読書メモ＋tips＋日記

・ 読書メモ： 基本的に読み終わったもののみ載せる（読みかけとか途中で飽きたものを含めるとキリがないから）
・ Tips　： 作ったスクリプトとかTips（WSH, JScript, Perl, Ruby, Rails, UNIXコマンド他）※その他のtipsはこっち → tips
・ 日記　： 自分が思ったことの覚え書き的に

2010年02月01日

[Flash] ActionScript 3.0 基礎文法最速マスター

巷で「○○基礎文法最速マスター」というのが流行っているので真似てみた。
（というノリで書くのをいちどやってみたかったんだよねー）
というわけで。

・Perl基礎文法最速マスター - Perl入門～サンプルコードによるPerl入門～
・Ruby基礎文法最速マスター - Route 477
・PHP基礎文法最速マスター - Shin x blog
・Python基礎文法最速マスター - D++のはまり日誌
・Brainf*ck基礎文法最速マスター - 医者を志す妻を応援する夫の日記
・Haskell基礎文法最速マスター - think and error
・JavaScript基礎文法最速マスター - なんとなく日記

これらを真似て、ActionScript 3.0 版を作ってみました。

間違いや、書きたした方が良さそうな点あったらコメントかメール頂けると嬉しいです。

ActionScript 3.0 基礎文法最速マスター

ActionScript 3.0 の文法一覧です。他の言語をある程度知っている人はこれを読めば ActionScript 3.0 の基礎をマスターして ActionScript 3.0 を書くことができるようになっています。簡易リファレンスとしても利用できます。

1. 基礎

開発環境

ActionScript 3.0 は、コンパイルしてFlash（SWFファイル）に組み込まれます。
開発環境には、Flash、Flex Builder、Flex SDKがあります。

(1) Flash

有償のオーサリングツールです。
タイムラインアニメーションを作成するデザイナが主に使います。
スクリプトを挿し込みたいフレーム（アニメーションの1コマ）を右クリックして
「アクション」を選択したときに出てくる編集ウィンドウに ActionScript コードを書きます。

(2) Flex Builder

記事検索

検索

<< 2010年2月

日	月	火	水	木	金	土
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28						

カテゴリ

読書メモ (146)
 読書メモ (一般) (106)
 読書メモ (IT) (32)
 読書メモ (リファレンス類) (7)
tips (102)
 tips (UNIX関連) (9)
 tips (EmEditorマクロ) (11)
 tips (JScript) (31)
 tips (ActionScript/Flash) (17)
 tips (Ruby) (27)
 tips (その他) (10)
日記 (123)
 日記2010 (2)
 日記2009 (9)
 日記2008 (7)
 日記2007 (19)
 日記2006 (13)
 日記2005 (37)
 日記2004 (10)
 日記 (IT) (26)
文房具オタク (3)

Archives

有償の統合開発環境(IDE)です。
Eclipseベースの環境なので、Java開発経験者は馴染みやすいです。
Visual Studioを使っている方はVisual Studioのようなものだと思ってください。
メニューから「ActionScriptプロジェクト」、「ActionScriptクラス」を作成後、
「ビルド」、「実行」のボタン操作をGUIから行ないます。(Eclipseベースの環境です)
※「Flexプロジェクト」を使うとGUIビルダを使った開発が出来ますが、この文章では触れません。

(3) Flex SDK

コマンドラインでのコンパイラです。
Flex SDKでのコンパイルは、コマンドラインで次のようにします。

```
mxmlc Sample.as
```

Hello, World

Flex Builder、Flex SDK では次のコードで、画面に Hello, World! と出せます。

```
package {
    import flash.display.Sprite;
    import flash.text.TextField;

    public class Sample extends Sprite {
        public function Sample()
        {
            var textField:TextField = new TextField();
            textField.text = "Hello, World!";
            addChild(textField);
        }
    }
}
```

trace 関数(print)

文字を出力するにはtrace関数を使います。
出力結果は Flash、Flex Builderでは、traceの結果は「出力」パネルに出力されます。
Flex SDKでのデバッグには デバッグ版FlashPlayer のインストールが必要で、
traceの結果はログファイルに出力されます

```
trace("Hello, world!");
```

※ ログファイルのデフォルト格納場所
Windows 2000/XP の場合:
C:\Documents and Settings\username\Application Data\Macromedia\Flash Player\Logs
Windows Visa の場合:
C:\Users\username\AppData\Roaming\Macromedia\Flash Player\Logs
Linux の場合
標準出力に出ます

コメント

最新記事

[Flash] ActionScript 3.0 基礎文法最速マスター
[book(IT)] 「OpenCL 入門」
[日記] トンボ
[books(IT)] 「Rubyスクリプティング・テクニック」(Brian Marick)
[日記] らくがき帖
[Ruby] 単体テストツールと、コマンド実行テスト ～ その3: Test::Unitのテストコード
[Ruby] 単体テストツールと、コマンド実行テスト ～ その2: Rubyからコマンド呼び出し
[Ruby] 単体テストツールと、コマンド実行テスト ～ その1: 検討
[book] 「ラクガキ・マスター」(寄藤文平)
[Ruby] PowerPoint に格子線を引く君

Profile

Contact me!

ホームページへ
メールフォームへ

コメントです。

```
// コメント
/* コメント */
```

変数の宣言

変数の宣言は「 var 変数名:型名; 」です。

```
// 数値
var num:int; // 整数
var num:Number; // 小数

// 配列
var arr:Array;

// ハッシュ（Objectクラスで代用）
var hash:Object;
```

スクリプトの実行

ブラウザまたは スタンドアロン版のFlashPlayer で、作成されたSWFファイルを起動します。（通常はダブルクリックでOK）

デバッガの起動

Flash、Flex BuilderにはGUIのデバッグモードがあるので、「デバッグ」ボタンでデバッガが起動します。
Flex SDK でデバッガを起動するにはコマンドラインで次のようにします。

```
mxmlc -debug=true Sample.as
fdb Sample.swf
```

2. 数値

数値の表現

整数には int クラス、小数には Number クラスを使います。

```
var num:int = 1;
var num:Number = 1.234;
```

四則演算

四則演算です。

```
num = 1 + 1;
num = 1 - 1;
```

[Ruby] Google Calendar API を使ってコマンドラインから予定を登録 (腰)
TODO係 (takaaki)
TODO係 (Dumpo)
[日記] あしたのてんきのきのう (takaaki_bb)

Recent TrackBacks

チルドレン 伊坂幸太郎 (粋な提案)
[book]「チルドレン」(伊坂幸太郎)
Astro de 3D(12) PerspectiveProjection は放棄<3> (閃光的網站・弛緩複合体 -Review Division-)
[ActionScript] Flashで3D ～ Papervision3dを使ってみる

```
num = 1 * 2;
num = 1 / 2;
```

余りと商の求め方です。商を求めるには普通の除算を行った後にint関数で整数部を取り出します。

```
// 商
div = int(7/2); // => 3

// 余り
mod = 7 % 2;  // => 1
```

インクリメントとデクリメント

インクリメントとデクリメントです。

```
// インクリメント
i++

// デクリメント
i--
```

3. 文字列

文字列の表現

文字列はシングルクォートかダブルクォートで囲みます。シングルクォート、ダブルクォートのどちらでも、\t(タブ)や\n(改行)などの特殊文字を利用することができます。またダブルクォートで囲まれた文字列の中での変数展開は出来ません。

```
var str1:String = 'abc';
var str2:String = "def";
var str3:String = "a¥tbc¥n";
```

文字列操作

各種文字列操作です。

```
// 結合
var join:String = 'aaa' + 'bbb';

// 分割
var str:String = 'aaa,bbb,ccc';
var record:Array = str.split(/,/);  // => ['aaa', 'bbb', 'ccc']

// 長さ
var str:String = 'abcdef';
var length:int = str.length;  // => 6

// 切り出し
```

```
var str:String = 'abcd';
var substr:String = str.substr(0, 2); // => ab

// 検索
str = 'abcd';
var result:int = str.indexOf('cd'); // => 2
  //見つかった場合はその位置、見つからなった場合は-1が返る）
```

※ 文字列操作についての他の言語との比較表は、下記ＵＲＬを参照してください。

文字列操作の比較表
http://blog.livedoor.jp/takaaki_bb/archives/51242781.html

4. 配列

配列変数の宣言と代入

配列です。

```
// 配列の宣言
var array:Array;

// 配列への代入
array = [1, 2, 3];
array = new Array(1, 2, 3);
```

配列の要素の参照と代入

配列の要素を参照と代入です。

```
// 要素の参照
array[0];
array[1];

// 要素の代入
array[0] = 1;
array[1] = 2;
```

配列の個数

配列の個数を取得するには length プロパティを使います。

```
var array_num:int = array.length;
```

配列の操作

配列を操作する関数です。

```
var array:Array = [1, 2, 3];

// shift(先頭を取り出す)
var first:int = array.shift; // => first は 1、array は [2, 3]

// unshift(先頭に追加)
array.unshift(5); // => array は [5, 2, 3]

// pop(末尾を取り出す)
var last:int = array.pop(); // => arrayは [5, 2]

// push(末尾に追加)
array.push(9); // => arrayは [5, 2, 9]
```

※ 配列操作について、他の言語との比較表は下記ＵＲＬを参照してください。

配列操作の比較表
http://blog.livedoor.jp/takaaki_bb/archives/51242767.html

5. ハッシュ（Objectクラスで代用）

ハッシュ宣言

```
var hash:Object = new Object();
```

ハッシュの要素の参照と代入

ハッシュの要素の参照と代入です。

```
// 要素の参照
hash['a'];
hash['b'];

// 要素の代入
hash['a'] = 5;
hash['b'] = 7;
```

ハッシュに関する関数

ハッシュに関する関数です。

```
// キーの取得（PerlとかRubyみたいに hash.keys というふうには取れない）
var keys:Array = [];
for (var k:* in hash) {
    keys.push(k);
}
trace(keys);  // ['b', 'a'] ※ 順番は保証されない

// 値の取得（PerlとかRubyみたいに hash.values というふうには取れない）
var values:Array = [];
for (var k:* in hash) {
```

```
    values.push(k);
}
trace(values);  // ['7', '5'] ※ 順番は保証されない

// キーの存在確認
hash['a'] != undefined

// ハッシュのキーの削除
delete hash['a'];
```

6. 制御文

if文

if文です。

```
if (条件) {

}
```

if ～ else文

if ～ else文です。

```
if (条件) {

} else {

}
```

if ～ else if 文

if ～ else if文です。

```
if (条件) {

} else if (条件) {

}
```

while文

while文です。

```
var i:int = 0;
while (i < 5) {

    // 処理
```

```
    i++;
}
```

for文

for文です。

```
for (var i:int = 0; i < 5; i++) {
    // 処理
}
```

※ ちなみに、こう書いても i はブロックスコープにはならなくて、関数スコープになります。
なので、こう続けて書くと「変数定義が重複しています」という警告が出ます。

```
for (var i:int = 0; i < 5; i++) {
        // 処理
}
for (var i:int = 0; i < 10; i++) { // NG
        // 処理
}
```

for .. in 文

```
for (var obj:String in obj) {
    xxx.text += p + ":" + obj[p] + "¥n";
}
```

※ foreach という名前の構文はないけど、for each (.. in ..) や Array クラスの
forEach() メソッド、というのはある

比較演算子

比較演算子の一覧です。Perlでは数値比較と文字列比較は厳密に区別されます。

数値比較演算子の一覧です。

```
num1 == num2  // num1 は num2 と等しい
num1 != num2  // num1 は num2 と等しくない
num1 <  num2  // num1 は num2 より小さい
num1 >  num2  // num1 は num2 より大きい
num1 <= num2  // num1 は num2 以下
num1 >= num2  // num1 は num2 以上
```

文字列比較演算子の一覧です。

```
str1 == str2  // str1 は str2 と等しい
```

```
str1 != str2  // str1 は str2 は等しくない
str1 <  str2  // str1 は str2 より小さい
str1 >  str2  // str1 は str2 より大きい
str1 <= str2  // str1 は str2 以下
str1 >= str2  // str1 は str2 以上
```

7. クラス、メソッド

ActionScript 3.0 でのクラスの定義は次のようなものです。メソッドの戻り値の型は、メソッド定義で指定します。

```
package 所属するパッケージ名 { // パッケージ名省略可
    import xxx;
    import xxx;

    public class クラス名 extends 親クラス名
    {
        // 変数宣言
        アクセス修飾子 var プロパティ名:プロパティの型;

        // コンストラクタ
        アクセス修飾子 function コンストラクタ名()
        {
            this.aaa = dddd;
        }

        // メソッド1
        アクセス修飾子 function メソッド名1(引数1:型 = デフォルト値):戻り値の型
        {
            return this.mmmm;
        }
    }
}
```

Flex Builder、Flex SDKを使っている場合で、画面に描画を行なうためのクラスは、次のようにSprite クラスを継承したコードの、コンストラクタに実行したいコードを書きます。(ここがエントリポイントになります)

```
package {
    import flash.display.Sprite;

    public class Sample extends Sprite
    {
        public function Sample()
        {
            //TODO: ここに実行したいコードを書く
        }
    }
}
```

8. 表示リスト（ディスプレイリスト）

ActionScript で表示するクラスは、次のような親子階層を定義しておくと、親から順に画面表示が行なわれます。

```
ステージ                                    Stage クラス
    +- SWFファイルのメインクラスのインスタンス  Sprite クラス
        +- 表示オブジェクトコンテナ            DisplayObjectContainer クラス
        |   +- 表示オブジェクト                DisplayObject クラス
        |   +- 表示オブジェクト                  （のそれぞれをを継承したクラス）
        +- 表示オブジェクト
```

ここでの「親子階層」は、クラスの継承関係のことではなく、DisplayObjectContainer クラスの addChild() を使って構築された親子階層を表します。
プログラマが編集できるのは「SWFファイルのメインクラス」以下の階層で、「ステージ」はFlashPlayerが勝手に用意します。

具体的に、次の親子階層を構築するコードを見ていきましょう。

```
ステージ                                    Stage クラス
    +- SWFファイルのメインクラスのインスタンス  Sample クラス
        +- 表示オブジェクトコンテナ            sprite01 : Sprite
        |   +- 表示オブジェクト                text01 : TextField
        |   +- 表示オブジェクト                text02 : TextField
        +- 表示オブジェクト                    text03 : TextField
```

上のような親子階層を構築するには、次のように addChild() を呼びます。(this は省略可能です)

```
package {
    import flash.display.Sprite;
    import flash.text.TextField;

    public class Sample extends Sprite {
        public function Sample()
        {
            var sprite01:Sprite = new Sprite();
            var text01:TextField = new TextField();
            var text02:TextField = new TextField();
            var text03:TextField = new TextField();

            text01.text = "1";
            text02.text = "2";
            text03.text = "3";

            text01.x = 0;
            text02.x = 50;
            text03.x = 100;

            // ディスプレイリスト（親子階層）の構築
            this.addChild(sprite01);
```

```
            sprite01.addChild(text01);
            sprite01.addChild(text02);
            this.addChild(text03);
        }
    }
}
```

※ Flex Builder、Flex SDK ではなくFlash で作成したSWFは、「メインクラス」はMovieClip クラスです。(ちなみにMovieClipクラスはSpriteを継承しています)

表示リストについて詳しい話は公式のドキュメントを参照してください。
http://livedocs.adobe.com/flex/2_jp/docs/wwhelp/wwhimpl/js/html/wwhelp.htm?href=00001853.html

9. ファイル入出力

Flash はブラウザで動作することを前提として作られているので、ファイル入出力の機能はありません。(AIRにはあります)

10. 知っておいたほうがよい文法

ActionScript 3.0 でよく出てくる知っておいたほうがよい文法の一覧です。

ActionScript 3.0 の真偽値

ActionScript 3.0 で偽と判断される値は、「false」「0」「0.0」「NaN」「''(空文字)」、「null」、「undefined」です。これ以外は真になります。

あと Boolean クラスがあります。

```
var b:Boolean = false;
```

※ 真偽値の扱いの、他の言語との比較表は下記URLを参照してください

真偽値の比較表
http://blog.livedoor.jp/takaaki_bb/archives/51242784.html

書式の指定

メインクラスの上に次のような指定をすると、サイズ、背景色、フレームレートが設定出来ます。

```
[SWF(width="320", height="320", backgroundColor="#ffffff", frameRate="60")]
public class Sample extends Sprite {
    // ...
}
```

以下の項目に該当する文法はありません。たぶん。

・unless文
・後置のif, 後置のunless
・リスト代入
・範囲演算子
・文字列のリストを簡単に書く構文があります。

例外処理

例外処理には try .. catch が使えます

```
try {
    // 例外が起きるかもしれない処理

} catch (e:Error) { // Exception ではない
    trace(e.getStackTrace());
    // 例外処理
}
```

3項演算子

３項演算子です。下のサンプルの場合は flg が真値の場合は 1 が、偽値の場合は 2 が num に代入されます。

```
var num:int = flg ? 1 : 2;
```

||=

左辺値が未定義の場合に右辺値を代入します。下のサンプルの場合は num が undefined であった場合に 2 が num に代入されます。(ActionScriptで出来るって知らなかった・・)

```
num ||= 2;
```

クラス定義の読み込み

クラス定義を読み込むには import を使います。

```
import パッケージ名.クラス名;

// 例
import flash.display.*; // これはちょっと行儀が悪い
import flash.text.TextField;
```

よく使用する(というか標準の)関数・ライブラリー覧

標準のクラスライブラリは必須です。これがないと絵が出ません。

Flex 用
http://livedocs.adobe.com/flex/3_jp/langref/index.html

Flash 用
http://livedocs.adobe.com/flash/9.0_jp/ActionScriptLangRefV3/

共通のパッケージは flash.* パッケージです。他に Flex 専用の mx.* パッケージ、Flash 専用の fl.* パッケージがあります。

ActionScript 3.0 コーディングガイドライン

Adobeが ActionScript 3.0 のコーディングガイドライン を出してます。
http://opensource.adobe.com/wiki/display/flexsdk/Coding+Conventions-ja

※ でも for 文の「 { 」の直前で改行するとか嫌だなあ・・

こまごましたtips

こまごました tips（「ActionScriptメモ」）は下記URLに載せているので、こちらも参考にどうぞ！
http://homepage2.nifty.com/takaaki024/tips/programs/flex/actionscript3.html

11. 書籍紹介

基本からしっかりわかるActionScript 3.0 (Web Designing BOOKS)
著者:森 巧尚
販売元:毎日コミュニケーションズ
発売日:2009-07-14
おすすめ度:
クチコミを見る

Amazonはこの本を薦めてきましたよ。

.fla 2 ―Idea of Flash Creation―
著者:新藤愛大
販売元:ワークスコーポレーション
発売日:2009-11-21
クチコミを見る

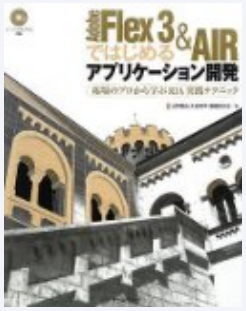
いまをときめく人たちが書いている本ってことで .fla 1 とともに気になる1冊。

Adobe Flash CS4 詳細!ActionScript3.0入門ノート[完全改訂版](CD-ROM付)
著者:大重 美幸
販売元:ソーテック社
発売日:2009-08-01
おすすめ度:
クチコミを見る

この人の本は分かりやすいです。あとこの人の作るFlashは面白いです。「キャスター先生」最高！

Flashデザインラボ -プロに学ぶ、一生枯れない永久不滅テクニック(Design Lab+ 1-2)
著者:デザインラボ編集部
販売元:ソフトバンククリエイティブ
発売日:2008-05-30
おすすめ度:
クチコミを見る

「ActionScriptプログラミングは分かるんだけど、タイムラインアニメーション(とくにデザイナさんがやってるような作業)がよく分からないのですよ」と思ってる人は必見。



Adobe Flex 3 & AIR ではじめるアプリケーション開発
著者:公門 和也
販売元:インプレスジャパン
発売日:2008-12-01
おすすめ度:
クチコミを見る

Flexでの業務アプリケーションについて書いてある本は少ないけど、この本はしっかり書いてあるから、良い感じですね。

takaaki_bb at 23:06|Comments(1)|TrackBack(0)|clip!| tips (ActionScript/Flash) | tips

このBlogのトップへ|前の記事

トラックバックURL

http://trackback.blogsys.jp/livedoor/takaaki_bb/5137 クイック

この記事へのコメント

1. Posted by fl 2010年02月02日 19:00

hashの値の取得はfor eachかhash[k]とする必要があるのでは。あとキーの存在確認は
in演算子のほうが良いと思います。値としてundefinedが入っている場合があるので。

この記事にコメントする

名前:

メール:

URL:

情報を記憶: 評価: -- 顔 星

投稿する