ActionScript FacileChapitre 7Création du composant AS3 ComboBox

par Matthieu (Accueil) (Blog)

Date de publication : 09 novembre 2010

Dernière mise à jour :

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

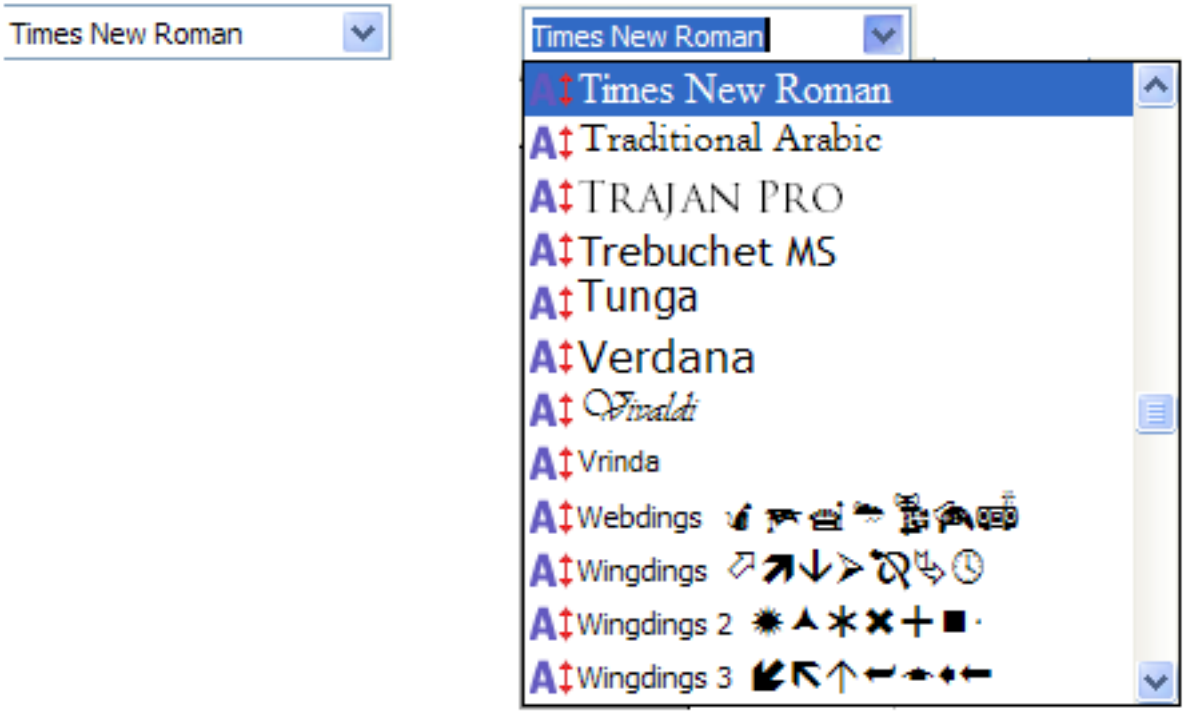
I - Introduction

En route pour la création du 4ème composant de notre **Bibliothèque AS3 de Composants Graphiques !**
J'ai nommée la fameuse ComboBox ou **Liste Déroulante**.
Comme d'habitude, dans ce chapitre, vous verrez **comment établir le cahier des charges de la ComboBox**. Et vous aurez bien évidemment accès aux sources commentées afin de mettre en pratique tout ce que vous aurez appris.

II - CDCF d'un composant ComboBox

*Un composant de type ComboBox est en en fait **une liste déroulante**. Il y a en sur la majorité des sites internet. Elle **se compose** d'**un bouton** et d'un composant de type List.*
La valeur du bouton change suivant l'élément qui est sélectionné dans la List.

Une fois n'est pas coutume, nous allons illustrer notre description :



Ici nous pouvons reconnaître deux composants que nous avons déjà développés:

- le **Button** (la partie qui contient le texte "Times New Roman")
- la **List** (le rectangle qui contient une liste de polices de caractères et qui peut être scrollée).

La rédaction du CDCF (Cahier Des Charges Fonctionnel) est donc quasiment terminé.
Nous allons juste ajouter une petite contrainte :

- Tous **les éléments** du composant List ont tous **la même hauteur et le même style**. Nous conservons ce comportement graphique cohérent.

III - CDCT d'un composant ComboBox

Traduisons en terme technique le CDCF (les spécifications fonctionnelles) du composant ComboBox.
Eh oui, c'est parti pour la rédaction du CDCT (Cahier Des Charges Techniques).

*Un composant de type ComboBox est une **liste déroulante** comme vous pouvez en voir sur n'importe quel site internet. Elle se **compose** d'**un bouton** et d'**un composant de type List**.*

La Classe ComboBoxButton

Le bouton représente le titre de la ComboBox. Pour cela nous développons une classe **ComboBoxButton**.
Aucun problème pour le bouton, nous avons déjà développé auparavant les composants graphiques nécessaires.

- *La valeur du bouton change suivant l'élément qui est sélectionné dans la List.*

Nous **écouterons** donc l'évènement **MOUSE_CLICK** de type flash.events.MouseEvent sur le composant List.
La classe ComboBoxButton reprend les fonctionnalités de la classe **Button** en ajoutant les suivantes :

- **un getter / setter** permettant de récupérer la valeur (un objet) d'un ComboBoxButton.
- la **gestion du skin** graphique propre à ComboBoxButton.

La Classe ComboBoxElement

- *Tous les éléments du composant List possèdent une hauteur et un style identique, nous devons conserver ce comportement graphique cohérent.*

Pour cela, nous créons une classe **ComboBoxElement** qui reprend les fonctionnalités de la classe **Button**.
Par extension, elle héritera également des fonctionnalités de la classe de base **UIComponent**.
Ainsi, tous les éléments qui seront ajoutés dans notre List seront du même type et possèderont tous une apparence identique.
Les **éléments** de la List de la ComboBox sont bien évidemment personnalisables graphiquement avec 3 états différents :

- **Normal** : L'**élément** est « **au repos** », il n'est ni survolé, ni cliqué.
- **Survol** : L'utilisateur passe et laisse sa **souris au dessus de l'élément** sans toutefois cliquer dessus.
- **Cliqué ou clic maintenu** : L'utilisateur réalise un clic gauche et laisse le **bouton de sa souris maintenu sur l'élément**.

IV - Conclusion

Voilà, **c'est terminé, toutes mes félicitations** pour avoir **suivi** et **appliqué** l'ensemble des **Tutoriels AS3**.
Vous venez de **créer** une **Bibliothèque AS3 de Composants Graphiques** ou plus communément appelé un **Framework de Composants AS3**.
La ComboBox est le dernier composant graphique que nous avions à développer et qui nécessitait tant d'efforts.

Cliquer sur ce lien pour lancer l'animation flash

Les prochains tutoriels seront des bonus.
C'est un peu ma façon de vous remercier et de vous encourager à développer vos propres composants graphiques.
Soit en apportant des améliorations au **Framework AS3 Facile** ou en développant votre propre **Framework de Composants**.

J'ai pris beaucoup de plaisir à rédiger ces articles et j'espère vous avoir communiqué l'envie d'aller plus loin. Les voies à explorer sont nombreuses et la programmation graphique passionnante.

Évolutions des fonctionnalités

Pour améliorer notre Composant AS3 ComboBox, voici quelques exemples de fonctionnalités supplémentaires à implémenter.
Allez, maintenant, **c'est à vous de coder** !

Méthode resizeListOnly

Je vous propose d'ajouter la méthode **resizeListOnly()** à notre composant ComboBox.
resizeListOnly ajoute la fonctionnalité suivante :

- permet de déterminer les **dimensions** de la List indépendamment du Titre (**ComboBoxButton**). En quelque sorte, la liste déroulante qui apparait sous la ComboBox peut être plus large que son titre. Pensez également à redimensionner les éléments de la List !

Création d'un menu en Flash
Avec le composant ComboBox et la méthode **resizeListOnly**, vous pouvez **créer un menu de navigation** tout simple. Une des solutions (il y a en plusieurs) consiste à utiliser plusieurs ComboBox, positionnées les unes à côté des autres et interagissant entre elles.
Si vous concevez souvent des menus pour vos applications Flash, je vous conseille de créer un composant Menu.

*Proposez vos exemples de code et posez vos questions sur le **Forum AS3 Developpez**.*

Sources commentées

- *com.actionscriptfacile.ui.combo.element.ComboBoxElement*
- *com.actionscriptfacile.ui.combo.element.ComboBoxButton*
- *com.actionscriptfacile.ui.combo.ComboBox*
- *ComboExample.as*
- *com.as3facileexemple.skin.classic.DefaultComboBoxSkin.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 ComboBox du Framework actionscript-facile**

com.actionscriptfacile.ui.combo.element.ComboBoxElement.as

```
package com.actionscriptfacile.ui.combo.element
{
 import com.actionscriptfacile.ui.button.Button;
 import com.actionscriptfacile.skin.ISkin;

 /**
  * Définition d'un élément ajouté dans la liste de la ComboBox.
  * Chaque élément est de type Button.
  *
  * @author Matthieu
  */
 public class ComboBoxElement extends Button
 {

  /**
   * On définit de nouvelles constantes pour le skin afin de ne pas les confondre
   * avec celle d'un composant Button classique, sémantiquement parlant, c'est plus propre
   * d'agir de la sorte.
   */
  public static const COMBOBOX_ELEMENT_OVER_SKIN:String = 'COMBOBOX_ELEMENT_OVER_SKIN';
  public static const COMBOBOX_ELEMENT_UP_SKIN:String = 'COMBOBOX_ELEMENT_UP_SKIN';
  public static const COMBOBOX_ELEMENT_DOWN_SKIN:String = 'COMBOBOX_ELEMENT_DOWN_SKIN';

  protected var m_value:Object; // la valeur courante

  public function ComboBoxElement( p_value:Object, p_label:String )
  {
   super();
   m_value = p_value;
   label = p_label;
  }

  /**
   * Getter / Setter
   *
   * Permet de récupérer / définir la propriété value d'un objet de type ComboBoxElement.
   */
  public function get value():Object { return m_value; }
```

com.actionscriptfacile.ui.combo.element.ComboBoxElement.as

```
   public function set value(value:Object):void { m_value = value; }

   /**
    *
    * @param p_skin Objet implémentant l'interface ISkin
    */
   override public function applySkin(p_skin:ISkin):void
   {

    /**
     * On procède exactement de la même façon qu'avec le composant Button à la différence près
     * que l'on n'utilise pas les mêmes constantes.
     */
    var definition:Class;

    definition = p_skin.getSkin( COMBOBOX_ELEMENT_OVER_SKIN ) as Class;

    if ( definition != null )
     m_overStyle = new definition();

    definition = p_skin.getSkin( COMBOBOX_ELEMENT_DOWN_SKIN ) as Class;

    if ( definition != null )
     m_downStyle = new definition();

    definition = p_skin.getSkin( COMBOBOX_ELEMENT_UP_SKIN ) as Class;

    if ( definition != null )
     m_upStyle = new definition();

    addChild( m_upStyle );
    addChild( m_downStyle );
    addChild( m_overStyle );

    setState ( UP_STATE );
   }

   /**
    * Tue proprement Un objet de type ComboBoxElement
    */
   override public function destroy():void
   {
    m_value = null;
    super.destroy();
   }

  }

 }
```

com.actionscriptfacile.ui.combo.element.ComboBoxButton.as

```
 package com.actionscriptfacile.ui.combo.element
 {
  import com.actionscriptfacile.ui.button.Button;
  import com.actionscriptfacile.skin.ISkin;

  /**
   * Définition de l'élément courant sélectionné dans la liste déroulante.
   * Le titre de la ComboBox.
   * Cet élément est de type Button.
   *
   * @author Matthieu
   */
  public class ComboBoxButton extends Button
  {

   /**
    * On définit les constantes pour le skin
    */
```

com.actionscriptfacile.ui.combo.element.ComboBoxButton.as

```
    public static const COMBOBOX_BUTTON_OVER_SKIN:String = 'COMBOBOX_BUTTON_OVER_SKIN';
    public static const COMBOBOX_BUTTON_UP_SKIN:String = 'COMBOBOX_BUTTON_UP_SKIN';
    public static const COMBOBOX_BUTTON_DOWN_SKIN:String = 'COMBOBOX_BUTTON_DOWN_SKIN';

    // variable qui va nous servir à stocker la valeur de l'objet ComboBoxButton
    protected var m_value:Object;

    public function ComboBoxButton( p_value:Object, p_label:String )
    {
     super();

     m_value = p_value;
     label = p_label;
    }

    /**
     * Définit / récupère définit la valeur de l'objet comboboxbutton
     */
    public function get value():Object { return m_value; }
    public function set value(value:Object):void { m_value = value; }

    /**
     *
     * @param p_skin Objet implémentant l'interface ISkin
     */
    override public function applySkin(p_skin:ISkin):void
    {

     /**
      * Même procédé que pour les autres composants
      */
     var definition:Class;

     // over skin
     definition = p_skin.getSkin( COMBOBOX_BUTTON_OVER_SKIN ) as Class;

     if ( definition != null )
      m_overStyle = new definition();

     // down skin
     definition = p_skin.getSkin( COMBOBOX_BUTTON_DOWN_SKIN ) as Class;

     if ( definition != null )
      m_downStyle = new definition();

     // up skin
     definition = p_skin.getSkin( COMBOBOX_BUTTON_UP_SKIN ) as Class;

     if ( definition != null )
      m_upStyle = new definition();

     // on ajoute les styles à la display list
     addChild( m_upStyle );
     addChild( m_downStyle );
     addChild( m_overStyle );

     setState ( UP_STATE );
    }

    /**
     * Fonction servant à détruire proprement un objet de type ComboBoxButton
     */
    override public function destroy():void
    {
     m_value = null;
     super.destroy();
    }

  }

 }
```

com.actionscriptfacile.ui.combo.ComboBox.as

```
package com.actionscriptfacile.ui.combo
{
 import com.actionscriptfacile.skin.ISkin;
 import com.actionscriptfacile.ui.combo.element.ComboBoxButton;
 import com.actionscriptfacile.ui.combo.element.ComboBoxElement;
 import com.actionscriptfacile.ui.list.List;
 import com.actionscriptfacile.ui.UIComponent;
 import com.actionscriptfacile.ui.utils.UIMargins;

 import flash.events.MouseEvent;

 /**
  * Composant ComboBox
  * @author Matthieu
  */
 public class ComboBox extends UIComponent
 {
  private var m_skin : ISkin; // objet implémentant l'interface ISkin

  /**
   * On définit les constantes de skin
   */
  public static const COMBOBOX_BUTTON_OVER_SKIN : String = 'COMBOBOX_BUTTON_OVER_SKIN';
  public static const COMBOBOX_BUTTON_UP_SKIN : String = 'COMBOBOX_BUTTON_UP_SKIN';
  public static const COMBOBOX_BUTTON_DOWN_SKIN : String = 'COMBOBOX_BUTTON_DOWN_SKIN';
  public static const COMBOBOX_ELEMENT_OVER_SKIN : String = 'COMBOBOX_ELEMENT_OVER_SKIN';
  public static const COMBOBOX_ELEMENT_UP_SKIN : String = 'COMBOBOX_ELEMENT_UP_SKIN';
  public static const COMBOBOX_ELEMENT_DOWN_SKIN : String = 'COMBOBOX_ELEMENT_DOWN_SKIN';
  // le currentElement est celui qui apparaîtra même lorsque le composant List sera caché
  private var m_currentElement : ComboBoxButton;
  //composant List
  private var m_list : List;
  // tableau des éléments
  private var m_elements : Array;
  // hauteur des éléments
  protected var m_componentsHeight : Number;

  public function ComboBox()
  {
   init();
  }

  /**
   * Fonction interne servant à initialiser un composant de type ComboBox
   */
  private function init() : void
  {
   // on définit une taille d'élément par défaut
   m_componentsHeight = 30;

   // l'objet skin est null
   m_skin = null;

   // le currentElement est crée
   m_currentElement = new ComboBoxButton(null, '');

   // le composant List aussi
   m_list = new List();

   // le tableau d'éléments est crée
   m_elements = new Array();

   // on ajoute à la displayList l'élément courant et le composant List
   addChild(m_currentElement);
   addChild(m_list);

   // on positionne la liste juste en dessous de l'élément courant
   m_list.y = m_currentElement.height;

   // et on passe son visible à false
   m_list.visible = false;
```

com.actionscriptfacile.ui.combo.ComboBox.as

```
    // on écoute les click sur la liste, comme ça on sait quel élément est cliqué
    m_list.addEventListener(MouseEvent.CLICK, chooseHandler, true);


   // on écoute les rollOut comme ça dès que l'on quitte les limites de notre composant, on peut cacher la List
    addEventListener(MouseEvent.ROLL_OUT, toggleListAppearHandler, false);

    // si on clique sur l'élément courant on fait apparaître la List
    m_currentElement.addEventListener(MouseEvent.CLICK, toggleListAppearHandler, false);
   }

   /**
    * Fonction interne gérant l'apparition / disparition du composant List
    * @param e Evenement de type MouseEvent
    */
   private function toggleListAppearHandler(e : MouseEvent) : void
   {
    // si on a cliqué sur l'élément courant alors on affiche la List sinon on ne l'affiche pas
    m_list.visible = ( e.type == MouseEvent.CLICK );
   }

   /**
    *
    * Fonction interne gérant le choix d'un élément dans la List
    * @param e Evenement de type MouseEvent
    */
   private function chooseHandler(e : MouseEvent) : void
   {
    // si l'origine du click ne provient pas d'un élement de la liste on ne fait rien
    if ( !( e.target is ComboBoxElement ) )
     return;

    // sinon on récupère cet élément
    var element : ComboBoxElement = e.target as ComboBoxElement;

    // et on affecte les valeurs de cet élément à l'élément courant
    m_currentElement.value = element.value;
    m_currentElement.label = element.label;

    // et cache la List
    m_list.visible = false;
   }

   /**
    *
    * Ajoute un élément à la liste déroulante et renvoit l'élément de la liste ainsi crée.
    *
    * @param p_label Le label ( texte ) qui sera affiché à l'utilisateur pour l'élément crée
    * @param p_value La valeur de l'élément crée
    * @return L'élément de la liste nouvellement crée de type ComboBoxElement
    */
   public function addElement( p_label : String, p_value : Object ) : ComboBoxElement
   {
    // on ajoute un nouvel élément à la liste auquel on attribue les valeurs passées en paramètre
    var element : ComboBoxElement = m_list.addElement(new ComboBoxElement(p_value, p_label)) as
ComboBoxElement;

    // Si la skin courante est définie on l'applique à l'élément
    if ( m_skin != null )
     element.applySkin(m_skin);

    // on redimensionne l'élément puis on l'ajoute au tableau d'éléments
    element.resize(m_currentElement.width, m_componentsHeight);
    m_elements.push(element);


   // la valeur de l'élément courant est null ou vide on prend les valeurs du premier élément de la liste
    if ( m_currentElement.value == null && m_currentElement.label == '' )
    {
     m_currentElement.value = ComboBoxElement(m_elements[ 0 ]).value;
```

com.actionscriptfacile.ui.combo.ComboBox.as

```
     m_currentElement.label = ComboBoxElement(m_elements[ 0 ]).label;
    }

    // on retourne l'élément ainsi crée
    return element;
   }

   /**
    *
    * Enlève un élément de la liste déroulante et le retourne
    *
    * @param p_element Objet de type ComboBoxElement
    * @return l'élément supprimé de la List
    */
   public function removeElement( p_element : ComboBoxElement ) : ComboBoxElement
   {
    var element : ComboBoxElement = m_list.removeElement(p_element) as ComboBoxElement;
    m_elements.splice(element);
    return element;
   }

   /**
    *
    * Retourne un objet de type ComboBoxElement dont la propriété p_prop a pour valeur p_value
    *
    * @param p_prop Nom de la propriété
    * @param p_value Valeur de la propriété
    * @return Un objet de type ComboBoxElement
    */
   public function getElementByProperty( p_prop : String, p_value : Object ) : ComboBoxElement
   {
    var max : int = m_elements.length;
    var i : int = 0;

    for ( ;i < max;i++)
    {
     if ( ComboBoxElement(m_elements[ i ]).hasOwnProperty(p_prop) && ComboBoxElement(m_elements[ i ])[
p_prop ] == p_value )
     {
      return m_elements[ i ] as ComboBoxElement;
     }
    }

    return null;
   }

   /**
    * Définit / récupère la valeur courante
    */
   public function set currentValue( p_value : Object ) : void
   {
    m_currentElement.value = p_value;
   }

   public function get currentValue() : Object
   {
    return m_currentElement.value;
   }

   /**
    * Définit / récupère le label courant
    */
   public function set currentLabel( p_label : String ) : void
   {
    m_currentElement.label = p_label;
   }

   public function get currentLabel() : String
   {
    return m_currentElement.label;
   }
```

com.actionscriptfacile.ui.combo.ComboBox.as

```
/**
 * Définit / récupère les marges de la liste déroulante
 */
public function set margins( p_margins : UIMargins ) : void
{
 m_list.margins = p_margins;
}

public function get margins() : UIMargins
{
 return m_list.margins;
}

/**
 * Retourne le tableau d'éléments
 */
public function get elements() : Array
{
 return m_elements;
}

/**
 * Définit / récupère la hauteur des éléments de la liste déroulante
 */
public function get componentsHeight() : Number
{
 return m_componentsHeight;
}

public function set componentsHeight(value : Number) : void
{
 m_componentsHeight = value;
 resize(width, height);
}

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

 /**
  * On applique la skin au ComboBoxButton ( l'élément courant ), à la scrollBar, à la List, et à tout les éléme
  * de la List
  */
 m_skin = p_skin;

 m_currentElement.applySkin(p_skin);
 m_list.applySkin(p_skin);
 m_list.scrollBar.applySkin(p_skin);

 var i : int = 0;
 var max : int = m_elements.length;

 for ( ;i < max;i++ )
 {
  ComboBoxElement(m_elements[i]).applySkin(p_skin);
 }

 // puis on redimensionne
 resize(m_currentElement.width, 100);
}

/**
 *
 * Fonction permettant de gérer intelligemment le redimensionnement
```

com.actionscriptfacile.ui.combo.ComboBox.as

```
     * d'un objet de type ComboBox
     *
     * @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
    {

     var i : int = 0;
     var max : int = m_elements.length;

     // on redimensionne la List
     m_list.resize(p_width, p_height);

     // on redimensionne l'élémen courant
     m_currentElement.resize(p_width, m_componentsHeight);

     // ... ainsi que tous les éléments de la liste
     for ( ;i < max;i++ )
     {
      ComboBoxElement(m_elements[ i]).resize(p_width, m_componentsHeight);
     }

     // on repositionne la liste
     m_list.y = m_currentElement.height;
    }

    /**
     * Tue proprement un objet de type ComboBox
     */
    override public function destroy() : void
    {
     /**
      * On tue tous les event listeners
      */
     m_list.addEventListener(MouseEvent.CLICK, chooseHandler, true);
     removeEventListener(MouseEvent.MOUSE_OUT, toggleListAppearHandler, false);
     m_currentElement.removeEventListener(MouseEvent.CLICK, toggleListAppearHandler, false);

     // puis on détruit l'élément courant, la List et le tableau d'éléments
     m_currentElement.destroy();
     m_list.destroy();
     m_elements = null;

     // on appelle la fonction destructrice parente
     super.destroy();
    }
   }
}
```

ComboExample.as

```
package
{
 import com.actionscriptfacile.ui.combo.element.ComboBoxElement;

 import com.actionscriptfacile.ui.utils.UIMargins;
 import com.as3facileexemple.skin.classic.DefaultComboBoxSkin;// import du skin de la ComboBox
 import com.actionscriptfacile.ui.combo.ComboBox;
 import flash.display.Sprite;

 /**
  * Exemple d'utilisation du composant ComboBox.
  * @author Matthieu
  */
 public class ComboExample extends Sprite
 {
  public function ComboExample()
  {
   // création d'une combobox
   var box:ComboBox = new ComboBox();
```

ComboExample.as

```
    // on lui applique la skin par défaut
    box.applySkin( new DefaultComboBoxSkin() );

    var boxElement:ComboBoxElement;
    // ajout des éléments
    for ( var i:int = 0; i < 35; i++ )
    {
     boxElement = box.addElement( "Élément "+ new String( i ), i );
     // acccès au composant de type UITextField (labelField)
     boxElement.labelField.alignCenter(); // centre le texte
     boxElement.labelField.changeFormat("color", 0xff33ff);// changement de la couleur du texte
     boxElement.labelField.changeFormat("size", 14);// changement de la taille de la police du texte
     boxElement.labelField.changeFormat("font", "Arial");// changement de la police du texte
    }

    // définition de la taille de la combobox
    box.resize( 230, 120 );

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

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

    // affichage - ajout à la displaylist
    addChild( box );
   }

  }

 }
```

com.as3facileexemple.skin.classic.DefaultComboBoxSkin.as

```
 package com.as3facileexemple.skin.classic
 {
  // 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.combo.ComboBoxButtonDownSkin;
  import com.as3facile.skin.combo.ComboBoxButtonOverSkin;
  import com.as3facile.skin.combo.ComboBoxButtonSkin;
  import com.as3facile.skin.combo.ComboBoxElementDownSkin;
  import com.as3facile.skin.combo.ComboBoxElementOverSkin;
  import com.as3facile.skin.combo.ComboBoxElementSkin;
  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;

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

  /**
   * Définition du skin utilisé pour un composant ComboBox
   *
   * @author Matthieu
   */
  public class DefaultComboBoxSkin extends Skin
  {

   public function DefaultComboBoxSkin()
   {
    setSkin( ComboBox.COMBOBOX_BUTTON_DOWN_SKIN , ComboBoxButtonDownSkin );
    setSkin( ComboBox.COMBOBOX_BUTTON_OVER_SKIN, ComboBoxButtonOverSkin );
    setSkin( ComboBox.COMBOBOX_BUTTON_UP_SKIN , ComboBoxButtonSkin );
```

com.as3facileexemple.skin.classic.DefaultComboBoxSkin.as

```
    setSkin( ComboBox.COMBOBOX_ELEMENT_DOWN_SKIN , ComboBoxElementDownSkin );
    setSkin( ComboBox.COMBOBOX_ELEMENT_OVER_SKIN, ComboBoxElementOverSkin );
    setSkin( ComboBox.COMBOBOX_ELEMENT_UP_SKIN , ComboBoxElementSkin );

    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 ComboBox**) 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**.