Autodesk®
Scaleform®

Scaleform 4.2 AS3 エクステンション リファレンス

この文書では Scaleform 4.2 で使用できる ActionScript 3.0 エクステンションについて解説します。

著者： Artem Bolgar、Prasad Silva

バージョン： 1.04

最終変更日： 2012 年 9 月 17 日

Autodesk®
GAMEWARE

著作権に関する情報

Autodesk® Scaleform® 4.2

OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE REGARDING THESE MATERIALS.

Autodesk Scaleform へのお問い合わせ先：

文書名	AS3 Extensions Reference
住所	Autodesk Scaleform Corporation 6305 Ivy Lane, Suite 310 Greenbelt, MD 20770, USA
Website	www.scaleform.com
Email	info@scaleform.com
直通電話	+1 (301) 446-3200
ファックス	+1 (301) 446-3199

目次

1 はじめに

Autodesk Scaleform® SDK™ はクリーンルーム開発プロセスで作られた軽く高性能のリッチメディア Adobe® Flash® ベクトルグラフィックスエンジンで、特にコンソールや PC ゲームのデベロッパー用に開発しました。Autodesk Scaleform は Adobe Creative Suite®などの実績あるビジュアルオーサリングツールのスケーラビリティと開発の容易さの両方を持ち合わせ、ゲームデベロッパーの要求する最新のハードウェア画像加速を取り入れました。

Flash が主にウェブ用でゲームやアプリケーション開発用ではないので、フォーカスの処理、マウスへのサポート、テキストフィールドのサポート、IME 入力の処理などの分野では機能が限られています。Autodesk Scaleform は ActionScript クラスに「エクステンション」を追加することでこれらの分野での基本的な機能性を高めました。Scaleform Player でのエクステンションのサポートをイネーブルするには、scaleform.gfx.Extensions.enabled のプロパティを true に設定する必要があります。

```
scaleform.gfx.Extensions.enabled = true;
```

または

```
import scaleform.gfx.*;
Extensions.enabled = true;
```

また、Flash に「Source path」として *GFXSDK/Resources/AS3/CLIK* を追加して ActionScript 3.0 コンパイラーからエクステンションクラスが見えるようにしておく必要があります（Flash Studio では、「*Edit*」→「*Preferences*」→「*ActionScript*」→「*ActionScript 3.0 Settings*…」→「*Source path*」）。多くの場合、このステートメントは FLA ファイルでエクステンションを使用する最初のフレームに追加するのが適切です。このプロパティを設定しない場合、Scaleform Player はエクステンションへの参照を全て無視して Flash への適合性を最大にするようにします。

この文書で記述するエクステンションの機能は Scaleform でのみ実装されている物ですので、標準の Flash Player では機能しないことに注意してください。各エクステンションには Scaleform のバージョン番号が関連付けられており、これがエクステンションの追加されたプレイヤーSDK のリリース番号を識別します。より古いバージョン番号の Scaleform Player ではエクステンションは動作しません。

Autodesk では各リリースを通してエクステンション API がサポートされ、一貫性があるように最大の努力を払いますが、将来の Scaleform のリリースでエクステンション API の変更、名称変更、除去をする権利を有します。大きな変更をする場合は、これを大きなリリースと同時に行い、できるだけ早期に通知を出すように努めます。

2 エクステンション

イネーブルされているスタティックなプロパティ

```
enabled:Boolean  [read-write]
```

Scaleform のバージョン : 4.0.12

エクステンションをイネーブル、ディスエーブルします。

noInvisibleAdvance スタティック プロパティ

```
noInvisibleAdvance:Boolean  [read-write]
```

Scaleform のバージョン : 4.0.12

これを true に設定すると、このプロパティは全ての目に見えないムービークリップを先に進めることをオフにします。これは多くの目に見えないムービークリップを含む SWF の動作を改善することに使用できます。ただし、Flash は目に見えないムービークリップを先に進めますので注意してください（タイムラインアニメーションを実行し、フレームの ActionScript コードを呼び出す、等をまだ行います）。このように、このプロパティを true に設定すると Scaleform と Flash での動作が異なる場合があります。

getTopMostEntity()スタティックの方法

```
public function getTopMostEntity(x :Number, y :Number, testAll:Boolean) :
DisplayObject
```

Scaleform のバージョン : 4.0.13

これは座標（x, y）（ステージ座標空間）の最も上にある DisplayObject のインスタンスを返します。この方法はまたボタンハンドラー（例えば CLICK, MOUSE_DOWN、ROLL_OVER 等）が設定されたキャラクターと設定されていないものの間では異なることがあります。この区別は、ハンドラーの無いキャラクターを除外してマウスのイベントを処理するときに役立つことがあります。

この方法は hitTest が x と y 座標が特定のオブジェクトの内側にあり、かつ getTopMostEntity が指定された x、y 座標で実際のオブジェクトを返すことをチェックするので、MovieClip.hitTest の逆です。したがって、Extensions.getTopMostEntity(x, y).hitTest(x, y, true) == true。

パラメーター

testAll :Boolean　　– ボタンハンドラーの付いたキャラクターのみ、（false）または任意のキャラクター（true）を探します。このパラメーターが指定されていない場合は、getTopMostEntity はこれが true であるとみなします。
x :Number, y :Number　– キャラクターを探すステージ座標です。

参照：

getMouseTopMostEntity

getMouseTopMostEntity()スタティックの方法

```
public function getTopMostEntity([testAll :Boolean],
   [mouseIndex :uint]) :DisplayObject
```

Scaleform のバージョン：4.0.13

現在のマウスのカーソルの位置で見つかる最も上にある DisplayObject のインスタンスを返します。この関数は getTopMostEntity と似ていますが、明示的な座標ではなく、マウスの座標を使用します。

パラメーター
testAll :Boolean　　– ボタンハンドラーの付いたキャラクターのみ、（false）または任意のキャラクター（true）を探します。このパラメーターが指定されていない場合は、getTopMostEntity はこれが true であるとみなします。
mouseIndex :Number– ゼロに基づいたマウスのインデックス。

参照：

getTopMostEntity

getMouseCursorType()スタティックの方法

```
public function getMouseCursorType([mouseIndex :uint]) :String
```

Scaleform のバージョン：4.0.13

このスタティックな方法は指定したマウスコントローラーに現在のマウスのカーソルタイプを返します。この方法は flash.ui.Mouse.cursor プロパティと似ていますが、この方法は複数のマウスに対してカーソルの変更ができることだけが異なります。

マウスカーソルを追跡し、変更しないようにするには、MouseCursorEvent.CURSOR_CHANGE エクステンションイベントを使用してください。

パラメーター
`mouseIndex :Number` – Zero-based mouse index (zero by default).

戻り値

カーソルのタイプを返します。次のような flash.ui.MouseCursor スタティック定数を参照してください。

- `flash.ui.MouseCursor.ARROW :String = "arrow"`
- `flash.ui.MouseCursor.BUTTON :String = "button"`
- `flash.ui.MouseCursor.HAND :String = "hand"`
- `flash.ui.MouseCursor.IBEAM :String = "ibeam"`

参照：

setMouseCursorType
MouseCursorEvent

setMouseCursorType()スタティックの方法

```
public function setMouseCursorType(cursor :String, [mouseIndex :uint]) : void
```

Scaleform のバージョン：4.0.13

このスタティックな方法はパラメーターcursor に応じてマウスのカーソルを変更します。この方法は flash.ui.Mouse.cursor プロパティと似ていますが、この方法は複数のマウスに対してカーソルの変更ができることだけが異なります。

マウスカーソルを追跡し、変更しないようにするには、MouseCursorEvent.CURSOR_CHANGE エクステンションイベントを使用してください。

パラメーター

`cursor :uint` – カーソルのタイプについては次のような flash.ui.MouseCursor スタティック定数を参照してください。

- `flash.ui.MouseCursor.ARROW :String = "arrow"`
- `flash.ui.MouseCursor.BUTTON :String = "button"`
- `flash.ui.MouseCursor.HAND :String = "hand"`
- `flash.ui.MouseCursor.IBEAM :String = "ibeam"`

`mouseIndex :Number` – ゼロに基づいたマウスのインデックス。（デフォルトではゼロ）。

参照：

getMouseCursorType
MouseCursorEvent

numControllers スタティック プロパティ

```
numControllers:uint  [read]
```

Scaleform のバージョン：4.0.12

システム中で検出されたコントローラーの数を返します。

setEdgeAAMode()スタティックの方法

```
public function setEdgeAAMode(dispObj:DisplayObject, mode:uint):void
```

Scaleform のバージョン：4.0.12

特定のディスプレーオブジェクトとその子供の EdgeAA モードを設定します。次のモード値を受け付けます。

- scaleform.gfx.Extensions.EDGEAA_INHERIT = 0; 親から EdgeAA モードを譲り受けます。デフォルトではオンです。
- scaleform.gfx.Extensions.EDGEAA_ON = 1; ディスエーブルされていなければ、ターゲットディスプレーオブジェクトとその子供に対して、EdgeAA モードをイネーブルします（EDGEAA_DISABLE 参照）。
- scaleform.gfx.Extensions.EDGEAA_OFF = 2;EdgeAA はターゲットディスプレーオブジェクトとその子供には使用しないでください。
- scaleform.gfx.Extensions.EDGEAA_DISABLE = 3; ターゲットディスプレーとその子供に対して EdgeAA をディスエーブルします。譲り受けた EDGEAA_ON モードの値をオーバーライドします。

パラメーター
`dispObj :DisplayObject` – 新しい EdgeAA モード値を適用するターゲットディスプレーオブジェクト。
mode :uint – EdgeAA モード値。

参照：
getEdgeAAMode

getEdgeAAMode()スタティックの方法

```
static public function getEdgeAAMode(dispObj:DisplayObject): uint
```

Scaleform のバージョン：4.0.12

特定のディスプレーオブジェクトから EdgeAA モード値を取得します。

パラメーター
`dispObj :DisplayObject` – EdgeAA モード値を取得するターゲットディスプレーオブジェクト。

参照：
setEdgeAAMode

3 DisplayObjectEx エクステンション

setRendererString スタティックの方法

```
static public function setRendererString(o:DisplayObject, s:String)
```

Scaleform のバージョン: 4.0.17

このプロパティを使って、任意の MovieClip インスタンスのために、カスタムのディレクティブを ActionScript からレンダラーに送ることができます。このプロパティが設定されている場合、ストリング値は、ユーザー データとしてレンダラーに送信されます。

デフォルト値はありません。

例:

```
import scaleform.gfx.*;

// m is a MovieClip on stage.

DisplayObjectEx.setRendererString(m, "Abc");

trace(DisplayObjectEx.getRendererString(m));
```

setRendererFloat スタティックの方法

```
static public function setRendererFloat(o:DisplayObject, f:Number)
```

Scaleform のバージョン: 4.0.17

このプロパティを使って、任意の MovieClip インスタンスのために、カスタムのディレクティブを ActionScript からレンダラーに送ることができます。このプロパティが設定されている場合、フロート値は、ユーザー データとしてレンダラーに送信されます。

デフォルト値はありません。

Example:

```
import scaleform.gfx.*;

// m is a MovieClip on stage.
DisplayObjectEx.setRendererFloat(m, 17.1717);
trace(DisplayObjectEx.getRendererFloat(m));
```

disableBatching スタティックの方法

```
static public function disableBatching(o:DisplayObject, b:Boolean)
```

Scaleform のバージョン: 4.0.17

このプロパティは、カスタム描画に対するメッシュ生成のバッチ処理を無効にします。

Example

```
import scaleform.gfx.*;

// m is a MovieClip on stage.

DisplayObjectEx.disableBatching(batchDisable, true);
trace(DisplayObjectEx.isBatchingDisabled(batchDisable));
```

4 FocusEventEx エクステンション

```
package scaleform.gfx
{
    import flash.events.FocusEvent;

    public final class FocusEventEx extends FocusEvent
    {
        public var controllerIdx : uint = 0;

        public function FocusEventEx(type:String) { super(type); }
    }
}
```

このイベントは標準の flash.events.FocusEvent のエクステンションです。これはイベントを起こしたコントローラーのゼロに基づいたインデックス「controllerIdx」メンバーを追加します。
Extensions.enabled プロパティが true に設定されている場合、Scaleform は常に標準の FocusEvent の替わりに FocusEventEx イベントを生成します。受け取ったイベントが FocusEventEx のインスタンスかどうかをチェックして、その通りならばイベントオブジェクトをエクステンションタイプに型変換してください。例：

```
import scaleform.gfx.*;
import flash.events.FocusEvent;

Extensions.enabled = true;

function ev(e:FocusEvent)
{
      if (e is FocusEventEx)
      {
            var ee:FocusEventEx = e as FocusEventEx;
            trace("    controllerIdx = "+ee.controllerIdx);
      }
}
stage.addEventListener(FocusEvent.MOUSE_FOCUS_CHANGE, ev);
```

controllerIdx プロパティ

```
controllerIdx :uint  [read]
```

Scaleform のバージョン：4.0.12

イベントに対してどのキーボード/コントローラーが使用されているかを示します（ゼロ基準のインデックス）。

5 FocusManager エクステンション

alwaysEnableArrowKeys スタティックプロパティ

```
alwaysEnableArrowKeys:Boolean  [read-write]
```

Scaleform のバージョン：4.0.12

このスタティックプロパティは_focusrect プロパティが false に設定されている（フォーカスがキャプチャーされているときに適用される）場合でもキーのフォーカスが変更できるようにします。デフォルトでは Flash は黄色いフォーカスの四角形が_focusrect = false でディスエーブルされている場合はキーのフォーカスの変更を許しません。この動作を変更するには alwaysEnableArrowKeys プロパティを true に変えてください。

disableFocusKeys スタティックプロパティ

```
disableFocusKeys:Boolean  [read-write]
```

Scaleform のバージョン：4.0.12

このスタティックプロパティは全てのフォーカスキー（タブ、シフトタブ、矢印キー）をディスエーブルして、ユーザーが自身のフォーカスキーの管理を実装できるようにします。

moveFocus()スタティックの方法

```
static public function moveFocus(keyToSimulate :String,
                          startFromMovie:InteractiveObject = null,
                          includeFocusEnabledChars :Boolean = false,
                          controllerIdx : uint = 0) :InteractiveObject
```

Scaleform のバージョン：4.0.12

このスタティックの方法は 4 つのフォーカスキーの内 1 つが押されることをシミュレートすることでフォーカスの四角形を移動することに使用します。タブ、シフトタブ、または矢印キー。この方法は disableFocusKeys と modalClip プロパティと合わせて、カスタムのフォーカス管理の実装に使用できます。

パラメーター

`keyToSimulate :String` - シミュレーションするキーの名称：「up」、「down」、「left」、「right」、「tab」、「shifttab」。

`startFromMovie:InteractiveObject` – キャラクターを指定するオプションのパラメーターです。moveFocus は現在フォーカスされている物に替えてこれを開始点として使用します。このプロパティ

はヌル（空値）または未定義であることがあり、その場合は現在フォーカスされているキャラクターが開始点として使用されることを示します。このことは第 3 のオプションパラメーターを指定する時に使用できます。
includeFocusEnabledChars :Boolean – オプションのフラッグで、moveFocus が focusEnabled プロパティだけ、また tabEnabled / tabIndex プロパティが設定された状態でキャラクター上に適用できるようにします。このフラッグが指定されていない場合、または false に設定されていない場合には、tabEnabled / tabIndex プロパティの設定されているキャラクターのみがフォーカスの移動に参加します。
controllerIdx：uint - 操作に使用されるコントローラー0 のインデックスです。これが指定されていない場合は、デフォルトのコントローラー（controller 0）が使用されます。

戻り値
次にフォーカスされるキャラクターを返すか、またはキャラクターが見つからない場合は null を返します。

参照：
findFocus
disableFocusKeys
setModalClip
getModalClip

findFocus()スタティックの方法

```
static public function findFocus(keyToSimulate :String,
                          parentMovie:DisplayObjectContainer = null,
                          loop :Boolean = false,
                          startFromMovie:InteractiveObject = null,
                          includeFocusEnabledChars :Boolean = false,
                          controllerIdx : uint = 0) :InteractiveObject
```

Scaleform のバージョン：4.0.12

このスタティックの方法は以下のキーの内 1 つが押されることをシミュレートすることでフォーカスアイテムの次のアイテムを見つけることに使用します。タブ、シフトタブ、または矢印キー。この方法は disableFocusKeys と setModalClip/getModalClip エクステンションと共に使用して、カスタムのフォーカス管理の実装に使用できます。

パラメーター
keyToSimulate :String - シミュレーションするキーの名称：「up」、「down」、「left」、「right」、「tab」、「shifttab」。
parentMovie:DisplayObjectContainer – モーダルクリップとして使用されるムービークリップです。フォーカスアイテムの検索はこのクリップの子供にのみ行われます。ヌル（空値）であっても良い。
loop :Boolean - フォーカスをループするブール値のフラッグです。例えば、現在フォーカスされているアイテムが一番下にあり、キーが「下」の場合、findFocus は「ヌル」を返す（このフラッグが「false」の場合）か、またはフォーカスのできる一番上のアイテム（このフラッグが「true」の場合）を返します。

`startFromMovie:InteractiveObject` – キャラクターを指定するオプションのパラメーターです。findFocus は現在フォーカスされている物に替えてこれを開始点として使用します。このプロパティはヌル（空値）または未定義であることがあり、その場合は現在フォーカスされているキャラクターが開始点として使用されることを示します。
`includeFocusEnabledChars :Boolean` – オプションのフラッグで、moveFocus が focusEnabled プロパティだけ、また tabEnabled / tabIndex プロパティが設定された状態でキャラクター上に適用できるようにします。このフラッグが指定されていない場合、または false に設定されていない場合には、tabEnabled / tabIndex プロパティの設定されているキャラクターのみがフォーカスの移動に参加します。
controllerIdx：uint - フォーカスを操作しているコントローラーのゼロに基づいたインデックスです。これはフォーカスグループと合わせて複数のフォーカスサポートを提供することに使用できます。

戻り値

次にフォーカスされるキャラクターを返すか、またはキャラクターが見つからない場合は null を返します。

参照：

moveFocus
disableFocusKeys
setModalClip
getModalClip

setFocus()スタティックの方法

```
static public function setFocus(obj:InteractiveObject, controllerIdx:uint = 0) : void
```

Scaleform のバージョン：4.0.12

このスタティックな方法はステージフォーカスのプロパティを割り当てるのと同じことを行いますが、操作に割り当てられているコントローラーのインデックスを指定できることだけが異なっています。

パラメーター

`obj:InteractiveObject` - 新規にフォーカスされた対話的なオブジェクト
`controllerIdx : uint` - イベントに対してどのキーボード/コントローラーが使用されているかを示します。

getFocus()スタティックの方法

```
static public function getFocus(controllerIdx:uint = 0) :InteractiveObject
```

Scaleform のバージョン：4.0.12

このスタティックな方法はステージフォーカスのプロパティと同じ値を返します。操作に割り当てられているコントローラーのインデックスを指定できることだけが異なっています。

パラメーター

controllerIdx : uint - イベントに対してどのキーボード/コントローラーが使用されているかを示します。

戻り値

現在フォーカスされている対話的オブジェクトです。

numFocusGroups スタティックプロパティ

```
numFocusGroups() :uint  [read]
```

Scaleform のバージョン : 4.0.12

setControllerFocusGroup 関数への呼び出しで設定されるフォーカスグループの数を返します。フォーカスグループ 0 と 3 がアクティブな場合は、numFocusGroups は 2 を返します。

setFocusGroupMask()スタティックの方法

```
public function setFocusGroupMask(obj:InteractiveObject, mask:uint) : void
```

Scaleform のバージョン : 4.0.12

この方法はキャラクターとその子供**全て**にビットマスクを設定します。このビットマスクはフォーカスグループの所有権をキャラクターに割り当てます。このことはビットマスクに示されたコントローラーのみがフォーカスをキャラクターへ動かし、またキャラクター内で動かすことを意味します。setControllerFocusGroup エクステンションの方法を使うと、フォーカスグループをコントローラーに割り当てることができます。

例えば、「button1」がコントローラー0 のみでフォーカス可能で、「movieclip2」がコントローラー 0 と 1 のみでフォーカス可能だと仮定します。この動作を達成するには、次のようにフォーカスグループをコントローラーに割り当てます。

```
FocusManager.setControllerFocusGroup(0, 0);
FocusManager.setControllerFocusGroup(1, 1);

FocusManager.setFocusGroupMask(button1, 0x1); // bit 0 – focus group 0
FocusManager.setFocusGroupMask(movieclip2,0x1 | 0x2); // bits 0 and 1 – focus
                                                      //  groups 0 and 1
```

「focusGroupMask」ビットマスクは親のムービークリップに設定できます。これはマスク値をその子供全てに伝えます。

パラメーター

obj:InteractiveObject - 対話的オブジェクト
mask:uint - フォーカスグループのビットマスク

参照：

setControllerFocusGroup
getFocusGroupMask

getFocusGroupMask()スタティックの方法

```
static public function getFocusGroupMask(obj:InteractiveObject) : uint
```

Scaleform のバージョン：4.0.12

現在のフォーカスグループのビットマスクを返します（詳細は setFocusGroupMask を参照）。

パラメーター
`obj:InteractiveObject` - 対話的オブジェクト

戻り値
特定の対話的オブジェクトに対するフォーカスグループのビットマスクです。

参照：

setControllerFocusGroup
setFocusGroupMask

setControllerFocusGroup()スタティックの方法

```
static public function setControllerFocusGroup(controllerIdx:uint,
                                                focusGroupIdx:uint) :   Boolean
```

Scaleform のバージョン：4.0.12

このスタティックの方法は controllerIndex で示すコントローラーをフォーカスグループに割り当てます。デフォルトでは、全てのコントローラーはフォーカスグループ 0 に割り当てられています。このことは全てのコントローラーは同じフォーカスを使用しているということです。しかし、各コントローラーがそれ自身のフォーカスに作用するようにすることも可能です。例えば、2 つのコントローラーが別々のフォーカスを持つ場合（スプリットスクリーン使用の場合）、setControllerFocusGroup(1,1)はコントローラー1 に対して別々のフォーカスグループを生成します。setControllerFocusGroup(1,0)を呼び出すとコントローラー0 と 1 は再び同じフォーカスを共有するようになります。

パラメーター
`controllerIdx:uint` - コントローラーのゼロに基づいたインデックスです。
`focusGroupIdx:uint` - フォーカスグループのゼロに基づいたインデックスです。

戻り値
成功した場合は「true」を返します。

getControllerFocusGroup()スタティックの方法

```
static public function getControllerFocusGroup(controllerIdx:uint) : uint
```

Scaleform のバージョン : 4.0.12

このスタティックな方法は指定したコントローラーに割り当てられたフォーカスグループインデックスを返します。

パラメーター
`controllerIndex` – 物理的なコントローラーのゼロに基づいたインデックスです。

戻り値
フォーカスグループのゼロに基づいたインデックス。

setModalClip()スタティックの方法

```
static public function setModalClip(mc:Sprite, controllerIdx:uint = 0) : void
```

Scaleform のバージョン : 4.0.12

このスタティックの方法は指定されたムービークリップをフォーカス管理の「モーダル」クリップとして設定します。このことはタブ、シフトタブ、矢印キーは、全ての「tabable」な子供にわたって指定されたムービークリップの内部のみでフォーカスを移動することを意味します。

パラメーター
`controllerIdx:uint` - コントローラーのゼロに基づいたインデックスです。
`mc:Sprite` - モーダルクリップです。

getModalClip()スタティックの方法

```
static public function getModalClip(controllerIdx:uint = 0) :Sprite
```

Scaleform のバージョン : 4.0.12

このスタティックな方法は指定したコントローラーに対するモーダルクリップを返します。

パラメーター
`controllerIdx` - コントローラーのゼロに基づいたインデックスです。

戻り値
見つからなかった場合はモーダルクリップまたは不定義。

6 GamePad エクステンション

コントロール定数

このクラスはトリガー、アナログスティック、ボタンなどの一般的なゲームパッドコントロール用のヘルパー定数を提供します。これらは SF_KeyCodes.h で定義された定数の正反射です。Scaleform はランタイムに正しい値を注入しますので、これらの定数を使用する SWF のコンパイルは意図したように動作します。

Scaleform の FxPlayer フレームワークは便宜上ゲームパッドのコントロールをキーボード上の対応するものにマップしますが、カスタム統合またはアプリケーションは、AS3 キーイベントに対して必要であれば、コントローラーとキーボードの区別をつけるためにこれらの定数を GFx::Movie::HandleEvent と共に使用することがあります。

Scaleform の FxPlayer フレームワークは、これらの定数をこのようなアナログ値への適切なフードバックを与えるためには GamePadAnalogEvents と共に使用しません。しかし、GamePad の定数を使用する替わりに、デベロッパーはキーボードコードを使用することもできます。ゲームパッドイベントをキーボードイベントと組み合わせるかどうかはデベロッパー次第です。

supportsAnalogEvents()スタティックの方法

```
public static function supportsAnalogEvents() :Boolean
```

Scaleform のバージョン：4.0.13

基盤となるハードウェアプラットフォームでの Scaleform の実装がトリガーやサムスティックなどのゲームパッド アナログイベントをサポートする場合は、true を返します。この値は GamePadAnalogEvents がサポートされているかどうかを確かめるのに使用できます。

参照

GamePadAnalogEvent

7 GamePadAnalogEvent エクステンション

```
package scaleform.gfx
{
    import flash.events.Event;


    public final class GamePadAnalogEvent extends Event
    {
        public static const CHANGE:String   = "gamePadAnalogChange";

        public var code : uint          = 0;    // See scaleform.gfx.GamePad for
                                                // valid pad codes
        public var controllerIdx : uint = 0;
        public var xvalue :Number      = 0;    // Normalized [-1, 1]
        public var yvalue :Number      = 0;    // Normalized [-1, 1]

        public function GamePadAnalogEvent(bubbles:Boolean, cancelable:Boolean,
                                           code:uint, controllerIdx:uint = 0,
                                           xvalue:Number = 0, yvalue:Number = 0)
        {
            super(GamePadAnalogEvent.CHANGE, bubbles, cancelable);
            this.code = code;
            this.controllerIdx = controllerIdx;
            this.xvalue = xvalue;
            this.yvalue = yvalue;
        }
    }
}
```

このイベントは、特定のプラットフォームに対して Scaleform がゲームパッドのアナログイベントをサポートする場合に発せられます。これは Stage からのみ出すことができ、全てのリスナーは Stage オブジェクトに付随することが必要です。Scaleform の FxPlayer フレームワークはこれらのイベントを「コード」プロパティに適切な GamePad 定数と共に発しますが、Scaleform の各統合でキーボード値などのその他の値を必要に応じて使用することも可能です。例：

```
import scaleform.gfx.*;

trace(“GamePadAnalogEvents supported?“ + GamePad.supportsAnalogEvents());

function ev(e:GamePadAnalogEvent)
{
      trace("code = " + ev.code);
```

```
        trace("controllerIdx = " + ev.controllerIdx);
        trace("xvalue = " + ev.xvalue);
        trace("yvalue = " + ev.yvalue);
}
stage.addEventListener(GamePadAnalogEvent.CHANGE, ev);
```

code プロパティ

```
code :uint
```

Scaleform のバージョン : 4.0.13

このイベントを生成するのにどのキー/コントロールが使用されたかを示します。Scaleform の FxPlayer フレームワークはゲームパッドの定数を使用します。GamePad 参照。

controllerIdx プロパティ

```
controllerIdx :uint
```

Scaleform のバージョン : 4.0.13

イベントに対してどのキーボード/コントローラーが使用されているかを示します（ゼロ基準のインデックス）。

xvalue プロパティ

```
xvalue :Number
```

Scaleform のバージョン : 4.0.13

x 軸での現在の値を示します。この値は-1 から 1（両端含む）で正規化されます。

yvalue プロパティ

```
yvalue :Number
```

Scaleform のバージョン : 4.0.13

y 軸での現在の値を示します。この値は-1 から 1（両端含む）で正規化されます。

8 InteractiveObjectEx エクステンション

getHitTestDisable()スタティックの方法

```
public function getHitTestDisable(o:InteractiveObject) :Boolean
```

Scaleform のバージョン：4.0.13

「hitTestDisable」フラッグのステートを返します。これが true に設定されている場合、MovieClip.hitTest 関数は、ヒットテスト検出の間はこの対話的オブジェクトを無視します。さらに、その他全てのマウスのイベントはオブジェクトに伝えられません。

デフォルト値は false です。

パラメーター
o　　- An interactive object.

戻り値
「hitTestDisable」フラッグのステートを表すブール値。

参照：
InteractiveObjectEx.setHitTestDisable

setHitTestDisable()スタティックの方法

```
public function setHitTestDisable(o:InteractiveObject, f:Boolean) : void
```

Scaleform のバージョン：4.0.13

「hitTestDisable」フラッグのステートを設定します。これが true に設定されている場合、MovieClip.hitTest 関数は、ヒットテスト検出の間はこの対話的オブジェクトを無視します。さらに、その他全てのマウスのイベントはオブジェクトに伝えられません。デフォルト値は false です。

パラメーター
o　　- An interactive object.
f - 「hitTestDisable」フラッグの新しいステートを表すブール値。

参照
InteractiveObjectEx.getHitTestDisable

getTopmostLevel()スタティックの方法

```
public function getTopmostLevel (o:InteractiveObject) :Boolean
```

Scaleform のバージョン：4.0.13

「topmostLevel」フラッグのステートを返します。true に設定されている場合、深さにかかわらずこのキャラクターはその他全ての物の上に表示されます。

パラメーター

o - An interactive object.

戻り値

「topmostLevel」フラッグのステートを表すブール値。

参照：

InteractiveObjectEx.setTopmostLevel

setTopmostLevel ()スタティックの方法

```
public function setTopmostLevel(o:InteractiveObject, f:Boolean) : void
```

Scaleform のバージョン：4.0.13

「topmostLevel」フラッグのステートを設定します。true に設定されている場合、深さにかかわらずこのキャラクターはその他全ての物の上に表示されます。このことは、カーソルが全てのレベルからのオブジェクトの上に描かれる必要のある場合にカスタムマウスカーソルを実装する場合に役に立ちます。デフォルト値は false です。

いくつかのキャラクターを「topmostLevel」にする場合、描画順はキャラクターを topmost にしない場合と同じです。つまり、オブジェクト A がオブジェクト B の下に描かれる場合、これらを topmost にした後でもオブジェクト A はオブジェクト B の下のままになり、これは「topmostLevel」プロパティを true に設定する順番には関係ありません。
注：キャラクターが「topmostLevel」にされた後では、swapDepth ActionScript 関数はこのキャラクターに対して何ら効力を持ちません。

デフォルト値は false です。

パラメーター

o - An interactive object.
f - 「topmostLevel」フラッグの新しいステートを表すブール値。

参照：

InteractiveObjectEx.getTopmostLevel

9 KeyboardEventEx エクステンション

```
package scaleform.gfx
{
    import flash.events.KeyboardEvent;

    public final class KeyboardEventEx extends KeyboardEvent
    {
        public var controllerIdx : uint = 0;

        public function KeyboardEventEx(type:String) { super(type); }
    }
}
```

このイベントは標準の flash.events.KeyboardEvent のエクステンションです。これはイベントを起こしたコントローラーのゼロに基づいたインデックス「controllerIdx」メンバーを追加します。Extensions.enabled プロパティが true に設定されている場合、Scaleform は常に標準の KeyboardEvent の替わりに KeyboardEventEx イベントを生成します。受け取ったイベントが KeyboardEventEx のインスタンスかどうかをチェックして、その通りならばイベントオブジェクトをエクステンションタイプに型変換してください。例：

```
import scaleform.gfx.*;
import flash.events.KeyboardEvent;

Extensions.enabled = true;

function ev(e:KeyboardEvent)
{
      if (e is KeyboardEventEx)
      {
            var ee:KeyboardEventEx = e as KeyboardEventEx;
            trace("    controllerIdx = "+ee.controllerIdx);
      }
}
stage.addEventListener(KeyboardEvent.KEY_DOWN, ev);
stage.addEventListener(KeyboardEvent.KEY_UP, ev);
```

controllerIdx プロパティ

```
controllerIdx :uint  [read]
```

Scaleform のバージョン：4.0.12

イベントに対してどのキーボード/コントローラーが使用されているかを示します（ゼロ基準のインデックス）。

10 MouseCursorEvent エクステンション

```
package scaleform.gfx
{
    import flash.events.Event;

    public final class MouseCursorEvent extends Event
    {
        public var cursor :String = "auto";
        public var mouseIdx : uint = 0;

        static public const CURSOR_CHANGE :String = "mouseCursorChange";

        public function MouseCursorEvent()
  {
      super("MouseCursorEvent", false, true);
  }
    }
}
```

このイベントはマウスのカーソルの変化をトラックするか、防ぐことの片方または両方を行います。これには次のようにイベントのプロパティが設定されています。
bubbles – false の場合、バブルしません
cancellable – true の場合、デフォルトのアクション（カーソルの変化）は preventDefault()方法を呼び出して防ぐことができます。

このイベントはカスタムでアニメ―トされたカーソルを実装するのに役立ちます。Scaleform がマウスカーソルの形を変えるたびに（テキストフィールドまたはボタンにマウスを重ねる、または flash.ui.Mouse.cursor プロパティを設定して）、このイベントは**ステージ用に**発せられます（Extensions.enabled は true に設定されている場合）。例：

```
Extensions.enabled = true;

function e(e:MouseCursorEvent)
{
   trace(e.type + "  " + e.mouseIdx + "  "+ e.cursor);
   e.preventDefault();
}
stage.addEventListener(MouseCursorEvent.CURSOR_CHANGE, e);
```

注：このイベントはリスナーがステージで設定されている場合に限って発せられます。

cursor プロパティ

```
cursor :String  [read]
```

Scaleform のバージョン : 4.0.13

このプロパティはカーソルの変更される先のタイプを示します。これは次のような
flash.ui.MouseCursor クラスからの文字列値の一つを含みます。

- `flash.ui.MouseCursor.ARROW :String = “arrow”`
- `flash.ui.MouseCursor.BUTTON :String = “button”`
- `flash.ui.MouseCursor.HAND :String = “hand”`
- `flash.ui.MouseCursor.IBEAM :String = “ibeam”`

mouseIdx プロパティ

```
mouseIdx :uint  [read]
```

Scaleform のバージョン : 4.0.13

イベントがどのマウス/コントローラーに対して生成されたかを示します（ゼロ基準のインデックス）。

11 MouseEventEx エクステンション

```
package scaleform.gfx
{
    import flash.events.MouseEvent;

    public final class MouseEventEx extends MouseEvent
    {
        public var mouseIdx : uint = 0;
        public var nestingIdx : uint = 0;
        public var buttonIdx : uint = 0; // LEFT_BUTTON, RIGHT_BUTTON, ...

        public static const LEFT_BUTTON : uint  = 0;
        public static const RIGHT_BUTTON : uint = 1;
        public static const MIDDLE_BUTTON : uint = 2;

        public function MouseEventEx(type:String) { super(type); }
    }
}
```

このイベントは標準の flash.events.MouseEvent のエクステンションです。これは複数のコントローラーと右/中のマウスのボタンへのサポート用のプロパティを付加します。Extensions.enabled プロパティが true に設定されている場合、Scaleform は常に標準の MouseEvent の替わりに MouseEventEx イベントを生成します。受け取ったイベントが MouseEventEx のインスタンスかどうかをチェックして、その通りならばイベントオブジェクトをエクステンションタイプに型変換してください。例：

```
import scaleform.gfx.*;
Extensions.enabled = true;

stage.doubleClickEnabled = true;

function ev(e:MouseEvent):void
{
      trace("!!!!EVENT." + cnt++);
      trace("    eventType     = "+e.type);
      trace("    bubbles       = "+e.bubbles);
      trace("    eventPhase    = "+e.eventPhase);
      trace("    target        = "+e.target.name);
      trace("    currentTarget = "+e.currentTarget.name);
      if (e is MouseEventEx)
      {
            var ee:MouseEventEx = e as MouseEventEx;
            trace("    mouseIdx      = "+ee.mouseIdx);
```

```
            trace("    nestingIdx    = "+ee. nestingIdx);
            trace("    buttonIdx     = "+ee.buttonIdx);
      }
      trace(e);
}
stage.addEventListener(MouseEvent.MOUSE_DOWN, ev);
stage.addEventListener(MouseEvent.MOUSE_UP, ev);
stage.addEventListener(MouseEvent.CLICK, ev);
stage.addEventListener(MouseEvent.DOUBLE_CLICK, ev);
```

注：エクステンションがイネーブルされるとマウスイベントは右、中、センターなどのマウスボタンイベントに対して発せられ、ユーザーは buttonIdx プロパティをチェックしてどのボタンがイベントを生成したかを見つける必要があります。

buttonIdx プロパティ

```
buttonIdx :uint  [read]
```

Scaleform のバージョン：4.0.13

イベントがどのボタンに対して生成されたかを示します（ゼロ基準のインデックス）。この値は次の内の何れかです。

- MouseEventEx.LEFT_BUTTON：uint = 0　　- 左マウスボタン
- MouseEventEx.RIGHT_BUTTON：uint = 1　　- 右マウスボタン
- MouseEventEx.MIDDLE_BUTTON：uint = 2　　- 中マウスボタン
- マウスにボタンが 4 つ以上ある場合は、2 よりも大きいどのような値でも正しいと認められます。

mouseIdx プロパティ

```
mouseIdx :uint  [read]
```

Scaleform のバージョン：4.0.13

イベントがどのマウス/コントローラーに対して生成されたかを示します（ゼロ基準のインデックス）。

nestingIdx プロパティ

```
nestingIdx :uint  [read]
```

Scaleform のバージョン：4.0.13

このプロパティは rollOver/Out、mouseOver/Out、dragOver/Out イベントにはオプションです。このパラメーターは同じキャラクター上をネストされるロールオーバーまたはドラッグオーバーのイベントのインデックスを指定します。
ネストされた rollOver/rollOut、mouseOver/Out、dragOver/dragOut イベントが生成されると（各マウスカーソル別々に）、このパラメーターはネスティングのゼロに基づいたインデックスを表します。つまり、最初のイベントはパラメーターとして 0 を持ち、 2 つめのカーソルが同じキャラクター上にロールオーバーする場合は 2 つ目の rollOver/rollOut イベントが、数を 1 に設定して発せられます。どれかのカーソルがキャラクターを離れると rollOut/mouseOut/dragOut がメンバーを 1 に設定して発せられ、最後の rollOut/mouseOut/dragOut イベントはメンバーを 0 に設定して発せられます。

12 TextEventEx エクステンション

```
package scaleform.gfx
{
    import flash.events.TextEvent;

    public final class TextEventEx extends TextEvent
    {
        public var controllerIdx : uint = 0;

        public function TextEventEx(type:String) { super(type); }
    }
}
```

このイベントは標準の flash.events.TexEvent のエクステンションです。これはイベントを起こしたコントローラーのゼロに基づいたインデックス「controllerIdx」メンバーを追加します。
Extensions.enabled プロパティが true に設定されている場合、Scaleform は常に標準の TexEvent の替わりに TexEventEx イベントを生成します。受け取ったイベントが TexEventEx のインスタンスかどうかをチェックして、その通りならばイベントオブジェクトをエクステンションタイプに型変換してください。例：

```
import scaleform.gfx.*;
import flash.events.TextEvent;

Extensions.enabled = true;

function ev(e:TextEvent)
{
      if (e is TextEventEx)
      {
            var ee:TextEventEx = e as TextEventEx;
            trace("    controllerIdx = "+ee.controllerIdx);
      }
}
txf.addEventListener(TextEvent.TEXT_INPUT, ev);
```

controllerIdx プロパティ

```
controllerIdx :uint  [read]
```

Scaleform のバージョン：4.0.12

イベントに対してどのキーボード/コントローラーが使用されているかを示します（ゼロ基準のインデックス）。

LINK_MOUSE_OVER/LINK_MOUSE_OUT イベント

```
public static const LINK_MOUSE_OVER:String = "linkMouseOver";
public static const LINK_MOUSE_OUT:String = "linkMouse";
```

Scaleform のバージョン：4.0.14

このバージョンではマウスの over/out イベントを TextField リンク（HTML タグで指定）で、TextFieldEx.LINK_MOUSE_OVER と TextFieldEx.LINK_MOUSE_OUT イベントを使用してリスンできるようになりました。これらのイベントは他の AS3 イベントのように動作し、addEventListener パラダイムを使用してリスンできます。

13 System エクステンション

actionVerbose スタティックプロパティ

```
actionVerbose:Boolean  [read-write]
```

Scaleform のバージョン：4.0.12

opcode 追跡のイネーブル/ディスエーブル。

getStackTrace()スタティックの方法

```
public function getStackTrace() :String
```

Scaleform のバージョン：4.0.12

現在のスタックのトレースを、文字列としてフォーマットして取得します。

getCodeFileName()スタティックの方法

```
public function getCodeFileName() :String
```

Scaleform のバージョン：4.0.12

現在実行中のコードのファイル名を取得します。

14 TextFieldEx エクステンション

appendHtml()スタティックの方法

```
public function appendHtml(textField:TextField, newHtml:String) : void
```

Scaleform のバージョン：4.0.12

newHtml パラメーターで指定した HTML をテキストフィールドの最後にあるテキストに付加します。この方法は htmlText プロパティの付加割り当て（+=）よりもより効率的です（例えば txt.htmlText += moreHtml）。htmlText プロパティに付加した標準の+=は HTML 文字列を作り、新しい HTML 部分を付加し、HTML 全体をはじめから解析します。この関数は HTML の増分解析を行います。つまり、これは HTML を newHtml 文字列のパラメーターから解析します（これが、+=演算子とは異なって、newHtml パラメーター中の HTML がうまく作られている必要のある理由です）。大量の内容を含むテキストフィールドにはこれがとりわけ重要です。

注：この方法はスタイルシートがテキストフィールドに適用されていなければ使用できません。

パラメーター
`textField:TextField` – HTML に付加するテキストフィールド。
`newHtml:String` - 既存のテキストに付加する HTML の付いた文字列。

setIMEEnabled()スタティックの方法

```
static public function setIMEEnabled(textField:TextField, isEnabled:Boolean): void
```

Scaleform のバージョン：4.0.12

指定したテキストフィールドに対して IME をイネーブル/ディスエーブルします。isEnabled が false に設定されている場合、このテキストフィールドでは IME はアクティブになりません。デフォルトでは IME はイネーブルされています。

パラメーター
`textField :TextField` – 作業すべきテキストフィールド。
`isEnabled :Boolean` – If true – IME はイネーブルされています。それ以外の場合はディスエーブルされています。

setVerticalAlign()スタティックの方法

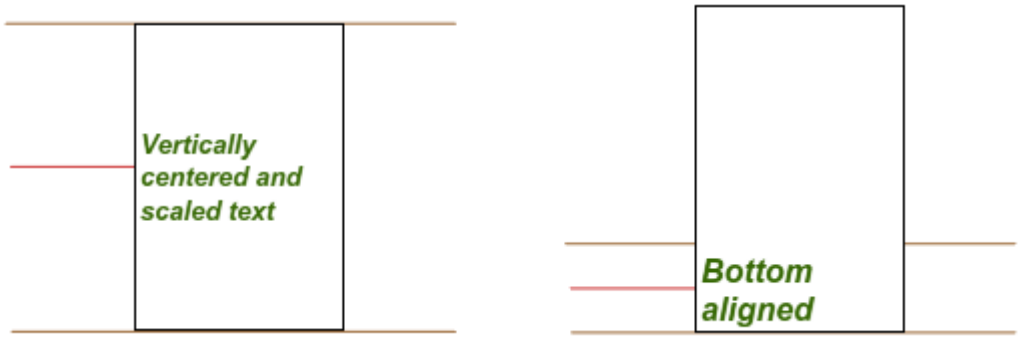
```
public function setVerticalAlign(textField:TextField, valign:String) : void
```

Scaleform のバージョン：4.0.12

テキストボックス内部でのテキストの垂直方向の整列を設定します。このプロパティに対する有効な値は次の定数で、TextFieldEx クラスで宣言します。

```
public static const VALIGN_TOP:String     = "top";
public static const VALIGN_CENTER:String  = "center";
public static const VALIGN_BOTTOM:String  = "bottom";
```

プロパティが center に設定されている場合、テキストはテキストボックスの中央に置かれます。プロパティが bottom に設定されている場合はテキストはテキストボックスの下部に置かれます（下図参照）。



デフォルト値は top です。

パラメーター

`textField:TextField` - 垂直整列を設定するテキストフィールド

`valign:String` - 整列値 ("top"、"center"、"bottom")。