

Flash Article

## Controlling Flash video with FLVPlayback programming の日本語訳

本ドキュメントは、Adobe 社のサイトにある“Controlling Flash video with FLVPlayback programming”をヒム・カンパニー 永井勝則が自主的に日本語に訳したものです。

<http://www.himco.jp>

[knagai@himco.jp](mailto:knagai@himco.jp)

(2007/9/14)

本ドキュメントの原文は

[http://www.adobe.com/devnet/flash/articles/flvplayback\\_programming.html](http://www.adobe.com/devnet/flash/articles/flvplayback_programming.html)

です。

## FLVPlayback プログラミングによる Flash Video の制御

Dan Carr

Dan Carr Design

Flash CS3 には新しい ActionScript 3.0 FLVPlayback ユーザーインターフェイスコンポーネントのセットが含まれています。ActionScript 3.0 コンポーネントは、標準化されたコーディング方法と、新しい ActionScript 仮想マシン (AVM2) の使用による向上した実行性能を提供します。ActionScript 3.0 コンポーネントを使用するには、コンポーネントとイベントの使用方法を少し変える必要がありますが、そのメリットにはビデオとユーザーインターフェイスコンポーネント間に一貫性ができたこと、スキン操作とカスタマイズが簡単になったこと、そして制御とエラー処理の範囲が大幅に広がったことがあります。

この記事では、ActionScript 3.0 FLVPlayback コンポーネントのカスタム操作に必要な基本的なコードの概略を述べます。ActionScript 3.0 ファイルではビヘイビアを通じた自動的なコード配置は使用できないので、この記事のコードのように少し手動で変更する必要があります。コードサンプルの使用とカスタマイズの簡単な説明によって、複数のビデオのロード、スタート、シーク、ストップが行える自作コントロールが作成できるようになります。

**Note:** FLVPlayback コンポーネントの ActionScript 3.0 バージョンは、Flash で ActionScript 3.0 ファイルが有効になっているときのみ、コンポーネントパネルで使用できます。Flash CS3 で既存の ActionScript 2.0 ファイルかそれ以下のファイルを開くと、コンポーネントパネルには Flash MX 2004 のメディアコンポーネントとほぼ同じに ActionScript 2.0 の FLVPlayback コンポーネントが表示されます。ActionScript 2.0 ファイルでは ActionScript 3.0 コンポーネントを動作させることはできず、その逆も同様です。意図したファイルバージョンで意図したコンポーネントを使っているかを必ず確認してください。

Flash CS3 Professional で ActionScript 3.0 を使用している場合は、このまま本記事を読み進め、最新のビデオコンポーネント技術を扱ってみてください。

Flash Professional 8 を使っているか、Flash CS3 Professional で ActionScript 2.0 ファイルを使っている場合は、本記事の前バージョン、「[FLVPlayback ビヘイビアを使った Flash Video の制御](#)」をご覧ください。Flash MX Professional 2004 を使っている場合は、本記事の前バージョン、「[Controlling Media Components with Flash Behaviors](#)」をご覧ください。

### 必要な環境

このチュートリアルを完了するには以下のソフトウェアとファイルが必要です。

## Flash CS3 Professional

### サンプルファイル

flvplayback\_samples.zip (ZIP、5MB)

### 必要な予備知識

本記事ではこれまでに Flash Video をエンコードした経験があり、FLVPlayback コンポーネントの基本的な使い方を理解していることが前提となります。追加的な情報が必要な場合は、以下のセクションで示すリソースを参照してください。では始めましょう。

### スタート

このセクションでは、ActionScript 3.0 を使ったビデオ再生のカスタマイズを始めるに当たって知っておく必要のある事柄の要点を説明します。すでにビヘイビアをインストールして ActionScript 3.0 を使って開発し、Flash Video と FLVPlayback コンポーネントを操作した経験のある方は、ここを飛ばして先へ進むこともできます。それ以外の方は、FLVPlayback コンポーネントをプログラミングで変更するのに必要な基礎をしっかりと身につけるため、以下に述べる情報をしっかり理解することが大切です。

### 始める前に

Flash Video のエンコーディングと Flash Video を表示する FLVPlayback コンポーネントの操作に関するドキュメンテーションをまだ読まれていない方は読んで理解してください。

[Flash video learning guide](#)

[Creating and Playing Flash Video Files Through Progressive Download](#)

本記事の説明にそって進めるには、ActionScript 3.0 FLVPlayback コンポーネントとコードサンプル用に提供している FLV ファイルが必要です。

### サンプルファイルの理解

提供しているサンプルファイルをダウンロードして展開すると、フォルダ内に短い FLV ファイルを制御する作業用コードサンプルが現れます。各ファイルはどれもメインのタイムラインに、assets と actions という名前のレイヤーがあります。assets レイヤーには display という名前の FLVPlayback コンポーネントがあり、actions レイヤーにはそのコ

ードがあります。サンプルファイルは、みなさんがご自分のプロジェクトにとりかかる際の基本として、作成し使用するファイルと比較するための目安としてお使いください。

## コードサンプルを実行するためのファイルの設定

開始するには、FLV ファイル、FLVPlayback コンポーネントのインスタンス、コードを記述した actions レイヤーが必要です。本記事のコードサンプルを使用するには以下の手順にしたがいます。

1. デスクトップかファイルを保存したい場所にフォルダを作成します。サンプルファイルのフォルダの FLV ファイルを作成したフォルダにコピーします。
2. Flash CS3 Professional で ActionScript 3.0 FLA ファイルを作成します。ファイルを手順1で作成したフォルダに、FLV ファイルとならべて保存します。
3. デフォルトレイヤーを assets という名前に変更します。
4. コンポーネントパネルを開き、Video フォルダから FLVPlayback コンポーネントのインスタンスをドラッグします。

**Note:** FLVPlayback コンポーネントやほかの UI コンポーネントをプログラミングで作成し配置することは可能ですが、そのコンポーネントはまず、コンポーネントパネルから現在の FLA ファイルのライブラリにコピーする必要があります。本記事の目的はそこからさらに一歩進めることにあるので、ここではまずステージに FLVPlayback コンポーネントをドラッグし、目で確認しながら配置することをおすすめします。

5. ステージにコンポーネントを配置し、自由変形ツールを使ってサイズを好きな大きさに調整します。
6. コンポーネントを選択した状態で、プロパティインスペクタでそのインスタンス名をつけます。

**Note:** この記事のサンプルでは FLVPlayback のインスタンス名は **display** にしています。このインスタンス名を使ってもけっこうですし、ビデオコンポーネントのインスタンス名に一致するようにコード内の名前を変更することもできます。

7. 新規レイヤーを作成し **actions** という名前をつけます。
8. actions レイヤーのフレーム1のキーフレームを選択してアクションパネルを開きます (F9)。
9. **この記事のコードサンプルかサンプルファイルのコードをコピーして、アクションパネルのテキストエディタにペーストします。**

注:

サンプルファイルのコードをそのままコピーしてもエラーが出ます。エラーを出さずに、一番簡単にビデオを再生するコ

ードは次のようになります。

```
import fl.video.*;

var flvControl = display;
var flvSource = "Call_to_Action.flv";
flvControl.source = flvSource;
```

10. コード内の変数 `flvControl` の値を使用するビデオコンポーネントのインスタンス名に変更します。また変数 `flvSource` の値を使用する FLV ファイルの URL に変更します。
11. SWF を書き出すかムービーをパブリッシュして結果を確認します。

### 作業ファイルの理解

上記手順を一度実行すると、FLV ファイル、場合によっては HTML ファイル、元になる FLA ファイル、SWF ファイル、スキン SWF ファイル(スキンを使っている場合)が手に入ります。プログレッシブビデオを配置するには、FLA ファイルをのぞくこれらのファイルをすべてサーバーにアップロードします。

ActionScript 3.0 FLVPlayback API に関する詳細は、Flash CS3 のヘルプページをご覧ください。

### ビデオのロード、プリロード、表示

プログラミングでビデオをロードする出発点はこれまでと変わりありません。コンポーネントのパラメータを使う場合にはただ FLV ファイルの URL を設定します。これは Flash の前のバージョンで行っていた方法と同じです。しかし Flash 8 で `contentPath` と呼んでいたプロパティは ActionScript 3.0 コンポーネントでは `source` プロパティに名前が変わりました。

このセクションでは以下のトピックを扱います。

- `fl.video` パッケージのインポート
- ビデオファイルのロードと `autoPlay` プロパティの設定
- プリロードの処理
- プログレスバーコンポーネントの追加

サンプルで扱うファイルの設定方法がよく分からない場合は本記事のスタートセクションをご覧ください。

## fl.video パッケージのインポート

パッケージは ActionScript 3.0 プログラミングを通して Flash ムービーに機能を提供するクラスのグループのことです。通常パッケージのインポートはスクリプトの最初で行います。これによってスクリプトがそのパッケージ内のクラス名に直接アクセスできるようになります。

ビデオのパッケージをインポートするには、コンポーネントのインスタンスを含むタイムラインのキーフレームに以下のコードを記述します。

```
import fl.video.*;
```

**Note:** アスタリスクを使用すると、Flash のコンパイラはそのパッケージ内のクラスをすべてインポートすることになります。アスタリスクは、この場合でいうとビデオのパッケージ内のすべてのパッケージをインポートする、ワイルドカード文字のようなものです。パッケージ内の全クラスをインポートしたくない場合は、たとえば `fl.video.FLVPlayback` のように、特定のクラスパスをインポートします。

## ビデオのロードと `autoPlay` プロパティの設定

ビデオのロードは、コンポーネントの `source` プロパティを FLV ファイルの URL に設定するのと同じように簡単です。同時に `autoPlay` プロパティを設定すると、ロードしたビデオの動作を制御できるようになるので便利です。`autoPlay` を `false` に設定すると、その表示の準備が整ってもビデオの再生はスタートしません。

ビデオをロードし `autoPlay` プロパティを `false` に設定するには、記述した `import` コードの後に以下のコードを追加します。`display` は `FLVPlayback` コンポーネントのインスタンス名です。

```
import fl.video.*;

display.autoPlay = false;
display.source = "Call_to_action.flv";
```

## プリロードの処理

`FLVPlayback` コンポーネントは、表示の準備が整うまでプログレッシブビデオの再生を一時停止します。プリロードを処理する通常の方法には、`ready` イベントが発生したら自動的に再生を開始する方法と、ビデオを一時停止しておいて十分にロードされるまで待つ方法です。ビデオ再生を滑らかにするキーの1つは、`FLVPlayback` コンポーネントからのタイミングイベントの配信を監視することです。

ビデオの ready イベントにตอบสนองするには、同じキーフレームに以下のコードを追加します。

```
function readyHandler(event:VideoEvent):void {
    display.pause();
    display.playWhenEnoughDownloaded();
}
display.addEventListener(VideoEvent.READY, readyHandler);
```

Note: このコードは、ready イベントの間、ムービー（ビデオ）が中断されることなく最後まで再生されるのに十分なだけダウンロードされない限り、ムービーを一時停止します。

おまけ: Flash ヘルプの“playWhenEnoughDownloaded”の引用

FLV ファイルを十分にダウンロードした後で再生します。FLV ファイルのダウンロードが完了している場合、または Flash Media Server (FMS) からストリーミングしている場合、playWhenEnoughDownloaded() メソッドを呼び出すことは、パラメータを指定しない play() メソッドと同じです。このメソッドを呼び出しても再生は一時停止されないで、多くの場合、このメソッドを呼び出す前に pause() メソッドを呼び出す必要があります。

## プログレスバーコンポーネントの追加

プログレッシブビデオを使用しているときは通常、ビデオのロード中、ビデオの再生が始まるまで、プログレスバーを表示するようにします。これは ProgressBar UI コンポーネントを使用することで簡単に実現できます。作業サンプルを作成するには次の手順にしたがいます。

プログレスバーを追加するには:

1. 新規 ActionScript 3.0 FLA ファイルを作成しデフォルトレイヤーの名前を assets に変更します。
2. コンポーネントパネルからステージに FLVPlayback コンポーネントのインスタンスをドラッグします。コンポーネントの位置とサイズを調整します。
3. プロパティインスペクタでインスタンス名を **display** にします。
4. コンポーネントパネルからステージに ProgressBar コンポーネントのインスタンスをドラッグします。コンポーネントの位置とサイズを調整します。
5. プロパティインスペクタでインスタンス名を **pb** にします。
6. 新規レイヤーを作成しレイヤー名を **actions** にします。
7. actions レイヤーのフレーム1のキーフレームを選択しアクションパネルを開きます。以下のコードを記述します。

```

import fl.video.*;
import fl.controls.ProgressBarMode;

// 変数の設定
var flvControl = display;
var flvSource = "Call_to_Action.flv";

//プログレスバーを制御するイベントハンドラ関数の作成
function progressHandler(event:VideoProgressEvent):void {
    var bl = Math.round(event.bytesLoaded/1000);
    var bt = Math.round(event.bytesTotal/1000);

    // 進行状況の更新
    pb.setProgress(bl,bt);
}

function readyHandler(event:VideoEvent):void {
    //再生が始まったらプログレスバーを削除する
    removeChild(pb);
}

//プログレスバーの状態の設定
pb.mode = ProgressBarMode.MANUAL;
pb.indeterminate = false;

//リスナーの追加とビデオのロード
flvControl.addEventListener(VideoProgressEvent.PROGRESS, progressHandler);
flvControl.addEventListener(VideoEvent.READY, readyHandler);
flvControl.source = flvSource;

```

8. flvSource 変数を使用する FLV ファイルの URL に一致するように変更します。
9. SWF を書き出して結果を確認します。

プログレスバーを使用するもう1つの方法は、ビデオが自動的に再生されないよう停止しておいて、進行状況のハンドラ関数で bytesLoaded と bytesTotal の値を監視します。ロードされたバイトのパーセンテージにもとづいてプ

ログレスバーを削除しビデオを再生する方法です。

**Note:** ActionScript 3.0 は FLVPlayback コンポーネントからのランタイムエラーを例外として報告します。例外は ActionScript の try...catch ステートメントを使ってとらえ処理します。ビデオの例外に関する詳細は、Flash CS3 ヘルプページの VideoError クラスをご覧ください。

作業サンプルを見るには、本記事の最初にリンクされているサンプルの ZIP ファイル内の flvplayback\_programming1.flx をご覧ください。

## ビデオの再生、一時停止、停止

ビデオの再生、一時停止、停止は、ビデオのユーザーインターフェイスをプログラミングするときに必要な非常によく使用されるコントロールです。これらのコントロールには、Flash CS3 Professional に含まれる FLVPlayback カスタム UI コンポーネントを通してアクセスします。カスタム UI コンポーネントの使用は簡単ですが、FLVPlayback のコマンドにプログラミングを通して直接アクセスするときには、次のセクションでお見せするように、何度も必要になります。

このセクションでは以下のトピックを取り上げます。

- 簡単なコマンド
- ループ
- 再生と一時停止のトグル
- 停止とリセット

## 簡単なコマンド

おそらくみなさんは、ビデオを使用するとき、ビデオに対するほとんどのカスタム機能は、発生するイベントを中心に展開することに気づかれるでしょう。しかしその中核をなす部分ではすべて、以下に示すような簡単な多くのコマンドも使用することになります。各コマンドは FLVPlayback のインスタンス名で始まっていることに注意してください。この例では、**display** というインスタンス名で始まります。

play コマンドを適用するには以下のコードを使用します。

```
display.play();
```

play コマンドは、source、totalTime、isLive という3つのパラメータをサポートするようになりました。この値は以

下のコードに示すように初期化できます。

```
display.play("videos/myVideo.flv", 32, false);
```

pause コマンドを適用するには以下のコードを使用します。

```
display.pause();
```

stop コマンドを適用するには以下のコードを使用します。

```
display.stop();
```

### ループ(連続再生)

ビデオのループは非常に便利な機能です。以下は、ビデオの complete イベントを監視し、最初から再生を再開する方法の手順です。

ループするビデオを作成するには：

1. 新規 ActionScript 3.0 FLA ファイルを作成しデフォルトレイヤーの名前を **assets** に変えます。
2. コンポーネントパネルからステージに FLVPlayback コンポーネントのインスタンスをドラッグします。コンポーネントの位置とサイズを調整します。
3. プロパティインスペクタでインスタンス名を **display** にします。
4. 新規レイヤーを作成しレイヤー名を **actions** にします。
5. actions レイヤーのフレーム1のキーフレームを選択しアクションパネルを開きます。以下のコードを記述します。

```
import fl.video.*;

//ビデオコンポーネントのインスタンス名
var flvControl = display;
var flvSource = "Call_to_Action.flv";

//再生が完了したらビデオをループする
function completeHandler(event:VideoEvent):void {
    flvControl.seek(0);
    flvControl.play();
}
```

```
}  
flvControl.addEventListener(VideoEvent.COMPLETE, completeHandler);  
  
// ビデオの設定  
flvControl.source = flvSource;
```

6. flvSource 変数を使用する FLV ファイルの URL に一致するように変更します。
7. SWF を書き出して結果を確認します。

### 再生と一時停止のトグル(オン/オフの切り替え)

FLVPlayback の ActionScript API を使用すると、カスタムボタンとコントロールが作成できます。以下に、カスタムボタンを使って再生のオンとオフをトグルするコードを記述する手順を示します。

1. 前のサンプルで使ったファイルで作業をつづけます。assets レイヤーを選択します。
2. ボタンシンボルを作成し、そのインスタンスを display の下に置きます。
3. プロパティインスペクタでボタンのインスタンス名を toggle\_btn にします。
4. actions レイヤーのフレーム1のキーフレームを選択しアクションパネルを開きます。前のコードの下に以下のコードを記述します。

```
// トグルするコードの追加  
function toggleHandler(event:MouseEvent):void {  
    if ( flvControl.playing ) {  
        flvControl.pause();  
    } else {  
        flvControl.play();  
    }  
}  
  
toggle_btn.addEventListener(MouseEvent.CLICK, toggleHandler);
```

5. SWF を書き出して結果を確認します。

### 停止とリセット

ビデオのループ機能を加えたので、ビデオ再生の停止方法を加えるのはよい考えです。以下のコードでは、ループ

を止めるための停止とリセット(巻き戻しと停止)の方法を示します。

停止とリセットを行うには:

1. 前のサンプルで使ったファイルで作業をつづけます。assets レイヤーを選択します。
2. ボタンシンボルを作成し、そのインスタンスを display の下に置きます。
3. プロパティインスペクタでボタンのインスタンス名を `reset_btn` にします。
4. actions レイヤーのフレーム1のキーフレームを選択しアクションパネルを開きます。前のコードの下に以下のコードを記述します。

```
// リセットコードの追加
function resetHandler(event:MouseEvent):void {
    flvControl.autoRewind = true;
    flvControl.stop();
}

reset_btn.addEventListener(MouseEvent.CLICK, resetHandler);
```

5. SWF を書き出して結果を確認します。

ここまでの作業サンプルは、本記事の最初にリンクされているサンプルの ZIP ファイルに含まれる `flvplayback_programming2.fla` ファイルに記述されています。

## ビデオ内へのシーク(移動)

ビデオ内にキューポイントを含めることで、ビデオ内の指定した場所にシークできるナビゲーションポイントを割り当て、Flash ムービー内のコンテンツにその場所を同期させることができます。まだ読まれていない場合は、キューポイントの作成で可能になるオプションを読んでください。一番簡単な方法はコンポーネントインスペクタの `cuePoints` パラメータを使う方法ですが、ビデオのエンコーディング中にキューポイントを埋め込んだり、ActionScript を使ってダイナミックにキューポイントを作成する方法もあります。

このセクションでは以下のトピックを取り上げます。

- キューポイント名へのシーク
- 時間へのシーク
- パーセントへのシーク
- キューポイントイベントの処理

**Tip:** プログレッシブファイル内の指定した時間へのシークは、エンコーディング時にナビゲーションキューポイントが埋め込まれている場合のみ正確に動作します。ナビゲーションキューポイントは、FLV に対して、そのキューポイントが追加されたビデオ内の場所のキーフレームを強制的に作成します。これは、ビデオファイルのナビゲーションを定義するときには、その映像内の正確な位置に再生ヘッドを移動させるために必要です。

訳者注: エンコーディング時にキューポイントを埋め込むとそのポイントは必ずキーフレームになるので、ビデオの再生時、そのポイントに正確に再生ヘッドを移動させることができます。これに対し、cuePoints パラメータや ActionScript で作成するキューポイントはあくまでもそのポイントに一番近いキーフレームに移動することになるので、精度は埋め込む方法よりも劣ります。

ナビゲーションキューポイントによって印づけされた正確な位置を越えるシークでは、プログレッシブダウンロードは正確にはシークしません。ビデオ内のキーフレームに関連づけられていない時間にスクラブ(前後の細かく移動)したりシークすると、再生ヘッドは一番近いキーフレームにジャンプします。すべての状況で正確にシークするには、Flash Media Server が提供するストリーミングビデオを使用する必要があります。

### キューポイント名へのシーク

キューポイント名を使用すると、埋め込まれたナビゲーションキューポイントに正確にシークできます。以下は、”introduction”という名前のキューポイントにシークするボタンを作成するコードの作成方法です。

1. 新規 ActionScript 3.0 FLA ファイルを作成しデフォルトレイヤーの名前を **assets** に変えます。
2. コンポーネントパネルからステージに FLVPlayback コンポーネントのインスタンスをドラッグします。コンポーネントの位置とサイズを調整します。
3. プロパティインスペクタでインスタンス名を **display** にします。
4. ボタンシンボルを作成し、そのインスタンスを display の下に置きます。
5. プロパティインスペクタでボタンのインスタンス名を **seek\_btn** にします。
6. 新規レイヤーを作成しレイヤー名を **actions** にします。
7. actions レイヤーのフレーム1のキーフレームを選択しアクションパネルを開きます。以下のコードを記述します。

```
import fl.video.*;

// ビデオコンポーネントのインスタンス名
var flvControl = display;
var flvSource = "Call_to_Action.flv";
```

```
// ビデオの設定
flvControl.source = flvSource;

//ボタンへのシーク機能の追加
function seekHandler(event:MouseEvent):void {
    flvControl.seekToNavCuePoint("introduction");
}
seek_btn.addEventListener(MouseEvent.CLICK, seekHandler);
```

8. コード内の introduction というキューポイント名を、シークしたいキューポイント名に変えます。
9. SWF を書き出して結果を確認します。

### 時間へのシーク

またビデオ内の指定した時間へのシークも可能です。

1. 前のサンプルで使ったファイルで作業をつづけます。assets レイヤーを選択します。
2. ボタンシンボルを作成し、そのインスタンスを display の下に置きます。
3. プロパティインスペクタでボタンのインスタンス名を seekToTime\_btn にします。
4. actions レイヤーのフレーム1のキーフレームを選択しアクションパネルを開きます。前のコードの下に以下のコードを記述します。

```
// タイムコードへのシークの追加
function seekToTimeHandler(event:MouseEvent):void {
    var sec = 2;
    flvControl.seek(sec);
}
seekToTime_btn.addEventListener(MouseEvent.CLICK, seekToTimeHandler);
```

5. コード内の sec 変数の値を移動したい時間に変えます。
6. SWF を書き出して結果を確認します。

### パーセントへのシーク

パーセントを指定することでその割合の場所へシークすることができます。

1. 前のサンプルで使ったファイルで作業をつづけます。assets レイヤーを選択します。
2. ボタンシンボルを作成し、そのインスタンスを display の下に置きます。
3. プロパティインスペクタでボタンのインスタンス名を seekToPercent\_btn にします。
4. actions レイヤーのフレーム1のキーフレームを選択しアクションパネルを開きます。前のコードの下に以下のコードを記述します。

```
// パーセントコードの追加
function seekToPercentHandler(event:MouseEvent):void {
    var percent = 20;
    flvControl.seekPercent(percent);
}
seekToPercent_btn.addEventListener(MouseEvent.CLICK, seekToPercentHandler);
```

5. コード内の sec 変数の値を移動したい時間に変えます。
6. SWF を書き出して結果を確認します。

**Tip:** パーセントへのシーク動作は、metadata を持たない古い FLV ファイルを使う場合には動作しません。その場合には、Flash CS3 Video Encoder を使ってビデオをもう一度 FLV 形式にエンコードすることでこの問題を回避できます。

## キューポイントイベントの処理

シークに関連する発展事項にキューポイントの処理があります。その場合埋め込みキューポイントはプログレッシブビデオでナビゲーションを成功させるためのキーになります。このテクニックは、ユーザーインターフェイス内の要素をビデオと同期させるためによく使用されます。以下は、ビデオのキューポイントを監視し、キューポイント名に対応するフレームラベルにナビゲートする例です。

1. 前のサンプルで使ったファイルで作業をつづけます。assets レイヤーを選択します。
2. actions レイヤーのフレーム1のキーフレームを選択しアクションパネルを開きます。前のコードの下に以下のコードを記述します。

```
//キューポイントハンドラコードの追加
function cuePointHandler(event:MetadataEvent):void {
    trace("name = "+event.info.name);
    trace("time = "+event.info.time);
}
```

```
//キューポイント名をもったフレームラベルへの移動
gotoAndStop(event.info.name);
}

flvControl.addEventListener(MetadataEvent.CUE_POINT, cuePointHandler);
```

3. SWF を書き出して結果を確認します(ただしこのまま実行してもエラーが出ます。移動先のフレームラベルを作成する必要があります)。

ここまでの作業サンプルは、本記事の最初にリンクされているサンプルの ZIP ファイルに含まれる flvplayback\_programming3 fla ファイルに記述されています。

### フルスクリーンモードの処理とコントロールのレイアウト

Flash CS3 の新しいビデオ機能の1つに、フルスクリーンモードで FLV を再生できる機能があります。ボタンをクリックすると、SWF が画面いっぱいに広がり、中央で拡大されたビデオが表示されます。ActionScript を使うと、ビデオのレイアウトや背景色、**スキンの遅延**を制御することができます。

フルスクリーン機能を使用するには、FLA ファイルのパブリッシュ設定と FLVPlayback カスタム UI コンポーネント、ActionScript を合わせて使う必要があります。

フルスクリーンボタンを使ってビデオのユーザーインターフェイスを設定するには以下の手順にしたがいます。

1. 新規 ActionScript 3.0 FLA ファイルを作成しデフォルトレイヤーの名前を **assets** に変えます。
2. コンポーネントパネルからステージに FLVPlayback コンポーネントのインスタンスをドラッグします。コンポーネントの位置とサイズを調整します。
3. プロパティインスペクタでインスタンス名を **display** にします。
4. コンポーネントパネルからステージに FullScreenButton コンポーネントのインスタンスをドラッグします。ボタンの位置とサイズを調整します。
5. ボタンのインスタンス名を **fullScrn\_btn** にします。
6. 新規レイヤーを作成しレイヤー名を **actions** にします。
7. actions レイヤーのフレーム1のキーフレームを選択しアクションパネルを開きます。以下のコードを記述します。

```
import fl.video.*;
```

//1.ビデオコンポーネントのインスタンス名

```
var flvControl = display;
```

```
var flvSource = "Call_to_Action.flv";
```

//2. ビデオのパラメータの設定

```
flvControl.fullScreenButton = fullScrn_btn;
```

```
flvControl.source = flvSource;
```

8. flvSource のパスを使用するビデオに一致させます。
9. SWF をプレビューします。この時点ではボタンはまだ動作しません。それは、フルスクリーン機能では、パラメータをフルスクリーンをサポートする HTML コード内に設定する必要があるからです。

フルスクリーンビデオを可能にする HTML パラメータを設定するには以下の手順にしたがいます。

1. 前のサンプルで使ったファイルで作業をつづけます。メインメニューから[ファイル]→[パブリッシュ設定]を選択します。
2. [パブリッシュ設定]ダイアログボックスの[HTML]タブをクリックし、[テンプレート]ポップアップメニューから[Flash のみ –フルスクリーンサポート]を選択します(図1 参照)



図1: [Flash のみ –フルスクリーンサポート]を選択します。

3. [パブリッシュ]ボタンをクリックして HTML と SWF ファイルをパブリッシュし、[OK]をクリックします。
4. ブラウザで作成された HTML ファイルを開き機能を確認します。フルスクリーンボタンをクリックするとビデオが画面いっぱいに表示されます。
5. フルスクリーンモードから抜けるにはキーボードの Esc ボタンを押します。

また FLVPlayback インスタンスの align と scaleMode プロパティを変更すると、拡大したりフルスクリーンしているときのレイアウトを変更することができます。

フルスクリーン表示で推奨されるデフォルトを適用するには以下の手順にしがいます。

1. 最後の手順まで行った FLA ファイルのコードに戻ります。
2. fullScreenButton の代入の行の下に以下のコードを追加します。

```
flvControl.align = VideoAlign.CENTER;  
flvControl.scaleMode = VideoScaleMode.MAINTAIN_ASPECT_RATIO;
```

3. SWF を書き出し、HTML ページ上で表示されるムービーを確認します。この設定を変えるとレイアウトの結果がどう変わるか試してみてください。通常、ビデオの幅と高さのプロパティは、そのレイアウトを大きく変えることとなります。

ビデオのレイアウト制御にさらに興味を持たれた方は、ヘルプページの FLVPlayback セクションで新しい registrationHeight、registrationWidth、registrationX、registrationY プロパティを調べてみてください。これらのプロパティを align や scaleMode プロパティと併用すると、伸縮やビデオからビデオへの切り替えなどで美での見え方を手直することができます。

ここまでの作業サンプルは、本記事の最初にリンクされているサンプルの ZIP ファイルに含まれる flvplayback\_programming4 fla ファイルに記述されています。

Note: フルスクリーンモードには、Flash Player 9.0.28.0 かそれ以降と、FLVPlayback コンポーネントの ActionScript 3.0 バージョン、フルスクリーン表示を正しく設定した HTML ファイルが必要です。

### ビデオのプレイリスト(複数のビデオの再生)の作成

ここまではユーザーインターフェイスにおいて1つのビデオを制御する方法を取り上げてきましたが、多くのプロジェクトではビデオのリストを操作する必要があります。よく使用される方法としては、外部の XML ファイル内にプレイリスト(テキストファイルとして編集できる)を作成するか、ActionScript でプレイリスト(Flashで編集できる)を作成

する方法があります。このサンプルでは、Flash 内に ActionScript の配列として作成するプレイリストを扱います。

ActionScript のプレイリストを作成し、複数のビデオをロードするには：

1. 新規 ActionScript 3.0 FLA ファイルを作成しデフォルトレイヤーの名前を **assets** に変えます。
2. コンポーネントパネルからステージに FLVPlayback コンポーネントのインスタンスをドラッグします。コンポーネントの位置とサイズを調整します。
3. プロパティインスペクタでインスタンス名を **display** にします。
4. 新規レイヤーを作成しレイヤー名を **actions** にします。
5. actions レイヤーのフレーム1のキーフレームを選択しアクションパネルを開きます。以下のコードを記述します。

```
import fl.video.*;

//ビデオコンポーネントのインスタンス名の変数定義
var flvControl = display;

//ビデオのプレイリストの作成
var videoList = ["video1.flv", "video2.flv", "video3.flv"];
var videoIndex = 0;
var loopAtEnd = true;

//ビデオ再生の完了処理(次のビデオのロード)
function completeHandler(event:MetadataEvent):void {
    // Get next item in list
    videoIndex++;

    //インデックスの確認
    if (
        videoIndex == videoList.length ) {
        if ( loopAtEnd ) {
            videoIndex = 0;
        } else {
            return;
        }
    }
}
```

```
    }  
  }  
  
  // 次のビデオのロード  
  flvControl.source = videoList[videoIndex];  
}  
  
flvControl.addEventListener(VideoEvent.COMPLETE, completeHandler);  
  
//デフォルトのビデオの設定(スタート)  
flvControl.source = videoList[0];
```

6. videoList をコンマで区切った、使用するビデオの URL のリストに書き換えます。各 URL スtring には引用符を忘れずに含めます。
7. loopAtEnd 変数は、最後のビデオの再生が終わったら再生を停止したい場合には false に設定します。
8. SWF を書き出し、結果を確認します。

**Note:** また複数のビデオのロードは、FLVPlayback コンポーネント内部で複数のビデオのインデックスを使用することも可能です。上記に示した方法はプログレッシブビデオを使うときメリットがあります。

ビデオのプレイリストを外部の XML ファイルを使ってロードする方法に関する詳細は、Flash CS3 ヘルプの [ActionScript 3.0 のプログラミング > ビデオの操作 > ビデオジュークボックス](#) をご覧ください。

ここまでの作業サンプルは、本記事の最初にリンクされているサンプルの ZIP ファイルに含まれる flvplayback\_programming4.flc ファイルに記述されています。