

# ActionScript-Facile UI Components Framework AS3

par Matthieu ([Accueil](#)) ([Blog](#))

Date de publication : 21 septembre 2010

Dernière mise à jour :

**Avec le Framework de Composants Graphiques AS3 "ActionScript Facile" ,  
développez facilement des RIA Flash intuitives et personnalisables.**

I - Présentation.....	3
II - Fonctionnalités.....	3
III - Exemple d'utilisation des composants.....	3
III-A - Création du skin des composants.....	3
III-B - Utilisation des Composants.....	6
III-C - Téléchargement du code source de l'exemple.....	10
IV - Tutoriels création des composants.....	11
V - Licence et Versions.....	11
VI - Pages du projet et téléchargement.....	11
VII - Support - Questions.....	11

## I - Présentation

Le « **Framework ActionScript-Facile** » est une « **bibliothèque de composants graphiques** » ou un « Visual Components Framework ».

Il vous permet d'ajouter à vos **applications flash** des **composants graphiques personnalisables**.

## II - Fonctionnalités

Ce Framework vous (Développeurs AS3) permet de vous concentrer sur le code AS3. Simultanément, les webdesigner peuvent concevoir le graphisme des composants. **Aucune connaissance en programmation** n'est nécessaire pour la **création des skins** (les graphismes) des composants du Framework ActionScript Facile.

De plus, toutes les **classes AS3** sont **commentées**, vous pouvez ainsi implémenter vos propres fonctionnalités et concevoir des nouveaux composants.

Le « *Framework ActionScript-Facile* » évolue selon vos besoins!

Un **espace de collaboration** est disponible pour partager vos évolutions, poser vos questions avec la **Communauté ActionScript-Facile**. Ensemble, vous **contribuez à l'évolution du Framework ActionScript-Facile**.

Le « Framework ActionScript-Facile » est développé en **pure AS3**. Vous pouvez donc utiliser vos logiciels habituels : à savoir FDT, FlashDevelop, le compilateur Flex, les compilateurs Flash CS3, CS4, CS5 ☐

## III - Exemple d'utilisation des composants

Ensemble, nous allons créer une interface utilisant une partie des fonctionnalités de chacun des composants.

### III-A - Création du skin des composants

Chaque composant est personnalisable graphiquement. En fonction du composant, vous devez créer un fichier fla avec plusieurs MovieClip.

Pour connaître le nom des classes / MovieClip à créer pour le composant sélectionné, il suffit de regarder les constantes de la classe.

Pour le composant Button, dans le code source de cette classe, il y a 3 constantes :

```
static public const BUTTON_UP_SKIN:String = "BUTTON_UP_SKIN";
static public const BUTTON_DOWN_SKIN:String = "BUTTON_DOWN_SKIN";
static public const BUTTON_OVER_SKIN:String = "BUTTON_OVER_SKIN";
```

Ensuite, il vous suffit de créer pour le composant Button, une classe de Skin. Si vous avez plusieurs Button, avec des graphismes différents, sur votre interface, il suffit de **créer une classe Skin par graphisme du Button**.

Ci-dessous, la classe qui gère le skin d'un des boutons de l'interface.

```
package com.as3facileexemple.skin.classic
{
    // Import des classes gérant la partie graphique du composant dans le fla (movieclip)
    // Proviens de ui.swc (créé avec la compilation de UI fla)
    import com.as3facile.skin.button.ButtonDownSkin;
    import com.as3facile.skin.button.ButtonOverSkin;
    import com.as3facile.skin.button.ButtonUpSkin;

    import com.actionscriptfacile.skin.Skin;
    import com.actionscriptfacile.ui.button.Button;

    /**
     * Définition du skin utilisé pour un composant Button
     *
     * @author Matthieu
     */
    public class DefaultButtonSkin extends Skin
```

```
{  
  
    public function DefaultButtonSkin()  
    {  
        // Affectation de chaque élément du bouton à un skin  
        setSkin( Button.BUTTON_DOWN_SKIN ,  
        ButtonDownSkin ); // élément du bouton, classe graphique (movieclip) associée  
        setSkin( Button.BUTTON_OVER_SKIN, ButtonOverSkin );  
        setSkin( Button.BUTTON_UP_SKIN , ButtonUpSkin );  
    }  
  
}  
  
}
```

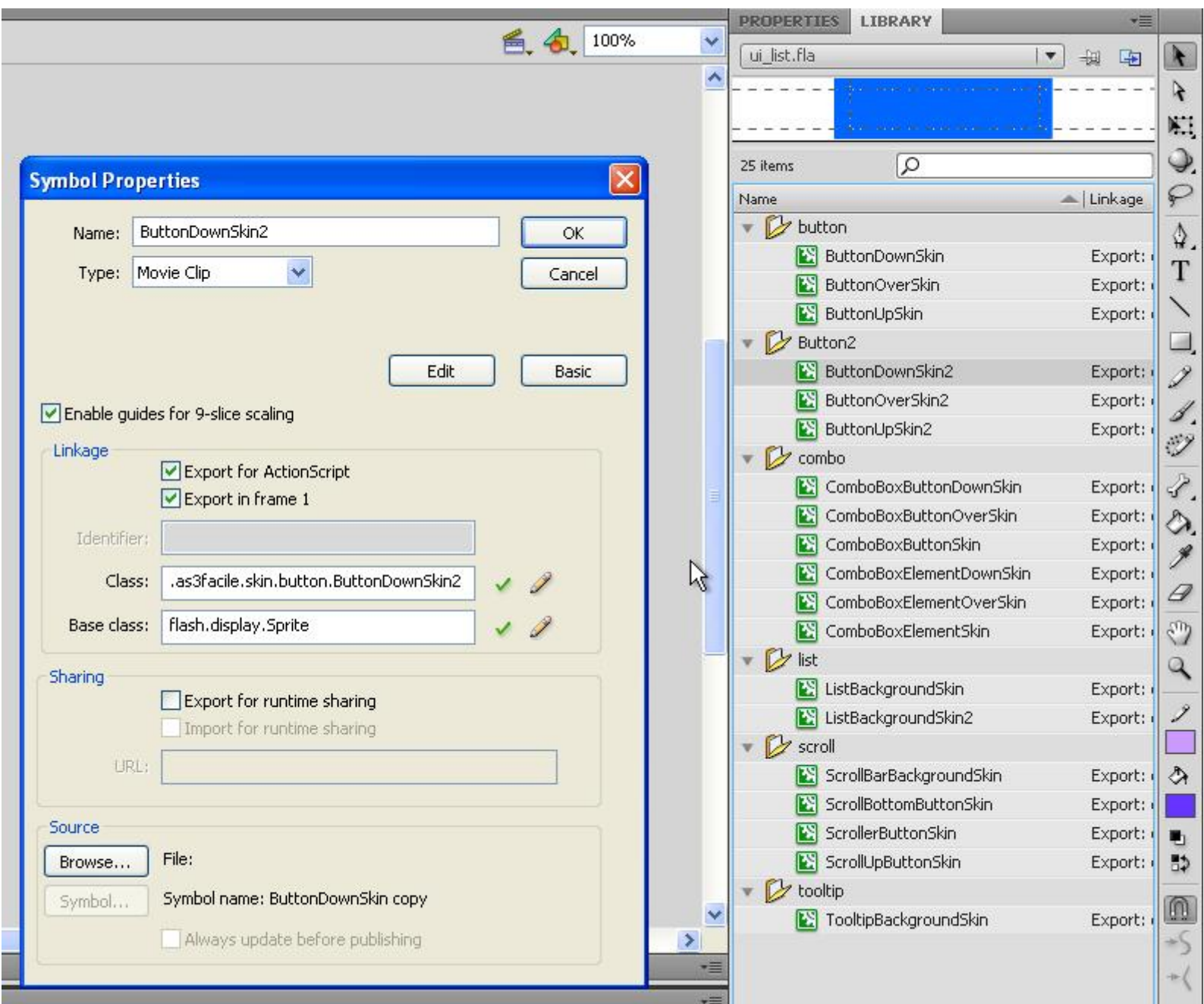
Puis, il vous reste à créer le fla avec les classes correspondantes.

Par exemple, pour créer le **Skin du composant Button**, il est nécessaire de créer 3 MovieClip différents. **1 MovieClip par état du bouton** (survolé, appuyé, normal).

### **Pensez-bien à paramétrer les propriétés de chacun des MovieClip :**

- Cochez la case Export For ActionScript
- Cochez la case Export in frame 1
- La Class : permet d'utiliser le graphisme du composant dans votre code AS3.
- La Base Class : permet de définir les propriétés de votre graphisme / composant disponible pour votre code AS3.

Une copie d'écran pour illustrer mes explications :




Les graphismes définis dans le fla sont très basiques. Libre à vous d'insérer des images, des animations, des dessins vectoriels plus créatifs et recherchés.

Il y a **énormément de possibilités pour personnaliser les composants du Framework Actionscript-Facile**.

Le principe des créations des skins est identique pour chacun des composants. Il y a plus ou moins de MovieClip à créer suivant les états graphiques possibles du composant.

Ensuite, une fois le fla créé, vous pouvez l'exporter soit au format swc ou au format swf. C'est en fonction de la manière dont vous souhaitez utiliser les MovieClip avec votre code AS3.

Pour cet exemple, nous allons **exporter le fla au format swc** pour intégrer directement les classes graphiques avec l'éditeur AS3 de notre choix.

Pour les Développeurs Actionscript professionnels, je vous conseille d'utiliser l'éditeur actionscript :  **FDT PowerFlasher**, l'essayer, c'est l'adopter.

Vous trouverez un **livre pdf gratuit** avec le **descriptif de ces fonctionnalités et un tutoriel d'utilisation** sur : **Livre Editeur FDT ActionScript-Facile**

## III-B - Utilisation des Composants

Tous les composants du Framework ActionScript-Facile s'utilisent de la même façon. Ils possèdent tous des fonctionnalités de base identiques : personnalisation graphiques, possibilité de l'afficher ou non dans la DisplayList, interaction possible avec l'utilisateur.

Pour des informations détaillées, vous pouvez lire l'article suivant : [Les fonctionnalités de base des composants Actionscript-Facile](#)

La 1ère étape consiste à **créer une nouvelle instance de notre composant**. Par exemple pour un Button, il suffit d'écrire le code suivant :

```
var bouton:Button = new Button();
```

2ème étape : en fonction du composant utilisé, vous **définissez ses paramètres** propres. Ci dessous, la création d'un composant Button et d'une liste.

```
// création un objet de type Button
var bouton:Button = new Button();

// définition du texte
bouton.label = 'Button Component';

// création d'une liste
var list>List = new List();

// définition de la taille de la List
list.resize( 230, 150 );

// ajout de plusieurs éléments dans la liste
for ( var i:int = 0; i < 35; i++ )
{
    list.addElement( getListElement(i+1) );
}
```

La 3ème étape consiste à **affecter à notre composant le skin** que nous lui avons conçu.

Un simple `button.applySkin( new DefaultButtonSkin() );` met à jour le graphisme de notre composant.

Ou `list.applySkin( new DefaultListSkin() );` pour le composant List.

En route pour la 4ème étape!

Il nous reste à **positionner notre composant sur la scène et à l'afficher**. Bien évidemment, il est possible de paramétrer les options de notre composant et de modifier sa taille.

```
// ajout à la displaylist
addChild( bouton );

// définition de sa taille en 150 * 30 pixels
bouton.resize( 150, 30 );

// définition de la position du bouton sur la scène
bouton.x = 60;
bouton.y = 60;
bouton.labelField.alignCenter(); // centre le texte
bouton.labelField.changeFormat("color", 0xffffffff); // changement de la couleur du texte
bouton.labelField.changeFormat("size", 14); // changement de la taille de la police du texte

// détermination de la position de la List
list.x = 30;
list.y = 30;

// affichage - ajout à la displaylist
addChild( list );

// ajout des marges au contenu de la liste
```

```
list.margins = new UIMargins( 5, 5, 5, 5 )
```

La dernière étape est plus ou moins compliquée en fonction de l'application flash que vous créez. Vous avez la possibilité d'**écouter les actions de l'utilisateur**, pour par exemple, afficher / masquer telle ou telle partie de l'interface, modifier le titre d'un élément...

Ci-dessous le code source complet d'utilisation des composants que nous avons créés.

```
package
{
    import flash.events.MouseEvent;
    import com.as3facileexemple.skin.classic.DefaultListSkin;
    import com.as3facileexemple.skin.classic.DefaultButtonSkin;
    import com.actionscriptfacile.ui.button.Button;
    import com.as3facileexemple.skin.classic.DefaultButtonSkin2;
    import com.as3facileexemple.skin.classic.DefaultListSkin2;
    import com.actionscriptfacile.ui.list.List;

    import flash.display.Shape;

    import com.actionscriptfacile.ui.utils.UIMargins;
    import com.actionscriptfacile.ui.combo.element.ComboBoxElement;
    import com.actionscriptfacile.ui.combo.ComboBox;
    import com.actionscriptfacile.ui.text.UITextField;

    import com.as3facileexemple.skin.classic.DefaultComboBoxSkin;

    import flash.display.Sprite;
    import flash.events.Event;
    import flash.text.Font;

    /**
     * Exemple d'utilisation du Framework de composants AS3 ActionScript-Facile
     *
     * @author Matthieu
     */
    public class DeveloppezExemple extends Sprite
    {

        protected var m_comboFonts:ComboBox;
        protected var m_list:List;
        protected var m_textField:UITextField;

        public function DeveloppezExemple()
        {
            /**
             * On construit les boutons
             */
            var buttonHide:Button = new Button();

            // définition du texte
            buttonHide.label = 'Arc en Ciel Liste';

            // Application de la skin par défaut

            // [ Attention ! Cette skin utilise le fichier ui.swc qui doit être ajouté à la liste des composants à passer a
            buttonHide.applySkin( new DefaultButtonSkin() );

            // on écoute les changements qui interviennent sur le bouton
            buttonHide.addEventListener(MouseEvent.CLICK, changeSkinList );

            // ajout à la displaylist
            addChild( buttonHide );

            // définition de sa taille en 150 * 30 pixels
            buttonHide.resize( 150, 30 );
        }
    }
}
```

```
// définition de la position du bouton sur la scène
buttonHide.x = 10;
buttonHide.y = 220;

// accès au composant de type UITextField (labelField)
buttonHide.labelField.alignCenter(); // centre le texte
buttonHide.labelField.changeFormat("color", 0xffffffff); // changement de la couleur du texte
buttonHide.labelField.changeFormat("size", 14); // changement de la taille de la police du texte
buttonHide.labelField.changeFormat("font", "Arial"); // changement de la police du texte

var buttonShow:Button = new Button();

// définition du texte
buttonShow.label = 'Classique Liste';

// Application de la skin par défaut

// [ Attention ! Cette skin utilise le fichier ui.swc qui doit être ajouté à la liste des composants à passer a
buttonShow.applySkin( new DefaultButtonSkin2() );

// on écoute les changements qui interviennent sur le bouton
buttonShow.addEventListener(MouseEvent.CLICK, changeSkinList2 );

// ajout à la displaylist
addChild( buttonShow );

// définition de sa taille en 150 * 30 pixels
buttonShow.resize( 150, 30 );

// définition de la position du bouton sur la scène
buttonShow.x = buttonHide.x;
buttonShow.y = buttonHide.y + buttonShow.height + 10;

// accès au composant de type UITextField (labelField)
buttonShow.labelField.alignCenter(); // centre le texte
buttonShow.labelField.changeFormat("color", 0xff33ff); // changement de la couleur du texte
buttonShow.labelField.changeFormat("size", 14); // changement de la taille de la police du texte
buttonShow.labelField.changeFormat("font", "Arial"); // changement de la police du texte

/**
 * On construit la liste
 */
m_list = new List();

// Application de la skin par défaut

// [ Attention ! Cette skin utilise le fichier ui.swc qui doit être ajouté à la liste des composants à passer a
m_list.applySkin( new DefaultListSkin2() );

// définition de la taille de la List
m_list.resize( 230, 150 );

// ajout de plusieurs éléments dans la liste
for ( var i:int = 0; i < 35; i++ )
{
    m_list.addElement( getListElement(i+1) );
}

// détermination de la position de la List
m_list.x = 200;
m_list.y = buttonHide.y;

// ajout des marges au contenu de la liste
m_list.margins = new UIMargins( 5, 5, 5, 5 );

// affichage - ajout à la displaylist
addChild( m_list );
```



```

/**
 * On construit la combobox
 */
m_comboFonts = new ComboBox();
m_comboFonts.applySkin( new DefaultComboBoxSkin() );

// on écoute les changements qui interviennent dans la combobox
m_comboFonts.addEventListener(Event.CHANGE, changeHandler );
m_comboFonts.resize( 300, 70 );
m_comboFonts.componentsHeight = 25;// hauteur des éléments de la ComboBox
m_comboFonts.margins = new UIMargins( 2, 2, 2, 2 );// ajout des marges au contenu de la liste

// on ajoute tous les noms de polices autorisées comme élément de la combobox
var fonts:Array = Font.enumerateFonts( true );

var boxElement:ComboBoxElement;

for each ( var font:Font in fonts )
{
    boxElement = m_comboFonts.addElement( font.fontName, font.fontName );
    boxElement.labelField.changeFormat("color", Math.random() *
0X00FFFFFF);// changement de la couleur du texte
    boxElement.labelField.changeFormat("size", 14);// changement de la taille de la police du texte
}

/**
 * On construit un UITextField
 */
m_textField = new UITextField();
m_textField.text = 'Développez Club des Professionnels :)';

m_textField.x = m_textField.y = 5;
m_textField.width = m_textField.maxWidth = 390;
m_textField.height = m_textField.maxHeight = 30;
m_textField.changeFormat("size", 20);// changement de la taille de la police du texte
m_textField.changeFormat("color", Math.random() * 0X00FFFFFF);// changement de la couleur

m_comboFonts.y = m_textField.y + m_textField.height;
m_comboFonts.x = 20;

addChild( m_comboFonts );
addChild( m_textField );
}

/**
 * Fonction servant à écouter le changement de police au sein de la combobox.
 * On applique la police à l'UITextField que l'on a créé et placé sur la
 * scène.
 *
 * @param e Evenement de type MouseEvent.CLICK
 */
private function changeHandler(e:Event):void
{
    m_textField.font = m_comboFonts.currentLabel;
    m_textField.changeFormat("color", Math.random() * 0X00FFFFFF);// changement de la couleur
}

/**
 * Fonction servant à écouter les click de l'utilisateur sur le buttonHide.
 * On modifie le Skin de La List.
 *
 * @param e Evenement de type MouseEvent.CLICK
 */
private function changeSkinList(e:MouseEvent) : void
{
    m_list.applySkin(new DefaultListSkin() );
}

/**

```

```

* Fonction servant à écouter les click de l'utilisateur sur le buttonHide.
* On modifie le Skin de La List.
*
* @param e Evenement de type MouseEvent.CLICK
*/
private function changeSkinList2(e:MouseEvent) : void
{
    m_list.applySkin(new DefaultListSkin2() );
}

/**
* Création des éléments du composant List
*/
private function getListElement(id:Number):Sprite
{
    // création de l'élément
    var oElement:Sprite = new Sprite();

    // ajout d'un fond avec une couleur aléatoire
    var shp:Shape = new Shape();
    var color:int = Math.random() * 0X00FFFFFF;

    shp.graphics.beginFill( color );
    shp.graphics.drawRect( 0, 0, 200, 30 );
    shp.graphics.endFill();

    // ajout d'un texte
    var oTxt:UITextField = new UITextField();

    // définition de la taille du texte
    oTxt.height = oTxt.maxHeight = 25;
    oTxt.width = oTxt.maxWidth = 195;

    oTxt.text = "Élément "+id;
    oTxt.selectable = false;
    oTxt.changeFormat("color", 0xffffffff);// changement de la couleur du texte
    oTxt.changeFormat("size", 14);// changement de la taille de la police du texte
    oTxt.changeFormat("font", "Arial");// changement de la police du texte

    oTxt.alignCenter();

    oTxt.background = false;
    oTxt.backgroundColor = 0xff33ff;

    // ajout à la displaylist (le fond et le texte) de l'élément
    oElement.addChild(shp);
    oElement.addChild(oTxt);

    return oElement;
}
}
}

```

### III-C - Téléchargement du code source de l'exemple

Vous pouvez télécharger le code source complet de notre tutoriel : **Fichier zip Exemple ActionScript-Facile**  
 Ci-dessous, voici le swf que nous venons de créer.

*Cliquer sur l'image ci-dessus pour lancer l'animation flash*

## IV - Tutoriels création des composants

J'ai rédigés plusieurs articles pour vous aider à appréhender au mieux Le Framework de Composants ActionScript-Facile.

### Sujets des articles

- Concevoir vos propres composants.
- Apprendre à utiliser les composants du Framework ActionScript-Facile.
- Implémenter vos fonctionnalités, modifier les composants existants.

### Liens vers les différents articles

- [Chapitre 1 : Pourquoi créer des composants graphiques ?](#)
- [Chapitre 2 : Comment créer des composants ?](#)
- [Chapitre 3 : Les fonctionnalités de base d'une bibliothèque de composants.](#)
- [Chapitre 4 : Création du composant Button.](#)
- [Chapitre 5 : Création du composant VerticalScrollBar.](#)
- [Chapitre 6 : Création du composant List.](#)
- [Chapitre 7 : Création du composant ComboBox.](#)

Il est possible de **concevoir plusieurs composants, de les améliorer**. L'ensemble de ces articles vous fournis une base solide pour : créer votre propre framework de composants ou améliorer Le Framework ActionScript-Facile.

## V - Licence et Versions

Le **Framework AS3 ActionScript-facile** utilise la Licence : **Creative Commons Attribution-NonCommercial-ShareAlike 3.0 Unported**

c'est à dire que vous êtes libres de :

**Partager**, copier et distribuer le Framework AS3 ActionScript-Facile.

**Adapter** le code source à vos besoins

Tout en respectant ces conditions :

**Paternité** - Vous devez citer le nom de l'auteur (Matthieu pour [www.actionscript-facile.com](http://www.actionscript-facile.com)).

**Pas d'Utilisation Commerciale** - Vous ne pouvez pas utiliser cette création à des fins commerciales. C'est à dire que vous n'avez pas la possibilité de revendre ce code source. Par contre, vous pouvez utiliser le Framework ActionScript-Facile pour développer des applications flash (vendues) à vos clients.

**Partage à l'identique** - Si vous modifiez, transformez ou adaptez cette création, vous pouvez distribuer le travail qui en résulte uniquement sous la même licence ou similaire.

Le Framework AS3 ActionScript-Facile est en **Version** : *0.1 - septembre 2010*.

## VI - Pages du projet et téléchargement

**Téléchargez** la dernière version du **Framework ActionScript-Facile** sur **ActionScript-Facile**

## VII - Support - Questions

Pour toutes questions sur ce tutoriel, vous pouvez utiliser le : **Forum Flash Developpez.com**