

Flash Quick Starts: Programming with ActionScript 3.0

Embedding fonts の日本語訳

本ドキュメントは、Adobe 社のサイトにある“Embedding fonts“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

(2009/1/7)

本ドキュメントの原文は

http://www.adobe.com/devnet/flash/quickstart/embedding_fonts/

です。

フォントの埋め込み

アプリケーションの TextField にフォントを指定すると、Flash Player は同じ名前のついたデバイスフォント(ユーザーのコンピュータ上にあるフォント)を探します。そのフォントがユーザーのシステム上で見つからないかまたは、ユーザーがその名前のついた少し異なるバージョンのフォントを持っている場合は、そのテキスト表示は期待したものとはかなり異なる可能性があります。

正しいフォントで確実にユーザーに表示するには、アプリケーションの SWF ファイルにそのフォントを埋め込みます。フォントの埋め込みにはいくつかのメリットがあります。

- 埋め込んだフォントの文字はアンチエイリアスがかかり、特に大きなテキストでそのエッジを滑らかに表示することができる
- 埋め込んだフォントを使用したテキストを回転させることができる
- 埋め込んだフォントのテキストは透明化や半透明化することができる
- 埋め込んだフォントではカーニングした CSS スタイルが使用できる
- TrueType フォントや Type 1 Postscript フォントなど、システムにインストールしているフォントならほとんどのフォントを埋め込むことができる

埋め込みフォントの使用に関する大きな制限は、埋め込んだフォントはアプリケーションのファイルサイズやダウンロードサイズを増大させることです。

Flash アプリケーションにフォントを埋め込む方法は、次に挙げる方法などたくさんあります。

- ステージ上に置いたダイナミックテキストかテキスト入力のフォントとスタイルのプロパティを設定して、[埋め込み]ボタンをクリックする
- フォントシンボルを作成し参照する、本記事で取り上げる方法
- 埋め込みフォントシンボルを含む共有ライブラリを作成し、実行時に使用する、本記事では取り上げない高度な方法

次のセクションではこの機能を使用した、最もよく知られ、役立つ方法について述べます。

新しいフォントシンボルの作成

新しいフォントシンボルを作成するには、ライブラリのポップアップメニュー(ライブラリパネルの右上隅)から[新

しいフォント]を選択します。フォントシンボルの指定が行える[フォントシンボルプロパティ]ダイアログボックスが開きます。

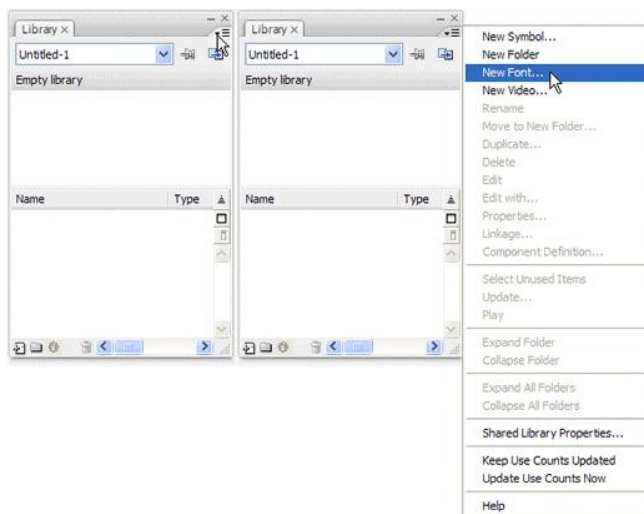


図1 : Flash ドキュメントのライブラリで新しいフォントシンボルを作成する

フォントシンボルのプロパティの設定

[フォントシンボルプロパティ]ダイアログボックスでは、フォントシンボルの名前を定義し、書体を選び、さまざまなスタイルの設定が行えます。このフォント名はライブラリパネルでシンボルが持つ名前になります。ActionScript を使ってダイナミックにこのフォントにアクセスするには、フォントのリンケージプロパティを設定する必要があります(“新しいフォントシンボルのリンケージプロパティの設定”を参照)。
[フォント]ポップアップメニューでは、現在使用しているコンピュータにインストールされているフォントのリストから希望する書体を選ぶことができます。

フォントのライブラリアイテムの作成

1. フォントシンボルを追加したいライブラリを開く
2. ライブラリパネルのメニューから[新しいフォント]を選択する
3. [名前]フィールドにフォントの名前を入力する
4. [フォント]メニューからフォントを選択するか、[フォント]フィールドにフォントの名前を入力する
5. (オプション)[ボールド]や[イタリック]を選択する
6. (オプション)フォント情報を、ベクターのアウトラインデータとしてでなく、ビットマップデータとして埋め込むには、[ビットマップテキスト]オプションを選択し、[サイズ]テキストフィールドにフォントサイズを入力します。(ビットマップフォントではアンチエイリアスを使用できません。[ビットマップテキスト]を選択する場合は、このフォントを使用するテキストに関して、プロパティインスペクタのアンチエイリアスオプションで[ピッ

トマップテキスト]を選択する必要があります。)

Note: [サイズ]設定は、[ビットマップテキスト]オプションを使用するときのみ適用します。

テキストのサイズは通常、TextFormat オブジェクト内で設定します。これについては後で見えていきます。



図2: フォントシンボルのプロパティの設定

フォントシンボルのリンケージプロパティの設定

新しい Font インスタンスをダイナミックにインスタンス化するにはその前に、リンケージクラス名を定義する必要があります。リンケージクラスを設定するには:

1. ライブラリでフォントシンボルを右クリックし、[リンケージ]を選択する
2. [ActionScript に書き出し]をクリックしクラス名を入力する(図3参照)
3. [OK]をクリックする

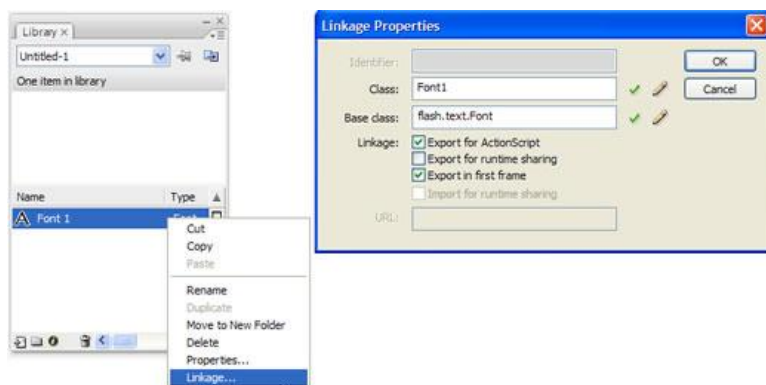


図3: [リンケージプロパティ]ダイアログボックスでリンケージクラスを設定

Note: [リンケージプロパティ]ダイアログボックスを閉じるとき、指定されたクラスの定義がクラスパス内に見つからなかったため、定義は自動的に作成されるという ActionScript クラスに関する警告が出る場合がありますが、そのときは[OK]をクリックします。

ActionScript を使った新しいフォントシンボルインスタンスの作成

ライブラリでフォントシンボルを作成し、リンケージクラスを定義すると、new 演算子を使ってそのフォントシンボルの新しいインスタンスを作成し、前に定義したリンケージクラス名を指定することができます。埋め込んだフォントをダイナミックに作成した TextField に適用するには、次のようにします。

- TextFormat オブジェクトを作成して、その fontFamily プロパティを埋め込んだフォントの名前に指定し、TextFormat オブジェクトを TextField に適用する。埋め込みフォントを指定するときには、fontFamily プロパティに単一の名前のみ含めるようにする。コンマで区切った複数のフォント名のリストは使用できない。
- CSS スタイルを使って TextField のフォントを設定する場合には、fontFamily CSS プロパティを埋め込みフォント名に設定する。埋め込んだフォントを指定する場合は、fontFamily プロパティは単一の名前のみを含み、名前リストであってはならない。

次のサンプルでは、リンケージクラス名を Font1 にしたライブラリのフォントシンボルにもとづき、新しいフォントシンボルを作成します。次に TextFormat オブジェクトを作成し、フォント名を myFont インスタンスで指定したフォントに設定します。

サンプル

次のサンプルでは、新しい Font、TextFormat、TextField インスタンスを作成し、embedFonts プロパティを true に設定します。

Note: このサンプルでは、ライブラリ内に Font1 というリンケージクラス名を持ったフォントシンボルが存在する必要があります。

```
// ドキュメントのライブラリから Font1 シンボルの新しいシンボルを作成
var myFont:Font = new Font1();

/* 新しい TextFormat オブジェクトを作成し、font プロパティを myFont オブジェクトの
   fontFamily プロパティに設定 */
var myFormat:TextFormat = new TextFormat();
myFormat.font = myFont.fontName;
myFormat.size = 24;

/* 新しい TextField オブジェクトを作成し、defaultTextFormat プロパティを使って、
   テキストフォーマットを代入。embedFonts プロパティを true に設定 */
```

```
var myTextField:TextField = new TextField();
myTextField.autoSize = TextFieldAutoSize.LEFT;
myTextField.defaultTextFormat = myFormat;
myTextField.embedFonts = true;
myTextField.text = "The quick brown fox jumped over the lazy dog.";
addChild(myTextField);
```

結果

The quick brown fox jumped over the lazy dog.

このサンプルのソースファイルのダウンロード

[embeddingfonts_section4_example1.zip](#) (ZIP, 8K)

高度なアンチエイリアスの設定

埋め込みフォントを扱うときには、テキストフィールドの `antiAliasType` プロパティを `AntiAliasType` クラス (`flash.text.AntiAliasType`) の値に設定することで、アンチエイリアスタイプを通常 (NORMAL) か高度 (ADVANCED) に設定できます。

サンプル

次のサンプルでは、新しい `Font` シンボルのインスタンスを作成し、テキストフィールドでフォントの埋め込みを可能にして、アンチエイリアスタイプを高度に設定しています。

Note: このサンプルでは、ライブラリ内に `Font1` というリンケージクラス名を持ったフォントシンボルが存在する必要があります。

```
// ドキュメントのライブラリから Font1 シンボルの新しいインスタンスを作成
var myFont:Font = new Font1();

/* 新しい TextFormat オブジェクトを作成し、font プロパティを myFont オブジェクトの
   fontName プロパティに設定 */
var myFormat:TextFormat = new TextFormat();
myFormat.font = myFont.fontName;
myFormat.size = 24;
```

```
/* 新しい TextField オブジェクトを作成し、defaultTextFormat プロパティを使って、  
テキストフォーマットを代入。embedFonts プロパティを true に設定し、  
antiAliasType プロパティを“advanced”に設定*/  
var myTextField:TextField = new TextField();  
myTextField.autoSize = TextFieldAutoSize.LEFT;  
myTextField.defaultTextFormat = myFormat;  
myTextField.embedFonts = true;  
myTextField.antiAliasType = AntiAliasType.ADVANCED;  
myTextField.text = “The quick brown fox jumped over the lazy dog.”;  
addChild(myTextField);
```

結果

The quick brown fox jumped over the lazy dog.

このサンプルのソースファイルのダウンロード

[embeddingfonts_section5_example1.zip](#) (ZIP, 5K)

サンプル

次のサンプルでは、新しいテキストフィールドを作成し、ユーザーがフォントを埋め込むかどうか、どちらのアンチエイリアスのタイプ(通常か高度)を使用するか、テキストフィールド内のテキストサイズを選ぶことができます。

Note: このサンプルでは、ライブラリ内に Font1 というリンケージクラス名を持ったフォントシンボルが存在する必要があります。また CheckBox コンポーネント、ComboBox コンポーネント、Slider コンポーネントもライブラリに存在する必要があります。ライブラリにコンポーネントを追加する簡単な方法は、一度それらをステージにドラッグしてからそれらをステージから消去する方法です。

```
// 必要なコンポーネントのクラスのインポート  
import fl.controls.CheckBox;  
import fl.controls.ComboBox;  
import fl.controls.Slider;  
import fl.events.SliderEvent;  
  
// ドキュメントのライブラリから Font1 シンボルの新しいインスタンスを作成  
var myFont:Font = new Font1();
```

```
/* 新しい TextFormat オブジェクトを作成し、font プロパティを myFont オブジェクトの
   fontName プロパティに設定 */
var myFormat:TextFormat = new TextFormat();
myFormat.font = myFont.fontName;
myFormat.size = 24;

/* 新しい TextField オブジェクトを作成し、defaultTextFormat プロパティを使って、
   テキストフォーマットを代入。embedFonts プロパティを true に設定し、
   antiAliasType プロパティを"normal"に設定*/
var myTextField:TextField = new TextField();
myTextField.autoSize = TextFieldAutoSize.LEFT;
myTextField.defaultTextFormat = myFormat;
myTextField.embedFonts = true;
myTextField.antiAliasType = AntiAliasType.NORMAL;
myTextField.text = "The quick brown fox jumped over the lazy dog.";
addChild(myTextField);

/* embedFonts プロパティのトグルに使用する CheckBox コンポーネントインスタンス
   の作成 */
var embedFontsCheckBox:CheckBox = new CheckBox();
embedFontsCheckBox.label = "embedFonts";
embedFontsCheckBox.move(0, 50);
embedFontsCheckBox.selected = myTextField.embedFonts;
embedFontsCheckBox.addEventListener(Event.CHANGE, checkBoxChangeHandler);
addChild(embedFontsCheckBox);

/* 通常と高度なアンチエイリアスのトグルに使用する
   ComboBox コンポーネントインスタンスの作成 */
var antiAliasTypeComboBox:ComboBox = new ComboBox();
antiAliasTypeComboBox.addItem({data:AntiAliasType.NORMAL, label:"AntiAliasType.NORMAL"});
antiAliasTypeComboBox.addItem({data:AntiAliasType.ADVANCED,
label:"AntiAliasType.ADVANCED"});
antiAliasTypeComboBox.enabled = myTextField.embedFonts;
antiAliasTypeComboBox.width = 200;
antiAliasTypeComboBox.move(150, 50);
antiAliasTypeComboBox.addEventListener(Event.CHANGE, comboBoxChangeHandler);
```

```
addChild(antiAliasTypeComboBox);

// フォントサイズの設定に使用する Slider コンポーネントインスタンスの作成
var fontSizeSlider:Slider = new Slider();
fontSizeSlider.minimum = 8;
fontSizeSlider.maximum = 24;
fontSizeSlider.value = Number(myFormat.size);
fontSizeSlider.tickInterval = 2;
fontSizeSlider.liveDragging = true;
fontSizeSlider.move(400, 56);
fontSizeSlider.addEventListener(SliderEvent.CHANGE, sliderChangeHandler);
addChild(fontSizeSlider);

/* CheckBox コンポーネントインスタンスのハンドラ関数。この関数は CheckBox の現在の
   値にもとづいて、ComboBox インスタンスの enabled プロパティをトグルさせる */
function checkBoxChangeHandler(event:Event):void {
    myTextField.embedFonts = CheckBox(event.currentTarget).selected;
    antiAliasTypeComboBox.enabled = CheckBox(event.currentTarget).selected;
}

/* ComboBox コンポーネントインスタンスのハンドラ関数。この関数は、ComboBox で
   現在選択されているアイテムの値にもとづいて、テキストフィールドの antiAliasType
   プロパティを設定する */
function comboBoxChangeHandler(event:Event):void {
    myTextField.antiAliasType = ComboBox(event.currentTarget).selectedItem.data;
}

/* Slider コンポーネントインスタンスのハンドラ関数。この関数は、スライダーの現在の値に
   もとづいて、テキストフォーマットのフォントサイズを設定し、setTextFormat( )メソッドを
   使ってテキストフィールドのテキストフォーマットを更新する */
function sliderChangeHandler(event:SliderEvent):void {
    myFormat.size = event.value;
    myTextField.setTextFormat(myFormat);
}
```

The quick brown fox jumped over the lazy dog.

embedFonts

AntiAliasType.NORMAL



このサンプルのソースファイルのダウンロード

[embeddingfonts_section5_example2.zip](#) (ZIP, 5K)

Tip: 高度なアンチエイリアス処理のメリットは小さいフォントの方が効果ははっきりします。前のサンプルでフォントの埋め込みを有効にし、高度なアンチエイリアス処理を行って、フォントサイズのスライダーを最小の値(8ポイント)に設定してみてください。

埋め込みフォントと Tween クラスを使ったテキストのフェード

フォントシンボルがライブラリにあると、テキストのフェードや回転が行え、ユーザーのコンピュータにはないフォントが使用できます。

次のサンプルでは、ActionScript 3.0 のさまざまなトウweenクラスを使って、フェードインとフェードアウトする回転したテキストを作成する方法を示しています。

サンプル

次のサンプルでは、Tween クラスを使ってテキストフィールドの alpha プロパティを制御します。

Note: このサンプルでは、ライブラリ内に Font1 というリンケージクラス名を持ったフォントシンボルが存在する必要があります。

```
// 必要なトランジションクラスのインポート
import fl.transitions.Tween;
import fl.transitions.TweenEvent;
import fl.transitions.easing.*;

// ドキュメントのライブラリから Font1 シンボルの新しいインスタンスを作成
var myFont:Font = new Font1();

/* 新しい TextFormat オブジェクトを作成し、font プロパティを myFont オブジェクトの
   fontName プロパティに設定 */
```

```

var myFormat:TextFormat = new TextFormat();
myFormat.font = myFont.fontName;
myFormat.size = 24;

/* 新しい TextField オブジェクトを作成し、defaultTextFormat プロパティを使って、
   テキストフォーマットを代入。embedFonts プロパティを true に設定し、
   antiAliasType プロパティを"advanced"に設定しテキストフィールドを回転*/
var myTextField:TextField = new TextField();
myTextField.autoSize = TextFieldAutoSize.LEFT;
myTextField.embedFonts = true;
myTextField.antiAliasType = AntiAliasType.ADVANCED;
myTextField.defaultTextFormat = myFormat;
myTextField.text = "The quick brown fox jumped over the lazy dog.";
myTextField.border = true;
myTextField.x = 15;
myTextField.y = 5;
myTextField.rotation = 15;
addChild(myTextField);

// テキストフィールドの alpha プロパティをフェードさせる新しい Tween オブジェクトの作成
var fadeTween:Tween = new Tween(myTextField, "alpha", Strong.easeIn, 1, 0, 2, true);
fadeTween.addEventListener(TweenEvent.MOTION_FINISH, motionFinishHandler);

/* フェードトゥイーンの手ドラ。トゥイーンが MOTION_FINISH イベントを送出すると、
   この関数が呼び出され、トゥイーンの方角を逆にする */
function motionFinishHandler(event:TweenEvent):void {
    Tween(event.currentTarget).yoyo();
}

```

結果

The quick brown fox jumped over the lazy dog.

このサンプルのソースファイルのダウンロード

[embeddingfonts_section6_example2.zip](#) (ZIP, 5K)