

FLASH[®] の拡張機能

Windows® および Macintosh 向けの Flash® の拡張機能

本マニュアルがエンドユーザー使用許諾契約を含むソフトウェアと共に提供される場合、本マニュアルおよびその中に記載されているソフトウェアは、エンドユーザー使用許諾契約にもとづいて提供されるものであり、当該エンドユーザー使用許諾契約の契約条件に従ってのみ使用または複製することが可能となるものです。当該エンドユーザー使用許諾契約により許可されている場合を除き、本マニュアルのいかなる部分といえども、Adobe Systems Incorporated (アドビ システムズ社) の書面による事前の許可なしに、電子的、機械的、録音、その他いかなる形式・手段であれ、複製、検索システムへの保存、または伝送を行うことはできません。本マニュアルの内容は、エンドユーザー使用許諾契約を含むソフトウェアと共に提供されていない場合であっても、著作権法により保護されていることにご注意ください。

本マニュアルに記載される内容は、あくまでも参照用としてのみ使用されること、また、なんら予告なしに変更されることを条件として、提供されるものであり、従って、当該情報が、アドビ システムズ社による確約として解釈されてはなりません。アドビ システムズ社は、本マニュアルにおけるいかなる誤りまたは不正確な記述に対しても、いかなる義務や責任を負うものではありません。

新しいアートワークを創作するためにテンプレートとして取り込もうとすると既存のアートワークまたは画像は、著作権法により保護されている可能性のあるものであることをご注意ください。保護されているアートワークまたは画像を新しいアートワークに許可なく取り込んだ場合、著作権者の権利を侵害することがあります。従って、著作権者から必要とするすべての許可を必ず取得してください。

例として使用されている会社名は、実在の会社・組織を示すものではありません。

Adobe、Adobe ロゴ、ActionScript、Dreamweaver、Fireworks、Flash、および FlashHelp は、Adobe Systems Incorporated (アドビ システムズ社) の米国ならびに他の国における登録商標または商標です。

Mac OS is a trademark of Apple Inc., registered in the United States and other countries. Windows is either a registered trademark or a trademark of Microsoft Corporation in the United States and/or other countries. All other trademarks are the property of their respective owners.

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.iis.fhg.de/amm/>). You cannot use the MP3 compressed audio within the Software for real time or live broadcasts. If you require an MP3 decoder for real time or live broadcasts, you are responsible for obtaining this MP3 technology license. Portions of this product contain code licensed from Nellymoser (www.nellymoser.com). Flash CS3 video 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/>)



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

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.

目次

はじめに.....	5
Adobe Flash JavaScript API の概要.....	6
JavaScript API の新機能	10
Flash ドキュメントオブジェクトモデル (DOM).....	12
サンプルの実装	18
 第 1 章：トップレベル関数とメソッド	 21
 第 2 章：オブジェクト	 35
actionsPanel オブジェクト	39
BitmapInstance オブジェクト	47
BitmapItem オブジェクト	51
CompiledClipInstance オブジェクト.....	54
compilerErrors オブジェクト	59
ComponentInstance オブジェクト	61
componentsPanel オブジェクト	62
Contour オブジェクト.....	64
Document オブジェクト.....	67
adrawingLayer オブジェクト	194
Edge オブジェクト	202
Effect オブジェクト.....	207
Element オブジェクト.....	211
Fill オブジェクト.....	228
Filter オブジェクト.....	234
flash オブジェクト (fl)	248
FLfile オブジェクト	298
folderItem オブジェクト	313
fontItem オブジェクト.....	314
Frame オブジェクト	315
HalfEdge オブジェクト	331
Instance オブジェクト.....	337
Item オブジェクト	339
Layer オブジェクト	349
library オブジェクト.....	355

Math オブジェクト	372
Matrix オブジェクト	376
outputPanel オブジェクト	380
Oval オブジェクト	383
Parameter オブジェクト	386
Path オブジェクト	392
Project オブジェクト	399
ProjectItem オブジェクト	408
Rectangle オブジェクト	415
Screen オブジェクト	419
ScreenOutline オブジェクト	428
Shape オブジェクト	440
SoundItem オブジェクト	447
Stroke オブジェクト	452
SymbolInstance オブジェクト	467
SymbolItem オブジェクト	482
Text オブジェクト	489
TextAttrs オブジェクト	509
TextRun オブジェクト	519
Timeline オブジェクト	521
ToolObj オブジェクト	554
Tools オブジェクト	565
Vertex オブジェクト	572
XMLUI オブジェクト	575
VideoItem オブジェクト	584
 第 3 章 : C レベルの拡張性	 587
C 関数の統合	588
データ型	594
C レベルの API	595

はじめに

本マニュアルの読者は、Adobe® Flash® CS3 Professional ユーザーとして、Adobe® Flash® Player でランタイムに実行されるスクリプトを作成できる ActionScript™ に習熟しているかもしれません。本マニュアルで取り上げた Flash JavaScript アプリケーションプログラミングインターフェイス (JavaScript API) は、オーサリング環境で実行されるスクリプトを作成できる補助的なプログラミングツールです。

本マニュアルでは、JavaScript API で使用できるオブジェクト、メソッド、およびプロパティについて説明します。オーサリング環境で作業時に記載されているコマンドの使用方法を理解していることが前提になります。特定コマンドの内容について質問がある場合は、『Flash ユーザーガイド』など、Flash ヘルプの他のマニュアルで該当する情報を検索してください。

本マニュアルはまた、JavaScript または ActionScript のシンタックスと、関数、パラメータ、データ型など、基本的なプログラミング概念についての知識があるユーザーを対象としています。

このセクションでは、次のトピックについて説明します。

Adobe Flash JavaScript API の概要	6
JavaScript API の新機能	10
Flash ドキュメントオブジェクトモデル (DOM)	12
サンプルの実装	18

Adobe Flash JavaScript API の概要

Flash JavaScript API を使用すると、Flash プログラムを実行している Flash オーサリング環境でアクションを実行するためのスクリプトを作成できます。この機能は、SWF ファイルを再生している Flash Player 環境でアクションを実行するためのスクリプトを作成できる ActionScript 言語とは異なります。また、Web ブラウザに表示されるページで使用する JavaScript コマンドとも異なります。

JavaScript API を使用すると、Flash アプリケーションのスクリプトを作成してオーサリング処理を効率化することができます。たとえば、繰り返し作業を自動化するスクリプトの作成、ツールパネルへのカスタムツールの追加、またはタイムラインエフェクトの追加を行うことができます。

Flash JavaScript API は、Netscape JavaScript API を基に設計されている Adobe® Dreamweaver® および Adobe® Fireworks® JavaScript API に似た設計になっています。Flash JavaScript API は Document Object Model (DOM) を基にしており、JavaScript のオブジェクトを使用して Flash ドキュメントにアクセスすることができます。Flash JavaScript API には、Netscape JavaScript API のすべてのエレメントと Flash DOM が含まれています。ここでは、これらの追加されたオブジェクトとそのメソッドおよびプロパティについて説明します。Flash スクリプトでは、ネイティブの JavaScript 言語であればどのエレメントでも使用できますが、Flash ドキュメントのコンテキストで意味のあるエレメントだけが有効です。

JavaScript API には、JavaScript やカスタム C コードと一緒に使用して機能を拡張するための多数のメソッドもあります。詳細については、[587 ページ](#)、[第 3 章の「C レベルの拡張性」](#)を参照してください。

Flash の JavaScript インタプリタは Mozilla SpiderMonkey エンジン、バージョン 1.5 で、www.mozilla.org/js/spidermonkey/ から入手できます。SpiderMonkey は、Mozilla.org が開発した JavaScript 言語の 2 つのリファレンス実装のうちの 1 つで、Mozilla ブラウザにはこれと同じエンジンが組み込まれています。

SpiderMonkey は、ECMAScript (ECMA-262) Edition 3 の言語仕様で定義されているコア JavaScript 言語を実装し、この仕様に完全準拠しています。サポートされていないのは、ECMA-262 仕様に含まれない Mozilla ブラウザ固有のホストオブジェクトだけです。同様に、多くの JavaScript リファレンスガイドでは、コア JavaScript とクライアントサイド (ブラウザ関連) JavaScript を区別しています。Flash JavaScript インタプリタにはコア JavaScript だけが適用されます。

JSFL ファイルの作成

Flash JavaScript (JSFL) ファイルを作成し編集するには、Adobe Flash CS3 Professional または任意のテキストエディタを使用します。Flash を使用する場合、デフォルトのファイル拡張子は .jsfl になります。

[ヒストリ] パネルでコマンドを選択し、[ヒストリ] パネルの [保存] ボタンをクリックするか、パネルメニューの [コマンドとして保存] を選択しても、JSFL ファイルを作成できます。コマンド (JSFL) ファイルは "Commands" フォルダに保存されます (7 ページの「JSFL ファイルの保存」を参照してください)。他のスクリプトファイルと同様に、このファイルを開いて編集することができます。

[ヒストリ] パネルには、他の役に立つオプションも用意されています。選択したコマンドをクリップボードにコピーすることや、Flash で作業中に生成された JavaScript コマンドを表示することもできます。

[ヒストリ] パネルのコマンドをクリップボードにコピーするには：

1. [ヒストリ] パネルでコマンドを選択します。
2. 次のいずれかの操作を行います。
 - [コピー] ボタンをクリックします。
 - パネルメニューで [ステップのコピー] を選択します。

[ヒストリ] パネルに JavaScript コマンドを表示するには：

- パネルメニューで [表示]-[パネル内の JavaScript] を選択します。

JSFL ファイルの保存

Flash オーサリング環境で JSFL スクリプトを使用できるようにするには、"Configuration" フォルダ内のいずれかのフォルダに格納します。"Configuration" フォルダは、デフォルトでは次の場所にあります。

- Windows® 2000 または Windows® XP :
 < ブートドライブ >¥Documents and Settings< ユーザー >¥Local Settings¥Application Data¥Adobe¥Flash CS3¥< 言語 >¥Configuration
- Mac OS® X :
 Macintosh HD/Users/< ユーザー名 >/Library/Application Support/Adobe/Flash CS3/< 言語 >/Configuration/

"Configuration" フォルダの場所を調べるには、次の例に示すように、[fl.configDirectory](#) または [fl.configURI](#) を使用します。

```
// ディレクトリを変数に保存
var configDir = fl.configDirectory;
// ディレクトリを [ 出力 ] パネルに表示
fl.trace(fl.configDirectory);
```

"Configuration" フォルダには、オーサリング環境でアクセスできるスクリプトを格納するフォルダがあります。それらは、"Behaviors" (ビヘイビアのユーザーインターフェイスのサポート)、"Commands" ([コマンド] メニューに表示されるスクリプト)、"Effects" (タイムラインエフェクト)、"JavaScript" (ユーザーインターフェイスコントロールのデータを設定するために [スクリプトアシスト] で使用されるスクリプト)、"Tools" (ツールパネルの拡張ツール)、"WindowSWF" ([ウィンドウ] メニューに表示されるパネル) の各フォルダです。本マニュアルでは、コマンド、エフェクト、ツールで使用されるスクリプトについて重点的に説明します。

"Commands" フォルダのスクリプトを編集すると、新しいスクリプトは Flash ですぐに使用可能になります。エフェクトまたは拡張ツール用のスクリプトを編集する場合は、Flash を終了して再起動するか、`fl.reloadEffects()` コマンドまたは `fl.reloadTools()` コマンドを使用する必要があります。ただし、スクリプトを使用して拡張ツールをツールパネルに追加してから、そのスクリプトを編集する場合は、ツールを削除してから再度ツールパネルに追加するか、編集したツールを使用可能にするために Flash を終了して再起動する必要があります。

オーサリング環境で使用するためのコマンド、エフェクト、およびツールファイルを保存する場所は 3 つあります。

- [コマンド] メニューのアイテムとして表示されるスクリプトの場合は、JSFL ファイルを以下の場所の "Commands" フォルダに保存します。
 - Windows 2000 または Windows XP :
 < ブートドライブ >¥Documents and Settings< ユーザー >¥Local Settings¥Application Data¥Adobe¥Flash CS3< 言語 >¥Configuration¥Commands
 - Mac OS X :
 Macintosh HD/Users/< ユーザー名 >/Library/Application Support/Adobe/Flash CS3/< 言語 >/Configuration/Commands
- ツールパネルの拡張ツールとして表示されるスクリプトの場合は、JSFL ファイルを以下の場所の "Tools" フォルダに保存します。
 - Windows 2000 または Windows XP :
 < ブートドライブ >¥Documents and Settings< ユーザー >¥Local Settings¥Application Data¥Adobe¥Flash CS3< 言語 >¥Configuration¥Tools
 - Mac OS X :
 Macintosh HD/Users/< ユーザー名 >/Library/Application Support/Adobe/Flash CS3/< 言語 >/Configuration/Tools
- [エフェクト] パネルのタイムラインエフェクトとして表示されるスクリプトの場合は、JSFL ファイルを以下の場所の "Effects" フォルダに保存します。
 - Windows 2000 または Windows XP :
 < ブートドライブ >¥Documents and Settings< ユーザー >¥Local Settings¥Application Data¥Adobe¥Flash CS3< 言語 >¥Configuration¥Effects
 - Mac OS X :
 Macintosh HD/Users/< ユーザー名 >/Library/Application Support/Adobe/Flash CS3/< 言語 >/Configuration/Effects

XML ファイルなど、JSFL ファイルに属するファイルがある場合は、そのファイルも JSFL ファイルと同じディレクトリに保存します。

JSFL ファイルの実行

JSFL ファイルを実行する方法はいくつかあります。このセクションでは最も一般的な方法を示します。

"Commands" フォルダにあるスクリプトを実行するには、次のいずれかの操作を行います。

- [コマンド]-[スクリプト名] を選択します。
- スクリプトに割り当てたキーボードショートカットを使用します。キーボードショートカットを割り当てるには、[編集]-[キーボードショートカット] を使用して、[コマンド] ポップアップメニューから、[描画メニューコマンド] を選択します。メニューツリーのコマンドノードを展開し、使用可能なスクリプトのリストを表示します。

"Commands" フォルダにないコマンドスクリプトを実行するには、次のいずれかの操作を行います。

- オーサリング環境から、[コマンド]-[コマンドの実行] を選択し、実行するスクリプトを選択します。
- スクリプト内から、`fl.runScript()` コマンドを使用します。
- ファイルシステムからスクリプトファイルをダブルクリックします。

JSFL ファイルに実装されているツールをツールパネルに追加するには：

1. ツールで使用する JSFL ファイルと関連ファイルを "Tools" フォルダにコピーします ([7 ページの「JSFL ファイルの保存」](#)を参照してください)。
2. [編集]-[ツールパネルのカスタマイズ](Windows) または [Flash]-[ツールパネルのカスタマイズ](Macintosh) を選択します。
3. 使用可能なツールのリストにツールを追加します。
4. [OK] をクリックします。

個々の JavaScript API コマンドを ActionScript ファイルに追加するには、MMExecute() 関数を使用します。この関数については、『ActionScript 3.0 言語リファレンス』で説明しています。ただし、MMExecute() 関数が有効であるのは、コンポーネントのプロパティインスペクタやオーサリング環境内の SWF パネルなど、カスタムユーザーインターフェイスエレメントのコンテキストで使用する場合だけです。JavaScript API コマンドを ActionScript から呼び出した場合でも、Flash Player やオーサリング環境外では無効になります。

ActionScript スクリプトからコマンドを発行するには：

- 次のシンタックスを使用します (いくつかのコマンドを1つのストリングに連結できます)。

```
MMEecute(Javascript command string);
```

コマンドラインからスクリプトを実行することもできます。

Windows でコマンドラインからスクリプトを実行するには：

- 次のシンタックスを使用します (必要に応じてパス情報を追加します)。

```
"flash.exe" myTestFile.jsfl
```

Macintosh で “Terminal” アプリケーションからスクリプトを実行するには：

- 次のシンタックスを使用します (必要に応じてパス情報を追加します)。

```
osascript -e 'tell application "flash" to open alias "Mac OS  
X:Users:user:myTestFile.jsfl" '
```

osascript コマンドは、ファイル内の AppleScript も実行できます。たとえば、myScript という名前のファイルに次のテキストを配置できます。

```
tell application "flash"  
    open alias "Mac OS X:Users:user:myTestFile.jsfl"  
end tell
```

その後、スクリプトを呼び出すために、次のコマンドを使用します。

```
osascript myScript
```

JavaScript API の新機能

Flash CS3 では、いくつかのオブジェクトが追加されています。さらに、既存のオブジェクトにも新しいメソッドやプロパティが追加されています。ここでは、他の変更と共に追加された機能について簡単に説明します。

JavaScript API をまだ使用したことがない場合は、このセクションをスキップして、ドキュメントオブジェクトモデル (12 ページの「[Flash ドキュメントオブジェクトモデル \(DOM\)](#)」を参照) に進んでください。

新しいオブジェクト

Flash CS3 で追加された新しいオブジェクトは次のとおりです。

[actionsPanel](#) オブジェクト

[compilerErrors](#) オブジェクト

[Oval](#) オブジェクト

[Rectangle](#) オブジェクト

新しいメソッドとプロパティ

Flash CS3 で、既存のオブジェクトに対して追加された新しいメソッドとプロパティは次のとおりです。

```
document.as3AutoDeclare
document.as3Dialect
document.as3ExportFrame
document.as3StrictMode
document.as3WarningsMode
document.asVersion
document.canSaveAVersion()
document.docClass
document.getMobileSettings()
document.getPlayerVersion()
document.id
document.resetOvalObject()
document.resetRectangleObject()
document.revertToLastVersion()
document.saveAVersion()
document.setMobileSettings()
document.setOvalObjectProperty()
document.setPlayerVersion()
document.setRectangleObjectProperty()
document.synchronizeWithHeadVersion()
element.getTransformationPoint()
element.rotation
element.scaleX
element.scaleY
element.setTransformationPoint()
element.skewX
element.skewY
element.transformX
element.transformY
element.x
element.y
filter.enabled
fl.actionsPanel
fl.as3PackagePaths
fl.addEventListener()
fl.closeAllPlayerDocuments()
fl.compilerErrors
fl.clipCopyString()
fl.downloadLatestVersion()
fl.packagePaths
fl.findDocumentDOM()
fl.findObjectInDocByName()
```

```
fl.findObjectInDocByType()
fl.removeEventListener()
fl.revertDocumentToLastVersion()
fl.resetPackagePaths()
fl.resetAS3PackagePaths()
fl.saveAVersionOfDocument()
fl.scriptURI
fl.selectElement()
fl.selectTool()
fl.synchronizeDocumentWithHeadVersion()
item.linkageBaseClass
shape.isOvalObject
shape.isRectangleObject
timeline.copyMotion()
timeline.copyMotionAsAS3()
timeline.pasteMotion()
```

その他の変更

次のメソッドは Flash CS3 で新しいパラメータを使用します。

```
document.addNewText()
fl.closeAll()
```

次のプロパティは使用されなくなっています。

```
stroke.breakAtCorners
```

Flash の拡張ファイルの実装 ([587 ページ](#)、[第3章の「C レベルの拡張性」](#)を参照) で使用する "mm_jsapi.h" ファイルが変更されました。

Flash ドキュメントオブジェクトモデル (DOM)

Flash JavaScript API の Flash ドキュメントオブジェクトモデル (DOM) は、1 組のトップレベル関数 ([21 ページの「トップレベル関数とメソッド」](#)を参照) と、FLfile オブジェクトおよび Flash オブジェクト (fl) の 2 つのトップレベルオブジェクトで構成されます。いずれのオブジェクトも、Flash オーサリング環境を開いている場合には必ず存在するので、いつでもスクリプトで使用できます。詳細については、[FLfile オブジェクト](#) および [flash オブジェクト \(fl\)](#) を参照してください。

Flash オブジェクトを参照する場合は、flash または fl を使用します。たとえば、開いている FLA ファイルをすべて閉じるには、次のいずれかのステートメントを使用します。

```
flash.closeAll();
fl.closeAll();
```

Flash オブジェクトには、次に示す子オブジェクトがあります。

オブジェクト	アクセス方法
actionsPanel オブジェクト	actionsPanel オブジェクトにアクセスするには、 <code>fl.actionsPanel</code> を使用します。このオブジェクトは、Flash オーサリング環境の [アクション] パネルに相当します。
compilerErrors オブジェクト	compilerErrors オブジェクトにアクセスするには、 <code>fl.compilerErrors</code> を使用します。このオブジェクトは、Flash オーサリング環境の [コンパイルエラー] パネルに相当します。
componentsPanel オブジェクト	componentsPanel オブジェクトにアクセスするには、 <code>fl.componentsPanel</code> を使用します。このオブジェクトは、Flash オーサリング環境の [コンポーネント] パネルに相当します。
Document オブジェクト	開いているすべてのドキュメントの配列を取得するには、 <code>fl.documents</code> を使用します。特定のドキュメントにアクセスするには、 <code>fl.documents[index]</code> を使用します。現在のドキュメント (フォーカスのあるドキュメント) にアクセスするには、 <code>fl.getDocumentDOM()</code> を使用します。
adrawingLayer オブジェクト	drawingLayer オブジェクトにアクセスするには、 <code>fl.drawingLayer</code> を使用します。
Effect オブジェクト	Flash 起動時に登録されるエフェクトに相当するエフェクト記述子の配列を取得するには、 <code>fl.effects</code> を使用します。特定のエフェクトにアクセスするには、 <code>fl.effects[index]</code> を使用します。現在適用されているエフェクトのエフェクト記述子にアクセスするには、 <code>fl.activeEffect</code> を使用します。
Math オブジェクト	Math オブジェクトにアクセスするには、 <code>fl.Math</code> を使用します。
outputPanel オブジェクト	outputPanel オブジェクトにアクセスするには、 <code>fl.outputPanel</code> を使用します。このオブジェクトは、Flash オーサリング環境の [出力] パネルに相当します。
Project オブジェクト	現在開いているプロジェクトの Project オブジェクトを返すには、 <code>fl.getProject()</code> を使用します。
Tools オブジェクト	Tools オブジェクトの配列にアクセスするには、 <code>fl.tools</code> を使用します。
XMLUI オブジェクト	XML User Interface (XMLUI) オブジェクトにアクセスするには、 <code>fl.xmlui</code> を使用します。XMLUI オブジェクトを使用すると、XMLUI ダイアログボックスのプロパティを取得、設定することができます。

Document オブジェクト

トップレベル Flash オブジェクトの重要なプロパティは `fl.documents` プロパティです ([fl.documents](#) プロパティを参照してください)。 `fl.documents` プロパティは、現在オーサリング環境で開いている FLA ファイルを表す Document オブジェクトの配列を含んでいます。各 Document オブジェクトのプロパティは、FLA ファイルに含めることのできるほとんどのエレメントを表します。したがって、DOM の大部分は Document オブジェクトの子オブジェクトとプロパティで構成されます。詳細については、[Document オブジェクト](#) を参照してください。

たとえば、最初に開いたドキュメントを参照するには、`flash.documents[0]` ステートメントまたは `fl.documents[0]` ステートメントを使用します。最初のドキュメントは、オーサリング環境の現在のセッションで開いた最初の Flash ドキュメントになります。最初に開いたドキュメントを閉じると、開いている他のドキュメントのインデックスがデクリメントされます。

特定のドキュメントのインデックスを検索するには、
`flash.findDocumentIndex(nameOfDocument)` または
`fl.findDocumentIndex(nameOfDocument)` を使用します。詳細については、
[fl.findDocumentIndex\(\)](#) を参照してください。

現在フォーカスのあるドキュメントにアクセスするには、`flash.getDocumentDOM()` ステートメントまたは `fl.getDocumentDOM()` ステートメントを使用します。詳細については、
[fl.getDocumentDOM\(\)](#) を参照してください。後者は、このドキュメントのほとんどの例で使用しているシンタックスです。

`fl.documents` 配列にある特定のドキュメントを検索するには、配列内を順に検索し、各ドキュメントの `document.name` プロパティを確認します。 [fl.documents](#) および [document.name](#) を参照してください。

前の表にない DOM のすべてのオブジェクト ([12 ページの「Flash ドキュメントオブジェクトモデル \(DOM\)」](#)を参照) は、Document オブジェクトからアクセスします。たとえば、ドキュメントのライブラリにアクセスするには、`document.library` プロパティを使用して、`library` オブジェクトを取得します。

```
fl.getDocumentDOM().library
```

ライブラリのアイテムの配列にアクセスするには、`library.items` プロパティを使用します。配列の各エレメントは `Item` オブジェクトです。

```
fl.getDocumentDOM().library.items
```

ライブラリの特定のアイテムにアクセスするには、`library.items` 配列のメンバーを指定します。

```
fl.getDocumentDOM().library.items[0]
```

つまり、`library` オブジェクトは Document オブジェクトの子オブジェクトであり、`Item` オブジェクトは `library` オブジェクトの子オブジェクトになります。詳細については、[document.library](#)、[library オブジェクト](#)、[library.items](#)、および [Item オブジェクト](#) を参照してください。

アクションのターゲットの指定

特に指定しない限り、現在のフォーカスまたは選択箇所メソッドが適用されます。たとえば、次のスクリプトでは、特定のオブジェクトを指定していないので、現在の選択箇所のサイズが2倍になります。

```
fl.getDocumentDOM().scaleSelection(2, 2);
```

また、Flashドキュメントの現在選択しているアイテムをターゲットにしてアクションを実行する必要がある場合もあります。この場合は、`document.selection` プロパティにある配列を使用します (`document.selection` を参照してください)。次の例に示すように、配列の最初の要素は、現在選択されているアイテムを表します。

```
var accDescription = fl.getDocumentDOM().selection[0].description;
```

次のスクリプトは、現在の選択箇所ではなく、エレメント配列に保存されているステージの最初のエレメントのサイズを2倍にします。

```
var element = fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0];
if (element) {
    element.width = element.width*2;
    element.height = element.height*2;
}
```

さらに、次の例に示すように、ステージのすべてのエレメントをループして、指定したサイズで幅と高さを増やすこともできます。

```
var elementArray =
    fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements;
for (var i=0; i < elementArray.length; i++) {
    var offset = 10;
    elementArray[i].width += offset;
    elementArray[i].height += offset;
}
```

DOM 構造の概要

次のリストに、DOM 構造をアウトラインフォーマットで示します。各行の先頭にある数字はオブジェクトのレベルを示します。たとえば、“03” で表されるオブジェクトは高位の“02” オブジェクトの子になり、この“02” オブジェクトは、さらに高位の“01” オブジェクトの子になります。

親オブジェクトのプロパティを指定することでオブジェクトが使用可能になる場合もあります。

たとえば、`document.timelines` プロパティは `Timeline` オブジェクトの配列を含んでいます (`document.timelines` および `Timeline オブジェクト` を参照してください)。これらのプロパティについては次のアウトラインで示しています。

最後に、オブジェクトの中には、他のオブジェクトの子オブジェクトではなく、オブジェクトのサブクラスになっているものもあります。別のオブジェクトのサブクラスであるオブジェクトには、スーパークラスオブジェクトのメソッドとプロパティの他に、独自のメソッドや親オブジェクトのプロパティがあります。サブクラスオブジェクトは、スーパークラスオブジェクトと同じ階層レベルを共有します。たとえば、Item オブジェクトは BitmapItem オブジェクトのスーパークラスです ([Item オブジェクト](#) および [BitmapItem オブジェクト](#) を参照してください)。これらの関係については次のアウトラインで示しています。

01 トップレベル関数とメソッド

01 FLfile オブジェクト

01 flash オブジェクト (fl)

02 compilerErrors オブジェクト

02 componentsPanel オブジェクト

02 Document オブジェクト (fl.documents 配列)

03 Filter オブジェクト

03 Matrix オブジェクト

03 Fill オブジェクト

03 Stroke オブジェクト

03 library オブジェクト

04 Item オブジェクト (library.items 配列)

04 BitmapItem オブジェクト (Item オブジェクトのサブクラス)

04 folderItem オブジェクト (Item オブジェクトのサブクラス)

04 fontItem オブジェクト (Item オブジェクトのサブクラス)

04 SoundItem オブジェクト (Item オブジェクトのサブクラス)

04 SymbolItem オブジェクト (Item オブジェクトのサブクラス)

04 VideoItem オブジェクト (Item オブジェクトのサブクラス)

03 Timeline オブジェクト (document.timelines 配列)

04 Layer オブジェクト (timeline.layers 配列)

05 Frame オブジェクト (layer.frames 配列)

06 Element オブジェクト (frame.elements 配列)

07 Matrix オブジェクト (Element.matrix)

06 Instance オブジェクト (抽象クラス、Element オブジェクトのサブクラス)

06 BitmapInstance オブジェクト (Instance オブジェクトのサブクラス)

06 CompiledClipInstance オブジェクト (Instance オブジェクトのサブクラス)

06 ComponentInstance オブジェクト (SymbolInstance オブジェクトのサブクラス)

- 07 [Parameter](#) オブジェクト ([componentInstance.parameters](#))
- 06 [SymbolInstance](#) オブジェクト ([Instance](#) オブジェクトのサブクラス)
- 06 [Text](#) オブジェクト ([Element](#) オブジェクトのサブクラス)
 - 07 [TextRun](#) オブジェクト ([text.textRuns](#) 配列)
 - 08 [TextAttrs](#) オブジェクト ([textRun.textAttrs](#) 配列)
- 06 [Shape](#) オブジェクト ([Element](#) オブジェクトのサブクラス)
 - 07 [Oval](#) オブジェクト
 - 07 [Rectangle](#) オブジェクト
 - 07 [Contour](#) オブジェクト ([shape.contours](#) 配列)
 - 08 [HalfEdge](#) オブジェクト
 - 09 [Vertex](#) オブジェクト
 - 09 [Edge](#) オブジェクト
 - 07 [Edge](#) オブジェクト ([shape.edges](#) 配列)
 - 08 [HalfEdge](#) オブジェクト
 - 09 [Vertex](#) オブジェクト
 - 09 [Edge](#) オブジェクト
 - 07 [Vertex](#) オブジェクト ([shape.vertices](#) 配列)
 - 08 [HalfEdge](#) オブジェクト
 - 09 [Vertex](#) オブジェクト
 - 09 [Edge](#) オブジェクト
- 03 [ScreenOutline](#) オブジェクト
 - 04 [Screen](#) オブジェクト ([screenOutline.screens](#) 配列)
 - 05 [Parameter](#) オブジェクト ([screen.parameters](#) 配列)
- 02 [adrawingLayer](#) オブジェクト
 - 03 [Path](#) オブジェクト
 - 04 [Contour](#) オブジェクト
- 02 [Effect](#) オブジェクト ([fl.effects](#) 配列)
- 02 [Math](#) オブジェクト
- 02 [outputPanel](#) オブジェクト
- 02 [Project](#) オブジェクト
 - 03 [ProjectItem](#) オブジェクト ([project.items](#) 配列)
- 02 [Tools](#) オブジェクト ([fl.tools](#) 配列)
 - 03 [ToolObj](#) オブジェクト ([tools.toolObjs](#) 配列)
- 02 [XMLUI](#) オブジェクト

サンプルの実装

Adobe Flash CS3 Professional には、JSFL 実装のサンプルがいくつか用意されています。JavaScript API に慣れるために、これらのファイルを参照およびインストールすることができます。サンプルは、www.adobe.com/go/learn_fl_samples_jp にある ZIP ファイル内の "Samples/ExtendingFlash" というフォルダにインストールされます。

サンプルの Shape コマンド

"ExtendingFlash/Shape" フォルダには、"Shape.jsfl" という名前のサンプルの JavaScript API スクリプトがあります (上記の「[サンプルの実装](#)」を参照してください)。このスクリプトは、シェイプの輪郭に関する情報を [出力] パネルに表示します。

Shape スクリプトをインストールして実行するには :

1. "Shape.jsfl" ファイルを "Configuration/Commands" フォルダにコピーします (7 ページの「[JSFL ファイルの保存](#)」を参照してください)。
2. Flash ドキュメント (FLA ファイル) 内でシェイプオブジェクトを選択します。
3. [コマンド]-[Shape] を選択してスクリプトを実行します。

サンプルのフィルタ取得および設定 (filtersGetSet) コマンド

"ExtendingFlash/filtersGetSet" フォルダには、"filtersGetSet.jsfl" という名前のサンプルの JavaScript API スクリプトがあります (18 ページの「[サンプルの実装](#)」を参照してください)。このスクリプトは、選択したオブジェクトにフィルタを追加し、追加するフィルタに関する情報を [出力] パネルに表示します。

filtersGetSet スクリプトをインストールして実行するには :

1. "filtersGetSet.jsfl" ファイルを "Configuration/Commands" フォルダにコピーします (7 ページの「[JSFL ファイルの保存](#)」を参照してください)。
2. Flash ドキュメント (FLA ファイル) 内で、テキスト、ムービークリップ、またはボタンオブジェクトを選択します。
3. [コマンド]-[filtersGetSet] を選択してスクリプトを実行します。

サンプルの PolyStar ツール

"ExtendingFlash/PolyStar" フォルダには、"PolyStar.jsfl" という名前のサンプルの JavaScript API スクリプトがあります (18 ページの「[サンプルの実装](#)」を参照してください)。

PolyStar.jsfl は、Flash のツールパネルにある PolyStar ツールを複製します。このスクリプトは、JavaScript API を使用して PolyStar ツールを作成する方法を示しており、コードが行う処理についての詳細なコメントが付けられています。このファイルを読むことにより、JavaScript API の使用方法について理解することができます。また、"Tools" ディレクトリにある "PolyStar.xml" ファイルを読むと、独自のツールの作成方法をさらに詳しく学ぶことができます。

サンプルの [ビットマップのトレース] パネル

"ExtendingFlash/TraceBitmapPanel" フォルダには、"TraceBitmap fla" および "TraceBitmap.swf" という名前の一対のファイルがあります (18 ページの「[サンプルの実装](#)」を参照してください)。これらのファイルでは、Flash の機能を制御するパネルの設計方法と作成方法を示します。MMExecute() 関数を使用して ActionScript スクリプトから JavaScript コマンドを呼び出す方法も示します。

TraceBitmap サンプルを実行するには：

1. Flash を実行中の場合は、Flash を終了します。
2. "TraceBitmap.swf" ファイルを "Configuration" フォルダのサブディレクトリである "WindowSWF" フォルダにコピーします。7 ページの「[JSFL ファイルの保存](#)」を参照してください。たとえば、Windows の場合、このフォルダは、ブートドライブ ¥Documents and Settings¥ ユーザー ¥Local Settings¥Application Data¥Adobe¥ Flash CS3¥ < 言語 ¥¥Configuration¥WindowSWF にあります。
3. Flash を起動します。
4. Flash ドキュメント (FLA ファイル) を作成するか開いて、ビットマップまたは JPEG イメージをファイルに読み込みます。
"TraceBitmapPanel" フォルダにある "flower.jpg" ファイル、またはその他のイメージを使用できます。
5. 読み込んだイメージを選択し、[ウィンドウ]-[他のパネル]-[TraceBitmap] を選択します。
6. [Submit] をクリックします。
イメージがシェイプのグループに変換されます。

サンプル DLL

"ExtendingFlash/dllSampleComputeSum" フォルダには、DLL 実装のサンプルがあります (18 ページの「[サンプルの実装](#)」を参照してください)。DLL 構築の詳細については、587 ページ、第 3 章の「[C レベルの拡張性](#)」を参照してください。

トップレベル関数とメソッド

この章では、Adobe Flash JavaScript アプリケーションプログラミングインターフェイス (JavaScript API) を使用する場合に利用可能なトップレベル関数とメソッドについて説明します。JavaScript API ファイルを保存する場所の詳細については、[7 ページの「JSFL ファイルの保存」](#)を参照してください。

各関数またはメソッドに関連するオーサリング環境の領域についての概要を次に示します。それに続いて、関数とメソッドについてアルファベット順に掲載します。

グローバルメソッド

次のメソッドは、任意の JavaScript API スクリプトから呼び出せます。

```
alert()  
confirm()  
prompt()
```

タイムラインエフェクト

次の関数は、タイムラインエフェクトに固有の関数です。

```
configureEffect()  
executeEffect()  
removeEffect()
```

拡張ツール

次の関数は、拡張ツールを作成するスクリプトで使用できます。

```
activate()  
configureTool()  
deactivate()  
keyDown()  
keyUp()  
mouseDoubleClick()  
mouseDown()  
mouseMove()  
mouseUp()  
notifySettingsChanged()  
setCursor()
```

activate()

使用できるバージョン

Flash MX 2004

シンタックス

```
function activate() {  
    // ステートメント  
}
```

パラメータ

なし

戻り値

なし。

説明

関数。拡張ツールが使用可能になったとき (ツールをツールパネルで選択したとき) に呼び出されます。ツールで初期化処理が必要なときに、この関数を使用します。

例

次の例では、ツールパネルで拡張ツールが選択されたときに `tools.activeTool` の値を設定します。

```
function activate() {  
    var theTool = fl.tools.activeTool  
}
```

関連項目

[tools.activeTool](#)

alert()

使用できるバージョン

Flash MX 2004

シンタックス

```
alert ( alertText )
```

パラメータ

alertText 警告ダイアログボックスに表示するメッセージを指定するストリング。

戻り値

なし。

説明

メソッド。モーダル警告ダイアログボックスに、ストリングおよび [OK] ボタンを表示します。

例

次の例では、“Process Complete” というメッセージを警告ダイアログボックスに表示します。

```
alert("Process Complete");
```

関連項目

[confirm\(\)](#), [prompt\(\)](#)

configureEffect()

使用できるバージョン

Flash MX 2004

シンタックス

```
function configureEffect() {  
    // ステートメント  
}
```

パラメータ

なし

戻り値

なし。

説明

関数。Flash のロード時に 1 回呼び出されます。エフェクトのグローバル初期化ステートメントをこの関数内に記述します。ここでは、エフェクトのインスタンスごとのパラメータデータにはアクセスできません。

関連項目

[executeEffect\(\)](#), [removeEffect\(\)](#)

configureTool()

使用できるバージョン

Flash MX 2004

シンタックス

```
function configureTool() {  
    // ステートメント  
}
```

パラメータ

なし

戻り値

なし。

説明

関数。Flash を起動し、ツールパネルに拡張ツールがロードされたときに呼び出されます。Flash で必要になるツールに関する情報を設定するときに、この関数を使用します。

例

次の例に、この関数の 2 種類の実装方法を示します。

```
function configureTool() {  
    theTool = fl.tools.activeTool;  
    theTool.setToolName("myTool");  
    theTool.setIcon("myTool.png");  
    theTool.setMenuString("My Tool's menu string");  
    theTool.setToolTip("my tool's tool tip");  
    theTool.setOptionsFile( "mtTool.xml" );  
}
```

```
function configureTool() {  
    theTool = fl.tools.activeTool;  
    theTool.setToolName("ellipse");  
    theTool.setIcon("Ellipse.png");  
    theTool.setMenuString("Ellipse");  
    theTool.setToolTip("Ellipse");  
    theTool.showTransformHandles( true );  
}
```


confirm()

使用できるバージョン

Flash 8

シンタックス

```
confirm ( strAlert )
```

パラメータ


strAlert 警告ダイアログボックスに表示するメッセージを指定するストリング。

戻り値

ブール値。ユーザーが [OK] をクリックしたときは true、[キャンセル] をクリックしたときは false。

説明

メソッド。モーダル警告ダイアログボックスに、ストリングおよび [OK] ボタンと [キャンセル] ボタンを表示します。

	開いているドキュメント (FLA ファイル) がない場合、このメソッドはエラー状態になり失敗します。
---	--

例

次の例では、“Sort data?” というメッセージを警告ダイアログボックスに表示します。

```
confirm("Sort data?");
```

関連項目

[alert\(\)](#), [prompt\(\)](#)

deactivate()

使用できるバージョン

Flash MX 2004

シンタックス

```
function deactivate() {  
    // ステートメント  
}
```

パラメータ

なし

戻り値

なし。

説明

関数。拡張ツールが非アクティブになったとき（別のツールがアクティブになったとき）に呼び出されます。ツールで後処理が必要なときに、この関数を使用します。

例

次の例では、ツールが非アクティブになったときに、[出力] パネルにメッセージを表示します。

```
function deactivate() {  
    fl.trace( "Tool is no longer active" );  
}
```

executeEffect()

使用できるバージョン

Flash MX 2004

シンタックス

```
function executeEffect() {  
    // ステートメント  
}
```

パラメータ

なし

戻り値

なし。

説明

関数。ユーザーがエフェクトを最初に適用したとき、またはエフェクトのプロパティを変更したときに呼び出されます。この関数に含まれるコードは、目的のエフェクトを作成するためにオリジナルのオブジェクトを変更します。また、removeEffect 関数で必要とされる場合、オリジナルのオブジェクトを非表示レイヤーにコピーします。

関連項目

[configureEffect\(\)](#), [removeEffect\(\)](#)

keyDown()

使用できるバージョン

Flash MX 2004

シンタックス

```
function keyDown() {  
    // ステートメント  
}
```

パラメータ

なし

戻り値

なし。

説明

関数。拡張ツールがアクティブで、ユーザーがキーを押したときに呼び出されます。どのキーが押されたのかを特定するには、スクリプトで `tools.getKeyDown()` を呼び出します。

例

次の例では、拡張ツールがアクティブで、ユーザーがキーを押したときに、どのキーが押されたのかを表示します。

```
function keyDown() {  
    fl.trace("key " + fl.tools.getKeyDown() + " was pressed");  
}
```

関連項目

[keyUp\(\)](#), [tools.getKeyDown\(\)](#)

keyUp()

使用できるバージョン

Flash MX 2004

シンタックス

```
function keyUp() {  
    // ステートメント  
}
```

パラメータ

なし

戻り値

なし。

説明

関数。拡張ツールがアクティブで、ユーザーがキーを離したときに呼び出されます。

例

次の例では、拡張ツールがアクティブで、ユーザーがキーを離したときに [出力] パネルにメッセージを表示します。

```
function keyUp() {  
    fl.trace("Key is released");  
}
```

関連項目

[keyDown\(\)](#)

mouseDoubleClick()

使用できるバージョン

Flash MX 2004

シンタックス

```
function mouseDoubleClick() {  
    // ステートメント  
}
```

パラメータ

なし

戻り値

なし。

説明

関数。拡張ツールがアクティブで、ステージ上でマウスボタンをダブルクリックしたときに呼び出されます。

例

次の例では、拡張ツールがアクティブで、マウスボタンをダブルクリックしたときに [出力] パネルにメッセージを表示します。

```
function mouseDoubleClick() {  
    fl.trace("Mouse was double-clicked");  
}
```

mouseDown()

使用できるバージョン

Flash MX 2004

シンタックス

```
function mouseDown( [ pt ] ) {  
    // ステートメント  
}
```

パラメータ

pt マウスボタンを押したときのマウスの位置を示すポイント。マウスボタンを押したときに、関数にポイントが渡されます。このパラメータは省略可能です。

戻り値

なし。

説明

関数。拡張ツールがアクティブで、マウスポインタがステージ上にある状態でマウスボタンを押すと呼び出されます。

例

次の例に、拡張ツールがアクティブなときのこの関数の使用方法を示します。最初の例では、マウスボタンを押したことを示すメッセージを [出力] パネルに表示します。2 番目の例では、マウスボタンを押したときのマウスの位置を示す *x* 座標と *y* 座標を表示します。

```
function mouseDown() {  
    fl.trace("Mouse button has been pressed");  
}  
  
function mouseDown(pt) {  
    fl.trace("x = "+ pt.x+" :: y = "+pt.y);  
}
```

mouseMove()

使用できるバージョン

Flash MX 2004

シンタックス

```
function mouseMove( [ pt ] ) {  
    // ステートメント  
}
```

パラメータ

pt マウスの現在位置を示すポイント。マウスが移動すると、マウスの位置を示すポイントが関数に渡されます。ステージが編集モードまたは同じ位置で編集モードの場合、ポイントの座標は編集集中のオブジェクトを基準にした値になります。その他のモードでは、ポイントの座標はステージを基準にした値になります。このパラメータは省略可能です。

戻り値

なし。

説明

関数。拡張ツールがアクティブで、ステージ上の指定ポイント上でマウスを移動すると呼び出されます。マウスボタンは押していても離していてもかまいません。

例

次の例に、この関数の使用方法を示します。最初の例では、マウスを移動していることを示すメッセージを[出力]パネルに表示します。2 番目の例では、マウスボタンを移動したときのマウスの位置を示す x 座標と y 座標を表示します。

```
function mouseMove() {  
    fl.trace("moving");  
}  
  
function mouseMove(pt) {  
    fl.trace("x = " + pt.x + " :: y = " + pt.y);  
}
```

mouseUp()

使用できるバージョン

Flash MX 2004

シンタックス

```
function mouseUp() {  
    // ステートメント  
}
```

パラメータ

なし

戻り値

なし。

説明

関数。拡張ツールがアクティブで、ステージでマウスボタンを押して離すと呼び出されます。

例

次の例では、拡張ツールがアクティブで、マウスボタンを離したときに [出力] パネルにメッセージを表示します。

```
function mouseUp() {  
    fl.trace("mouse is up");  
}
```

notifySettingsChanged()

使用できるバージョン

Flash MX 2004

シンタックス

```
function notifySettingsChanged() {  
    // ステートメント  
}
```

パラメータ

なし

戻り値

なし。

説明

関数。拡張ツールがアクティブで、ユーザーがプロパティインスペクタでオプションを変更したときに呼び出されます。`tools.activeTool` プロパティを使用して、オプションの現在値を調べることができます([tools.activeTool](#) を参照してください)。

例

次の例では、拡張ツールがアクティブで、ユーザーがプロパティインスペクタでオプションを変更したときに [出力] パネルにメッセージを表示します。

```
function notifySettingsChanged() {
    var theTool = fl.tools.activeTool;
    var newValue = theTool.myProp;
}
```

prompt()

使用できるバージョン

Flash MX 2004

シンタックス

```
prompt(promptMsg [,text])
```

パラメータ

promptMsg プロンプトダイアログボックスに表示するストリング (Mac OS X では 256 文字まで)。

text テキストフィールドにデフォルト値として表示するオプションのストリング。

戻り値

ユーザーが [OK] をクリックした場合はユーザーが入力したストリング、[キャンセル] をクリックした場合は `null`。

説明

メソッド。モーダル警告ダイアログボックスに、プロンプトとオプションのテキストおよび [OK] ボタンと [キャンセル] ボタンを表示します。

例

次の例では、ユーザーに対して、ユーザー名を入力するためのプロンプトを表示します。ユーザーがユーザー名を入力して [OK] をクリックすると、そのユーザー名が [出力] パネルに表示されます。

```
var userName = prompt("Enter user name", "Type user name here");
fl.trace(userName);
```

関連項目

[alert\(\)](#), [confirm\(\)](#)

removeEffect()

使用できるバージョン

Flash MX 2004

シンタックス

```
function removeEffect() {  
    // ステートメント  
}
```

パラメータ

なし

戻り値

なし。

説明

関数。ユーザーがエフェクトのプロパティを変更したとき、または[エフェクトの削除]メニューアイテムを使用したときに呼び出されます。この関数のコードは、オブジェクトを元の状態に戻します。たとえば、テキストストリングを分解するエフェクトの場合、removeEffect() メソッドは、分解されたテキストストリングを削除し、元のストリングで置き換えます。

関連項目

[configureEffect\(\)](#), [executeEffect\(\)](#)

setCursor()

使用できるバージョン

Flash MX 2004

シンタックス

```
function setCursor() {  
    // ステートメント  
}
```

パラメータ

なし

戻り値

なし。

説明

関数。拡張ツールがアクティブでマウスが移動すると呼び出されます。スクリプトでカスタムポインタを設定することができます。使用するポインタをスクリプトで指定するには、`tools.setCursor()` を呼び出します。各ポインタと整数値との対応一覧は、[tools.setCursor\(\)](#) を参照してください。

例

```
function setCursor() {  
    fl.tools.setCursor( 1 );  
}
```

オブジェクト

この章では、Flash JavaScript アプリケーションプログラミングインターフェイス (JavaScript API) で利用可能な各オブジェクトについて簡単に説明します。次の表に、オブジェクトをアルファベット順に示します。

オブジェクト	説明
actionsPanel オブジェクト	actionsPanel オブジェクトは、現在表示されている [アクション] パネルを表します。
BitmapInstance オブジェクト	BitmapInstance オブジェクトは、 Instance オブジェクト のサブクラスで、フレームのビットマップを表します。
BitmapItem オブジェクト	BitmapItem オブジェクトは、ドキュメントのライブラリにあるビットマップに相当します。BitmapItem オブジェクトは、 Item オブジェクト のサブクラスです。
CompiledClipInstance オブジェクト	CompiledClipInstance オブジェクトは、 Instance オブジェクト のサブクラスです。
compilerErrors オブジェクト	compilerErrors オブジェクトは、[コンパイルエラー] パネルを表します。 flash オブジェクト (fl) のプロパティで、 <code>fl.compilerErrors</code> でアクセスできます。
ComponentInstance オブジェクト	ComponentInstance オブジェクトは、 SymbolInstance オブジェクト のサブクラスで、フレームのコンポーネントを表します。
componentsPanel オブジェクト	componentsPanel オブジェクトは、[コンポーネント] パネルを表します。 flash オブジェクト (fl) のプロパティで、 <code>fl.componentsPanel</code> でアクセスできます。
Contour オブジェクト	Contour オブジェクトは、シェイプの境界にあるハーフエッジの閉じたパスを表します。
Document オブジェクト	Document オブジェクトは、ステージを表します。
drawingLayer オブジェクト	drawingLayer オブジェクトは、JavaScript から Flash オブジェクトの子としてアクセスできます。
Edge オブジェクト	Edge オブジェクトは、ステージ上のシェイプのエッジを表します。

オブジェクト	説明
Effect オブジェクト	Effect オブジェクトは、タイムラインエフェクトのインスタンスを表します。
Element オブジェクト	ステージにあるすべてのエレメントの種類は Element です。
Fill オブジェクト	Fill オブジェクトには、ツールパネルまたは選択したシェイプの塗りのカラー設定のすべてのプロパティがあります。
Filter オブジェクト	Filter オブジェクトには、すべてのフィルタのすべてのプロパティがあります。
flash オブジェクト (fl)	flash オブジェクトは、Flash アプリケーションを表します。
FLfile オブジェクト	FLfile オブジェクトを使用すると、ローカルファイルシステム上のファイルとフォルダにアクセスしたり、修正および削除したりするための Flash 拡張機能を記述することができます。
folderItem オブジェクト	folderItem オブジェクトは、 Item オブジェクト のサブクラスです。
fontItem オブジェクト	fontItem オブジェクトは、 Item オブジェクト のサブクラスです。
Frame オブジェクト	Frame オブジェクトは、レイヤーのフレームを表します。
HalfEdge オブジェクト	Shape オブジェクト のエッジの指定サイドです。
Instance オブジェクト	Instance オブジェクトは、 Element オブジェクト のサブクラスです。
Item オブジェクト	Item オブジェクトは、抽象基本クラスです。
Layer オブジェクト	Layer オブジェクトは、タイムラインにあるレイヤーを表します。
library オブジェクト	library オブジェクトは、[ライブラリ] パネルを表します。
Math オブジェクト	Math オブジェクトは、flash オブジェクトの読み取り専用プロパティとして使用できます。 fl.Math を参照してください。
Matrix オブジェクト	Matrix オブジェクトは、変換マトリックスを表します。
outputPanel オブジェクト	outputPanel オブジェクトは、[出力] パネルを表します。シンタックスエラーなどのトラブルシューティング情報を表示します。
Oval オブジェクト	Oval オブジェクトは、楕円ツールを使用して描画されるシェイプです。アイテムが Rectangle オブジェクト かどうかを調べるには、 shape.isOvalObject を使用します。
Parameter オブジェクト	Parameter オブジェクト型は、 screen.parameters 配列 (Flash オーサリングツールのスクリーンのプロパティインスペクタに対応) または componentInstance.parameters 配列 (オーサリングツールのコンポーネントのプロパティインスペクタに対応) によってアクセスされます。
Path オブジェクト	Path オブジェクトは、線のセグメント (直線、曲線、または両方) のシーケンスを定義します。このシーケンスは通常、拡張ツールを作成する場合に使用します。

オブジェクト	説明
Project オブジェクト	Project オブジェクトは、Flash プロジェクト (FLP) ファイルを表します。
ProjectItem オブジェクト	ProjectItem オブジェクトは、プロジェクトに追加されたアイテム (ディスク上のファイル) を表します。
Rectangle オブジェクト	Rectangle オブジェクトは、矩形ツールを使用して描画されるシェイプです。アイテムが Rectangle オブジェクトかどうかを調べるには、 shape.isRectangleObject を使用します。
Screen オブジェクト	Screen オブジェクトは、スライドまたはフォームドキュメント内の1つのスクリーンを表します。
ScreenOutline オブジェクト	ScreenOutline オブジェクトは、スライドまたはフォームドキュメント内のスクリーンのグループを表します。
Shape オブジェクト	Shape オブジェクトは、 Element オブジェクト のサブクラスです。Shape オブジェクトは、ステージで図形座標を操作または作成するときに、drawing API よりも精度を高くすることができます。
SoundItem オブジェクト	SoundItem オブジェクトは、 Item オブジェクト のサブクラスです。このオブジェクトは、サウンドを作成するのに使用するライブラリアイテムを表します。
Stroke オブジェクト	Stroke オブジェクトは、カスタム設定など、線に関するすべての設定を含みます。
SymbolInstance オブジェクト	SymbolInstance オブジェクトは、 Instance オブジェクト のサブクラスで、フレイムのシンボルを表します。
SymbolItem オブジェクト	SymbolItem オブジェクトは、 Item オブジェクト のサブクラスです。
Text オブジェクト	Text オブジェクトは、ドキュメント内の1つのテキストアイテムを表します。
TextAttrs オブジェクト	TextAttrs オブジェクトは、ダイレクト選択範囲に適用できるテキストのすべてのプロパティを含みます。このオブジェクトは、 Text オブジェクト のサブクラスです。
TextRun オブジェクト	TextRun オブジェクトは、 TextAttrs オブジェクト のすべてのプロパティと一致する属性を持つ連続した文字を表します。
Timeline オブジェクト	Timeline オブジェクトは、Flash タイムラインを表します。現在のドキュメントのタイムラインにアクセスするには、 <code>fl.getDocumentDOM().getTimeline()</code> を使用します。
ToolObj オブジェクト	ToolObj オブジェクトは、ツールパネルの個々のツールを表します。
Tools オブジェクト	Tools オブジェクトは、Flash オブジェクト (fl.tools) からアクセスできます。

オブジェクト	説明
Vertex オブジェクト	Vertex オブジェクトは、座標データを保持するシェイプデータ構造の一部です。
Videoltem オブジェクト	Videoltem オブジェクトは、 Item オブジェクト のサブクラスです。
XMLUI オブジェクト	XMLUI オブジェクトは、XMLUI ダイアログボックスのプロパティを取得、設定することができます。受け取ることもキャンセルすることもできます。

actionsPanel オブジェクト

使用できるバージョン

Flash CS3 Professional

説明

actionsPanel オブジェクトは、現在表示されている [アクション] パネルを表します。flash オブジェクト (fl) のプロパティであり、fl.actionsPanel でアクセスできます ([flash オブジェクト \(fl\)](#) を参照してください)。

actionsPanel オブジェクトのメソッド一覧

actionsPanel オブジェクトでは次のメソッドを使用できます。

メソッド	説明
<code>actionsPanel.getClassForObject()</code>	指定した変数のクラスを返します。
<code>actionsPanel.getScriptAssistMode()</code>	スクリプトアシストモードが有効かどうかを指定します。
<code>actionsPanel.getSelectedText()</code>	[アクション] パネルで現在選択されているテキストを返します。
<code>actionsPanel.getText()</code>	[アクション] パネルのテキストを返します。
<code>actionsPanel.hasSelection()</code>	[アクション] パネルのテキストが現在選択されているかどうかを指定します。
<code>actionsPanel.replaceSelectedText()</code>	現在選択されているテキストを指定したテキストで置換します。
<code>actionsPanel.setScriptAssistMode()</code>	スクリプトアシストモードを有効または無効にします。
<code>actionsPanel.setSelection()</code>	[アクション] パネル内の指定した文字のセットを選択します。
<code>actionsPanel.setText()</code>	[アクション] パネルのテキストを削除して、指定したテキストを追加します。

actionsPanel.getClassForObject()

使用できるバージョン

Flash CS3 Professional

シンタックス

```
actionsPanel.getClassForObject(ASvariableName)
```

パラメータ

ASvariableName ActionScript 変数の名前を表すストリング。

戻り値

ASvariableName がどのクラスのメンバーであるかを表すストリング。

説明

メソッド。指定した変数のクラスを返します。変数は現在表示されている [アクション] パネルで定義されている必要があります。また、[アクション] パネルで、変数定義の後にカーソルまたは選択されているテキストを配置する必要があります。

例

次の例では、[アクション] パネルでカーソルを `var myVar:ActivityEvent;` ステートメントの後に配置した場合、変数 `myVar` に割り当てられたクラスが表示されます。

```
// [ アクション ] パネルで、次のコードを入力します。  
// 行末の後にカーソルを置く  
var myVar:ActivityEvent;  
  
// 次のコードを JSFL ファイルに配置する  
var theClass = fl.actionsPanel.getClassForObject("myVar");  
fl.trace(theClass); // トレース対象 : "ActivityEvent"
```

actionsPanel.getScriptAssistMode()

使用できるバージョン

Flash CS3 Professional

シンタックス

```
actionsPanel.getScriptAssistMode()
```

パラメータ

なし

戻り値

スクリプトアシストモードを有効にするか(true)、無効にするか(false)を指定するブール値。

説明

メソッド。スクリプトアシストモードが有効かどうかを指定します。

例

次の例では、スクリプトアシストモードが有効でない場合、メッセージが表示されます。

```
mAssist = fl.actionsPanel.getScriptAssistMode();
if (!mAssist) {
    alert("For more guidance when writing ActionScript code, try Script Assist mode");
}
```

関連項目

[actionsPanel.setScriptAssistMode\(\)](#)

actionsPanel.getSelectedText()

使用できるバージョン

Flash CS3 Professional

シンタックス

```
actionsPanel.getSelectedText()
```

パラメータ

なし

戻り値

[アクション] パネルで現在選択されているテキストを含むストリング。

説明

メソッド。[アクション] パネルで現在選択されているテキストを返します。

例

次の例では、[アクション] パネルで現在選択されているテキストが表示されます。

```
var apText = fl.actionsPanel.getSelectedText();
fl.trace(apText);
```

関連項目

[actionsPanel.getText\(\)](#), [actionsPanel.hasSelection\(\)](#),
[actionsPanel.replaceSelectedText\(\)](#), [actionsPanel.setSelection\(\)](#)

actionsPanel.getText()

使用できるバージョン

Flash CS3 Professional

シンタックス

```
actionsPanel.getText()
```

パラメータ

なし

戻り値

[アクション] パネル内のすべてのテキストを含むストリング。

説明

メソッド。[アクション] パネル内のテキストを返します。

例

次の例では、[アクション] パネル内のテキストが表示されます。

```
var apText = fl.actionsPanel.getText();  
fl.trace(apText);
```

関連項目

[actionsPanel.getSelectedText\(\)](#), [actionsPanel.setText\(\)](#)

actionsPanel.hasSelection()

使用できるバージョン

Flash CS3 Professional

シンタックス

```
actionsPanel.hasSelection()
```

パラメータ

なし

戻り値

[アクション] パネル内でテキストが選択されている (true) か、選択されていない (false) かを示すブール値。

説明

メソッド。[アクション] パネル内のテキストが現在選択されているかどうかを示します。

例

次の例では、[アクション] パネルで現在選択されているテキストが表示されます。テキストが選択されていない場合、[アクション] パネル内のすべてのテキストが表示されます。

```
if (fl.actionsPanel.hasSelection()) {  
    var apText = fl.actionsPanel.getSelectedText();  
}  
else {  
    var apText = fl.actionsPanel.getText();  
}  
fl.trace(apText);
```

関連項目

[actionsPanel.getSelectedText\(\)](#), [actionsPanel.getText\(\)](#),
[actionsPanel.replaceSelectedText\(\)](#), [actionsPanel.setSelection\(\)](#)

actionsPanel.replaceSelectedText()

使用できるバージョン

Flash CS3 Professional

シンタックス

```
actionsPanel.replaceSelectedText(replacementText)
```

パラメータ

replacementText [アクション] パネル内の選択されているテキストを置換するテキストを表すストリング。

戻り値

ブール値。[アクション] パネルが見つかった場合は true を返します。それ以外の場合は false を返します。

説明

メソッド。現在選択されているテキストを *replacementText* で指定したテキストに置き換えます。*replacementText* に含まれる文字数が、選択されたテキストの文字数より多い場合、選択されたテキストの直後にある文字は *replacementText* の後に配置されます。したがって、文字が上書きされることはありません。

例

次の例では、[アクション] パネルで現在選択されているテキストを置換します。

```
if (fl.actionsPanel.hasSelection()) {  
    fl.actionsPanel.replaceSelectedText("// © 2006 Adobe Inc.");  
}
```

関連項目

[actionsPanel.getSelectedText\(\)](#), [actionsPanel.hasSelection\(\)](#),
[actionsPanel.setSelection\(\)](#), [actionsPanel.setText\(\)](#)

actionsPanel.setScriptAssistMode()

使用できるバージョン

Flash CS3 Professional

シンタックス

```
actionsPanel.setScriptAssistMode(bScriptAssist)
```

パラメータ

bScriptAssist スクリプトアシストモードを有効にするか無効にするかを指定するブール値。

戻り値

スクリプトアシストモードの有効化または無効化が正常に行われたかどうかを示すブール値。

説明

メソッド。スクリプトアシストモードを有効または無効にします。

例

次の例では、スクリプトアシストモードの状態を切り替えます。

```
fl.trace(fl.actionsPanel.getScriptAssistMode());  
if (fl.actionsPanel.getScriptAssistMode()){  
    fl.actionsPanel.setScriptAssistMode(false);  
}  
else {  
    fl.actionsPanel.setScriptAssistMode(true);  
}  
fl.trace(fl.actionsPanel.getScriptAssistMode());
```

関連項目

[actionsPanel.getScriptAssistMode\(\)](#)

actionsPanel.setSelection()

使用できるバージョン

Flash CS3 Professional

シンタックス

```
actionsPanel.setSelection(startIndex, numberOfChars)
```

パラメータ

startIndex 選択する先頭の文字をゼロから始まる整数で指定します。

numberOfChars 選択する文字数を整数で指定します。

戻り値

要求された文字が選択されるか(true)、そうでないか(false)を指定するブール値。

説明

メソッド。[アクション]パネル内の指定した文字のセットを選択します。

例

次の例では、[アクション]パネル内の“2006”という文字を指定したテキストで置換します。

```
// 次の行を [アクション] パネルの 1 行目として入力する
// 2006 - ユーザーの要求 40196 に対処
// 次の行を JSFL ファイルに入力する
fl.actionsPanel.setSelection(3,4);
fl.actionsPanel.replaceSelectedText("// 最終更新日 : 2007");
```

関連項目

[actionsPanel.getSelectedText\(\)](#), [actionsPanel.hasSelection\(\)](#),
[actionsPanel.replaceSelectedText\(\)](#)

actionsPanel.setText()

使用できるバージョン

Flash CS3 Professional

シンタックス

```
actionsPanel.setText(replacementText)
```

パラメータ

replacementText [アクション]パネルに配置するテキストを表す文字列。

戻り値

ブール値。指定したテキストが[アクション]パネルに配置された場合は true、そうでない場合は false を返します。

説明

メソッド。[アクション]パネル内のテキストを削除して、*replacementText* で指定したテキストを追加します。

例

次の例では、[アクション]パネル内の現在のテキストを指定したテキストで置き換えます。

```
fl.actionsPanel.setText("// このコードを削除 - 必要でないため");
```

関連項目

[actionsPanel.getText\(\)](#), [actionsPanel.replaceSelectedText\(\)](#)

BitmapInstance オブジェクト

継承 [Element オブジェクト](#) > [Instance オブジェクト](#) > BitmapInstance オブジェクト

使用できるバージョン

Flash MX 2004

説明

BitmapInstance オブジェクトは、Instance オブジェクトのサブクラスで、フレームのビットマップを表します。[Instance オブジェクト](#)を参照してください。

BitmapInstance オブジェクトのメソッド一覧

BitmapInstance オブジェクトでは、[Instance オブジェクト](#)のメソッドの他に、次のメソッドを使用できます。

メソッド	説明
bitmapInstance.getBits()	ビットマップのビットからビットマップエフェクトを作成し、操作し、Flash に戻します。
bitmapInstance.setBits()	既存のビットマップエレメントのビットを設定します。

BitmapInstance オブジェクトのプロパティ一覧

BitmapInstance オブジェクトでは、[Instance オブジェクト](#)のプロパティの他に、以下のプロパティを使用できます。

プロパティ	説明
bitmapInstance.hPixels	読み取り専用。ビットマップの幅を表すピクセル単位の整数です。
bitmapInstance.vPixels	読み取り専用。ビットマップの高さを表すピクセル単位の整数です。

bitmapInstance.getBits()

使用できるバージョン

Flash MX 2004

シンタックス

```
bitmapInstance.getBits()
```

パラメータ

なし

戻り値

width、height、depth、bits の各プロパティ、およびビットマップにカラーテーブルがある場合は cTab プロパティを含むオブジェクト。bits エLEMENT はバイト配列です。cTab エLEMENT は、"#RRGGBB" 形式のカラー値の配列です。配列の長さは、カラーテーブルの長さです。

バイト配列は、DLL または共有ライブラリによって参照される場合にだけ有効です。通常は、拡張ツールまたはエフェクトを作成するときにだけ使用します。Flash JavaScript で使用する DLL の作成方法については、[第 3 章の「C レベルの拡張性」](#)を参照してください。

説明

メソッド。ビットマップのビットからビットマップエフェクトを作成し、操作し、Flash に戻します。[bitmapInstance.setBits\(\)](#) も参照してください。

例

次のコードでは、現在選択しているオブジェクトへの参照を作成して、オブジェクトがビットマップかどうかをテストし、ビットマップの高さ、幅、ビット深度をトレースします。

```
var isBitmap = fl.getDocumentDOM().selection[0].instanceType;
if(isBitmap == "bitmap"){
    var bits = fl.getDocumentDOM().selection[0].getBits();
    fl.trace("height = " + bits.height);
    fl.trace("width = " + bits.width);
    fl.trace("depth = " + bits.depth);
}
```

関連項目

[bitmapInstance.setBits\(\)](#)

bitmapInstance.hPixels

使用できるバージョン

Flash MX 2004

シンタックス

bitmapInstance.hPixels

説明

読み取り専用プロパティ。ビットマップの幅を水平方向のピクセル数で表す整数です。

例

次のコードでは、ビットマップの幅をピクセル単位で取得します。

```
// 水平方向のピクセル数を取得します。
var bmObj = fl.getDocumentDOM().selection[0];
var isBitmap = bmObj.instanceType;
if(isBitmap == "bitmap"){
    var numHorizontalPixels = bmObj.hPixels;
}
```

関連項目

[bitmapInstance.vPixels](#)

bitmapInstance.setBits()

使用できるバージョン

Flash MX 2004

シンタックス

`bitmapInstance.setBits(bitmap)`

パラメータ

bitmap height、width、depth、bits、および cTab の各プロパティを含むオブジェクト。height、width、および depth の各プロパティは整数です。bits プロパティはバイト配列です。cTab プロパティは、ビット深度が 8 以下のビットマップだけに必要で、"#RRGGBB" 形式でカラー値を表すストリングです。



バイト配列は、外部ライブラリによって参照される場合にだけ有効です。通常は、拡張ツールまたはエフェクトを作成するときにだけ使用します。

戻り値

なし

説明

メソッド。既存のビットマップ要素のビットを設定します。ビットマップのビットからビットマップエフェクトを作成し、操作し、ビットマップを Flash に戻します。

例

次のコードでは、現在の選択がビットマップかどうかをテストし、ビットマップの高さを 150 ピクセルに設定します。

```
var isBitmap = fl.getDocumentDOM().selection[0].instanceType;
if(isBitmap == "bitmap"){
    var bits = fl.getDocumentDOM().selection[0].getBits();
    bits.height = 150;
    fl.getDocumentDOM().selection[0].setBits(bits);
}
```

関連項目

[bitmapInstance.getBits\(\)](#)

bitmapInstance.vPixels

使用できるバージョン

Flash MX 2004

シンタックス

`bitmapInstance.vPixels`

説明

読み取り専用プロパティ。ビットマップの高さを垂直方向のピクセル数で表す整数です。

例

次のコードでは、ビットマップの高さをピクセル単位で取得します。

```
// 垂直方向のピクセル数を取得します。
var bmObj = fl.getDocumentDOM().selection[0];
var isBitmap = bmObj.instanceType;
if(isBitmap == "bitmap"){
    var numVerticalPixels = bmObj.vPixels;
}
```

関連項目

[bitmapInstance.hPixels](#)

BitmapItem オブジェクト

継承 [Item オブジェクト](#) > BitmapItem オブジェクト

使用できるバージョン

Flash MX 2004

説明

BitmapItem オブジェクトは、ドキュメントのライブラリにあるビットマップに相当します。BitmapItem オブジェクトは、Item オブジェクトのサブクラスです。[Item オブジェクト](#)を参照してください。

BitmapItem オブジェクトのプロパティ一覧

BitmapItem オブジェクトには、[Item オブジェクト](#)のプロパティの他に、次のプロパティがあります。

プロパティ	説明
bitmapItem.allowSmoothing	ビットマップのスムージングを許可するかどうかを指定するブール値。
bitmapItem.compressionType	ビットマップに適用する画像圧縮の種類を指定するストリング。
bitmapItem.quality	ビットマップの画質を指定する整数。
bitmapItem.useImportedJPEGQuality	デフォルトの読み込み JPEG 画質を使用するかどうかを指定するブール値。

bitmapItem.allowSmoothing

使用できるバージョン

Flash MX 2004

シンタックス

`bitmapItem.allowSmoothing`

説明

プロパティ。ビットマップのスムージングを許可するか (true)、許可しないか (false) を指定するブール値です。

例

次のコードでは、現在のドキュメントのライブラリにある最初のアイテムの allowSmoothing プロパティに true を設定します。

```
fl.getDocumentDOM().library.items[0].allowSmoothing = true;
alert(fl.getDocumentDOM().library.items[0].allowSmoothing);
```

bitmapItem.compressionType

使用できるバージョン

Flash MX 2004

シンタックス

bitmapItem.compressionType

説明

プロパティ。ビットマップに適用する画像圧縮の種類を指定するストリングです。指定できる値は、"photo" または "lossless" です。bitmapItem.useImportedJPEGQuality の値が false の場合、"photo" は、画質が 0 ～ 100 の JPEG に相当します。bitmapItem.useImportedJPEGQuality の値が true の場合、"photo" は、デフォルトドキュメント画質の JPEG に相当します。"lossless" は GIF 形式または PNG 形式に相当します (bitmapItem.useImportedJPEGQuality を参照してください)。

例

次のコードでは、現在のドキュメントのライブラリにある最初のアイテムの compressionType プロパティに "photo" を設定します。

```
fl.getDocumentDOM().library.items[0].compressionType = "photo";  
alert(fl.getDocumentDOM().library.items[0].compressionType);
```

bitmapItem.quality

使用できるバージョン

Flash MX 2004

シンタックス

bitmapItem.quality

説明

プロパティ。ビットマップの画質を指定する整数です。デフォルトドキュメント画質を使用する場合は -1 を指定します。使用しない場合は、0 ～ 100 の整数を指定します。JPEG 圧縮だけで使用できます。

例

次のコードでは、現在のドキュメントのライブラリにある最初のアイテムの quality プロパティに 65 を設定します。

```
fl.getDocumentDOM().library.items[0].quality = 65;  
alert(fl.getDocumentDOM().library.items[0].quality);
```

bitmapItem.useImportedJPEGQuality

使用できるバージョン

Flash MX 2004

シンタックス

`bitmapItem.useImportedJPEGQuality`

説明

プロパティ。デフォルトの読み込み JPEG 画質を使用するか (`true`)、使用しないか (`false`) を指定するブール値です。JPEG 圧縮だけで使用できます。

例

次のコードでは、現在のドキュメントのライブラリにある最初のアイテムの `useImportedJPEGQuality` プロパティに `true` を設定します。

```
fl.getDocumentDOM().library.items[0].useImportedJPEGQuality = true;  
alert(fl.getDocumentDOM().library.items[0].useImportedJPEGQuality);
```

CompiledClipInstance オブジェクト

継承 [Element オブジェクト](#) > [Instance オブジェクト](#) > CompiledClipInstance オブジェクト

使用できるバージョン

Flash MX 2004

説明

CompiledClipInstance オブジェクトは、Instance オブジェクトのサブクラスです。ムービークリップのインスタンスがコンパイル済みクリップライブラリアイテムに変換されている必要があります ([Instance オブジェクト](#)を参照してください)。

CompiledClipInstance オブジェクトのプロパティ一覧

CompiledClipInstance オブジェクトには、[Instance オブジェクト](#)のプロパティの他に、次のプロパティがあります。

プロパティ	説明
<code>compiledClipInstance.accName</code>	[アクセシビリティ] パネルの [名前] フィールドと等しいストリング。
<code>compiledClipInstance.actionScript</code>	このインスタンスの <code>ActionScript</code> を表すストリング。 <code>symbolInstance.actionScript</code> と同等です。
<code>compiledClipInstance.description</code>	[アクセシビリティ] パネルの [説明] フィールドと等しいストリング。
<code>compiledClipInstance.forceSimple</code>	子オブジェクトのアクセシビリティを有効または無効にするブール値。
<code>compiledClipInstance.shortcut</code>	[アクセシビリティ] パネルの [ショートカット] フィールドと等しいストリング。
<code>compiledClipInstance.silent</code>	オブジェクトのアクセシビリティを有効または無効にするブール値。これは、[アクセシビリティ] パネルの [オブジェクトをアクセス可能にする] の設定の逆ロジックと等価です。
<code>compiledClipInstance.tabIndex</code>	[アクセシビリティ] パネルの [タブインデックス] フィールドと等価の整数。

compiledClipInstance.accName

使用できるバージョン

Flash MX 2004

シンタックス

compiledClipInstance.accName

説明

プロパティ。[アクセシビリティ] パネルの [名前] フィールドと等しいストリングです。スクリーンリーダーは、名前を読み上げてオブジェクトを識別します。

例

次の例では、最初に選択したオブジェクトのアクセシビリティ名を取得、設定します。

```
// オブジェクト名を取得します。
var theName = fl.getDocumentDOM().selection[0].accName;
// オブジェクト名を設定します。
fl.getDocumentDOM().selection[0].accName = 'Home Button';
```

compiledClipInstance.actionScript

使用できるバージョン

Flash MX 2004

シンタックス

compiledClipInstance.actionScript

説明

プロパティ。このインスタンスの **ActionScript** を表すストリングで、[symbolInstance.actionScript](#) と同等です。

例

次のコードでは、指定したエレメントに **ActionScript** を割り当てます。

```
// 指定したボタンコンパイル済みクリップインスタンスに ActionScript を割り当てます。
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0]
    .actionScript = "on(click) {trace('button is clicked');}";
// 現在選択しているボタンコンパイル済みクリップインスタンスに ActionScript を割り当てます。
fl.getDocumentDOM().selection[0].actionScript =
    "on(click) {trace('button is clicked');}";
```

compiledClipInstance.description

使用できるバージョン

Flash MX 2004

シンタックス

compiledClipInstance.description

説明

プロパティ。[アクセシビリティ] パネルの [説明] フィールドと等しいストリングです。この説明はスクリーンリーダーによって読み上げられます。

例

次の例では、description プロパティの取得と設定について示します。

```
// 現在の選択の説明を取得します。
var theDescription = fl.getDocumentDOM().selection[0].description;
// 現在の選択の説明を設定します。
fl.getDocumentDOM().selection[0].description =
    "This is compiled clip number 1";
```

compiledClipInstance.forceSimple

使用できるバージョン

Flash MX 2004

シンタックス

compiledClipInstance.forceSimple

説明

プロパティ。子オブジェクトのアクセシビリティを有効または無効にするブール値です。[アクセシビリティ] パネルの [子オブジェクトをアクセス可能にする] の逆ロジックに等しくなります。forceSimple が true の場合、[子オブジェクトをアクセス可能にする] をオフにした場合と同じになります。forceSimple が false の場合、[子オブジェクトをアクセス可能にする] をオンにした場合と同じになります。

例

次の例では、forceSimple プロパティの取得と設定について示します。

```
// 子オブジェクトがアクセス可能かどうかを調べます。
var areChildrenAccessible = fl.getDocumentDOM().selection[0].forceSimple;
// 子オブジェクトをアクセス可能にします。
fl.getDocumentDOM().selection[0].forceSimple = false;
```


compiledClipInstance.shortcut

使用できるバージョン

Flash MX 2004

シンタックス

compiledClipInstance.shortcut

説明

プロパティ。[アクセシビリティ] パネルの [ショートカット] フィールドと等しいストリングです。このショートカットはスクリーンリーダーによって読み上げられます。ダイナミックテキストフィールドでは、このプロパティは使用できません。

例

次の例では、shortcut プロパティの取得と設定について示します。

```
// オブジェクトのショートカットキーを取得します。
var theShortcut = fl.getDocumentDOM().selection[0].shortcut;
// オブジェクトのショートカットキーを設定します。
fl.getDocumentDOM().selection[0].shortcut = "Ctrl+I";
```

compiledClipInstance.silent

使用できるバージョン

Flash MX 2004

シンタックス

compiledClipInstance.silent

説明

プロパティ。オブジェクトのアクセシビリティを有効または無効にするブール値です。[アクセシビリティ] パネルの [オブジェクトをアクセス可能にする] の逆ロジックに等しくなります。つまり、silent が true の場合、[オブジェクトをアクセス可能にする] をオフにした場合と同じになります。silent が false の場合、[オブジェクトをアクセス可能にする] をオンにした場合と同じになります。

例

次の例では、silent プロパティの取得と設定について示します。

```
// オブジェクトがアクセス可能かどうかを調べます。
var isSilent = fl.getDocumentDOM().selection[0].silent;
// オブジェクトをアクセス可能に設定します。
fl.getDocumentDOM().selection[0].silent = false;
```

compiledClipInstance.tabIndex

使用できるバージョン

Flash MX 2004

シンタックス

compiledClipInstance.tabIndex

説明

プロパティ。[アクセシビリティ] パネルの [タブインデックス] フィールドと等しい整数です。ユーザーが Tab キーを押したときにアクセスされるオブジェクトのタブ順序を作成します。

例

次の例では、tabIndex プロパティの取得と設定について示します。

```
// オブジェクトの tabIndex を取得します。  
var theTabIndex = fl.getDocumentDOM().selection[0].tabIndex;  
// オブジェクトの tabIndex を設定します。  
fl.getDocumentDOM().selection[0].tabIndex = 1;
```

compilerErrors オブジェクト

使用できるバージョン

Flash CS3 Professional

説明

compilerErrors オブジェクトは、[コンパイルエラー] パネルを表します。flash オブジェクト (fl) のプロパティで、fl.compilerErrors でアクセスできます ([flash オブジェクト \(fl\)](#) を参照してください)。

compilerErrors オブジェクトのメソッド一覧

compilerErrors オブジェクトでは次のメソッドを使用できます。

メソッド	説明
compilerErrors.clear()	[コンパイルエラー] パネルの内容をクリアします。
compilerErrors.save()	[コンパイルエラー] パネルの内容をローカルのテキストファイルに保存します。

compilerErrors.clear()

使用できるバージョン

Flash CS3 Professional

シンタックス

```
compilerErrors.clear()
```

パラメータ

なし

戻り値

なし。

説明

メソッド。[コンパイルエラー] パネルの内容をクリアします。

例

次の例では、[コンパイルエラー] パネルの内容をクリアします。

```
fl.compilerErrors.clear();
```

関連項目

[compilerErrors.save\(\)](#)

compilerErrors.save()

使用できるバージョン

Flash CS3 Professional

シンタックス

```
compilerErrors.save(fileURI [, bAppendToFile [, bUseSystemEncoding]])
```

パラメータ

fileURI 保存したファイルの名前を指定するストリング。file:///URI で表します。*fileURI* が既に存在し、*bAppendToFile* に対して true の値を指定しなかった場合、*fileURI* は警告なしで上書きされます。

bAppendToFile オプションのブール値。[コンパイルエラー] パネルの内容を *fileURI* に追加する (true) か、追加しない (false) かを指定します。デフォルト値は false です。

bUseSystemEncoding オプションのブール値。システムエンコーディングを使用して [コンパイルエラー] パネルのテキストを保存するかどうかを指定します。false (デフォルト) の場合、UTF-8 エンコーディングを使用して [コンパイルエラー] パネルのテキストを保存します。このとき、テキストの先頭にバイト順マーク (BOM) 文字が付きます。デフォルト値は false です。

戻り値

なし。

説明

メソッド。[コンパイルエラー] パネルの内容をローカルのテキストファイルに保存します。

例

次の例では、[コンパイルエラー] パネルの内容を "C:\tests" フォルダの "errors.log" というファイルに保存します。

```
fl.compilerErrors.save("file:///c:/tests/errors.log");
```

関連項目

[compilerErrors.clear\(\)](#)

ComponentInstance オブジェクト

継承 [Element オブジェクト](#) > [Instance オブジェクト](#) > [SymbolInstance オブジェクト](#) > [ComponentInstance オブジェクト](#)

使用できるバージョン

Flash MX 2004

説明

ComponentInstance オブジェクトは、SymbolInstance オブジェクトのサブクラスで、フレーム上のコンポーネントを表します ([SymbolInstance オブジェクト](#) を参照してください)。

ComponentInstance オブジェクトのプロパティ一覧

ComponentInstance オブジェクトには、[SymbolInstance オブジェクト](#) のプロパティの他に、次のプロパティがあります。

プロパティ	説明
componentInstance.parameters	読み取り専用。コンポーネントのプロパティインスペクタからアクセス可能な ActionScript 2.0 のプロパティの配列です。

componentInstance.parameters

使用できるバージョン

Flash MX 2004

シンタックス

`componentInstance.parameters`

説明

読み取り専用プロパティ。コンポーネントのプロパティインスペクタからアクセス可能な ActionScript 2.0 のプロパティの配列です。詳細については、[386 ページの「Parameter オブジェクト」](#) を参照してください。

例

次の例では、parameters プロパティの取得と設定について示します。

```
var parms = fl.getDocumentDOM().selection[0].parameters;
parms[0].value = "some value";
```

関連項目

[Parameter オブジェクト](#)

componentsPanel オブジェクト

使用できるバージョン

Flash MX 2004

説明

componentsPanel オブジェクトは、[コンポーネント] パネルを表します。flash オブジェクト (fl) のプロパティで、fl.componentsPanel でアクセスできます ([flash オブジェクト \(fl\)](#) を参照してください)。

componentsPanel オブジェクトのメソッド一覧

componentsPanel オブジェクトで使用できるメソッドは次のとおりです。

メソッド	説明
<code>componentsPanel.addItemToDocument()</code>	指定したコンポーネントを、ドキュメントの指定した位置に追加します。
<code>componentsPanel.reload()</code>	[コンポーネント] パネルのコンポーネントのリストを更新します。

componentsPanel.addItemToDocument()

使用できるバージョン

Flash MX 2004

シンタックス

`componentsPanel.addItemToDocument(position, categoryName, componentName)`

パラメータ

position コンポーネントを追加する位置を指定するポイント ({x:0, y:100} など)。 *position* は、コンポーネントの基準点 (あるいは 原点 または ゼロ点) ではなく、コンポーネントの中心点を基準に指定します。

categoryName コンポーネントカテゴリの名前を指定するストリング ("Data" など)。有効なカテゴリ名は [コンポーネント] パネルに表示されます。

componentName 指定したカテゴリのコンポーネント名を指定するストリング ("WebServiceConnector" など)。有効なコンポーネント名は [コンポーネント] パネルに表示されます。

戻り値

なし

説明

指定したコンポーネントを、ドキュメントの指定した位置に追加します。

例

次の例では、このメソッドの使用例を示します。

```
fl.componentsPanel.addItemToDocument({x:0, y:0}, "User Interface", "CheckBox");
fl.componentsPanel.addItemToDocument({x:0, y:100}, "Data",
    "WebServiceConnector");
fl.componentsPanel.addItemToDocument({x:0, y:200}, "User Interface", "Button");
```

componentsPanel.reload()

使用できるバージョン

Flash 8

シンタックス

```
componentsPanel.reload()
```

パラメータ

なし

戻り値

ブール値。[コンポーネント] パネルのリストを更新した場合は true を返します。それ以外の場合は false を返します。

説明

メソッド。[コンポーネント] パネルのコンポーネントのリストを更新します。

例

次の例では、[コンポーネント] パネルを更新します。

```
fl.componentsPanel.reload();
```

Contour オブジェクト

使用できるバージョン

Flash MX 2004

説明

Contour オブジェクトは、シェイプの境界にあるハーフエッジの閉じたパスを表します。

Contour オブジェクトのメソッド一覧

Contour オブジェクトで使用できるメソッドは次のとおりです。

プロパティ	説明
<code>contour.getHalfEdge()</code>	選択している輪郭の HalfEdge オブジェクト を返します。

Contour オブジェクトのプロパティ一覧

Contour オブジェクトで使用できるプロパティは次のとおりです。

プロパティ	説明
<code>contour.interior</code>	読み取り専用。輪郭内にエリアが含まれる場合は <code>true</code> で、含まれない場合は <code>false</code> です。
<code>contour.orientation</code>	読み取り専用。輪郭の方向を示す整数。

contour.getHalfEdge()

使用できるバージョン

Flash MX 2004

シンタックス

```
contour.getHalfEdge()
```

パラメータ

なし

戻り値

[HalfEdge オブジェクト](#)

説明

メソッド。選択している輪郭の [HalfEdge オブジェクト](#) を返します。

例

次の例では、選択したシェイプのすべての輪郭について、その頂点の座標を [出力] パネルに表示します。

// 選択したシェイプで

```
var elt = fl.getDocumentDOM().selection[0];
elt.beginEdit();

var contourArray = elt.contours;
var contourCount = 0;
for (i=0; i<contourArray.length; i++)
{
    var contour = contourArray[i];
    contourCount++;
    var he = contour.getHalfEdge();

    var iStart = he.id;
    var id = 0;
    while (id != iStart)
    {
        // 次の頂点を取得します。
        var vrt = he.getVertex();

        var x = vrt.x;
        var y = vrt.y;
        fl.trace("vrt: " + x + ", " + y);

        he = he.getNext();
        id = he.id;
    }
}
elt.endEdit();
```

contour.interior

使用できるバージョン

Flash MX 2004

シンタックス

contour.interior

説明

読み取り専用プロパティ。輪郭内にエリアが含まれる場合は true で、含まれない場合は false です。

例

次の例では、選択したシェイプのすべての輪郭について、各輪郭の interior プロパティの値を [出力] パネルに表示します。

```
var elt = fl.getDocumentDOM().selection[0];
elt.beginEdit();

var contourArray = elt.contours;

var contourCount = 0;
for (i=0; i<contourArray.length; i++) {
    var contour = contourArray[i];
    fl.trace("Next Contour, interior:" + contour.interior );
    contourCount++;
}
elt.endEdit();
```

contour.orientation

使用できるバージョン

Flash MX 2004

シンタックス

contour.orientation

説明

読み取り専用プロパティ。輪郭の方向を示す整数です。方向が反時計回りである場合は -1、時計回りである場合は 1、輪郭内にエリアが含まれない場合は 0 です。

例

次の例では、選択したシェイプのすべての輪郭について、各輪郭の orientation プロパティの値を [出力] パネルに表示します。

```
var elt = fl.getDocumentDOM().selection[0];
elt.beginEdit();

var contourArray = elt.contours;

var contourCount = 0;
for (i=0; i<contourArray.length; i++) {
    var contour = contourArray[i];
    fl.trace("Next Contour, orientation:" + contour.orientation);
    contourCount++;
}
elt.endEdit();
```

Document オブジェクト

使用できるバージョン

Flash MX 2004

説明

Document オブジェクトは、ステージを表します。つまり、FLA ファイルだけがドキュメントとして見なされます。現在のドキュメントの Document オブジェクトを返すには、`fl.getDocumentDOM()` を使用します。

Document オブジェクトのメソッド一覧

Document オブジェクトで使用できるメソッドは次のとおりです。

メソッド	説明
<code>document.addDataToDocument()</code>	指定したデータを、ドキュメントとともに保存します。
<code>document.addDataToSelection()</code>	指定したデータを、選択したオブジェクトとともに保存します。
<code>document.addFilter()</code>	選択したオブジェクトにフィルタを適用します。
<code>document.addItem()</code>	開いているドキュメントまたはライブラリから、指定した Document オブジェクトにアイテムを追加します。
<code>document.addNewLine()</code>	2 点間の新規のパスを追加します。
<code>document.addNewOval()</code>	指定した境界の矩形に新規の楕円オブジェクトを追加します。
<code>document.addNewPublishProfile()</code>	新規のパブリッシュプロファイルを追加し、現在のパブリッシュプロファイルとして設定します。
<code>document.addNewRectangle()</code>	新規の矩形または角丸矩形を追加し、指定した範囲に合わせます。
<code>document.addNewScene()</code>	新規のシーン (Timeline オブジェクト) を、現在選択しているシーンの後の次のシーンとして追加し、新規に追加したシーンを現在選択しているシーンとして設定します。
<code>document.addNewText()</code>	新規の空のテキストフィールドを挿入します。
<code>document.align()</code>	選択したオブジェクトを整列します。

メソッド	説明
<code>document.allowScreens()</code>	<code>document.screenOutline</code> プロパティを使用する前に、このメソッドを使用します。
<code>document.arrange()</code>	ステージで選択したオブジェクトを並べ替えます。
<code>document.breakApart()</code>	現在選択しているオブジェクトで分解処理を実行します。
<code>document.canEditSymbol()</code>	[シンボルの編集] メニューと機能が使用できるかどうかを示します。
<code>document.canRevert()</code>	<code>document.revert()</code> または <code>fl.revertDocument()</code> メソッドを正常に使用できるかどうかを判断します。
<code>document.canSaveAVersion()</code>	指定したドキュメントのバージョンを Version Cue サーバーに保存できるかどうかを判断します。
<code>document.canTestMovie()</code>	<code>document.testMovie()</code> メソッドを正常に使用できるかどうかを判断します。
<code>document.canTestScene()</code>	<code>document.testScene()</code> メソッドを正常に使用できるかどうかを判断します。
<code>document.changeFilterOrder()</code>	フィルタリストで、フィルタのインデックスを変更します。
<code>document.clipCopy()</code>	現在選択しているオブジェクトをドキュメントからクリップボードにコピーします。
<code>document.clipCut()</code>	現在選択しているオブジェクトをドキュメントから切り取り、クリップボードに書き込みます。
<code>document.clipPaste()</code>	クリップボードの内容をドキュメントにペーストします。
<code>document.close()</code>	指定したドキュメントを閉じます。
<code>document.convertLinesToFills()</code>	選択したオブジェクトの線を塗りに変換します。
<code>document.convertToSymbol()</code>	選択したステージアイテムを新規のシンボルに変換します。
<code>document.crop()</code>	選択した一番上の描画オブジェクトを使用して、選択した下位のすべての描画オブジェクトを切り抜きます。
<code>document.deleteEnvelope()</code>	選択したオブジェクトからエンベロープ (オブジェクトを含む境界ボックス) を削除します。
<code>document.deletePublishProfile()</code>	現在アクティブなプロファイルがある場合に削除します。

メソッド	説明
<code>document.deleteScene()</code>	現在のシーン (Timeline オブジェクト) を削除し、削除したシーンが最後のシーンでない場合は、次のシーンを現在のタイムラインオブジェクトとして設定します。
<code>document.deleteSelection()</code>	ステージで現在選択しているオブジェクトを削除します。
<code>document.disableAllFilters()</code>	選択したオブジェクトのすべてのフィルタを無効にします。
<code>document.disableFilter()</code>	フィルタリストで、指定したフィルタを無効にします。
<code>document.disableOtherFilters()</code>	フィルタリストで、指定した位置にあるフィルタ以外のすべてのフィルタを無効にします。
<code>document.distribute()</code>	選択したオブジェクトを配分します。
<code>document.distributeToLayers()</code>	現在選択しているオブジェクトでレイヤーに配分処理を実行します。[レイヤーに配分] を選択する操作と同じです。
<code>document.documentHasData()</code>	指定した名前の永続データがドキュメントにあるかどうかを確認します。
<code>document.duplicatePublishProfile()</code>	現在アクティブなプロファイルを複製し、複製したプロファイルにフォーカスを移動します。
<code>document.duplicateScene()</code>	現在選択しているシーンをコピーし、新規のシーンに固有の名前を付け、現在のシーンとして設定します。
<code>document.duplicateSelection()</code>	ステージで選択したオブジェクトを複製します。
<code>document.editScene()</code>	指定したシーンを編集するために、現在選択しているシーンとして設定します。
<code>document.enableAllFilters()</code>	選択したオブジェクトのフィルタリストですべてのフィルタを有効にします。
<code>document.enableFilter()</code>	選択したオブジェクトの指定したフィルタを有効にします。
<code>document.enterEditMode()</code>	オーサリングツールを、パラメータで指定した編集モードに切り替えます。
<code>document.exitEditMode()</code>	シンボル編集モードを終了し、編集モードから次の上位レベルにフォーカスを戻します。
<code>document.exportPNG()</code>	ドキュメントを PNG ファイルとして書き出します。
<code>document.exportPublishProfile()</code>	現在アクティブなプロファイルを XML ファイルに書き出します。

メソッド	説明
<code>document.exportSWF()</code>	ドキュメントを Flash SWF 形式で書き出します。
<code>document.getAlignToDocument()</code>	[整列] パネルにある [ステージを基準] ボタンの値を取得する操作と同じです。
<code>document.getBlendMode()</code>	選択したオブジェクトのブレンドモードを指定するストリングを返します。
<code>document.getCustomFill()</code>	選択したシェイプの塗りオブジェクト、またはツールパネルとプロパティインスペクタの塗りオブジェクト (指定した場合) を取得します。
<code>document.getCustomStroke()</code>	選択したシェイプの線オブジェクト、またはツールパネルとプロパティインスペクタの線オブジェクト (指定した場合) を返します。
<code>document.getDataFromDocument()</code>	指定したデータの値を取得します。
<code>document.getElementProperty()</code>	現在選択しているオブジェクトの指定した Element プロパティを取得します。
<code>document.getElementTextAttr()</code>	選択したテキストオブジェクトの指定した TextAttrs プロパティを取得します。
<code>document.getFilters()</code>	現在選択しているオブジェクトに適用されるフィルタのリストを含む配列を返します。
<code>document.getMetadata()</code>	ドキュメントと関連付けられる XML メタデータを含むストリングを返します。
<code>document.getMobileSettings()</code>	<code>document.setMobileSettings()</code> に渡されるストリングを返します。
<code>document.getPlayerVersion()</code>	指定したドキュメントの対象 Player バージョンを表すストリングを返します。
<code>document.getSelectionRect()</code>	現在選択しているオブジェクトの境界の矩形を取得します。
<code>document.getTextString()</code>	現在選択しているテキストを取得します。
<code>document.getTimeline()</code>	ドキュメントにある現在の Timeline オブジェクト を取得します。
<code>document.getTransformationPoint()</code>	現在選択しているオブジェクトの変形点の位置を取得します。
<code>document.group()</code>	現在選択しているオブジェクトをグループに変換します。
<code>document.importFile()</code>	ファイルをドキュメントに読み込みます。
<code>document.importPublishProfile()</code>	ファイルからプロファイルを読み込みます。

メソッド	説明
<code>document.importSWF()</code>	SWF ファイルをドキュメントに読み込みます。
<code>document.intersect()</code>	選択したすべての描画オブジェクトから交差描画オブジェクトを作成します。
<code>document.match()</code>	選択したオブジェクトのサイズを同じにします。
<code>document.mouseClick()</code>	選択ツールからマウスのクリックを実行します。
<code>document.mouseDoubleClick()</code>	選択ツールからマウスのダブルクリックを実行します。
<code>document.moveSelectedBezierPointsBy()</code>	最低1つのパスと最低1つのベジェポイントを選択している場合、選択したすべてのパスで選択したすべてのベジェポイントを、指定距離だけ移動します。
<code>document.moveSelectionBy()</code>	選択したオブジェクトを、指定距離だけ移動します。
<code>document.optimizeCurves()</code>	現在選択しているオブジェクトのスムージングを最適化します。複数処理を適用してスムージングの品質を上げることもできます。[修正]-[シェイプ]-[最適化]を選択する操作と同じです。
<code>document.publish()</code>	アクティブな [パブリッシュ設定]([ファイル]-[パブリッシュ設定])に従って、ドキュメントをパブリッシュします。[ファイル]-[パブリッシュ]を選択する操作と同じです。
<code>document.punch()</code>	選択した一番上の描画オブジェクトを使用して、選択した下位のすべての描画オブジェクトを型抜きします。
<code>document.removeAllFilters()</code>	選択したオブジェクトからすべてのフィルタを削除します。
<code>document.removeDataFromDocument()</code>	ドキュメントに追加されている指定した名前の永続データを削除します。
<code>document.removeDataFromSelection()</code>	選択したオブジェクトに追加されている指定した名前の永続データを削除します。
<code>document.removeFilter()</code>	選択したオブジェクトのフィルタリストから、指定したフィルタを削除します。
<code>document.renamePublishProfile()</code>	現在のプロファイルの名前を変更します。
<code>document.renameScene()</code>	[シーン] パネルで現在選択しているシーンの名前を変更します。
<code>document.reorderScene()</code>	指定したシーンを、指定した別のシーンの前に移動します。

メソッド	説明
<code>document.resetOvalObject()</code>	プロパティインスペクタのすべての値を Oval オブジェクトのデフォルト値に設定します。
<code>document.resetRectangleObject()</code>	プロパティインスペクタのすべての値を Rectangle オブジェクトのデフォルト値に設定します。
<code>document.resetTransformation()</code>	変換マトリックスをリセットします。[修正]-[変形]-[変形の解除]を選択する操作と同じです。
<code>document.revert()</code>	指定したドキュメントを前回保存した状態に復帰します。[ファイル]-[復帰]を選択する操作と同じです。
<code>document.revertToLastVersion()</code>	指定したドキュメントを Version Cue サーバーに保存されたバージョンのドキュメントに戻し、[出力]パネルにエラーをログします。
<code>document.rotateSelection()</code>	選択したオブジェクトを指定した角度だけ回転します。
<code>document.save()</code>	ドキュメントをデフォルトの場所に保存します。[ファイル]-[保存]を選択する操作と同じです。
<code>document.saveAndCompact()</code>	ファイルを保存して最適化します。[ファイル]-[保存して最適化]を選択する操作と同じです。
<code>document.saveAVersion()</code>	指定したドキュメントのバージョンを Version Cue サーバーに保存します。
<code>document.scaleSelection()</code>	選択したオブジェクトを、指定した量だけ拡大 / 縮小します。自由変形ツールを使用してオブジェクトを拡大 / 縮小する操作と同じです。
<code>document.selectAll()</code>	ステージにあるすべてのアイテムを選択します。 Ctrl + A (Windows) または Command + A (Macintosh) を押す操作、または [編集]-[すべて選択]を選択する操作と同じです。
<code>document.selectNone()</code>	選択したアイテムの選択を解除します。
<code>document.setAlignToDocument()</code>	<code>document.align()</code> 、 <code>document.distribute()</code> 、 <code>document.match()</code> 、および <code>document.space()</code> の環境設定を行います。[整列]パネルにある [ステージを基準] ボタンを有効にする操作と同じです。
<code>document.setBlendMode()</code>	選択したオブジェクトのブレンドモードを設定します。
<code>document.setCustomFill()</code>	ツールパネル、プロパティインスペクタ、および選択したシェイプの塗り設定を行います。
<code>document.setCustomStroke()</code>	ツールバー、プロパティインスペクタ、および選択したシェイプの線の設定を行います。

メソッド	説明
<code>document.setElementProperty()</code>	ドキュメントで選択したオブジェクトの指定した Element プロパティを設定します。
<code>document.setElementTextAttr()</code>	選択したテキストアイテムの指定した TextAttrs プロパティに指定値を設定します。
<code>document.setFillColor()</code>	選択したオブジェクトの塗りのカラーを指定したカラーに変更します。
<code>document.setFilterProperty()</code>	現在選択しているオブジェクトの指定したフィルタ プロパティを設定します。
<code>document.setFilters()</code>	選択したオブジェクトにフィルタを適用します。
<code>document.setInstanceAlpha()</code>	インスタンスの不透明度を設定します。
<code>document.setInstanceBrightness()</code>	インスタンスの明度を設定します。
<code>document.setInstanceTint()</code>	インスタンスの濃淡を設定します。
<code>document.setMetadata()</code>	指定したドキュメントの XML メタデータを設定します。既存のメタデータは上書きされます。
<code>document.setMobileSettings()</code>	モバイル FLA ファイルの XML 設定ストリングの値を設定します。
<code>document.setOvalObjectProperty()</code>	指定したプリミティブ Oval オブジェクトのプロパティ値を指定します。
<code>document.setPlayerVersion()</code>	指定したドキュメントの対象となる Flash Player のバージョンを設定します。
<code>document.setRectangleObjectProperty()</code>	指定したプリミティブ Rectangle オブジェクトのプロパティ値を指定します。
<code>document.setSelectionBounds()</code>	選択したオブジェクトを、1回の操作で移動しサイズ変更します。
<code>document.setSelectionRect()</code>	ステージを基準にして、指定した座標で矩形の選択マーキーを描画します。
<code>document.setStroke()</code>	選択した線のカラー、幅、スタイルを設定します。
<code>document.setStrokeColor()</code>	選択したオブジェクトの線のカラーを指定したカラーに変更します。
<code>document.setStrokeSize()</code>	選択したオブジェクトの線のサイズを指定したサイズに変更します。
<code>document.setStrokeStyle()</code>	選択したオブジェクトの線のスタイルを指定したスタイルに変更します。
<code>document.setTextRectangle()</code>	選択したテキストアイテムの境界の矩形を指定したサイズに変更します。

メソッド	説明
<code>document.setTextSelection()</code>	現在選択しているテキストフィールドのテキストに、 <i>startIndex</i> 値と <i>endIndex</i> 値で指定される値を設定します。
<code>document.setTextString()</code>	テキストのストリングを挿入します。
<code>document.setTransformationPoint()</code>	現在選択しているオブジェクトの変形点を移動します。
<code>document.skewSelection()</code>	選択したオブジェクトを指定した角度だけ傾斜します。
<code>document.smoothSelection()</code>	選択した塗りのアウトラインまたは曲線を滑らかにします。
<code>document.space()</code>	選択したオブジェクトを均等な間隔で配置します。
<code>document.straightenSelection()</code>	現在選択している線を直線にします。ツールパネルの [ストレート] ボタンを使用する操作と同じです。
<code>document.swapElement()</code>	現在選択しているオブジェクトを、指定したオブジェクトと入れ替えます。
<code>document.swapStrokeAndFill()</code>	線と塗りのカラーを入れ替えます。
<code>document.synchronizeWithHeadVersion()</code>	指定したドキュメントを Version Cue サーバーにある最新バージョンと同期し、[出力] パネルにエラーをログします。
<code>document.testMovie()</code>	ドキュメントで [ムービープレビュー] を実行します。
<code>document.testScene()</code>	ドキュメントの現在のシーンで [シーンプレビュー] を実行します。
<code>document.traceBitmap()</code>	現在選択しているオブジェクトでビットマップのトレースを実行します。[修正]-[ビットマップ]-[ビットマップのトレース] を選択する操作と同じです。
<code>document.transformSelection()</code>	現在選択しているオブジェクトで、パラメータで指定したマトリックスを適用して、変形を実行します。
<code>document.unGroup()</code>	現在選択しているオブジェクトのグループを解除します。
<code>document.union()</code>	選択したすべてのシェイプを描画オブジェクトに結合します。
<code>document.unlockAllElements()</code>	現在選択しているフレームでロックされているすべてのエレメントをロック解除します。
<code>document.xmlPanel()</code>	[XMLUI] ダイアログボックスを表示します。

Document オブジェクトのプロパティ一覧

Document オブジェクトで利用できるプロパティは次のとおりです。

プロパティ	説明
<code>document.accName</code>	[アクセシビリティ] パネルの [名前] フィールドと等しい文字列です。
<code>document.as3AutoDeclare</code>	ステージ上に置かれたインスタンスをユーザー定義のタイムラインクラスに自動的に追加するかどうかを示すブール値です。
<code>document.as3Dialect</code>	指定したドキュメントで使用する ActionScript 3.0 の " 言語 " を表す文字列です。
<code>document.as3ExportFrame</code>	ActionScript 3.0 クラスをエクスポートするフレームを指定する整数です。
<code>document.as3StrictMode</code>	ActionScript 3.0 コンパイラでのコンパイルを Strict モードで行うか否かを指定するブール値です。
<code>document.as3WarningsMode</code>	ActionScript 3.0 コンパイラでのコンパイルを Warnings モードで行うか否かを指定するブール値です。
<code>document.asVersion</code>	指定したファイルで使用する ActionScript のバージョンを指定する整数です。
<code>document.autoLabel</code>	[アクセシビリティ] パネルの [自動ラベル] チェックボックスと等しいブール値です。
<code>document.backgroundColor</code>	背景色を表す文字列、16 進数値、または整数です。
<code>document.currentPublishProfile</code>	指定したドキュメントのアクティブなパブリッシュプロファイルの名前を指定する文字列です。
<code>document.currentTimeline</code>	アクティブなタイムラインのインデックスを指定する整数です。
<code>document.description</code>	[アクセシビリティ] パネルの [説明] フィールドと等しい文字列です。
<code>document.docClass</code>	ドキュメントに関連付けられたトップレベルの ActionScript 3.0 クラスを指定します。
<code>document.forceSimple</code>	指定したオブジェクトの子がアクセス可能かどうかを指定するブール値です。
<code>document.frameRate</code>	SWF ファイルの再生時に 1 秒間に表示されるフレーム数を指定する浮動小数値です。デフォルトは 12 です。
<code>document.height</code>	ドキュメント (ステージ) の高さをピクセル単位で指定する整数です。

プロパティ	説明
<code>document.id</code>	Flash セッションの実行中に各ドキュメントを識別するために自動的に割り当てられる一意の整数です。
<code>document.library</code>	読み取り専用。ドキュメントの library オブジェクト です。
<code>document.livePreview</code>	[ライブプレビュー] を有効にするかどうかを指定するブール値です。
<code>document.name</code>	読み取り専用。ドキュメント (FLA ファイル) の名前を表すストリングです。
<code>document.path</code>	読み取り専用。ドキュメントのパスを表すストリングです。
<code>document.publishProfiles</code>	読み取り専用。ドキュメントのパブリッシュプロファイル名の配列です。
<code>document.screenOutline</code>	読み取り専用。ドキュメントの現在の ScreenOutline オブジェクト です。
<code>document.selection</code>	ドキュメントで選択したオブジェクトの配列です。
<code>document.silent</code>	オブジェクトがアクセス可能かどうかを指定するブール値です。
<code>document.timelines</code>	読み取り専用。Timeline オブジェクトの配列です (Timeline オブジェクト を参照してください)。
<code>document.viewMatrix</code>	読み取り専用。 Matrix オブジェクト 。
<code>document.width</code>	ドキュメント (ステージ) の幅をピクセル単位で指定する整数です。
<code>document.zoomFactor</code>	オーサリング時にステージのズーム率を指定します。

document.accName

使用できるバージョン

Flash MX 2004

シンタックス

`document.accName`

説明

プロパティ。[アクセシビリティ] パネルの [名前] フィールドと等しいストリングです。スクリーンリーダーは、名前を読み上げてオブジェクトを識別します。

例

次の例では、ドキュメントのアクセシビリティ名に "Main Movie" を設定します。

```
fl.getDocumentDOM().accName = "Main Movie";
```

次の例では、ドキュメントのアクセシビリティ名を取得します。

```
fl.trace(fl.getDocumentDOM().accName);
```

document.addDataToDocument()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.addDataToDocument(name, type, data)
```

パラメータ

name 追加するデータの名前を指定するストリング。

type 追加するデータの型を定義するストリング。指定できる値は、"integer"、"integerArray"、"double"、"doubleArray"、"string"、および "byteArray" です。

data 追加する値。使用できるデータ型は *type* パラメータにより異なります。

戻り値

なし

説明

メソッド。指定したデータを、ドキュメントとともに保存します。データが FLA ファイルに書き込まれます。ファイルを再び開くと JavaScript で使用できます。

例

次の例では、整数値 12 を現在のドキュメントに追加します。

```
fl.getDocumentDOM().addDataToDocument("myData", "integer", 12);
```

次の例では、"myData" という名前のデータを返し、結果を [出力] パネルに表示します。

```
fl.trace(fl.getDocumentDOM().getDataFromDocument("myData"));
```

関連項目

[document.getDataFromDocument\(\)](#), [document.removeDataFromDocument\(\)](#)

document.addDataToSelection()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.addDataToSelection(name, type, data)
```

パラメータ

name 永続データの名前を指定するストリング。

type データ型を定義します。指定できる値は、"integer"、"integerArray"、"double"、"doubleArray"、"string"、および "byteArray" です。

data 追加する値。使用できるデータ型は *type* パラメータにより異なります。

戻り値

なし

説明

メソッド。指定したデータを、選択したオブジェクトとともに保存します。データが FLA ファイルに書き込まれます。ファイルを再び開くと JavaScript で使用できます。永続データがサポートされるのはシンボルとビットマップだけです。

例

次の例では、選択したオブジェクトに整数値 12 を追加します。

```
fl.getDocumentDOM().addDataToSelection("myData", "integer", 12);
```

関連項目

[document.removeDataFromSelection\(\)](#)

document.addFilter()

使用できるバージョン

Flash 8

シンタックス

```
document.addFilter(filterName)
```

パラメータ

filterName フィルタリストに追加し、選択したオブジェクトに対して有効にするフィルタを指定するストリング。指定できる値は、"adjustColorFilter"、"bevelFilter"、"blurFilter"、"dropShadowFilter"、"glowFilter"、"gradientBevelFilter"、および"gradientGlowFilter"です。

戻り値

なし

説明

メソッド。選択したオブジェクトにフィルタを適用し、フィルタリストの最後にそのフィルタを格納します。

例

次の例では、選択したオブジェクトにグローフィルタを適用します。

```
fl.getDocumentDOM().addFilter("glowFilter");
```

関連項目

[document.changeFilterOrder\(\)](#), [document.disableFilter\(\)](#),
[document.enableFilter\(\)](#), [document.getFilters\(\)](#), [document.removeFilter\(\)](#),
[document.setBlendMode\(\)](#), [document.setFilterProperty\(\)](#)

document.addItem()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.addItem(position, item)
```

パラメータ

position アイテムを追加する位置の x 座標と y 座標を指定するポイント。シンボルの中心または、ビットマップまたはビデオの左上隅を使用します。

item 追加するアイテムと追加元のライブラリを指定する Item オブジェクト。[Item オブジェクト](#) を参照してください。

戻り値

ブール値。成功した場合は true を返します。それ以外の場合は false を返します。

説明

メソッド。開いているドキュメントまたはライブラリから、指定した `Document` オブジェクトにアイテムを追加します。

例

次の例では、ライブラリの最初のアイテムを、選択したシンボル、ビットマップまたはビデオの指定した位置にある最初のドキュメントに追加します。

```
var item = fl.documents[0].library.items[0];
fl.documents[0].addItem({x:0,y:0}, item);
```

次の例では、現在のドキュメントのライブラリにあるシンボル `myMovieClip` を、現在のドキュメントに追加します。

```
var itemIndex = fl.getDocumentDOM().library.findItemIndex("myMovieClip");
var theItem = fl.getDocumentDOM().library.items[itemIndex];
fl.getDocumentDOM().addItem({x:0,y:0}, theItem);
```

次の例では、ドキュメント配列の 2 番目のドキュメントにあるシンボル `myMovieClip` を、ドキュメント配列の 3 番目のドキュメントに追加します。

```
var itemIndex = fl.documents[1].library.findItemIndex("myMovieClip");
var theItem = fl.documents[1].library.items[itemIndex];
fl.documents[2].addItem({x:0,y:0}, theItem);
```

document.addNewLine()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.addNewLine(startPoint, endpoint)
```

パラメータ

startpoint 線の開始点を示す *x* 座標と *y* 座標を指定する浮動小数値のペア。

endpoint 線の終了点を示す *x* 座標と *y* 座標を指定する浮動小数値のペア。

戻り値

なし

説明

メソッド。2 点間の新規のパスを追加します。このメソッドでは、ドキュメントの現在の線の属性を使用して、現在のフレームと現在のレイヤーにパスを追加します。このメソッドは、線ツールをクリックして線を描画する操作と同じように機能します。

例

次の例では、指定した開始点と終了点との間に線を追加します。

```
fl.getDocumentDOM().addNewLine({x:216.7, y:122.3}, {x:366.8, y:165.8});
```

document.addNewOval()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.addNewOval(boundingRectangle [, bSuppressFill [, bSuppressStroke]])
```

パラメータ

boundingRectangle 追加する楕円の範囲を指定する矩形。*boundingRectangle* の形式の詳細については、[document.addNewRectangle\(\)](#) を参照してください。

bSuppressFill ブール値。true を設定すると、塗りなしのシェイプを作成します。デフォルト値は false です。このパラメータはオプションです。

bSuppressStroke ブール値。true を設定すると、線なしのシェイプを作成します。デフォルト値は false です。このパラメータはオプションです。

戻り値

なし

説明

メソッド。指定した境界の矩形に新規の楕円オブジェクトを追加します。このメソッドは、楕円ツールと同じ操作を実行します。このメソッドでは、ドキュメントの現在のデフォルトの線の属性と塗りの属性を使用して、現在のフレームとレイヤーに楕円を追加します。*bSuppressFill* に true を設定すると、塗りなしの楕円を描画します。*bSuppressStroke* に true を設定すると、線なしの楕円を描画します。*bSuppressFill* と *bSuppressStroke* の両方に true を設定すると、このメソッドは何もしません。

例

次の例では、指定した座標内に新規の楕円を追加します。楕円の幅は 164 ピクセルで、高さは 178 ピクセルです。

```
flash.getDocumentDOM().addNewOval({left:72,top:50,right:236,bottom:228});
```

次の例では、塗りなしの楕円を描画します。

```
flash.getDocumentDOM().addNewOval({left:72,top:50,right:236,bottom:228}, true);
```

次の例では、線なしの楕円を描画します。

```
flash.getDocumentDOM().addNewOval({left:72,top:50,right:236,bottom:228}, false,  
    true);
```

document.addNewPublishProfile()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.addNewPublishProfile([profileName])
```

パラメータ

profileName 新規プロファイルの一意の名前。名前を指定しない場合は、デフォルト名が使用されます。このパラメータは省略可能です。

戻り値

プロファイルリストの新規プロファイルのインデックスである整数です。新規プロファイルを作成できない場合は -1 を返します。

説明

メソッド。新規のパブリッシュプロファイルを追加し、現在のパブリッシュプロファイルとして設定します。

例

次の例では、デフォルト名の新規パブリッシュプロファイルを追加し、プロファイル名を [出力] パネルに表示します。

```
fl.getDocumentDOM().addNewPublishProfile();  
fl.outputPanel.trace(fl.getDocumentDOM().currentPublishProfile());
```

次の例では、新規パブリッシュプロファイルを "my profile" という名前で追加します。

```
fl.getDocumentDOM().addNewPublishProfile("my profile");
```

関連項目

[document.deletePublishProfile\(\)](#)

document.addNewRectangle()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.addNewRectangle(boundingRectangle, roundness  
    [, bSuppressFill [, bSuppressStroke]])
```

パラメータ

boundingRectangle 新規の矩形を追加する範囲の境界を指定する矩形。

{left:value1,top:value2,right:value3,bottom:value4} という形式で指定します。left および top の値で、左上隅の位置を指定します (たとえば、left:0,top:0 はステージの左上を表します)。right および bottom の値で右下隅の位置を指定します。したがって、矩形の幅は left と right の値の差となり、矩形の高さは top と bottom の値の差となります。

つまり、矩形の境界が、プロパティインスペクタに表示されるすべての値と対応するわけではありません。left および top の値は、それぞれプロパティインスペクタの X と Y の値に対応します。ただし、right および bottom の値は、プロパティインスペクタの W と H の値に対応しません。たとえば、次の境界を持つ矩形について考えてみましょう。

```
{left:10,top:10,right:50,bottom:100}
```

この矩形では、プロパティインスペクタに次の値が表示されます。

X = 10, Y = 10, W = 40, H = 90

roundness 角に適用する丸みを指定する 0 ~ 999 の整数値。値は、ポイント数で指定します。値が大きいほど、丸みが大きくなります。

bSuppressFill ブール値。true を設定すると、塗りなしのシェイプを作成します。デフォルト値は false です。このパラメータはオプションです。

bSuppressStroke ブール値。true を設定すると、線なしの矩形を作成します。デフォルト値は false です。このパラメータはオプションです。

戻り値

なし

説明

メソッド。新規の矩形または角丸矩形を追加し、指定した範囲に合わせます。このメソッドは、矩形ツールと同じ操作を実行します。このメソッドでは、ドキュメントの現在のデフォルト線の属性と塗りの属性を使用して、現在のフレームとレイヤーに矩形を追加します。*bSuppressFill* パラメータに true を設定すると、塗りなしの矩形を描画します。*bSuppressStroke* パラメータに true を設定すると、線なしの矩形を描画します。*bSuppressFill* と *bSuppressStroke* の両方に true を設定すると、このメソッドは何もしません。

例

次の例では、角の丸みのない新規の矩形を指定した座標内に追加します。矩形の幅と高さはともに 100 ピクセルです。

```
flash.getDocumentDOM().addNewRectangle({left:0,top:0,right:100,bottom:100},0);
```

次の例では、角の丸みのない、塗りなしの新規の矩形を追加します。矩形の幅は 100 ピクセル、高さは 200 ピクセルです。

```
flash.getDocumentDOM().addNewRectangle({left:10,top:10,right:110,bottom:210},0,true);
```

次の例では、角の丸みのない、線なしの新規の矩形を追加します。矩形の幅は 200 ピクセル、高さは 100 ピクセルです。

```
flash.getDocumentDOM().addNewRectangle({left:20,top:20,right:220,bottom:120},0,false,true);
```

document.addNewScene()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.addNewScene([name])
```

パラメータ

name シーンの名前を指定します。名前を指定しない場合は、新規のシーン名が生成されます。

戻り値

ブール値。シーンが正常に追加された場合は true を返します。追加されなかった場合は false を返します。

説明

メソッド。新規のシーン ([Timeline オブジェクト](#)) を、現在選択しているシーンの後の次のシーンとして追加し、新規に追加したシーンを現在選択しているシーンとして設定します。指定したシーン名が既に存在する場合は、シーンは追加されず、メソッドはエラーを返します。

例

次の例では、現在のドキュメントの現在のシーンの後に、myScene という名前の新規シーンを追加します。新規シーンが作成されると、変数 success は true になり、作成されなかった場合は false になります。

```
var success = flash.getDocumentDOM().addNewScene("myScene");
```

次の例では、デフォルトの命名規則を使用して新規シーンを追加します。シーンが1つだけ存在する場合、新規に作成されたシーン名前は "Scene 2" になります。

```
fl.getDocumentDOM().addNewScene();
```

document.addNewText()

使用できるバージョン

Flash MX 2004。Flash CS3 Professional でオプションの *text* パラメータが追加されました。

シンタックス

```
document.addNewText(boundingRectangle [, text ])
```

パラメータ

boundingRectangle テキストフィールドのサイズと位置を指定します。*boundingRectangle* の形式の詳細については、[document.addNewRectangle\(\)](#) を参照してください。

text フィールド内に配置するテキストを指定するオプションのストリング。このパラメータを省略すると、ツールパネルで選択したツールがテキストツールに切り替わります。したがって、選択したツールを変更しない場合は、*text* の値を渡します。

戻り値

なし。

説明

メソッド。新しいテキストフィールドを挿入し、オプションでフィールド内にテキストを配置します。*text* パラメータを省略する場合は、[document.setTextString\(\)](#) を呼び出してテキストフィールドにテキストを挿入できます。

例

次の例では、ステージの左上隅に新規のテキストフィールドを作成し、テキストストリングに "Hello World" を設定します。

```
fl.getDocumentDOM().addNewText({left:0, top:0, right:100, bottom:100} , "Hello  
World!" );  
fl.getDocumentDOM().setTextString('Hello World!');
```

関連項目

[document.setTextString\(\)](#)

document.align()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.align(alignmode [, bUseDocumentBounds])
```

パラメータ

alignmode 選択したオブジェクトを整列する方法を指定するストリング。指定できる値は、"left"、"right"、"top"、"bottom"、"vertical center"、および "horizontal center" です。

bUseDocumentBounds ブール値。true を設定すると、ドキュメントの領域を基準に整列します。それ以外の場合は、選択したオブジェクトの境界を使用します。デフォルト値は false です。このパラメータはオプションです。

戻り値

なし

説明

メソッド。選択したオブジェクトを整列します。

例

次の例では、ステージの左側にオブジェクトを整列します。[整列] パネルにある [ステージを基準] をオンにして、[左揃え] ボタンをクリックした場合と同じです。

```
fl.getDocumentDOM().align("left", true);
```

関連項目

[document.distribute\(\)](#), [document.getAlignToDocument\(\)](#),
[document.setAlignToDocument\(\)](#)

document.allowScreens()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.allowScreens()
```

パラメータ

なし

戻り値

ブール値。document.screenOutline を正常に使用できる場合は true、正常に使用できない場合は false を返します。

説明

メソッド。document.screenOutline プロパティを使用する前に使用します。このメソッドが true を返した場合は、document.screenOutline に正常にアクセスできます。スクリーンを使用しないドキュメントで document.screenOutline にアクセスすると、エラーが表示されます。

例

次の例では、現在のドキュメントで screens メソッドが使用できるかどうかを確認します。

```
if(f1.getDocumentDOM().allowScreens()) {  
    f1.trace("screen outline is available.");  
}  
else {  
    f1.trace("whoops, no screens.");  
}
```

関連項目

[document.screenOutline](#)

document.arrange()

使用できるバージョン

Flash MX 2004

シンタックス

document.arrange(*arrangeMode*)

パラメータ

arrangeMode 選択したオブジェクトを移動する方向を指定します。指定できる値は、"back"、"backward"、"forward"、および "front" です。[修正]-[整列]メニューのオプションと同じ機能を持ちます。

戻り値

なし。

説明

メソッド。ステージで選択したオブジェクトを並べ替えます。このメソッドは、シェイプ以外のオブジェクトに適用されます。

例

次の例では、現在選択しているオブジェクトを前面に移動します。

```
fl.getDocumentDOM().arrange("front");
```

document.as3AutoDeclare

使用できるバージョン

Flash CS3 Professional

シンタックス

```
document.as3AutoDeclare
```

説明

プロパティ。ステージ上に置かれたインスタンスをユーザー定義のタイムラインクラスに自動的に追加するかどうかを示すブール値です。デフォルト値は `true` です。

例

次の例では、現在のドキュメントでステージ上に配置されたインスタンスをユーザー定義のタイムラインクラスに手動で追加する必要があります。

```
fl.getDocumentDOM().as3AutoDeclare=false;
```

document.as3Dialect

使用できるバージョン

Flash CS3 Professional

シンタックス

```
document.as3Dialect
```

説明

プロパティ。指定したドキュメントで使用する ActionScript 3.0 の " 言語 " を表す文字列です。デフォルト値は "AS3" です。以前の ECMAScript 仕様で許可されていたように、プロトタイプクラスを許可する場合は、この値を "ES" に設定します。

例

次の例では、現在のドキュメントで使用している言語は ECMAScript であることを指定します。

```
fl.getDocumentDOM().as3Dialect="ES";
```

関連項目

[document.asVersion](#)

document.as3ExportFrame

使用できるバージョン

Flash CS3 Professional

シンタックス

`document.as3ExportFrame`

説明

プロパティ。ActionScript 3.0 クラスをエクスポートするフレームを指定する整数です。デフォルトでは、フレーム 1 でクラスをエクスポートします。

例

次の例では、クラスをエクスポートするフレームを 1 (デフォルト) から 5 に変更します。

```
var myDocument = fl.getDocumentDOM();
fl.outputPanel.trace("'Export classes in frame:' value before modification is "
    + myDocument.as3ExportFrame);
myDocument.as3ExportFrame = 5;
fl.outputPanel.trace("'Export classes in frame:' value after modification is " +
    myDocument.as3ExportFrame);
```

document.as3StrictMode

使用できるバージョン

Flash CS3 Professional

シンタックス

`document.as3StrictMode`

説明

プロパティ。ActionScript 3.0 コンパイラで、Strict モードオプションをオンにしてコンパイルする (true) か、オフでコンパイルする (false) かを指定するブール値です。Strict モードでは警告がエラーとして報告され、このようなエラーが存在すると、コンパイルは成功しません。デフォルト値は true です。

例

次の例では、Strict モードコンパイラオプションをオフにします。

```
var myDocument = fl.getDocumentDOM();
fl.outputPanel.trace("Strict Mode value before modification is " +
    myDocument.as3StrictMode);
myDocument.as3StrictMode = false;
```

```
fl.outputPanel.trace("Strict Mode value after modification is " +  
    myDocument.as3StrictMode);
```

関連項目

[document.as3WarningsMode](#)

document.as3WarningsMode

使用できるバージョン

Flash CS3 Professional

シンタックス

`document.as3WarningsMode`

説明

プロパティ。ActionScript 3.0 コンパイラで、Warnings モードオプションをオンにしてコンパイルする (true) か、オフでコンパイルする (false) かを指定するブール値です。Warnings モードでは追加の警告が報告され、ActionScript 2.0 のコードを ActionScript 3.0 に更新するときの非互換性の検出に役立ちます。デフォルト値は true です。

例

次の例では、Warnings モードコンパイラオプションをオフにします。

```
var myDocument = fl.getDocumentDOM();  
fl.outputPanel.trace("Warnings Mode value before modification is " +  
    myDocument.as3WarningsMode);  
myDocument.as3WarningsMode = false;  
fl.outputPanel.trace("Warnings Mode value after modification is " +  
    myDocument.as3WarningsMode);
```

関連項目

[document.as3StrictMode](#)

document.asVersion

使用できるバージョン

Flash CS3 Professional

シンタックス

`document.asVersion`

説明

プロパティ。指定したドキュメントで使用する ActionScript のバージョンを指定する整数です。指定できる値は 1、2、および 3 です。

指定したドキュメントの対象となる Player バージョンを判断するには、`document.getPlayerVersion()` を使用します。このメソッドは文字列を返し、Flash Lite プレーヤーで使用できます。

例

次の例では、現在の設定が ActionScript 1.0 である場合、現在のドキュメントの ActionScript バージョンを ActionScript 2.0 に設定します。

```
if(fl.getDocumentDOM().asVersion == 1){  
    fl.getDocumentDOM().asVersion = 2;  
}
```

関連項目

[document.as3Dialect](#), [document.getPlayerVersion\(\)](#)

document.autoLabel

使用できるバージョン

Flash MX 2004

シンタックス

`document.autoLabel`

説明

プロパティ。[アクセシビリティ] パネルの [自動ラベル] チェックボックスと等しいブール値です。ステージのオブジェクトに、そのオブジェクトに関連付けられているテキストを使用して、自動的にラベルを付けることができます。

例

次の例では、autoLabel プロパティの値を取得し、結果を [出力] パネルに表示します。

```
var isAutoLabel = fl.getDocumentDOM().autoLabel;  
fl.trace(isAutoLabel);
```

次の例では、autoLabel プロパティに true を設定し、ステージのオブジェクトに自動的にラベルを付けます。

```
fl.getDocumentDOM().autoLabel = true;
```

document.backgroundColor

使用できるバージョン

Flash MX 2004

シンタックス

`document.backgroundColor`

説明

プロパティ。背景色。次のいずれかの形式です。

- "#RRGGBB" または "#RRGGBBAA" 形式のストリング
- 0xRRGGBB 形式の 16 進数
- 16 進数と等しい 10 進表現の整数

例

次の例では、背景色に黒を設定します。

```
fl.getDocumentDOM().backgroundColor = '#000000';
```

document.breakApart()

使用できるバージョン

Flash MX 2004

シンタックス

`document.breakApart()`

パラメータ

なし

戻り値

なし

説明

メソッド。現在選択しているオブジェクトで分解処理を実行します。

例

次の例では、現在選択しているオブジェクトを分解します。

```
fl.getDocumentDOM().breakApart();
```

document.canEditSymbol()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.canEditSymbol()
```

パラメータ

なし

戻り値

ブール値。[シンボルの編集] メニューと機能が使用できる場合は true を返し、使用できない場合は false を返します。

説明

メソッド。[シンボルの編集] メニューと機能が使用できるかどうかを示します。選択したオブジェクトが編集できるかどうかには関係ありません。このメソッドは、
fl.getDocumentDOM().enterEditMode() が使用できるかどうかの確認には使用しないでください。

例

次の例では、[シンボルの編集] メニューと機能の状態を [出力] パネルに表示します。

```
fl.trace("fl.getDocumentDOM().canEditSymbol() returns: " +  
    fl.getDocumentDOM().canEditSymbol());
```

document.canRevert()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.canRevert()
```

パラメータ

なし

戻り値

ブール値。document.revert() メソッドまたは fl.revertDocument() メソッドを正常に使用できる場合は true を返し、使用できない場合は false を返します。

説明

メソッド。 `document.revert()` または `fl.revertDocument()` メソッドを正常に使用できるかどうかを判断します。

例

次の例では、現在のドキュメントを、前回保存した状態に復帰できるかどうかを確認します。復帰できる場合は、 `fl.getDocumentDOM().revert()` で前回保存した状態に戻します。

```
if(fl.getDocumentDOM().canRevert()){
    fl.getDocumentDOM().revert();
}
```

document.canSaveAVersion()

使用できるバージョン

Flash CS3 Professional

シンタックス

```
document.canSaveAVersion()
```

パラメータ

なし

戻り値

ブール値。ファイルのバージョンを Version Cue サーバーに保存できる場合は `true`、それ以外の場合は `false` を返します。

説明

メソッド。指定したドキュメントのバージョンを Version Cue サーバーに保存できるかどうかを判断します。

例

次の例では、 `document.saveAVersion()` を使用できるかどうかをテストします。使用できる場合は、メソッドを呼び出します。

```
if(fl.getDocumentDOM().canSaveAVersion()){
    fl.getDocumentDOM().saveAVersion;
}
```

関連項目

[document.revertToLastVersion\(\)](#), [document.saveAVersion\(\)](#)

document.canTestMovie()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.canTestMovie()
```

パラメータ

なし

戻り値

ブール値。[document.testMovie\(\)](#) メソッドを正常に使用できる場合は true、それ以外の場合は false を返します。

説明

メソッド。[document.testMovie\(\)](#) メソッドを正常に使用できるかどうかを判断します。

例

次の例では、`fl.getDocumentDOM().testMovie()` が使用できるかどうかを確認します。使用できる場合は、メソッドを呼び出します。

```
if(fl.getDocumentDOM().canTestMovie()){  
    fl.getDocumentDOM().testMovie();  
}
```

関連項目

[document.canTestScene\(\)](#), [document.testScene\(\)](#)

document.canTestScene()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.canTestScene()
```

パラメータ

なし

戻り値

ブール値。document.testScene() メソッドを正常に使用できる場合は true を返し、使用できない場合は false を返します。

説明

メソッド。document.testScene() メソッドを正常に使用できるかどうかを判断します。

例

次の例では、fl.getDocumentDOM().testScene() を正常に使用できるかどうかを先に確認します。使用できる場合は、メソッドを呼び出します。

```
if(fl.getDocumentDOM().canTestScene()){
    fl.getDocumentDOM().testScene();
}
```

関連項目

[document.canTestMovie\(\)](#), [document.testMovie\(\)](#)

document.changeFilterOrder()

使用できるバージョン

Flash 8

シンタックス

document.changeFilterOrder(*oldIndex*, *newIndex*)

パラメータ

oldIndex フィルタリストで位置を変更するフィルタのゼロから始まるインデックス位置を表す整数。

newIndex リスト内のフィルタの新規インデックス位置を表す整数。

戻り値

なし

説明

メソッド。フィルタリストで、フィルタのインデックスを変更します。*newIndex* より上または下のフィルタは、適宜上下にシフトされます。たとえば、次に示すフィルタを使用する場合、コマンド fl.getDocumentDOM().changeFilterOrder(3, 0) を実行すると、フィルタは次に示すように並べ替えられます。

実行前: blurFilter、dropShadowFilter、glowFilter、gradientBevelFilter

実行後: gradientBevelFilter、blurFilter、dropShadowFilter、glowFilter

ここで、コマンド `fl.getDocumentDOM().changeFilterOrder(0, 2)` を実行すると、フィルタは次のように並べ替えられます。

実行前: `gradientBevelFilter`、`blurFilter`、`dropShadowFilter`、`glowFilter`

実行後: `blurFilter`、`dropShadowFilter`、`gradientBevelFilter`、`glowFilter`

例

次の例では、フィルタリスト内の現在の位置が 2 番目のフィルタを 1 番目に移動します。

```
fl.getDocumentDOM().changeFilterOrder(1,0);
```

関連項目

[document.addFilter\(\)](#)、[document.disableFilter\(\)](#)、[document.enableFilter\(\)](#)、[document.getFilters\(\)](#)、[document.removeFilter\(\)](#)、[Filter](#) オブジェクト

document.clipCopy()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.clipCopy()
```

パラメータ

なし

戻り値

なし

説明

メソッド。現在選択しているオブジェクトをドキュメントからクリップボードにコピーします。

ストリングをクリップボードにコピーするには、[fl.clipCopyString\(\)](#) を使用します。

例

次の例では、現在選択しているオブジェクトをドキュメントからクリップボードにコピーします。

```
fl.getDocumentDOM().clipCopy();
```

関連項目

[document.clipCut\(\)](#)、[document.clipPaste\(\)](#)

document.clipCut()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.clipCut()
```

パラメータ

なし

戻り値

なし。

説明

メソッド。現在選択しているオブジェクトをドキュメントから切り取り、クリップボードに書き込みます。

例

次の例では、現在選択しているオブジェクトをドキュメントから切り取り、クリップボードに書き込みます。

```
fl.getDocumentDOM().clipCut();
```

関連項目

[document.clipCopy\(\)](#), [document.clipPaste\(\)](#), [fl.clipCopyString\(\)](#)

document.clipPaste()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.clipPaste([bInPlace])
```

パラメータ

bInPlace ブール値。trueを設定すると、同じ位置にペースト処理を実行します。デフォルト値はfalseで、ドキュメントの中心でペースト処理を実行します。このパラメータはオプションです。

戻り値

なし

説明

メソッド。クリップボードの内容をドキュメントにペーストします。

例

次の例では、クリップボードの内容をドキュメントの中心にペーストします。

```
fl.getDocumentDOM().clipPaste();
```

次の例では、クリップボードの内容を、現在のドキュメントの同じ位置にペーストします。

```
fl.getDocumentDOM().clipPaste(true);
```

関連項目

[document.clipCopy\(\)](#), [document.clipCut\(\)](#), [fl.clipCopyString\(\)](#)

document.close()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.close([bPromptToSaveChanges])
```

パラメータ

bPromptToSaveChanges ブール値。true を設定すると、保存していない変更がドキュメントにないかどうかを確認するためのプロンプトを表示します。*bPromptToSaveChanges* に false を設定すると、変更したドキュメントを保存するためのプロンプトは表示されません。デフォルト値は true です。このパラメータはオプションです。

戻り値

なし

説明

メソッド。指定したドキュメントを閉じます。

例

次の例では、現在のドキュメントを閉じて、変更を保存するためのプロンプトを表示します。

```
fl.getDocumentDOM().close();
```

次の例では、変更を保存せずに現在のドキュメントを閉じます。

```
fl.getDocumentDOM().close(false);
```

document.convertLinesToFills()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.convertLinesToFills()
```

パラメータ

なし

戻り値

なし。

説明

メソッド。選択したオブジェクトの線を塗りに変換します。

例

次の例では、現在選択している線を塗りに変換します。

```
fl.getDocumentDOM().convertLinesToFills();
```

document.convertToSymbol()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.convertToSymbol(type, name, registrationPoint)
```

パラメータ

type 作成するシンボルのタイプを指定するストリング。有効な値は、"movie clip"、"button"、および "graphic" です。

name 新規シンボルの一意の名前を指定するストリング。空のストリングを送信すると、一意のシンボル名をメソッドで作成することができます。

registration point シンボルの 0,0 位置を表す点を指定します。指定できる値は、"top left"、"top center"、"top right"、"center left"、"center"、"center right"、"bottom left"、"bottom center"、および "bottom right" です。

戻り値

新規に作成されたシンボルのオブジェクトを返します。シンボルを作成できない場合は `null` を返します。

説明

メソッド。選択したステージアイテムを新規のシンボルに変換します。シンボルのリンケージと共有アセットプロパティの定義の詳細については、[Item オブジェクト](#) を参照してください。

例

次の例では、指定した名前のムービークリップシンボル、指定した名前のボタンシンボル、およびデフォルト名のムービークリップシンボルを作成します。

```
newMc = fl.getDocumentDOM().convertToSymbol("movie clip", "mcSymbolName", "top left");
newButton = fl.getDocumentDOM().convertToSymbol("button", "btnSymbolName", "bottom right");
newClipWithDefaultName = fl.getDocumentDOM().convertToSymbol("movie clip", "", "top left");
```

document.crop()

使用できるバージョン

Flash 8

シンタックス

`document.crop()`

パラメータ

なし

戻り値

ブール値。成功した場合は `true` を返します。それ以外の場合は `false` を返します。

説明

メソッド。選択した一番上の描画オブジェクトを使用して、選択した下位のすべての描画オブジェクトを切り抜きます。描画オブジェクトを選択していない場合、または選択したアイテムが描画オブジェクトでない場合は `false` を返します。

例

次の例では、現在選択しているオブジェクトを切り抜きます。

```
fl.getDocumentDOM().crop();
```

関連項目

[document.deleteEnvelope\(\)](#), [document.intersect\(\)](#), [document.punch\(\)](#),
[document.union\(\)](#), [shape.isDrawingObject](#)

document.currentPublishProfile

使用できるバージョン

Flash MX 2004

シンタックス

`document.currentPublishProfile`

説明

プロパティ。指定したドキュメントのアクティブなパブリッシュプロファイルの名前を指定するストリングです。

例

次の例では、デフォルト名の新規パブリッシュプロファイルを追加し、プロファイル名を [出力] パネルに表示します。

```
fl.getDocumentDOM().addNewPublishProfile();  
fl.outputPanel.trace(fl.getDocumentDOM().currentPublishProfile);
```

次の例では、選択したパブリッシュプロファイルを "Default" に変更します。

```
fl.getDocumentDOM().currentPublishProfile = "Default";
```

document.currentTimeline

使用できるバージョン

Flash MX 2004

シンタックス

`document.currentTimeline`

説明

プロパティ。アクティブタイムラインのインデックスを指定する整数です。アクティブタイムラインを設定するには、このプロパティの値を変更します。[document.editScene\(\)](#) の呼び出しとほとんど同じエフェクトになります。唯一の違いは、タイムラインのインデックスが有効でない場合でもエラーメッセージが表示されないことです (プロパティが設定されていないので、エラーの通知がありません)。

例

次の例では、現在のタイムラインのインデックスを表示します。

```
var myCurrentTL = fl.getDocumentDOM().currentTimeline;
fl.trace("The index of the current timeline is: "+ myCurrentTL);
```

次の例では、アクティブなタイムラインを、メインタイムラインから "myScene" という名前のシーンに変更します。

```
var i = 0;
var curTimelines = fl.getDocumentDOM().timelines;
while(i < fl.getDocumentDOM().timelines.length){
    if(curTimelines[i].name == "myScene"){
        fl.getDocumentDOM().currentTimeline = i;
    }
    ++i;
}
```

関連項目

[document.getTimeline\(\)](#)

document.deleteEnvelope()

使用できるバージョン

Flash 8

シンタックス

```
document.deleteEnvelope();
```

パラメータ

なし

戻り値

ブール値。成功した場合は true を返します。それ以外の場合は false を返します。

説明

メソッド。選択したオブジェクトからエンベロープ (オブジェクトを含む境界ボックス) を削除します。

例

次の例では、選択したオブジェクトからエンベロープを削除します。

```
fl.getDocumentDOM().deleteEnvelope();
```

関連項目

[document.crop\(\)](#), [document.intersect\(\)](#), [document.punch\(\)](#), [document.union\(\)](#),
[shape.isDrawingObject](#)

document.deletePublishProfile()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.deletePublishProfile()
```

パラメータ

なし

戻り値

現在の新規プロファイルのインデックスである整数。新規プロファイルがない場合は、現在のプロファイルは変更せずに、そのプロファイルのインデックスを返します。

説明

メソッド。現在アクティブなプロファイルが複数ある場合に削除します。最低でも1つのプロファイルは残しておく必要があります。

例

次の例では、現在アクティブなプロファイルが複数ある場合に削除し、新規の現在アクティブなプロファイルのインデックスを表示します。

```
alert(fl.getDocumentDOM().deletePublishProfile());
```

関連項目

[document.addNewPublishProfile\(\)](#)

document.deleteScene()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.deleteScene()
```

パラメータ

なし

戻り値

ブール値。シーンが正常に削除された場合は `true` を返し、削除されなかった場合は `false` を返します。

説明

メソッド。現在のシーン ([Timeline オブジェクト](#)) を削除し、削除したシーンが最後のシーンでない場合は、次のシーンを現在のタイムラインオブジェクトとして設定します。削除したシーンが最後のシーンであった場合は、最初のオブジェクトを現在のタイムラインオブジェクトとして設定します。タイムラインオブジェクト (シーン) が1つだけ存在する場合は、false を返します。

例

次の例では、現在のドキュメントにある 3 つのシーン (Scene0、Scene1、および Scene2) のうち、Scene2 を現在のシーンとして設定して削除します。

```
fl.getDocumentDOM().editScene(2);  
var success = fl.getDocumentDOM().deleteScene();
```

document.deleteSelection()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.deleteSelection()
```

パラメータ

なし

戻り値

なし

説明

メソッド。ステージで現在選択しているオブジェクトを削除します。オブジェクトを選択していない場合は、エラーメッセージを表示します。

例

次の例では、ドキュメントで現在選択しているオブジェクトを削除します。

```
fl.getDocumentDOM().deleteSelection();
```

document.description

使用できるバージョン

Flash MX 2004

シンタックス

`document.description`

説明

プロパティ。[アクセシビリティ] パネルの [説明] フィールドと等しいストリングです。この説明はスクリーンリーダーによって読み上げられます。

例

次の例では、ドキュメントの説明を設定します。

```
fl.getDocumentDOM().description= "This is the main movie";
```

次の例では、ドキュメントの説明を取得して、[出力] パネルに表示します。

```
fl.trace(fl.getDocumentDOM().description);
```

document.disableAllFilters()

使用できるバージョン

Flash 8

シンタックス

`document.disableAllFilters()`

パラメータ

なし

戻り値

なし

説明

メソッド。選択したオブジェクトのすべてのフィルタを無効にします。

例

次の例では、選択したオブジェクトのすべてのフィルタを無効にします。

```
fl.getDocumentDOM().disableAllFilters();
```

関連項目

[document.addFilter\(\)](#), [document.changeFilterOrder\(\)](#), [document.disableFilter\(\)](#),
[document.disableOtherFilters\(\)](#), [document.enableAllFilters\(\)](#),
[document.getFilters\(\)](#), [document.removeAllFilters\(\)](#), [Filter](#) オブジェクト

document.disableFilter()

使用できるバージョン

Flash 8

シンタックス

```
document.disableFilter(filterIndex)
```

パラメータ

filterIndex フィルタリストのフィルタのゼロから始まるインデックスを表す整数。

戻り値

なし

説明

メソッド。フィルタリストで、指定したフィルタを無効にします。

例

次の例では、選択したオブジェクトのフィルタリストで、1番目と3番目 (インデックス値 0 と 2) のフィルタを無効にします。

```
fl.getDocumentDOM().disableFilter(0);  
fl.getDocumentDOM().disableFilter(2);
```

関連項目

[document.addFilter\(\)](#), [document.changeFilterOrder\(\)](#),
[document.disableAllFilters\(\)](#), [document.disableOtherFilters\(\)](#),
[document.enableFilter\(\)](#), [document.getFilters\(\)](#), [document.removeFilter\(\)](#),
[Filter](#) オブジェクト

document.disableOtherFilters()

使用できるバージョン

Flash 8

シンタックス

```
document.disableOtherFilters(enabledFilterIndex)
```

パラメータ

enabledFilterIndex その他のフィルタを無効にした後、有効のままにしておくフィルタのゼロから始まるインデックスを表す整数。

戻り値

なし

説明

メソッド。フィルタリストで、指定した位置にあるフィルタ以外のすべてのフィルタを無効にします。

例

次の例では、リストの 2 番目 (インデックス値 1) のフィルタ以外のすべてのフィルタを無効にします。

```
fl.getDocumentDom().disableOtherFilters(1);
```

関連項目

[document.addFilter\(\)](#), [document.changeFilterOrder\(\)](#),
[document.disableAllFilters\(\)](#), [document.disableFilter\(\)](#),
[document.enableFilter\(\)](#), [document.getFilters\(\)](#), [document.removeFilter\(\)](#),
[Filter](#) オブジェクト

document.distribute()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.distribute(distributemode [, bUseDocumentBounds])
```

パラメータ

distributemode 選択したオブジェクトを配分する位置を指定するストリング。指定できる値は、
"left edge"、"horizontal center"、"right edge"、"top edge"、"vertical center"、
および "bottom edge" です。

bUseDocumentBounds ブール値。true を設定すると、選択したオブジェクトを、そのドキュメントの境界を使用して配分します。それ以外の場合は、選択したオブジェクトの境界を使用します。デフォルト値は false です。

戻り値

なし

説明

メソッド。選択したオブジェクトを配分します。

例

次の例では、選択したオブジェクトを上揃えで配分します。

```
fl.getDocumentDOM().distribute("top edge");
```

次の例では、選択したオブジェクトを上揃えで配分し、*bUseDocumentBounds* パラメータを明示的に設定します。

```
fl.getDocumentDOM().distribute("top edge", false);
```

次の例では、選択したオブジェクトを、そのドキュメントの境界を使用して上揃えで配分します。

```
fl.getDocumentDOM().distribute("top edge", true);
```

関連項目

[document.getAlignToDocument\(\)](#), [document.setAlignToDocument\(\)](#)

document.distributeToLayers()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.distributeToLayers()
```

パラメータ

なし

戻り値

なし。

説明

メソッド。現在選択しているオブジェクトでレイヤーに配分処理を実行します。[レイヤーに配分] を選択する操作と同じです。オブジェクトを選択していない場合は、エラーメッセージを表示します。

例

次の例では、現在選択しているオブジェクトをレイヤーに配分します。

```
fl.getDocumentDOM().distributeToLayers();
```

document.docClass

使用できるバージョン

Flash CS3 Professional

シンタックス

```
document.docClass
```

説明

プロパティ。ドキュメントに関連付けられたトップレベルの **ActionScript 3.0** クラスを示す文字列です。**ActionScript 3.0** を使用するようにドキュメントが設定されていない場合、このプロパティは無視されます。

例

次の例では、ドキュメントに関連付けられた **ActionScript 3.0** クラスは、`com/mycompany/ManagerClass.as` に定義された `com.mycompany.ManagerClass` であることを指定します。

```
var myDocument = fl.getDocumentDOM();  
// プロパティの設定  
myDocument.docClass = "com.mycompany.ManagerClass";  
// プロパティの取得  
fl.outputPanel.trace("document.docClass has been set to " +  
    myDocument.docClass);
```

関連項目

[item.linkageBaseClass](#)

document.documentHasData()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.documentHasData(name)
```

パラメータ

name 確認するデータの名前を指定する文字列。

戻り値

ブール値。ドキュメントに永続データがある場合は `true` を返し、ない場合は `false` を返します。

説明

メソッド。指定した名前の永続データがドキュメントにあるかどうかを確認します。

例

次の例では、ドキュメントに "myData" という名前の永続データがあるかどうかを確認します。

```
var hasData = fl.getDocumentDOM().documentHasData("myData");
```

関連項目

[document.addDataToDocument\(\)](#), [document.getDataFromDocument\(\)](#),
[document.removeDataFromDocument\(\)](#)

document.duplicatePublishProfile()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.duplicatePublishProfile([profileName])
```

パラメータ

profileName 複製したプロファイルの一意の名前を指定するストリング。名前を指定しない場合は、デフォルト名が使用されます。このパラメータはオプションです。

戻り値

プロファイルリストの新規プロファイルのインデックスである整数。プロファイルを複製できない場合は -1 を返します。

説明

メソッド。現在アクティブなプロファイルを複製し、複製したプロファイルにフォーカスを移動します。

例

次の例では、現在アクティブなプロファイルを複製し、新規プロファイルのインデックスを [出力] パネルに表示します。

```
fl.trace(fl.getDocumentDOM().duplicatePublishProfile("dup profile"));
```

document.duplicateScene()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.duplicateScene()
```

パラメータ

なし

戻り値

ブール値。シーンが正常に複製された場合は true を返し、複製されなかった場合は false を返します。

説明

メソッド。現在選択しているシーンをコピーし、新規のシーンに固有の名前を付け、現在のシーンとして設定します。

例

次の例では、ドキュメントの 2 番目のシーンを複製します。

```
fl.getDocumentDOM().editScene(1); // 真ん中のシーンを現在のシーンとして設定する  
var success = fl.getDocumentDOM().duplicateScene();
```

document.duplicateSelection()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.duplicateSelection()
```

パラメータ

なし

戻り値

なし

説明

メソッド。ステージで選択したオブジェクトを複製します。

例

次の例では、現在選択しているオブジェクトを複製します。Alt キーを押しながらアイテムをクリックしてドラッグする操作と同じです。

```
fl.getDocumentDOM().duplicateSelection();
```

document.editScene()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.editScene(index)
```

パラメータ

index 編集するシーンを指定するゼロから始まる整数

戻り値

なし

説明

メソッド。指定したシーンを編集するために、現在選択しているシーンとして設定します。

例

次の例では、現在のドキュメントにある 3 つのシーン (Scene0、Scene1、および Scene2) のうち、Scene2 を現在のシーンとして設定して削除します。

```
fl.getDocumentDOM().editScene(2);  
fl.getDocumentDOM().deleteScene();
```

document.enableAllFilters()

使用できるバージョン

Flash 8

シンタックス

```
document.enableAllFilters()
```

パラメータ

なし

戻り値

なし

説明

メソッド。選択したオブジェクトのフィルタリストですべてのフィルタを有効にします。

例

次の例では、選択したオブジェクトのフィルタリストですべてのフィルタを有効にします。

```
fl.getDocumentDOM().enableAllFilters()
```

関連項目

[document.addFilter\(\)](#), [document.changeFilterOrder\(\)](#),
[document.disableAllFilters\(\)](#), [document.enableFilter\(\)](#), [document.getFilters\(\)](#),
[document.removeAllFilters\(\)](#), [Filter](#) オブジェクト

document.enableFilter()

使用できるバージョン

Flash 8

シンタックス

```
document.enableFilter(filterIndex)
```

パラメータ

filterIndex フィルタリストで有効にするフィルタのゼロから始まるインデックスを表す整数。

戻り値

なし

説明

メソッド。選択したオブジェクトの指定したフィルタを有効にします。

例

次の例では、選択したオブジェクトの 2 番目のフィルタを有効にします。

```
fl.getDocumentDOM().enableFilter(1);
```

関連項目

[document.addFilter\(\)](#), [document.changeFilterOrder\(\)](#), [document.disableFilter\(\)](#),
[document.enableAllFilters\(\)](#), [document.getFilters\(\)](#), [document.removeFilter\(\)](#),
[Filter](#) オブジェクト

document.enterEditMode()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.enterEditMode([editMode])
```

パラメータ

editMode 編集モードを指定するストリング。指定できる値は、"inPlace" または "newWindow" です。パラメータを指定しない場合は、シンボル編集モードがデフォルトになります。このパラメータはオプションです。

戻り値

なし

説明

メソッド。オーサリングツールを、パラメータで指定した編集モードに切り替えます。パラメータを指定しない場合は、シンボル編集モードをデフォルトとします。シンボルを右クリックしてコンテキストメニューを表示し、[編集] を選択する操作と同じです。

例

次の例では、現在選択しているシンボルを同じ位置で編集モードに設定します。

```
fl.getDocumentDOM().enterEditMode('inPlace');
```

次の例では、現在選択しているシンボルを新規ウィンドウで編集モードに設定します。

```
fl.getDocumentDOM().enterEditMode('newWindow');
```

関連項目

[document.exitEditMode\(\)](#)

document.exitEditMode()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.exitEditMode()
```

パラメータ

なし

戻り値

なし

説明

メソッド。シンボル編集モードを終了し、編集モードから次の上位レベルにフォーカスを戻します。たとえば、別のシンボル内のシンボルを編集する場合、編集中のシンボルの上位である親シンボルに移ります。

例

次の例では、シンボル編集モードを終了します。

```
fl.getDocumentDOM().exitEditMode();
```

関連項目

[document.enterEditMode\(\)](#)

document.exportPNG()

使用できるバージョン

Flash 8

シンタックス

```
document.exportPNG([fileURI [, bCurrentPNGSettings [, bCurrentFrame]])
```

パラメータ

fileURI 書き出したファイルの名前を指定する文字列。file:/// URI で表します。*fileURI* が空の文字列の場合または指定されていない場合は、[ムービーの書き出し]ダイアログボックスを表示します。

bCurrentPNGSettings 現在の PNG のパブリッシュ設定を使用するか (true)、[PNG 書き出し]ダイアログボックスを表示するか (false) を指定するブール値。このパラメータはオプションです。デフォルト値は false です。

bCurrentFrame 現在のフレームのみを書き出すか (true)、すべてのフレームをフレームごとに別の PNG ファイルとして書き出すか (false) を指定するブール値。このパラメータはオプションです。デフォルト値は false です。

戻り値

ブール値。ファイルを PNG ファイルとして正常に書き出した場合は true を返します。それ以外の場合は false を返します。

説明

メソッド。ドキュメントを PNG ファイルとして書き出します。*fileURI* を指定し、そのファイルが既に存在する場合は、警告なしで上書きされます。

例

次の例では、現在の PNG のパブリッシュ設定を使用して、現在のドキュメントの現在のフレームを myFile.png に書き出します。

```
fl.getDocumentDOM().exportPNG("file:///C:/myProject/myFile.png", true, true);
```

document.exportPublishProfile()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.exportPublishProfile(fileURI)
```

パラメータ

fileURI プロファイルを書き出す XML ファイルのパスを指定するストリング。file:/// URI で表します。

戻り値

なし

説明

メソッド。現在アクティブなプロファイルを XML ファイルに書き出します。

例

次の例では、現在アクティブなプロファイルを、C ドライブの "/Documents and Settings/<ユーザー名>/Desktop" フォルダにある "profile.xml" に書き出します。

```
fl.getDocumentDOM().exportPublishProfile('file:///C:/Documents and Settings/  
  <ユーザー名>/Desktop/profile.xml');
```

document.exportSWF()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.exportSWF([fileURI [, bCurrentSettings]])
```

パラメータ

fileURI 書き出したファイルの名前を指定するストリング。file:/// URI で表します。*fileURI* が空の場合または指定されていない場合は、[ムービーの書き出し] ダイアログボックスを表示します。このパラメータはオプションです。

bCurrentSettings ブール値。true を設定すると、現在の SWF パブリッシュ設定を使用します。設定しない場合は、[Flash Player 書き出し] ダイアログボックスを表示します。デフォルト値は false です。このパラメータはオプションです。

戻り値

なし

説明

メソッド。ドキュメントを Flash SWF 形式で書き出します。

例

次の例では、指定したファイル位置に、現在のパブリッシュ設定でドキュメントを書き出します。

```
fl.getDocumentDOM().exportSWF("file:///C:/Documents and Settings/joe_user/Desktop/qwerty.swf");
```

次の例では、[ムービーの書き出し] ダイアログボックスと [Flash Player 書き出し] ダイアログボックスを表示し、指定した設定に基づいてドキュメントを書き出します。

```
fl.getDocumentDOM().exportSWF("", true);
```

次の例では、[ムービーの書き出し] ダイアログボックスを表示し、指定した設定に基づいてドキュメントを書き出します。

```
fl.getDocumentDOM().exportSWF();
```

document.forceSimple

使用できるバージョン

Flash MX 2004

シンタックス

`document.forceSimple`

説明

プロパティ。指定したオブジェクトの子がアクセス可能かどうかを指定するブール値です。[アクセシビリティ] パネルの [子オブジェクトをアクセス可能にする] の逆ロジックに等しくなります。forceSimple が true の場合、[子オブジェクトをアクセス可能にする] をオフにした場合と同じになります。forceSimple が false の場合、[子オブジェクトをアクセス可能にする] をオンにした場合と同じになります。

例

次の例では、areChildrenAccessible 変数を forceSimple プロパティの値に設定します。値 false は、子がアクセス可能であることを示します。

```
var areChildrenAccessible = fl.getDocumentDOM().forceSimple;
```

次の例では、forceSimple プロパティを設定して、ドキュメントの子をアクセス可能にします。

```
fl.getDocumentDOM().forceSimple = false;
```

document.frameRate

使用できるバージョン

Flash MX 2004

シンタックス

`document.frameRate`

説明

プロパティ。SWF ファイルの再生時に1秒間に表示されるフレーム数を指定する浮動小数値です。デフォルトは 12 です。このプロパティの設定は、[ドキュメントプロパティ] ダイアログボックス ([修正] [ドキュメント]) でデフォルトのフレームレートを設定するのと同じです。

例

次の例では、1 秒間に 25.5 フレームのフレームレートを設定します。

```
fl.getDocumentDOM().frameRate = 25.5;
```

document.getAlignToDocument()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.getAlignToDocument()
```

パラメータ

なし

戻り値

ブール値。環境設定でオブジェクトをステージに整列するように設定している場合は true を返し、それ以外の場合は false を返します。

説明

メソッド。[整列] パネルにある [ステージを基準] ボタンの値を取得する操作と同じです。ドキュメントの [document.align\(\)](#) メソッド、[document.distribute\(\)](#) メソッド、[document.match\(\)](#) メソッド、および [document.space\(\)](#) メソッドで使用できる環境設定を取得します。

例

次の例では、[整列] パネルにある [ステージを基準] ボタンの値を取得します。戻り値が true である場合、[ステージを基準] ボタンはアクティブです。それ以外の場合はアクティブではありません。

```
var isAlignToDoc = fl.getDocumentDOM().getAlignToDocument();
fl.getDocumentDOM().align("left", isAlignToDoc);
```

関連項目

[document.setAlignToDocument\(\)](#)

document.getBlendMode()

使用できるバージョン

Flash 8

シンタックス

```
document.getBlendMode()
```

パラメータ

なし

戻り値

選択したオブジェクトのブレンドモードを指定するストリング。複数のオブジェクトを選択していて、設定されているブレンドモードが異なる場合、ストリングは最上位深度のオブジェクトのブレンドモードを示します。



ブレンドモードをサポートしていないオブジェクト、またはブレンドモード値が "normal" のオブジェクトが、選択したオブジェクトに含まれている場合、戻り値は予測できません。

説明

メソッド。選択したオブジェクトのブレンドモードを指定するストリングを返します。

例

次の例では、ブレンドモードの名前を [出力] パネルに表示します。

```
fl.trace(fl.getDocumentDom().getBlendMode());
```

document.getCustomFill()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.getCustomFill([ objectToFill ])
```

パラメータ

objectToFill 塗りオブジェクトの位置を指定するストリング。使用できる値は次のとおりです。

- "toolbar"。ツールパネルとプロパティインスペクタの塗りオブジェクトを返します。
- "selection"。選択範囲の塗りオブジェクトを返します。

このパラメータを省略すると、"selection" がデフォルトになります。何も選択していない場合は、`undefined` を返します。このパラメータは省略可能です。

戻り値

正常に終了した場合は、*objectToFill* パラメータで指定される **Fill オブジェクト** を返します。それ以外の場合は `undefined` を返します。

説明

メソッド。選択したシェイプの塗りオブジェクト、またはツールパネルとプロパティインスペクタの塗りオブジェクト (指定した場合) を取得します。

例

次の例では、選択範囲の塗りオブジェクトを取得し、その選択範囲のカラーを白色に変更します。

```
var fill = fl.getDocumentDOM().getCustomFill();
fill.color = '#FFFFFF';
fill.style = "solid";
fl.getDocumentDOM().setCustomFill(fill);
```

次の例では、ツールパネルとプロパティインスペクタの塗りオブジェクトを返し、色見本を線状グラデーションに変更します。

```
var fill = fl.getDocumentDOM().getCustomFill("toolbar");
fill.style = "linearGradient";
fill.colorArray = [ 0x00ff00, 0xff0000, 0x0000ff ];
fill.posArray = [0, 100, 200];
fl.getDocumentDOM().setCustomFill( fill );
```

関連項目

[document.setCustomFill\(\)](#)

document.getCustomStroke()

使用できるバージョン

Flash MX 2004

シンタックス

`document.getCustomStroke([locationOfStroke])`

パラメータ

locationOfStroke 線オブジェクトの位置を指定するストリング。使用できる値は次のとおりです。

- "toolbar"。ツールパネルとプロパティインスペクタの線オブジェクトを返します。
- "selection"。選択範囲の線オブジェクトを返します。

このパラメータを省略すると、"selection" がデフォルトになります。何も選択していない場合は、`undefined` を返します。このパラメータは省略可能です。

戻り値

正常に終了した場合は、*locationOfStroke* パラメータで指定される [Stroke オブジェクト](#) を返します。それ以外の場合は `undefined` を返します。

説明

選択したシェイプの線オブジェクト、またはツールパネルとプロパティインスペクタの線オブジェクト (指定した場合) を返します。

例

次の例では、選択範囲の現在の線設定を返し、線の太さを 2 に変更します。

```
var stroke = fl.getDocumentDOM().getCustomStroke("selection");
stroke.thickness = 2;
fl.getDocumentDOM().setCustomStroke(stroke);
```

次の例では、ツールパネルとプロパティインスペクタの現在の線設定を返し、線のカラーを赤色に設定します。

```
var stroke = fl.getDocumentDOM().getCustomStroke("toolbar");
stroke.color = "#FF0000";
fl.getDocumentDOM().setCustomStroke(stroke);
```

関連項目

[document.setCustomStroke\(\)](#)

document.getDataFromDocument()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.getDataFromDocument(name)
```

パラメータ

name 返すデータの名前を指定するストリング。

戻り値

指定したデータ

説明

メソッド。指定されたデータの値を取得します。返されるデータの型は保存されたデータ型によって異なります。

例

次の例では、整数値 12 を現在のドキュメントに追加し、その値を [出力] パネルに表示します。

```
fl.getDocumentDOM().addDataToDocument("myData", "integer", 12);
fl.trace(fl.getDocumentDOM().getDataFromDocument("myData"));
```

関連項目

[document.addDataToDocument\(\)](#), [document.documentHasData\(\)](#),
[document.removeDataFromDocument\(\)](#)

document.getElementProperty()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.getElementProperty(propertyName)
```

パラメータ

propertyName 値を取得する Element プロパティの名前を指定するストリング。

戻り値

指定したプロパティの値。複数のエレメントを選択していて、各プロパティ値が異なる場合など、プロパティが未確定の場合は `null` を返します。選択したエレメントに適したプロパティでない場合は `undefined` を返します。

説明

メソッド。現在選択しているオブジェクトの指定した Element プロパティを取得します。指定できる値の一覧については、[211 ページの「Element オブジェクトのプロパティ一覧」](#)を参照してください。

例

次の例では、現在選択しているオブジェクトの Element プロパティの `name` を取得します。

```
// elementName = 選択したオブジェクトのインスタンス名。  
var elementName = fl.getDocumentDOM().getElementProperty("name");
```

関連項目

[document.setElementProperty\(\)](#)

document.getElementTextAttr()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.getElementTextAttr(attrName [, startIndex [, endIndex]])
```

パラメータ

attrName 返される TextAttrs プロパティの名前を指定するストリング。プロパティ名と使用できる値については、[509 ページの「TextAttrs オブジェクトのプロパティ一覧」](#)を参照してください。

startIndex 最初の文字のインデックスを指定するゼロから始まる整数。このパラメータはオプションです。

endIndex 最後の文字のインデックスを指定する整数。このパラメータはオプションです。

戻り値

選択しているテキストフィールドが1つである場合、テキストで使用している値が1つだけの場合にプロパティを返します。テキストフィールド内で複数の値を使用している場合は `undefined` を返します。複数のテキストフィールドを選択しているとき、すべてのテキストの整列値が同じである場合は、この値を返します。複数のテキストフィールドを選択していて、テキストの整列値が同じでない場合は、`undefined` を返します。オプションのパラメータを渡していない場合、このルールは、現在選択しているテキスト範囲に適用され、テキストが現在編集中でない場合は、テキストフィールド全体に適用されます。*startIndex* だけを渡している場合、選択しているテキストオブジェクトすべてがこの値と一致した場合に、インデックスの右側の文字のプロパティを返します。*startIndex* と *endIndex* を渡している場合、戻り値は、*startIndex* から *endIndex* の前の文字までの範囲を表します。

説明

メソッド。選択したテキストオブジェクトの指定した `TextAttrs` プロパティを取得します。選択したオブジェクトのうち、テキストフィールドでないものは無視されます。プロパティ名と使用できる値については、[509 ページの「TextAttrs オブジェクトのプロパティ一覧」](#)を参照してください。

`document.setElementTextAttr()` も参照してください。

例

次の例では、選択したテキストフィールドのサイズを取得します。

```
fl.getDocumentDOM().getElementTextAttr("size");
```

次の例では、選択したテキストフィールドのインデックス 3 の文字のカラーを取得します。

```
fl.getDocumentDOM().getElementTextAttr("fillColor", 3);
```

次の例では、選択したテキストフィールドのインデックス 2 から 9 までのテキストのフォント名を取得します。

```
fl.getDocumentDOM().getElementTextAttr("face", 2, 10);
```

document.getFilters()

使用できるバージョン

Flash 8

シンタックス

```
document.getFilters()
```

パラメータ

なし

戻り値

現在選択しているオブジェクトに適用されるフィルタリストを含む配列。

説明

メソッド。現在選択しているオブジェクトに適用されるフィルタリストを含む配列を返します。複数のオブジェクトを選択していて、適用されているフィルタが同じでない場合、このメソッドは、最初に選択したオブジェクトに適用されているフィルタリストを返します。

例

詳細については、`document.setFilters()` を参照してください。

関連項目

`document.addFilter()`, `document.changeFilterOrder()`, `document.setFilters()`,
`Filter` オブジェクト

document.getMetadata()

使用できるバージョン

Flash 8

シンタックス

```
document.getMetadata()
```

パラメータ

なし

戻り値

ドキュメントに関連付けられている XML メタデータを含むストリング。メタデータがない場合は空のストリング。

説明

メソッド。ドキュメントに関連付けられている XML メタデータを含むストリングです。メタデータがない場合は空のストリングです。

例

次の例では、現在開いているドキュメントの XML メタデータを [出力] パネルに表示します。

```
fl.trace("XML Metadata is :" + fl.getDocumentDOM().getMetadata());
```

関連項目

`document.setMetadata()`

document.getMobileSettings()

使用できるバージョン

Flash CS3 Professional

シンタックス

```
document.getMobileSettings()
```

パラメータ

なし

戻り値

ドキュメントの XML 設定を表すストリング。値が設定されていない場合は、空のストリングを返します。

説明

メソッド。ドキュメントのモバイル XML 設定を返します。

例

次の例では、現在のドキュメントの XML 設定を表すストリングが表示されます。

```
fl.trace(fl.getDocumentDOM().getMobileSettings());  
// 次のようなストリングをトレースする  
"<? xml version="1.0" encoding="UTF-16" standalone="no" ?><mobileSettings>  
  <contentType id="standalonePlayer" name="Standalone Player"/> <testDevices>  
    <testDevice id="1170" name="Generic Phone" selected="yes"/> </testDevices>  
    <outputMsgFiltering info="no" trace="yes" warning="yes"/> <testWindowState  
      height="496" splitterClosed="No" splitterXPos="400" width="907"/> </  
mobileSettings>"
```

関連項目

[document.setMobileSettings\(\)](#)

document.getPlayerVersion()

使用できるバージョン

Flash CS3 Professional

シンタックス

```
document.getPlayerVersion()
```

パラメータ

なし

戻り値

`document.setPlayerVersion()` で指定した Flash Player バージョンを表す文字列。値が設定されていない場合は、[パブリッシュ設定] ダイアログボックスで指定した値を返します。

説明

メソッド。指定したドキュメントの対象 Player バージョンを表す文字列を返します。

指定したファイルの対象となる ActionScript バージョンを調べるには、`document.asVersion` を使用します。

例

次の例では、指定した Player バージョンを現在のドキュメントの対象として指定し、その値を取得します。

```
fl.getDocumentDOM().setPlayerVersion("6");
var version = fl.getDocumentDOM().getPlayerVersion();
fl.trace(version) // "6" を表示する
fl.getDocumentDOM().setPlayerVersion("FlashLite20");
var version = fl.getDocumentDOM().getPlayerVersion();
fl.trace(version) // "'FlashLite20'" を表示する
```

関連項目

[document.setPlayerVersion\(\)](#)

document.getSelectionRect()

使用できるバージョン

Flash MX 2004

シンタックス

`document.getSelectionRect()`

パラメータ

なし

戻り値

現在選択しているオブジェクトの境界の矩形を返し、何も選択していない場合は 0 を返します。戻り値の形式の詳細については、[document.addNewRectangle\(\)](#) を参照してください。

説明

メソッド。現在選択しているオブジェクトの境界の矩形を取得します。選択範囲が矩形でない場合は、選択範囲全体を囲む最小の矩形を返します。矩形はドキュメントスペースを基にしており、編集モードの場合は、編集集中のシンボルの基準点 (あるいは原点またはゼロ点) を基にしています。

例

次の例では、現在選択しているオブジェクトの境界の矩形を取得し、そのプロパティを表示します。

```
var newRect = fl.getDocumentDOM().getSelectionRect();
var outputStr = "left: " + newRect.left + " top: " + newRect.top + " right: " +
    newRect.right + " bottom: " + newRect.bottom;
alert(outputStr);
```

関連項目

[document.selection](#), [document.setSelectionRect\(\)](#)

document.getTextString()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.getTextString([startIndex [, endIndex]])
```

パラメータ

startIndex 取得する最初の文字のインデックスである整数。このパラメータはオプションです。

endIndex 取得する最後の文字のインデックスである整数。このパラメータはオプションです。

戻り値

選択したテキストを含むストリング

説明

メソッド。現在選択しているテキストを取得します。オプションのパラメータを渡していない場合は、現在選択しているテキストが使用されます。編集するテキストを現在開いていない場合は、テキストストリング全体を返します。*startIndex* だけを渡している場合は、その開始インデックスの位置からフィールドの最後までまでのストリングを返します。*startIndex* と *endIndex* を渡している場合は、*startIndex* から *endIndex* の前までのストリングを返します。

複数のテキストフィールドを選択している場合は、すべてのストリングを連結して返します。

例

次の例では、選択したテキストフィールドのストリングを取得します。

```
fl.getDocumentDOM().getTextString();
```

次の例では、選択したテキストフィールドの文字インデックス 5 のストリングを取得します。

```
fl.getDocumentDOM().getTextString(5);
```

次の例では、文字インデックス 2 から 9 までのストリングを取得します。

```
fl.getDocumentDOM().getTextString(2, 10);
```

関連項目

[document.setTextString\(\)](#)

document.getTimeline()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.getTimeline()
```

パラメータ

なし

戻り値

現在の Timeline オブジェクト

説明

メソッド。ドキュメントにある現在の [Timeline オブジェクト](#) を取得します。現在のタイムラインは、現在のシーン、編集中の現在のシンボル、または現在のスクリーンになります。

例

次の例では、Timeline オブジェクトを取得し、最長レイヤーのフレーム数を返します。

```
var longestLayer = fl.getDocumentDOM().getTimeline().frameCount;  
fl.trace("The longest layer has" + longestLayer + "frames");
```

次の例では、ステージで選択したシンボルの同じ位置で編集モードに移行し、シンボルのタイムラインにフレームを挿入します。

```
fl.getDocumentDOM().enterEditMode("inPlace");  
fl.getDocumentDOM().getTimeline().insertFrames();
```

次の例では、Timeline オブジェクトを取得し、その名前を表示します。

```
var timeline = fl.getDocumentDOM().getTimeline();  
alert(timeline.name);
```

関連項目

[document.currentTimeline](#), [document.timelines](#), [symbolItem.timeline](#)

document.getTransformationPoint()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.getTransformationPoint()
```

パラメータ

なし

戻り値

{x:10, y:20} などの点 (x と y は浮動小数点数)。選択したエレメントの座標系の中で、変形点 (あるいは原点またはゼロ点) の位置を指定します。

説明

メソッド。現在選択しているオブジェクトの変形点の位置を取得します。変形点は、回転や傾斜などの連結で使用できます。



変形点の位置は、選択したアイテムのタイプに応じて異なります。詳細については、[document.setTransformationPoint\(\)](#) を参照してください。

例

次の例では、現在選択しているオブジェクトの変形点を取得します。transPoint.x プロパティは、変形点の x 座標を表します。transPoint.y プロパティは、変形点の y 座標を表します。

```
var transPoint = fl.getDocumentDOM().getTransformationPoint();
```

関連項目

[document.setTransformationPoint\(\)](#), [element.getTransformationPoint\(\)](#)

document.group()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.group()
```

パラメータ

なし

戻り値

なし

説明

メソッド。現在選択しているオブジェクトをグループに変換します。

例

次の例では、現在選択しているオブジェクトをグループに変換します。

```
fl.getDocumentDOM().group();
```

関連項目

[document.unGroup\(\)](#)

document.height

使用できるバージョン

Flash MX 2004

シンタックス

```
document.height
```

説明

プロパティ。ドキュメント (ステージ) の高さをピクセル単位で指定する整数です。

例

次の例では、ステージの高さを 400 ピクセルに設定します。

```
fl.getDocumentDOM().height = 400;
```

関連項目

[document.width](#)

document.id

使用できるバージョン

Flash CS3 Professional

シンタックス

```
document.id
```

説明

読み取り専用のプロパティ。Flash セッションの実行中に各ドキュメントを識別するために自動的に割り当てられる一意の整数です。特定のドキュメントのアクションを指定するには、このプロパティを `fl.findDocumentDOM()` と組み合わせて使用します。

例

次の例では、現在のドキュメントのドキュメント ID を表示します。

```
fl.trace("Current doc's internal ID is: " + fl.getDocumentDOM().id);
```

関連項目

[fl.findDocumentDOM\(\)](#)

document.importFile()

使用できるバージョン

Flash 8

シンタックス

```
document.importFile(fileURI [, importToLibrary])
```

パラメータ

fileURI 読み込むファイルのパスを指定するストリング。file:/// URI で表します。

importToLibrary ファイルのみをドキュメントのライブラリに読み込むか (true)、コピーもステージ上に配置するか (false) を指定するブール値。デフォルト値は false です。

戻り値

ファイルの読み込みが正常に終了したかどうかを示すブール値。

説明

メソッド。ファイルをドキュメントに読み込みます。このメソッドは、[ライブラリに読み込み] または [ステージに読み込み] メニューコマンドと同じ操作を実行します。パブリッシュプロファイルを読み込むには、[document.importPublishProfile\(\)](#) を使用します。

例

次の例では、ユーザーがステージ上に読み込むファイルを参照できるようにします。

```
var dom = fl.getDocumentDOM();  
var URI = fl.browseForFileURL("select", "Import File");  
dom.importFile(URI);
```

関連項目

[document.importSWF\(\)](#), [fl.browseForFileURL\(\)](#)

document.importPublishProfile()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.importPublishProfile( fileURI )
```

パラメータ

fileURI 読み込むプロファイルを定義する XML ファイルのパスを指定するストリング。
file:/// URI で表します。

戻り値

プロファイルリストの読み込んだプロファイルのインデックスである整数です。プロファイルを読み込めない場合は -1 を返します。

説明

メソッド。ファイルからプロファイルを読み込みます。

例

次の例では、"profile.xml" ファイルに含まれるプロファイルを読み込み、そのインデックスをプロファイルリストに表示します。

```
alert(fl.getDocumentDOM().importPublishProfile('file:///C:/Documents and  
Settings/janeUser/Desktop/profile.xml'));
```

document.importSWF()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.importSWF( fileURI )
```

パラメータ

fileURI 読み込む SWF ファイルを指定するストリング。file:/// URI で表します。

戻り値

なし

説明

メソッド。SWF ファイルをドキュメントに読み込みます。このメソッドは、[読み込み]メニューオプションを使用して SWF ファイルを指定するのと同じ操作を実行します。Flash 8 以降では、`document.importFile()` を使用して、SWF ファイル (他のファイルタイプも同様) を読み込むこともできます。

例

次の例では、"Flash Configuration" フォルダから "mySwf.swf" ファイルを読み込みます。

```
fl.getDocumentDOM().importSWF(fl.configURI+"mySwf.swf");
```

関連項目

[document.importFile\(\)](#)

document.intersect()

使用できるバージョン

Flash 8

シンタックス

```
document.intersect();
```

パラメータ

なし

戻り値

ブール値。成功した場合は true を返します。それ以外の場合は false を返します。

説明

メソッド。選択したすべての描画オブジェクトから交差描画オブジェクトを作成します。描画オブジェクトを選択していない場合、または選択したアイテムが描画オブジェクトでない場合は false を返します。

例

次の例では、選択したすべての描画オブジェクトから交差描画オブジェクトを作成します。

```
fl.getDocumentDOM().intersect();
```

関連項目

[document.crop\(\)](#), [document.deleteEnvelope\(\)](#), [document.punch\(\)](#), [document.union\(\)](#), [shape.isDrawingObject](#)

document.library

使用できるバージョン

Flash MX 2004

シンタックス

`document.library`

説明

読み取り専用プロパティ。ドキュメントの [library オブジェクト](#) です。

例

次の例では、現在フォーカスのあるドキュメントのライブラリを取得します。

次の例では、`var myCurrentLib = fl.getDocumentDOM().library;`

現在フォーカスのあるドキュメントが `fl.documents[1]` でない場合、フォーカスのないライブラリのライブラリを取得するか、[ファイル]-[開く] で外部ライブラリとして開いたライブラリのライブラリを取得します。

```
var externalLib = fl.documents[1].library;
```

document.livePreview

使用できるバージョン

Flash MX 2004

シンタックス

`document.livePreview`

説明

プロパティ。[ライブプレビュー] を有効にするかどうかを指定するブール値です。true を設定すると、コンポーネントが、パブリッシュされた Flash コンテンツに表示される場合と同じように、おおよそのサイズとともにステージに表示されます。false を設定すると、コンポーネントはアウトラインとしてだけ表示されます。デフォルト値は true です。

例

次の例では、[ライブプレビュー] に false を設定します。

```
fl.getDocumentDOM().livePreview = false;
```


document.match()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.match(bWidth, bHeight [, bUseDocumentBounds])
```

パラメータ

bWidth ブール値。true を設定すると、選択したアイテムの幅を同じにします。

bHeight ブール値。true を設定すると、選択したアイテムの高さを同じにします。

bUseDocumentBounds ブール値。true を設定すると、オブジェクトのサイズとドキュメントの境界を一致させます。それ以外の場合は、最大のオブジェクトの境界を使用します。デフォルト値は false です。このパラメータはオプションです。

戻り値

なし

説明

メソッド。選択したオブジェクトのサイズを同じにします。

例

次の例では、選択したオブジェクトの幅だけを一致させます。

```
fl.getDocumentDOM().match(true,false);
```

次の例では、高さだけを一致させます。

```
fl.getDocumentDOM().match(false,true);
```

次の例では、幅だけをドキュメントの境界と一致させます。

```
fl.getDocumentDOM().match(true,false,true);
```

関連項目

[document.getAlignToDocument\(\)](#), [document.setAlignToDocument\(\)](#)

document.mouseClick()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.mouseClick(position, bToggleSel, bShiftSel)
```

パラメータ

position クリックをピクセル単位で示す *x* 座標と *y* 座標を指定する浮動小数値のペア。

bToggleSel Shift キーの状態を指定するブール値。キーが押されている場合は true、押されていない場合は false になります。

bShiftSel アプリケーションの環境設定の Shift 選択状態を指定するブール値。オンの場合は true、オフの場合は false になります。

戻り値

なし。

説明

メソッド。矢印ツールからマウスのクリックを実行します。

例

次の例では、指定した位置でマウスのクリックを実行します。

```
fl.getDocumentDOM().mouseClick({x:300, y:200}, false, false);
```

関連項目

[document.mouseDbClick\(\)](#)

document.mouseDbClick()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.mouseDbClick(position, bAltDown, bShiftDown, bShiftSelect)
```

パラメータ

position クリックをピクセル単位で示す *x* 座標と *y* 座標を指定する浮動小数値のペア。

bAltdown イベント発生時の Alt キーの状態を記録するブール値。キーが押されている場合は true、押されていない場合は false になります。

bShiftDown イベント発生時の Shift キーの状態を記録するブール値。キーが押されている場合は true、押されていない場合は false になります。

bShiftSelect アプリケーションの環境設定の Shift 選択状態を示すブール値。オンの場合は true、オフの場合は false になります。

戻り値

なし。

説明

メソッド。矢印ツールからマウスのダブルクリックを実行します。

例

次の例では、指定した位置でマウスのダブルクリックを実行します。

```
fl.getDocumentDOM().mouseDbClick({x:392.9, y:73}, false, false, true);
```

関連項目

[document.mouseClick\(\)](#)

document.moveSelectedBezierPointsBy()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.moveSelectedBezierPointsBy(delta)
```

パラメータ

delta 選択したベジェポイントを移動するのに使用する x 座標と y 座標を指定するピクセル単位の浮動小数値のペア。たとえば、(*x*:1,*y*:2) は、現在位置から右に 1 ピクセル、下に 2 ピクセルの位置を指定します。

戻り値

なし

説明

メソッド。最低 1 つのパスと最低 1 つのベジェポイントを選択している場合、選択したすべてのパスで選択したすべてのベジェポイントを、指定距離だけ移動します。

例

次の例では、選択したベジェポイントを、右に 10 ピクセル、下に 5 ピクセル移動します。

```
fl.getDocumentDOM().moveSelectedBezierPointsBy({x:10, y:5});
```

document.moveSelectionBy()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.moveSelectionBy(distanceToMove)
```

パラメータ

distanceToMove 選択したオブジェクトを移動するのに使用する *x* 座標と *y* 座標を指定する浮動小数値のペア。たとえば、(*{x:1,y:2}*) は、現在位置から右に 1 ピクセル、下に 2 ピクセルの位置を指定します。

戻り値

なし

説明

メソッド。選択したオブジェクトを、指定距離だけ移動します。

×
中

矢印キーを使用してアイテムを移動するとき、矢印キーのすべての押し下げが [ヒストリ] パネルで 1 回の移動として結合されます。[ヒストリ] パネルで複数手順を踏むのではなく、矢印キーを繰り返し押した場合、メソッドでは 1 回の手順で実行し、パラメータは、矢印キーを繰り返し押し下げた状態に反映されます。

選択方法の詳細については、[document.setSelectionRect\(\)](#)、[document.mouseClick\(\)](#)、[document.mouseDblClick\(\)](#)、および [Element オブジェクト](#) を参照してください。

例

次の例では、選択したアイテムを、右に 62 ピクセル、下に 84 ピクセル移動します。

```
flash.getDocumentDOM().moveSelectionBy({x:62, y:84});
```

document.name

使用できるバージョン

Flash MX 2004

シンタックス

```
document.name
```

説明

読み取り専用プロパティ。ドキュメント (FLA ファイル) の名前を表す文字列です。

例

次の例では、変数 `fileName` に、ドキュメント配列の最初のドキュメントのファイル名を設定します。

```
var fileName = fl.documents[0].name;
```

次の例では、開いているすべてのドキュメントの名前を [出力] パネルに表示します。

```
var openDocs = fl.documents;
for(var i=0;i < openDocs.length; i++){
    fl.trace(i + " " + openDocs[i].name +"\n");
}
```

document.optimizeCurves()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.optimizeCurves(smoothing, bUseMultiplePasses)
```

パラメータ

smoothing スムージングを指定する 0 から 100 までの整数。0 はスムージングなし、100 は最大スムージングを指定します。

bUseMultiplePasses ブール値。true を設定すると、複数処理を適用することを示します。処理は遅くなりますが、品質は向上します。このパラメータは、[曲線の最適化] ダイアログボックスの [複数処理を適用] ボタンをクリックする操作と同じエフェクトが得られます。

戻り値

なし

説明

メソッド。現在選択しているオブジェクトのスムージングを最適化します。複数処理を適用してスムージングの品質を上げることもできます。[修正]-[シェイプ]-[最適化] を選択する操作と同じです。

例

次の例では、現在選択しているオブジェクトの曲線を、複数処理を適用して 50 度のスムージングに最適化します。

```
fl.getDocumentDOM().optimizeCurves(50, true);
```

document.path

使用できるバージョン

Flash MX 2004

シンタックス

`document.path`

説明

読み取り専用プロパティ。ドキュメントのパスをプラットフォーム固有の形式で表すストリングです。ドキュメントをまだ保存したことがない場合は、このプロパティは `undefined` になります。

例

次の例では、ドキュメント配列の最初のドキュメントのパスを [出力] パネルに表示します。

```
var filePath = flash.documents[0].path;  
fl.trace(filePath);
```

document.publish()

使用できるバージョン

Flash MX 2004

シンタックス

`document.publish()`

パラメータ

なし

戻り値

なし

説明

メソッド。アクティブな [パブリッシュ設定]([ファイル]-[パブリッシュ設定])に従って、ドキュメントをパブリッシュします。[ファイル]-[パブリッシュ]を選択する操作と同じです。

例

次の例では、現在のドキュメントをパブリッシュします。

```
fl.getDocumentDOM().publish();
```

document.publishProfiles

使用できるバージョン

Flash MX 2004

シンタックス

`document.publishProfiles`

説明

読み取り専用プロパティ。ドキュメントのパブリッシュプロファイル名の配列です。

例

次の例では、ドキュメントのパブリッシュプロファイルの名前を表示します。

```
var myPubProfiles = fl.getDocumentDOM().publishProfiles;
for (var i=0; i < myPubProfiles.length; i++){
    fl.trace(myPubProfiles[i]);
}
```

document.punch()

使用できるバージョン

Flash 8

シンタックス

`document.punch()`

パラメータ

なし

戻り値

ブール値。成功した場合は `true` を返します。それ以外の場合は `false` を返します。

説明

メソッド。選択した一番上の描画オブジェクトを使用して、選択した下位のすべての描画オブジェクトを型抜きします。描画オブジェクトを選択していない場合、または選択したアイテムが描画オブジェクトでない場合は `false` を返します。

例

次の例では、選択した描画オブジェクトの下にある描画オブジェクトを型抜きします。

```
fl.getDocumentDOM().punch();
```

関連項目

[document.crop\(\)](#), [document.deleteEnvelope\(\)](#), [document.intersect\(\)](#),
[document.union\(\)](#), [shape.isDrawingObject](#)

document.removeAllFilters()

使用できるバージョン

Flash 8

シンタックス

```
document.removeAllFilters()
```

パラメータ

なし

戻り値

なし

説明

メソッド。選択したオブジェクトからすべてのフィルタを削除します。

例

次の例では、選択したオブジェクトからすべてのフィルタを削除します。

```
fl.getDocumentDOM().removeAllFilters();
```

関連項目

[document.addFilter\(\)](#), [document.changeFilterOrder\(\)](#),
[document.disableAllFilters\(\)](#), [document.getFilters\(\)](#), [document.removeFilter\(\)](#),
[Filter](#) オブジェクト

document.removeDataFromDocument()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.removeDataFromDocument(name)
```

パラメータ

name 削除するデータの名前を示すストリング

戻り値

なし

説明

メソッド。ドキュメントに追加されている指定名の永続データを削除します。

例

次の例では、"myData" という名前の永続データをドキュメントから削除します。

```
fl.getDocumentDOM().removeDataFromDocument("myData");
```

関連項目

[document.addDataToDocument\(\)](#), [document.documentHasData\(\)](#),
[document.getDataFromDocument\(\)](#)

document.removeDataFromSelection()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.removeDataFromSelection(name)
```

パラメータ

name 削除する永続データの名前を示すストリング。

戻り値

なし

説明

メソッド。選択範囲に追加されている指定名の永続データを削除します。

例

次の例では、"myData" という名前の永続データを選択範囲から削除します。

```
fl.getDocumentDOM().removeDataFromSelection("myData");
```

関連項目

[document.addDataToSelection\(\)](#)

document.removeFilter()

使用できるバージョン

Flash 8

シンタックス

```
document.removeFilter(filterIndex)
```

パラメータ

filterIndex 選択したオブジェクトから削除するフィルタのゼロから始まるインデックスを表す整数。

戻り値

なし

説明

メソッド。選択したオブジェクトのフィルタリストから、指定したフィルタを削除します。

例

次の例では、選択したオブジェクトのフィルタリストから、最初のフィルタ (インデックス値 0) を削除します。

```
fl.getDocumentDOM().removeFilter(0);
```

関連項目

[document.addFilter\(\)](#), [document.changeFilterOrder\(\)](#), [document.disableFilter\(\)](#), [document.getFilters\(\)](#), [document.removeAllFilters\(\)](#), [Filter](#) オブジェクト

document.renamePublishProfile()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.renamePublishProfile([profileNewName])
```

パラメータ

profileNewName プロファイルの新規の名前を指定するオプションのパラメータ。新規の名前は一意である必要があります。名前を指定しない場合は、デフォルト名が使用されます。

戻り値

ブール値。名前が正常に変更された場合は true を返し、変更されなかった場合は false を返します。

説明

メソッド。現在のプロファイルの名前を変更します。

例

次の例では、現在のプロファイルの名前をデフォルト名に変更して表示します。

```
alert(fl.getDocumentDOM().renamePublishProfile());
```

document.renameScene()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.renameScene(name)
```

パラメータ

name シーンの新規の名前を示すストリング。

戻り値

ブール値。名前が正常に変更された場合は `true` を返し、変更されなかった場合は `false` を返します。新規の名前が一意でない場合は、`false` を返します。

説明

メソッド。[シーン] パネルで現在選択しているシーンの名前を変更します。選択したシーンの新規の名前は一意である必要があります。

例

次の例では、現在のシーンの名前を "new name" に変更します。

```
var success = fl.getDocumentDOM().renameScene("new name");
```

document.reorderScene()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.reorderScene(sceneToMove, sceneToPutItBefore)
```

パラメータ

sceneToMove 移動するシーンを指定する整数。0 (ゼロ) は最初のシーンを示します。

*sceneToPutItBefore sceneToMove*で指定したシーンの移動の前に、シーンを指定する整数。最初のシーンは0を指定します。たとえば、*sceneToMove*に1、*sceneToPutItBefore*に0を指定すると、2番目のシーンが最初のシーンの前に移動します。シーンを最後に移動するには-1を指定します。

戻り値

なし

説明

メソッド。指定したシーンを、指定した別のシーンの前に移動します。

例

次の例では、2番目のシーンを最初のシーンの前に移動します。

```
fl.getDocumentDOM().reorderScene(1, 0);
```

document.resetOvalObject()

使用できるバージョン

Flash CS3 Professional

シンタックス

```
document.resetOvalObject()
```

パラメータ

なし

戻り値

なし。

説明

メソッド。プロパティインスペクタのすべての値を **Oval** オブジェクトのデフォルト値に設定します。**Oval** オブジェクトが選択されている場合、それらのプロパティはデフォルト値にリセットされます。

例

次の例では、現在のドキュメントの **Oval** オブジェクトのプロパティをデフォルト値にリセットします。

```
fl.getDocumentDOM().resetOvalObject();
```

関連項目

[document.resetRectangleObject\(\)](#)

document.resetRectangleObject()

使用できるバージョン

Flash CS3 Professional

シンタックス

```
document.resetRectangleObject()
```

パラメータ

なし

戻り値

なし。

説明

メソッド。プロパティインスペクタのすべての値を Rectangle オブジェクトのデフォルト値に設定します。Rectangle オブジェクトが選択されている場合、それらのプロパティはデフォルト値にリセットされます。

例

次の例では、現在のドキュメントの Rectangle オブジェクトのプロパティをデフォルト値にリセットします。

```
fl.getDocumentDOM().resetRectangleObject();
```

関連項目

[document.resetOvalObject\(\)](#)

document.resetTransformation()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.resetTransformation()
```

パラメータ

なし

戻り値

なし

説明

メソッド。変換マトリックスをリセットします。[修正]-[変形]-[変形の解除] を選択する操作と同じです。

例

次の例では、現在選択しているオブジェクトの変換マトリックスをリセットします。

```
fl.getDocumentDOM().resetTransformation();
```

document.revert()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.revert()
```

パラメータ

なし

戻り値

なし。

説明

メソッド。指定したドキュメントを、前回保存したときの状態に戻します。[ファイル]-[復帰] を選択する操作と同じです。

例

次の例では、現在のドキュメントを前回保存したときの状態に戻します。

```
fl.getDocumentDOM().revert();
```

関連項目

[document.canRevert\(\)](#), [fl.revertDocument\(\)](#)

document.revertToLastVersion()

使用できるバージョン

Flash CS3 Professional

シンタックス

```
document.revertToLastVersion()
```

パラメータ

なし

戻り値

ブール値。ドキュメントの復帰に成功した場合は `true`、それ以外の場合は `false` になります。

説明

メソッド。ファイルを復帰できる場合は、ファイルを復帰するかどうかを確認するダイアログボックスが表示されます。ユーザーが確認すると、ファイルを **Version Cue** サーバーに保存されたバージョンに復帰し、エラーを [出力] パネルにログします。

例

次の例では、現在のドキュメントを **Version Cue** サーバーに保存されたバージョンに復帰します。

```
fl.getDocumentDOM().revertToLastVersion();
```

関連項目

[document.canSaveAVersion\(\)](#), [document.saveAVersion\(\)](#),
[document.synchronizeWithHeadVersion\(\)](#), [fl.revertDocumentToLastVersion\(\)](#)

document.rotateSelection()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.rotateSelection(angle [, rotationPoint])
```

パラメータ

angle 回転角度を指定する浮動小数値。

rotationPoint 回転する境界ボックスの面を指定するストリング。指定できる値は、"top right"、"top left"、"bottom right"、"bottom left"、"top center"、"right center"、"bottom center"、および "left center" です。指定していない場合は、変形点が使用されます。このパラメータはオプションです。

戻り値

なし

説明

メソッド。選択したオブジェクトを指定した角度だけ回転します。自由変形ツールを使用してオブジェクトを回転する操作と同じです。

例

次の例では、選択範囲を変形点を中心に 45 度回転します。

```
flash.getDocumentDOM().rotateSelection(45);
```

次の例では、選択範囲を左下隅を中心に 45 度回転します。

```
fl.getDocumentDOM().rotateSelection(45, "bottom left");
```

document.save()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.save([boolToSaveAs])
```

パラメータ

boolToSaveAs オプションのパラメータ。true を設定するか省略した場合、ファイルをまだ保存したことがない場合は、[名前を付けて保存] ダイアログボックスを表示します。false を設定した場合は、ファイルをまだ保存したことがない場合でも、ファイルは保存されません。

戻り値

ブール値。保存操作が正常に終了した場合は true を返します。それ以外の場合は false を返します。

説明

メソッド。ドキュメントをデフォルトの場所に保存します。[ファイル]-[保存] を選択する操作と同じです。

同じ名前前で保存するのではなく、ファイルの名前を指定するには、[fl.saveDocument\(\)](#) を使用します。

×
❗

ファイルが新しく、まだ変更も保存もされていない場合、またはファイルを最後に保存してから変更していない場合、このメソッドは何も実行せず、false を返します。未保存または未修正のファイルを保存できるようにするには、[document.saveAndCompact\(\)](#) または [fl.saveDocumentAs\(\)](#) を使用します。

例

次の例では、現在のドキュメントをデフォルトの場所に保存します。

```
fl.getDocumentDOM().save();
```

関連項目

[document.saveAndCompact\(\)](#), [fl.saveAll\(\)](#), [fl.saveDocument\(\)](#),
[fl.saveDocumentAs\(\)](#)

document.saveAndCompact()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.saveAndCompact([bOkToSaveAs])
```

パラメータ

bOkToSaveAs オプションのパラメータ。true を設定するか省略した場合、ファイルをまだ保存したことがない場合は、[名前を付けて保存] ダイアログボックスを表示します。false を設定した場合は、ファイルをまだ保存したことがない場合でも、ファイルは保存されません。デフォルト値は true です。

戻り値

ブール値。保存して最適化操作が正常に終了した場合は true を返します。正常に終了しない場合は false を返します。

説明

メソッド。ファイルを保存して最適化します。[ファイル]-[保存して最適化] を選択する操作と同じです。

✕
#

ファイルをまだ保存したことがない場合は、ユーザーが[名前を付けて保存] ダイアログボックスでキャンセルを選択しても、このメソッドは true を返します。ファイルが保存されたかどうかを正確に判定するには、[fl.saveDocumentAs\(\)](#) を使用します。

例

次の例では、現在のドキュメントを保存して最適化します。

```
fl.getDocumentDOM().saveAndCompact();
```

関連項目

[document.save\(\)](#), [fl.saveDocumentAs\(\)](#), [fl.saveDocument\(\)](#), [fl.saveAll\(\)](#)

document.saveAVersion()

使用できるバージョン

Flash CS3 Professional

シンタックス

```
document.saveAVersion()
```

パラメータ

なし

戻り値

ブール値。ドキュメントのバージョンが Version Cue サーバーに正常に保存された場合は true、それ以外の場合は false になります。

説明

メソッド。ファイルを Version Cue サーバーに保存できる場合は、ユーザーがバージョンのコメントを入力するためのダイアログボックスを表示し、指定したドキュメントのバージョンをサーバーに保存し、エラーを [出力] パネルにログします。

× 中	サーバーの資格情報が現在のアプリケーションセッションにキャッシュされていないために Flash でファイルを保存できない場合は、[出力] パネルに認証エラーが表示されます。このエラーが発生した場合は、[ファイル]-[開く] ダイアログボックスを使用して、正しい資格情報で Version Cue ワークスペースを開く必要があります。次回から、このサーバーの JavaScript API 呼び出しは成功します。
--------	---

例

詳細については、[document.canSaveAVersion\(\)](#) を参照してください。

関連項目

[document.canSaveAVersion\(\)](#), [document.revertToLastVersion\(\)](#),
[document.synchronizeWithHeadVersion\(\)](#)

document.scaleSelection()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.scaleSelection(xScale, yScale [, whichCorner])
```

パラメータ

xScale 拡大 / 縮小する x の量を指定する浮動小数値。

yScale 拡大 / 縮小する y の量を指定する浮動小数値。

whichCorner 変形の中心になるエッジを指定するストリング。省略すると、変形点を中心に拡大 / 縮小します。指定できる値は、"bottom left"、"bottom right"、"top right"、"top left"、"top center"、"right center"、"bottom center"、および "left center" です。このパラメータはオプションです。

戻り値

なし

説明

メソッド。選択したオブジェクトを指定した量だけ拡大 / 縮小します。自由変形ツールを使用してオブジェクトを拡大 / 縮小する操作と同じです。

例

次の例では、現在選択しているオブジェクトの幅を元の幅の 2 倍に拡大し、高さを半分に縮小します。

```
fl.getDocumentDOM().scaleSelection(2.0, 0.5);
```

次の例では、選択したオブジェクトを縦に反転します。

```
fl.getDocumentDOM().scaleSelection(1, -1);
```

次の例では、選択したオブジェクトを横に反転します。

```
fl.getDocumentDOM().scaleSelection(-1, 1);
```

次の例では、選択したオブジェクトを上部中心から 1.9 に縦に拡大 / 縮小します。

```
fl.getDocumentDOM().scaleSelection(1, 1.90, 'top center');
```

document.screenOutline

使用できるバージョン

Flash MX 2004

シンタックス

`document.screenOutline`

説明

読み取り専用プロパティ。ドキュメントの現在の **ScreenOutline** オブジェクトです。オブジェクトに初めてアクセスする前に、`document.allowScreens()` を使用して、プロパティが存在するかどうかを確認する必要があります。

例

次の例では、値の配列を `screenOutline` プロパティに表示します。

```
var myArray = new Array();
for(var i in fl.getDocumentDOM().screenOutline) {
    myArray.push(" "+i+" : "+fl.getDocumentDOM().screenOutline[i]) ;
}
fl.trace("Here is the property dump for screenOutline: "+myArray);
```

関連項目

[document.allowScreens\(\)](#), [ScreenOutline](#) オブジェクト

document.selectAll()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.selectAll()
```

パラメータ

なし

戻り値

なし。

説明

メソッド。ステージにあるすべてのアイテムを選択します。Ctrl + A (Windows) または Command + A (Macintosh) を押す操作、または [編集]-[すべての選択] を選択する操作と同じです。

例

次の例では、現在表示されているすべてのアイテムを選択します。

```
fl.getDocumentDOM().selectAll();
```

関連項目

[document.selection](#), [document.selectNone\(\)](#)

document.selection

使用できるバージョン

Flash MX 2004

シンタックス

```
document.selection
```

説明

プロパティ。ドキュメントで選択したオブジェクトの配列です。何も選択していない場合は、長さゼロの配列を返します。開いているドキュメントがない場合は、null を返します。

オブジェクトを配列に追加するには、最初に、次のいずれかの方法でオブジェクトを選択します。

- 手動でステージのオブジェクトを選択する。

- `document.setSelectionRect()`、`document.setSelectionBounds()`、`document.onClick()`、`document.ondblClick()`、または `document.selectAll()` など、選択メソッドの1つを使用する。
- フレームを手動で選択する。
- `timeline.getSelectedFrames()`、`timeline.setSelectedFrames()`、または `timeline.selectAllFrames()` など、`Timeline` オブジェクトのメソッドを使用して次のようにフレームを選択する。

```
fl.getDocumentDOM().selection =
  fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0];
```
- 特定のフレーム内のすべてのエレメントを指定する ([Element オブジェクト](#) を参照)。次に示す最初の例を参照してください。
- エレメントの配列を作成し、その配列を `document.selection` 配列に割り当てる。次に示す3番目の例を参照してください。

例

次の例では、フレーム 11 のすべてのエレメントを現在の選択範囲に割り当てます (インデックス値はフレーム番号値とは異なります)。

```
fl.getDocumentDOM().getTimeline().currentFrame = 10;
fl.getDocumentDOM().selection =
  fl.getDocumentDOM().getTimeline().layers[0].frames[10].elements;
```

次の例では、ステージの左上隅に矩形を作成し、矩形の下にテキストストリングを作成します。次に、`document.setSelectionRect()` を使用して両方のオブジェクトを選択し、`document.selection` 配列に追加します。最後に、`document.selection` の内容を [出力] パネルに表示します。

```
fl.getDocumentDOM().addNewRectangle({left:0, top:0, right:99, bottom:99}, 0);
fl.getDocumentDOM().addNewText({left:-1, top:117.3, right:9.2, bottom:134.6});
fl.getDocumentDOM().setTextString('Hello World');
fl.getDocumentDOM().setSelectionRect({left:-28, top:-22, right:156.0,
  bottom:163});
```

```
var theSelectionArray = fl.getDocumentDOM().selection;
```

```
for(var i=0;i<theSelectionArray.length;i++){
  fl.trace("fl.getDocumentDOM().selection["+i+"] = " + theSelectionArray[i]);
}
```

次の例は、上級者向けの例です。レイヤー配列内とエレメント配列内をループして、特定シンボルのインスタンスを特定して選択する方法を示します。この例を拡張して、複数のフレームまたはシーンのループを含めることができます。次の例では、最初のフレームにあるムービークリップ `myMovieClip` のすべてのインスタンスを、現在の選択範囲に割り当てます。

```

// レイヤー配列を変数 "theLayers" に割り当てます。
var theLayers = fl.getDocumentDOM().getTimeline().layers;
// "myMovieClip" のインスタンスであるすべてのエレメントを
// 保持するための配列を作成します。
var myArray = new Array();
// カウンタ変数
var x = 0;
// 全レイヤーでループを開始します。
for (var i = 0; i < theLayers.length; i++) {
    // フレーム 1 のエレメントの配列を取得し、
    // 配列 "theElems" に割り当てます。
    var theElems = theLayers[i].frames[0].elements;
    // レイヤー上のエレメントでのループを開始します。
    for (var c = 0; c < theElems.length; c++) {
        // エレメントタイプが "instance" であるかどうかを確認します。
        if (theElems[c].elementType == "instance") {
            // エレメントがインスタンスである場合は、"myMovieClip" の
            // インスタンスであるかどうかを確認します。
            if (theElems[c].libraryItem.name == "myMovieClip") {
                // "myMovieClip" のインスタンスであるエレメントを "myArray" に割り当てます。
                myArray[x] = theElems[c];
                // カウンタ変数をインクリメントします。
                x++;
            }
        }
    }
}
// "myMovieClip" のすべてのインスタンスの "myArray" への
// 割り当てが終了したら、document.selection 配列を myArray に
// 設定します。これにより、ステージのオブジェクトが選択されます。
fl.getDocumentDOM().selection = myArray;

```

document.selectNone()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.selectNone()
```

パラメータ

なし

戻り値

なし。

説明

メソッド。選択したアイテムの選択を解除します。

例

次の例では、選択したアイテムの選択を解除します。

```
fl.getDocumentDOM().selectNone();
```

関連項目

[document.selectAll\(\)](#), [document.selection](#)

document.setAlignToDocument()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.setAlignToDocument(bToStage)
```

パラメータ

bToStage ブール値。true を設定すると、オブジェクトをステージに整列します。false を設定すると、整列は実行されません。

戻り値

なし

説明

メソッド。[document.align\(\)](#)、[document.distribute\(\)](#)、[document.match\(\)](#)、および [document.space\(\)](#) の環境設定を行います。[整列] パネルにある [ステージを基準] ボタンを有効にする操作と同じです。

例

次の例では、[整列] パネルにある [ステージを基準] ボタンを有効にして、オブジェクトをステージに整列します。

```
fl.getDocumentDOM().setAlignToDocument(true);
```

関連項目

[document.getAlignToDocument\(\)](#)

document.setBlendMode()

使用できるバージョン

Flash 8

シンタックス

```
document.setBlendMode(mode)
```

パラメータ

mode 選択したオブジェクトのブレンドモードを表すストリング。有効な値は、"normal"、"layer"、"multiply"、"screen"、"overlay"、"hardlight"、"lighten"、"darken"、"difference"、"add"、"subtract"、"invert"、"alpha"、および "erase" です。

戻り値

なし

説明

メソッド。選択したオブジェクトのブレンドモードを設定します。

例

次の例では、選択したオブジェクトのブレンドモードに "add" を設定します。

```
fl.getDocumentDOM().setBlendMode("add");
```

関連項目

[document.addFilter\(\)](#), [document.setFilterProperty\(\)](#), [symbolInstance.blendMode](#)

document.setCustomFill()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.setCustomFill(fill)
```

パラメータ

fill 使用する塗り設定を指定する Fill オブジェクト。詳細については、[Fill オブジェクト](#) を参照してください。

戻り値

なし

説明

メソッド。ツールパネル、プロパティインスペクタ、および選択したシェイプの塗り設定を行います。このメソッドにより、オブジェクトを描画して選択してから塗り設定を変更するのではなく、オブジェクトを描画する前にスクリプトで塗り設定を行うことができます。ツールパネルとプロパティインスペクタの塗り設定をスクリプトで変更することもできます。

例

次の例では、ツールパネル、プロパティインスペクタ、および選択したシェイプの塗り色見本の色を白色に変更します。

```
var fill = fl.getDocumentDOM().getCustomFill();
fill.color = '#FFFFFF';
fill.style = "solid";
fl.getDocumentDOM().setCustomFill(fill);
```

関連項目

[document.getCustomFill\(\)](#)

document.setCustomStroke()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.setCustomStroke(stroke)
```

パラメータ

stroke [Stroke](#) オブジェクト。

戻り値

なし

説明

メソッド。ツールパネル、プロパティインスペクタ、および選択したシェイプの線の設定を行います。このメソッドにより、オブジェクトを描画して選択してから線の設定を変更するのではなく、オブジェクトを描画する前にスクリプトで線の設定を行うことができます。ツールパネルとプロパティインスペクタの線の設定をスクリプトで変更することもできます。

例

次の例では、ツールパネル、プロパティインスペクタ、および選択したシェイプの線の太さの設定を変更します。

```
var stroke = fl.getDocumentDOM().getCustomStroke();
stroke.thickness += 2;
fl.getDocumentDOM().setCustomStroke(stroke);
```

関連項目

[document.getCustomStroke\(\)](#)

document.setElementProperty()

使用できるバージョン

Flash MX 2004

シンタックス

`document.setElementProperty(property, value)`

パラメータ

property 設定する Element プロパティの名前を示すストリング。プロパティ名と使用できる値については、[211 ページの「Element オブジェクトのプロパティ一覧」](#)を参照してください。



`element.elementType`、`element.top`、および `element.left` などの読み取り専用プロパティの値を、このメソッドで設定することはできません。

value 指定した Element プロパティで設定したい値を指定する整数。

戻り値

なし

説明

メソッド。ドキュメントで選択したオブジェクトの指定した Element プロパティを設定します。オブジェクトを選択していない場合は、何も実行しません。

例

次の例では、選択しているすべてのオブジェクトの幅を 100 に設定し、高さを 50 に設定します。

```
fl.getDocumentDOM().setElementProperty("width", 100);
fl.getDocumentDOM().setElementProperty("height", 50);
```

document.setElementTextAttr()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.setElementTextAttr(attrName, attrValue [, startIndex  
    [, endIndex]])
```

パラメータ

attrName 変更する TextAttrs プロパティの名前を指定するストリング。

attrValue TextAttrs プロパティに設定する値。プロパティ名と使用できる値については、[509 ページの「TextAttrs オブジェクトのプロパティ一覧」](#)を参照してください。

startIndex 影響を受ける最初の文字のインデックスを指定する整数。このパラメータはオプションです。

endIndex 影響を受ける最後の文字のインデックスを指定する整数。このパラメータはオプションです。

戻り値

ブール値。テキスト属性プロパティが1つでも変更されている場合は true を返し、それ以外の場合は false を返します。

説明

メソッド。選択したテキストアイテムの指定した textAttrs プロパティに指定値を設定します。プロパティ名と使用できる値については、[509 ページの「TextAttrs オブジェクトのプロパティ一覧」](#)を参照してください。オプションのパラメータを渡していない場合は、現在選択しているテキスト範囲のスタイルを設定するか、テキストを選択していない場合は、テキストフィールド全体のスタイルを設定します。*startIndex* だけを渡した場合は、文字の属性を設定します。*startIndex* と *endIndex* を渡している場合、*startIndex* から *endIndex* の前の文字までの文字の属性を設定します。段落スタイルを指定すると、範囲内にあるすべての段落が影響を受けます。

例

次の例では、選択したテキストアイテムに fillColor、italic、bold のテキスト属性を設定します。

```
var success = fl.getDocumentDOM().setElementTextAttr("fillColor", "#00ff00");  
var pass = fl.getDocumentDOM().setElementTextAttr("italic", true, 10);  
var ok = fl.getDocumentDOM().setElementTextAttr("bold", true, 5, 15);
```

document.setFillColor()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.setFillColor(color)
```

パラメータ

color 塗りのカラー。次のいずれかの形式です。

- "#RRGGBB" または "#RRGGBBAA" 形式のストリング
- 0xRRGGBB 形式の 16 進数
- 16 進数と等しい 10 進表現の整数

null を設定すると、塗りのカラーは設定されません。塗りの色見本で塗りなしに設定する操作と同じです。

戻り値

なし

説明

メソッド。選択したオブジェクトの塗りのカラーを指定カラーに変更します。ツールパネルおよびプロパティインスペクタの塗りのカラーを変更する方法については、[document.setCustomFill\(\)](#) を参照してください。

例

次の例の最初の 3 つのステートメントでは、それぞれ別のカラー指定形式を使用して塗りのカラーを設定します。4 番目のステートメントでは、塗りなしに設定します。

```
flash.getDocumentDOM().setFillColor("#cc00cc");  
flash.getDocumentDOM().setFillColor(0xcc00cc);  
flash.getDocumentDOM().setFillColor(120000);  
flash.getDocumentDOM().setFillColor(null);
```

document.setFilterProperty()

使用できるバージョン

Flash 8

シンタックス

```
document.setFilterProperty(property, filterIndex, value)
```

パラメータ

property 設定するプロパティを指定するストリング。指定できる値は、"blurX"、"blurY"、"quality"、"angle"、"distance"、"strength"、"knockout"、"inner"、"bevelType"、"color"、"shadowColor"、および "highlightColor" です。

filterIndex フィルタリストのフィルタのゼロから始まるインデックスを指定する整数。

value 指定したフィルタプロパティに設定する値を指定する数値またはストリング。指定できる値は、プロパティと設定中のフィルタにより異なります。

戻り値

なし

説明

メソッド。フィルタプロパティをサポートする現在選択しているオブジェクトに対し、指定したフィルタプロパティを設定します。

例

次の例では、選択したオブジェクトのフィルタリストにある 2 番目のフィルタ (インデックス値 1) の *quality* プロパティに 2 を設定し、選択したオブジェクトのフィルタリストにある 1 番目のフィルタの *shadowColor* プロパティを設定します。

```
fl.getDocumentDOM().setFilterProperty("quality", 1, 2);
fl.getDocumentDOM().setFilterProperty("shadowColor", 0, "#FF00FF");
```

関連項目

[document.addFilter\(\)](#), [document.getFilters\(\)](#), [document.setBlendMode\(\)](#),
[document.setFilters\(\)](#), [Filter](#) オブジェクト

document.setFilters()

使用できるバージョン

Flash 8

シンタックス

```
document.setFilters(filterArray)
```

パラメータ

filterArray 現在指定しているフィルタの配列。

戻り値

なし

説明

メソッド。選択したオブジェクトにフィルタを適用します。document.getFilters() を呼び出してフィルタに変更を加えた後で、このメソッドを使用します。

例

次の例では、選択したオブジェクトのフィルタを取得し、すべてのぼかしフィルタの blurX プロパティに 50 を設定します。

```
var myFilters = fl.getDocumentDOM().getFilters();
for (i=0; i < myFilters.length; i++) {
    if (myFilters[i].name == "blurFilter"){
        myFilters[i].blurX = 50;
    }
}
fl.getDocumentDOM().setFilters(myFilters);
```

関連項目

[document.addFilter\(\)](#), [document.getFilters\(\)](#), [document.setFilterProperty\(\)](#), [Filter](#) オブジェクト

document.setInstanceAlpha()

使用できるバージョン

Flash MX 2004

シンタックス

document.setInstanceAlpha(*opacity*)

パラメータ

opacity インスタンスの透明度を調整する整数。0 (透明) から 100 (完全に濃く塗りつぶした状態) までの値。

戻り値

なし

説明

メソッド。インスタンスの不透明度を設定します。

例

次の例では、着色の不透明度を 50 に設定します。

```
fl.getDocumentDOM().setInstanceAlpha(50);
```

document.setInstanceBrightness()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.setInstanceBrightness(brightness)
```

パラメータ

brightness 明度を指定する整数。-100 (黒) から 100 (白) までの値。

戻り値

なし

説明

メソッド。インスタンスの明度を設定します。

例

次の例では、インスタンスの明度を 50 に設定します。

```
fl.getDocumentDOM().setInstanceBrightness(50);
```

document.setInstanceTint()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.setInstanceTint( color, strength )
```

パラメータ

color 着色のカラー。次のいずれかの形式です。

- "#RRGGBB" または "#RRGGBBAA" 形式のストリング
- 0xRRGGBB 形式の 16 進数
- 16 進数と等しい 10 進表現の整数

このパラメータは、プロパティインスペクタでシンボルの Color: Tint 値を選択する操作と同じです。

strength 着色の不透明度を指定する整数。0 から 100 までの値。

戻り値

なし

説明

メソッド。インスタンスの着色を設定します。

例

次の例では、選択したインスタンスの着色に、不透明度 50 の赤色を設定します。

```
fl.getDocumentDOM().setInstanceTint(0xff0000, 50);
```

document.setMetadata()

使用できるバージョン

Flash 8

シンタックス

```
document.setMetadata(strMetadata)
```

パラメータ

strMetadata ドキュメントと関連付けられる XML メタデータを含むストリング。詳細については、次の説明を参照してください。

戻り値

ブール値。成功した場合は true を返します。それ以外の場合は false を返します。

説明

メソッド。指定したドキュメントの XML メタデータを設定します。既存のメタデータは上書きされません。*strMetadata* として渡した XML は検証され、保存する前に書き込みされる場合があります。正しい XML として検証されない場合、または特定規則に違反する場合は、XML メタデータを設定せずに、false を返します (false である場合、詳細なエラー情報を取得することはできません)。

×
中

true が返された場合でも、設定された XML は、渡したストリングと同じストリングになるとは限りません。XML に設定された正確な値を取得するには、`document.getMetadata()` を使用します。

メタデータの形式は、XMP 仕様に準拠した RDF です。RDF と XMP の詳細については、次のソースを参照してください。

- RDF 入門 (www.w3.org/TR/rdf-primer/)
- RDF 仕様 (www.w3.org/TR/1999/REC-rdf-syntax-19990222/)
- XMP のホームページ (www.adobe.com/products/xmp/)

例

次の例では、同じデータを表すいくつかの異なる有効な方法を示します。2 番目以外の例では、データが `Document.setMetadata()` に送られても、データは再書き込みされません (改行の削除は除く)。

最初の例では、メタデータがタグの中にあり、さまざまなスキーマがそれぞれ別の `rdf:Description` タグに分けて入れられています。

```
<rdf:RDF xmlns:rdf='http://www.w3.org/1999/02/22-rdf-syntax-ns#'>
  <rdf:Description rdf:about='' xmlns:dc='http://purl.org/dc/1.1/'>
    <dc:title>Simple title</dc:title>
    <dc:description>Simple description</dc:description>
  </rdf:Description>
  <rdf:Description rdf:about='' xmlns:xmp='http://ns.adobe.com/xap/1.0/'>
    <xmp:CreateDate>2004-10-12T10:29-07:00</xmp:CreateDate>
    <xmp:CreatorTool>Flash Authoring WIN 8,0,0,215</xmp:CreatorTool>
  </rdf:Description>
</rdf:RDF>
```

2 番目の例では、メタデータがタグの中にあり、さまざまなスキーマがすべて 1 つの `rdf:Description` タグに入っています。この例にはコメントも含まれています。このコメントは、`Document.setMetadata()` では無視されて破棄されます。

```
<rdf:RDF xmlns:rdf='http://www.w3.org/1999/02/22-rdf-syntax-ns#'>
  <!-- これは最初の rdf:Description タグの前 -->
  <rdf:Description rdf:about='' xmlns:dc='http://purl.org/dc/1.1/'>
    <dc:title>Simple title</dc:title>
    <dc:description>Simple description</dc:description>
  </rdf:Description>
  <!-- これは 2 つの rdf:Description タグの間 -->
  <rdf:Description rdf:about='' xmlns:xmp='http://ns.adobe.com/xap/1.0/'>
    <xmp:CreateDate>2004-10-12T10:29-07:00</xmp:CreateDate>
    <xmp:CreatorTool>Flash Authoring WIN 8,0,0,215</xmp:CreatorTool>
  </rdf:Description>
  <!-- これは 2 番目の rdf:Description タグの後 -->
</rdf:RDF>
```

3 番目の例では、メタデータが属性の中にあり、さまざまなスキーマがすべて 1 つの `rdf:Description` タグに入っています。

```
<rdf:RDF xmlns:rdf='http://www.w3.org/1999/02/22-rdf-syntax-ns#'>
  <rdf:Description rdf:about='' xmlns:dc='http://purl.org/dc/1.1/'
    dc:title='Simple title'
    dc:description='Simple description' />
  <rdf:Description rdf:about='' xmlns:xmp='http://ns.adobe.com/xap/1.0/'
    xmp:CreateDate='2004-10-12T10:29-07:00' xmp:CreatorTool='Flash Authoring WIN
    8,0,0,215' />
</rdf:RDF>
```

関連項目

[document.getMetadata\(\)](#)

document.setMobileSettings()

使用できるバージョン

Flash CS3 Professional

シンタックス

```
document.setMobileSettings(xmlString)
```

パラメータ

xmlString モバイル FLA ファイルの XML 設定を表すストリング。

戻り値

設定が成功した場合は true、それ以外の場合は false を返します。

説明

メソッド。モバイル FLA ファイルの XML 設定ストリングの値を設定します。ほとんどのモバイル FLA ファイルには、ドキュメント内に設定を示す XML ストリングがあります。

例

次の例では、モバイル FLA ファイルの XML 設定ストリングを設定します。この例は、単一のコード行を表しています。

```
fl.getDocumentDOM().setMobileSettings("<? xml version='1.0' encoding='UTF-16' standalone='no' ?> <mobileSettings> <contentType id='standalonePlayer' name='Standalone Player'/> <testDevices> <testDevice id='1170' name='Generic Phone' selected='yes'/> </testDevices> <outputMsgFiltering info='no' trace='yes' warning='yes'/> <testWindowState height='496' splitterClosed='No' splitterXPos='400' width='907'/> </mobileSettings>");
```

関連項目

[document.getMobileSettings\(\)](#)

document.setOvalObjectProperty()

使用できるバージョン

Flash CS3 Professional

シンタックス

```
document.setOvalObjectProperty(propertyName, value)
```

パラメータ

propertyName 設定するプロパティを指定するストリング。指定できる値については、[383 ページの「Oval オブジェクトのプロパティ一覧」](#)を参照してください。

value プロパティに割り当てられる値。指定できる値は、*propertyName* で指定するプロパティごとに異なります。

戻り値

なし。

説明

メソッド。指定したプリミティブ `Oval` オブジェクトのプロパティ値を指定します。

例

[Oval オブジェクト](#) の個々のプロパティの例を参照してください。

関連項目

[Oval オブジェクト](#) , [shape.isOvalObject](#)

document.setPlayerVersion()

使用できるバージョン

Flash CS3 Professional

シンタックス

```
document.setPlayerVersion(version)
```

パラメータ

version 指定したドキュメントの対象となる `Flash Player` のバージョンを表すストリング。指定できる値は、"`FlashLite`"、"`FlashLite11`"、"`FlashLite20`"、"`1`"、"`2`"、"`3`"、"`4`"、"`5`"、"`6`"、"`7`"、"`8`"、および "`9`" です。これらの値は、[`パブリッシュ設定`] ダイアログボックスの [`バージョン`] ドロップダウンリストに対応しています。

戻り値

`Flash Player` バージョンの設定に成功した場合は `true`、それ以外の場合は `false` を返します。

説明

メソッド。指定したドキュメントの対象となる `Flash Player` のバージョンを設定します。この値は、[`パブリッシュ設定`] ダイアログボックスに設定された値と同じです。

例

次の例では、`Flash Lite 1.1` を現在のドキュメントの対象 `Player` バージョンとして指定します。

```
fl.getDocumentDOM().setPlayerVersion("FlashLite11");
```

関連項目

[document.getPlayerVersion\(\)](#)

document.setRectangleObjectProperty()

使用できるバージョン

Flash CS3 Professional

シンタックス

```
document.setRectangleObjectProperty(propertyName, value)
```

パラメータ

propertyName 設定するプロパティを指定するストリング。指定できる値については、[415 ページの「Rectangle オブジェクトのプロパティ一覧」](#)を参照してください。

value プロパティに割り当てられる値。指定できる値は、*propertyName* で指定するプロパティごとに異なります。

戻り値

なし。

説明

メソッド。指定したプリミティブ Rectangle オブジェクトのプロパティ値を指定します。

例

[Rectangle オブジェクト](#) の個々のプロパティの例を参照してください。

関連項目

[Rectangle オブジェクト](#), [shape.isRectangleObject](#)

document.setSelectionBounds()

使用できるバージョン

Flash MX 2004。Flash 8 では、*bContactSensitiveSelection* パラメータが追加されました。

シンタックス

```
document.setSelectionBounds(boundingRectangle [, bContactSensitiveSelection])
```

パラメータ

boundingRectangle 選択範囲の新規の位置とサイズを指定する矩形。*boundingRectangle* の形式の詳細については、[document.addNewRectangle\(\)](#) を参照してください。

bContactSensitiveSelection オブジェクトの選択中、接点を区別する選択モードが有効か (true)、無効か (false) を指定するブール値。デフォルト値は false です。

戻り値

なし

説明

メソッド。選択したオブジェクトを、1回の操作で移動しサイズ変更します。

bContactSensitiveSelection の値を渡すと、その値はこのメソッドのみで有効となり、ドキュメントの接点を区別する選択モードには影響を与えません ([fl.contactSensitiveSelection](#) を参照)。

例

次の例では、現在選択しているオブジェクトを 10、20 の地点に移動し、サイズを 100、200 に変更します。

```
var l = 10;
var t = 20;
fl.getDocumentDOM().setSelectionBounds({left:l, top:t, right:(100+l),
    bottom:(200+t)});
```

関連項目

[document.selection](#), [document.setSelectionRect\(\)](#)

document.setSelectionRect()

使用できるバージョン

Flash MX 2004。Flash 8 では、*bContactSensitiveSelection* パラメータが追加されました。

シンタックス

```
document.setSelectionRect(rect [, bReplaceCurrentSelection
    [, bContactSensitiveSelection]])
```

パラメータ

rect 選択範囲として設定する矩形オブジェクト。*rect* の形式の詳細については、[document.addNewRectangle\(\)](#) を参照してください。

bReplaceCurrentSelection 現在の選択内容を置き換えるか (true)、現在の選択内容に追加するか (false) を指定するブール値。デフォルト値は true です。

bContactSensitiveSelection オブジェクトの選択中、接点を区別する選択モードが有効か (true)、無効か (false) を指定するブール値。デフォルト値は false です。

戻り値

なし

説明

メソッド。ステージを基準にして、指定した座標で矩形の選択マーキーを描画します。編集集中のオブジェクトを基準とする `document.getSelectionRect()` とは異なります。

このメソッドは、選択ツールで矩形をドラッグする操作と同じです。インスタンスを選択するには、矩形で完全に囲む必要があります。

`bContactSensitiveSelection` の値を渡すと、その値はこのメソッドのみで有効となり、ドキュメントの接点を区別する選択モードには影響を与えません ([fl.contactSensitiveSelection](#) を参照)。



[ヒストリ] パネルまたはメニューアイテムを使用して `setSelectionRect()` を繰り返すと、`setSelectionRect()` 操作の前に手順を繰り返します。

例

次の例では、2 番目の選択範囲を 1 番目の選択範囲と置き換えます。

```
fl.getDocumentDOM().setSelectionRect({left:1, top:1, right:200, bottom:200});
fl.getDocumentDOM().setSelectionRect({left:364.0, top:203.0, right:508.0,
    bottom:434.0}, true);
```

次の例では、2 番目の選択範囲を 1 番目の選択範囲に追加します。Shift キーを押しながら 2 番目のオブジェクトを選択する操作と同じです。

```
fl.getDocumentDOM().setSelectionRect({left:1, top:1, right:200, bottom:200});
fl.getDocumentDOM().setSelectionRect({left:364.0, top:203.0, right:508.0,
    bottom:434.0}, false);
```

関連項目

[document.getSelectionRect\(\)](#), [document.selection](#),
[document.setSelectionBounds\(\)](#)

document.setStroke()

使用できるバージョン

Flash MX 2004

シンタックス

`document.setStroke(color, size, strokeType)`

パラメータ

color 線のカラー。次のいずれかの形式です。

- "#RRGGBB" または "#RRGGBBAA" 形式のストリング
- 0xRRGGBB 形式の 16 進数
- 16 進数と等しい 10 進表現の整数

size 選択したオブジェクトの新規の線のサイズを指定する浮動小数値。

strokeType 選択したオブジェクトの新規の線のタイプを指定するストリング。指定できる値は、`"hairline"`、`"solid"`、`"dashed"`、`"dotted"`、`"ragged"`、`"stipple"`、および `"hatched"` です。

戻り値

なし

説明

メソッド。選択した線のカラー、幅、スタイルを設定します。ツールパネルおよびプロパティインスペクタの線を変更する方法については、[document.setStrokeColor\(\)](#) を参照してください。

例

次の例では、線のカラーに赤色を、サイズに 3.25 を、タイプに破線を設定します。

```
fl.getDocumentDOM().setStroke("#ff0000", 3.25, "dashed");
```

document.setStrokeColor()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.setStrokeColor(color)
```

パラメータ

color 線のカラー。次のいずれかの形式です。

- `"#RRGGBB"` または `"#RRGGBBAA"` 形式のストリング
- `0xRRGGBB` 形式の 16 進数
- 16 進数と等しい 10 進表現の整数

戻り値

なし

説明

メソッド。選択したオブジェクトの線のカラーを指定カラーに変更します。ツールパネルおよびプロパティインスペクタの線を変更する方法については、[document.setStrokeColor\(\)](#) を参照してください。

例

次の例の 3 つのステートメントでは、それぞれ別のカラー指定形式を使用して線のカラーを設定します。

```
flash.getDocumentDOM().setStrokeColor("#cc00cc");  
flash.getDocumentDOM().setStrokeColor(0xcc00cc);  
flash.getDocumentDOM().setStrokeColor(120000);
```

document.setStrokeSize()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.setStrokeSize(size)
```

パラメータ

size 線のサイズを指定する浮動小数値。0.25 から 10 までの値。小数点以下 3 桁以上の数値は無視されます。

戻り値

なし。

説明

メソッド。選択したオブジェクトの線のサイズを指定したサイズに変更します。ツールパネルおよびプロパティインスペクタの線を変更する方法については、[document.setCustomStroke\(\)](#) を参照してください。

例

次の例では、選択したオブジェクトの線のサイズを 5 に変更します。

```
fl.getDocumentDOM().setStrokeSize(5);
```

document.setStrokeStyle()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.setStrokeStyle(strokeType)
```

パラメータ

strokeType 現在選択しているオブジェクトの線のスタイルを指定するストリング。指定できる値は、"hairline"、"solid"、"dashed"、"dotted"、"ragged"、"stipple"、および "hatched" です。

戻り値

なし

説明

メソッド。選択したオブジェクトの線のスタイルを指定スタイルに変更します。ツールパネルおよびプロパティインスペクタの線を変更する方法については、[document.setCustomStroke\(\)](#) を参照してください。

例

次の例では、選択したオブジェクトの線のスタイルを "dashed" に変更します。

```
fl.getDocumentDOM().setStrokeStyle("dashed");
```

document.setTextRectangle()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.setTextRectangle(boundingRectangle)
```

パラメータ

boundingRectangle テキストアイテムのフロー内で使用する新規サイズを指定するテキスト矩形オブジェクト。*boundingRectangle* の形式の詳細については、[document.addNewRectangle\(\)](#) を参照してください。

戻り値

ブール値。テキストフィールドのサイズが1つでも変更されている場合は true を返し、それ以外の場合は false を返します。

説明

メソッド。選択したテキストアイテムの境界の矩形を指定したサイズに変更します。新規矩形内でテキストを再フローします。テキストアイテムは、拡大 / 縮小や変形はされません。*boundingRectangle* に渡した値は、次のように使用されます。

- 横書きの静止テキストの場合、*boundingRectangle* に渡した幅だけが考慮されます。高さは、すべてのテキストが収まるように自動的に計算されます。
- 縦書きのテキスト (したがって、静止テキスト) の場合、*boundingRectangle* に渡した高さだけが考慮されます。幅は、すべてのテキストが収まるように自動的に計算されます。

- ダイナミックテキストまたは入力テキストの場合、*boundingRectangle* に渡した幅と高さの両方が考慮され、矩形は、すべてのテキストを収容するために必要なサイズよりも大きくなる場合があります。ただし、パラメータで指定した矩形のサイズが、すべてのテキストを収容するには小さすぎる場合、*boundingRectangle* に渡した幅だけが考慮されます (高さは、すべてのテキストが収まるように自動的に計算されます)。

例

次の例では、境界テキスト矩形のサイズを指定サイズに変更します。

```
fl.getDocumentDOM().setTextRectangle({left:0, top:0, right:50, bottom:200})
```

document.setTextSelection()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.setTextSelection(startIndex, endIndex)
```

パラメータ

startIndex 選択する最初の文字の位置を指定する整数。先頭の文字位置は 0 になります。

endIndex 選択範囲の最終位置を指定する整数。*endIndex* は含みません。先頭の文字位置は 0 になります。

戻り値

ブール値。テキスト選択が正常に設定された場合は true を返し、それ以外の場合は false を返します。

説明

メソッド。現在選択しているテキストフィールドのテキストに、*startIndex* 値と *endIndex* 値で指定される値を設定します。テキスト編集がアクティブになります。

例

次の例では、6 文字目から 25 文字目までのテキストを選択します。

```
fl.document.setTextSelection(5, 25);
```

document.setTextString()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.setTextString(text [, startIndex [, endIndex]])
```

パラメータ

text テキストフィールドに挿入する文字のストリング。

startIndex 置換する最初の文字を指定する整数。先頭の文字位置は 0 になります。このパラメータはオプションです。

endIndex 置換する最後の文字を指定する整数。このパラメータはオプションです。

戻り値

ブール値。テキストストリングのテキストが1つでも設定されている場合は true、それ以外の場合は false を返します。

説明

メソッド。テキストのストリングを挿入します。オプションのパラメータを渡していない場合は、既存のテキスト範囲が置換されます。テキストオブジェクトが現在編集集中でない場合は、テキストストリング全体が置換されます。*startIndex* だけを渡している場合は、指定したストリングがこの位置に挿入されます。*startIndex* と *endIndex* を渡している場合、*startIndex* から *endIndex* の前の文字までのテキストセグメントが、指定したストリングで置換されます。

例

次の例では、現在のテキスト選択範囲を "Hello World" で置換します。

```
var success = fl.getDocumentDOM().setTextString("Hello World!");
```

次の例では、現在のテキスト範囲の 6 文字目に "hello" を挿入します。

```
var pass = fl.getDocumentDOM().setTextString("hello", 6);
```

次の例では、現在のテキスト範囲の 2 文字目から 6 文字目までに "Howdy" を挿入します。

```
var ok = fl.getDocumentDOM().setTextString("Howdy", 2, 7);
```

関連項目

[document.getTextString\(\)](#)

document.setTransformationPoint()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.setTransformationPoint( transformationPoint )
```

パラメータ

transformationPoint {x:10, y:20} などの点 (x と y は浮動小数点数)。次の各エレメントの変形点の値を指定します。

- シェイプ: *transformationPoint* は、ドキュメントを基準に設定されます (0,0 がステージの左上隅になる)。
- シンボル: *transformationPoint* は、シンボルの基準点を基準に設定されます (0,0 が基準点の位置になる)。
- テキスト: *transformationPoint* は、テキストフィールドを基準に設定されます (0,0 がテキストフィールドの左上隅になる)。
- ビットマップとビデオ: *transformationPoint* は、ビットマップまたはビデオを基準に設定されます (0,0 がビットマップまたはビデオの左上隅になる)。
- 描画オブジェクト、プリミティブな楕円と矩形、およびグループ: *transformationPoint* は、ドキュメントを基準に設定されます (0,0 がステージの左上隅になる)。オブジェクト、プリミティブ、またはグループの中心点を基準に *transformationPoint* を設定するには、[element.setTransformationPoint\(\)](#) を使用します。

戻り値

なし

説明

メソッド。現在の選択範囲の変形点の位置を設定します。

例

次の例では、現在の選択範囲の変形点を 100,200 に設定します。

```
fl.getDocumentDOM().setTransformationPoint({x:100, y:200});
```

関連項目

[document.getTransformationPoint\(\)](#), [element.setTransformationPoint\(\)](#)

document.silent

使用できるバージョン

Flash MX 2004

シンタックス

`document.silent`

説明

プロパティ。オブジェクトがアクセス可能かどうかを指定するブール値です。[アクセシビリティ] パネルの [ムービーをアクセス可能にする] の逆ロジックに等しくなります。`document.silent` が `true` の場合、[ムービーをアクセス可能にする] をオフにした場合と同じになります。`false` の場合、[ムービーをアクセス可能にする] をオンにした場合と同じになります。

例

次の例では、`isSilent` 変数に `silent` プロパティの値を設定します。

```
var isSilent = fl.getDocumentDOM().silent;
```

次の例では、`silent` プロパティに `false` を設定し、ドキュメントをアクセス可能にします。

```
fl.getDocumentDOM().silent = false;
```

document.skewSelection()

使用できるバージョン

Flash MX 2004

シンタックス

`document.skewSelection(xSkew, ySkew [, whichEdge])`

パラメータ

xSkew 傾斜する *x* の量を度単位で指定する浮動小数値。

ySkew 傾斜する *y* の量を度単位で指定する浮動小数値。

whichEdge 変形の発生するエッジを指定するストリング。省略すると、変形点で傾斜が発生します。指定できる値は、"top center"、"right center"、"bottom center"、および "left center" です。このパラメータはオプションです。

戻り値

なし

説明

メソッド。選択したオブジェクトを指定した角度だけ傾斜します。自由変形ツールを使用してオブジェクトを傾斜する操作と同じです。

例

次の例では、選択したオブジェクトを垂直方向に 2.0、水平方向に 1.5 傾斜します。2 番目の例では、上部中央のエッジでオブジェクトを変形します。

```
flash.getDocumentDOM().skewSelection(2.0, 1.5);  
flash.getDocumentDOM().skewSelection(2.0, 1.5, "top center");
```

document.smoothSelection()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.smoothSelection()
```

パラメータ

なし

戻り値

なし

説明

メソッド。選択した塗りのアウトラインまたは曲線を滑らかにします。このメソッドは、ツールパネルの [スムーズ] ボタンと同じ操作を実行します。

例

次の例では、現在の選択範囲の曲線を滑らかにします。

```
fl.getDocumentDOM().smoothSelection();
```

document.space()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.space(direction [, bUseDocumentBounds])
```

パラメータ

direction 選択範囲内のオブジェクトを均等に配置する方向を指定するストリング。指定できる値は、"horizontal" または "vertical" です。

bUseDocumentBounds ブール値。true を設定すると、オブジェクトをドキュメント境界に合わせて均等に配置します。それ以外の場合は、選択したオブジェクトの境界を使用します。デフォルト値は false です。このパラメータはオプションです。

戻り値

なし

説明

メソッド。選択したオブジェクトを均等な間隔で配置します。

例

次の例では、ステージを基準にして、オブジェクトを水平方向に均等に配置します。

```
fl.getDocumentDOM().space("horizontal",true);
```

次の例では、各オブジェクトを基準にして、オブジェクトを水平方向に均等に配置します。

```
fl.getDocumentDOM().space("horizontal");
```

次の例では、*bUseDocumentBounds* に明示的に false を設定して、各オブジェクトを基準にして、オブジェクトを水平方向に均等に配置します。

```
fl.getDocumentDOM().space("horizontal",false);
```

関連項目

[document.getAlignToDocument\(\)](#), [document.setAlignToDocument\(\)](#)

document.straightenSelection()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.straightenSelection()
```

パラメータ

なし

戻り値

なし

説明

メソッド。現在選択している線を直線にします。ツールパネルの [ストレート] ボタンを使用する操作と同じです。

例

次の例では、現在の選択範囲の曲線を直線にします。

```
fl.getDocumentDOM().straightenSelection();
```

document.swapElement()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.swapElement(name)
```

パラメータ

name 使用するライブラリアイテムの名前を指定するストリング。

戻り値

なし

説明

メソッド。現在選択しているオブジェクトを、指定したオブジェクトと入れ替えます。選択範囲には、グラフィック、ボタン、ムービークリップ、ビデオ、またはビットマップを含んでいる必要があります。オブジェクトを選択していない場合、または指定したオブジェクトが見つからない場合は、エラーメッセージを表示します。

例

次の例では、現在選択しているオブジェクトと、ライブラリの Symbol 1 を入れ替えます。

```
fl.getDocumentDOM().swapElement('Symbol 1');
```

document.swapStrokeAndFill()

使用できるバージョン

Flash 8

シンタックス

```
document.swapStrokeAndFill();
```


パラメータ

なし

戻り値

なし

説明

メソッド。線と塗りのカラーを入れ替えます。

例

次の例では、現在のドキュメントの線と塗りのカラーを入れ替えます。

```
fl.getDocumentDOM().swapStrokeAndFill();
```

document.synchronizeWithHeadVersion()

使用できるバージョン

Flash CS3 Professional

シンタックス

```
fl.document.getDocumentDOM().synchronizeWithHeadVersion().swapStrokeAndFill();
```

パラメータ

なし

戻り値

ブール値。指定したファイルが Version Cue サーバーとの同期に成功した場合は true、それ以外の場合は false を返します。

説明

メソッド。指定したドキュメントを Version Cue サーバーにある最新バージョンと同期し、[出力] パネルにエラーをログします。

このメソッドを使用できるのは、現在開いているドキュメントだけです。現在開いていないファイルの最新バージョンを取得するには、[fl.downloadLatestVersion\(\)](#) を使用します。

例

次の例では、現在のドキュメントを Version Cue サーバーのバージョンと同期します。

```
fl.getDocumentDOM().synchronizeWithHeadVersion();
```

関連項目

[document.canSaveAVersion\(\)](#), [fl.downloadLatestVersion\(\)](#),
[document.revertToLastVersion\(\)](#), [document.saveAVersion\(\)](#),
[fl.synchronizeDocumentWithHeadVersion\(\)](#)

document.testMovie()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.testMovie()
```

パラメータ

なし

戻り値

なし

説明

メソッド。ドキュメントで [ムービープレビュー] を実行します。

例

次の例では、現在のドキュメントのムービーをテストします。

```
fl.getDocumentDOM().testMovie();
```

関連項目

[document.canTestMovie\(\)](#), [document.testScene\(\)](#)

document.testScene()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.testScene()
```

パラメータ

なし

戻り値

なし

説明

メソッド。ドキュメントの現在のシーンで [シーンプレビュー] を実行します。

例

次の例では、ドキュメントの現在のシーンをテストします。

```
fl.getDocumentDOM().testScene();
```

関連項目

[document.canTestScene\(\)](#), [document.testMovie\(\)](#)

document.timelines

使用できるバージョン

Flash MX 2004

シンタックス

```
document.timelines
```

説明

読み取り専用プロパティ。Timeline オブジェクトの配列です ([Timeline オブジェクト](#) を参照)。

例

次の例では、アクティブドキュメントの現在のタイムラインの配列を取得し、その名前を [出力] パネルに表示します。

```
var i = 0;
var curTimelines = fl.getDocumentDOM().timelines;
while(i < fl.getDocumentDOM().timelines.length){
    alert(curTimelines[i].name);
    ++i;
}
```

関連項目

[document.currentTimeline](#), [document.getTimeline\(\)](#)

document.traceBitmap()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.traceBitmap(threshold, minimumArea, curveFit, cornerThreshold)
```

パラメータ

threshold トレースしたビットマップの色数を制御する整数。指定できる値は 0～500 の整数です。

minimumArea 半径をピクセル単位で指定する整数。指定できる値は 1～1000 の整数です。

curveFit アウトラインの滑らかさを指定するストリング。指定できる値は、"pixels"、"very tight"、"tight"、"normal"、"smooth"、および "very smooth" です。

cornerThreshold *curveFit* と同じストリング。ただし、ビットマップイメージの角に属するものに限りです。指定できる値は、"many corners"、"normal"、および "few corners" です。

戻り値

なし

説明

メソッド。現在選択しているオブジェクトでビットマップのトレースを実行します。[修正]-[ビットマップ]-[ビットマップのトレース]を選択する操作と同じです。

例

次の例では、指定したパラメータを使用して、選択したビットマップをトレースします。

```
fl.getDocumentDOM().traceBitmap(0, 500, 'normal', 'normal');
```

document.transformSelection()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.transformSelection(a, b, c, d)
```

パラメータ

a 変形マトリックスの (0,0) エlementを指定する浮動小数値。

b 変形マトリックスの (0,1) エlementを指定する浮動小数値。

c 変形マトリックスの (1,0) エlementを指定する浮動小数値。

d 変形マトリックスの (1,1) エlementを指定する浮動小数値。

戻り値

なし

説明

メソッド。現在選択しているオブジェクトで、パラメータで指定したマトリックスを適用して、一般変形を実行します。詳細については、[element.matrix](#) プロパティを参照してください。

例

次の例では、選択したオブジェクトを x 方向に 2 だけ伸縮します。

```
fl.getDocumentDOM().transformSelection(2.0, 0.0, 0.0, 1.0);
```

document.unGroup()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.unGroup()
```

パラメータ

なし

戻り値

なし

説明

メソッド。現在選択しているオブジェクトのグループを解除します。

例

次の例では、現在の選択範囲にあるエレメントのグループを解除します。

```
fl.getDocumentDOM().unGroup();
```

関連項目

[document.group\(\)](#)

document.union()

使用できるバージョン

Flash 8

シンタックス

```
document.union()
```

パラメータ

なし

戻り値

ブール値。成功した場合は true を返します。それ以外の場合は false を返します。

説明

メソッド。選択したすべてのシェイプを描画オブジェクトに結合します。

例

次の例では、選択したすべてのシェイプを描画オブジェクトに結合します。

```
fl.getDocumentDOM().union();
```

関連項目

[document.crop\(\)](#), [document.deleteEnvelope\(\)](#), [document.intersect\(\)](#),
[document.punch\(\)](#), [shape.isDrawingObject](#)

document.unlockAllElements()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.unlockAllElements()
```

パラメータ

なし

戻り値

なし。

説明

メソッド。現在選択しているフレームでロックされているすべてのエレメントをロック解除します。

例

次の例では、現在のフレームでロックされているすべてのオブジェクトのロックを解除します。

```
fl.getDocumentDOM().unlockAllElements();
```

関連項目

[element.locked](#)

document.viewMatrix

使用できるバージョン

Flash MX 2004

シンタックス

`document.viewMatrix`

説明

読み取り専用プロパティ。**Matrix** オブジェクトです。ドキュメントが編集モードである場合、オブジェクトスペースをドキュメントスペースに変形するときに `viewMatrix` を使用します。ツールに送られるマウスの位置は、現在編集中のオブジェクトを基準とした位置になります。詳細については、[Matrix オブジェクト](#) を参照してください。

たとえば、シンボルを作成し、ダブルクリックして編集し、多角形ツールで描画すると、点 (0,0) がシンボルの基準点となります。ただし、`drawingLayer` オブジェクトではドキュメントスペースの値と見なしているので、`drawingLayer` を使用して (0,0) から線を描画すると、ステージの左上隅が開始点になります。`viewMatrix` では、編集中のオブジェクトスペースをドキュメントスペースに変換します。

例

次の例では、`viewMatrix` プロパティの値を取得します。

```
var mat = fl.getDocumentDOM().viewMatrix;
```

document.width

使用できるバージョン

Flash MX 2004

シンタックス

`document.width`

説明

プロパティ。ドキュメント (ステージ) の幅をピクセル単位で指定する整数。

例

次の例では、ステージの幅を 400 ピクセルに設定します。

```
fl.getDocumentDOM().width= 400;
```

関連項目

[document.height](#)

document.xmlPanel()

使用できるバージョン

Flash MX 2004

シンタックス

```
document.xmlPanel(fileURI)
```

パラメータ

fileURI パネルのコントロールを定義する XML ファイルのパスを指定するストリング。
file:/// URI で表します。完全パスを指定する必要があります。

戻り値

XML ファイルで定義されるすべてのコントロールを定義するプロパティを持つオブジェクト。プロパティはすべて、ストリングとして返されます。返されたオブジェクトには、あらかじめ定義された "dismiss" プロパティが1つあり、そのストリング値は "accept" または "cancel" です。

説明

メソッド。[\[XMLUI\]](#) ダイアログボックスを表示します。詳細については、[fl.xmlui](#) を参照してください。

例

次の例では、"Test.xml" ファイルをロードし、そのファイルにある各プロパティを表示します。

```
var obj = fl.getDocumentDOM().xmlPanel(fl.configURI + "Commands/Test.xml");  
for (var prop in obj) {  
    fl.trace("property " + prop + " = " + obj[prop]);  
}
```


document.zoomFactor

使用できるバージョン

Flash 8

シンタックス

```
document.zoomFactor
```

説明

プロパティ。オーサリング時にステージのズーム率を指定します。1 は 100% ズーム、8 は 800% ズーム、.5 は 50% ズームになります。

例

次の例では、ステージのズーム率を 200% に設定します。

```
fl.getDocumentDOM().zoomFactor = 2;
```

adrawingLayer オブジェクト

使用できるバージョン

Flash MX 2004

説明

drawingLayer オブジェクトは、JavaScript から Flash オブジェクトの子としてアクセスできます。drawingLayer オブジェクトは、選択マーキーを作成するときなど、ユーザーがドラッグ中に一時的に描画する場合に拡張ツールで使います。他の drawingLayer メソッドを呼び出す前に、[drawingLayer.beginFrame\(\)](#) を呼び出す必要があります。

drawingLayer オブジェクトのメソッド一覧

drawingLayer オブジェクトでは次のメソッドを使用できます。

メソッド	説明
drawingLayer.beginDraw()	Flash を描画モードにします。
drawingLayer.beginFrame()	drawingLayer で以前に描画されたものを消去し、さらに多くの描画コマンドを使用できるよう準備します。
drawingLayer.cubicCurveTo()	パラメータを三次セグメントの座標として使用し、現在のペンの位置から三次曲線を描画します。
drawingLayer.curveTo()	現在の描画位置から指定位置まで、二次曲線のセグメントを描画します。
drawingLayer.drawPath()	指定したパスを描画します。
drawingLayer.endDraw()	描画モードを終了します。
drawingLayer.endFrame()	描画コマンドのグループの終了を通知します。
drawingLayer.lineTo()	現在の描画位置から (x,y) 地点まで線を描画します。
drawingLayer.moveTo()	現在の描画位置を設定します。
drawingLayer.newPath()	新しい Path オブジェクト を返します。
drawingLayer.setColor()	以降に描画するデータの色を設定します。

drawingLayer.beginDraw()

使用できるバージョン

Flash MX 2004

シンタックス

```
drawingLayer.beginDraw([persistentDraw])
```

パラメータ

persistentDraw ブール値 (オプション)。true に設定すると、beginDraw() または beginFrame() を新しく呼び出すまで、最後のフレームの描画がステージに残ります。ここでのフレームとは、描画を開始し終了する場所を指します。タイムラインフレームではありません。たとえば、ユーザーが矩形を描画する際に、マウスをドラッグしながらシェイプのアウトラインをプレビューすることができます。マウスボタンを離れた後でも、そのプレビューシェイプを残しておく場合は、*persistentDraw* を true に設定します。

戻り値

なし

説明

メソッド。Flash を描画モードにします。描画モードは、マウスボタンを押しているときの一時的な描画に使用します。通常、このメソッドは、拡張ツールを作成する場合にのみ使用します。

例

次の例では、Flash を描画モードにします。

```
fl.drawingLayer.beginDraw();
```

drawingLayer.beginFrame()

使用できるバージョン

Flash MX 2004

シンタックス

```
drawingLayer.beginFrame()
```

パラメータ

なし

戻り値

なし。

説明

メソッド。drawingLayer で以前に描画されたものを消去し、さらに多くの描画コマンドを使用できるよう準備します。drawingLayer.beginDraw() の後に呼び出します。drawingLayer.beginFrame() と drawingLayer.endFrame() の間で描画されたものはすべて、次回 beginFrame() と endFrame() を呼び出すまで、ステージに残ります。ここでのフレームとは、描画を開始し終了する場所を指します。タイムラインフレームではありません。通常、このメソッドは、拡張ツールを作成する場合にのみ使用します。詳細については、[drawingLayer.beginDraw\(\)](#) を参照してください。

drawingLayer.cubicCurveTo()

使用できるバージョン

Flash MX 2004

シンタックス

```
drawingLayer.cubicCurveTo(x1Ctrl, y1Ctrl, x2Ctrl, y2Ctrl, xEnd, yEnd)
```

パラメータ

x1Ctrl 最初のコントロールポイントの x 位置を示す浮動小数値。

y1Ctrl 最初のコントロールポイントの y 位置を示す浮動小数値。

x2Ctrl 中央のコントロールポイントの x 位置を示す浮動小数値。

y2Ctrl 中央のコントロールポイントの y 位置を示す浮動小数値。

xEnd 最後のコントロールポイントの x 位置を示す浮動小数値。

yEnd 最後のコントロールポイントの y 位置を示す浮動小数値。

戻り値

なし

説明

メソッド。パラメータを三次セグメントの座標として使用し、現在のペンの位置から三次曲線を描画します。通常、このメソッドは、拡張ツールを作成する場合にのみ使用します。

例

次の例では、指定したコントロールポイントを使用して三次曲線を描画します。

```
fl.drawingLayer.cubicCurveTo(0, 0, 1, 1, 2, 0);
```

drawingLayer.curveTo()

使用できるバージョン

Flash MX 2004

シンタックス

```
drawingLayer.curveTo(xCtl, yCtl, xEnd, yEnd)
```

パラメータ

xCtl コントロールポイントの *x* 位置を示す浮動小数値。

yCtl コントロールポイントの *y* 位置を示す浮動小数値。

xEnd 最後のコントロールポイントの *x* 位置を示す浮動小数値。

yEnd 最後のコントロールポイントの *y* 位置を示す浮動小数値。

戻り値

なし

説明

メソッド。現在の描画位置から指定位置まで、二次曲線のセグメントを描画します。通常、このメソッドは、拡張ツールを作成する場合にのみ使用します。

例

次の例では、指定したコントロールポイントを使用して二次曲線を描画します。

```
fl.drawingLayer.curveTo(0, 0, 2, 0);
```

drawingLayer.drawPath()

使用できるバージョン

Flash MX 2004

シンタックス

```
drawingLayer.drawPath(path)
```

パラメータ

path 描画対象の [Path オブジェクト](#)

戻り値

なし

説明

メソッド。 *path* パラメータで指定したパスを描画します。通常、このメソッドは、拡張ツールを作成する場合にのみ使用します。

例

次の例では、 *gamePath* という名前の **Path** オブジェクトで指定したパスを描画します。

```
fl.drawingLayer.drawPath(gamePath);
```

drawingLayer.endDraw()

使用できるバージョン

Flash MX 2004

シンタックス

```
drawingLayer.endDraw()
```

パラメータ

なし

戻り値

なし

説明

メソッド。描画モードを終了します。描画モードは、マウスボタンを押しているとき、一時的に描画する際に使用します。通常、このメソッドは、拡張ツールを作成する場合にのみ使用します。

例

次の例では、描画モードを終了します。

```
fl.drawingLayer.endDraw();
```

drawingLayer.endFrame()

使用できるバージョン

Flash MX 2004

シンタックス

```
drawingLayer.endFrame()
```

パラメータ

なし

戻り値

なし

説明

メソッド。描画コマンドのグループの終了を通知します。描画コマンドのグループとは、`drawingLayer.beginFrame()` と `drawingLayer.endFrame()` の間に描画されたすべてのものを指します。次回、`drawingLayer.beginFrame()` を呼び出すと、このグループの描画コマンドで描画したものはすべて消去されます。通常、このメソッドは、拡張ツールを作成する場合にのみ使用します。

drawingLayer.lineTo()

使用できるバージョン

Flash MX 2004

シンタックス

```
drawingLayer.lineTo(x, y)
```

パラメータ

x 描画する線の終点の *x* 座標を表す浮動小数値。

y 描画する線の終点の *y* 座標を表す浮動小数値。

戻り値

なし

説明

メソッド。現在の描画位置から (*x*,*y*) 地点まで線を描画します。通常、このメソッドは、拡張ツールを作成する場合にのみ使用します。

例

次の例では、現在の描画位置から (20,30) 地点まで線を描画します。

```
fl.drawingLayer.lineTo(20, 30);
```

drawingLayer.moveTo()

使用できるバージョン

Flash MX 2004

シンタックス

```
drawingLayer.moveTo(x, y)
```

パラメータ

x 描画の始点の *x* 座標を表す浮動小数値。

y 描画の始点の *y* 座標を表す浮動小数値。

戻り値

なし

説明

メソッド。現在の描画位置を設定します。通常、このメソッドは、拡張ツールを作成する場合にのみ使用します。

例

次の例では、現在の描画位置を (10,15) 地点に設定します。

```
fl.drawingLayer.moveTo(10, 15);
```

drawingLayer.newPath()

使用できるバージョン

Flash MX 2004

シンタックス

```
drawingLayer.newPath()
```

パラメータ

なし

戻り値

Path オブジェクト

説明

メソッド。新しい Path オブジェクトを返します。通常、このメソッドは、拡張ツールを作成する場合にのみ使用します。[Path オブジェクト](#) を参照してください。

例

次の例では、新しい Path オブジェクトを返します。

```
fl.drawingLayer.newPath();
```


drawingLayer.setColor()

使用できるバージョン

Flash MX 2004

シンタックス

`drawingLayer.setColor(color)`

パラメータ

color 以降に描画するデータの色。次のいずれかの形式です。

- "#RRGGBB" または "#RRGGBBAA" 形式のストリング
- 0xRRGGBB 形式の 16 進数
- 16 進数と等しい 10 進表現の整数

戻り値

なし。

説明

メソッド。以降に描画するデータの色を設定します。永続データのみに適用されます。このメソッドを使用するには、`drawingLayer.beginDraw()` に渡すパラメータを `true` に設定しておく必要があります。通常、このメソッドは、拡張ツールを作成する場合にのみ使用します。詳細については、[drawingLayer.beginDraw\(\)](#) を参照してください。

例

次の例では、ステージ上に赤い線を描画します。

```
f1.drawingLayer.beginDraw( true );
f1.drawingLayer.beginFrame();
f1.drawingLayer.setColor( "#ff0000" );
f1.drawingLayer.moveTo(0,0);
f1.drawingLayer.lineTo(100,100);
f1.drawingLayer.endFrame();
f1.drawingLayer.endDraw();
```

Edge オブジェクト

使用できるバージョン

Flash MX 2004

説明

Edge オブジェクトは、ステージ上のシェイプのエッジを表します。

Edge オブジェクトのメソッド一覧

Edge オブジェクトでは次のメソッドを使用できます。

メソッド	説明
<code>edge.getControl()</code>	エッジの指定されたコントロールポイントの位置に設定されているポイントオブジェクトを取得します。
<code>edge.getHalfEdge()</code>	HalfEdge オブジェクト を返します。
<code>edge.setControl()</code>	エッジのコントロールポイントの位置を設定します。
<code>edge.splitEdge()</code>	エッジを 2 つに分割します。

Edge オブジェクトのプロパティ一覧

Edge オブジェクトでは次のプロパティを使用できます。

プロパティ	説明
<code>edge.id</code>	読み取り専用。エッジの固有の識別子を表す整数です。
<code>edge.isLine</code>	読み取り専用。0 または 1 の整数です。

edge.getControl()

使用できるバージョン

Flash MX 2004

シンタックス

`edge.getControl(i)`

パラメータ

i 返すエッジのコントロールポイントを指定する整数。最初のコントロールポイントには 0 を指定し、中央のコントロールポイントには 1 を指定し、最後のコントロールポイントには 2 を指定します。
`edge.isLine` プロパティが `true` の場合、中央のコントロールポイントは、最初と最後のコントロールポイントを結合するセグメントの中間点に設定されます。

戻り値

指定したコントロールポイント

説明

メソッド。エッジの指定されたコントロールポイントの位置に設定されているポイントオブジェクトを取得します。

例

次の例では、指定したシェイプの最初のコントロールポイントを `pt` 変数に格納します。

```
var shape = fl.getDocumentDOM().selection[0];  
var pt = shape.edges[0].getControl(0);
```

edge.getHalfEdge()

使用できるバージョン

Flash MX 2004

シンタックス

`edge.getHalfEdge(index)`

パラメータ

index 返すハーフエッジを指定する整数。*index* の値には、最初のハーフエッジの場合は 0、2 番目のハーフエッジの場合は 1 を指定します。

戻り値

HalfEdge オブジェクト

説明

メソッド。[HalfEdge オブジェクト](#) を返します。

例

次の例では、指定したエッジのハーフエッジを hEdge0 変数と hEdge1 変数に格納します。

```
var shape = fl.getDocumentDOM().selection[0];
var edge = shape.edges[0];
var hEdge0 = edge.getHalfEdge(0);
var hEdge1 = edge.getHalfEdge(1);
```

edge.id

使用できるバージョン

Flash MX 2004

シンタックス

edge.id

説明

読み取り専用プロパティ。エッジの固有の識別子を表す整数です。

例

次の例では、指定したエッジの固有の識別子を my_shape_id 変数に格納します。

```
var shape = fl.getDocumentDOM().selection[0];
var my_shape_id = shape.edges[0].id;
```

edge.isLine

使用できるバージョン

Flash MX 2004

シンタックス

edge.isLine

説明

読み取り専用プロパティ。0 または 1 の整数です。1 の場合は、エッジが直線であることを示します。この場合、中間のコントロールポイントで、2 つの端点をつなぐ線が 2 等分されます。

例

次の例では、指定したエッジが直線であるかどうかを判断し、[出力] パネルに 1 (直線の場合) または 0 (直線でない場合) を表示します。

```
var shape = fl.getDocumentDOM().selection[0];
fl.trace(shape.edges[0].isLine);
```

edge.setControl()

使用できるバージョン

Flash MX 2004

シンタックス

```
edge.setControl(index, x, y)
```

パラメータ

index 設定するコントロールポイントを指定する整数。値 0、1、または 2 を使用して、最初、中間、最後の各コントロールポイントを指定します。

x コントロールポイントの水平位置を指定する浮動小数値。ステージが編集モードまたは同じ位置で編集モードの場合、ポイントの座標は、編集したオブジェクトを基準にした値になります。その他のモードでは、ポイントの座標はステージを基準にした値になります。

y コントロールポイントの垂直位置を指定する浮動小数値。ステージが編集モードまたは同じ位置で編集モードの場合、ポイントの座標は、編集したオブジェクトを基準にした値になります。その他のモードでは、ポイントの座標はステージを基準にした値になります。

戻り値

なし

説明

メソッド。エッジのコントロールポイントの位置を設定します。このメソッドを使用する前に、`shape.beginEdit()` を呼び出す必要があります。 [shape.beginEdit\(\)](#) を参照してください。

例

次の例では、指定したエッジの最初のコントロールポイントを (0,1) 座標に設定します。

```
x = 0; y = 1;
var shape = fl.getDocumentDOM().selection[0];
shape.beginEdit();
shape.edges[0].setControl(0, x, y);
shape.endEdit();
```

edge.splitEdge()

使用できるバージョン

Flash MX 2004

シンタックス

```
edge.splitEdge(t)
```

パラメータ

t エッジを分割する位置を指定する 0～1 の範囲の浮動小数値。0 は一方の端点を表し、1 はもう一方の端点を表します。たとえば、0.5 を指定すると、エッジが中央で分割され、直線の場合は 2 等分されます。曲線の場合は、0.5 を指定すると、曲線の中点になります。

戻り値

なし

説明

メソッド。エッジを 2 つに分割します。このメソッドを使用する前に、`shape.beginEdit()` を呼び出す必要があります。

例

次の例では、指定したエッジを 2 つに分割します。

```
var shape = fl.getDocumentDOM().selection[0];
shape.beginEdit()
shape.edges[0].splitEdge( 0.5 );
shape.endEdit()
```

Effect オブジェクト

使用できるバージョン

Flash MX 2004

説明

単独のエフェクト記述子オブジェクトです。`fl.activeEffect` プロパティと `fl.effects` プロパティには、この種類のオブジェクトが含まれています。Effect オブジェクトは、タイムラインエフェクトのインスタンスを表します。[fl.activeEffect](#) および [fl.effects](#) を参照してください。

Effect オブジェクトのプロパティ一覧

Effect オブジェクトには、次の表に示すプロパティの他に、ユーザー定義パラメータを使用することができます。ユーザー定義パラメータは、[effect.effectName](#) プロパティと [effect.sourceFile](#) プロパティを指定する XML ファイルと同じファイルで指定する必要があります。これらのパラメータでは、作成するユーザーインターフェイスエレメント (編集フィールド、チェックボックス、リストボックスなど) や、作成しているエフェクトの制御対象を指定します。デフォルト値だけでなく、コントロールに表示するラベルを指定することもできます。

プロパティ	説明
effect.effectName	読み取り専用。エフェクトのコンテキストメニューに表示されるストリングです。
effect.groupName	読み取り専用。エフェクトの階層化コンテキストメニューに使用するエフェクトグループの名前を表すストリングです。
effect.sourceFile	読み取り専用。指定したエフェクトの JSFL ソースファイル名を指定するストリングです。
effect.symbolType	読み取り専用。エフェクトの最初の適用時に作成するシンボルの種類を指定するストリングです。
effect.useXMLToUI	1つ以上のコントロールから構成されるダイアログボックスを、XMLUI を使用して作成する際のデフォルトの動作を無効にするブール値。

effect.effectName

使用できるバージョン

Flash MX 2004

シンタックス

`effect.effectName`

説明

読み取り専用プロパティ。エフェクトのコンテキストメニューに表示されるストリングです。各エフェクトの名前は一意である必要があります。

例

次の例では、現在のエフェクトの名前を `efName` 変数に格納します。

```
var efName = fl.activeEffect.effectName;
```

effect.groupName

使用できるバージョン

Flash MX 2004

シンタックス

`effect.groupName`

説明

読み取り専用プロパティ。エフェクトの階層化コンテキストメニューに使用するエフェクトグループの名前を表すストリングです。空のストリングである場合、エフェクトはグループ解除されて、コンテキストメニューの最上レベルに表示されます。グループ名とエフェクト名は、そのエフェクトのXML ファイルで指定されます。

例

次の例では、現在のエフェクトのグループ名を `efGroupName` 変数に格納します。

```
var efGroupName = fl.activeEffect.groupName;
```


effect.sourceFile

使用できるバージョン

Flash MX 2004

シンタックス

`effect.sourceFile`

説明

読み取り専用プロパティ。指定したエフェクトの JSFL ソースファイル名を指定するストリングです。このストリングを使用して、XML パラメータファイルをその JSFL エフェクトの実装にバインドします。この XML パラメータは、エフェクトの XML ファイルに含める必要があります。

例

次の例では、JSFL エフェクトのソースファイルの名前を `efSourceFile` 変数に格納します。

```
var efSourceFile = fl.activeEffect.sourceFile;
```

effect.symbolType

使用できるバージョン

Flash MX 2004

シンタックス

`effect.symbolType`

説明

読み取り専用プロパティ。エフェクトの最初の適用時に作成するシンボルの種類を指定するストリングです。サポートされる種類は、"graphic"、"movie clip"、および "button" です。エフェクトを作成したときに、シンボルの種類が指定されていない場合、デフォルト値は "graphic" になります。

例

次の例では、現在のエフェクトのシンボルの種類を `efType` 変数に格納します。

```
var efType = fl.activeEffect.symbolType;
```

effect.useXMLToUI

使用できるバージョン

Flash MX 2004

シンタックス

`effect.useXMLToUI`

説明

プロパティ。1つ以上のコントロールから構成されるダイアログボックスを、XMLUI を使用して作成する際のデフォルトの動作を無効にするブール値です。デフォルト値は `true` です。`false` に設定すると、標準 XMLUI ダイアログボックスは表示されないのので、自分で UI を表示する必要があります。

例

次の例では、エフェクトで独自の UI を表示するように指定します。

```
function configureEffect() {  
    fl.activeEffect.useXMLToUI = false;  
}
```

Element オブジェクト

使用できるバージョン

Flash MX 2004

説明

ステージにあるすべてのエレメントの種類は `Element` です。次のコード例では、エレメントを選択できるようにします。

```
var el = fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0];
```

Element オブジェクトのメソッド一覧

Element オブジェクトでは次のメソッドを使用できます。

メソッド	説明
<code>element.getPersistentData()</code>	<code>name</code> パラメータで指定されるデータの値を取得します。
<code>element.getTransformationPoint()</code>	指定したエレメントの変形点の値を取得します。
<code>element.hasPersistentData()</code>	指定したデータが、指定したエレメントに割り当てられているかどうかを判断します。
<code>element.removePersistentData()</code>	オブジェクトに割り当てられた指定の名前を持つ永続データを削除します。
<code>element.setPersistentData()</code>	エレメントと共にデータを格納します。
<code>element.setTransformationPoint()</code>	エレメントの変形点の位置を設定します。

Element オブジェクトのプロパティ一覧

Element オブジェクトでは次のプロパティを使用できます。

プロパティ	説明
<code>element.depth</code>	読み取り専用。ビュー内のオブジェクトの深度が 0 より大きい整数です。
<code>element.elementType</code>	読み取り専用。指定したエレメントの種類を表すストリングです。
<code>element.height</code>	エレメントの高さをピクセル単位で指定する浮動小数値です。
<code>element.layer</code>	読み取り専用。エレメントがある Layer オブジェクト を表します。
<code>element.left</code>	読み取り専用。エレメントの左側を表す浮動小数値です。

プロパティ	説明
<code>element.locked</code>	ブール値。エレメントがロックされている場合は <code>true</code> で、ロックされていない場合は <code>false</code> です。
<code>element.matrix</code>	Matrix オブジェクト <code>matrix</code> には <code>a</code> 、 <code>b</code> 、 <code>c</code> 、 <code>d</code> 、 <code>tx</code> 、 <code>ty</code> の各プロパティがあります。 <code>a</code> 、 <code>b</code> 、 <code>c</code> 、 <code>d</code> は浮動小数値であり、 <code>tx</code> と <code>ty</code> は座標です。
<code>element.name</code>	エレメントの名前を指定するストリングです。通常、インスタンス名と呼ばれます。
<code>element.rotation</code>	オブジェクトの時計回りの回転角(度数)を指定する <code>-180</code> ～ <code>180</code> の整数値または浮動小数値です。
<code>element.scaleX</code>	シンボル、描画オブジェクト、およびプリミティブな矩形と楕円の水平方向の拡大/縮小値を指定する浮動小数値です。
<code>element.scaleY</code>	シンボル、描画オブジェクト、およびプリミティブな矩形と楕円の垂直方向の拡大/縮小値を指定する浮動小数値です。
<code>element.selected</code>	エレメントが選択されているかどうかを指定するブール値です。
<code>element.skewX</code>	シンボル、描画オブジェクト、およびプリミティブな矩形と楕円の水平方向の傾斜値を指定する <code>-180</code> ～ <code>180</code> の浮動小数値です。
<code>element.skewY</code>	シンボル、描画オブジェクト、およびプリミティブな矩形と楕円の垂直方向の傾斜値を指定する <code>-180</code> ～ <code>180</code> の浮動小数値です。
<code>element.top</code>	読み取り専用。エレメントの上端です。
<code>element.transformX</code>	エレメントの親の座標系の中で、選択したエレメントの変形点の <code>x</code> 値を指定する浮動小数点数です。
<code>element.transformY</code>	エレメントの親の座標系の中で、選択したエレメントの変形点の <code>y</code> 値を指定する浮動小数点数です。
<code>element.width</code>	エレメントの幅をピクセル単位で指定する浮動小数値です。
<code>element.x</code>	選択したエレメントの基準点の <code>x</code> 値を指定する浮動小数値です。
<code>element.y</code>	選択したエレメントの基準点の <code>y</code> 値を指定する浮動小数値です。

element.depth

使用できるバージョン

Flash MX 2004

シンタックス

`element.depth`

説明

読み取り専用プロパティ。ビュー内のオブジェクトの深度が0より大きい整数です。ステージ上でオブジェクトを描画する順序により、重ね順が決まります。オブジェクトの順序は、[修正]-[重ね順]メニューアイテムで指定することもできます。

例

次の例では、指定したエレメントの深度を [出力] パネルに表示します。

```
// オブジェクトを選択して、このスクリプトを実行します。  
fl.trace("Depth of selected object: " + fl.getDocumentDOM().selection[0].depth);  
  
element.elementType の例を参照してください。
```

element.elementType

使用できるバージョン

Flash MX 2004

シンタックス

`element.elementType`

説明

読み取り専用プロパティ。指定したエレメントの種類を表すストリングです。値は、"shape"、"text"、"instance"、または "shapeObj" のいずれかです。"shapeObj" は拡張ツールを使用して作成されます。

例

次の例では、最初のエレメントの種類を eType 変数に格納します。

```
// 新しいファイルで、最初のフレームの上部レイヤーにムービークリップを配置し、  
// このスクリプトを実行します。  
var eType =  
    fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].elementType;  
// eType = インスタンス
```

次の例では、現在のレイヤーまたはフレームにあるすべてのエレメントのプロパティを表示します。

```
var tl = fl.getDocumentDOM().getTimeline()
var elts = tl.layers[tl.currentLayer].frames[tl.currentFrame].elements;
for (var x = 0; x < elts.length; x++) {
    var elt = elts[x];
    fl.trace("Element "+ x + " Name = " + elt.name + " Type = " + elt.elementType +
        " location = " + elt.left + "," + elt.top + " Depth = " + elt.depth);
}
```

element.getPersistentData()

使用できるバージョン

Flash MX 2004

シンタックス

```
element.getPersistentData(name)
```

パラメータ

name 返されるデータを識別するストリング。

戻り値

name パラメータで指定されるデータ。データがない場合は 0 です。

説明

メソッド。 *name* パラメータで指定されるデータの値を取得します。データの種類は、格納されているデータの種類により異なります ([element.setPersistentData\(\)](#) を参照してください)。永続データは、シンボルとビットマップでのみサポートされています。

例

次の例では、指定したエレメントのデータを設定して取得し、その値を [出力] パネルに表示した後、データを削除します。

```
// 最初のフレームの最初のレイヤーで、少なくとも 1 つのシンボルまたはビットマップが選択されています。
var elt = fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0];
elt.setPersistentData("myData", "integer", 12);
if (elt.hasPersistentData("myData")){
    fl.trace("myData = "+ elt.getPersistentData("myData"));
    elt.removePersistentData("myData");
    fl.trace("myData = "+ elt.getPersistentData("myData"));
}
```

element.getTransformationPoint()

使用できるバージョン

Flash CS3 Professional

シンタックス

```
element.getTransformationPoint()
```

パラメータ

なし

戻り値

{x:10, y:20} などの点 (x と y は浮動小数点数)。エレメントの座標系の中で、変形点 (あるいは原点またはゼロ点) の位置を指定します。

説明

メソッド。指定したエレメントの変形点の値を取得します。

✕
中

変形点の位置は、選択したアイテムのタイプに応じて異なります。詳細については、[element.setTransformationPoint\(\)](#) を参照してください。

例

次の例では、ドキュメントの最初のレイヤーの 9 番目のフレーム内の 3 番目のエレメントの変形点を取得します。transPoint.x プロパティは、変形点の x 座標を表します。transPoint.y プロパティは、変形点の y 座標を表します。

```
var transPoint = fl.getDocumentDOM().getTimeline().layers[0].frames[8].elements[2].getTransformationPoint();
```

関連項目

[document.getTransformationPoint\(\)](#), [element.setTransformationPoint\(\)](#), [element.transformX](#), [element.transformY](#)

element.hasPersistentData()

使用できるバージョン

Flash MX 2004

シンタックス

```
element.hasPersistentData(name)
```

パラメータ

name テストするデータアイテムの名前を示すストリング。

戻り値

ブール値。指定したデータがオブジェクトに割り当てられている場合は `true`、割り当てられていない場合は `false` です。

説明

メソッド。指定したデータが、指定したエレメントに割り当てられているかどうかを判断します。永続データは、シンボルとビットマップでのみサポートされています。

例

詳細については、[element.getPersistentData\(\)](#) を参照してください。

element.height

使用できるバージョン

Flash MX 2004

シンタックス

`element.height`

説明

プロパティ。エレメントの高さをピクセル単位で指定する浮動小数値です。

×
ⓘ

このプロパティを使用してテキストフィールドのサイズを変更しないでください。サイズを変更する場合は、テキストフィールドを選択して、[document.setTextRectangle\(\)](#) を使用します。このプロパティをテキストフィールドで使用すると、テキストのサイズが変更されます。

例

次の例では、指定したエレメントの高さを `100` に設定します。

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].height = 100;
```

element.layer

使用できるバージョン

Flash 8

シンタックス

`element.layer`

説明

読み取り専用プロパティ。エレメントがある [Layer オブジェクト](#) を表します。

例

次の例では、エレメントがある Layer オブジェクトを theLayer 変数に格納します。

```
var theLayer = element.layer;
```

element.left

使用できるバージョン

Flash MX 2004

シンタックス

`element.left`

説明

読み取り専用プロパティ。エレメントの左側を表す浮動小数値です。element.left の値は、シーン内にあるエレメントの場合はステージの左上が基準になり、エレメントがシンボル内に格納されている場合はシンボルの基準点 (あるいは原点またはゼロ点) が基準になります。このプロパティを設定するには、[document.setSelectionBounds\(\)](#) または [document.moveSelectionBy\(\)](#) を使用します。

例

次の例では、エレメントを移動した際に、このプロパティの値がどのように変化するかを示します。

```
// ステージのエレメントを選択してから、このスクリプトを実行します。
var sel = fl.getDocumentDOM().selection[0];
fl.trace("Left (before) = " + sel.left);
fl.getDocumentDOM().moveSelectionBy({x:100, y:0});
fl.trace("Left (after) = " + sel.left);
```

[element.elementType](#) の例を参照してください。

element.locked

使用できるバージョン

Flash MX 2004

シンタックス

`element.locked`

説明

プロパティ。ブール値です。エレメントがロックされている場合は true、それ以外の場合は false になります。[element.elementType](#) の値が "shape" である場合、このプロパティは無視されます。

例

次の例では、最初のフレームの上側のレイヤーにある最初のエレメントをロックします。

```
// [ 修正 ]-[ 重ね順 ]-[ ロック ] と同じです。  
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].locked = true;
```

element.matrix

使用できるバージョン

Flash MX 2004

シンタックス

`element.matrix`

説明

プロパティ。**Matrix** オブジェクト。マトリックスには、a、b、c、d、tx、および ty の各プロパティがあります。a、b、c、および d の各プロパティは浮動小数値で、tx と ty の各プロパティは座標です。詳細については、[Matrix オブジェクト](#)を参照してください。

例

次の例では、指定したエレメントを x 方向に 10 ピクセル、y 方向に 20 ピクセル移動します。

```
var mat =  
    fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].matrix;  
mat.tx += 10;  
mat.ty += 20;  
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].matrix = mat;
```

element.name

使用できるバージョン

Flash MX 2004

シンタックス

`element.name`

説明

プロパティ。エレメントの名前を指定する文字列です。通常、インスタンス名と呼ばれます。`element.elementType` の値が "shape" である場合、このプロパティは無視されます。詳細については、[element.elementType](#)を参照してください。

例

次の例では、フレーム 1 の上側のレイヤーにある最初のエレメントのインスタンス名を "clip_mc" に設定します。

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].name =  
    "clip_mc";
```

[element.elementType](#) の例を参照してください。

element.removePersistentData()

使用できるバージョン

Flash MX 2004

シンタックス

```
element.removePersistentData(name)
```

パラメータ

name 削除するデータの名前を示すストリング。

戻り値

なし

説明

メソッド。オブジェクトに割り当てられた指定の名前を持つ永続データを削除します。永続データは、シンボルとビットマップでのみサポートされています。

例

詳細については、[element.getPersistentData\(\)](#) を参照してください。

element.rotation

使用できるバージョン

Flash CS3 Professional

シンタックス

```
element.rotation
```

説明

プロパティ。オブジェクトの時計回りの回転角 (度数) を指定する -180 ~ 180 の整数値または浮動小数値です。

例

次の例では、現在選択されているエレメントの回転角を 45 度に設定します。

```
var element = fl.getDocumentDOM().selection[0];
fl.trace("Element rotation = " + element.rotation);
element.rotation = 45;
fl.trace("After setting rotation to 45: rotation = " + element.rotation);
```

element.scaleX

使用できるバージョン

Flash CS3 Professional

シンタックス

`element.scaleX`

説明

プロパティ。シンボル、描画オブジェクト、およびプリミティブな矩形と楕円の水平方向の拡大 / 縮小値を指定する浮動小数値です。値が 1 の場合、縮尺は 100% です。

例

次の例では、現在選択しているエレメントの水平方向の拡大 / 縮小値を 2 (値を 2 倍にする) に設定します。

```
var element = fl.getDocumentDOM().selection[0];
element.scaleX = 2;
```

関連項目

[element.scaleY](#)

element.scaleY

使用できるバージョン

Flash CS3 Professional

シンタックス

`element.scaleY`

説明

プロパティ。シンボル、描画オブジェクト、およびプリミティブな矩形と楕円の垂直方向の拡大 / 縮小値を指定する浮動小数値です。値が 1 の場合、縮尺は 100% です。

例

次の例では、現在選択しているエレメントの垂直方向の拡大 / 縮小値を 2 (値を 2 倍にする) に設定します。

```
var element = fl.getDocumentDOM().selection[0];
element.scaleY = 2;
```

関連項目

[element.scaleX](#)

element.selected

使用できるバージョン

Flash 8

シンタックス

```
element.selected
```

説明

プロパティ。エレメントが選択されているか (true)、選択されていないか (false) を指定するブール値です。

例

次の例では、エレメントを選択します。

```
element.selected = true;
```

element.setPersistentData()

使用できるバージョン

Flash MX 2004

シンタックス

```
element.setPersistentData(name, type, value)
```

パラメータ

name データに関連付ける名前を示すストリング。この名前を使用してデータを取得します。

type データ型を示すストリング。指定できる値は、"integer"、"integerArray"、"double"、"doubleArray"、"string"、および "byteArray" です。

value オブジェクトに関連付ける値を指定します。*value* のデータ型は *type* パラメータの値により決まります。指定した値は、*type* パラメータで指定するデータ型に対応している必要があります。

戻り値

なし

説明

メソッド。エレメントと共にデータを格納します。このデータは、エレメントが含まれる FLA ファイルを再度開いている場合に使用できます。永続データは、シンボルとビットマップでのみサポートされています。

例

詳細については、`element.getPersistentData()` を参照してください。

element.setTransformationPoint()

使用できるバージョン

Flash CS3 Professional

シンタックス

```
element.setTransformationPoint(transformationPoint)
```

パラメータ

transformationPoint {x:10, y:20} などの点 (x と y は浮動小数点数)。エレメントまたはグループの変形点の値を指定します。

- シェイプ: *transformationPoint* は、ドキュメントを基準に設定されます (0,0 がステージの左上隅になる)。
- シンボル: *transformationPoint* は、シンボルの基準点を基準に設定されます (0,0 が基準点の位置になる)。
- テキスト: *transformationPoint* は、テキストフィールドを基準に設定されます (0,0 がテキストフィールドの左上隅になる)。
- ビットマップとビデオ: *transformationPoint* は、ビットマップまたはビデオを基準に設定されます (0,0 がビットマップまたはビデオの左上隅になる)。
- 描画オブジェクト、プリミティブなオブジェクト、およびグループ: *transformationPoint* は、エレメントまたはグループの中心点を基準に設定されます (0,0 がエレメントまたはグループの中心点になる)。

戻り値

なし。

説明

メソッド。エレメントの変形点の位置を設定します。

このメソッドは、`document.setTransformationPoint()` とほとんど同じです。ただし、次の点で異なります。

- 描画オブジェクト、プリミティブなオブジェクト、およびグループの変形点は、ステージではなく、エレメントまたはグループの中心を基準に設定されます。
- エレメントの変形点を設定するとき、エレメントを最初に選択しておく必要がありません。

このメソッドは変形点を移動しますが、エレメントを移動しません。一方、`element.transformX` と `element.transformY` の各プロパティはエレメントを移動します。

例

次の例では、ステージの 3 番目のエレメントの変形点を 100, 200 に設定します。

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].  
  elements[2].setTransformationPoint({x:100, y:200});
```

関連項目

[document.setTransformationPoint\(\).element.getTransformationPoint\(\)](#),
[element.transformX](#), [element.transformY](#)

element.skewX

使用できるバージョン

Flash CS3 Professional

シンタックス

`element.skewX`

説明

プロパティ。シンボル、描画オブジェクト、およびプリミティブな矩形と楕円の水平方向の傾斜値を指定する $-180 \sim 180$ の浮動小数値です。

例

次の例では、現在選択しているエレメントの水平方向の傾斜値を 10 に設定します。

```
var element = fl.getDocumentDOM().selection[0];  
element.skewX = 10;
```

関連項目

[document.setTransformationPoint\(\)](#), [element.skewY](#)

element.skewY

使用できるバージョン

Flash CS3 Professional

シンタックス

`element.skewY`

説明

プロパティ。シンボル、描画オブジェクト、およびプリミティブな矩形と楕円の垂直方向の傾斜値を指定する $-180 \sim 180$ の浮動小数値です。

例

次の例では、現在選択しているエレメントの垂直方向の傾斜値を 10 に設定します。

```
var element = fl.getDocumentDOM().selection[0];
element.skewY = 10;
```

関連項目

[document.setTransformationPoint\(\)](#), [element.skewX](#)

element.top

使用できるバージョン

Flash MX 2004

シンタックス

`element.top`

説明

読み取り専用プロパティ。エレメントの上端です。element.top の値は、シーン内にあるエレメントの場合はステージの左上が基準になり、エレメントがシンボル内に格納されている場合はシンボルの基準点が基準になります。このプロパティを設定するには、[document.setSelectionBounds\(\)](#) または [document.moveSelectionBy\(\)](#) を使用します。

例

次の例では、エレメントを移動した際に、このプロパティの値がどのように変化するかを示します。

```
// ステージのエレメントを選択してから、このスクリプトを実行します。
var sel = fl.getDocumentDOM().selection[0];
fl.trace("Top (before) = " + sel.top);
fl.getDocumentDOM().moveSelectionBy({x:0, y:100});
fl.trace("Top (after) = " + sel.top);
```

[element.elementType](#) の例を参照してください。

element.transformX

使用できるバージョン

Flash CS3 Professional

シンタックス

`element.transformX`

説明

プロパティ。エレメントの親の座標系の中で、選択したエレメントの変形点の *x* 値を指定する浮動小数点数です。このプロパティに新しい値を設定すると、エレメントが移動します。一方、`element.setTransformationPoint()` メソッドは、変形点を移動しますが、エレメントを移動しません。

関連項目

`element.getTransformationPoint()`, `element.setTransformationPoint()`,
`element.transformY`

element.transformY

使用できるバージョン

Flash CS3 Professional

シンタックス

`element.transformY`

説明

プロパティ。エレメントの親の座標系の中で、選択したエレメントの変形点の *y* 値を指定する浮動小数点数です。このプロパティに新しい値を設定すると、エレメントが移動します。一方、`element.setTransformationPoint()` メソッドは、変形点を移動しますが、エレメントを移動しません。

関連項目

`element.getTransformationPoint()`, `element.setTransformationPoint()`,
`element.transformX`

element.width

使用できるバージョン

Flash MX 2004

シンタックス

`element.width`

説明

プロパティ。エレメントの幅をピクセル単位で指定する浮動小数値です。

×
❌

このプロパティを使用してテキストフィールドのサイズを変更しないでください。サイズを変更する場合は、テキストフィールドを選択して、`document.setTextRectangle()` を使用します。このプロパティをテキストフィールドで使用すると、テキストのサイズが変更されます。

例

次の例では、指定したエレメントの幅を 100 に設定します。

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].width= 100;
```

element.x

使用できるバージョン

Flash CS3 Professional

シンタックス

`element.x`

説明

プロパティ。選択したエレメントの基準点の x 値を指定する浮動小数値です。

例

次の例では、指定したエレメントの基準点を 100, 200 に設定します。

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].widthx= 100;  
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].y= 200;
```

関連項目

[element.y](#)

element.y

使用できるバージョン

Flash CS3 Professional

シンタックス

`element.y`

説明

プロパティ。選択したエレメントの基準点の y 値を指定する浮動小数値です。

例

詳細については、[element.x](#) を参照してください。

Fill オブジェクト

使用できるバージョン

Flash MX 2004

説明

このオブジェクトには、ツールパネルまたは選択したシェイプの塗りのカラー設定のすべてのプロパティがあります。Fill オブジェクトを取得するには、`document.getCustomFill()` を使用します。

Fill オブジェクトのプロパティ一覧

Fill オブジェクトでは次のプロパティを使用できます。

プロパティ	説明
<code>fill.color</code>	塗りのカラーを表す、ストリング、16 進数値、または整数。
<code>fill.colorArray</code>	グラデーションカラーの配列。
<code>fill.focalPoint</code>	変形点からのグラデーションの焦点の水平オフセットを指定する整数。
<code>fill.linearRGB</code>	塗りを線状または放射状の RGB グラデーションでレンダリングするかどうかを指定するブール値。
<code>fill.matrix</code>	グラデーションの塗りの配置、方向、および拡大 / 縮小を定義する Matrix オブジェクト 。
<code>fill.overflow</code>	グラデーションのオーバーフローの動作を指定するストリング。
<code>fill.posArray</code>	整数の配列。0 ～ 255 の範囲で、対応するカラーの位置を示す。
<code>fill.style</code>	塗りのスタイルを指定するストリング。

fill.color

使用できるバージョン

Flash MX 2004

シンタックス

`fill.color`

説明

プロパティ。塗りのカラー。次のいずれかの形式です。

- "#RRGGBB" または "#RRGGBBAA" 形式のストリング
- 0xRRGGBB 形式の 16 進数
- 16 進数と等しい 10 進表現の整数

例

次の例では、現在の選択範囲の塗りのカラーを設定します。

```
var fill = fl.getDocumentDOM().getCustomFill();
fill.color = '#FFFFFF';
fl.getDocumentDOM().setCustomFill( fill );
```

fill.colorArray

使用できるバージョン

Flash MX 2004

シンタックス

fill.colorArray

説明

プロパティ。整数で表されるグラデーションカラーの配列です。このプロパティは、fill.style プロパティの値が "radialGradient" または "linearGradient" のいずれかである場合にのみ使用できます。詳細については、[fill.style](#) を参照してください。

例

次の例では、現在の選択範囲のカラー配列を、必要に応じて [出力] パネルに表示します。

```
var fill = fl.getDocumentDOM().getCustomFill();
if(fill.style == "linearGradient" || fill.style == "radialGradient")
    alert(fill.colorArray);
```

次の例では、指定した線状グラデーションの塗りを設定します。

```
var fill = fl.getDocumentDOM().getCustomFill();
fill.style = "linearGradient";
fill.colorArray = ["#00ff00", "#ff00ff"];
fill.posArray = [0, 255];
fl.getDocumentDOM().setCustomFill(fill);
```

fill.focalPoint

使用できるバージョン

Flash 8

シンタックス

fill.focalPoint

説明

プロパティ。変形点からのグラデーションの焦点の水平オフセットを指定する整数です。たとえば、値を **10** にすると、変形点からグラデーションのエッジまでの距離が **10/255** の地点に焦点を配置します。値を **-255** にすると、グラデーションの左端に焦点を配置します。デフォルト値は **0** です。

このプロパティは、**fill.style** プロパティの値が "radialGradient" である場合にのみ使用できます。

例

次の例では、現在の選択範囲の放射状グラデーションの焦点を、シェイプの中心に対して **100** ピクセル右側に設定します。

```
var fill = fl.getDocumentDOM().getCustomFill();
fill.style = "radialGradient";
fill.colorArray = ["#00ff00", "#ff00ff"];
fill.posArray = [0, 255];
fill.focalPoint = 10100;
fl.getDocumentDOM().setCustomFill(fill);
```

fill.linearRGB

使用できるバージョン

Flash 8

シンタックス

fill.linearRGB

説明

プロパティ。塗りを線状または放射状の **RGB** グラデーションでレンダリングするかどうかを指定するブール値です。グラデーションの線状補間を指定する場合は **true** に設定し、放射状補間を指定する場合は **false** に設定します。デフォルト値は **false** です。

例

次の例では、現在の選択範囲のグラデーションを線状の **RGB** でレンダリングするように指定します。

```
var fill = fl.getDocumentDOM().getCustomFill();
fill.linearRGB style = true"radialGradient";
fill.colorArray = ["#00ff00", "#ff00ff"];
fill.posArray = [0, 255];
fill.focalPoint = 100;
fill.linearRGB = true;
fl.getDocumentDOM().setCustomFill(fill);
```

fill.matrix

使用できるバージョン

Flash MX 2004

シンタックス

fill.matrix

説明

プロパティ。グラデーションの塗りの配置、方向、および拡大 / 縮小を定義する [Matrix オブジェクト](#) です。

例

次の例では、fill.matrix プロパティを使用して、現在の選択範囲のグラデーションの塗りを指定します。

```
var fill = fl.getDocumentDOM().getCustomFill();
fill.style = 'radialGradient';
fill.colorArray = ['#00ff00', '#ff00ff'];
fill.posArray = [0, 255];
fill.focalPoint = 100;
fill.linearRGB = false;
fill.overflow = 'repeat';
var mat = fl.getDocumentDOM().selection[0].matrix;
mat.a = 0.0167083740234375;
mat.b = -0.0096435546875;
mat.c = 0.0312957763671875;
mat.d = 0.05419921875;
mat.tx = 288.65;
mat.ty = 193.05;
for (i in mat) {
    fl.trace(i+' : '+mat[i]);
}
fl.getDocumentDOM().setCustomFill(fill);
```

fill.overflow

使用できるバージョン

Flash 8

シンタックス

fill.overflow

説明

プロパティ。グラデーションのオーバーフローの動作を指定するストリングです。指定できる値は、"extend"、"repeat"、および "reflect" です。ストリングの大文字小文字は区別されません。デフォルト値は "extend" です。

例

次の例では、現在の選択範囲のオーバーフローの動作を "extend" と指定します。

```
var fill = fl.getDocumentDOM().getCustomFill();
fill.overflow = "extend";
fl.getDocumentDOM().setCustomFill(fill);
```

fill.posArray

使用できるバージョン

Flash MX 2004

シンタックス

fill.posArray

説明

プロパティ。0 ～ 255 の範囲で、対応するカラーの位置を示す整数の配列です。このプロパティは、[fill.style](#) プロパティの値が "radialGradient" または "linearGradient" のいずれかである場合にのみ使用できます。

例

次の例では、現在の選択範囲の線状グラデーションで使用するカラーを指定します。

```
var fill = fl.getDocumentDOM().getCustomFill();
fill.style = "linearGradient";
fill.colorArray = [ 0x00ff00, 0xff0000, 0x0000ff ];
fill.posArray = [0, 100, 200];
fl.getDocumentDOM().setCustomFill( fill );
```

fill.style

使用できるバージョン

Flash MX 2004

シンタックス

fill.style

説明

プロパティ。塗りのスタイルを指定するストリングです。指定できる値は、"solid"、"linearGradient"、"radialGradient"、および "noFill" です。オブジェクトに塗りがいない場合、このプロパティの値は "noFill" です。

この値が "linearGradient" または "radialGradient" である場合、`fill.colorArray` プロパティと `fill.posArray` プロパティも指定できます。

例

次の例では、現在の選択範囲の線状グラデーションで使用するカラーを指定します。

```
var fill = fl.getDocumentDOM().getCustomFill();
fill.style= "linearGradient";
fill.colorArray = [ 0x00ff00, 0xff0000, 0x0000ff ];
fill.posArray = [0, 100, 200];
fl.getDocumentDOM().setCustomFill( fill );
```

Filter オブジェクト

使用できるバージョン

Flash 8

説明

このオブジェクトには、すべてのフィルタのすべてのプロパティがあります。filter.name はフィルタの種類を指定し、各フィルタに適用可能なプロパティを決定します。詳細については、[filter.name](#) を参照してください。

オブジェクトのフィルタリストを返すには、`document.getFilters()` を使用します。オブジェクトにフィルタを適用するには、`document.setFilters()` を使用します。[document.getFilters\(\)](#) および [document.setFilters\(\)](#) を参照してください。

Filter オブジェクトのプロパティ一覧

Filter オブジェクトでは次のプロパティを使用できます。

プロパティ	説明
filter.angle	シャドウやハイライトカラーの角度を度数で指定する浮動小数値。
filter.blurX	x 方向のぼかし量を指定する浮動小数値。
filter.blurY	y 方向のぼかし量を指定する浮動小数値。
filter.brightness	フィルタの明度を指定する浮動小数値。
filter.color	フィルタのカラーを表す、ストリング、16 進数値、または整数。
filter.contrast	フィルタのコントラスト値を指定する浮動小数値。
filter.distance	フィルタのエフェクトとオブジェクトの距離をピクセル単位で指定する浮動小数値。
filter.enabled	指定したフィルタが有効であるか無効であるかを指定するブール値です。
filter.hideObject	ソースイメージが非表示かどうかを指定するブール値。
filter.highlightColor	ハイライトカラーを表す、ストリング、16 進数値、または整数。
filter.hue	フィルタの色相を指定する浮動小数値。
filter.inner	影が内側の影かどうかを指定するブール値。
filter.knockout	フィルタがノックアウトフィルタかどうかを指定するブール値。
filter.name	読み取り専用。フィルタのタイプを指定するストリングです。

プロパティ	説明
<code>filter.quality</code>	ぼかしの品質を指定するストリング。
<code>filter.saturation</code>	フィルタの彩度値を指定する浮動小数値。
<code>filter.shadowColor</code>	影の色を表す、ストリング、16 進数値、または整数。
<code>filter.strength</code>	フィルタの強さをパーセンテージで指定する整数。
<code>filter.type</code>	ベベルまたはグローの種類を指定するストリング。

filter.angle

使用できるバージョン

Flash 8

シンタックス

`filter.angle`

説明

プロパティ。シャドウやハイライトカラーの角度を度数で指定する浮動小数値です。指定できる値は 0 ～ 360 です。`filter.name` プロパティの値が "bevelFilter"、"dropShadowFilter"、"gradientBevelFilter"、または "gradientGlowFilter" の Filter オブジェクトに対して、このプロパティを定義します。

例

次の例では、選択したオブジェクトに適用するベベルフィルタの角度を 120 に設定します。

```
var myFilters = fl.getDocumentDOM().getFilters();
for(i=0; i < myFilters.length; i++) {
    if(myFilters[i].name == 'bevelFilter'){
        myFilters[i].angle = 120;
    }
}
fl.getDocumentDOM().setFilters(myFilters);
```

関連項目

`document.setFilterProperty()`

filter.blurX

使用できるバージョン

Flash 8

シンタックス

`filter.blurX`

説明

プロパティ。x 方向のぼかし量を指定する浮動小数値です。指定できる値は 0 ～ 255 です。

`filter.name` プロパティの値が "bevelFilter"、"blurFilter"、"dropShadowFilter"、"glowFilter"、"gradientBevelFilter"、または "gradientGlowFilter" の Filter オブジェクトに対して、このプロパティを定義します。

例

次の例では、選択したオブジェクトに適用するぼかしフィルタの blurX 値を 30 に設定し、blurY 値を 20 に設定します。

```
var myFilters = fl.getDocumentDOM().getFilters();
for(i=0; i < myFilters.length; i++){
    if(myFilters[i].name == 'blurFilter'){
        myFilters[i].blurX = 30;
        myFilters[i].blurY = 20;
    }
}
fl.getDocumentDOM().setFilters(myFilters);
```

関連項目

`document.setFilterProperty()`, `filter.blurY`

filter.blurY

使用できるバージョン

Flash 8

シンタックス

`filter.blurY`

説明

プロパティ。y 方向のぼかし量を指定する浮動小数値です。指定できる値は 0 ～ 255 です。

`filter.name` プロパティの値が "bevelFilter"、"blurFilter"、"dropShadowFilter"、"glowFilter"、"gradientBevelFilter"、または "gradientGlowFilter" の Filter オブジェクトに対して、このプロパティを定義します。

例

詳細については、[filter.blurX](#) を参照してください。

関連項目

[document.setFilterProperty\(\)](#), [filter.blurX](#)

filter.brightness

使用できるバージョン

Flash 8

シンタックス

`filter.brightness`

説明

プロパティ。フィルタの明度を指定する浮動小数値です。指定できる値は -100 ～ 100 です。

[filter.name](#) プロパティの値が "adjustColorFilter" の `Filter` オブジェクトに対して、このプロパティを定義します。

例

次の例では、選択したオブジェクトに適用するカラー調整フィルタの明度を 30.5 に設定します。

```
var myFilters = fl.getDocumentDOM().getFilters();
for(i=0; i < myFilters.length; i++){
    if(myFilters[i].name == 'adjustColorFilter'){
        myFilters[i].brightness = 30.5;
    }
}
fl.getDocumentDOM().setFilters(myFilters);
```

filter.color

使用できるバージョン

Flash 8

シンタックス

`filter.color`

説明

プロパティ。フィルタのカラー。次のいずれかの形式です。

- "#RRGGBB" または "#RRGGBBAA" 形式のストリング
- 0xRRGGBB 形式の 16 進数
- 16 進数と等しい 10 進表現の整数

`filter.name` プロパティの値が "dropShadowFilter" または "glowFilter" の Filter オブジェクトに対して、このプロパティを定義します。

例

次の例では、選択したオブジェクトに適用するドロップシャドウフィルタのカラーを "#ff00003e" に設定します。

```
var myFilters = fl.getDocumentDOM().getFilters();
for(i=0; i < myFilters.length; i++){
    if(myFilters[i].name == 'dropShadowFilter'){
        myFilters[i].color = '#ff00003e';
    }
}
fl.getDocumentDOM().setFilters(myFilters);
```

関連項目

[document.setFilterProperty\(\)](#)

filter.contrast

使用できるバージョン

Flash 8

シンタックス

`filter.contrast`

説明

プロパティ。フィルタのコントラスト値を指定する浮動小数値です。指定できる値は -100 ～ 100 です。
`filter.name` プロパティの値が "adjustColorFilter" の Filter オブジェクトに対して、このプロパティを定義します。

例

次の例では、選択したオブジェクトに適用するカラー調整フィルタのコントラスト値を -15.5 に設定します。

```
var myFilters = fl.getDocumentDOM().getFilters();
for(i=0; i < myFilters.length; i++){
    if(myFilters[i].name == 'adjustColorFilter'){
        myFilters[i].contrast = -15.5;
    }
}
fl.getDocumentDOM().setFilters(myFilters);
```

filter.distance

使用できるバージョン

Flash 8

シンタックス

`filter.distance`

説明

プロパティ。フィルタのエフェクトとオブジェクトの距離をピクセル単位で指定する浮動小数値です。指定できる値は 255 ~ 255 です。`filter.name` プロパティの値が "bevelFilter"、"dropShadowFilter"、"gradientBevelFilter"、または "gradientGlowFilter" の Filter オブジェクトに対して、このプロパティを定義します。

例

次の例では、選択したオブジェクトに適用するドロップシャドウフィルタの距離を 10 ピクセルに設定します。

```
var myFilters = fl.getDocumentDOM().getFilters();
for(i=0; i < myFilters.length; i++){
    if(myFilters[i].name == 'dropShadowFilter'){
        myFilters[i].distance = 10;
    }
}
fl.getDocumentDOM().setFilters(myFilters);
```

関連項目

[document.setFilterProperty\(\)](#)

filter.enabled

使用できるバージョン

Flash CS3 Professional

シンタックス

`filter.enabled`

説明

プロパティ。指定したフィルタが有効か (true)、そうでないか (false) を指定するブール値です。

例

次の例では、選択したオブジェクトのカラーフィルタを無効にします。

```
var myFilters = fl.getDocumentDOM().getFilters();
for(i=0; i < myFilters.length; i++){
    if(myFilters[i].name == 'adjustColorFilter'){
        myFilters[i].enabled = false;
    }
}
fl.getDocumentDOM().setFilters(myFilters);
```

filter.hideObject

使用できるバージョン

Flash 8

シンタックス

filter.hideObject

説明

プロパティ。ソースイメージの表示 (true) または非表示 (false) を指定するブール値です。
filter.name プロパティの値が "dropShadowFilter" の Filter オブジェクトに対して、このプロパティを定義します。

例

次の例では、選択したオブジェクトに適用するドロップシャドウフィルタの hideObject 値を true に設定します。

```
var myFilters = fl.getDocumentDOM().getFilters();
for(i=0; i < myFilters.length; i++){
    if(myFilters[i].name == 'dropShadowFilter'){
        myFilters[i].hideObject = true;
    }
}
fl.getDocumentDOM().setFilters(myFilters);
```

filter.highlightColor

使用できるバージョン

Flash 8

シンタックス

filter.highlightColor

説明

プロパティ。ハイライトカラー。次のいずれかの形式です。

- "#RRGGBB" または "#RRGGBBAA" 形式のストリング
- 0xRRGGBB 形式の 16 進数
- 16 進数と等しい 10 進表現の整数

`filter.name` プロパティの値が "bevelFilter" の `Filter` オブジェクトに対して、このプロパティを定義します。

例

次の例では、選択したオブジェクトに適用するベベルフィルタのハイライトカラーを "#ff00003e" に設定します。

```
var myFilters = fl.getDocumentDOM().getFilters();
for(i=0; i < myFilters.length; i++){
    if(myFilters[i].name == 'bevelFilter'){
        myFilters[i].highlightColor = '#ff00003e';
    }
}
fl.getDocumentDOM().setFilters(myFilters);
```

filter.hue

使用できるバージョン

Flash 8

シンタックス

`filter.hue`

説明

プロパティ。フィルタの色相を指定する浮動小数値です。指定できる値は -180 ～ 180 です。

`filter.name` プロパティの値が "adjustColorFilter" の `Filter` オブジェクトに対して、このプロパティを定義します。

例

次の例では、選択したオブジェクトに適用するカラー調整フィルタの色相を 120 に設定します。

```
var myFilters = fl.getDocumentDOM().getFilters();
for(i=0; i < myFilters.length; i++){
    if(myFilters[i].name == 'adjustColorFilter'){
        myFilters[i].hue = 120;
    }
}
fl.getDocumentDOM().setFilters(myFilters);
```

filter.inner

使用できるバージョン

Flash 8

シンタックス

`filter.inner`

説明

プロパティ。影が内側の影か (true)、そうでないか (false) を指定するブール値です。[filter.name](#) プロパティの値が "dropShadowFilter" または "glowFilter" の `Filter` オブジェクトに対して、このプロパティを定義します。

例

次の例では、選択したオブジェクトに適用するグローフィルタの inner プロパティの値を true に設定します。

```
var myFilters = fl.getDocumentDOM().getFilters();
for(i=0; i < myFilters.length; i++){
    if(myFilters[i].name == 'glowFilter'){
        myFilters[i].inner = true;
    }
}
fl.getDocumentDOM().setFilters(myFilters);
```

関連項目

[document.setFilterProperty\(\)](#)

filter.knockout

使用できるバージョン

Flash 8

シンタックス

`filter.knockout`

説明

プロパティ。フィルタがノックアウトフィルタか (true)、そうでないか (false) を指定するブール値です。[filter.name](#) プロパティの値が "bevelFilter"、"dropShadowFilter"、"glowFilter"、"gradientBevelFilter"、または "gradientGlowFilter" の `Filter` オブジェクトに対して、このプロパティを定義します。

例

次の例では、選択したオブジェクトに適用するグローフィルタの knockout プロパティを true に設定します。

```
var myFilters = fl.getDocumentDOM().getFilters();
for(i=0; i < myFilters.length; i++){
    if(myFilters[i].name == 'glowFilter'){
        myFilters[i].knockout = true;
    }
}
fl.getDocumentDOM().setFilters(myFilters);
```

関連項目

[document.setFilterProperty\(\)](#)

filter.name

使用できるバージョン

Flash 8

シンタックス

filter.name

説明

読み取り専用プロパティ。フィルタの種類を指定するストリングです。このプロパティの値は、他に使用可能な Filter オブジェクトのプロパティを決定します。値は、"adjustColorFilter"、"bevelFilter"、"blurFilter"、"dropShadowFilter"、"glowFilter"、"gradientBevelFilter"、または "gradientGlowFilter" のいずれかです。

例

次の例では、フィルタ名とインデックス位置を [出力] パネルに表示します。

```
var myFilters = fl.getDocumentDOM().getFilters();
var traceStr = "";
for(i=0; i < myFilters.length; i++){
    traceStr = traceStr + " At index " + i + ": " + myFilters[i].name;
}
fl.trace(traceStr);
```

関連項目

[document.getFilters\(\)](#), [document.setFilterProperty\(\)](#)

filter.quality

使用できるバージョン

Flash 8

シンタックス

`filter.quality`

説明

プロパティ。ぼかしの品質を指定するストリングです。指定できる値は、"low"、"medium"、および "high" ("high" はガウスぼかしと同じ) です。[filter.name](#) プロパティの値が "bevelFilter"、"blurFilter"、"dropShadowFilter"、"glowFilter"、"gradientGlowFilter"、または "gradientBevelFilter" の `Filter` オブジェクトに対して、このプロパティを定義します。

例

次の例では、選択したオブジェクトに適用するグローフィルタのぼかしの品質を "medium" に設定します。

```
var myFilters = fl.getDocumentDOM().getFilters();
for(i=0; i < myFilters.length; i++){
    if(myFilters[i].name == 'glowFilter'){
        myFilters[i].quality = 'medium';
    }
}
fl.getDocumentDOM().setFilters(myFilters);
```

関連項目

[document.setFilterProperty\(\)](#)

filter.saturation

使用できるバージョン

Flash 8

シンタックス

`filter.saturation`

説明

プロパティ。フィルタの彩度値を指定する浮動小数値です。指定できる値は -100 ~ 100 です。[filter.name](#) プロパティの値が "adjustColorFilter" の `Filter` オブジェクトに対して、このプロパティを定義します。

例

次の例では、選択したオブジェクトに適用するカラー調整フィルタの彩度値を 100 (グレースケール) に設定します。

```
var myFilters = fl.getDocumentDOM().getFilters();
for(i=0; i < myFilters.length; i++){
    if(myFilters[i].name == 'adjustColorFilter'){
        myFilters[i].saturation = 0-100;
    }
}
fl.getDocumentDOM().setFilters(myFilters);
```

関連項目

[document.setFilterProperty\(\)](#)

filter.shadowColor

使用できるバージョン

Flash 8

シンタックス

filter.shadowColor

説明

プロパティ。影の色。次のいずれかの形式です。

- "#RRGGBB" または "#RRGGBBAA" 形式のストリング
- 0xRRGGBB 形式の 16 進数
- 16 進数と等しい 10 進表現の整数

[filter.name](#) プロパティの値が "bevelFilter" の Filter オブジェクトに対して、このプロパティを定義します。

例

次の例では、選択したオブジェクトに適用するベベルフィルタの影の色を "#ff00003e" に設定します。

```
var myFilters = fl.getDocumentDOM().getFilters();
for(i=0; i < myFilters.length; i++){
    if(myFilters[i].name == 'bevelFilter'){
        myFilters[i].shadowColor = '#ff00003e';
    }
}
fl.getDocumentDOM().setFilters(myFilters);
```

関連項目

[document.setFilterProperty\(\)](#)

filter.strength

使用できるバージョン

Flash 8

シンタックス

`filter.strength`

説明

プロパティ。フィルタの強さをパーセンテージで指定する整数です。指定できる値は 0 ～ 25,500 です。`filter.name` プロパティの値が "bevelFilter"、"dropShadowFilter"、"glowFilter"、"gradientGlowFilter"、または "gradientBevelFilter" の `Filter` オブジェクトに対して、このプロパティを定義します。

例

次の例では、選択したオブジェクトに適用するグローフィルタの強さを 50 に設定します。

```
var myFilters = fl.getDocumentDOM().getFilters();
for(i=0; i < myFilters.length; i++){
    if(myFilters[i].name == 'glowFilter'){
        myFilters[i].strength = 50;
    }
}
fl.getDocumentDOM().setFilters(myFilters);
```

関連項目

[document.setFilterProperty\(\)](#)

filter.type

使用できるバージョン

Flash 8

シンタックス

`filter.type`

説明

プロパティ。ベベルまたはグローのタイプを指定するストリングです。指定できる値は、"inner"、"outer"、および "full" です。`filter.name` プロパティの値が "bevelFilter"、"gradientGlowFilter"、または "gradientBevelFilter" の `Filter` オブジェクトに対して、このプロパティを定義します。

例

次の例では、選択したオブジェクトに適用するベベルフィルタのタイプを "full" に設定します。

```
var myFilters = fl.getDocumentDOM().getFilters();
for(i=0; i < myFilters.length; i++){
    if(myFilters[i].name == 'bevelFilter'){
        myFilters[i].type = 'full';
    }
}
fl.getDocumentDOM().setFilters(myFilters);
```

関連項目

[document.setFilterProperty\(\)](#)

flash オブジェクト (fl)

使用できるバージョン

Flash MX 2004

説明

flash オブジェクトは、Flash アプリケーションを表します。flash または fl を使用して、このオブジェクトを参照することができます。本マニュアルでは、すべてのコードサンプルで fl を使用しています。

flash オブジェクトのメソッド一覧

flash オブジェクトでは次のメソッドを使用できます。

メソッド	説明
<code>fl.addEventListener()</code>	特定のイベントを受け取ったときに呼び出す関数を登録します。
<code>fl.browseForFileURL()</code>	[ファイルを開く] または [ファイルを保存] システムダイアログボックスを開き、ユーザーは開いたり保存したりするファイルを指定できます。
<code>fl.browseForFolderURL()</code>	[フォルダを参照] ダイアログボックスを表示し、ユーザーがフォルダを選択できるようにします。
<code>fl.clipCopyString()</code>	指定されたストリングがクリップボードにコピーされます。
<code>fl.closeAll()</code>	開いているドキュメントをすべて閉じ、1回も保存されていないドキュメントに対しては [名前を付けて保存] ダイアログボックスを表示します。
<code>fl.closeAllPlayerDocuments()</code>	[制御]-[ムービープレビュー] で開いたすべての SWF ファイルを閉じます。
<code>fl.closeDocument()</code>	指定したドキュメントを閉じます。
<code>fl.closeProject()</code>	現在開いている Flash プロジェクトファイル (FLP) を閉じます。
<code>fl.createDocument()</code>	新しいドキュメントを開いて、そのドキュメントを選択します。
<code>fl.createProject()</code>	指定した名前で作成した Flash プロジェクトファイル (FLP) を作成します。
<code>fl.downloadLatestVersion()</code>	Version Cue サーバーから、現在開いていないファイルの最新バージョンをダウンロードします。
<code>fl.enableImmediateUpdates()</code>	スクリプト開発者は、エフェクトの実行中にタイムラインを視覚的に即時更新できるようになります。

メソッド	説明
<code>fl.fileExists()</code>	ファイルがディスクに既に存在するかどうかを確認します。
<code>fl.findDocumentDOM()</code>	固有の識別子を使用して特定のファイルを対象指定できます。
<code>fl.findDocumentIndex()</code>	<code>fl.documents</code> 配列内のドキュメントの位置を表す整数の配列を返します。
<code>fl.findObjectInDocByName()</code>	ドキュメント内で、インスタンス名に指定したテキストが含まれているエレメントを見つけます。
<code>fl.findObjectInDocByType()</code>	ドキュメント内で、指定したエレメント型のエレメントを見つけます。
<code>fl.getAppMemoryInfo()</code>	Flash.exe メモリの指定領域で使用されているバイト数を表す整数を返します。
<code>fl.getDocumentDOM()</code>	現在アクティブなドキュメントの DOM (Document オブジェクト) を取得します。
<code>fl.getProject()</code>	現在開いているプロジェクトを表す Project オブジェクトを返します。
<code>fl.mapPlayerURL()</code>	エスケープした Unicode URL を UTF-8 または MBCS URL にマッピングします。
<code>fl.openDocument()</code>	新しい Flash ドキュメントウィンドウで編集する Flash (FLA) ドキュメントを開き、フォーカスを移します。
<code>fl.openProject()</code>	Flash オーサリングツールで編集する Flash プロジェクトファイル (FLP) を開きます。
<code>fl.openScript()</code>	Flash テキストエディタでスクリプト (JSFL、AS、ASC) または他のファイル (XML、TXT) を開きます。
<code>fl.quit()</code>	Flash を終了し、変更したドキュメントを保存するためのプロンプトを表示します。
<code>fl.reloadEffects()</code>	ユーザーの "Configuration Effects" フォルダで定義されるすべてのエフェクト記述子をリロードします。
<code>fl.reloadTools()</code>	"toolconfig.xml" ファイルからツールパネルを再構築します。拡張ツールを作成する場合のみに使用します。
<code>fl.removeEventListener()</code>	<code>fl.addEventListener()</code> を使用して登録した関数を登録解除します。
<code>fl.resetAS3PackagePaths()</code>	[ActionScript 3.0 設定] ダイアログボックスのグローバルクラスパスの設定をデフォルト値にリセットします。
<code>fl.resetPackagePaths()</code>	[ActionScript 2.0 設定] ダイアログボックスのグローバルクラスパスの設定をデフォルト値にリセットします。

メソッド	説明
<code>fl.revertDocumentToLastVersion()</code>	指定したドキュメントを Version Cue サーバーのバージョンに復歸します。
<code>fl.runScript()</code>	JavaScript ファイルを実行します。
<code>fl.saveAll()</code>	開いているドキュメントをすべて保存し、1回も保存されていないドキュメントに対しては [名前を付けて保存] ダイアログボックスを表示します。
<code>fl.saveAVersionOfDocument()</code>	指定したドキュメントのバージョンを Version Cue サーバーに保存します。
<code>fl.saveDocument()</code>	指定したドキュメントを FLA ドキュメントとして保存します。
<code>fl.saveDocumentAs()</code>	指定したドキュメントの [名前を付けて保存] ダイアログボックスを表示します。
<code>fl.selectElement()</code>	エレメントを選択または編集できます。
<code>fl.selectTool()</code>	ツールパネルから指定したツールを選択します。
<code>fl.setActiveWindow()</code>	指定したドキュメントをアクティブウィンドウに設定します。
<code>fl.showIdleMessage()</code>	スクリプトの実行時間が長すぎる場合にメッセージを表示しません。
<code>fl.synchronizeDocumentWithHeadVersion()</code>	指定したドキュメントを Version Cue サーバーの最新バージョンと同期します。
<code>fl.trace()</code>	テキストストリングを [出力] パネルに送ります。

flash オブジェクトのプロパティ一覧

flash オブジェクトでは次のプロパティを使用できます。

プロパティ	説明
<code>fl.actionsPanel</code>	読み取り専用。 actionsPanel オブジェクト です。
<code>fl.activeEffect</code>	読み取り専用。現在適用されているエフェクトの Effect オブジェクト です。
<code>fl.as3PackagePaths</code>	[ActionScript 3.0 設定] ダイアログボックスのグローバルクラスパスの設定に対応するストリング。
<code>fl.compilerErrors</code>	読み取り専用。 compilerErrors オブジェクト 。
<code>fl.componentsPanel</code>	読み取り専用。 [コンポーネント] パネルを表す componentsPanel オブジェクト です。

プロパティ	説明
<code>fl.configDirectory</code>	読み取り専用。ローカルユーザーの "Configuration" フォルダをプラットフォーム固有のパスとして表す完全パスを指定する文字列です。
<code>fl.configURI</code>	読み取り専用。ローカルユーザーの "Configuration" ディレクトリを <code>file:///URI</code> として表す完全パスを指定する文字列です。
<code>fl.contactSensitiveSelection</code>	[接点を区別する] 選択モードを有効にするかどうかを指定するブール値です。
<code>fl.createNewDocList</code>	読み取り専用。作成可能なドキュメントの種類を表す文字列の配列です。
<code>fl.createNewDocListType</code>	読み取り専用。作成可能なドキュメントのファイル拡張子を表す文字列の配列です。
<code>fl.createNewTemplateList</code>	読み取り専用。作成可能なテンプレートの種類を表す文字列の配列です。
<code>fl.documents</code>	読み取り専用。現在、編集用に開いているドキュメント (FLA ファイル) を表す Document オブジェクト (Document オブジェクト を参照) の配列です。
<code>fl.drawingLayer</code>	読み取り専用。ユーザーがドラッグ中に一時的に描画する場合に拡張ツールで使用する adrawingLayer オブジェクト です。
<code>fl.effects</code>	読み取り専用。XML パラメータファイルを基にした Effect オブジェクト (Effect オブジェクト を参照) の配列です。
<code>fl.Math</code>	読み取り専用。マトリックスとポイントの操作に関するメソッドを提供する Math オブジェクト です。
<code>fl.mruRecentFileList</code>	読み取り専用。Flash オーサリングツールで管理する最後に使用した (MRU) リストにある完全なファイル名の配列です。
<code>fl.mruRecentFileListType</code>	読み取り専用。Flash オーサリングツールで管理する MRU リストにあるファイル形式の配列です。
<code>fl.packagePaths</code>	[ActionScript 2.0 設定] ダイアログボックスのグローバルクラスパスの設定に対応する文字列。
<code>fl.objectDrawingMode</code>	有効なオブジェクト描画モードを表す整数。
<code>fl.outputPanel</code>	読み取り専用。 outputPanel オブジェクト への参照です。
<code>fl.scriptURI</code>	読み取り専用。現在実行中の JSFL スクリプトのパスを <code>file:///URI</code> で表す文字列です。
<code>fl.tools</code>	読み取り専用。Tools オブジェクトの配列です。

プロパティ	説明
<code>fl.version</code>	読み取り専用。プラットフォームなど、Flash オーサリングツールの長いストリングのバージョンを返します。
<code>fl.xmlui</code>	読み取り専用。 XMLUI オブジェクト です。

fl.actionsPanel

使用できるバージョン

Flash CS3 Professional

シンタックス

```
fl.actionsPanel
```

説明

読み取り専用プロパティ。現在表示されている [アクション] パネルを表す `actionsPanel` オブジェクト。このプロパティの使用方法の詳細については、[actionsPanel オブジェクト](#)を参照してください。

fl.activeEffect

使用できるバージョン

Flash MX 2004

シンタックス

```
fl.activeEffect
```

説明

読み取り専用プロパティ。現在適用されているエフェクトの [Effect オブジェクト](#)です。

`fl.activeEffect` で使用できるプロパティについては、[207 ページの「Effect オブジェクトのプロパティ一覧」](#)を参照してください。

例

次の例では、現在のエフェクトを表すオブジェクトを `ef` 変数に格納します。

```
var ef = fl.activeEffect;
```

fl.addEventListener()

使用できるバージョン

Flash CS3 Professional

シンタックス

```
fl.addEventListener(eventType, callbackFunction)
```

パラメータ

eventType このコールバック関数に渡されるイベントタイプを指定するストリング。指定できる値は、"documentNew"、"documentOpened"、"documentClosed"、"mouseMove"、"documentChanged"、"layerChanged"、および"frameChanged"です。



documentChanged の値は、ドキュメントの内容が変更されたことを意味するのではなく、別のドキュメントが前面に配置されたことを意味します。つまり、fl.getDocumentDOM() は、このイベントが発生する前と後では異なる値を返します。

callbackFunction イベントが発生することに実行される関数を指定するストリング。

戻り値

なし。

説明

メソッド。特定のイベントが発生したときに呼び出す関数を登録します。

このメソッドを使用する場合は、mouseMove などのようにイベントが頻繁に発生し、関数の実行に時間がかかると、アプリケーションがハングするか、その他のエラー状態になる可能性があることに注意してください。

例

次の例では、ドキュメントが閉じられたときに、[出力] パネルにメッセージを表示します。

```
myFunction = function () {  
    fl.trace('document was closed');  
}  
fl.addEventListener("documentClosed", myFunction);
```

関連項目

[fl.removeEventListener\(\)](#)

fl.as3PackagePaths

使用できるバージョン

Flash CS3 Professional

シンタックス

`fl.as3PackagePaths`

説明

プロパティ。`[ActionScript 3.0 設定]` ダイアログボックスのグローバルクラスパスの設定に対応する文字列。文字列の中で各クラスパスはセミコロン (;) で区切られています。`ActionScript 2.0` クラスパスの設定を表示または変更するには、`fl.packagePaths` を使用します。

例

次の例では、`ActionScript 3.0` クラスパスの設定を変更します。

```
fl.trace(fl.as3PackagePaths);  
// 出力 ( デフォルト値で開始したと仮定する )  
// .:$(AppConfig)/ActionScript 3.0/Classes  
fl.as3PackagePaths="buying;selling";  
fl.trace(fl.as3PackagePaths);  
// 出力  
// 購入 ; 販売
```

関連項目

[fl.resetAS3PackagePaths\(\)](#)

fl.browseForFileURL()

使用できるバージョン

Flash MX 2004

シンタックス

`fl.browseForFileURL(browseType [, title [, previewArea]])`

パラメータ

browseType ファイルの参照操作の種類を指定する文字列です。指定できる値は、"open"、"select"、または "save" です。"open" と "select" では、システムの [ファイルを開く] ダイアログボックスを開きます。各値は Dreamweaver との互換性を維持するために用意されています。"save" では、システムの [ファイルを保存] ダイアログボックスを開きます。

title [ファイルを開く]ダイアログボックスまたは[ファイルを保存]ダイアログボックスのタイトルを指定する文字列。このパラメータを省略すると、デフォルト値が使用されます。このパラメータは省略可能です。

previewArea Flash と Fireworks で無視される省略可能なパラメータ。Dreamweaver との互換性を維持するためにだけ用意されています。

戻り値

file:/// URI で表したファイルの URL。ユーザーがダイアログボックスでキャンセルを選択した場合は null を返します。

説明

メソッド。[ファイルを開く]または[ファイルを保存]システムダイアログボックスを開き、ユーザーは開いたり保存したりするファイルを指定できます。

例

次の例では、ユーザーが FLA ファイルを選択して開くことができますようにします
fl.browseForFileURL() メソッドはすべての種類のファイルを参照できますが、
fl.openDocument() で開けるのは FLA ファイルだけです。

```
var fileURL = fl.browseForFileURL("open", "Select file");  
var doc = fl.openDocument(fileURL);
```

関連項目

[fl.browseForFolderURL\(\)](#)

fl.browseForFolderURL()

使用できるバージョン

Flash 8

シンタックス

```
fl.browseForFolderURL([description])
```

パラメータ

description [フォルダを参照]ダイアログボックスの説明を指定する省略可能な文字列。
このパラメータを省略すると、説明エリアには何も表示されません。

戻り値

file:/// URI で表したフォルダの URL。ユーザーがダイアログボックスでキャンセルを選択した場合は null を返します。

説明

メソッド。[フォルダを参照] ダイアログボックスを表示し、ユーザーがフォルダを選択できるようにします。

×
中

ダイアログボックスのタイトルは、必ず " フォルダを参照 " になります。タイトルの下の説明領域に「フォルダを選択」や「読み込むプロファイルのパスを選択」などの詳細情報を追加するには、`description` パラメータを使用します。

例

次の例では、ユーザーがフォルダを選択して、そのフォルダ内のファイルの一覧を表示できるようにします。

```
var folderURI = fl.browseForFolderURL("Select a folder.");  
var folderContents = FLfile.listFolder(folderURI);
```

関連項目

[fl.browseForFileURL\(\)](#), [FLfile](#) オブジェクト

fl.clipCopyString()

使用できるバージョン

Flash CS3 Professional

シンタックス

```
fl.clipCopyString(string)
```

パラメータ

string クリップボードにコピーするストリング。

戻り値

なし。

説明

メソッド。指定したストリングをクリップボードにコピーします。

現在の選択内容をクリップボードにコピーするには、[document.clipCopy\(\)](#) を使用します。

例

次の例では、現在のドキュメントのパスをドキュメントからクリップボードにコピーします。

```
var documentPath = fl.getDocumentDOM().path;  
fl.clipCopyString(documentPath);
```


fl.closeAll()

使用できるバージョン

Flash MX 2004

シンタックス

```
fl.closeAll([bPromptToSave])
```

パラメータ

bPromptToSave オプションのブール値。前回保存された後に変更されているファイルについて [保存] ダイアログボックスを表示するのか、または保存されたことがないファイルについて [名前を付けて保存] ダイアログボックスを表示するのかを指定します。デフォルト値は true です。

戻り値

なし。

説明

メソッド。開いているすべてのファイルを閉じます (FLA ファイル、SWF ファイル、JSFL ファイルなど)。開いているすべてのファイルを、変更内容を保存せずに閉じる場合は、*bPromptToSave* に false を指定します。このメソッドはアプリケーションを終了しません。

例

次のコードは、開いているすべてのファイルを閉じ、新規のファイルまたは変更したファイルを保存するためのプロンプトを表示します。

```
fl.closeAll();
```

関連項目

[fl.closeAllPlayerDocuments\(\)](#), [fl.closeDocument\(\)](#)

fl.closeAllPlayerDocuments()

使用できるバージョン

Flash CS3 Professional

シンタックス

```
fl.closeAllPlayerDocuments()
```

パラメータ

なし

戻り値

ブール値。ムービーウィンドウが開いていた場合は `true`、それ以外の場合は `false` を返します。

説明

メソッド。[制御]-[ムービープレビュー] で開いたすべての SWF ファイルを閉じます。

例

次の例では、[制御]-[ムービープレビュー] で開いたすべての SWF ファイルを閉じます。

```
fl.closeAllcloseAllPlayerDocuments();
```

関連項目

[fl.closeAll\(\)](#), [fl.closeDocument\(\)](#)

fl.closeDocument()

使用できるバージョン

Flash MX 2004

シンタックス

```
fl.closeDocument(documentObject [, bPromptToSaveChanges])
```

パラメータ

documentObject [Document オブジェクト](#)。*documentObject* でアクティブなドキュメントを参照している場合は、このメソッドを呼び出しているスクリプトの実行が終了するまで、ドキュメントウィンドウが閉じない場合があります。

bPromptToSaveChanges ブール値。*bPromptToSaveChanges* が `false` の場合は、ドキュメントの変更が保存されていなくても、ユーザーにプロンプトは表示されません。つまり、ファイルはそのまま閉じられ、変更内容は破棄されます。*bPromptToSaveChanges* が `true` の場合、ドキュメントの変更が保存されていないときには、標準の確認ダイアログボックスが表示されます。デフォルト値は `true` です。このパラメータは省略可能です。

戻り値

ブール値。成功した場合は `true` を返します。それ以外の場合は `false` を返します。

説明

メソッド。指定したドキュメントを閉じます。

例

次の例では、ドキュメントを閉じる 2 つの方法を示します。

```
// 指定したドキュメントを閉じ、変更を保存するためのプロンプトを表示します。
fl.closeDocument(fl.documents[0]);
fl.closeDocument(fl.documents[0] , true); // true は省略可能です。
// 指定したドキュメントを閉じ、変更を保存するためのプロンプトは表示しません。
fl.closeDocument(fl.documents[0], false);
```

関連項目

[fl.closeAll\(\)](#)

fl.closeProject()

使用できるバージョン

Flash 8

シンタックス

```
fl.closeProject()
```

パラメータ

なし

戻り値

ブール値。プロジェクトを正常に閉じた場合は true を返します。開いているプロジェクトファイルがない場合は false を返します。

説明

メソッド。現在開いている Flash プロジェクトファイル (FLP) を閉じます。

例

次の例では、プロジェクトファイルを閉じ、ファイルを正常に閉じたかどうかを示すメッセージを表示します。

```
fl.trace("The project was" + (fl.closeProject() ? "closed" : "not closed"));
```

関連項目

[fl.getProject\(\)](#), [fl.openProject\(\)](#), [Project](#) オブジェクト

fl.compilerErrors

使用できるバージョン

Flash CS3 Professional

シンタックス

`fl.compilerErrors`

説明

読み取り専用プロパティ。[エラー] パネルを表す `compilerErrors` オブジェクトです。このプロパティの使用方法の詳細については、[compilerErrors オブジェクト](#)を参照してください。

fl.componentsPanel

使用できるバージョン

Flash MX 2004

シンタックス

`fl.componentsPanel`

説明

読み取り専用プロパティ。[コンポーネント] パネルを表す [componentsPanel オブジェクト](#)です。

例

次の例では、`componentsPanel` オブジェクトを `comPanel` 変数に保存します。

```
var comPanel = fl.componentsPanel;
```

fl.configDirectory

使用できるバージョン

Flash MX 2004

シンタックス

`fl.configDirectory`

説明

読み取り専用プロパティ。ローカルユーザーの "Configuration" ディレクトリをプラットフォーム固有の形式で表す完全パスを指定するストリングです。このパスをプラットフォーム固有ではない `file:///URI` 形式で指定するには、[fl.configURI](#) を使用します。

例

次の例では、"Configuration" ディレクトリを [出力] パネルに表示します。

```
fl.trace("My local configuration directory is " + fl.configDirectory);
```

fl.configURI

使用できるバージョン

Flash MX 2004

シンタックス

```
fl.configURI
```

説明

読み取り専用プロパティ。ローカルユーザーの "Configuration" ディレクトリを file:/// URI として表示する完全パスを指定するストリングです。[fl.configDirectory](#) も参照してください。

例

次の例では、指定したスクリプトを実行します。fl.configURI を使用すると、スクリプトを実行するプラットフォームがわからなくても、スクリプトの場所を指定できます。

```
// コマンドメニューのコマンドを実行するには、次の行で "Test.jsfl" を  
// 実行したいコマンドに変更します。  
fl.runScript( fl.configURI + "Commands/Test.jsfl" );
```

fl.contactSensitiveSelection

使用できるバージョン

Flash 8

シンタックス

```
fl.contactSensitiveSelection
```

説明

接点を区別する選択モードを有効にするか(true)、無効にするか(false)を指定するブール値。

例

次の例では、選択する前に接点を区別する選択モードを無効にし、選択後にこのモードを元の値にリセットする方法を示します。

```
var contact = fl.contactSensitiveSelection;  
fl.contactSensitiveSelection = false;  
// ここに選択したコードを入力  
fl.contactSensitiveSelection = contact;
```

fl.createDocument()

使用できるバージョン

Flash MX 2004

シンタックス

```
fl.createDocument([docType])
```

パラメータ

docType 作成するドキュメントの種類を指定するストリング。指定できる値は、"timeline"、"presentation"、および "application" です。デフォルト値は "timeline" であり、[ファイル]-[新規]-[Flash ファイル (ActionScript 3.0)] を選択した場合と同じです。このパラメータは省略可能です。



このメソッドは Flash モバイルドキュメントの作成をサポートしません。

戻り値

メソッドが正常に終了した場合は、新しく作成されたドキュメントの Document オブジェクトを返します。エラーが発生した場合は、undefined を返します。

説明

メソッド。新しいドキュメントを開いて、そのドキュメントを選択します。サイズ、解像度、およびカラーの値は、現在のデフォルト値と同じです。

例

次の例では、異なる種類のドキュメントを作成します。

```
// タイムラインベースの Flash ドキュメントを作成します。
fl.createDocument();
fl.createDocument("timeline");
// スライドプレゼンテーションのドキュメントを作成します。
fl.createDocument("presentation");
// フォームアプリケーションのドキュメントを作成します。
fl.createDocument("application");
```

fl.createNewDocList

使用できるバージョン

Flash MX 2004

シンタックス

`fl.createNewDocList`

説明

読み取り専用プロパティ。作成可能なドキュメントの種類を表すストリングの配列です。

例

次の例では、作成できるドキュメントの種類を [出力] パネルに表示します。

```
fl.trace("Number of choices " + fl.createNewDocList.length);
for (i = 0; i < fl.createNewDocList.length; i++)
    fl.trace("choice: " + fl.createNewDocList[i]);
```

fl.createNewDocListType

使用できるバージョン

Flash MX 2004

シンタックス

`fl.createNewDocListType`

説明

読み取り専用プロパティ。作成可能なドキュメントのファイル拡張子を表すストリングの配列です。配列内のエントリは、インデックスごとに `fl.createNewDocList` 配列のエントリに直接対応します。

例

次の例では、作成できるドキュメントのファイル拡張子を [出力] パネルに表示します。

```
fl.trace("Number of types " + fl.createNewDocListType.length);
for (i = 0; i < fl.createNewDocListType.length; i++) fl.trace("type: " +
    fl.createNewDocListType[i]);
```

fl.createNewTemplateList

使用できるバージョン

Flash MX 2004

シンタックス

```
fl.createNewTemplateList
```

説明

読み取り専用プロパティ。作成可能なテンプレートの種類を表すstringの配列です。

例

次の例では、作成できるテンプレートの種類を [出力] パネルに表示します。

```
fl.trace("Number of template types: " + fl.createNewTemplateList.length); for (i  
    = 0; i < fl.createNewTemplateList.length; i++) fl.trace("type: " +  
    fl.createNewTemplateList[i]);
```

fl.createProject()

使用できるバージョン

Flash 8

シンタックス

```
fl.createProject(fileURI [ , name ])
```

パラメータ

fileURI 作成するFlashプロジェクトファイル (FLP) の名前を指定するstring。*file:///* URI で表します。

name [プロジェクト] パネルにプロジェクト名として表示される省略可能なstring。*name* を省略すると、FLP ファイルの名前 (パスまたは拡張子を除く) が [プロジェクト] パネルに表示されます。

戻り値

メソッドが正常に終了した場合は [Project オブジェクト](#) を返します。ファイルを作成できない場合 (たとえば、*fileURI* が存在しないディレクトリを含んでいる場合) は *undefined* を返します。

説明

メソッド。指定した名前で作成するFlashプロジェクトファイル (FLP) を作成します。ファイルを作成できない場合は、情報ダイアログボックスが表示されます。ファイルが既に存在する場合は、ファイルを上書きするかどうかを尋ねるダイアログボックスが表示されます。

例

次の例では、指定したディレクトリが存在する場合、そのディレクトリにプロジェクトファイルを作成し、[プロジェクト] パネルに表示する名前を指定します。

```
var myProject = fl.createProject("file:///C:/Projects/MasterProject_2005.flp",
    "Master Project");
```

関連項目

[fl.getProject\(\)](#), [fl.openProject\(\)](#), [Project オブジェクト](#)

fl.documents

使用できるバージョン

Flash MX 2004

シンタックス

```
fl.documents
```

説明

読み取り専用プロパティ。現在、編集用に開いているドキュメント (FLA ファイル) を表す Document オブジェクト ([Document オブジェクト](#)を参照) の配列です。

例

次の例では、開いているドキュメントの配列を docs 変数に格納します。

```
var docs = fl.documents;
```

次の例では、現在開いているドキュメントの名前を [出力] パネルに表示します。

```
for (doc in fl.documents) {
    fl.trace(fl.documents[doc].name);
}
```

fl.downloadLatestVersion()

使用できるバージョン

Flash CS3 Professional

シンタックス

```
fl.downloadLatestVersion(fileURI)
```

パラメータ

fileURI Version Cue サーバーからダウンロードするファイルのローカルパスを指定するストリング。file:///URI で表します。開いていないファイルだけをダウンロードできます。*fileURI* で指定したファイルが既に開いている場合、このメソッドは何も実行しません。

戻り値

ブール値。ファイルのダウンロードに成功した場合は true、それ以外の場合は false を返します。

説明

メソッド。Version Cue サーバーから、現在開いていないファイルの最新バージョンをダウンロードします。開いているファイルの最新バージョンをダウンロードするには、[document.synchronizeWithHeadVersion\(\)](#) を使用します。

例

次の例では、Version Cue サーバーから myFile.flc というファイルをダウンロードします。

```
fl.downloadLatestVersion("file:///C:/MyFiles/Version Cue/docs/myFile.flc");
```

関連項目

[document.synchronizeWithHeadVersion\(\)](#), [fl.revertDocumentToLastVersion\(\)](#),
[fl.saveAVersionOfDocument\(\)](#), [fl.synchronizeDocumentWithHeadVersion\(\)](#)

fl.drawingLayer

使用できるバージョン

Flash MX 2004

シンタックス

fl.drawingLayer

説明

読み取り専用プロパティ。選択マーキーを作成するときなど、ユーザーがドラッグ中に一時的に描画するときに拡張ツールで使用する [adrawingLayer オブジェクト](#) です。

例

詳細については、[drawingLayer.setColor\(\)](#) を参照してください。

fl.effects

使用できるバージョン

Flash MX 2004

シンタックス

`fl.effects`

説明

読み取り専用プロパティ。XML パラメータファイルを基にした Effect オブジェクト ([Effect オブジェクト](#)を参照) の配列です。これらはエフェクトではなく、エフェクトの説明です。配列長は、JSFL 実装ファイルの数ではなく、XML パラメータ定義ファイルに基づいて、プログラムの起動時に登録されるエフェクトの数に対応しています。

例

次の例では、最初に登録されたエフェクトを返します。

```
ef = fl.effects[0]
```

fl.enableImmediateUpdates()

使用できるバージョン

Flash MX 2004

シンタックス

`fl.enableImmediateUpdates(bEnableUpdates)`

パラメータ

bEnableUpdates エフェクトの実行中にタイムラインの視覚的な即時更新を有効にするか (true)、または無効にするか (false) を指定するブール値。

戻り値

なし

説明

メソッド。スクリプト開発者は、エフェクトの実行中にタイムラインを視覚的に即時更新できるようになります。即時更新は、目障りにならないようにするため、また必要以上に時間がかかっているように見えるのを避けるため、通常は非表示で実行され、ユーザーには途中の手順が表示されません。このメソッドは、デバッグでの使用を目的としています。フィールドに導入するエフェクトでは使用しないでください。エフェクトが完了すると、内部状態がリセットされ、即時更新が非表示になります。

例

次の例では、即時更新を有効にします。

```
fl.enableImmediateUpdates(true) ;  
fl.trace("Immediate updates are enabled");
```

fl.fileExists()

使用できるバージョン

Flash MX 2004

シンタックス

```
fl.fileExists(fileURI)
```

パラメータ

fileURI ファイルへのパスを含むストリング。file:/// URI で表します。

戻り値

ブール値。ファイルがディスクに存在する場合は true、存在しない場合は false を返します。

説明

メソッド。ファイルがディスクに既に存在するかどうかを確認します。

例

次の例では、指定した各ファイルが存在するかどうかに応じて、true または false を [出力] パネルに表示します。

```
alert(fl.fileExists("file:///C:/example fla"));  
alert(fl.fileExists("file:///C:/example.jsfl"));  
alert(fl.fileExists(""));
```

fl.findDocumentDOM()

使用できるバージョン

Flash CS3 Professional

シンタックス

```
fl.findDocumentDOM(id)
```

パラメータ

id ドキュメントを一意に識別する整数。

戻り値

Document オブジェクト。指定した *id* のドキュメントが存在しない場合は `null` を返します。

説明

メソッド。インデックス値などではなく、固有の識別子を使用して特定のファイルを対象指定できます。このメソッドを `document.id` と組み合わせて使用します。

例

次の例では、ドキュメントの ID を読み取り、その ID を使用してドキュメントを対象指定します。

```
var originalDocID = fl.getDocumentDOM().id;
// 別のファイルを使用している可能性があるため、他のコードをここに置く
var targetDoc = fl.findDocumentDOM(originalDocID);
// 元のドキュメントでステージの高さを 400 ピクセルに設定する。
targetDoc.height = 400;
```

関連項目

[fl.findDocumentIndex\(\)](#)

fl.findDocumentIndex()

使用できるバージョン

Flash MX 2004

シンタックス

```
fl.findDocumentIndex(name)
```

パラメータ

name インデックスを検索するドキュメントの名前。ドキュメントは開いている必要があります。

戻り値

`fl.documents` 配列内のドキュメント名の位置を表す整数の配列。

説明

メソッド。`fl.documents` 配列内のドキュメント名の位置を表す整数の配列を返します。同じ名前の複数のドキュメントを開くことができます。ただし、ドキュメントが異なるフォルダにある場合に限りです。

例

次の例では、`test.fla` という名前の開いているドキュメントのインデックス位置に関する情報を [出力] パネルに表示します。

```
var filename = "test.fla"
var docIndex = fl.findDocumentIndex(filename);
for (var index in docIndex)
    fl.trace(filename + " is open at index " + docIndex[index]);
```

関連項目

[fl.documents](#), [fl.findDocumentDOM\(\)](#)

fl.findObjectInDocByName()

使用できるバージョン

Flash CS3 Professional

シンタックス

`fl.findObjectInDocByName(instanceName, document)`

パラメータ

instanceName 指定したドキュメント内のアイテムのインスタンス名の一部または全部を指定するストリング。

document 指定したアイテムを検索する [Document オブジェクト](#)。

戻り値

汎用オブジェクトの配列です。配列の各アイテムの `.obj` プロパティを使用してオブジェクトを取得します。オブジェクトのプロパティには、`keyframe`、`layer`、`timeline`、および `parent` があります。これらのプロパティを使用して、オブジェクト階層にアクセスできます。プロパティおよびアクセス方法の詳細については、[fl.findObjectInDocByName\(\)](#) を参照してください。

`layer` および `timeline` 値のメソッドおよびプロパティにもアクセスできます。これらは、[Layer オブジェクト](#) および [Timeline オブジェクト](#) と同等です。

説明

メソッド。ドキュメント内で、インスタンス名に指定したテキストが含まれている要素を見つけます。

例

次の例では、インスタンス名に "text" が含まれているテキストフィールドを現在のドキュメントで検索し、テキストフィールドの内容を変更します。

```
var nameToSearchFor = "text";
var doc = fl.getDocumentDOM();
var results = fl.findObjectInDocByName(nameToSearchFor, doc);
if (results.length > 0) {
    for (var i = 0; i < results.length; i++) {
        results[i].obj.setTextString("new text");
    }
    alert("success, found " + results.length + " objects");
}
else {
    alert("failed, no objects of type " + nameToSearchFor + " found");
}
```

関連項目

[fl.findObjectInDocByType\(\)](#)

fl.findObjectInDocByType()

使用できるバージョン

Flash CS3 Professional

シンタックス

```
fl.findObjectInDocByType(elementType, document)
```

パラメータ

elementType 検索するエレメントの型を表すストリング。指定できる値については、[element.elementType](#) を参照してください。

document 指定したアイテムを検索する [Document オブジェクト](#)。

戻り値

汎用オブジェクトの配列です。配列の各アイテムの .obj プロパティを使用してエレメントオブジェクトを取得します。各オブジェクトには、keyframe、layer、timeline、および parent の各プロパティがあります。これらのプロパティを使用して、オブジェクト階層にアクセスできます。

layer および timeline 値のメソッドおよびプロパティにもアクセスできます。これらは、[Layer オブジェクト](#) および [Timeline オブジェクト](#) と同等です。

「例」セクションの 2 番目と 3 番目の例に、これらのプロパティにアクセスする方法を示しています。

説明

メソッド。ドキュメント内で、指定したエレメント型のエレメントを見つけます。

例

次の例では、現在のドキュメントのテキストフィールドを検索し、テキストフィールドの内容を変更します。

```
var doc = fl.getDocumentDOM();
var typeToSearchFor = "text";
var results = fl.findObjectInDocByType(typeToSearchFor, doc);
if (results.length > 0) {
    for (var i = 0; i < results.length; i++) {
        results[i].obj.setTextString("new text");
    }
    alert("success, found " + results.length + " objects");
}
else {
    alert("failed, no objects of type " + typeToSearchFor + " found");
}
```

次の例では、このメソッドから返された特殊なプロパティにアクセスする方法を示します。

```
var doc = fl.getDocumentDOM();
var resultsArray = findObjectInDocByType("text", doc);
if (resultsArray.length > 0)
{
    var firstItem = resultsArray[0];

    // firstItem.obj- 見つかったエレメントオブジェクト

    // このオブジェクトでは、次のプロパティにアクセスできる
    // firstItem.keyframe- エレメントのキーフレーム
    // firstItem.layer- キーフレームのレイヤー
    // firstItem.timeline- レイヤーのタイムライン
    // firstItem.parent- タイムラインの親。次に例を示します。
    // シンボルインスタンスのタイムラインなど。
}
```

次の例では、DOM をバックアップして、**resultArray.obj** オブジェクトを使用することで、テキストフィールドが見つかったレイヤーの名前を検出する方法を示します。

```
var doc = fl.getDocumentDOM();
var typeToSearchFor = "text";
var resultsArray = fl.findObjectInDocByType(typeToSearchFor, doc);
if (resultsArray.length > 0) {
    for (var i = 0; i < resultsArray.length; i++) {
        resultsArray[i].obj.setTextString("new text");
        var firstItem = resultsArray[0];
        firstItemObj = firstItem.obj;
        fl.trace(firstItemObj.layer.name+" layerName");
    }
} else {
    alert("failed, no objects of type " + typeToSearchFor + " found");
}
```


関連項目

```
fl.findObjectInDocByName()
```

fl.getAppMemoryInfo()

使用できるバージョン

Flash 8 (Windows のみ)

シンタックス

```
fl.getAppMemoryInfo(memType)
```

パラメータ

memType クエリー対象のメモリ使用領域を指定する整数。指定できる値の一覧については、次の説明を参照してください。

戻り値

Flash.exe メモリの指定領域で使用されているバイト数を表す整数。

説明

メソッド (Windows のみ)。Flash.exe メモリの指定領域で使用されているバイト数を表す整数を返します。*memType* として渡す値については、次の表を参照してください。

memType	リソースデータ
0	PAGEFAULTCOUNT
1	PEAKWORKINGSETSIZE
2	WORKINGSETSIZE
3	QUOTAPEAKPAGEDPOOLUSAGE
4	QUOTAPAGEDPOOLUSAGE
5	QUOTAPEAKNONPAGEDPOOLUSAGE
6	QUOTANONPAGEDPOOLUSAGE
7	PAGEFILEUSAGE
8	PEAKPAGEFILEUSAGE

例

次の例では、現在の作業メモリの消費量を表示します。

```
var memsize = fl.getAppMemoryInfo(2);
fl.trace("Flash current memory consumption is " + memsize + " bytes or " +
    memsize/1024 + " KB");
```

fl.getDocumentDOM()

使用できるバージョン

Flash MX 2004

シンタックス

```
fl.getDocumentDOM()
```

パラメータ

なし

戻り値

Document オブジェクトを返します。ドキュメントが開いていない場合は null を返します。

説明

メソッド。現在アクティブなドキュメント (FLA ファイル) の DOM ([Document オブジェクト](#)) を取得します。ドキュメントが開いていて、現在ドキュメントにフォーカスがない場合 (たとえば、JSFL ファイルにフォーカスがある場合)、最後にアクティブになっていたドキュメントの DOM を取得します。

例

次の例では、現在アクティブなドキュメント、または最後にアクティブになっていたドキュメントの名前を [出力] パネルに表示します。

```
var currentDoc = fl.getDocumentDOM();  
fl.trace(currentDoc.name);
```

fl.getProject()

使用できるバージョン

Flash 8

シンタックス

```
fl.getProject()
```

パラメータ

なし

戻り値

現在開いているプロジェクトを表す [Project オブジェクト](#)。現在開いているプロジェクトがない場合は、undefined を返します。

説明

メソッド。現在開いているプロジェクトを表す [Project オブジェクト](#) を返します。

例

次の例では、現在開いているプロジェクトの名前を [出力] パネルに表示します。

```
fl.trace("Current project: " + fl.getProject().name);
```

関連項目

[fl.createProject\(\)](#), [fl.openProject\(\)](#), [Project オブジェクト](#)

fl.mapPlayerURL()

使用できるバージョン

Flash MX 2004

シンタックス

```
fl.mapPlayerURL(URI [, returnMBCS])
```

パラメータ

URI マッピング対象のエスケープした Unicode URL を含むストリング。

returnMBCS ブール値。エスケープした MBCS パスを返す場合は true に設定する必要があります。それ以外の場合は UTF-8 を返します。デフォルト値は false です。このパラメータは省略可能です。

戻り値

変換された URL を表すストリング

説明

メソッド。エスケープした Unicode URL を UTF-8 または MBCS URL にマッピングします。このメソッドは、ストリングを **ActionScript** で使用して外部リソースにアクセスする場合に使用します。マルチバイト文字を処理する必要がある場合は、このメソッドを使用します。

例

次の例では、プレーヤーでロードできるよう URL を UTF-8 に変換します。

```
var url = MMExecute( "fl.mapPlayerURL(" + myURL + ", false);" );
mc.loadMovie( url);
```

fl.Math

使用できるバージョン

Flash MX 2004

シンタックス

fl.Math

説明

読み取り専用プロパティ。マトリックスとポイントの操作に関するメソッドを提供する [Math オブジェクト](#) です。

例

選択したオブジェクトの変換マトリックスと、その逆マトリックスを次に示します。

```
// ステージの要素を選択してから、このスクリプトを実行します。
var mat =fl.getDocumentDOM().selection[0].matrix;
for(var prop in mat){
    fl.trace("mat."+prop+" = " + mat[prop]);
}
var invMat = fl.Math.invertMatrix( mat );
for(var prop in invMat) {
    fl.trace("invMat."+prop+" = " + invMat[prop]);
}
```

fl.mruRecentFileList

使用できるバージョン

Flash MX 2004

シンタックス

fl.mruRecentFileList

説明

読み取り専用プロパティ。Flash オーサリングツールで管理する最後に使用した (MRU) リストにある完全なファイル名の配列です。

例

次の例では、最後に開いたファイルの数、および各ファイルの名前を [出力] パネルに表示します。

```
fl.trace("Number of recently opened files: " + fl.mruRecentFileList.length);
for (i = 0; i < fl.mruRecentFileList.length; i++) fl.trace("file: " +
    fl.mruRecentFileList[i]);
```

fl.mruRecentFileListType

使用できるバージョン

Flash MX 2004

シンタックス

`fl.mruRecentFileListType`

説明

読み取り専用プロパティ。Flash オーサリングツールで管理する MRU リストにあるファイル形式の配列です。この配列は、`fl.mruRecentFileList` プロパティの配列に対応します。

例

次の例では、最後に開いたファイルの数、および各ファイル形式を [出力] パネルに表示します。

```
fl.trace("Number of recently opened files: " + fl.mruRecentFileListType.length);
for (i = 0; i < fl.mruRecentFileListType.length; i++) fl.trace("type: " +
    fl.mruRecentFileListType[i]);
```

fl.objectDrawingMode

使用できるバージョン

Flash 8

シンタックス

`fl.objectDrawingMode`

説明

プロパティ。オブジェクト描画モードが有効か (true)、結合描画モードが有効か (false) を指定するブール値です。

例

次の例では、オブジェクト描画モードの状態を切り替えます。

```
var toggleMode = fl.objectDrawingMode;
if (toggleMode) {
    fl.objectDrawingMode = false;
} else {
    fl.objectDrawingMode = true;
}
```

fl.openDocument()

使用できるバージョン

Flash MX 2004

シンタックス

```
fl.openDocument(fileURI)
```

パラメータ

fileURI 開くファイルの名前を指定するストリング。file:/// URI で表します。

戻り値

メソッドが正常に終了した場合は、新しく作成されたドキュメントの [Document オブジェクト](#) を返します。ファイルが見つからない場合、または有効な FLA ファイルでない場合は、エラーが報告され、スクリプトはキャンセルされます。

説明

メソッド。新しい Flash ドキュメントウィンドウで編集する Flash ドキュメント (FLA ファイル) を開き、フォーカスを移します。[ファイル]-[開く] を選択してからファイルを選択するのと同じです。指定したファイルが既に開いている場合は、そのドキュメントのウィンドウが前面に表示されます。指定したファイルのウィンドウが、現在選択しているドキュメントになります。

例

次の例では、C ドライブのルートディレクトリに保存されている "Document.fla" ファイルを開きます。このドキュメントを表す Document オブジェクトを doc 変数に格納し、ドキュメントを現在選択されているドキュメントとして設定します。フォーカスが変更されるまで、fl.getDocumentDOM() はこのドキュメントを参照します。

```
var doc = fl.openDocument("file:///c:/Document.fla");
```

fl.openProject()

使用できるバージョン

Flash MX 2004。Flash 8 では、戻り値が異なります。

シンタックス

```
fl.openProject(fileURI)
```

パラメータ

fileURI 開く Flash プロジェクトファイル (FLP) のパスを指定するストリング。file:/// URI で表します。

戻り値

Flash MX 2004 の場合、戻り値はありません。Flash 8 の場合は、[Project オブジェクト](#)を返します。

説明

メソッド。Flash オーサリングツールで編集する Flash プロジェクトファイル (FLP) を開きます。

例

次の例では、C ドライブのルートディレクトリに保存されている "myProjectFile.flp" プロジェクトファイルを開きます。

```
fl.openProject("file:///c:/myProjectFile.flp");
```

関連項目

[fl.closeProject\(\)](#), [fl.createProject\(\)](#), [fl.getProject\(\)](#), [Project オブジェクト](#)

fl.openScript()

使用できるバージョン

Flash MX 2004

シンタックス

```
fl.openScript(fileURI)
```

パラメータ

fileURI Flash テキストエディタにロードする JSFL、AS、ASC、XML、TXT などのファイルのパスを指定する文字列。file:///URI で表します。

戻り値

なし

説明

メソッド。Flash テキストエディタでスクリプト (JSFL、AS、ASC) または他のファイル (XML、TXT) を開きます。

例

次の例では、C ドライブの "/temp" ディレクトリに保存されている "my_test.jsfl" ファイルを開きます。

```
fl.openScript("file:///c:/temp/my_test.jsfl");
```

fl.outputPanel

使用できるバージョン

Flash MX 2004

シンタックス

`fl.outputPanel`

説明

読み取り専用プロパティ。[outputPanel オブジェクト](#)への参照です。

例

詳細については、[outputPanel オブジェクト](#)を参照してください。

fl.packagePaths

使用できるバージョン

Flash CS3 Professional

シンタックス

`fl.packagePaths`

説明

プロパティ。`[ActionScript 2.0 設定]` ダイアログボックスのグローバルクラスパスの設定に対応するストリング。ストリングの中で各クラスパスはセミコロン (;) で区切られています。ActionScript 3.0 クラスパスの設定を表示または変更するには、[fl.as3PackagePaths](#) を使用します。

例

次の例では、ActionScript 2.0 クラスパスの設定を変更します。

```
fl.trace(fl.packagePaths);  
// 出力 ( デフォルト値で開始したと仮定する )  
// .;$(LocalData)/Classes  
fl.packagePaths="buying;selling";  
fl.trace(fl.packagePaths);  
// 出力 :  
// 購入 ; 販売
```

関連項目

[fl.resetPackagePaths\(\)](#)

fl.quit()

使用できるバージョン

Flash MX 2004

シンタックス

```
fl.quit([bPromptIfNeeded])
```

パラメータ

bPromptIfNeeded ブール値。変更したドキュメントを保存するためのプロンプトを表示する場合は、`true` (デフォルト) に設定します。変更したドキュメントを保存するためのプロンプトを表示しない場合は、このパラメータに `false` を設定します。後者の場合、開いているドキュメントの変更内容は破棄され、アプリケーションが終了します。バッチ処理を行う場合には便利ですが、使用する際には注意が必要です。このパラメータは省略可能です。

戻り値

なし

説明

メソッド。Flash を終了し、変更したドキュメントを保存するためのプロンプトを表示します。

例

次の例では、終了する際に、変更したドキュメントを保存するためのプロンプトを表示する場合と表示しない場合を示します。

```
// 終了する際に、変更したドキュメントを保存するためのプロンプトを表示します。
fl.quit();
fl.quit(true); // true は省略可能です。
// 終了する際に、ファイルを保存しません。
fl.quit(false);
```

fl.reloadEffects()

使用できるバージョン

Flash MX 2004

シンタックス

```
fl.reloadEffects()
```

パラメータ

なし

戻り値

なし

説明

メソッド。ユーザーの "Configuration Effects" フォルダで定義されるすべてのエフェクト記述子をリロードします。開発中にスクリプトをすばやく変更することができ、アプリケーションを再起動しなくても、エフェクトを改良することができます。このメソッドは、"Commands" フォルダのコマンドで使用する場合に最適です。

例

次の例は、"Commands" フォルダで利用できる 1 行のスクリプトです。エフェクトをリロードする必要がある場合は、[コマンド] メニューからスクリプトを実行します。

```
fl.reloadEffects();
```

fl.reloadTools()

使用できるバージョン

Flash MX 2004

シンタックス

```
fl.reloadTools()
```

パラメータ

なし

戻り値

なし

説明

メソッド。"toolconfig.xml" ファイルからツールパネルを再構築します。このメソッドは、拡張ツールを作成する場合にのみ使用します。ツールパネルをリロードする必要がある場合、たとえば、パネルに既に存在しているツールを定義する JSFL ファイルを修正した後、このメソッドを使用します。

例

次の例は、"Commands" フォルダで利用できる 1 行のスクリプトです。ツールパネルをリロードする必要がある場合は、[コマンド] メニューからスクリプトを実行します。

```
fl.reloadTools();
```

fl.removeEventListener()

使用できるバージョン

Flash CS3 Professional

シンタックス

```
fl.removeEventListener(eventType)
```

パラメータ

eventType このコールバック関数から削除するイベントタイプを指定する文字列。指定できる値は、"documentNew"、"documentOpened"、"documentClosed"、"mouseMove"、"documentChanged"、"layerChanged"、および "frameChanged" です。

戻り値

ブール値。イベントリスナの削除に成功した場合は true、fl.addEventListener() メソッドによって関数がリストに追加されていなかった場合は false を返します。

説明

[fl.addEventListener\(\)](#) を使用して登録した関数を登録解除します。

例

次の例では、documentClosed イベントに関連付けられたイベントリスナを削除します。

```
fl.removeEventListener("documentClosed");
```

関連項目

[fl.addEventListener\(\)](#)

fl.resetAS3PackagePaths()

使用できるバージョン

Flash CS3 Professional

シンタックス

```
fl.resetAS3PackagePaths()
```

パラメータ

なし

説明

メソッド。[ActionScript 3.0 設定] ダイアログボックスのグローバルクラスパスの設定をデフォルト値にリセットします。ActionScript 2.0 のグローバルクラスパスをリセットするには、[fl.resetPackagePaths\(\)](#) を使用します。

例

次の例では、ActionScript 3.0 クラスパスの設定をデフォルト値にリセットします。

```
fl.resetAS3PackagePaths();
```

関連項目

[fl.as3PackagePaths](#)

fl.resetPackagePaths()

使用できるバージョン

Flash CS3 Professional

シンタックス

```
fl.resetPackagePaths()
```

パラメータ

なし

説明

メソッド。[ActionScript 2.0 設定] ダイアログボックスのグローバルクラスパスの設定をデフォルト値にリセットします。ActionScript 3.0 のグローバルクラスパスをリセットするには、[fl.resetAS3PackagePaths\(\)](#) を使用します。

例

次の例では、ActionScript 2.0 クラスパスの設定をデフォルト値にリセットします。

```
fl.resetPackagePaths();
```

関連項目

[fl.packagePaths](#)

fl.revertDocument()

使用できるバージョン

Flash MX 2004

シンタックス

```
fl.revertDocument(documentObject)
```

パラメータ

documentObject [Document オブジェクト](#)。*documentObject* でアクティブなドキュメントを参照している場合は、このメソッドを呼び出しているスクリプトの実行が終了するまで、ドキュメントウィンドウが戻らない場合があります。

戻り値

ブール値。復帰操作が正常に終了した場合は `true` を返します。それ以外の場合は `false` を返します。

説明

メソッド。指定した FLA ドキュメントを、最後に保存したときの状態に戻します。[ファイル]-[復帰] メニューオプションとは異なり、このメソッドでは、ユーザーが操作を確認するためのメッセージは表示されません。[document.revert\(\)](#) および [document.canRevert\(\)](#) も参照してください。

Version Cue サーバーのバージョンにドキュメントを復帰するには、

[fl.revertDocumentToLastVersion\(\)](#) を使用します。

例

次の例では、現在の FLA ドキュメントを、最後に保存したときの状態に戻します。最後に保存してから行った変更内容は破棄されます。

```
fl.revertDocument(fl.getDocumentDOM());
```

fl.revertDocumentToLastVersion()

使用できるバージョン

Flash CS3 Professional

シンタックス

```
fl.revertDocumentToLastVersion(documentObject)
```

パラメータ

documentObject [Document オブジェクト](#)。

戻り値

ブール値。ドキュメントの復帰に成功した場合は `true`、それ以外の場合は `false` になります。

説明

メソッド。ファイルを復帰できる場合は、指定したドキュメントを Version Cue サーバーのバージョンに復帰し、[出力] パネルにエラーをログします。

ドキュメントをローカルに保存された最新バージョンに復帰するには、`fl.revertDocument()` を使用します。

例

次の例では、現在のドキュメントを Version Cue サーバーに保存されたバージョンに復帰します。

```
fl.revertDocumentToLastVersion(fl.getDocumentDOM());
```

関連項目

`document.revertToLastVersion()`, `fl.downloadLatestVersion()`,
`fl.saveAVersionOfDocument()`, `fl.synchronizeDocumentWithHeadVersion()`

fl.runScript()

使用できるバージョン

Flash MX 2004

シンタックス

```
fl.runScript(fileURI [, funcName [, arg1, arg2, ...]])
```

パラメータ

fileURI 実行するスクリプトファイルの名前を指定するストリング。file:/// URI で表します。

funcName *fileURI* で指定した JSFL ファイルで実行する関数を識別するストリング。このパラメータはオプションです。

arg *funcname* に渡すパラメータを指定する省略可能なパラメータ。

戻り値

funcName を指定すると、関数の結果をストリングで返します。指定しない場合は、何も返されません。

説明

メソッド。JavaScript ファイルを実行します。関数をパラメータの 1 つとして指定している場合は、その関数が実行されます。また関数内にはないスクリプトのコードも実行されます。スクリプト内の他のコードは、関数の実行前に実行されます。

例

Cドライブのルートディレクトリに "testScript.jsfl" スクリプトファイルがあり、その内容が次のとおりであるとしてします。

```
function testFuncnt(num, minNum) {  
    fl.trace("in testFuncnt: 1st arg: " + num + " 2nd arg: " + minNum);  
}  
for (i=0; i<2; i++) {  
    fl.trace("in for loop i=" + i);  
}  
fl.trace("end of for loop");  
// testScript.jsfl の終わり
```

次のコマンドを実行します。

```
fl.runScript("file:///C:/testScript.jsfl", "testFuncnt", 10, 1);
```

[出力] パネルに次の情報が表示されます。

```
in for loop i=0  
in for loop i=1  
end of for loop  
in testFuncnt: 1st arg: 10 2nd arg: 1
```

次のように、関数を実行せずに testScript.jsfl を呼び出すこともできます。

```
fl.runScript("file:///C:/testScript.jsfl");
```

次の内容が [出力] パネルに表示されます。

```
in for loop i=0  
in for loop i=1  
end of for loop
```

fl.saveAll()

使用できるバージョン

Flash MX 2004

シンタックス

```
fl.saveAll()
```

パラメータ

なし

戻り値

なし。

説明

メソッド。開いているすべてのドキュメントを保存します。

✕
中

ファイルはまだ保存したことがないか、ファイルを最後に保存してから、まだ修正していない場合、ファイルは保存されません。未保存または未修正のファイルを保存できるようにするには、`fl.saveDocumentAs()` を使用します。

例

次の例では、以前に保存され、前回の保存以降に変更されたドキュメントについて、開いているすべてのドキュメントを保存します。

```
fl.saveAll();
```

関連項目

[document.save\(\)](#), [document.saveAndCompact\(\)](#), [fl.saveDocument\(\)](#),
[fl.saveDocumentAs\(\)](#)

fl.saveAVersionOfDocument()

使用できるバージョン

Flash CS3 Professional

シンタックス

```
fl.saveAVersionOfDocument(document)
```

パラメータ

document [Document](#) オブジェクト。

戻り値

ブール値。ドキュメントのバージョンが Version Cue サーバーに正常に保存された場合は `true`、それ以外の場合は `false` になります。

説明

メソッド。ファイルを Version Cue サーバーに保存できる場合は、ユーザーがバージョンのコメントを入力するためのダイアログボックスを表示し、指定したドキュメントのバージョンをサーバーに保存し、エラーを [出力] パネルにログします。

例

次の例では、現在のドキュメントを Version Cue サーバーに保存します。

```
fl.saveAVersionOfDocument(fl.getDocumentDOM());
```

関連項目

[document.saveAVersion\(\)](#)

fl.saveDocument()

使用できるバージョン

Flash MX 2004

シンタックス

```
fl.saveDocument(document [, fileURI])
```

パラメータ

document 保存するドキュメントを指定する [Document オブジェクト](#)。*document* が null である場合、アクティブなドキュメントが保存されます。

fileURI 保存したドキュメントの名前を指定する文字列。file:///URI で表します。*fileURI* パラメータが null、または省略されている場合、ドキュメントは現在の名前で保存されます。このパラメータは省略可能です。

戻り値

ブール値。保存操作が正常に終了した場合は true を返します。それ以外の場合は false を返します。

×
#

ファイルはまだ保存したことがないか、ファイルを最後に保存してから、まだ修正していない場合、ファイルは保存されず、false が返されます。未保存または未修正のファイルを保存できるようにするには、[fl.saveDocumentAs\(\)](#) を使用します。

説明

メソッド。指定したドキュメントを FLA ドキュメントとして保存します。

例

次の例では、現在のドキュメントと、指定した 2 つのドキュメントを保存します。

```
// 現在のドキュメントを保存します。
alert(fl.saveDocument(fl.getDocumentDOM()));
// 指定したドキュメントを保存します。
alert(fl.saveDocument(fl.documents[0], "file:///C:/example1 fla"));
alert(fl.saveDocument(fl.documents[1], "file:///C:/example2 fla"));
```

関連項目

[document.save\(\)](#), [document.saveAndCompact\(\)](#), [fl.saveAll\(\)](#), [fl.saveDocumentAs\(\)](#)

fl.saveDocumentAs()

使用できるバージョン

Flash MX 2004

シンタックス

```
fl.saveDocumentAs(document)
```

パラメータ

document 保存するドキュメントを指定する [Document オブジェクト](#)。*document* が null である場合、アクティブなドキュメントが保存されます。

戻り値

ブール値。[名前を付けて保存] コマンドの操作が正常に終了した場合は true を返します。それ以外の場合は false を返します。

説明

メソッド。指定したドキュメントの [名前を付けて保存] ダイアログボックスを表示します。

例

次の例では、指定したドキュメントを保存するためのプロンプトを表示し、ドキュメントが保存されたかどうかを示すメッセージを表示します。

```
alert(fl.saveDocumentAs(fl.documents[1]));
```

関連項目

[document.save\(\)](#), [document.saveAndCompact\(\)](#), [fl.saveAll\(\)](#), [fl.saveDocument\(\)](#)

fl.scriptURI

使用できるバージョン

Flash CS3 Professional

シンタックス

```
fl.scriptURI
```

説明

読み取り専用プロパティ。現在実行中の JSFL スクリプトのパスを file:/// URI で表すストリングです。スクリプトを [fl.runScript\(\)](#) から呼び出した場合は、すぐ上の親スクリプトのパスを表します。つまり、[fl.runScript\(\)](#) に対する複数の呼び出しを通じて、最初の呼び出し元スクリプトのパスを見つけることはしません。

例

次の例では、現在実行されている JSFL スクリプトのパスを [出力] パネルに表示します。

```
fl.trace(fl.scriptURI);
```

関連項目

[fl.runScript\(\)](#)

fl.selectElement()

使用できるバージョン

Flash CS3 Professional

シンタックス

```
fl.selectElement(elementObject, editMode)
```

パラメータ

elementObject 選択する [Element オブジェクト](#)。

editMode エレメントを編集する (true) か、選択のみ (false) かを指定するブール値。

戻り値

ブール値。エレメントが正常に選択された場合は true、それ以外の場合は false を返します。

説明

メソッド。エレメントを選択または編集できます。通常では、[fl.findObjectInDocByName\(\)](#) または [fl.findObjectInDocByType\(\)](#) で返されたオブジェクトに対してこのメソッドを使用します。

例

次の例では、ドキュメントで見つかった場合、"second text field" というエレメントを選択します。

```
var nameToSearchFor = "second text field";
var doc = fl.getDocumentDOM();

// シーン 1 (インデックス値 0) の表示から開始する
document.editScene(0);

// エレメントを名前で検索する
var results = fl.findObjectInDocByName(nameToSearchFor, doc);
if (results.length > 0) {
    // 最初に見つかったエレメントを選択する
    // false を渡して、検索中の symbolInstance を選択する
    // true を渡すと、シンボルインスタンスが編集モードに切り替わる
    fl.selectElement(results[0], false);
    alert("success, found " + results.length + " objects")
}
```

```
else {  
    alert("failed, no objects with name " + nameToSearchFor + " found");  
}
```

関連項目

[fl.findObjectInDocByName\(\)](#), [fl.findObjectInDocByType\(\)](#)

fl.selectTool()

使用できるバージョン

Flash CS3 Professional

シンタックス

```
fl.selectTool(toolName)
```

パラメータ

toolName 選択するツールの名前を指定するストリング。このパラメータの指定できる値については、以下の「説明」を参照してください。

説明

メソッド。ツールパネルの指定したツールを選択します。*toolName* に指定できるデフォルト値は、"arrow"、"bezierSelect"、"freeXform"、"fillXform"、"lasso"、"pen"、"penplus"、"penminus"、"penmodify"、"text"、"line"、"rect"、"oval"、"rectPrimitive"、"ovalPrimitive"、"polystar"、"pencil"、"brush"、"inkBottle"、"bucket"、"eyeDropper"、"eraser"、"hand"、および "magnifier" です。

カスタムツールを作成した場合、そのツールの名前を *toolName* パラメータとして渡すこともできます。次のファイルに、ツール名のリストがあります。

- Windows 2000 または Windows XP :
 < ブートドライブ >¥Documents and Settings¥< ユーザー >¥Local Settings¥
 Application Data¥Adobe¥Flash CS3¥< 言語 >¥Configuration¥Tools¥toolConfig.xml
- Mac OS X :
 Macintosh HD/Users/< ユーザー名 >/Library/Application Support/Adobe/Flash CS3/
 < 言語 >/Configuration/Tools/toolConfig.xml

例

次の例では、ペンツールを選択します。

```
fl.selectTool("pen");
```

関連項目

[Tools オブジェクト](#), [ToolObj オブジェクト](#)

fl.setActiveWindow()

使用できるバージョン

Flash MX 2004

シンタックス

```
fl.setActiveWindow(document [, bActivateFrame])
```

パラメータ

document アクティブウィンドウとして選択するドキュメントを指定する [Document オブジェクト](#)。

bActivateFrame Flash と Fireworks で無視される省略可能なパラメータ。Dreamweaver との互換性を維持するためにだけ用意されています。

戻り値

なし。

説明

メソッド。指定したドキュメントをアクティブウィンドウに設定します。このメソッドは、Dreamweaver および Fireworks でもサポートされています。ドキュメントに、[新規ウィンドウで編集] コマンドで作成した複数のビューがある場合、最初のビューが選択されます。

例

次の例では、指定したドキュメントを保存する 2 つの方法を示します。

```
fl.setActiveWindow(fl.documents[0]);

var theIndex = fl.findDocumentIndex("myFile fla");
fl.setActiveWindow(fl.documents[theIndex]);
```

fl.showIdleMessage()

使用できるバージョン

Flash 8

シンタックス

```
fl.showIdleMessage(show)
```

パラメータ

show スクリプトの実行時間が長すぎる場合に、メッセージを表示するかどうかを指定するブール値

戻り値

なし。

説明

メソッド。スクリプトの実行時間が長すぎる場合にメッセージを表示しません (*show* に `false` を指定)。完了するのに時間がかかるバッチ処理を実行する場合に便利です。メッセージを表示するように変更するには、*show* に `true` を指定して、再度コマンドを実行します。

例

次の例では、スクリプトの実行時間が長すぎる場合にメッセージを表示しないように設定し、次に設定を変更してメッセージを表示するように設定します。

```
fl.showIdleMessage(false);
var result = timeConsumingFunction();
fl.showIdleMessage(true); ;
var result = timeConsumingFunction;
```

fl.synchronizeDocumentWithHeadVersion()

使用できるバージョン

Flash CS3 Professional

シンタックス

```
fl.showIdleMessage(truesynchronizeDocumentWithHeadVersion(documentObject);
```

パラメータ

documentObject [Document](#) オブジェクト。

戻り値

ブール値。指定したファイルが Version Cue サーバーとの同期に成功した場合は `true`、それ以外の場合は `false` を返します。

説明

メソッド。指定したドキュメントを Version Cue サーバーにある最新バージョンと同期し、[出力] パネルにエラーをログします。このメソッドは [document.synchronizeWithHeadVersion\(\)](#) と同じです。

例

次の例では、現在のドキュメントを Version Cue サーバーに保存します。

```
fl.synchronizeWithHeadVersion(fl.getDocumentDOM());
```

関連項目

[fl.downloadLatestVersion\(\)](#), [fl.revertDocumentToLastVersion\(\)](#),
[fl.saveAVersionOfDocument\(\)](#)

fl.tools

使用できるバージョン

Flash MX 2004

シンタックス

`fl.tools`

説明

読み取り専用プロパティ。Tools オブジェクト ([Tools オブジェクト](#)を参照) の配列です。このプロパティは、拡張ツールを作成する場合にのみ使用します。

fl.trace()

使用できるバージョン

Flash MX 2004

シンタックス

`fl.trace(message)`

パラメータ

message [出力] パネルに表示するストリング

戻り値

なし。

説明

メソッド。テキストストリングを [出力] パネルに送り、改行で終了させます。[出力] パネルが表示されていない場合は、表示します。このメソッドは、[outputPanel.trace\(\)](#) と同じで、ActionScript の `trace()` ステートメントと同じように機能します。

空白行を送るには、`fl.trace("")` または `fl.trace("\n")` を使用します。この後者のコマンドをインラインで使用して、`\n` を *message* ストリングの一部とすることができます。

例

次の例では、数行のテキストを [出力] パネルに表示します。

```
fl.outputPanel.clear();
fl.trace("Hello World!!!");
var myPet = "cat";
fl.trace("\nI have a " + myPet);
fl.trace("");
fl.trace("I love my " + myPet);
fl.trace("Do you have a " + myPet + "?");
```

fl.version

使用できるバージョン

Flash MX 2004

シンタックス

`fl.version`

説明

読み取り専用プロパティ。プラットフォームなど、Flash オーサリングツールの長いストリングのバージョンを返します。

例

次の例では、Flash オーサリングツールのバージョンを [出力] パネルに表示します。

```
alert(fl.version); // 例 : WIN 9,0,0,375
```

fl.xmlui

使用できるバージョン

Flash MX 2004

シンタックス

`fl.xmlui`

説明

読み取り専用プロパティ。[XMLUI オブジェクト](#)です。このプロパティを使用すると、XMLUI ダイアログボックスの XMLUI プロパティを取得および設定したり、ダイアログボックスの承認またはキャンセルをプログラムで制御したりできます。

例

詳細については、[XMLUI オブジェクト](#)を参照してください。

FLfile オブジェクト

使用できるバージョン

Flash MX 2004 7.2

説明

FLfile オブジェクトを使用すると、ローカルファイルシステム上のファイルとフォルダにアクセスしたり、修正および削除したりするための Flash 拡張機能を記述することができます。FLfile API は、JavaScript API の拡張機能として提供されます。この拡張機能は共有ライブラリと呼ばれ、次のフォルダに保存されます。

- Windows 2000 または Windows XP :

ブートドライブ ¥Documents and Settings¥ ユーザー ¥¥Local Settings¥

Application Data¥Adobe¥Flash CS3¥< 言語 >¥Configuration¥External Libraries¥FLfile.dll

- Mac OS X :

Macintosh HD/Users/< ユーザー名 >/Library/Application Support/Adobe/Flash CS3/

< 言語 >/Configuration/External Libraries/FLfile.dll



Flash ドキュメント内のシンボルを含む共有ライブラリと、JavaScript API の共有ライブラリを混同しないでください。この 2 つは異なるものです。

FLfile のメソッドは、ディスク上のファイルまたはフォルダ (ディレクトリ) で使用できます。したがって、各メソッドには、ファイルまたはフォルダの場所を指定するためのパラメータがあります。ファイルまたはフォルダの場所は、Web サイト URL と似た形式のストリングで表します。これをファイル URI (Uniform Resource Identifier) と言い、引用符を使用した次の形式で表します。

```
"file:///drive|/folder 1/folder 2/.../filename"
```

たとえば、C ドライブに "config" フォルダを作成し、"Program Files/MyApp" フォルダに配置するには、次のコマンドを使用します。

```
FLfile.createFolder("file:///C|/Program Files/MyApp/config");
```

上記フォルダに "config.ini" ファイルを配置する場合は、次のコマンドを使用します。

```
FLfile.write("file:///C|/Program Files/MyApp/config/config.ini", "");
```

Macintosh でフォルダを作成するには、次のコマンドを使用します。

```
FLfile.createFolder("file:///Macintosh/MyApp/config");
```

FLfile オブジェクトのメソッド一覧

FLfile オブジェクトでは次のメソッドを使用できます。

メソッド	説明
<code>FLfile.copy()</code>	ファイルをコピーします。
<code>FLfile.createFolder()</code>	フォルダを作成します。
<code>FLfile.exists()</code>	ファイルまたはフォルダが存在するかどうかを確認します。
<code>FLfile.getAttributes()</code>	ファイルの属性が、書き込み可能、読み取り専用、非表示、表示、またはシステムフォルダであるかどうかを確認します。
<code>FLfile.getCreationDate()</code>	1970 年 1 月 1 日からファイルまたはフォルダの作成時刻までに経過した秒数を指定します。
<code>FLfile.getCreationDateObj()</code>	ファイルまたはフォルダの作成日を取得します。
<code>FLfile.getModificationDate()</code>	1970 年 1 月 1 日からファイルまたはフォルダの最後の修正時刻までに経過した秒数を指定します。
<code>FLfile.getModificationDateObj()</code>	ファイルまたはフォルダの最後の修正日を取得します。
<code>FLfile.getSize()</code>	ファイルのサイズを取得します。
<code>FLfile.listFolder()</code>	フォルダの内容を一覧表示します。
<code>FLfile.read()</code>	ファイルの内容を読み取ります。
<code>FLfile.remove()</code>	ファイルまたはフォルダを削除します。
<code>FLfile.setAttributes()</code>	ファイルまたはフォルダを読み取り専用、書き込み可能、非表示、または表示に設定します。
<code>FLfile.write()</code>	ファイルの作成、ファイルへの書き込みまたは追加を行います。

FLfile.copy()

使用できるバージョン

Flash MX 2004 7.2

シンタックス

`FLfile.copy(fileURI, copyURI)`

パラメータ

fileURI コピー対象のファイルを指定するストリング。file:/// URI で表します。

copyURI コピーしたファイルの場所と名前を指定するストリング。file:/// URI で表します。

戻り値

ブール値。成功した場合は `true` を返します。それ以外の場合は `false` を返します。

説明

メソッド。ファイルを別の場所にコピーします。`copyURI` が既に存在する場合は、`false` を返します。

例

次の例では、`config.ini` という名前の設定ファイルのバックアップコピーを作成し、同じフォルダ内に別の名前で保存します。

```
var originalFileURI="file:///C:/Program Files/MyApp/config.ini";
var newFileURI="file:///C:/Program Files/MyApp/config_backup.ini";
FLfile.copy(originalFileURI, newFileURI);
```

単一のコマンドを使用して同じ操作を行うこともできます。

```
FLfile.copy("file:///C|:/Program Files/MyApp/config.ini", file:///C|/Program
  Files/MyApp/config_backup.ini");
```

FLfile.createFolder()

使用できるバージョン

Flash MX 2004 7.2

シンタックス

```
FLfile.createFolder(folderURI)
```

パラメータ

folderURI 作成するフォルダ構造を指定するフォルダ URI。

戻り値

ブール値。成功した場合は `true`、`folderURI` が既に存在している場合は `false` を返します。

説明

メソッド。指定した場所にフォルダを作成します。

一度に複数のフォルダを作成できます。たとえば、次のコマンドは、`"MyData"` フォルダと `"TempData"` のいずれも存在しなければ、両方を作成します。

```
FLfile.createFolder("file:///c|/MyData/TempData")
```

例

次の例では、設定フォルダ (`fl.configURI`) の下にフォルダとサブフォルダを作成します。

```
fl.trace(FLfile.createFolder(fl.configURI+"folder01/subfolder01"));
```

次の例では、Cドライブのルートに "tempFolder" フォルダを作成し、操作が成功したかどうかを示すメッセージボックスを表示します。

```
var folderURI = "file:///c:/tempFolder";
if (FLfile.createFolder(folderURI)) {
    alert("Created " + folderURI);
}
else {
    alert(folderURI + " already exists");
}
```

関連項目

[FLfile.remove\(\)](#), [FLfile.write\(\)](#)

FLfile.exists()

使用できるバージョン

Flash MX 2004 7.2

シンタックス

`FLfile.exists(fileURI)`

パラメータ

fileURI 確認するファイルを指定するストリング。file:/// URI で表します。

戻り値

ブール値。成功した場合は true を返します。それ以外の場合は false を返します。

説明

メソッド。指定したファイルが存在するかどうかを確認します。ファイルを作成するフォルダを指定する場合は、そのフォルダが既に存在する必要があります。フォルダを作成するには、

[FLfile.createFolder\(\)](#) を参照してください。

例

次の例では、一時フォルダ内で mydata.txt という名前のファイルを検索し、ファイルが存在するかどうかを示すメッセージボックスを表示します。

```
var fileURI = "file:///c:/temp/mydata.txt";
if (FLfile.exists(fileURI)) {
    alert( fileURI + " exists.");
}
else {
    alert( fileURI + " does not exist.");
}
```

次の例では、必要な設定ファイルが **MyApplication** フォルダに存在するかどうかを確認します。ファイルが存在しない場合は作成されます。

```
var configFile = "file:///C:/MyApplication/config.ini";
if (!FLfile.exists(configFile)) {
    FLfile.write(configFile,"");
}
```

関連項目

[FLfile.write\(\)](#)

FLfile.getAttributes()

使用できるバージョン

Flash MX 2004 7.2

シンタックス

`FLfile.getAttributes(fileOrFolderURI)`

パラメータ

fileOrFolderURI 属性を取得するファイルまたはフォルダを指定するストリング。file:///URI で表します。

戻り値

指定したファイルまたはフォルダの属性を表すストリング。



ファイルやフォルダが存在しない場合は、予期しない結果が生じる可能性があります。このメソッドを使用する前に、[FLfile.exists\(\)](#) を使用する必要があります。

説明

メソッド。指定したファイルまたはフォルダの属性を表すストリングを返します。ファイルに読み取り専用や非表示などの特定の属性がない場合は、空のストリングを返します。このメソッドを使用する前に、必ず [FLfile.exists\(\)](#) を使用してファイルやフォルダの存在をテストする必要があります。

ストリング内の文字は、次の属性を表します。

R – *fileOrFolderURI* は読み取り専用

D – *fileOrFolderURI* はフォルダ (ディレクトリ)

H – *fileOrFolderURI* は非表示 (Windows のみ)

S – *fileOrFolderURI* はシステムファイルまたはシステムフォルダ (Windows のみ)

A – *fileOrFolderURI* はアーカイブ可能 (Windows のみ)

たとえば、*fileOrFolderURI* が隠しフォルダである場合、"DH" というストリングを返します。

例

次の例では、"mydata.txt" ファイルの属性を取得し、ファイルが読み取り専用である場合は、メッセージボックスを表示します。

```
var URI = "file:///c:/temp/mydata.txt";
if (FLfile.exists(URI)){
    var attr = FLfile.getAttributes(URI);
    if (attr && (attr.indexOf("R") != -1)) { // R を含むストリングが返されます。
        alert(URI + " is read only!");
    }
}
```

関連項目

[FLfile.setAttributes\(\)](#)

FLfile.getCreationDate()

使用できるバージョン

Flash MX 2004 7.2

シンタックス

`FLfile.getCreationDate(fileOrFolderURI)`

パラメータ

fileOrFolderURI 作成日時を 16 進数のストリングで取得するファイルまたはフォルダを指定するストリング。file:///URI で表します。

戻り値

1970 年 1 月 1 日からファイルまたはフォルダの作成時刻までに経過した秒数を表す 16 進数のストリングを返します。ファイルやフォルダが存在しない場合は、"00000000" を返します。

説明

メソッド。1970 年 1 月 1 日からファイルまたはフォルダの作成時刻までに経過した秒数を指定します。このメソッドは、主にファイルやフォルダの作成日または修正日を比較するために使用されます。

例

次の例では、ファイルの作成後、ファイルが修正されているかどうかを確認します。

```
// 指定したファイルが存在することを確認します。
var fileURI = "file:///C:/MyApplication/MyApp fla";
var creationTime = FLfile.getCreationDate(fileURI);
var modificationTime = FLfile.getModificationDate(fileURI);
if ( modificationTime > creationTime ) {
    alert("The file has been modified since it was created.");
}
else {
    alert("The file has not been modified since it was created.");
}
```

関連項目

[FLfile.getCreationDateObj\(\)](#), [FLfile.getModificationDate\(\)](#)

FLfile.getCreationDateObj()

使用できるバージョン

Flash MX 2004 7.2

シンタックス

FLfile.getCreationDateObj(*fileOrFolderURI*)

パラメータ

fileOrFolderURI 作成日時を JavaScript Date オブジェクトとして取得するファイルを指定するストリング。file:/// URI で表します。

戻り値

指定したファイルまたはフォルダの作成日時を表す JavaScript Date オブジェクトを返します。ファイルが存在しない場合は、オブジェクトには、ファイルまたはフォルダが 1969 年 12 月 31 日 0 時 (グリニッジ標準時) に作成されたことを示す情報が格納されます。

説明

メソッド。指定したファイルまたはフォルダの作成日時を表す JavaScript Date オブジェクトを返します。

例

次の例では、ファイルの作成日を人間が理解できる形式で [出力] パネルに表示します。

```
// 指定したファイルが存在することを確認します。
var file1Date = FLfile.getCreationDateObj("file:///c:/temp/file1.txt");
fl.trace(file1Date);
```


関連項目

[FLfile.getCreationDate\(\)](#), [FLfile.getModificationDateObj\(\)](#)

FLfile.getModificationDate()

使用できるバージョン

Flash MX 2004 7.2

シンタックス

`FLfile.getModificationDate(fileOrFolderURI)`

パラメータ

fileOrFolderURI 修正日時を 16 進数のストリングで取得するファイルを指定するストリング。
file:/// URI で表します。

戻り値

1970 年 1 月 1 日からファイルまたはフォルダの最後の修正時刻までに経過した秒数を表す 16 進数のストリングを返します。ファイルが存在しない場合は、"00000000" を返します。

説明

メソッド。1970 年 1 月 1 日からファイルまたはフォルダの最後の修正時刻までに経過した秒数を指定します。このメソッドは、主にファイルやフォルダの作成日または修正日を比較するために使用されます。

例

次の例では、2 つのファイルの修正日時を比較し、どちらのファイルが最後に修正されたかを確認します。

```
// 指定したファイルが存在することを確認します。
file1 = "file:///C:/MyApplication/MyApp fla"
file2 = "file:///C:/MyApplication/MyApp.as"
modificationTime1 = FLfile.getModificationDate(file1)
modificationTime2 = FLfile.getModificationDate(file2)

if(modificationTime1 > modificationTime2) {
    alert("File 2 is older than File 1")
}
else if(modificationTime1 < modificationTime2) {
    alert("File 1 is older than File 2")
}
else {
    alert("File 1 and File 2 were saved at the same time")
}
```

関連項目

[FLfile.getCreationDate\(\)](#), [FLfile.getModificationDateObj\(\)](#)

FLfile.getModificationDateObj()

使用できるバージョン

Flash MX 2004 7.2

シンタックス

```
FLfile.getModificationDateObj(fileOrFolderURI)
```

パラメータ

fileOrFolderURI 修正日時を JavaScript Date オブジェクトとして取得するファイルやフォルダを指定する文字列。file:///URI で表します。

戻り値

指定したファイルまたはフォルダの最後の修正日時を表す JavaScript Date オブジェクトを返します。ファイルやフォルダが存在しない場合は、オブジェクトには、ファイルまたはフォルダが1969年12月31日0時(グリニッジ標準時)に作成されたことを示す情報が格納されます。

説明

メソッド。指定したファイルまたはフォルダの最後の修正日時を表す JavaScript Date オブジェクトを返します。

例

次の例では、ファイルの最後の修正日を人間が理解できる形式で [出力] パネルに表示します。

```
// 指定したファイルが存在することを確認します。
var file1Date = FLfile.getModificationDateObj("file:///c:/temp/file1.txt");
trace(file1Date);
```

関連項目

[FLfile.getCreationDateObj\(\)](#), [FLfile.getModificationDate\(\)](#)

FLfile.getSize()

使用できるバージョン

Flash MX 2004 7.2

シンタックス

```
FLfile.getSize(fileURI)
```

パラメータ

fileURI サイズを取得するファイルを指定する文字列。file:///URI で表します。

戻り値

指定したファイルのサイズをバイト単位で表す整数を返します。ファイルが存在しない場合は 0 を返します。

説明

メソッド。指定したファイルのサイズをバイト単位で表す整数を返します。ファイルが存在しない場合は 0 を返します。戻り値が 0 の場合、`FLfile.exists()` を使用して、ファイルがゼロバイトファイルか、またはファイルが存在しないかを調べることができます。

このメソッドは、ファイルサイズが 2 GB 以下の場合にのみ正しいファイルサイズを返します。

例

次の例では、`mydata.txt` ファイルのサイズを `fileSize` 変数に格納します。

```
var URL = "file:///c:/temp/mydata.txt";  
var fileSize = FLfile.getSize(URL);
```

FLfile.listFolder()

使用できるバージョン

Flash MX 2004 7.2

シンタックス

```
FLfile.listFolder(folderURI [, filesOrDirectories])
```

パラメータ

folderURI 内容を取得するフォルダを指定するストリング。`file:///URI` で表します。*folderURI* にワイルドカードマスクを含めることもできます。指定できるワイルドカードは、`*` (1文字以上と一致) と `?` (1文字と一致) です。

filesOrDirectories ファイル名のみを返すかフォルダ名(ディレクトリ名)のみを返すかを指定する省略可能なストリング。省略した場合、ファイル名とフォルダ名の両方が返されます。指定できる値は、`"files"` および `"directories"` です。

戻り値

フォルダの内容を表すストリングの配列を返します。フォルダが存在しない場合は `false` を返します。

説明

メソッド。フォルダの内容を表すストリングの配列を返します。フォルダが存在しない場合は 空の配列を返します。

例

次の例では、"Program Files"ディレクトリにあるファイル、フォルダ、またはファイルとフォルダの両方を表す配列を返します。

```
var folderURI = "file:///C:/WINDOWS/Program Files" ;
var fileList = FLfile.listFolder(folderURI, "files") // ファイル
var fileList = FLfile.listFolder(folderURI, "directories") // フォルダ
var fileList = FLfile.listFolder(folderURI) // ファイルとフォルダ
```

次の例では、一時フォルダにあるすべてのテキストファイル(.txt)の配列を返し、その一覧をメッセージボックスに表示します。

```
var folderURI = "file:///c:/temp";
var fileMask = "*.txt";
var list = FLfile.listFolder(folderURI + "/" + fileMask, "files");
if (list) {
    alert(folderURI + " contains: " + list.join(" "));
}
```

次の例では、指定した *folderURI* でファイルマスクを使用して、Windows アプリケーションフォルダにあるすべての実行可能ファイルの名前を返します。

```
var executables = FLfile.listFolder("file:///C:/WINDOWS/*.exe", "files")
alert(executables.join("\n"))
```

FLfile.read()

使用できるバージョン

Flash MX 2004 7.2

シンタックス

FLfile.read()

パラメータ

fileOrFolderURI 属性を取得するファイルまたはフォルダを指定するストリング。file:/// URI で表します。

戻り値

指定したファイルの内容をストリングで返します。読み取りに失敗した場合は null を返します。

説明

メソッド。指定したファイルの内容をストリングで返します。読み取りに失敗した場合は null を返します。

例

次の例では、"mydata.txt" ファイルを読み取り、読み取りに成功した場合は、ファイルの内容をメッセージボックスに表示します。

```
var fileURI = "file:///c:/temp/mydata.txt";
var str = FLfile.read( fileURI);
if (str) {
    alert( fileURL + " contains: " + str);
}
```

次の例では、クラスファイルから **ActionScript** コードを読み取り、code 変数に格納します。

```
var classFileURI = "file:///C:/MyApplication/TextCarousel.as";
var code = FLfile.read(classFileURI);
```

FLfile.remove()

使用できるバージョン

Flash MX 2004 7.2

シンタックス

```
FLfile.remove(fileOrFolderURI)
```

パラメータ

fileOrFolderURI 削除するファイルまたはフォルダを指定するストリング。file:/// URI で表します。

戻り値

ブール値。成功した場合は true を返します。それ以外の場合は false を返します。

説明

メソッド。指定したファイルまたはフォルダを削除します。フォルダにファイルが含まれる場合は、そのファイルも削除されます。R(読み取り専用)属性を持つファイルは削除できません。

例

次の例では、ファイルが存在する場合にユーザーに通知し、ユーザーが削除するよう指定した場合に、そのファイルを削除します。

```
var fileURI = prompt ("Enter file/folder to be deleted: ", "file:///c:/temp/delete.txt");
if (FLfile.exists(fileURI)) {
    var confirm = prompt("File exists. Delete it? (y/n)", "y");
    if (confirm == "y" || confirm == "Y") {
        if(FLfile.remove(fileURI)) {
            alert(fileURI + " is deleted.");
        }
        else {
            alert("fail to delete " + fileURI);
        }
    }
}
else {
    alert(fileURI + " does not exist");
}
```

次の例では、アプリケーションで作成された設定ファイルを削除します。

```
if(FLfile.remove("file:///C:/MyApplication/config.ini")) {
    alert("Configuration file deleted");
}
```

次の例では、"Configuration" フォルダとその内容を削除します。

```
FLfile.remove("file:///C:/MyApplication/Configuration/");
```

関連項目

[FLfile.createFolder\(\)](#), [FLfile.getAttributes\(\)](#)

FLfile.setAttributes()

使用できるバージョン

Flash MX 2004 7.2

シンタックス

```
FLfile.setAttributes(fileURI, strAttrs)
```

パラメータ

fileURI 属性を設定するファイルを指定するストリング。file:/// URI で表します。

strAttrs 設定する属性の値を指定するストリング。*strAttrs* に指定できる値については、説明を参照してください。

戻り値

ブール値。成功した場合は true を返します。



ファイルやフォルダが存在しない場合は、予期しない結果が生じる可能性があります。このメソッドを使用する前に、`FLfile.exists()` を使用する必要があります。

説明

メソッド。指定したファイルにシステムレベルの属性を指定します。

`strAttrs` には次の値を指定できます。

N – 特定の属性なし (読み取り専用でない、非表示でないなど)

A – アーカイブ可能 (Windows のみ)

R – 読み取り専用 (Macintosh の場合、"ロック" の意味)

W – 書き込み可能 (R より優先)

H – 非表示 (Windows のみ)

V – 表示 (H より優先、Windows のみ)

`strAttrs` に R と W の両方を指定した場合、R は無視され、ファイルは書き込み可能に設定されます。同様に、H と V の両方を指定した場合、H は無視され、ファイルは表示に設定されます。

アーカイブ属性が未設定であることを確認する場合は、属性の設定前にこのコマンドを N パラメータ付きで使用してください。つまり、アーカイブ属性をオフにする A に直接対応するものではありません。

例

次の例では、"mydata.txt" ファイルを読み取り専用と非表示に設定します。アーカイブ属性には影響を与えません。

```
var URI = "file:///c:/temp/mydata.txt";
if (FLfile.exists(URI)) {
  FLfile.setAttributes(URI, "RH");
}
```

次の例では、"mydata.txt" ファイルを読み取り専用と非表示に設定します。アーカイブ属性も確実に未設定にします。

```
var URI = "file:///c:/temp/mydata.txt";

if (FLfile.exists(URI)) {
  FLfile.setAttributes(URI, "N");
  FLfile.setAttributes(URI, "RH");
}
```

関連項目

[FLfile.getAttributes\(\)](#)

FLfile.write()

使用できるバージョン

Flash MX 2004 7.2

シンタックス

```
FLfile.write(fileURI, textToWrite, [ strAppendMode ])
```

パラメータ

fileURI 書き込むファイルを指定するストリング。file:/// URI で表します。

textToWrite ファイルに書き込むテキストを表すストリング。

strAppendMode 値 "append" を持つ省略可能なストリング。これにより、既存ファイルに *textToWrite* を追加することを指定します。省略した場合、*fileURI* は *textToWrite* で上書きされます。

戻り値

ブール値。成功した場合は true を返します。それ以外の場合は false を返します。

説明

メソッド。指定したストリングを、指定したファイルに UTF-8 で書き込みます。指定したファイルが存在しない場合は、ファイルが作成されます。ただし、このメソッドを使用する前に、ファイルを配置するフォルダが存在している必要があります。フォルダを作成するには、[FLfile.createFolder\(\)](#) を使用します。

例

次の例では、ストリング "xxx" を "mydata.txt" ファイルに書き込み、書き込みに成功した場合にメッセージを表示します。次に、ストリング "aaa" をファイルに追加しようと試みます。書き込みに成功した場合にはメッセージを表示します。このスクリプトを実行した後、"mydata.txt" ファイルの内容は "xxxxaaa" だけになります。

```
var URI = "file:///c:/temp/mydata.txt";
if (FLfile.write(URI, "xxx")) {
    alert("Wrote xxx to " + URI);
}
if (FLfile.write(URI, "aaa", "append")) {
    alert("Appended aaa to " + fileURI);
}
```

関連項目

[FLfile.createFolder\(\)](#), [FLfile.exists\(\)](#)

folderItem オブジェクト

継承 [Item オブジェクト](#) > folderItem オブジェクト

使用できるバージョン

Flash MX 2004

説明

folderItem オブジェクトは、Item オブジェクトのサブクラスです。folderItem に固有のメソッドやプロパティはありません。詳細については、[Item オブジェクト](#)を参照してください。

fontItem オブジェクト

継承 [Item オブジェクト](#) > fontItem オブジェクト

使用できるバージョン

Flash MX 2004

説明

fontItem オブジェクトは、Item オブジェクトのサブクラスです。fontItem に固有のメソッドやプロパティはありません。詳細については、[Item オブジェクト](#)を参照してください。

Frame オブジェクト

使用できるバージョン

Flash MX 2004

説明

Frame オブジェクトは、レイヤーのフレームを表します。

Frame オブジェクトのメソッド一覧

Frame オブジェクトでは次のメソッドを使用できます。

メソッド	説明
<code>frame.getCustomEase()</code>	JavaScript のオブジェクトの配列を返します。それぞれ x プロパティと y プロパティがあります。
<code>frame.setCustomEase()</code>	カスタムイーasing曲線として三次ベジェ曲線を使用することを指定します。

Frame オブジェクトのプロパティ一覧

Frame オブジェクトでは次のプロパティを使用できます。

プロパティ	説明
<code>frame.actionScript</code>	ActionScript コードを表す文字列。
<code>frame.duration</code>	読み取り専用。フレームシーケンスのフレーム数を表す整数です。
<code>frame.elements</code>	読み取り専用。Element オブジェクト (Element オブジェクト を参照) の配列です。
<code>frame.hasCustomEase</code>	カスタムイーasing曲線からフレームのイーasing情報を取得するかどうかを指定するブール値。
<code>frame.labelType</code>	フレーム名の種類を指定する文字列。
<code>frame.motionTweenOrientToPath</code>	トゥイーンされたエレメントがパスに沿って移動するときに、パス上の各点を基準にエレメントの角度を維持するためにエレメントを回転するかどうかを指定するブール値。
<code>frame.motionTweenRotate</code>	トゥイーンされたエレメントをどのように回転するかを指定する文字列。
<code>frame.motionTweenRotateTimes</code>	開始キーフレームと次のキーフレーム間でトゥイーンされたエレメントを回転する数を指定する整数。

プロパティ	説明
<code>frame.motionTweenScale</code>	ブール値。トゥイーンされたエレメントを、後続のキーフレーム内にあるオブジェクトのサイズに拡大 / 縮小するかどうかを指定します。トゥイーン内の各フレームでサイズを拡大する場合は <code>true</code> を指定し、拡大 / 縮小しない場合は <code>false</code> を指定します。
<code>frame.motionTweenSnap</code>	ブール値。トゥイーンされたエレメントを、このフレームのレイヤーに関連付けられているモーションガイドレイヤーで最も近くにあるポイントに自動的に吸着するかどうかを指定します。自動的に吸着する場合は <code>true</code> を指定し、吸着しない場合は <code>false</code> を指定します。
<code>frame.motionTweenSync</code>	ブール値。 <code>true</code> に設定すると、トゥイーンされたオブジェクトのアニメーションをメインタイムラインと同期します。
<code>frame.name</code>	フレームの名前を指定するストリング。
<code>frame.shapeTweenBlend</code>	トゥイーンの最初にあるキーフレームのシェイプと、次のキーフレームのシェイプとの間で、シェイプトゥイーンをどのようにブレンドするかを指定するストリング。
<code>frame.soundEffect</code>	フレームに直接割り当てられているサウンド (<code>frame.soundLibraryItem</code>) のエフェクトを指定するストリング。
<code>frame.soundLibraryItem</code>	サウンドの作成に使用するライブラリアイテム (SoundItem オブジェクト を参照)。
<code>frame.soundLoop</code>	フレームに直接追加されているサウンド (<code>frame.soundLibraryItem</code>) の再生回数を指定する整数値。
<code>frame.soundLoopMode</code>	フレームに直接割り当てられているサウンド (<code>frame.soundLibraryItem</code>) の再生回数を指定するか、無限にループさせるかを指定するストリング。
<code>frame.soundName</code>	フレームに直接割り当てられているサウンド (<code>frame.soundLibraryItem</code>) の名前に、ライブラリに格納されているのと同じ名前を指定するストリング。
<code>frame.soundSync</code>	フレームに直接割り当てられているサウンド (<code>frame.soundLibraryItem</code>) の同期の動作を指定するストリング。
<code>frame.startFrame</code>	読み取り専用。シーケンス内の最初のフレームのインデックスです。
<code>frame.tweenEasing</code>	トゥイーンオブジェクトに適用するイー징ング量を指定する整数。

プロパティ	説明
<code>frame.tweenType</code>	トゥイーンの種類を指定するストリング。
<code>frame.useSingleEaseCurve</code>	単一カスタムイーasing曲線をすべてのプロパティのイーasing情報に使用するかどうかを指定するブール値。

frame.actionScript

使用できるバージョン

Flash MX 2004

シンタックス

`frame.actionScript`

説明

プロパティ。ActionScript コードを表すストリングです。改行文字を挿入するには、"`\n`" を使用します。

例

次の例では、`stop()` を最初のフレームの1番上のレイヤーのアクションに割り当てます。

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].actionScript = 'stop();';
```

frame.duration

使用できるバージョン

Flash MX 2004

シンタックス

`frame.duration`

説明

読み取り専用プロパティ。フレームシーケンスのフレーム数を表す整数です。

例

次の例では、最初のレイヤーの最初のフレームから開始するフレームシーケンス内のフレーム数を `frameSpan` 変数に格納します。

```
var frameSpan = fl.getDocumentDOM().getTimeline().layers[0].frames[0].duration;
```

frame.elements

使用できるバージョン

Flash MX 2004

シンタックス

`frame.elements`

説明

読み取り専用プロパティ。Element オブジェクト ([Element オブジェクト](#)を参照) の配列です。エレメントの順序は、FLA ファイルに保存されている順序になります。ステージに複数のシェイプがあり、各シェイプがグループ化されていない場合は、1つのエレメントとして扱われます。各シェイプがグループ化されていて、ステージに複数のグループがある場合は、別個のエレメントとして扱われます。つまり、ステージに個別のシェイプがいくつあるかに関係なく、未処理のグループ化されていないシェイプは単一エレメントとして扱われます。たとえば、未処理のグループ化されていないシェイプがフレームに3つある場合、このフレームの `elements.length` は1を返します。この問題を回避するには、各シェイプを個別に選択して、グループ化してください。

例

次の例では、上側のレイヤーの最初のフレームにある現在のエレメントの配列を `myElements` 変数に格納します。

```
var myElements = fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements;
```

frame.getCustomEase()

使用できるバージョン

Flash 8

シンタックス

`Frame.getCustomEase([property])`

パラメータ

property カスタムイージング値を返すプロパティを指定する省略可能なストリング。指定できる値は、"all"、"position"、"rotation"、"scale"、"color"、および "filters" です。デフォルト値は "all" です。

戻り値

JavaScript のオブジェクトの配列を返します。それぞれ *x* プロパティと *y* プロパティがあります。

説明

メソッド。イージング曲線を定義する三次ベジェ曲線のコントロールポイントを表すオブジェクトの配列を返します。

例

次の例では、上側のレイヤーの最初のフレームにある `position` プロパティのカスタムイーasing値を返します。

```
var theFrame = fl.getDocumentDOM().getTimeline().layers[0].frames[0]
var easeArray = theFrame.getCustomEase("position");
```

関連項目

[frame.hasCustomEase](#), [frame.setCustomEase\(\)](#), [frame.useSingleEaseCurve](#)

frame.hasCustomEase

使用できるバージョン

Flash 8

シンタックス

`frame.hasCustomEase`

説明

プロパティ。ブール値です。true の場合、カスタムイーasing曲線からフレームのイーasing情報を取得します。false の場合、イーasing値からフレームのイーasing情報を取得します。

例

次の例では、上側のレイヤーの最初のフレームで、カスタムイーasing曲線からではなく、イーasing値からフレームのイーasing情報を取得するよう指定します。

```
var theFrame = fl.getDocumentDOM().getTimeline().layers[0].frames[0]
theFrame.hasCustomEase = false;
```

関連項目

[frame.getCustomEase\(\)](#), [frame.setCustomEase\(\)](#), [frame.useSingleEaseCurve](#)

frame.labelType

使用できるバージョン

Flash MX 2004

シンタックス

`frame.labelType`

説明

プロパティ。フレーム名の種類を指定する文字列です。指定できる値は、"none"、"name"、"comment"、および "anchor" です。ラベルに "none" を設定すると、`frame.name` プロパティがクリアされます。

例

次の例では、上側のレイヤーの最初のフレームの名前を "First Frame" に設定し、ラベルを "comment" に設定します。

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].name = 'First Frame';  
fl.getDocumentDOM().getTimeline().layers[0].frames[0].labelType = 'comment';
```

frame.motionTweenOrientToPath

使用できるバージョン

Flash MX 2004

シンタックス

`frame.motionTweenOrientToPath`

説明

プロパティ。ブール値です。トゥイーンされたエレメントが、パスに沿って移動するときに、パス上の各点を基準にエレメントの角度を維持するためにエレメントを回転するか (true)、または回転しないか (false) を指定します。

このプロパティの値を指定する場合は、`frame.motionTweenRotate` を "none" に設定する必要があります。

frame.motionTweenRotate

使用できるバージョン

Flash MX 2004

シンタックス

`frame.motionTweenRotate`

説明

プロパティ。トゥイーンされたエレメントをどのように回転するかを指定する文字列です。指定できる値は、"none"、"auto"、"clockwise"、および "counter-clockwise" です。"auto" を指定すると、後続のキーフレームにあるオブジェクトの回転と一致するように、最小限必要な方向にオブジェクトが回転します。

`frame.motionTweenOrientToPath` の値を指定する場合は、このプロパティを "none" に設定する必要があります。

例

詳細については、`frame.motionTweenRotateTimes` を参照してください。

frame.motionTweenRotateTimes

使用できるバージョン

Flash MX 2004

シンタックス

`frame.motionTweenRotateTimes`

説明

プロパティ。開始キーフレームと次のキーフレーム間でトゥイーンされたエレメントを回転する数を指定する整数です。

例

次の例では、このフレームのエレメントを、後続のキーフレームに達するまで反時計回りに 3 回転します。

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].motionTweenRotate =  
    "counter-clockwise";  
fl.getDocumentDOM().getTimeline().layers[0].frames[0].motionTweenRotateTimes =  
    3;
```

frame.motionTweenScale

使用できるバージョン

Flash MX 2004

シンタックス

`frame.motionTweenScale`

説明

プロパティ。ブール値です。トゥイーンされたエレメントを、後続のキーフレーム内にあるオブジェクトのサイズに拡大・縮小するかどうかを指定します。トゥイーン内の各フレームでサイズを拡大する場合は true を指定し、拡大・縮小しない場合は false を指定します。

例

次の例では、トゥイーンされたエレメントのサイズをトゥイーン内の各フレームで拡大し、後続のキーフレーム内のオブジェクトのサイズに拡大します。

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].motionTweenScale = true;
```

frame.motionTweenSnap

使用できるバージョン

Flash MX 2004

シンタックス

```
frame.motionTweenSnap
```

説明

プロパティ。ブール値です。トゥイーンされたエレメントを、このフレームのレイヤーに関連付けられているモーションガイドレイヤーで最も近くにあるポイントに自動的に吸着するかどうかを指定します。自動的に吸着する場合は true を指定し、吸着しない場合は false を指定します。

frame.motionTweenSync

使用できるバージョン

Flash MX 2004

シンタックス

```
frame.motionTweenSync
```

説明

プロパティ。ブール値です。true に設定すると、トゥイーンされたオブジェクトのアニメーションをメインタイムラインと同期します。

例

次の例では、トゥイーンオブジェクトをタイムラインと同期するように指定します。

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].motionTweenSync = true;
```

frame.name

使用できるバージョン

Flash MX 2004

シンタックス

`frame.name`

説明

プロパティ。フレームの名前を指定するストリングです。

例

次の例では、上側のレイヤーの最初のフレームの名前を "First Frame" に設定し、name 値を frameLabel 変数に格納します。

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].name = 'First Frame';  
var frameLabel = fl.getDocumentDOM().getTimeline().layers[0].frames[0].name;
```

frame.setCustomEase()

使用できるバージョン

Flash 8

シンタックス

`frame.setCustomEase(property, easeCurve)`

パラメータ

property イージング曲線を使用するプロパティを指定するストリング。指定できる値は、"all"、"position"、"rotation"、"scale"、"color"、および "filters" です。

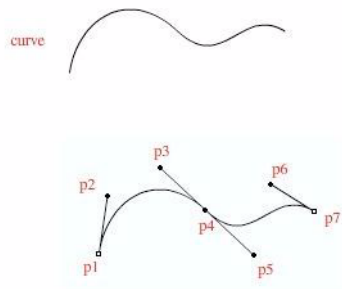
easeCurve イージング曲線を定義するオブジェクトの配列。各配列エレメントは、x プロパティと y プロパティを持つ JavaScript オブジェクトである必要があります。

戻り値

なし

説明

メソッド。カスタムイージング曲線として使用する三次ベジェ曲線を表す、コントロールポイントおよびタンジェントエンドポイントの座標の配列を指定します。この配列は、コントロールポイントおよびタンジェントエンドポイントの水平位置(左から右の順番)で構成されます。たとえば、次の図は、*easeCurve* 配列が *p1*~*p7* という7つの点の値を含む場合に作成されるイージング曲線を示しています。



例

次の例では、最初のレイヤーの最初のフレームにあるすべてのプロパティのイージング曲線に、*myCurve* 配列に格納されたコントロールポイントおよびタンジェントエンドポイントによって指定されるベジェ曲線を設定します。

```
var theFrame = fl.getDocumentDOM().getTimeline().layers[0].frames[0]
var myCurve = [{x:100, y:200},{x:200, y:100}, {x:10, y:0}]
theFrame.setCustomEase("all", myCurve);
```

関連項目

[frame.getCustomEase\(\)](#), [frame.hasCustomEase](#), [frame.useSingleEaseCurve](#)

frame.shapeTweenBlend

使用できるバージョン

Flash MX 2004

シンタックス

`frame.shapeTweenBlend`

説明

プロパティ。トゥイーンの最初にあるキーフレームのシェイプと、次のキーフレームのシェイプとの間で、シェイプトゥイーンをどのようにブレンドするかを指定するストリングです。指定できる値は、*"distributive"* と *"angular"* です。

frame.soundEffect

使用できるバージョン

Flash MX 2004

シンタックス

`frame.soundEffect`

説明

プロパティ。フレームに直接割り当てられているサウンド ([frame.soundLibraryItem](#)) のエフェクトを指定するストリングです。指定できる値は、"none"、"left channel"、"right channel"、"fade left to right"、"fade right to left"、"fade in"、"fade out"、および "custom" です。

例

次の例では、最初のフレームに割り当てられているサウンドをフェードインすることを指定します。

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].soundEffect = "fade in";
```

frame.soundLibraryItem

使用できるバージョン

Flash MX 2004

シンタックス

`frame.soundLibraryItem`

説明

プロパティ。サウンドの作成に使用するライブラリアイテム ([SoundItem オブジェクト](#) を参照) です。サウンドは直接フレームに割り当てられます。

例

次の例では、ライブラリの最初のアイテムを最初のフレームの `soundLibraryItem` プロパティに割り当てます。

```
// ライブラリの最初のアイテムはサウンドオブジェクトである必要があります。  
fl.getDocumentDOM().getTimeline().layers[0].frames[0].soundLibraryItem  
  =fl.getDocumentDOM().library.items[0];
```

frame.soundLoop

使用できるバージョン

Flash MX 2004

シンタックス

`frame.soundLoop`

説明

プロパティ。フレームに直接追加されているサウンド ([frame.soundLibraryItem](#)) の再生回数を指定する整数値です。このプロパティの値を指定する場合は、[frame.soundLoopMode](#) を "repeat" に設定する必要があります。

例

詳細については、[frame.soundLoopMode](#) を参照してください。

frame.soundLoopMode

使用できるバージョン

Flash MX 2004

シンタックス

frame.soundLoopMode

説明

プロパティ。フレームに直接割り当てられているサウンド ([frame.soundLibraryItem](#)) の再生回数を指定するか、無限にループさせるかを指定するストリングです。指定できる値は、"repeat" と "loop" です。サウンドの再生回数を指定するには、[frame.soundLoop](#) の値を設定します。

例

次の例では、サウンドの再生回数を 2 回に指定します。

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].soundLoopMode = "repeat";  
fl.getDocumentDOM().getTimeline().layers[0].frames[0].soundLoop = 2;
```

frame.soundName

使用できるバージョン

Flash MX 2004

シンタックス

frame.soundName

説明

プロパティ。フレームに直接割り当てられているサウンド ([frame.soundLibraryItem](#)) の名前に、ライブラリに格納されているのと同じ名前を指定するストリングです。

例

次の例では、最初のフレームの soundName プロパティを "song1.mp3" に変更します。song1.mp3 はライブラリに存在する必要があります。

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].soundName = "song1.mp3";
```

frame.soundSync

使用できるバージョン

Flash MX 2004

シンタックス

`frame.soundSync`

説明

プロパティ。フレームに直接割り当てられているサウンド ([frame.soundLibraryItem](#)) の同期の動作を指定する文字列です。指定できる値は、"event"、"stop"、"start"、および "stream" です。

例

次の例では、サウンドをストリーミングすることを指定します。

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].soundSync = 'stream';
```

frame.startFrame

使用できるバージョン

Flash MX 2004

シンタックス

`frame.startFrame`

説明

読み取り専用プロパティ。シーケンス内の最初のフレームのインデックスです。

例

次の例では、`stFrame` がフレームシーケンスの最初のフレームのインデックスです。この例では、フレームシーケンスの範囲はフレーム 5 からフレーム 10 までの 6 つのフレームです。したがって、フレーム 5 からフレーム 10 の間にあるフレームの `stFrame` 値は 4 になります (インデックス値はフレーム番号値とは異なります)。

```
var stFrame = fl.getDocumentDOM().getTimeline().layers[0].frames[4].startFrame;  
fl.trace(stFrame); // 4  
var stFrame = fl.getDocumentDOM().getTimeline().layers[0].frames[9].startFrame;  
fl.trace(stFrame); // 4
```


frame.tweenEasing

使用できるバージョン

Flash MX 2004

シンタックス

`frame.tweenEasing`

説明

プロパティ。トゥイーンオブジェクトに適用するイーasing量を指定する整数です。指定できる値は、-100～100です。モーショントゥイーンを低速で開始し、アニメーションの終わりに向けて加速するには、-1～-100の値を使用します。モーショントゥイーンを高速で開始し、アニメーションの終わりに向けて減速するには、1～100の正の値を使用します。

例

次の例では、トゥイーンオブジェクトのモーションを高速で開始し、アニメーションの終わりに向けて減速するように指定します。

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].tweenEasing = 50;
```

frame.tweenType

使用できるバージョン

Flash MX 2004

シンタックス

`frame.tweenType`

説明

プロパティ。トゥイーンの種類を指定するストリングです。指定できる値は "motion"、"shape"、または "none" です。"none" を指定すると、モーショントゥイーンが削除されます。モーショントゥイーンを作成するには、[timeline.createMotionTween\(\)](#) メソッドを使用します。

"motion" を指定する場合は、フレーム内のオブジェクトはシンボル、テキストフィールド、またはグループ化されたオブジェクトである必要があります。現在のキーフレームの位置から後続のキーフレームの位置までトゥイーンされます。

"shape" を指定する場合は、フレームのオブジェクトはシェイプである必要があります。現在のキーフレームのシェイプから後続のキーフレームのシェイプにブレンドされます。

例

次の例では、オブジェクトがモーショントゥイーンであることを指定します。これにより、オブジェクトは現在のキーフレームの位置から後続のキーフレームの位置までトゥイーンされます。

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].tweenType = "motion";
```

frame.useSingleEaseCurve

使用できるバージョン

Flash 8

シンタックス

`frame.useSingleEaseCurve`

説明

プロパティ。ブール値です。true の場合、単一カスタムイーザリング曲線が、すべてのプロパティのイーザリング情報に使用されます。false の場合、各プロパティに個別のイーザリング曲線が使用されます。

フレームにカスタムイーザリングが適用されない場合、このプロパティは無視されます。

例

次の例では、単一カスタムイーザリング曲線を、最初のレイヤーの最初のフレームにあるすべてのプロパティで使用することを指定します。

```
var theFrame = fl.getDocumentDOM().getTimeline().layers[0].frames[0]
theFrame.useSingleEaseCurve = true;
```

関連項目

[frame.getCustomEase\(\)](#), [frame.hasCustomEase](#), [frame.setCustomEase\(\)](#)

HalfEdge オブジェクト

使用できるバージョン

Flash MX 2004

説明

HalfEdge オブジェクトは、[Shape オブジェクト](#)のエッジの指定された側です。エッジには 2 つのハーフエッジがあります。これらのハーフエッジをたどることで、シェイプの輪郭を横切ることができます。たとえば、1 つのハーフエッジを基点として、シェイプの輪郭の周囲のすべてのハーフエッジをトレースし、元のハーフエッジに戻ることができます。

ハーフエッジには順序があります。1 つのハーフエッジは、エッジの 1 つの側を表し、他のハーフエッジは、エッジの反対側を表します。

HalfEdge オブジェクトのメソッド一覧

HalfEdge オブジェクトでは次のメソッドを使用できます。

メソッド	説明
halfEdge.getEdge()	HalfEdge オブジェクトの Edge オブジェクト を取得します。
halfEdge.getNext()	現在の輪郭上の次のハーフエッジを取得します。
halfEdge.getOppositeHalfEdge()	エッジの反対側の HalfEdge オブジェクトを取得します。
halfEdge.getPrev()	現在の輪郭上の前の HalfEdge オブジェクトを取得します。
halfEdge.getVertex()	HalfEdge オブジェクトの先頭にある Vertex オブジェクト を取得します。

HalfEdge オブジェクトのプロパティ一覧

HalfEdge オブジェクトでは次のプロパティを使用できます。

プロパティ	説明
halfEdge.id	読み取り専用。HalfEdge オブジェクトの一意的整数の識別子です。
halfEdge.index	親エッジにおけるこの HalfEdge オブジェクトのインデックスを指定する 0 または 1 の整数値です。

halfEdge.getEdge()

使用できるバージョン

Flash MX 2004

シンタックス

```
halfEdge.getEdge()
```

パラメータ

なし

戻り値

[Edge オブジェクト](#)。

説明

メソッド。HalfEdge オブジェクトの Edge オブジェクトを取得します。[Edge オブジェクト](#) を参照してください。

例

次の例では、指定されたシェイプのエッジとハーフエッジを取得する方法を示します。

```
var shape = fl.getDocumentDOM().selection[0];  
var hEdge = shape.edges[0].getHalfEdge(0);  
var edge = hEdge.getEdge();
```

halfEdge.getNext()

使用できるバージョン

Flash MX 2004

シンタックス

```
halfEdge.getNext()
```

パラメータ

なし

戻り値

HalfEdge オブジェクト

説明

メソッド。現在の輪郭上の次のハーフエッジを取得します。



ハーフエッジには方向と順序がありますが、エッジにはありません。

例

次の例では、指定された輪郭の次のハーフエッジを `nextHalfEdge` 変数に格納します。

```
var shape = fl.getDocumentDOM().selection[0];
var hEdge = shape.edges[0].getHalfEdge( 0 );
var nextHalfEdge = hEdge.getNext();
```

halfEdge.getOppositeHalfEdge()

使用できるバージョン

Flash MX 2004

シンタックス

```
halfEdge.getOppositeHalfEdge()
```

パラメータ

なし

戻り値

HalfEdge オブジェクト

説明

メソッド。エッジの反対側の HalfEdge オブジェクトを取得します。

例

次の例では、ハーフエッジの反対側の hEdge を `otherHalfEdge` 変数に格納します。

```
var shape = fl.getDocumentDOM().selection[0];
var hEdge = shape.edges[0].getHalfEdge(0);
var otherHalfEdge = hEdge.getOppositeHalfEdge();
```

halfEdge.getPrev()

使用できるバージョン

Flash MX 2004

シンタックス

```
halfEdge.getPrev()
```

パラメータ

なし

戻り値

HalfEdge オブジェクト

説明

メソッド。現在の輪郭上の前の HalfEdge オブジェクトを取得します。



ハーフエッジには方向と順序がありますが、エッジにはありません。

例

次の例では、指定された輪郭の前のハーフエッジを prevHalfEdge 変数に格納します。

```
var shape = fl.getDocumentDOM().selection[0];  
var hEdge = shape.edges[0].getHalfEdge( 0 );  
var prevHalfEdge = hEdge.getPrev();
```

halfEdge.getVertex()

使用できるバージョン

Flash MX 2004

シンタックス

```
halfEdge.getVertex()
```

パラメータ

なし

戻り値

[Vertex オブジェクト](#)

説明

メソッド。HalfEdge オブジェクトの先頭にある Vertex オブジェクトを取得します。[Vertex オブジェクト](#)を参照してください。

例

次の例では、hEdge の先頭にある Vertex オブジェクトを vertex 変数に格納します。

```
var shape = fl.getDocumentDOM().selection[0];
var edge = shape.edges[0];
var hEdge = edge.getHalfEdge(0);
var vertex = hEdge.getVertex();
```

halfEdge.id

使用できるバージョン

Flash MX 2004

シンタックス

halfEdge.id

説明

読み取り専用プロパティ。HalfEdge オブジェクトの一意の整数の識別子です。

例

次の例では、指定されたハーフエッジの一意の識別子を [出力] パネルに表示します。

```
var shape = fl.getDocumentDOM().selection[0];
alert(shape.contours[0].getHalfEdge().id);
```

halfEdge.index

使用できるバージョン

Flash MX 2004

シンタックス

halfEdge.index

説明

読み取り専用プロパティ。親エッジにおけるこの HalfEdge オブジェクトのインデックスを指定する 0 または 1 の整数値です。

例

次の例では、指定されたハーフエッジのインデックス値を [出力] パネルに表示します。

```
var shape = fl.getDocumentDOM().selection[0];  
var hEdge = shape.edges[0].getHalfEdge(0);  
var heIndex = hEdge.index;
```


Instance オブジェクト

継承 [Element オブジェクト](#) > Instance オブジェクト

使用できるバージョン

Flash MX 2004

説明

Instance は [Element オブジェクト](#) のサブクラスです。

Instance オブジェクトのプロパティ一覧

Instance には、[Element オブジェクト](#) のすべてのプロパティの他に、次のプロパティがあります。

プロパティ	説明
<code>instance.instanceType</code>	読み取り専用。インスタンスのタイプを表すストリングです。
<code>instance.libraryItem</code>	このインスタンスをインスタンス化するために使用されるライブラリアイテムです。

instance.instanceType

使用できるバージョン

Flash MX 2004。Flash 8 では、指定できる値に "video" が追加されました。

シンタックス

`instance.instanceType`

説明

読み取り専用プロパティ。インスタンスのタイプを表すストリングです。指定できる値は、"symbol"、"bitmap"、"embedded video"、"linked video"、"video"、および "compiled clip" です。

✕
例

Flash MX 2004 では、`library.addItem("video")` を使用してライブラリに追加したアイテムの `instance.instanceType` の値は、`"embedded_video"` となります。Flash 8 では、この値は `"video"` となります。詳細については、[library.addItem\(\)](#) を参照してください。

例

次の例は、ムービークリップのインスタンスタイプが "symbol" であることを示しています。

```
// ムービークリップを選択してから、このスクリプトを実行します。  
var type = fl.getDocumentDOM().selection[0].instanceType;  
fl.trace("This instance type is " + type);
```

instance.libraryItem

使用できるバージョン

Flash MX 2004

シンタックス

`instance.libraryItem`

説明

プロパティ。このインスタンスをインスタンス化するために使用されるライブラリアイテムです。このプロパティは、同じタイプの別のライブラリアイテムにのみ変更できます(つまり、symbol インスタンスがビットマップを参照するように設定することはできません)。 [library オブジェクト](#) を参照してください。

例

次の例では、選択したシンボルがライブラリ内の最初のアイテムを参照するように変更します。

```
fl.getDocumentDOM().selection[0].libraryItem =  
    fl.getDocumentDOM().library.items[0];
```

Item オブジェクト

使用できるバージョン

Flash MX 2004

説明

Item オブジェクトは、抽象基本クラスです。ライブラリ内のすべてのものは Item から継承します。[library オブジェクト](#)も参照してください。

Item オブジェクトのメソッド一覧

Item オブジェクトでは次のメソッドを使用できます。

メソッド	説明
<code>item.addData()</code>	指定されたデータをライブラリアイテムに追加します。
<code>item.getData()</code>	指定したデータの値を取得します。
<code>item.hasData()</code>	ライブラリアイテムに指定されたデータが含まれているかどうかを判断します。
<code>item.removeData()</code>	ライブラリアイテムから永続データを削除します。

Item オブジェクトのプロパティ一覧

Item オブジェクトでは次のプロパティを使用できます。

プロパティ	説明
<code>item.itemType</code>	読み取り専用。エレメントのタイプを指定するストリングです。
<code>item.linkageBaseClass</code>	シンボルに関連付けられる ActionScript 3.0 クラスを指定するストリング。
<code>item.linkageClassName</code>	シンボルに関連付けられる ActionScript 2.0 クラスを指定するストリング。
<code>item.linkageExportForAS</code>	ブール値。true の場合、アイテムは ActionScript 用に書き出されます。
<code>item.linkageExportForRS</code>	ブール値。true の場合、アイテムはランタイム共有用に書き出されます。
<code>item.linkageExportInFirstFrame</code>	ブール値。true の場合、アイテムは最初のフレームに書き出されます。

プロパティ	説明
<code>item.linkageIdentifier</code>	Flash がリンク先の SWF ファイルにリンクするときにアセットを識別するために使用する名前を指定するストリング。
<code>item.linkageImportForRS</code>	ブール値。true の場合、アイテムはランタイム共有用に読み込まれます。
<code>item.linkageURL</code>	共有アセットが格納されている SWF ファイルの URL を指定するストリング。
<code>item.name</code>	フォルダ構造が格納されているライブラリアイテムの名前を指定するストリング。

item.addData()

使用できるバージョン

Flash MX 2004

シンタックス

```
item.addData(name, type, data)
```

パラメータ

name データの名前を示すストリング。

type データ型を示すストリング。有効なデータ型は、"integer"、"integerArray"、"double"、"doubleArray"、"string"、および "byteArray" です。

data 指定されたライブラリアイテムに追加するデータ。データの型は、*type* パラメータの値に応じて異なります。たとえば、*type* が "integer" の場合、データの値は整数である必要があります。

戻り値

なし

説明

メソッド。指定されたデータをライブラリアイテムに追加します。

例

次の例では、整数値 12 が含まれる `myData` というデータをライブラリ内の最初のアイテムに追加します。

```
fl.getDocumentDOM().library.items[0].addData("myData", "integer", 12);
```

item.getData()

使用できるバージョン

Flash MX 2004

シンタックス

```
item.getData(name)
```

パラメータ

name 取得するデータの名前を示すストリング。

戻り値

name パラメータで指定されたデータです。返されるデータの型は保存されるデータの型によって異なります。

説明

メソッド。指定されたデータの値を取得します。

例

次の例では、ライブラリ内の最初のアイテムから **myData** というデータの値を取得し、そのデータを変数 **libData** に格納します。

```
var libData = fl.getDocumentDOM().library.items[0].getData("myData");
```

item.hasData()

使用できるバージョン

Flash MX 2004

シンタックス

```
item.hasData(name)
```

パラメータ

name ライブラリアイテム内でチェックするデータの名前を示すストリング。

戻り値

ブール値。指定されたデータが存在する場合は **true** を返します。それ以外の場合は **false** を返します。

説明

メソッド。ライブラリアイテムに指定されたデータが含まれているかどうかを判断します。

例

次の例では、ライブラリ内の最初のアイテムに myData というデータポイントが含まれている場合に、[出力] パネルにメッセージを表示します。

```
if (fl.getDocumentDOM().library.items[0].hasData("myData")){
    fl.trace("Yep, it's there!");
}
```

item.itemType

使用できるバージョン

Flash MX 2004

シンタックス

item.itemType

説明

読み取り専用プロパティ。エレメントのタイプを示す文字列です。値は、"undefined"、"component"、"movie clip"、"graphic"、"button"、"folder"、"font"、"sound"、"bitmap"、"compiled clip"、"screen"、または "video" のいずれかです。このプロパティが "video" の場合は、ビデオの種類を確認できます。[videoItem.videoType](#) を参照してください。

例

次の例では、指定されたライブラリアイテムのタイプを [出力] パネルに表示します。

```
fl.trace(fl.getDocumentDOM().library.items[0].itemType);
```

item.linkageBaseClass

使用できるバージョン

Flash CS3 Professional

シンタックス

item.linkageBaseClass

説明

プロパティ。シンボルに関連付けられる **ActionScript 3.0** クラスを指定する文字列です。ここで指定する値は、オーサリング環境の [リンケージ] ダイアログボックス、およびリンケージダイアログのコントロールを含むその他のダイアログボックス ([シンボルプロパティ] ダイアログボックスなど) で表示されます。**ActionScript 2.0** クラスでこの値を指定するには、[item.linkageClassName](#) を使用します。

基本クラスがシンボルタイプのデフォルトである場合 (たとえば、ムービークリップの "flash.display.MovieClip"、ボタンの "flash.display.SimpleButton" など)、このプロパティは空の文字列("")です。同様に、アイテムをデフォルトの基本クラスにするには、この値を空の文字列に設定します。

この値を設定するとき、[リンケージ] ダイアログボックスで実行されるようなチェックは行われず、Flash で基本クラスを指定された値に設定できない場合、エラーは発生しません。たとえば、[リンケージ] ダイアログボックスでこの値を設定すると、FLA ファイルのクラスパスで基本クラスが見つかるかどうかのチェックが実行されます。[パブリッシュ設定] ダイアログボックスの [Flash] タブで、ActionScript 3.0 が選択されていることなどが確認されます。このプロパティをスクリプトで設定する場合、これらのチェックは実行されません。

例

次のコード行では、このプロパティのいくつかの使用方法を示しています。

```
// ライブラリアイテムの基本クラスを "Sprite" に設定する
fl.getDocumentDOM().library.items[0].linkageBaseClass = "flash.display.Sprite";
// ライブラリアイテムの基本クラスをそのアイテムタイプのデフォルトに設定する
fl.getDocumentDOM().library.items[0].linkageBaseClass = "";
// ライブラリアイテムの基本クラスを検索して表示する
fl.trace(fl.getDocumentDOM().library.items[0].linkageBaseClass);
```

関連項目

[document.docClass](#)

item.linkageClassName

使用できるバージョン

Flash MX 2004

シンタックス

`item.linkageClassName`

説明

プロパティ。シンボルに関連付けられる ActionScript 2.0 クラスを指定する文字列です。ActionScript 3.0 クラスでこの値を指定するには、[item.linkageBaseClass](#) を使用します。

このプロパティを定義するには、[item.linkageExportForAS](#) プロパティまたは [item.linkageExportForRS](#) プロパティを true に設定し、[item.linkageImportForRS](#) プロパティを false に設定する必要があります。

例

次の例では、ライブラリ内の最初のアイテムに関連付けられる `ActionScript 2.0` クラス名が `myClass` であることを指定します。

```
fl.getDocumentDOM().library.items[0].linkageClassName = "myClass";
```

item.linkageExportForAS

使用できるバージョン

Flash MX 2004

シンタックス

```
item.linkageExportForAS
```

説明

プロパティ。ブール値です。このプロパティが `true` の場合、アイテムは `ActionScript` 用書き出されます。`item.linkageExportForAS` プロパティと `item.linkageExportInFirstFrame` プロパティを `true` に設定することもできます。

このプロパティを `true` に設定する場合、`item.linkageImportForRS` プロパティが `false` に設定されている必要があります。さらに、識別子 (`item.linkageIdentifier`) と URL (`item.linkageURL`) を指定する必要があります。

例

次の例では、指定されたライブラリアイテムのこのプロパティを設定します。

```
fl.getDocumentDOM().library.items[0].linkageExportForAS = true;
```

item.linkageExportForRS

使用できるバージョン

Flash MX 2004

シンタックス

```
item.linkageExportForRS
```

説明

プロパティ。ブール値です。このプロパティが `true` の場合、ランタイム共有用書き出されます。`item.linkageExportForAS` プロパティと `item.linkageExportInFirstFrame` プロパティを `true` に設定することもできます。

このプロパティを `true` に設定する場合、`item.linkageImportForRS` プロパティが `false` に設定されている必要があります。さらに、識別子 (`item.linkageIdentifier`) と URL (`item.linkageURL`) を指定する必要があります。

例

次の例では、指定されたライブラリアイテムのこのプロパティを設定します。

```
fl.getDocumentDOM().library.items[0].linkageExportForRS = true;
```

item.linkageExportInFirstFrame

使用できるバージョン

Flash MX 2004

シンタックス

```
item.linkageExportInFirstFrame
```

説明

プロパティ。ブール値です。`true` の場合、アイテムは最初のフレームに書き出されます。`false` の場合、アイテムは最初のインスタンスの最初のフレームに書き出されます。アイテムがステージに表示されない場合、アイテムは書き出されません。

このプロパティは、`item.linkageExportForAS` または `item.linkageExportForRS` が `true` に設定されている場合のみ、`true` に設定できます。

例

次の例では、指定されたライブラリアイテムを最初のフレームに書き出すことを指定します。

```
fl.getDocumentDOM().library.items[0].linkageExportInFirstFrame = true;
```

item.linkageIdentifier

使用できるバージョン

Flash MX 2004

シンタックス

```
item.linkageIdentifier
```

説明

プロパティ。Flash がリンク先の SWF ファイルにリンクするときにアセットを識別するために使用する名前を指定するストリングです。`item.linkageImportForRS`、`item.linkageExportForAS`、および `item.linkageExportForRS` が `false` に設定されている場合、このプロパティは無視されます。逆に、これらのプロパティが `true` に設定されている場合、このプロパティを設定する必要があります。

例

次の例では、ライブラリアイテムが書き出し先の SWF ファイルにリンクされている場合にライブラリアイテムを識別するためにストリング my_mc が使用されることを指定します。

```
fl.getDocumentDOM().library.items[0].linkageIdentifier = "my_mc";
```

関連項目

[item.linkageURL](#)

item.linkageImportForRS

使用できるバージョン

Flash MX 2004

シンタックス

```
item.linkageImportForRS
```

説明

プロパティ。ブール値です。true の場合、アイテムはランタイム共有のために読み込まれます。このプロパティが true に設定されている場合、[item.linkageExportForAS](#) と [item.linkageExportForRS](#) の両方を false に設定する必要があります。さらに、識別子 ([item.linkageIdentifier](#)) と URL ([item.linkageURL](#)) を指定する必要があります。

例

次の例では、指定されたライブラリアイテムのこのプロパティを true に設定します。

```
fl.getDocumentDOM().library.items[0].linkageImportForRS = true;
```

item.linkageURL

使用できるバージョン

Flash MX 2004

シンタックス

```
item.linkageURL
```

説明

プロパティ。共有アセットが格納されている SWF ファイルの URL を指定するストリングです。
`item.linkageImportForRS`、`item.linkageExportForAS`、および
`item.linkageExportForRS` が `false` に設定されている場合、このプロパティは無視されます。
逆に、これらのプロパティが `true` に設定されている場合、このプロパティを設定する必要があります。
Web URL またはファイル名は、プラットフォームに依存する形式で (つまり、プラットフォームに応じてスラッシュ [/] またはバックスラッシュ [\] で) 指定できます。

例

次の例では、指定されたライブラリアイテムのリンケージ URL を指定します。

```
fl.getDocumentDOM().library.items[0].linkageURL = "theShareSWF.swf";
```

関連項目

`item.linkageIdentifier`

item.name

使用できるバージョン

Flash MX 2004

シンタックス

`item.name`

説明

メソッド。フォルダ構造が格納されているライブラリアイテムの名前を指定するストリングです。たとえば、`Symbol_1` が、`Folder_1` フォルダ内にある場合、`Symbol_1` の `name` プロパティは "`Folder_1/Symbol_1`" になります。

例

次の例では、指定されたライブラリアイテムの名前を [出力] パネルに表示します。

```
fl.trace(fl.getDocumentDOM().library.items[0].name);
```

item.removeData()

使用できるバージョン

Flash MX 2004

シンタックス

`item.removeData(name)`

パラメータ

name ライブラリアイテムから削除するデータの名前を指定します。

戻り値

なし

説明

プロパティ。ライブラリアイテムから永続データを削除します。

例

次の例では、ライブラリ内の最初のアイテムから **myData** というデータを削除します。

```
fl.getDocumentDOM().library.items[0].removeData("myData");
```

Layer オブジェクト

使用できるバージョン

Flash MX 2004

説明

Layer オブジェクトは、タイムラインにあるレイヤーを表します。`timeline.layers` プロパティは、Layer オブジェクトの配列を含んでいます。このプロパティには、`fl.getDocumentDOM().getTimeline().layers` によってアクセスできます。

Layer オブジェクトのプロパティ一覧

Layer オブジェクトでは次のプロパティを使用できます。

プロパティ	説明
<code>layer.color</code>	レイヤーのアウトラインに割り当てられるカラーを指定する、ストリング、16 進数値、または整数。
<code>layer.frameCount</code>	読み取り専用。レイヤー内のフレームの数を指定する整数です。
<code>layer.frames</code>	読み取り専用。Frame オブジェクトの配列です。
<code>layer.height</code>	レイヤーの高さのパーセント値を示す整数。[レイヤープロパティ] ダイアログボックスのレイヤーの高さの値と同じです。
<code>layer.layerType</code>	レイヤーの現在の使用方法を示すストリング。[レイヤープロパティ] ダイアログボックスの [タイプ] の設定と同じです。
<code>layer.locked</code>	レイヤーのロックステータスを示すブール値。
<code>layer.name</code>	レイヤーの名前を指定するストリング。
<code>layer.outline</code>	レイヤー内のすべてのオブジェクトのアウトラインのステータスを示すブール値。
<code>layer.parentLayer</code>	レイヤーが含まれるフォルダ、ガイド、またはマスクレイヤーを表す Layer オブジェクト。
<code>layer.visible</code>	レイヤーのオブジェクトがステージ上に表示されるかどうかを指定するブール値。

layer.color

使用できるバージョン

Flash MX 2004

シンタックス

layer.color

説明

プロパティ。レイヤーのアウトラインに割り当てられるカラー。次のいずれかの形式です。

- "#RRGGBB" または "#RRGGBBAA" 形式のストリング
- 0xRRGGBB 形式の 16 進数
- 16 進数と等しい 10 進表現の整数

このプロパティは、[レイヤープロパティ] ダイアログボックスの [アウトラインカラー] の設定と同じです。

例

次の例では、最初のレイヤーの値を colorValue 変数に格納します。

```
var colorValue = fl.getDocumentDOM().getTimeline().layers[0].color;
```

次の例では、最初のレイヤーのカラーを赤に設定する 3 つの方法を示します。

```
fl.getDocumentDOM().getTimeline().layers[0].color=16711680;  
fl.getDocumentDOM().getTimeline().layers[0].color="#ff0000";  
fl.getDocumentDOM().getTimeline().layers[0].color=0xFF0000;
```

layer.frameCount

使用できるバージョン

Flash MX 2004

シンタックス

layer.frameCount

説明

読み取り専用プロパティ。レイヤー内のフレームの数を指定する整数です。

例

次の例では、最初のレイヤー内のフレームの数を fcNum 変数に格納します。

```
var fcNum = fl.getDocumentDOM().getTimeline().layers[0].frameCount;
```

layer.frames

使用できるバージョン

Flash MX 2004

シンタックス

layer.frames

説明

読み取り専用プロパティ。Frame オブジェクトの配列です。[Frame オブジェクト](#) を参照してください。

例

次の例では、frameArray 変数を、現在のドキュメント内のフレームの Frame オブジェクトの配列に設定します。

```
var frameArray = fl.getDocumentDOM().getTimeline().layers[0].frames;
```

フレームがキーフレームかどうかを調べるには、次の例に示すように、`frame.startFrame` プロパティが配列のインデックスと一致しているかどうかを確認します。

```
var frameArray = fl.getDocumentDOM().getTimeline().layers[0].frames;
var n = frameArray.length;
for (i=0; i<n; i++) {
    if (i==frameArray[i].startFrame) {
        alert("Keyframe at: " + i);
    }
}
```

layer.height

使用できるバージョン

Flash MX 2004

シンタックス

layer.height

説明

プロパティ。レイヤーの高さのパーセント値を示す整数です。[レイヤープロパティ] ダイアログボックスのレイヤーの高さの値と同じです。指定できる値は、デフォルトの高さに対するパーセント値 (100、200、または 300) を表します。

例

次の例では、最初のレイヤーの高さの設定のパーセント値を格納します。

```
var layerHeight = fl.getDocumentDOM().getTimeline().layers[0].height;
```

次の例では、最初のレイヤーの高さを 300% に設定します。

```
fl.getDocumentDOM().getTimeline().layers[0].height = 300;
```

layer.layerType

使用できるバージョン

Flash MX 2004

シンタックス

layer.layerType

説明

プロパティ。レイヤーの現在の使用方法を示すストリングです。[レイヤープロパティ] ダイアログボックスの [タイプ] の設定と同じです。指定できる値は、"normal"、"guide"、"guided"、"mask"、"masked"、および "folder" です。

例

次の例では、タイムライン内の最初のレイヤーのタイプを "folder" に設定します。

```
fl.getDocumentDOM().getTimeline().layers[0].layerType = "folder";
```

layer.locked

使用できるバージョン

Flash MX 2004

シンタックス

layer.locked

説明

プロパティ。レイヤーのロックステータスを示すブール値です。true に設定した場合、レイヤーはロックされます。デフォルト値は false です。

例

次の例では、最初のレイヤーのステータスのブール値を lockStatus 変数に格納します。

```
var lockStatus = fl.getDocumentDOM().getTimeline().layers[0].locked;
```

次の例では、最初のレイヤーのステータスをロック解除に設定します。

```
fl.getDocumentDOM().getTimeline().layers[0].locked = false;
```


layer.name

使用できるバージョン

Flash MX 2004

シンタックス

layer.name

説明

プロパティ。レイヤーの名前を指定するストリングです。

例

次の例では、現在のドキュメント内の最初のレイヤーの名前を“foreground”に設定します。

```
fl.getDocumentDOM().getTimeline().layers[0].name = "foreground";
```

layer.outline

使用できるバージョン

Flash MX 2004

シンタックス

layer.outline

説明

プロパティ。レイヤー内のすべてのオブジェクトのアウトラインのステータスを示すブール値です。
true に設定されている場合、レイヤー内のすべてのオブジェクトのアウトラインのみが表示されます。
false の場合、オブジェクトは作成されたとおりに表示されます。

例

次の例では、最初のレイヤー上のすべてのオブジェクトのアウトラインのみを表示します。

```
fl.getDocumentDOM().getTimeline().layers[0].outline = true;
```

layer.parentLayer

使用できるバージョン

Flash MX 2004

シンタックス

layer.parentLayer

説明

プロパティ。レイヤーが含まれるフォルダ、ガイド、またはマスクレイヤーを表す **Layer** オブジェクトです。親レイヤーは、レイヤーの前に位置するフォルダ、ガイド、またはマスクレイヤーか、前または次のレイヤーの `parentLayer` でなければなりません。レイヤーの `parentLayer` を設定しても、リスト内でのレイヤーの位置は移動しません。レイヤーの `parentLayer` を、レイヤーの移動が必要になるレイヤーに移動しても効果はありません。最上位のレイヤーの場合は `null` を使用します。

例

次の例では、同じタイムライン上の同じレベルで 2 つのレイヤーを使用します。最初のレイヤー (`layers[0]`) は、フォルダに変換されてから、2 番目のレイヤー (`layers[1]`) の親フォルダとして設定されます。このアクションにより、2 番目のレイヤーが最初のレイヤーの内部に移動します。

```
var parLayer = fl.getDocumentDOM().getTimeline().layers[0];
parLayer.layerType = "folder";
fl.getDocumentDOM().getTimeline().layers[1].parentLayer = parLayer;
```

layer.visible

使用できるバージョン

Flash MX 2004

シンタックス

`layer.visible`

説明

プロパティ。レイヤーのオブジェクトがステージ上に表示されるかどうかを指定するブール値です。`true` に設定されている場合、レイヤー内のすべてのオブジェクトが表示されます。`false` の場合、それらのオブジェクトは非表示になります。デフォルト値は `true` です。

例

次の例では、最初のレイヤー内のすべてのオブジェクトを非表示にします。

```
fl.getDocumentDOM().getTimeline().layers[0].visible = false;
```

library オブジェクト

使用できるバージョン

Flash MX 2004

説明

library オブジェクトは、[ライブラリ] パネルを表します。これは、Document オブジェクト ([document.library](#) を参照) のプロパティであり、`fl.getDocumentDOM().library` によってアクセスできます。

library オブジェクトは、シンボル、ビットマップ、サウンド、ビデオなどのさまざまなタイプのアイテムの配列を含んでいます。

library オブジェクトのメソッド一覧

library オブジェクトでは次のメソッドを使用できます。

メソッド	説明
<code>library.addItemToDocument()</code>	現在のアイテムまたは指定されたアイテムをステージの指定された位置に追加します。
<code>library.addNewItem()</code>	[ライブラリ] パネル内に指定されたタイプの新しいアイテムを作成し、新しいアイテムを現在選択されているアイテムに設定します。
<code>library.deleteItem()</code>	現在のアイテムまたは指定されたアイテムを [ライブラリ] パネルから削除します。
<code>library.duplicateItem()</code>	現在選択されているアイテムまたは指定されたアイテムのコピーを作成します。
<code>library.editItem()</code>	現在選択されているアイテムまたは指定されたアイテムを編集モードで開きます。
<code>library.expandFolder()</code>	ライブラリ内の現在選択されているフォルダまたは指定されたフォルダを開くか、閉じます。
<code>library.findItemIndex()</code>	ライブラリアイテムのインデックス値 (ゼロから始まる値) を返します。
<code>library.getItemProperty()</code>	選択されたアイテムのプロパティを取得します。
<code>library.getItemType()</code>	現在選択されているオブジェクト、またはライブラリパスで指定されたオブジェクトのタイプを取得します。
<code>library.getSelectedItems()</code>	ライブラリ内の現在選択されているすべてのアイテムの配列を取得します。

メソッド	説明
<code>library.importEmbeddedSWF()</code>	SWF ファイルをコンパイルされたクリップとしてライブラリに読み込みます。
<code>library.itemExists()</code>	指定されたアイテムがライブラリ内に存在するかどうかを確認します。
<code>library.moveToFolder()</code>	現在選択されているライブラリアイテムまたは指定されたライブラリアイテムを指定されたフォルダに移動します。
<code>library.newFolder()</code>	指定された名前の新しいフォルダ、または <code>folderName</code> パラメータが指定されていない場合はデフォルトの名前 (" 名称未設定フォルダ #") の新しいフォルダを現在選択されているフォルダ内に作成します。
<code>library.renameItem()</code>	[ライブラリ] パネルで現在選択されているライブラリアイテムの名前を変更します。
<code>library.selectAll()</code>	ライブラリ内のすべてのアイテムを選択または選択解除します。
<code>library.selectItem()</code>	指定されたライブラリアイテムを選択します。
<code>library.selectNone()</code>	すべてのライブラリアイテムの選択を解除します。
<code>library.setItemProperty()</code>	フォルダを無視してすべての選択されたライブラリアイテムのプロパティを設定します。
<code>library.updateItem()</code>	指定されたアイテムを更新します。

library オブジェクトのプロパティ一覧

library オブジェクトでは次のプロパティを使用できます。

プロパティ	説明
<code>library.items</code>	ライブラリ内の item オブジェクトの配列。

library.addItemToDocument()

使用できるバージョン

Flash MX 2004

シンタックス

```
library.addItemToDocument(position [, namePath])
```

パラメータ

position ステージ上のアイテムの中心の x,y 座標を指定する位置。

namePath アイテムの名前を指定するストリング。アイテムがフォルダ内にある場合は、スラッシュ表記を使用してアイテムの名前とパスを指定できます。*namePath*が指定されていない場合、現在のライブラリの選択内容が使用されます。このパラメータはオプションです。

戻り値

ブール値。ドキュメントにアイテムが正常に追加された場合は `true` を返します。それ以外の場合は `false` を返します。

説明

メソッド。現在のアイテムまたは指定されたアイテムをステージの指定された位置に追加します。

例

次の例では、現在選択されているアイテムをステージの (3, 60) の位置に追加します。

```
fl.getDocumentDOM().library.addItemToDocument({x:3, y:60});
```

次の例では、ライブラリの `folder1` にあるアイテム `Symbol1` をステージの (550, 485) の位置に追加します。

```
fl.getDocumentDOM().library.addItemToDocument({x:550.0, y:485.0}, "folder1/Symbol1");
```

library.addItem()

使用できるバージョン

Flash MX 2004

シンタックス

```
library.addItem(type [, namePath])
```

パラメータ

type 作成するアイテムのタイプを指定するストリング。*type* の指定できる値は、"video"、"movie clip"、"button"、"graphic"、"bitmap"、"screen"、および "folder" のみです。そのため、たとえばこのメソッドを使用してサウンドをライブラリに追加することはできません。フォルダパスの指定は、このメソッドを呼び出す前に `library.newFolder()` を使用するのと同じです。

namePath 追加するアイテムの名前を指定するストリング。アイテムがフォルダ内にある場合は、スラッシュ表記を使用してアイテムの名前とパスを指定できます。このパラメータはオプションです。

戻り値

ブール値。アイテムが正常に作成された場合は `true` を返します。それ以外の場合は `false` を返します。

説明

メソッド。[ライブラリ] パネル内に指定されたタイプの新しいアイテムを作成し、新しいアイテムを現在選択されているアイテムに設定します。サウンドなどのアイテムを含むアイテムをライブラリに読み込む方法の詳細については、`document.importFile()` を参照してください。

例

次の例では、"folderTwo" という新しいフォルダ内に "start" という新しいボタンを作成します。

```
fl.getDocumentDOM().library.addNewItem("button", "folderTwo/start");
```

library.deleteItem()

使用できるバージョン

Flash MX 2004

シンタックス

```
library.deleteItem([namePath])
```

パラメータ

namePath 削除するアイテムの名前を指定するストリング。アイテムがフォルダ内にある場合は、スラッシュ表記を使用してアイテムの名前とパスを指定できます。フォルダ名を渡した場合は、フォルダとそのすべてのアイテムが削除されます。名前を指定していない場合は、現在選択されている1つまたは複数のアイテムが削除されます。[ライブラリ] パネル内のすべてのアイテムを削除するには、すべてのアイテムを選択してからこのメソッドを使用します。このパラメータは省略可能です。

戻り値

ブール値。アイテムが正常に削除された場合は true を返します。それ以外の場合は false を返します。

説明

メソッド。現在のアイテムまたは指定されたアイテムを [ライブラリ] パネルから削除します。複数のアイテムを選択した場合は、このメソッドを複数のアイテムに適用できます。

例

次の例では、現在選択されているアイテムを削除します。

```
fl.getDocumentDOM().library.deleteItem();
```

次の例では、アイテム Symbol_1 をライブラリフォルダ "Folder_1" から削除します。

```
fl.getDocumentDOM().library.deleteItem("Folder_1/Symbol_1");
```

library.duplicateItem()

使用できるバージョン

Flash MX 2004

シンタックス

```
library.duplicateItem( [ namePath ] )
```

パラメータ

namePath 複製するアイテムの名前を指定するストリング。アイテムがフォルダ内にある場合は、スラッシュ表記を使用してアイテムの名前とパスを指定できます。このパラメータは省略可能です。

戻り値

ブール値。アイテムが正常に複製された場合は true を返します。それ以外の場合は false を返します。複数のアイテムが選択されている場合は、false を返します。

説明

メソッド。現在選択されているアイテムまたは指定されたアイテムのコピーを作成します。新しいアイテムにはデフォルトの名前 (item copy など) が付けられ、現在選択されているアイテムとして設定されます。複数のアイテムが選択されている場合、コマンドは失敗します。

例

次の例では、ライブラリフォルダ "test" 内にアイテム "square" のコピーを作成します。

```
fl.getDocumentDOM().library.duplicateItem("test/square");
```

library.editItem()

使用できるバージョン

Flash MX 2004

シンタックス

```
library.editItem([namePath])
```

パラメータ

namePath アイテムの名前を指定するストリング。アイテムがフォルダ内にある場合は、スラッシュ表記を使用してアイテムの名前とパスを指定できます。*namePath* が指定されていない場合は、1つの選択されたライブラリアイテムが編集モードで開かれます。ライブラリでアイテムが現在選択されていないか複数のアイテムが選択されている場合、メインのタイムラインの最初のシーンが編集用に表示されます。このパラメータはオプションです。

戻り値

ブール値。指定したアイテムが存在し、編集可能な場合は `true` を返します。それ以外の場合は `false` を返します。

説明

メソッド。現在選択されているアイテムまたは指定されたアイテムを編集モードで開きます。

例

次の例では、ライブラリの "test" フォルダ内のアイテム "circle" を編集用を開きます。

```
fl.getDocumentDOM().library.editItem("test/circle");
```

library.expandFolder()

使用できるバージョン

Flash MX 2004

シンタックス

```
library.expandFolder(bExpand [, bRecurseNestedParents [, namePath]])
```

パラメータ

bExpand ブール値。フォルダが開いている場合は `true` を返します。フォルダが閉じている場合は `false` (デフォルト) を返します。

bRecurseNestedParents ブール値。`true` の場合、*bExpand* の値を基にして、指定されたフォルダ内のすべてのフォルダが開くか閉じます。デフォルト値は `false` です。このパラメータはオプションです。

namePath 開くまたは閉じるフォルダの名前と、オプションでパスを指定する文字列。このパラメータが指定されていない場合、メソッドは現在選択されているフォルダにのみ適用されます。このパラメータはオプションです。

戻り値

ブール値。アイテムを正常に開くか閉じることができた場合は `true` を返します。正常に開くか閉じることができない場合、または指定されたアイテムがフォルダではない場合は `false` を返します。

説明

メソッド。ライブラリ内の現在選択されているフォルダまたは指定されたフォルダを開くか、閉じます。

例

次の例では、ライブラリ内の "test" フォルダと "test" フォルダ内のすべてのフォルダ (存在する場合) を閉じます。

```
fl.getDocumentDOM().library.expandFolder(false, true, "test");
```

library.findItemIndex()

使用できるバージョン

Flash MX 2004

シンタックス

```
library.findItemIndex(namePath)
```

パラメータ

namePath アイテムの名前を指定するストリング。アイテムがフォルダ内にある場合は、スラッシュ表記を使用してアイテムの名前とパスを指定できます。

戻り値

アイテムのゼロから始まるインデックス値を表す整数値

説明

メソッド。ライブラリアイテムのインデックス値 (ゼロから始まる値) を返します。ライブラリインデックスはフラットなので、フォルダはメインのインデックスの一部と見なされます。フォルダパスを使用して、ネストされたアイテムを指定することができます。

例

次の例では、"test" フォルダ内にあるライブラリアイテム "square" のゼロから始まるインデックス値を sqIndex 変数に格納し、ダイアログボックスにインデックス値を表示します。

```
var sqIndex = fl.getDocumentDOM().library.findItemIndex("test/square");  
alert(sqIndex);
```

library.getItemProperty()

使用できるバージョン

Flash MX 2004

シンタックス

```
library.getItemProperty(property)
```

パラメータ

property スtring。 *property* パラメータとして使用できる値の一覧は、 [Item オブジェクトのプロパティ一覧](#) およびそのサブクラスのプロパティ一覧を参照してください。

戻り値

プロパティのString値

説明

メソッド。選択されたアイテムのプロパティを取得します。

例

次の例では、 `ActionScript` を使用してシンボルを参照する場合またはランタイム共有にシンボルを参照する場合に、シンボルのリンケージ識別子の値を含むダイアログボックスを表示します。

```
alert(fl.getDocumentDOM().library.getItemProperty("linkageIdentifier"));
```

library.getItemType()

使用できるバージョン

Flash MX 2004

シンタックス

```
library.getItemType([namePath])
```

パラメータ

namePath アイテムの名前を指定するString。アイテムがフォルダ内にある場合は、スラッシュ表記を使用してアイテムの名前とパスを指定できます。 *namePath* が指定されていない場合、Flash によって現在選択されているアイテムのタイプが指定されます。複数のアイテムが現在選択され、 *namePath* が指定されていない場合、コマンドが無視されます。このパラメータはオプションです。

戻り値

オブジェクトのタイプを指定するString値戻り値については、 [item.itemType](#) を参照してください。

説明

メソッド。現在選択されている、またはライブラリパスで指定されたオブジェクトのタイプを取得します。

例

次の例では、"Folder_1/Folder_2" フォルダ内にあるタイプ Symbol_1 のアイテムを含ダイアログボックスを表示します。

```
alert(fl.getDocumentDOM().library.getItemType("Folder_1/Folder_2/Symbol_1"));
```

library.getSelectedItems()

使用できるバージョン

Flash MX 2004

パラメータ

なし

戻り値

ライブラリ内の現在選択されているすべてのアイテムの値の配列

説明

メソッド。ライブラリ内の現在選択されているすべてのアイテムの配列を取得します。

例

次の例では、現在選択されているライブラリアイテム (この場合は複数のオーディオファイル) の配列を selItems 変数に格納し、配列内の最初のオーディオファイルの sampleRate プロパティの値を "11 kHz" に変更します。

```
var selItems = fl.getDocumentDOM().library.getSelectedItems();
selItems[0].sampleRate = "11 kHz";
```

library.importEmbeddedSWF()

使用できるバージョン

Flash MX 2004

シンタックス

```
library.importEmbeddedSWF(linkageName, swfData [, libName])
```

パラメータ

linkageName ルートムービークリップの SWF リンケージの名前を提供するストリング。

swfData 外部ライブラリまたは DLL から取得したバイナリ SWF データの配列。

libName 作成されるアイテムのライブラリ名を指定するストリング。この名前が既に使用されている場合、代わりの名前が作成されます。このパラメータはオプションです。

戻り値

なし

説明

メソッド。SWF ファイルをコンパイルされたクリップとしてライブラリに読み込みます。[ファイル]-[読み込み]-[SWF]とは異なり、このメソッドを使用すると、コンパイルされた SWF ファイルをライブラリ内に埋め込むことができます。これに対応するユーザーインターフェイス機能はありません。外部ライブラリまたは DLL を使用する場合はこのメソッドを使用する必要があります ([587 ページ](#)、[第 3 章の「C レベルの拡張性」](#)を参照してください)。

読み込む SWF ファイルには、すべてのコンテンツを含むトップレベルのムービークリップが1つある必要があります。そのムービークリップのリンケージ識別子は、このメソッドに渡される *linkageName* パラメータと同じ値に設定されている必要があります。

例

次の例では、*linkageName* 値が MyMovie の SWF ファイルを Intro というコンパイルされたクリップとしてライブラリに追加します。

```
fl.getDocumentDOM().library.importEmbeddedSWF("MyMovie", swfData, "Intro");
```

library.itemExists()

使用できるバージョン

Flash MX 2004

シンタックス

```
library.itemExists(namePath)
```

パラメータ

namePath アイテムの名前を指定するストリング。アイテムがフォルダ内にある場合は、スラッシュ表記を使用してアイテムの名前とパスを指定できます。

戻り値

ブール値。指定したアイテムがライブラリ内に存在する場合は true を返します。それ以外の場合は false を返します。

説明

メソッド。指定されたアイテムがライブラリ内に存在するかどうかを確認します。

例

次の例では、アイテム `Symbol_1` がライブラリフォルダ `"Folder_1"` 内に存在するかどうかに応じて、ダイアログボックスに `true` または `false` を表示します。

```
alert(fl.getDocumentDOM().library.itemExists('Folder_1/Symbol_1'));
```

library.items

使用できるバージョン

Flash MX 2004

シンタックス

`library.items`

説明

プロパティ。ライブラリ内の `item` オブジェクトの配列です。

例

次の例では、ライブラリ名の配列を `itemArray` 変数に格納します。

```
var itemArray = fl.getDocumentDOM().library.items;
```

library.moveToFolder()

使用できるバージョン

Flash MX 2004

シンタックス

```
library.moveToFolder(folderPath [, itemToMove [, bReplace]])
```

パラメータ

folderPath フォルダのパスを `"FolderName"` または `"FolderName/FolderName"` の形式で指定する文字列。アイテムを最上位レベルに移動するには、*folderPath* に空の文字列 (`""`) を指定します。

itemToMove 移動するアイテムの名前を指定する文字列。*itemToMove* が指定されていない場合は、現在選択されているアイテムを移動します。このパラメータはオプションです。

bReplace ブール値。同じ名前のアイテムが既に存在している場合は、*bReplace* パラメータに `true` を指定して、既存のアイテムを移動するアイテムに置き換えます。`false` の場合、削除されたアイテムの名前が一意的な名前になります。デフォルト値は `false` です。このパラメータはオプションです。

戻り値

ブール値。アイテムが正常に移動された場合は `true` を返します。それ以外の場合は `false` を返します。

説明

メソッド。現在選択されているライブラリアイテムまたは指定されたライブラリアイテムを指定されたフォルダに移動します。`folderPath` パラメータが空の場合、アイテムは最上位レベルに移動します。

例

次の例では、アイテム `Symbol_1` をライブラリフォルダ `"new"` に移動し、そのフォルダ内の同じ名前のアイテムを置き換えます。

```
fl.getDocumentDOM().library.moveToFolder("new", "Symbol_1", true);
```

library.newFolder()

使用できるバージョン

Flash MX 2004

シンタックス

```
library.newFolder([folderPath])
```

パラメータ

folderPath 作成するフォルダの名前を指定するストリング。このストリングがパスとして指定され、パスが存在していない場合、パスが作成されます。このパラメータはオプションです。

戻り値

ブール値。フォルダが正常に作成された場合は `true` を返します。それ以外の場合は `false` を返します。

説明

メソッド。指定された名前の新しいフォルダ、または *folderName* パラメータが指定されていない場合はデフォルトの名前 (" 名称未設定フォルダ #") の新しいフォルダを現在選択されているフォルダ内に作成します。

例

次の例では、新しい 2 つのライブラリフォルダを作成します。2 番目のフォルダは最初のフォルダのサブフォルダです。

```
fl.getDocumentDOM().library.newFolder("first/second");
```

library.renameItem()

使用できるバージョン

Flash MX 2004

シンタックス

```
library.renameItem(name)
```

パラメータ

name ライブラリアイテムの新しい名前を指定するストリング。

戻り値

ブール値。アイテムの名前が正常に変更された場合は true、それ以外の場合は false を返します。複数のアイテムが選択されている場合、名前は変更されず、ユーザーインターフェイスの動作と一致させるために戻り値は false になります。

説明

メソッド。[ライブラリ] パネルで現在選択されているライブラリアイテムの名前を変更します。

例

次の例では、現在選択されているライブラリアイテムの名前を new_name に変更します。

```
fl.getDocumentDOM().library.renameItem("new_name");
```

library.selectAll()

使用できるバージョン

Flash MX 2004

シンタックス

```
library.selectAll([bSelectAll])
```

パラメータ

bSelectAll ライブラリ内のすべてのアイテムを選択するか選択解除するかを指定するブール値。このパラメータを省略するか、デフォルト値の true を使用すると、ライブラリ内のすべてのアイテムが選択されます。false を使用すると、すべてのライブラリアイテムが選択解除されます。このパラメータは省略可能です。

戻り値

なし

説明

メソッド。ライブラリ内のすべてのアイテムを選択または選択解除します。

例

次の例では、ライブラリ内のすべてのアイテムを選択します。

```
fl.getDocumentDOM().library.selectAll();  
fl.getDocumentDOM().library.selectAll(true);
```

次の例では、ライブラリ内のすべてのアイテムを選択解除します。

```
fl.getDocumentDOM().library.selectAll(false);  
fl.getDocumentDOM().library.selectNone();
```

library.selectItem()

使用できるバージョン

Flash MX 2004

シンタックス

```
library.selectItem(namePath [, bReplaceCurrentSelection [, bSelect]])
```

パラメータ

namePath アイテムの名前を指定するストリング。アイテムがフォルダ内にある場合は、スラッシュ表記を使用してアイテムの名前とパスを指定できます。

bReplaceCurrentSelection 現在の選択内容を置き換えるかアイテムを現在の選択内容に追加するかを指定するブール値。デフォルト値は `true` (現在の選択内容を置き換え) です。このパラメータはオプションです。

bSelect アイテムを選択するか選択解除するかを指定するブール値。デフォルト値は `true` (選択) です。このパラメータは省略可能です。

戻り値

ブール値。指定されたアイテムが存在する場合は `true` を返します。それ以外の場合は `false` を返します。

説明

メソッド。指定されたライブラリアイテムを選択します。

例

次の例では、ライブラリ内の現在の選択内容を "名称未設定フォルダ 1" 内の "シンボル 1" に変更します。

```
fl.getDocumentDOM().library.selectItem("untitled Folder_1/Symbol_1");
```


次の例では、" 名称未設定フォルダ 1" 内の " シンボル 1" が含まれるように、ライブラリ内の現在の選択内容を拡張します。

```
fl.getDocumentDOM().library.selectItem("untitled Folder_1/Symbol_1", false);
```

次の例では、" 名称未設定フォルダ 1" 内の " シンボル 1" を選択解除し、他の選択されているアイテムは変更しません。

```
fl.getDocumentDOM().library.selectItem("untitled Folder_1/Symbol_1", true, false);
```

library.selectNone()

使用できるバージョン

Flash MX 2004

シンタックス

```
library.selectNone()
```

パラメータ

なし

戻り値

なし

説明

メソッド。すべてのライブラリアイテムを選択解除します。

例

次の例では、ライブラリ内のすべてのアイテムを選択解除します。

```
fl.getDocumentDOM().library.selectNone();  
fl.getDocumentDOM().library.selectAll(false);
```

library.setItemProperty()

使用できるバージョン

Flash MX 2004

シンタックス

```
library.setItemProperty(property, value)
```

パラメータ

property 設定するプロパティの名前を示すストリング。プロパティの一覧は、[Item オブジェクトのプロパティ一覧](#)およびそのサブクラスのプロパティ一覧を参照してください。Item オブジェクトのサブクラスになっているオブジェクトを確認するには、[DOM 構造の概要](#)を参照してください。

value 指定されたプロパティに割り当てる値。

戻り値

なし

説明

メソッド。フォルダを無視してすべての選択されたライブラリアイテムのプロパティを設定します。

例

次の例では、値 `button` を選択された1つまたは複数のライブラリアイテムの `symbolType` プロパティに割り当てます。この場合、アイテムは [SymbolItem オブジェクト](#)である必要があります。`symbolType` は、[SymbolItem](#) オブジェクトの有効なプロパティです。

```
fl.getDocumentDOM().library.setItemProperty("symbolType", "button");
```

library.updateItem()

使用できるバージョン

Flash MX 2004

シンタックス

```
library.updateItem([namePath])
```

パラメータ

namePath アイテムの名前を指定するストリング。アイテムがフォルダ内にある場合は、スラッシュ表記を使用してアイテムの名前とパスを指定できます。これは、アイテムを右クリックして、メニューから [更新] を選択するのと同じです。名前を指定していない場合は、現在の選択内容が更新されます。このパラメータはオプションです。

戻り値

ブール値。アイテムが正常に更新された場合は `true` を返します。それ以外の場合は `false` を返します。

説明

メソッド。指定されたアイテムを更新します。

例

次の例では、現在選択されているアイテムが更新されたか (true) または更新されなかったか (false) を示すダイアログボックスを表示します。

```
alert(fl.getDocumentDOM().library.updateItem());
```

Math オブジェクト

使用できるバージョン

Flash MX 2004

説明

Math オブジェクトは、flash オブジェクトの読み取り専用プロパティとして使用できます。[fl.Math](#) を参照してください。このオブジェクトは、共通の算術演算を実行するメソッドを提供します。

Math オブジェクトのメソッド一覧

Math オブジェクトでは次のメソッドを使用できます。

メソッド	説明
Math.concatMatrix()	マトリックスの連結を実行し、結果を返します。
Math.invertMatrix()	指定したマトリックスの逆マトリックスを返します。
Math.pointDistance()	2つのポイントの間の距離を計算します。

Math.concatMatrix()

使用できるバージョン

Flash MX 2004

シンタックス

```
Math.concatMatrix(mat1, mat2)
```

パラメータ

mat1 および *mat2* 連結する Matrix オブジェクトを指定します。[Matrix オブジェクト](#) を参照してください。各パラメータは、a、b、c、d、tx、および ty のフィールドを含むオブジェクトである必要があります。

戻り値

連結されたオブジェクトのマトリックス

説明

メソッド。マトリックスの連結を実行し、結果を返します。

例

次の例では、現在選択されているオブジェクトを `elt` 変数に格納し、オブジェクトのマトリックスとビューのマトリックスを乗算し、その値を `mat` 変数に格納します。

```
var elt = fl.getDocumentDOM().selection[0];  
var mat = fl.Math.concatMatrix( elt.matrix , fl.getDocumentDOM().viewMatrix );
```

Math.invertMatrix()

使用できるバージョン

Flash MX 2004

シンタックス

`Math.invertMatrix(mat)`

パラメータ

mat 逆にする Matrix オブジェクトを示します ([Matrix オブジェクト](#)を参照)。このオブジェクトには、a、b、c、d、tx、および ty の各フィールドが必要です。

戻り値

元のマトリックスの逆の Matrix オブジェクト

説明

メソッド。指定されたマトリックスの逆マトリックスを返します。

例

次の例では、現在選択されているオブジェクトを *elt* 変数に格納し、そのマトリックスを *mat* 変数に代入し、マトリックスの逆を *inv* 変数に格納します。

```
var elt = fl.getDocumentDOM().selection[0];  
var mat = elt.matrix;  
var inv = fl.Math.invertMatrix( mat );
```

Math.pointDistance()

使用できるバージョン

Flash MX 2004

シンタックス

`Math.pointDistance(pt1, pt2)`

パラメータ

pt1 および *pt2* ポイント間の距離を測定する 2 つのポイントを指定します。

戻り値

ポイント間の距離を表す浮動小数点値

説明

メソッド。2つのポイント間の距離を計算します。

例

次の例では、*pt1* と *pt2* の間の距離の値を *dist* 変数に格納します。

```
var pt1 = {x:10, y:20}  
var pt2 = {x:100, y:200}  
var dist = fl.Math.pointDistance(pt1, pt2);
```


Matrix オブジェクト

使用できるバージョン

Flash MX 2004

説明

Matrix オブジェクトは、変換マトリックスを表します。

Matrix オブジェクトのプロパティ一覧

Matrix オブジェクトでは次のプロパティを使用できます。

プロパティ	説明
<code>matrix.a</code>	変換マトリックス内の (0,0) エlementを指定する浮動小数点値。
<code>matrix.b</code>	変換マトリックス内の (0,1) エlementを指定する浮動小数点値。
<code>matrix.c</code>	変換マトリックス内の (1,0) エlementを指定する浮動小数点値。
<code>matrix.d</code>	変換マトリックス内の (1,1) エlementを指定する浮動小数点値。
<code>matrix.tx</code>	シンボルの基準点またはシェイプの中心の x 軸の位置を指定する浮動小数点値。
<code>matrix.ty</code>	シンボルの基準点またはシェイプの中心の y 軸の位置を指定する浮動小数点値。

matrix.a

使用できるバージョン

Flash MX 2004

シンタックス

`matrix.a`

説明

プロパティ。変換マトリックス内の (0,0) エlementを指定する浮動小数点値です。この値は、オブジェクトの x 軸の拡大 / 縮小を表します。

例

マトリックス内の a および d のプロパティは、拡大 / 縮小を表します。次の例では、値をそれぞれ 2 と 3 に設定し、選択したオブジェクトの幅を 2 倍、高さを 3 倍に設定します。

```
var mat = fl.getDocumentDOM().selection[0].matrix;  
mat.a = 2;  
mat.d = 3;  
fl.getDocumentDOM().selection[0].matrix = mat;
```


a、b、c、およびdのマトリックスプロパティを $a = d$ および $b = -c$ という相対関係に基づいて設定することにより、オブジェクトを回転できます。たとえば、0.5、0.8、~~0.8~~、および 0.5 という値は、オブジェクトを 60° 回転します。

```
var mat = fl.getDocumentDOM().selection[0].matrix;  
mat.a = 0.5;  
mat.b = 0.8;  
mat.c = 0.8*(-1);  
mat.d = 0.5;  
fl.getDocumentDOM().selection[0].matrix = mat;
```

$a = d = 1$ および $c = b = 0$ に設定して、オブジェクトを元のシェイプに戻すことができます。

matrix.b

使用できるバージョン

Flash MX 2004

シンタックス

matrix.b

説明

プロパティ。マトリックス内の (0,1) エレメントを指定する浮動小数点値です。この値は、シェイプの垂直方向の傾斜を表します。これにより、シェイプの右端が垂直軸に沿って移動します。

マトリックス内の matrix.b および matrix.c プロパティは、傾斜を表します。[matrix.c](#) を参照してください。

例

次の例では、b と c をそれぞれ -1 と 0 に設定します。この設定により、オブジェクトが垂直方向に 45° 傾斜します。

```
var mat = fl.getDocumentDOM().selection[0].matrix;  
mat.b = -1;  
mat.c = 0;  
fl.getDocumentDOM().selection[0].matrix = mat;
```

オブジェクトを元のシェイプに戻すには、b と c を 0 に設定します。

[matrix.a](#) の例も参照してください。

matrix.c

使用できるバージョン

Flash MX 2004

シンタックス

`matrix.c`

説明

プロパティ。マトリックス内の (1,0) エlementを指定する浮動小数点値です。この値により、オブジェクトの下端を水平軸に沿って移動させることでオブジェクトを傾斜させます。

マトリックス内の `matrix.b` および `matrix.c` プロパティは、傾斜を表します。

例

`matrix.b` の例を参照してください。

matrix.d

使用できるバージョン

Flash MX 2004

シンタックス

`matrix.d`

説明

プロパティ。マトリックス内の (1,1) エlementを指定する浮動小数点値です。この値は、オブジェクトの y 軸の拡大 / 縮小を表します。

例

`matrix.a` の例を参照してください。

matrix.tx

使用できるバージョン

Flash MX 2004

シンタックス

`matrix.tx`

説明

プロパティ。シンボルの基準点 (あるいは原点またはゼロ点) またはシェイプの中心の x 軸の位置を指定する浮動小数点値です。これは、変形の x 変換を定義します。

`matrix.tx` および `matrix.ty` プロパティを設定することでオブジェクトを移動できます。

[matrix.ty](#) を参照してください。

例

次の例では、`tx` と `ty` を 0 に設定し、オブジェクトの基準点をドキュメント内のポイント 0,0 に移動します。

```
var mat = fl.getDocumentDOM().selection[0].matrix;  
mat.tx = 0;  
mat.ty = 0;  
fl.getDocumentDOM().selection[0].matrix = mat;
```

matrix.ty

使用できるバージョン

Flash MX 2004

シンタックス

`matrix.ty`

説明

プロパティ。シンボルの基準点またはシェイプの中心の y 軸の位置を指定する浮動小数点値です。これは、変形の y 変換を定義します。

`matrix.tx` および `matrix.ty` プロパティを設定することでオブジェクトを移動できます。

例

[matrix.tx](#) の例を参照してください。

outputPanel オブジェクト

使用できるバージョン

Flash MX 2004

説明

このオブジェクトは、シンタックスエラーなどのトラブルシューティング情報を表示する [出力] パネルを表します。このオブジェクトにアクセスするには、`fl.outputPanel` (または `flash.outputPanel`) を使用します。[fl.outputPanel](#) を参照してください。

outputPanel オブジェクトのメソッド一覧

outputPanel オブジェクトでは次のメソッドを使用します。

メソッド	説明
outputPanel.clear()	[出力] パネルの内容をクリアします。
outputPanel.save()	[出力] パネルの内容をローカルのテキストファイルに保存します。
outputPanel.trace()	[出力] パネルの内容に行を追加し、新しい行で終了させます。

outputPanel.clear()

使用できるバージョン

Flash MX 2004

シンタックス

```
outputPanel.clear()
```

パラメータ

なし

戻り値

なし

説明

メソッド。[出力] パネルの内容をクリアします。このメソッドをバッチ処理アプリケーションで使用することで、エラーのリストをクリアしたり、[outputPanel.save\(\)](#) と共に使用してエラーの増分を保存したりできます。

例

次の例では、[出力] パネルの現在の内容をクリアします。

```
fl.outputPanel.clear();
```

outputPanel.save()

使用できるバージョン

Flash MX 2004。Flash 8 では、*bUseSystemEncoding* パラメータが追加されました。

シンタックス

```
outputPanel.save(fileURI [ , bAppendToFile [ , bUseSystemEncoding ] ])
```

パラメータ

fileURI file:/// URI として表されるストリング。[出力] パネルの内容を格納するローカルファイルを指定します。

bAppendToFile オプションのブール値。true の場合、[出力] パネルの内容を出力ファイルに追加します。false の場合、出力ファイルが既に存在している場合に出力ファイルを上書きします。デフォルト値は false です。

bUseSystemEncoding オプションのブール値。true の場合、システムエンコーディングを使用して [出力] パネルのテキストを保存します。false の場合、UTF-8 エンコーディングを使用して [出力] パネルのテキストを保存します。このとき、テキストの先頭にバイト順マーク (BOM) 文字が付きます。デフォルト値は false です。

戻り値

なし

説明

メソッド。[出力] パネルの内容をローカルのテキストファイルに上書きまたは追加し、保存します。

fileURI が無効または指定されていない場合は、エラーが報告されます。

このメソッドは、バッチ処理で役に立ちます。たとえば、複数のコンポーネントをコンパイルする JSFL ファイルを作成するとします。すべてのコンパイルエラーは [出力] パネルに表示されます。このメソッドを使用してコンパイルエラーをテキストファイルに保存し、使用している構築システムで自動的にそれらを解析することができます。

例

次の例では、[出力] パネルの内容を "/tests" フォルダ内の "batch.log" ファイルに保存します。"batch.log" が既に存在する場合は上書きされます。

```
fl.outputPanel.save("file:///c:/tests/batch.log");
```

outputPanel.trace()

使用できるバージョン

Flash MX 2004

シンタックス

```
outputPanel.trace(message)
```

パラメータ

message [出力] パネルに追加するテキストを含むストリング。

戻り値

なし

説明

メソッド。テキストストリングを [出力] パネルに送り、改行で終了させます。[出力] パネルが表示されていない場合は、表示します。このメソッドは、[fl.trace\(\)](#) と同じで、ActionScript の `trace()` ステートメントと同じように機能します。

空白行を送るには、`outputPanel.trace("")` または `outputPanel.trace("\n")` を使用します。この後者のコマンドをインラインで使用して、`\n` を *message* ストリングの一部とすることができます。

例

次の例では、数行のテキストを [出力] パネルに表示します。

```
fl.outputPanel.clear();
fl.outputPanel.trace("Hello World!!!");
var myPet = "cat";
fl.outputPanel.trace("\nI have a " + myPet);
fl.outputPanel.trace("");
fl.outputPanel.trace("I love my " + myPet);
fl.outputPanel.trace("Do you have a " + myPet + "?");
```

Oval オブジェクト

継承 [Element オブジェクト](#) > [Shape オブジェクト](#) > Oval オブジェクト

使用できるバージョン

Flash CS3 Professional

説明

Oval オブジェクトは、楕円プリミティブツールを使用して描画されるシェイプです。アイテムが Oval オブジェクトかどうかを判断するには、[shape.isOvalObject](#) を使用します。

Oval オブジェクトのプロパティ一覧

Oval オブジェクトでは、[Shape オブジェクト](#) のプロパティの他に、以下のプロパティを使用できます。Oval オブジェクトのプロパティを設定するには、[document.setOvalObjectProperty\(\)](#) を使用します。

プロパティ	説明
OvalObject.closePath	読み取り専用。プロパティインスペクタで [パスを閉じる] チェックボックスがオンであるかどうかを指定するブール値です。
OvalObject.endAngle	読み取り専用。Oval オブジェクトの終了角度を指定する浮動小数値です。
OvalObject.innerRadius	読み取り専用。Oval オブジェクトの内側の半径をパーセンテージで指定する浮動小数値です。
OvalObject.startAngle	読み取り専用。Oval オブジェクトの開始角度を指定する浮動小数値です。

OvalObject.closePath

使用できるバージョン

Flash CS3 Professional

シンタックス

[OvalObject.closePath](#)

説明

読み取り専用プロパティ。プロパティインスペクタで [パスを閉じる] チェックボックスがオンであるかどうかを指定するブール値です。オブジェクトの開始角度と終了角度の値が等しい場合は、これらの値が変更されるまで、このプロパティを設定しても効果がありません。

この値を設定するには、`document.setOvalObjectProperty()` を使用します。

例

次の例では、`OvalObject.closePath` プロパティを選択解除します。

```
fl.getDocumentDOM().setOvalObjectProperty("closePath",false);
```

関連項目

[document.setOvalObjectProperty\(\)](#), [shape.isOvalObject](#)

OvalObject.endAngle

使用できるバージョン

Flash CS3 Professional

シンタックス

```
OvalObject.endAngle
```

説明

読み取り専用プロパティ。`Oval` オブジェクトの終了角度を指定する浮動小数値です。指定できる値は、0 ～ 360 です。

この値を設定するには、`document.setOvalObjectProperty()` を使用します。

例

次の例では、選択された `Oval` オブジェクトの終了角度を 270 に設定します。

```
fl.getDocumentDOM().setOvalObjectProperty("endAngle",270);
```

関連項目

[document.setOvalObjectProperty\(\)](#), [OvalObject.startAngle](#), [shape.isOvalObject](#)

OvalObject.innerRadius

使用できるバージョン

Flash CS3 Professional

シンタックス

```
OvalObject.innerRadius
```

説明

読み取り専用プロパティ。`Oval` オブジェクトの内側の半径をパーセンテージで指定する浮動小数値です。指定できる値は、0 ～ 99 です。

この値を設定するには、[document.setOvalObjectProperty\(\)](#) を使用します。

例

次の例では、選択された Oval オブジェクトの内側の半径を 50% に設定します。

```
fl.getDocumentDOM().setOvalObjectProperty("innerRadius",50);
```

関連項目

[document.setOvalObjectProperty\(\)](#), [shape.isOvalObject](#)

OvalObject.startAngle

使用できるバージョン

Flash CS3 Professional

シンタックス

```
OvalObject.startAngle
```

説明

読み取り専用プロパティ。Oval オブジェクトの開始角度を指定する浮動小数値です。指定できる値は、0 ～ 360 です。

この値を設定するには、[document.setOvalObjectProperty\(\)](#) を使用します。

例

次の例では、選択された Oval オブジェクトの開始角度を 270 に設定します。

```
fl.getDocumentDOM().setOvalObjectProperty("startAngle",270);
```

関連項目

[document.setOvalObjectProperty\(\)](#), [OvalObject.endAngle](#), [shape.isOvalObject](#)

Parameter オブジェクト

使用できるバージョン

Flash MX 2004

説明

Parameter オブジェクト型は、`screen.parameters` 配列 (Flash オーサリングツールのスクリーンのプロパティインスペクタに対応) または `componentInstance.parameters` 配列 (オーサリングツールのコンポーネントのプロパティインスペクタに対応) によってアクセスされます。

[screen.parameters](#) および [componentInstance.parameters](#) を参照してください。

Parameter オブジェクトのメソッド一覧

Parameter オブジェクトでは次のメソッドを使用できます。

メソッド	説明
parameter.insertItem()	リスト、オブジェクト、または配列にアイテムを挿入します。
parameter.removeItem()	スクリーンまたはコンポーネントのパラメータのリスト、オブジェクト、または配列型のエレメントを削除します。

Parameter オブジェクトのプロパティ一覧

Parameter オブジェクトでは次のプロパティを使用できます。

プロパティ	説明
parameter.category	screen パラメータまたは componentInstance パラメータの category プロパティを指定するストリングです。
parameter.listIndex	選択されたリストアイテムの値を指定する整数。
parameter.name	読み取り専用。パラメータの名前を指定するストリングです。
parameter.value	コンポーネントインスペクタの [パラメータ] タブの [値] フィールド、プロパティインスペクタの [パラメータ] タブの [値] フィールド、またはスクリーンのプロパティインスペクタの [値] フィールドに対応します。
parameter.valueType	読み取り専用。スクリーンまたはコンポーネントのパラメータの型を示すストリングです。
parameter.verbose	パラメータを表示する場所を指定します。

parameter.category

使用できるバージョン

Flash MX 2004

シンタックス

`parameter.category`

説明

プロパティ。screen パラメータまたは componentInstance パラメータの category プロパティを指定する文字列です。このプロパティは、パラメータのリストを表示する方法の1つを提供します。この機能は、Flash のユーザーインターフェイスでは使用できません。

parameter.insertItem()

使用できるバージョン

Flash MX 2004

シンタックス

`parameter.insertItem(index, name, value, type)`

パラメータ

index リスト、オブジェクト、または配列内のアイテムが挿入される場所を示すゼロから始まる整数のインデックス。*index* が 0 の場合、アイテムはリストの先頭に挿入されます。*index* がリストのサイズよりも大きい場合、新しいアイテムは、配列の末尾に挿入されます。

name 挿入するアイテムの名前を示す文字列。これは、オブジェクトパラメータの必須のパラメータです。

value 挿入するアイテムの値を指定する文字列。

type 挿入するアイテムの型を指定する文字列。

戻り値

なし

説明

メソッド。リスト、オブジェクト、または配列にアイテムを挿入します。パラメータがリスト、オブジェクト、または配列の場合、*value* プロパティは配列です。

例

次の例では、"New Value" という値を `labelPlacement` パラメータに挿入します。

```
// ステージで Button コンポーネントのインスタンスを選択します。
var parms = fl.getDocumentDOM().selection[0].parameters;
parms[2].insertItem(0, "name", "New Value", "String");
var values = parms[2].value;
for(var prop in values){
    fl.trace("labelPlacement parameter value = " + values[prop].value);
}
```

parameter.listIndex

使用できるバージョン

Flash MX 2004

シンタックス

`parameter.listIndex`

説明

プロパティ。選択されたリストアイテムの値です。このプロパティは、`valueType` パラメータが "List" になっている場合のみ有効です。

例

次の例では、`Slide` の最初のパラメータである `autoKeyNav` パラメータを設定します。このパラメータをいずれかの有効な値 (`true`、`false`、または `inherit`) に設定するために、`parameter.listIndex` はリスト内のアイテムのインデックス (`true` の場合は 0、`false` の場合は 1、`inherit` の場合は 2) に設定されます。

```
var parms = fl.getDocumentDOM().screenOutline.screens[1].parameters;
parms[0].listIndex = 1;
```

parameter.name

使用できるバージョン

Flash MX 2004

シンタックス

`parameter.name`

説明

読み取り専用プロパティ。パラメータの名前を指定する文字列です。

例

次の例では、選択されたコンポーネントの 5 番目のパラメータの名前を示します。

```
var parms = fl.getDocumentDOM().selection[0].parameters;  
fl.trace("name: " + parms[4].name);
```

次の例では、指定されたスクリーンの 5 番目のパラメータの名前を示します。

```
var parms = fl.getDocumentDOM().screenOutline.screens[1].parameters;  
fl.trace("name: " + parms[4].name);
```

parameter.removeItem()

使用できるバージョン

Flash MX 2004

シンタックス

```
parameter.removeItem(index)
```

パラメータ

index スクリーンまたはコンポーネントのプロパティから削除するアイテムのゼロから始まる整数のインデックス。

戻り値

なし

説明

メソッド。スクリーンまたはコンポーネントのパラメータのリスト、オブジェクト、または配列型のエレメントを削除します。

例

次の例では、インデックス 1 のエレメントをコンポーネントの labelPlacement パラメータから削除します。

```
// ステージで Button コンポーネントのインスタンスを選択します。  
var parms = fl.getDocumentDOM().selection[0].parameters;  
var values = parms[2].value;  
fl.trace("--Original--");  
for(var prop in values){  
    fl.trace("labelPlacement value = " + values[prop].value);  
}  
parms[2].removeItem(1);  
  
var newValues = parms[2].value;  
fl.trace("--After Removing Item--");  
for(var prop in newValues){
```

```
        fl.trace("labelPlacement value = " + newValues[prop].value);
    }
}
```

次の例では、インデックス 1 のエレメントをスクリーンの autoKeyNav パラメータから削除します。

```
// プレゼンテーションドキュメントを開きます。
var parms = fl.getDocumentDOM().screenOutline.screens[1].parameters;
var values = parms[0].value;
fl.trace("--Original--");
for(var prop in values){
    fl.trace("autoKeyNav value = " + values[prop].value);
}
parms[0].removeItem(1);

var newValues = parms[0].value;
fl.trace("--After Removing Item--");
for(var prop in newValues){
    fl.trace("autoKeyNav value = " + newValues[prop].value);
}
}
```

parameter.value

使用できるバージョン

Flash MX 2004

シンタックス

parameter.value

説明

プロパティ。コンポーネントインスペクタの [パラメータ] タブの [値] フィールド、プロパティインスペクタの [パラメータ] タブの [値] フィールド、またはスクリーンのプロパティインスペクタの [値] フィールドに対応します。value プロパティの型は、パラメータの valueType プロパティによって決まります ([parameter.valueType](#) を参照)。

parameter.valueType

使用できるバージョン

Flash MX 2004

シンタックス

parameter.valueType

説明

読み取り専用プロパティ。スクリーンまたはコンポーネントのパラメータの型を示す文字列です。型として、"Default"、"Array"、"Object"、"List"、"String"、"Number"、"Boolean"、"Font Name"、"Color"、"Collection"、"Web Service URL"、または "Web Service Operation" のいずれかの値を指定できます。

関連項目

[parameter.value](#)

parameter.verbose

使用できるバージョン

Flash MX 2004

シンタックス

`parameter.verbose`

説明

プロパティ。パラメータを表示する場所を指定します。このプロパティの値が 0 (簡潔) の場合、パラメータはコンポーネントインスペクタにのみ表示されます。1 (詳細) の場合、コンポーネントインスペクタおよびプロパティインスペクタの [パラメータ] タブに表示されます。

Path オブジェクト

使用できるバージョン

Flash MX 2004

説明

Path オブジェクトは、線のセグメント (直線、曲線、または両方) のシーケンスを定義します。このシーケンスは通常、拡張ツールを作成する場合に使用します。次の例では、flash オブジェクトから返される Path オブジェクトのインスタンスを表示します。

```
path = fl.drawingLayer.newPath();
```

[adrawingLayer](#) オブジェクトも参照してください。

Path オブジェクトのメソッド一覧

Path オブジェクトでは次のメソッドを使用できます。

メソッド	説明
path.addCubicCurve()	パスに三次ベジェセグメントを追加します。
path.addCurve()	パスに二次ベジェセグメントを追加します。
path.addPoint()	パスにポイントを追加します。
path.clear()	パスからすべてのポイントを削除します。
path.close()	パスの最初のポイントの位置にポイントを追加し、そのポイントまでパスを延長することでパスを閉じます。
path.makeShape()	現在の線と塗りの設定を使用して、ステージにシェイプを作成します。
path.newContour()	パス内で新しい輪郭を開始します。

Path オブジェクトのプロパティ一覧

Path オブジェクトでは次のプロパティを使用できます。

プロパティ	説明
path.nPts	読み取り専用。パス内のポイントの数を表す整数です。

path.addCubicCurve()

使用できるバージョン

Flash MX 2004

シンタックス

```
path.addCubicCurve(xAnchor, yAnchor, x2, y2, x3, y3, x4, y4)
```

パラメータ

xAnchor 最初のコントロールポイントの *x* 座標を指定する浮動小数点数。

yAnchor 最初のコントロールポイントの *y* 座標を指定する浮動小数点数。

x2 2 番目のコントロールポイントの *x* 座標を指定する浮動小数点数。

y2 2 番目のコントロールポイントの *y* 座標を指定する浮動小数点数。

x3 3 番目のコントロールポイントの *x* 座標を指定する浮動小数点数。

y3 3 番目のコントロールポイントの *y* 座標を指定する浮動小数点数。

x4 4 番目のコントロールポイントの *x* 座標を指定する浮動小数点数。

y4 4 番目のコントロールポイントの *y* 座標を指定する浮動小数点数。

戻り値

なし

説明

メソッド。パスに三次ベジェセグメントを追加します。

例

次の例では、新しいパスを作成して *myPath* 変数に格納し、そのパスに曲線を割り当てます。

```
var myPath = fl.drawingLayer.newPath();
myPath.addCubicCurve(0, 0, 10, 20, 20, 20, 30, 0);
```

path.addCurve()

使用できるバージョン

Flash MX 2004

シンタックス

```
path.addCurve(xAnchor, yAnchor, x2, y2, x3, y3)
```

パラメータ

xAnchor 最初のコントロールポイントの *x* 座標を指定する浮動小数点数。

yAnchor 最初のコントロールポイントの *y* 座標を指定する浮動小数点数。

x2 2 番目のコントロールポイントの *x* 座標を指定する浮動小数点数。

y2 2 番目のコントロールポイントの *y* 座標を指定する浮動小数点数。

x3 3 番目のコントロールポイントの *x* 座標を指定する浮動小数点数。

y3 3 番目のコントロールポイントの *y* 座標を指定する浮動小数点数。

戻り値

なし。

説明

メソッド。パスに二次ベジェセグメントを追加します。

例

次の例では、新しいパスを作成して *myPath* 変数に格納し、そのパスに曲線を割り当てます。

```
var myPath = fl.drawingLayer.newPath();  
myPath.addCurve(0, 0, 10, 20, 20, 0);
```

path.addPoint()

使用できるバージョン

Flash MX 2004

シンタックス

```
path.addPoint(x, y)
```

パラメータ

x ポイントの *x* 座標を指定する浮動小数点数。

y ポイントの *y* 座標を指定する浮動小数点数。

戻り値

なし。

説明

メソッド。パスにポイントを追加します。

例

次の例では、新しいパスを作成して myPath 変数に格納し、そのパスに新しいポイントを割り当てます。

```
var myPath = fl.drawingLayer.newPath();  
myPath.addPoint(10, 100);
```

path.clear()

使用できるバージョン

Flash MX 2004

シンタックス

```
path.clear()
```

パラメータ

なし

戻り値

なし

説明

メソッド。パスからすべてのポイントを削除します。

例

次の例では、myPath 変数に格納されたパスからすべてのポイントを削除します。

```
var myPath = fl.drawingLayer.newPath();  
myPath.clear();
```

path.close()

使用できるバージョン

Flash MX 2004

シンタックス

```
path.close()
```

パラメータ

なし

戻り値

なし

説明

メソッド。パスの最初のポイントの位置にポイントを追加し、そのポイントまでパスを延長することでパスを閉じます。パスにポイントがない場合は、ポイントは追加されません。

例

次の例では、閉じたパスを作成します。

```
var myPath = fl.drawingLayer.newPath();
myPath.close();
```

path.makeShape()

使用できるバージョン

Flash MX 2004

シンタックス

```
path.makeShape([bSupressFill [, bSupressStroke]])
```

パラメータ

bSupressFill ブール値。true に設定されている場合、シェイプに適用される塗りを無効にします。デフォルト値は false です。このパラメータはオプションです。

bSupressStroke ブール値。true に設定されている場合、シェイプに適用される線を無効にします。デフォルト値は false です。このパラメータはオプションです。

戻り値

なし

説明

メソッド。現在の線と塗りの設定を使用して、ステージにシェイプを作成します。パスは、シェイプが作成された後にクリアされます。このメソッドには、結果のシェイプオブジェクトに塗りと線を適用しないようにする 2 つのオプションのパラメータがあります。これらのパラメータを省略するか、false に設定した場合、塗りと線の現在の値が使用されます。

例

次の例では、現在の塗りと線を使用してシェイプを作成します。

```
var myPath = fl.drawingLayer.newPath();
myPath.makeShape(false, true);
```

path.newContour()

使用できるバージョン

Flash MX 2004

シンタックス

```
path.newContour()
```

パラメータ

なし

戻り値

なし

説明

メソッド。パス内で新しい輪郭を開始します。

例

次の例では、白抜きの四角形を作成します。

```
var myPath = fl.drawingLayer.newPath();
myPath.addPoint(0, 0);
myPath.addPoint(0, 30);
myPath.addPoint(30, 30);
myPath.addPoint(30, 0);
myPath.addPoint(0, 0);

myPath.newContour();
myPath.addPoint(10, 10);
myPath.addPoint(10, 20);
myPath.addPoint(20, 20);
myPath.addPoint(20, 10);
myPath.addPoint(10, 10);

myPath.makeShape();
```

path.nPts

使用できるバージョン

Flash MX 2004

シンタックス

path.nPts

説明

読み取り専用プロパティ。パス内のポイントの数を表す整数です。新しいパスにはポイントはありません。

例

次の例では、[出力] パネルを使用して、myPath 変数によって参照されているパス内のポイントの数を表示します。

```
var myPath = fl.drawingLayer.newPath();
var numOfPoints = myPath.nPts;
fl.trace("Number of points in the path: " + numOfPoints);
// 表示結果 : Number of points in the path: 0
```

Project オブジェクト

使用できるバージョン

Flash 8

説明

Project オブジェクトは、Flash プロジェクト (FLP) ファイルを表します。次のコマンドを使用して、Project オブジェクトを返すことができます。

- 新しいプロジェクトファイルを作成するには、`fl.createProject()` を使用します。
- 既存のプロジェクトファイルを開くには、`fl.openProject()` を使用します。
- 現在開いているプロジェクトの Project オブジェクトを返すには、`fl.getProject()` を使用します。

Project オブジェクトのメソッド一覧

Project オブジェクトでは次のメソッドを使用できます。

メソッド	説明
<code>project.addFile()</code>	指定されたファイルをプロジェクトに追加します。
<code>project.canPublishProject()</code>	プロジェクトをパブリッシュできるかどうかを調べます。
<code>project.canTestProject()</code>	プロジェクトをテストできるかどうかを調べます。
<code>project.findProjectItem()</code>	指定されたファイルをプロジェクト内で検索します。
<code>project.publishProject()</code>	プロジェクト内の FLA ファイルをパブリッシュします。
<code>project.testProject()</code>	プロジェクトをテストします。

Project オブジェクトのプロパティ一覧

Project オブジェクトでは次のプロパティを使用できます。

プロパティ	説明
<code>project.defaultItem</code>	プロジェクト内のデフォルトのドキュメントを表す ProjectItem オブジェクト を指定します。
<code>project.items</code>	読み取り専用プロパティ。ProjectItem オブジェクトの配列です (ProjectItem オブジェクト を参照)。

プロパティ	説明
<code>project.name</code>	[プロジェクト] パネルに表示されるプロジェクトの名前。
<code>project.projectURI</code>	読み取り専用。プロジェクトファイルのパスと名前を表す文字列です。file:/// URI で表します。

project.addFile()

使用できるバージョン

Flash 8

シンタックス

```
project.addFile(fileURI [ , autoCreateFolder])
```

パラメータ

fileURI file:/// URI として表され、プロジェクトに追加するファイルを指定する文字列。

autoCreateFolder *fileURI* 内のパスをミラーリングするために [プロジェクト] パネルにフォルダを自動的に作成するかどうかを指定するオプションのブール値。デフォルト値は false です。

戻り値

成功した場合は `ProjectItem` オブジェクトを返します。それ以外の場合は `undefined` を返します。
[ProjectItem オブジェクト](#) を参照してください。

説明

メソッド。指定されたファイルをプロジェクトに追加します。*autoCreateFolder* を使用して、[プロジェクト] パネル内で新しいファイルを配置する位置を決定することができます。

- *autoCreateFolder* を省略するか、値 `false` を渡した場合、ファイルはプロジェクトのルートレベルに追加されます。
- *autoCreateFolder* に対して値 `true` を渡し、*fileURI* が、ディスク上のフォルダ構造内で FLP ファイルの下にある場合、ファイルのフォルダ構造が [プロジェクト] パネルにミラーリングされます。つまり、新しいフォルダは、ディスク上のファイルの場所をミラーリングするために必要な場合に、[プロジェクト] パネルに追加されます。
- *autoCreateFolder* に対して値 `true` を渡し、*fileURI* がディスク上のフォルダ構造内で FLP ファイルの上にある場合、ファイルはルートレベルに追加されます。つまり、*autoCreateFolder* は無視されます。

例

次の例では、このコマンドのいくつかの使用法を示します。この場合、開いているプロジェクトファイルは、`c:\Projects` ディレクトリ内にあり、現在プロジェクト内にあるファイルのみがルートレベルで追加されています。

```
// Project オブジェクトを取得します。
var myProject = fl.getProject();

// 次のコマンドは、プロジェクト内のルートレベルの下に "files" というフォルダを作成し、
// "myFile.fla" をそのフォルダ内に置きます。
var newFile = myProject.addFile("file:///C|Projects/files/myFile.fla", true)
fl.trace(newFile.isMissing); // false

// 次の 2 つのコマンドの効果は同じです。つまり、myFile_02.fla をプロジェクトのルートレベルに
// 配置します。
var newFile = myProject.addFile("file:///C|Projects/files/myFile_02.fla" ,
    false)
var newFile = myProject.addFile("file:///C|Projects/files/myFile_02.fla")
fl.trace(newFile.isMissing); // false

// 次のコマンドは、"myFile_03" を見つからないファイルとしてプロジェクトのルートレベルに配置し
// ます。
var newFile = myProject.addFile("file:///C|myFile_03.fla")
fl.trace(newFile.isMissing); // true
```

次の例では、プロジェクトへの新しいファイルの追加を試行し、ファイルが追加されたかどうかを示すメッセージを [出力] パネルに表示します。

```
var myProject = fl.getProject();
var newItem = myProject.addFile("file:///C|Projects/files/Integra.fla", true);
fl.trace( "Item " + ( newItem ? "was" : "was not" ) + " added!" );
```

関連項目

[fl.getProject\(\)](#), [project.items](#), [ProjectItem](#) オブジェクト

project.canPublishProject()

使用できるバージョン

Flash 8

シンタックス

```
project.canPublishProject()
```

パラメータ

なし

戻り値

プロジェクトをパブリッシュできるかどうかを示すブール値

説明

メソッド。プロジェクトをパブリッシュできるかどうかを調べます。プロジェクトは1つ以上の FLA ファイルを含んでいる場合にパブリッシュできます。

例

次の例では、プロジェクトをパブリッシュできない場合に [出力] パネルにメッセージを表示します。

```
if (!fl.getProject().canPublishProject()) {  
    fl.trace("Project cannot be published!");  
}
```

関連項目

[fl.getProject\(\)](#), [project.publishProject\(\)](#), [projectItem.canPublish\(\)](#)

project.canTestProject()

使用できるバージョン

Flash 8

シンタックス

`project.canTestProject()`

パラメータ

なし

戻り値

プロジェクトをテストできるかどうかを示すブール値

説明

メソッド。プロジェクトをテストできるかどうかを調べます。プロジェクトは、デフォルトドキュメントが指定されている場合にテストできます。

例

次の例では、プロジェクトをテストできない場合に [出力] パネルにメッセージを表示します。

```
if (!fl.getProject().canTestProject()) {  
    fl.trace("Project cannot be tested!");  
}
```

関連項目

[fl.getProject\(\)](#), [project.testProject\(\)](#), [projectItem.canTest\(\)](#)

project.defaultItem

使用できるバージョン

Flash 8

シンタックス

project.defaultItem

説明

プロパティ。プロジェクト内のデフォルトのドキュメントを表す `ProjectItem` オブジェクトを指定します。プロジェクトをテストする場合は、デフォルトのアイテムを指定する必要があります。[ProjectItem オブジェクト](#) を参照してください。

例

次の例では、プロジェクト内のデフォルトドキュメントを "Flower.fla" ファイルに設定します。

```
var myProject = fl.getProject();
var item = myProject.findProjectItem("file:///C:/Projects/files/Flower.fla");
fl.myProject.defaultItem = item;
```

次の例では、デフォルトドキュメントの名前を [出力] パネルに表示します。

```
fl.trace(fl.getProject().defaultItem.displayName);
```

関連項目

[fl.getProject\(\)](#), [project.findProjectItem\(\)](#), [ProjectItem オブジェクト](#)

project.findProjectItem()

使用できるバージョン

Flash 8

シンタックス

project.findProjectItem(*fileURI*)

パラメータ

fileURI file:/// URI として表され、プロジェクト内で検索するファイルを指定するストリング。

戻り値

成功した場合はアイテムの `ProjectItem` オブジェクト。それ以外の場合は `false`。詳細については、[ProjectItem オブジェクト](#) を参照してください。

説明

メソッド。指定されたファイルをプロジェクト内で検索します。

例

次の例では、指定されたファイルがプロジェクト内で見つからない場合に [出力] パネルにエラーメッセージを表示します。

```
var myProject = fl.getProject();
var item = myProject.findProjectItem("file:///C|/Projects/files/Integra fla");
if (item == undefined) {
    fl.trace("Integra fla is missing!");
}
```

関連項目

[fl.getProject\(\)](#), [ProjectItem オブジェクト](#), [projectItem.isMissing](#)

project.items

使用できるバージョン

Flash 8

シンタックス

`project.items`

説明

読み取り専用プロパティ。ProjectItem オブジェクトの配列です ([ProjectItem オブジェクト](#)を参照してください)。

例

次の例では、プロジェクト内のすべてのアイテムの名前を表示します。名前が [出力] パネルに表示されます。

```
for (i = 0; i < fl.getProject().items.length; i++) {
    fl.trace(fl.getProject().items[i].displayName);
}
```

関連項目

[fl.getProject\(\)](#), [ProjectItem オブジェクト](#)

project.name

使用できるバージョン

Flash 8

シンタックス

project.name

説明

プロパティ。[プロジェクト] パネルに表示されるプロジェクトの名前です。

例

次の例では、[プロジェクト] パネルに表示する新しい名前を指定します。

```
fl.getProject().name = "New project name";
```

関連項目

[fl.getProject\(\), project.projectURI](#)

project.projectURI

使用できるバージョン

Flash 8

シンタックス

project.projectURI

説明

読み取り専用プロパティ。プロジェクトファイルのパスと名前を表すストリングです。file:/// URI で表します。

例

次の例では、現在開いているプロジェクトファイルのパスと名前を表示します。名前とパスが[出力] パネルに表示されます。

```
fl.trace("Project is located at: " + fl.getProject().projectURI);
```

関連項目

[fl.getProject\(\), project.name](#)

project.publishProject()

使用できるバージョン

Flash 8

シンタックス

```
project.publishProject()
```

パラメータ

なし

戻り値

プロジェクトが正常にパブリッシュされたかどうかを示すブール値。

説明

メソッド。プロジェクト内の FLA ファイルをパブリッシュします。

例

次の例では、プロジェクトをパブリッシュできることを確認した後でプロジェクトをパブリッシュし、プロジェクトがパブリッシュされたかどうかを [出力] パネルに示します。

```
if (fl.getProject().canPublishProject()) {  
    var bSucceeded = fl.getProject().publishProject();  
}  
fl.trace(bSucceeded);
```

関連項目

[fl.getProject\(\)](#), [project.canPublishProject\(\)](#), [projectItem.publish\(\)](#)

project.testProject()

使用できるバージョン

Flash 8

シンタックス

```
project.testProject()
```

パラメータ

なし

戻り値

プロジェクトが正常にテストされたかどうかを示すブール値。

説明

メソッド。プロジェクトをテストします。プロジェクトには、テストするデフォルトドキュメントが含まれている必要があります。

例

次の例では、プロジェクトをテストできることを確認した後でプロジェクトをテストし、プロジェクトがテストされたかどうかを [出力] パネルに示します。

```
if (fl.getProject().canTestProject()) {  
    var bSucceeded = fl.getProject().testProject();  
}  
fl.trace(bSucceeded);
```

関連項目

[fl.getProject\(\)](#), [project.canTestProject\(\)](#), [project.defaultItem](#),
[projectItem.test\(\)](#)

ProjectItem オブジェクト

使用できるバージョン

Flash 8

説明

ProjectItem オブジェクトは、プロジェクトに追加されたアイテム (ディスク上のファイル) を表します。このオブジェクトは、Project オブジェクトのプロパティです。[project.items](#) を参照してください。次のコマンドを使用して、ProjectItem オブジェクトを返すことができます。

- 新しいファイルをプロジェクトに追加するには、[project.addFile\(\)](#) を使用します。
- プロジェクトに既に追加されているアイテムを検索するには、[project.findProjectItem\(\)](#) を使用します。

ProjectItem オブジェクトのメソッド一覧

ProjectItem オブジェクトでは次のメソッドを使用できます。

メソッド	説明
projectItem.canPublish()	プロジェクトアイテムをパブリッシュできるかどうかを調べます。
projectItem.canTest()	プロジェクトアイテムをテストできるかどうかを調べます。
projectItem.publish()	プロジェクトアイテムをパブリッシュします。
projectItem.test()	プロジェクトアイテムをテストします。

ProjectItem オブジェクトのプロパティ一覧

ProjectItem オブジェクトでは次のプロパティを使用できます。

プロパティ	説明
projectItem.displayName	読み取り専用。プロジェクトアイテムの名前を指定するストリングです。
projectItem.isMissing	読み取り専用。ファイルがディスク上で見つからないかどうかを指定するブール値です。
projectItem.itemURI	読み取り専用。プロジェクトアイテムのパスと名前を指定するストリングです。
projectItem.publishProfile	プロジェクトアイテム (FLA ファイル) をパブリッシュするときに使用するパブリッシュプロファイルを指定するストリング。

projectItem.canPublish()

使用できるバージョン

Flash 8

シンタックス

```
projectItem.canPublish()
```

パラメータ

なし

戻り値

プロジェクトアイテムをパブリッシュできるかどうかを示すブール値。

説明

メソッド。アイテムをパブリッシュできるかどうかを調べます。アイテムがFLA ファイルの場合、アイテムをパブリッシュできます。

例

次の例では、プロジェクト内の最初のアイテムをパブリッシュできない場合に [出力] パネルにメッセージを表示します。

```
var item = fl.getProject().items[0];
if (!item.canPublish()) {
    fl.trace(item.displayName + " cannot be published!");
}
```

関連項目

[fl.getProject\(\)](#), [project.canPublishProject\(\)](#), [project.items](#),
[projectItem.publish\(\)](#)

projectItem.canTest()

使用できるバージョン

Flash 8

シンタックス

```
projectItem.canTest()
```

パラメータ

なし

戻り値

プロジェクトアイテムをテストできるかどうかを示すブール値。

説明

メソッド。アイテムをテストできるかどうかを調べます。アイテムが FLA ファイルまたは HTML ファイルの場合、アイテムをテストできます。

例

次の例では、プロジェクト内の最初のアイテムをテストできない場合に [出力] パネルにメッセージを表示します。

```
var item = fl.getProject().items[0];
if (!item.canTest()) {
    fl.trace(item.name + " cannot be tested!");
}
```

関連項目

[fl.getProject\(\)](#), [project.canTestProject\(\)](#), [project.items](#), [projectItem.test\(\)](#)

projectItem.displayName

使用できるバージョン

Flash 8

シンタックス

`projectItem.displayName`

説明

読み取り専用プロパティ。"file fla" などのプロジェクトアイテムの名前を指定するストリングです。

例

次の例では、プロジェクト内のすべてのファイルの名前を [出力] パネルに表示します。

```
fl.trace("These are all the files in the project: ");
var files = fl.getProject().items;
for (i = 0; i < files.length; i++) {
    fl.trace(files[i].displayName + " ");
}
```

関連項目

[fl.getProject\(\)](#), [project.items](#), [projectItem.itemURI](#)

projectItem.isMissing

使用できるバージョン

Flash 8

シンタックス

`projectItem.isMissing`

説明

読み取り専用プロパティ。ファイルがディスク上で見つからないかどうかを示すブール値です (たとえば、アイテムが移動または削除されたか、名前が変更された場合など)。

例

次の例では、特定のファイルがディスクの予想されるフォルダ内にあるかどうかを示すメッセージを [出力] パネルに表示します。

```
var item = fl.getProject().findProjectItem("file:///C:/Projects/files/
    DynamicHighAscii.flc");
fl.trace("DynamicHighAscii.flc is missing: " + item.isMissing);
```

関連項目

[fl.getProject\(\)](#), [project.findProjectItem\(\)](#), [project.items](#)

projectItem.itemURI

使用できるバージョン

Flash 8

シンタックス

`projectItem.itemURI`

説明

読み取り専用プロパティ。file:/// URI として指定され、プロジェクトアイテムのパスと名前を表すストリングです。フォルダアイテムには空のストリング ("") が含まれます。

例

次の例では、プロジェクト内の各アイテムのパスと名前を [出力] パネルに表示します。

```
files = fl.getProject().items;
for (i = 0; i < files.length; i++) {
    fl.trace(files[i].itemURI);
}
```

関連項目

[fl.getProject\(\), projectItem.displayName, project.items](#)

projectItem.publish()

使用できるバージョン

Flash 8

シンタックス

```
projectItem.publish()
```

パラメータ

なし

戻り値

ブール値。成功した場合は true を返します。それ以外の場合は false を返します。

説明

メソッド。プロジェクトアイテムをパブリッシュします。パブリッシュできるのは FLA ファイルのみです。

例

次の例では、プロジェクト内のすべてのパブリッシュ可能なアイテムをパブリッシュします。

```
for (var i in fl.getProject().items) {  
    var item = fl.getProject().items[i];  
    if (item.canPublish()) {  
        item.publish();  
    }  
}
```

関連項目

[fl.getProject\(\), project.canPublishProject\(\), project.items,](#)
[projectItem.canPublish\(\), projectItem.publishProfile](#)

projectItem.publishProfile

使用できるバージョン

Flash 8

シンタックス

```
projectItem.publishProfile
```

説明

プロパティ。プロジェクトアイテム (FLA ファイル) をパブリッシュするときに使用するパブリッシュプロファイルを指定する文字列です。パブリッシュプロファイルは、アイテム内の既存のプロファイルである必要があります。それ以外の場合、その後の `projectItem.publish()` の呼び出しが失敗します。詳細については、[projectItem.publish\(\)](#) を参照してください。

アイテムが FLA ファイルではない場合、このプロパティは空の文字列 ("") になり、このプロパティの設定が失敗します。

例

次の例では、プロジェクト内のすべてのアイテムのパブリッシュプロファイルをアイテム内に既に存在する指定されたプロファイルに設定し、その後で各アイテムをパブリッシュします。プロファイルがファイル内に存在しない場合、そのファイルはパブリッシュされません。

```
var items = fl.getProject().items;
for ( i = 0 ; i < items.length ; i++ ) {
    items[i].publishProfile = "mySpecialProfile";
    items[i].publish();
}
```

関連項目

[fl.getProject\(\)](#), [project.canPublishProject\(\)](#), [project.items](#),
[projectItem.canPublish\(\)](#), [projectItem.publish\(\)](#)

projectItem.test()

使用できるバージョン

Flash 8

シンタックス

`projectItem.test()`

パラメータ

なし

戻り値

アイテムが正常にテストされたかどうかを示すブール値。

説明

メソッド。プロジェクトアイテムをテストします。アイテムが FLA ファイルまたは HTML ファイルではないために、テストが失敗した場合、このメソッドは `false` を返します。

例

次の例では、プロジェクト内のすべての FLA ファイルと HTML ファイルをテストします。

```
for (var i in fl.getProject().items) {  
    var item = fl.getProject().items[i];  
    if (item.canTest()) {  
        item.test();  
    }  
}
```

関連項目

[fl.getProject\(\)](#), [project.canTestProject\(\)](#), [project.items](#),
[projectItem.canTest\(\)](#)

Rectangle オブジェクト

継承 [Element オブジェクト](#) > [Shape オブジェクト](#) > Rectangle オブジェクト

使用できるバージョン

Flash CS3 Professional

説明

Rectangle オブジェクトは、矩形プリミティブツールを使用して描画されるシェイプです。アイテムが Rectangle オブジェクトかどうかを調べるには、[shape.isRectangleObject](#) を使用します。

Rectangle オブジェクトのプロパティ一覧

Rectangle オブジェクトでは、[Shape オブジェクト](#) のプロパティの他に、以下のプロパティを使用できます。Rectangle オブジェクトのプロパティを設定するには、[document.setRectangleObjectProperty\(\)](#) を使用します。

プロパティ	説明
RectangleObject.bottomLeftRadius	読み取り専用。Rectangle オブジェクトの左下隅の半径を設定する浮動小数値。
RectangleObject.bottomRightRadius	読み取り専用。Rectangle オブジェクトの右下隅の半径を設定する浮動小数値。
RectangleObject.lockFlag	読み取り専用。矩形の角をそれぞれ異なる半径値に設定できるかどうかを判定するブール値。
RectangleObject.topLeftRadius	読み取り専用。矩形のすべての角の半径、または Rectangle オブジェクトの左上隅の半径のみを設定する浮動小数値。
RectangleObject.topRightRadius	読み取り専用。Rectangle オブジェクトの右上隅の半径を設定する浮動小数値。

RectangleObject.bottomLeftRadius

使用できるバージョン

Flash CS3 Professional

シンタックス

[RectangleObject.bottomLeftRadius](#)

説明

読み取り専用プロパティ。Rectangle オブジェクトの左下隅の半径を設定する浮動小数値です。

`RectangleObject.lockFlag` が `true` の場合、この値を設定しても効果はありません。

この値を設定するには、`document.setRectangleObjectProperty()` を使用します。

関連項目

`document.setRectangleObjectProperty()`, `RectangleObject.bottomRightRadius`,
`RectangleObject.lockFlag`, `RectangleObject.topLeftRadius`,
`RectangleObject.topRightRadius`

RectangleObject.bottomRightRadius

使用できるバージョン

Flash CS3 Professional

シンタックス

`RectangleObject.bottomRightRadius`

説明

読み取り専用プロパティ。Rectangle オブジェクトの右下隅の半径を設定する浮動小数値です。

`RectangleObject.lockFlag` が `true` の場合、この値を設定しても効果はありません。

この値を設定するには、`document.setRectangleObjectProperty()` を使用します。

関連項目

`document.setRectangleObjectProperty()`, `RectangleObject.bottomLeftRadius`,
`RectangleObject.lockFlag`, `RectangleObject.topLeftRadius`,
`RectangleObject.topRightRadius`

RectangleObject.lockFlag

使用できるバージョン

Flash CS3 Professional

シンタックス

`RectangleObject.lockFlag`

説明

読み取り専用プロパティ。矩形の角をそれぞれ異なる半径値に設定できるかどうかを判定するブール値です。この値が `true` の場合は、すべての角に `RectangleObject.topLeftRadius` の値が割り当てられます。 `false` の場合は、それぞれの角の半径を別々に設定できます。

この値を設定するには、 `document.setRectangleObjectProperty()` を使用します。

関連項目

`document.setRectangleObjectProperty()`, `RectangleObject.bottomLeftRadius`, `RectangleObject.bottomRightRadius`, `RectangleObject.topLeftRadius`, `RectangleObject.topRightRadius`

RectangleObject.topLeftRadius

使用できるバージョン

Flash CS3 Professional

シンタックス

`RectangleObject.topLeftRadius`

説明

読み取り専用プロパティ。矩形のすべての角の半径 (`RectangleObject.lockFlag` が `true` の場合)、または左上隅の半径のみ (`RectangleObject.lockFlag` が `false` の場合) を設定する浮動小数値です。

この値を設定するには、 `document.setRectangleObjectProperty()` を使用します。

関連項目

`document.setRectangleObjectProperty()`, `RectangleObject.bottomLeftRadius`, `RectangleObject.bottomRightRadius`, `RectangleObject.lockFlag`, `RectangleObject.topRightRadius`

RectangleObject.topRightRadius

使用できるバージョン

Flash CS3 Professional

シンタックス

`RectangleObject.topRightRadius`

説明

読み取り専用プロパティ。Rectangle オブジェクトの右上隅の半径を設定する浮動小数値です。

`RectangleObject.lockFlag` が `true` の場合、この値を設定しても効果はありません。

この値を設定するには、`document.setRectangleObjectProperty()` を使用します。

関連項目

`document.setRectangleObjectProperty()`, `RectangleObject.bottomLeftRadius`,
`RectangleObject.bottomRightRadius`, `RectangleObject.lockFlag`,
`RectangleObject.topLeftRadius`

Screen オブジェクト

使用できるバージョン

Flash MX 2004

説明

Screen オブジェクトは、スライドまたはフォームドキュメント内の1つのスクリーンを表します。このオブジェクトには、スライドまたはフォームに関連するプロパティが含まれます。ドキュメント内のすべての Screen オブジェクトの配列にアクセスするには、次のコードを使用します。

```
fl.getDocumentDOM().screenOutline.screens
```

Screen オブジェクトのプロパティ一覧

Screen オブジェクトには次のプロパティがあります。

プロパティ	説明
<code>screen.accName</code>	[アクセシビリティ] パネルの [名前] フィールドと等しいストリング。
<code>screen.childScreens</code>	読み取り専用。このスクリーンの子スクリーンの配列です。子スクリーンがない場合、配列は空です。
<code>screen.description</code>	[アクセシビリティ] パネルの [説明] フィールドと等しいストリング。
<code>screen.forceSimple</code>	オブジェクトの子のアクセシビリティを有効または無効にするブール値。
<code>screen.hidden</code>	スクリーンが表示されるかどうかを指定するブール値。
<code>screen.instanceName</code>	読み取り専用。ActionScript からオブジェクトにアクセスするために使用されるインスタンス名を表すストリングです。
<code>screen.name</code>	読み取り専用。スクリーンの名前を表すストリングです。
<code>screen.nextScreen</code>	読み取り専用。親の childScreens 配列内の次のピアスクリーンを表すオブジェクトです。
<code>screen.parameters</code>	読み取り専用。スクリーンのプロパティインスペクタからアクセス可能な ActionScript 2.0 プロパティの配列です。
<code>screen.parentScreen</code>	読み取り専用。親スクリーンを表すオブジェクトです。
<code>screen.prevScreen</code>	読み取り専用。親の childScreens 配列内の前のピアスクリーンを表すオブジェクトです。
<code>screen.silent</code>	オブジェクトがアクセス可能かどうかを指定するブール値です。

プロパティ	説明
<code>screen.tabIndex</code>	[アクセシビリティ] パネルの [タブインデックス] フィールドと等価です。
<code>screen.timeline</code>	読み取り専用。スクリーンの Timeline オブジェクトです。詳細については、 Timeline オブジェクトを参照してください 。

screen.accName

使用できるバージョン

Flash MX 2004

シンタックス

`screen.accName`

説明

プロパティ。[アクセシビリティ] パネルの [名前] フィールドと等しいストリングです。スクリーンリーダーは、名前を読み上げてオブジェクトを識別します。

例

次の例では、オブジェクトの名前の値を `theName` 変数に格納します。

```
var theName = fl.getDocumentDOM().screenOutline.screens[1].accName;
```

次の例では、オブジェクトの名前を "Home Button" に設定します。

```
fl.getDocumentDOM().screenOutline.screens[1].accName = 'Home Button';
```

screen.childScreens

使用できるバージョン

Flash MX 2004

シンタックス

`screen.childScreens`

説明

読み取り専用プロパティ。このスクリーンの子スクリーンの配列です。子スクリーンがない場合、配列は空です。

例

次の例では、現在のドキュメントがスライドまたはフォームかどうかを確認し、その場合は、子スクリーンの配列を `myChildren` 変数に格納し、それらの名前を [出力] パネルに表示します。

```
var myChildren = new Array();
if(fl.getDocumentDOM().allowScreens) {
    var myParent = fl.getDocumentDOM().screenOutline.rootScreen.name
    for (i in fl.getDocumentDOM().screenOutline.rootScreen.childScreens) {
        myChildren.push("
"+fl.getDocumentDOM().screenOutline.rootScreen.childScreens[i].name);
    }
    fl.trace(" The child screens of "+myParent+" are "+myChildren+". ");
}
```

screen.description

使用できるバージョン

Flash MX 2004

シンタックス

`screen.description`

説明

プロパティ。[アクセシビリティ] パネルの [説明] フィールドと等しいストリングです。この説明がスクリーンリーダーによって読み上げられます。

例

次の例では、オブジェクトの説明を取得し、`theDescription` 変数に格納します。

```
var theDescription = fl.getDocumentDOM().screenOutline.screens[1].description;
```

次の例では、オブジェクトの説明を "This is Screen 1" に設定します。

```
fl.getDocumentDOM().screenOutline.screens[1].description = "This is Screen 1"
```

screen.forceSimple

使用できるバージョン

Flash MX 2004

シンタックス

`screen.forceSimple`

説明

プロパティ。オブジェクトの子のアクセシビリティを有効または無効にするブール値です。[アクセシビリティ] パネルの [子オブジェクトをアクセス可能にする] の逆ロジックに等しくなります。つまり、forceSimple が true の場合、[子オブジェクトをアクセス可能にする] が選択解除されていることと同じです。forceSimple が false の場合、[子オブジェクトをアクセス可能にする] が選択されていることと同じです。

例

次の例では、forceSimple の値を areChildrenAccessible 変数に格納します (値が false の場合は、オブジェクトの子がアクセス可能なことを意味します)。

```
var areChildrenAccessible =  
    fl.getDocumentDOM().screenOutline.screens[1].forceSimple
```

次の例では、オブジェクトの子をアクセス可能にします。

```
fl.getDocumentDOM().screenOutline.screens[1].forceSimple = false;
```

screen.hidden

使用できるバージョン

Flash MX 2004

シンタックス

screen.hidden

説明

プロパティ。スクリーンが表示されるかどうかを指定するブール値です。hidden プロパティが true に設定されているスクリーンは、他のどのスクリーンにも表示されません。

例

次の例では、アウトライン内の最初のスクリーンが非表示かどうかを確認し、それに応じてスクリーンの表示 / 非表示を変更します。その後で、変更前にスクリーンが表示されていたかどうかを示すメッセージを [出力] パネルに表示します。

```
if (fl.getDocumentDOM().screenOutline.screens[0].hidden) {  
    fl.getDocumentDOM().screenOutline.setScreenProperty("hidden", false);  
    fl.trace(fl.getDocumentDOM().screenOutline.screens[0].name+" had its 'hidden'  
    property set to 'false'");  
}  
else {  
    fl.getDocumentDOM().screenOutline.setScreenProperty("hidden", true);  
    fl.trace(fl.getDocumentDOM().screenOutline.screens[0].name+" had its 'hidden'  
    property set to 'true'");  
}
```

screen.instanceName

使用できるバージョン

Flash MX 2004

シンタックス

screen.instanceName

説明

読み取り専用プロパティ。ActionScript からオブジェクトにアクセスするために使用されるインスタンス名を表すストリングです。

例

次の例では、現在のドキュメントがスクリーンを使用できるかどうか (スライドまたはフォームか) を確認します。その後で、配列内の最初の子スクリーンの instanceName 値を myInstanceName 変数に代入し、[出力] パネルを開いて、スクリーンのインスタンス名を表示します。

```
var myChildren = new Array();
if(fl.getDocumentDOM().allowScreens) {
    var myInstanceName =
        fl.getDocumentDOM().screenOutline.rootScreen.childScreens[0].instanceName;
    fl.trace(" The instanceName is "+myInstanceName+". ");
}
```

screen.name

使用できるバージョン

Flash MX 2004

シンタックス

screen.name

説明

読み取り専用プロパティ。スクリーンの名前を表すストリングです。

例

次の例では、現在のドキュメントが、スクリーンを使用できるかどうか (スライドまたはフォームドキュメントか) を確認します。その後で、配列内の最初の子スクリーンの name 値を myName 変数に代入し、[出力] パネルを開いて、スクリーンの名前を表示します。

```
var myChildren = new Array();
if(fl.getDocumentDOM().allowScreens) {
    var myName =
        fl.getDocumentDOM().screenOutline.rootScreen.childScreens[0].name;
    fl.trace("The name of the screen is "+myName+". ");
}
```

screen.nextScreen

使用できるバージョン

Flash MX 2004

シンタックス

screen.nextScreen

説明

読み取り専用プロパティ。親の childScreens 配列内の次のピアスクリーンを表すオブジェクトです。つまり、子スクリーンの配列で配列内の次のスクリーンに移動することで、screen.nextScreen が検出されます。[screen.prevScreen](#) を参照してください。

ピアスクリーンがない場合、値は null になります。

例

次の例では、初めに現在のドキュメントがスライドまたはフォームかどうかを確認し、その場合は、スクリーンのシーケンスを取得して [出力] パネルに表示します。

```
if(fl.getDocumentDOM().allowScreens) {  
    var myCurrent =  
        fl.getDocumentDOM().screenOutline.rootScreen.childScreens[0].name;  
    var myNext =  
        fl.getDocumentDOM().screenOutline.rootScreen.childScreens[0].nextScreen.name;  
    fl.trace(" The next screen to "+myCurrent+" is "+myNext+" ");  
}
```

screen.parameters

使用できるバージョン

Flash MX 2004

シンタックス

screen.parameters

説明

読み取り専用プロパティ。スクリーンのプロパティインスペクタからアクセスできる ActionScript 2.0 プロパティの配列です。

例

次の例では、アウトライン内の 2 番目のスクリーンのパラメータを parms 変数に格納し、"some value" 値を最初のプロパティに代入します。


```
var parms = fl.getDocumentDOM().screenOutline.screens[1].parameters;  
parms[0].value = "some value";
```

関連項目

[Parameter オブジェクト](#)

screen.parentScreen

使用できるバージョン

Flash MX 2004

シンタックス

screen.parentScreen

説明

読み取り専用プロパティ。親スクリーンを表すオブジェクトです。parentScreen が null の場合、スクリーンは最上位のスクリーンです。

例

次の例では、childScreens および parentScreen プロパティの値を変数に格納し、それらの値とそれらの親子関係を [出力] パネルに表示します。

```
if(fl.getDocumentDOM().allowScreens) {  
    var myCurrent =  
        fl.getDocumentDOM().screenOutline.rootScreen.childScreens[1].name;  
    var myParent =  
        fl.getDocumentDOM().screenOutline.rootScreen.childScreens[1].parentScreen.name;  
    fl.trace(" The parent screen to "+myCurrent+" is "+myParent+" ");  
}
```

screen.prevScreen

使用できるバージョン

Flash MX 2004

シンタックス

screen.prevScreen

説明

読み取り専用プロパティ。親の childScreens 配列内の前のピアスクリーンを表すオブジェクトです。ピアスクリーンがない場合、値は null になります。[screen.nextScreen](#) も参照してください。

例

次の例では、現在のドキュメントがスライドまたはフォームかどうかを確認し、その場合は、スクリーンのシーケンスを取得して [出力] パネルに表示します。

```
if(fl.getDocumentDOM().allowScreens) {  
    var myCurrent =  
        fl.getDocumentDOM().screenOutline.rootScreen.childScreens[1].name;  
    var myNext =  
        fl.getDocumentDOM().screenOutline.rootScreen.childScreens[1].prevScreen.name;  
    fl.trace(" The previous screen to "+myCurrent+" is "+myNext+". ");  
}
```

screen.silent

使用できるバージョン

Flash MX 2004

シンタックス

screen.silent

説明

プロパティ。オブジェクトがアクセス可能かどうかを指定するブール値です。これは、[アクセシビリティ] パネルの [オブジェクトをアクセス可能にする] の設定の逆ロジックと等価です。つまり、silent が true の場合、[アクセシビリティ] パネルの [オブジェクトをアクセス可能にする] が選択解除されていることと同じです。silent が false の場合、[アクセシビリティ] パネルの [オブジェクトをアクセス可能にする] が選択されていることと同じです。

例

次の例では、オブジェクトの silent 値を取得します (値が false の場合、オブジェクトがアクセス可能なことを意味します)。

```
var isSilent = fl.getDocumentDOM().screenOutline.screens[1].silent;
```

次の例では、オブジェクトをアクセス可能に設定します。

```
fl.getDocumentDOM().screenOutline.screens[1].silent = false;
```

screen.tabIndex

使用できるバージョン

Flash MX 2004

シンタックス

screen.tabIndex

説明

プロパティ。[アクセシビリティ] パネルの [タブインデックス] フィールドと等価です。この値を使用して、ユーザーが Tab キーを押したときにオブジェクトがアクセスされる順序を決定することができます。

例

次の例では、オブジェクトのタブインデックスを取得します。

```
var theTabIndex = fl.getDocumentDOM().screenOutline.screens[1].tabIndex;
```

次の例では、オブジェクトのタブインデックスを 1 に設定します。

```
fl.getDocumentDOM().screenOutline.screens[1].tabIndex = 1;
```

screen.timeline

使用できるバージョン

Flash MX 2004

シンタックス

screen.timeline

説明

読み取り専用プロパティ。スクリーンの [Timeline オブジェクト](#) です。

例

次の例では、現在のスライドドキュメントの screenOutline プロパティを取得し、最初のスクリーンの timeline プロパティの配列を myArray に代入し、それらのプロパティを [出力] パネルに表示します。

```
myArray = new Array();
if(fl.getDocumentDOM().screenOutline) {
    for(i in fl.getDocumentDOM().screenOutline.screens[0].timeline) {
        myArray.push(" "+i+" : "+fl.getDocumentDOM().screenOutline.screens[0].timeline[i]+" ");
    }
    fl.trace("Here are the properties of the screen named "+
        fl.getDocumentDOM().screenOutline.screens[0].name+": "+myArray);
}
```

ScreenOutline オブジェクト

使用できるバージョン

Flash MX 2004

説明

ScreenOutline オブジェクトは、スライドまたはフォームドキュメント内のスクリーンのグループを表します。このオブジェクトにアクセスするには、`fl.getDocumentDOM().screenOutline` を使用します。

ScreenOutline オブジェクトは、ドキュメントがスライドまたはフォームドキュメントの場合にのみ存在します。そのため、プロパティにアクセスする前に、次の例に示すように `document.allowScreens()` を使用して、スクリーンドキュメントが存在することを確認します。

```
if(fl.getDocumentDOM().allowScreens) {  
    var myName =  
        fl.getDocumentDOM().screenOutline.rootScreen.childScreens[0].name;  
    fl.trace("The name of the screen is " + myName + ".");  
}
```

ScreenOutline オブジェクトのメソッド一覧

ScreenOutline オブジェクトでは次のメソッドを使用できます。

メソッド	説明
<code>screenOutline.copyScreenFromFile()</code>	すべてのスクリーンまたは指定されたスクリーンとその子を、指定されたドキュメントから現在選択されているスクリーンの下に挿入します。
<code>screenOutline.deleteScreen()</code>	現在選択されているスクリーンまたは指定されたスクリーンとそのスクリーンの子を削除します。
<code>screenOutline.duplicateScreen()</code>	現在選択されているスクリーンまたは指定されたスクリーンを複製します。
<code>screenOutline.getSelectedScreens()</code>	スクリーンアウトラインで現在選択されている Screen オブジェクトの配列を返します。
<code>screenOutline.insertNestedScreen()</code>	特定のタイプのネストされたスクリーンをスクリーンアウトラインの特定の場所に挿入します。
<code>screenOutline.insertScreen()</code>	指定されたタイプの新しい空のスクリーンをドキュメントの指定された場所に挿入します。

メソッド	説明
<code>screenOutline.moveScreen()</code>	指定されたスクリーンを <i>referenceScreen</i> パラメータの値を基準として、前または後、あるいは最初の子または最後の子として移動します。
<code>screenOutline.renameScreen()</code>	指定されたスクリーンの名前を新しい名前に変更します。
<code>screenOutline.setCurrentScreen()</code>	スクリーンアウトラインで現在選択されているスクリーンを、指定されたスクリーンにします。
<code>screenOutline.setScreenProperty()</code>	選択されたスクリーンの指定されたプロパティを指定された値に設定します。
<code>screenOutline.setSelectedScreens()</code>	スクリーンアウトラインペインで、指定されたスクリーンを選択します。

ScreenOutline オブジェクトのプロパティ一覧

ScreenOutline オブジェクトでは次のプロパティを使用できます。

プロパティ	説明
<code>screenOutline.currentScreen</code>	Screen オブジェクト 。現在選択されているスクリーンです。
<code>screenOutline.rootScreen</code>	読み取り専用。スクリーンアウトラインの最初のスクリーンです。
<code>screenOutline.screens</code>	読み取り専用。ドキュメントに含まれるトップレベルの Screen オブジェクトの配列です (Screen オブジェクト を参照してください)。

screenOutline.copyScreenFromFile()

使用できるバージョン

Flash MX 2004

シンタックス

```
screenOutline.copyScreenFromFile(fileURI [, screenName])
```

パラメータ

fileURI file:/// URI として表され、ドキュメント内にコピーするスクリーンが含まれるオーサリングファイルのファイル名を指定するストリング。

screenName コピーするスクリーンの名前。*screenName* パラメータが存在している場合、そのスクリーンとその子がコピーされます。*screenName* が指定されていない場合、ドキュメント全体がコピーされます。このパラメータはオプションです。

戻り値

なし。ファイルが見つからないか有効な FLA ファイルではない場合、または指定されたスクリーンが見つからない場合、エラーが報告され、スクリプトはキャンセルされます。

説明

メソッド。すべてのスクリーンまたは指定されたスクリーンとその子を、指定されたドキュメントから現在選択されているスクリーンの下に挿入します。複数のスクリーンが選択されている場合、スクリーンは最後の選択されたスクリーンの下に兄弟ノードとして挿入されます。

例

次の例では、デスクトップ上の "myTarget fla" ファイルから "slide1" スクリーンを現在のドキュメントにコピーします。〈ユーザー名〉は実際のユーザー名に置き換えてください。

```
fl.getDocumentDOM().screenOutline.copyScreenFromFile("file:///C:/Documents and Settings/<ユーザー名>/Desktop/myTarget fla", "slide1");
```

screenOutline.currentScreen

使用できるバージョン

Flash MX 2004

シンタックス

screenOutline.currentScreen

説明

プロパティ。Screen オブジェクト。現在選択されているスクリーンです。[Screen オブジェクト](#)を参照してください。

例

次の例では、currentScreen オブジェクトを myScreen 変数に格納し、スクリーンの名前を [出力] パネルに表示します。

```
var myScreen = fl.getDocumentDOM().screenOutline.currentScreen;  
fl.trace(myScreen.name);
```

screenOutline.deleteScreen()

使用できるバージョン

Flash MX 2004

シンタックス

```
screenOutline.deleteScreen([screenName])
```

パラメータ

screenName 削除するスクリーンの名前を指定するストリング。*screenName* の値を指定しない場合は、現在選択されているスクリーンとそれらの子が削除されます。このパラメータはオプションです。

戻り値

なし

説明

メソッド。現在選択されているスクリーンまたは指定されたスクリーンとそのスクリーンの子を削除します。

例

次の例では、apple というスクリーンとそのすべての子を削除します。

```
fl.getDocumentDOM().screenOutline.deleteScreen("apple");
```

screenOutline duplicateScreen()

使用できるバージョン

Flash MX 2004

シンタックス

```
screenOutline.duplicateScreen([screenName])
```

パラメータ

screenName 複製するスクリーンの名前を指定するストリング値。*screenName* の値を指定しない場合は、現在選択されているスクリーンが複製されます。このパラメータはオプションです。

戻り値

ブール値。スクリーンが正常に複製された場合は true を返します。それ以外の場合は false を返します。

説明

メソッド。現在選択されているスクリーンまたは指定されたスクリーンを複製します。複製されたスクリーンには、元の名前に `_copy` を付加したデフォルトの名前が付けられます。たとえば、`Screen_copy`、`Screen_copy2` などのようになります。複数のスクリーンを複製した場合、複製は、スクリーンアウトライン階層で、選択されたスクリーンのうち最下位にあるスクリーンのすぐ下に置かれます。

例

次の例では、`apple` というスクリーンを複製します。

```
fl.getDocumentDOM().screenOutline.duplicateScreen("apple");
```

screenOutline.getSelectedScreens()

使用できるバージョン

Flash MX 2004

シンタックス

```
screenOutline.getSelectedScreens()
```

パラメータ

なし

戻り値

選択された `Screen` オブジェクトの配列 ([Screen オブジェクト](#) を参照)。

説明

メソッド。スクリーンアウトラインで現在選択されている `Screen` オブジェクトの配列を返します。

例

次の例では、選択された `Screen` オブジェクトを `myArray` 変数に格納し、スクリーン名を [出力] パネルに表示します。

```
var myArray = fl.getDocumentDOM().screenOutline.getSelectedScreens();
for (var i in myArray) {
    fl.trace(myArray[i].name)
}
```


screenOutline.insertNestedScreen()

使用できるバージョン

Flash MX 2004

シンタックス

```
screenOutline.insertNestedScreen([name [, referenceScreen  
    [, screenTypeName]])
```

パラメータ

name 挿入する新しいスクリーンの名前を示すストリング。空の名前を指定すると、Slide *n* または Form *n* (*n* は、最も小さい使用可能な一意の番号) などのデフォルトのスクリーン名のスクリーンが挿入されます。このパラメータはオプションです。

referenceScreen 新しいスクリーンを子として挿入する挿入先のスクリーンの名前を示すストリング。このパラメータを省略した場合、新しいスクリーンは、現在選択されているスクリーンの子として挿入されます。このパラメータはオプションです。

screenTypeName 新しいネストされたスクリーンに割り当てるスクリーンタイプを指定するストリング。このスクリーンのスクリーンタイプとクラス名が設定されます。有効な値は、"Form" および "Slide" です。このパラメータはオプションです。このパラメータを省略した場合、タイプは、親スクリーンから継承されます。

戻り値

[Screen](#) オブジェクト。

説明

メソッド。特定のタイプのネストされたスクリーンをスクリーンアウトラインの特定の場所に挿入します。

例

次の例では、slide2 を slide1 の子として挿入します。

```
fl.getDocumentDOM().screenOutline.insertNestedScreen("slide2", "slide1",  
    "Slide");
```

screenOutline.insertScreen()

使用できるバージョン

Flash MX 2004

シンタックス

```
screenOutline.insertScreen([name [, referenceScreen [, screenTypeName]])
```

パラメータ

name 挿入する新しいスクリーンの名前を示すストリング。このパラメータが省略されている場合、メソッドは、**Slide *n*** または **Form *n*** (*n* は最も小さい使用可能な一意の番号) などのデフォルトのスクリーン名のスクリーンを挿入します。このパラメータはオプションです。

referenceScreen 新しいスクリーンの前のスクリーンの名前を示すストリング。このパラメータが省略されている場合、新しいスクリーンは、現在選択されているスクリーンの後に挿入されます。*referenceScreen* パラメータが、子オブジェクトを示す場合、新しいスクリーンは、子スクリーンのピアになり、同じ親の子スクリーンになります。このパラメータはオプションです。

screenTypeName 新しいスクリーンに割り当てるスクリーンタイプを指定するストリング。このスクリーンのスクリーンタイプとクラス名が設定されます。有効な値は、"Form" および "Slide" です。このパラメータはオプションです。

戻り値

[Screen](#) オブジェクト。

説明

メソッド。指定されたタイプの新しい空のスクリーンをドキュメントの指定された場所に挿入します。

例

次の例では、**slide2** というフォームを **slide1** というスクリーンの後に挿入します。

```
fl.getDocumentDOM().screenOutline.insertScreen("slide2","slide1","Form");
```

次の例では、**slide4** というスライドを **slide3** というスライドの後に挿入します。

```
fl.getDocumentDOM().screenOutline.insertScreen("slide4","slide3","Slide");
```

screenOutline.moveScreen()

使用できるバージョン

Flash MX 2004

シンタックス

```
screenOutline.moveScreen(screenToMove, referenceScreen, position)
```

パラメータ

screenToMove 移動するスクリーンの名前を示すストリング。

referenceScreen *screenToMove* を配置する場所の近くにあるスクリーンを指定するストリング。

position *referenceScreen* を基準としてスクリーンを移動する場所を指定するストリング。有効な値は、"before"、"after"、"firstChild"、および "lastChild" です。

戻り値

ブール値。移動が成功した場合は true を返し、それ以外の場合は false を返します。

説明

メソッド。指定されたスクリーンを *referenceScreen* パラメータの値を基準として、前または後、あるいは最初の子または最後の子として移動します。

例

次の例では、slide2 の最初の子になるようにスクリーン slide1 を移動します。

```
fl.getDocumentDOM().screenOutline.moveScreen("slide1", "slide2", "firstChild");
```

screenOutline.renameScreen()

使用できるバージョン

Flash MX 2004

シンタックス

```
screenOutline.renameScreen(newScreenName [, oldScreenName  
    [, bDisplayError]])
```

パラメータ

newScreenName スクリーンの新しい名前を指定するストリング。

oldScreenName 変更する既存のスクリーンの名前を指定するストリング。指定されていない場合、現在選択されているスクリーンの名前が変更されます。このパラメータはオプションです。

bDisplayError ブール値。true に設定されている場合、エラーが発生したときにエラーメッセージが表示されます。たとえば、*newScreenName* に渡された値と同じ名前のスクリーンが既に存在している場合にエラーになります。デフォルト値は false です。

戻り値

ブール値。名前の変更が成功した場合は true を返し、それ以外の場合は false を返します。

説明

メソッド。指定されたスクリーンの名前を新しい名前に変更します。

例

次の例では、slide1 の名前を Intro に変更します。

```
fl.getDocumentDOM().screenOutline.renameScreen("Intro", "slide1");
```

screenOutline.rootScreen

使用できるバージョン

Flash MX 2004

シンタックス

```
screenOutline.rootScreen
```

説明

読み取り専用プロパティ。スクリーンアウトラインの最初のスクリーンです。

screenOutline.rootScreen を screenOutline.screens[0] のショートカットとして使用することができます。

例

次の例では、スクリーンアウトラインの最初のスクリーンの最初の子の名前を表示します。

```
var n = fl.getDocumentDOM().screenOutline.rootScreen.childScreens[0].name;  
fl.trace(n);
```

screenOutline.screens

使用できるバージョン

Flash MX 2004

シンタックス

screenOutline.screens

説明

読み取り専用プロパティ。ドキュメントに含まれるトップレベルの Screen オブジェクトの配列です ([Screen オブジェクト](#) を参照してください)。

例

次の例では、Screen オブジェクトの配列を myArray 変数に格納し、それらの名前を [出力] パネルに表示します。

```
var myArray = new Array();
if(fl.getDocumentDOM().allowScreens) {
    for(var i in fl.getDocumentDOM().screenOutline.screens) {
        myArray.push(" "+fl.getDocumentDOM().screenOutline.screens[i].name);
    }
    fl.trace(2"The screens array contains objects whose names are: "+myArray+"
");
}
```

screenOutline.setCurrentScreen()

使用できるバージョン

Flash MX 2004

シンタックス

screenOutline.setCurrentScreen(*name*)

パラメータ

name 現在選択されているスクリーンにするスクリーンの名前を指定するストリング。スクリーンが別のスクリーンの子の場合、パスまたは階層を指定する必要はありません。

戻り値

なし

説明

メソッド。スクリーンアウトラインで現在選択されているスクリーンを、指定されたスクリーンにします。

例

次の例では、現在のスクリーンを ChildOfSlide_1 というスクリーンにします。

```
fl.getDocumentDOM().screenOutline.setCurrentScreen("ChildOfSlide_1");
```

screenOutline.setScreenProperty()

使用できるバージョン

Flash MX 2004

シンタックス

```
screenOutline.setScreenProperty(property, value)
```

パラメータ

property 設定するプロパティを指定するストリング。

value プロパティの新しい値。値の型は、設定するプロパティによって異なります。

使用可能なプロパティと値の一覧は、[Screen オブジェクトのプロパティ一覧](#)を参照してください。

戻り値

なし

説明

メソッド。選択されたスクリーンの指定されたプロパティを指定された値に設定します。

例

次の例では、現在選択されているスクリーンを非表示から表示に変更します。

```
fl.getDocumentDOM().screenOutline.setScreenProperty("hidden", false);
```

screenOutline.setSelectedScreens()

使用できるバージョン

Flash MX 2004

シンタックス

```
screenOutline.setSelectedScreens(selection [, bReplaceCurrentSelection])
```

パラメータ

selection スクリーンアウトラインで選択するスクリーン名の配列。

bReplaceCurrentSelection ブール値。true の場合、現在の選択を選択解除できます。デフォルト値は true です。false の場合、指定したスクリーンが含まれるように現在の選択が拡張されます。このパラメータはオプションです。

戻り値

なし

説明

メソッド。スクリーンアウトラインで指定されたスクリーンを選択します。複数のスクリーンが指定されている場合、**selection** 配列の最後のインデックス値を持つスクリーンがステージ上でフォーカスを受け取ります。

例

次の例では、現在選択されているすべてのスクリーンを選択解除し、スクリーンアウトラインの `slide1`、`slide2`、`slide3`、および `slide4` を選択します。

```
myArray = new Array("slide1", "slide2", "slide3", "slide4");  
fl.getDocumentDOM().screenOutline.setSelectedScreens(myArray, true);
```

Shape オブジェクト

継承 [Element オブジェクト](#) › Shape オブジェクト

使用できるバージョン

Flash MX 2004

説明

Shape オブジェクトは、Element オブジェクトのサブクラスです。Shape オブジェクトを使用すると、ステージ上でジオメトリを操作または作成するときに、**drawing API** よりも正確に制御できます。便利な効果や描画コマンドをスクリプトで作成できるようにするためにこの制御が必要です([Element オブジェクト](#)を参照)。

シェイプまたはその従属部分を変更する Shape オブジェクトのすべてのメソッドとプロパティを正しく機能させるには、それらを `shape.beginEdit()` と `shape.endEdit()` の呼び出しの間に入れる必要があります。

Shape オブジェクトのメソッド一覧

Shape オブジェクトでは、[Element オブジェクト](#)のメソッドに加えて、次のメソッドを使用することができます。

メソッド	説明
<code>shape.beginEdit()</code>	編集セッションの開始を定義します。
<code>shape.deleteEdge()</code>	指定されたエッジを削除します。
<code>shape.endEdit()</code>	シェイプの編集セッションの終了を定義します。

Shape オブジェクトのプロパティ一覧

Shape オブジェクトでは、[Element オブジェクト](#)のプロパティに加えて、次のプロパティを使用することができます。

プロパティ	説明
<code>shape.contours</code>	読み取り専用。シェイプの Contour オブジェクトの配列です (Contour オブジェクト を参照)。
<code>shape.edges</code>	読み取り専用。Edge オブジェクトの配列です (Edge オブジェクト を参照)。
<code>shape.isDrawingObject</code>	読み取り専用。true の場合、シェイプは描画オブジェクトです。
<code>shape.isGroup</code>	読み取り専用。true の場合、シェイプはグループです。

プロパティ	説明
<code>shape.isOvalObject</code>	読み取り専用。 <code>true</code> の場合、シェイプはプリミティブ楕円オブジェクトです (楕円ツールを使用して作成)。
<code>shape.isRectangleObject</code>	読み取り専用。 <code>true</code> の場合、シェイプはプリミティブ矩形オブジェクトです (矩形ツールを使用して作成)。
<code>shape.vertices</code>	読み取り専用。 <code>Vertex</code> オブジェクトの配列です (Vertex オブジェクト を参照)。

shape.beginEdit()

使用できるバージョン

Flash MX 2004

シンタックス

```
shape.beginEdit()
```

パラメータ

なし

戻り値

なし

説明

メソッド。編集セッションの開始を定義します。`Shape` オブジェクトまたはその従属部分を変更するコマンドを発行する前にこのメソッドを使用する必要があります。

例

次の例では、現在選択されているシェイプを取得し、`edge` 配列内の最初のエッジを削除します。

```
var shape = fl.getDocumentDOM().selection[0];
shape.beginEdit();
shape.deleteEdge(0);
shape.endEdit();
```

shape.contours

使用できるバージョン

Flash MX 2004

シンタックス

shape.contours

説明

読み取り専用プロパティ。シェイプの Contour オブジェクトの配列です ([Contour オブジェクト](#)を参照)。

例

次の例では、contours 配列内の最初の輪郭を c 変数に格納し、輪郭の [HalfEdge オブジェクト](#) を he 変数に格納します。

```
var c = fl.getDocumentDOM().selection[0].contours[0];
var he = c.getHalfEdge();
```

shape.deleteEdge()

使用できるバージョン

Flash MX 2004

シンタックス

shape.deleteEdge(*index*)

パラメータ

index [shape.edges](#) 配列から削除するエッジを指定するゼロから始まるインデックス値。このメソッドは、shape.edges 配列の長さを変更します。

戻り値

なし

説明

メソッド。指定されたエッジを削除します。このメソッドを使用する前に、[shape.beginEdit\(\)](#) を呼び出す必要があります。

例

次の例では、現在選択されているシェイプを取得し、edge 配列内の最初のエッジを削除します。

```
var shape = fl.getDocumentDOM().selection[0];
shape.beginEdit();
shape.deleteEdge(0);
shape.endEdit();
```

shape.edges

使用できるバージョン

Flash MX 2004

シンタックス

`shape.edges`

説明

読み取り専用プロパティ。Edge オブジェクトの配列です ([Edge オブジェクト](#)を参照)。

shape.endEdit()

使用できるバージョン

Flash MX 2004

シンタックス

`shape.endEdit()`

パラメータ

なし

戻り値

なし

説明

メソッド。シェイプの編集セッションの終了を定義します。Shape オブジェクトまたはその従属部分に加えられたすべての変更がシェイプに適用されます。Shape オブジェクトまたはその従属部分を変更するコマンドを発行した後にこのメソッドを使用する必要があります。

例

次の例では、現在選択されているシェイプを取得し、`edge` 配列内の最初のエッジを削除します。

```
var shape = fl.getDocumentDOM().selection[0];
shape.beginEdit();
shape.deleteEdge(0);
shape.endEdit();
```

shape.isDrawingObject

使用できるバージョン

Flash 8

シンタックス

shape.isDrawingObject

説明

読み取り専用プロパティ。true の場合、シェイプは描画オブジェクトです。

例

次の例では、選択されている最初の item オブジェクトを sel 変数に格納し、[element.elementType](#) および [shape.isDrawingObject](#) プロパティを使用して、選択されているアイテムが描画オブジェクトかどうかを判断します。

```
var sel = fl.getDocumentDOM().selection[0];
var shapeDrawingObject = (sel.elementType == "shape") && sel.isDrawingObject;
fl.trace(shapeDrawingObject);
```

関連項目

[document.crop\(\)](#), [document.deleteEnvelope\(\)](#), [document.intersect\(\)](#),
[document.punch\(\)](#), [document.union\(\)](#), [shape.isGroup](#)

shape.isGroup

使用できるバージョン

Flash MX 2004

シンタックス

shape.isGroup

説明

読み取り専用プロパティ。true の場合、シェイプはグループです。

例

次の例では、選択されている最初の item オブジェクトを sel 変数に格納し、[element.elementType](#) および [shape.isGroup](#) プロパティを使用して、選択されているアイテムがグループかどうかを判断します。

```
var sel = fl.getDocumentDOM().selection[0];
var shapeGroup = (sel.elementType == "shape") && sel.isGroup;
fl.trace(shapeGroup);
```

関連項目

[shape.isDrawingObject](#)

shape.isOvalObject

使用できるバージョン

Flash CS3 Professional

シンタックス

`shape.isOvalObject`

説明

読み取り専用プロパティ。true の場合、シェイプはプリミティブ楕円オブジェクトです (楕円プリミティブツールを使用して作成)。

例

次の例では、最初に選択したアイテムがプリミティブ楕円オブジェクトの場合は "true"、そうでない場合は "false" を表示します。

```
var sel = fl.getDocumentDOM().selection[0];  
fl.trace(sel.isOvalObject);
```

関連項目

[shape.isRectangleObject](#)

shape.isRectangleObject

使用できるバージョン

Flash CS3 Professional

シンタックス

`shape.isRectangleObject`

説明

読み取り専用プロパティ。true の場合、シェイプはプリミティブ矩形オブジェクトです (矩形プリミティブツールを使用して作成)。

例

次の例では、最初に選択したアイテムがプリミティブ矩形オブジェクトの場合は "true"、そうでない場合は "false" を表示します。

```
var sel = fl.getDocumentDOM().selection[0];  
fl.trace(sel.isRectangleObject);
```

関連項目

[shape.isOvalObject](#)

shape.vertices

使用できるバージョン

Flash MX 2004

シンタックス

`shape.vertices`

説明

読み取り専用プロパティ。Vertex オブジェクトの配列です ([Vertex オブジェクト](#)を参照)。

例

次の例では、選択されている最初の item オブジェクトを someShape 変数に格納し、そのオブジェクトの頂点の数を [出力] パネルに表示します。

```
var someShape = fl.getDocumentDOM().selection[0];  
fl.trace("The shape has " + someShape.vertices.length + " vertices.");
```

SoundItem オブジェクト

継承 [Item オブジェクト](#) > SoundItem オブジェクト

使用できるバージョン

Flash MX 2004

説明

SoundItem オブジェクトは、Item オブジェクトのサブクラスです。このオブジェクトは、サウンドを作成するのに使用するライブラリアイテムを表します。[frame.soundLibraryItem](#) および [Item オブジェクト](#) も参照してください。

SoundItem オブジェクトのプロパティ一覧

SoundItem オブジェクトでは、[Item オブジェクト](#) のプロパティに加えて、次のプロパティを使用することができます。

プロパティ	説明
soundItem.bitRate	ライブラリ内のサウンドのビットレートを指定するストリング。MP3 圧縮タイプでのみ使用できます。
soundItem.bits	ライブラリ内の ADPCM 圧縮を使用するサウンドのビット値を指定するストリング。
soundItem.compressionType	ライブラリ内のサウンドの圧縮タイプを指定するストリング。
soundItem.convertStereoToMono	MP3 および Raw 圧縮タイプでのみ使用可能なブール値。
soundItem.quality	ライブラリ内のサウンドの再生品質を指定するストリング。MP3 圧縮タイプでのみ使用できます。
soundItem.sampleRate	オーディオクリップのサンプルレートを指定するストリング。
soundItem.useImportedMP3Quality	ブール値。true の場合、他のすべてのプロパティが無視され、読み込んだ MP3 の音質が使用されます。

soundItem.bitRate

使用できるバージョン

Flash MX 2004

シンタックス

soundItem.bitRate

説明

プロパティ。ライブラリ内のサウンドのビットレートを指定するストリングです。このプロパティは、MP3 圧縮タイプでのみ使用できます。有効な値は、"8 kbps"、"16 kbps"、"20 kbps"、"24 kbps"、"32 kbps"、"48 kbps"、"56 kbps"、"64 kbps"、"80 kbps"、"112 kbps"、"128 kbps"、および "160 kbps" です。8 kbps または 16 kbps で書き出されたステレオサウンドはモノラルに変換されます。他の圧縮タイプの場合、このプロパティは undefined になります。



このプロパティの値を指定する場合は、[soundItem.useImportedMP3Quality](#) を false に設定する必要があります。

例

次の例では、ライブラリ内の指定されたアイテムの圧縮タイプが MP3 の場合に [出力] パネルに bitRate 値を表示します。

```
alert(fl.getDocumentDOM().library.items[0].bitRate);
```

関連項目

[soundItem.compressionType](#), [soundItem.convertStereoToMono](#)

soundItem.bits

使用できるバージョン

Flash MX 2004

シンタックス

soundItem.bits

説明

プロパティ。ライブラリ内の ADPCM 圧縮を使用するサウンドのビット値を指定するストリングです。有効な値は、"2 bit"、"3 bit"、"4 bit"、および "5 bit" です。



このプロパティの値を指定する場合は、[soundItem.useImportedMP3Quality](#) を false に設定する必要があります。

例

次の例では、ライブラリ内の現在選択されているアイテムの圧縮タイプが ADPCM の場合に [出力] パネルにビット値を表示します。

```
alert(fl.getDocumentDOM().library.items[0].bits);
```

関連項目

[soundItem.compressionType](#)

soundItem.compressionType

使用できるバージョン

Flash MX 2004

シンタックス

```
soundItem.compressionType
```

説明

プロパティ。ライブラリ内のサウンドの圧縮タイプを指定するストリングです。有効な場合は、"Default"、"ADPCM"、"MP3"、"Raw"、および "Speech" です。



このプロパティの値を指定する場合は、[soundItem.useImportedMP3Quality](#) を false に設定する必要があります。

例

次の例では、ライブラリ内のアイテムの圧縮タイプを Raw に変更します。

```
fl.getDocumentDOM().library.items[0].compressionType = "Raw";
```

次の例では、選択されたアイテムの圧縮タイプを Speech (スピーチ) に変更します。

```
fl.getDocumentDOM().library.getSelectedItems()[0].compressionType = "Speech";
```

soundItem.convertStereoToMono

使用できるバージョン

Flash MX 2004

シンタックス

```
soundItem.convertStereoToMono
```

説明

プロパティ。MP3 および Raw 圧縮タイプでのみ使用可能なブール値です。このプロパティを true に設定すると、ステレオサウンドがモノラルに変換されます。false の場合はステレオのままになります。MP3 圧縮タイプの場合、soundItem.bitRate が 20 Kbps 未満のときには、このプロパティが無視され、強制的に true に設定されます。soundItem.bitRate を参照してください。



このプロパティの値を指定する場合は、soundItem.useImportedMP3Quality を false に設定する必要があります。

例

次の例では、アイテムの圧縮タイプが MP3 または Raw の場合のみ、ライブラリ内のアイテムをモノラルに変換します。

```
fl.getDocumentDOM().library.items[0].convertStereoToMono = true;
```

関連項目

[soundItem.compressionType](#)

soundItem.quality

使用できるバージョン

Flash MX 2004

シンタックス

soundItem.quality

説明

プロパティ。ライブラリ内のサウンドの再生品質を指定するストリングです。このプロパティは、MP3 圧縮タイプでのみ使用できます。有効な値は、"Fast"、"Medium"、および "Best" です。



このプロパティの値を指定する場合は、soundItem.useImportedMP3Quality を false に設定する必要があります。

例

次の例では、アイテムが MP3 圧縮タイプの場合、ライブラリ内のアイテムの再生品質を Best (品質優先) に設定します。

```
fl.getDocumentDOM().library.items[0].quality = "Best";
```

関連項目

[soundItem.compressionType](#)

soundItem.sampleRate

使用できるバージョン

Flash MX 2004

シンタックス

soundItem.sampleRate

説明

プロパティ。オーディオクリップのサンプルレートを指定するストリングです。このプロパティは、ADPCM、Raw、および Speech 圧縮タイプでのみ使用できます。有効な値は、"5 kHz"、"11 kHz"、"22 kHz"、および "44 kHz" です。



このプロパティの値を指定する場合は、[soundItem.useImportedMP3Quality](#) を false に設定する必要があります。

例

次の例では、ライブラリ内のアイテムの圧縮タイプが、ADPCM、Raw、または Speech の場合に、アイテムのサンプルレートを 5 kHz に設定します。

```
fl.getDocumentDOM().library.items[0].sampleRate = "5 kHz";
```

関連項目

[soundItem.compressionType](#)

soundItem.useImportedMP3Quality

使用できるバージョン

Flash MX 2004

シンタックス

soundItem.useImportedMP3Quality

説明

プロパティ。ブール値です。true の場合、他のすべてのプロパティが無視され、読み込んだ MP3 の音質が使用されます。

例

次の例では、読み込んだ MP3 の音質を使用するようにライブラリ内のアイテムを設定します。

```
fl.getDocumentDOM().library.items[0].useImportedMP3Quality = true;
```

関連項目

[soundItem.compressionType](#)

Stroke オブジェクト

使用できるバージョン

Flash MX 2004

説明

Stroke オブジェクトは、カスタム設定など、線に関するすべての設定を含みます。このオブジェクトは、プロパティインスペクタに含まれる情報を表します。Stroke オブジェクトを `document.setCustomStroke()` メソッドと共に使用して、ツールパネル、プロパティインスペクタ、および現在の選択内容の線の設定を変更することができます。また、`document.getCustomStroke()` メソッドを使用して、ツールパネル、プロパティインスペクタ、または現在の選択内容の線の設定を取得することもできます。

このオブジェクトには常に、`style`、`thickness`、`color`、および `breakAtCorners` の4つのプロパティがあります (Flash CS3 では、`breakAtCorners` プロパティの代わりに `stroke.joinType` を使用します)。他のプロパティは、`stroke.style` プロパティの値に応じて設定できます。

Stroke オブジェクトのプロパティ一覧

Stroke オブジェクトでは次のプロパティを使用できます。

プロパティ	説明
<code>stroke.breakAtCorners</code>	カスタムの [線のスタイル] ダイアログボックスの [角を離さない] 設定と同じ。
<code>stroke.capType</code>	線のキャップの種類を指定するストリング。
<code>stroke.color</code>	線のカラーを表す、ストリング、16 進数値、または整数。
<code>stroke.curve</code>	線のハッチの種類を指定するストリング。
<code>stroke.dash1</code>	破線の実線部分の長さを指定する整数。
<code>stroke.dash2</code>	破線の空白部分の長さを指定する整数。
<code>stroke.density</code>	点描の密度を指定するストリング。
<code>stroke.dotSize</code>	点描の点の大きさを指定するストリング。
<code>stroke.dotSpace</code>	点描の点の間隔を指定する整数。
<code>stroke.hatchThickness</code>	ハッチの線の太さを指定するストリング。
<code>stroke.jiggle</code>	ハッチの揺れのプロパティを指定するストリング。
<code>stroke.joinType</code>	線の結合の種類を指定するストリング。
<code>stroke.length</code>	ハッチの線の長さを指定するストリング。

プロパティ	説明
<code>stroke.miterLimit</code>	マイターの先端の角度の上限を指定する浮動小数点値。この角度より大きい部分はセグメントによって切り詰められます。
<code>stroke.pattern</code>	ラフのパターンを指定するストリング。
<code>stroke.rotate</code>	ハッチの回転を指定するストリング。
<code>stroke.scaleType</code>	線に適用する拡大 / 縮小の種類を指定するストリング。
<code>stroke.shapeFill</code>	線の塗りの設定を表す Fill オブジェクト 。
<code>stroke.space</code>	ハッチの線の間隔を指定するストリング。
<code>stroke.strokeHinting</code>	線のヒントが設定されているかどうかを指定するブール値。
<code>stroke.style</code>	線のスタイルを示すストリング。
<code>stroke.thickness</code>	線のサイズを指定する整数。
<code>stroke.variation</code>	点描のスタイルを指定するストリング。
<code>stroke.waveHeight</code>	ラフの波の高さを指定するストリング。
<code>stroke.waveLength</code>	ラフの波の長さを指定するストリング。

stroke.breakAtCorners

使用できるバージョン

Flash MX 2004。Flash CS3 では、このプロパティを使用しないでください。代わりに [stroke.joinType](#) を使用してください。

シンタックス

`stroke.breakAtCorners`

説明

プロパティ。ブール値です。このプロパティは、カスタムの [線のスタイル] ダイアログボックスの [角を離さない] 設定と同じです。

例

次の例では、`breakAtCorners` プロパティを `true` に設定します。

```
var myStroke = fl.getDocumentDOM().getCustomStroke();
myStroke.breakAtCorners = true;
fl.getDocumentDOM().setCustomStroke(myStroke);
```

stroke.capType

使用できるバージョン

Flash 8

シンタックス

stroke.capType

説明

プロパティ。線のキャップの種類を指定するストリングです。有効な値は、"none"、"round"、および "square" です。

例

次の例では、線のキャップの種類を "round" に設定します。

```
var myStroke = fl.getDocumentDOM().getCustomStroke();
myStroke.capType = "round";
fl.getDocumentDOM().setCustomStroke(myStroke);
```

stroke.color

使用できるバージョン

Flash MX 2004。Flash 8 以降では、このプロパティの代わりに stroke.shapeFill.color を使用します。

シンタックス

stroke.color

説明

プロパティ。線のカラー。次のいずれかの形式です。

- "#RRGGBB" または "#RRGGBBAA" 形式のストリング
- 0xRRGGBB 形式の 16 進数
- 16 進数と等しい 10 進表現の整数

例

次の例では、線のカラーを設定します。

```
var myStroke = fl.getDocumentDOM().getCustomStroke();
myStroke.color = "#000000";
fl.getDocumentDOM().setCustomStroke(myStroke);
```

関連項目

[stroke.shapeFill](#)

stroke.curve

使用できるバージョン

Flash MX 2004

シンタックス

stroke.curve

説明

プロパティ。線のハッチの種類を指定するストリングです。このプロパティは、stroke.style プロパティが "hatched" に設定された場合のみ設定できます。[stroke.style](#) を参照してください。有効な値は、"straight"、"slight curve"、"medium curve"、および "very curved" です。

例

次の例では、"hatched" スタイルの線の curve および他のプロパティを設定します。

```
var myStroke = fl.getDocumentDOM().getCustomStroke();
myStroke.style = "hatched";
myStroke.curve = "straight";
myStroke.space = "close";
myStroke.jiggle = "wild";
myStroke.rotate = "free";
myStroke.length = "slight";
myStroke.hatchThickness = "thin";
fl.getDocumentDOM().setCustomStroke(myStroke);
```

stroke.dash1

使用できるバージョン

Flash MX 2004

シンタックス

stroke.dash1

説明

プロパティ。破線の実線部分の長さを指定する整数です。このプロパティは、stroke.style プロパティが "dashed" に設定されている場合のみ使用できます。[stroke.style](#) を参照してください。

例

次の例では、dashed スタイルの線の dash1 および dash2 プロパティを設定します。

```
var myStroke = fl.getDocumentDOM().getCustomStroke();
myStroke.style = "dashed";
myStroke.dash1 = 1;
myStroke.dash2 = 2;
fl.getDocumentDOM().setCustomStroke(myStroke);
```

stroke.dash2

使用できるバージョン

Flash MX 2004

シンタックス

`stroke.dash2`

説明

プロパティ。破線の空白部分の長さを指定する整数です。このプロパティは、`stroke.style` プロパティが "dashed" に設定されている場合のみ使用できます。[stroke.style](#) を参照してください。

例

詳細については、[stroke.dash1](#) を参照してください。

stroke.density

使用できるバージョン

Flash MX 2004

シンタックス

`stroke.density`

説明

プロパティ。点描の密度を指定するストリングです。このプロパティは、`stroke.style` プロパティが "stipple" に設定されている場合のみ使用できます。[stroke.style](#) を参照してください。有効な値は、"very dense"、"dense"、"sparse"、および "very sparse" です。

例

次の例では、stipple スタイルの線の `density` プロパティを "sparse" に設定します。

```
var myStroke = fl.getDocumentDOM().getCustomStroke();
myStroke.style = "stipple";
myStroke.dotSpace = 3;
myStroke.variation = "random sizes";
myStroke.density = "sparse";
fl.getDocumentDOM().setCustomStroke(myStroke);
```


stroke.dotSize

使用できるバージョン

Flash MX 2004

シンタックス

stroke.dotSize

説明

プロパティ。点描の点の大きさを指定するストリングです。このプロパティは、stroke.style プロパティが "stipple" に設定されている場合のみ使用できます。[stroke.style](#) を参照してください。有効な値は、"tiny"、"small"、"medium"、および "large" です。

次の例では、stipple スタイルの線の dotSize プロパティを "tiny" に設定します。

```
var myStroke = fl.getDocumentDOM().getCustomStroke();
myStroke.style = "stipple";
myStroke.dotSpace= 3;
myStroke.dotsize = "tiny";
myStroke.variation = "random sizes";
myStroke.density = "sparse";
fl.getDocumentDOM().setCustomStroke(myStroke);
```

stroke.dotSpace

使用できるバージョン

Flash MX 2004

シンタックス

stroke.dotSpace

説明

プロパティ。点描の点の間隔を指定する整数です。このプロパティは、stroke.style プロパティが "dotted" に設定されている場合のみ使用できます。詳細については、[stroke.style](#) を参照してください。

例

次の例では、dotted スタイルの線の dotSpace プロパティを 3 に設定します。

```
var myStroke = fl.getDocumentDOM().getCustomStroke();
myStroke.style = "dotted";
myStroke.dotSpace= 3;
fl.getDocumentDOM().setCustomStroke(myStroke);
```

stroke.hatchThickness

使用できるバージョン

Flash MX 2004

シンタックス

stroke.hatchThickness

説明

プロパティ。ハッチの線の太さを指定するストリングです。このプロパティは、stroke.style プロパティが "hatched" に設定されている場合のみ使用できます。[stroke.style](#) を参照してください。有効な値は、"hairline"、"thin"、"medium"、および "thick" です。

例

次の例では、hatched スタイルの線の hatchThickness プロパティを "thin" に設定します。

```
var myStroke = fl.getDocumentDOM().getCustomStroke();
myStroke.style = "hatched";
myStroke.curve = "straight";
myStroke.space = "close";
myStroke.jiggle = "wild";
myStroke.rotate = "free";
myStroke.length = "slight";
myStroke.hatchThickness = "thin";
fl.getDocumentDOM().setCustomStroke(myStroke);
```

stroke.jiggle

使用できるバージョン

Flash MX 2004

シンタックス

stroke.jiggle

説明

プロパティ。ハッチの揺れのプロパティを指定するストリングです。このプロパティは、stroke.style プロパティが "hatched" に設定されている場合のみ使用できます。[stroke.style](#) を参照してください。有効な値は、"none"、"bounce"、"loose"、および "wild" です。

例

次の例では、hatched スタイルの線の jiggle プロパティを "wild" に設定します。

```
var myStroke = fl.getDocumentDOM().getCustomStroke();
myStroke.style = "hatched";
myStroke.curve = "straight";
myStroke.space = "close";
myStroke.jiggle = "wild";
myStroke.rotate = "free";
myStroke.length = "slight";
myStroke.hatchThickness = "thin";
fl.getDocumentDOM().setCustomStroke(myStroke);
```

stroke.joinType

使用できるバージョン

Flash 8

シンタックス

stroke.joinType

説明

プロパティ。線の結合の種類を指定するストリングです。有効な値は、"miter"、"round"、および "bevel" です。

関連項目

[stroke.capType](#)

stroke.length

使用できるバージョン

Flash MX 2004

シンタックス

stroke.length

説明

プロパティ。ハッチの線の長さを指定するストリングです。このプロパティは、stroke.style プロパティが "hatched" に設定されている場合のみ使用できます。[stroke.style](#) を参照してください。有効な値は、"equal"、"slight"、"variation"、"medium variation"、および "random" です。

例

次の例では、hatched スタイルの線の length プロパティを "slight" に設定します。

```
var myStroke = fl.getDocumentDOM().getCustomStroke();
myStroke.style = "hatched";
myStroke.curve = "straight";
myStroke.space = "close";
myStroke.jiggle = "wild";
myStroke.rotate = "free";
myStroke.length = "slight";
myStroke.hatchThickness = "thin";
fl.getDocumentDOM().setCustomStroke(myStroke);
```

stroke.miterLimit

使用できるバージョン

Flash 8

シンタックス

stroke.miterLimit

説明

プロパティ。マイターの先端の角度の上限を指定する浮動小数点値です。この角度より大きい部分はセグメントによって切り詰められます。つまり、マイターの角度が miterLimit の値より大きい場合のみ、マイターが切り詰められます。

例

次の例では、線のマイターの限度の設定を 3 に変更します。マイターの角度が 3 より大きい場合、マイターが切り詰められます。

```
var myStroke = fl.getDocumentDOM().getCustomStroke();
myStroke.miterLimit = 3;
var myStroke = fl.getDocumentDOM().setCustomStroke();
```

stroke.pattern

使用できるバージョン

Flash MX 2004

シンタックス

stroke.pattern

説明

プロパティ。ラフのパターンを指定するストリングです。このプロパティは、`stroke.style` プロパティが "ragged" に設定されている場合のみ使用できます。[stroke.style](#) を参照してください。有効な値は、"solid"、"simple"、"random"、"dotted"、"random dotted"、"triple dotted"、および "random triple dotted" です。

例

次の例では、ragged スタイルの線の pattern プロパティを "random" に設定します。

```
var myStroke = fl.getDocumentDOM().getCustomStroke();
myStroke.style = "ragged";
myStroke.pattern = "random";
fl.getDocumentDOM().setCustomStroke(myStroke);
```

stroke.rotate

使用できるバージョン

Flash MX 2004

シンタックス

`stroke.rotate`

説明

プロパティ。ハッチの回転を指定するストリングです。このプロパティは、`stroke.style` プロパティが "hatched" に設定されている場合のみ使用できます。[stroke.style](#) を参照してください。有効な値は、"none"、"slight"、"medium"、および "free" です。

例

次の例では、hatched スタイルの線の rotate プロパティを "free" に設定します。

```
var myStroke = fl.getDocumentDOM().getCustomStroke();
myStroke.style = "hatched";
myStroke.curve = "straight";
myStroke.space = "close";
myStroke.jiggle = "wild";
myStroke.rotate = "free";
myStroke.length = "slight";
myStroke.hatchThickness = "thin";
```

stroke.scaleType

使用できるバージョン

Flash 8

シンタックス

stroke.scaleType

説明

プロパティ。線に適用する拡大 / 縮小の種類を指定するストリングです。有効な値は、"normal"、"horizontal"、"vertical"、および "none" です。

例

次の例では、線の拡大 / 縮小の種類を "horizontal" に設定します。

```
var myStroke = fl.getDocumentDOM().getCustomStroke();
myStroke.scaleType = "horizontal";
fl.getDocumentDOM().setCustomStroke(myStroke);
```

stroke.shapeFill

使用できるバージョン

Flash 8

シンタックス

stroke.shapeFill

説明

プロパティ。線の塗りの設定を表す [Fill オブジェクト](#) です。

例

次の例では、塗りの設定を指定し、それらを線に適用します。

```
var fill = fl.getDocumentDOM().getCustomFill();
fill.linearGradient = true;
fill.colorArray = [ 00ff00, ff0000, fffff ];
var stroke = fl.getDocumentDOM().getCustomStroke();
stroke.shapeFill = fill;
fl.getDocumentDOM().setCustomStroke(stroke);
```

stroke.space

使用できるバージョン

Flash MX 2004

シンタックス

stroke.space

説明

プロパティ。ハッチの線の間隔を指定するストリングです。このプロパティは、stroke.style プロパティが "hatched" に設定されている場合のみ使用できます。[stroke.style](#) を参照してください。有効な値は、"very close"、"close"、"distant"、および "very distant" です。

例

次の例では、hatched スタイルの線の space プロパティを "close" に設定します。

```
var myStroke = fl.getDocumentDOM().getCustomStroke();
myStroke.style = "hatched";
myStroke.curve = "straight";
myStroke.space = "close";
myStroke.jiggle = "wild";
myStroke.rotate = "free";
myStroke.length = "slight";
myStroke.hatchThickness = "thin";
fl.getDocumentDOM().setCustomStroke(myStroke);
```

stroke.strokeHinting

使用できるバージョン

Flash 8

シンタックス

stroke.strokeHinting

説明

プロパティ。線のヒントが設定されているかどうかを指定するブール値です。

例

次の例では、線のヒントを有効にします。

```
var myStroke = fl.getDocumentDOM().getCustomStroke();
myStroke.strokeHinting = true;
fl.getDocumentDOM().setCustomStroke(myStroke);
```

stroke.style

使用できるバージョン

Flash MX 2004

シンタックス

stroke.style

説明

プロパティ。線のスタイルを示すストリングです。有効な値は、"noStroke"、"solid"、"dashed"、"dotted"、"ragged"、"stipple"、および "hatched" です。これらの値の一部では、次の一覧に示すように、**Stroke** オブジェクトの追加のプロパティを設定する必要があります。

- 値が "solid" または "noStroke" の場合、他のプロパティはありません。
- 値が "dashed" の場合、"dash1" および "dash2" という 2 つの追加のプロパティがあります。
- 値が "dotted" の場合、"dotSpace" という 1 つの追加のプロパティがあります。
- 値が "ragged" の場合、"pattern"、"waveHeight"、および "waveLength" という 3 つの追加のプロパティがあります。
- 値が "stipple" の場合、"dotSize"、"variation"、および "density" という 3 つの追加のプロパティがあります。
- 値が "hatched" の場合、"hatchThickness"、"space"、"jiggle"、"rotate"、"curve"、および "length" という 6 つの追加のプロパティがあります。

例

次の例では、線のスタイルを "ragged" に設定します。

```
var myStroke = fl.getDocumentDOM().getCustomStroke();  
myStroke.style = "ragged";  
fl.getDocumentDOM().setCustomStroke(myStroke);
```

stroke.thickness

使用できるバージョン

Flash MX 2004

シンタックス

stroke.thickness

説明

プロパティ。線のサイズを指定する整数です。

例

次の例では、線の `thickness` プロパティの値を 2 に設定します。

```
var myStroke = fl.getDocumentDOM().getCustomStroke();
myStroke.thickness = 2;
fl.getDocumentDOM().setCustomStroke(myStroke);
```

stroke.variation

使用できるバージョン

Flash MX 2004

シンタックス

`stroke.variation`

説明

プロパティ。点描のスタイルを指定するストリングです。このプロパティは、`stroke.style` プロパティが `"stipple"` に設定されている場合のみ使用できます。[stroke.style](#) を参照してください。有効な値は、`"one size"`、`"small variation"`、`"varied sizes"`、および `"random sizes"` です。

例

次の例では、`stipple` スタイルの線の `variation` プロパティを `"random sizes"` に設定します。

```
var myStroke = fl.getDocumentDOM().getCustomStroke();
myStroke.style = "stipple";
myStroke.dotSpace = 3;
myStroke.variation = "random sizes";
myStroke.density = "sparse";
fl.getDocumentDOM().setCustomStroke(myStroke);
```

stroke.waveHeight

使用できるバージョン

Flash MX 2004

シンタックス

`stroke.waveHeight`

説明

プロパティ。ラフの波の高さを指定するストリングです。このプロパティは、`stroke.style` プロパティが `"ragged"` に設定されている場合のみ使用できます。[stroke.style](#) を参照してください。有効な値は、`"flat"`、`"wavy"`、`"very wavy"`、および `"wild"` です。

例

次の例では、ragged スタイルの線の waveHeight プロパティを "flat" に設定します。

```
var myStroke = fl.getDocumentDOM().getCustomStroke();
myStroke.style = "ragged";
myStroke.pattern = "random";
myStroke.waveHeight = "flat";
myStroke.waveLength = "short";
fl.getDocumentDOM().setCustomStroke(myStroke);
```

stroke.waveLength

使用できるバージョン

Flash MX 2004

シンタックス

stroke.waveLength

説明

プロパティ。ラフの波の長さを指定するストリングです。このプロパティは、stroke.style プロパティが "ragged" に設定されている場合のみ使用できます。[stroke.style](#) を参照してください。有効な値は、"very short"、"short"、"medium"、および "long" です。

例

次の例では、ragged スタイルの線の waveLength プロパティを "short" に設定します。

```
var myStroke = fl.getDocumentDOM().getCustomStroke();
myStroke.style = "ragged";
myStroke.pattern = "random";
myStroke.waveHeight = 'flat';
myStroke.waveLength = "short";
fl.getDocumentDOM().setCustomStroke(myStroke);
```

SymbolInstance オブジェクト

継承 [Element オブジェクト](#) > [Instance オブジェクト](#) > SymbolInstance オブジェクト

使用できるバージョン

Flash MX 2004

説明

SymbolInstance は、Instance オブジェクトのサブクラスで、フレーム内のシンボルを表します。
[Instance オブジェクト](#) を参照してください。

SymbolInstance オブジェクトのプロパティ一覧

SymbolInstance オブジェクトには、[Instance オブジェクト](#) のプロパティに加えて次のプロパティがあります。

プロパティ	説明
symbolInstance.accName	[アクセシビリティ] パネルの [名前] フィールドと等価のストリング。
symbolInstance.actionScript	シンボルに割り当てられたアクションを指定するストリング。
symbolInstance.blendMode	ムービークリップシンボルに適用するブレンドモードを指定するストリング。
symbolInstance.buttonTracking	ボタンシンボルの場合にのみ、プロパティインスペクタの [ボタントラック] または [メニューアイテムトラック] のポップアップメニューと同じプロパティを設定するストリング。
symbolInstance.cacheAsBitmap	実行時のビットマップキャッシングが有効かどうかを指定するブール値。
symbolInstance.colorAlphaAmount	[拡張効果] の [アルファ] 設定を指定することで、インスタンスのカラー変換の一部を指定する整数。これは、プロパティインスペクタで [カラー]-[詳細] を選択し、[設定] をクリックして表示するダイアログボックスの右側のコントロールを調整するのと等価です。
symbolInstance.colorAlphaPercent	インスタンスのカラー変換の一部を指定する整数。これは、インスタンスのプロパティインスペクタで [カラー]-[詳細] を選択し、[設定] をクリックして表示するダイアログボックスの左側のパーセンテージコントロールを調整するのと等価です。

プロパティ	説明
<code>symbolInstance.colorBlueAmount</code>	インスタンスのカラー変換の一部を指定する整数。これは、インスタンスのプロパティインスペクタで [カラー]-[詳細] を選択し、[設定] をクリックして表示するダイアログボックスを使用するのと等価です。
<code>symbolInstance.colorBluePercent</code>	インスタンスのカラー変換の一部を指定する整数。これは、インスタンスのプロパティインスペクタで [カラー]-[詳細] を選択し、[設定] をクリックして表示するダイアログボックスの左側のパーセンテージコントロールを使用するのと等価です。
<code>symbolInstance.colorGreenAmount</code>	インスタンスのカラー変換の一部を指定する整数。これは、インスタンスのプロパティインスペクタで [カラー]-[詳細] を選択し、[設定] をクリックして表示するダイアログボックスを使用するのと等価です。有効な値は、-255～255 です。
<code>symbolInstance.colorGreenPercent</code>	インスタンスのカラー変換の一部。これは、インスタンスのプロパティインスペクタで [カラー]-[詳細] を選択し、[設定] をクリックして表示するダイアログボックスの左側のパーセンテージコントロールを使用するのと等価です。
<code>symbolInstance.colorMode</code>	シンボルのプロパティインスペクタの [カラー] ポップアップメニューで示されるカラーモードを指定するストリング。
<code>symbolInstance.colorRedAmount</code>	インスタンスのカラー変換の一部を指定する整数。これは、インスタンスのプロパティインスペクタで [カラー]-[詳細] を選択し、[設定] をクリックして表示するダイアログボックスを使用するのと等価です。
<code>symbolInstance.colorRedPercent</code>	インスタンスのカラー変換の一部。これは、インスタンスのプロパティインスペクタで [カラー]-[詳細] を選択し、[設定] をクリックして表示するダイアログボックスの左側のパーセンテージコントロールを使用するのと等価です。
<code>symbolInstance.description</code>	[アクセシビリティ] パネルの [説明] フィールドと等価のストリング。
<code>symbolInstance.filters</code>	Filter オブジェクトの配列。 Filter オブジェクト を参照してください。
<code>symbolInstance.firstFrame</code>	グラフィックのタイムラインに表示される最初のフレームを指定するゼロから始まる整数。
<code>symbolInstance.forceSimple</code>	オブジェクトの子のアクセシビリティを有効または無効にするブール値。これは、[アクセシビリティ] パネルの [子オブジェクトをアクセス可能にする] の設定の逆ロジックと等価です。

プロパティ	説明
<code>symbolInstance.loop</code>	グラフィックシンボルの場合に、プロパティインスペクタの [ループ] ポップアップメニューと同じプロパティを設定するストリング。
<code>symbolInstance.shortcut</code>	シンボルに割り当てられているショートカットキーと等価のストリング。これは、[アクセシビリティ] パネルの [ショートカット] フィールドと等価です。
<code>symbolInstance.silent</code>	オブジェクトのアクセシビリティを有効または無効にするブール値。これは、[アクセシビリティ] パネルの [オブジェクトをアクセス可能にする] の設定の逆ロジックと等価です。
<code>symbolInstance.symbolType</code>	シンボルの種類を指定するストリング。これは、[新規シンボルの作成] および [シンボルに変換] ダイアログボックスの [ビヘイビア] の値と等価です。
<code>symbolInstance.tabIndex</code>	[アクセシビリティ] パネルの [タブインデックス] フィールドと等価の整数。

symbolInstance.accName

使用できるバージョン

Flash MX 2004

シンタックス

`symbolInstance.accName`

説明

プロパティ。[アクセシビリティ] パネルの [名前] フィールドと等価のストリングです。スクリーンリーダーは、名前を読み上げてオブジェクトを識別します。このプロパティは、グラフィックシンボルでは使用できません。

例

次の例では、オブジェクトの [アクセシビリティ] パネル名の値を `theName` 変数に格納します。

```
var theName = fl.getDocumentDOM().selection[0].accName;
```

次の例では、オブジェクトの [アクセシビリティ] パネル名の値を "Home Button" に設定します。

```
fl.getDocumentDOM().selection[0].accName = "Home Button";
```

symbolInstance.actionScript

使用できるバージョン

Flash MX 2004

シンタックス

symbolInstance.actionScript

説明

プロパティ。シンボルに割り当てられたアクションを指定する文字列です。これは、ムービークリップとボタンのインスタンスにのみ適用されます。グラフィックシンボルインスタンスの場合は、`undefined` という値が返されます。

例

次の例では、タイムラインの最初のレイヤーの最初のフレームの最初のアイテムに `onClipEvent` アクションを割り当てます。

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].actionScript  
    = "onClipEvent(enterFrame) {trace('movie clip enterFrame')}";
```

symbolInstance.blendMode

使用できるバージョン

Flash 8

シンタックス

symbolInstance.blendMode

説明

プロパティ。ムービークリップシンボルに適用するブレンドモードを指定する文字列です。有効な値は、`"normal"`、`"layer"`、`"multiply"`、`"screen"`、`"overlay"`、`"hardlight"`、`"lighten"`、`"darken"`、`"difference"`、`"add"`、`"subtract"`、`"invert"`、`"alpha"`、および `"erase"` です。

例

次の例では、最初のレベルの最初のフレームの最初のムービークリップシンボルのブレンドモードを `"add"` に設定します。

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].blendMode =  
    "add";
```

関連項目

[document.setBlendMode\(\)](#)

symbolInstance.buttonTracking

使用できるバージョン

Flash MX 2004

シンタックス

symbolInstance.buttonTracking

説明

プロパティ。ボタンシンボルの場合にのみ、プロパティインスペクタの [ボタントラック] または [メニューアイテムトラック] のポップアップメニューと同じプロパティを設定するストリングです。他の種類のシンボルの場合、このプロパティは無視されます。有効な値は、"button" または "menu" です。

例

次の例では、シンボルがボタンの場合にのみ、タイムラインの最初のレイヤーの最初のフレームの最初のシンボルを [メニューアイテムトラック] に割り当てます。

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].buttonTracking  
    = "menu";
```

symbolInstance.cacheAsBitmap

使用できるバージョン

Flash 8

シンタックス

symbolInstance.cacheAsBitmap

説明

プロパティ。実行時のビットマップキャッシングが有効かどうかを指定するブール値です。

例

次の例では、最初のレイヤーの最初のフレームの最初のエレメントに対して実行時のビットマップキャッシングを有効にします。

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].cacheAsBitmap  
    = true;
```

symbolInstance.colorAlphaAmount

使用できるバージョン

Flash MX 2004

シンタックス

symbolInstance.colorAlphaAmount

説明

プロパティ。インスタンスのカラー変換の一部を指定する整数です。[拡張効果] の [アルファ] 設定を指定します。このプロパティは、プロパティインスペクタの [カラー]-[詳細] を選択し、[設定] をクリックして表示するダイアログボックスの右側のコントロールを調整するのと等価です。この値は、回転値とアルファ値を一定量ずつ増減させます。この値は現在の値に加算されます。このプロパティは、[symbolInstance.colorAlphaPercent](#) と共に使用する場合に最も役に立ちます。有効な値は、-255 ～ 255 です。

例

次の例では、選択されたシンボルインスタンスのアルファの設定から 100 を減算します。

```
fl.getDocumentDOM().selection[0].colorAlphaAmount = -100;
```

symbolInstance.colorAlphaPercent

使用できるバージョン

Flash MX 2004

シンタックス

symbolInstance.colorAlphaPercent

説明

プロパティ。インスタンスのカラー変換の一部を指定する整数です。このプロパティは、インスタンスのプロパティインスペクタで [カラー]-[詳細] を選択し、[設定] をクリックして表示されるダイアログボックスの左側のコントロールを調整するのと等価です。この値は、回転値とアルファ値を、指定されたパーセンテージに変更します。有効な値は、-100 ～ 100 です。

[symbolInstance.colorAlphaAmount](#) も参照してください。

例

次の例では、選択されたシンボルインスタンスの colorAlphaPercent を 80 に設定します。

```
fl.getDocumentDOM().selection[0].colorAlphaPercent = 80;
```


symbolInstance.colorBlueAmount

使用できるバージョン

Flash MX 2004

シンタックス

`symbolInstance.colorBlueAmount`

説明

プロパティ。インスタンスのカラー変換の一部を指定する整数です。このプロパティは、インスタンスのプロパティインスペクタの[カラー]-[詳細]を使用するのと等価です。有効な値は、255 ~ 255 です。

symbolInstance.colorBluePercent

使用できるバージョン

Flash MX 2004

シンタックス

`symbolInstance.colorBluePercent`

説明

プロパティ。インスタンスのカラー変換の一部を指定する整数です。このプロパティは、インスタンスのプロパティインスペクタで[カラー]-[詳細]を選択し、[設定]をクリックして表示されるダイアログボックスの左側のコントロールを調整するのと等価です。この値は、青の値を指定されたパーセンテージに設定します。有効な値は、100 ~ 100 です。

例

次の例では、選択されたシンボルインスタンスの `colorBluePercent` を 80 に設定します。

```
fl.getDocumentDOM().selection[0].colorBluePercent = 80;
```

symbolInstance.colorGreenAmount

使用できるバージョン

Flash MX 2004

シンタックス

`symbolInstance.colorGreenAmount`

説明

プロパティ。インスタンスのカラー変換の一部を指定する整数です。このプロパティは、インスタンスのプロパティインスペクタの [カラー]-[詳細] を使用するのと等価です。有効な値は、255 ~ 255 です。

symbolInstance.colorGreenPercent

使用できるバージョン

Flash MX 2004

シンタックス

`symbolInstance.colorGreenPercent`

説明

プロパティ。インスタンスのカラー変換の一部を指定する整数です。このプロパティは、インスタンスのプロパティインスペクタで [カラー]-[詳細] を選択し、[設定] をクリックして表示されるダイアログボックスの左側のコントロールを調整するのと等価です。この値は、緑の値を指定されたパーセンテージに設定します。有効な値は、0 ~ 100 です。

例

次の例では、選択されたシンボルインスタンスの `colorGreenPercent` を 70 に設定します。

```
fl.getDocumentDOM().selection[0].colorGreenPercent = 70;
```

symbolInstance.colorMode

使用できるバージョン

Flash MX 2004

シンタックス

`symbolInstance.colorMode`

説明

プロパティ。シンボルのプロパティインスペクタの [カラー] ポップアップメニューで示されるカラーモードを指定するストリングです。有効な値は、"none"、"brightness"、"tint"、"alpha"、および "advanced" です。

例

次の例では、タイムラインの最初のレイヤーの最初のフレームの最初のエレメントの `colorMode` プロパティを "alpha" に変更します。

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].colorMode =  
    "alpha";
```

symbolInstance.colorRedAmount

使用できるバージョン

Flash MX 2004

シンタックス

`symbolInstance.colorRedAmount`

説明

プロパティ。インスタンスのカラー変換の一部を指定する整数です。このプロパティは、インスタンスのプロパティインスペクタの [カラー]-[詳細] を使用するのと等価です。有効な値は、255 ~ 255 です。

例

次の例では、選択されたシンボルインスタンスの `colorRedAmount` を 255 に設定します。

```
fl.getDocumentDOM().selection[0].colorRedAmount = 255;
```

symbolInstance.colorRedPercent

使用できるバージョン

Flash MX 2004

シンタックス

`symbolInstance.colorRedPercent`

説明

プロパティ。インスタンスのカラー変換の一部を指定する整数です。このプロパティは、インスタンスのプロパティインスペクタで [カラー]-[詳細] を選択し、[設定] をクリックして表示されるダイアログボックスの左側のコントロールを調整するのと等価です。この値は、赤の値を指定されたパーセンテージに設定します。有効な値は、100 ~ 100 です。

例

次の例では、選択されたシンボルインスタンスの `colorRedPercent` を 10 に設定します。

```
fl.getDocumentDOM().selection[0].colorRedPercent = 10;
```

symbolInstance.description

使用できるバージョン

Flash MX 2004

シンタックス

symbolInstance.description

説明

プロパティ。[アクセシビリティ] パネルの [説明] フィールドと等価のストリングです。この説明がスクリーンリーダーによって読み上げられます。このプロパティは、グラフィックシンボルでは使用できません。

例

次の例では、オブジェクトの [アクセシビリティ] パネルの説明の値を theDescription 変数に格納します。

```
var theDescription = fl.getDocumentDOM().selection[0].description;
```

次の例では、[アクセシビリティ] パネルの説明の値を "Click the home button to go to home" に設定します。

```
fl.getDocumentDOM().selection[0].description= "Click the home button to go to  
home";
```

symbolInstance.filters

使用できるバージョン

Flash 8

シンタックス

symbolInstance.filters

説明

プロパティ。Filter オブジェクトの配列です。[Filter オブジェクト](#)を参照してください。フィルタプロパティを変更する場合は、この配列に直接書き込むのではなく、配列を取得して個々のプロパティを設定した後、その配列を設定して新しいプロパティを反映します。

例

次の例では、インデックス 0 のフィルタの名前をトレースします。フィルタがグローフィルタの場合、blurX プロパティを 100 に設定し、フィルタの配列にこの新しい値を書き込みます。

```
var filterName =  
    fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].filters[0].  
    name;  
fl.trace(filterName);  
var filterArray =  
    fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].filters;  
if (filterName == 'glowFilter'){  
    filterArray[0].blurX = 100;  
}  
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].filters =  
    filterArray;
```

symbolInstance.firstFrame

使用できるバージョン

Flash MX 2004

シンタックス

symbolInstance.firstFrame

説明

プロパティ。グラフィックのタイムラインに表示される最初のフレームを指定するゼロから始まる整数です。このプロパティは、グラフィックシンボルにのみ適用され、プロパティインスペクタの[名前] フィールドと同じプロパティを設定します。他の種類のシンボルの場合、このプロパティは undefined になります。

例

次の例では、Frame 10 が、指定されたエレメントのタイムラインに表示される最初のフレームになることを指定します。

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].firstFrame =  
    10;
```

symbolInstance.forceSimple

使用できるバージョン

Flash MX 2004

シンタックス

symbolInstance.forceSimple

説明

プロパティ。オブジェクトの子のアクセシビリティを有効または無効にするブール値です。このプロパティは、[アクセシビリティ] パネルの [子オブジェクトをアクセス可能にする] の設定の逆ロジックと等価です。たとえば、forceSimple が true の場合、[子オブジェクトをアクセス可能にする] がオフになっていることと同じです。forceSimple が false の場合、[子オブジェクトをアクセス可能にする] をオンにした場合と同じになります。

このプロパティは、MovieClip オブジェクトでのみ使用できます。

例

次の例では、オブジェクトの子がアクセス可能かどうかを確認します。戻り値が false の場合、子がアクセス可能なことを意味します。

```
var areChildrenAccessible = fl.getDocumentDOM().selection[0].forceSimple;
```

次の例では、オブジェクトの子をアクセス可能にします。

```
fl.getDocumentDOM().selection[0].forceSimple = false;
```

symbolInstance.loop

使用できるバージョン

Flash MX 2004

シンタックス

symbolInstance.loop

説明

プロパティ。グラフィックシンボルの場合に、プロパティインスペクタの [ループ] ポップアップメニューと同じプロパティを設定するストリングです。他の種類のシンボルの場合、このプロパティは undefined になります。有効な値は、"loop"、"play once"、および "single frame" です。各値に応じて、グラフィックのアニメーションが設定されます。

例

次の例では、シンボルがグラフィックの場合に限り、タイムラインの最初のレイヤーの最初のフレームの最初のシンボルをシングルフレームに設定します。グラフィックのタイムラインの 1 つの指定されたフレームが表示されます。

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].loop = 'single frame';
```

symbolInstance.shortcut

使用できるバージョン

Flash MX 2004

シンタックス

`symbolInstance.shortcut`

説明

プロパティ。シンボルに割り当てられたショートカットキーと等価のストリングです。このプロパティは、[アクセシビリティ] パネルの [ショートカット] フィールドと等価です。このキーがスクリーンリーダーによって読み上げられます。このプロパティは、グラフィックシンボルでは使用できません。

例

次の例では、オブジェクトのショートカットキーの値を `theShortcut` 変数に格納します。

```
var theShortcut = fl.getDocumentDOM().selection[0].shortcut;
```

次の例では、オブジェクトのショートカットキーを "Ctrl+i" に設定します。

```
fl.getDocumentDOM().selection[0].shortcut = "Ctrl+i";
```

symbolInstance.silent

使用できるバージョン

Flash MX 2004

シンタックス

`symbolInstance.silent`

説明

プロパティ。オブジェクトのアクセシビリティを有効または無効にするブール値です。このプロパティは、[アクセシビリティ] パネルの [オブジェクトをアクセス可能にする] の設定の逆ロジックと等価です。たとえば、`silent` が `true` の場合、[オブジェクトをアクセス可能にする] がオフになっていることと同じです。`silent` が `false` の場合、[子オブジェクトをアクセス可能にする] がオンになっていることと同じです。

このプロパティは、グラフィックオブジェクトでは使用できません。

例

次の例では、オブジェクトがアクセス可能かどうかを確認します。戻り値が `false` の場合、オブジェクトがアクセス可能なことを意味します。

```
var isSilent = fl.getDocumentDOM().selection[0].silent;
```

次の例では、オブジェクトをアクセス可能に設定します。

```
fl.getDocumentDOM().selection[0].silent = false;
```

symbolInstance.symbolType

使用できるバージョン

Flash MX 2004

シンタックス

symbolInstance.symbolType

説明

プロパティ。シンボルの種類を指定するストリングです。このプロパティは、[新規シンボルの作成] および [シンボルに変換] ダイアログボックスの [ビヘイビア] の値と等価です。有効な値は、"button"、"movie clip"、および "graphic" です。

例

次の例では、現在のドキュメントのタイムラインの最初のレイヤーの最初のフレームの最初のシンボルをグラフィックシンボルとして動作するように設定します。

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].symbolType =  
    "graphic";
```

symbolInstance.tabIndex

使用できるバージョン

Flash MX 2004

シンタックス

symbolInstance.tabIndex

説明

プロパティ。[アクセシビリティ] パネルの [タブインデックス] フィールドと等価の整数です。ユーザーが Tab キーを押したときにアクセスされるオブジェクトのタブ順序を作成します。このプロパティは、グラフィックシンボルでは使用できません。

例

次の例では、`mySymbol` オブジェクトの `tabIndex` プロパティを **3** に設定し、その値を [出力] パネルに表示します。

```
var mySymbol = fl.getDocumentDOM().selection[0];
mySymbol.tabIndex = 3;
fl.trace(mySymbol.tabIndex);
```

SymbolItem オブジェクト

継承 [Item オブジェクト](#) > SymbolItem オブジェクト

使用できるバージョン

Flash MX 2004

説明

SymbolItem オブジェクトは、[Item オブジェクト](#)のサブクラスです。

SymbolItem オブジェクトのメソッド一覧

SymbolItem オブジェクトでは、[Item オブジェクト](#)のメソッドに加えて、次のメソッドを使用することができます。

メソッド	説明
<code>symbolItem.convertToCompiledClip()</code>	ライブラリ内のシンボルアイテムをコンパイルされたムービークリップに変換します。
<code>symbolItem.exportSWC()</code>	シンボルアイテムを SWC ファイルに書き出します。
<code>symbolItem.exportSWF()</code>	シンボルアイテムを SWF ファイルに書き出します。

SymbolItem オブジェクトのプロパティ一覧

SymbolItem オブジェクトでは、[Item オブジェクト](#)のプロパティに加えて、次のプロパティを使用することができます。

プロパティ	説明
<code>symbolItem.scalingGrid</code>	アイテムの 9 スライスの拡大 / 縮小が有効かどうかを指定するブール値。
<code>symbolItem.scalingGridRect</code>	4 つの 9 スライスのガイドの位置を指定する矩形オブジェクト。
<code>symbolItem.sourceAutoUpdate</code>	FLA ファイルのパブリッシュ時にアイテムを更新するかどうかを指定するブール値。
<code>symbolItem.sourceFilePath</code>	ソースの FLA ファイルのパスを file:/// URI として指定するストリング。
<code>symbolItem.sourceLibraryName</code>	ソースファイルライブラリ内のアイテムの名前を指定するストリング。

プロパティ	説明
<code>symbolItem.symbolType</code>	シンボルの種類を指定するストリング。
<code>symbolItem.timeline</code>	読み取り専用。 Timeline オブジェクト。

symbolItem.convertToCompiledClip()

使用できるバージョン

Flash MX 2004

シンタックス

```
symbolItem.convertToCompiledClip()
```

パラメータ

なし

戻り値

なし

説明

メソッド。ライブラリ内のシンボルアイテムをコンパイルされたムービークリップに変換します。

例

次の例では、ライブラリ内のアイテムをコンパイルされたムービークリップに変換します。

```
fl.getDocumentDOM().library.items[3].convertToCompiledClip();
```

symbolItem.exportSWC()

使用できるバージョン

Flash MX 2004

シンタックス

```
symbolItem.exportSWC(outputURI)
```

パラメータ

outputURI file:/// URI として表され、メソッドでシンボルを書き出す先の SWC ファイルを指定するストリング。*outputURI* は、ローカルファイルを参照する必要があります。*outputURI* が存在しない場合、フォルダは作成されません。

戻り値

なし

説明

メソッド。シンボルアイテムを SWC ファイルに書き出します。

例

次の例では、ライブラリ内のアイテムを "tests" フォルダ内の "mySymbol.swc" という SWC ファイルに書き出します。

```
fl.getDocumentDOM().library.selectItem("mySymbol");  
var currentSelection = fl.getDocumentDOM().library.getSelectedItems();  
currentSelection[0].exportSWC("file:///Macintosh HD/SWCDirectory/mySymbol.swc");
```

symbolItem.exportSWF()

使用できるバージョン

Flash MX 2004

シンタックス

```
symbolItem.exportSWF(outputURI)
```

パラメータ

outputURI file:/// URI として表され、メソッドでシンボルを書き出す先の SWF ファイルを指定するストリング。*outputURI* は、ローカルファイルを参照する必要があります。*outputURI* が存在しない場合、フォルダは作成されません。

戻り値

なし

説明

メソッド。シンボルアイテムを SWF ファイルに書き出します。

例

次の例では、ライブラリ内のアイテムを "tests" フォルダ内の "my.swf" ファイルに書き出します。

```
fl.getDocumentDOM().library.items[0].exportSWF("file:///c:/tests/my.swf");
```

symbolItem.scalingGrid

使用できるバージョン

Flash 8

シンタックス

`symbolItem.scalingGrid`

説明

プロパティ。アイテムの 9 スライスの拡大 / 縮小が有効かどうかを指定するブール値です。

例

次の例では、ライブラリ内のアイテムの 9 スライスの拡大 / 縮小を有効にします。

```
fl.getDocumentDOM().library.items[0].scalingGrid = true;
```

関連項目

[symbolItem.scalingGridRect](#)

symbolItem.scalingGridRect

使用できるバージョン

Flash 8

シンタックス

`symbolItem.scalingGridRect`

説明

プロパティ。4 つの 9 スライスのガイドの位置を指定する矩形オブジェクトです。矩形の形式の詳細については、[document.addNewRectangle\(\)](#) を参照してください。

例

次の例では、9 スライスのガイドの位置を指定します。

```
fl.getDocumentDOM().library.items[0].scalingGridRect = {left:338, top:237,  
    right:3859, bottom:713};
```

関連項目

[symbolItem.scalingGrid](#)

symbolItem.sourceAutoUpdate

使用できるバージョン

Flash MX 2004

シンタックス

`symbolItem.sourceAutoUpdate`

説明

プロパティ。FLA ファイルのパブリッシュ時にアイテムを更新するかどうかを指定するブール値です。デフォルト値は `false` です。共有ライブラリシンボルで使用されます。

例

次の例では、ライブラリアイテムの `sourceAutoUpdate` プロパティを設定します。

```
fl.getDocumentDOM().library.items[0].sourceAutoUpdate = true;
```

symbolItem.sourceFilePath

使用できるバージョン

Flash MX 2004

シンタックス

`symbolItem.sourceFilePath`

説明

プロパティ。ソースの FLA ファイルのパスを `file:/// URI` として指定するストリングです。パスは、相対パスではなく絶対パスにする必要があります。このプロパティは共有ライブラリシンボルで使用されます。

例

次の例では、`sourceFilePath` プロパティの値を [出力] パネルに表示します。

```
fl.trace(fl.getDocumentDOM().library.items[0].sourceFilePath);
```

symbolItem.sourceLibraryName

使用できるバージョン

Flash MX 2004

シンタックス

symbolItem.sourceLibraryName

説明

プロパティ。ソースファイルライブラリ内のアイテムの名前を指定する文字列です。共有ライブラリシンボルで使用されます。

例

次の例では、sourceLibraryName プロパティの値を [出力] パネルに表示します。

```
fl.trace(fl.getDocumentDOM().library.items[0].sourceLibraryName);
```

symbolItem.symbolType

使用できるバージョン

Flash MX 2004

シンタックス

symbolItem.symbolType

説明

プロパティ。シンボルの種類を指定する文字列です。有効な値は、"movie clip"、"button"、および "graphic" です。

例

次の例では、symbolType プロパティの現在の値を表示し、その値を "button" に変更して、もう一度表示します。

```
alert(fl.getDocumentDOM().library.items[0].symbolType);  
fl.getDocumentDOM().library.items[0].symbolType = "button";  
alert(fl.getDocumentDOM().library.items[0].symbolType);
```

symbolItem.timeline

使用できるバージョン

Flash MX 2004

シンタックス

`symbolItem.timeline`

説明

読み取り専用プロパティ。[Timeline](#) オブジェクトです。

例

次の例では、ライブラリ内の選択されたムービークリップに含まれるレイヤーの数を取得して表示します。

```
var tl = fl.getDocumentDOM().library.getSelectedItems()[0].timeline;  
alert(tl.layerCount);
```


Text オブジェクト

継承 [Element オブジェクト](#) > Text オブジェクト

使用できるバージョン

Flash MX 2004

説明

Text オブジェクトは、ドキュメント内の1つのテキストアイテムを表します。テキストのすべてのプロパティがテキストブロック全体に関係します。

テキストフィールド内の連続したテキストのプロパティを設定するには、[519 ページの「TextRun オブジェクトのプロパティ一覧」](#)を参照してください。テキストフィールド内の選択範囲のプロパティを変更するには、`document.setTextAttr()` を使用し、テキストの範囲を指定するか、現在の選択範囲を使用します。

選択したテキストフィールドのテキストのプロパティを設定するには、`document.setProperty()` を使用します。次の例では、現在選択されているテキストフィールドを変数 `textVar` に代入します。

```
fl.getDocumentDOM().setProperty("variableName", "textVar");
```

Text オブジェクトのメソッド一覧

Text オブジェクトでは、[Element オブジェクト](#)のメソッドに加えて、次のメソッドを使用することができます。

メソッド	説明
<code>text.getTextAttr()</code>	オプションの <i>startIndex</i> および <i>endIndex</i> パラメータで識別されるテキストの指定した属性を取得します。
<code>text.getTextString()</code>	指定した範囲のテキストを取得します。
<code>text.setTextAttr()</code>	<i>startIndex</i> および <i>endIndex</i> で識別されるテキストに関連付けられる指定された属性を設定します。
<code>text.setTextString()</code>	この Text オブジェクト内のテキストストリングを変更します。

Text オブジェクトのプロパティ一覧

Text オブジェクトでは、[Element オブジェクト](#)のプロパティに加えて、次のプロパティを使用することができます。

プロパティ	説明
<code>text.accName</code>	[アクセシビリティ] パネルの [名前] フィールドと等しいストリング。
<code>text.antiAliasSharpness</code>	テキストのアンチエイリアスのシャープネスを指定する浮動小数値。
<code>text.antiAliasThickness</code>	テキストのアンチエイリアスの太さを指定する浮動小数値。
<code>text.autoExpand</code>	静止テキストフィールドの境界の幅、あるいはダイナミックテキストまたは入力テキストの境界の幅と高さを制御するブール値。
<code>text.border</code>	ダイナミックテキストまたは入力テキストの周囲に境界を表示するか (true) または表示しないか (false) を制御するブール値。
<code>text.description</code>	[アクセシビリティ] パネルの [説明] フィールドと等しいストリング。
<code>text.embeddedCharacters</code>	埋め込む文字を指定するストリング。これは、[文字オプション] ダイアログボックスでテキストを入力するのと等価です。
<code>text.embedRanges</code>	[文字オプション] ダイアログボックスで選択できるアイテムに対応する、区切られた複数の整数で構成されるストリング。
<code>text.fontRenderingMode</code>	テキストのレンダリングモードを指定するストリング。
<code>text.length</code>	読み取り専用。Text オブジェクト内の文字数を表す整数です。
<code>text.lineType</code>	行の種類を "single line"、"multiline"、"multiline no wrap"、または "password" に設定するストリング。
<code>text.maxCharacters</code>	ユーザーがこの Text オブジェクトに入力できる最大文字数を指定する整数。
<code>text.orientation</code>	テキストフィールドの向きを指定するストリング。
<code>text.renderAsHTML</code>	テキストを HTML として描画し、埋め込まれた HTML タグを解釈するかどうかを制御するブール値。
<code>text.scrollable</code>	テキストがスクロール可能か (true) またはスクロール不可か (false) を制御するブール値。
<code>text.selectable</code>	テキストが選択可能か (true) または選択不可か (false) を制御するブール値。入力テキストは常に選択可能です。
<code>text.selectionEnd</code>	テキストのダイレクト選択範囲が終了する位置のオフセットを指定するゼロから始まる整数。
<code>text.selectionStart</code>	テキストのダイレクト選択範囲が開始する位置のオフセットを指定するゼロから始まる整数。

プロパティ	説明
<code>text.shortcut</code>	[アクセシビリティ] パネルの [ショートカット] フィールドと等しいストリング。
<code>text.silent</code>	オブジェクトがアクセス可能かどうかを指定するブール値です。
<code>text.tabIndex</code>	[アクセシビリティ] パネルの [タブインデックス] フィールドと等しい整数です。
<code>text.textRuns</code>	読み取り専用。 <code>TextRun</code> オブジェクトの配列です。
<code>text.textType</code>	テキストフィールドの種類を指定するストリング。有効な値は、 <code>"static"</code> 、 <code>"dynamic"</code> 、および <code>"input"</code> です。
<code>text.useDeviceFonts</code>	ブール値。値が <code>true</code> の場合、デバイスフォントを使用してテキストが描画されます。
<code>text.variableName</code>	<code>Text</code> オブジェクトの内容を含むストリング。

text.accName

使用できるバージョン

Flash MX 2004

シンタックス

`text.accName`

説明

プロパティ。[アクセシビリティ] パネルの [名前] フィールドと等しいストリングです。スクリーンリーダーは、名前を読み上げてオブジェクトを識別します。このプロパティは、ダイナミックテキストでは使用できません。

例

次の例では、オブジェクトの名前を取得します。

```
var doc = fl.getDocumentDOM();
var theName = doc.getTimeline().layers[0].frames[0].elements[0].accName;
```

次の例では、現在選択されているオブジェクトの名前を設定します。

```
fl.getDocumentDOM().selection[0].accName = "Home Button";
```

text.antiAliasSharpness

使用できるバージョン

Flash 8

シンタックス

`text.antiAliasSharpness`

説明

プロパティ。テキストのアンチエイリアスのシャープネスを指定する浮動小数値です。このプロパティは、テキスト描画のくっきり度を制御します。高い値を指定すると、テキストがシャープになり、鮮明になります。値 0 は通常のシャープネスを示します。このプロパティは、`text.fontRenderingMode` が "customThicknessSharpness" に設定されている場合のみ使用できます。

例

詳細については、[text.fontRenderingMode](#) を参照してください。

関連項目

[text.antiAliasThickness](#), [text.fontRenderingMode](#)

text.antiAliasThickness

使用できるバージョン

Flash 8

シンタックス

`text.antiAliasThickness`

説明

プロパティ。テキストのアンチエイリアスの太さを指定する浮動小数値です。このプロパティは、テキスト描画の太さを制御します。高い値を指定すると、テキストが太くなります。値 0 は通常の太さを示します。このプロパティは、`text.fontRenderingMode` が "customThicknessSharpness" に設定されている場合のみ使用できます。

例

詳細については、[text.fontRenderingMode](#) を参照してください。

関連項目

[text.antiAliasSharpness](#), [text.fontRenderingMode](#)

text.autoExpand

使用できるバージョン

Flash MX 2004

シンタックス

`text.autoExpand`

説明

プロパティ。ブール値です。静止テキストフィールドの場合、値が `true` になっていると、すべてのテキストが表示されるように境界が拡張されます。ダイナミックテキストフィールドまたは入力テキストフィールドの場合、値が `true` になっていると、すべてのテキストが表示されるように境界の幅と高さが拡張されます。

例

次の例では、`autoExpand` プロパティの値を `true` に設定します。

```
fl.getDocumentDOM().selection[0].autoExpand = true;
```

text.border

使用できるバージョン

Flash MX 2004

シンタックス

`text.border`

説明

プロパティ。ブール値です。値が `true` の場合は、テキストの周囲に境界が表示されます。

例

次の例では、`border` プロパティの値を `true` に設定します。

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].border = true;
```

text.description

使用できるバージョン

Flash MX 2004

シンタックス

text.description

説明

プロパティ。[アクセシビリティ] パネルの [説明] フィールドと等しいストリングです。この説明がスクリーンリーダーによって読み上げられます。

例

次の例では、オブジェクトの説明を取得します。

```
var doc = fl.getDocumentDOM();  
var desc = doc.getTimeline().layers[0].frames[0].elements[0].description;
```

次の例では、オブジェクトの説明を設定します。

```
var doc = fl.getDocumentDOM();  
doc.getTimeline().layers[0].frames[0].elements[0].description= "Enter your name  
here";
```

text.embeddedCharacters

使用できるバージョン

Flash MX 2004

シンタックス

text.embeddedCharacters

説明

プロパティ。埋め込む文字を指定するストリングです。これは、[文字オプション] ダイアログボックスでテキストを入力するのと等価です。

このプロパティはダイナミックテキストまたは入力テキストでのみ使用できます。他の種類のテキストで使用すると警告が表示されます。

例

次の例では、embeddedCharacters プロパティを "abc" に設定します。

```
fl.getDocumentDOM().selection[0].embeddedCharacters = "abc";
```

text.embedRanges

使用できるバージョン

Flash MX 2004

シンタックス

text.embedRanges

説明

プロパティ。[文字オプション] ダイアログボックスで選択できるアイテムに対応する、区切られた複数の整数で構成されるストリングです。このプロパティはダイナミックテキストまたは入力テキストでのみ使用できます。静止テキストで使用すると警告が表示されます。



このプロパティは、"Configuration/Font Embedding" フォルダ内の XML ファイルに対応します。

例

次の例では、embedRanges プロパティを "1|3|7" に設定します。

```
var doc = fl.getDocumentDOM();  
doc.getTimeline().layers[0].frames[0].elements[0].embedRanges = "1|3|7";
```

次の例では、プロパティをリセットします。

```
var doc = fl.getDocumentDOM();  
doc.getTimeline().layers[0].frames[0].elements[0].embedRanges = "";
```

text.fontRenderingMode

使用できるバージョン

Flash 8

シンタックス

text.fontRenderingMode

説明

プロパティ。テキストのレンダリングモードを指定するストリングです。このプロパティは、ステージと Flash Player の両方でのテキストの表示方法に影響します。有効な値を次の表に示します。

プロパティ値	テキストのレンダリング方法
device	デバイスフォントでテキストをレンダリングします。
bitmap	エイリアス処理されたテキストを、ビットマップまたはピクセルフォントとしてレンダリングします。

プロパティ値	テキストのレンダリング方法
standard	Flash MX 2004 で使用される標準のアンチエイリアス方式を使用してテキストをレンダリングします。アニメーション化された非常に大きなテキストや傾斜したテキストに使用するには最適の設定です。
advanced	Flash 8 に実装されている高度なアンチエイリアスフォントレンダリング技術を使用して、テキストをレンダリングします。この技術は、より優れたアンチエイリアス処理を提供し、特に小さなテキストの読みやすさを向上します。
customThicknessSharpness	Flash 8 に実装されている高度のアンチエイリアスフォントレンダリング技術の使用時に、テキストのシャープネスおよび太さをカスタム設定することができます。

例

次の例では、customThicknessSharpness 値を使用して、テキストのシャープネスおよび太さを指定する方法を示します。

```
fl.getDocumentDOM().setElementProperty("fontRenderingMode",
    "customThicknessSharpness");
fl.getDocumentDOM().setElementProperty("antiAliasSharpness", 400);
fl.getDocumentDOM().setElementProperty("antiAliasThickness", -200);
```

関連項目

[text.antiAliasSharpness](#), [text.antiAliasThickness](#)

text.getTextAttr()

使用できるバージョン

Flash MX 2004

シンタックス

```
text.getTextAttr(attrName [, startIndex [, endIndex]])
```

パラメータ

attrName 返される TextAttrs オブジェクトプロパティの名前を指定するストリング。

	<i>attrName</i> の有効な値の一覧については、 TextAttrs オブジェクトのプロパティ一覧 を参照してください。
---	--

startIndex 最初の文字のインデックスである整数。このパラメータはオプションです。

endIndex テキストの範囲の終了位置を指定する整数。この範囲は *startIndex* から始まり、*endIndex* の直前で終わります。このパラメータはオプションです。

戻り値

attrName パラメータで指定された属性の値

説明

メソッド。オプションの *startIndex* および *endIndex* パラメータで識別されるテキストの *attrName* パラメータによって指定された属性を取得します。属性が指定された範囲と矛盾している場合、*undefined* が返されます。オプションのパラメータ *startIndex* と *endIndex* を省略すると、メソッドではテキスト範囲全体が使用されます。*startIndex* のみを指定した場合、その位置の1つの文字が範囲として使用されます。*startIndex* と *endIndex* の両方を指定した場合、範囲は、*startIndex* から始まり、*endIndex* の直前で終わります。

例

次の例では、現在選択されているテキストフィールドのフォントサイズを取得し、それを表示します。

```
var TheTextSize = fl.getDocumentDOM().selection[0].getTextAttr("size");
fl.trace(TheTextSize);
```

次の例では、選択したテキストフィールドのテキストの塗りのカラーを取得します。

```
var TheFill = fl.getDocumentDOM().selection[0].getTextAttr("fillColor");
fl.trace(TheFill);
```

次の例では、3 番目の文字のサイズを取得します。

```
var Char3 = fl.getDocumentDOM().selection[0].getTextAttr("size", 2);
fl.trace(Char3);
```

次の例では、選択したテキストフィールドの 3 番目から 8 番目までの文字のカラーを取得します。

```
fl.getDocumentDOM().selection[0].getTextAttr("fillColor", 2, 8);
```

text.getTextString()

使用できるバージョン

Flash MX 2004

シンタックス

```
text.getTextString([startIndex [, endIndex]])
```

パラメータ

startIndex 最初の文字のゼロから始まるインデックスである整数。このパラメータはオプションです。

endIndex テキストの範囲の終了位置を指定する整数。この範囲は *startIndex* から始まり、*endIndex* の直前で終わります。このパラメータはオプションです。

戻り値

指定した範囲のテキストのストリング

説明

メソッド。指定した範囲のテキストを取得します。オプションのパラメータ *startIndex* および *endIndex* を省略した場合、テキストストリング全体が返されます。*startIndex* のみを指定した場合は、インデックスの場所からフィールドの最後までまでのテキストが返されます。*startIndex* と *endIndex* の両方を指定した場合、*startIndex* から *endIndex* の直前までのストリングが返されます。

例

次の例では、選択したテキストフィールドの 5 番目の文字から最後までまでの文字を取得します。

```
var myText = fl.getDocumentDOM().selection[0].getTextString(4);
fl.trace(myText);
```

次の例では、選択したテキストフィールド内の 4 番目から 9 番目までの文字を取得します。

```
var myText = fl.getDocumentDOM().selection[0].getTextString(3, 9);
fl.trace(myText);
```

text.length

使用できるバージョン

Flash MX 2004

シンタックス

`text.length`

説明

読み取り専用プロパティ。Text オブジェクト内の文字数を表す整数です。

例

次の例では、選択したテキストの文字数を返します。

```
var textLength = fl.getDocumentDOM().selection[0].length;
```

text.lineType

使用できるバージョン

Flash MX 2004

シンタックス

text.lineType

説明

プロパティ。行の種類を設定するストリングです。有効な値は、"single line"、"multiline"、"multiline no wrap"、および "password" です。

このプロパティはダイナミックテキストまたは入力テキストでのみ使用できます。静止テキストで使用すると警告が表示されます。値 "password" は、入力テキストでのみ使用できます。

例

次の例では、lineType プロパティの値を "multiline no wrap" に設定します。

```
fl.getDocumentDOM().selection[0].lineType = "multiline no wrap";
```

text.maxCharacters

使用できるバージョン

Flash MX 2004

シンタックス

text.maxCharacters

説明

プロパティ。ユーザーがこの Text オブジェクトに入力できる最大文字数を指定する整数です。

このプロパティは入力テキストでのみ使用できます。他の種類のテキストでこのプロパティを使用すると、警告が表示されます。

例

次の例では、maxCharacters プロパティの値を 30 に設定します。

```
fl.getDocumentDOM().selection[0].maxCharacters = 30;
```

text.orientation

使用できるバージョン

Flash MX 2004

シンタックス

text.orientation

説明

プロパティ。テキストフィールドの向きを指定するストリングです。有効な値は、"horizontal"、"vertical left to right"、および "vertical right to left" です。

このプロパティは静止テキストでのみ使用できます。他の種類のテキストで使用すると警告が表示されます。

例

次の例では、orientation プロパティを "vertical right to left" に設定します。

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].orientation =  
    "vertical right to left";
```

text.renderAsHTML

使用できるバージョン

Flash MX 2004

シンタックス

text.renderAsHTML

説明

プロパティ。ブール値です。値が true の場合、テキストが HTML として描画され、埋め込まれた HTML タグが解釈されます。

このプロパティはダイナミックテキストまたは入力テキストでのみ使用できます。他の種類のテキストで使用すると警告が表示されます。

例

次の例では、renderAsHTML プロパティを true に設定します。

```
fl.getDocumentDOM().selection[0].renderAsHTML = true;
```

text.scrollable

使用できるバージョン

Flash MX 2004

シンタックス

`text.scrollable`

説明

プロパティ。ブール値です。値が `true` の場合、テキストがスクロール可能になります。

このプロパティはダイナミックテキストまたは入力テキストでのみ使用できます。静止テキストで使用すると警告が表示されます。

例

次の例では、`scrollable` プロパティを `false` に設定します。

```
fl.getDocumentDOM().selection[0].scrollable = false;
```

text.selectable

使用できるバージョン

Flash MX 2004

シンタックス

`text.selectable`

説明

プロパティ。ブール値です。値が `true` の場合、テキストが選択可能になります。

入力テキストは常に選択可能です。このプロパティを `false` に設定して、入力テキストで使用すると、警告が表示されます。

例

次の例では、`selectable` プロパティを `true` に設定します。

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].selectable =  
    true;
```

text.selectionEnd

使用できるバージョン

Flash MX 2004

シンタックス

`text.selectionEnd`

説明

プロパティ。テキストのダイレクト選択範囲の終了位置を指定するゼロから始まる整数です。詳細については、[text.selectionStart](#) を参照してください。

text.selectionStart

使用できるバージョン

Flash MX 2004

シンタックス

`text.selectionStart`

説明

プロパティ。テキストのダイレクト選択範囲の開始位置を指定するゼロから始まる整数です。このプロパティを `text.selectionEnd` と共に使用して、文字の範囲を選択することができます。
`text.selectionEnd` の直前までの文字が選択されます。[text.selectionEnd](#) を参照してください。

- 挿入ポイントがあるか何も選択されていない場合、`text.selectionEnd` は `text.selectionStart` と同じになります。
- `text.selectionStart` が `text.selectionEnd` より大きい値に設定されている場合、`text.selectionEnd` は `text.selectionStart` に設定され、テキストは選択されません。

例

次の例では、テキストのダイレクト選択範囲の開始位置を 6 番目の文字に設定します。

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].selectionStart = 5;
```

次の例では、"My name is Barbara" というテキストが含まれるテキストフィールドから "Barbara" という文字を選択し、緑色の太字にフォーマットします。

```
fl.getDocumentDOM().selection[0].selectionStart = 11;  
fl.getDocumentDOM().selection[0].selectionEnd = 18;  
var s = fl.getDocumentDOM().selection[0].selectionStart;  
var e = fl.getDocumentDOM().selection[0].selectionEnd;
```

```
fl.getDocumentDOM().setElementTextAttr('bold', true, s, e);
fl.getDocumentDOM().setElementTextAttr("fillColor", "#00ff00", s, e);
```

text.setTextAttr()

使用できるバージョン

Flash MX 2004

シンタックス

```
text.setTextAttr(attrName, attrValue [, startIndex [, endIndex]])
```

パラメータ

attrName 変更する TextAttrs オブジェクトプロパティの名前を指定するストリング。

attrValue TextAttrs オブジェクトプロパティの値。

× 中	<i>attrName</i> および <i>attrValue</i> の有効な値の一覧については、 509 ページの「TextAttrs オブジェクトのプロパティ一覧」 を参照してください。
--------	---

startIndex 配列内の最初の文字のゼロから始まるインデックスである整数。このパラメータはオプションです。

endIndex 選択したテキストストリングの終了位置のインデックスを指定する整数。選択範囲は *startIndex* から始まり、*endIndex* の直前で終わります。このパラメータはオプションです。

戻り値

なし

説明

メソッド。 *startIndex* および *endIndex* で識別されるテキストに関連付けられる *attrName* パラメータによって指定された属性を *attrValue* によって指定された値に設定します。このメソッドを使用して、TextRun エlement を範囲とするテキスト ([TextRun オブジェクト](#)を参照) または既存の TextRun エlement の一部の属性を変更することができます。このメソッドを使用すると、このオブジェクトの text.textRuns 配列内の TextRun エlement の位置と数に変更される場合があります。[text.textRuns](#) を参照してください。

オプションのパラメータを省略した場合、メソッドで Text オブジェクトの文字範囲全体が使用されます。 *startIndex* のみを指定した場合、その位置の1つの文字が範囲になります。 *startIndex* と *endIndex* の両方を指定した場合、範囲は、 *startIndex* から始まり、 *endIndex* にある文字の直前で終わります。

例

次の例では、選択したテキストフィールドをイタリックに設定します。

```
fl.getDocumentDOM().selection[0].setTextAttr("italic", true);
```

次の例では、3 番目の文字のサイズを 10 に設定します。

```
fl.getDocumentDOM().selection[0].setTextAttr("size", 10, 2);
```

次の例では、選択したテキストの 3 番目から 8 番目までの文字のカラーを赤に設定します。

```
fl.getDocumentDOM().selection[0].setTextAttr("fillColor", 0xff0000, 2, 8);
```

text.setTextString()

使用できるバージョン

Flash MX 2004

シンタックス

```
text.setTextString(text [, startIndex [, endIndex]])
```

パラメータ

text Text オブジェクトに挿入される文字で構成されるストリング。

startIndex ストリング内のテキストが挿入される場所にある文字のゼロから始まるインデックスを指定する整数。このパラメータはオプションです。

endIndex 選択したテキストストリング内の終了位置のインデックスを指定する整数。新しいテキストによって、*startIndex* から *endIndex* の直前までのテキストが上書きされます。このパラメータはオプションです。

戻り値

なし

説明

プロパティ。この Text オブジェクト内のテキストストリングを変更します。オプションのパラメータを省略した場合、Text オブジェクト全体が置き換えられます。*startIndex* のみを指定した場合、指定したストリングが *startIndex* の位置に挿入されます。*startIndex* と *endIndex* の両方を指定した場合、指定したストリングによって、*startIndex* から *endIndex* の直前までのテキストセグメントが置き換えられます。

例

次の例では、ストリング "this is a string" を選択したテキストフィールドに代入します。

```
fl.getDocumentDOM().selection[0].setTextString("this is a string");
```

次の例では、選択したテキストフィールドの 5 番目の文字の前に "abc" を挿入します。

```
fl.getDocumentDOM().selection[0].setTextString("01234567890");  
fl.getDocumentDOM().selection[0].setTextString("abc", 4);  
// これでテキストフィールドは "0123abc4567890" になります。
```


次の例では、選択したテキストストリングの 3 番目から 8 番目までの文字のテキストをストリング "abcdefghij" に置き換えます。 *startIndex* から *endIndex* までの文字が上書きされます。 *endIndex* から始まる文字が、挿入されたストリングの後に続きます。

```
fl.getDocumentDOM().selection[0].setTextString("01234567890");  
fl.getDocumentDOM().selection[0].setTextString("abcdefghij", 2, 8);  
// これでテキストフィールドは "01abcdefghij890" になります。
```

text.shortcut

使用できるバージョン

Flash MX 2004

シンタックス

text.shortcut

説明

プロパティ。[アクセシビリティ] パネルの [ショートカット] フィールドと等しいストリングです。このショートカットがスクリーンリーダーによって読み上げられます。このプロパティは、ダイナミックテキストでは使用できません。

例

次の例では、選択したオブジェクトのショートカットキーを取得し、その値を表示します。

```
var theShortcut = fl.getDocumentDOM().selection[0].shortcut;  
fl.trace(theShortcut);
```

次の例では、選択したオブジェクトのショートカットキーを設定します。

```
fl.getDocumentDOM().selection[0].shortcut = "Ctrl+i";
```

text.silent

使用できるバージョン

Flash MX 2004

シンタックス

text.silent

説明

プロパティ。オブジェクトがアクセス可能かどうかを指定するブール値です。これは、[アクセシビリティ] パネルの [オブジェクトをアクセス可能にする] の設定の逆ロジックと等価です。つまり、silent が true の場合、[オブジェクトをアクセス可能にする] をオフにした場合と同じになります。false の場合、[オブジェクトをアクセス可能にする] がオンになります。

例

次の例では、オブジェクトがアクセス可能かどうかを確認します。値が `false` の場合、オブジェクトがアクセス可能なことを意味します。

```
var isSilent =  
    fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].silent;
```

次の例では、オブジェクトをアクセス可能に設定します。

```
fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].silent =  
    false;
```

text.tabIndex

使用できるバージョン

Flash MX 2004

シンタックス

`text.tabIndex`

説明

プロパティ。[アクセシビリティ] パネルの [タブインデックス] フィールドと等しい整数です。この値を使用して、ユーザーが **Tab** キーを押したときにオブジェクトがアクセスされる順序を決定することができます。

例

次の例では、現在選択されているオブジェクトの `tabIndex` を取得します。

```
var theTabIndex = fl.getDocumentDOM().selection[0].tabIndex;
```

次の例では、現在選択されているオブジェクトの `tabIndex` を設定します。

```
fl.getDocumentDOM().selection[0].tabIndex = 1;
```

text.textRuns

使用できるバージョン

Flash MX 2004

シンタックス

`text.textRuns`

説明

読み取り専用プロパティ。TextRun オブジェクトの配列です ([TextRun オブジェクト](#)を参照)。

例

次の例では、textRuns プロパティの値を myTextRuns 変数に格納します。

```
var myTextRuns = fl.getDocumentDOM().selection[0].textRuns;
```

text.textType

使用できるバージョン

Flash MX 2004

シンタックス

text.textType

説明

プロパティ。テキストフィールドの種類を指定するストリングです。有効な値は、"static"、"dynamic"、および "input" です。

例

次の例では、textType プロパティを "input" に設定します。

```
fl.getDocumentDOM().selection[0].textType = "input";
```

text.useDeviceFonts

使用できるバージョン

Flash MX 2004

シンタックス

text.useDeviceFonts

説明

プロパティ。ブール値です。値が true の場合、デバイスフォントを使用してテキストが描画されます。

例

次の例では、テキストの描画時にデバイスフォントを使用します。

```
fl.getDocumentDOM().selection[0].useDeviceFonts = true;
```

text.variableName

使用できるバージョン

Flash MX 2004

シンタックス

`text.variableName`

説明

プロパティ。Text オブジェクトに関連付けられている変数の名前を含むストリングです。このプロパティはダイナミックテキストまたは入力テキストでのみ使用できます。他の種類のテキストで使用するすると警告が表示されます。

TextAttrs オブジェクト

使用できるバージョン

Flash MX 2004

説明

TextAttrs オブジェクトは、ダイレクト選択範囲に適用できるテキストのすべてのプロパティを含みます。このオブジェクトは、TextRun オブジェクトのプロパティ ([textRun.textAttrs](#)) です。

TextAttrs オブジェクトのプロパティ一覧

TextAttrs オブジェクトでは次のプロパティを使用できます。

プロパティ	説明
textAttrs.aliasText	小さなテキストの読みやすさを向上させるために最適化されたメソッドを使用してテキストが描画されるように指定するブール値。
textAttrs.alignment	段落の位置揃えを指定するストリング。有効な値は、"left"、"center"、"right"、および "justify" です。
textAttrs.autoKern	テキストをカーニングするためにフォントの自動カーニング情報を使用するか (true) または無視するか (false) を決定するブール値。
textAttrs.bold	ブール値。値が true の場合、テキストは太字のフォントで表示されます。
textAttrs.characterPosition	テキストのベースラインを決定するストリング。
textAttrs.characterSpacing	このプロパティは使用しないでください。代わりに textAttrs.letterSpacing を使用してください。文字の間隔を表す整数。
textAttrs.face	"Arial" などのフォントの名前を表すストリング。
textAttrs.fillColor	塗りのカラーを表す、ストリング、16 進数値、または整数。
textAttrs.indent	段落のインデントを指定する整数。
textAttrs.italic	ブール値。値が true の場合、テキストはイタリックのフォントで表示されます。
textAttrs.leftMargin	段落の左余白を指定する整数。
textAttrs.letterSpacing	文字の間隔を表す整数。
textAttrs.lineSpacing	段落の行間 (行送り) を指定する整数。

プロパティ	説明
<code>textAttrs.rightMargin</code>	段落の右余白を指定する整数。
<code>textAttrs.rotation</code>	ブール値。値が <code>true</code> の場合、テキストの文字が 90 度回転します。デフォルト値は <code>false</code> です。
<code>textAttrs.size</code>	フォントのサイズを指定する整数。
<code>textAttrs.target</code>	テキストフィールドの <code>target</code> プロパティを表すストリング。
<code>textAttrs.url</code>	テキストフィールドの <code>URL</code> プロパティを表すストリング。

textAttrs.aliasText

使用できるバージョン

Flash MX 2004

シンタックス

`textAttrs.aliasText`

説明

プロパティ。小さなテキストの読みやすさを向上させるために最適化されたメソッドを使用してテキストが描画されるように指定するブール値です。

例

次の例では、現在選択されているテキストフィールド内のすべてのテキストの `aliasText` プロパティを `true` に設定します。

```
fl.getDocumentDOM().setElementTextAttr('aliasText', true);
```

textAttrs.alignment

使用できるバージョン

Flash MX 2004

シンタックス

`textAttrs.alignment`

説明

プロパティ。段落の位置揃えを指定するストリングです。有効な値は、"`left`"、"`center`"、"`right`"、および "`justify`" です。

例

次の例では、インデックス 0 からインデックス 3 の直前までの文字を含む段落を設定します。
これは、同じ段落内にある指定範囲外の文字に影響する可能性があります。

```
fl.getDocumentDOM().setTextSelection(0, 3);  
fl.getDocumentDOM().setElementTextAttr("alignment", "justify");
```

textAttrs.autoKern

使用できるバージョン

Flash MX 2004

シンタックス

textAttrs.autoKern

説明

プロパティ。テキストをカーニングするときにフォントの自動カーニング情報を使用するか (true) または無視するか (false) を決定するブール値です。

例

次の例では、インデックス 2 からインデックス 6 の直前までの文字を選択し、autoKern プロパティを true に設定します。

```
fl.getDocumentDOM().setTextSelection(3, 6);  
fl.getDocumentDOM().setElementTextAttr('autoKern', true);
```

textAttrs.bold

使用できるバージョン

Flash MX 2004

シンタックス

textAttrs.bold

説明

プロパティ。ブール値です。値が true の場合、テキストは太字のフォントで表示されます。

例

次の例では、選択したテキストオブジェクトの最初の文字を選択し、bold プロパティを true に設定します。

```
fl.getDocumentDOM().setTextSelection(0, 1);  
fl.getDocumentDOM().setElementTextAttr('bold', true);
```

textAttrs.characterPosition

使用できるバージョン

Flash MX 2004

シンタックス

textAttrs.characterPosition

説明

プロパティ。テキストのベースラインを決定するストリングです。有効な値は、"normal"、"subscript"、および "superscript" です。このプロパティは静止テキストにのみ適用されます。

例

次の例では、テキストフィールドのインデックス 2 からインデックス 6 の直前までの文字を選択し、characterPosition プロパティを "subscript" に設定します。

```
fl.getDocumentDOM().setTextSelection(2, 6);  
fl.getDocumentDOM().setElementTextAttr("characterPosition", "subscript");
```

textAttrs.characterSpacing

使用できるバージョン

Flash MX 2004。Flash 8 以降では、このプロパティを使用しないでください。代わりに [textAttrs.letterSpacing](#) を使用してください。

シンタックス

textAttrs.characterSpacing

説明

プロパティ。文字の間隔を表す整数です。有効な値は、-60 ～ 60 です。

このプロパティは静止テキストにのみ適用されます。他の種類のテキストで使用すると警告が表示されます。

例

次の例では、選択したテキストフィールドの文字間隔を 10 に設定します。

```
fl.getDocumentDOM().setElementTextAttr("characterSpacing", 10);
```


textAttrs.face

使用できるバージョン

Flash MX 2004

シンタックス

textAttrs.face

説明

プロパティ。"Arial" などのフォントの名前を表すストリングです。

例

次の例では、選択したテキストフィールドのインデックス 2 からインデックス 8 の直前までの文字のフォントを "Arial" に設定します。

```
fl.getDocumentDOM().selection[0].setTextAttr("face", "Arial", 2, 8);
```

textAttrs.fillColor

使用できるバージョン

Flash MX 2004

シンタックス

textAttrs.fillColor

説明

プロパティ。塗りのカラー。次のいずれかの形式です。

- "#RRGGBB" または "#RRGGBBAA" 形式のストリング
- 0xRRGGBB 形式の 16 進数
- 16 進数と等しい 10 進表現の整数

例

次の例では、選択したテキストフィールドのインデックス 2 からインデックス 8 の直前までの文字のカラーを赤に設定します。

```
fl.getDocumentDOM().selection[0].setTextAttr("fillColor", 0xff0000, 2, 8);
```

textAttrs.indent

使用できるバージョン

Flash MX 2004

シンタックス

`textAttrs.indent`

説明

プロパティ。段落のインデントを指定する整数です。有効な値は、-720 ～ 720 です。

例

次の例では、選択したテキストフィールドのインデックス 2 からインデックス 8 の直前までの文字のインデントを 100 に設定します。これは、同じ段落内にある指定範囲外の文字に影響する可能性があります。

```
fl.getDocumentDOM().selection[0].setTextAttr("indent", 100, 2, 8);
```

textAttrs.italic

使用できるバージョン

Flash MX 2004

シンタックス

`textAttrs.italic`

説明

プロパティ。ブール値です。値が true の場合、テキストはイタリックのフォントで表示されます。

例

次の例では、選択したテキストフィールドをイタリックに設定します。

```
fl.getDocumentDOM().selection[0].setTextAttr("italic", true);
```

textAttrs.leftMargin

使用できるバージョン

Flash MX 2004

シンタックス

`textAttrs.leftMargin`

説明

プロパティ。段落の左余白を指定する整数です。有効な値は、0 ～ 720 です。

例

次の例では、選択したテキストフィールドのインデックス 2 からインデックス 8 の直前までの文字の `leftMargin` プロパティを 100 に設定します。これは、同じ段落内にある指定範囲外の文字に影響する可能性があります。

```
fl.getDocumentDOM().selection[0].setTextAttr("leftMargin", 100, 2, 8);
```

textAttrs.letterSpacing

使用できるバージョン

Flash 8

シンタックス

`textAttrs.letterSpacing`

説明

プロパティ。文字の間隔を表す整数です。有効な値は、-60 ～ 60 です。

このプロパティは静止テキストにのみ適用されます。他の種類のテキストで使用すると警告が表示されます。

例

次のコードでは、インデックス 0 からインデックス 10 の直前までの文字を選択し、文字間隔を 60 に設定します。

```
fl.getDocumentDOM().setTextSelection(0, 10);  
fl.getDocumentDOM().setElementTextAttr("letterSpacing", 60);
```

textAttrs.lineSpacing

使用できるバージョン

Flash MX 2004

シンタックス

`textAttrs.lineSpacing`

説明

プロパティ。段落の行間 (行送り) を指定する整数です。有効な値は、-360 ～ 720 です。

例

次の例では、選択したテキストフィールドの `lineSpacing` プロパティを 100 に設定します。

```
fl.getDocumentDOM().selection[0].setTextAttr("lineSpacing", 100);
```

textAttrs.rightMargin

使用できるバージョン

Flash MX 2004

シンタックス

`textAttrs.rightMargin`

説明

プロパティ。段落の右余白を指定する整数です。有効な値は、0 ～ 720 です。

例

次の例では、選択したテキストフィールドのインデックス 2 からインデックス 8 の直前までの文字の `rightMargin` プロパティを 100 に設定します。これは、同じ段落内にある指定範囲外の文字に影響する可能性があります。

```
fl.getDocumentDOM().selection[0].setTextAttr("rightMargin", 100, 2, 8);
```

textAttrs.rotation

使用できるバージョン

Flash MX 2004

シンタックス

`textAttrs.rotation`

説明

プロパティ。ブール値です。値が `true` の場合、テキストの文字が 90 度回転します。デフォルト値は `false` です。このプロパティは、縦書きの静止テキストにのみ適用されます。他の種類のテキストで使用するすると警告が表示されます。

例

次の例では、選択したテキストフィールドの回転を `true` に設定します。

```
fl.getDocumentDOM().setElementTextAttr("rotation", true);
```

textAttrs.size

使用できるバージョン

Flash MX 2004

シンタックス

textAttrs.size

説明

プロパティ。フォントのサイズを指定する整数です。

例

次の例では、インデックス 2 にある文字のサイズを取得し、その結果を [出力] パネルに表示します。

```
fl.outputPanel.trace(fl.getDocumentDOM().selection[0].getTextAttr("size", 2));
```

textAttrs.target

使用できるバージョン

Flash MX 2004

シンタックス

textAttrs.target

説明

プロパティ。テキストフィールドの target プロパティを表すストリングです。このプロパティは静止テキストでのみ使用できます。

例

次の例では、現在のシーンの最上位レイヤーの最初のフレームのテキストフィールドの target プロパティを取得し、[出力] パネルに表示します。

```
fl.outputPanel.trace(fl.getDocumentDOM().getTimeline().layers[0].frames[0].elements[0].getTextAttr("target"));
```

textAttrs.url

使用できるバージョン

Flash MX 2004

シンタックス

`textAttrs.url`

説明

プロパティ。テキストフィールドの URL プロパティを表す文字列です。このプロパティは静止テキストでのみ使用できます。

例

次の例では、選択したテキストフィールドの URL を `http://www.adobe.com` に設定します。

```
fl.getDocumentDOM().setElementTextAttr("url", "http://www.adobe.com");
```

TextRun オブジェクト

使用できるバージョン

Flash MX 2004

説明

TextRun オブジェクトは、[TextAttrs オブジェクト](#)のすべてのプロパティと一致する属性を持つ連続した文字を表します。このオブジェクトは、Text オブジェクトのプロパティ ([text.textRuns](#)) です。

TextRun オブジェクトのプロパティ一覧

TextRun オブジェクトでは、[Text オブジェクト](#)で使用するプロパティに加えて、次のプロパティを使用できます。

プロパティ	説明
textRun.characters	TextRun オブジェクトに含まれるテキストを表すストリング。
textRun.textAttrs	連続したテキストの属性を含む TextAttrs オブジェクト 。

textRun.characters

使用できるバージョン

Flash MX 2004

シンタックス

`textRun.characters`

説明

プロパティ。TextRun オブジェクトに含まれるテキストです。

例

次の例では、選択したテキストフィールド内の最初の連続した文字を構成する文字を [出力] パネルに表示します。

```
fl.trace(fl.getDocumentDOM().selection[0].textRuns[0].characters);
```

textRun.textAttrs

使用できるバージョン

Flash MX 2004

シンタックス

`textRun.textAttrs`

説明

プロパティ。連続したテキストの属性を含む [TextAttrs オブジェクト](#) です。

例

次の例では、選択したテキストフィールドの最初の連続した文字のプロパティを [出力] パネルに表示します。

```
var curTextAttrs = fl.getDocumentDOM().selection[0].textRuns[0].textAttrs;
for (var prop in curTextAttrs) {
    fl.trace(prop + " = " + curTextAttrs[prop]);
}
```


Timeline オブジェクト

使用できるバージョン

Flash MX 2004

説明

Timeline オブジェクトは、Flash タイムラインを表します。現在のドキュメントのタイムラインにアクセスするには、`fl.getDocumentDOM().getTimeline()` を使用します。このメソッドは、現在のシーンまたは編集されているシンボルのタイムラインを返します。

シーンを操作している場合は、各シーンのタイムラインにインデックス値が割り当てられており、`fl.getDocumentDOM().timelines[i]` を使用して現在のドキュメントのタイムラインにアクセスすることができます。この例では、`i` がタイムラインの値のインデックスです。

Timeline オブジェクトのメソッドとプロパティを使用してフレームを操作する場合、フレーム値は、ゼロから始まるインデックスであり、タイムライン内のフレームのシーケンスの実際のフレーム番号ではないことに注意してください。つまり、最初のフレームのインデックスは 0 になります。

Timeline オブジェクトのメソッド一覧

Timeline オブジェクトでは次のメソッドを使用できます。

メソッド	説明
<code>timeline.addMotionGuide()</code>	カレントレイヤーの上にモーションガイドレイヤーを追加し、カレントレイヤーを新しく追加したガイドレイヤーに割り当てます。
<code>timeline.addNewLayer()</code>	新しいレイヤーをドキュメントに追加し、そのレイヤーをカレントレイヤーにします。
<code>timeline.clearFrames()</code>	カレントレイヤー上のフレームまたはフレームの範囲からすべての内容を削除します。
<code>timeline.clearKeyframes()</code>	カレントレイヤー上のキーフレームを通常のフレームに変換し、その内容を削除します。
<code>timeline.convertToBlankKeyframes()</code>	カレントレイヤー上のフレームを空白のキーフレームに変換します。
<code>timeline.convertToKeyframes()</code>	カレントレイヤー上のフレームの範囲をキーフレームに変換します。フレームが指定されていない場合は、選択範囲を変換します。
<code>timeline.copyFrames()</code>	カレントレイヤー上のフレームの範囲をクリップボードにコピーします。

メソッド	説明
<code>timeline.copyMotion()</code>	モーショントウィーンまたはフレームアニメーションから、選択したフレーム上のモーションをコピーし、他のフレームに適用できるようにします。
<code>timeline.copyMotionAsAS3()</code>	モーショントウィーンまたはフレームアニメーションから、選択したフレーム上のモーションを ActionScript 3.0 コードとしてクリップボードにコピーします。
<code>timeline.createMotionTween()</code>	カレントレイヤー上の選択した各キーフレームの <code>frame.tweenType</code> プロパティを <code>motion</code> に設定し、必要に応じて各フレームの内容を1つのシンボルインスタンスに変換します。
<code>timeline.cutFrames()</code>	カレントレイヤー上のフレームの範囲をタイムラインからカットし、クリップボードに保存します。
<code>timeline.deleteLayer()</code>	レイヤーを削除します。
<code>timeline.expandFolder()</code>	指定した1つまたは複数のフォルダを開く、または閉じます。
<code>timeline.findLayerIndex()</code>	特定の名前を持つレイヤーのインデックスの配列を検索します。
<code>timeline.getFrameProperty()</code>	選択したフレームの指定したプロパティの値を取得します。
<code>timeline.getLayerProperty()</code>	選択したレイヤーの指定したプロパティの値を取得します。
<code>timeline.getSelectedFrames()</code>	配列内の現在選択されているフレームを取得します。
<code>timeline.getSelectedLayers()</code>	現在選択されているレイヤーのゼロから始まるインデックス値を取得します。
<code>timeline.insertBlankKeyframe()</code>	指定したフレームインデックスに空白のキーフレームを挿入します。インデックスが指定されていない場合は、再生ヘッドまたは選択位置を使用して空白のキーフレームを挿入します。
<code>timeline.insertFrames()</code>	指定した数のフレームを指定したフレーム番号に挿入します。
<code>timeline.insertKeyframe()</code>	指定したフレームにキーフレームを挿入します。
<code>timeline.pasteFrames()</code>	フレームの範囲をクリップボードから指定したフレームにペーストします。
<code>timeline.pasteMotion()</code>	<code>timeline.copyMotion()</code> で取得したモーションフレームの範囲をタイムラインにペーストします。
<code>timeline.removeFrames()</code>	フレームを削除します。
<code>timeline.reorderLayer()</code>	最初に指定したレイヤーを 2 番目に指定したレイヤーの前または後に移動します。

メソッド	説明
<code>timeline.reverseFrames()</code>	フレームの範囲を逆にします。
<code>timeline.selectAllFrames()</code>	現在のタイムライン内のすべてのフレームを選択します。
<code>timeline setFrameProperty()</code>	選択したフレームの <code>Frame</code> オブジェクトのプロパティを設定します。
<code>timeline.setLayerProperty()</code>	すべての選択したレイヤー上の指定したプロパティを指定した値に設定します。
<code>timeline.setSelectedFrames()</code>	カレントレイヤー内のフレームの範囲を選択するか、選択フレームをこのメソッドに渡された選択配列に設定します。
<code>timeline.setSelectedLayers()</code>	レイヤーを選択された状態に設定し、さらに指定したレイヤーを現在のレイヤーにします。
<code>timeline.showLayerMasking()</code>	マスクおよびマスクされたレイヤーをロックすることにより、オーサリング中にレイヤーのマスクを表示します。

Timeline オブジェクトのプロパティ一覧

Timeline オブジェクトでは次のメソッドを使用できます。

プロパティ	説明
<code>timeline.currentFrame</code>	現在の再生ヘッドの位置にあるフレームのゼロから始まるインデックス。
<code>timeline.currentLayer</code>	現在アクティブなレイヤーのゼロから始まるインデックス。
<code>timeline.frameCount</code>	読み取り専用。タイムラインの最も長いレイヤー内のフレームの数を表す整数です。
<code>timeline.layerCount</code>	読み取り専用。指定したタイムラインのレイヤーの数を表す整数です。
<code>timeline.layers</code>	読み取り専用。 <code>layer</code> オブジェクトの配列です。
<code>timeline.name</code>	現在のタイムラインの名前を表すストリング。

timeline.addMotionGuide()

使用できるバージョン

Flash MX 2004

シンタックス

```
timeline.addMotionGuide()
```

パラメータ

なし

戻り値

新しく追加されたガイドレイヤーのゼロから始まるインデックスを表す整数。現在のレイヤーのタイプが "Normal" ではない場合、1が返されます。

説明

メソッド。モーションガイドレイヤーを現在のレイヤーの上に追加して、現在のレイヤーを新しく追加したガイドレイヤーに割り当て、現在のレイヤーを "Guided" タイプのレイヤーに変換します。

このメソッドは、"Normal" タイプのレイヤーでのみ使用できます。タイプが "Folder"、"Mask"、"Masked"、"Guide"、または "Guided" のレイヤーには適用されません。

例

次の例では、モーションガイドレイヤーを現在のレイヤーの上に追加し、現在のレイヤーを "Guided" に変換します。

```
fl.getDocumentDOM().getTimeline().addMotionGuide();
```

timeline.addNewLayer()

使用できるバージョン

Flash MX 2004

シンタックス

```
timeline.addNewLayer([name] [, layerType [, bAddAbove]])
```

パラメータ

name 新しいレイヤーの名前を指定するストリング。このパラメータを省略した場合、新しいデフォルトのレイヤー名 ("レイヤー n"。n はレイヤーの総数) が新しいレイヤーに割り当てられます。このパラメータは省略可能です。

layerType 追加するレイヤーのタイプを指定するストリング。このパラメータを省略した場合は、"Normal" タイプのレイヤーが作成されます。このパラメータは省略可能です。指定できる値は、"normal"、"guide"、"guided"、"mask"、"masked"、および "folder" です。

bAddAbove ブール値。true (デフォルト) に設定した場合は、現在のレイヤーの上に新しいレイヤーが追加されます。false の場合は、現在のレイヤーの下にレイヤーが追加されます。このパラメータは省略可能です。

戻り値

新しく追加されたレイヤーのゼロから始まるインデックスの整数値

説明

メソッド。新しいレイヤーをドキュメントに追加し、そのレイヤーを現在のレイヤーにします。

例

次の例では、Flash によって生成されるデフォルト名の新しいレイヤーをタイムラインに追加します。

```
fl.getDocumentDOM().getTimeline().addNewLayer();
```

次の例では、現在のレイヤーの上に新しいフォルダレイヤーを追加し、そのレイヤーに "Folder1" という名前を付けます。

```
fl.getDocumentDOM().getTimeline().addNewLayer("Folder1", "folder", true);
```

timeline.clearFrames()

使用できるバージョン

Flash MX 2004

シンタックス

```
timeline.clearFrames([startFrameIndex [, endFrameIndex]])
```

パラメータ

startFrameIndex クリアするフレームの範囲の始まりを定義するゼロから始まるインデックス。*startFrameIndex* を省略した場合は、現在の選択範囲が使用されます。このパラメータは省略可能です。

endFrameIndex クリアするフレームの範囲の終わりを定義するゼロから始まるインデックス。範囲は、*endFrameIndex* の直前までです。*startFrameIndex* のみを指定した場合、*endFrameIndex* はデフォルトで *startFrameIndex* の値に設定されます。このパラメータは省略可能です。

戻り値

なし

説明

メソッド。カレントレイヤー上のフレームまたはフレームの範囲からすべての内容を削除します。

例

次の例では、フレーム 6 からフレーム 11 の直前までのフレームを削除します (インデックス値はフレーム番号値とは異なることに注意してください)。

```
fl.getDocumentDOM().getTimeline().clearFrames(5, 10);
```

次の例では、フレーム 15 をクリアします。

```
fl.getDocumentDOM().getTimeline().clearFrames(14);
```

timeline.clearKeyframes()

使用できるバージョン

Flash MX 2004

シンタックス

```
timeline.clearKeyframes([startFrameIndex [, endFrameIndex]])
```

パラメータ

startFrameIndex クリアするフレームの範囲の始まりを定義するゼロから始まるインデックス。*startFrameIndex* を省略した場合は、現在の選択範囲が使用されます。このパラメータは省略可能です。

endFrameIndex クリアするフレームの範囲の終わりを定義するゼロから始まるインデックス。範囲は、*endFrameIndex* の直前までです。*startFrameIndex* のみを指定した場合、*endFrameIndex* はデフォルトで *startFrameIndex* の値に設定されます。このパラメータは省略可能です。

戻り値

なし

説明

メソッド。カレントレイヤー上のキーフレームを通常のフレームに変換し、その内容を削除します。

例

次の例では、フレーム 5 からフレーム 10 の直前までのフレームをクリアします (インデックス値はフレーム番号値とは異なることに注意してください)。

```
fl.getDocumentDOM().getTimeline().clearKeyframes(4, 9);
```

次の例では、フレーム 15 にあるキーフレームをクリアし、通常のフレームに変換します。

```
fl.getDocumentDOM().getTimeline().clearKeyframes(14);
```

timeline.convertToBlankKeyframes()

使用できるバージョン

Flash MX 2004

シンタックス

```
timeline.convertToBlankKeyframes([startFrameIndex [, endFrameIndex]])
```

パラメータ

startFrameIndex キーフレームに変換する最初のフレームを指定するゼロから始まるインデックス。*startFrameIndex* を省略した場合は、現在選択されているフレームが変換されます。このパラメータは省略可能です。

endFrameIndex キーフレームへの変換を停止する位置のフレームを指定するゼロから始まるインデックス。変換するフレームの範囲は、*endFrameIndex* の直前までです。*startFrameIndex* のみを指定した場合、*endFrameIndex* はデフォルトで *startFrameIndex* の値に設定されます。このパラメータは省略可能です。

戻り値

なし

説明

メソッド。カレントレイヤー上のフレームを空白のキーフレームに変換します。

例

次の例では、フレーム 2 からフレーム 10 の直前までを空白のキーフレームに変換します (インデックス値はフレーム番号値とは異なることに注意してください)。

```
fl.getDocumentDOM().getTimeline().convertToBlankKeyframes(1, 9);
```

次の例では、フレーム 5 を空白のキーフレームに変換します。

```
fl.getDocumentDOM().getTimeline().convertToBlankKeyframes(4);
```

timeline.convertToKeyframes()

使用できるバージョン

Flash MX 2004

シンタックス

```
timeline.convertToKeyframes([startFrameIndex [, endFrameIndex]])
```

パラメータ

startFrameIndex キーフレームに変換する最初のフレームを指定するゼロから始まるインデックス。*startFrameIndex* を省略した場合は、現在選択されているフレームが変換されます。このパラメータは省略可能です。

endFrameIndex キーフレームへの変換を停止する位置のフレームを指定するゼロから始まるインデックス。変換するフレームの範囲は、*endFrameIndex* の直前までです。*startFrameIndex* のみを指定した場合、*endFrameIndex* はデフォルトで *startFrameIndex* の値に設定されます。このパラメータは省略可能です。

戻り値

なし

説明

メソッド。カレントレイヤー上のフレームの範囲をキーフレームに変換します。フレームが指定されていない場合は、選択範囲を変換します。

例

次の例では、選択したフレームをキーフレームに変換します。

```
fl.getDocumentDOM().getTimeline().convertToKeyframes();
```

次の例では、フレーム 2 からフレーム 10 の直前までのフレームをキーフレームに変換します (インデックス値はフレーム番号値とは異なることに注意してください)。

```
fl.getDocumentDOM().getTimeline().convertToKeyframes(1, 9);
```

次の例では、フレーム 5 をキーフレームに変換します。

```
fl.getDocumentDOM().getTimeline().convertToKeyframes(4);
```


timeline.copyFrames()

使用できるバージョン

Flash MX 2004

シンタックス

```
timeline.copyFrames([startFrameIndex [, endFrameIndex]])
```

パラメータ

startFrameIndex コピーするフレームの範囲の始まりを指定するゼロから始まるインデックス。*startFrameIndex* を省略した場合は、現在の選択範囲が使用されます。このパラメータは省略可能です。

endFrameIndex コピーを停止する位置のフレームを指定するゼロから始まるインデックス。コピーするフレームの範囲は、*endFrameIndex* の直前までです。*startFrameIndex* のみを指定した場合、*endFrameIndex* はデフォルトで *startFrameIndex* の値に設定されます。このパラメータは省略可能です。

戻り値

なし

説明

メソッド。カレントレイヤー上のフレームの範囲をクリップボードにコピーします。

例

次の例では、選択したフレームをクリップボードにコピーします。

```
fl.getDocumentDOM().getTimeline().copyFrames();
```

次の例では、フレーム 2 からフレーム 10 の直前までをクリップボードにコピーします (インデックス値はフレーム番号値とは異なることに注意してください)。

```
fl.getDocumentDOM().getTimeline().copyFrames(1, 9);
```

次の例では、フレーム 5 をクリップボードにコピーします。

```
fl.getDocumentDOM().getTimeline().copyFrames(4);
```

timeline.copyMotion()

使用できるバージョン

Flash CS3 Professional

シンタックス

```
timeline.copyMotion()
```

パラメータ

なし

戻り値

なし

説明

メソッド。モーショントウweenまたはフレームアニメーションから、選択したフレーム上のモーションをコピーします。[timeline.pasteMotion\(\)](#)を使用して、モーションを他のフレームに適用できます。

スクリプトにペーストできるテキスト (コード) としてモーションをコピーする方法については、[timeline.copyMotionAsAS3\(\)](#)を参照してください。

例

次の例では、選択したフレームのモーションをコピーします。

```
fl.getDocumentDOM().getTimeline().copyMotion();
```

関連項目

[timeline.copyMotionAsAS3\(\)](#), [timeline.pasteMotion\(\)](#)

timeline.copyMotionAsAS3()

使用できるバージョン

Flash CS3 Professional

シンタックス

```
timeline.copyMotionAsAS3()
```

パラメータ

なし

戻り値

なし

説明

メソッド。モーショントウイーンまたはフレームアニメーションから、選択したフレーム上のモーションを **ActionScript 3.0** コードとしてクリップボードにコピーします。その後、このコードをスクリプトにペーストできます。

他のフレームに適用できる形式でモーションをコピーする方法については、[`timeline.copyMotion\(\)`](#) を参照してください。

例

次の例では、選択したフレームのモーションを **ActionScript 3.0** コードとしてクリップボードにコピーします。

```
fl.getDocumentDOM().getTimeline().copyMotionAsAS3();
```

関連項目

[`timeline.copyMotion\(\)`](#)

timeline.createMotionTween()

使用できるバージョン

Flash MX 2004

シンタックス

```
timeline.createMotionTween([startFrameIndex [, endFrameIndex]])
```

パラメータ

startFrameIndex モーショントウイーンの作成を開始するフレームを指定するゼロから始まるインデックス。*startFrameIndex* を省略した場合は、現在の選択範囲が使用されます。このパラメータは省略可能です。

endFrameIndex モーショントウイーンを停止する位置のフレームを指定するゼロから始まるインデックス。フレームの範囲は *endFrameIndex* の直前までです。*startFrameIndex* のみを指定した場合、*endFrameIndex* はデフォルトで *startFrameIndex* の値に設定されます。このパラメータは省略可能です。

戻り値

なし

説明

メソッド。カレントレイヤー上の選択した各キーフレームの [`frame.tweenType`](#) プロパティを `motion` に設定し、必要に応じて各フレームの内容を1つのシンボルインスタンスに変換します。このプロパティは、Flash オーサリングツールの [モーショントウイーンを作成] メニューと同じです。

例

次の例では、最初のフレームからフレーム 10 の直前までのシェイプをグラフィックシンボルインスタンスに変換し、`frame.tweenType` を `motion` に設定します (インデックス値はフレーム番号値とは異なることに注意してください)。

```
fl.getDocumentDOM().getTimeline().createMotionTween(0, 9);
```

timeline.currentFrame

使用できるバージョン

Flash MX 2004

シンタックス

```
timeline.currentFrame
```

説明

プロパティ。現在の再生ヘッドの位置にあるフレームのゼロから始まるインデックスです。

例

次の例では、現在のタイムラインの再生ヘッドをフレーム 10 に設定します。インデックス値はフレーム番号値とは異なることに注意してください。

```
fl.getDocumentDOM().getTimeline().currentFrame = 9;
```

次の例では、現在の再生ヘッドの位置の値を `curFrame` 変数に格納します。

```
var curFrame = fl.getDocumentDOM().getTimeline().currentFrame;
```

timeline.currentLayer

使用できるバージョン

Flash MX 2004

シンタックス

```
timeline.currentLayer
```

説明

プロパティ。現在アクティブなレイヤーのゼロから始まるインデックスです。値 0 は最上位のレイヤーを示します。値 1 はその下のレイヤーを示します。

例

次の例では、最上位のレイヤーをアクティブにします。

```
fl.getDocumentDOM().getTimeline().currentLayer = 0;
```

次の例では、現在アクティブなレイヤーのインデックスを `curLayer` 変数に格納します。

```
var curLayer = fl.getDocumentDOM().getTimeline().currentLayer;
```

timeline.cutFrames()

使用できるバージョン

Flash MX 2004

シンタックス

```
timeline.cutFrames([startFrameIndex [, endFrameIndex]])
```

パラメータ

startFrameIndex カットするフレームの範囲の始まりを指定するゼロから始まるインデックス。*startFrameIndex* を省略した場合は、現在の選択範囲が使用されます。このパラメータは省略可能です。

endFrameIndex カットを停止する位置のフレームを指定するゼロから始まるインデックス。フレームの範囲は *endFrameIndex* の直前までです。*startFrameIndex* のみを指定した場合、*endFrameIndex* はデフォルトで *startFrameIndex* の値に設定されます。このパラメータは省略可能です。

戻り値

なし

説明

メソッド。カレントレイヤー上のフレームの範囲をタイムラインからカットし、クリップボードに保存します。

例

次の例では、選択したフレームをタイムラインからカットし、クリップボードに保存します。

```
fl.getDocumentDOM().getTimeline().cutFrames();
```

次の例では、フレーム 2 からフレーム 10 の直前までをタイムラインからカットしてクリップボードに保存します。インデックス値はフレーム番号値とは異なることに注意してください。

```
fl.getDocumentDOM().getTimeline().cutFrames(1, 9);
```

次の例では、フレーム 5 をタイムラインからカットし、クリップボードに保存します。

```
fl.getDocumentDOM().getTimeline().cutFrames(4);
```

timeline.deleteLayer()

使用できるバージョン

Flash MX 2004

シンタックス

```
timeline.deleteLayer([index])
```

パラメータ

index 削除するレイヤーを指定するゼロから始まるインデックス。タイムライン内にレイヤーが1つしかない場合、このメソッドは効果がありません。このパラメータは省略可能です。

戻り値

なし

説明

メソッド。レイヤーを削除します。レイヤーがフォルダの場合は、フォルダ内のすべてのレイヤーが削除されます。レイヤーのインデックスを指定しない場合は、現在選択されているレイヤーが削除されます。

例

次の例では、上から 2 番目のレイヤーを削除します。

```
fl.getDocumentDOM().getTimeline().deleteLayer(1);
```

次の例では、現在選択されているレイヤーを削除します。

```
fl.getDocumentDOM().getTimeline().deleteLayer();
```

timeline.expandFolder()

使用できるバージョン

Flash MX 2004

シンタックス

```
timeline.expandFolder(bExpand [, bRecurseNestedParents [, index]])
```

パラメータ

bExpand ブール値。true に設定されている場合はフォルダを開きます。false の場合はフォルダを閉じます。

bRecurseNestedParents ブール値。true に設定されている場合、*bExpand* パラメータに応じて、指定したフォルダ内のすべてのレイヤーを開く、または閉じます。このパラメータは省略可能です。

index 開く、または閉じるフォルダのゼロから始まるインデックス。すべてのフォルダに適用するには、*-1*を使用します。さらに、*bRecurseNestedParents* を *true* に設定する必要があります。このプロパティは、Flash オーサリングツールの [すべてのフォルダを開く] または [すべてのフォルダを閉じる] メニューと同じです。このパラメータは省略可能です。

戻り値

なし

説明

メソッド。指定した1つまたは複数のフォルダを開く、または閉じます。レイヤーを指定しない場合、このメソッドは現在のレイヤーを処理します。

例

次の例では、このフォルダ構造を使用します。

```
Folder 1 ***
--layer 7
--Folder 2 ****
----Layer 5
```

次の例では、フォルダ1のみを開きます。

```
fl.getDocumentDOM().getTimeline().currentLayer = 1;
fl.getDocumentDOM().getTimeline().expandFolder(true);
```

次の例では、フォルダ1のみを開きます。ここでは、フォルダ1を最後に閉じたときにフォルダ2が閉じていたものとしています。そうでない場合は、フォルダ2は開いた状態で表示されます。

```
fl.getDocumentDOM().getTimeline().expandFolder(true, false, 0);
```

次の例では、現在のタイムライン内のすべてのフォルダを閉じます。

```
fl.getDocumentDOM().getTimeline().expandFolder(false, true, -1);
```

timeline.findLayerIndex()

使用できるバージョン

Flash MX 2004

シンタックス

```
timeline.findLayerIndex(name)
```

パラメータ

name 検索するレイヤーの名前を指定するストリング。

戻り値

指定したレイヤーのインデックス値の配列。指定したレイヤーが見つからない場合は、`undefined` を返します。

説明

メソッド。特定の名前を持つレイヤーのインデックスの配列を検索します。レイヤーインデックスはフラットなので、フォルダはメインのインデックスの一部と見なされます。

例

次の例では、**Layer 7** という名前のすべてのレイヤーのインデックス値を [出力] パネルに表示します。

```
var layerIndex = fl.getDocumentDOM().getTimeline().findLayerIndex("Layer 7");
fl.trace(layerIndex);
```

次の例では、このメソッドから返された値を `timeline.setSelectedLayers()` に戻す方法を示します。

```
var layerIndex = fl.getDocumentDOM().getTimeline().findLayerIndex("Layer 1");
fl.getDocumentDOM().getTimeline().setSelectedLayers(layerIndex[0], true);
```

timeline.frameCount

使用できるバージョン

Flash MX 2004

シンタックス

`timeline.frameCount`

説明

読み取り専用プロパティ。タイムラインの最も長いレイヤー内のフレームの数を表す整数です。

例

次の例では、`countNum` 変数を使用して、現在のドキュメントの最も長いレイヤー内のフレームの数を格納します。

```
var countNum = fl.getDocumentDOM().getTimeline().frameCount;
```


timeline.getFrameProperty()

使用できるバージョン

Flash MX 2004

シンタックス

```
timeline.getFrameProperty(property [, startFrameIndex [, endFrameIndex]])
```

パラメータ

property 値を取得するプロパティの名前を指定するストリング。すべてのプロパティの一覧については、[315 ページの「Frame オブジェクトのプロパティ一覧」](#)を参照してください。

startFrameIndex 値を取得する最初のフレームの番号を指定するゼロから始まるインデックス。*startFrameIndex* を省略した場合は、現在の選択範囲が使用されます。このパラメータは省略可能です。

endFrameIndex 選択するフレームの範囲の終わりを定義するゼロから始まるインデックス。範囲は、*endFrameIndex* の直前までです。*startFrameIndex* のみを指定した場合、*endFrameIndex* はデフォルトで *startFrameIndex* の値に設定されます。このパラメータは省略可能です。

戻り値

指定したプロパティの値、または選択したすべてのフレームのプロパティ値が同じではない場合は undefined

説明

メソッド。選択したフレームの指定したプロパティの値を取得します。

例

次の例では、現在のドキュメント内の最初のフレームの名前を取得し、その名前を [出力] パネルに表示します。

```
fl.getDocumentDOM().getTimeline().currentLayer = 0;
fl.getDocumentDOM().getTimeline().setSelectedFrames(0, 0, true);
var frameName = fl.getDocumentDOM().getTimeline().getFrameProperty("name");
fl.trace(frameName);
```

timeline.getLayerProperty()

使用できるバージョン

Flash MX 2004

シンタックス

```
timeline.getLayerProperty(property)
```

パラメータ

property 値を取得するプロパティの名前を指定するストリング。プロパティの一覧については、[349 ページの「Layer オブジェクトのプロパティ一覧」](#)を参照してください。

戻り値

指定したプロパティの値。Flash は、レイヤーのプロパティを参照してタイプを判断します。すべての指定したレイヤーのプロパティ値が同じではない場合、`undefined` が返されます。

説明

メソッド。選択したレイヤーの指定したプロパティの値を取得します。

例

次の例では、現在のドキュメント内の最上位のレイヤーの名前を取得し、[出力] パネルに表示します。

```
fl.getDocumentDOM().getTimeline().currentLayer = 0;  
var layerName = fl.getDocumentDOM().getTimeline().getLayerProperty("name");  
fl.trace(layerName);
```

timeline.getSelectedFrames()

使用できるバージョン

Flash MX 2004

パラメータ

なし

戻り値

$3n$ の整数を含む配列。 n は、選択した領域の数です。各グループの最初の整数は、レイヤーのインデックスです。2 番目の整数は、選択範囲の最初のフレームです。3 番目の整数は選択範囲の最後のフレームを指定します。最後のフレームは、選択範囲に含まれません。

説明

メソッド。配列内の現在選択されているフレームを取得します。

例

次の例では、最上位のレイヤーが現在のレイヤーになっている状態で、[出力] パネルに 0,5,10,0,20,25 を表示します。

```
var timeline = fl.getDocumentDOM().getTimeline();
timeline.setSelectedFrames(5,10);
timeline.setSelectedFrames(20,25,false);
var theSelectedFrames = timeline.getSelectedFrames();
fl.trace(theSelectedFrames);
```

関連項目

[timeline.setSelectedFrames\(\)](#)

timeline.getSelectedLayers()

使用できるバージョン

Flash MX 2004

パラメータ

なし

戻り値

選択したレイヤーのゼロから始まるインデックス値の配列

説明

メソッド。現在選択されているレイヤーのゼロから始まるインデックス値を取得します。

例

次の例では、[出力] パネルに 1,0 を表示します。

```
fl.getDocumentDOM().getTimeline().setSelectedLayers(0);
fl.getDocumentDOM().getTimeline().setSelectedLayers(1, false);
var layerArray = fl.getDocumentDOM().getTimeline().getSelectedLayers();
fl.trace(layerArray);
```

関連項目

[timeline.setSelectedLayers\(\)](#)

timeline.insertBlankKeyframe()

使用できるバージョン

Flash MX 2004

シンタックス

```
timeline.insertBlankKeyframe([ frameNumIndex])
```

パラメータ

frameNumIndex キーフレームを挿入する位置のフレームを指定するゼロから始まるインデックス。
frameNumIndex を省略した場合は、現在の再生ヘッドのフレーム番号が使用されます。このパラメータは省略可能です。

指定または選択したフレームが通常のフレームの場合、キーフレームはそのフレームの位置に挿入されます。たとえば、1〜10の番号が付いた10のフレームによる範囲があるときにフレーム5を選択した場合、このメソッドにより、フレーム5が空白のキーフレームになり、フレームの範囲の長さは10フレームのままになります。フレーム5が選択され、そのフレームが通常のフレームの隣にあるキーフレームである場合、空白のキーフレームがフレーム6の位置に挿入されます。フレーム5がキーフレームで、その隣にあるフレームが既にキーフレームである場合、フレームは挿入されませんが、再生ヘッドがフレーム6に移動します。

戻り値

なし

説明

メソッド。指定したフレームインデックスに空白のキーフレームを挿入します。インデックスが指定されていない場合は、再生ヘッドまたは選択位置を使用して空白のキーフレームを挿入します。[timeline.insertKeyframe\(\)](#) も参照してください。

例

次の例では、空白のキーフレームをフレーム20に挿入します (インデックス値はフレーム番号値とは異なることに注意してください)。

```
fl.getDocumentDOM().getTimeline().insertBlankKeyframe(19);
```

次の例では、空白のキーフレームを現在選択されているフレーム (またはフレームが選択されていない場合は再生ヘッドの位置) に挿入します。

```
fl.getDocumentDOM().getTimeline().insertBlankKeyframe();
```

timeline.insertFrames()

使用できるバージョン

Flash MX 2004

シンタックス

```
timeline.insertFrames([numFrames [, bAllLayers [, frameNumIndex]])
```

パラメータ

numFrames 挿入するフレームの数を指定する整数。このパラメータを省略した場合、現在のレイヤーの現在の選択位置にフレームが挿入されます。このパラメータは省略可能です。

bAllLayers ブール値。true (デフォルト) に設定した場合、*numFrames* パラメータで指定した数のフレームがすべてのレイヤーに挿入されます。false に設定した場合、現在のレイヤーにフレームが挿入されます。このパラメータは省略可能です。

frameNumIndex 新しいフレームを挿入する位置のフレームを指定するゼロから始まるインデックス。このパラメータは省略可能です。

戻り値

なし

説明

メソッド。指定した数のフレームを指定したインデックスに挿入します。

パラメータが指定されていない場合、このメソッドは次のように機能します。

- 1つ以上のフレームが選択されている場合、選択された数のフレームを現在のレイヤーの最初に選択されたフレームの場所に挿入します。つまり、フレーム 6 から 10 が選択されている場合 (合計で 5 フレーム)、選択されたフレームを含むレイヤー内のフレーム 6 の位置に 5 つのフレームを挿入します。
- フレームが選択されていない場合、すべてのレイヤーの現在のフレームの位置に 1 つのフレームを挿入します。

パラメータが指定されていない場合、メソッドは次のように機能します。

- *numFrames* のみが指定されている場合、指定した数のフレームを現在のレイヤーの現在のフレームの位置に挿入します。
- *numFrames* が指定され、*bAllLayers* が true の場合、指定した数のフレームをすべてのレイヤーの現在のフレームの位置に挿入します。
- 3 つのパラメータがすべて指定されている場合、指定した数のフレームを指定したインデックス (*frameIndex*) の位置に挿入します。*bAllLayers* に対して渡した値によって、フレームを現在のレイヤーのみまたはすべてのレイヤーのどちらかに追加するかが決まります。

指定または選択したフレームが通常のフレームの場合、フレームはそのフレームの位置に挿入されます。たとえば、1～10の番号が付いた10のフレームによる範囲があるときにフレーム5を選択した場合（または *frameIndex* に対して値4を渡した場合）、このメソッドにより、フレーム5の位置にフレームが追加され、フレームの範囲の長さは11フレームになります。フレーム5を選択し、それがキーフレームである場合、その隣にあるフレームもキーフレームであるかどうかに関係なく、フレーム6の位置にフレームが挿入されます。

例

次の例では、現在のレイヤーの現在の選択位置に、選択内容に応じて1つまたは複数のフレームを挿入します。

```
fl.getDocumentDOM().getTimeline().insertFrames();
```

次の例では、すべてのレイヤーの現在のフレームの位置に5つのフレームを挿入します。

```
fl.getDocumentDOM().getTimeline().insertFrames(5);
```

×
中

フレームを含む複数のレイヤーがあり、前のコマンドを使用するときに1つのレイヤーのフレームを選択した場合、選択したレイヤーにのみフレームが挿入されます。フレームが選択されていない複数のレイヤーがある場合、すべてのレイヤーにフレームが挿入されます。

次の例では、現在のレイヤーにのみ3つのフレームを挿入します。

```
fl.getDocumentDOM().getTimeline().insertFrames(3, false);
```

次の例では、すべてのレイヤーに最初のフレームの位置から4つのフレームを挿入します。

```
fl.getDocumentDOM().getTimeline().insertFrames(4, true, 0);
```

timeline.insertKeyframe()

使用できるバージョン

Flash MX 2004

シンタックス

```
timeline.insertKeyframe([ frameNumIndex ])
```

パラメータ

frameNumIndex 現在のレイヤー内のキーフレームを挿入する位置のフレームインデックスを指定するゼロから始まるインデックス。*frameNumIndex* を省略した場合は、現在の再生ヘッドまたは選択したフレームのフレーム番号が使用されます。このパラメータは省略可能です。

戻り値

なし。

説明

メソッド。指定したフレームにキーフレームを挿入します。パラメータを省略した場合、再生ヘッドまたは選択位置を使用してキーフレームが挿入されます。

このメソッドは、挿入されたキーフレームに変換されたフレームの内容が含まれる (つまり空白ではない) 点を除いて `timeline.insertBlankKeyframe()` と同様に機能します。

例

次の例では、再生ヘッドまたは選択位置にキーフレームを挿入します。

```
fl.getDocumentDOM().getTimeline().insertKeyframe();
```

次の例では、キーフレームを 2 番目のレイヤーのフレーム 10 に挿入します (インデックス値はフレーム番号またはレイヤー番号の値とは異なることに注意してください)。

```
fl.getDocumentDOM().getTimeline().currentLayer = 1;  
fl.getDocumentDOM().getTimeline().insertKeyframe(9);
```

timeline.layerCount

使用できるバージョン

Flash MX 2004

シンタックス

```
timeline.layerCount
```

説明

読み取り専用プロパティ。指定したタイムラインのレイヤーの数を表す整数です。

例

次の例では、NumLayer 変数を使用して、カレントシーンのフレームの数を格納します。

```
var NumLayer = fl.getDocumentDOM().getTimeline().layerCount;
```

timeline.layers

使用できるバージョン

Flash MX 2004

シンタックス

```
timeline.layers
```

説明

読み取り専用プロパティ。Layer オブジェクトの配列です。

例

次の例では、`currentLayers` 変数を使用して、現在のドキュメントの **Layer** オブジェクトの配列を格納します。

```
var currentLayers = fl.getDocumentDOM().getTimeline().layers;
```

timeline.name

使用できるバージョン

Flash MX 2004

シンタックス

```
timeline.name
```

説明

プロパティ。現在のタイムラインの名前を指定するストリングです。この名前は、編集されているカレントシーン、スクリーン (スライドまたはフォーム)、またはシンボルの名前です。

例

次の例では、最初のシーンの名前を取得します。

```
var sceneName = fl.getDocumentDOM().timelines[0].name;
```

次の例では、最初のシーンの名前を `FirstScene` に設定します。

```
fl.getDocumentDOM().timelines[0].name = "FirstScene";
```

timeline.pasteFrames()

使用できるバージョン

Flash MX 2004

シンタックス

```
timeline.pasteFrames([startFrameIndex [, endFrameIndex]])
```

パラメータ

startFrameIndex ペーストするフレームの範囲の始まりを指定するゼロから始まるインデックス。*startFrameIndex* を省略した場合は、現在の選択範囲が使用されます。このパラメータは省略可能です。

endFrameIndex フレームのペーストを停止する位置のフレームを指定するゼロから始まるインデックス。このメソッドは、*endFrameIndex* の直前までフレームをペーストします。*startFrameIndex* のみを指定した場合、*endFrameIndex* はデフォルトで *startFrameIndex* の値に設定されます。このパラメータは省略可能です。

戻り値

なし

説明

メソッド。フレームの範囲をクリップボードから指定したフレームにペーストします。

例

次の例では、クリップボードのフレームを現在選択されているフレームまたは再生ヘッドの位置にペーストします。

```
fl.getDocumentDOM().getTimeline().pasteFrames();
```

次の例では、クリップボードのフレームをフレーム 2 からフレーム 10 の直前までにペーストします (インデックス値はフレーム番号値とは異なることに注意してください)。

```
fl.getDocumentDOM().getTimeline().pasteFrames(1, 9);
```

次の例では、クリップボードのフレームをフレーム 5 を開始位置としてペーストします。

```
fl.getDocumentDOM().getTimeline().pasteFrames(4);
```

timeline.pasteMotion()

使用できるバージョン

Flash CS3 Professional

シンタックス

```
timeline.pasteMotion()
```

パラメータ

なし

戻り値

なし

説明

メソッド。[timeline.copyMotion\(\)](#) で取得したモーションフレームの範囲をタイムラインにペーストします。必要に応じて、フレームをペーストするために既存フレームが置き換えられます (右側に移動)。

例

次の例では、現在選択されているフレームの位置をペーストするフレームの右側に移動して、クリップボード上のモーションを現在選択されているフレームまたは再生ヘッドの位置にペーストします。

```
fl.getDocumentDOM().getTimeline().pasteMotion();
```

関連項目

[timeline.copyMotion\(\)](#)

timeline.removeFrames()

使用できるバージョン

Flash MX 2004

シンタックス

```
timeline.removeFrames([startFrameIndex [, endFrameIndex]])
```

パラメータ

startFrameIndex フレームの削除を開始する位置の最初のフレームを指定するゼロから始まるインデックス。*startFrameIndex* を省略した場合、現在の選択範囲が使用されます。選択範囲がない場合は、すべてのレイヤーの現在の再生ヘッドのすべてのフレームが削除されます。このパラメータは省略可能です。

endFrameIndex フレームの削除を停止する位置のフレームを指定するゼロから始まるインデックス。フレームの範囲は *endFrameIndex* の直前までです。*startFrameIndex* のみを指定した場合、*endFrameIndex* はデフォルトで *startFrameIndex* の値に設定されます。このパラメータは省略可能です。

戻り値

なし

説明

メソッド。フレームを削除します。

例

次の例では、カレントシーンの最上位のレイヤーのフレーム 5 からフレーム 10 の直前までを削除します (インデックス値はフレーム番号値とは異なります)。

```
fl.getDocumentDOM().getTimeline().currentLayer = 0;  
fl.getDocumentDOM().getTimeline().removeFrames(4, 9);
```

次の例では、カレントシーンの最上位のレイヤーのフレーム 8 を削除します。

```
fl.getDocumentDOM().getTimeline().currentLayer = 0;  
fl.getDocumentDOM().getTimeline().removeFrames(7);
```

timeline.reorderLayer()

使用できるバージョン

Flash MX 2004

シンタックス

```
timeline.reorderLayer(layerToMove, layerToPutItBy [, bAddBefore])
```

パラメータ

layerToMove 移動するレイヤーを指定するゼロから始まるインデックス。

layerToPutItBy レイヤーの移動先の隣のレイヤーを指定するゼロから始まるインデックス。たとえば、*layerToMove* に対して 1 を指定し、*layerToPutItBy* に対して 0 を指定した場合、2 番目のレイヤーが最初のレイヤーの隣に配置されます。

bAddBefore レイヤーを *layerToPutItBy* の前または後のどちらに移動するかを指定します。*false* を指定した場合、レイヤーは *layerToPutItBy* の後に移動されます。デフォルト値は *true* です。このパラメータは省略可能です。

戻り値

なし

説明

メソッド。最初に指定したフレームを 2 番目に指定したレイヤーの前または後に移動します。

例

次の例では、インデックス 2 のレイヤーを一番上 (インデックス 0 のレイヤーの上) に移動します。

```
fl.getDocumentDOM().getTimeline().reorderLayer(2, 0);
```

次の例では、インデックス 3 のレイヤーをインデックス 5 のレイヤーの後に配置します。

```
fl.getDocumentDOM().getTimeline().reorderLayer(3, 5, false);
```

timeline.reverseFrames()

使用できるバージョン

Flash MX 2004

シンタックス

```
timeline.reverseFrames([startFrameIndex [, endFrameIndex]])
```

パラメータ

startFrameIndex フレームの反転を開始する位置の最初のフレームを指定するゼロから始まるインデックス。*startFrameIndex* を省略した場合は、現在の選択範囲が使用されます。このパラメータは省略可能です。

endFrameIndex フレームの反転を停止する位置のフレームを指定するゼロから始まるインデックス。フレームの範囲は *endFrameIndex* の直前までです。*startFrameIndex* のみを指定した場合、*endFrameIndex* はデフォルトで *startFrameIndex* の値に設定されます。このパラメータは省略可能です。

戻り値

なし

説明

メソッド。フレームの範囲を逆にします。

例

次の例では、現在選択されているフレームの位置を逆にします。

```
fl.getDocumentDOM().getTimeline().reverseFrames();
```

次の例では、フレーム 10 からフレーム 15 の直前までのフレームを逆にします (インデックス値はフレーム番号値とは異なります)。

```
fl.getDocumentDOM().getTimeline().reverseFrames(9, 14);
```

timeline.selectAllFrames()

使用できるバージョン

Flash MX 2004

シンタックス

```
timeline.selectAllFrames()
```

パラメータ

なし

戻り値

なし

説明

メソッド。現在のタイムライン内のすべてのフレームを選択します。

例

次の例では、現在のタイムライン内のすべてのフレームを選択します。

```
fl.getDocumentDOM().getTimeline().selectAllFrames();
```

timeline setFrameProperty()

使用できるバージョン

Flash MX 2004

シンタックス

```
timeline.setFrameProperty(property, value [, startFrameIndex [,  
    endFrameIndex]])
```

パラメータ

property 変更するプロパティの名前を示すストリング。プロパティ名と使用できる値については、[315 ページの「Frame オブジェクトのプロパティ一覧」](#)を参照してください。

×
中

このメソッドを使用して、[frame.duration](#) や [frame.elements](#) などの読み取り専用のプロパティの値を設定することはできません。

value プロパティを設定する値を指定します。適切な値とタイプを調べるには、[315 ページの「Frame オブジェクトのプロパティ一覧」](#)を参照してください。

startFrameIndex 変更する最初のフレーム番号を指定するゼロから始まるインデックス。

startFrameIndex を省略した場合は、現在の選択範囲が使用されます。このパラメータは省略可能です。

endFrameIndex 停止する位置のフレームを指定するゼロから始まるインデックス。フレームの範囲は *endFrameIndex* の直前までです。*startFrameIndex* を指定し、*endFrameIndex* を省略した場合、*endFrameIndex* はデフォルトで *startFrameIndex* の値に設定されます。このパラメータは省略可能です。

戻り値

なし

説明

メソッド。選択したフレームの Frame オブジェクトのプロパティを設定します。

例

次の例では、`ActionScript stop()` コマンドを現在のドキュメントの最上位のレイヤーの最初のフレームに割り当てます。

```
fl.getDocumentDOM().getTimeline().currentLayer = 0;  
fl.getDocumentDOM().getTimeline().setSelectedFrames(0,0,true);  
fl.getDocumentDOM().getTimeline().setFrameProperty("actionScript", "stop();");
```

次の例では、現在のレイヤーのフレーム 2 からフレーム 5 の直前までにモーショントゥイーンを設定します (インデックス値はフレーム番号値とは異なることに注意してください)。

```
var doc = fl.getDocumentDOM();  
doc.getTimeline().setFrameProperty("tweenType","motion",1,4);
```

timeline.setLayerProperty()

使用できるバージョン

Flash MX 2004

シンタックス

```
timeline.setLayerProperty(property, value [, layersToChange])
```

パラメータ

property 設定するプロパティを指定するストリング。プロパティの一覧については、[349 ページの「Layer オブジェクト」](#)を参照してください。

value プロパティを設定する値。Layer オブジェクトのプロパティを設定する場合と同じタイプの値を使用します。

layersToChange 変更するレイヤーを識別するストリング。指定できる値は、"selected"、"all"、および "others" です。このパラメータを省略した場合のデフォルト値は "selected" です。このパラメータは省略可能です。

戻り値

なし

説明

メソッド。すべての選択したレイヤー上の指定したプロパティを指定した値に設定します。

例

次の例では、選択したレイヤーを非表示にします。

```
fl.getDocumentDOM().getTimeline().setLayerProperty("visible", false);
```

次の例では、選択したレイヤーの名前を "selLayer" に設定します。

```
fl.getDocumentDOM().getTimeline().setLayerProperty("name", "selLayer");
```

timeline.setSelectedFrames()

使用できるバージョン

Flash MX 2004

シンタックス

```
timeline.setSelectedFrames(startFrameIndex, endFrameIndex [,  
    bReplaceCurrentSelection])  
timeline.setSelectedFrames(selectionList [, bReplaceCurrentSelection])
```

パラメータ

startFrameIndex 設定する最初のフレームを指定するゼロから始まるインデックス。

endFrameIndex 選択範囲の終わりを指定するゼロから始まるインデックス。*endFrameIndex* は、選択範囲の最後のフレームの後のフレームです。

bReplaceCurrentSelection ブール値。true に設定されている場合、現在選択されているフレームが選択解除されてから指定したフレームが選択されます。デフォルト値は true です。

selectionList timeline.getSelectedFrames() によって返される 3 つの整数の配列。

戻り値

なし

説明

メソッド。カレントレイヤー内のフレームの範囲を選択するか、選択フレームをこのメソッドに渡された選択配列に設定します。

例

次の例では、最上位のレイヤーのフレーム 1 から フレーム 10 の直前までを選択し、同じレイヤーのフレーム 12 からフレーム 15 の直前までを現在の選択範囲に追加する 2 つの方法を示しています (インデックス値はフレーム番号値とは異なることに注意してください)。

```
fl.getDocumentDOM().getTimeline().setSelectedFrames(0, 9);  
fl.getDocumentDOM().getTimeline().setSelectedFrames(11, 14, false);  
fl.getDocumentDOM().getTimeline().setSelectedFrames([0, 0, 9]);  
fl.getDocumentDOM().getTimeline().setSelectedFrames([0, 11, 14], false);
```

次の例では、まず選択したフレームの配列を savedSelectionList 変数に格納してから、コード内で後で配列を使用し、コマンドまたはユーザーの操作によって選択範囲が変更された後にそれらのフレームを再び選択します。

```
var savedSelectionList = fl.getDocumentDOM().getTimeline().getSelectedFrames();  
// 選択範囲を変更する操作を実行します。  
fl.getDocumentDOM().getTimeline().setSelectedFrames(savedSelectionList);
```

関連項目

[timeline.getSelectedFrames\(\)](#)

timeline.setSelectedLayers()

使用できるバージョン

Flash MX 2004

シンタックス

```
timeline.setSelectedLayers(index [, bReplaceCurrentSelection])
```

パラメータ

index 選択するレイヤーのゼロから始まるインデックス。

bReplaceCurrentSelection ブール値。true に設定されている場合は、現在の選択範囲を置き換えます。false の場合は、現在の選択範囲を拡張します。デフォルト値は true です。このパラメータは省略可能です。

戻り値

なし

説明

メソッド。レイヤーを選択された状態に設定し、さらに指定したレイヤーを現在のレイヤーにします。レイヤーを選択すると、レイヤー内のすべてのフレームも選択されます。

例

次の例では、最上位のレイヤーを選択します。

```
fl.getDocumentDOM().getTimeline().setSelectedLayers(0);
```

次の例では、次のレイヤーを選択範囲に追加します。

```
fl.getDocumentDOM().getTimeline().setSelectedLayers(1, false);
```

関連項目

[timeline.getSelectedLayers\(\)](#)

timeline.showLayerMasking()

使用できるバージョン

Flash MX 2004

シンタックス

```
timeline.showLayerMasking([layer])
```


パラメータ

layer オーサリング中にマスクを表示するための、マスクまたはマスクされたレイヤーのゼロから始まるインデックス。このパラメータは省略可能です。

戻り値

なし

説明

メソッド。マスクおよびマスクされたレイヤーをロックすることにより、オーサリング中にレイヤーのマスクを表示します。レイヤーが指定されていない場合、カレントレイヤーを使用します。**Mask** または **Masked** 以外のタイプのレイヤーに対してこのメソッドを使用すると、[出力] パネルにエラーが表示されます。

例

次の例では、オーサリング中に最初のレイヤーのレイヤーマスクが表示されるように指定します。

```
fl.getDocumentDOM().getTimeline().showLayerMasking(0);
```

ToolObj オブジェクト


使用できるバージョン
Flash MX 2004

説明

ToolObj オブジェクトは、ツールパネルの個々のツールを表します。ToolObj オブジェクトにアクセスするには、**Tools オブジェクト**のプロパティ (`tools.toolObjs` 配列または `tools.activeTool`) を使用します。

ToolObj オブジェクトのメソッド一覧

ToolObj オブジェクトでは次のメソッドを使用できます。

 次

次のメソッドは、拡張ツールを作成する場合にのみ使用します。

メソッド	説明
<code>toolObj.enablePControl()</code>	プロパティインスペクタ内の指定したコントロールを有効または無効にします。拡張ツールを作成する場合のみに使用します。
<code>toolObj.setIcon()</code>	Flash のツールパネルでツールアイコンとして使用する PNG ファイルを識別します。
<code>toolObj.setMenuString()</code>	ポップアップメニューにツールの名前として表示するストリングを設定します。
<code>toolObj.setOptionsFile()</code>	XML ファイルをツールに関連付けます。
<code>toolObj.setPI()</code>	ツールがアクティブになったときに使用される特定のプロパティインスペクタを設定します。
<code>toolObj.setToolName()</code>	ツールパネルの設定用にツールに名前を割り当てます。
<code>toolObj.setToolTip()</code>	マウスをツールアイコンの上に置いたときに表示されるツールヒントを設定します。
<code>toolObj.showPControl()</code>	プロパティインスペクタ内のコントロールを表示または非表示にします。
<code>toolObj.showTransformHandles()</code>	拡張ツールの JavaScript ファイルの <code>configureTool()</code> メソッドで呼び出されます。ツールがアクティブなときに自由変形ハンドルを表示することを指定します。

ToolObj オブジェクトのプロパティ一覧

ToolObj オブジェクトでは次のプロパティを使用できます。

プロパティ	説明
<code>toolObj.depth</code>	ツールパネル内のポップアップメニューのツールの深度を指定する整数です。
<code>toolObj.iconID</code>	ツールのリソース ID を指定する整数です。
<code>toolObj.position</code>	読み取り専用。ツールパネル内のツールの位置を指定する整数です。

toolObj.depth

使用できるバージョン

Flash MX 2004

シンタックス

```
toolObj.depth
```

説明

読み取り専用プロパティ。ツールパネル内のポップアップメニューのツールの深度を指定する整数です。このプロパティは、拡張ツールを作成する場合にのみ使用します。

例

次の例では、ツールの深度が1、つまり、ツールパネルのツールの1レベル下であることを指定します。

```
fl.tools.activeTool.depth = 1;
```

toolObj.enablePIControl()

使用できるバージョン

Flash MX 2004

シンタックス

```
toolObj.enablePIControl(control, bEnable)
```

パラメータ

control 有効または無効にするコントロールの名前を指定するストリング。使用できる値は、このツールによって起動されるプロパティインスペクタによって異なります (`toolObj.setPI()` を参照)。

シェイプのプロパティインスペクタには次のコントロールがあります。

stroke	fill
--------	------

テキストのプロパティインスペクタには次のコントロールがあります。

type	font	pointsize
color	bold	italic
direction	alignLeft	alignCenter
alignRight	alignJustify	spacing
position	autoKern	small
rotation	format	lineType
selectable	html	border
deviceFonts	varEdit	options
link	maxChars	target

ムービーのプロパティインスペクタには次のコントロールがあります。

size	publish	background
framerate	player	profile

bEnable コントロールの有効(true)または無効(false)を決定するブール値。

戻り値

なし

説明

メソッド。プロパティインスペクタ内の指定したコントロールを有効または無効にします。拡張ツールを作成する場合のみに使用します。

例

拡張ツールの JavaScript ファイル内の次のコマンドは、そのツールのプロパティインスペクタに線のオプションを表示しないように Flash を設定します。

```
theTool.enablePIControl("stroke", false);
```

toolObj.iconID

使用できるバージョン

Flash MX 2004

シンタックス

`toolObj.iconID`

説明

読み取り専用プロパティ。1の整数値です。このプロパティは、拡張ツールを作成する場合にのみ使用します。iconID 値が1の場合、ツールアイコンの検索が行われません。代わりに、ツールのスクリプトで、ツールパネルに表示するアイコンを指定する必要があります。[toolObj.setIcon\(\)](#) を参照してください。

例

次の例では、値1(現在のツールのアイコン ID) を toolIconID 変数に割り当てます。

```
var toolIconID = fl.tools.activeTool.iconID
```

toolObj.position

使用できるバージョン

Flash MX 2004

シンタックス

`toolObj.position`

説明

読み取り専用プロパティ。ツールパネル内のツールの位置を指定する整数です。このプロパティは、拡張ツールを作成する場合にのみ使用します。

例

ツールの JavaScript ファイルにある `mouseDown()` メソッドの次のコマンドは、ツールパネル内のそのツールの位置を整数として [出力] パネルに表示します。

```
myToolPos = fl.tools.activeTool.position;  
fl.trace(myToolPos);
```

toolObj.setIcon()

使用できるバージョン

Flash MX 2004

シンタックス

```
toolObj.setIcon(file)
```

パラメータ

file アイコンとして使用する PNG ファイルの名前を指定するストリング。PNG ファイルは、JSFL ファイルと同じフォルダに置かれている必要があります。

戻り値

なし

説明

メソッド。ツールパネルでツールアイコンとして使用する PNG ファイルを識別します。このメソッドは、拡張ツールを作成する場合にのみ使用します。

例

次の例では、"PolyStar.png" ファイル内のイメージを PolyStar というツールのアイコンとして使用することを指定します。このコードは、サンプルの "PolyStar.jsfl" ファイルから抜粋したものです。

[19 ページの「サンプルの PolyStar ツール」](#)を参照してください。

```
theTool = fl.tools.activeTool;  
theTool.setIcon("PolyStar.png");
```

toolObj.setMenuString()

使用できるバージョン

Flash MX 2004

シンタックス

```
toolObj.setMenuString(menuStr)
```

パラメータ

menuStr ポップアップメニューにツールの名前として表示する名前を指定するストリング。

戻り値

なし

説明

メソッド。ポップアップメニューにツールの名前として表示するストリングを設定します。このメソッドは、拡張ツールを作成する場合にのみ使用します。

例

次の例では、theTool というツールをポップアップメニューに “PolyStar Tool” という名前で表示することを指定します。このコードは、サンプルの "PolyStar.jsfl" ファイルから抜粋したものです。

[19 ページの「サンプルの PolyStar ツール」](#)を参照してください。

```
theTool = fl.tools.activeTool;  
theTool.setMenuString("PolyStar Tool");
```

toolObj.setOptionsFile()

使用できるバージョン

Flash MX 2004

シンタックス

```
toolObj.setOptionsFile(xmlFile)
```

パラメータ

xmlFile ツールのオプションの説明を含む XML ファイルの名前を指定するストリング。XML ファイルは、JSFL ファイルと同じフォルダに置かれている必要があります。

戻り値

なし

説明

メソッド。XML ファイルをツールに関連付けます。このファイルは、プロパティインスペクタの [オプション] ボタンによって起動されるモーダルパネルに表示されるオプションを指定します。通常は、JSFL ファイル内の configureTool() 関数でこのメソッドを使用します。詳細については、[configureTool\(\)](#) を参照してください。

たとえば、"PolyStar.xml" ファイルでは、Polygon ツールに関連付ける 3 つのオプションを指定しています。

```
<properties>  
  <property name="Style"  
    variable="style"  
    list="polygon,star"  
    defaultValue="0"  
    type="Strings"  />  
  
  <property name="Number of Sides"
```

```
variable="nsides"  
min="3"  
max="32"  
defaultValue="5"  
type="Number" />
```

```
<property name="Star point size"  
variable="pointParam"  
min="0"  
max="1"  
defaultValue=".5"  
type="Double" />
```

```
</properties>
```

例

次の例では、"PolyStar.xml" というファイルを現在アクティブなツールに関連付けることを指定します。このコードは、サンプルの "PolyStar.jsfl" ファイルから抜粋したものです。[19 ページの「サンプルの PolyStar ツール」](#)を参照してください。

```
theTool = fl.tools.activeTool;  
theTool.setOptionsFile("PolyStar.xml");
```

toolObj.setPI()

使用できるバージョン

Flash MX 2004

シンタックス

```
toolObj.setPI(pi)
```

パラメータ

pi このツール用に起動するプロパティインスペクタを指定するストリング。

戻り値

なし

説明

メソッド。ツールがアクティブになったときに使用されるプロパティインスペクタを指定します。このメソッドは、拡張ツールを作成する場合にのみ使用します。有効な値は、"shape" (デフォルト)、"text"、および "movie" です。

例

次の例では、ツールがアクティブになったときにシェイプのプロパティインスペクタを使用することを指定します。このコードは、サンプルの "PolyStar.jsfl" ファイルから抜粋したものです。

[19 ページの「サンプルの PolyStar ツール」](#)を参照してください。

```
theTool = fl.tools.activeTool;  
theTool.setPI("shape");
```

toolObj.setToolName()

使用できるバージョン

Flash MX 2004

シンタックス

```
toolObj.setToolName(name)
```

パラメータ

name ツールの名前を指定するストリング。

戻り値

なし

説明

メソッド。ツールパネルの設定用にツールに名前を割り当てます。このメソッドは、拡張ツールを作成する場合にのみ使用します。この名前は、Flash がツールパネルを構築するために読み込む XML レイアウトファイルでのみ使用されます。この名前は Flash のユーザーインターフェイスには表示されません。

例

次の例では、“polystar” という名前を theTool というツールに割り当てます。このコードは、サンプルの "PolyStar.jsfl" ファイルから抜粋したものです。[19 ページの「サンプルの PolyStar ツール」](#)を参照してください。

```
theTool = fl.tools.activeTool;  
theTool.setToolName("polystar");
```

toolObj.setToolTip()

使用できるバージョン

Flash MX 2004

シンタックス

```
toolObj.setToolTip(toolTip)
```

パラメータ

toolTip ツールで使用するツールヒントを指定するストリング。

戻り値

なし

説明

メソッド。マウスをツールアイコンの上に置いたときに表示されるツールヒントを設定します。このメソッドは、拡張ツールを作成する場合にのみ使用します。

例

次の例では、ツールのツールヒントが“PolyStar Tool”であることを指定します。このコードは、サンプルの“PolyStar.jsfl”ファイルから抜粋したものです。[19 ページ](#)の「[サンプルの PolyStar ツール](#)」を参照してください。

```
theTool = fl.tools.activeTool;  
theTool.setToolTip("PolyStar Tool");
```

toolObj.showPIControl()

使用できるバージョン

Flash MX 2004

シンタックス

```
toolObj.showPIControl(control, bShow)
```

パラメータ

control 表示または非表示にするコントロールの名前を指定するストリング。このメソッドは、拡張ツールを作成する場合にのみ使用します。使用できる値は、このツールによって起動されるプロパティンスペクタによって異なります。[toolObj.setPI\(\)](#)を参照してください。

シェイブのプロパティンスペクタには次のコントロールがあります。

stroke

fill

テキストのプロパティインスペクタには次のコントロールがあります。

type	font	pointsize
color	bold	italic
direction	alignLeft	alignCenter
alignRight	alignJustify	spacing
position	autoKern	small
rotation	format	lineType
selectable	html	border
deviceFonts	varEdit	options
link	maxChars	target

ムービーのプロパティインスペクタには次のコントロールがあります。

size	publish	background
framerate	player	profile

bShow 指定したコントロールを表示するか非表示にするかを決定するブール値。*true* の場合はコントロールが表示され、*false* の場合はコントロールが非表示になります。

戻り値

なし

説明

メソッド。プロパティインスペクタ内のコントロールを表示または非表示にします。このメソッドは、拡張ツールを作成する場合にのみ使用します。

例

拡張ツールの JavaScript ファイル内の次のコマンドは、そのツールのプロパティインスペクタに塗りのオプションを表示しないように **Flash** を設定します。

```
fl.tools.activeTool.showPIColorControl("fill", false);
```

toolObj.showTransformHandles()

使用できるバージョン

Flash MX 2004

シンタックス

```
toolObj.showTransformHandles(bShow)
```

パラメータ

bShow 現在のツールの自由変形ハンドルを表示するか非表示にするかを決定するブール値。`true` の場合はハンドルが表示され、`false` の場合はハンドルが非表示になります。

戻り値

なし

説明

メソッド。拡張可能なツールの **JavaScript** ファイルの `configureTool()` メソッドで呼び出されます。ツールがアクティブなときに自由変形ハンドルを表示することを指定します。このメソッドは、拡張ツールを作成する場合にのみ使用します。

例

詳細については、[configureTool\(\)](#) を参照してください。

Tools オブジェクト

使用できるバージョン

Flash MX 2004

説明

Tools オブジェクトは、flash オブジェクト ([fl.tools](#)) からアクセスできます。[tools.toolObj](#) プロパティは、ToolObj オブジェクトの配列を含んでおり、[tools.activeTool](#) プロパティは、現在アクティブなツールの ToolObj オブジェクトを返します。[554 ページの「ToolObj オブジェクト」](#) および [21 ページの「拡張ツール」](#) も参照してください。



次のメソッドとプロパティは、拡張可能なツールを作成する場合にのみ使用します。

Tools オブジェクトのメソッド一覧

Tools オブジェクトでは次のメソッドを使用できます。

メソッド	説明
tools.constrainPoint()	2つのポイントを取得し、調整または制限された新しいポイントを返します。
tools.getKeyDown()	最後に押されたキーを返します。
tools.setCursor()	ポインタを指定した外観に設定します。
tools.snapPoint()	入力としてポイントを取得し、最も近い図形オブジェクトに調整または吸着される新しいポイントを返します。

Tools オブジェクトのプロパティ一覧

Tools オブジェクトでは次のプロパティを使用できます。

プロパティ	説明
tools.activeTool	読み取り専用。現在アクティブなツールの ToolObj オブジェクト を返します。
tools.altIsDown	読み取り専用。Alt キーが押されているかどうかを識別するブール値です。
tools.ctrlIsDown	読み取り専用。Ctrl キーが押されているかどうかを識別するブール値です。
tools.mouseIsDown	読み取り専用。左マウスボタンが現在押されているかどうかを識別するブール値です。
tools.penDownLoc	読み取り専用。ステージ上で最後にマウスの down イベントがあったときの位置を表すポイントです。

プロパティ	説明
<code>tools.penLoc</code>	読み取り専用。マウスの現在の場所を表すポイントです。
<code>tools.shiftIsDown</code>	読み取り専用。 Shift キーが押されているかどうかを識別するブール値です。
<code>tools.toolObjjs</code>	読み取り専用。 ToolObj オブジェクトの配列です。

tools.activeTool

使用できるバージョン

Flash MX 2004

シンタックス

`tools.activeTool`

説明

読み取り専用プロパティ。現在アクティブなツールの **ToolObj** オブジェクトを返します。

例

次の例では、現在アクティブなツールを表すオブジェクトを `theTool` 変数に格納します。

```
var theTool = fl.tools.activeTool;
```

tools.altIsDown

使用できるバージョン

Flash MX 2004

シンタックス

`tools.altIsDown`

説明

読み取り専用プロパティ。**Alt** キーが押されているかどうかを識別するブール値です。**Alt** キーが押されている場合は値が `true` になります。それ以外の場合は `false` になります。

例

次の例では、**Alt** キーが押されているかどうかを調べます。

```
var isAltDown = fl.tools.altIsDown;
```

tools.constrainPoint()

使用できるバージョン

Flash MX 2004

シンタックス

```
tools.constrainPoint(pt1, pt2)
```

パラメータ

pt1 および *pt2* 最初のクリックのポイントおよびドラッグ先のポイントを指定するポイントです。

戻り値

調整または制限された新しいポイント。

説明

メソッド。2つのポイントを取得し、調整または制限された新しいポイントを返します。コマンドの実行時に Shift キーが押された場合、返されるポイントは 45° の制限に従うように制限されるか (矢印の付いた線などで有効)、オブジェクトの縦横比を維持するように制限されます (矩形ツールで完全な正方形を描画する場合など)。

例

次の例では、制限されたポイントを返します。

```
pt2 = fl.tools.constrainPoint(pt1, tempPt);
```

tools.ctrlIsDown

使用できるバージョン

Flash MX 2004

シンタックス

```
tools.ctrlIsDown
```

説明

読み取り専用プロパティ。ブール値です。Ctrl キーが押されている場合は true になり、それ以外の場合は false になります。

例

次の例では、Ctrl キーが押されているかどうかを調べます。

```
var isCtrlDown = fl.tools.ctrlIsDown;
```

tools.getKeyDown()

使用できるバージョン

Flash MX 2004

シンタックス

```
tools.getKeyDown()
```

パラメータ

なし

戻り値

キーの整数値

説明

メソッド。最後に押されたキーを返します。

例

次の例では、最後に押されたキーの整数値を表示します。

```
var theKey = fl.tools.getKeyDown();  
fl.trace(theKey);
```

tools.mouseIsDown

使用できるバージョン

Flash MX 2004

シンタックス

```
tools.mouseIsDown
```

説明

読み取り専用プロパティ。ブール値です。左マウスボタンが現在押されている場合は true になり、それ以外の場合は false になります。

例

次の例では、左マウスボタンが押されているかどうかを調べます。

```
var isMouseDown = fl.tools.mouseIsDown;
```


tools.penDownLoc

使用できるバージョン

Flash MX 2004

シンタックス

`tools.penDownLoc`

説明

読み取り専用プロパティ。ステージ上で最後にマウスの `down` イベントがあったときの位置を表すポイントです。`tools.penDownLoc` プロパティは、`x` と `y` の2つのプロパティで構成され、マウスポインタの位置の `x,y` 座標に対応します。

例

次の例では、ステージ上で最後にマウスの `down` イベントがあったときの位置を調べ、`x` と `y` の値を [出力] パネルに表示します。

```
var pt1 = fl.tools.penDownLoc;  
fl.trace("x,y location of last mouseDown event was " + pt1.x + ", " + pt1.y)
```

関連項目

[tools.penLoc](#)

tools.penLoc

使用できるバージョン

Flash MX 2004

シンタックス

`tools.penLoc`

説明

読み取り専用プロパティ。マウスポインタの現在の位置を表すポイントです。`tools.penLoc` プロパティは、`x` と `y` の2つのプロパティで構成され、マウスポインタの位置の `x,y` 座標に対応します。

例

次の例では、マウスの現在の位置を調べます。

```
var tempPt = fl.tools.penLoc;
```

関連項目

[tools.penDownLoc](#)

tools.setCursor()

使用できるバージョン

Flash MX 2004

シンタックス

```
tools.setCursor(cursor)
```

パラメータ

cursor 次の一覧で説明するようにポインタの外観を定義する整数

- 0 プラスカーソル (+)
- 1 黒の矢印
- 2 白の矢印
- 3 4方向の矢印
- 4 2方向の水平の矢印
- 5 2方向の垂直の矢印
- 6 X
- 7 ハンドカーソル

戻り値

なし

説明

メソッド。ポインタを指定した外観に設定します。

例

次の例では、ポインタを黒の矢印に設定します。

```
fl.tools.setCursor(1);
```

tools.shiftIsDown

使用できるバージョン

Flash MX 2004

シンタックス

```
tools.shiftIsDown
```

説明

読み取り専用プロパティ。ブール値です。Shift キーが押されている場合は true になり、それ以外の場合は false になります。

例

次の例では、Shift キーが押されているかどうかを調べます。

```
var isShiftDown = fl.tools.shiftIsDown;
```

tools.snapPoint()

使用できるバージョン

Flash MX 2004

シンタックス

```
tools.snapPoint(pt)
```

パラメータ

pt 吸着ポイントを返すポイントの位置を指定します。

戻り値

最も近い図形オブジェクトに調整または吸着される新しいポイント。

説明

メソッド。入力としてポイントを取得し、最も近い図形オブジェクトに調整または吸着される新しいポイントを返します。Flash ユーザーインターフェイスの [表示] メニューで吸着が無効になっている場合、返されるポイントは元のポイントになります。

例

次の例では、最も近い図形オブジェクトに吸着される新しいポイントを返します。

```
var theSnapPoint = fl.tools.snapPoint(pt1);
```

tools.toolObjs

使用できるバージョン

Flash MX 2004

シンタックス

```
tools.toolObjs
```

説明

読み取り専用プロパティ。ToolObj オブジェクト ([ToolObj オブジェクト](#) を参照) の配列です。

Vertex オブジェクト

使用できるバージョン

Flash MX 2004

説明

Vertex オブジェクトは、座標データを保持するシェイプデータ構造の一部です。

Vertex オブジェクトのメソッド一覧

Vertex オブジェクトでは次のメソッドを使用できます。

メソッド	説明
<code>vertex.getHalfEdge()</code>	この頂点を共有する HalfEdge オブジェクト を取得します。
<code>vertex.setLocation()</code>	頂点の場所を設定します。

Vertex オブジェクトのプロパティ一覧

Vertex オブジェクトでは次のプロパティを使用できます。

プロパティ	説明
<code>vertex.x</code>	読み取り専用。頂点のピクセル単位の x 座標です。
<code>vertex.y</code>	読み取り専用。頂点のピクセル単位の y 座標です。

vertex.getHalfEdge()

使用できるバージョン

Flash MX 2004

シンタックス

`vertex.getHalfEdge()`

パラメータ

なし

戻り値

[HalfEdge オブジェクト](#)

説明

メソッド。この頂点を共有する [HalfEdge オブジェクト](#) を取得します。

例

次の例では、同じ頂点を共有する他のハーフエッジを取得する方法を示します。

```
var shape = fl.getDocumentDOM().selection[0];
var hEdge = shape.edges[0].getHalfEdge(0);
var theVertex = hEdge.getVertex();
var someHEdge = theVertex.getHalfEdge(); // 同じハーフエッジとは限りません。
var theSameVertex = someHEdge.getVertex();
fl.trace('the same vertex: ' + theSameVertex);
```

vertex.setLocation()

使用できるバージョン

Flash MX 2004

シンタックス

```
vertex.setLocation(x, y)
```

パラメータ

x 頂点を配置する場所の *x* 座標をピクセル単位で指定する浮動小数値。

y 頂点を配置する場所の *y* 座標をピクセル単位で指定する浮動小数値。

戻り値

なし

説明

メソッド。頂点の場所を設定します。このメソッドを使用する前に、[shape.beginEdit\(\)](#) を呼び出す必要があります。

例

次の例では、頂点を原点に設定します。

```
var shape = fl.getDocumentDOM().selection[0];
shape.beginEdit();
var hEdge = shape.edges[0].getHalfEdge(0);
var vertex = hEdge.getVertex();
var someHEdge = vertex.getHalfEdge();
var vertex = someHEdge.getVertex();
// 頂点を原点に移動します。
vertex.setLocation(0.0, 0.0);
shape.endEdit();
```

vertex.x

使用できるバージョン

Flash MX 2004

シンタックス

vertex.x

説明

読み取り専用プロパティ。頂点のピクセル単位の x 座標です。

例

次の例では、頂点の x 値と y 値の位置を [出力] パネルに表示します。

```
var shape = fl.getDocumentDOM().selection[0];
var hEdge = shape.edges[0].getHalfEdge(0);
var vertex = hEdge.getVertex();

fl.trace('x location of vertex is: ' + vertex.x);
fl.trace('y location of vertex is: ' + vertex.y);
```

vertex.y

使用できるバージョン

Flash MX 2004

シンタックス

vertex.y

説明

読み取り専用プロパティ。頂点のピクセル単位の y 座標です。

例

[vertex.x](#) を参照してください。

XMLUI オブジェクト

使用できるバージョン

Flash MX 2004

説明

Flash 8 は、XUL (XML User Interface Language) で作成されたカスタムダイアログボックスをサポートします。コマンドやビヘイビアなどの複数の Flash の機能で XMLUI (XML User Interface) ダイアログボックスを使用し、拡張性を使用して構築する機能のユーザーインターフェイスを提供することができます。XMLUI オブジェクトは、XMLUI ダイアログボックスのプロパティを取得、設定することができます。受け取ることもキャンセルすることもできます。XMLUI メソッドは、ボタンの oncommand ハンドラなどのコールバックで使用できます。

"dialog.xml" ファイルを作成し、`document.xmlPanel()` メソッドを使用して、JavaScript API からそのファイルを呼び出すことができます。現在の XMLUI ダイアログボックスを表すオブジェクトを取得するには、`fl.xmlui` を使用します。

XMLUI オブジェクトのメソッド一覧

XMLUI オブジェクトでは次のメソッドを使用できます。

メソッド	説明
<code>xmlui.accept()</code>	現在の XMLUI ダイアログボックスを承認状態で閉じます。
<code>xmlui.cancel()</code>	現在の XMLUI ダイアログボックスをキャンセル状態で閉じます。
<code>xmlui.get()</code>	現在の XMLUI ダイアログボックスの指定したプロパティの値を取得します。
<code>xmlui.getControlItemElement()</code>	指定されたコントロールの現在のコントロールアイテムを返します。
<code>xmlui.setEnabled()</code>	コントロールが有効か無効 (グレー表示) かを示すブール値を返します。
<code>xmlui.getVisible()</code>	コントロールが表示か非表示かを示すブール値を返します。
<code>xmlui.set()</code>	現在の XMLUI ダイアログボックスの指定したプロパティの値を変更します。
<code>xmlui.setControlItemElement()</code>	現在のアイテムのラベルと値を設定します。
<code>xmlui.setControlItemElements()</code>	現在のアイテムのラベルと値のペアを複数設定します。

メソッド	説明
<code>xmlui.setEnabled()</code>	コントロールを有効または無効 (グレー表示) にします。
<code>xmlui.setVisible()</code>	コントロールを表示または非表示にします。

xmlui.accept()

使用できるバージョン

Flash MX 2004

シンタックス

`xmlui.accept()`

パラメータ

なし

戻り値

なし

説明

メソッド。現在の XMLUI ダイアログボックスを承認状態で閉じます。これは、ユーザーが [OK] ボタンをクリックするのと同じです。

関連項目

`fl.xmlui.document.xmlPanel()`, `xmlui.cancel()`

xmlui.cancel()

使用できるバージョン

Flash MX 2004

シンタックス

`xmlui.cancel()`

パラメータ

なし

戻り値

なし

説明

メソッド。現在の XMLUI ダイアログボックスをキャンセル状態で閉じます。これは、ユーザーが [キャンセル] ボタンをクリックするのと同じです。

関連項目

[fl.xmlui, document.xmlPanel\(\), xmlui.accept\(\)](#)

xmlui.get()

使用できるバージョン

Flash MX 2004

シンタックス

`xmlui.get(controlPropertyName)`

パラメータ

controlPropertyName 値を取得する XMLUI プロパティの名前を指定するストリング。

戻り値

指定したプロパティの値を表すストリング。true または false のブール値が予想される場合は、"true" または "false" のストリングを返します。

説明

メソッド。現在の XMLUI ダイアログボックスの指定したプロパティの値を取得します。

例

次の例では、"URL" というプロパティの値を返します。

```
fl.xmlui.get("URL");
```

関連項目

[fl.xmlui, document.xmlPanel\(\), xmlui.getControlItemElement\(\), xmlui.set\(\)](#)

xmlui.getControlItemElement()

使用できるバージョン

Flash 8

シンタックス

```
xmlui.getControlItemElement(controlPropertyName)
```

パラメータ

controlPropertyName コントロールアイテムエレメントを取得するプロパティを指定するストリング

戻り値

controlPropertyName で指定されたコントロールの現在のコントロールアイテムを表すオブジェクト

説明

メソッド。 *controlPropertyName* で指定されるコントロールの `ListBox` コントロールまたは `ComboBox` コントロールで選択されている行のラベルと値を返します。

例

次の例では、 `myListBox` コントロールの現在選択されている行のラベルと値を返します。

```
var elem = new Object();
elem = fl.xmlui.getControlItemElement("myListBox");
fl.trace("label = " + elem.label + " value = " + elem.value);
```

関連項目

[fl.xmlui.document.xmlPanel\(\)](#), [xmlui.get\(\)](#), [xmlui.setControlItemElement\(\)](#), [xmlui.setControlItemElements\(\)](#)

xmlui.setEnabled()

使用できるバージョン

Flash 8

シンタックス

```
xmlui.setEnabled(controlID)
```

パラメータ

controlID 状態を取得するコントロールの ID 属性を指定するストリング

戻り値

ブール値。コントロールが有効な場合は `true` を返し、それ以外の場合は `false` を返します。

説明

メソッド。コントロールが有効か無効 (グレー表示) かを示すブール値を返します。

例

次の例では、ID 属性 `myListBox` を持つコントロールが有効かどうかを示す値を返します。

```
var isEnabled = fl.xmlui.setEnabled("myListBox");
fl.trace(isEnabled);
```

関連項目

[fl.xmlui, document.xmlPanel\(\), xmlui.setEnabled\(\)](#)

xmlui.getVisible()

使用できるバージョン

Flash 8

シンタックス

```
xmlui.getVisible(controlID)
```

パラメータ

controlID 表示状態を取得するコントロールの ID 属性を指定するストリング。

戻り値

ブール値。コントロールが表示されている場合は `true` を返します。コントロールが非表示の場合は `false` を返します。

説明

メソッド。コントロールが表示か非表示かを示すブール値を返します。

例

次の例では、ID 属性 `myListBox` を持つコントロールが表示されているかどうかを示す値を返します。

```
var isVisible = fl.xmlui.getVisible("myListBox");
fl.trace(isVisible);
```

関連項目

[xmlui.setVisible\(\)](#)

xmlui.set()

使用できるバージョン

Flash MX 2004

シンタックス

```
xmlui.set(controlPropertyName, value)
```

パラメータ

controlPropertyName 変更する XMLUI プロパティの名前を指定するストリング

value XMLUI プロパティを設定する値を指定するストリング

戻り値

なし

説明

メソッド。現在の XMLUI ダイアログボックスの指定したプロパティの値を変更します。

例

次の例では、URL というプロパティの値を "www.adobe.com" に設定します。

```
fl.xmlui.set("URL", "www.adobe.com");
```

関連項目

[fl.xmlui](#), [document.xmlPanel\(\)](#), [xmlui.get\(\)](#), [xmlui.setControlItemElement\(\)](#), [xmlui.setControlItemElements\(\)](#)

xmlui.setControlItemElement()

使用できるバージョン

Flash 8

シンタックス

```
xmlui.setControlItemElement(controlPropertyName, elementItem)
```

パラメータ

controlPropertyName 設定するコントロールアイテムエレメントを指定するストリング。

elementItem label というストリングプロパティと value という省略可能なストリングプロパティを持つ JavaScript オブジェクト。value プロパティが存在しない場合は、プロパティが作成されて、label と同じ値が割り当てられます。

戻り値

なし

説明

メソッド。 *controlPropertyName* で指定された **ListBox** コントロールまたは **ComboBox** コントロールで現在選択されている行のラベルと値を設定します。

例

次の例では、 "PhoneNumber" というコントロールプロパティの現在のアイテムのラベルと値を設定します。

```
var elem = new Object();
elem.label = "Fax";
elem.value = "707-555-5555";
fl.xmlui.setControlItemElement("PhoneNumber",elem);
```

関連項目

[fl.xmlui.document.xmlPanel\(\)](#), [xmlui.getControlItemElement\(\)](#), [xmlui.set\(\)](#), [xmlui.setControlItemElements\(\)](#)

xmlui.setControlItemElements()

使用できるバージョン

Flash 8

シンタックス

```
xmlui.setControlItemElements(controlID, elementItemArray)
```

パラメータ

controlID 設定するコントロールの ID 属性を指定するストリング。

elementItemArray JavaScript オブジェクトの配列。各オブジェクトは *label* というストリングプロパティと *value* という省略可能なストリングプロパティを持ちます。 *value* プロパティが存在しない場合は、プロパティが作成されて、 *label* と同じ値が割り当てられます。

戻り値

なし

説明

メソッド。 *controlID* で指定された **ListBox** コントロールまたは **ComboBox** コントロールの値をクリアし、そのリストアイテムまたはメニューアイテムを *elementItemArray* で指定された *label*, *value* ペアで置き換えます。

例

次の例では、ID 属性が myControlID のコントロール内のアイテムのラベルと値を、指定された label, value ペアに設定します。

```
var nameArray = new Array("January", "February", "March");
var monthArray = new Array();
for (i=0;i<nameArray.length;i++){
    elem = new Object();
    elem.label = nameArray[i];
    elem.value = i;
    monthArray[i] = elem;
}
fl.xmlui.setControlItemElements("myControlID", monthArray);
```

関連項目

[xmlui.getControlItemElement\(\)](#), [xmlui.set\(\)](#), [xmlui.setControlItemElement\(\)](#)

xmlui.setEnabled()

使用できるバージョン

Flash 8

シンタックス

```
xmlui.setEnabled(controlID, enable)
```

パラメータ

controlID 有効または無効にするコントロールの ID 属性を指定するストリング。

enable ブール値。コントロールを有効にする場合は true、コントロールを無効 (グレー表示) にする場合は false。

戻り値

なし

説明

メソッド。コントロールを有効または無効 (グレー表示) にします。

例

次の例では、ID 属性 myControl を持つコントロールをグレー表示にします。

```
fl.xmlui.setEnabled("myControl", false);
```

関連項目

[xmlui.setEnabled\(\)](#)

xmlui.setVisible()

使用できるバージョン

Flash 8

シンタックス

```
xmlui.setVisible(controlID, visible)
```

パラメータ

controlID 表示または非表示にするコントロールの ID 属性を指定するストリング。

visible ブール値。コントロールを表示する場合は true、コントロールを非表示にする場合は false。

戻り値

なし

説明

メソッド。コントロールを表示または非表示にします。

例

次の例では、ID 属性 myControl を持つコントロールを非表示にします。

```
fl.xmlui.setVisible("myControl", false);
```

関連項目

[xmlui.getVisible\(\)](#)

VideolItem オブジェクト

継承 [Item オブジェクト](#) > VideolItem オブジェクト

使用できるバージョン

Flash MX 2004

説明

VideolItem オブジェクトは、[Item オブジェクト](#)のサブクラスです。

VideolItem オブジェクトのプロパティ一覧

VideolItem オブジェクトでは、[Item オブジェクト](#)のプロパティの他に、以下のプロパティを使用できます。

プロパティ	説明
<code>videolItem.sourceFilePath</code>	読み取り専用。ビデオアイテムへのパスを指定するストリングです。
<code>videolItem.videoType</code>	読み取り専用。アイテムが表すビデオの種類を指定するストリングです。

videolItem.sourceFilePath

使用できるバージョン

Flash 8

シンタックス

`videolItem.sourceFilePath`

説明

読み取り専用プロパティ。ビデオアイテムへのパスを指定するストリングです。file:/// URI で表します。

例

次の例では、ライブラリ内で種類が "video" のアイテムの名前とソースファイルパスを表示します。

```
for (idx in fl.getDocumentDOM().library.items) {
    if (fl.getDocumentDOM().library.items[idx].itemType == "video") {
        var myItem = fl.getDocumentDOM().library.items[idx];
        fl.trace(myItem.name + " source is " + myItem.sourceFilePath);
    }
}
```


関連項目

[library.items](#)

videoItem.videoType

使用できるバージョン

Flash 8

シンタックス

`videoItem.videoType`

説明

読み取り専用プロパティ。アイテムが表すビデオの種類を指定するストリングです。指定できる値は、`"embedded video"`、`"linked video"`、および `"video"` です。

例

次の例では、ライブラリ内で種類が `"video"` のアイテムの名前と種類を表示します。

```
for (idx in fl.getDocumentDOM().library.items) {
    if (fl.getDocumentDOM().library.items[idx].itemType == "video") {
        var myItem = fl.getDocumentDOM().library.items[idx];
        fl.trace(myItem.name + " is " + myItem.videoType);
    }
}
```

関連項目

[library.items](#)

C レベルの拡張性

C レベルの拡張性メカニズムを使用すると、JavaScript とカスタム C コードの組み合わせを使用して Adobe Flash CS3 Professional の拡張ファイルを実装することができます。C を使用して関数を定義し、それらをダイナミックリンクライブラリ (DLL) または共有ライブラリにまとめて、そのライブラリを適切なディレクトリに保存します。その後で、Adobe Flash JavaScript API を使用して JavaScript からその関数を呼び出します。

C レベルの拡張性メカニズムは Macromedia Flash MX 2004 で実装され、Flash 8 では変更されませんでした。Flash CS3 での変更点は、サンプルの ZIP ファイルまたは SIT ファイルに含まれる "mm_jsapi.h" ファイルのみです (593 ページの「[サンプルの DLL 実装](#)」を参照してください)。

たとえば、JavaScript よりも効率的に多くの計算を実行する関数を定義してパフォーマンスを向上させたり、より高度なツールやエフェクトを作成する必要がある場合などがあります。

この拡張性メカニズムは、Adobe Dreamweaver CS3 API のサブセットです。この API に関する詳しい知識がある場合は、C レベルの拡張性メカニズムの API 内の関数を知っているかもしれません。ただし、この API は、次の点で Dreamweaver API とは異なっています。

- この API には、Dreamweaver API のすべてのコマンドが含まれているわけではありません。
- Dreamweaver API での型 `wchar_t` および `char` のすべての宣言は、この API では、ストリングが渡されるときに Unicode をサポートするために `unsigned short` 宣言として実装されます。
- この API の `JSVal JS_BytesToValue()` 関数は、Dreamweaver API には含まれていません。
- DLL または共有ライブラリファイルの格納場所が異なります (588 ページの「[C 関数の統合](#)」を参照してください)。

C 関数の統合

C レベルの拡張性メカニズムを使用すると、JavaScript と C コードの組み合わせを使用して Flash の拡張ファイルを実装することができます。次の手順では、この機能の実装方法について説明します。

1. C または C++ 言語を使用して関数を定義します。
2. それらを DLL ファイル (Windows) または共有ライブラリ (Macintosh) に実装します。
3. DLL ファイルまたはライブラリを次の適切な場所に保存します。
 - Windows 2000 または Windows XP :
ブートドライブ ¥Documents and Settings¥< ユーザー >¥Local Settings¥
Application Data¥Adobe¥Flash CS3¥< 言語 >¥Configuration¥External Libraries
 - Macintosh OS X :
Macintosh HD/Users/< ユーザー名 >/Library/Application Support/Adobe/Flash CS3/
< 言語 >/Configuration/External Libraries
4. 関数を呼び出す JSFL ファイルを作成します。
5. Flash オーサリング環境の [コマンド] メニューから JSFL ファイルを実行します。

詳細については、[593 ページの「サンプルの DLL 実装」](#)を参照してください。

C レベルの拡張性と JavaScript インタプリタ

DLL または共有ライブラリ内の C コードは、次の 3 つの異なる時点で Flash JavaScript API とのやり取りを実行します。

- 起動時にライブラリの関数を登録する
- C 関数が呼び出されるときに、JavaScript から C に渡される引数を取得する
- C 関数を返す前に、戻り値をパッケージ化する

これらのタスクを実行するために、インタプリタはいくつかのデータ型を定義し、API を公開します。このセクションに記載されているデータ型と関数の定義は、"mm_jsapi.h" ファイルに示されています。ライブラリを適切に機能させるには、次の行を使用して、ライブラリ内の各ファイルの一番上に "mm_jsapi.h" ファイルを含める必要があります。

```
#include "mm_jsapi.h"
```

"mm_jsapi.h" ファイルを含めると、"mm_jsapi_environment.h" ファイルが含まれます。このファイルは MM_Environment 構造体を定義します。

"mm_jsapi.h" ファイルのコピーを取得するには、サンプルの ZIP ファイルまたは SIT ファイルから抽出するか ([593 ページの「サンプルの DLL 実装」](#)を参照)、mm_jsapi.h という名前にするファイルに次のコードをコピーします。

```
#ifndef _MM_JSAPI_H_
```

```

#define _MM_JSAPI_H_

/*****
 * パブリックデータ型
 *****/

typedef struct JSContext JSContext;
typedef struct JSObject JSObject;
typedef long jsval;
#ifndef JSBool
typedef long JSBool;
#endif

typedef JSBool (*JSNative)(JSContext *cx, JSObject *obj, unsigned int argc,
                           jsval *argv, jsval *rval);

/* JSBool の有効な値 */
#define JS_TRUE 1
#define JS_FALSE 0

/*****
 * パブリック関数
 *****/

/* JSBool JS_DefineFunction(unsigned short *name, JSNative call, unsigned int
   nargs) */
#define JS_DefineFunction(n, c, a) \
    (mmEnv.defineFunction ? (*(mmEnv.defineFunction))(mmEnv.libObj, n, c, a) \
     : JS_FALSE)

/* unsigned short *JS_ValueToString(JSContext *cx, jsval v, unsigned int
   *pLength) */
#define JS_ValueToString(c, v, l) \
    (mmEnv.valueToString ? (*(mmEnv.valueToString))(c, v, l) : (char *)0)

/* unsigned char *JS_ValueToBytes(JSContext *cx, jsval v, unsigned int *pLength)
   */
#define JS_ValueToBytes(c, v, l) \
    (mmEnv.valueToBytes ? (*(mmEnv.valueToBytes))(c, v, l) : (unsigned char *)0)

/* JSBool JS_ValueToInteger(JSContext *cx, jsval v, long *lp); */
#define JS_ValueToInteger(c, v, l) \
    (mmEnv.valueToInteger ? (*(mmEnv.valueToInteger))(c, v, l) : JS_FALSE)

/* JSBool JS_ValueToDouble(JSContext *cx, jsval v, double *dp); */
#define JS_ValueToDouble(c, v, d) \
    (mmEnv.valueToDouble ? (*(mmEnv.valueToDouble))(c, v, d) : JS_FALSE)

```

```

/* JSBool JS_ValueToBoolean(JSContext *cx, jsval v, JSBool *bp); */
#define JS_ValueToBoolean(c, v, b) \
    (mmEnv.valueToBoolean ? (*(mmEnv.valueToBoolean))(c, v, b) : JS_FALSE)

/* JSBool JS_ValueToObject(JSContext *cx, jsval v, JSObject **op); */
#define JS_ValueToObject(c, v, o) \
    (mmEnv.valueToObject ? (*(mmEnv.valueToObject))(c, v, o) : JS_FALSE)

/* JSBool JS_StringToValue(JSContext *cx, unsigned short *bytes, uint sz, jsval *vp); */
#define JS_StringToValue(c, b, s, v) \
    (mmEnv.stringToValue ? (*(mmEnv.stringToValue))(c, b, s, v) : JS_FALSE)

/* JSBool JS_BytesToValue(JSContext *cx, unsigned char *bytes, uint sz, jsval *vp); */
#define JS_BytesToValue(c, b, s, v) \
    (mmEnv.bytesToValue ? (*(mmEnv.bytesToValue))(c, b, s, v) : JS_FALSE)

/* JSBool JS_DoubleToValue(JSContext *cx, double dv, jsval *vp); */
#define JS_DoubleToValue(c, d, v) \
    (mmEnv.doubleToValue ? (*(mmEnv.doubleToValue))(c, d, v) : JS_FALSE)

/* jsval JS_IntegerToValue(long lv); */
#define JS_IntegerToValue(lv) \
    (((jsval)(lv) << 1) | 0x1)

/* jsval JS_BooleanToValue(JSBool bv); */
#define JS_BooleanToValue(bv) \
    (((jsval)(bv) << 3) | 0x6)

/* jsval JS_ObjectToValue(JSObject *obj); */
#define JS_ObjectToValue(ov) \
    ((jsval)(ov))

/* unsigned short *JS_ObjectType(JSObject *obj); */
#define JS_ObjectType(o) \
    (mmEnv.objectType ? (*(mmEnv.objectType))(o) : (char *)0)

/* JSObject *JS_NewArrayObject(JSContext *cx, unsigned int length, jsval *v) */
#define JS_NewArrayObject(c, l, v) \
    (mmEnv.newArrayObject ? (*(mmEnv.newArrayObject))(c, l, v) : (JSObject *)0)

/* long JS_GetArrayLength(JSContext *cx, JSObject *obj) */
#define JS_GetArrayLength(c, o) \
    (mmEnv.getArrayLength ? (*(mmEnv.getArrayLength))(c, o) : -1)

/* JSBool JS_GetElement(JSContext *cx, JSObject *obj, jsint idx, jsval *vp) */
#define JS_GetElement(c, o, i, v) \
    (mmEnv.getElement ? (*(mmEnv.getElement))(c, o, i, v) : JS_FALSE)

/* JSBool JS_SetElement(JSContext *cx, JSObject *obj, jsint idx, jsval *vp) */
#define JS_SetElement(c, o, i, v) \
    (mmEnv.setElement ? (*(mmEnv.setElement))(c, o, i, v) : JS_FALSE)

```

```

/* JSBool JS_ExecuteScript(JSContext *cx, JSObject *obj, unsigned short *script,
 *      unsigned int sz, jsval *rval) */
#define JS_ExecuteScript(c, o, s, z, r) \
    (mmEnv.executeScript ? (*(mmEnv.executeScript))(c, o, s, z, \
        (LPCTSTR)___FILE___, \
        ___LINE___, r) : JS_FALSE)

/* JSBool JS_ReportError(JSContext *cx, unsigned short *error, unsigned int sz)
 */
#define JS_ReportError(c, e, s) \
    (mmEnv.reportError ? (*(mmEnv.reportError))(c, e, s) : JS_FALSE)

/*****
 * プライベートデータ型、マクロ、およびグローバル変数
 *****/

typedef struct {
    JSObject *libObj;
    JSBool (*defineFunction)(JSObject *libObj, unsigned short *name, JSNative
        call,
        unsigned int nargs);
    unsigned short *(*valueToString)(JSContext *cx, jsval v, unsigned int
        *pLength);
    unsigned char *(*valueToBytes)(JSContext *cx, jsval v, unsigned int
        *pLength);
    JSBool (*valueToInteger)(JSContext *cx, jsval v, long *lp);
    JSBool (*valueToDouble)(JSContext *cx, jsval v, double *dp);
    JSBool (*valueToBoolean)(JSContext *cx, jsval v, JSBool *bp);
    JSBool (*valueToObject)(JSContext *cx, jsval v, JSObject **op);
    JSBool (*stringValue)(JSContext *cx, unsigned short *b, unsigned int sz,
        jsval *vp);
    JSBool (*bytesToValue)(JSContext *cx, unsigned char *b, unsigned int sz,
        jsval *vp);
    JSBool (*doubleToValue)(JSContext *cx, double dv, jsval *vp);
    unsigned short *(*objectType)(JSObject *obj);
    JSObject *(*newArrayObject)(JSContext *cx, unsigned int length, jsval *vp);
    long (*getArrayLength)(JSContext *cx, JSObject *obj);
    JSBool (*getElement)(JSContext *cx, JSObject *obj, unsigned int idx,
        jsval *vp);
    JSBool (*setElement)(JSContext *cx, JSObject *obj, unsigned int idx,
        jsval *vp);
    JSBool (*executeScript)(JSContext *cx, JSObject *obj, unsigned short *script,
        unsigned int sz, unsigned short *file, unsigned int lineNum, jsval
        *rval);
    JSBool (*reportError)(JSContext *cx, unsigned short *error, unsigned int sz);
} MM_Environment;

```

```

extern MM_Environment mmEnv;

// 外部エントリポイントとリンケージを宣言します
#ifdef _WIN32
#   ifndef _MAC
       // Windows
       __declspec( dllexport ) void MM_InitWrapper( MM_Environment *env, unsigned
int envSize );
#   endif
#else
    extern void MM_InitWrapper( MM_Environment *env, unsigned int envSize );
#endif

#define MM_STATE
/* グローバル変数の定義 */
MM_Environment mmEnv;

void
MM_InitWrapper(MM_Environment *env, unsigned int envSize)
{
    extern void MM_Init();

    char **envPtr = (char **)env;
    char **mmPtr = (char **>(&mmEnv);
    char **envEnd = (char **)((char *)envPtr + envSize);
    char **mmEnd = (char **)((char *)mmPtr + sizeof(MM_Environment));

    /* env から mmEnv にフィールドをコピーします (一度にポインタ 1 つずつ) */
    while (mmPtr < mmEnd && envPtr < envEnd)
        *mmPtr++ = *envPtr++;

    /* mmEnv のすべてのフィールドが env で定義されていない場合、定義されていないフィールド
を NULL に設定します */
    while (mmPtr < mmEnd)
        *mmPtr++ = (char *)0;

    /* ユーザーの MM_Init 関数を呼び出します */
    MM_Init();
}

#endif /* _MM_JSAPI_H_ */

```


サンプルの DLL 実装

"ExtendingFlash/dllSampleComputeSum" フォルダの ZIP ファイルと SIT ファイルには、サンプルの DLL 実装があります。[18 ページの「サンプルの実装」](#)を参照してください。次の操作を実行することで、実際に DLL を構築せずにプロセスを確認できます。

- "Sample.jsfl" ファイルを "Configuration/Commands" ディレクトリに格納します。
[7 ページの「JSFL ファイルの保存」](#)を参照してください。
- "Sample.dll" ファイルを "Configuration/External Libraries" ディレクトリに格納します。
[588 ページの「C 関数の統合」](#)を参照してください。
- Flash オーサリング環境で、[コマンド]-[サンプル] を選択します。JSFL ファイル内の trace ステートメントは、Sample.dll で定義された関数の結果を [出力] パネルに送信します。

このセクションでは、サンプルの開発について説明します。このケースでは、DLL は 2 つの数値を追加する 1 つの関数のみを含んでいます。C コードは、次の例のようになります。

```
// C のソースコード
// DLL または共有ライブラリを "Sample" という名前で保存します。
#include <windows.h>
#include <stdlib.h>

#include "mm_jsapi.h"

// サンプルの関数
// JavaScript 関数のすべての実装でこのシングネチャが必要です。
JSBool computeSum(JSContext *cx, JSObject *obj, unsigned int argc, jsval *argv,
                  jsval *rval)
{
    long a, b, sum;

    // 正しい数の引数が渡されたことを確認します。
    if (argc != 2)
        return JS_FALSE;

    // 2 つの引数を jsvals から longs に変換します。
    if (JS_ValueToInteger(cx, argv[0], &a) == JS_FALSE ||
        JS_ValueToInteger(cx, argv[1], &b) == JS_FALSE)
        return JS_FALSE;

    /* 実際の作業を実行します。*/
    sum = a + b;

    /* 戻り値を jsval としてパッケージ化します。*/
    *rval = JS_IntegerToValue(sum);

    /* 正常に終了したことを示します。*/
    return JS_TRUE;
}
```

このコードを作成した後で、DLL ファイルまたは共有ライブラリを作成し、それを適切な "Configuration/External Libraries" ディレクトリに格納します。[588 ページの「C 関数の統合」](#)を参照してください。次に、次のコードを使用して JSFL ファイルを作成し、"Configuration/Commands" ディレクトリに格納します。[7 ページの「JSFL ファイルの保存」](#)を参照してください。

```
// 上で定義した C 関数を実行する JSFL ファイル。  
var a = 5;  
var b = 10;  
var sum = Sample.computeSum(a, b);  
fl.trace("The sum of " + a + " and " + b + " is " + sum );
```

DLL 内で定義された関数を実行するには、Flash オーサリング環境で [コマンド]-[サンプル] を実行します。

データ型

JavaScript インタプリタは、次のデータ型を定義します。

- JSContext
- JSObject
- jsval
- JSBool

typedef struct JSContext JSContext

この不透明なデータ型へのポインタが C レベル関数に渡されます。API の一部の関数は、引数の 1 つとしてこのポインタを受け入れます。

typedef struct JSObject JSObject

この不透明なデータ型へのポインタが C レベル関数に渡されます。このデータ型は、オブジェクトを表します。このオブジェクトは、配列オブジェクトまたは他のオブジェクトタイプである可能性があります。

typedef struct jsval jsval

整数、浮動小数へのポインタ、ストリングへのポインタ、またはオブジェクトへのポインタを含むことができる不透明なデータ構造体。API の一部の関数は、jsval 構造体の内容を読み取ることで関数の引数の値を読み取ることができます。また、一部の関数を使用し、jsval 構造体を記述することで、関数の戻り値を記述することができます。

```
typedef enum { JS_FALSE = 0, JS_TRUE = 1 }
JSBool
```

ブール値を格納する単純なデータ型。

C レベルの API

C レベルの拡張 API は、JSBool (*JSNative) 関数シグネチャと次の関数で構成されます。

- JSBool JS_DefineFunction()
- unsigned short *JS_ValueToString()
- JSBool JS_ValueToInteger()
- JSBool JS_ValueToDouble()
- JSBool JS_ValueToBoolean()
- JSBool JS_ValueToObject()
- JSBool JS_StringToValue()
- JSBool JS_DoubleToValue()
- JSVal JS_BooleanToValue()
- JSVal JS_BytesToValue()
- JSVal JS_IntegerToValue()
- JSVal JS_ObjectToValue()
- unsigned short *JS_ObjectType()
- JSObject *JS_NewArrayObject()
- long JS_GetArrayLength()
- JSBool JS_GetElement()
- JSBool JS_SetElement()
- JSBool JS_ExecuteScript()

```
typedef JSBool (*JSNative)(JSContext *cx,  
JSObject *obj, unsigned int argc, jsval *argv, jsval  
*rval)
```

説明

メソッド。次の状況での JavaScript 関数の C レベルの実装を記述します。

- *cx* ポインタは不透明な JSContext 構造体へのポインタです。JavaScript API の一部の関数にこのポインタを渡す必要があります。この変数は、インタプリタの実行コンテキストを格納します。
- *obj* ポインタは、スクリプトが実行されるコンテキストを持つオブジェクトへのポインタです。スクリプトが実行されている間、*this* キーワードは、このオブジェクトと同じです。
- *argc* 整数は、関数に渡される引数の数です。
- *argv* ポインタは、*jsval* 構造体の配列へのポインタです。配列の長さは、エレメント *argc* 個です。
- *rval* ポインタは、単一の *jsval* 構造体へのポインタです。関数の戻り値は、**rval* に書き込まれる必要があります。

関数は、成功した場合 JS_TRUE を返します。それ以外の場合は JS_FALSE を返します。関数が JS_FALSE を返した場合、現在のスクリプトの実行が停止され、エラーメッセージが表示されます。

JSBool JS_DefineFunction()

シンタックス

```
JSBool JS_DefineFunction(unsigned short *name, JSNative call, unsigned int  
nargs)
```

説明

メソッド。Flash の JavaScript インタプリタに C レベル関数を登録します。JS_DefineFunction() 関数を使用して、*call* 引数で指定した C レベル関数を登録した後に、*name* 引数で指定した名前を参照することによって、JavaScript スクリプト内でその関数を呼び出すことができます。*name* 引数では大文字と小文字が区別されます。

一般的に、この関数は、Flash が起動中に呼び出す MM_Init() 関数から呼び出されます。

引数

```
unsigned short *name, JSNative call, unsigned int nargs
```

- *name* 引数は、関数が JavaScript に公開されるとき関数の名前です。
- *call* 引数は、C レベル関数へのポインタです。関数は、成功または失敗を示す JSBool を返す必要があります。
- *nargs* 引数は、関数が受け取ることを想定している引数の数です。

戻り値

ブール値。JS_TRUE は成功を示し、JS_FALSE は失敗を示します。

unsigned short *JS_ValueToString()

シンタックス

```
unsigned short *JS_ValueToString(JSContext *cx, jsval v, unsigned int *pLength)
```

説明

メソッド。jsval 構造体から関数の引数を抽出し、可能な場合、それをストリングに変換して、変換した値を呼び出し元に返します。

×
#

返されたバッファポインタを変更しないでください。変更すると、JavaScript インタプリタのデータ構造が壊れる場合があります。ストリングを変更するには、文字を別のバッファにコピーし、新しい JavaScript ストリングを作成する必要があります。

引数

```
JSContext *cx, jsval v, unsigned int *pLength
```

- cx 引数は、JavaScript 関数に渡される不透明な JSContext ポインタです。
- v 引数は、ストリングの抽出元の jsval 構造体です。
- pLength 引数は、符号なし整数へのポインタです。この関数は、*pLength をストリングの長さと同じ長さにバイト単位で設定します。

戻り値

成功した場合は、null で終了されたストリングを指すポインタを返します。失敗した場合は null 値を指すポインタを返します。呼び出し元のルーチンは、終了時にこのストリングを開放できません。

JSBool JS_ValueToInteger()

シンタックス

```
JSBool JS_ValueToInteger(JSContext *cx, jsval v, long *lp);
```

説明

メソッド。jsval 構造体から関数の引数を抽出し、可能な場合、それを整数に変換して、変換した値を呼び出し元に返します。

引数

```
JSContext *cx, jsval v, long *lp
```

- cx 引数は、JavaScript 関数に渡される不透明な JSContext ポインタです。

- *v* 引数は、整数の抽出元の `jsval` 構造体です。
- *lp* 引数は、4 バイトの整数へのポインタです。この関数は、変換された値を `*lp` に格納します。

戻り値

ブール値。JS_TRUE は成功を示し、JS_FALSE は失敗を示します。

JSBool JS_ValueToDouble()

シンタックス

```
JSBool JS_ValueToDouble(JSContext *cx, jsval v, double *dp);
```

説明

メソッド。`jsval` 構造体から関数の引数を抽出し、可能な場合、それを倍精度数に変換して、変換した値を呼び出し元に返します。

引数

```
JSContext *cx, jsval v, double *dp
```

- *cx* 引数は、JavaScript 関数に渡される不透明な JSContext ポインタです。
- *v* 引数は、倍精度数の抽出元の `jsval` 構造体です。
- *dp* 引数は、8 バイトの倍精度数へのポインタです。この関数は、変換された値を `*dp` に格納します。

戻り値

ブール値。JS_TRUE は成功を示し、JS_FALSE は失敗を示します。

JSBool JS_ValueToBoolean()

シンタックス

```
JSBool JS_ValueToBoolean(JSContext *cx, jsval v, JSBool *bp);
```

説明

メソッド。`jsval` 構造体から関数の引数を抽出し、可能な場合、それをブール値に変換して、変換した値を呼び出し元に返します。

引数

```
JSContext *cx, jsval v, JSBool *bp
```

- *cx* 引数は、JavaScript 関数に渡される不透明な JSContext ポインタです。
- *v* 引数は、ブール値の抽出元の `jsval` 構造体です。
- *bp* 引数は、JSBool ブール値へのポインタです。この関数は、変換された値を `*bp` に格納します。

戻り値

ブール値。JS_TRUE は成功を示し、JS_FALSE は失敗を示します。

JSBool JS_ValueToObject()

シンタックス

```
JSBool JS_ValueToObject(JSContext *cx, jsval v, JSObject **op);
```

説明

メソッド。jsval 構造体から関数の引数を抽出し、可能な場合、それをオブジェクトに変換して、変換した値を呼び出し元に返します。オブジェクトが配列の場合は、JS_GetArrayLength() および JS_GetElement() を使用してその内容を読み取ります。

引数

JSContext *cx, jsval v, JSObject **op

- cx 引数は、JavaScript 関数に渡される不透明な JSContext ポインタです。
- v 引数は、オブジェクトの抽出元の jsval 構造体です。
- op 引数は、JSObject ポインタへのポインタです。この関数は、変換された値を *op に格納します。

戻り値

ブール値。JS_TRUE は成功を示し、JS_FALSE は失敗を示します。

JSBool JS_StringToValue()

シンタックス

```
JSBool JS_StringToValue(JSContext *cx, unsigned short *bytes, uint sz, jsval *vp);
```

説明

メソッド。ストリングの戻り値を jsval 構造体に格納します。このメソッドは、新しい JavaScript ストリングオブジェクトを割り当てます。

引数

JSContext *cx, unsigned short *bytes, size_t sz, jsval *vp

- cx 引数は、JavaScript 関数に渡される不透明な JSContext ポインタです。
- bytes 引数は、jsval 構造体に格納されるストリングです。ストリングデータがコピーされるので、呼び出し元は、不要になったときにストリングを開放する必要があります。ストリングサイズが指定されていない場合 (sz 引数を参照)、ストリングを null で終了する必要があります。

- `sz` 引数は、バイト単位のストリングのサイズです。`sz` が 0 の場合、`null` で終了されたストリングの長さが自動的に計算されます。
- `vp` 引数は、ストリングの内容のコピー先の `jsval` 構造体へのポインタです。

戻り値

ブール値。`JS_TRUE` は成功を示し、`JS_FALSE` は失敗を示します。

JSBool JS_DoubleToValue()

シンタックス

```
JSBool JS_DoubleToValue(JSContext *cx, double dv, jsval *vp);
```

説明

メソッド。浮動小数点数の戻り値を `jsval` 構造体に格納します。

引数

```
JSContext *cx, double dv, jsval *vp
```

- `cx` 引数は、JavaScript 関数に渡される不透明な `JSContext` ポインタです。
- `dv` 引数は、8 バイトの浮動小数点数です。
- `vp` 引数は、倍精度数の内容のコピー先の `jsval` 構造体へのポインタです。

戻り値

ブール値。`JS_TRUE` は成功を示し、`JS_FALSE` は失敗を示します。

JSVal JS_BooleanToValue()

シンタックス

```
jsval JS_BooleanToValue(JSBool bv);
```

説明

メソッド。ブール値の戻り値を `jsval` 構造体に格納します。

引数

```
JSBool bv
```

- `bv` 引数はブール値です。`JS_TRUE` は成功を示し、`JS_FALSE` は失敗を示します。

戻り値

引数として関数に渡されるブール値が含まれる `JSVal` 構造体。

JSVal JS_BytesToValue()

シンタックス

```
JSBool JS_BytesToValue(JSContext *cx, unsigned short *bytes, uint sz, jsval *vp);
```

説明

メソッド。バイトを JavaScript 値に変換します。

引数

JSContext *cx, unsigned short *bytes*, uint sz, jsval *vp

- *cx* 引数は JavaScript のコンテキストです。
- *bytes* 引数は、JavaScript オブジェクトに変換するバイトのストリングです。
- *sz* 引数は、変換されるバイト数です。
- *vp* 引数は JavaScript 値です。

戻り値

ブール値。JS_TRUE は成功を示し、JS_FALSE は失敗を示します。

JSVal JS_IntegerToValue()

シンタックス

```
jsval JS_IntegerToValue(long lv);
```

説明

メソッド。長整数値を JSVal 構造体に変換します。

引数

lv

- *lv* 引数は、jsval 構造体に変換する長整数値です。

戻り値

引数として関数に渡される整数が含まれる JSVal 構造体。

JSVal JS_ObjectToValue()

シンタックス

```
jsval JS_ObjectToValue(JSObject *obj);
```

説明

メソッド。オブジェクトの戻り値を JSVal に格納します。配列オブジェクトを作成するには JS_NewArrayObject() を使用します。配列オブジェクトの内容を定義するには JS_SetElement() を使用します。

引数

JSObject *obj

- obj 引数は、JSVal 構造体に変換する JSObject オブジェクトへのポインタです。

戻り値

引数として関数に渡されるオブジェクトが含まれる JSVal 構造体。

unsigned short *JS_ObjectType()

シンタックス

```
unsigned short *JS_ObjectType(JSObject *obj);
```

説明

メソッド。オブジェクトへの参照を受け取り、オブジェクトのクラス名を返します。たとえば、オブジェクトが DOM オブジェクトの場合、この関数は "Document" を返します。オブジェクトがドキュメント内のノードの場合、関数は "Element" を返します。配列オブジェクトの場合、関数は "Array" を返します。



返されたバッファポインタを変更しないでください。変更すると、JavaScript インタプリタのデータ構造が壊れる場合があります。

引数

JSObject *obj

- 一般的に、この引数は JS_ValueToObject() 関数を使用して渡され、変換されます。

戻り値

null で終了されたストリングへのポインタ。呼び出し元は、終了時にこのストリングを開放できません。

JSObject *JS_NewArrayObject()

シンタックス

```
JSObject *JS_NewArrayObject(JSContext *cx, unsigned int length [, jsval *v])
```

説明

メソッド。JSVals の配列を含む新しいオブジェクトを作成します。

引数

```
JSContext *cx, unsigned int length, jsval *v
```

- *cx* 引数は、JavaScript 関数に渡される不透明な JSContext ポインタです。
- *length* 引数は、配列に格納できるエレメントの数です。
- *v* 引数は、配列に格納される jsvals への省略可能なポインタです。戻り値が null ではない場合、*v* は、*length* 個のエレメントを含む配列です。戻り値が null の場合、配列オブジェクトの初期の内容は定義されていません。これは、JS_SetElement() 関数を使用して設定することができます。

戻り値

新しい配列オブジェクトへのポインタを返します。失敗した場合は値 null を返します。

long JS_GetArrayLength()

シンタックス

```
long JS_GetArrayLength(JSContext *cx, JSObject *obj)
```

説明

メソッド。配列オブジェクトへのポインタを受け取り、配列内のエレメントの数を取得します。

引数

```
JSContext *cx, JSObject *obj
```

- *cx* 引数は、JavaScript 関数に渡される不透明な JSContext ポインタです。
- *obj* 引数は、配列オブジェクトへのポインタです。

戻り値

配列内のエレメントの数を返します。失敗した場合は -1 を返します。

JSBool JS_GetElement()

シンタックス

```
JSBool JS_GetElement(JSContext *cx, JSObject *obj, jsint idx, jsval *vp)
```

説明

メソッド。配列オブジェクトの1つのエレメントを読み取ります。

引数

```
JSContext *cx, JSObject *obj, jsint idx, jsval *vp
```

- *cx* 引数は、JavaScript 関数に渡される不透明な JSContext ポインタです。
- *obj* 引数は、配列オブジェクトへのポインタです。
- *idx* 引数は、配列内の整数のインデックスです。最初のエレメントはインデックス 0 で、最後のエレメントはインデックス (length - 1) です。
- *vp* 引数は、配列内の jsval 構造体の内容のコピー先である jsval へのポインタです。

戻り値

ブール値。JS_TRUE は成功を示し、JS_FALSE は失敗を示します。

JSBool JS_SetElement()

シンタックス

```
JSBool JS_SetElement(JSContext *cx, JSObject *obj, jsint idx, jsval *vp)
```

説明

メソッド。配列オブジェクトの1つのエレメントを書き込みます。

引数

```
JSContext *cx, JSObject *obj, jsint idx, jsval *vp
```

- *cx* 引数は、JavaScript 関数に渡される不透明な JSContext ポインタです。
- *obj* 引数は、配列オブジェクトへのポインタです。
- *idx* 引数は、配列内の整数のインデックスです。最初のエレメントはインデックス 0 で、最後のエレメントはインデックス (length - 1) です。
- *vp* 引数は、配列内の jsval に内容をコピーする jsval 構造体へのポインタです。

戻り値

ブール値。JS_TRUE は成功を示し、JS_FALSE は失敗を示します。

JSBool JS_ExecuteScript()

シンタックス

```
JS_ExecuteScript (JSContext *cx, JSObject *obj, unsigned short *script, unsigned  
int sz, jsval *rval)
```

説明

メソッド。JavaScript スtring をコンパイルして実行します。スクリプトが戻り値を生成する場合、戻り値は *rval* に返されます。

引数

```
JSContext *cx, JSObject *obj, unsigned short *script, unsigned int sz, jsval  
*rval
```

- *cx* 引数は、JavaScript 関数に渡される不透明な JSContext ポインタです。
- *obj* 引数は、スクリプトが実行されるコンテキストを持つオブジェクトへのポインタです。スクリプトが実行されている間、*this* キーワードは、このオブジェクトと同じです。通常、これは JavaScript 関数に渡される JSObject ポインタです。
- *script* 引数は、JavaScript コードを含む String です。String サイズが指定されていない場合 (*sz* 引数を参照)、String を null で終了する必要があります。
- *sz* 引数は、バイト単位の String のサイズです。*sz* が 0 の場合、null で終了された String の長さが自動的に計算されます。
- *rval* 引数は、単一の jsval 構造体へのポインタです。関数の戻り値は、*rval* に格納されます。

戻り値

ブール値。JS_TRUE は成功を示し、JS_FALSE は失敗を示します。