

Displaying Library Content is AS 3.0 の日本語訳

本ドキュメントは、KIRUPA.COM のサイトにある“Displaying Library Content is AS 3.0”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

(2007/10/8)

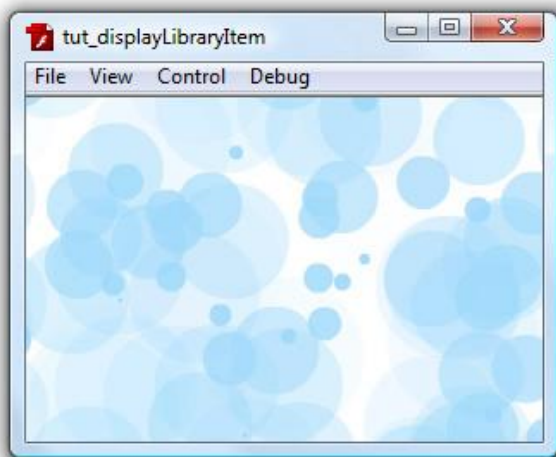
本ドキュメントの原文は

http://www.kirupa.com/developer/flashcs3/displayingContent_library_AS3_pg1.htm

です。

Flash のライブラリは、カスタムグラフィックやボタン、ムービークリップ、外部コンテンツなどすべてを保持するレポジトリ(入れ物)としての機能を果たします。デザインビューからライブラリにある要素にアクセスすることは非常に簡単です。ステージに置きたいものをただドラッグ & ドロップするだけです。しかし必要に応じてコードを使ってライブラリの項目を表示するよう求められることも多くあります。このチュートリアルではライブラリにあるコンテンツ(ライブラリコンテンツ)をステージに表示する場合を取り上げます。同時に Flash CS 3 の ActionScript 3.0 言語の使い方も学んでいきます。

このチュートリアルの最後では、以下の画像のようなものを作成します。ここではさまざまな円がステージに表示されていますが、これらはすべてコードを使って行ったものです。

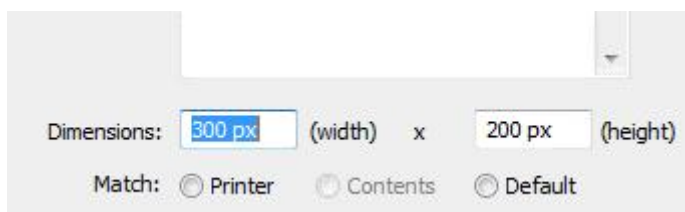


AS 2.0か AS 1.0 を使っていた方にとっては、AS 3 での大きな変更点の1つは `attachMovie` 関数がサポートされなくなったことです。以前ならステージに表示したいライブラリコンテンツは、`attachMovie` 関数を呼び出し、ライブラリ項目のリンケージ名を渡し、いくつかのデータを加えると、そのコンテンツは画面上に表示されました。AS 3 で使用する方法は、この方法と似たところもありますが、異なるところもあります。

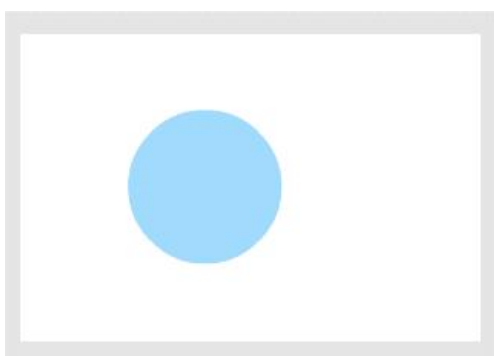
始めましょう

1. まず Flash CS3 で新規 FLA ファイルを作成します (AS 3.0)。プロパティインスペクタで [サイズ] の横にあるボ

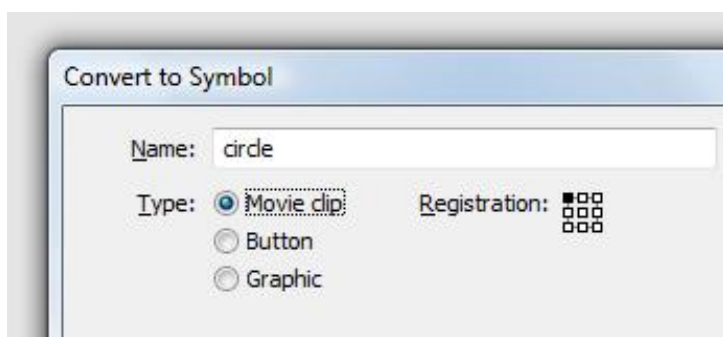
タンをクリックし、FLA ドキュメントの幅と高さをそれぞれ 300 ピクセルと 200 ピクセルに設定します。



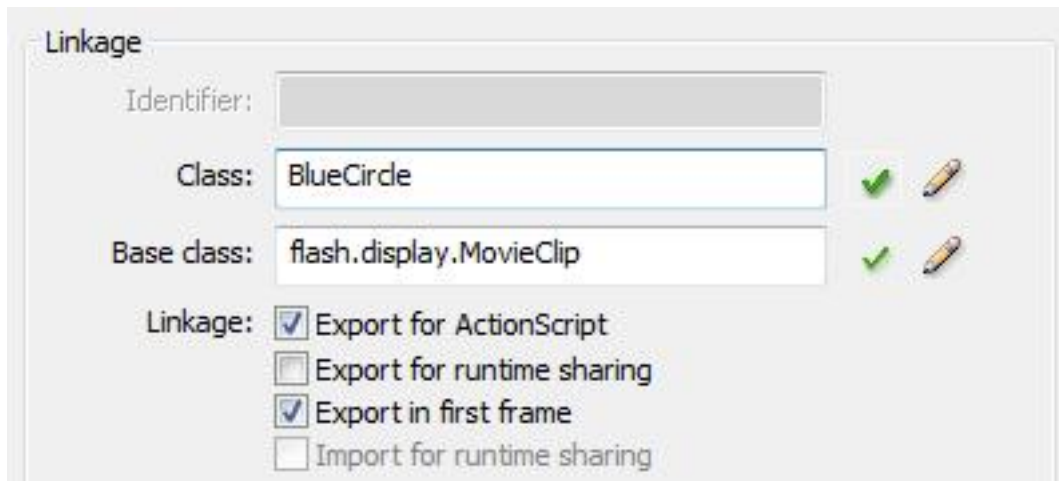
2. ステージの幅と高さが希望通りに設定できたので、円を描画します。[楕円]ツールで青い単色塗りの円を描きます。



3. 円を選択し F8 を押して [シンボルに変換] ダイアログボックスを表示させます。[名前]に **circle** を入力し、[タイプ]に [ムービークリップ] オプションが選択されていることを確認します。まだ [OK] はクリックしません。もう少し変更を加えます。

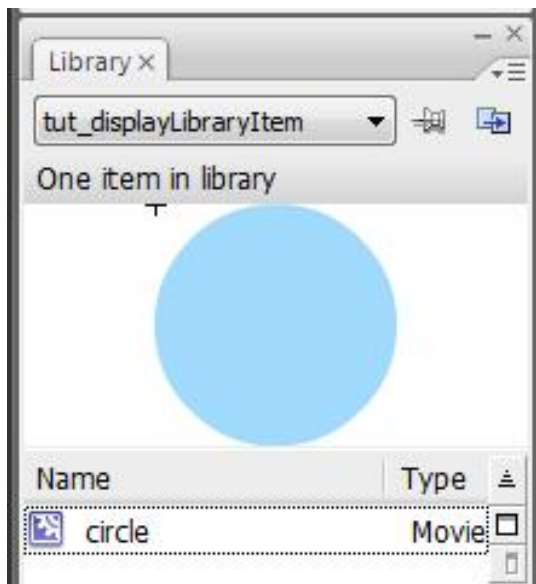


4. [シンボルに変換] ダイアログボックスで [リンケージ] 領域を探します。[リンケージ] 領域が表示されていない場合は、[詳細] ボタンをクリックします。[リンケージ] 領域で [ActionScript に書き出し] チェックボックスを選択します。その2つ上の [クラス] フィールドで表示されている文字 (circle) を **BlueCircle** に変更します。



[基本クラス]フィールドには自動的に値が入ります。もし入っていない場合は、`flash.display.MovieClip` と入力します。

5. [OK]ボタンをクリックしてダイアログボックスを閉じます。[OK]をクリックすると、ライブラリに新しく作成したシンボルが表示されます。

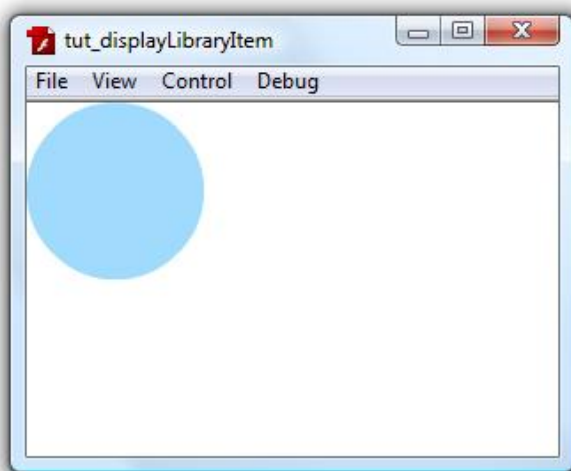


6. ここでは作成した円のムービークリップはライブラリに保持され、その同じクリップのコピーがステージ上にあります。ステージにある円のムービークリップを選択し Delete キーを押してそれを消去します。これでステージ上には何もなくなりました。
7. ではコードを記述しましょう。タイムラインの最初のフレームを右クリックし[アクション]を選択します。アクション

パネルが開きます。以下のコードを記述します。

```
function Main() {  
    this.addChild(new BlueCircle());  
}  
Main();
```

- ムービーをプレビューします。以下のように表示されます（作成したシンボルの基準点の位置によって青いムービークリップの位置は変わります）。



専門的な言い方をすると、ここでやっていることはライブラリからコンテンツを取り出しそれを表示することです。実際に行えることはもっとたくさんあるので、次ページでコードの説明とともに新しいティップスを紹介していきます。

page 2

前のページでは、ライブラリからロードした円を表示する小さな Flash アプリケーションを作成しました。このページでは、前のページで学んだことを拡張しクールにする方法を理解していきます。

ムービークリップのプロパティの調整

今のところわれわれの円はただ左上隅に表示されるだけです。もし円の位置などのプロパティを調整することができたとしたら素晴らしいことでしょう。そのためにはコードに戻る必要があります。コードを書いたフレームを右クリックして[アクション]を選びアクションパネルを開きます。

```
1 function Main() {
2     this.addChild(new BlueCircle());
3 }
4 Main();
```

今のところコードはごくわずかで、新しい BlueCircle ムービークリップを子要素として追加する Main 関数があるのみです。BlueCircle ムービークリップの後には2つのカッコ、(と)がつづいていることに注意してください。このカッコは、BlueCircle ムービークリップが実際にはクラスのように扱われていることを示しています。今のところは、クラスに不慣れでも心配いりません。

では Main 関数のコードを修正し、BlueClass に関連づけられたものすべての面倒をみる特別な変数を持たせます。今のコードを以下のコードに置き換えます。

```
function Main() {
    var newCircle:BlueCircle=new BlueCircle();
    this.addChild(newCircle);
}
Main();
```

アプリケーションを実行しても前と変わったところはありません。青い円は前と同じように左上隅に表示されます。大きな違いは、addChild 関数に new BlueCircle()を渡しているのはでなく、BlueCircle 型のオブジェクトへの参照を保持する newCircle という名前の変数を作成していることです。

その後は同じで addChild 関数に新しい BlueCircle オブジェクトへの参照を渡します。唯一異なるのは、前のように直接オブジェクトを渡すのではなく、そのオブジェクトを参照する変数を渡していることです。

別の変数に BlueCircle オブジェクトを参照させることで、この変数を使って実際に処理できる柔軟性が増します。では円の位置を x 位置が 50、y 位置が 75 に変更しましょう。コードは次のようになります。

```
newCircle.x=50;
newCircle.y=75;
```

このコードを追加したコード全体は次のようになります。

```
function Main() {  
    var newCircle:BlueCircle=new BlueCircle();  
    newCircle.x=50;  
    newCircle.y=75;  
  
    this.addChild(newCircle);  
}  
Main();
```

新しい行は this.addChild(newCircle)行の前に追加していますが、必ずしもこのようにしなくてはならないわけでは
ありません。この2行は addChild の後に置くこともでき、同じように動作します。

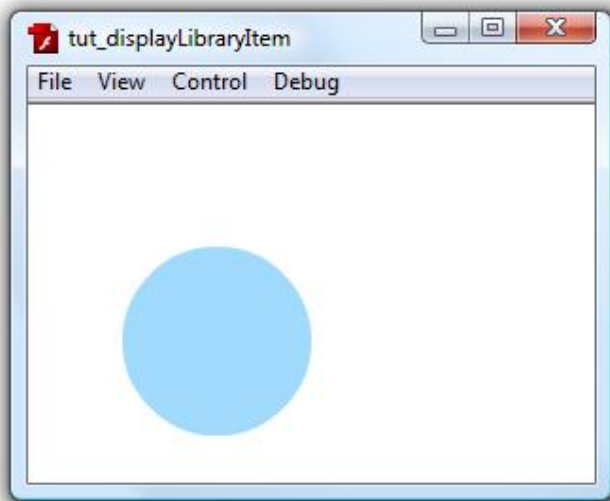
訳者注:

addChild()メソッドの後に x と y を指定する場合、次のコードを試すと 0 と 50 が出力されます。

```
function Main() {  
    var newCircle:BlueCircle=new BlueCircle();  
    this.addChild(newCircle);  
  
    trace(newCircle.x);  
  
    newCircle.x=50;  
    newCircle.y=75;  
  
    trace(newCircle.x);  
}  
Main();
```

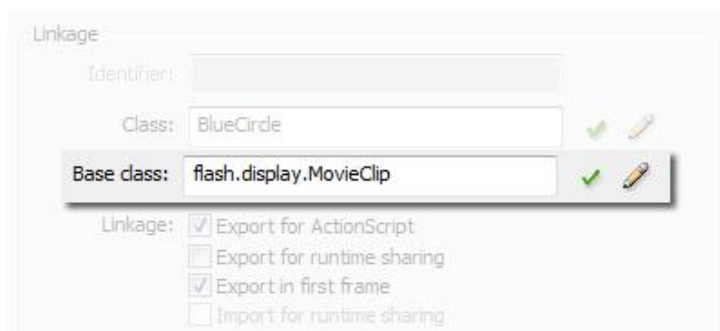
ここから、this.addChild(newCircle)で x=0 の位置に配置され、newCircle.x=50 で x=50 の位置に配置されているこ
とが分かります。

ムービーをプレビューすると、円は 50, 75 の位置に表示されることが分かります。



追加した2行に戻りましょう。ここでは円の x と y プロパティを設定するために newCircle オブジェクトをその x と y プロパティを使って移動させています。ここで x と y が使用できるのは、BlueCircle オブジェクトが実際には MovieClip の拡張であるからです。

青い円を MovieClip に変換したとき、[シンボルに変換]ダイアログボックスの[リンケージ]領域に[基本クラス]フィールドがあったことを覚えておられるでしょう。



[基本クラス]のテキストは、新しいシンボルは機能性を継承するソースの場所を示すものです。この場合には、BlueCircle クラスは、flash.display.MovieClip にある MovieClip クラスから派生しています。

これは、BlueCircle オブジェクトが MovieClip クラスのイベントやメソッド、x と y 以外のプロパティなど**すべてに**アクセスできるということを意味しています。これを確認するために今のコードを以下のコードと置き換えてみましょう。

```
function Main() {  
    for (var i:int=0; i<200; i++) {
```

```
var newCircle:BlueCircle=new BlueCircle();

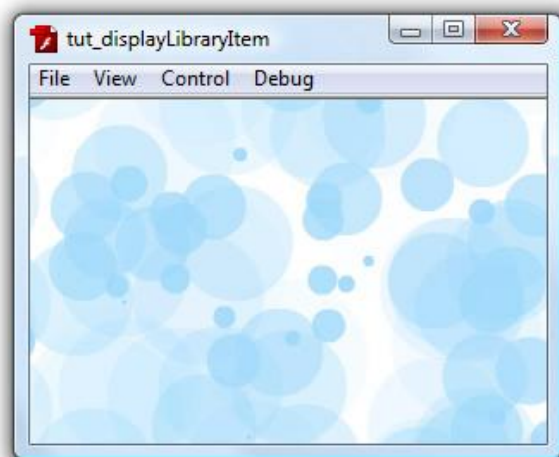
var randomValue:Number=Math.random()*1;

newCircle.x=-100+Math.random()*500;
newCircle.y=-100+Math.random()*400;

newCircle.scaleX=newCircle.scaleY=randomValue;
newCircle.alpha=1-randomValue;

this.addChild(newCircle);
}
}
Main();
```

このコードでは前のコードにいくつかの修正を加えることで前のコードを拡張しています。大きな変化はステージに 200 個の BlueCircle を作成するループを加えたことです。たくさんの BlueCircle オブジェクトのプロパティをランダム化しているので、円はさまざまな位置と大きさ、透明度で表示されます。



まとめ

ライブラリからコンテンツを取り出しステージに動的に配置することは非常に重要で有用な機能です。このチュートリアルではコンテンツの表示とその修正に関する基本的な方法を学びましたが、実際にはここで取り上げていない多くの事柄があります。

たとえば舞台裏では、BlueCircle()クラスが使用され自動的に提供されています。またみなさん自身のBlueCircleクラスファイル(BlueCircle.as)を作成し、このチュートリアルでは試さなかった多くのカスタム機能を付加することもできます。attachMovie には、新しく作成されるムービークリップに任意の数の引数を渡すことができるというすぐれた機能があるのですが、AS 3 ではこの機能を複製することも可能です。今後のチュートリアルではこの機能やそのほかのトピックにも触れていきたいと思っています。