ActionScript FacileChapitre 6Création du composant AS3 List

par Matthieu (Accueil) (Blog)

Date de publication : 06 novembre 2010

Dernière mise à jour :

Article original : **ActionScript-Facile.com - Création du composant AS3 List**
Commentez cet article :

I - Introduction

Allez c'est parti, continuons avec la conception de notre **Bibliothèque de Composants Graphiques** !
Nous allons créer notre 3ème composant, il s'agit du Composant List.
Dans ce chapitre, vous verrez comment établir un cahier des charges pour un composant de type List. Vous aurez bien évidemment accès aux sources commentées afin de mettre en pratique tout ce que vous aurez appris.

II - CDCF d'un composant List

Un composant de type List peut se **comparer** visuellement à une **maison**.
Dans une maison (en général), l'espace habitable ne se limite pas à ce que l'on peut distinguer par la fenêtre. Depuis l'extérieur : **nous n'en voyons donc qu'une partie**.
Le composant List **est à la fois la maison et la fenêtre.**

Il peut contenir autant d'objets graphiques que l'on souhaite (comme une maison peut contenir un certain nombre de meubles). Toutefois, il ne pourra en afficher qu'une partie (la fenêtre).

Là où le composant List **innove par rapport à la maison**. Effectivement, il est possible de faire **défiler tous les objets graphiques** qu'il contient par le biais d'une ScrollBar (verticale ou horizontale). Ainsi, tout son contenu pourra être visualisé.
C'est un peu comme si sur vous aviez une ScrollBar sur la fenêtre de votre maison qui déplacerait vos meubles de sorte qu'ils arrivent automatiquement devant la fenêtre.
Maintenant que nous avons défini ce qu'est un composant List, nous allons pouvoir nous attaquer à sa **conception**.
Comme pour le **Composant ScrollBar**, le sujet augmente en difficulté au fur et à mesure de la création des Composants AS3 Facile.
Nous allons ainsi nous simplifier la vie en implémentant un composant List :

- qui gère uniquement le scroll vertical.
- qui va éditer pour nous les positions de l'ensemble des objets graphiques qu'on lui envoie de manière à ce que ceux-ci apparaissent à la "queue leu leu".

Un composant List est constitué de 4 entités :

- Un **clip conteneur** (la maison), qui va se charger de réceptionner des objets graphiques. La taille du conteneur augmente et diminue en fonction du nombre d'objets graphiques qu'il contient.
- Un **clip masque** (la fenêtre), qui nous servira à cacher une partie des objets graphiques.
- Un **clip background** (le papier peint) qui servira juste de fond à notre composant, ceci est purement esthétique et vous n'êtes en aucun cas obligé de l'implémenter.
- Une **scrollBar verticale** qui ne s'affichera que lorsque la taille du conteneur dépassera les dimensions de la fenêtre.

Pour compléter notre CDCF, **la taille du masque** doit pouvoir être **éditée**. De plus, le **fond** et la **scrollBar sont personnalisables graphiquement.**
Avec toutes ces informations dans notre CDCF (Cahier Des Charges Fonctionnel), nous pouvons passer à la rédaction du CDCT (Cahier Des Charges Technique).

III - CDCT d'un composant List

Nous allons donc de traduire, en termes techniques, les spécifications fonctionnelles du composant List.
La classe List

- *Un clip conteneur (la maison) va se charger de réceptionner des objets graphiques. La taille du conteneur augmente et diminue en fonction du nombre d'objets graphiques qu'il contient.*

En lieu et place d'un clip, nous allons utiliser un objet Sprite.

- *Un clip masque (la fenêtre), qui nous servira à cacher une partie des objets graphiques.*

Ici, un simple **objet** de type Shape suffira largement.

- *Un clip background (le papier peint) qui servira juste de fond à notre composant. Celui-ci est purement esthétique et vous n'êtes en aucun cas obligé de l'implémenter.*

Un objet de type Sprite qui contiendra le fond du composant List.

- *Une scrollBar verticale qui ne s'affichera que lorsque la taille du conteneur dépassera les dimensions de la fenêtre.*

Un objet de type com.actionscriptfacile.ui.scroll.components.VerticalScrollBar (veuillez lire le Tutoriel 5 sur la **Création du Composant Vertical ScrollBar**).

- *La taille du masque peut être éditée.*

Pour cela, nous allons **overrider** la **méthode** de base **resize** de la classe parente **UIComponent**.

- *Le fond personnalisable graphiquement ainsi que la scrollBar.*

Pour cette fonctionnalité, nous allons effectuer un **override** de la **méthode** de base **applySkin** de la classe parente **UIComponent**.

La classe UIMargins

Et comme toujours, nous allons ajouter une petite difficulté en intégrant une fonctionnalité supplémentaire à notre composant. J'ai nommé : **les marges !**

Le conteneur doit pouvoir appliquer des marges au contenu dans les 4 directions suivantes :

- Haut
- Bas
- Gauche
- Droite

La solution technique reste simple :

- Pour respecter la **marge gauche** nous décalerons **le masque et le conteneur vers la droite**.
- Pour respecter la **marge haute** nous décalerons **le masque et le conteneur vers le bas**.
- Pour respecter la **marge droite** nous réduirons **la largeur du masque**.
- Pour respecter la **marge basse** nous réduirons **la hauteur du masque**.

Pour définir nos marges, nous allons créer une classe nommée **UIMargins**. Elle se trouvera dans le package com.actionscriptfacile.ui.utils.

IV - Conclusion

Et voila, notre composant List est terminé. Regardez le résultat ci-dessous.

Cliquer sur ce lien pour lancer l'animation flash

Dans le prochain tutoriel AS3, nous attaquerons notre dernier composant. Et avec une difficulté accrue : la ComboBox ! (ou liste déroulante).

Évolutions des fonctionnalités

Pour améliorer notre Composant AS3 List, voici quelques exemples de fonctionnalités supplémentaires à implémenter. Proposez vos exemples dans les commentaires.
Allez, maintenant, **c'est à vous de jouer !**

List avec une ScrollBar Horizontale

Ajoutez au composant List la gestion d'une ScrollBar Horizontale.
Dans le tutoriel précédent (**Création d'une VerticalScrollbar**), un des exercices consistait à concevoir une ScrollBar Horizontale. Il suffit de réutiliser cette classe et de l'instancier / afficher (si besoin) par le composant List.

List avec des composants Button

Vous pouvez également remplacer les différents éléments de la List par des objets de type Button (lisez le **Tutoriel 4 : Création du composant Button**).
Puis vous **écoutez les évènements** des objets Button. Et vous avez la possibilité de créer une action spécifique, en fonction du Button sélectionné par l'utilisateur.

Sources commentées

- *com.actionscriptfacile.ui.list.List.as*
- *com.actionscriptfacile.ui.utils.UIMargins.as*
- *ListExample.as*
- *com.as3facileexemple.skin.classic.DefaultListSkin.as*

Vous trouverez ci-dessous l'ensemble des classes créées. Elles sont commentées pour vous permettre de comprendre au mieux leur fonctionnement.
Vous pouvez télécharger le fichier zip : **Component List du Framework actionscript-facile**

Posez vos questions sur le **Forum AS3 Developpez**.

com.actionscriptfacile.ui.list.List.as

```
package  com.actionscriptfacile.ui.list
{
 import com.actionscriptfacile.skin.ISkin;
 import com.actionscriptfacile.ui.scroll.components.VerticalScrollBar;
 import com.actionscriptfacile.ui.UIComponent;
 import com.actionscriptfacile.ui.utils.UIMargins;
 import flash.display.DisplayObject;
 import flash.display.Shape;
 import flash.display.Sprite;

 /**
  * Un composant List est un objet graphique qui peut stocker d'autres objets graphiques au sein d'un clip
  * conteneur qui lui-même, est masqué.
  *
  * A chaque fois qu'un objet est ajouté au composant, celui-
ci se charge d'éditer la position ".y" du dit
  * objet afin de le placer à la suite des autres.
  *
  *
  * @author Matthieu
  */
 public class List extends UIComponent
 {

  static public const LIST_BACKGROUND_SKIN:String = 'LIST_BACKGROUND_SKIN'; // constantes pour le skin

  private var m_scrollBar:VerticalScrollBar; // objet de type VerticalScrollBar
  private var m_containerMask:Shape; // masque du clip conteneur
  private var m_container:Sprite; // conteneur
  private var m_background:Sprite; // background ou "fond" du composant.
  private var m_margins:UIMargins; // objet servant à définir les marges à appliquer au contenu.

  public function List()
  {
   init();
  }

  private function init():void
  {
```

com.actionscriptfacile.ui.list.List.as

```
    m_margins = new UIMargins(); // Marges = 0 dans toutes les directions

    m_scrollBar = new VerticalScrollBar(); // on crée une scrollBar
    m_containerMask = new Shape(); // on crée le mask
    m_container =  new Sprite(); // on crée le conteneur
    m_background = new Sprite(); // on crée le fond

    /**
     * On dessiner le masque
     */
    m_containerMask.graphics.beginFill( 0XFF0000, .5 );
    m_containerMask.graphics.drawRect( 0, 0, 100, 100 );
    m_containerMask.graphics.endFill();

    /**
     * On dessine un fond par défaut
     */
    m_background.graphics.beginFill( 0X999999, .1  );
    m_background.graphics.drawRect( 0, 0, 100, 100 );
    m_background.graphics.endFill();

    /**
     * Puis nous ajoutons tout les éléments de notre composant que nous avons crée
     * au préalable à sa displayList
     */
    addChild( m_scrollBar );
    addChild( m_background );
    addChild( m_container );
    addChild( m_containerMask );

    // on applique le masque au conteneur
    m_container.mask = m_containerMask;

    // on cache la scrollBar
    m_scrollBar.hide();

  }

  /**
   * Fonction permettant de gérer la customisation des différents éléments
   * graphiques d'un objet de type List
   *
   * @param p_skin Objet implémentant l'interface ISkin
   */
  override public function applySkin( p_skin:ISkin ):void
  {

    // on applique la skin à la scrollBar
    m_scrollBar.applySkin( p_skin );

    // nous récuperons la définition de classe pour le background de l'objet List
    var definition:Class = p_skin.getSkin( LIST_BACKGROUND_SKIN ) as Class;

    // si la définition a été trouvée
    if ( definition )
    {
     // nous vidons proprement l'objet background
     var child:DisplayObject;
     while ( m_background.numChildren > 0 )
     {
      child = m_background.removeChildAt( 0 );
      child = null;
     }
     // puis nous ajoutons une instance de cette définition au conteneur background
     m_background = addChildAt( new definition(), 0 ) as Sprite;
    }
  }

  /**
   *
```

com.actionscriptfacile.ui.list.List.as

```
     * Fonction permettant d'ajouter un clip graphique au conteneur.
     *
     * @param p_element Un objet de type flash.display.DisplayObject
     * @return Retourne une référence vers le DisplayObject ajouté.
     */
    public function addElement( p_element:DisplayObject ):DisplayObject
    {
     // On place p_element dans la displayList du conteneur
     m_container.addChild( p_element );

     // puis on effectue un redraw.
     redraw();

     // enfin, nous retournons l'élément envoyé en paramètre
     return p_element;
    }

    /**
     *
     * Fonction permettant d'enlever un clip graphique au conteneur.
     *
     * @param p_element Un objet de type flash.display.DisplayObject
     * @return Retourne une référence vers le DisplayObject supprimé de la displayList du conteneur.
     */
    public function removeElement( p_element:DisplayObject ):DisplayObject
    {
     // on enlève p_element de la displayList du conteneur
     m_container.removeChild( p_element );

     // on effectue un redraw
     redraw();

     // puis on retourne l'élément envoyé en paramètre
     return p_element;
    }

    /**
     *
     * Fonction interne permettant d'actualiser l'état graphique des différents
     * éléments d'un composant de type List
     *
     * @param p_width Nouvelle taille en largeur du composant
     * @param p_height Nouvelle taille en hauteur du composant
     */
    private function redraw( p_width:Number = -1, p_height:Number = - 1 ):void
    {

     // on effectue des contrôles sur les paramètres envoyés
     if ( p_width < 0 ) p_width = m_background.width;
     if ( p_height < 0 ) p_height = m_background.height;

     var curY:Number = 0; // y maximum courant, au départ il est égal à 0
     var max:int = m_container.numChildren; // nombre d'enfants du clip conteneur
     var i:int = 0;
     var child:DisplayObject;

     for ( ; i < max; i++ )
     {
      // nous récupérons l'enfant du conteneur à l'index i
      child = m_container.getChildAt( i );

      // nous actualisons sa position en y ( verticale )
      child.y = curY;

      // puis on ajoute la hauteur de l'enfant à la variable curY
      // ainsi le prochain enfant sera placé juste après de façon verticale.
      curY += child.height;
     }

     // On actualise la largeur et la hauteur du background
     m_background.width = p_width;
```

com.actionscriptfacile.ui.list.List.as

```
    m_background.height = p_height;

    // la largeur du masque est égale à la largeur passée en paramètre - la marge
    // de gauche + celle de droite
    m_containerMask.width = p_width - m_margins.marginLeft - m_margins.marginRight;

    // la hauteur du masque est égale à la hauteur passée en paramètre - la marge
    // du haut + celle du bas
    m_containerMask.height = p_height - m_margins.marginTop - m_margins.marginBottom;

    // si la taille du conteneur ( la maison ) ne dépasse pas les dimensions du masque
    // ( la fenêtre ) alors
    if ( m_container.height < m_containerMask.height )
    {
     // on cache la scrollBar
     m_scrollBar.hide();
    }
    else // sinon
    {
     // on affiche la scrollBar
     m_scrollBar.show();

     // la largeur du masque diminue encore pour laisser de la place afin de laisse de la
     // place à la scrollBar
     m_containerMask.width -= m_scrollBar.width;

     // on positionne la scrollBar juste à droite du masque
     m_scrollBar.x = m_background.width - m_scrollBar.width - m_margins.marginRight;

     // enfin on actualise la hauteur de la scrollBar
     m_scrollBar.resize( m_scrollBar.width, m_containerMask.height );
    }

    // on positionne le masque et le conteneur de façon à respecter la marge de gauche
    m_containerMask.x = m_container.x = m_margins.marginLeft;

    // Même chose pour la marge du haut avec la scrollBar en plus
    m_scrollBar.y = m_containerMask.y = m_container.y = m_margins.marginTop;

    // on dit bien à la scrollBar que le contenu qu'elle doit scroller est le conteneur...
    m_scrollBar.content = m_container;

    // et que la zone de scroll correspond aux dimensions du masque ( la fenêtre ).
    m_scrollBar.scrollArea = m_containerMask.getRect(this);

   }

   /**
    *
    * Fonction permettant de gérer intelligemment le redimensionnement
    * d'un objet de type List
    *
    * @param p_width  la nouvelle largeur ( en pixels ) du composant
    * @param p_height la nouvelle hauteur ( en pixels ) du composant
    */
   override public function resize( p_width:Number, p_height:Number ):void
   {
    // on effectue un redraw
    redraw( p_width, p_height );
   }

   /**
    * Retourne une référence de l'objet VerticalScrollBa utilisé au sein du composant
    */
   public function get scrollBar():VerticalScrollBar { return m_scrollBar; }

   /**
    * Retourne une référence de l'objet UIMargins utilisé pour définir les marges des contenus
    */
   public function get margins():UIMargins { return m_margins; }
```

com.actionscriptfacile.ui.list.List.as

```
    /**
     * Définit les marges à appliquer aux contenus
     */
    public function set margins( p_margins:UIMargins ):void
    {
     // on change les margins actuelles puis on effectue un redraw
     m_margins = p_margins;
     redraw( m_background.width, m_background.height );
    }

    /**
     * Détruit proprement un objet de type List
     */
    override public function destroy():void
    {
     // on tue proprement l'objet UIMargins
     m_margins = null;

     // on tue proprement la scrollBar
     m_scrollBar.destroy();

     /**
      * Nous tuons tous les enfants du conteneur
      */
     var child:DisplayObject;

     while ( m_container.numChildren > 0 )
     {
      child = m_container.removeChildAt( 0 );

      if ( child is UIComponent )
      {
       UIComponent( child ).destroy();
      }

      child = null;
     }

     // Enfin, on supprime le conteneur, son masque et le fond
     removeChild( m_container );
     removeChild( m_containerMask );
     removeChild( m_background );

     m_container = null;
     m_containerMask = null;
     m_background = null;
     m_scrollBar = null;

     // on appelle la fonction de destruction parente par précaution
     super.destroy();

    }
   }

  }
```

com.actionscriptfacile.ui.utils.UIMargins.as

```
  package  com.actionscriptfacile.ui.utils
  {
   /**
    * Permet de définir des marges à tout type d'élément
    *
    * @author Matthieu
    */
   public class UIMargins
   {

    private var m_marginLeft:Number;
    private var m_marginRight:Number;
    private var m_marginBottom:Number;
```

com.actionscriptfacile.ui.utils.UIMargins.as

```
   private var m_marginTop:Number;

   public function UIMargins( p_margTop:Number = 0, p_margRight:Number = 0, p_margBottom:Number = 0,
            p_margLeft:Number = 0 )
   {
    m_marginTop = p_margTop;
    m_marginLeft = p_margLeft;
    m_marginBottom = p_margBottom;
    m_marginRight = p_margRight;
   }

   public function get marginLeft():Number { return m_marginLeft; }
   public function set marginLeft(value:Number):void { m_marginLeft = value; }

   public function get marginRight():Number { return m_marginRight; }
   public function set marginRight(value:Number):void { m_marginRight = value; }

   public function get marginBottom():Number { return m_marginBottom; }
   public function set marginBottom(value:Number):void { m_marginBottom = value; }

   public function get marginTop():Number { return m_marginTop; }
   public function set marginTop(value:Number):void { m_marginTop = value; }

  }

 }
```

ListExample.as

```
 package
 {
  import com.actionscriptfacile.ui.text.UITextField;
  import com.actionscriptfacile.ui.list.List;
  import com.actionscriptfacile.ui.utils.UIMargins;
  import com.as3facileexemple.skin.classic.DefaultListSkin;// import du skin de la VerticalScrollBar
  import flash.display.Shape;
  import flash.display.Sprite;

  /**
   * Exemple d'utilisation du composant List.
   * @author Matthieu
   */
  public class ListExample extends Sprite
  {

   public function ListExample()
   {
    // création d'une liste
    var list: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 au compilateur ]
    list.applySkin( new DefaultListSkin() );

    // 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) );
    }

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

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

ListExample.as

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

   /**
    * 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.selectable = false;
    oTxt.changeFormat("color", 0xffffff);// 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.text = "Élément "+id;
    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;
   }

  }

 }
```

com.as3facileexemple.skin.classic.DefaultListSkin.as

```
 package com.as3facileexemple.skin.classic
 {

  import com.actionscriptfacile.skin.Skin;
  import com.actionscriptfacile.ui.list.List;
  import com.actionscriptfacile.ui.scroll.components.VerticalScrollBar;

  // Import des classes gérant la partie graphique du composant dans le fla (movieclip)
  // Provient de ui.swc (créé avec la compilation de UI.fla)
  import com.as3facile.skin.list.ListBackgroundSkin;
  import com.as3facile.skin.scroll.ScrollBarBackgroundSkin;
  import com.as3facile.skin.scroll.ScrollBottomButtonSkin;
  import com.as3facile.skin.scroll.ScrollerButtonSkin;
  import com.as3facile.skin.scroll.ScrollUpButtonSkin;

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

com.as3facileexemple.skin.classic.DefaultListSkin.as

```
  {

   public function DefaultListSkin()
   {
    setSkin( List.LIST_BACKGROUND_SKIN, ListBackgroundSkin  );
    setSkin( VerticalScrollBar.SCROLL_VERTICAL_BACKGROUND_SKIN, ScrollBarBackgroundSkin );
    setSkin( VerticalScrollBar.SCROLL_VERTICAL_BOTTOM_SKIN, ScrollBottomButtonSkin );
    setSkin( VerticalScrollBar.SCROLL_VERTICAL_UP_SKIN, ScrollUpButtonSkin );
    setSkin( VerticalScrollBar.SCROLL_VERTICAL_SCROLLER_SKIN, ScrollerButtonSkin );
   }

  }

 }
```

V - Remerciements

Je tiens ici à remercier **La Rédactrice Kalyparker** pour la mise au gabarit de l'article original (**ActionScript-Facile.com - Création du composant AS3 List**) au format Developpez.com.
Merci beaucoup à l'Equipe Developpez.com de contribuer à la diffusion du Framework ActionScript-Facile.
Rendez-vous sur **ActionScript-Facile** pour consulter **des tutoriels**, télécharger **des codes sources supplémentaires** et recevoir **un guide sur l'utilisation de l'Editeur ActionScript FDT**.