UTILISATION DES COMPOSANTS ACTIONSCRIPT™ 3.0

Adobe

Utilisation des composants ActionScript™ 3.0

Adobe Systems Incorporated, 345 Park Avenue, San Jose, Californie 95110, Etats-Unis.

Table des matières

Introduction

Adobe® Flash® CS3 Professional est l'outil de programmation standard pour la création de contenu Web percutant. La création de ces applications Internet enrichies repose sur des unités élémentaires appelées composants. Un *composant* est un clip qui contient des paramètres permettant de personnaliser le composant pendant la phase de programmation dans Flash ou lors de l'exécution à l'aide des méthodes, des propriétés et des événements ActionScript™. Les composants sont conçus pour permettre aux développeurs de réutiliser et de partager du code. Ils permettent également d'encapsuler une fonctionnalité complexe que les concepteurs peuvent utiliser et personnaliser sans avoir à se servir d'ActionScript.

Les composants vous permettent de créer facilement et rapidement des applications robustes à la présentation et au comportement cohérents. Ce manuel explique comment créer des applications avec les composants ActionScript 3.0. Le manuel intitulé *Guide de référence du langage et des composants ActionScript 3.0* décrit tous les composants, ainsi que l'interface de programmation (API) de chacun d'entre eux.

Vous pouvez utiliser les composants créés par Adobe, télécharger des composants créés par d'autres développeurs ou créer vos propres composants.

Ce chapitre contient les sections suivantes :

Public visé

Ce manuel est destiné aux développeurs qui créent des applications Flash et qui souhaitent exploiter des composants pour accélérer le développement. Il demande des notions de développement d'applications dans Flash et d'écriture de code ActionScript.

Si vous avez peu d'expérience en écriture de code ActionScript, vous pouvez ajouter des composants à un document, définir leurs paramètres dans l'Inspecteur des propriétés ou dans l'Inspecteur des composants, puis gérer leurs événements via le panneau Comportements. Par exemple, sans écrire aucun code ActionScript, vous pouvez affecter un comportement Atteindre la page Web à un composant Button pour qu'une adresse URL s'ouvre dans un navigateur Web lorsque l'utilisateur clique sur ce bouton.

Si vous êtes programmeur et que vous souhaitez créer des applications plus robustes, vous pouvez créer les composants dynamiquement, utiliser ActionScript pour définir les propriétés et appeler les méthodes à l'exécution. Vous pouvez également exploiter le modèle d'événement écouteur pour gérer les événements.

Pour plus d'informations, consultez le Chapitre 2, « Utilisation des composants », à la page 37.

Configuration système requise

Aucune configuration particulière n'est requise pour les composants Flash outre Flash.

Tout fichier SWF qui utilise les composants Flash CS3 doit être affiché avec Adobe® Flash® Player 9.0.28.0 ou une version ultérieure et doit être publié pour ActionScript 3.0 (vous pouvez définir ceci dans Fichier > Paramètres de publication, sous l'onglet Flash).

Présentation de la documentation

Ce document explique comment utiliser les composants pour développer des applications Flash. Il présume que vous connaissez déjà Flash et ActionScript 3.0. La documentation spécifique à Flash et aux produits apparentés est disponible séparément.

Ce document est disponible sous forme de fichier PDF et d'aide en ligne. Pour afficher l'aide en ligne, lancez Flash et choisissez Aide > Aide de Flash > Utilisation des composants ActionScript 3.0.

Pour plus d'informations sur Flash, consultez les documents suivants :

- *Utilisation de Flash*
- *Programmation d'ActionScript 3.0*
- *Guide de référence du langage et des composants ActionScript 3.0*

Conventions typographiques

Ce manuel utilise les conventions typographiques suivantes :

- *La police en italique* indique une valeur qui devrait être remplacée (par exemple, dans le chemin d'un dossier).
- `La police de code` indentifie le code ActionScript, y compris les noms de méthode et de propriété.
- *`La police de code en italique`* désigne un élément de code à remplacer (par exemple, un paramètre ActionScript).
- **La police en gras** désigne une valeur à saisir.

Termes employés dans ce manuel

Ce manuel emploie les termes suivants :

à l'exécution Lorsque le code est exécuté dans Flash Player.

pendant la programmation Lors du travail exécuté dans l'environnement auteur de Flash.

Ressources supplémentaires

Outre le contenu de ces manuels, Adobe fournit des articles mis à jour régulièrement, des idées de conception et des exemples dans le Centre des développeurs Adobe et dans le Centre de conception Adobe.

Vous trouverez d'autres exemples de composants à l'adresse www.adobe.com/go/learn_fl_samples_fr.

Centre des développeurs Adobe

Le Centre des développeurs Adobe est votre ressource d'accès aux informations les plus récentes sur ActionScript, aux articles sur le développement d'applications réelles et aux informations sur les problèmes importants. Le Centre des développeurs est accessible à l'adresse www.adobe.com/go/flash_devcenter_fr.

Centre de conception Adobe

Accédez aux informations les plus récentes en matière de conception numérique et d'animations. Recherchez les travaux effectués par les artistes éminents, découvrez les nouvelles tendances en matière de conception et améliorez vos compétences grâce aux didacticiels, aux flux de travail fondamentaux et aux techniques avancées. Connectez-vous deux fois par mois pour accéder aux didacticiels et aux articles les plus récents, ainsi qu'aux galeries qui seront votre source d'inspiration. Le Centre de conception est accessible à l'adresse www.adobe.com/go/fl_designcenter_fr.

CHAPITRE 1

A propos des composants ActionScript 3.0

1

Les composants Adobe® Flash® CS3 Professional sont des clips vidéo dont les paramètres vous permettent de modifier l'apparence et le comportement. Un composant peut être une simple commande d'interface utilisateur, tel qu'un bouton radio ou une case à cocher, ou peut contenir du contenu, tel qu'un objet List ou DataGrid.

Les composants vous permettent de créer facilement et rapidement des applications Flash robustes à la présentation et au comportement cohérents. Plutôt que de créer vos propres boutons, listes et zones déroulantes, vous pouvez utiliser les composants Flash qui implémentent ces commandes. Il vous suffit de les faire glisser du panneau Composants dans le document de votre application. Vous pouvez également personnaliser facilement l'aspect de ces composants pour les adapter à votre application.

Même si vous pouvez effectuer toutes ces opérations sans pour autant maîtriser ActionScript, vous pouvez également utiliser ActionScript 3.0 pour modifier le comportement d'un composant ou implémenter un nouveau comportement. Chaque composant possède un jeu unique de méthodes, de propriétés et d'événements ActionScript constituant son *interface de programmation* (API). L'API vous permet de créer et de manipuler des composants lorsque l'application s'exécute.

L'API vous permet également de créer vos propres composants personalisés. Vous pouvez télécharger des composants développés par des membres de la communauté Flash sur le site Adobe Exchange à l'adresse http://www.adobe.com/go/flash_exchange_fr. Pour plus d'informations sur la création d'un composant, consultez le site www.adobe.com/go/learn_fl_creating_components_fr.

L'architecture des composants d'ActionScript 3.0 inclut des classes sur lesquelles sont basés tous les composants, des enveloppes et des styles qui vous permettent de personaliser l'aspect des composants, un modèle de gestion des événements, la gestion du focus, une interface d'accessibilité et bien plus encore.

REMARQUE	Adobe Flash CS3 inclut les composants ActionScript 2.0 et les composants ActionScript 3.0. Vous ne pouvez pas mélanger ces deux ensembles de composants. Vous devez utiliser un ensemble ou l'autre pour une application donnée. Flash CS3 intègre les composants ActionScript 2.0 ou les composants ActionScript 3.0 selon que vous ouvrez un fichier ActionScript 2.0 ou un fichier ActionScript 3.0. Lorsque vous créez un nouveau document Flash CS3, vous devez spécifier un fichier Flash (ActionScript 3.0) ou un fichier Flash (ActionScript 2.0). Lorsque vous ouvrez un document existant, Flash examine les paramètres de publication afin de déterminer quel ensemble de composants utiliser. Pour plus d'informations sur les composants ActionScript 2.0, consultez *Utilisation des composants ActionScript 2.0*.

Pour obtenir une liste complète des composants Flash ActionScript 3.0, reportez-vous à « Types de composants », à la page 18.

Ce chapitre contient les sections suivantes :

Avantages des composants

Les composants vous permettent de séparer le processus de conception de votre application du processus de codage. Ils permettent aux développeurs de créer des fonctionnalités que les concepteurs pourront exploiter dans les applications. Les développeurs peuvent encapsuler les fonctionnalités fréquemment utilisées dans des composants. Les concepteurs peuvent, quant à eux, personnaliser la taille, l'emplacement et le comportement de ces composants en modifiant leurs paramètres. Ils peuvent également changer l'apparence d'un composant en modifiant ses éléments graphiques ou enveloppes.

Les composants partagent les mêmes fonctionnalités de base : styles, enveloppes et gestion du focus. Lorsque vous ajoutez le premier composant dans une application, ces fonctionnalités de base représentent environ 20 Ko de la taille. Lorsque vous ajoutez d'autres composants, cette allocation de mémoire initiale est partagée par les composants ajoutés, réduisant ainsi la croissance de la taille de votre application.

Cette section présente les principaux avantages des composants ActionScript 3.0.

La puissance d'ActionScript 3.0 offre un langage de programmation orienté objet puissant qui constitue une étape importante de l'évolution des fonctionnalités de Flash Player. Le langage est conçu pour créer des applications Internet enrichies sur une base de code réutilisable. ActionScript 3.0 repose sur ECMAScript, le langage de script international normalisé. Il est conforme à la spécification du langage ECMAScript (ECMA-262), 3ème édition. Vous trouverez une présentation complète d'ActionScript 3.0 dans le guide *Programmation d'ActionScript 3.0*. Pour obtenir des informations de référence sur le langage, consultez le *Guide de référence du langage et des composants ActionScript 3.0*.

Les composants de l'interface utilisateur basés sur un fichier FLA permettent d'accéder facilement aux enveloppes afin de les personnaliser aisément au cours de la programmation. Ces composants fournissent également des styles, notamment des styles d'enveloppe, qui vous permettent de personnaliser l'aspect des composants et de charger des enveloppes lors de l'exécution. Pour plus d'informations, consultez le Chapitre 4, « Personnalisation des composants de l'interface utilisateur », à la page 153 et le *Guide de référence du langage et des composants ActionScript 3.0*.

Le nouveau composant FVLPlayback ajoute le composant FLVPlaybackCaptioning et intègre la prise en charge plein écran, l'aperçu en direct amélioré, des enveloppes qui vous permettent d'ajouter des paramètres de couleur et alpha, ainsi que des fonctionnalités de téléchargement et de mise en forme de fichiers FLV améliorées.

L'Inspecteur des propriétés et l'Inspecteur des composants vous permettent de modifier les paramètres des composants au cours de la programmation dans Flash. Pour plus d'informations, consultez les sections « Ajout et suppression dans un document », à la page 21 et « Définition des paramètres et des propriétés », à la page 45.

La nouvelle boîte de dialogue de collection des composants ComboBox, List et TileList vous permet de renseigner leur propriété `dataProvider` via l'interface utilisateur. Pour plus d'informations, consultez la section « Création d'un fournisseur de données », à la page 60.

Le modèle d'événements ActionScript 3.0 permet à votre application d'écouter des événements et d'appeler des gestionnaires d'événements pour y répondre. Pour plus d'informations, consultez les sections « Modèle de gestion des événements ActionScript 3.0 », à la page 26 et « Gestion d'événements », à la page 51.

Les classes Manager fournissent un moyen aisé de traiter le focus et de gérer les styles dans une application. Pour plus d'informations, consultez le *Guide de référence du langage et des composants ActionScript 3.0.*

La classe de base UIComponent fournit les méthodes, les propriétés et les événements élémentaires aux composants qui l'étendent. Tous les composants de l'interface utilisateur ActionScript 3.0 héritent de la classe UIComponent. Pour plus d'informations, consultez la classe UIComponent du *Guide de référence du langage et des composants ActionScript 3.0*.

L'utilisation d'un fichier SWC dans les composants basés sur un fichier FLA de l'interface utilisateur fournit des définitions ActionScript comme un actif faisant partie intégrante du scénario du composant afin d'accélérer la compilation.

Une hiérarchie des classes facilement extensible avec ActionScript 3.0 permet de créer des espaces de nom uniques et d'importer des classes si nécessaire et des sous-classes afin d'étendre les fonctionnalités des composants. Pour plus d'informations, consultez le *Guide de référence du langage et des composants ActionScript 3.0*.

REMARQUE

Flash CS3 prend en charge à la fois les composants basés sur un fichier FLA et SWC. Pour plus d'informations, consultez la section « Architecture des composants », à la page 37.

Types de composants

L'installation des composants Flash s'effectue lors de l'installation de Flash CS3.

Les composants ActionScript 3.0 incluent les composants de l'interface utilisateur suivants :

Composants de l'interface utilisateur		
Button	List	TextArea
CheckBox	NumericStepper	TextInput
ColorPicker	RadioButton	TileList
ComboBox	ProgressBar	UILoader
DataGrid	ScrollPane	UIScrollBar
Label	Slider	

Outre les composants de l'interface utilisateur, les composants Flash ActionScript 3.0 incluent les composants et les classes de prise en charge suivants :

- Le composant FLVPlayback (fl.video.FLVPlayback), qui est un composant basé sur un fichier SWC.

  Le composant FLVPlayback vous permet d'inclure un lecteur vidéo dans votre application Flash afin de lire de la vidéo progressive en continu sur HTTP depuis un service FVSS (Adobe® Flash® Video Streaming Service - service de diffusion en continu des vidéos de Flash) ou FMS (Macromedia® Flash® Media Server) de Adobe. Pour plus d'informations, consultez le Chapitre 5, « Utilisation du composant FLVPlayback », à la page 207.
- Les composants de l'interface utilisateur personnalisée FLVPlayback, basés sur un fichier FLA et compatibles avec les versions ActionScript 2.0 et ActionScript 3.0 du composant FLVPlayback. Pour plus d'informations, consultez le Chapitre 5, « Utilisation du composant FLVPlayback », à la page 207.
- Le composant de sous-titrage FLVPlayback, permettant de créer le sous-titrage fermé pour FLVPlayback. Chapitre 6, « Utilisation du composant FLVPlaybackCaptioning », à la page 261

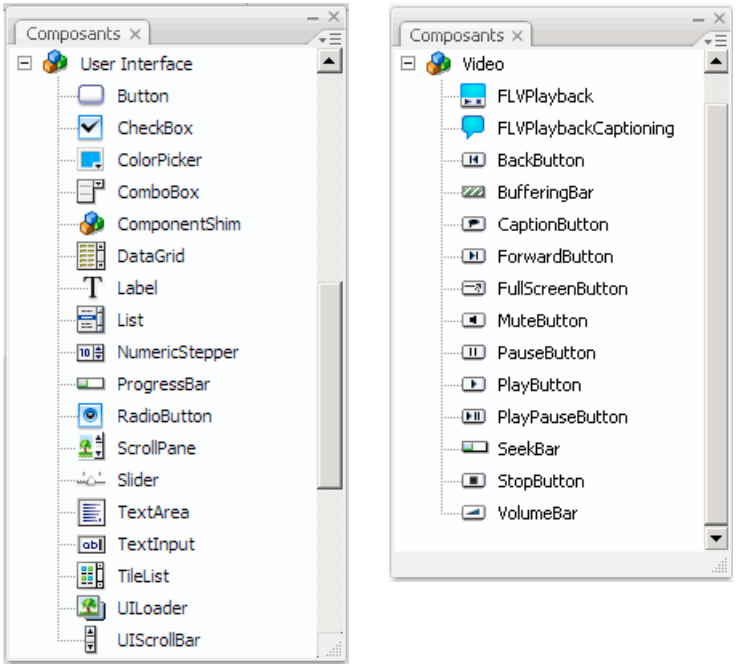
Pour obtenir une liste complète des classes de composant ActionScript 3.0 et de leurs classes de prise en charge, consultez le *Guide de référence du langage et des composants ActionScript 3.0*.

Pour afficher les composants Flash :

Vous pouvez afficher les composants Flash ActionScript 3.0 dans le panneau Composants en effectuant les étapes suivantes.

1. Démarrez Flash.
2. Créez un nouveau fichier Flash (ActionScript 3.0) ou ouvrez un document Flash existant dans lequel les paramètres de publication spécifient ActionScript 3.0.

3. Choisissez Fenêtre > Composants pour ouvrir le panneau Composants s'il n'est pas déjà visible.



Panneau Composants incluant les composants de l'interface utilisateur et vidéo

Les composants de l'interface utilisateur et vidéo sont affichés séparément pour économiser de l'espace. Le panneau Composants contient tous les composants affichés.

Vous pouvez également télécharger des composants supplémentaires depuis le site Adobe Exchange à l'adresse http://www.adobe.com/go/flash_exchange_fr. Pour installer des composants téléchargés à partir du site Exchange, téléchargez et installez Adobe® Extension Manager à l'adresse http://www.adobe.com/go/exchange_fr. Cliquez sur le lien Accueil Adobe Exchange et recherchez le lien Extension Manager.

Tous les composants doivent apparaître dans le panneau Composants de Flash. Pour installer les composants sur un ordinateur Windows® ou Macintosh®, respectez la procédure suivante.

Pour installer des composants sur un ordinateur Windows ou Macintosh :

1. Fermez Flash.
2. Placez le fichier SWC ou FLA contenant le composant dans le dossier suivant de votre disque dur :
   - Sous Windows :
     C:\Program Files\Adobe\Flash CS3*langue*\Configuration\Components
   - Sous Macintosh :
     DD Macintosh:Applications:Adobe Flash CS3:Configuration:Components
3. Démarrez Flash.
4. Choisissez Fenêtre > Composants pour visualiser le composant dans le panneau Composants s'il n'est pas déjà ouvert.

Pour plus d'informations sur les fichiers de composant, consultez la section « Utilisation des fichiers de composants », à la page 41

Ajout et suppression dans un document

Lorsque vous faites glisser un composant basé sur un fichier FLA depuis le panneau Composants vers la scène, Flash importe un clip modifiable dans la bibliothèque. Lorsque vous faites glisser un composant basé sur un fichier SWC sur la scène, Flash importe un clip compilé dans la bibliothèque. Une fois le composant importé dans la bibliothèque, vous pouvez faire glisser ses occurrences sur la scène depuis le panneau Bibliothèque ou le panneau Composants.

Ajout d'un composant pendant la programmation

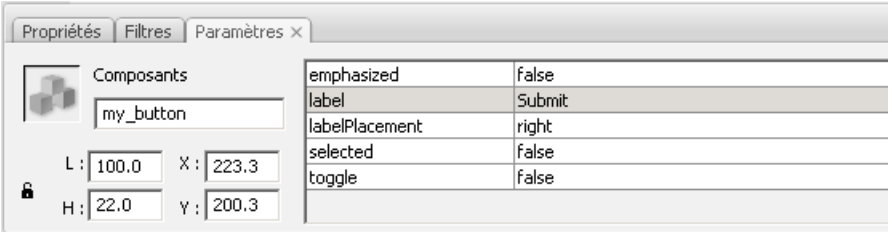
Vous pouvez ajouter un composant à un document en le faisant glisser depuis le panneau Composants. Vous pouvez définir les propriétés de chaque occurrence d'un composant dans l'onglet Paramètres de l'Inspecteur des propriétés ou de l'Inspecteur des composants.

Pour ajouter un composant à un document Flash via le panneau Composants :

1. Choisissez Fenêtre> Composants.
2. Dans le panneau Composants, double-cliquez sur le composant ou faites-le glisser sur la scène.
3. Sélectionnez le composant sur la scène.
4. Si l'Inspecteur des propriétés n'est pas visible, sélectionnez Fenêtre > Propriétés > Propriétés.

5. Dans l'Inspecteur des propriétés, nommez l'occurrence de composant.
6. Cliquez sur l'onglet Paramètres et définissez les paramètres de l'occurrence.

   L'illustration suivante présente l'Inspecteur des propriétés pour un composant Button.



Paramètres du composant dans l'Inspecteur des propriétés

   Pour plus d'informations, consultez la section « Définition des paramètres et des propriétés », à la page 45.
7. Changez la taille du composant comme vous le souhaitez en modifiant les valeurs de la largeur (L:) et de la hauteur (H:).

   Pour plus d'informations sur le dimensionnement des types spécifiques de composants, consultez le Chapitre 4, « Personnalisation des composants de l'interface utilisateur », à la page 153.
8. Choisissez Contrôle > Tester l'animation ou appuyez sur Ctrl+Entrée pour compiler le document et afficher les résultats de vos paramètres.

Vous pouvez également modifier la couleur et la mise en forme du texte d'un composant en définissant ses propriétés de style ou en personnalisant son apparence en modifiant les enveloppes du composant. Pour plus d'informations sur ces rubriques, consultez le Chapitre 4, « Personnalisation des composants de l'interface utilisateur », à la page 153.

Lorsque vous faites glisser un composant sur la scène au cours de la programmation, il suffit de l'appeler par le nom de son occurrence pour y faire référence (par exemple, `myButton`).

Ajout de composants à l'exécution avec ActionScript

Pour ajouter un composant à un document à l'exécution avec ActionScript, le composant doit d'abord se trouver dans la bibliothèque de l'application (Fenêtre > Bibliothèque) lorsque le fichier SWF est compilé. Pour ajouter un composant à la bibliothèque, faites-le glisser du panneau Composants vers le panneau Bibliothèque. Pour plus d'informations sur la bibliothèque, consultez la section « La bibliothèque », à la page 47.

Vous devez également importer le fichier de classe du composant afin que son API soit disponible pour votre application. Les fichiers de classe de composant sont installés dans des *paquets* contenant une ou plusieurs classes. Pour importer une classe de composant, utilisez l'instruction `import` et spécifiez le nom de paquet et le nom de classe. Par exemple, vous pouvez importer la classe Button à l'aide de l'instruction `import` suivante :

```
import fl.controls.Button;
```

Pour savoir dans quel paquet se trouve un composant, consultez le *Guide de référence du langage et des composants ActionScript 3.0*. Pour plus d'informations sur l'emplacement des fichiers source de composant, consultez la section « Utilisation des fichiers de composants », à la page 41.

Pour créer une occurrence du composant, vous devez appeler la méthode constructeur ActionScript du composant. Par exemple, l'instruction suivante crée une occurrence d'un composant Button appelée `aButton` :

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

La dernière étape consiste à appeler la méthode statique `addChild()` pour ajouter l'occurrence du composant sur la scène ou dans le conteneur de l'application. Par exemple, l'instruction suivante ajoute l'occurrence `aButton` :

```
addChild(aButton);
```

A ce stade, vous pouvez utiliser l'API du composant pour spécifier de façon dynamique la taille et la position du composant sur la scène, écouter des événements et définir des propriétés pour modifier son comportement. Pour plus d'informations sur l'API d'un composant particulier, consultez le *Guide de référence du langage et des composants ActionScript 3.0*.

Pour plus d'informations sur la méthode `addChild()`, consultez la section « Utilisation de la liste d'affichage », à la page 53.

Suppression d'un composant

Pour supprimer une occurrence de composant dans la scène au cours de la programmation, il vous suffit de la sélectionner et d'appuyer sur la touche Suppr. Vous supprimerez ainsi l'occurrence de la scène ; en revanche, le composant se trouve toujours dans votre application.

Pour supprimer un composant dans votre document Flash après l'avoir placé sur la scène ou dans la bibliothèque, vous devez le supprimer, ainsi que les actifs qui lui sont associés, de la bibliothèque. Supprimer le composant de la scène ne suffit pas. Si vous ne le supprimez pas de la bibliothèque, il sera inclus à votre application lors de la compilation.

Pour supprimer un composant d'un document :

1. Dans le panneau Bibliothèque, sélectionnez le symbole du composant.
2. Cliquez sur le bouton Supprimer en bas du panneau Bibliothèque ou choisissez Supprimer dans le menu du panneau Bibliothèque.

   Répétez ces étapes pour supprimer les actifs associés au composant.

Pour plus d'informations sur la suppression d'un composant de son conteneur lors de l'exécution de votre application, consultez la section « Suppression d'un composant dans la liste d'affichage », à la page 55.

Recherche de la version

Les composants Flash ActionScript 3.0 possèdent une propriété version que vous pouvez afficher si vous devez la fournir au support technique d'Adobe ou si vous devez connaître la version du composant que vous utilisez.

Pour afficher le numéro de version d'un composant de l'interface utilisateur :

1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Faites glisser le composant sur la scène et attribuez-lui un nom d'occurrence. Par exemple, faites glisser un composant ComboBox dénommé **aCb** sur la scène.
3. Appuyez sur la touche **F9** ou choisissez Fenêtre > Actions pour ouvrir le panneau Actions.
4. Cliquez sur l'image 1 du scénario principal et ajoutez le code suivant dans le panneau Actions.

```
trace(aCb.version);
```

Le numéro de version, similaire à celui de l'illustration suivante, doit apparaître dans le panneau Sortie.

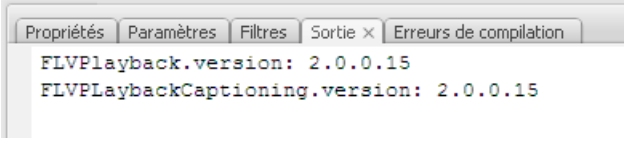
Pour les composants FLVPlayback et FLVPlaybackCaptioning, vous devez vous référer au nom de la classe plutôt qu'au nom de l'occurrence car le numéro de version est stocké dans une constante de classe.

Pour afficher le numéro de version des composants FLVPlayback et FLVPlaybackCaptioning :

1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Faites glisser les composants FLVPlayback et FLVPlaybackCaptioning dans le panneau Bibliothèque.
3. Appuyez sur la touche **F9** ou choisissez Fenêtre > Actions pour ouvrir le panneau Actions.
4. Cliquez sur l'image 1 du scénario principal et ajoutez le code suivant dans le panneau Actions.

```
import fl.video.*;
trace("FLVPlayback.VERSION: " + FLVPlayback.VERSION);
trace("FLVPLaybackCaptioning.VERSION: " +
  FLVPlaybackCaptioning.VERSION);
```

Les numéros de version, similaires à ceux de l'illustration suivante, doivent apparaître dans le panneau Sortie.



Numéros de version de FLVPlayback et FLVPlaybackCaptioning

Modèle de gestion des événements ActionScript 3.0

ActionScript 3.0 intègre un modèle de gestion des événements unique qui remplace les différents mécanismes de gestion des événements qui existaient dans les versions précédentes d'ActionScript. Le nouveau modèle d'événements repose sur la spécification d'événements de niveau 3 DOM (Document Object Model).

Pour les développeurs habitués à utiliser la méthode `addListener()` ActionScript 2.0, il peut s'avérer judicieux de souligner les différences qui existent entre le modèle d'écouteur d'événement ActionScript 2.0 et le modèle d'événement ActionScript 3.0. La liste suivante décrit les principales différences entre les deux modèles d'événement :

- Pour ajouter des écouteurs d'événements dans ActionScript 2.0, vous utilisez la méthode `addListener()` dans certains cas et la méthode `addEventListener()` dans d'autres ; en revanche, dans ActionScript 3.0, vous utilisez la méthode `addEventListener()` dans tous les cas.
- Il n'existe pas de flux d'événements dans ActionScript 2.0, ce qui signifie que la méthode `addListener()` peut être appelée uniquement sur l'objet qui diffuse l'événement ; en revanche, dans ActionScript 3.0, la méthode `addEventListener()` peut être appelée sur tous les objets qui font partie du flux d'événements.
- Dans ActionScript 2.0, les écouteurs d'événements peuvent être des fonctions, des méthodes ou des objets ; en revanche, dans ActionScript 3.0, seules les fonctions ou les méthodes peuvent faire office d'écouteurs d'événements.
- La syntaxe `on(`*`event`*`)` n'est plus prise en charge dans ActionScript 3.0 ; par conséquent, vous ne pouvez pas lier du code d'événement ActionScript à un clip. Vous pouvez uniquement utiliser la méthode `addEventListener()` pour ajouter un écouteur d'événements.

L'exemple suivant, qui écoute un événement `MouseEvent.CLICK` sur un composant Button appelé `aButton`, illustre le modèle de gestion des événements ActionScript 3.0 de base :

```
aButton.addEventListener(MouseEvent.CLICK, clickHandler);
function clickHandler(event:MouseEvent):void {
    trace("clickHandler detected an event of type: " + event.type);
    trace("the event occurred on: " + event.target.name);
}
```

Pour plus d'informations sur la gestion des événements ActionScript 3.0, consultez le guide *Programmation d'ActionScript 3.0*. Pour plus d'informations sur la gestion des événements ActionScript 3.0 des composants, consultez la section « Gestion d'événements », à la page 51.

Application simple

Cette section vous indique les étapes à suivre pour créer une application ActionScript 3.0 simple à l'aide des composants Flash et de l'outil de programmation Flash. L'exemple est fourni en tant que fichier FLA dont le code ActionScript est inclus sur le scénario et en tant que fichier de classe ActionScript externe incluant un fichier FLA qui contient uniquement les composants de la bibliothèque. En règle générale, vous avez recours aux fichiers de classe externes pour développer des applications volumineuses ; ces fichiers vous permettent en effet de partager du code entre les classes et les applications, et facilitent la mise à jour de vos applications. Pour plus d'informations sur la programmation avec ActionScript 3.0, consultez le guide *Programmation d'ActionScript 3.0*.

Conception de l'application

Notre premier exemple d'application de composant ActionScript est une variante de l'application standard « Hello World » par conséquent, sa conception est relativement simple :

- L'application sera appelée Greetings.
- Elle utilise un composant TextArea pour afficher un message de bienvenue, initialement Hello World.
- Elle utilise un composant ColorPicker qui vous permet de modifier la couleur du texte.
- Elle utilise trois composants RadioButton qui vous permettent de définir la taille du texte sur Petite, Grande ou Maximale.
- Elle utilise un composant ComboBox qui vous permet de sélectionner un message de bienvenue différent dans une liste déroulante.
- L'application utilise les composants du panneau Composants et crée également des éléments d'application via le code ActionScript.

Après avoir établi cette définition, vous pouvez commencer à créer l'application.

Création de l'application Greetings

Les instructions fournies aux étapes suivantes permettent de créer l'application Greetings à l'aide de l'outil de programmation Flash pour créer un fichier FLA, placer des composants sur la scène et ajouter du code ActionScript au scénario.

Pour créer l'application Greetings dans un fichier FLA :

1. Sélectionnez Fichier > Nouveau.
2. Dans la boîte de dialogue Nouveau document, sélectionnez Fichier Flash (ActionScript 3.0) et cliquez sur OK.

   Une nouvelle fenêtre Flash s'ouvre.
3. Choisissez Fichier > Enregistrer, nommez le fichier Flash **Greetings.fla**, puis cliquez sur le bouton Enregistrer.
4. Dans le panneau Composants Flash, sélectionnez un composant TextArea et faites-le glisser sur la scène.
5. Dans la fenêtre Propriétés, après avoir sélectionné le composant TextArea sur la scène, tapez **aTa** pour le nom d'occurrence et entrez les informations suivantes :
   - Entrez **230** pour la valeur W (largeur).
   - Entrez **44** pour la valeur H (hauteur).
   - Entrez **165** pour la valeur X (position horizontale).
   - Entrez **57** pour la valeur Y (position verticale).
   - Entrez **Hello World!** pour le paramètre texte, dans l'onglet Paramètres.
6. Faites glisser un composant ColorPicker sur la scène, placez-le à gauche du composant TextArea et donnez-lui le nom d'occurrence **txtCp.** Entrez les informations suivantes dans l'Inspecteur des propriétés :
   - Entrez **96** pour la valeur X.
   - Entrez **72** pour la valeur Y.
7. Faites glisser trois composants RadioButton sur la scène, l'un après l'autre, et donnez-leur les noms d'occurrence **smallRb**, **largerRb** et **largestRb**. Entrez les informations suivantes les concernant dans l'Inspecteur des propriétés :
   - Entrez **100** pour la valeur W et **22** pour la valeur H de chacun d'entre eux.
   - Entrez **155** pour la valeur X.
   - Entrez **120** pour la valeur Y de smallRb, **148** pour largerRb et **175** pour largestRb.
   - Entrez **fontRbGrp** pour le paramètre groupName de chacun d'entre eux.
   - Entrez leurs étiquettes sur l'onglet Paramètres de **Petite**, **Grande**, **Maximale**.

8. Faites glisser un composant ComboBox sur la scène et nommez son occurrence **msgCb**. Entrez les informations suivantes le concernant dans l'Inspecteur des propriétés :
   - Entrez **130** pour la valeur W.
   - Entrez **265** pour la valeur X.
   - Entrez **120** pour la valeur Y.
   - Dans l'onglet Paramètres, entrez **Greetings** pour le paramètre d'invite.
   - Double-cliquez sur le champ texte du paramètre dataProvider pour ouvrir la boîte de dialogue Valeurs.
   - Cliquez sur le signe plus et remplacez la valeur de l'étiquette par **Hello World!**
   - Répétez l'étape précédente pour ajouter les valeurs de l'étiquette **Have a nice day!** et **Top of the Morning!**
   - Cliquez sur OK pour fermer la boîte de dialogue Valeurs.
9. Enregistrez le fichier.
10. S'il n'est pas déjà ouvert, ouvrez le panneau Actions en appuyant sur **F9** ou en sélectionnant Actions dans le menu Fenêtre. Cliquez sur l'image 1 du scénario principal et entrez le code suivant dans le panneau Actions :

```
import flash.events.Event;
import fl.events.ComponentEvent;
import fl.events.ColorPickerEvent;
import fl.controls.RadioButtonGroup;

var rbGrp:RadioButtonGroup = RadioButtonGroup.getGroup("fontRbGrp");
rbGrp.addEventListener(MouseEvent.CLICK, rbHandler);
txtCp.addEventListener(ColorPickerEvent.CHANGE,cpHandler);
msgCb.addEventListener(Event.CHANGE, cbHandler);
```

    Les trois premières lignes importent les classes d'événement utilisées par l'application. Un événement se produit lorsqu'un utilisateur interagit avec l'un des composants. Les cinq lignes suivantes enregistrent des gestionnaires d'événements pour les événements que l'application souhaite écouter. Un événement `click` se produit pour un composant RadioButton lorsqu'un utilisateur clique dessus. Un événement `change` se produit lorsqu'un utilisateur sélectionne une autre couleur dans le composant ColorPicker. Un événement `change` se produit sur le composant ComboBox lorsqu'un utilisateur choisit un autre message de bienvenue dans la liste déroulante.

    La quatrième ligne importe la classe RadioButtonGroup de manière à ce que l'application puisse affecter un écouteur d'événements au groupe de composants RadioButtons, plutôt que d'affecter l'écouteur à chaque bouton individuellement.

11. Ajoutez la ligne de code suivante au panneau Actions pour créer l'objet TextFormat `tf`, utilisé par l'application pour modifier les propriétés de style `size` et `color` du texte dans le composant TextArea.

```
var tf:TextFormat = new TextFormat();
```

12. Ajoutez le code suivant pout créer la fonction de gestion des événements `rbHandler`. Cette fonction gère un événement `click` lorsqu'un utilisateur clique sur l'un des composants RadioButton.

```
function rbHandler(event:MouseEvent):void {
  switch(event.target.selection.name) {
    case "smallRb":
      tf.size = 14;
      break;
    case "largerRb":
      tf.size = 18;
      break;
    case "largestRb":
      tf.size = 24;
      break;
    }
    aTa.setStyle("textFormat", tf);
}
```

Cette fonction utilise une instruction `switch` pour examiner la propriété `target` de l'objet `événement` afin de déterminer quel composant RadioButton a déclenché l'événement. La propriété `currentTarget` contient le nom de l'objet qui a déclenché l'événement. Selon le composant RadioButton sur lequel l'utilisateur a cliqué, l'application modifie la taille du texte dans le composant TextArea pour la définir sur 14, 18 ou 24 points.

13. Ajoutez le code suivant pour implémenter la fonction `cpHandler()`, qui gère une modification apportée à la valeur dans le composant ColorPicker :

```
function cpHandler(event:ColorPickerEvent):void {
  tf.color = event.target.selectedColor;
  aTa.setStyle("textFormat", tf);
}
```

Cette fonction définit la propriété `color` de l'objet TextFormat `tf` sur la couleur sélectionnée dans le composant ColorPicker, puis appelle la méthode `setStyle()` pour l'appliquer au texte dans l'occurrence TextArea `aTa`.

14. Ajoutez le code suivant pour implémenter la fonction `cbHandler()`, qui gère une modification apportée à la sélection dans le composant ComboBox :

```
function cbHandler(event:Event):void {
  aTa.text = event.target.selectedItem.label;
}
```

Cette fonction remplace simplement le texte du composant TextArea par le texte sélectionné dans le composant ComboBox, `event.target.selectedItem.label`.

15. Choisissez Contrôle > Tester l'animation ou appuyez sur Ctrl+Entrée pour compiler le code et tester l'application Greetings.

La section suivante explique comment créer la même application avec une classe ActionScript externe et un fichier FLA incluant uniquement les composants requis dans la bibliothèque.

Pour créer l'application Greetings2 à l'aide d'un fichier de classe externe :

1. Sélectionnez Fichier > Nouveau.
2. Dans la boîte de dialogue Nouveau document, sélectionnez Fichier Flash (ActionScript 3.0) et cliquez sur OK.

   Une nouvelle fenêtre Flash s'ouvre.
3. Choisissez Fichier > Enregistrer, nommez le fichier Flash **Greetings2.fla**, puis cliquez sur le bouton Enregistrer.
4. Faites glisser les composants suivants du panneau Composants vers la bibliothèque :
   - ColorPicker
   - ComboBox
   - RadioButton
   - TextArea

   Le fichier SWF compilé utilisera chacun de ces actifs ; vous devez donc les ajouter à la bibliothèque. Faites glisser les composants en bas du panneau Bibliothèque. Lorsque vous ajoutez ces composants à la bibliothèque, d'autres actifs (tels que List, TextInput et UIScrollBox) sont ajoutés automatiquement.
5. Dans la fenêtre Propriétés, pour la classe Document, tapez **Greetings2**.

   Si Flash affiche un message d'avertissement indiquant que la « définition de la classe de document est introuvable », ignorez-le. Vous définirez la classe Greetings2 lors des étapes suivantes. Cette classe définit les principales fonctionnalités de l'application.
6. Enregistrez le fichier Greetings2.fla.
7. Sélectionnez Fichier > Nouveau.

8. Dans la boîte de dialogue Nouveau document, sélectionnez Fichier ActionScript et cliquez sur OK.

   Une nouvelle fenêtre de script s'ouvre.

9. Ajoutez le code suivant dans la fenêtre de script :

```
package {
  import flash.display.Sprite;
  import flash.events.Event;
  import flash.events.MouseEvent;
  import flash.text.TextFormat;
  import fl.events.ComponentEvent;
  import fl.events.ColorPickerEvent;
  import fl.controls.ColorPicker;
  import fl.controls.ComboBox;
  import fl.controls.RadioButtonGroup;
  import fl.controls.RadioButton;
  import fl.controls.TextArea;
  public class Greetings2 extends Sprite {
    private var aTa:TextArea;
    private var msgCb:ComboBox;
    private var smallRb:RadioButton;
    private var largerRb:RadioButton;
    private var largestRb:RadioButton;
    private var rbGrp:RadioButtonGroup;
    private var txtCp:ColorPicker;
    private var tf:TextFormat = new TextFormat();
    public function Greetings2() {
```

   Le script définit une classe ActionScript 3.0, intitulée Greetings2. Le script effectue les opérations suivantes :

   - Il importe les classes que nous utiliserons dans le fichier. La procédure habituelle consiste à ajouter ces instructions d'importation au fur et à mesure que vous faites référence à différentes classes dans le code ; cependant, pour plus de concision, cet exemple les importe toutes au cours de cette étape unique.
   - Il déclare les variables qui représentent les différents types d'objets de composant que nous ajouterons au code. Une autre variable crée l'objet TextFormat `tf`.
   - Il définit une fonction constructeur, `Greetings2()`, pour la classe. Nous ajouterons des lignes à cette fonction et d'autres méthodes à la classe lors des étapes suivantes.

10. Choisissez Fichier > Enregistrer, nommez le fichier **Greetings2.as**, puis cliquez sur le bouton Enregistrer.

11. Ajoutez les lignes de code suivantes à la fonction `Greeting2()` :

```
    createUI();
    setUpHandlers();
}
```

La fonction doit désormais avoir l'aspect suivant :

```
public function Greetings2() {
  createUI();
  setUpHandlers();
}
```

12. Ajoutez les lignes de code suivantes après l'accolade fermante de la méthode `Greeting2()` :

```
private function createUI() {
  bldTxtArea();
  bldColorPicker();
  bldComboBox();
  bldRadioButtons();
}
private function bldTxtArea() {
  aTa = new TextArea();
  aTa.setSize(230, 44);
  aTa.text = "Hello World!";
  aTa.move(165, 57);
  addChild(aTa);
}
private function bldColorPicker() {
  txtCp = new ColorPicker();
  txtCp.move(96, 72);
  addChild(txtCp);
}
private function bldComboBox() {
  msgCb = new ComboBox();
  msgCb.width = 130;
  msgCb.move(265, 120);
  msgCb.prompt = "Greetings";
  msgCb.addItem({data:"Hello.", label:"English"});
  msgCb.addItem({data:"Bonjour.", label:"Français"});
  msgCb.addItem({data:"¡Hola!", label:"Español"});
  addChild(msgCb);
}
private function bldRadioButtons() {
  rbGrp = new RadioButtonGroup("fontRbGrp");
  smallRb = new RadioButton();
  smallRb.setSize(100, 22);
  smallRb.move(155, 120);
  smallRb.group = rbGrp; //"fontRbGrp";
  smallRb.label = "Small";
  smallRb.name = "smallRb";
  addChild(smallRb);
  largerRb = new RadioButton();
  largerRb.setSize(100, 22);
  largerRb.move(155, 148);
  largerRb.group = rbGrp;
  largerRb.label = "Larger";
  largerRb.name = "largerRb";
```

```
    addChild(largerRb);
    largestRb = new RadioButton();
    largestRb.setSize(100, 22);
    largestRb.move(155, 175);
    largestRb.group = rbGrp;
    largestRb.label = "Largest";
    largestRb.name = "largestRb";
    addChild(largestRb);
  }
```

Ces lignes :

- instancient les composants utilisés dans l'application ;
- définissent la taille, la position et les propriétés de chaque composant ;
- ajoutent chaque composant sur la scène via la méthode `addChild()`.

13. Après l'accolade fermante de la méthode `bldRadioButtons()`, ajoutez le code suivant pour la méthode `setUpHandlers()` :

```
private function setUpHandlers():void {
  rbGrp.addEventListener(MouseEvent.CLICK, rbHandler);
  txtCp.addEventListener(ColorPickerEvent.CHANGE,cpHandler);
  msgCb.addEventListener(Event.CHANGE, cbHandler);
}
private function rbHandler(event:MouseEvent):void {
  switch(event.target.selection.name) {
    case "smallRb":
      tf.size = 14;
      break;
    case "largerRb":
      tf.size = 18;
      break;
    case "largestRb":
      tf.size = 24;
      break;
  }
  aTa.setStyle("textFormat", tf);
}
private function cpHandler(event:ColorPickerEvent):void {
  tf.color = event.target.selectedColor;
  aTa.setStyle("textFormat", tf);
}
private function cbHandler(event:Event):void {
  aTa.text = event.target.selectedItem.data;
}
}
}
```

Ces fonctions définissent les écouteurs d'événements des composants.

14. Sélectionnez Fichier > Enregistrer pour enregistrer le fichier.
15. Choisissez Contrôle > Tester l'animation ou appuyez sur Ctrl+Entrée pour compiler le code et tester l'application Greetings2.

Exécution des exemples suivants

Après avoir développé et exécuté l'application Greetings, vous devriez posséder les connaissances de base requises pour pouvoir exécuter les autres exemples de code présentés dans ce manuel. Le code ActionScript 3.0 propre à chaque exemple sera mis en surbrillance et présenté de manière détaillée. De plus, vous pourrez couper et coller chaque exemple présenté dans ce manuel dans un fichier FLA, le compiler et l'exécuter.

CHAPITRE 2

Utilisation des composants

2

Dans ce chapitre, vous allez apprendre à utiliser les composants dans un document.
Ce chapitre contient les rubriques suivantes :

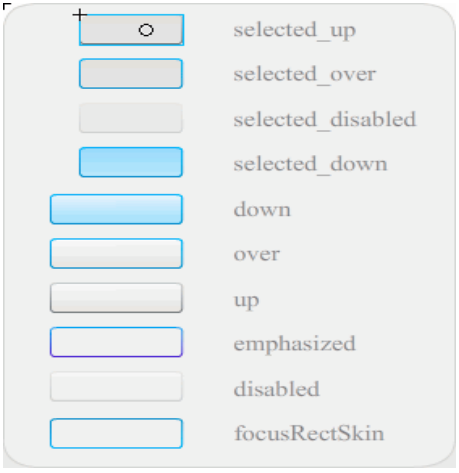
Architecture des composants

Les composants ActionScript 3.0 sont pris en charge par Abobe Flash Player 9.0.28.0 et les versions ultérieures. Ces composants ne sont pas compatibles avec les composants antérieurs à Flash CS3. Pour plus d'informations sur l'utilisation des composants ActionScript 2.0, consultez *Utilisation des composants ActionScript 2.0* et le *Guide de référence du langage des composants ActionScript 2.0*.

Les composants de l'interface utilisateur ActionScript 3.0 sont implémentés en tant que composants basés sur un fichier FLA mais Flash CS3 prend en charge à la fois les composants basés sur un fichier SWC et FLA. Par exemple, les composants FLVPlayback et FLVPlaybackCaptioning sont basés sur un fichier SWC. Vous pouvez placer l'un des types de composants dans le dossier Composants de manière à ce qu'il apparaisse dans le panneau Composants. Ces deux types de composants étant créés différemment, ils sont décrits séparement ici.

Composants ActionScript 3.0 basés sur un fichier FLA

Les composants de l'interface utilisateur ActionScript 3.0 sont des fichiers FLA (.fla) intégrant des enveloppes auxquelles vous pouvez accéder afin de les modifier. Pour ce faire, il vous suffit de double-cliquer sur le composant qui se trouve sur la scène. Les enveloppes et les autres éléments du composant sont placés sur l'image 2 du scénario. Lorsque vous double-cliquez sur le composant, Flash passe directement à l'image 2 et ouvre une palette des enveloppes du composant. L'illustration suivante présente la palette des enveloppes qui s'affichent pour le composant Button.



Enveloppes du composant Button

Pour plus d'informations sur les enveloppes des composants et la personnalisation des composants, consultez le Chapitre 4, « Personnalisation des composants de l'interface utilisateur », à la page 153 et la section « Personnalisation du composant FLVPlayback », à la page 232.

Pour accélérer la compilation de vos applications et éviter les conflits avec vos paramètres ActionScript 3.0, les composants de l'interface utilisateur basés sur un fichier FLA Flash CS3 contiennent également un fichier SWC incluant le code ActionScript déjà compilé du composant. Le fichier SWC ComponentShim est placé sur la scène, sur l'image 2, dans chaque composant de l'interface utilisateur de manière à rendre les définitions précompilées disponibles. Pour être accessible à ActionScript, un composant doit se trouver sur la scène ou dans la bibliothèque et l'option Exporter dans la première image doit être cochée dans la boîte de dialogue Propriétés de liaison correspondante. Pour créer un composant via ActionScript, vous devez également importer la classe avec une instruction `import` pour y accéder. Pour plus d'informations sur l'instruction `import`, consultez le *Guide de référence du langage et des composants ActionScript 3.0*.

Composants basés sur un fichier SWC

Les composants basés sur un fichier SWC possèdent un fichier FLA et un fichier de classe ActionScript également ; en revanche, ils ont été compilés et exportés en tant que fichier SWC. Un fichier SWC est un paquet de symboles Flash précompilés et de code ActionScript qui vous évite de devoir recompiler les symboles et le code qui ne changeront pas.

Les composants FLVPlayback et FLVPlaybackCaptioning sont basés sur un fichier SWC. Ils possèdent des enveloppes externes, et non des enveloppes intégrées. Le composant FLVPlayback possède une enveloppe par défaut que vous pouvez modifier. Pour ce faire, il vous suffit de sélectionner une autre enveloppe dans un ensemble d'enveloppes prédéfinies, de personnaliser les commandes de l'interface utilisateur dans le panneau Composants (BackButton, BufferingBar, etc.) ou de créer une enveloppe personnalisée. Pour plus d'informations, consultez la section « Personnalisation du composant FLVPlayback », à la page 232.

Dans Flash, vous pouvez convertir un clip en clip compilé comme suit.

Pour compiler un clip :

- Dans le panneau Bibliothèque, cliquez sur le clip du bouton droit (Windows) ou avec la touche Contrôle enfoncée (Macintosh), puis choisissez Convertir en clip compilé.

Le clip compilé se comporte exactement comme le clip à partir duquel il a été compilé, mais les clips compilés sont affichés et publiés beaucoup plus rapidement que les clips ordinaires. Les clips compilés ne peuvent pas être modifiés, mais leurs propriétés peuvent apparaître dans l'Inspecteur des propriétés et l'Inspecteur des composants.

Les composants SWC contiennent un clip compilé, les définitions ActionScript précompilées du composant et d'autres fichiers qui décrivent le composant. Si vous créez votre propre composant, vous pouvez l'exporter en tant que fichier SWC pour le distribuer.

Pour exporter un fichier SWC :

- Choisissez le clip dans le panneau Bibliothèque et cliquez du bouton droit (Windows) ou maintenez la touche Contrôle enfoncée (Macintosh), puis sélectionnez Exporter le fichier SWC.

REMARQUE	Le format d'un fichier SWC Flash CS3 est compatible avec le format Flex SWC, ce qui permet d'échanger les fichiers SWC entre les deux produits, mais pas nécessairement sans modification.

Pour plus d'informations sur la création de composants basés sur un fichier SWC, consultez le site www.adobe.com/go/learn_fl_creating_components_fr.

API des composants ActionScript 3.0

Chaque composant ActionScript 3.0 est basé sur une classe ActionScript 3.0 située dans un dossier de paquet dont le nom est au format fl.*packagename.classname*. Par exemple, le composant Button, dont le nom de paquet est `fl.controls.Button`, est une occurrence de la classe Button. Vous devez faire référence au nom de paquet lorsque vous importez une classe de composant dans votre application. Vous pouvez importer la classe Button à l'aide de l'instruction suivante :

```
import fl.controls.Button;
```

Pour plus d'informations sur l'emplacement des fichiers de classe de composant, consultez la section « Utilisation des fichiers de composants », à la page 41.

La classe d'un composant définit les méthodes, propriétés, événements et styles qui vous permettent d'interagir avec votre application. Les composants de l'interface utilisateur ActionScript 3.0 sont des sous-classes des classes Sprite et UIComponent et héritent de leurs propriétés, méthodes et événements. La classe Sprite est l'élément fondamental de la liste d'affichage : elle est similaire à un clip mais ne possède pas de scénario. La classe UIComponent constitue la classe de base de tous les composants visuels, interactifs ou non. Le chemin de l'héritage de chaque composant, ainsi que ses propriétés, méthodes, événements et styles sont décrits dans le *Guide de référence du langage et des composants ActionScript 3.0*.

Tous les composants ActionScript 3.0 utilisent le modèle de gestion des événements ActionScript 3.0. Pour plus d'informations sur la gestion des événements, consultez la section « Gestion d'événements », à la page 51 et le guide *Programmation d'ActionScript 3.0*.

Utilisation des fichiers de composants

Cette section explique à quel emplacement sont stockés les fichiers de composants, où se trouvent les fichiers source ActionScript et comment ajouter et supprimer des composants dans le panneau Composants.

Emplacement de stockage des fichiers de composants

Les composants Flash sont stockés dans le dossier Configuration au niveau de l'application.

REMARQUE

Pour plus d'informations sur ces dossiers, consultez la section « Dossiers de configuration installés avec Flash » dans le guide *Utilisation de Flash*.

Les composants sont installés dans les emplacements suivants :

- Windows 2000 ou Windows XP :

  C:\Program Files\Adobe\Adobe Flash CS3*langue*\Configuration\Components
- Mac OS X :

  DD Macintosh:Applications:Adobe Flash CS3:Configuration:Components

Dans le dossier Composants, les composants de l'interface utilisateur se trouvent dans le fichier User Interface.fla et les composants FLVPlayback (FLVPlaybackAS3.swc) et FLVPlaybackCaptioning dans le dossier Video.

Vous pouvez également stocker les composants dans les emplacements utilisateur suivants :

- Windows 2000 ou Windows XP :

  C:\Documents and Settings*nom d'utilisateur*\Local Settings\Application Data\Adobe\Adobe Flash CS3\fr\Configuration\Components
- Mac OS X :

  Disque dur Macintosh:Utilisateurs:<utilisateur>:Library:Application Support:Adobe Flash CS3:Configuration:Components

Emplacement de stockage des fichiers source de composant

Les fichiers de classe ActionScript (.as) (ou *fichiers source*) des composants sont installés dans les dossiers d'application suivants pour Windows 2000 ou Windows XP :

- Composants d'interface utilisateur

  C:\Program Files\Adobe\Adobe Flash CS3\fr\Configuration\Component Source\ActionScript 3.0\User Interface\fl
- FLVPlayback

  C:\Program Files\Adobe\Adobe Flash CS3\fr\Configuration\Component Source\ActionScript 3.0\FLVPlayback\fl\video
- FLVPlaybackCaptioning

  C:\Program Files\Adobe\Adobe Flash CS3\fr\Configuration\Component Source\ActionScript 3.0\FLVPlaybackCaptioning\fl\video

Pour Mac OS X, les fichiers source de composant sont situés aux emplacements suivants :

- Composants d'interface utilisateur

  Disque dur Macintosh:Applications:Adobe Flash CS3:Configuration:Component Source:ActionScript 3.0:User Interface:fl
- FLVPlayback

  Disque dur Macintosh:Applications:Adobe Flash CS3:Configuration:Component Source:ActionScript 3.0:FLVPlayback:fl:video
- FLVPlaybackCaptioning

  Disque dur Macintosh:Applications:Adobe Flash CS3:Configuration:Component Source:ActionScript 3.0:FLVPlaybackCaptioning:fl:video

Fichiers source de composant et variable Classpath

Etant donné que le code des composants ActionScript 3.0 est compilé au sein de ces derniers, vous ne devez pas spécifier l'emplacement des fichiers de classe ActionScript dans votre variable Classpath. Si vous incluez leur emplacement à la variable Classpath, le temps requis pour compiler vos applications augmentera. Toutefois, si Flash découvre des fichiers de classe de composant dans votre paramètre Classpath, le fichier de classe est toujours prioritaire par rapport au code compilé dans le composant.

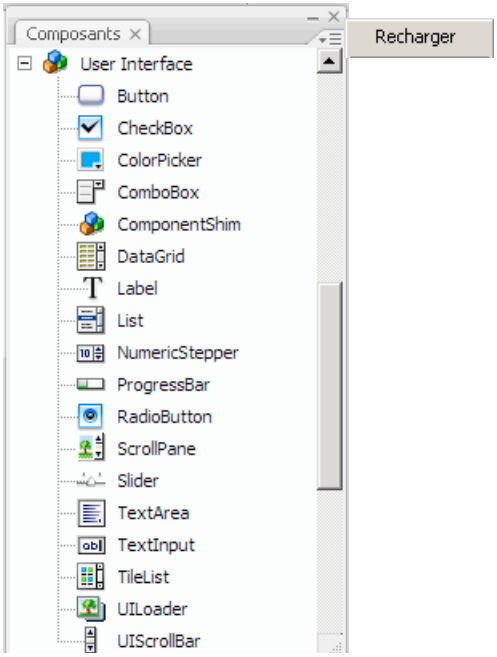
Lors du débogage d'une application avec les composants, vous voudrez peut-être ajouter l'emplacement des fichiers source de composant à votre paramètre Classpath. Pour plus d'informations, consultez la section « Débogage des applications de composants », à la page 44.

Modification des fichiers de composant

Si vous mettez à jour, ajoutez ou supprimez des composants basés sur un fichier SWC ou ajoutez de nouveaux composants basés sur un fichier FLA à Flash, vous devez les recharger dans le panneau Composants afin de les rendre disponibles. Pour recharger les composants, il vous suffit de redémarrer Flash ou de sélectionner Recharger dans le menu du panneau Composants. Flash collectera ainsi les composants que vous avez ajoutés dans le dossier Composants.

Pour recharger des composants dans le panneau Composants lors de l'exécution de Flash :

- Dans le menu du panneau Composants, choisissez Recharger.



Elément de menu Recharger dans le menu du panneau Composants

Pour supprimer un composant dans le panneau Composants :

- Supprimez le fichier FLA, SWC ou MXP du dossier Composants et redémarrez Flash ou sélectionnez Recharger dans le menu du panneau Composants. Un fichier MXP est un fichier de composant téléchargé depuis le site Adobe Exchange.

Vous pouvez supprimer et remplacer les composants basés sur un fichier SWC lors de l'exécution de Flash : le rechargement reflètera alors les modifications ; en revanche, si vous modifiez ou supprimez des composants basés sur un fichier FLA, les modifications ne sont pas implémentées tant que vous n'avez pas terminé et redémarré Flash. Vous pouvez, cependant, ajouter des composants basés sur un fichier FLA et les charger à l'aide de la commande Recharger.

CONSEIL	Adobe vous recommande d'abord de faire une copie du fichier de composant Flash (.fla ou .as) que vous allez modifier. Vous pouvez ensuite le restaurer, si nécessaire.

Débogage des applications de composants

Les composants ActionScript 3.0 contiennent tous leur code source afin de réduire le temps de compilation de votre application. Toutefois, le débogueur Flash ne peut pas inspecter le code des clips compilés. Par conséquent, si vous souhaitez déboguer votre application dans le code source du composant, vous devez ajouter les fichiers source du composant à votre paramètre Classpath.

L'emplacement des dossiers de paquet de composants dépend de l'emplacement des fichiers source du type de composant. Pour faire référence à tous les fichiers source ActionScript 3.0 de tous les composants de l'interface utilisateur, ajoutez l'emplacement suivant à votre paramètre Classpath pour les paquets de l'interface utilisateur :

```
$(AppConfig)/Component Source/ActionScript 3.0/User Interface
```

REMARQUE	Vous écraserez ainsi le code compilé dans tous les composants de l'interface utilisateur et augmenterez le temps de compilation de votre application. Si, pour une raison quelconque, vous avez modifié le fichier source d'un composant, celui-ci peut de ce fait se comporter différemment.

Pour définir le paramètre Classpath, sélectionnez Préférences dans le menu Edition, puis ActionScript dans la liste Catégorie et cliquez sur le bouton Paramètres ActionScript 3.0. Pour ajouter une nouvelle entrée, cliquez sur le signe plus au-dessus de la fenêtre qui affiche les paramètres actuels.

La variable `$(AppConfig)` renvoie au dossier Configuration Flash CS3 à l'emplacement où vous avez installé Flash CS3. En général, le chemin a l'aspect suivant :

pour Windows 2000 ou Windows XP

```
C:\Program Files\Adobe\Adobe Flash CS3\language\Configuration\
```

pour Mac OS X

```
Macintosh HD:Applications:Adobe Flash CS3:Configuration:
```

REMARQUE

Si vous devez modifier un fichier source de composant, Adobe vous recommande fortement de copier le fichier source d'origine dans un emplacement différent et d'ajouter ce dernier à votre paramètre Classpath.

Pour plus d'informations sur l'emplacement des fichiers source de composant, consultez la section « Emplacement de stockage des fichiers source de composant », à la page 42.

Définition des paramètres et des propriétés

Chaque composant dispose de paramètres que vous pouvez définir pour modifier son apparence et son comportement. Un paramètre est une propriété de la classe du composant et apparaît dans l'Inspecteur des propriétés et dans l'Inspecteur des composants. Les propriétés les plus utilisées apparaissent sous forme de paramètres de création ; les autres doivent être définies à l'aide d'ActionScript. Tous les paramètres définissables pendant la programmation peuvent également être modifiés avec ActionScript. La définition d'un paramètre par ActionScript remplace toutes les valeurs définies lors de la programmation.

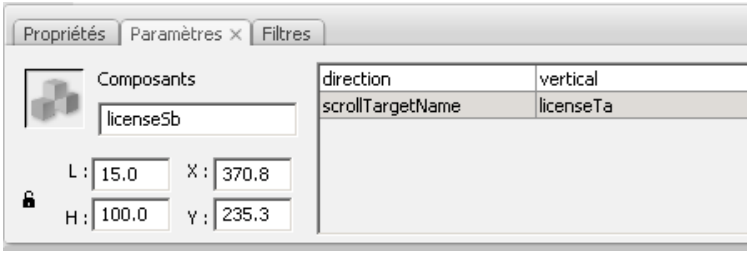
La plupart des composants de l'interface utilisateur ActionScript 3.0 héritent des propriétés et des méthodes de la classe UIComponent, ainsi que d'une classe de base. Par exemple, les classes Button et CheckBox héritent des propriétés de la classe UIComponent et de la classe BaseButton. Vous pouvez accéder aux propriétés héritées d'un composant, ainsi qu'à ses propres propriétés de classe. Par exemple, le composant ProgressBar hérite de la propriété `ProgressBar.enabled` de la classe UIComponent, mais possède également sa propre propriété `ProgressBar.percentComplete`. Vous pouvez accéder à ces deux propriétés pour interagir avec une occurrence du composant ProgressBar. Pour plus d'informations sur les propriétés d'un composant, consultez l'entrée de sa classe dans le *Guide de référence du langage et des composants ActionScript 3.0*.

Vous pouvez définir les paramètres d'une occurrence de composant via l'Inspecteur des propriétés ou l'Inspecteur des composants.

Pour entrer le nom d'une occurrence de composant dans l'Inspecteur des propriétés :

1. Choisissez Fenêtre> Propriétés> Propriétés.
2. Sélectionnez une occurrence de composant sur la scène.
3. Entrez un nom d'occurrence de composant dans la zone intitulée <Nom d'occurrence>, située sous la liste déroulante intitulée Clip. Ou cliquez sur l'onglet Paramètres et entrez le nom dans la zone située sous le mot *Composant*. Entrez les valeurs des paramètres que vous voulez définir.

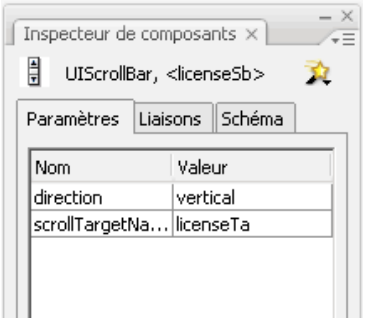
   Pour faciliter la compréhension de votre code ActionScript, il est conseillé de préciser le type du composant en ajoutant un suffixe au nom de l'occurrence. Dans cet exemple, le nom d'occurrence est **licenseSb** car le composant est une barre de défilement qui fait défiler un contrat de licence dans la zone de texte **licenseTa**.



Champ Nom d'occurrence du composant

Pour entrer les paramètres d'une occurrence de composant dans l'Inspecteur des composants :

1. Choisissez Fenêtre> Inspecteur des composants.
2. Sélectionnez une occurrence de composant sur la scène.
3. Cliquez sur l'onglet Paramètres et entrez les valeurs des paramètres répertoriés.



Paramètres de composant dans l'Inspecteur des composants

Définition de propriétés de composant dans ActionScript

Dans ActionScript, un opérateur point (.) (syntaxe à point) est utilisé pour accéder aux propriétés ou méthodes associées à un objet ou à une occurrence de la scène. Une expression en syntaxe à point commence par le nom de l'occurrence, suivi d'un point, et se termine par l'élément que vous souhaitez spécifier. Par exemple, le code ActionScript suivant définit la propriété `width` de l'occurrence CheckBox `aCh` de manière à ce qu'elle mesure 50 pixels de large :

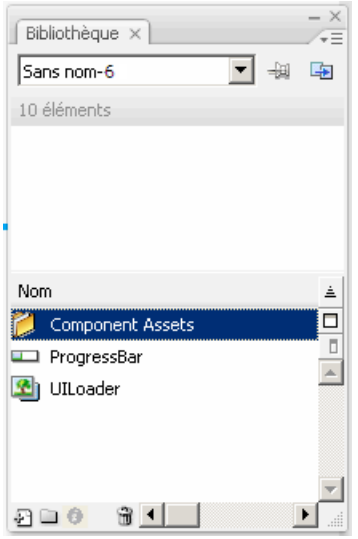
```
aCh.width = 50;
```

L'instruction `if` suivante vérifie si l'utilisateur a coché la case :

```
if (aCh.selected == true) {
  displayImg(redCar);
}
```

La bibliothèque

Lorsque vous ajoutez un composant à un document pour la première fois, Flash l'importe en tant que clip dans le panneau Bibliothèque. Vous pouvez également faire glisser un composant du panneau Composants directement vers le panneau bibliothèque, puis ajouter une occurrence de celui-ci sur la scène. Vous devez à chaque fois ajouter un composant à la bibliothèque pour pouvoir accéder aux éléments de sa classe.

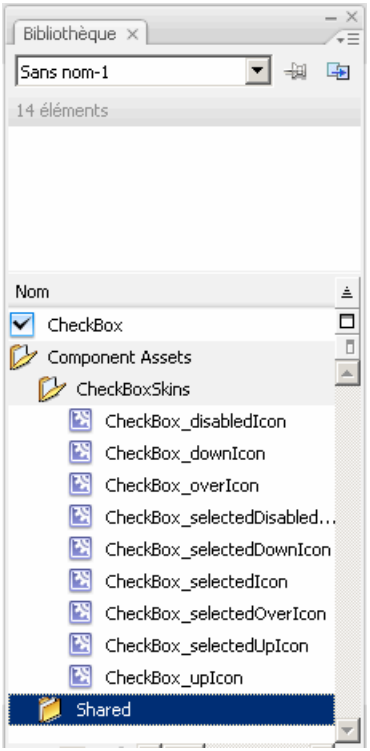


Composant ProgressBar dans le panneau Bibliothèque

Si vous ajoutez un composant à la bibliothèque et créez une occurrence de celui-ci à l'aide d'ActionScript, vous devez d'abord importer sa classe avec l'instruction `import`. Dans l'instruction `import`, vous devez spécifier le nom de paquet et le nom de classe du composant. Par exemple, l'instruction suivante importe la classe Button :

```
import fl.controls.Button;
```

Lorsque vous placez un composant dans la bibliothèque, Flash importe également un dossier de ses actifs contenant les enveloppes de ses différents états. Les *enveloppes* d'un composant comprennent l'ensemble des symboles constituant son affichage graphique dans l'application. Une enveloppe unique est la représentation graphique, ou clip, qui indique un état particulier du composant. Par exemple, dans le dossier Component Assets du composant CheckBox, l'enveloppe CheckBox_disabledIcon fournit la représentation graphique du composant à l'état désactivé. L'enveloppe CheckBox_selectedDownIcon fournit l'image graphique du composant CheckBox qui s'affiche lorsque vous cliquez sur celui-ci tout en maintenant enfoncé le bouton de la souris.



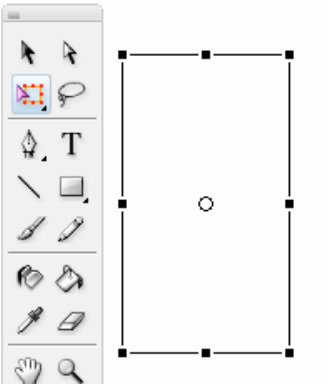
Actifs du composant dans le panneau Bibliothèque

Le contenu du dossier Component Assets vous permet de modifier les enveloppes du composant si vous le souhaitez. Pour plus d'informations, consultez le Chapitre 4, « Personnalisation des composants de l'interface utilisateur », à la page 153.

Lorsqu'un composant se trouve dans la bibliothèque, vous pouvez ajouter plusieurs occurrences de celui-ci à votre document en faisant glisser son icône du panneau Composants ou du panneau Bibliothèque vers la scène.

Dimensionnement des composants

Pour redimensionner les occurrences de vos composants, employez l'outil Transformation libre ou la méthode `setSize()`.

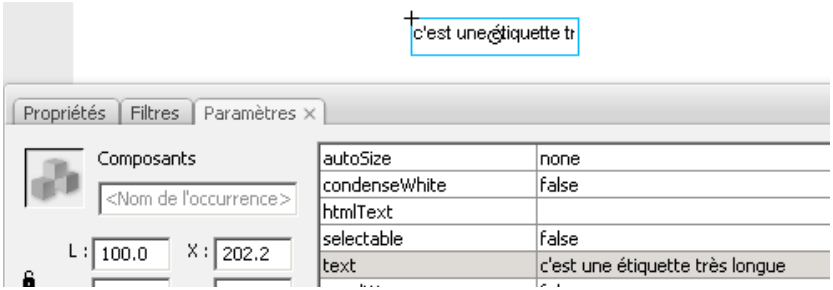


Redimensionnement du composant List sur la scène à l'aide de l'outil Transformation libre

Vous pouvez appeler la méthode `setSize()` à partir de n'importe quelle occurrence de composant pour le redimensionner (voir `UIComponent.setSize()`). Le code suivant redimensionne une occurrence du composant List avec une largeur de 200 pixels et une hauteur de 300 pixels :

```
aList.setSize(200, 300);
```

La taille d'un composant n'est pas automatiquement ajustée à son étiquette. Si une occurrence de composant ajoutée à un document n'est pas assez grande pour contenir son étiquette, le texte de celle-ci sera tronqué. Vous devez donc adapter le composant à la taille de son étiquette.



Texte tronqué dans un composant Label

Pour plus d'informations sur le redimensionnement des composants, consultez les entrées correspondantes dans le *Guide de référence du langage et des composants ActionScript 3.0*.

Aperçu en direct

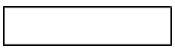
La fonction Aperçu en direct, activée par défaut, permet de visualiser les composants sur la scène tels qu'ils apparaîtront dans le contenu Flash publié, avec leur taille approximative.

Pour activer ou désactiver l'aperçu en direct :

- Choisissez Contrôle > Activer l'aperçu en direct. Une coche en regard de l'option indique qu'elle est activée.

L'aperçu en direct reflète les différents paramètres des composants utilisés. Pour plus d'informations sur les paramètres de composant visibles dans l'aperçu en direct, consultez les entrées de chaque composant dans le *Guide de référence du langage et des composants ActionScript 3.0*.

Composant Button avec Aperçu en direct activé



Composant Button avec Aperçu en direct désactivé

Les composants de l'aperçu en direct ne sont pas opérationnels. Pour tester leur fonctionnement, vous devez utiliser la commande Contrôle> Tester l'animation.

Gestion d'événements

Chaque composant diffuse des événements lorsqu'un utilisateur interagit avec lui. Par exemple, lorsqu'un utilisateur clique sur un composant Button, celui distribue un événement `MouseEvent.CLICK` ; lorsqu'il sélectionne un élément dans un composant List, celui-ci distribue un événement Event.`CHANGE`. Un événement peut également se produire lorsqu'une opération importante intervient sur un composant, par exemple à la fin du chargement du contenu pour une occurrence UILoader, ce qui génère un événement `Event.COMPLETE`. Pour gérer un événement, vous rédigez du code ActionScript qui s'exécute lorsque l'événement se produit.

Les événements d'un composant incluent les événements de toutes les classes héritées par le composant. Cela signifie que tous les composants de l'interface utilisateur ActionScript 3.0 héritent des événements de la classe UIComponent car il s'agit de la classe de base des composants de l'interface utilisateur ActionScript 3.0. Pour afficher la liste des événements diffusés par un composant, consultez la section Evénements de l'entrée de classe du composant dans le *Guide de référence du langage et des composants ActionScript 3.0*.

Pour obtenir une description détaillée de la gestion des événements dans ActionScript 3.0, consultez le guide *Programmation d'ActionScript 3.0*.

Ecouteurs d'événements

Les points importants suivants s'appliquent à la gestion des événements des composants ActionScript 3.0 :

- Tous les événements sont diffusés par l'occurrence d'une classe de composant. L'occurrence de composant est le *diffuseur*.
- Vous pouvez enregistrer un *écouteur* d'événements en appelant la méthode `addEventListener()` de l'occurrence du composant. Par exemple, la ligne de code suivante ajoute un écouteur pour l'événement `MouseEvent.CLICK` à l'occurrence de bouton `aButton` :

  ```
  aButton.addEventListener(MouseEvent.CLICK, clickHandler);
  ```

  Le deuxième paramètre de la méthode `addEventListener()` enregistre le nom de la fonction, `clickHandler`, à appeler lorsque l'événement se produit. Cette fonction est également appelée *fonction de rappel*.
- Vous pouvez associer plusieurs écouteurs à une seule occurrence de composant.

  ```
  aButton.addEventListener(MouseEvent.CLICK, clickHandler1);
  aButton.addEventListener(MouseEvent.CLICK, clickHandler2);
  ```

- Vous pouvez associer un seul écouteur à plusieurs occurrences de composant.

```
aButton.addEventListener(MouseEvent.CLICK, clickHandler1);
bButton.addEventListener(MouseEvent.CLICK, clickHandler1);
```

- Un objet événement contenant des informations sur le type d'événement et l'occurrence qui diffuse l'événement est transmis à la fonction de gestionnaire d'événements. Pour plus d'informations, consultez la section « Objet événement », à la page 52.
- L'écouteur reste actif tant que l'application n'est pas terminée ou que vous ne l'avez pas supprimé de manière explicite via la méthode `removeEventListener()`. Par exemple, la ligne de code suivante supprime l'écouteur de l'événement `MouseEvent.CLICK` sur `aButton`:

```
aButton.removeEventListener(MouseEvent.CLICK, clickHandler);
```

Objet événement

L'objet événement hérite de la classe d'objet Event et possède des propriétés qui contiennent des informations sur l'événement qui s'est produit, notamment les propriétés `target` et `type` qui fournissent des informations essentielles sur l'événement :

Propriété	Description
`type`	Chaîne indiquant le type de l'événement.
`target`	Référence à l'occurrence de composant qui émet l'événement.

Lorsqu'un événement possède des propriétés supplémentaires, celles-ci sont répertoriées dans la description de la classe de l'événement dans le *Guide de référence du langage et des composants ActionScript 3.0*.

L'objet événement est automatiquement généré et transmis à la fonction de gestionnaire d'événements lorsqu'un événement se produit.

Vous pouvez l'utiliser à l'intérieur de la fonction pour connaître le nom de l'événement diffusé ou le nom d'occurrence du composant qui a diffusé l'événement. A partir du nom de l'occurrence, vous pouvez accéder à d'autres propriétés du composant. Le code suivant, par exemple, utilise la propriété `target` de l'objet événement `evtObj` pour accéder à la propriété `label` de `aButton` et l'afficher dans le panneau Sortie :

```
import fl.controls.Button;
import flash.events.MouseEvent;

var aButton:Button = new Button();
aButton.label = "Submit";
addChild(aButton);
aButton.addEventListener(MouseEvent.CLICK, clickHandler);

function clickHandler(evtObj:MouseEvent){
  trace("The " + evtObj.target.label + " button was clicked");
}
```

Utilisation de la liste d'affichage

Tous les composants ActionScript 3.0 héritent de la classe DisplayObject, ce qui leur permet d'accéder à ses méthodes et propriétés pour interagir avec la liste d'affichage. La *liste d'affichage* est la hiérarchie des objets affichés et des éléments visuels dans une application. Cette hiérarchie inclut les éléments suivants :

- La scène, qui est le conteneur de niveau supérieur
- Les objets d'affichage qui incluent notamment les formes, les clips et les champs de texte
- Les conteneurs d'objet d'affichage, qui sont des types spéciaux d'objets d'affichage pouvant contenir des objets d'affichage enfant.

L'ordre des objets dans la liste d'affichage détermine leur profondeur dans le conteneur parent. La profondeur d'un objet désigne sa position de haut en bas ou de l'avant vers l'arrière sur la scène ou dans son conteneur d'affichage. L'ordre de profondeur est apparent, indépendamment du fait que les objets se chevauchent ou non. A chaque objet de la liste d'affichage correspond une profondeur sur la scène. Si vous souhaitez modifier la profondeur d'un objet en le plaçant devant ou derrière les autres objets, vous devez modifier sa position dans la liste d'affichage. L'ordre par défaut des objets dans la liste d'affichage correspond à l'ordre dans lequel ils sont placés sur la scène. Dans la liste d'affichage, la position 0 correspond à l'objet au bas de l'ordre de profondeur.

Ajout d'un composant à la liste d'affichage

Vous pouvez ajouter un objet à un objet DisplayObjectContainer en appelant la méthode `addChild()` ou `addChildAt()` du conteneur. Dans le cas de la scène, vous pouvez également ajouter un objet à sa liste d'affichage lors de la programmation en le créant ou, dans le cas des composants, en le faisant glisser sur la scène depuis le panneau Composants. Pour ajouter un objet à un conteneur à l'aide d'ActionScript, commencez par créer une occurrence de celui-ci en appelant son constructeur à l'aide de l'opérateur `new`, puis la méthode `addChild()` ou `addChildAt()` pour le placer sur la scène et dans la liste d'affichage. La méthode `addChild()` place l'objet à la position suivante dans la liste d'affichage, tandis que la méthode `addChildAt()` spécifie la position à laquelle l'objet doit être ajouté. Si vous spécifiez une position déjà occupée, l'objet qui se trouve à cette position, et ceux des positions de niveau supérieur, sont décalés d'une position vers le haut. La propriété `numChildren` d'un objet DisplayObjectContainer contient le nombre d'objets d'affichage inclus dans celui-ci. Vous pouvez extraire un objet de la liste d'affichage en appelant la méthode `getChildAt()` et en spécifiant sa position, ou si vous connaissez le nom de l'objet, en appelant la méthode `getChildByName()`.

REMARQUE	Lorsque vous ajoutez un composant à l'aide d'ActionScript, vous devez affecter un nom à sa propriété de nom pour pouvoir y accéder par nom dans la liste d'affichage.

L'exemple suivant affiche les noms et positions de trois composants dans la liste d'affichage. Commencez par faire glisser les composants NumericStepper, Button et ComboBox sur la scène de manière à ce qu'ils se chevauchent et affectez-leur les noms d'occurrence **aNs**, **aButton** et **aCb**. Ajoutez ensuite le code suivant dans le panneau Actions, sur l'image1 du scénario :

```
var i:int = 0;
while(i < numChildren) {
  trace(getChildAt(i).name + " is at position: " + i++);
}
```

Les lignes suivantes doivent s'afficher dans le panneau Sortie :

```
aNs is at position: 0
aButton is at position: 1
aCb is at position: 2
```

Déplacement d'un composant dans la liste d'affichage

Vous pouvez modifier la position d'un objet dans la liste d'affichage, ainsi que sa profondeur d'affichage, en appelant la méthode `addChildAt()` et en spécifiant le nom d'un objet et la position où vous voulez le placer en tant que paramètres de la méthode. Par exemple, ajoutez le code suivant à l'exemple précédent de manière à placer le composant NumericStepper sur le dessus et répétez la boucle pour afficher les nouvelles positions des composants dans la liste d'affichage :

```
this.addChildAt(aNs, numChildren - 1);
i = 0;
while(i < numChildren) {
  trace(getChildAt(i).name + " is at position: " + i++);
}
```

Les lignes suivantes doivent s'afficher dans le panneau Sortie :

```
aNs is at position: 0
aButton is at position: 1
aCb is at position: 2
aButton is at position: 0
aCb is at position: 1
aNs is at position: 2
```

Le composant NumericStepper doit également s'afficher devant les autres composants à l'écran.

Notez que `numChildren` est le nombre d'objets (de 1 à *n*) dans la liste d'affichage tandis que la première position de la liste est 0. Par conséquent, si la liste contient trois objets, la position d'index du troisième objet est 2. Cela signifie que vous pouvez référencer la dernière position de la liste d'affichage, ou l'objet du dessus en termes de profondeur d'affichage, en tant que `numChildren - 1`.

Suppression d'un composant dans la liste d'affichage

Vous pouvez supprimer un composant d'un conteneur d'objet d'affichage et de sa liste d'affichage à l'aide des méthodes `removeChild()` et `removeChildAt()`. L'exemple suivant place trois composants Button l'un devant l'autre sur la scène et ajoute un écouteur d'événements pour chacun d'entre eux. Lorsque vous cliquez sur chaque composant Button, le gestionnaire d'événements le supprime de la liste d'affichage et de la scène.

Pour supprimer un composant dans la liste d'affichage :

1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Faites glisser un composant Button du panneau Composants vers le panneau Bibliothèque.
3. Ouvrez le panneau Actions, sélectionnez l'image 1 dans le scénario principal et ajoutez le code suivant :

```
import fl.controls.Button;

var i:int = 0;
while(i++ < 3) {
  makeButton(i);
}
function removeButton(event:MouseEvent):void {
  removeChildAt(numChildren -1);
}
function makeButton(num) {
  var aButton:Button = new Button();
  aButton.name = "Button" + num;
  aButton.label = aButton.name;
  aButton.move(200, 200);
  addChild(aButton);
  aButton.addEventListener(MouseEvent.CLICK, removeButton);
}
```

Pour obtenir une description détaillée de la liste d'affichage, consultez le Chapitre 12, « Programmation de l'affichage » du guide *Programmation d'ActionScript 3.0*.

Utilisation de FocusManager

Lorsqu'un utilisateur appuie sur la touche de tabulation pour naviguer dans une application Flash ou clique dans une application, la classe FocusManager détermine quel composant reçoit le focus d'entrée. Il n'est pas nécessaire d' ajouter une occurrence FocusManager à une application ou du code pour activer le gestionnaire de focus, sauf si vous créez un composant.

Si un objet RadioButton reçoit le focus, le gestionnaire de focus l'examine, de même que tous les objets ayant la même valeur `groupName`, puis attribue le focus à l'objet dont la propriété `selected` est définie sur `true`.

Chaque composant Window modal contenant une occurrence du gestionnaire de focus, ses contrôles constituent son propre jeu de tabulation. Ainsi, l'utilisateur ne risque pas de naviguer par erreur vers les composants des autres fenêtres en appuyant sur la touche de tabulation.

Le gestionnaire de focus utilise le niveau de profondeur (ou ordre *z*) des éléments du conteneur comme système de navigation par défaut ou *boucle de tabulation*. Un utilisateur navigue généralement dans la boucle de tabulation à l'aide de la touche de tabulation, le focus se déplaçant du premier composant ayant le focus jusqu'au dernier, et vice versa. Les niveaux de profondeur sont principalement définis par l'ordre dans lequel les composants sont déposés sur la scène. Vous pouvez néanmoins utiliser les commandes Modifier > Réorganiser > Mettre au premier plan/Mettre à l'arrière-plan pour déterminer l'ordre *z* final. Pour plus d'informations sur les niveaux de profondeur, consultez la section « Utilisation de la liste d'affichage », à la page 53.

Vous pouvez appeler la méthode `setFocus()` pour attribuer le focus à une occurrence de composant dans une application. Ainsi, l'exemple suivant crée une occurrence FocusManager pour le conteneur actuel (`this`) et attribue le focus à l'occurrence de bouton `aButton`.

```
var fm:FocusManager = new FocusManager(this);
fm.setFocus(aButton);
```

Vous pouvez déterminer quel composant a le focus en appelant la méthode `getFocus()` et quel composant de la boucle de tabulation le recevra ensuite en appelant la méthode `getNextFocusManagerComponent()`. Dans l'exemple suivant, un composant CheckBox, un composant RadioButton et un composant Button sont sur la scène et chacun d'entre eux possède des écouteurs pour les événements `MouseEvent.CLICK` et `FocusEvent.MOUSE_FOCUS_CHANGE`. Lorsque l'événement `MouseEvent.CLICK` se produit car l'utilisateur a cliqué sur le composant, la fonction `showFocus()` appelle la méthode `getNextFocusManagerComponent()` afin de déterminer le composant de la boucle de tabulation qui recevra ensuite le focus. Elle appelle ensuite la méthode `setFocus()` pour attribuer le focus à ce composant. Lorsque l'événement `FocusEvent.MOUSE_FOCUS_CHANGE` se produit, la fonction `fc()` affiche le nom du composant sur lequel cet événement s'est produit. Cet événement est déclenché lorsque l'utilisateur clique sur un composant autre que le suivant dans la boucle de tabulation.

```
// This example assumes a CheckBox (aCh), a RadioButton (aRb) and a Button
// (aButton) have been placed on the Stage.

import fl.managers.FocusManager;
import flash.display.InteractiveObject;

var fm:FocusManager = new FocusManager(this);

aCh.addEventListener(MouseEvent.CLICK, showFocus);
aRb.addEventListener(MouseEvent.CLICK, showFocus);
aButton.addEventListener(MouseEvent.CLICK, showFocus);
aCh.addEventListener(FocusEvent.MOUSE_FOCUS_CHANGE, fc);
aRb.addEventListener(FocusEvent.MOUSE_FOCUS_CHANGE, fc);
aButton.addEventListener(FocusEvent.MOUSE_FOCUS_CHANGE, fc);
```

```
function showFocus(event:MouseEvent):void {
  var nextComponent:InteractiveObject = fm.getNextFocusManagerComponent();
  trace("Next component in tab loop is: " + nextComponent.name);
  fm.setFocus(nextComponent);
}

function fc(fe:FocusEvent):void {
  trace("Focus Change: " + fe.target.name);
}
```

Pour créer un bouton recevant le focus lorsqu'un utilisateur appuie sur Entrée (Windows) ou sur Retour (Macintosh), définissez la propriété `FocusManager.defaultButton` sur l'occurrence du bouton devant être défini par défaut, comme dans le code suivant :

```
import fl.managers.FocusManager;

var fm:FocusManager = new FocusManager(this);
fm.defaultButton = okButton;
```

La classe FocusManager remplace le rectangle de focus par défaut de Flash Player par un rectangle de focus personnalisé dont les bords sont arrondis.

Pour plus d'informations sur la création d'un programme de focus dans une application Flash, consultez la Classe FocusManager du *Guide de référence du langage et des composants ActionScript 3.0*. Pour créer un gestionnaire de focus personnalisé, vous devez créer une classe qui implémente l'interface *IFocusManager*. Pour plus d'informations, reportez-vous à IFocusManager dans le *Guide de référence du langage et des composants ActionScript 3.0*.

Utilisation des composants basés sur des listes

Les composants List, DataGrid et TileList héritent tous de la classe de base SelectableList. Pour cette raison, ces composants sont considérés comme étant des composants basés sur des listes. Un composant ComboBox comprend une zone de texte et une liste : il s'agit donc également d'un composant basé sur des listes.

Un composant List est composé de lignes. Les composants DataGrid et TileList sont composés de lignes pouvant être divisées en plusieurs colonnes. L'intersection d'une ligne et d'une colonne est une cellule. Dans une liste, qui est une colonne unique de lignes, chaque ligne est une cellule. Une cellule se caractérise par les deux aspects importants suivants :

- Les valeurs de données contenues dans les cellules sont appelées éléments. Un *élément* est un objet ActionScript utilisé pour stocker les unités d'informations dans une liste. Une liste peut être considérée comme un tableau ; chaque espace indexé du tableau constitue un élément. Dans une liste, un élément est un objet qui dispose, en règle générale, d'une propriété `label` affichée et d'une propriété `data` utilisée pour stocker des données. Le *fournisseur de données* correspond au modèle de données des éléments d'une liste. Le fournisseur de données vous permet d'alimenter un composant basé sur des listes en l'affectant tout simplement à la propriété `dataProvider` du composant.
- Une cellule peut contenir différents types de données, comme du texte, des images, des clips ou les classes que vous voulez créer. Pour cette raison, le tracé ou le rendu de la cellule doit être approprié à son contenu. Par conséquent, les composants basés sur des listes disposent d'un composant *CellRenderer* leur permettant de définir le rendu de leurs cellules. Dans le cas du composant DataGrid, chaque colonne est un objet DataGridColumn, qui dispose également d'une propriété `cellRenderer`, permettant ainsi de définir le rendu du contenu de chaque colonne de manière appropriée.

Tous les composants basés sur des listes disposent de propriétés `cellRenderer` et `dataProvider` que vous pouvez définir pour charger et effectuer le rendu des cellules de ces composants. Pour plus d'informations sur l'utilisation de ces propriétés et des composants basés sur des listes, consultez la section « Utilisation d'un fournisseur de données », à la page 59 et la section « Utilisation d'un composant CellRenderer », à la page 70.

Utilisation d'un fournisseur de données

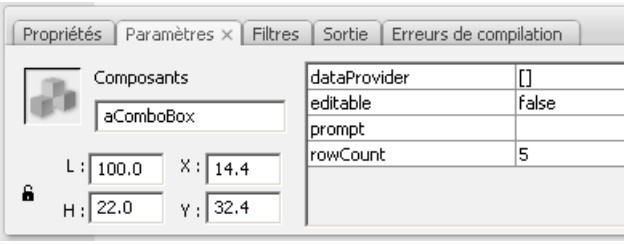
Le fournisseur de données est une source de données que vous pouvez utiliser pour fournir des données aux composants ComboBox, DataGrid, List et TileList. Chacune de ces classes de composant possède une propriété `dataProvider` à laquelle vous pouvez affecter un objet DataProvider afin d'inclure des données aux cellules du composant. En règle générale, un fournisseur de données est une collection de données tel qu'un tableau ou un objet XML.

Création d'un fournisseur de données

Vous pouvez créer un fournisseur de données pour les composants ComboBox, List et TileList à l'aide du paramètre `dataProvider` dans l'environnement de programmation. Le composant DataGrid ne dispose pas de paramètre dataProvider dans l'Inspecteur des propriétés car il peut contenir plusieurs colonnes, ce qui rend son fournisseur de données plus complexe. Vous pouvez également utiliser ActionScript pour créer un fournisseur de données pour ces composants, ainsi que pour le composant DataGrid.

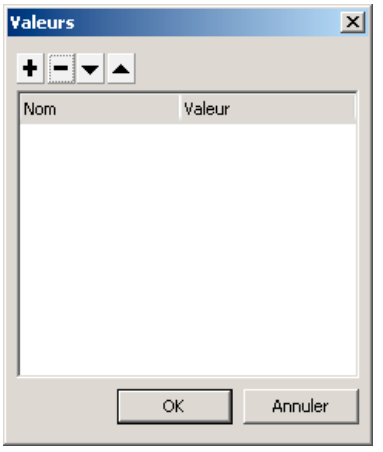
Utilisation du paramètre dataProvider

Vous pouvez créer un fournisseur de données simple pour les composants ComboBox, List et TileList en cliquant sur le paramètre dataProvider de l'onglet Paramètres de l'Inspecteur des propriétés ou de l'Inspecteur des composants. L'illustration suivante affiche le paramètre dans l'Inspecteur des propriétés.



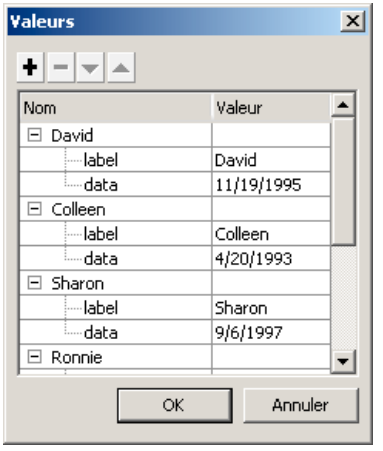
Paramètre dataProvider dans l'Inspecteur des propriétés

Si vous double-cliquez sur la cellule de valeur, qui affiche initialement un tableau vide, la boîte de dialogue Valeurs vous permettant d'entrer plusieurs valeurs d'étiquettes et de données afin de créer le fournisseur de données s'ouvre.



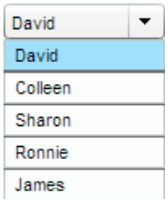
Boîte de dialogue Valeurs de dataProvider

Cliquez sur le signe plus pour ajouter un élément au fournisseur de données. Cliquez sur le signe moins pour supprimer un élément. Cliquez sur la flèche vers le haut pour déplacer vers le haut un élément sélectionné dans la liste ou cliquez sur la flèche vers le bas pour déplacer vers le bas un élément sélectionné dans la liste. L'illustration suivante affiche la boîte de dialogue Valeurs permettant de créer une liste de noms d'enfants incluant également leurs anniversaires.



Boîte de dialogue Valeurs incluant des données

Le tableau que vous créez se compose de paires de champs d'étiquette et de valeur. Les champs d'étiquette sont `label` et `data`, et les champs de valeur représentent les noms des enfants, ainsi que leurs anniversaires. Le champ d'étiquette identifie le contenu qui apparaît dans la liste, dans ce cas les noms des enfants. Le composant ComboBox obtenu a l'aspect suivant :



Composant ComboBox alimenté par le fournisseur de données

Lorsque vous avez terminé d'ajouter des données, cliquez sur OK pour fermer la boîte de dialogue. Le tableau du paramètre dataProvider inclut désormais les éléments que vous avez créés.

allowMultipleSelection	false
dataProvider	[{label:David,data:11/19/1995},{label:Colleen,data:4/20/1993},{label:Sharon,data:9/6/1997},
horizontalLineScrollSize	1
horizontalPageScrollSize	0
horizontalScrollPolicy	auto
verticalLineScrollSize	1

Paramètre dataProvider avec des données

Vous pouvez accéder aux valeurs d'étiquettes et de données que vous avez créées à l'aide d'ActionScript pour accéder à la propriété `dataProvider` du composant.

Utilisation d'ActionScript

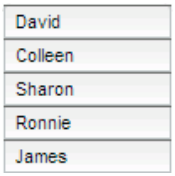
Vous pouvez créer un fournisseur de données en créant les données dans un tableau ou un objet XML et en affectant l'objet en tant que paramètre `value` au constructeur DataProvider.

REMARQUE	Dans ActionScript 3.0, vous ne pouvez pas affecter de tableau ou d'objet XML directement à une propriété dataProvider car celle-ci est définie en tant qu'objet DataProvider et peut uniquement recevoir un objet de type DataProvider.

L'exemple suivant remplit un composant List, qui est une colonne unique de lignes, avec les noms de plusieurs enfants et leurs anniversaires. Cet exemple définit la liste dans le tableau des `éléments` et la fournit en tant que paramètre lorsqu'il crée l'occurrence DataProvider (`new DataProvider(items)`), puis l'affecte à la propriété `dataProvider` du composant List.

```
import fl.controls.List;
import fl.data.DataProvider;

var aList:List = new List();
var items:Array = [
{label:"David", data:"11/19/1995"},
{label:"Colleen", data:"4/20/1993"},
{label:"Sharon", data:"9/06/1997"},
{label:"Ronnie", data:"7/6/1993"},
{label:"James", data:"2/15/1994"},
];
aList.dataProvider = new DataProvider(items);
addChild(aList);
aList.move(150,150);
```

Le tableau se compose de paires de champs d'étiquette et de valeur. Les champs d'étiquette sont `label` et `data`, et les champs de valeur représentent les noms des enfants, ainsi que leurs anniversaires. Le champ d'étiquette identifie le contenu qui apparaît dans la liste, dans ce cas les noms des enfants. Le composant List obtenu a l'aspect suivant :

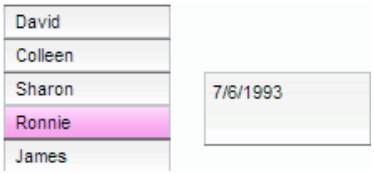


Composant List alimenté par un fournisseur de données

La valeur du champ de données est disponible lorsque l'utilisateur sélectionne un élément dans la liste en cliquant dessus pour déclencher un événement `change`. L'exemple suivant ajoute un composant TextArea (`aTa`) et un gestionnaire d'événements (`changeHandler`) à l'exemple précédent pour afficher l'anniversaire de l'enfant lorsqu'un utilisateur sélectionne un nom dans la liste.

```
import fl.controls.List;
import fl.controls.TextArea;
import flash.events.Event;
import fl.data.DataProvider;

var aList:List = new List();
var aTa:TextArea = new TextArea();
var items:Array = [
{label:"David", data:"1/19/1995"},
{label:"Colleen", data:"4/20/1993"},
{label:"Sharon", data:"9/06/1994"},
{label:"Ronnie", data:"7/6/1993"},
{label:"James", data:"2/15/1994"},
];
aList.dataProvider = new DataProvider(items);

addChild(aList);
addChild(aTa);

aList.move(150,150);
aTa.move(150, 260);

aList.addEventListener(Event.CHANGE, changeHandler);

function changeHandler(event:Event):void {
  aTa.text = event.target.selectedItem.data;
};
```

Désormais, lorsqu'un utilisateur sélectionne le nom d'un enfant dans la liste, la date de son anniversaire s'affiche dans le composant TextArea, comme indiqué dans l'illustration suivante. Cette opération est effectuée par la fonction `changeHandler()` lorsqu'elle définit la propriété `text` du composant TextArea (`aTa.text`) sur la valeur du champ de données dans l'élément sélectionné (`event.target.selectedItem.data`). La propriété `event.target` est l'objet qui a déclenché l'événement, soit l'objet List dans le cas présent.



Affichage du champ de données du fournisseur de données d'un composant List

Vous pouvez inclure des données autres que du texte dans un fournisseur de données. L'exemple suivant inclut des clips dans un fournisseur de données qui affecte des données à un composant TileList. Il crée le fournisseur de données en appelant la méthode `addItem()` pour ajouter chaque élément après avoir créé le clip, une zone colorée.

```
import fl.data.DataProvider;
import flash.display.DisplayObject;

var aBox:MovieClip = new MovieClip();
var i:uint = 0;
var colors:Array = new Array(0x00000, 0xFF0000, 0x0000CC, 0x00CC00,
  0xFFFF00);
var colorNames:Array = new Array("Midnight", "Cranberry", "Sky", "Forest",
  "July");
var dp:DataProvider = new DataProvider();
for(i=0; i < colors.length; i++) {
  drawBox(aBox, colors[i]);// draw box w next color in array
  dp.addItem( {label:colorNames[i], source:aBox} );
}
aTl.dataProvider = dp;
aTl.columnWidth = 110;
aTl.rowHeight = 130;
aTl.setSize(280,150);
aTl.move(150, 150);
aTl.setStyle("contentPadding", 5);

function drawBox(box:MovieClip,color:uint):void {
       box.graphics.beginFill(color, 1.0);
       box.graphics.drawRect(0, 0, 100, 100);
       box.graphics.endFill();
```

Vous pouvez également utiliser des données XML (au lieu d'un tableau) pour remplir un objet DataProvider. Par exemple, le code suivant stocke des données dans un objet XML intitulé `employeesXML`, puis transmet cet objet en tant que paramètre de valeur de la fonction constructeur `DataProvider()` :

```
import fl.controls.DataGrid;
import fl.data.DataProvider;

var aDg:DataGrid = new DataGrid();
addChild(aDg);

var employeesXML:XML =
  <employees>
    <employee Name="Edna" ID="22" />
    <employee Name="Stu" ID="23" />
  </employees>;

var myDP:DataProvider = new DataProvider(employeesXML);

aDg.columns = ["Name", "ID"];
aDg.dataProvider = myDP;
```

Vous pouvez fournir des données en tant qu'attributs des données XML, comme dans le code précédent, ou en tant que propriétés des données XML, comme dans le code suivant :

```
var employeesXML:XML =
  <employees>
    <employee>
      <Name>Edna</Name>
      <ID>22</ID>
    </employee>
    <employee>
      <Name>Stu</Name>
      <ID>23</ID>
    </employee>
  </employees>;
```

Le fournisseur de données dispose également d'un ensemble de méthodes et de propriétés vous permettant d'y accéder et de le manipuler. Vous pouvez utiliser l'API du fournisseur de données pour ajouter, supprimer, remplacer, trier et fusionner des éléments dans un fournisseur de données.

Manipulation d'un fournisseur de données

Vous pouvez ajouter des éléments à un fournisseur de données à l'aide des méthodes `addItem()` et `addItemAt()`. L'exemple suivant ajoute les éléments entrés par un utilisateur dans le champ texte d'un composant ComboBox modifiable. Il suppose qu'un composant ComboBox a été placé sur la scène et a reçu le nom d'occurrence `aCb`.

```
import fl.data.DataProvider;
import fl.events.ComponentEvent;

var items:Array = [
{label:"Roger"},
{label:"Carolyn"},
{label:"Darrell"},
{label:"Rebecca"},
{label:"Natalie"},
{label:"Mitchell"},
];
aCb.dataProvider = new DataProvider(items);

aCb.addEventListener(ComponentEvent.ENTER, newItemHandler);

function newItemHandler(event:ComponentEvent):void {
  var newRow:int = event.target.length + 1;
  event.target.addItemAt({label:event.target.selectedLabel},
    event.target.length);
}
```

Vous pouvez également remplacer et supprimer des éléments d'un composant par l'intermédiaire de son fournisseur de données. L'exemple suivant implémente deux composants List distincts, `listA` et `listB`, et contient le bouton Sync. Lorsqu'un utilisateur clique sur le bouton, l'exemple utilise la méthode `replaceItemAt()` pour remplacer les éléments du composant `listB` par ceux du composant `listA`. Si le composant `listA` est plus long que le composant `listB`, l'exemple appelle la méthode `addItem()` pour ajouter les éléments supplémentaires au composant `listB`. Si le composant `listB` est plus long que le composant `listA`, l'exemple appelle la méthode `removeItemAt()` pour supprimer les éléments supplémentaires du composant `ListB`.

```
// Requires the List and Button components to be in the library

import fl.controls.List;
import fl.controls.Button;
import flash.events.Event;
import fl.data.DataProvider;

var listA:List = new List();
var listB:List = new List();
var syncButton:Button = new Button();
syncButton.label = "Sync";

var itemsA:Array = [
{label:"David"},
{label:"Colleen"},
{label:"Sharon"},
{label:"Ronnie"},
{label:"James"},
];
var itemsB:Array = [
{label:"Roger"},
{label:"Carolyn"},
{label:"Darrell"},
{label:"Rebecca"},
{label:"Natalie"},
{label:"Mitchell"},
];
listA.dataProvider = new DataProvider(itemsA);
listB.dataProvider = new DataProvider(itemsB);

addChild(listA);
addChild(listB);
addChild(syncButton);

listA.move(100, 100);
listB.move(250, 100);
syncButton.move(175, 220);

syncButton.addEventListener(MouseEvent.CLICK, syncHandler);
```

```
function syncHandler(event:MouseEvent):void {
  var i:uint = 0;
  if(listA.length > listB.length) { //if listA is longer, add items to B
    while(i < listB.length) {
      listB.dataProvider.replaceItemAt(listA.dataProvider.getItemAt(i),
  i);
      ++i;
    }
    while(i < listA.length) {
      listB.dataProvider.addItem(listA.dataProvider.getItemAt(i++));
    }
  } else if(listA.length == listB.length) { //if listA and listB are
                                            //equal length
    while(i < listB.length) {
      listB.dataProvider.replaceItemAt(listA.dataProvider.getItemAt(i),
  i);
      ++i;
    }
  } else { //if listB is longer, remove extra items from B
    while(i < listA.length) {
      listB.dataProvider.replaceItemAt(listA.dataProvider.getItemAt(i),
  i);
      ++i;
    }
    while(i < listB.length) {
      listB.dataProvider.removeItemAt(i++);
    }
  }
}
```

Vous pouvez également fusionner avec un fournisseur de données et le trier à l'aide des méthodes `merge()`, `sort()` et `sortOn()`. L'exemple suivant remplit deux occurrences DataGrid (`aDg` et `bDg`) avec des listes partielles pour deux équipes de softball. Il ajoute le bouton Fusionner et lorsque l'utilisateur clique sur celui-ci, le gestionnaire d'événements (`mrgHandler`) fusionne la liste de l'occurrence `bDg` avec celle de l'occurrence `aDg` et trie l'occurrence DataGrid résultante selon la colonne Nom.

```
import fl.data.DataProvider;
import fl.controls.DataGrid;
import fl.controls.Button;

var aDg:DataGrid = new DataGrid();
var bDg:DataGrid = new DataGrid();
var mrgButton:Button = new Button();
addChild(aDg);
addChild(bDg);
addChild(mrgButton);
bldRosterGrid(aDg);
bldRosterGrid(bDg);
```

```
var aRoster:Array = new Array();
var bRoster:Array = new Array();
aRoster = [
    {Name:"Wilma Carter", Bats:"R", Throws:"R", Year:"So", Home: "Redlands,
  CA"},
    {Name:"Sue Pennypacker", Bats:"L", Throws:"R", Year:"Fr", Home:
  "Athens, GA"},
    {Name:"Jill Smithfield", Bats:"R", Throws:"L", Year:"Sr", Home:
  "Spokane, WA"},
    {Name:"Shirley Goth", Bats:"R", Throws:"R", Year:"Sr", Home: "Carson,
  NV"}
];
bRoster = [
    {Name:"Angelina Davis", Bats:"R", Throws:"R", Year:"So", Home: "Odessa,
  TX"},
    {Name:"Maria Santiago", Bats:"L", Throws:"L", Year:"Sr", Home: "Tacoma,
  WA"},
    {Name:"Debbie Ferguson", Bats:"R", Throws:"R", Year: "Jr", Home: "Bend,
  OR"}
];
aDg.dataProvider = new DataProvider(aRoster);
bDg.dataProvider = new DataProvider(bRoster);
aDg.move(50,50);
aDg.rowCount = aDg.length;
bDg.move(50,200);
bDg.rowCount = bDg.length;
mrgButton.label = "Merge";
mrgButton.move(200, 315);
mrgButton.addEventListener(MouseEvent.CLICK, mrgHandler);

function bldRosterGrid(dg:DataGrid){
  dg.setSize(400, 300);
  dg.columns = ["Name", "Bats", "Throws", "Year", "Home"];
  dg.columns[0].width = 120;
  dg.columns[1].width = 50;
  dg.columns[2].width = 50;
  dg.columns[3].width = 40;
  dg.columns[4].width = 120;
};

function mrgHandler(event:MouseEvent):void {
  aDg.dataProvider.merge(bDg.dataProvider);
  aDg.dataProvider.sortOn("Name");
}
```

Pour plus d'informations, consultez la classe DataProvider dans le *Guide de référence du langage et des composants ActionScript 3.0*.

Utilisation d'un composant CellRenderer

CellRenderer est une classe utilisée par les composants basés sur des listes, tels que List, DataGrid, TileList et ComboBox pour manipuler et afficher le contenu des cellules personnalisées dans leurs lignes. Une cellule personnalisée peut contenir du texte, un composant prédéfini, par exemple un objet CheckBox, ou n'importe quelle classe d'objet d'affichage que vous pouvez créer. Pour procéder au rendu des données à l'aide d'un composant CellRenderer personnalisé, vous pouvez étendre la classe CellRenderer ou implémenter l'interface ICellRenderer pour créer votre propre classe CellRenderer personnalisée.

Les classes List, DataGrid, TileList et ComboBox sont des sous-classes de la classe SelectableList. La classe SelectableList inclut un style `cellRenderer`. Ce style définit l'objet d'affichage utilisé par le composant pour procéder au rendu des cellules.

Vous pouvez ajuster la mise en forme des styles utilisés par le composant CellRenderer, en appelant la méthode `setRendererStyle()` de l'objet List (consultez la section suivante, « Mise en forme des cellules »). Vous pouvez également définir une classe personnalisée à utiliser en tant que composant CellRenderer (consultez la section « Définition d'une classe CellRenderer personnalisée », à la page 71).

Mise en forme des cellules

La classe CellRenderer inclut un certain nombre de styles vous permettant de contrôler le format de la cellule.

Les styles suivants vous permettent de définir les enveloppes utilisées pour les différents états de la cellule (désactivé, abaissé, survolé et relevé) :

- `disabledSkin` et `selectedDisabledSkin`
- `downSkin` et `selectedDownSkin`
- `overSkin` et `selectedOverSkin`
- `upSkin` et `selectedUpSkin`

Les styles suivants s'appliquent à la mise en forme du texte :

- `disabledTextFormat`
- `textFormat`
- `textPadding`

Vous pouvez définir ces styles en appelant la méthode `setRendererStyle()` de l'objet List ou en appelant la méthode `setStyle()` de l'objet CellRenderer. Vous pouvez obtenir ces styles en appelant la méthode `getRendererStyle()` de l'objet List ou en appelant la méthode `setStyle()` de l'objet CellRenderer. Vous pouvez également accéder à un objet qui définit tous les styles de rendu (en tant que propriétés nommées de l'objet) via la propriété `rendererStyles` de l'objet List ou la méthode `getStyleDefinition()` de l'objet CellRenderer.

Vous pouvez appeler la méthode `clearRendererStyle()` pour restaurer la valeur par défaut d'un style.

Pour obtenir ou définir la hauteur des lignes dans la liste, utilisez la propriété `rowHeight` de l'objet List.

Définition d'une classe CellRenderer personnalisée

Par exemple, le code suivant inclut deux classes. La classe ListSample instancie un composant List et utilise l'autre classe, CustomRenderer, pour définir la classe CellRenderer à utiliser pour le composant List. La classe CustomRenderer étend la classe CellRenderer.

Pour utiliser une classe qui étend la classe CellRenderer pour définir une classe CellRenderer personnalisée :

1. Sélectionnez Fichier > Nouveau.
2. Dans la boîte de dialogue Nouveau document qui s'affiche, sélectionnez Fichier Flash (ActionScript 3.0), puis cliquez sur OK.
3. Choisissez Fenêtre> Composants pour afficher le panneau Composants.
4. Dans le panneau Composants, faites glisser un composant List sur la scène.
5. Si Flash n'affiche pas l'Inspecteur des propriétés, choisissez Fenêtre > Propriétés > Propriétés.
6. Après avoir sélectionné le composant List, définissez les propriétés dans l'Inspecteur des propriétés :
   - Nom d'occurrence : myList
   - L (largeur) : 200
   - H (hauteur) : 300
   - X : 20
   - Y : 20
7. Sélectionnez l'image 1 du calque 1 dans le scénario, puis Fenêtre > Actions.

8. Tapez le script suivant dans le panneau Actions :

```
myList.setStyle("cellRenderer", CustomCellRenderer);
myList.addItem({label:"Burger -- $5.95"});
myList.addItem({label:"Fries -- $1.95"});
```

9. Choisissez Fichier > Enregistrer. Attribuez un nom au fichier et cliquez sur le bouton OK.

10. Sélectionnez Fichier > Nouveau.

11. Dans la boîte de dialogue Nouveau document qui s'affiche, sélectionnez Fichier ActionScript, puis cliquez sur le bouton OK.

12. Dans la fenêtre de script, entrez le code suivant pour définir la classe CustomCellRenderer :

```
package {
  import fl.controls.listClasses.CellRenderer;
  import flash.text.TextFormat;
  import flash.filters.BevelFilter;
  public class CustomCellRenderer2 extends CellRenderer {
    public function CustomCellRenderer2() {
      var format:TextFormat = new TextFormat("Verdana", 12);
      setStyle("textFormat", format);
      this.filters = [new BevelFilter()];
    }
  }
}
```

13. Choisissez Fichier > Enregistrer. Nommez le fichier CustomCellRenderer.as, placez-le dans le même répertoire que le fichier FLA, puis cliquez sur le bouton OK.

14. Choisissez Contrôle> Tester l'animation.

Vous pouvez également définir une classe CellRenderer à l'aide des classes qui héritent de la classe DisplayObject et qui implémentent l'interface ICellRenderer. Par exemple, le code suivant définit deux classes. La classe ListSample2 ajoute un objet List à la liste d'affichage et définit sa classe CellRenderer pour utiliser la classe CustomRenderer. La classe CustomRenderer étend la classe CheckBox (qui étend la classe DisplayObject) et implémente l'interface ICellRenderer. Notez que la classe CustomRenderer définit les méthodes de lecture et de définition des propriétés `data` et `listData`, définies dans l'interface ICellRenderer. D'autres propriétés et méthodes définies dans l'interface ICellRenderer (la propriété `selected` et la méthode `setSize()`) sont déjà définies dans la classe CheckBox :

Pour utiliser une classe qui implémente l'interface ICellRenderer pour définir une classe CellRenderer personnalisée :

1. Sélectionnez Fichier > Nouveau.

2. Dans la boîte de dialogue Nouveau document qui s'affiche, sélectionnez Fichier Flash (ActionScript 3.0), puis cliquez sur OK.

3. Choisissez Fenêtre> Composants pour afficher le panneau Composants.

4. Dans le panneau Composants, faites glisser un composant List sur la scène.
5. Si Flash n'affiche pas l'Inspecteur des propriétés, choisissez Fenêtre > Propriétés > Propriétés.
6. Après avoir sélectionné le composant List, définissez les propriétés dans l'Inspecteur des propriétés :
   - Nom d'occurrence : myList
   - L (largeur) : 100
   - H (hauteur) : 300
   - X : 20
   - Y : 20
7. Sélectionnez l'image 1 du calque 1 dans le scénario, puis Fenêtre > Actions.
8. Tapez le script suivant dans le panneau Actions :

```
myList.setStyle("cellRenderer", CustomCellRenderer);
myList.addItem({name:"Burger", price:"$5.95"});
myList.addItem({name:"Fries", price:"$1.95"});
```

9. Choisissez Fichier > Enregistrer. Attribuez un nom au fichier et cliquez sur le bouton OK.
10. Sélectionnez Fichier > Nouveau.
11. Dans la boîte de dialogue Nouveau document qui s'affiche, sélectionnez Fichier ActionScript, puis cliquez sur le bouton OK.
12. Dans la fenêtre de script, entrez le code suivant pour définir la classe CustomCellRenderer :

```
package
{
  import fl.controls.CheckBox;
  import fl.controls.listClasses.ICellRenderer;
  import fl.controls.listClasses.ListData;
  public class CustomCellRenderer extends CheckBox implements ICellRenderer
  {
    private var _listData:ListData;
    private var _data:Object;
    public function CustomCellRenderer() {
    }
    public function set data(d:Object):void {
      _data = d;
      label = d.label;
    }
    public function get data():Object {
      return _data;
    }
```

```
    public function set listData(ld:ListData):void {
      _listData = ld;
    }
    public function get listData():ListData {
      return _listData;
    }
  }
}
```

13. Choisissez Fichier > Enregistrer. Nommez le fichier CustomCellRenderer.as, placez-le dans le même répertoire que le fichier FLA, puis cliquez sur le bouton OK.
14. Choisissez Contrôle> Tester l'animation.

Vous pouvez également utiliser un symbole de la bibliothèque pour définir une classe CellRenderer. Vous devez exporter le symbole pour ActionScript. En outre, un fichier de classe qui implémente l'interface ICellRenderer ou qui étend la classe CellRenderer (ou l'une de ses sous-classes) doit être associé au nom de classe du symbole de la bibliothèque.

L'exemple suivant définit une classe CellRenderer personnalisée à l'aide d'un symbole de la bibliothèque.

Pour utiliser un symbole de la bibliothèque pour définir une classe CellRenderer :

1. Sélectionnez Fichier > Nouveau.
2. Dans la boîte de dialogue Nouveau document qui s'affiche, sélectionnez Fichier Flash (ActionScript 3.0), puis cliquez sur OK.
3. Choisissez Fenêtre> Composants pour afficher le panneau Composants.
4. Dans le panneau Composants, faites glisser un composant List sur la scène.
5. Si Flash n'affiche pas l'Inspecteur des propriétés, choisissez Fenêtre > Propriétés > Propriétés.
6. Après avoir sélectionné le composant List, définissez les propriétés dans l'Inspecteur des propriétés :
   - Nom d'occurrence : myList
   - L (largeur) : 100
   - H (hauteur) : 400
   - X : 20
   - Y : 20
7. Cliquez sur le panneau Paramètres, puis double-cliquez sur la deuxième colonne de la ligne dataProvider.

8. Dans la boîte de dialogue Valeurs qui s'affiche, cliquez sur le signe plus à deux reprises pour ajouter deux éléments de données (avec les étiquettes définies sur label0 et label1), puis cliquez sur le bouton OK.
9. A l'aide de l'outil Texte, dessinez un champ texte sur la scène.
10. Après avoir sélectionné le champ texte, définissez les propriétés dans l'Inspecteur des propriétés :
    - Type de texte : Texte dynamique
    - Nom d'occurrence : textField
    - L (largeur) : 100
    - Taille de police : 24
    - X : 0
    - Y : 0
11. Après avoir sélectionné le champ texte, choisissez Modification > Convertir en symbole.
12. Dans la boîte de dialogue Convertir en symbole, définissez les paramètres suivants, puis cliquez sur OK.
    - Nom : MyCellRenderer
    - Type : MovieClip
    - Exporter pour ActionScript : Sélectionné
    - Exporter dans la première image : Sélectionné
    - Classe : MyCellRenderer
    - Classe de base : flash.display.SimpleButton

    Si Flash affiche un avertissement de classe ActionScript, cliquez sur le bouton OK dans la boîte de dialogue d'avertissement.
13. Supprimez l'occurrence du nouveau symbole de clip sur la scène.
14. Sélectionnez l'image 1 du calque 1 dans le scénario, puis Fenêtre > Actions.
15. Tapez le script suivant dans le panneau Actions :

```
myList.setStyle("cellRenderer", MyCellRenderer);
```

16. Choisissez Fichier> Enregistrer. Attribuez un nom au fichier et cliquez sur le bouton OK.
17. Sélectionnez Fichier > Nouveau.
18. Dans la boîte de dialogue Nouveau document qui s'affiche, sélectionnez Fichier ActionScript, puis cliquez sur le bouton OK.

19. Dans la fenêtre de script, entrez le code suivant pour définir la classe MyCellRenderer :

```
package {
  import flash.display.MovieClip;
  import flash.filters.GlowFilter;
  import flash.text.TextField;
  import fl.controls.listClasses.ICellRenderer;
  import fl.controls.listClasses.ListData;
  import flash.utils.setInterval;
  public class MyCellRenderer extends MovieClip implements ICellRenderer
  {
    private var _listData:ListData;
    private var _data:Object;
    private var _selected:Boolean;
    private var glowFilter:GlowFilter;
    public function MyCellRenderer() {
      glowFilter = new GlowFilter(0xFFFF00);
      setInterval(toggleFilter, 200);
    }
    public function set data(d:Object):void {
      _data = d;
      textField.text = d.label;
    }
    public function get data():Object {
      return _data;
    }
    public function set listData(ld:ListData):void {
      _listData = ld;
    }
    public function get listData():ListData {
      return _listData;
    }
    public function set selected(s:Boolean):void {
      _selected = s;
    }
    public function get selected():Boolean {
      return _selected;
    }
    public function setSize(width:Number, height:Number):void {
    }
    public function setStyle(style:String, value:Object):void {
    }
    private function toggleFilter():void {
      if (textField.filters.length == 0) {
        textField.filters = [glowFilter];
      } else {
        textField.filters = [];
      }
    }
  }
}
```

20. Choisissez Fichier > Enregistrer. Nommez le fichier MyCellRenderer.as, placez-le dans le même répertoire que le fichier FLA, puis cliquez sur le bouton OK.

21. Choisissez Contrôle> Tester l'animation.

Propriétés CellRenderer

La propriété `data` est un objet qui contient toutes les propriétés définies pour la classe CellRenderer. Par exemple, dans la classe suivante, qui définit une classe CellRenderer personnalisée qui étend la classe Checkbox, vous remarquerez que la fonction de lecture de la propriété `data` transmet la valeur de `data.label` à la propriété `label` héritée de la classe CheckBox :

```
  public class CustomRenderer extends CheckBox implements ICellRenderer {
    private var _listData:ListData;
    private var _data:Object;
    public function CustomRenderer() {
    }
    public function set data(d:Object):void {
      _data = d;
      label = d.label;
    }
    public function get data():Object {
      return _data;
    }
    public function set listData(ld:ListData):void {
      _listData = ld;
    }
    public function get listData():ListData {
      return _listData;
    }
  }
}
```

La propriété `selected` définit si une cellule est sélectionnée dans la liste ou non.

Application d'une classe CellRenderer à une colonne d'un objet DataGrid

Un objet DataGrid peut disposer de plusieurs colonnes ; vous pouvez spécifier différentes classes CellRenderer pour chacune d'entre elles. Chaque colonne d'un objet DataGrid est représentée par un objet DataGridColumn ; la classe DataGridColumn inclut une propriété `cellRenderer` pour laquelle vous pouvez définir la classe CellRenderer de la colonne.

Définition d'une classe CellRenderer pour une cellule modifiable

La classe DataGridCellEditor définit un rendu utilisé pour les cellules modifiables d'un objet DataGrid. Il devient le rendu d'une cellule lorsque la propriété `editable` de l'objet DataGrid est définie sur `true` et lorsque l'utilisateur clique sur la cellule à modifier. Pour définir une classe CellRenderer pour la cellule modifiable, définissez la propriété `itemEditor` de chaque élément du tableau `columns` de l'objet DataGrid.

Utilisation d'une image, d'un fichier SWF ou d'un clip en tant que classe CellRenderer

La classe ImageCell, sous-classe de la classe CellRenderer, définit un objet utilisé pour procéder au rendu des cellules dans lesquelles le contenu principal est une image, un fichier SWF ou un clip. La classe ImageCell inclut les styles suivants pour définir l'aspect de la cellule :

- `imagePadding` : marge intérieure qui sépare le bord de la cellule du bord de l'image, en pixels
- `selectedSkin` : enveloppe utilisée pour indiquer l'état sélectionné
- `textOverlayAlpha` : opacité de la superposition derrière l'étiquette de la cellule
- `textPadding` : marge intérieure qui sépare le bord de la cellule du bord du texte, en pixels

La classe ImageCell est le CellRenderer par défaut de la classe TileList.

Accessibilité des composants

Vous pouvez permettre aux utilisateurs malvoyants d'accéder au contenu visuel de vos applications Flash grâce à un logiciel de lecture d'écran qui fournit une description sonore du contenu de l'écran. Pour plus d'informations sur le déploiement d'un logiciel de lecture d'écran dans votre application Flash, consultez le chapitre 18, « Création de contenu accessible » du guide *Utilisation de Flash*.

Pour rendre un composant ActionScript 3.0 accessible à un logiciel de lecture d'écran, vous devez également importer sa classe d'accessibilité et appeler la méthode `enableAccessibility()` de cette classe. Vous pouvez rendre les composants ActionScript 3.0 suivants accessibles à un logiciel de lecture d'écran :

Composant	Classe d'accessibilité
Button	`ButtonAccImpl`
CheckBox	`CheckBoxAccImpl`
ComboBox	`ComboBoxAccImpl`
List	`ListAccImpl`
RadioButton	`RadioButtonAccImpl`
TileList	`TileListAccImpl`

Les classes d'accessibilité des composants se trouvent dans le paquet `fl.accessibility`. Pour rendre un composant CheckBox accessible à un logiciel de lecture d'écran, par exemple, vous devez ajouter les instructions suivantes à votre application :

```
import fl.accessibility.CheckBoxAccImpl;

CheckBoxAccImpl.enableAccessibility();
```

Quel que soit le nombre d'occurrences d'un composant que vous créez, l'activation de son accessibilité ne se fait qu'une fois.

REMARQUE

L'activation de l'accessibilité augmente légèrement la taille du fichier en incluant les classes requises lors de la compilation.

Vous pouvez également utiliser le clavier pour accéder à la plupart des composants. Pour plus d'informations sur l'activation des composants accessibles et sur l'utilisation du clavier pour y accéder, consultez les sections Interaction de l'utilisateur du guide « Utilisation des composants de l'interface utilisateur » et les classes d'accessibilité du *Guide de référence du langage et des composants ActionScript 3.0*.

CHAPITRE 3

Utilisation des composants de l'interface utilisateur

3

Ce chapitre explique comment utiliser les composants de l'interface utilisateur ActionScript 3.0 suivants, inclus dans Flash :

Utilisation du composant Button

Le composant Button est un bouton rectangulaire pouvant être redimensionné sur lequel un utilisateur peut appuyer à l'aide de la souris ou de la barre d'espace pour déclencher une action dans l'application. Vous pouvez ajouter une icône personnalisée à un bouton. Vous pouvez également modifier son comportement pour transformer un bouton-poussoir en bouton bascule. Un bouton bascule reste enfoncé lorsque l'utilisateur clique dessus, puis reprend l'état relevé au clic suivant.

Les boutons sont les éléments de base de nombreux formulaires et de nombreuses applications Web. Chaque fois que l'utilisateur doit pouvoir déclencher un événement, vous utilisez des boutons. Par exemple, la plupart des formulaires comportent un bouton Envoyer.
Vous pouvez également ajouter des boutons Précédent et Suivant à une présentation.

Interaction de l'utilisateur avec le bouton

Vous pouvez activer ou désactiver un bouton dans une application. En état désactivé, un bouton ne réagit pas aux commandes de la souris ou du clavier. Un bouton activé reçoit le focus si vous cliquez sur son entrée ou si vous appuyez sur la touche de tabulation pour l'atteindre. Lorsque l'occurrence d'un bouton a le focus, vous pouvez la contrôler à l'aide des touches suivantes :

Touche	Description
Maj+Tab	Place le focus sur l'objet précédent.
Espace	Sélectionne ou libère le bouton et déclenche l'événement `click`.
Tab	Place le focus sur l'objet suivant.
Entrée/Retour	Place le focus sur l'objet suivant si un bouton est défini comme étant le bouton par défaut de FocusManager.

Pour plus d'informations sur le contrôle du focus, consultez l'interface IFocusManager et la classe FocusManager dans le *Guide de référence du langage et des composants ActionScript 3.0* et « Utilisation de FocusManager », à la page 56.

L'aperçu en direct des occurrences de bouton reflète les modifications apportées aux paramètres dans l'inspecteur Propriétés ou l'inspecteur des composants pendant la programmation.

REMARQUE	Lorsque l'icône est plus grande que le bouton, elle dépasse ses bordures.

Pour désigner un bouton comme bouton-poussoir par défaut dans une application (bouton qui reçoit l'événement click lorsque l'utilisateur appuie sur Entrée), définissez `FocusManager.defaultButton`. Par exemple, le code suivant définit le bouton par défaut comme étant une occurrence de bouton intitulée `submitButton`.

```
FocusManager.defaultButton = submitButton;
```

Lorsque vous ajoutez le composant Button à une application, vous pouvez le rendre accessible à un lecteur d'écran en ajoutant les lignes de code ActionScript suivantes :

```
import fl.accessibility.ButtonAccImpl;

ButtonAccImpl.enableAccessibility();
```

Quel que soit le nombre d'occurrences d'un composant que vous créez, l'activation de son accessibilité ne se fait qu'une fois.

Paramètres du composant Button

Vous pouvez définir les paramètres de programmation suivants dans l'Inspecteur des propriétés (Fenêtre > Propriétés > Propriétés) ou dans l'Inspecteur des composants (Fenêtre > Inspecteur des composants) pour chaque occurrence Button : `emphasized`, `label`, `labelPlacement`, `selected` et `toggle`. Chacun de ces paramètres possède une propriété ActionScript correspondante du même nom. Lorsque vous affectez une valeur à ces paramètres, vous définissez l'état initial de la propriété dans l'application. La définition de la propriété dans ActionScript remplace la valeur que vous avez définie dans le paramètre. Pour plus d'informations sur les valeurs possibles de ces paramètres, consultez la classe Button dans le *Guide de référence du langage et des composants ActionScript 3.0*.

Création d'une application avec le composant Button

La procédure suivante décrit l'ajout d'un composant Button à une application lors de la programmation. Dans cet exemple, le bouton change l'état d'un composant ColorPicker lorsque vous cliquez dessus.

Pour créer une application avec le composant Button :

1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Faites glisser un composant Button du panneau Composants vers la scène et entrez les valeurs suivantes pour celui-ci dans l'Inspecteur des propriétés :
   - Entrez le nom d'occurrence **aButton**.
   - Entrez **Show** pour le paramètre label.
3. Ajoutez un composant ColorPicker sur la scène et nommez son occurrence **aCp**.
4. Ouvrez le panneau Actions, sélectionnez l'image 1 dans le scénario principal et entrez le code ActionScript suivant :

```
aCp.visible = false;

aButton.addEventListener(MouseEvent.CLICK, clickHandler);

function clickHandler(event:MouseEvent):void {

  switch(event.currentTarget.label) {
    case "Show":
      aCp.visible = true;
      aButton.label = "Disable";
      break;
    case "Disable":
      aCp.enabled = false;
      aButton.label = "Enable";
      break;
    case "Enable":
      aCp.enabled = true;
      aButton.label = "Hide";
      break;
    case "Hide":
      aCp.visible = false;
      aButton.label = "Show";
      break;
  }
}
```

La deuxième ligne de code enregistre la fonction `clickHandler()` en tant que fonction de gestionnaire d'événements de l'événement `MouseEvent.CLICK`. L'événement se produit lorsqu'un utilisateur clique sur le bouton, contraignant la fonction `clickHandler()` à effectuer l'une des actions suivantes en fonction de la valeur du bouton :

- Afficher : rend le composant ColorPicker visible et définit l'étiquette du bouton sur Désactiver.
- Désactiver : désactive le composant ColorPicker et définit l'étiquette du bouton sur Activer.
- Activer : active le composant ColorPicker et définit l'étiquette du bouton sur Masquer.
- Masquer : rend le composant ColorPicker invisible et définit l'étiquette du bouton sur Afficher.

5. Choisissez Contrôle > Tester l'animation pour exécuter l'application.

La procédure suivante crée un bouton bascule à l'aide d'ActionScript et affiche le type d'événement dans le panneau Sortie lorsque vous cliquez sur le bouton. L'exemple crée l'occurrence de bouton en invoquant le constructeur de la classe et l'ajoute sur la scène en appelant la méthode `addChild()`.

Pour créer un composant Button à l'aide d'ActionScript :

1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Faites glisser le composant Button du panneau Composants vers le panneau Bibliothèque du document en cours.

   Cette opération permet d'ajouter le composant à la bibliothèque, mais elle ne permet pas de le rendre visible dans l'application.
3. Ouvrez le panneau Actions, sélectionnez l'image 1 dans le scénario principal et entrez le code suivant pour créer une occurrence de bouton :

```
import fl.controls.Button;

var aButton:Button = new Button();
addChild(aButton);
aButton.label = "Click me";
aButton.toggle =  true;
aButton.move(50, 50);
```

   La méthode `move()` positionne le bouton à l'emplacement 50 (coordonnée x), 50 (coordonnée y) sur la scène.

4. Ajoutez maintenant le code ActionScript suivant pour créer un écouteur d'événements et une fonction de gestionnaire d'événements :

```
aButton.addEventListener(MouseEvent.CLICK, clickHandler);

function clickHandler(event:MouseEvent):void {
  trace("Event type: " + event.type);
}
```

5. Choisissez Contrôle> Tester l'animation.

Lorsque vous cliquez sur le bouton, Flash affiche le message « Type d'événement : click » dans le panneau Sortie.

Utilisation du composant CheckBox

Un composant CheckBox est un carré pouvant être activé ou désactivé. Lorsqu'il est activé, une coche apparaît à l'intérieur. Vous pouvez ajouter une étiquette de texte au composant CheckBox et la placer à gauche, à droite, au-dessus ou au-dessous de celui-ci.

Vous pouvez utiliser des cases à cocher pour réunir un jeu de valeurs `true` ou `false` qui ne s'excluent pas réciproquement. Par exemple, une application qui recueille des informations sur le type de voiture que vous voulez acheter peut utiliser des cases à cocher vous permettant de sélectionner ses caractéristiques.

Interaction de l'utilisateur avec le composant CheckBox

Vous pouvez activer ou désactiver un composant CheckBox dans une application. Lorsqu'un composant CheckBox est activé et qu'un utilisateur clique sur son entrée ou sur son étiquette, celui-ci reçoit le focus d'entrée et son état enfoncé apparaît à l'écran. Si un utilisateur place le pointeur à l'extérieur du cadre de délimitation d'un composant CheckBox ou de son étiquette en appuyant sur le bouton de la souris, l'aspect du composant revient à son état d'origine et conserve le focus d'entrée. L'état d'un composant CheckBox ne change pas tant que le bouton de la souris n'est pas relâché sur le composant. En outre, le composant CheckBox possède deux états désactivés, sélectionné et désélectionné, qui utilisent respectivement `selectedDisabledSkin` et `disabledSkin`, et qui ne permettent pas d'interagir avec la souris ou le clavier.

Si un composant CheckBox est désactivé, il affiche son aspect désactivé, quelle que soit l'interaction de l'utilisateur. Lorsqu'il est désactivé, le composant CheckBox ne reçoit aucune information provenant du clavier ou de la souris.

Une occurrence CheckBox reçoit le focus si un utilisateur clique sur son entrée ou utilise la touche de tabulation pour l'atteindre. Lorsqu'une occurrence CheckBox a le focus, les touches suivantes permettent de le contrôler :

Touche	Description
Maj+Tab	Déplace le focus vers l'élément précédent.
Espace	Sélectionne ou désélectionne le composant et déclenche l'événement `change`.
Tab	Déplace le focus vers l'élément suivant.

Pour plus d'informations sur le contrôle du focus, consultez la section « Utilisation de FocusManager », à la page 56 et la classe FocusManager dans le *Guide de référence du langage et des composants ActionScript 3.0*.

L'aperçu en direct des occurrences CheckBox reflète les modifications apportées aux paramètres dans l'inspecteur Propriétés ou l'inspecteur des composants pendant la programmation.

Lorsque vous ajoutez le composant CheckBox à une application, vous pouvez le rendre accessible à un lecteur d'écran en ajoutant les lignes de code ActionScript suivantes :

```
import fl.accessibility.CheckBoxAccImpl;

CheckBoxAccImpl.enableAccessibility();
```

Quel que soit le nombre d'occurrences d'un composant, l'activation de son accessibilité ne se fait qu'une fois.

Paramètres du composant CheckBox

Dans l'Inspecteur des propriétés ou l'Inspecteur des composants, vous pouvez définir les paramètres de programmation suivants pour chaque occurrence de composant CheckBox : `label`, `labelPlacement` et `selected`. Chacun de ces paramètres possède une propriété ActionScript correspondante du même nom. Pour plus d'informations sur les valeurs possibles de ces paramètres, consultez la classe CheckBox dans le *Guide de référence du langage et des composants ActionScript 3.0*.

Création d'une application avec le composant CheckBox

La procédure suivante explique comment ajouter un composant CheckBox à une application lors de la programmation, à l'aide d'un extrait tiré du formulaire d'une application de calcul de crédits. Le formulaire demande si le demandeur est propriétaire et prévoit une case à cocher pour que l'utilisateur puisse répondre « oui ». S'il est propriétaire, le formulaire affiche deux boutons radio permettant à l'utilisateur d'indiquer la valeur relative de sa maison.

Pour créer une application avec le composant CheckBox :

1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Faites glisser un composant CheckBox du panneau Composants vers la scène.
3. Dans l'Inspecteur des propriétés, procédez comme suit :
   - Entrez le nom d'occurrence **homeCh**.
   - Entrez **140** pour la valeur de la largeur (L).
   - Entrez « **Etes-vous propriétaire** ? » pour le paramètre d'étiquette.
4. Faites glisser deux composants RadioButton du panneau Composants vers la scène et placez-les à droite sous le composant CheckBox. Entrez les valeurs suivantes les concernant dans l'Inspecteur des propriétés :
   - Entrez les noms d'occurrence **underRb** et **overRb**.
   - Entrez **120** pour le paramètre L (largeur) des deux composants RadioButton.
   - Entrez **Inférieur à 500 000 € ?** pour le paramètre d'étiquette de `underRb`.
   - Entrez **Supérieur à 500 000 € ?** pour le paramètre d'étiquette de `overRb`.
   - Entrez **valueGrp** pour le paramètre groupName des deux composants RadioButton.
5. Ouvrez le panneau Actions, sélectionnez l'image 1 dans le scénario principal et entrez le code ActionScript suivant :

```
homeCh.addEventListener(MouseEvent.CLICK, clickHandler);
underRb.enabled = false;
overRb.enabled = false;

function clickHandler(event:MouseEvent):void {
  underRb.enabled = event.target.selected;
  overRb.enabled = event.target.selected;
}
```

   Ce code crée un gestionnaire d'événements pour un événement `CLICK` qui active les boutons radio `underRb` et `overRb` si le composant CheckBox `homeCh` est sélectionné, et les désactive si `homeCh` n'est pas sélectionné. Pour plus d'informations, consultez la classe `MouseEvent` dans le *Guide de référence du langage et des composants ActionScript 3.0*.

6. Choisissez Contrôle> Tester l'animation.

L'exemple suivant duplique l'application précédente mais crée les composants CheckBox et RadioButton à l'aide d'ActionScript.

Pour créer un composant CheckBox à l'aide d'ActionScript :

1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Faites glisser le composant CheckBox et le composant RadioButton du panneau Composants vers le panneau Bibliothèque du document en cours. Si le panneau Bibliothèque n'est pas déjà ouvert, appuyez sur Ctrl+L ou choisissez Fenêtre > Bibliothèque pour l'ouvrir.

   Cette opération met les composants à la disposition de votre application mais ne les place pas sur la scène.
3. Ouvrez le panneau Actions, sélectionnez l'image 1 dans le scénario principal et entrez le code suivant pour créer et positionner les occurrences de composant :

```
import fl.controls.CheckBox;
import fl.controls.RadioButton;

var homeCh:CheckBox = new CheckBox();
var underRb:RadioButton = new RadioButton();
var overRb:RadioButton = new RadioButton();
addChild(homeCh);
addChild(underRb);
addChild(overRb);
underRb.groupName = "valueGrp";
overRb.groupName = "valueGrp";
homeCh.move(200, 100);
homeCh.width = 120;
homeCh.label = "Own your home?";
underRb.move(220, 130);
underRb.enabled = false;
underRb.width = 120;
underRb.label = "Under $500,000?";
overRb.move(220, 150);
overRb.enabled = false;
overRb.width = 120;
overRb.label = "Over $500,000?";
```

   Ce code utilise les constructeurs `CheckBox()` et `RadioButton()` pour créer les composants et la méthode `addChild()` pour les placer sur la scène. Il utilise la méthode `move()` pour positionner les composants sur la scène.

4. Ajoutez maintenant le code ActionScript suivant pour créer un écouteur d'événements et une fonction de gestionnaire d'événements :

```
homeCh.addEventListener(MouseEvent.CLICK, clickHandler);

function clickHandler(event:MouseEvent):void {
  underRb.enabled = event.target.selected;
  overRb.enabled = event.target.selected;
}
```

   Ce code crée un gestionnaire d'événements pour l'événement `CLICK` qui active les boutons radio `underRb` et `overRb` si le composant CheckBox `homeCh` est sélectionné, et les désactive si `homeCh` n'est pas sélectionné. Pour plus d'informations, consultez la classe `MouseEvent` dans le *Guide de référence du langage et des composants ActionScript 3.0*.

5. Choisissez Contrôle> Tester l'animation.

Utilisation du composant ColorPicker

Le composant ColorPicker permet à l'utilisateur de sélectionner une couleur dans un nuancier. Le mode par défaut du composant ColorPicker présente une seule couleur dans un bouton carré. Lorsqu'un utilisateur clique sur le bouton, la liste des couleurs disponibles apparaît dans un nuancier, ainsi qu'un champ texte qui affiche la valeur hexadécimale de la sélection de couleur active.

Vous pouvez spécifier les couleurs qui apparaissent dans le composant ColorPicker en définissant sa propriété `colors` avec les valeurs de couleur que vous souhaitez afficher.

Interaction de l'utilisateur avec le composant ColorPicker

Un composant ColorPicker permet à l'utilisateur de sélectionner une couleur et de l'appliquer à un autre objet dans l'application. Par exemple, si vous souhaitez que l'utilisateur puisse personnaliser les éléments de l'application, comme par exemple la couleur d'arrière-plan ou la couleur de texte, vous pouvez inclure un composant ColorPicker et appliquer la couleur sélectionnée par l'utilisateur.

L'utilisateur choisit une couleur en cliquant sur le nuancier dans le panneau ou en entrant sa valeur hexadécimale dans le champ texte. Une fois la couleur sélectionnée par l'utilisateur, vous pouvez utiliser la propriété `selectedColor` du composant ColorPicker pour l'appliquer au texte ou à un autre objet dans l'application.

Une occurrence ColorPicker reçoit le focus si un utilisateur place le pointeur au-dessus de celle-ci ou utilise la touche de tabulation pour l'atteindre. Lorsque le nuancier d'un composant ColorPicker est ouvert, vous pouvez utiliser les touches suivantes pour le contrôler :

Touche	Description
Origine	Déplace la sélection vers la première couleur dans le panneau Nuanciers.
Flèche vers le haut	Déplace la sélection d'une ligne vers le haut dans le panneau Nuanciers.
Flèche vers le bas	Déplace la sélection d'une ligne vers le bas dans le panneau Nuanciers.
Flèche droite	Déplace la sélection d'une couleur vers la droite dans le panneau Nuanciers.
Flèche gauche	Déplace la sélection d'une couleur vers la gauche dans le panneau Nuanciers.
Fin	Déplace la sélection vers la dernière couleur dans le panneau Nuanciers.

Paramètres du composant ColorPicker

Dans l'Inspecteur des propriétés ou l'Inspecteur des composants, vous pouvez définir les paramètres de programmation suivants pour chaque occurrence ColorPicker : `selectedColor` et `showTextField`. Chacun de ces paramètres possède une propriété ActionScript correspondante du même nom. Pour plus d'informations sur les valeurs possibles de ces paramètres, consultez la classe ColorPicker dans le *Guide de référence du langage et des composants ActionScript 3.0*.

Création d'une application avec le composant ColorPicker

L'exemple suivant ajoute un composant ColorPicker à une application lors de la programmation. Dans cet exemple, à chaque fois que vous modifiez la couleur dans le composant ColorPicker, la fonction `changeHandler()` appelle la fonction `drawBox()` pour dessiner un nouveau cadre avec la couleur sélectionnée dans le composant ColorPicker.

Pour créer une application avec le composant ColorPicker :

1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Faites glisser un composant ColorPicker du panneau Composants au centre de la scène et nommez l'occurrence **aCp**.
3. Ouvrez le panneau Actions, sélectionnez l'image 1 dans le scénario principal et entrez le code ActionScript suivant :

```
import fl.events.ColorPickerEvent;

var aBox:MovieClip = new MovieClip();
drawBox(aBox, 0xFF0000);//draw a red box
addChild(aBox);

aCp.addEventListener(ColorPickerEvent.CHANGE,changeHandler);

function changeHandler(event:ColorPickerEvent):void {
  drawBox(aBox, event.target.selectedColor);
}

function drawBox(box:MovieClip,color:uint):void {
       box.graphics.beginFill(color, 1);
       box.graphics.drawRect(100, 150, 100, 100);
       box.graphics.endFill();
}
```

4. Choisissez Contrôle> Tester l'animation.
5. Cliquez sur le composant ColorPicker et sélectionnez une couleur à affecter au cadre.

L'exemple suivant utilise le constructeur `ColorPicker()` et la méthode `addChild()` pour créer un composant ColorPicker sur la scène. Il définit la propriété `colors` sur les valeurs de couleur rouge (0xFF0000), vert (0x00FF00) et bleu (0x0000FF) pour spécifier les couleurs affichées par le composant ColorPicker. Il crée également un composant TextArea, et à chaque fois que vous sélectionnez une couleur différente dans le composant ColorPicker, l'exemple modifie la couleur du texte dans le composant TextArea afin d'établir une correspondance.

Pour créer un composant ColorPicker à l'aide d'ActionScript :

1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Faites glisser le composant ColorPicker du panneau Composants vers le panneau Bibliothèque.
3. Faites glisser le composant TextArea du panneau Composants vers le panneau Bibliothèque.
4. Ouvrez le panneau Actions, sélectionnez l'image 1 dans le scénario principal et entrez le code ActionScript suivant :

```
import fl.controls.ColorPicker;
import fl.controls.TextArea;
import fl.events.ColorPickerEvent;

var aCp:ColorPicker = new ColorPicker();
var aTa:TextArea = new TextArea();
var aTf:TextFormat = new TextFormat();

aCp.move(100, 100);
aCp.colors = [0xff0000, 0x00ff00, 0x0000ff];
aCp.addEventListener(ColorPickerEvent.CHANGE, changeHandler);

aTa.text = "Lorem ipsum dolor sit amet, consectetuer adipiscing elit.
  Vivamus quis nisl vel tortor nonummy vulputate. Quisque sit amet eros
  sed purus euismod tempor. Morbi tempor. Class aptent taciti sociosqu
  ad litora torquent per conubia nostra, per inceptos hymenaeos.
  Curabitur diam. Suspendisse at purus in ipsum volutpat viverra. Nulla
  pellentesque libero id libero.";
aTa.setSize(200, 200);
aTa.move(200,100);

addChild(aCp);
addChild(aTa);

function changeHandler(event:ColorPickerEvent):void {
  if(TextFormat(aTa.getStyle("textFormat"))){
    aTf = TextFormat(aTa.getStyle("textFormat"));
  }
  aTf.color = event.target.selectedColor;
  aTa.setStyle("textFormat", aTf);
}
```

5. Choisissez Contrôle> Tester l'animation.

Utilisation du composant ComboBox

Un composant ComboBox permet à l'utilisateur d'effectuer une sélection unique dans une liste déroulante. Ce composant ComboBox peut être statique ou modifiable. Un composant ComboBox modifiable permet à l'utilisateur d'entrer du texte directement dans le champ texte en haut de la liste. Si la liste déroulante atteint le bas du document, elle se déroule vers le haut et non vers le bas. Le composant ComboBox est constitué de trois sous-composants : BaseButton, TextInput et List.

Dans un composant ComboBox modifiable, le bouton est la seule zone active, pas le champ de texte. Dans le cas d'un composant ComboBox statique, le bouton et le champ de texte constituent la zone active. La zone active répond par l'ouverture ou la fermeture de la liste déroulante.

Lorsque l'utilisateur effectue une sélection dans la liste, à l'aide de la souris ou du clavier, l'étiquette de la sélection est copiée dans le champ texte en haut du composant ComboBox.

Interaction de l'utilisateur avec le composant ComboBox

Vous pouvez utiliser un composant ComboBox dans tout formulaire ou toute application qui requiert un choix unique dans une liste. Par exemple, vous pouvez fournir une liste déroulante de pays dans un formulaire où les clients doivent saisir leur adresse. Les composants ComboBox modifiables conviennent pour des scénarios plus complexes. Par exemple, dans une application d'itinéraire routier, utilisez un composant ComboBox modifiable pour que l'utilisateur y saisisse ses adresses de départ et d'arrivée. La liste déroulante pourrait alors contenir des adresses déjà saisies.

Si le composant ComboBox est modifiable, ce qui signifie que la propriété `editable` est définie sur `true`, les touches suivantes suppriment le focus de la zone de saisie et conservent la valeur précédente. La touche Entrée constitue une exception car elle applique la nouvelle valeur d'abord, si l'utilisateur a entré du texte.

Touche	Description
Maj + Tab	Place le focus sur l'élément précédent. Si un nouvel élément est sélectionné, un événement `change` est distribué.
Tab	Place le focus sur l'élément suivant. Si un nouvel élément est sélectionné, un événement `change` est distribué.
Flèche vers le bas	Déplace la sélection d'un élément vers le bas.
Fin	Déplace la sélection en bas de la liste.

Touche	Description
Echap	Ferme la liste déroulante et place à nouveau le focus sur le composant ComboBox.
Entrée	Ferme la liste déroulante et place à nouveau le focus sur le composant ComboBox. Lorsque le composant ComboBox est modifiable, et si l'utilisateur entre du texte, la touche Entrée définit la valeur sur le texte entré.
Origine	Déplace la sélection en haut de la liste.
Pg. Préc.	Déplace la sélection d'une page vers le haut.
Pg. Suiv.	Déplace la sélection d'une page vers le bas.

Lorsque vous ajoutez le composant ComboBox à une application, vous pouvez le rendre accessible à un lecteur d'écran en ajoutant les lignes de code ActionScript suivantes :

```
import fl.accessibility.ComboBoxAccImpl;

ComboBoxAccImpl.enableAccessibility();
```

Quel que soit le nombre d'occurrences d'un composant, l'activation de son accessibilité ne se fait qu'une fois.

Paramètres du composant ComboBox

Dans l'Inspecteur des propriétés ou l'Inspecteur des composants, vous pouvez définir les paramètres suivants pour chaque occurrence ComboBox : `dataProvider`, `editable`, `prompt` et `rowCount`. Chacun de ces paramètres possède une propriété ActionScript correspondante du même nom. Pour plus d'informations sur les valeurs possibles de ces paramètres, consultez la classe ComboBox dans le *Guide de référence du langage et des composants ActionScript 3.0*. Pour plus d'informations sur l'utilisation du paramètre dataProvider, consultez la section « Utilisation du paramètre dataProvider », à la page 60.

Création d'une application avec le composant ComboBox

La procédure suivante décrit l'ajout d'un composant ComboBox à une application au cours de la programmation. Le composant ComboBox est modifiable et si vous tapez **Add** dans le champ texte, l'exemple ajoute un élément dans la liste déroulante.

Pour créer une application avec le composant ComboBox :

1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Faites glisser un composant ComboBox sur la scène et nommez son occurrence **aCb**. Dans l'onglet Paramètres, définissez le paramètre `editable` sur `true`.
3. Ouvrez le panneau Actions, sélectionnez l'image 1 dans le scénario principal et entrez le code suivant :

```
import fl.data.DataProvider;
import fl.events.ComponentEvent;

var items:Array = [
{label:"screen1", data:"screenData1"},
{label:"screen2", data:"screenData2"},
{label:"screen3", data:"screenData3"},
{label:"screen4", data:"screenData4"},
{label:"screen5", data:"screenData5"},
];
aCb.dataProvider = new DataProvider(items);

aCb.addEventListener(ComponentEvent.ENTER, onAddItem);

function onAddItem(event:ComponentEvent):void {
  var newRow:int = 0;
  if (event.target.text == "Add") {
    newRow = event.target.length + 1;
    event.target.addItemAt({label:"screen" + newRow, data:"screenData"
  + newRow},
       event.target.length);
  }
}
```

4. Choisissez Contrôle> Tester l'animation.

L'exemple suivant crée un composant ComboBox à l'aide d'ActionScript et lui affecte une liste des universités de la région de San Francisco, en Californie. Il définit la propriété `width` du composant ComboBox en fonction de la largeur du texte de l'invite et définit la propriété `dropdownWidth` de manière à ce qu'elle soit légèrement plus étendue pour pouvoir prendre en charge le nom d'université le plus long.

Cet exemple crée la liste des universités dans une occurrence de tableau et utilise la propriété `label` pour stocker leurs noms et la propriété `data` pour stocker les URL de chacun de leurs sites Web. Il affecte le tableau au composant ComboBox en définissant sa propriété `dataProvider`.

Lorsqu'un utilisateur sélectionne une université dans la liste, il déclenche un événement Event.`CHANGE` et un appel de la fonction `changeHandler()`, qui charge la propriété `data` dans une requête d'URL pour accéder à son site Web.

Veuillez noter que la dernière ligne définit la propriété `selectedIndex` de l'occurrence ComboBox sur -1 pour afficher de nouveau l'invite lorsque la liste se ferme. Sinon, l'invite serait remplacée par le nom de l'université sélectionné.

Pour créer un composant ComboBox à l'aide d'ActionScript :

1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Faites glisser le composant ComboBox du panneau Composants vers le panneau Bibliothèque.
3. Ouvrez le panneau Actions, sélectionnez l'image 1 dans le scénario principal et entrez le code ActionScript suivant :

```
import fl.controls.ComboBox;
import fl.data.DataProvider;
import flash.net.navigateToURL;

var sfUniversities:Array = new Array(
       {label:"University of California, Berkeley",
         data:"http://www.berkeley.edu/"},
       {label:"University of San Francisco",
         data:"http://www.usfca.edu/"},
       {label:"San Francisco State University",
         data:"http://www.sfsu.edu/"},
       {label:"California State University, East Bay",
         data:"http://www.csuhayward.edu/"},
       {label:"Stanford University", data:"http://www.stanford.edu/"},
       {label:"University of Santa Clara", data:"http://www.scu.edu/"},
       {label:"San Jose State University", data:"http://www.sjsu.edu/"}
);

var aCb:ComboBox = new ComboBox();
aCb.dropdownWidth = 210;
aCb.width = 200;
aCb.move(150, 50);
aCb.prompt = "San Francisco Area Universities";
aCb.dataProvider = new DataProvider(sfUniversities);
aCb.addEventListener(Event.CHANGE, changeHandler);

addChild(aCb);

function changeHandler(event:Event):void {
       var request:URLRequest = new URLRequest();
       request.url = ComboBox(event.target).selectedItem.data;
       navigateToURL(request);
       aCb.selectedIndex = -1;
}
```

4. Choisissez Contrôle> Tester l'animation.

   Vous pouvez implémenter et exécuter cet exemple dans l'environnement de programmation Flash mais vous recevrez des messages d'avertissement si vous essayez d'accéder aux sites Web des universités en cliquant sur les éléments du composant ComboBox. Pour accéder au composant ComboBox parfaitement fonctionnel sur Internet, rendez-vous à l'adresse suivante :

   http://www.helpexamples.com/peter/bayAreaColleges/bayAreaColleges.html

Utilisation du composant DataGrid

Le composant DataGrid vous permet d'afficher des données dans une grille de lignes et de colonnes, de tracer les données d'un tableau ou d'un fichier XML externe que vous pouvez analyser dans un tableau pour le fournisseur de données. Le composant DataGrid inclut la fonction de défilement horizontal et vertical, une prise en charge des événements (notamment dans les cellules modifiables) et des fonctionnalités de tri.

Vous pouvez redimensionner et personnaliser des caractéristiques telles que la police, la couleur et les bordures des colonnes d'une grille. Vous pouvez utiliser un clip personnalisé en tant qu'objet CellRenderer pour toute colonne d'une grille. (Un objet CellRenderer affiche le contenu d'une cellule.) Vous pouvez désactiver les barres de défilement et utiliser les méthodes DataGrid pour créer un affichage de style mode Page. Pour plus d'informations sur la personnalisation, consultez la classe DataGridColumn dans le *Guide de référence du langage et des composants ActionScript 3.0*.

Interaction de l'utilisateur avec le composant DataGrid

La souris et le clavier permettent d'interagir avec un composant DataGrid.

Si la propriété `sortableColumns` et la propriété `sortable` de la colonne sont toutes deux définies sur `true`, l'utilisateur peut trier les données en fonction des valeurs de la colonne en cliquant sur l'en-tête de cette dernière. Vous pouvez désactiver le tri d'une colonne individuelle en définissant sa propriété `sortable` sur `false`.

Si la propriété `resizableColumns` est définie sur `true`, vous pouvez redimensionner les colonnes en faisant glisser les séparateurs de colonne dans la ligne d'en-tête.

Si l'utilisateur clique sur une cellule modifiable, cette cellule reçoit le focus. S'il clique sur une cellule non modifiable, cela n'a aucun impact sur le focus. Une cellule est modifiable si ses propriétés `DataGrid.editable` et `DataGridColumn.editable` ont la valeur `true`.

Pour plus d'informations, consultez les classes DataGrid et DataGridColumn dans le *Guide de référence du langage et des composants ActionScript 3.0*.

Lorsqu'une occurrence de DataGrid a le focus (l'utilisateur a cliqué ou utilisé la tabulation), les touches suivantes permettent de la contrôler :

Touche	Description
Flèche vers le bas	Lorsque la cellule fait l'objet d'une modification, le point d'insertion se positionne à la fin du texte de la cellule. Si la cellule n'est pas modifiable, la flèche vers le bas gère la sélection de la même façon que le composant List.
Flèche vers le haut	Lorsque la cellule fait l'objet d'une modification, le point d'insertion se positionne au début du texte de la cellule. Si la cellule n'est pas modifiable, la flèche vers le haut gère la sélection de la même façon que le composant List.
Maj+Flèche vers le haut/bas	Si le composant DataGrid n'est pas modifiable et si `allowMultipleSelection` a la valeur `true`, des lignes contiguës sont sélectionnées. L'inversion de la direction avec la flèche opposée désélectionne les lignes sélectionnées jusqu'à ce que vous transmettiez la ligne de départ à partir de laquelle les lignes dans cette direction sont sélectionnées.
Maj+Clic	Si `allowMultipleSelection` a la valeur `true`, toutes les lignes entre la ligne sélectionnée et la position actuelle du signe insertion (cellule mise en surbrillance) sont sélectionnées.
Ctrl+Clic	Si `allowMultipleSelection` a la valeur `true`, les lignes supplémentaires, qui n'ont pas besoin d'être contiguës, sont sélectionnées.
Flèche droite	Lorsque la cellule fait l'objet d'une modification, le point d'insertion se déplace d'un caractère vers la droite. Si la cellule n'est pas modifiable, cette action n'a aucune incidence.
Flèche gauche	Lorsque la cellule fait l'objet d'une modification, le point d'insertion se déplace d'un caractère vers la gauche. Si la cellule n'est pas modifiable, cette action n'a aucune incidence.
Origine	Sélectionne la première ligne du composant DataGrid.
Fin	Sélectionne la dernière ligne du composant DataGrid.
Pg. Préc.	Sélectionne la première ligne d'une page du composant DataGrid. Une page comprend le nombre de lignes que le composant DataGrid peut afficher sans défilement.

Touche	Description
Pg. Suiv.	Sélectionne la dernière ligne d'une page du composant DataGrid. Une page comprend le nombre de lignes que le composant DataGrid peut afficher sans défilement.
Retour/Entrée/ Maj+Entrée	Lorsque la cellule est modifiable, la modification est validée et le point d'insertion se place dans la cellule de la ligne suivante (vers le haut ou vers le bas, en fonction du sens de basculement) dans la même colonne.
Maj+Tab/Tab	Si le composant DataGrid est modifiable, le focus est placé sur l'élément précédent/suivant jusqu'à ce que la fin de la colonne soit atteinte, puis sur la ligne précédente/suivante jusqu'à ce que la première ou la dernière cellule soit atteinte. Si la première cellule est sélectionnée, Maj+Tab place le focus sur le contrôle précédent. Si la dernière cellule est sélectionnée, Tab place le focus sur le contrôle suivant. Si le composant DataGrid n'est pas modifiable, le focus est placé sur le contrôle précédent/suivant.

Le composant DataGrid peut servir de base à de nombreux types d'applications de données. Vous pouvez aisément afficher une vue tabulaire de données, puis utiliser les fonctionnalités du composant CellRenderer pour créer des éléments d'interface utilisateur plus élaborés et modifiables. Voici des exemples concrets d'utilisation du composant DataGrid :

- Client de messagerie Web
- Pages de résultats de recherches
- Tableurs (applications de calculs de crédits et de formulaires de déclaration d'impôt)

Lors du développement d'une application avec le composant DataGrid, il est important de bien comprendre la conception du composant List car la classe DataGrid étend la classe SelectableList. Pour plus d'informations sur la classe SelectableList et sur le composant List, consultez les classes SelectableList et List dans le *Guide de référence du langage et des composants ActionScript 3.0*.

Lorsque vous ajoutez un composant DataGrid à votre application, vous pouvez le rendre accessible à un lecteur d'écran en ajoutant les lignes de code ActionScript suivantes :

```
import fl.accessibility.DataGridAccImpl;
DataGridAccImpl.enableAccessibility();
```

Quel que soit le nombre d'occurrences d'un composant, l'activation de son accessibilité ne se fait qu'une fois. Pour plus d'informations, consultez le Chapitre 18, « Création de contenu accessible » du guide *Utilisation de Flash*.

Paramètres du composant DataGrid

Dans l'Inspecteur des propriétés ou l'Inspecteur des composants, vous pouvez définir les paramètres de programmation suivants pour chaque occurrence du composant DataGrid : `allowMultipleSelection`, `editable`, `headerHeight`, `horizontalLineScrollSize`, `horizontalPageScrollSize`, `horizontalScrolllPolicy`, `resizableColumns`, `rowHeight`, `showHeaders`, `verticalLineScrollSize`, `verticalPageScrollSize` et `verticalScrollPolicy`. Chacun de ces paramètres possède une propriété ActionScript correspondante du même nom. Pour plus d'informations sur les valeurs possibles de ces paramètres, consultez la classe DataGrid dans le *Guide de référence du langage et des composants ActionScript 3.0*.

Création d'une application avec le composant DataGrid

Pour créer une application avec le composant DataGrid, commencez par déterminer d'où proviennent vos données. Généralement, les données proviennent d'un tableau que vous pouvez intégrer à la grille en définissant la propriété `dataProvider`. Vous pouvez également utiliser les méthodes des classes DataGrid et DataGridColumn pour ajouter des données à la grille.

L'exemple suivant crée un composant DataGrid pour afficher la liste d'une équipe de softball. Il définit la liste dans un tableau (`aRoster`) et l'affecte à la propriété `dataProvider` du composant DataGrid.

Pour utiliser un fournisseur de données local avec un composant DataGrid :

1. Dans Flash, sélectionnez Fichier> Nouveau, puis Fichier Flash (ActionScript 3.0).
2. Faites glisser le composant DataGrid du panneau Composants vers la scène.
3. Dans l'Inspecteur des propriétés, entrez le nom d'occurrence **aDg**.

4. Ouvrez le panneau Actions, sélectionnez l'image 1 dans le scénario principal et entrez le code ActionScript suivant :

```
import fl.data.DataProvider;

bldRosterGrid(aDg);
var aRoster:Array = new Array();
aRoster = [
     {Name:"Wilma Carter", Bats:"R", Throws:"R", Year:"So", Home:
  "Redlands, CA"},
     {Name:"Sue Pennypacker", Bats:"L", Throws:"R", Year:"Fr", Home:
  "Athens, GA"},
     {Name:"Jill Smithfield", Bats:"R", Throws:"L", Year:"Sr", Home:
  "Spokane, WA"},
     {Name:"Shirley Goth", Bats:"R", Throws:"R", Year:"Sr", Home:
  "Carson, NV"},
     {Name:"Jennifer Dunbar", Bats:"R", Throws:"R", Year:"Fr", Home:
  "Seaside, CA"},
     {Name:"Patty Crawford", Bats:"L", Throws:"L", Year:"Jr", Home:
  "Whittier, CA"},
     {Name:"Angelina Davis", Bats:"R", Throws:"R", Year:"So", Home:
  "Odessa, TX"},
     {Name:"Maria Santiago", Bats:"L", Throws:"L", Year:"Sr", Home:
  "Tacoma, WA"},
     {Name:"Debbie Ferguson", Bats:"R", Throws:"R", Year: "Jr", Home:
  "Bend, OR"},
     {Name:"Karen Bronson", Bats:"R", Throws:"R", Year: "Sr", Home:
  "Billings, MO"},
     {Name:"Sylvia Munson", Bats:"R", Throws:"R", Year: "Jr", Home:
  "Pasadena, CA"},
     {Name:"Carla Gomez", Bats:"R", Throws:"L", Year: "Sr", Home:
  "Corona, CA"},
     {Name:"Betty Kay", Bats:"R", Throws:"R", Year: "Fr", Home:
  "Palo Alto, CA"},
];
aDg.dataProvider = new DataProvider(aRoster);
aDg.rowCount = aDg.length;

function bldRosterGrid(dg:DataGrid){
  dg.setSize(400, 300);
  dg.columns = ["Name", "Bats", "Throws", "Year", "Home"];
  dg.columns[0].width = 120;
  dg.columns[1].width = 50;
  dg.columns[2].width = 50;
  dg.columns[3].width = 40;
  dg.columns[4].width = 120;
  dg.move(50,50);
};
```

La fonction `bldRosterGrid()` définit la taille du composant DataGrid, ainsi que l'ordre des colonnes et leurs tailles.

5. Choisissez Contrôle> Tester l'animation.

   Notez que vous pouvez cliquer sur un en-tête de colonne quelconque pour trier le contenu du composant DataGrid par ordre décroissant en fonction des valeurs de cette colonne.

L'exemple suivant utilise la méthode `addColumn()` pour ajouter des occurrences `DataGridColumn` à un composant DataGrid. Les colonnes représentent les noms des joueurs et leurs scores. Cet exemple définit également la propriété `sortOptions` pour spécifier les options de tri de chaque colonne : `Array.CASEINSENSITIVE` pour la colonne Nom et `Array.NUMERIC` pour la colonne Score. Il dimensionne le composant DataGrid de manière appropriée en définissant la longueur en fonction du nombre de lignes et la largeur sur 200.

Pour spécifier des colonnes et ajouter un tri pour un composant DataGrid dans une application :

1. Dans Flash, sélectionnez Fichier> Nouveau, puis Fichier Flash (ActionScript 3.0).
2. Faites glisser le composant DataGrid du panneau Composants vers la scène.
3. Dans l'Inspecteur des propriétés, entrez le nom d'occurrence **aDg**.
4. Ouvrez le panneau Actions, sélectionnez l'image 1 dans le scénario principal et entrez le code ActionScript suivant :

```
import fl.controls.dataGridClasses.DataGridColumn;
import fl.events.DataGridEvent;
import fl.data.DataProvider;
// Create columns to enable sorting of data.
var nameDGC:DataGridColumn = new DataGridColumn("name");
nameDGC.sortOptions = Array.CASEINSENSITIVE;
var scoreDGC:DataGridColumn = new DataGridColumn("score");
scoreDGC.sortOptions = Array.NUMERIC;
aDg.addColumn(nameDGC);
aDg.addColumn(scoreDGC);
var aDP_array:Array = new Array({name:"clark", score:3135},
  {name:"Bruce", score:403}, {name:"Peter", score:25})
aDg.dataProvider = new DataProvider(aDP_array);
aDg.rowCount = aDg.length;
aDg.width = 200;
```

5. Choisissez Contrôle> Tester l'animation.

L'exemple suivant crée un composant DataGrid à l'aide d'ActionScript et le remplit avec un tableau de noms de joueurs et de scores.

Pour créer une occurrence du composant DataGrid à l'aide d'ActionScript :

1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Faites glisser le composant DataGrid du panneau Composants vers le panneau Bibliothèque du document en cours.

   Cette opération permet d'ajouter le composant à la bibliothèque, mais elle ne permet pas de le rendre visible dans l'application.
3. Ouvrez le panneau Actions, sélectionnez l'image 1 dans le scénario principal et entrez le code ActionScript suivant :

```
import fl.controls.DataGrid;
import fl.data.DataProvider;

var aDg:DataGrid = new DataGrid();
addChild(aDg);
aDg.columns = [ "Name", "Score" ];
aDg.setSize(140, 100);
aDg.move(10, 40);
```

   Ce code crée l'occurrence DataGrid, puis dimensionne et positionne la grille.
4. Créez maintenant un tableau, ajoutez-y des données et identifiez-le comme fournisseur de données du composant DataGrid :

```
var aDP_array:Array = new Array();
aDP_array.push({Name:"Clark", Score:3135});
aDP_array.push({Name:"Bruce", Score:403});
aDP_array.push({Name:"Peter", Score:25});
aDg.dataProvider = new DataProvider(aDP_array);
aDg.rowCount = aDg.length;
```

5. Choisissez Contrôle> Tester l'animation.

L'exemple suivant utilise la classe DataGridColumn pour créer les colonnes du composant DataGrid. Il remplit le composant DataGrid en transmettant un objet XML en tant que paramètre `value` du constructeur `DataProvider()`.

Pour charger un composant DataGrid avec un fichier XML :

1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Dans le panneau Composants, double-cliquez sur le composant DataGrid pour l'ajouter sur la scène.
3. Dans l'Inspecteur des propriétés, entrez le nom d'occurrence **aDg**.

4. Ouvrez le panneau Actions, sélectionnez l'image 1 dans le scénario principal et entrez le code ActionScript suivant :

```
import fl.controls.dataGridClasses.DataGridColumn;
import fl.data.DataProvider;

var teamXML:XML = <team>
    <player name="Player A" avg="0.293" />
    <player name="Player B" avg="0.214" />
    <player name="Player C" avg="0.317" />
</team>;

var nameCol:DataGridColumn = new DataGridColumn("name");
nameCol.headerText = "Name";
nameCol.width = 120;
var avgCol:DataGridColumn = new DataGridColumn("avg");
avgCol.headerText = "Average";
avgCol.width = 60;

var myDP:DataProvider = new DataProvider(teamXML);

aDg.columns = [nameCol, avgCol];
aDg.width = 200;
aDg.dataProvider = myDP;
aDg.rowCount = aDg.length;
```

5. Choisissez Contrôle> Tester l'animation.

Utilisation du composant Label

Le composant Label affiche une ligne unique de texte, généralement pour identifier un autre élément ou l'activité d'une page Web. Vous pouvez spécifier qu'une étiquette soit mise au format HTML pour pouvoir tirer parti de ses balises de formatage de texte. Vous pouvez également contrôler son alignement et sa taille. Les étiquettes n'ont pas de bordures, ne peuvent pas recevoir le focus et ne diffusent pas d'événements.

L'aperçu en direct des occurrences Label reflète les modifications apportées aux paramètres dans l'inspecteur des propriétés ou des composants pendant la programmation. Le composant Label n'ayant pas de bordures, la définition de son paramètre de texte constitue le seul moyen de visualiser son aperçu en direct.

Interaction de l'utilisateur avec le composant Label

Utilisez des composants Label afin de créer des étiquettes de texte pour les autres composants de formulaire, tels que l'étiquette « Nom : » placée à gauche d'un champ TextInput où l'on saisirait un nom d'utilisateur. Il s'avère judicieux d'utiliser un composant Label au lieu d'un champ de texte ordinaire afin d'employer des styles qui vous permettront de conserver un aspect cohérent dans l'ensemble du document.

Si vous souhaitez faire pivoter un composant Label, vous devez intégrer les polices ; sinon, elles ne s'afficheront pas lors du test de l'animation.

Paramètres du composant Label

Dans l'Inspecteur des propriétés ou l'Inspecteur des composants, vous pouvez définir les paramètres de programmation suivants pour chaque occurrence du composant Label : `autoSize`, `condenseWhite`, `selectable`, `text` et `wordWrap`. Chacun de ces paramètres possède une propriété ActionScript correspondante du même nom. Pour plus d'informations sur les valeurs possibles de ces paramètres, consultez la classe Label dans le *Guide de référence du langage et des composants ActionScript 3.0*.

Création d'une application avec le composant Label

La procédure suivante décrit l'ajout d'un composant Label à une application au cours de la programmation. Dans cet exemple, l'étiquette affiche simplement le texte « Expiration Date ».

Pour créer une application avec le composant Label :

1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Faites glisser un composant Label du panneau Composants vers la scène et spécifiez les valeurs suivantes pour celui-ci dans l'Inspecteur des propriétés :
   - Entrez le nom d'occurrence **aLabel**.
   - Entrez **80** pour la valeur W.
   - Entrez **100** pour la valeur X.
   - Entrez **100** pour la valeur Y.
   - Entrez **Expiration Date** pour le paramètre `text`.

3. Faites glisser un composant TextArea sur la scène et spécifiez les valeurs suivantes pour celui-ci dans l'Inspecteur des propriétés :
   - Entrez le nom d'occurrence **aTa**.
   - Entrez **22** pour la valeur H.
   - Entrez **200** pour la valeur X.
   - Entrez **100** pour la valeur Y.
4. Ouvrez le panneau Actions, sélectionnez l'image 1 dans le scénario principal et entrez le code ActionScript suivant :

```
var today:Date = new Date();
var expDate:Date = addDays(today, 14);
aTa.text = expDate.toDateString();

function addDays(date:Date, days:Number):Date {
    return addHours(date, days*24);
}

function addHours(date:Date, hrs:Number):Date {
    return addMinutes(date, hrs*60);
}

function addMinutes(date:Date, mins:Number):Date {
    return addSeconds(date, mins*60);
}

function addSeconds(date:Date, secs:Number):Date {
        var mSecs:Number = secs * 1000;
        var sum:Number = mSecs + date.getTime();
        return new Date(sum);
}
```

5. Choisissez Contrôle> Tester l'animation.

L'exemple suivant crée un paramètre du composant Label à l'aide d'ActionScript. Il utilise une étiquette pour identifier la fonction d'un composant ColorPicker et la propriété `htmlText` pour appliquer le formatage au texte de l'étiquette.

Pour créer une occurrence du composant Label à l'aide d'ActionScript :

1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Faites glisser le composant Label du panneau Composants vers le panneau Bibliothèque du document en cours.
3. Faites glisser le composant ColorPicker du panneau Composants vers le panneau Bibliothèque du document en cours.

4. Ouvrez le panneau Actions, sélectionnez l'image 1 dans le scénario principal et entrez le code ActionScript suivant :

```
import fl.controls.Label;
import fl.controls.ColorPicker;

var aLabel:Label = new Label();
var aCp:ColorPicker = new ColorPicker();

addChild(aLabel);
addChild(aCp);

aLabel.htmlText = '<font face="Arial" color="#FF0000" size="14">Fill:</
  font>';
aLabel.x = 200;
aLabel.y = 150;
aLabel.width = 25;
aLabel.height = 22;

aCp.x = 230;
aCp.y = 150;
```

5. Choisissez Contrôle> Tester l'animation.

Utilisation du composant List

Le composant List est une zone de liste défilante à sélection unique ou multiple. Les listes peuvent également contenir des graphiques et d'autres composants. L'ajout des éléments affichés dans la zone de liste s'effectue via la boîte de dialogue Valeurs qui s'ouvre lorsque vous cliquez dans les champs de paramètres des étiquettes ou des données. Vous pouvez également utiliser les méthodes `List.addItem()` et `List.addItemAt()` pour ajouter des éléments à la liste.

Le composant List utilise un index basé sur zéro, où l'élément possédant l'index 0 est le premier affiché. Lorsque vous ajoutez, supprimez ou remplacez les éléments d'une liste au moyen des méthodes et des propriétés de la classe List, il peut s'avérer nécessaire d'indiquer leur index.

Interaction de l'utilisateur avec le composant List

Vous pouvez définir une liste dans laquelle les utilisateurs pourront effectuer un ou plusieurs choix. Par exemple, un utilisateur qui visite un site de commerce électronique doit pouvoir choisir l'article à acheter. Imaginons que 30 articles lui soient proposés. Il fait défiler la liste et en choisit un en cliquant sur son entrée.

Vous pouvez également concevoir une liste qui affiche des lignes de clips personnalisés pour mieux renseigner les utilisateurs. Par exemple, dans une application de courrier électronique, chaque boîte de réception peut être un composant List et chaque ligne peut être accompagnée d'icônes indiquant la priorité et l'état des messages.

La liste reçoit le focus lorsque l'utilisateur clique sur son entrée ou appuie sur la touche de tabulation pour y accéder. Les touches suivantes permettent de la contrôler :

Touche	Description
Touches alphanumériques	Passe à l'élément suivant dont `Key.getAscii()` est le premier caractère de l'étiquette.
Contrôle	Bouton bascule autorisant plusieurs sélections et désélections non contiguës.
Flèche vers le bas	Déplace la sélection d'un élément vers le bas.
Origine	Déplace la sélection jusqu'au sommet de la liste.
Pg. Suiv.	Déplace la sélection sur la page suivante.
Pg. Préc.	Déplace la sélection sur la page précédente.
Maj	Permet une sélection contiguë.
Flèche vers le haut	Déplace la sélection d'un élément vers le haut.

REMARQUE

La taille de page utilisée par les touches Page précédente et Page suivante correspond au nombre d'éléments contenus dans l'affichage, moins un. Par exemple, le passage à la page suivante dans une liste déroulante à dix lignes affiche les éléments 0-9, 9-18, 18-27, etc., avec un élément commun par page.
Notez également que les tailles de défilement sont en pixels et non pas en lignes.

Pour plus d'informations sur le contrôle du focus, consultez l'interface IFocusManager et la classe FocusManager dans le *Guide de référence du langage et des composants ActionScript 3.0* et « Utilisation de FocusManager », à la page 56.

L'aperçu en direct de chaque occurrence de List sur la scène reflète les modifications apportées aux paramètres dans l'inspecteur Propriétés ou des composants au cours de la programmation.

Lorsque vous ajoutez le composant List à une application, vous pouvez le rendre accessible à un lecteur d'écran en ajoutant les lignes de code ActionScript suivantes :

```
import fl.accessibility.ListAccImpl;

ListAccImpl.enableAccessibility();
```

Quel que soit le nombre d'occurrences d'un composant, l'activation de son accessibilité ne se fait qu'une fois. Pour plus d'informations, consultez le Chapitre 18, « Création de contenu accessible» du guide *Utilisation de Flash*.

Paramètres du composant List

Dans l'Inspecteur des propriétés ou l'Inspecteur des composants, vous pouvez définir les paramètres suivants pour chaque occurrence du composant List : `allowMultipleSelection`, `dataProvider`, `horizontalLineScrollSize`, `horizontalPageScrollSize`, `horizontalScrollPolicy`, `multipleSelection`, `verticalLineScrollSize`, `verticalPageScrollSize` et `verticalScrollPolicy`. Chacun de ces paramètres possède une propriété ActionScript correspondante du même nom. Pour plus d'informations sur les valeurs possibles de ces paramètres, consultez la classe List dans le *Guide de référence du langage et des composants ActionScript 3.0*. Pour plus d'informations sur l'utilisation du paramètre dataProvider, consultez la section « Utilisation du paramètre dataProvider », à la page 60.

Création d'une application avec le composant List

L'exemple suivant décrit l'ajout d'un composant List à une application au cours de la programmation. Dans cet exemple, le composant List comprend des étiquettes qui identifient les modèles de voiture et des champs de données contenant leurs prix.

Pour ajouter un composant List simple à une application :

1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Faites glisser un composant List du panneau Composants sur la scène.
3. Dans l'Inspecteur des propriétés, procédez comme suit :
   - Entrez le nom d'occurrence **aList**.
   - Affectez la valeur **200** au paramètre L (largeur).
4. Utilisez l'outil Texte pour créer un champ texte sous `aList` et attribuez-lui le nom d'occurrence **aTf**.
5. Ouvrez le panneau Actions, sélectionnez l'image 1 dans le scénario principal et entrez le code ActionScript suivant :

```
import fl.controls.List;
import flash.text.TextField;

aTf.type = TextFieldType.DYNAMIC;
aTf.border = false;

// Create these items in the Property inspector when data and label
// parameters are available.
aList.addItem({label:"1956 Chevy (Cherry Red)", data:35000});
aList.addItem({label:"1966 Mustang (Classic)", data:27000});
aList.addItem({label:"1976 Volvo (Xcllnt Cond)", data:17000});
aList.allowMultipleSelection = true;
```

```
aList.addEventListener(Event.CHANGE, showData);

function showData(event:Event) {
  aTf.text = "This car is priced at: $" +
  event.target.selectedItem.data;
}
```

Ce code utilise la méthode `addItem()` pour affecter trois éléments à l'occurrence `aList`, en leur attribuant à chacun d'entre eux une valeur `label`, qui apparaît dans la liste, et une valeur `data`. Lorsque vous sélectionnez un élément dans la liste, l'écouteur d'événements appelle la fonction`showData()` qui affiche la valeur `data` de l'élément sélectionné.

6. Choisissez Contrôle > Tester l'animation pour compiler et exécuter cette application.

L'exemple suivant crée également une liste des modèles de voiture et de leurs prix. Toutefois, il utilise un fournisseur de données pour remplir le composant List plutôt que la méthode `addItem()`.

Pour alimenter une occurrence List avec un fournisseur de données :

1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Faites glisser un composant List du panneau Composants sur la scène.
3. Dans l'Inspecteur des propriétés, procédez comme suit :
   - Entrez le nom d'occurrence **aList**.
   - Affectez la valeur **200** au paramètre L (largeur).
4. Utilisez l'outil Texte pour créer un champ texte sous `aList` et attribuez-lui le nom d'occurrence **aTf**.
5. Ouvrez le panneau Actions, sélectionnez l'image 1 dans le scénario principal et entrez le code ActionScript suivant :

```
import fl.controls.List;
import fl.data.DataProvider;
import flash.text.TextField;

aTf.type = TextFieldType.DYNAMIC;
aTf.border = false;

var cars:Array = [
{label:"1956 Chevy (Cherry Red)", data:35000},
{label:"1966 Mustang (Classic)", data:27000},
{label:"1976 Volvo (Xcllnt Cond)", data:17000},
];
aList.dataProvider = new DataProvider(cars);
aList.allowMultipleSelection = true;

aList.addEventListener(Event.CHANGE, showData);
```

```
function showData(event:Event) {
  aTf.text = "This car is priced at: $" +
  event.target.selectedItem.data;
}
```

6. Choisissez Contrôle > Tester l'animation pour visualiser la liste et ses éléments.

L'exemple suivant crée une liste de noms de couleur. Lorsque vous sélectionnez une couleur, il l'applique à un clip.

Pour contrôler une occurrence de clip à l'aide d'un composant List :

1. Créez un document de fichier Flash (ActionScript 3.0).
2. Faites glisser le composant List du panneau Composants vers la scène et spécifiez les valeurs suivantes pour celui-ci dans l'Inspecteur des propriétés :
   - Entrez le nom d'occurrence **aList**.
   - Entrez **60** pour la valeur H.
   - Entrez **100** pour la valeur X.
   - Entrez **150** pour la valeur Y.
3. Ouvrez le panneau Actions, sélectionnez l'image 1 dans le scénario principal et entrez le code ActionScript suivant :

```
aList.addItem({label:"Blue", data:0x0000CC});
aList.addItem({label:"Green", data:0x00CC00});
aList.addItem({label:"Yellow", data:0xFFFF00});
aList.addItem({label:"Orange", data:0xFF6600});
aList.addItem({label:"Black", data:0x000000});

var aBox:MovieClip = new MovieClip();
addChild(aBox);

aList.addEventListener(Event.CHANGE, changeHandler);
function changeHandler(event:Event) {
  drawBox(aBox, event.target.selectedItem.data);
};

function drawBox(box:MovieClip,color:uint):void {
       box.graphics.beginFill(color, 1.0);
       box.graphics.drawRect(225, 150, 100, 100);
       box.graphics.endFill();
}
```

4. Choisissez Contrôle > Tester l'animation pour exécuter l'application.
5. Dans la liste, cliquez sur les couleurs pour les afficher dans un clip.

L'exemple suivant utilise ActionScript pour créer une liste simple qu'il remplit à l'aide de la méthode `addItem()`.

Pour créer une occurrence du composant List à l'aide d'ActionScript :

1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Faites glisser le composant List du panneau Composants vers le panneau Bibliothèque.
3. Ouvrez le panneau Actions, sélectionnez l'image 1 dans le scénario principal et entrez le code ActionScript suivant :

```
import fl.controls.List;

var aList:List = new List();
aList.addItem({label:"One", data:1});
aList.addItem({label:"Two", data:2});
aList.addItem({label:"Three", data:3});
aList.addItem({label:"Four", data:4});
aList.addItem({label:"Five", data:5});
aList.setSize(60, 40);
aList.move(200,200);
addChild(aList);
aList.addEventListener(Event.CHANGE, changeHandler);
function changeHandler(event:Event):void {
  trace(event.target.selectedItem.data);
}
```

4. Choisissez Contrôle > Tester l'animation pour exécuter l'application.

Utilisation du composant NumericStepper

Le composant NumericStepper permet à un utilisateur de faire défiler un ensemble ordonné de nombres. Il s'agit d'un nombre dans une zone de texte affiché à côté de petits boutons fléchés vers le haut et vers le bas. Lorsqu'un utilisateur appuie sur les boutons, le nombre augmente ou diminue de façon incrémentielle en fonction de l'unité spécifiée dans le paramètre `stepSize` jusqu'à ce que l'utilisateur relâche les boutons ou que la valeur maximale ou minimale soit atteinte. Le texte dans la zone de texte du composant NumericStepper peut également être modifié.

Un aperçu en direct de chaque occurrence NumericStepper reflète la valeur du paramètre value dans l'Inspecteur des propriétés ou des composants. Cependant, il n'y a aucune interaction entre le clavier ou la souris et les boutons fléchés du composant NumericStepper dans l'aperçu en direct.

Interaction de l'utilisateur avec le composant NumericStepper

Vous pouvez utiliser le composant NumericStepper là où vous souhaitez qu'un utilisateur sélectionne une valeur numérique. Par exemple, vous pouvez utiliser un composant NumericStepper dans un formulaire pour définir la la date d'expiration d'une carte bancaire en spécifiant le mois, le jour et l'année. Vous pouvez également utiliser un composant NumericStepper pour permettre à un utilisateur d'augmenter ou de diminuer la taille d'une police.

Le composant NumericStepper gère uniquement les données numériques. Vous devez également redimensionner l'incrémenteur lors de la programmation pour afficher plus de deux chiffres (par exemple, les nombres 5246 ou 1,34).

Vous pouvez activer ou désactiver un composant NumericStepper dans une application. Lorsqu'il est désactivé, le composant NumericStepper ne reçoit aucune information provenant du clavier ou de la souris. Lorsqu'il est activé, le composant NumericStepper reçoit le focus si vous cliquez dessus ou si vous utilisez la tabulation pour l'ouvrir et son focus interne est défini dans la zone de texte. Lorsqu'une occurrence NumericStepper a le focus, vous pouvez utiliser les touches suivantes pour la contrôler :

Touche	Description
Flèche vers le bas	La valeur est modifiée d'une unité.
Flèche gauche	Déplace le point d'insertion vers la gauche à l'intérieur de la zone de texte.
Flèche droite	Déplace le point d'insertion vers la droite à l'intérieur de la zone de texte.
Maj+Tab	Place le focus sur l'objet précédent.
Tab	Place le focus sur l'objet suivant.
Flèche vers le haut	La valeur est modifiée d'une unité.

Pour plus d'informations sur le contrôle du focus, consultez la classe FocusManager dans le *Guide de référence du langage et des composants ActionScript 3.0* et la section « Utilisation de FocusManager », à la page 56.

Paramètres du composant NumericStepper

Dans l'Inspecteur des propriétés ou l'Inspecteur des composants, vous pouvez définir les paramètres suivants pour chaque occurrence NumericStepper : `maximum`, `minimum`, `stepSize` et `value`. Chacun de ces paramètres possède une propriété ActionScript correspondante du même nom. Pour plus d'informations sur les valeurs possibles de ces paramètres, consultez la classe NumericStepper dans le *Guide de référence du langage et des composants ActionScript 3.0*.

Création d'une application avec le composant NumericStepper

La procédure suivante explique comment ajouter un composant NumericStepper à une application lors de la programmation. L'exemple place un composant NumericStepper et un composant Label sur la scène et crée un écouteur pour un événement `Event.CHANGE` sur l'occurrence de composant NumericStepper. Lorsque la valeur dans l'incrémenteur numérique change, l'exemple affiche la nouvelle valeur dans la propriété `text` de l'occurrence Label.

Pour créer une application avec le composant NumericStepper :

1. Faites glisser un composant NumericStepper du panneau Composants vers la scène.
2. Dans l'Inspecteur des propriétés, entrez le nom d'occurrence **aNs**.
3. Faites glisser un composant Label du panneau Composants vers la scène.
4. Dans l'Inspecteur des propriétés, entrez le nom d'occurrence **aLabel**.
5. Ouvrez le panneau Actions, sélectionnez l'image 1 dans le scénario principal et entrez le code ActionScript suivant :

```
import flash.events.Event;

aLabel.text = "value = " + aNs.value;

aNs.addEventListener(Event.CHANGE, changeHandler);
function changeHandler(event:Event) :void {
 aLabel.text = "value = " + event.target.value;
};
```

Cet exemple définit la propriété `text` de l'étiquette sur la valeur du composant NumericStepper. La fonction `changeHandler()` met à jour la propriété `text` de l'étiquette à chaque fois que la valeur de l'occurrence NumericStepper change.

6. Choisissez Contrôle> Tester l'animation.

L'exemple suivant crée trois composants NumericStepper à l'aide du code ActionScript, chacun d'entre eux représentant respectivement le mois, le jour et l'année de la date de naissance de l'utilisateur. Il ajoute également des étiquettes pour une invite et des identificateurs pour chaque composant NumericStepper.

Pour créer un composant NumericStepper à l'aide d'ActionScript :

1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Faites glisser une étiquette vers le panneau Bibliothèque.
3. Faites glisser un composant NumericStepper vers le panneau Bibliothèque.
4. Ouvrez le panneau Actions, sélectionnez l'image 1 dans le scénario principal et entrez le code ActionScript suivant :

```
import fl.controls.Label;
import fl.controls.NumericStepper;

var dobPrompt:Label = new Label();
var moPrompt:Label = new Label();
var dayPrompt:Label = new Label();
var yrPrompt:Label = new Label();

var moNs:NumericStepper = new NumericStepper();
var dayNs:NumericStepper = new NumericStepper();
var yrNs:NumericStepper = new NumericStepper();

addChild(dobPrompt);
addChild(moPrompt);
addChild(dayPrompt);
addChild(yrPrompt);
addChild(moNs);
addChild(dayNs);
addChild(yrNs);

dobPrompt.setSize(65, 22);
dobPrompt.text = "Date of birth:"
dobPrompt.move(80, 150);

moNs.move(150, 150);
moNs.setSize(40, 22);
moNs.minimum = 1;
moNs.maximum = 12;
moNs.stepSize = 1;
moNs.value = 1;
```

```
moPrompt.setSize(25, 22);
moPrompt.text = "Mo.";
moPrompt.move(195, 150);

dayNs.move(225, 150);
dayNs.setSize(40, 22);
dayNs.minimum = 1;
dayNs.maximum = 31;
dayNs.stepSize = 1;
dayNs.value = 1;

dayPrompt.setSize(25, 22);
dayPrompt.text = "Day";
dayPrompt.move(270, 150);

yrNs.move(300, 150);
yrNs.setSize(55, 22);
yrNs.minimum = 1900;
yrNs.maximum = 2006;
yrNs.stepSize = 1;
yrNs.value = 1980;

yrPrompt.setSize(30, 22);
yrPrompt.text = "Year";
yrPrompt.move(360, 150);
```

5. Choisissez Contrôle > Tester l'animation pour exécuter l'application.

Utilisation du composant ProgressBar

Le composant ProgressBar affiche la progression du contenu en chargement, ce qui est rassurant pour l'utilisateur car du contenu volumineux peut retarder l'exécution de l'application. Le composant ProgressBar permet d'afficher la progression de chargement des images et des parties d'une application. Le processus de chargement peut être déterminé ou indéterminé. Une barre de progression *determinate* est une représentation linéaire de la progression d'une tâche dans le temps. Elle est utilisée lorsque la quantité de contenu à charger est connue. Une barre de progression *indeterminate* est utilisée lorsque la quantité de contenu à charger est inconnue. Vous pouvez également ajouter un composant Label pour afficher la progression de chargement en pourcentage.

Le composant ProgressBar utilise la mise à l'échelle à 9 découpes et possède une enveloppe de barre, une enveloppe de rail et une enveloppe indéterminée.

Interaction de l'utilisateur avec le composant ProgressBar

Il existe trois modes d'utilisation du composant ProgressBar. Les modes les plus couramment utilisés sont event et polled. Ces modes spécifient un processus de chargement qui émet des événements `progress` et `complete` (mode event et polled) ou expose des propriétés `bytesLoaded` et `bytesTotal` (mode polled). Vous pouvez également utiliser le composant ProgressBar en mode manuel en définissant les propriétés `maximum`, `minimum` et `value` et en appelant la méthode `ProgressBar.setProgress()`. Vous pouvez définir la propriété indeterminate pour indiquer si le composant ProgressBar a un remplissage rayé et une source de taille inconnue (`true`), ou un remplissage uni et une source de taille connue (`false`).

Spécifiez le mode du composant ProgressBar en définissant sa propriété `mode`, via le paramètre `mode` dans l'Inspecteur des propriétés ou l'Inspecteur des composants, ou via ActionScript.

Si vous utilisez le composant ProgressBar pour afficher l'état de traitement, comme l'analyse de 100 000 éléments, alors que celui-ci se trouve dans une boucle image par image, aucune mise à jour ne sera visible sur le composant ProgressBar car il n'y a pas de retraçage de l'écran.

Paramètres du composant ProgressBar

Dans l'Inspecteur des propriétés ou l'Inspecteur des composants, vous pouvez définir les paramètres suivants pour chaque occurrence ProgressBar : `direction`, `mode` et `source`. Chacun de ces paramètres possède une propriété ActionScript correspondante du même nom.

Vous pouvez contrôler ces options et d'autres options du composant ProgressBar à l'aide des propriétés, méthodes et événements d'ActionScript. Pour plus d'informations, consultez la classe ProgressBar dans le *Guide de référence du langage et des composants ActionScript 3.0*.

Création d'une application avec le composant ProgressBar

La procédure suivante indique comment ajouter un composant ProgressBar à une application lors de la programmation. Dans cet exemple, le composant ProgressBar utilise le mode event. En mode event, le contenu en cours de chargement émet des événements `progress` et `complete` distribués par le composant ProgressBar pour afficher la progression. Lorsque l'événement `progress` se produit, l'exemple met à jour une étiquette pour afficher le pourcentage de contenu chargé. Lorsque l'événement `complete` se produit, l'exemple affiche « Chargement terminé » et la valeur de la propriété `bytesTotal`, qui représente la taille du fichier.

Pour créer une application avec le composant ProgressBar en mode event :

1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Faites glisser le composant ProgressBar du panneau Composants vers la scène.
   - Dans l'Inspecteur des propriétés, entrez le nom d'occurrence **aPb**.
   - Dans la section Paramètres, entrez **200** pour la valeur X.
   - Entrez **260** pour la valeur Y.
   - Sélectionnez `event` pour le paramètre `mode`.
3. Faites glisser le composant Button du panneau Composants vers la scène.
   - Dans l'Inspecteur des propriétés, entrez le nom d'occurrence **loadButton**.
   - Entrez **220** pour le paramètre X.
   - Entrez **290** pour le paramètre Y.
   - Entrez **Load Sound** pour le paramètre `label`.
4. Faites glisser le composant Label sur la scène et donnez-lui le nom d'occurrence **progLabel**.
   - Entrez **150** pour la valeur W.
   - Entrez **200** pour le paramètre X.
   - Entrez **230** pour le paramètre Y.
   - Dans la section Paramètres, effacez la valeur du paramètre `text`.
5. Ouvrez le panneau Actions, sélectionnez l'image 1 dans le scénario principal et entrez le code ActionScript suivant, qui permet de charger un fichier audio .mp3 :

```
import fl.controls.ProgressBar;
import flash.events.ProgressEvent;
import flash.events.IOErrorEvent;

var aSound:Sound = new Sound();
aPb.source = aSound;
var url:String = "http://www.helpexamples.com/flash/sound/song1.mp3";
var request:URLRequest = new URLRequest(url);

aPb.addEventListener(ProgressEvent.PROGRESS, progressHandler);
aPb.addEventListener(Event.COMPLETE, completeHandler);
aSound.addEventListener(IOErrorEvent.IO_ERROR, ioErrorHandler);
loadButton.addEventListener(MouseEvent.CLICK, clickHandler);

function progressHandler(event:ProgressEvent):void {
  progLabel.text = ("Sound loading ... " + aPb.percentComplete);
}
```

```
function completeHandler(event:Event):void {
  trace("Loading complete");
  trace("Size of file: " + aSound.bytesTotal);
  aSound.close();
  loadButton.enabled = false;
}

function clickHandler(event:MouseEvent) {
  aSound.load(request);
}

function ioErrorHandler(event:IOErrorEvent):void {
  trace("Load failed due to: " + event.text);
}
```

6. Choisissez Contrôle> Tester l'animation.

L'exemple suivant définit le composant ProgressBar en mode polled. En mode polled, la progression est déterminée en écoutant les événements `progress` du contenu en cours de chargement et en utilisant ses propriétés `bytesLoaded` et `bytesTotal` pour calculer la progression. Cet exemple charge un objet Sound, écoute ses événements `progress` et calcule le pourcentage chargé à l'aide de ses propriétés `bytesLoaded` et `bytesTotal`. Il affiche le pourcentage chargé dans une étiquette et dans le panneau Sortie.

Pour créer une application avec le composant ProgressBar en mode polled :

1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Faites glisser un composant ProgressBar du panneau Composants vers la scène et entrez les valeurs suivantes dans l'Inspecteur des propriétés :
   - Entrez le nom d'occurrence **aPb**.
   - Entrez **185** pour la valeur X.
   - Entrez **225** pour la valeur Y.
3. Faites glisser le composant Label sur la scène et entrez les valeurs suivantes dans l'Inspecteur des propriétés :
   - Entrez le nom d'occurrence **progLabel**.
   - Entrez **180** pour la valeur X.
   - Entrez **180** pour la valeur Y.
   - Dans la section Paramètres, effacez la valeur du paramètre text.

4. Ouvrez le panneau Actions, sélectionnez l'image 1 dans le scénario principal et entrez le code ActionScript suivant, qui crée un objet Sound (`aSound`) et appelle la méthode `loadSound()` pour charger un son dans l'objet Sound :

```
import fl.controls.ProgressBarMode;
import flash.events.ProgressEvent;
import flash.media.Sound;

var aSound:Sound = new Sound();
var url:String = "http://www.helpexamples.com/flash/sound/song1.mp3";
var request:URLRequest = new URLRequest(url);

aPb.mode = ProgressBarMode.POLLED;
aPb.source = aSound;
aSound.addEventListener(ProgressEvent.PROGRESS, loadListener);

aSound.load(request);

function loadListener(event:ProgressEvent) {
  var percentLoaded:int = event.target.bytesLoaded /
  event.target.bytesTotal * 100;
  progLabel.text = "Percent loaded: " + percentLoaded + "%";
  trace("Percent loaded: " + percentLoaded + "%");
}
```

5. Choisissez Contrôle > Tester l'animation pour exécuter l'application.

L'exemple suivant définit le composant ProgressBar en mode manuel. En mode manuel, vous devez définir la progression manuellement en appelant la méthode `setProgress()` et en lui affectant les valeurs actuelles et maximales afin de déterminer le degré de progression. Vous ne définissez pas la propriété `source` en mode manuel . Cet exemple utilise un composant NumericStepper, avec une valeur maximale de 250, pour incrémenter le composant ProgressBar. Lorsque la valeur du composant NumericStepper change et déclenche un événement `CHANGE`, le gestionnaire d'événements (`nsChangeHander`) appelle la méthode `setProgress()` pour atteindre le composant ProgressBar. Il affiche également le pourcentage de progression effectué, en fonction de la valeur maximale.

Pour créer une application avec le composant ProgressBar en mode manuel :

1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Faites glisser le composant ProgressBar du panneau Composants vers la scène et spécifiez les valeurs suivantes pour celui-ci dans l'Inspecteur des propriétés :
   - Entrez le nom d'occurrence **aPb**.
   - Entrez **180** pour la valeur X.
   - Entrez **175** pour la valeur Y.

3. Faites glisser un composant NumericStepper sur la scène et entrez les valeurs suivantes dans l'Inspecteur des propriétés :
   - Entrez le nom d'occurrence **aNs**.
   - Entrez **220** pour la valeur X.
   - Entrez **215** pour la valeur Y.
   - Dans la section Paramètres, entrez **250** pour le paramètre maximum, **0** pour la valeur minimum, **1** pour le paramètre stepSize et **0** pour le paramètre value.
4. Faites glisser un composant Label sur la scène et entrez les valeurs suivantes dans l'Inspecteur des propriétés :
   - Entrez le nom d'occurrence **progLabel**.
   - Entrez **150** pour la valeur W.
   - Entrez **180** pour la valeur X.
   - Entrez **120** pour la valeur Y.
   - Dans l'onglet Paramètres, effacez la valeur Label du paramètre text.
5. Ouvrez le panneau Actions, sélectionnez l'image 1 dans le scénario principal et entrez le code suivant :

```
import fl.controls.ProgressBarDirection;
import fl.controls.ProgressBarMode;
import flash.events.Event;

aPb.direction = ProgressBarDirection.RIGHT;
aPb.mode = ProgressBarMode.MANUAL;
aPb.minimum = aNs.minimum;
aPb.maximum = aNs.maximum;
aPb.indeterminate = false;

aNs.addEventListener(Event.CHANGE, nsChangeHandler);

function nsChangeHandler(event:Event):void {
  aPb.value = aNs.value;
  aPb.setProgress(aPb.value, aPb.maximum);
   progLabel.text = "Percent of progress = " + int(aPb.percentComplete)
  + "%";
}
```

6. Choisissez Contrôle > Tester l'animation pour exécuter l'application.
7. Cliquez sur la flèche vers le haut sur le composant NumericStepper pour atteindre le composant ProgressBar.

L'exemple suivant crée un composant ProgressBar à l'aide d'ActionScript. Il duplique en outre la fonctionnalité de l'exemple précédent, qui crée un composant ProgressBar en mode manuel.

Pour créer un composant ProgressBar à l'aide d'ActionScript :

1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Faites glisser le composant ProgressBar vers le panneau Bibliothèque.
3. Faites glisser le composant NumericStepper vers le panneau Bibliothèque.
4. Faites glisser le composant Label vers le panneau Bibliothèque.
5. Ouvrez le panneau Actions, sélectionnez l'image 1 dans le scénario principal et entrez le code suivant :

```
import fl.controls.ProgressBar;
import fl.controls.NumericStepper;
import fl.controls.Label;
import fl.controls.ProgressBarDirection;
import fl.controls.ProgressBarMode;
import flash.events.Event;

var aPb:ProgressBar = new ProgressBar();
var aNs:NumericStepper = new NumericStepper();
var progLabel:Label = new Label();

addChild(aPb);
addChild(aNs);
addChild(progLabel);

aPb.move(180,175);
aPb.direction = ProgressBarDirection.RIGHT;
aPb.mode = ProgressBarMode.MANUAL;

progLabel.setSize(150, 22);
progLabel.move(180, 150);
progLabel.text = "";

aNs.move(220, 215);
aNs.maximum = 250;
aNs.minimum = 0;
aNs.stepSize = 1;
aNs.value = 0;

aNs.addEventListener(Event.CHANGE, nsChangeHandler);

function nsChangeHandler(event:Event):void {
    aPb.setProgress(aNs.value, aNs.maximum);
    progLabel.text = "Percent of progress = " + int(aPb.percentComplete)
  + "%";
}
```

6. Choisissez Contrôle > Tester l'animation pour exécuter l'application.
7. Cliquez sur la flèche vers le haut sur le composant NumericStepper pour atteindre le composant ProgressBar.

Utilisation du composant RadioButton

Le composant RadioButton vous permet d'obliger un utilisateur à choisir un seul élément parmi plusieurs possibilités. Ce composant doit être utilisé dans un groupe comprenant au moins deux occurrences de RadioButton. Seul un membre peut être sélectionné au sein du groupe. La sélection d'un bouton radio dans un groupe désélectionne le bouton jusqu'alors sélectionné dans le groupe. Vous définissez le paramètre `groupName` pour désigner le groupe auquel appartient un bouton radio.

Le bouton radio est un élément de base de nombreuses applications de formulaires sur le Web. Vous pouvez utiliser les boutons radio partout où vous souhaitez qu'un utilisateur opte pour un choix dans un groupe d'options. Par exemple, utilisez des boutons radio dans un formulaire pour demander quelle carte bancaire un utilisateur souhaite utiliser.

Interaction de l'utilisateur avec le composant RadioButton

Un bouton radio peut être activé ou désactivé. En état désactivé, le bouton radio ne réagit pas aux commandes de la souris ou du clavier. Lorsque l'utilisateur clique dans un groupe de composants RadioButton ou l'atteint via une touche de tabulation, seul le bouton radio sélectionné reçoit le focus. L'utilisateur peut ensuite utiliser les touches suivantes pour le contrôler :

Touche	Description
Flèche vers le haut/Flèche vers la gauche	La sélection se déplace vers le bouton radio précédent dans le groupe de boutons radio.
Flèche vers le bas/Flèche vers la droite	La sélection se déplace vers le bouton radio suivant dans le groupe de boutons radio.
Tab	Déplace le focus du groupe de boutons radio vers le composant suivant.

Pour plus d'informations sur le contrôle du focus, consultez l'interface IFocusManager et la classe FocusManager dans le *Guide de référence du langage et des composants ActionScript 3.0* et « Utilisation de FocusManager », à la page 56.

Un aperçu en direct de chaque occurrence de RadioButton sur la scène reflète les modifications effectuées sur les paramètres dans l'inspecteur Propriétés ou des composants lors de la programmation. Cependant, l'exclusion mutuelle de la sélection ne s'affiche pas dans l'aperçu en direct. Si vous définissez le paramètre sélectionné sur `true` pour deux boutons radio dans le même groupe, ils apparaissent tous deux comme étant sélectionnés même si seule la dernière occurrence créée apparaît comme étant sélectionnée lors de l'exécution. Pour plus d'informations, consultez la section « Paramètres du composant RadioButton », à la page 125.

Lorsque vous ajoutez le composant RadioButton à une application, vous pouvez le rendre accessible à un lecteur d'écran en ajoutant les lignes de code suivantes :

```
import fl.accessibility.RadioButtonAccImpl;
RadioButtonAccImpl.enableAccessibility();
```

Quel que soit le nombre d'occurrences d'un composant, l'activation de son accessibilité ne se fait qu'une fois. Pour plus d'informations, consultez le Chapitre 18, « Création de contenu accessible » du guide *Utilisation de Flash*.

Paramètres du composant RadioButton

Dans l'Inspecteur des propriétés ou l'Inspecteur des composants, vous pouvez définir les paramètres de programmation suivants pour chaque occurrence du composant RadioButton : `groupName`, `label`, `labelPlacement`, `selected` et `value`. Chacun de ces paramètres possède une propriété ActionScript correspondante du même nom. Pour plus d'informations sur les valeurs possibles de ces paramètres, consultez la classe RadioButton dans le *Guide de référence du langage et des composants ActionScript 3.0*.

ActionScript vous permet de définir des options supplémentaires pour les occurrences de RadioButton à l'aide des méthodes, propriétés et événements de la classe RadioButton.

Création d'une application avec le composant RadioButton

La procédure suivante explique comment ajouter des composants RadioButton à une application lors de la programmation. Dans cet exemple, les composants RadioButton sont utilisés pour présenter une question dont la réponse sera oui ou non. Les données du composant RadioButton sont affichées dans un composant TextArea.

Pour créer une application avec le composant RadioButton :

1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Faites glisser deux composants RadioButton du panneau Composants vers la scène.
3. Sélectionnez le premier bouton radio. Dans l'Inspecteur des composants, attribuez-lui le nom d'occurrence **yesRb** et le nom de groupe **rbGroup.**
4. Sélectionnez le deuxième bouton radio. Dans l'Inspecteur des composants, attribuez-lui le nom d'occurrence **noRb** et le nom de groupe **rbGroup.**
5. Faites glisser un composant TextArea du panneau Composants vers la scène et nommez l'occurrence **aTa**.
6. Ouvrez le panneau Actions, sélectionnez l'image 1 dans le scénario principal et entrez le code ActionScript suivant :

```
yesRb.label = "Yes";
yesRb.value = "For";
noRb.label = "No";
noRb.value = "Against";

yesRb.move(50, 100);
noRb.move(100, 100);
aTa.move(50, 30);
noRb.addEventListener(MouseEvent.CLICK, clickHandler);
yesRb.addEventListener(MouseEvent.CLICK, clickHandler);

function clickHandler(event:MouseEvent):void {
  aTa.text = event.target.value;
}
```

7. Choisissez Contrôle > Tester l'animation pour exécuter l'application.

L'exemple suivant utilise ActionScript pour créer trois composants RadioButton pour les couleurs rouge, bleu et vert, et trace un cadre gris. La propriété `value` de chaque composant RadioButton spécifie la valeur hexadécimale de la couleur associée au bouton. Lorsqu'un utilisateur clique sur l'un des composants RadioButton, la fonction `clickHandler()` appelle la fonction `drawBox()`, transmettant ainsi la couleur de la propriété `value` du composant RadioButton au cadre.

Pour créer un composant RadioButton à l'aide d'ActionScript :

1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Faites glisser le composant RadioButton vers le panneau Bibliothèque.

3. Ouvrez le panneau Actions, sélectionnez l'image 1 dans le scénario principal et entrez le code ActionScript suivant :

```
import fl.controls.RadioButton;
import fl.controls.RadioButtonGroup;

var redRb:RadioButton = new RadioButton();
var blueRb:RadioButton = new RadioButton();
var greenRb:RadioButton = new RadioButton();
var rbGrp:RadioButtonGroup = new RadioButtonGroup("colorGrp");

var aBox:MovieClip = new MovieClip();
drawBox(aBox, 0xCCCCCC);

addChild(redRb);
addChild(blueRb);
addChild(greenRb);
addChild(aBox);

redRb.label = "Red";
redRb.value = 0xFF0000;
blueRb.label = "Blue";
blueRb.value = 0x0000FF;
greenRb.label = "Green";
greenRb.value = 0x00FF00;
redRb.group = blueRb.group = greenRb.group = rbGrp;
redRb.move(100, 260);
blueRb.move(150, 260);
greenRb.move(200, 260);

rbGrp.addEventListener(MouseEvent.CLICK, clickHandler);

function clickHandler(event:MouseEvent):void {
  drawBox(aBox, event.target.selection.value);
}

function drawBox(box:MovieClip,color:uint):void {
       box.graphics.beginFill(color, 1.0);
       box.graphics.drawRect(125, 150, 100, 100);
       box.graphics.endFill();
}
```

4. Choisissez Contrôle > Tester l'animation pour exécuter l'application.

Pour plus d'informations, consultez la classe RadioButton dans le *Guide de référence du langage et des composants ActionScript 3.0*.

Utilisation du composant ScrollPane

Vous pouvez utiliser le composant ScrollPane pour afficher le contenu qui ne rentre pas dans la zone dans laquelle il est chargé. Par exemple, si vous devez afficher une image de grande taille mais que vous avez peu de place dans une application, vous pouvez la charger dans un composant ScrollPane. Le composant ScrollPane peut accepter les clips, les fichiers JPEG, PNG, GIF et SWF.

Les composants tels que ScrollPane et UILoader possèdent des événements `complete` qui vous permettent de déterminer à quel moment le chargement du contenu est terminé. Si vous souhaitez définir des propriétés sur le contenu d'un composant ScrollPane ou UILoader, écoutez l'événement `complete` et définissez la propriété dans le gestionnaire d'événements. Par exemple, le code suivant crée un écouteur pour l'événement Event.`COMPLETE` et un gestionnaire d'événements qui définit la propriété `alpha` du contenu du composant ScrollPane sur 0,5 :

```
function spComplete(event:Event):void{
  aSp.content.alpha = .5;
}
aSp.addEventListener(Event.COMPLETE, spComplete);
```

Si vous spécifiez un emplacement lors du chargement du contenu dans le composant ScrollPane, vous devez spécifier les valeurs 0, 0 (coordonnées X et Y). Par exemple, le code suivant charge le composant ScrollPane correctement car le cadre est tracé à l'emplacement 0,0 :

```
var box:MovieClip = new MovieClip();
box.graphics.beginFill(0xFF0000, 1);
box.graphics.drawRect(0, 0, 150, 300);
box.graphics.endFill();
aSp.source = box;//load ScrollPane
```

Pour plus d'informations, consultez la classe ScrollPane dans le *Guide de référence du langage et des composants ActionScript 3.0*.

■

Interaction de l'utilisateur avec le composant ScrollPane

Un composant ScrollPane peut être activé ou désactivé. En état désactivé, le composant ScrollPane ne réagit pas aux commandes de la souris ou du clavier. L'utilisateur peut utiliser les touches suivantes pour contrôler un composant ScrollPane lorsqu'il détient le focus.

Touche	Description
Flèche vers le bas	Le contenu se déplace d'une ligne de défilement verticale vers le haut.
Flèche vers le haut	Le contenu se déplace d'une ligne de défilement verticale vers le bas.
Fin	Le contenu se déplace en bas du composant ScrollPane.
Flèche gauche	Le contenu se déplace d'une ligne de défilement horizontale vers la droite.
Flèche droite	Le contenu se déplace d'une ligne de défilement horizontale vers la gauche.
Origine	Le contenu se déplace en haut du composant ScrollPane.
Fin	Le contenu se déplace en bas du composant ScrollPane.
Pg. Suiv.	Le contenu se déplace d'une page de défilement verticale vers le haut.
Pg. Préc.	Le contenu se déplace d'une page de défilement verticale vers le bas.

Un utilisateur peut utiliser la souris pour interagir avec le composant ScrollPane sur son contenu et sur les barres de défilement horizontale et verticale. L'utilisateur peut faire glisser le contenu à l'aide de la souris lorsque la propriété `scrollDrag` est définie sur `true`. L'apparition du curseur en forme de main sur le contenu indique que l'utilisateur peut faire glisser le contenu. Contrairement à la plupart des autres commandes, les actions se produisent dès que l'utilisateur appuie sur le bouton de la souris et se poursuivent jusqu'à ce qu'il le relâche. Si le contenu présente des arrêts de tabulation valides, vous devez définir la propriété `scrollDrag` sur false. Sinon, chaque sollicitation de la souris sur le contenu invoquera le déplacement par défilement.

Paramètres du composant ScrollPane

Vous pouvez définir les paramètres suivants pour chaque occurrence ScrollPane dans l'Inspecteur des propriétés ou l'Inspecteur des composants : `horizontalLineScrollSize`, `horizontalPageScrollSize`, `horizontalScrollPolicy`, `scrollDrag`, `source`, `verticalLineScrollSize`, `verticalPageScrollSize` et `verticalScrollPolicy`. Chacun de ces paramètres possède une propriété ActionScript correspondante du même nom. Pour plus d'informations sur les valeurs possibles de ces paramètres, consultez la classe ScrollPane dans le *Guide de référence du langage et des composants ActionScript 3.0*.

Vous pouvez rédiger du code ActionScript pour contrôler ces options et d'autres options d'un composant ScrollPane en utilisant ses propriétés, méthodes et événements.

Création d'une application avec le composant ScrollPane

La procédure suivante explique comment ajouter un composant ScrollPane à une application pendant la programmation. Dans cet exemple, le composant ScrollPane charge une image depuis un chemin spécifié par la propriété `source`.

Pour créer une application avec le composant ScrollPane :

1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Faites glisser le composant ScrollPane du panneau Composants vers la scène et nommez l'occurrence **aSp**.
3. Ouvrez le panneau Actions, sélectionnez l'image 1 dans le scénario principal et entrez le code ActionScript suivant :

```
import fl.events.ScrollEvent;

aSp.setSize(300, 200);

function scrollListener(event:ScrollEvent):void {
 trace("horizontalScPosition: " + aSp.horizontalScrollPosition +
      ", verticalScrollPosition = " + aSp.verticalScrollPosition);
};
aSp.addEventListener(ScrollEvent.SCROLL, scrollListener);

function completeListener(event:Event):void {
 trace(event.target.source + " has completed loading.");
};
// Add listener.
aSp.addEventListener(Event.COMPLETE, completeListener);

aSp.source = "http://www.helpexamples.com/flash/images/image1.jpg";
```

4. Choisissez Contrôle > Tester l'animation pour exécuter l'application.

Cet exemple crée un composant ScrollPane, définit sa taille et y charge une image à l'aide de la propriété `source`. Il crée également deux écouteurs. Le premier écoute un événement `scroll` et affiche la position de l'image lorsque l'utilisateur utilise le défilement vertical ou horizontal. Le deuxième écoute un événement `complete` et affiche un message dans le panneau Sortie qui indique que le chargement de l'image est terminé.

Cet exemple crée un composant ScrollPane à l'aide d'ActionScript et y place un clip (cadre rouge) mesurant 150 pixels de large et 300 pixels de haut.

Pour créer une occurrence du composant ScrollPane à l'aide d'ActionScript :

1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Faites glisser le composant ScrollPane du panneau Composants vers le panneau Bibliothèque.
3. Faites glisser le composant DataGrid du panneau Composants vers le panneau Bibliothèque.
4. Ouvrez le panneau Actions, sélectionnez l'image 1 dans le scénario principal et entrez le code ActionScript suivant :

```
import fl.containers.ScrollPane;
import fl.controls.ScrollPolicy;
import fl.controls.DataGrid;
import fl.data.DataProvider;

var aSp:ScrollPane = new ScrollPane();
var aBox:MovieClip = new MovieClip();
drawBox(aBox, 0xFF0000);//draw a red box

aSp.source = aBox;
aSp.setSize(150, 200);
aSp.move(100, 100);

addChild(aSp);

function drawBox(box:MovieClip,color:uint):void {
       box.graphics.beginFill(color, 1);
       box.graphics.drawRect(0, 0, 150, 300);
       box.graphics.endFill();
}
```

5. Choisissez Contrôle > Tester l'animation pour exécuter l'application.

Utilisation du composant Slider

Le composant Slider permet à l'utilisateur de sélectionner une valeur en déplaçant un *curseur de défilement* graphique entre les points d'extrémité d'un rail correspondant à une plage de valeurs. Vous pouvez utiliser un curseur pour permettre à un utilisateur de choisir une valeur comme un nombre ou un pourcentage, par exemple. Vous pouvez également utiliser ActionScript pour contraindre la valeur du curseur à influencer le comportement d'un deuxième objet. Par exemple, vous pouvez associer le curseur à une image et la rétrécir ou l'agrandir en fonction de la position relative, ou de la valeur, du curseur.

La valeur actuelle du composant Slider est déterminée par l'emplacement relatif du curseur de défilement entre les extrémités du rail ou les valeurs minimales et maximales du composant Slider.

Le composant Slider accepte une plage continue de valeurs entre ses valeurs minimales et maximales, mais vous pouvez également définir le paramètre `snapInterval` pour spécifier les intervalles entre la valeur minimale et maximale. Le curseur peut afficher des graduations, qui ne dépendent pas de ses valeurs affectées, à intervalles spécifiés le long du rail.

Le curseur a une orientation horizontale par défaut, mais vous pouvez lui attribuer une orientation verticale en définissant la valeur du paramètre `direction` sur vertical. Le rail du curseur s'étend d'une extrémité à l'autre et les graduations sont placées de gauche à droite juste au-dessus du rail.

Interaction de l'utilisateur avec le composant Slider

Lorsque l'occurrence d'un composant Slider a le focus, vous pouvez la contrôler à l'aide des touches suivantes :

Touche	Description
Flèche droite	Augmente la valeur associée d'un curseur horizontal.
Flèche vers le haut	Augmente la valeur associée d'un curseur vertical.
Flèche gauche	Diminue la valeur associée d'un curseur horizontal.
Flèche vers le bas	Diminue la valeur associée d'un curseur vertical.
Maj+Tab	Place le focus sur l'objet précédent.
Tab	Place le focus sur l'objet suivant.

Pour plus d'informations sur le contrôle du focus, consultez l'interface IFocusManager et la classe FocusManager dans le *Guide de référence du langage et des composants ActionScript 3.0* et « Utilisation de FocusManager », à la page 56.

L'aperçu en direct des occurrences Slider reflète les modifications apportées aux paramètres dans l'Inspecteur des propriétés ou l'Inspecteur des composants pendant la programmation.

Paramètres du composant Slider

Dans l'Inspecteur des propriétés ou l'Inspecteur des composants, vous pouvez définir les paramètres de programmation suivants pour chaque occurrence du composant Slider : `direction`, `liveDragging`, `maximum`, `minimum`, `snapInterval`, `tickInterval` et `value`. Chacun de ces paramètres possède une propriété ActionScript correspondante du même nom. Pour plus d'informations sur les valeurs possibles de ces paramètres, consultez la classe Slider dans le *Guide de référence du langage et des composants ActionScript 3.0*.

Création d'une application avec le composant Slider

L'exemple suivant crée une occurrence Slider pour permettre à l'utilisateur d'exprimer son niveau de satisfaction concernant un événement hypothétique. L'utilisateur déplace le curseur vers la droite ou vers la gauche pour indiquer un niveau de satisfaction supérieur ou inférieur.

Pour créer une application avec le composant Slider :

1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Faites glisser un composant Label du panneau Composants au centre de la scène.
   - Attribuez-lui le nom d'occurrence **valueLabel**.
   - Affectez la valeur **0 %** au paramètre `text`.
3. Faites glisser un composant Slider du panneau Composants et centrez-le sous `value_lbl`.
   - Attribuez-lui le nom d'occurrence **aSlider**.
   - Affectez-lui la valeur de largeur (L:) **200**.
   - Affectez-lui la valeur de hauteur (H:) **10**.
   - Affectez la valeur **100** au paramètre `maximum`.
   - Affectez la valeur **10** aux paramètres `snapInterval` et `tickInterval`.
4. Faites glisser une autre occurrence Label du panneau Bibliothèque et centrez-la sous `aSlider`.
   - Attribuez-lui le nom d'occurrence **promptLabel.**
   - Affectez-lui la valeur de largeur (L:) 250.
   - Affectez-lui la valeur de hauteur (H:) 22.
   - Entrez **Veuillez indiquer votre niveau de satisfaction** pour le paramètre `text`.

5. Ouvrez le panneau Actions, sélectionnez l'image 1 dans le scénario principal et entrez le code ActionScript suivant :

```
import fl.controls.Slider;
import fl.events.SliderEvent;
import fl.controls.Label;

aSlider.addEventListener(SliderEvent.CHANGE, changeHandler);

function changeHandler(event:SliderEvent):void {
  valueLabel.text = event.value + "percent";
}
```

6. Choisissez Contrôle> Tester l'animation.

   Dans cet exemple, lorsque vous déplacez le curseur d'un intervalle vers un autre, l'écouteur de l'événement `SliderEvent.CHANGE` met à jour la propriété `text` de `valueLabel` pour afficher le pourcentage correspondant à la position du curseur.

L'exemple suivant crée un composant Slider à l'aide d'ActionScript. Il télécharge l'image d'une fleur et utilise le curseur pour permettre à l'utilisateur d'appliquer un fondu à l'image ou de l'éclaircir en modifiant sa propriété `alpha` pour la faire correspondre à la valeur du curseur.

Pour créer une application avec le composant Slider à l'aide d'ActionScript :

1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Faites glisser le composant Label et le composant Slider du panneau Composants vers le panneau Bibliothèque du document en cours.

   Cette opération permet d'ajouter les composants à la bibliothèque, mais elle ne permet pas de les rendre visibles dans l'application.
3. Ouvrez le panneau Actions, sélectionnez l'image 1 dans le scénario principal et entrez le code suivant pour créer et positionner les occurrences de composant :

```
import fl.controls.Slider;
import fl.events.SliderEvent;
import fl.controls.Label;
import fl.containers.UILoader;

var sliderLabel:Label = new Label();
sliderLabel.width = 120;
sliderLabel.text = "< Fade - Brighten >";
sliderLabel.move(170, 350);

var aSlider:Slider = new Slider();
aSlider.width = 200;
aSlider.snapInterval = 10;
aSlider.tickInterval = 10;
aSlider.maximum = 100;
aSlider.value = 100;
aSlider.move(120, 330);
```

```
var aLoader:UILoader = new UILoader();
aLoader.source = "http://www.flash-mx.com/images/image1.jpg";
aLoader.scaleContent = false;

addChild(sliderLabel);
addChild(aSlider);
addChild(aLoader);

aLoader.addEventListener(Event.COMPLETE, completeHandler);

function completeHandler(event:Event) {
  trace("Number of bytes loaded: " + aLoader.bytesLoaded);
}

aSlider.addEventListener(SliderEvent.CHANGE, changeHandler);

function changeHandler(event:SliderEvent):void {
      aLoader.alpha = event.value * .01;
}
```

4. Choisissez Contrôle > Tester l'animation pour exécuter l'application.
5. Déplacez le curseur du composant Slider vers la gauche pour appliquer un fondu à l'image et vers la droite pour l'éclaircir.

Utilisation du composant TextArea

Le composant TextArea renvoie à la ligne automatiquement l'objet TextField ActionScript natif. Vous pouvez utiliser le composant TextArea pour afficher le texte et également pour modifier et recevoir la saisie de texte si la propriété `editable` est définie sur `true`. Le composant peut afficher ou recevoir plusieurs lignes de texte et renvoyer automatiquement à la ligne les lignes de texte trop longues si la propriété `wordWrap` est définie sur `true`. La propriété `restrict` vous permet de limiter le nombre de caractères pouvant être saisis par l'utilisateur et la propriété `maxChars` vous permet de spécifier le nombre maximum de caractères que l'utilisateur peut entrer. Si le texte dépasse les limites horizontales ou verticales de la zone de texte, les barres de défilement horizontale et verticale apparaissent automatiquement sauf si les propriétés qui y sont associées, `horizontalScrollPolicy` et `verticalScrollPolicy`, sont définies sur `off`.

Vous pouvez utiliser un composant TextArea partout où vous avez besoin d'un champ de texte multiligne. Par exemple, vous pouvez utiliser le composant TextArea comme un champ de commentaires dans un formulaire. Vous pouvez définir un écouteur qui vérifie si le champ est vide lorsqu'un utilisateur sort du champ. Cet écouteur peut afficher un message d'erreur indiquant qu'un commentaire doit être entré dans ce champ.

Si vous avez besoin d'un champ de texte à ligne unique, utilisez le composant TextInput.

Vous pouvez définir le style `textFormat` à l'aide de la méthode `setStyle()` pour modifier le style du texte qui apparaît dans une occurrence TextArea. Vous pouvez également définir un composant TextArea au format HTML à l'aide de la propriété `htmlText` dans ActionScript, et vous pouvez définir la propriété `displayAsPassword` sur `true` pour masquer le texte avec des astérisques. Si vous définissez la propriété `condenseWhite` sur `true`, Flash supprime les espaces blancs supplémentaires du nouveau texte réservés aux espaces, sauts de ligne, etc. Cela n'a aucun effet sur le texte apparaissant déjà dans le contrôle.

Interaction de l'utilisateur avec le composant TextArea

Un composant TextArea peut être activé ou désactivé dans une application. Lorsqu'il est désactivé, il ne peut pas recevoir les informations en provenance de la souris ou du clavier. Lorsqu'il est activé, il suit les mêmes règles de focus, de sélection et de navigation qu'un objet TextField ActionScript. Lorsque l'occurrence d'un composant TextArea a le focus, vous pouvez la contrôler à l'aide des touches suivantes.

Touche	Description
Touches fléchées	Déplacent le point d'insertion vers le haut, le bas, la gauche ou la droite dans le texte, si celui-ci est modifiable.
Pg. Suiv.	Déplace le point d'insertion à la fin du texte, si celui-ci est modifiable.
Pg. Préc.	Déplace le point d'insertion au début du texte, si celui-ci est modifiable.
Maj+Tab	Place le focus sur l'objet précédent dans la boucle de tabulation.
Tab	Place le focus sur l'objet suivant dans la boucle de tabulation.

Pour plus d'informations sur le contrôle du focus, consultez la classe FocusManager dans le *Guide de référence du langage et des composants ActionScript 3.0* et la section « Utilisation de FocusManager », à la page 56.

Paramètres du composant TextArea

Vous pouvez définir les paramètres de programmation suivants pour chaque occurrence du composant TextArea dans l'Inspecteur des propriétés ou l'Inspecteur des composants : `condenseWhite`, `editable`, `hortizontalScrollPolicy`, `maxChars`, `restrict`, `text`, `verticalScrollPolicy` et `wordwrap`. Chacun de ces paramètres possède une propriété ActionScript correspondante du même nom. Pour plus d'informations sur les valeurs possibles de ces paramètres, consultez la classe TextArea dans le *Guide de référence du langage et des composants ActionScript 3.0*.

L'aperçu en direct des occurrences TextArea reflète les modifications apportées aux paramètres dans l'Inspecteur des propriétés ou l'Inspecteur des composants pendant la programmation. Si une barre de défilement s'avère nécessaire, elle apparaît lors d'un aperçu en direct, mais ne fonctionnera pas. Lors d'un aperçu en direct, il n'est pas possible de sélectionner du texte et vous ne pouvez pas entrer de texte dans l'occurrence du composant sur la scène.

Vous pouvez contrôler ces options et d'autres options du composant TextArea à l'aide des propriétés, méthodes et événements d'ActionScript. Pour plus d'informations, consultez la classe TextArea dans le *Guide de référence du langage et des composants ActionScript 3.0*.

Création d'une application avec le composant TextArea

La procédure suivante explique comment ajouter un composant TextArea à une application pendant la programmation. L'exemple configure un gestionnaire d'événements `focusOut` sur l'occurrence TextArea qui vérifie que l'utilisateur a tapé des données dans la zone de texte avant de donner le focus à une autre partie de l'interface.

Pour créer une application avec le composant TextArea :

1. Créez un nouveau fichier Flash (ActionScript 3.0).
2. Faites glisser un composant TextArea du panneau Composants vers la scène et nommez l'occurrence **aTa**. Conservez les valeurs par défaut de ses paramètres.
3. Faites glisser un deuxième composant TextArea du panneau Composants vers la scène, placez-le sous le premier et nommez l'occurrence **bTa**. Conservez les valeurs par défaut de ses paramètres.

4. Ouvrez le panneau Actions, sélectionnez l'image 1 dans le scénario principal et entrez le code ActionScript suivant :

```
import flash.events.FocusEvent;

aTa.restrict = "a-z,'\" \"";
aTa.addEventListener(Event.CHANGE,changeHandler);
aTa.addEventListener(FocusEvent.KEY_FOCUS_CHANGE, k_m_fHandler);
aTa.addEventListener(FocusEvent.MOUSE_FOCUS_CHANGE, k_m_fHandler);

function changeHandler(ch_evt:Event):void {
  bTa.text = aTa.text;
}
function k_m_fHandler(kmf_event:FocusEvent):void {
  kmf_event.preventDefault();
}
```

   Cet exemple limite les caractères que vous pouvez entrer dans la zone de texte `aTa` aux minuscules, à la virgule, à l'apostrophe et aux espaces. Il configure également les gestionnaires d'événements des événements `change`, `KEY_FOCUS_CHANGE` et `MOUSE_FOCUS_CHANGE` de la zone de texte `aTa`. La fonction `changeHandler()` permet d'afficher automatiquement le texte que vous entrez dans la zone de texte `aTa` dans la zone de texte `bTa` en affectant `aTa.text` à `bTa.text` sur chaque événement `change`. La fonction `k_m_fHandler()` des événements `KEY_FOCUS_CHANGE` et `MOUSE_FOCUS_CHANGE` vous empêche d'appuyer sur la touche de tabulation afin de passer au champ suivant sans entrer de texte. Elle y parvient en bloquant le comportement par défaut.

5. Choisissez Contrôle> Tester l'animation.

   Si vous appuyez sur la touche de tabulation pour déplacer le focus vers la deuxième zone de texte sans entrer de texte, un message d'erreur doit s'afficher et le focus doit être renvoyé à la première zone de texte. Lorsque vous entrez du texte dans la première zone de texte, celui-ci est dupliqué dans la deuxième zone de texte.

L'exemple suivant crée un composant TextArea à l'aide d'ActionScript. Il définit la propriété `condenseWhite` sur `true` pour comprimer les espaces blancs et affecte du texte à la propriété `htmlText` afin de tirer parti des attributs de formatage de texte HTML.

Pour créer une occurrence du composant TextArea à l'aide d'ActionScript :

1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Faites glisser le composant TextArea vers le panneau Bibliothèque.

3. Ouvrez le panneau Actions, sélectionnez l'image 1 dans le scénario principal et entrez le code ActionScript suivant :

```
import fl.controls.TextArea;

var aTa:TextArea = new TextArea();

aTa.move(100,100);
aTa.setSize(200, 200);
aTa.condenseWhite = true;
aTa.htmlText = '      <b>Lorem ipsum dolor</b> sit amet, consectetuer
  adipiscing elit. <u>Vivamus quis nisl vel tortor nonummy vulputate.</
  u> Quisque sit amet eros sed purus euismod tempor. Morbi tempor. <font
  color="#FF0000">Class aptent taciti sociosqu ad litora torquent per
  conubia nostra, per inceptos hymenaeos.</font> Curabitur diam.
  Suspendisse at purus in ipsum volutpat viverra. Nulla pellentesque
  libero id libero.';
addChild(aTa);
```

Cet exemple utilise la propriété `htmlText` pour appliquer les attributs HTML gras et souligné à un bloc de texte et l'afficher dans la zone de texte `a_ta`. L'exemple définit également la propriété `condenseWhite` sur `true` pour comprimer les espaces blancs dans le bloc de texte. La méthode `setSize()` définit la hauteur et la largeur de la zone de texte, tandis que la méthode `move()` définit sa position. La méthode `addChild()` ajoute l'occurrence TextArea sur la scène.

4. Choisissez Contrôle> Tester l'animation.

Utilisation du composant TextInput

TextInput est un composant à une seule ligne qui renvoie à la ligne automatiquement l'objet TextField ActionScript natif. Si vous avez besoin d'un champ de texte multiligne, utilisez le composant Composant TextArea. Par exemple, vous pouvez utiliser un composant TextInput en tant que champ de mot de passe dans un formulaire. Vous pouvez également définir un écouteur qui vérifie si le champ comporte suffisamment de caractères lorsque l'utilisateur sort du champ. Cet écouteur peut afficher un message d'erreur indiquant que l'utilisateur n'a pas entré le nombre de caractères adéquat.

Vous pouvez définir la propriété `textFormat` à l'aide de la méthode `setStyle()` pour modifier le style du texte qui apparaît dans une occurrence TextInput. Un composant TextInput peut également être formaté en HTML ou en tant que champ de mot de passe masquant le texte.

Interaction de l'utilisateur avec le composant TextInput

Un composant TextInput peut être activé ou désactivé dans une application. Lorsqu'il est désactivé, il ne reçoit pas les informations en provenance de la souris ou du clavier. Lorsqu'il est activé, il suit les mêmes règles de focus, de sélection et de navigation qu'un objet TextField ActionScript. Lorsqu'une occurrence TextInput a le focus, vous pouvez également utiliser les touches suivantes pour le contrôler.

Touche	Description
Touches fléchées	Déplacent le point d'insertion d'un caractère vers la gauche et vers la droite.
Maj+Tab	Place le focus sur l'objet précédent.
Tab	Place le focus sur l'objet suivant.

Pour plus d'informations sur le contrôle du focus, consultez la classe FocusManager dans le *Guide de référence du langage et des composants ActionScript 3.0* et la section « Utilisation de FocusManager », à la page 56.

L'aperçu en direct des occurrences TextInput reflète les modifications apportées aux paramètres dans l'Inspecteur des propriétés ou l'Inspecteur des composants pendant la programmation. Lors d'un aperçu en direct, il n'est pas possible de sélectionner du texte et vous ne pouvez pas entrer de texte dans l'occurrence du composant sur la scène.

Lorsque vous ajoutez un composant TextInput à une application, vous pouvez utiliser le panneau Accessibilité pour le rendre accessible aux lecteurs d'écran.

Paramètres du composant TextInput

Vous pouvez définir les paramètres de programmation suivants pour chaque occurrence du composant TextInput dans
l'Inspecteur des propriétés ou l'Inspecteur des composants : `editable`, `displayAsPassword`, `maxChars`, `restrict` et `text`. Chacun de ces paramètres possède une propriété ActionScript correspondante du même nom. Pour plus d'informations sur les valeurs possibles de ces paramètres, consultez la classe TextInput dans le *Guide de référence du langage et des composants ActionScript 3.0*.

Vous pouvez contrôler ces options et d'autres options du composant TextInput à l'aide des propriétés, méthodes et événements d'ActionScript. Pour plus d'informations, consultez la classe TextInput dans le *Guide de référence du langage et des composants ActionScript 3.0*.

Création d'une application avec le composant TextInput

La procédure suivante explique comment ajouter un composant TextInput à une application. L'exemple utilise deux champs TextInput pour recevoir et confirmer un mot de passe. Il utilise un écouteur d'événements pour s'assurer que huit caractères au minimum ont été entrés et que le texte des deux champs correspond.

Pour créer une application avec le composant TextInput :

1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Faites glisser un composant Label du panneau Composants vers la scène et spécifiez les valeurs suivantes pour celui-ci dans l'Inspecteur des propriétés :
   - Entrez le nom d'occurrence **pwdLabel**.
   - Entrez la valeur **100** pour W.
   - Entrez la valeur **50** pour X.
   - Entrez la valeur **150** pour Y.
   - Dans la section Paramètres, entrez la valeur **Password:** pour le paramètre de texte.
3. Faites glisser un deuxième composant Label du panneau Composants vers la scène et spécifiez les valeurs suivantes pour celui-ci :
   - Entrez le nom d'occurrence **confirmLabel**.
   - Entrez la valeur **100** pour W.
   - Entrez la valeur **50** pour X.
   - Entrez la valeur **200** pour Y.
   - Dans la section Paramètres, entrez la valeur **Confirm Password:** pour le paramètre de texte.
4. Faites glisser un composant TextInput du panneau Composants vers la scène et spécifiez les valeurs suivantes pour celui-ci :
   - Entrez le nom d'occurrence **pwdTi**.
   - Entrez la valeur **150** pour W.
   - Entrez la valeur **190** pour X.
   - Entrez la valeur **150** pour Y.
   - Dans la section Paramètres, double-cliquez sur la valeur du paramètre `displayAsPassword` et sélectionnez **true**. La valeur entrée dans le champ texte est ainsi masquée avec des astérisques.
5. Faites glisser un deuxième composant TextInput du panneau Composants vers la scène et spécifiez les valeurs suivantes pour celui-ci :

- Entrez le nom d'occurrence **confirmTi**.
- Entrez la valeur **150** pour W.
- Entrez la valeur **190** pour X.
- Entrez la valeur **200** pour Y.
- Dans la section Paramètres, double-cliquez sur la valeur du paramètre `displayAsPassword` et sélectionnez **true**. La valeur entrée dans le champ texte est ainsi masquée avec des astérisques.

6. Ouvrez le panneau Actions, sélectionnez l'image 1 dans le scénario principal et entrez le code ActionScript suivant :

```
function tiListener(evt_obj:Event){
  if(confirmTi.text != pwdTi.text || confirmTi.length < 8)
  {
   trace("Password is incorrect. Please reenter it.");
  }
  else {
   trace("Your password is: " + confirmTi.text);
  }
}
confirmTi.addEventListener("enter", tiListener);
```

   Ce code configure un gestionnaire d'événements `enter` sur l'occurrence de composant TextInput appelée `confirmTi`. Si les deux mots de passe ne correspondent pas ou si l'utilisateur saisit moins de huit caractères, l'exemple affiche ce message : « Mot de passe incorrect. Veuillez le saisir de nouveau ». Si les mots de passe contiennent au moins huit caractères et correspondent, l'exemple affiche la valeur entrée dans le panneau Sortie.

7. Choisissez Contrôle> Tester l'animation.

L'exemple suivant crée un composant TextInput à l'aide d'ActionScript. Il crée également un composant Label qu'il utilise pour inviter l'utilisateur à entrer son nom. L'exemple définit la propriété `restrict` du composant de manière à autoriser uniquement les majuscules et les minuscules, les points et les espaces. Il crée également un objet TextFormat qu'il utilise pour formater le texte des composants Label et TextInput.

Pour créer une occurrence du composant TextInput à l'aide d'ActionScript :

1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Faites glisser un composant TextInput du panneau Composants vers le panneau Bibliothèque.
3. Faites glisser un composant Label du panneau Composants vers le panneau Bibliothèque.

4. Ouvrez le panneau Actions, sélectionnez l'image 1 dans le scénario principal et entrez le code ActionScript suivant :

```
import fl.controls.Label;
import fl.controls.TextInput;

var nameLabel:Label = new Label();
var nameTi:TextInput = new TextInput();
var tf:TextFormat = new TextFormat();

addChild(nameLabel);
addChild(nameTi);

nameTi.restrict = "A-Z .a-z";

tf.font = "Georgia";
tf.color = 0x0000CC;
tf.size = 16;

nameLabel.text = "Name: "
nameLabel.setSize(50, 25);
nameLabel.move(100,100);
nameLabel.setStyle("textFormat", tf);
nameTi.move(160, 100);
nameTi.setSize(200, 25);
nameTi.setStyle("textFormat", tf);
```

5. Choisissez Contrôle > Tester l'animation pour exécuter l'application.

Utilisation du composant TileList

Le composant TileList comprend une liste de lignes et de colonnes accompagnant les données attribuées par un fournisseur de données. Un *élément* fait référence à une unité de données stockées dans une cellule du composant TileList. Un élément provenant du fournisseur de données possède généralement une propriété `label` et une propriété `source`. La propriété `label` identifie le contenu à afficher dans une cellule et la propriété `source` fournit sa valeur.

Vous pouvez créer une occurrence de tableau ou en récupérer une sur le serveur. Le composant TileList dispose de méthodes agissant comme proxy pour son fournisseur de données, telles que les méthodes `addItem()` et `removeItem()`. Si vous ne procurez aucun fournisseur de données externe à la liste, ces méthodes créent automatiquement une occurrence de DataProvider, exposée par le biais de `List.dataProvider`.

Interaction de l'utilisateur avec le composant TileList

Le composant TileList restitue chaque cellule à l'aide d'un objet Sprite implémentant l'interface ICellRenderer. Vous pouvez spécifier cet objet de rendu à l'aide de la propriété `cellRenderer` de l'objet TileList. L'objet CellRenderer par défaut du composant TileList est le composant ImageCell, qui affiche une image (classe, bitmap, occurrence ou URL) et une étiquette facultative. L'étiquette constitue une seule ligne, toujours alignée au bas de la cellule. Vous pouvez faire défiler un composant TileList dans une seule direction.

Lorsqu'une occurrence TileList a le focus, vous pouvez également utiliser les touches suivantes pour accéder à ses éléments :

Touche	Description
Flèche vers le haut et flèche vers le bas	Vous permettent de vous déplacer vers le haut et vers le bas dans une colonne. Si la propriété `allowMultipleSelection` a la valeur `true`, vous pouvez utiliser ces touches conjointement avec la touche Maj pour sélectionner plusieurs cellules.
Flèche gauche et flèche droite	Vous permettent de vous déplacer vers la gauche ou vers la droite sur une ligne. Si la propriété `allowMultipleSelection` a la valeur `true`, vous pouvez utiliser ces touches conjointement avec la touche Maj pour sélectionner plusieurs cellules.
Origine	Sélectionne la première cellule d'un composant TileList. Si la propriété `allowMultipleSelection` a la valeur `true`, maintenez enfoncée la touche Maj et appuyez sur la touche Origine pour sélectionner toutes les cellules de votre sélection actuelle dans la première cellule.
Fin	Sélectionne la dernière cellule d'un composant TileList. Si la propriété `allowMultipleSelection` a la valeur `true`, maintenez enfoncée la touche Maj et appuyez sur la touche Fin pour sélectionner toutes les cellules de votre sélection actuelle dans la dernière cellule.
Ctrl	Si la propriété `allowMultipleSelection` est définie sur `true`, vous permet de sélectionner plusieurs cellules, dans un ordre quelconque.

Lorsque vous ajoutez le composant TileList à une application, vous pouvez le rendre accessible à un lecteur d'écran en ajoutant les lignes de code ActionScript suivantes :

```
import fl.accessibility.TileListAccImpl;

TileListAccImpl.enableAccessibility();
```

Quel que soit le nombre d'occurrences d'un composant, l'activation de son accessibilité ne se fait qu'une fois. Pour plus d'informations, consultez le Chapitre 18, « Création de contenu accessible » du guide *Utilisation de Flash*.

Paramètres du composant TileList

Dans l'Inspecteur des propriétés ou l'Inspecteur des composants, vous pouvez définir les paramètres de programmation suivants pour chaque occurrence du composant TileList : `allowMultipleSelection`, `columnCount`, `columnWidth`, `dataProvider`, `direction`, `horizontalScrollLineSize`, `horizontalScrollPageSize`, `labels`, `rowCount`, `rowHeight`, `ScrollPolicy`, `verticalScrollLineSize` et `verticalScrollPageSize`. Chacun de ces paramètres possède une propriété ActionScript correspondante du même nom. Pour plus d'informations sur l'utilisation du paramètre dataProvider, consultez la section « Utilisation du paramètre dataProvider », à la page 60.

Vous pouvez rédiger du code ActionScript afin de définir d'autres options pour les occurrences TileList à l'aide de ses méthodes, propriétés et événements. Pour plus d'informations, consultez la classe TileList dans le *Guide de référence du langage et des composants ActionScript 3.0*.

Création d'une application avec le composant TileList

Cet exemple utilise des clips pour affecter un tableau de couleurs à un composant TileList.

Pour créer une application avec le composant TileList :

1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Faites glisser un composant TileList sur la scène et nommez l'occurrence **aTl**.
3. Ouvrez le panneau Actions, sélectionnez l'image 1 dans le scénario principal et entrez le code ActionScript suivant :

```
import fl.data.DataProvider;
import flash.display.DisplayObject;

var aBoxes:Array = new Array();
var i:uint = 0;
var colors:Array = new Array(0x00000, 0xFF0000, 0x0000CC, 0x00CC00,
  0xFFFF00);
var colorNames:Array = new Array("Midnight", "Cranberry", "Sky",
  "Forest", "July");
var dp:DataProvider = new DataProvider();
for(i=0; i < colors.length; i++) {
  aBoxes[i] = new MovieClip();
  drawBox(aBoxes[i], colors[i]);// draw box w next color in array
  dp.addItem( {label:colorNames[i], source:aBoxes[i]} );
}
```

```
aTl.dataProvider = dp;
aTl.columnWidth = 110;
aTl.rowHeight = 130;
aTl.setSize(280,150);
aTl.move(150, 150);
aTl.setStyle("contentPadding", 5);

function drawBox(box:MovieClip,color:uint):void {
       box.graphics.beginFill(color, 1.0);
       box.graphics.drawRect(0, 0, 100, 100);
       box.graphics.endFill();
}
```

4. Sélectionnez Contrôle > Tester l'animation pour tester l'application.

L'exemple suivant crée de manière dynamique une occurrence TileList et y ajoute des occurrences des composants ColorPicker, ComboBox, NumericStepper et CheckBox. Il crée un tableau contenant les étiquettes et les noms du composant à afficher, et affecte le tableau (`dp`) à la propriété `dataProvider` du composant TileList. Il utilise les propriétés `columnWidth` et `rowHeight`, et la méthode `setSize()` pour disposer le composant TileList, la méthode `move()` pour le positionner sur la scène, le style `contentPadding` pour insérer des espaces entre les bordures de l'occurrence TileList et son contenu et la méthode `sortItemsOn()` pour trier le contenu en fonction de ses étiquettes.

Pour créer un composant TileList à l'aide d'ActionScript :

1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Faites glisser les composants suivants du panneau Composants vers le panneau Bibliothèque : ColorPicker, ComboBox, NumericStepper, CheckBox et TileList.
3. Ouvrez le panneau Actions, sélectionnez l'image 1 dans le scénario principal et entrez le code ActionScript suivant :

```
import fl.controls.CheckBox;
import fl.controls.ColorPicker;
import fl.controls.ComboBox;
import fl.controls.NumericStepper;
import fl.controls.TileList;
import fl.data.DataProvider;

var aCp:ColorPicker = new ColorPicker();
var aCb:ComboBox = new ComboBox();
var aNs:NumericStepper = new NumericStepper();
var aCh:CheckBox = new CheckBox();
var aTl:TileList = new TileList();
```

```
var dp:Array = [
    {label:"ColorPicker", source:aCp},
    {label:"ComboBox", source:aCb},
    {label:"NumericStepper", source:aNs},
    {label:"CheckBox", source:aCh},
];
aTl.dataProvider = new DataProvider(dp);
aTl.columnWidth = 110;
aTl.rowHeight = 100;
aTl.setSize(280,130);
aTl.move(150, 150);
aTl.setStyle("contentPadding", 5);
aTl.sortItemsOn("label");
addChild(aTl);
```

4. Sélectionnez Contrôle > Tester l'animation pour tester l'application.

Utilisation du composant UILoader

Le composant UILoader est un conteneur qui peut afficher des fichiers SWF, JPEG, JPEG progressifs, PNG et GIF. Vous pouvez utiliser un composant UILoader chaque fois que vous devez récupérer du contenu depuis un emplacement distant et le placer dans une application Flash. Par exemple, vous pouvez utiliser un composant UILoader pour ajouter un logo d'entreprise (fichier JPEG) dans un formulaire. Vous pouvez également utiliser le composant UILoader dans une application qui affiche des photos. Utilisez la méthode `load()` pour charger du contenu, la propriété `percentLoaded` afin de déterminer la quantité de contenu chargé et l'événement `complete` pour déterminer le moment de la fin du chargement.

Vous pouvez redimensionner le contenu du composant UILoader ou le composant UILoader lui-même pour l'adapter à la taille du contenu. Par défaut, le contenu est dimensionné pour s'ajuster au composant UILoader. Vous pouvez également charger du contenu à l'exécution et surveiller la progression du chargement (même si une fois que le contenu est chargé, il est mis en mémoire cache et la progression passe donc rapidement à 100 %). Si vous spécifiez un emplacement lors du chargement du contenu dans le composant UILoader, vous devez spécifier les valeurs 0, 0 (coordonnées X et Y).

Interaction de l'utilisateur avec le composant UILoader

Un composant UILoader ne peut recevoir le focus. Cependant, le contenu chargé dans le composant UILoader peut accepter le focus et avoir ses propres interactions de focus. Pour plus d'informations sur le contrôle du focus, consultez la classe FocusManager dans le *Guide de référence du langage et des composants ActionScript 3.0* et la section « Utilisation de FocusManager », à la page 56.

Paramètres du composant UILoader

Dans l'Inspecteur des propriétés ou l'Inspecteur des composants, vous pouvez définir les paramètres de programmation suivants pour chaque occurrence du composant UILoader : `autoLoad`, `maintainAspectRatio`, `source` et `scaleContent`. Chacun de ces paramètres possède une propriété ActionScript correspondante du même nom.

L'aperçu en direct des occurrences UILoader reflète les modifications apportées aux paramètres dans l'Inspecteur des propriétés ou l'Inspecteur des composants pendant la programmation.

ActionScript vous permet de définir des options supplémentaires pour les occurrences UILoader en utilisant ses méthodes, propriétés et événements. Pour plus d'informations, consultez la classe UILoader dans le *Guide de référence du langage et des composants ActionScript 3.0*.

Création d'une application avec le composant UILoader

La procédure suivante explique comment ajouter un composant UILoader à une application pendant la programmation. Dans cet exemple, le chargeur charge l'image GIF d'un logo.

Pour créer une application avec le composant UILoader :

1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Faites glisser un composant UILoader du panneau Composants vers la scène.
3. Dans l'Inspecteur des propriétés, entrez le nom d'occurrence **aUI**.
4. Sélectionnez le chargeur sur la scène et dans l'Inspecteur des composants, puis entrez **http://www.helpexamples.com/images/logo.gif** pour le paramètre `source`.

Cet exemple crée un composant UILoader à l'aide d'ActionScript et charge l'image GIF d'une fleur. Lorsque l'événement `complete` se produit, il affiche le nombre d'octets chargés dans le panneau Sortie.

Pour créer une occurrence du composant UILoader à l'aide d'ActionScript :

1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Faites glisser le composant UILoader du panneau Composants vers le panneau Bibliothèque.

3. Ouvrez le panneau Actions, sélectionnez l'image 1 dans le scénario principal et entrez le code ActionScript suivant :

```
import fl.containers.UILoader;

var aLoader:UILoader = new UILoader();
aLoader.source = "http://www.flash-mx.com/images/image1.jpg";
aLoader.scaleContent = false;
addChild(aLoader);

aLoader.addEventListener(Event.COMPLETE, completeHandler);
function completeHandler(event:Event) {
  trace("Number of bytes loaded: " + aLoader.bytesLoaded);
}
```

4. Choisissez Contrôle> Tester l'animation.

Utilisation du composant UIScrollBar

Le composant UIScrollBar permet d'ajouter une barre de défilement à un champ de texte. Vous pouvez ajouter une barre de défilement à un champ de texte pendant la programmation ou lors de l'exécution avec ActionScript. Pour utiliser le composant UIScrollBar, créez un champ texte sur la scène et faites glisser le composant UIScrollBar du panneau Composants vers n'importe quel quadrant du cadre de sélection du champ texte.

Si la longueur de la barre de défilement est inférieure à la taille combinée de ses flèches de défilement, elle ne s'affiche pas correctement. L'une des touches fléchées est masquée derrière l'autre. Flash ne fournit pas de détection des erreurs pour ceci. Dans ce cas, il peut être utile de masquer la barre de défilement avec ActionScript. Si la barre de défilement est dimensionnée de façon à ce qu'il y ait assez de place pour le curseur de défilement, Flash rend ce dernier invisible.

Le composant UIScrollBar fonctionne comme toute autre barre de défilement. Il contient des boutons fléchés aux deux extrémités et un rail et un curseur de défilement entre les deux. Il peut être associé à n'importe quel bord d'un champ de texte et utilisé verticalement et horizontalement.

Pour plus d'informations sur l'objet TextField, consultez la classe TextField dans le *Guide de référence du langage et des composants ActionScript 3.0*.

Interaction de l'utilisateur avec le composant UIScrollBar

Contrairement à de nombreux autres composants, le composant UIScrollBar peut recevoir des actions de la souris continues comme lorsque l'utilisateur maintient le bouton de la souris enfoncé plutôt que d'exiger des clics répétés.

Il n'existe pas d'interaction clavier avec le composant UIScrollBar.

Paramètres du composant UIScrollBar

Dans l'Inspecteur des propriétés ou l'Inspecteur des composants, vous pouvez définir les paramètres de programmation suivants pour chaque occurrence du composant UIScrollBar : `direction` et `scrollTargetName`. Chacun de ces paramètres possède une propriété ActionScript correspondante du même nom.

Vous pouvez rédiger du code ActionScript afin de définir d'autres options pour les occurrences UIScrollBar à l'aide des méthodes de classe, propriétés et événements. Pour plus d'informations, consultez la classe UIScrollBar dans le *Guide de référence du langage et des composants ActionScript 3.0*.

Création d'une application avec le composant UIScrollBar

La procédure suivante décrit l'ajout d'un composant UIScrollBar à une application au cours de la programmation.

Pour créer une application avec le composant UIScrollBar :

1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Créez un champ de texte dynamique suffisamment grand pour pouvoir contenir une ou deux lignes de texte et attribuez-lui le nom d'occurrence **myText** dans l'Inspecteur des propriétés.
3. Dans l'Inspecteur des propriétés, définissez le type de ligne du champ de saisie de texte sur Multiligne ou Multiligne sans retour si vous envisagez d'utiliser la barre de défilement horizontalement.

4. Ouvrez le panneau Actions, sélectionnez l'image 1 dans le scénario principal et entrez le code ActionScript suivant pour remplir la propriété `text` de manière à ce que l'utilisateur doive le faire défiler pour pouvoir l'afficher dans son intégralité :

```
myText.text="When the moon is in the seventh house and Jupiter aligns
  with Mars, then peace will guide the planet and love will rule the
  stars."
```

> **REMARQUE** Vérifiez que le champ de texte sur la scène est assez petit pour que vous deviez le faire défiler afin de visualiser tout le texte. Si ce n'est pas le cas, la barre de défilement ne s'affiche pas ou risque d'apparaître sur deux lignes sans poignée miniature (partie que vous faites glisser pour faire défiler le contenu).

5. Vérifiez que l'accrochage aux objets est activé (Affichage > Accrochage > Accrocher aux objets).
6. Faites glisser une occurrence de UIScrollBar du panneau Composants sur le champ de saisie de texte près du côté auquel vous souhaitez l'associer. Le champ de texte et le composant doivent se chevaucher lorsque vous relâchez la souris de façon à ce que le composant soit correctement lié au champ. Attribuez-lui le nom d'occurrence **mySb.**

   La propriété `scrollTargetName` du composant est renseignée automatiquement avec le nom de l'occurrence du champ de texte dans l'Inspecteur des propriétés et l'Inspecteur des composants. Si elle n'apparaît pas dans l'onglet Paramètres, vous n'avez peut-être pas suffisamment recouvert l'occurrence UIScrollBar.
7. Choisissez Contrôle> Tester l'animation.

Vous pouvez également créer une occurrence UIScrollBar à l'aide d'ActionScript et l'associer à un champ de texte lors de l'exécution. L'exemple suivant crée une occurrence UIScrollBar orientée horizontalement et l'associe au bas d'une occurrence de champ de texte nommée **myTxt**, chargée avec du texte d'une URL. L'exemple définit également la taille de la barre de défilement en fonction de la taille du champ de texte :

Pour créer une occurrence du composant UIScrollBar à l'aide d'ActionScript :

1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Faites glisser le composant ScrollBar vers le panneau Bibliothèque.
3. Ouvrez le panneau Actions, sélectionnez l'image 1 dans le scénario principal et entrez le code ActionScript suivant :

```
import flash.net.URLLoader;
import fl.controls.UIScrollBar;
import flash.events.Event;
```

```
var myTxt:TextField = new TextField();
myTxt.border = true;
myTxt.width = 200;
myTxt.height = 16;
myTxt.x = 200;
myTxt.y = 150;

var mySb:UIScrollBar = new UIScrollBar();
mySb.direction = "horizontal";
// Size it to match the text field.
mySb.setSize(myTxt.width, myTxt.height);

// Move it immediately below the text field.
mySb.move(myTxt.x, myTxt.height + myTxt.y);

// put them on the Stage
addChild(myTxt);
addChild(mySb);
// load text
var loader:URLLoader = new URLLoader();
var request:URLRequest = new URLRequest("http://www.helpexamples.com/
  flash/lorem.txt");
loader.load(request);
loader.addEventListener(Event.COMPLETE, loadcomplete);

function loadcomplete(event:Event) {
  // move loaded text to text field
  myTxt.text = loader.data;
  // Set myTxt as target for scroll bar.
  mySb.scrollTarget = myTxt;
}
```

4. Choisissez Contrôle> Tester l'animation.

CHAPITRE 4

Personnalisation des composants de l'interface utilisateur

4

Ce chapitre explique comment personnaliser les composants de l'interface utilisateur Flash ActionScript 3.0. Il inclut les rubriques suivantes :

Pour plus d'informations sur la personnalisation du composant FLVPlayback, consultez le Chapitre 5, « Utilisation du composant FLVPlayback ».

A propos de la personnalisation des composants de l'interface utilisateur

Vous pouvez personnaliser l'aspect des composants dans vos applications en modifiant l'un des éléments suivants, ou les deux :

Styles : chaque composant possède un ensemble de styles que vous pouvez définir pour spécifier les valeurs utilisées par Flash pour définir le rendu de l'aspect du composant. Les styles spécifient généralement les enveloppes et les icônes à utiliser pour un composant dans ses différents états, ainsi que les valeurs de formatage de texte et de marge intérieure à utiliser.

Enveloppes : une *enveloppe* comprend la collection de symboles constituant l'apparence graphique du composant dans un état donné. Tandis qu'un style spécifie l'enveloppe à utiliser, une enveloppe est un élément graphique utilisé par Flash pour dessiner le composant. *L'application d'une enveloppe* est le processus mis en œuvre pour changer l'aspect d'un composant en modifiant ou en remplaçant ses graphiques.

REMARQUE

L'aspect par défaut des composants ActionScript 3.0 peut être considéré comme étant un thème (Aeon Halo) mais ces enveloppes sont intégrées aux composants. Les composants ActionScript 3.0 ne prennent pas en charge les fichiers de thème externes, contrairement aux composants ActionScript 2.0.

Définition de styles

Les styles d'un composant spécifient généralement les valeurs des enveloppes, des icônes, de formatage de texte et de marge intérieure lorsque Flash dessine le composant dans ses divers états. Par exemple, Flash dessine un bouton avec une enveloppe spécifique pour afficher son état abaissé, qui se produit lorsque vous cliquez dessus à l'aide du bouton de la souris ; il utilise des enveloppes différentes pour afficher son état relevé ou normal. Il utilise également une enveloppe différente lorsque son état est désactivé ; cet état est généré en définissant la propriété `enabled` sur `false`.

Vous pouvez définir les styles des composants au niveau du document, de la classe et de l'occurrence. En outre, certaines propriétés de style peuvent être héritées d'un composant parent. Par exemple, le composant List hérite des styles ScrollBar hérités de BaseScrollPane.

Vous pouvez définir des styles pour personnaliser un composant de l'une des manières suivantes :

- Définir des styles pour une occurrence de composant.

  Vous pouvez modifier les propriétés de couleur et de texte d'une seule occurrence de composant. Cette méthode est efficace dans certaines situations, mais elle s'avère laborieuse si vous devez définir des propriétés individuelles pour chacun des composants d'un document.
- Définir des styles pour tous les composants d'un type donné dans un document.

  Si vous souhaitez appliquer un aspect cohérent à tous les composants d'un type donné, par exemple à tous les composants CheckBox ou Button d'un document, vous pouvez définir des styles au niveau du composant.

Les composants contenus héritent des valeurs des propriétés de style définies sur ces conteneurs.

Flash ne présente pas les modifications apportées aux propriétés de style lorsque vous affichez des composants sur la scène via la fonction Aperçu en direct.

Présentation des paramètres de style

Voici quelques points-clés relatifs à l'utilisation des styles :

Héritage : un composant enfant est conçu pour hériter d'un style du composant parent. Vous ne pouvez pas définir l'héritage des styles dans ActionScript.

Caractère prioritaire : si un style de composant est défini de plusieurs façons, Flash utilise le premier style qu'il rencontre selon l'ordre de priorité. Flash recherche les styles dans l'ordre suivant, jusqu'à la détection d'une valeur :

1. Flash cherche une propriété de style dans l'occurrence du composant.
2. Si le style fait partie des styles hérités, Flash recherche la valeur héritée dans la hiérarchie apparentée.
3. Flash recherche le style sur le composant.
4. Flash recherche une définition globale sur StyleManager.
5. Si la propriété n'est toujours pas définie, elle prend la valeur `undefined`.

Accès aux styles par défaut d'un composant

Vous pouvez accéder aux styles par défaut d'un composant à l'aide de la méthode statique `getDefaultStyles()` de la classe du composant. Par exemple, le code suivant extrait les styles par défaut du composant ComboBox et affiche les valeurs par défaut des propriétés `buttonWidth` et `downArrowDownSkin` :

```
import fl.controls.ComboBox;
var styleObj:Object = ComboBox.getDefaultStyles();
trace(styleObj.buttonWidth); // 21
trace(styleObj.downArrowDownSkin); // downArrowDownSkin
```

Définition et obtention de styles pour l'occurrence d'un composant

Toutes les occurrences du composant de l'interface utilisateur peuvent appeler les méthodes `setStyle()` et `getStyle()` directement pour définir ou récupérer un style. La syntaxe suivante définit un style et une valeur pour l'occurrence d'un composant :

```
instanceName.setStyle("styleName", value);
```

Cette syntaxe récupère un style pour l'occurrence d'un composant :

```
var a_style:Object = new Object();
a_style = instanceName.getStyle("styleName");
```

Notez que la méthode `getStyle()` renvoie le type Objet car il peut renvoyer plusieurs styles ayant des types de données différents. Par exemple, le code suivant définit le style de police d'une occurrence TextArea (`aTa`), puis le récupère à l'aide de la méthode `getStyle()`. L'exemple attribue la valeur renvoyée à un objet TextFormat pour l'affecter à une variable TextFormat. Sans l'attribution, le compilateur émettrait une erreur pour tentative de mise en correspondance forcée d'une variable Objet à une variable TextFormat.

```
import flash.text.TextFormat;

var tf:TextFormat = new TextFormat();
tf.font = "Georgia";
aTa.setStyle("textFormat",tf);
aTa.text = "Hello World!";
var aStyle:TextFormat = aTa.getStyle("textFormat") as TextFormat;
trace(aStyle.font);
```

Utilisation de l'objet TextFormat pour définir des propriétés de texte

Utilisez l'objet TextFormat pour formater le texte de l'occurrence d'un composant. L'objet TextFormat possède des propriétés qui vous permettent de spécifier des caractéristiques de texte telles que `gras`, `puce`, `couleur`, `police`, `italique`, `taille`, et bien d'autres encore. Vous pouvez définir ces propriétés dans l'objet TextFormat, puis appeler la méthode `setStyle()` afin de les appliquer à une occurrence de composant. Par exemple, le code suivant définit les propriétés `police`, `taille` et `gras` d'un objet TextFormat et les applique à une occurrence de bouton :

```
/* Create a new TextFormat object to set text formatting properties. */
var tf:TextFormat = new TextFormat();
tf.font = "Arial";
tf.size = 16;
tf.bold = true;
a_button.setStyle("textFormat", tf);
```

L'illustration suivante montre l'effet de ces paramètres sur un bouton incluant une étiquette Submit :

Les propriétés de style définies pour l'occurrence d'un composant via `setStyle()` sont prioritaires et remplacent tous les autres paramètres de style. Néanmoins, plus vous définissez de propriétés en utilisant `setStyle()` sur une seule occurrence de composant, plus le rendu du composant est lent au moment de l'exécution.

Définition d'un style pour toutes les occurrences d'un composant

Vous pouvez définir un style pour toutes les occurrences d'une classe de composant à l'aide de la méthode statique `setComponentStyle()` de la classe StyleManager. Par exemple, vous pouvez définir la couleur du texte sur rouge pour tous les boutons en commençant par faire glisser un bouton sur la scène, puis en ajoutant le code ActionScript suivant dans le panneau Actions de l'image 1 du scénario :

```
import fl.managers.StyleManager;
import fl.controls.Button;

var tf:TextFormat = new TextFormat();
tf.color = 0xFF0000;
StyleManager.setComponentStyle(Button, "textFormat", tf);
```

Tous les boutons que vous ajouterez par la suite sur la scène auront des étiquettes rouges.

Définition d'un style pour tous les composants

Vous pouvez définir un style pour tous les composants à l'aide de la méthode statique `setStyle()` de la classe StyleManager.

Pour définir un style pour tous les composants :

1. Faites glisser un composant List sur la scène et nommez l'occurrence **aList**.
2. Faites glisser un composant Button sur la scène et nommez l'occurrence **aButton**.
3. Appuyez sur **F9** ou sélectionnez Actions dans le menu Fenêtre pour ouvrir le panneau Actions, s'il n'est pas déjà ouvert, et entrez le code suivant dans l'image 1 du scénario pour définir la couleur du texte sur rouge pour tous les composants :

```
import fl.managers.StyleManager;

var tf:TextFormat = new TextFormat();
tf.color = 0xFF0000;
StyleManager.setStyle("textFormat", tf);
```

4. Ajoutez le code suivant dans le panneau Actions pour insérer du texte dans le composant List.

```
aList.addItem({label:"1956 Chevy (Cherry Red)", data:35000});
aList.addItem({label:"1966 Mustang (Classic)", data:27000});
aList.addItem({label:"1976 Volvo (Xcllnt Cond)", data:17000});
aList.allowMultipleSelection = true;
```

5. Sélectionnez Contrôle > Tester l'animation ou appuyez sur Ctrl+Entrée pour compiler le code et tester votre contenu. Le texte inclus dans l'étiquette du bouton et dans la liste doit être de couleur rouge.

A propos des enveloppes

L'aspect d'un composant est constitué d'éléments graphiques tels que le contour, la couleur de remplissage, les icônes et bien d'autres composants encore. Un composant ComboBox, par exemple, contient un composant List qui lui-même contient un composant ScrollBar. Les éléments graphiques réunis constituent l'aspect du composant ComboBox. Toutefois, l'aspect d'un composant change en fonction de son état actuel. Par exemple, un composant CheckBox, sans son étiquette, a l'aspect suivant lorsqu'il apparaît dans votre application :

Etat relevé normal d'un composant CheckBox

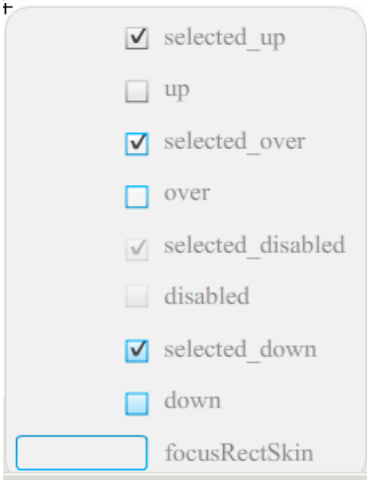
Si vous cliquez sur le bouton de la souris en le maintenant enfoncé sur le composant CheckBox, son aspect devient semblable à celui-ci :

Etat abaissé d'un composant CheckBox

Et lorsque vous relâchez le bouton de la souris, le composant CheckBox reprend son aspect d'origine mais inclut désormais une coche qui indique qu'il a été sélectionné.

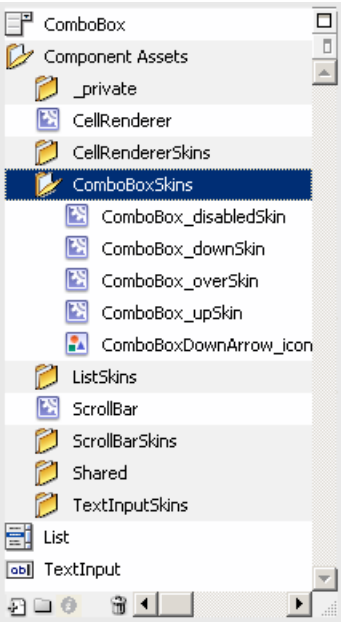
Etat sélectionné d'un composant CheckBox

Pour plus de clarté, les icônes qui représentent le composant dans ses divers états sont désignées comme étant ses *enveloppes.* Vous pouvez modifier l'aspect d'un composant indépendamment de son état en modifiant ses enveloppes dans Flash, comme vous le feriez avec tout autre symbole Flash. Vous pouvez accéder aux enveloppes d'un composant de deux façons. La façon la plus simple consiste à faire glisser le composant sur la scène, puis à double-cliquer dessus. Une palette des enveloppes du composant, semblable à celle-ci pour un composant CheckBox, s'ouvre.



Enveloppes d'un composant CheckBox

Vous pouvez également accéder aux enveloppes d'un composant individuellement dans le panneau Bibliothèque. Lorsque vous faites glisser un composant sur la scène, vous le copier également dans la bibliothèque avec un dossier de ses actifs et des autres composants qu'il contient. Par exemple, si vous faites glisser un composant ComboBox sur la scène, les éléments suivants apparaissent dans le panneau Bibliothèque :



Panneau Bibliothèque du composant ComboBox

Outre le composant ComboBox, le panneau Bibliothèque contient les composants List, ScrollBar et TextInput, intégrés au composant ComboBox, ainsi qu'un dossier incluant les enveloppes de chacun de ces composants et un dossier des actifs partagés contenant les éléments partagés par ces composants. Vous pouvez modifier les enveloppes de ces composants en ouvrant le dossier des enveloppes correspondant (ComboBoxSkins, ListSkins, ScrollBarSkins ou TextInputSkins) et en double-cliquant sur l'icône de l'enveloppe que vous souhaitez modifier. Par exemple, si vous double-cliquez sur ComboBox_downSkin, l'enveloppe s'ouvre en mode d'édition de symbole, comme illustré ci-dessous :

ComboBox_downSkin

Création d'une enveloppe

Si vous souhaitez créer un nouvel aspect pour un composant dans votre document, modifiez les enveloppes du composant de manière à changer leur apparence. Pour accéder aux enveloppes d'un composant, il vous suffit de double-cliquer sur celui-ci sur la scène pour ouvrir la palette de ses enveloppes. Ensuite, double-cliquez sur l'enveloppe à modifier pour l'ouvrir en mode d'édition de symbole. Par exemple, double-cliquez sur le composant TextArea qui se trouve sur la scène pour ouvrir ses actifs en mode d'édition de symbole. Définissez le contrôle de zoom sur 400 %, ou plus si vous le souhaitez, et modifiez le symbole de manière à changer son aspect. Lorsque vous aurez terminé, la modification apportée affectera toutes les occurrences du composant se trouvant dans le document. Une autre méthode consiste à double-cliquer sur une enveloppe particulière dans le panneau Bibliothèque pour l'ouvrir sur la scène en mode d'édition de symbole.

Vous pouvez modifier les enveloppes de composant de l'une des manières suivantes :

- Créer une nouvelle enveloppe pour toutes les occurrences
- Créer de nouvelles enveloppes pour certaines occurrences

Création d'une enveloppe pour toutes les occurrences

Lorsque vous modifiez l'enveloppe d'un composant, vous modifiez par défaut l'aspect du composant de toutes ses occurrences dans le document. Si vous souhaitez créer différents aspects pour le même composant, vous devez dupliquer les enveloppes à modifier et leur attribuer des noms différents, les modifier, puis définir les styles appropriés pour les appliquer. Pour plus d'informations, consultez la section « Création d'enveloppes pour certaines occurrences », à la page 161.

Ce chapitre décrit comment modifier une ou plusieurs enveloppes pour chacun des composants de l'interface utilisateur. Si vous suivez l'une de ces procédures pour modifier une ou plusieurs enveloppes d'un composant de l'interface utilisateur, la modification affectera toutes les occurrences du document.

Création d'enveloppes pour certaines occurrences

Vous pouvez créer une enveloppe pour certaines occurrences d'un composant à l'aide de la procédure générale suivante :

- Sélectionnez l'enveloppe dans le dossier Assets du composant du panneau Bibliothèque.
- Dupliquez l'enveloppe et affectez-lui un nom de classe unique.
- Modifiez l'enveloppe pour lui attribuer l'aspect souhaité.

- Appelez la méthode `setStyle()` de l'occurrence de composant de manière à affecter la nouvelle enveloppe au style de l'enveloppe.

La procédure suivante crée une nouvelle enveloppe selectedDownSkin pour l'une des deux occurrences de bouton.

Pour créer une nouvelle enveloppe selectedDownSkin pour un bouton :

1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Faites glisser deux boutons du panneau Composants sur la scène et nommez leurs occurrences **aButton** et **bButton**.
3. Ouvrez le panneau Bibliothèque, puis les dossiers Component Assets et ButtonSkins qui s'y trouvent.
4. Cliquez sur l'enveloppe selectedDownSkin pour la sélectionner.
5. Cliquez avec le bouton droit pour ouvrir le menu contextuel et choisissez Dupliquer.
6. Dans la boîte de dialogue Dupliquer le symbole, attribuez un nom unique à la nouvelle enveloppe, par exemple **Button_mySelectedDownSkin**. Cliquez sur OK.
7. Dans le panneau Bibliothèque, ouvrez le dossier > Component Assets > ButtonSkins, sélectionnez Button_mySelectedDownSkin et cliquez sur le bouton droit de la souris pour ouvrir le menu contextuel. Sélectionnez Liaison pour ouvrir la boîte de dialogue Propriétés de liaison.
8. Cochez la case Exporter pour ActionScript. Ne décochez pas la case Exporter dans la première image et assurez-vous que le nom de classe est unique. Cliquez sur OK, puis sur OK de nouveau en réponse au message d'avertissement indiquant que la définition de classe est introuvable et qu'une nouvelle définition sera donc créée.
9. Double-cliquez sur l'enveloppe Button_mySelectedDownSkin dans le panneau Bibliothèque pour l'ouvrir en mode d'édition de symbole.
10. Cliquez sur le remplissage bleu au centre de l'enveloppe jusqu'à ce que la couleur apparaisse dans le sélecteur de couleur de remplissage de l'Inspecteur des propriétés. Cliquez sur le sélecteur de couleur et sélectionnez la couleur #00CC00 pour le remplissage de l'enveloppe.
11. Cliquez sur le bouton de retour figurant dans la partie gauche de la barre d'édition en haut de la scène pour revenir en mode d'édition de document.
12. Dans l'Inspecteur des propriétés, cliquez sur l'onglet Paramètres pour chaque bouton et définissez le paramètre toggle sur `true`.
13. Ajoutez le code suivant dans le panneau Actions, sur l'image 1 du scénario :

```
bButton.setStyle("selectedDownSkin", Button_mySelectedDownSkin);
bButton.setStyle("downSkin", Button_mySelectedDownSkin);
```

14. Choisissez Contrôle> Tester l'animation.
15. Cliquez sur chaque bouton. Notez que l'enveloppe abaissée (sélectionnée et déselectionnée) de l'objet bButton utilise le nouveau symbole d'enveloppe.

Personnalisation du composant Button

Vous pouvez transformer un composant Button horizontalement et verticalement pendant la programmation et lors de l'exécution. Lors de la programmation, choisissez le composant sur la scène et utilisez l'outil Transformer librement ou une commande de modification > transformation. Lors de l'exécution, utilisez la méthode `setSize()` ou toute propriété applicable de la classe Button, telle que `height` et `width`, `scaleX` et `scaleY`.

Le redimensionnement du bouton n'affecte pas la taille de l'icône ni celle de l'étiquette. Le cadre de sélection d'un bouton correspond à la bordure du bouton et désigne également la zone active de l'occurrence. Si vous augmentez la taille de l'occurrence, vous augmentez également celle de la zone active. Si le cadre de sélection est trop petit pour contenir l'étiquette, celle-ci est découpée à la bonne taille.

Si le bouton possède une icône plus grande que sa taille, l'icône dépasse les bordures du bouton.

Utilisation de styles avec le composant Button

Les styles d'un composant Button spécifient généralement les valeurs des enveloppes, des icônes, de formatage de texte et de marge intérieure d'un bouton lorsque le composant est dessiné dans ses divers états.

La procédure suivante place deux composants Button sur la scène et définit la propriété `emphasized` sur `true` pour les deux boutons lorsque l'utilisateur clique sur l'un d'entre eux. Elle définit également le style `emphasizedSkin` du deuxième composant Button sur `selectedOverSkin` lorsque l'utilisateur clique dessus de manière à ce que les deux boutons affichent des enveloppes différentes pour le même état.

Pour modifier le style emphasizedSkin d'un bouton :

1. Créez un fichier Flash (ActionScript 3.0).
2. Faites glisser deux composants Button sur la scène, l'un après l'autre, et donnez-leur les noms d'occurrence **aBtn** et **bBtn**. Dans l'onglet Paramètres de l'Inspecteur des propriétés, donnez-leur les étiquettes de composant Button A et Button B.

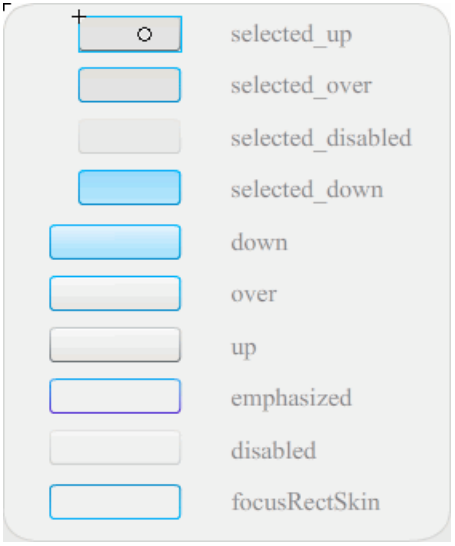
3. Ajoutez le code suivant dans le panneau Actions, sur l'image 1 du scénario :

```
bBtn.emphasized = true;
aBtn.emphasized = true;
bBtn.addEventListener(MouseEvent.CLICK, Btn_handler);
function Btn_handler(evt:MouseEvent):void {
  bBtn.setStyle("emphasizedSkin", "Button_selectedOverSkin");
}
```

4. Choisissez Contrôle> Tester l'animation.
5. Cliquez sur l'un des boutons pour visualiser l'effet du style `emphasizedSkin` sur chaque bouton.

Utilisation d'enveloppes avec le composant Button

Le composant Button utilise les enveloppes suivantes qui correspondent à ses différents états. Pour modifier une ou plusieurs enveloppes de manière à changer l'aspect du composant Button, double-cliquez sur l'occurrence de bouton sur la scène pour ouvrir la palette de ses enveloppes, comme illustré ci-dessous :

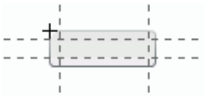


Enveloppes du composant Button

Si un bouton est activé, il affiche l'état survolé lorsque le pointeur de la souris est placé au-dessus. Le bouton reçoit le focus d'entrée et affiche l'état abaissé lorsque l'utilisateur clique dessus. Le bouton retrouve l'état survolé lorsque la souris est relâchée. Si le pointeur s'éloigne du bouton alors que le bouton de la souris est enfoncé, le bouton reprend son état d'origine. Si le paramètre toggle est défini sur `true`, l'état enfoncé s'affiche avec l'enveloppe selectedDownSkin, l'état relevé avec l'enveloppe selectedUpSkin et l'état survolé avec l'enveloppe selectedOverSkin.

Si un bouton est désactivé, il affiche son état désactivé, quelle que soit l'interaction de l'utilisateur.

Pour modifier l'une des enveloppes, double-cliquez dessus pour l'ouvrir en mode d'édition de symbole, comme illustré ci-dessous :



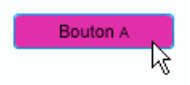
A ce stade, vous pouvez utiliser les outils de programmation Flash pour modifier l'enveloppe à votre guise.

La procédure suivante modifie la couleur de l'enveloppe selected_over du bouton.

Pour modifier la couleur de l'enveloppe selected_over du composant Button :

1. Créez un nouveau fichier Flash (ActionScript 3.0).
2. Faites glisser un composant Button du panneau Composants vers la scène. Dans l'onglet Paramètres, définissez le paramètre toggle sur `true`.
3. Double-cliquez sur le bouton pour ouvrir la palette de ses enveloppes.
4. Double-cliquez sur l'enveloppe selected_over pour l'ouvrir en mode d'édition de symbole.
5. Définissez le contrôle de zoom sur 400 % pour agrandir l'icône en vue de la modification.
6. Double-cliquez sur l'arrière-plan jusqu'à ce que sa couleur apparaisse dans le sélecteur de couleur de remplissage de l'Inspecteur des propriétés.
7. Sélectionnez la couleur #CC0099 dans le sélecteur de couleur de remplissage pour l'appliquer à l'arrière-plan de l'enveloppe selected_over.
8. Cliquez sur le bouton de retour figurant dans la partie gauche de la barre d'édition en haut de la scène pour revenir en mode d'édition de document.
9. Choisissez Contrôle> Tester l'animation.
10. Cliquez sur le bouton pour le placer dans l'état sélectionné.

Lorsque vous survolez le bouton à l'aide du pointeur de la souris, l'état selected_over doit apparaître tel qu'illustré ci-dessous.



Personnalisation du composant CheckBox

Vous pouvez transformer un composant CheckBox horizontalement et verticalement pendant la programmation et lors de l'exécution. Lors de la programmation, choisissez le composant sur la scène et utilisez l'outil Transformer librement ou une commande de modification > transformation. Lors de l'exécution, utilisez la méthode `setSize()` ou les propriétés applicables de la classe CheckBox. Par exemple, vous pouvez modifier la taille d'un composant CheckBox en définissant ses propriétés `height`, `width`, `scaleX` et `scaleY`. Le redimensionnement du composant CheckBox ne modifie pas la taille de l'étiquette ni de l'icône, mais uniquement la taille du cadre de sélection.

Le cadre de sélection d'une occurrence CheckBox est invisible et désigne également la zone active de l'occurrence. Si vous augmentez la taille de l'occurrence, vous augmentez également celle de la zone active. Si le cadre de sélection est trop petit pour contenir l'étiquette, celle-ci est découpée à la bonne taille.

Utilisation de styles avec le composant CheckBox

Vous pouvez définir des propriétés de style pour modifier l'aspect d'une occurrence CheckBox. Par exemple, la procédure suivante modifie la taille et la couleur d'une étiquette CheckBox.

Pour modifier la taille et la couleur d'une étiquette CheckBox :

1. Faites glisser le composant CheckBox du panneau Composants vers la scène et nommez l'occurrence **myCb.**
2. Dans l'Inspecteur des propriétés, cliquez sur l'onglet Paramètres et entrez la valeur suivante pour le paramètre d'étiquette : **Inférieur à 500 € ?**

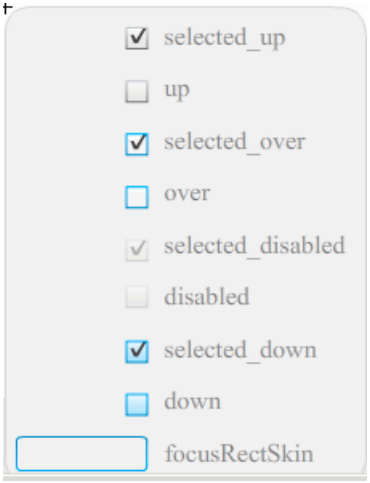
3. Sur l'image 1 du scénario principal, entrez le code suivant dans le panneau Actions :

```
var myTf:TextFormat = new TextFormat();
myCb.setSize(150, 22);
myTf.size = 16;
myTf.color = 0xFF0000;
myCb.setStyle("textFormat", myTf);
```

Pour plus d'informations, consultez la section « Définition de styles », à la page 154. Pour plus d'informations sur la définition des propriétés de style afin de modifier les icônes et les enveloppes du composant, consultez la section « Création d'une enveloppe », à la page 161 et « Utilisation d'enveloppes avec le composant CheckBox », à la page 167.

Utilisation d'enveloppes avec le composant CheckBox

Le composant CheckBox possède les enveloppes suivantes que vous pouvez modifier pour changer son apparence.



Enveloppes du composant CheckBox

Cet exemple modifie la couleur de contour et la couleur d'arrière-plan du composant dans ses états `up` et `selectedUp`. Vous effectuez des étapes similaires pour modifier les enveloppes des autres états.

Pour personnaliser les enveloppes du composant CheckBox :

1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Faites glisser le composant CheckBox sur la scène, ce qui le place également dans la bibliothèque avec un dossier de ses actifs.
3. Double-cliquez sur le composant CheckBox qui se trouve sur la scène pour ouvrir son panneau d'icônes d'enveloppe.
4. Double-cliquez sur l'icône selected_up pour l'ouvrir en mode d'édition de symbole.
5. Définissez le contrôle de zoom sur 800 % pour agrandir l'icône en vue de la modification.
6. Cliquez sur la bordure du composant CheckBox pour le sélectionner. Dans l'Inspecteur des propriétés, utilisez le sélecteur de couleur de remplissage pour sélectionner la couleur #0033FF et l'appliquer à la bordure.
7. Double-cliquez sur l'arrière-plan du composant CheckBox pour le sélectionner et utilisez de nouveau le sélecteur de couleur de remplissage pour définir la couleur de l'arrière-plan sur #00CCFF.
8. Répétez les étapes 4 à 8 pour l'enveloppe du composant CheckBox dont l'état est Relevé.
9. Choisissez Contrôle> Tester l'animation.

Personnalisation du composant ColorPicker

Vous pouvez uniquement redimensionner un composant ColorPicker par l'intermédiaire de ses styles : `swatchWidth`, `swatchHeight`, `backgroundPadding`, `textFieldWidth` et `textFieldHeight`. Si vous essayez de modifier la taille du composant ColorPicker à l'aide de l'outil Transformer ou d'ActionScript via la méthode `setSize()`, ou via les propriétés `width`, `height`, `scaleX` ou `scaleY`, ces valeurs sont ignorées lorsque vous créez le fichier SWF et le composant ColorPicker s'affiche selon sa taille par défaut. L'arrière-plan de la palette est redimensionnée en fonction du nombre de colonnes défini via la méthode `setStyle()` pour le style `columnCount`. Le nombre par défaut de colonnes est 18. Vous pouvez définir les couleurs personnalisées sur 1024 et la palette est redimensionnée verticalement en fonction du nombre de nuanciers.

Utilisation de styles avec le composant ColorPicker

Vous pouvez définir plusieurs styles pour modifier l'aspect du composant ColorPicker. Par exemple, la procédure suivante modifie le nombre de colonnes (`columnCount`) dans le composant ColorPicker pour le définir sur 12, modifie la hauteur (`swatchHeight`) et la largeur (`swatchWidth`) des nuanciers, et modifie la marge intérieure du champ texte (`textPadding`) et de l'arrière-plan (`backgroundPadding`).

Pour modifier l'aspect du composant ColorPicker à l'aide des styles :

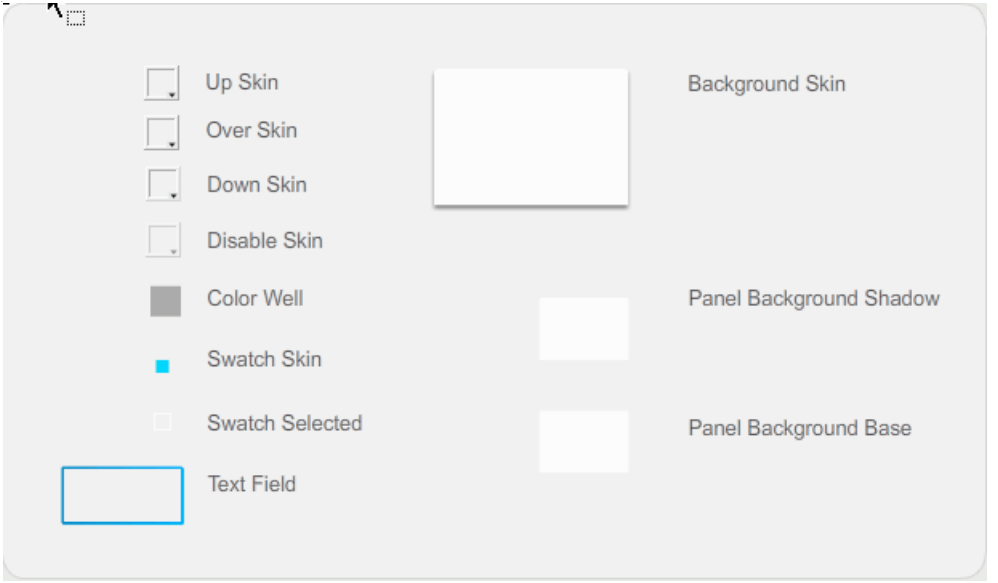
1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Faites glisser le composant ColorPicker sur la scène et nommez son occurrence **aCp**.
3. Ouvrez le panneau Actions, sélectionnez l'image 1 dans le scénario principal et entrez le code suivant :

```
aCp.setStyle("columnCount", 12);
aCp.setStyle("swatchWidth", 8);
aCp.setStyle("swatchHeight", 12);
aCp.setStyle("swatchPadding", 2);
aCp.setStyle("backgroundPadding", 3);
aCp.setStyle("textPadding", 7);
```

4. Choisissez Contrôle> Tester l'animation.
5. Cliquez sur le composant ColorPicker pour l'ouvrir et découvrir la façon dont ces paramètres ont modifié son apparence.

Utilisation d'enveloppes avec le composant ColorPicker

Le composant ColorPicker utilise les enveloppes suivantes pour représenter ses états visuels.



Enveloppes du composant ColorPicker

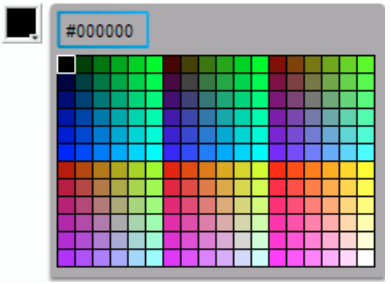
Vous pouvez modifier la couleur de l'enveloppe d'arrière-plan pour modifier la couleur d'arrière-plan de la palette.

Pour modifier la couleur de l'enveloppe d'arrière-plan :

1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Faites glisser le composant ColorPicker sur la scène.
3. Double-cliquez dessus pour ouvrir la palette de ses enveloppes.
4. Double-cliquez sur l'enveloppe d'arrière-plan jusqu'à ce qu'elle soit sélectionnée : le sélecteur de couleur de remplissage apparaît alors dans l'Inspecteur des propriétés.
5. Sélectionnez la couleur #999999 à l'aide du sélecteur de couleur de remplissage pour l'appliquer à l'enveloppe d'arrière-plan.

6. Cliquez sur le bouton de retour figurant dans la partie gauche de la barre d'édition en haut de la scène pour revenir en mode d'édition de document.
7. Choisissez Contrôle> Tester l'animation.

   Lorsque vous cliquez sur le composant ColorPicker, l'arrière-plan de la palette doit apparaître en gris, comme indiqué dans l'illustration suivante.



Personnalisation du composant ComboBox

Vous pouvez transformer un composant ComboBox horizontalement et verticalement pendant la programmation et lors de l'exécution. Lors de la programmation, choisissez le composant sur la scène et utilisez l'outil Transformer librement ou une commande de modification > transformation. Lors de l'exécution, utilisez la méthode `setSize()` ou les propriétés applicables de la classe ComboBox, telles que `height`, `width`, `scaleX` et `scaleY`.

Le composant ComboBox est redimensionné en fonction de la largeur et de la hauteur spécifiées. La liste est redimensionnée en fonction de la largeur du composant, sauf si la propriété `dropdownWidth` a été définie.

Si le texte est trop long pour tenir dans le composant ComboBox, il est tronqué. Vous devez redimensionner le composant ComboBox et définir la propriété `dropdownWidth` pour contenir le texte.

Utilisation de styles avec le composant ComboBox

Vous pouvez définir des propriétés de style pour modifier l'aspect d'un composant ComboBox. Les styles spécifient les valeurs des enveloppes, de la classe CellRenderer, de la marge intérieure et de la largeur de bouton du composant. L'exemple suivant définit les styles `buttonWidth` et `textPadding`. Le style `buttonWidth` définit la largeur de la zone active du bouton et est en vigueur lorsque le composant ComboBox est modifiable et que vous pouvez uniquement appuyer sur le bouton pour ouvrir la liste déroulante. Le style textPadding spécifie la quantité d'espace répartie entre la bordure extérieure du champ texte et le texte. Il est particulièrement utile pour centrer le texte verticalement dans le champ texte si vous agrandissez le composant ComboBox. Sinon, le texte pourrait apparaître dans la partie supérieure du champ texte.

Pour définir les styles sur le composant ComboBox :

1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Faites glisser le composant ComboBox sur la scène et nommez son occurrence **aCb**.
3. Ouvrez le panneau Actions, sélectionnez l'image 1 dans le scénario principal et entrez le code suivant :

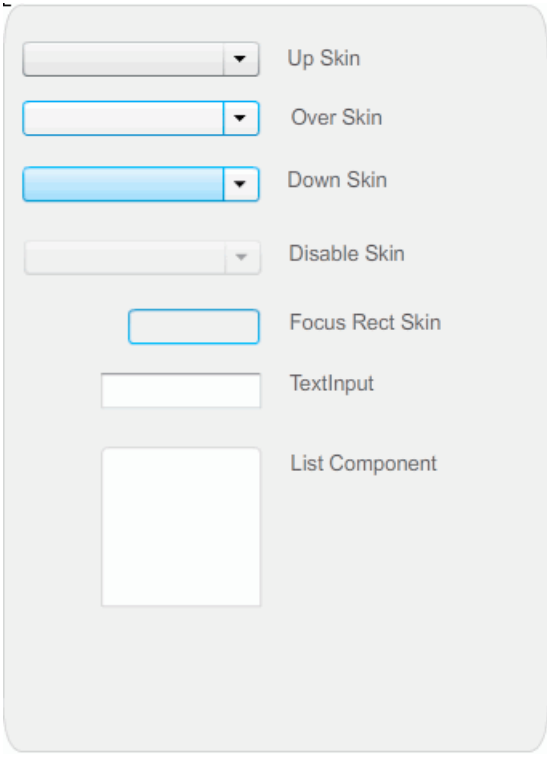
```
import fl.data.DataProvider;

aCb.setSize(150, 35);
aCb.setStyle("textPadding", 10);
aCb.setStyle("buttonWidth", 10);
aCb.editable = true;

var items:Array = [
{label:"San Francisco", data:"601 Townsend St."},
{label:"San Jose", data:"345 Park Ave."},
{label:"San Diego", data:"10590 West Ocean Air Drive, Suite 100"},
{label:"Santa Rosa", data:"2235 Mercury Way, Suite 105"},
{label:"San Luis Obispo", data:"3220 South Higuera Street, Suite 311"}
];
aCb.dataProvider = new DataProvider(items);
```

4. Choisissez Contrôle> Tester l'animation.

   Sachez que la zone du bouton sur laquelle vous pouvez cliquer pour ouvrir la liste déroulante représente uniquement une zone étroite sur le côté droit. Notez également que le texte est centré verticalement dans le champ texte. Vous pouvez essayer d'exécuter l'exemple sans les deux instructions `setStyle()` pour visualiser leur effet.

Utilisation d'enveloppes avec le composant ComboBox

Le composant ComboBox utilise les enveloppes suivantes pour représenter ses états visuels :



Enveloppes du composant ComboBox

Vous pouvez modifier la couleur de l'enveloppe dont l'état est Relevé de manière à modifier la couleur du composant dans son état inactif sur la scène.

Pour modifier la couleur de l'enveloppe d'arrière-plan :

1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Faites glisser le composant ComboBox sur la scène.
3. Double-cliquez dessus pour ouvrir la palette de ses enveloppes.
4. Double-cliquez sur l'enveloppe dont l'état est Relevé jusqu'à ce qu'elle soit sélectionnée et ouverte en vue de la modification.
5. Définissez le contrôle de zoom sur 400 %.

6. Cliquez sur le centre de l'enveloppe jusqu'à ce que sa couleur apparaisse dans le sélecteur de couleur de remplissage de l'Inspecteur des propriétés.
7. Sélectionnez la couleur #33FF99 à l'aide du sélecteur de couleur de remplissage pour l'appliquer à l'enveloppe dont l'état est Relevé.
8. Cliquez sur le bouton de retour figurant dans la partie gauche de la barre d'édition en haut de la scène pour revenir en mode d'édition de document.
9. Choisissez Contrôle> Tester l'animation.

   Le composant ComboBox doit apparaître sur la scène, comme illustré ci-dessous.

Personnalisation du composant DataGrid

Vous pouvez transformer un composant DataGrid horizontalement et verticalement durant la programmation et à l'exécution. Lors de la programmation, choisissez le composant sur la scène et utilisez l'outil Transformer librement ou une commande de modification > transformation. Lors de l'exécution, utilisez la méthode `setSize()` ou les propriétés applicables, telles que `width`, `height`, `scaleX` et `scaleY`. Lorsque aucune barre de défilement horizontale n'est présente, la largeur des colonnes s'ajuste proportionnellement. Si une modification de la taille des colonnes (et donc des cellules) intervient, le texte des cellules peut être tronqué.

Utilisation de styles avec le composant DataGrid

Vous pouvez définir des propriétés de style afin de modifier l'aspect d'un composant DataGrid. Le composant DataGrid hérite des styles du composant List. (Consultez la section « Utilisation des styles avec le composant List », page 766.

Définition des styles pour une colonne individuelle

Un objet DataGrid peut disposer de plusieurs colonnes ; vous pouvez spécifier différentes classes CellRenderer pour chacune d'entre elles. Chaque colonne d'un objet DataGrid est représentée par un objet DataGridColumn ; la classe DataGridColumn inclut une propriété `cellRenderer` pour laquelle vous pouvez définir la classe CellRenderer de la colonne.

Pour créer une colonne multiligne dans un composant DataGrid :

1. Créez un nouveau fichier Flash (ActionScript 3.0).
2. Faites glisser le composant DataGrid vers le panneau Bibliothèque.

3. Ajoutez le code suivant dans le panneau Actions, sur l'image 1 du scénario. Ce code crée un composant DataGrid avec une longue chaîne de texte dans la troisième colonne. A la fin, il définit la propriété `cellRenderer` de la colonne sur le nom d'un objet CellRenderer qui exécute le rendu d'une cellule à plusieurs lignes.

```
/* This is a simple cell renderer example.  It invokes
the MultiLineCell cell renderer to display a multiple
line text field in one of a DataGrid's columns. */

import fl.controls.DataGrid;
import fl.controls.dataGridClasses.DataGridColumn;
import fl.data.DataProvider;
import fl.controls.ScrollPolicy;

// Create a new DataGrid component instance.
var aDg:DataGrid = new DataGrid();

var aLongString:String = "An example of a cell renderer class that
  displays a multiple line TextField"
var myDP:Array = new Array();
myDP = [{firstName:"Winston", lastName:"Elstad", note:aLongString,
  item:100},
  {firstName:"Ric", lastName:"Dietrich", note:aLongString, item:101},
  {firstName:"Ewing", lastName:"Canepa", note:aLongString, item:102},
  {firstName:"Kevin", lastName:"Wade", note:aLongString, item:103},
  {firstName:"Kimberly", lastName:"Dietrich", note:aLongString,
  item:104},
  {firstName:"AJ", lastName:"Bilow", note:aLongString, item:105},
  {firstName:"Chuck", lastName:"Yushan", note:aLongString, item:106},
  {firstName:"John", lastName:"Roo", note:aLongString, item:107},
];

// Assign the data provider to the DataGrid to populate it.
// Note: This has to be done before applying the cellRenderers.
aDg.dataProvider = new DataProvider(myDP);

/* Set some basic grid properties.
Note: The data grid's row height should reflect
the number of lines you expect to show in the multiline cell.
The cell renderer wil size to the row height.
About 40 for 2 lines or 60 for 3 lines.*/

aDg.columns = ["firstName", "lastName", "note", "item"];
aDg.setSize(430,190);
aDg.move(40,40);
aDg.rowHeight = 40;  // Allows for 2 lines of text at default text size.
aDg.columns[0].width = 70;
aDg.columns[1].width = 70;
aDg.columns[2].width = 230;
aDg.columns[3].width = 60;
```

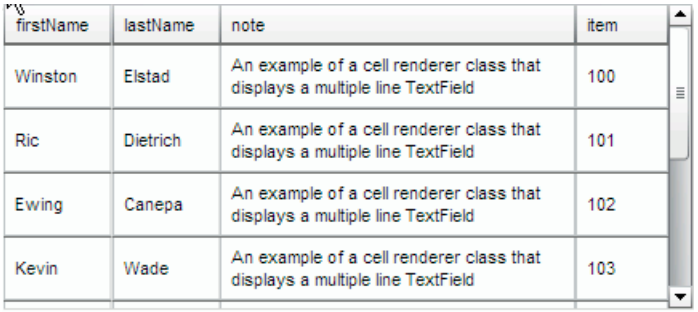
```
aDg.resizableColumns = true;
aDg.verticalScrollPolicy = ScrollPolicy.AUTO;
addChild(aDg);
// Assign cellRenderers.
var col3:DataGridColumn = new DataGridColumn();
col3 = aDg.getColumnAt(2);
col3.cellRenderer = MultiLineCell;
```

4. Enregistrez le fichier FLA sous MultiLineGrid.fla.
5. Créez un fichier ActionScript.
6. Copiez le code ActionScript suivant dans la fenêtre de script :

```
package {


  import fl.controls.listClasses.CellRenderer;

  public class MultiLineCell extends CellRenderer
  {

    public function MultiLineCell()
    {
      textField.wordWrap = true;
      textField.autoSize = "left";
    }
    override protected function drawLayout():void {
      textField.width = this.width;
      super.drawLayout();
    }
  }
}
```

7. Enregistrez le fichier ActionScript sous MultiLineCell.as dans le dossier dans lequel vous avez enregistré le fichier MultiLineGrid.fla.
8. Revenez dans l'application MultiLineGrid.fla et choisissez Contrôle > Tester l'animation.

   Le composant DataGrid doit avoir l'aspect suivant :



Définition de styles d'en-tête

Vous pouvez définir le style de texte d'une ligne d'en-tête à l'aide du style `headerTextFormat`. L'exemple suivant utilise l'objet TextFormat pour définir le style `headerTextFormat` sur la police Arial, la couleur rouge, la taille de police 14 et le style italique.

Pour définir le style headerTextFormat d'un composant DataGrid :

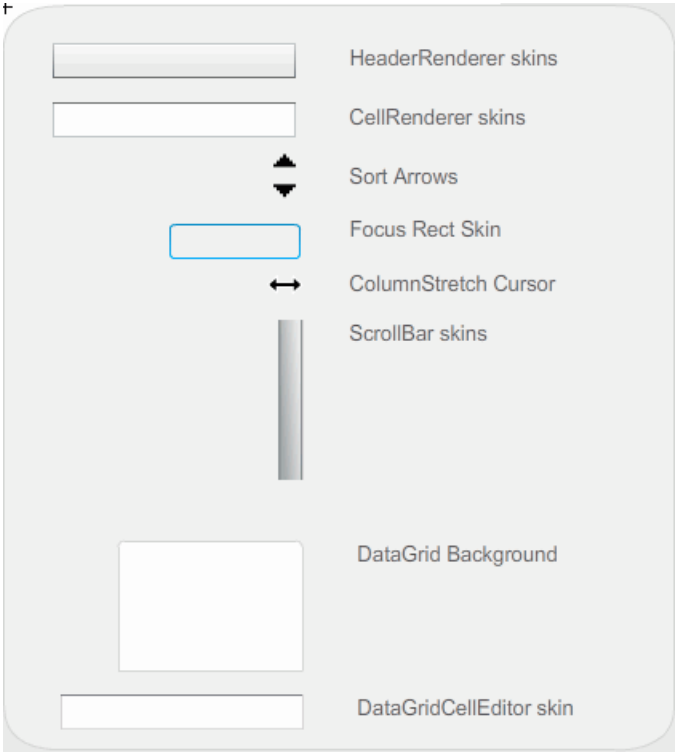
1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Faites glisser le composant DataGrid sur la scène et nommez l'occurrence **aDg**.
3. Ouvrez le panneau Actions, sélectionnez l'image 1 dans le scénario principal et entrez le code suivant :

```
import fl.data.DataProvider;
import fl.controls.dataGridClasses.DataGridColumn;

var myDP:Array = new Array();
myDP = [{FirstName:"Winston", LastName:"Elstad"},
  {FirstName:"Ric", LastName:"Dietrich"},
  {FirstName:"Ewing", LastName:"Canepa"},
  {FirstName:"Kevin", LastName:"Wade"},
  {FirstName:"Kimberly", LastName:"Dietrich"},
  {FirstName:"AJ", LastName:"Bilow"},
  {FirstName:"Chuck", LastName:"Yushan"},
  {FirstName:"John", LastName:"Roo"},
];

// Assign the data provider to the DataGrid to populate it.
// Note: This has to be done before applying the cellRenderers.
aDg.dataProvider = new DataProvider(myDP);
aDg.setSize(160,190);
aDg.move(40,40);
aDg.columns[0].width = 80;
aDg.columns[1].width = 80;
var tf:TextFormat = new TextFormat();
tf.size = 14;
tf.color = 0xff0000;
tf.italic = true;
tf.font = "Arial"
aDg.setStyle("headerTextFormat", tf);
```

4. Choisissez Contrôle > Tester l'animation pour exécuter l'application.

Utilisation d'enveloppes avec le composant DataGrid

Le composant DataGrid utilise les enveloppes suivantes pour représenter ses états visuels :



Enveloppes du composant DataGrid

L'enveloppe CellRenderer est l'enveloppe utilisée pour les cellules du corps du composant DataGrid, tandis que l'enveloppe HeaderRenderer est utilisée pour la ligne d'en-tête. La procédure suivante modifie la couleur d'arrière-plan de la ligne d'en-tête mais vous pouvez utiliser le même processus pour modifier la couleur d'arrière-plan des cellules du corps du composant DataGrid en modifiant l'enveloppe CellRenderer.

Pour modifier la couleur d'arrière-plan de la ligne d'en-tête du composant DataGrid :

1. Créez un nouveau fichier Flash (ActionScript 3.0).
2. Faites glisser le composant DataGrid sur la scène et nommez l'occurrence **aDg**.
3. Double-cliquez sur le composant pour ouvrir la palette de ses enveloppes.

4. Définissez le contrôle de zoom sur 400 % pour agrandir les icônes en vue de la modification.
5. Double-cliquez sur l'enveloppe HeaderRenderer pour ouvrir la palette de ses enveloppes.
6. Double-cliquez sur l'enveloppe Up_Skin pour l'ouvrir en mode d'édition de symbole et cliquez sur son arrière-plan jusqu'à ce qu'il soit sélectionné et que le sélecteur de couleur de remplissage apparaisse dans l'Inspecteur des propriétés.
7. Sélectionnez la couleur #00CC00 dans le sélecteur de couleur de remplissage pour l'appliquer à l'arrière-plan de l'enveloppe Up_Skin HeaderRenderer.
8. Cliquez sur le bouton de retour figurant dans la partie gauche de la barre d'édition en haut de la scène pour revenir en mode d'édition de document.
9. Ajoutez le code suivant dans le panneau Actions, sur l'image 1 du scénario pour ajouter des données au composant DataGrid :

```
import fl.data.DataProvider;

bldRosterGrid(aDg);
var aRoster:Array = new Array();
aRoster = [
     {Name:"Wilma Carter",  Home: "Redlands, CA"},
     {Name:"Sue Pennypacker",  Home: "Athens, GA"},
     {Name:"Jill Smithfield",  Home: "Spokane, WA"},
     {Name:"Shirley Goth", Home: "Carson, NV"},
     {Name:"Jennifer Dunbar",  Home: "Seaside, CA"}
];
aDg.dataProvider = new DataProvider(aRoster);
function bldRosterGrid(dg:DataGrid){
  dg.setSize(400, 130);
  dg.columns = ["Name", "Home"];
  dg.move(50,50);
  dg.columns[0].width = 120;
  dg.columns[1].width = 120;
};
```

10. Sélectionnez Contrôle > Tester l'animation pour tester l'application.

Le composant DataGrid doit apparaître comme illustré ci-dessous, l'arrière-plan de la ligne d'en-tête devant apparaître en vert.

Name	Home
Wilma Carter	Redlands, CA
Sue Pennypacker	Athens, GA
Jill Smithfield	Spokane, WA
Shirley Goth	Carson, NV
Jennifer Dunbar	Seaside, CA

Personnalisation du composant Label

Vous pouvez transformer un composant Label horizontalement et verticalement pendant la programmation et lors de l'exécution. Lors de la programmation, choisissez le composant sur la scène et utilisez l'outil Transformer librement ou une commande de modification > transformation. Vous pouvez également définir le paramètre de programmation `autoSize` ; la définition de ce paramètre ne change pas le cadre de sélection dans l'aperçu en direct, mais l'étiquette est redimensionnée. L'étiquette est redimensionnée en fonction du paramètre `wordwrap`. Si le paramètre a la valeur `true`, l'étiquette est redimensionnée verticalement pour contenir le texte. Si le paramètre a la valeur `false`, l'étiquette est redimensionnée horizontalement. Lors de l'exécution, utilisez la méthode `setSize()`. Pour plus d'informations, consultez la méthode `Label.setSize()` et la propriété `Label.autoSize` dans le *Guide de référence du langage et des composants ActionScript 3.0*. Voir aussi « Création d'une application avec le composant Label », à la page 106.

Utilisation de styles avec le composant Label

Vous pouvez définir des propriétés de style afin de modifier l'aspect d'une occurrence d'étiquette. Tout le texte contenu dans une occurrence de composant Label doit partager le même style. Le composant Label dispose d'un style `textFormat`, qui possède les mêmes attributs que l'objet TextFormat et qui vous permet de définir les mêmes propriétés pour le contenu de `Label.text` que pour un objet TextField Flash ordinaire. L'exemple suivant définit la couleur du texte d'une étiquette sur rouge.

Pour modifier la couleur du texte d'une étiquette :

1. Faites glisser le composant Label du panneau Composants vers la scène et nommez l'occurrence `a_label`.
2. Cliquez sur l'onglet Paramètres et remplacez la valeur de la propriété de texte par le texte :

   Color me red
3. Sélectionnez l'image 1 dans le scénario principal, ouvrez le panneau Actions et saisissez le code suivant :

```
/* Create a new TextFormat object, which allows you to set multiple text
   properties at a time. */

var tf:TextFormat = new TextFormat();
tf.color = 0xFF0000;
/* Apply this specific text format (red text) to the Label instance. */
a_label.setStyle("textFormat", tf);
```

4. Choisissez Contrôle> Tester l'animation.

Pour plus d'informations sur les styles d'étiquette, consultez la classe Label dans le *Guide de référence du langage et des composants ActionScript 3.0*.

Utilisation d'enveloppes avec le composant Label

Le composant Label ne présente aucun élément visuel susceptible de recevoir des enveloppes.

Personnalisation du composant List

Vous pouvez transformer un composant List horizontalement et verticalement pendant la programmation et lors de l'exécution. Lors de la programmation, choisissez le composant sur la scène et utilisez l'outil Transformer librement ou une commande de modification > transformation. Lors de l'exécution, utilisez la méthode `setSize()` et les propriétés applicables de la classe List, telles que `height`, `width`, `scaleX` et `scaleY`.

Lorsqu'une liste est redimensionnée, ses lignes diminuent horizontalement, ce qui tronque leur texte. Verticalement, la liste ajoute ou supprime le nombre de lignes approprié. Les barres de défilement sont positionnées automatiquement si nécessaire.

Utilisation de styles avec le composant List

Vous pouvez définir des propriétés de style afin de modifier l'aspect d'un composant List. Les styles spécifient les valeurs des enveloppes et de la marge intérieure du composant lorsque celui-ci est tracé.

Les divers styles d'enveloppe vous permettent de spécifier différentes classes à utiliser pour l'enveloppe. Pour plus d'informations sur l'utilisation des styles d'enveloppe, consultez la section « A propos des enveloppes », à la page 158.

La procédure suivante définit la valeur du style `contentPadding` pour le composant List. Notez que la valeur de ce paramètre est soustraite de la taille du composant List pour définir la marge intérieure autour du contenu de sorte que vous devrez peut-être augmenter la taille du composant List afin d'éviter que le texte qui y est inclus ne soit recadré.

Pour définir le style contentPadding pour le composant List :

1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Faites glisser le composant List du panneau Composants vers la scène et nommez l'occurrence **aList**.

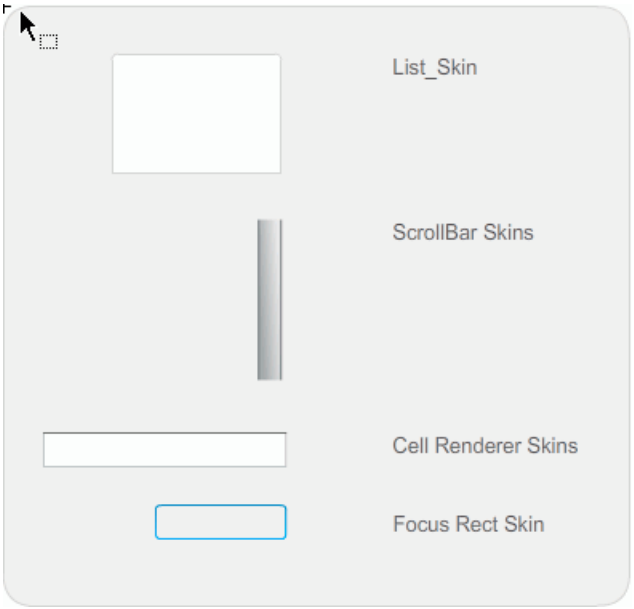
3. Sélectionnez l'image 1 dans le scénario principal, ouvrez le panneau Actions, puis entrez le code suivant, qui définit le style contentPadding et ajoute des données au composant List :

```
aList.setStyle("contentPadding", 5);
aList.setSize(145, 200);
aList.addItem({label:"1956 Chevy (Cherry Red)", data:35000});
aList.addItem({label:"1966 Mustang (Classic)", data:27000});
aList.addItem({label:"1976 Volvo (Xcllnt Cond)", data:17000});
aList.rowCount = aList.length;
```

4. Choisissez Contrôle> Tester l'animation.

Utilisation d'enveloppes avec le composant List

Le composant List utilise les enveloppes suivantes pour représenter ses états visuels :

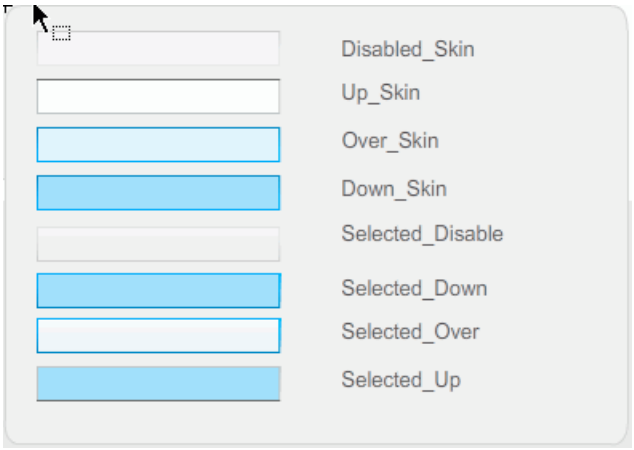


Enveloppes du composant List

Pour plus d'informations sur l'application d'enveloppes au composant ScrollBar, consultez la section « Personnalisation du composant UIScrollBar », à la page 204. Pour plus d'informations sur l'application d'enveloppes à l'enveloppe Focus Rect, consultez la section « Personnalisation du composant TextArea », à la page 196.

REMARQUE	La modification de l'enveloppe ScrollBar dans un composant affecte tous les autres composants qui utilisent le composant ScrollBar.

Double-cliquez sur l'enveloppe CellRenderer pour ouvrir une deuxième palette d'enveloppes pour les différents états d'une cellule List.



Enveloppes CellRenderer du composant List

Vous pouvez modifier l'aspect des cellules du composant List en modifiant ces enveloppes. La procédure suivante modifie la couleur de l'enveloppe dont l'état est Relevé de manière à modifier l'aspect du composant List dans son état inactif normal.

Pour modifier la couleur de l'enveloppe Up_Skin CellRenderer du composant List :

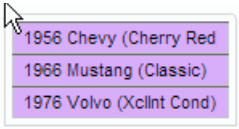
1. Créez un nouveau document de fichier Flash (ActionScript 3.0).
2. Faites glisser le composant List du panneau Composants vers la scène et nommez l'occurrence **aList**.
3. Double-cliquez sur le composant List pour ouvrir la palette de ses enveloppes.
4. Double-cliquez sur l'enveloppe CellRenderer pour ouvrir la palette des enveloppes CellRenderer.

5. Double-cliquez sur l'enveloppe Up_Skin pour l'ouvrir en vue de la modification.
6. Cliquez sur la zone de remplissage de l'enveloppe pour la sélectionner. Le sélecteur de couleur de remplissage incluant la couleur de remplissage actuelle de l'enveloppe doit apparaître dans l'Inspecteur des propriétés.
7. Sélectionnez la couleur #CC66FF dans le sélecteur de couleur de remplissage pour l'appliquer à l'enveloppe Up_Skin.
8. Cliquez sur le bouton de retour figurant dans la partie gauche de la barre d'édition en haut de la scène pour revenir en mode d'édition de document.
9. Ajoutez le code suivant dans le panneau Actions, sur l'image 1 du scénario pour ajouter des données au composant List :

```
aList.setStyle("contentPadding", 5);
aList.setSize(145, 200);
aList.addItem({label:"1956 Chevy (Cherry Red)", data:35000});
aList.addItem({label:"1966 Mustang (Classic)", data:27000});
aList.addItem({label:"1976 Volvo (Xcllnt Cond)", data:17000});
aList.rowCount = aList.length;
```

10. Choisissez Contrôle> Tester l'animation.

    Le composant List doit apparaître tel qu'illustré ci-dessous :



    La structure d'image résulte du paramétrage du style `contentPadding`.

Personnalisation du composant NumericStepper

Vous pouvez transformer un composant NumericStepper horizontalement et verticalement pendant la programmation et lors de l'exécution. Lors de la programmation, choisissez le composant sur la scène et utilisez l'outil Transformer librement ou une commande de modification > transformation. Lors de l'exécution, utilisez la méthode `setSize()` ou toutes les propriétés et méthodes applicables de la classe NumericStepper, telles que `width`, `height`, `scaleX` et `scaleY`.

Le redimensionnement du composant NumericStepper ne modifie pas la largeur des boutons fléchés vers le haut et le bas. Si l'incrémenteur est redimensionné au-delà de la hauteur par défaut, le comportement par défaut place les boutons fléchés en haut et en bas du composant. Sinon, la mise à l'échelle à 9 découpes détermine la façon dont les boutons sont dessinés. Les boutons fléchés apparaissent toujours à droite de la zone de texte.

Utilisation de styles avec le composant NumericStepper

Vous pouvez définir les propriétés de style du composant NumericStepper pour modifier son aspect. Les styles spécifient les valeurs des enveloppes, de la marge intérieure et du format de texte du composant lorsque celui-ci est tracé. Le style `textFormat` vous permet de modifier la taille et l'aspect de la valeur du composant NumericStepper. Les divers styles d'enveloppe vous permettent de spécifier différentes classes à utiliser pour l'enveloppe. Pour plus d'informations sur l'utilisation des styles d'enveloppe, consultez la section « A propos des enveloppes », à la page 158.

Pour modifier l'aspect de la valeur du composant Numeric Stepper :

Cette procédure utilise le style `textFormat` pour modifier l'aspect de la valeur affichée par le composant NumericStepper.

1. Créez un nouveau fichier Flash (ActionScript 3.0).
2. Faites glisser le composant NumericStepper du panneau Composants vers la scène et nommez l'occurrence **myNs**.
3. Ajoutez le code suivant dans le panneau Actions, sur l'image 1 du scénario principal.

```
var tf:TextFormat = new TextFormat();
myNs.setSize(100, 50);
tf.color = 0x0000CC;
tf.size = 24;
tf.font = "Arial";
tf.align = "center";
myNs.setStyle("textFormat", tf);
```

4. Choisissez Contrôle> Tester l'animation.

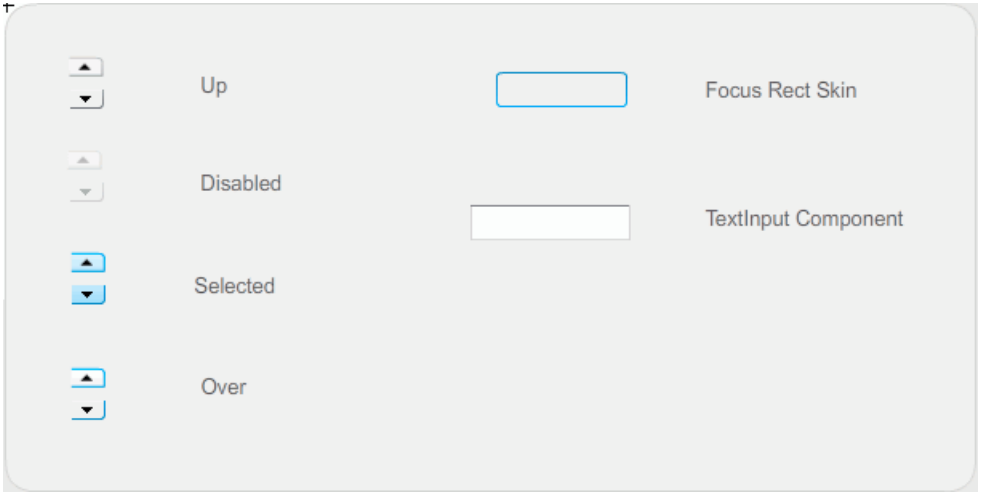
Utilisation d'enveloppes avec le composant NumericStepper

Le composant NumericStepper dispose d'enveloppes permettant de représenter les états Relevé, Abaissé, Désactivé et Sélectionné de ses boutons.

Si un incrémenteur est activé, les boutons fléchés vers le haut et vers le bas affichent leur état survolé lorsque le pointeur se déplace au-dessus d'eux. Les boutons affichent leur état enfoncé lorsque l'utilisateur clique dessus. Les boutons reviennent à l'état survolé lorsque le bouton de la souris est relâché. Si le pointeur s'éloigne des boutons alors que le bouton de la souris est enfoncé, les boutons reviennent à leur état original.

Si un incrémenteur est désactivé, il affiche son état désactivé, quelle que soit l'interaction de l'utilisateur.

Un composant NumericStepper possède les enveloppes suivantes :



Enveloppes du composant NumericStepper

Pour modifier la couleur d'arrière-plan du texte du composant NumericStepper et des boutons à l'état Relevé :

1. Créez un fichier FLA.
2. Faites glisser le composant NumericStepper sur la scène.
3. Définissez le contrôle de zoom sur 400 % pour agrandir l'image en vue de la modification.
4. Double-cliquez sur l'arrière-plan de l'enveloppe TextInput sur le panneau des enveloppes jusqu'à ce que vous développiez le niveau Groupe et que la couleur d'arrière-plan apparaisse dans le sélecteur de couleur de remplissage de l'Inspecteur des propriétés.

5. Dans le sélecteur de couleur de remplissage de l'Inspecteur des propriétés, sélectionnez la couleur #9999FF pour l'appliquer à l'arrière-plan de l'enveloppe TextInput.
6. Cliquez sur le bouton de retour figurant dans la partie gauche de la barre d'édition en haut de la scène pour revenir en mode d'édition de document.
7. Double-cliquez de nouveau sur le composant NumericStepper pour rouvrir la palette des enveloppes.
8. Double-cliquez sur l'arrière-plan du bouton fléché vers le haut dans le groupe Relevé jusqu'à ce que l'arrière-plan soit sélectionné et que sa couleur apparaisse dans le sélecteur de couleur de remplissage de l'Inspecteur des propriétés.
9. Sélectionnez la couleur #9966FF pour l'appliquer à l'arrière-plan du bouton fléché vers le haut.
10. Répétez les étapes 8 et 9 pour le bouton fléché vers le bas dans le groupe Relevé.
11. Choisissez Contrôle> Tester l'animation.

L'occurrence NumericStepper doit apparaître telle qu'illustrée ci-dessous :

Personnalisation du composant ProgressBar

Vous pouvez transformer un composant ProgressBar horizontalement et verticalement pendant la programmation et lors de l'exécution. Lors de la programmation, choisissez le composant sur la scène et utilisez l'outil Transformer librement ou une commande de modification > transformation. Lors de l'exécution, utilisez la méthode `setSize()` ou les propriétés appropriées de la classe ProgressBar, telles que `height`, `width`, `scaleX` et `scaleY`.

Le composant ProgressBar possède trois enveloppes : une enveloppe de rail, une enveloppe de barre et une enveloppe indéterminée. Il utilise la mise à l'échelle à 9 découpes pour mettre à l'échelle les actifs.

Utilisation de styles avec le composant ProgressBar

Vous pouvez définir des propriétés de style pour modifier l'aspect d'une occurrence ProgressBar. Les styles du composant ProgressBar spécifient les valeurs de son enveloppe et de sa marge intérieure lorsque le composant est tracé. L'exemple suivant agrandit la taille d'une occurrence ProgressBar et définit son style barPadding.

Pour définir les styles de marge intérieure d'une occurrence ProgressBar :

1. Créez un fichier FLA.
2. Faites glisser le composant ProgressBar du panneau Composants vers la scène et nommez l'occurrence **myPb.**
3. Sur l'image 1 du scénario principal, entrez le code suivant dans le panneau Actions :

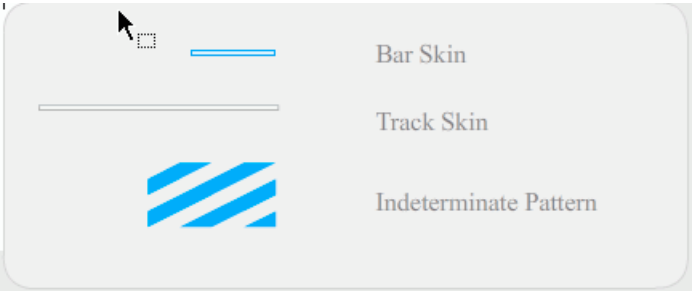
```
myPb.width = 300;
myPb.height = 30;

myPb.setStyle("barPadding", 3);
```

4. Choisissez Contrôle> Tester l'animation.

Pour plus d'informations sur la définition des styles d'enveloppe, consultez la section « A propos des enveloppes », à la page 158.

Utilisation d'enveloppes avec le composant ProgressBar

Le composant ProgressBar utilise des enveloppes pour représenter le rail de la barre de progression, la barre terminée et une barre indéterminée, comme illustré ci-dessous.



Enveloppes du composant ProgressBar

La barre est placée sur l'enveloppe de rail, à l'aide du style barPadding qui permet de déterminer le positionnement. Les actifs sont mis à l'échelle à l'aide de la mise à l'échelle à 9 découpes.

La barre indéterminée est utilisée lorsque la propriété `indeterminate` de l'occurrence ProgressBar est définie sur `true`. L'enveloppe est redimensionnée verticalement et horizontalement en fonction de la taille du composant ProgressBar.

Vous pouvez modifier ces enveloppes pour modifier l'aspect du composant ProgressBar. Ainsi, l'exemple suivant modifie la couleur de la barre indéterminée.

Pour changer la couleur de la barre indéterminée en modifiant son enveloppe :

1. Créez un fichier FLA.
2. Faites glisser un composant ProgressBar sur la scène et double-cliquez dessus pour ouvrir son panneau d'icônes d'enveloppe.
3. Double-cliquez sur l'enveloppe de la barre indéterminée.
4. Définissez le contrôle de zoom sur 400 % pour agrandir l'icône en vue de la modification.
5. Double-cliquez sur l'une des barres diagonales, maintenez enfoncée la touche Maj, puis cliquez sur toutes les autres barres diagonales. La couleur actuelle apparaît dans le sélecteur de couleur de remplissage de l'Inspecteur des propriétés.
6. Dans l'Inspecteur des propriétés, cliquez sur le sélecteur de couleur de remplissage pour l'ouvrir et sélectionnez la couleur #00CC00 pour l'appliquer aux barres diagonales sélectionnées.
7. Cliquez sur le bouton de retour figurant dans la partie gauche de la barre d'édition en haut de la scène pour revenir en mode d'édition de document.
8. Choisissez Contrôle> Tester l'animation.

Le composant ProgressBar doit apparaître tel qu'illustré ci-dessous.

Personnalisation du composant RadioButton

Vous pouvez transformer un composant RadioButton horizontalement et verticalement pendant la programmation et lors de l'exécution. Lors de la programmation, choisissez le composant sur la scène et utilisez l'outil Transformer librement ou une commande de modification > transformation. Lors de l'exécution, utilisez la méthode `setSize()`.

Le cadre de sélection d'un composant RadioButton est invisible et désigne également la zone active du composant. Si vous augmentez la taille du composant, vous augmentez également la taille de la zone active.

Si la dimension du cadre de sélection du composant est trop petite pour l'étiquette du composant, celle-ci sera rognée.

Utilisation de styles avec le composant RadioButton

Vous pouvez définir les propriétés des styles afin de modifier l'apparence d'un bouton radio. Les propriétés de style du composant ScrollPane spécifient les valeurs des enveloppes, des icônes, de formatage de texte et de marge intérieure lorsque le composant est tracé. Les styles du composant ScrollPane spécifient les valeurs de ses enveloppes et de sa marge intérieure lorsque le composant est tracé.

L'exemple suivant extrait le style `textFormat` d'un composant CheckBox et l'applique à un composant RadioButton de manière à ce que le style de leurs étiquettes soit identique.

Pour appliquer le style textFormat d'un composant CheckBox à un composant RadioButton :

1. Créez un nouveau fichier Flash (ActionScript 3.0).
2. Faites glisser un composant CheckBox sur la scène et nommez l'occurrence **myCh** dans l'Inspecteur des propriétés.
3. Faites glisser un composant RadioButton sur la scène et nommez l'occurrence **myRb** dans l'Inspecteur des propriétés.
4. Ajoutez le code suivant dans le panneau Actions, sur l'image 1 du scénario.

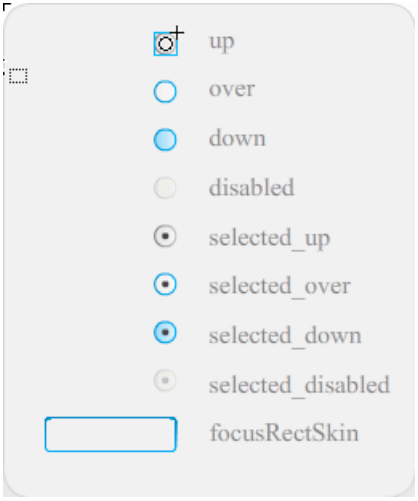
```
var tf:TextFormat = new TextFormat();
tf.color = 0x00FF00;
tf.font = "Georgia";
tf.size = 18;
myCh.setStyle("textFormat", tf);
myRb.setStyle("textFormat", myCh.getStyle("textFormat"));
```

   Ce code définit le style `textFormat` du composant CheckBox, puis l'applique au composant RadioButton en appelant la méthode `getStyle()` sur le composant CheckBox.

5. Choisissez Contrôle> Tester l'animation.

Utilisation d'enveloppes avec le composant RadioButton

Le composant RadioButton possède les enveloppes suivantes que vous pouvez modifier pour changer son apparence :



Enveloppes du composant RadioButton

Si un composant RadioButton est activé mais n'est pas sélectionné, il affiche son enveloppe à l'état survolé lorsque l'utilisateur place le pointeur de la souris au-dessus du bouton. Lorsqu'un utilisateur clique sur un composant RadioButton, il reçoit le focus d'entrée et affiche son enveloppe selected_down. Lorsque l'utilisateur relâche le bouton de la souris, le composant RadioButton affiche son enveloppe selected_up. Si l'utilisateur éloigne le pointeur de la zone active d'un composant RadioButton en appuyant sur le bouton de la souris, le composant RadioButton affiche de nouveau son enveloppe à l'état Relevé.

Si un composant RadioButton est désactivé, il affiche son état désactivé, quelle que soit l'interaction de l'utilisateur.

L'exemple suivant remplace l'enveloppe selected_up indiquant l'état sélectionné.

Pour créer une nouvelle enveloppe selected_up du composant RadioButton :

1. Créez un nouveau fichier Flash (ActionScript 3.0).
2. Faites glisser le composant RadioButton sur la scène et double-cliquez dessus pour ouvrir sa palette d'enveloppes.
3. Définissez le contrôle de zoom sur 800 % pour agrandir l'icône en vue de la modification.

4. Double-cliquez sur l'enveloppe selected_up pour la sélectionner et appuyez sur la touche Suppr pour la supprimer.
5. Choisissez l'outil Rectangle dans le panneau Outils.
6. Dans l'Inspecteur des propriétés, définissez la couleur de ligne sur rouge (#FF0000) et la couleur de remplissage sur noir (#000000).
7. En commençant au niveau de la mire qui indique le point d'alignement du symbole (également appelé *point d'origine* ou *point zéro*), cliquez sur le pointeur et faites-le glisser pour tracer un rectangle.
8. Cliquez sur le bouton de retour figurant dans la partie gauche de la barre d'édition en haut de la scène pour revenir en mode d'édition de document.
9. Choisissez Contrôle> Tester l'animation.
10. Cliquez sur le composant RadioButton pour le sélectionner.

Le composant RadioButton à l'état Sélectionné doit apparaître tel qu'illustré ci-dessous.

Personnalisation du composant ScrollPane

Vous pouvez transformer un composant ScrollPane horizontalement et verticalement pendant la programmation et lors de l'exécution. Lors de la programmation, choisissez le composant sur la scène et utilisez l'outil Transformer librement ou une commande de modification > transformation. Lors de l'exécution, utilisez la méthode `setSize()` ou les propriétés et méthodes applicables de la classe ScrollPane, telles que `height`, `width`, `scaleX` et `scaleY`.

Le composant ScrollPane possède les caractéristiques graphiques suivantes :

- Le point d'alignement (également appelé *point d'origine* ou *point zéro*) de son contenu correspond au coin supérieur gauche du panneau.
- Lorsque la barre de défilement horizontale est désactivée, la barre de défilement verticale s'affiche de haut en bas sur le côté droit du panneau défilant. Lorsque la barre de défilement verticale est désactivée, la barre de défilement horizontale s'affiche de gauche à droite en bas du panneau défilant. Vous pouvez également désactiver ces deux barres.

- Si le panneau défilant n'est pas assez grand, il se peut que le contenu ne s'affiche pas correctement.
- Lorsque le panneau défilant est redimensionné, le rail et le curseur de défilement sont agrandis ou réduits et leurs zones réactives sont redimensionnées. Les boutons conservent la même taille.

Utilisation de styles avec le composant ScrollPane

Les propriétés de style du composant ScrollPane spécifient les valeurs des enveloppes et de marge intérieure de sa disposition lorsque le composant est tracé. Les divers styles d'enveloppe vous permettent de spécifier différentes classes à utiliser pour les enveloppes du composant. Pour plus d'informations sur l'utilisation des styles d'enveloppe, consultez la section « A propos des enveloppes », à la page 158.

Pour définir le style contentPadding d'un composant ScrollPane :

1. Créez un fichier Flash.
2. Faites glisser un composant ScrollPane sur la scène et nommez l'occurrence **mySp**.
3. Dans l'Inspecteur des propriétés, cliquez sur l'onglet Paramètres et entrez la valeur suivante pour le paramètre `source` : **http://www.helpexamples.com/flash/images/image1.jpg**.
4. Sur l'image 1 du scénario principal, ajoutez le code suivant dans le panneau Actions.

   ```
   mySp.setStyle("contentPadding", 5);
   ```

   Notez que la marge intérieure est appliquée entre la bordure du composant et son contenu, à l'extérieur des barres de défilement.
5. Choisissez Contrôle> Tester l'animation.

Utilisation d'enveloppes avec le composant ScrollPane

Le composant ScrollPane utilise une bordure et des barres de défilement pour des éléments de défilement. Pour plus d'informations sur l'application d'enveloppes aux barres de défilement, consultez la section « Utilisation d'enveloppes avec le composant UIScrollBar », à la page 205.

Personnalisation du composant Slider

Vous pouvez transformer un composant Slider horizontalement pendant la programmation et lors de l'exécution. Lors de la programmation, choisissez le composant sur la scène et utilisez l'outil Transformer librement ou une commande de modification > transformation. Lors de l'exécution, utilisez la méthode `setSize()` ou les propriétés applicables de la classe Slider, telles que les propriétés `width` et `scaleX`.

Vous pouvez uniquement accroître la longueur du curseur. Vous ne pouvez pas augmenter sa hauteur. Flash ignore la propriété `height` et le paramètre height de la méthode `setSize()`. Toutefois, vous pouvez créer un curseur vertical et accroître sa longueur verticalement.

Utilisation de styles avec le composant Slider

Les styles du composant Slider spécifient uniquement les classes de ses enveloppes et la valeur de `FocusRectPadding`, qui spécifie le nombre de pixels à utiliser pour la marge intérieure entre le cadre de sélection du composant et sa limite extérieure. Pour plus d'informations sur l'utilisation des styles d'enveloppe, consultez la section « A propos des enveloppes », à la page 158.

Utilisation d'enveloppes avec le composant Slider

Le composant Slider utilise les enveloppes suivantes que vous pouvez modifier pour changer son apparence.

Enveloppes du composant Slider

L'exemple suivant modifie le rail relevé pour définir sa couleur sur le bleu.

Pour modifier la couleur du rail relevé du composant Slider :

1. Créez un nouveau fichier Flash (ActionScript 3.0).
2. Faites glisser le composant Slider du panneau Composants vers la scène.
3. Double-cliquez sur le composant Slider pour ouvrir son panneau d'enveloppes.
4. Double-cliquez sur le rail relevé au niveau du point d'alignement pour l'ouvrir en mode d'édition de symbole.
5. Définissez le contrôle de zoom sur 800 % pour agrandir l'icône en vue de la modification. Notez que le rail du composant Slider comprend trois barres.
6. Cliquez sur la barre supérieure afin de la sélectionner. Une fois sélectionnée, sa couleur apparaît dans le sélecteur de couleur de remplissage de l'Inspecteur des propriétés.
7. Dans le sélecteur de couleur de remplissage de l'Inspecteur des propriétés, sélectionnez la couleur #000066 pour l'appliquer à la barre supérieure du rail du composant Slider.
8. Cliquez sur la barre centrale du rail du composant Slider afin de la sélectionner. Une fois sélectionnée, sa couleur apparaît dans le sélecteur de couleur de remplissage de l'Inspecteur des propriétés.
9. Dans le sélecteur de couleur de remplissage de l'Inspecteur des propriétés, sélectionnez la couleur #0066FF pour l'appliquer à la barre centrale du rail du composant Slider.
10. Cliquez sur la barre inférieure du rail du composant Slider afin de la sélectionner. Une fois sélectionnée, sa couleur apparaît dans le sélecteur de couleur de remplissage de l'Inspecteur des propriétés.
11. Dans le sélecteur de couleur de remplissage de l'Inspecteur des propriétés, sélectionnez la couleur #00CCFF pour l'appliquer à la barre inférieure du rail du composant Slider.
12. Cliquez sur le bouton de retour figurant dans la partie gauche de la barre d'édition en haut de la scène pour revenir en mode d'édition de document.
13. Choisissez Contrôle> Tester l'animation.

Le composant Slider doit apparaître tel qu'illustré ci-dessous

Personnalisation du composant TextArea

Vous pouvez transformer un composant TextArea horizontalement et verticalement pendant la programmation et lors de l'exécution. Lors de la programmation, choisissez le composant sur la scène et utilisez l'outil Transformer librement ou une commande de modification > transformation. Lors de l'exécution, utilisez la méthode `setSize()` ou les propriétés applicables, telles que `height`, `width`, `scaleX` et `scaleY` de la classe TextArea.

Lorsqu'un composant TextArea est redimensionné, la bordure est redimensionnée en fonction du nouveau cadre de sélection. Le cas échéant, les barres de défilement s'affichent sur les bords inférieur et droit du cadre. La zone de texte est alors redimensionnée dans la zone restante. Dans un composant TextArea, les éléments n'ont pas de taille fixe. Si la largeur du composant TextArea est trop étroite pour pouvoir afficher la taille du texte, ce dernier est tronqué.

Utilisation de styles avec le composant TextArea

Les styles du composant TextArea spécifient les valeurs des enveloppes, de la marge intérieure et du format de texte lorsque le composant est tracé. Les styles `texFormat` et `disabledTextFormat` gèrent le style du texte affiché par le composant TextArea. Pour plus d'informations sur les propriétés de style de l'enveloppe, consultez la section « Utilisation d'enveloppes avec le composant TextArea », à la page 197.

L'exemple suivant définit le style `disabledTextFormat` pour modifier l'aspect du texte lorsque le composant TextArea est désactivé mais le même processus s'applique au paramètre du style `textFormat` pour un composant TextArea activé.

Pour définir le style disabledTextFormat d'un composant TextArea :

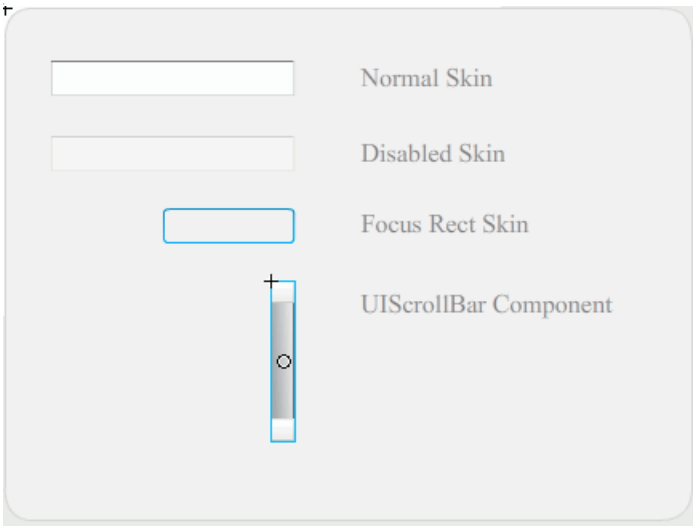
1. Créez un fichier Flash.
2. Faites glisser un composant TextArea sur la scène et nommez son occurrence **myTa**.
3. Ajoutez le code suivant dans le panneau Actions, sur l'image 1 du scénario principal.

```
var tf:TextFormat = new TextFormat();
tf.color = 0xCC99FF;
tf.font = "Arial Narrow";
tf.size = 24;
myTa.setStyle("disabledTextFormat", tf);
myTa.text = "Hello World";
myTa.setSize(120, 50);
myTa.move(200, 50);
myTa.enabled = false;
```

4. Choisissez Contrôle> Tester l'animation.

Utilisation d'enveloppes avec le composant TextArea

Le composant TextArea utilise les enveloppes suivantes que vous pouvez modifier pour changer son apparence.



Enveloppes du composant TextArea

REMARQUE	La modification de l'enveloppe ScrollBar dans un composant affecte tous les autres composants qui utilisent le composant ScrollBar.

La procédure suivante modifie les couleurs de bordure de l'enveloppe Focus Rect, qui apparaît lorsque le composant TextArea a le focus, et de l'enveloppe dont l'état est Normal.

Pour modifier la couleur des bordures du composant TextArea :

1. Créez un fichier Flash.
2. Faites glisser un composant TextArea sur la scène et double-cliquez dessus pour ouvrir son panneau d'icônes d'enveloppe.
3. Double-cliquez sur l'enveloppe Focus Rect.
4. Cliquez sur la bordure de l'enveloppe Focus Rect pour la sélectionner. Une fois sélectionnée, sa couleur actuelle apparaît dans le sélecteur de couleur de remplissage de l'Inspecteur des propriétés.

5. Dans l'Inspecteur des propriétés, cliquez sur le sélecteur de couleur de remplissage pour l'ouvrir et sélectionnez la couleur #CC0000 pour l'appliquer à la bordure.
6. Cliquez sur le bouton de retour figurant dans la partie gauche de la barre d'édition en haut de la scène pour revenir en mode d'édition de document.
7. Double-cliquez sur le composant TextArea pour ouvrir son panneau d'icônes d'enveloppe.
8. Double-cliquez sur l'enveloppe dont l'état est Normal.
9. Sélectionnez chaque bord de la bordure de l'enveloppe dont l'état est Normal, l'un après l'autre, et définissez sa couleur sur #990099.
10. Cliquez sur le bouton de retour figurant dans la partie gauche de la barre d'édition en haut de la scène pour revenir en mode d'édition de document.
11. Choisissez Contrôle> Tester l'animation.

Lorsque vous sélectionnez le composant TextArea pour commencer à entrer du texte, sa bordure doit apparaître telle qu'illustrée ci-dessous :

La bordure extérieure est l'enveloppe Focus Rect ; la bordure intérieure est la bordure de l'enveloppe dont l'état est Normal.

Pour plus d'informations sur la modification de l'enveloppe du composant UIScrollBar, consultez la section « Personnalisation du composant UIScrollBar », à la page 204.

Personnalisation du composant TextInput

Vous pouvez modifier la taille d'une occurrence TextInput au cours de la programmation et lors de l'exécution. Lors de la programmation, choisissez le composant sur la scène et utilisez l'outil Transformer librement ou une commande de modification > transformation. Lors de l'exécution, utilisez la méthode `setSize()` ou les propriétés applicables de la classe TextInput, telles que `height`, `width`, `scaleX` et `scaleY`.

Lorsqu'un composant TextInput est redimensionné, la bordure prend la taille du nouveau cadre de sélection. Le composant TextInput n'utilise pas de barres de défilement, mais le point d'insertion défile automatiquement lorsque l'utilisateur intervient sur le texte. Le champ de texte est alors redimensionné dans la zone restante. Les éléments d'un composant TextInput n'ont pas de taille fixe. Si le composant TextInput est trop petit pour afficher le texte, le texte est rogné.

Utilisation de styles avec le composant TextInput

Les styles du composant TextInput spécifient les valeurs des enveloppes, de la marge intérieure et du formatage de texte lorsque le composant est tracé. Les styles `texFormat` et `disabledTextFormat` gèrent le style du texte affiché par le composant. Pour plus d'informations sur les propriétés de style de l'enveloppe, consultez la section « Utilisation d'enveloppes avec le composant TextInput », à la page 199.

L'exemple suivant définit le style `textFormat` pour spécifier la police, la taille et la couleur du texte affiché par le composant TextInput. Le même processus s'applique au paramètre du style `disabledTextFormat` appliqué lorsque le composant est désactivé.

Pour définir le style textFormat d'une occurrence TextInput :

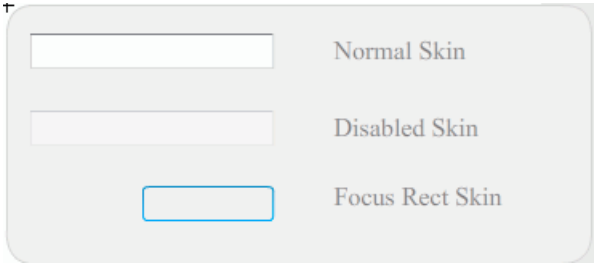
1. Créez un fichier Flash.
2. Faites glisser un composant TextInput sur la scène et nommez l'occurrence **myTi**.
3. Ajoutez le code suivant dans le panneau Actions, sur l'image 1 du scénario principal.

```
var tf:TextFormat = new TextFormat();
tf.color = 0x0000FF;
tf.font = "Verdana";
tf.size = 30;
tf.align = "center";
tf.italic = true;
myTi.setStyle("textFormat", tf);
myTi.text = "Enter your text here";
myTi.setSize(350, 50);
myTi.move(100, 50);
```

4. Choisissez Contrôle> Tester l'animation.

Utilisation d'enveloppes avec le composant TextInput

Le composant TextInput utilise les enveloppes suivantes que vous pouvez modifier pour changer son apparence :



Légende du composant TextInput

La procédure suivante modifie les couleurs de bordure et d'arrière-plan d'un composant TextInput.

Pour modifier les couleurs de bordure et d'arrière-plan du composant TextInput :

1. Créez un fichier Flash.
2. Faites glisser un composant TextInput sur la scène et double-cliquez dessus pour ouvrir son panneau d'enveloppes.
3. Double-cliquez sur l'enveloppe dont l'état est Normal.
4. Définissez le contrôle de zoom sur 800 % pour agrandir l'icône en vue de la modification.
5. Sélectionnez chaque bord de la bordure de l'enveloppe d'état normal, l'un après l'autre, et définissez sa couleur sur #993399 pour l'appliquer.
6. Double-cliquez sur l'arrière-plan jusqu'à ce que sa couleur apparaisse dans le sélecteur de couleur de remplissage de l'Inspecteur des propriétés. Sélectionnez la couleur #99CCCC pour l'appliquer à l'arrière-plan.
7. Cliquez sur le bouton de retour figurant dans la partie gauche de la barre d'édition en haut de la scène pour revenir en mode d'édition de document.
8. Choisissez Contrôle> Tester l'animation.

Le composant TextInput doit apparaître tel qu'illustré ci-dessous :

Personnalisation du composant TileList

Vous pouvez transformer le composant TileList horizontalement et verticalement pendant la programmation et lors de l'exécution. Lors de la programmation, choisissez le composant sur la scène et utilisez l'outil Transformer librement ou une commande de modification > transformation. Lors de l'exécution, utilisez la méthode `setSize()` ou les propriétés appropriées telles que les propriétés `width`, `height`, `columnCount`, `rowCount`, `scaleX` et `scaleY`. Le composant ScrollBar, inclus dans le composant TileList, est redimensionné dans la zone de liste.

Utilisation de styles avec le composant TileList

Les styles du composant TileList spécifient les valeurs des enveloppes, de la marge intérieure et du formatage de texte lorsque le composant est tracé. Les styles `texFormat` et `disabledTextFormat` gèrent le style du texte affiché par le composant. Pour plus d'informations sur les styles d'enveloppe, consultez la section « Utilisation d'enveloppes avec le composant TileList », à la page 202.

L'exemple suivant appelle la méthode `setRendererStyle()` à l'aide du style `textFormat` pour définir la police, la taille, la couleur et les attributs de texte des étiquettes affichées dans une occurrence TileList. Le même processus s'applique également au paramètre du style `disabledTextFormat` appliqué lorsque la propriété `enabled` est définie sur `false`.

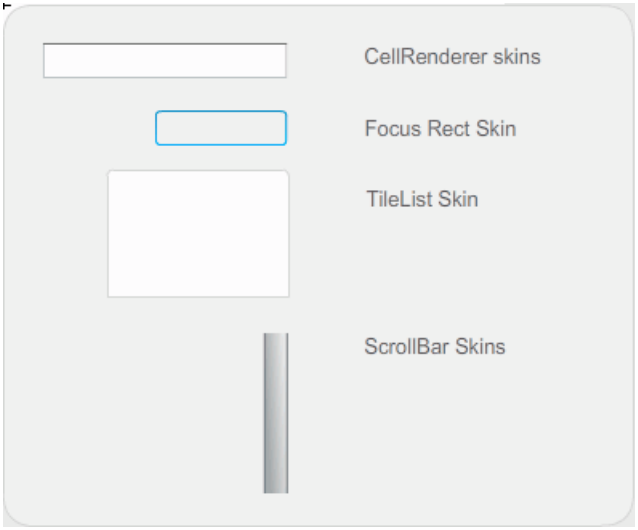
Pour définir le style textFormat d'une occurrence TileList :

1. Créez un nouveau fichier Flash (ActionScript 3.0).
2. Faites glisser le composant TileList sur la scène et nommez l'occurrence **myTl**.
3. Ajoutez le code suivant dans le panneau Actions, sur l'image 1 du scénario.

```
myTl.setSize(100, 100);
myTl.addItem({label:"#1"});
myTl.addItem({label:"#2"});
myTl.addItem({label:"#3"});
myTl.addItem({label:"#4"});
var tf:TextFormat = new TextFormat();
tf.font = "Arial";
tf.color = 0x00FF00;
tf.size = 16;
tf.italic = true;
tf.bold = true;
tf.underline = true;
tf.align = "center";
myTl.setRendererStyle("textFormat", tf);
```

Utilisation d'enveloppes avec le composant TileList

Le composant TileList possède une enveloppe TileList, une enveloppe CellRenderer et une enveloppe ScrollBar. Vous pouvez modifier ces enveloppes pour modifier l'aspect du composant TileList :



Enveloppes du composant TileList

REMARQUE	La modification de l'enveloppe ScrollBar dans un composant affecte tous les autres composants qui utilisent le composant ScrollBar.

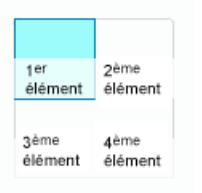
La procédure suivante modifie la couleur de l'enveloppe CellRenderer Selected_Up du composant TileList.

Pour modifier la couleur de l'enveloppe CellRenderer du composant TileList :

1. Créez un fichier Flash (ActionScript 3.0).
2. Faites glisser le composant TileList sur la scène et double-cliquez dessus pour ouvrir son panneau d'enveloppes.
3. Double-cliquez sur l'enveloppe CellRenderer, double-cliquez sur l'enveloppe Selected_Up, puis cliquez sur l'arrière-plan rectangulaire.
4. Sélectionnez la couleur #99FFFF à l'aide du sélecteur de couleur de remplissage de l'Inspecteur des propriétés pour l'appliquer à l'enveloppe Selected_Up.

5. Cliquez sur le bouton de retour figurant dans la partie gauche de la barre d'édition en haut de la scène pour revenir en mode d'édition de document.
6. Dans l'onglet Paramètres de l'Inspecteur des propriétés, double-cliquez dans la deuxième colonne de la ligne du fournisseur de données pour ouvrir la boîte de dialogue Valeurs. Ajoutez les éléments portant les étiquettes suivantes : 1er élément, 2ème élément, 3ème élément, 4ème élément.
7. Choisissez Contrôle> Tester l'animation.
8. Cliquez sur l'une des cellules du composant TileList pour la sélectionner, puis éloignez le curseur de la souris de la cellule sélectionnée.

La cellule sélectionnée doit apparaître telle qu'illustrée ci-dessous :



Personnalisation du composant UILoader

Vous pouvez transformer un composant UILoader horizontalement et verticalement pendant la programmation et lors de l'exécution. Lors de la programmation, choisissez le composant sur la scène et utilisez l'outil Transformer librement ou une commande de modification > transformation. Lors de l'exécution, utilisez la méthode `setSize()` ou les propriétés appropriées, telles que les propriétés `width`, `height`, `scaleX` et `scaleY`.

Le comportement de dimensionnement du composant UILoader est contrôlé par la propriété `scaleContent`. Lorsque `scaleContent` est défini sur `true`, le contenu est dimensionné afin de rester dans les limites du chargeur (et redimensionné lorsque la méthode `setSize()` est appelée). Lorsque `scaleContent` est défini sur `false`, la taille du composant est fixée sur celle du contenu, et la méthode `setSize()` et les propriétés de dimensionnement n'ont aucun effet.

Le composant UILoader n'intègre pas d'éléments d'interface utilisateur auxquels vous pouvez appliquer des styles ou des enveloppes.

Personnalisation du composant UIScrollBar

Vous pouvez transformer un composant UIScrollBar horizontalement et verticalement pendant la programmation et lors de l'exécution. Néanmoins, un composant UIScrollBar vertical ne vous permet pas de modifier la largeur, et un composant UIScrollBar horizontal ne vous permet pas de modifier la hauteur. Lors de la programmation, choisissez le composant sur la scène et utilisez l'outil Transformer librement ou une commande de modification > transformation. Lors de l'exécution, utilisez la méthode `setSize()` ou les propriétés applicables de la classe UIScrollBar, telles que les propriétés `width`, `height`, `scaleX` et `scaleY`.

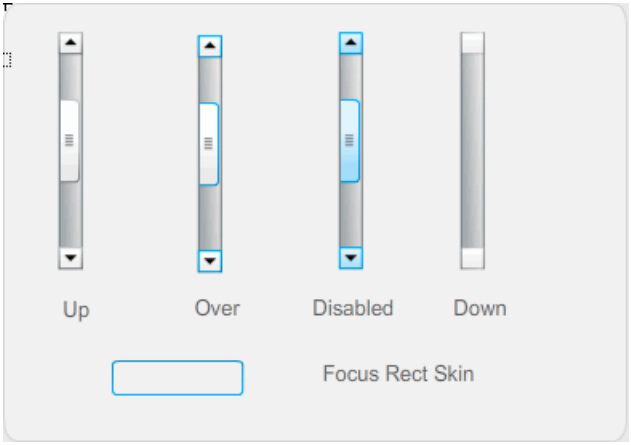
REMARQUE	Si vous utilisez la méthode `setSize()`, vous pouvez modifier uniquement la largeur d'une barre de défilement horizontale ou la hauteur d'une barre de défilement verticale. Lors de la programmation, vous pouvez définir la hauteur d'une barre de défilement horizontale, ou la largeur d'une barre de défilement verticale, mais les valeurs seront réinitialisées à la publication de l'animation. Seule la dimension d'une barre de défilement qui correspond à sa longueur peut être modifiée.

Utilisation de styles avec le composant UIScrollBar

Les styles du composant UIScrollBar spécifient uniquement les classes de ses enveloppes et la valeur de FocusRectPadding, qui spécifie le nombre de pixels à utiliser pour la marge intérieure entre le cadre de sélection du composant et sa limite extérieure. Pour plus d'informations sur l'utilisation des styles d'enveloppe, consultez la section « A propos des enveloppes », à la page 158.

Utilisation d'enveloppes avec le composant UIScrollBar

Le composant UIScrollBar utilise les enveloppes suivantes.



Enveloppes du composant UIScrollBar

Les barres de défilement horizontale et verticale utilisent les mêmes enveloppes. Lors de l'affichage d'une barre de défilement horizontale, le composant UIScrollBar fait pivoter les enveloppes comme nécessaire.

REMARQUE	La modification de l'enveloppe ScrollBar dans un composant affecte tous les autres composants qui utilisent le composant ScrollBar.

L'exemple suivant montre comment modifier la couleur du curseur du composant UIScrollBar et des

boutons fléchés.

Pour modifier la couleur des enveloppes du composant UIScrollBar :

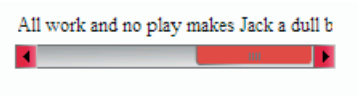
1. Créez un nouveau fichier Flash (ActionScript 3.0).
2. Faites glisser le composant UIScrollBar sur la scène et nommez l'occurrence **mySb**. Dans l'onglet Paramètres, spécifiez la direction horizontale.
3. Double-cliquez sur la barre de défilement pour ouvrir son panneau d'enveloppes.
4. Cliquez sur l'enveloppe dont l'état est Relevé pour la sélectionner.

5. Définissez le contrôle de zoom sur 400 % pour agrandir l'icône en vue de la modification.
6. Double-cliquez sur l'arrière-plan de la flèche droite (ou de la flèche vers le haut pour une barre de défilement verticale) jusqu'à ce que l'arrière-plan soit sélectionné et que sa couleur apparaisse dans le sélecteur de couleur de remplissage de l'Inspecteur des propriétés.
7. Sélectionnez la couleur #CC0033 pour l'appliquer à l'arrière-plan du bouton.
8. Cliquez sur le bouton de retour figurant dans la partie gauche de la barre d'édition en haut de la scène pour revenir en mode d'édition de document.
9. Répétez les étapes 6, 7 et 8 pour le curseur de défilement et la flèche pointant vers la gauche (ou la flèche vers le bas pour une barre de défilement verticale).
10. Ajoutez le code suivant dans le panneau Actions, sur l'image 1 du scénario, pour associer la barre de défilement à un objet TextField.

```
var tf:TextField = new TextField();
addChild(tf);
tf.x = 150;
tf.y = 100;
mySb.width = tf.width = 200;
tf.height = 22;
tf.text = "All work and no play makes Jack a dull boy. All work and no
  play makes Jack a dull boy. All . . .";
mySb.y = tf.y + tf.height;
mySb.x = tf.x + tf.width;x
mySb.scrollTarget = tf;
```

11. Choisissez Contrôle> Tester l'animation.

Le composant UIScrollBar doit apparaître tel qu'illustré ci-dessous.



CHAPITRE 5

Utilisation du composant FLVPlayback

5

Le composant FLVPlayback vous permet d'inclure facilement un lecteur vidéo à votre application Adobe Flash CS3 Professional afin de lire des fichiers FLV (Vidéo Adobe Flash) progressivement téléchargés sur HTTP ou de lire les fichiers FLV en continu à partir de Macromedia Flash Media Server d'Adobe ou du service FVSS (Flash Video Streaming Service).

Le composant FLVPlayback facile à utiliser présente les caractéristiques et les avantages suivants :

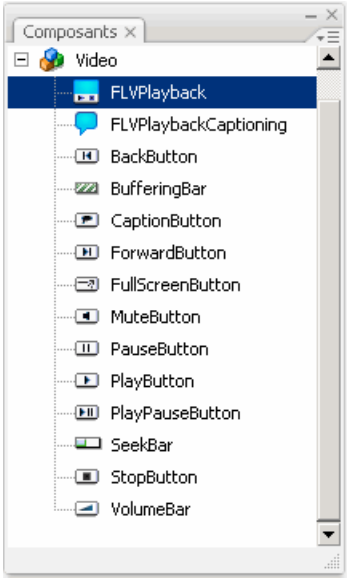
- peut être glissé sur la scène et implémenté rapidement
- prend en charge le format plein écran
- fournit un ensemble d'*enveloppes* prédéfinies qui vous permettent de personnaliser l'apparence de ses contrôles de lecture
- permet de sélectionner des valeurs de couleur et alpha pour les enveloppes prédéfinies
- permet aux utilisateurs avancés de créer leurs propres enveloppes
- fournit un aperçu en direct pendant la programmation
- fournit des propriétés de disposition pour conserver le fichier FLV centré lors du redimensionnement
- permet de commencer la lecture lorsqu'un fichier FLV téléchargé progressivement est suffisamment téléchargé
- propose des points de repère qui vous permettent de synchroniser votre vidéo avec du texte, des graphiques et de l' animation
- conserve un fichier SWF de taille raisonnable.

Utilisation du composant FLVPlayback

L'utilisation de base du composant FLVPlayback consiste à le placer sur la scène et à spécifier un fichier FLV à lire. Par ailleurs, vous pouvez définir d'autres paramètres qui régissent son comportement et décrivent le fichier FLV.

Le composant FLVPlayback inclut également une API ActionScript. L'API inclut les classes suivantes, décrites de manière détaillée dans le *Guide de référence du langage et des composants ActionScript 3.0:* CuePointType, FLVPlayback, FLVPlaybackCaptioning, NCManager, NCManagerNative, VideoAlign, VideoError, VideoPlayer, VideoState et plusieurs classes d'événements, à savoir AutoLayoutEvent, LayoutEvent, MetadataEvent. SkinErrorEvent, SoundEvent, VideoEvent et VideoProgressEvent.

Le composant FLVPlayback comprend les composants de l'interface utilisateur personnalisée Lecture de fichiers FLV. Il constitue une combinaison de la zone d'affichage, ou lecteur vidéo, dans laquelle vous affichez le fichier FLV et des commandes qui vous permettent de l'utiliser. Les composants de l'interface utilisateur personnalisée Lecture de fichiers FLV fournissent des boutons de commande et des mécanismes qui vous permettent de lire, d'arrêter, de mettre en pause et de contrôler autrement le fichier FLV. Ces commandes sont les suivantes : BackButton, BufferingBar, CaptionButton (pour FLVPlaybackCaptioning), ForwardButton, FullScreenButton, MuteButton, PauseButton, PlayButton, PlayPauseButton, SeekBar, StopButton et VolumeBar. Le composant FLVPlayback et les commandes de l'interface utilisateur personnalisée Lecture de fichiers FLV apparaissent dans le panneau Composants comme indiqué sur l'illustration suivante :



Composants FLVPlayback du panneau Composants

La procédure consistant à ajouter des commandes de lecture au composant FLVPlayback est appelée *application d'une enveloppe*. Le composant FLVPlayback possède une enveloppe initiale par défaut, SkinOverAll.swf, qui fournit les commandes de lecture, arrêt, défilement arrière, défilement avant, barre de recherche, muette, volume, plein écran et sous-titrage. Pour modifier cette enveloppe, vous pouvez effectuer les tâches suivantes :

- effectuer une sélection dans un ensemble d'enveloppes prédéfinies
- créer une enveloppe personnalisée, puis l'ajouter à l'ensemble d'enveloppes prédéfinies
- sélectionner des commandes particulières dans les composants de l'interface utilisateur personnalisée Lecture de fichiers FLV, puis les personnaliser.

Lorsque vous sélectionnez une enveloppe prédéfinie, vous pouvez choisir ses valeurs de couleur et alpha séparément, lors de la programmation ou de l'exécution. Pour plus d'informations, consultez la section « Sélection d'une enveloppe prédéfinie », à la page 233.

Une fois que vous avez sélectionné une autre enveloppe, celle-ci devient la nouvelle enveloppe par défaut.

Pour plus d'informations sur la sélection ou la création d'une enveloppe pour le composant FLVPlayback, reportez-vous à « Personnalisation du composant FLVPlayback », à la page 232.

Création d'une application avec le composant FLVPlayback

Vous pouvez inclure le composant FLVPlayback à votre application des trois façons décrites ci-dessous :

- Faites glisser le composant FLVPlayback du panneau Composants sur la scène, puis indiquez la valeur du paramètre `source`.
- Utilisez l'Assistant Importation vidéo pour créer le composant sur la scène, puis personnalisez-le en choisissant une enveloppe.
- Utilisez le constructeur `FLVPlayback()` pour créer de façon dynamique une occurrence FLVPlayback sur la scène, en supposant que le composant soit dans la bibliothèque.

REMARQUE	Si vous créez une occurrence FLVPlayback avec ActionScript, vous devez également lui affecter une enveloppe en définissant la propriété `skin` avec ActionScript. Lorsque vous appliquez une enveloppe de cette manière, elle n'est pas automatiquement publiée avec le fichier SWF. Vous devez copier le fichier SWF d'application et le fichier SWF d'enveloppe sur votre serveur d'application ; si vous ne le faites pas, le fichier SWF d'enveloppe n'est pas disponible lorsque vous exécutez l'application.

Pour faire glisser le composant FLVPlayback à partir du panneau Composants :

1. Dans le panneau Composants, cliquez sur le signe plus (+) pour ouvrir l'entrée vidéo.
2. Faites glisser le composant FLVPlayback sur la scène.
3. Dans l'onglet Paramètres de l'Inspecteur des composants, après avoir sélectionné le composant FLVPlayback sur la scène, recherchez la cellule Valeur correspondant au paramètre `source`, puis entrez une chaîne qui spécifie l'un des éléments suivants :
   - le chemin local d'un fichier FLV
   - l'URL d'un fichier FLV
   - l'URL d'un fichier SMIL (Synchronized Multimedia Integration Language) qui décrit comment lire un fichier FLV.

   Pour plus d'informations sur la création d'un fichier SMIL pour décrire un ou plusieurs fichiers FLV, reportez-vous à « Utilisation d'un fichier SMIL », à la page 250.
4. Dans l'onglet Paramètres de l'Inspecteur des composants, le composant FLVPlayback étant sélectionné sur la scène, cliquez sur la cellule Valeur correspondant au paramètre `skin`.
5. Cliquez sur l'icône en forme de loupe pour ouvrir la boîte de dialogue Sélectionner une enveloppe.
6. Choisissez l'une des options suivantes :
   - Dans la liste déroulante Enveloppe, sélectionnez l'une des enveloppes prédéfinies pour associer un ensemble de commandes de lecture au composant.
   - Si vous avez créé une enveloppe personnalisée, sélectionnez URL d'enveloppe personnalisée dans la liste déroulante et entrez, dans la zone URL, l'URL du fichier SWF contenant l'enveloppe.
   - Sélectionnez Aucune, puis faites glisser des composants de l'interface utilisateur personnalisée Lecture de fichiers FLV pour ajouter des commandes de lecture.

   REMARQUE

   Dans les deux premiers cas, un aperçu de l'enveloppe s'affiche dans le panneau de visualisation situé au-dessus du menu contextuel. Le sélecteur de couleur permet de modifier la couleur de l'enveloppe.
   Pour modifier la couleur d'une commande de l'interface utilisateur personnalisée, vous devez la personnaliser. Pour plus d'informations sur l'utilisation des commandes de l'interface utilisateur personnalisée, reportez-vous à « Application d'enveloppe aux composants particuliers de l'interface utilisateur personnalisée Lecture de fichiers FLV », à la page 235
7. Cliquez sur OK pour fermer la boîte de dialogue Sélectionner une enveloppe.
8. Choisissez Contrôle > Tester l'animation pour exécuter le fichier SWF et démarrer la vidéo.

Dans l'exemple suivant, un composant FLVPlayback est ajouté à l'aide de l'Assistant d'importation vidéo :

Pour utiliser l'Assistant d'importation vidéo :

1. Choisissez Fichier > Importer > Importer de la vidéo.
2. Indiquez l'emplacement du fichier vidéo en sélectionnant l'une des options suivantes :
   - Sur l'ordinateur local
   - Déjà déployée sur un serveur Web, Flash Video Streaming Service ou Flash Media Server.
3. Selon votre choix, entrez le chemin du fichier ou l'URL spécifiant l'emplacement du fichier vidéo, puis cliquez sur Suivant.
4. Si vous avez sélectionné un chemin de fichier, une boîte de dialogue Déploiement s'affiche ; vous pouvez alors sélectionner l'une des options répertoriées pour indiquer comment vous souhaitez déployer votre vidéo :
   - Téléchargement progressif à partir d'un serveur Web standard
   - Diffusion en continu avec le service FVSS
   - Diffusion en continu à partir de Flash Media Server
   - Intégration de la vidéo dans un fichier SWF et diffusion dans le scénario

AVERTISSEMENT	Ne sélectionnez pas l'option Incorporer la vidéo. Le composant FLVPlayback lit uniquement la vidéo en continu externe. Cette option ne permet pas de placer un composant FLVPlayback sur la scène.

5. Cliquez sur Suivant.
6. Choisissez l'une des options suivantes :
   - Dans la liste déroulante Enveloppe, sélectionnez l'une des enveloppes prédéfinies pour associer un ensemble de commandes de lecture au composant.
   - Si vous avez créé une enveloppe personnalisée pour le composant, sélectionnez URL d'enveloppe personnalisée dans la liste déroulante et entrez, dans la zone URL, l'URL du fichier SWF contenant l'enveloppe.

- Sélectionnez Aucune, puis faites glisser des composants de l'interface utilisateur personnalisée Lecture de fichiers FLV pour ajouter des commandes de lecture.

REMARQUE	Dans les deux premiers cas, un aperçu de l'enveloppe s'affiche dans le panneau de visualisation situé au-dessus du menu contextuel.

7. Cliquez sur OK pour fermer la boîte de dialogue Sélectionner une enveloppe.
8. Lisez la boîte de dialogue Terminer l'importation de vidéos pour savoir ce qui se passe ensuite, puis cliquez sur Terminer.
9. Si vous n'avez pas enregistré votre fichier FLA, une boîte de dialogue Enregistrer sous s'affiche.
10. Choisissez Contrôle > Tester l'animation pour exécuter le fichier SWF et démarrer la vidéo.

Dans l'exemple suivant, un composant FLVPlayback est ajouté à l'aide d'ActionScript.

Pour créer une occurrence de façon dynamique à l'aide d'ActionScript :

1. Faites glisser le composant FLVPlayback du panneau Composants vers le panneau Bibliothèque (Fenêtre > Bibliothèque).
2. Ajoutez le code suivant dans le panneau Actions, sur l'image 1 du scénario. Remplacez *`lecteur_installation`* par le lecteur sur lequel vous avez installé Flash, puis modifiez le chemin pour refléter l'emplacement du dossier Skins de votre installation.

   Sur un ordinateur Windows :

```
import fl.video.*;
var my_FLVPlybk = new FLVPlayback();
my_FLVPlybk.x = 100;
my_FLVPlybk.y = 100;
addChild(my_FLVPlybk);
my_FLVPlybk.skin = "file:///install_drive|/Program Files/Adobe/Adobe
  Flash CS3/en/Configuration/FLVPlayback Skins/ActionScript 3.0/
  SkinOverPlaySeekMute.swf"
my_FLVPlybk.source = "http://www.helpexamples.com/flash/video/
  water.flv";
```

Sur un ordinateur Macintosh :

```
import fl.video.*;
var my_FLVPlybk = new FLVPlayback();
my_FLVPlybk.x = 100;
my_FLVPlybk.y = 100;
addChild(my_FLVPlybk);
my_FLVPlybk.skin = "file:///Macintosh HD:Applications:Adobe Flash
  CS3:Configuration:FLVPlayback Skins:ActionScript
  3.0SkinOverPlaySeekMute.swf"
my_FLVPlybk.source = "http://www.helpexamples.com/flash/video/
  water.flv";
```

REMARQUE

Si vous ne définissez pas les propriétés `source` et `skin`, le clip généré apparaît vide.

3. Choisissez Contrôle > Tester l'animation pour exécuter le fichier SWF et démarrer le fichier FLV.

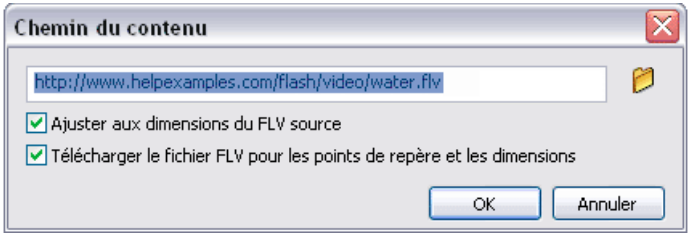
Paramètres du composant FLVPlayback

Vous pouvez définir les paramètres suivants pour chaque occurrence du composant FLVPlayback dans l'Inspecteur des composants ou l'Inspecteur des propriétés : `align`, `autoPlay`, `cuePoints`, `preview`, `scaleMode`, `skin`, `skinAutoHide`, `skinBackgroundAlpha`, `skinBackgroundColor`, `source` et `volume`. Chacun de ces paramètres possède une propriété ActionScript correspondante du même nom. Lorsque vous affectez une valeur à ces paramètres, vous définissez l'état initial de la propriété dans l'application. La définition de la propriété dans ActionScript remplace la valeur que vous avez définie dans le paramètre. Pour plus d'informations sur les valeurs possibles de ces paramètres, reportez-vous à la classe FLVPlayback dans le *Guide de référence du langage et des composants ActionScript 3.0*.

Définition du paramètre source

Le paramètre `source` permet d'indiquer le nom et l'emplacement du fichier FLV, ces deux informations indiquant à Flash comment lire le fichier.

Dans l'Inspecteur des composants, ouvrez la boîte de dialogue Chemin du contenu en double-cliquant sur la cellule Valeur correspondant au paramètre `source`. La boîte de dialogue a l'aspect suivant :



Boîte de dialogue Chemin du contenu de FLVPlayback

La boîte de dialogue contient deux cases à cocher qui permettent de déterminer les dimensions de l'occurrence FLVPlayback et d'indiquer si vous souhaitez acquérir les dimensions et les informations sur les points de repère à partir du fichier FLV. Pour plus d'informations, consultez la section « Options de fichier FLV », à la page 216.

La source

Entrez l'URL ou le chemin local du fichier FLV ou d'un fichier XML qui décrit comment lire le fichier FLV. Si vous ne connaissez pas l'emplacement exact d'un fichier FLV, cliquez sur l'icône de dossier pour ouvrir une boîte de dialogue Navigateur afin de vous aider à le trouver. Lorsque vous recherchez un fichier FLV, s'il se trouve à l'emplacement du fichier SWF cible (ou au-dessous), Flash utilise automatiquement le chemin relatif de cet emplacement afin que vous puissiez l'utiliser à partir d'un serveur Web. Autrement, il s'agit d'un chemin absolu, Windows ou Macintosh. Pour indiquer le nom d'un fichier XML local, vous devez taper son chemin et son nom.

Si vous spécifiez une URL HTTP, le fichier FLV est lu à mesure de son téléchargement progressif. Si vous spécifiez une URL RTMP, le fichier FLV est diffusé en continu à partir de Flash Media Server ou d'un FVSS. L'URL d'un fichier XML peut également être un fichier FLV à diffusion en flux continu à partir de Flash Media Server ou d'un FVSS.

ATTENTION

Si vous cliquez sur OK dans la boîte de dialogue Chemin du contenu, le composant met à jour la valeur du paramètre `cuePoints`, car il ne peut plus s'appliquer si le chemin du contenu est modifié. Par conséquent, vous pouvez perdre les points de repère désactivés, sauf s'il s'agit de points de repère ActionScript. (Vous ne perdez pas les points de repère désactivés si le nouveau fichier FLV contient ces mêmes points, ce qui peut se produire si vous changez simplement de chemin.) Pour cette raison, vous pouvez désactiver les points de repère non ActionScript via ActionScript plutôt qu'au moyen de la boîte de dialogue Points de repère.

Vous pouvez également spécifier l'emplacement d'un fichier SMIL qui décrit comment lire plusieurs flux continus de fichier FLV pour plusieurs bandes passantes. Ce fichier utilise le langage SMIL (Synchronized Multimedia Integration Language) pour décrire les fichiers FLV. Pour obtenir une description du fichier SMIL, reportez-vous à « Utilisation d'un fichier SMIL », à la page 250.

Vous pouvez également indiquer le nom et l'emplacement du fichier FLV à l'aide de la propriété `FLVPlayback.source` et des méthodes `FLVPlayback.play()` et `FLVPlayback.load()` ActionScript. Ces trois solutions sont prioritaires par rapport au paramètre `source` de l'Inspecteur des composants. Pour plus d'informations, reportez-vous aux entrées `FLVPlayback.source`, `FLVPlayback.play()` et `FLVPlayback.load()` de la classe FLVPlayback dans le *Guide de référence du langage et des composants ActionScript 3.0*.

Options de fichier FLV

La boîte de dialogue Chemin du contenu contient également deux options. La première, Correspondre aux dimensions FLV source, indique si l'occurrence FLVPlayback sur la scène doit correspondre aux dimensions du fichier FLV source. Le fichier FLV source contient la hauteur et la largeur souhaitées pour la lecture. Si vous activez cette première case à cocher, l'occurrence FLVPlayback est redimensionnée pour correspondre à ces dimensions souhaitées. Cette option est cependant disponible uniquement si la seconde case à cocher est également activée.

La seconde option, Télécharger les points de repère et dimensions FLV, est activée uniquement si le chemin du contenu est une URL HTTP ou RTMP, en d'autres termes si le fichier FLV n'est pas local. (Tout chemin qui ne se termine pas par .flv est également considéré comme un chemin réseau, car il peut s'agir d'un fichier XML et il peut pointer vers des fichiers FLV situés n'importe où). Cette option détermine si vous souhaitez télécharger ou diffuser en flux continu une partie du fichier FLV pour acquérir les dimensions et les définitions des points de repère intégrées dans ce fichier FLV. Flash utilise les dimensions pour redimensionner l'occurrence FLVPlayback et charge les définitions de points de repère dans le paramètre `cuePoints` dans l'Inspecteur des composants. Si cette case à cocher n'est pas activée, la première ne l'est pas non plus.

Utilisation de l’aperçu en direct

Le paramètre `preview` FLVPlayback permet d’afficher une image du fichier FLV source du composant sur la scène et également les modifications apportées au composant. Si vous cliquez sur le paramètre `preview`, la boîte de dialogue suivante permettant de lire un fichier SWF du fichier FLV source s’ouvre.

Boîte de dialogue de sélection d’une image d’aperçu en direct

Cliquez sur OK lorsque le fichier FLV atteint la séquence que vous souhaitez capturer pour afficher l’aperçu du composant sur la scène. L’affichage d’une image du fichier FLV du composant sur la scène vous permet de l’afficher sur la scène en relation avec d’autres éléments de l’application.

Vous pouvez également exporter l’image sélectionnée afin de l’enregistrer en tant que fichier PNG (portable network graphics) à l’emplacement de votre choix.

Prise en charge du plein écran

La version ActionScript 3.0 de FLVPlayback prend en charge le mode plein écran qui nécessite Flash Player 9.0.28.0, ainsi que la configuration adéquate des fichiers HTML en vue de l'affichage plein écran. Certaines enveloppes prédéfinies incluent un bouton bascule permettant d'activer et de désactiver le mode plein écran. La commande FullScreenButton est située à droite de la barre de contrôle dans l'illustration suivante.

Icône Plein écran de la barre de contrôle

La prise en charge du mode plein écran inclut les propriétés suivantes : `fullScreenBackgroundColor`, `fullScreenSkinDelay` et `fullScreenTakeOver`. Pour plus d'informations sur ces propriétés, consultez le *Guide de référence du langage et des composants ActionScript 3.0*.

Alignement de la disposition pour lire plusieurs fichiers FLV

ActionScript 3.0 FLVPlayback possède une propriété `align` qui indique si le fichier FLV doit être centré lorsqu'il est redimensionné ou positionné en haut, en bas, à gauche ou à droite du composant. Outre les propriétés `x`, `y`, `width` et `height`, le composant ActionScript 3.0 possède également les propriétés `registrationX`, `registrationY`, `registrationWidth` et `registrationHeight`. Initialement, celles-ci correspondent aux propriétés `x`, `y`, `width` et `height`. Lors du chargement de fichiers FLV supplémentaires, la nouvelle mise en forme automatique ne les modifie pas, ce qui permet de centrer le nouveau fichier FLV au même emplacement. Si `scaleMode = VideoScaleMode.MAINTAIN_ASPECT_RATIO`, vous pouvez faire correspondre les fichiers FLV supplémentaires aux dimensions d'origine du composant, au lieu de modifier sa largeur et sa hauteur.

Lecture automatique des fichiers FLV téléchargés progressivement

Lors du chargement d'un fichier FLV téléchargé progressivement, FLVPlayback commence à le lire uniquement lorqu'il est suffisamment téléchargé de manière à pouvoir le lire intégralement du début à la fin.

Si vous voulez lire le fichier FLV avant qu'il ne soit suffisamment téléchargé, appelez la méthode `play()` sans paramètre.

Si vous voulez revenir à l'état d'attente afin de permettre au fichier FLV d'être suffisamment téléchargé, appelez la méthode `pause()`, puis la méthode `playWhenEnoughDownloaded()`.

Utilisation des points de repère

Un point de repère est un point au niveau duquel le lecteur vidéo distribue un événement `cuePoint` pendant la lecture d'un fichier FLV. Vous pouvez ajouter des points de repère dans un fichier FLV aux moments auxquels vous souhaitez qu'une action se produise pour un autre élément de la page Web. Par exemple, vous pouvez afficher du texte ou un graphique ou bien effectuer une synchronisation avec une animation Flash ou encore modifier la lecture du fichier FLV en l'interrompant, en recherchant un autre point de repère dans la vidéo ou en basculant vers un autre fichier FLV. Les points de repère permettent de recevoir des commandes dans le code ActionScript afin de synchroniser les points contenus dans votre fichier FLV avec d'autres actions de la page Web.

Il existe trois types de points de repère : de navigation, d'événement et ActionScript. Les points de repère de navigation et d'événement sont également désignés sous le nom de points de repère *intégrés*, car ils sont intégrés dans le flux continu du fichier FLV et dans le paquet de métadonnées du fichier FLV.

Un *point de repère de navigation* permet de rechercher une image particulière du fichier FLV, car il y crée une image-clé la plus près possible de l'heure indiquée. Une *image-clé* est un segment de données qui intervient entre les images du flux continu du fichier FLV. Lorsque vous recherchez un point de repère de navigation, le composant recherche cette image-clé et démarre l'événement `cuePoint`.

Un *point de repère d'événement* permet de synchroniser un point dans le temps du fichier FLV avec un événement externe de la page Web. L'événement `cuePoint` se produit exactement au moment spécifié. Vous pouvez intégrer des points de repère de navigation et d'événement dans un fichier FLV à l'aide de l'Assistant Importation vidéo ou de l'encodeur vidéo de Flash. Pour plus d'informations sur l'Assistant d'importation vidéo et l'encodeur vidéo de Flash, consultez le chapitre 16, « Utilisation de la vidéo » du guide *Utilisation de Flash*.

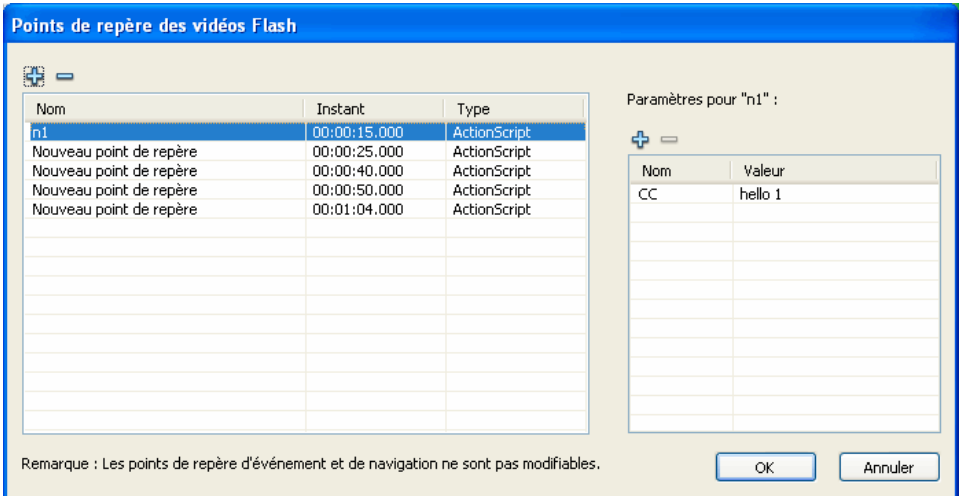
Un *point de repère ActionScript* est un point de repère externe que vous pouvez ajouter dans la boîte de dialogue Points de repère des vidéos Flash du composant ou via la méthode `FLVPlayback.addASCuePoint()`. Le composant stocke et suit les points de repère ActionScript séparément du fichier FLV. Ils sont par conséquent moins précis que les points de repère intégrés. La précision des points de repère ActionScript est d'un dixième de seconde. Vous pouvez améliorer la précision des points de repère ActionScript en diminuant la valeur de la propriété `playheadUpdateInterval`, car le composant génère l'événement `cuePoint` pour les points de repère ActionScript lors des mises à jour de la tête de lecture. Pour plus d'informations, reportez-vous à la propriété `FLVPlayback.playheadUpdateInterval` dans le *Guide de référence du langage et des composants ActionScript 3.0*.

Dans ActionScript et dans les métadonnées du fichier FLV, un point de repère est représenté sous la forme d'un objet pourvu des propriétés suivantes : `name`, `time`, `type` et `parameters`. La propriété `name` est une chaîne qui contient le nom affecté au point de repère. La propriété `time` est un nombre qui représente l'heure exprimée en heures, minutes, secondes et millisecondes (HH:MM:SS.mmm) à laquelle se présente le point de repère. La propriété `type` est une chaîne dont la valeur est `navigation`, `event` ou `actionscript`, en fonction du type de point de repère que vous avez créé. La propriété `parameters` est un tableau de paires nom/valeur spécifiées.

Lorsqu'un événement `cuePoint` se produit, l'objet point de repère est disponible dans l'objet événement par le biais de la propriété `info`. Pour plus d'informations, consultez la section « Ecoute des événements cuePoint », à la page 224.

Utilisation de la boîte de dialogue Points de repère des vidéos Flash

Ouvrez la boîte de dialogue Points de repère des vidéos Flash en double-cliquant sur la cellule Valeur du paramètre `cuePoints` dans l'Inspecteur des composants. La boîte de dialogue est semblable à l'illustration suivante :



Boîte de dialogue Points de repère

La boîte de dialogue affiche des points de repère intégrés et ActionScript. Vous pouvez l'utiliser pour ajouter et supprimer des points de repère ActionScript et des paramètres cuePoints. Vous pouvez également activer ou désactiver des points de repère intégrés. Vous ne pouvez cependant pas en ajouter, les modifier ou les supprimer.

Pour ajouter un point de repère ActionScript :

1. Dans l'Inspecteur des composants, double-cliquez sur la cellule Valeur du paramètre `cuePoints` pour ouvrir la boîte de dialogue Points de repère Flash.
2. Cliquez sur le signe plus (+) dans le coin supérieur gauche, au-dessus de la liste des points de repère, pour ajouter une entrée de point de repère ActionScript par défaut.
3. Cliquez sur le texte Nouveau point de repère dans la colonne Nom, puis ajoutez du texte pour nommer le point de repère.

4. Cliquez sur la valeur horaire 00:00:00:000 pour la modifier, puis affectez l'heure du point de repère. Vous pouvez spécifier l'heure en heures, minutes, secondes et millisecondes (HH:MM:SS.mmm).

   Si plusieurs points de repère existent, la boîte de dialogue place ce nouveau point à sa position chronologique dans la liste.
5. Pour ajouter un paramètre au point de repère sélectionné, cliquez sur le signe plus (+) au-dessus de la section Paramètres, puis entrez des valeurs dans les colonnes Nom et Valeur. Répétez cette étape pour chaque paramètre.
6. Pour ajouter plusieurs points de repère ActionScript, répétez les étapes 2 à 5 pour chaque point.
7. Cliquez sur OK pour enregistrer les changements.

Pour supprimer un point de repère ActionScript :

1. Dans l'Inspecteur des composants, double-cliquez sur la cellule Valeur du paramètre `cuePoints` pour ouvrir la boîte de dialogue Points de repère Flash.
2. Sélectionnez le point de repère à supprimer.
3. Cliquez sur le signe moins (-) dans le coin supérieur gauche, au-dessus de la liste des points de repère, pour le supprimer.
4. Répétez les étapes 2 et 3 pour chaque point de repère à supprimer.
5. Cliquez sur OK pour enregistrer les changements.

Pour activer ou désactiver un point de repère intégré à un fichier FLV :

1. Dans l'Inspecteur des composants, double-cliquez sur la cellule Valeur du paramètre `cuePoints` pour ouvrir la boîte de dialogue Points de repère Flash.
2. Sélectionnez le point de repère à activer ou à désactiver.
3. Dans la colonne Type, cliquez sur la valeur pour déclencher le menu contextuel ou cliquez sur la flèche vers le bas.
4. Cliquez sur le nom du type de point de repère (par exemple, Evénement ou Navigation) pour l'activer. Cliquez sur Désactivé pour le désactiver.
5. Cliquez sur OK pour enregistrer les changements.

Utilisation d'ActionScript avec des points de repère

Vous pouvez utiliser ActionScript pour ajouter des points de repère ActionScript, écouter des événements `cuePoint`, rechercher des points de repère d'un type quelconque ou particulier, chercher un point de repère de navigation, activer ou désactiver un point de repère, vérifier si un point de repère est activé ou en supprimer un.

Les exemples figurant dans cette section utilisent le fichier FLV cuepoints.flv. Il contient les trois points de repère suivants :

Nom	Heure	Type
point1	00:00:00.418	Navigation
point2	00:00:07.748	Navigation
point3	00:00:16.020	Navigation

Ajout de points de repère ActionScript

Vous pouvez ajouter des points de repère ActionScript dans un fichier FLV à l'aide de la méthode `addASCuePoint()`. L'exemple suivant ajoute deux points de repère ActionScript dans le fichier FLV lorsqu'il est prêt. Il ajoute le premier point de repère à l'aide d'un objet point de repère qui spécifie l'heure, le nom et le type du point dans ses propriétés. Le second appel spécifie l'heure et le nom à l'aide des paramètres `time` et `name` de la méthode.

```
import fl.video.*;
import fl.video.MetadataEvent;
my_FLVPlybk.source = "http://www.helpexamples.com/flash/video/
  cuepoints.flv"
var cuePt:Object = new Object(); //create cue point object
cuePt.time = 2.02;
cuePt.name = "ASpt1";
cuePt.type = "actionscript";
my_FLVPlybk.addASCuePoint(cuePt);  //add AS cue point
// add 2nd AS cue point using time and name parameters
my_FLVPlybk.addASCuePoint(5, "ASpt2");
```

Pour plus d'informations, reportez-vous à la méthode `FLVPlayback.addASCuePoint()` dans le *Guide de référence du langage et des composants ActionScript 3.0*.

Ecoute des événements cuePoint

L'événement `cuePoint` permet de recevoir le contrôle dans le code ActionScript lorsqu'un événement `cuePoint` se produit. Lorsque des points de repère sont trouvés dans l'exemple suivant, l'écouteur `cuePoint` appelle un gestionnaire d'événements qui affiche la valeur de la propriété `playheadTime`, ainsi que le nom et le type du point de repère.

```
my_FLVPlybk.addEventListener(MetadataEvent.CUE_POINT, cp_listener);
function cp_listener(eventObject:MetadataEvent):void {
    trace("Elapsed time in seconds: " + my_FLVPlybk.playheadTime);
    trace("Cue point name is: " + eventObject.info.name);
    trace("Cue point type is: " + eventObject.info.type);
}
```

Pour plus d'informations sur l'événement `cuePoint`, reportez-vous à l'événement `FLVPlayback.cuePoint` dans le *Guide de référence du langage et des composants ActionScript 3.0*.

Recherche de points de repère

En utilisant ActionScript, vous pouvez rechercher un point de repère de tout type, le point de repère le plus proche d'une heure ou un point ayant un nom particulier.

Dans l'exemple suivant, le gestionnaire d'événements `ready_listener()` appelle la méthode `findCuePoint()` pour rechercher le point de repère `ASpt1`, puis la méthode `findNearestCuePoint()` pour rechercher le point de repère de navigation le plus proche de l'heure du point `ASpt1`:

```
import fl.video.FLVPlayback;
import fl.video.CuePointType;
import fl.video.VideoEvent;
my_FLVPlybk.source = "http://www.helpexamples.com/flash/video/
  cuepoints.flv"
var rtn_obj:Object; //create cue point object
my_FLVPlybk.addASCuePoint(2.02, "ASpt1");  //add AS cue point
function ready_listener(eventObject:VideoEvent):void {
  rtn_obj = my_FLVPlybk.findCuePoint("ASpt1", CuePointType.ACTIONSCRIPT);
  traceit(rtn_obj);
  rtn_obj = my_FLVPlybk.findNearestCuePoint(rtn_obj.time,
  CuePointType.NAVIGATION);
  traceit(rtn_obj);
}
my_FLVPlybk.addEventListener(VideoEvent.READY, ready_listener);
function traceit(cuePoint:Object):void {
  trace("Cue point name is: " + cuePoint.name);
  trace("Cue point time is: " + cuePoint.time);
  trace("Cue point type is: " + cuePoint.type);
}
```

Dans l'exemple suivant, le gestionnaire d'événements `ready_listener()` recherche le point de repère `ASpt` et appelle la méthode `findNextCuePointWithName()` pour rechercher le point de repère suivant portant le même nom :

```
import fl.video.*;
my_FLVPlybk.source = "http://www.helpexamples.com/flash/video/
  cuepoints.flv"
var rtn_obj:Object; //create cue point object
my_FLVPlybk.addASCuePoint(2.02, "ASpt");  //add AS cue point
my_FLVPlybk.addASCuePoint(3.4, "ASpt");  //add 2nd Aspt
my_FLVPlybk.addEventListener(VideoEvent.READY, ready_listener);
function ready_listener(eventObject:VideoEvent):void {
  rtn_obj = my_FLVPlybk.findCuePoint("ASpt", CuePointType.ACTIONSCRIPT);
  traceit(rtn_obj);
  rtn_obj = my_FLVPlybk.findNextCuePointWithName(rtn_obj);
  traceit(rtn_obj);
}
function traceit(cuePoint:Object):void {
  trace("Cue point name is: " + cuePoint.name);
  trace("Cue point time is: " + cuePoint.time);
  trace("Cue point type is: " + cuePoint.type);
}
```

Pour plus d'informations sur la recherche de points de repère, reportez-vous aux méthodes `FLVPlayback.findCuePoint()`, `FLVPlayback.findNearestCuePoint()` et `FLVPlayback.findNextCuePointWithName()` dans le *Guide de référence du langage et des composants ActionScript 3.0*.

Recherche de points de repère de navigation

Vous pouvez rechercher un point de repère de navigation et le point de repère de navigation suivant ou précédent à une heure spécifiée. L'exemple suivant lit le fichier FLV cuepoints.flv et recherche le point de repère situé à 7.748 lorsque l'événement `ready` a lieu. Lorsque l'événement `cuePoint` se produit, l'exemple appelle la méthode `seekToPrevNavCuePoint()` pour rechercher le premier point de repère. Lorsque cet `événement` se produit, l'exemple appelle la méthode `seekToNextNavCuePoint()` pour rechercher le dernier point de repère en ajoutant 10 secondes à `eventObject.info.time`, qui correspond à l'heure du point de repère actuel.

```
import fl.video.*;

my_FLVPlybk.addEventListener(VideoEvent.READY, ready_listener);
function ready_listener(eventObject:Object):void {
  my_FLVPlybk.seekToNavCuePoint("point2");
}
my_FLVPlybk.addEventListener(MetadataEvent.CUE_POINT, cp_listener);
function cp_listener(eventObject:MetadataEvent):void {
```

```
  trace(eventObject.info.time);
  if(eventObject.info.time == 7.748)
    my_FLVPlybk.seekToPrevNavCuePoint(eventObject.info.time - .005);
  else
    my_FLVPlybk.seekToNextNavCuePoint(eventObject.info.time + 10);
}
my_FLVPlybk.source = "http://helpexamples.com/flash/video/cuepoints.flv";
```

Pour plus d'informations, reportez-vous aux méthodes `FLVPlayback.seekToNavCuePoint()`, `FLVPlayback.seekToNextNavCuePoint()` et FLVPlayback.seekToPrevNavCuePoint() dans le *Guide de référence du langage et des composants ActionScript 3.0*.

Activation et désactivation des points de repère intégrés au fichier FLV

Vous pouvez activer et désactiver les points de repère intégrés au fichier FLV à l'aide de la méthode `setFLVCuePointEnabled()`. Les points de repère désactivés ne déclenchent pas d'événements `cuePoint` et n'utilisent pas les méthodes `seekToCuePoint()`, `seekToNextNavCuePoint()` ni `seekToPrevNavCuePoint()`. Toutefois, vous pouvez rechercher des points de repère désactivés à l'aide des méthodes `findCuePoint()`, `findNearestCuePoint()` et `findNextCuePointWithName()`.

Vous pouvez vérifier si un point de repère intégré au fichier FLV est activé en utilisant la méthode `isFLVCuePointEnabled()`. L'exemple suivant désactive les points de repère intégrés `point2` et `point3` lorsque la vidéo est prête à être lue. Cependant, lorsque le premier événement `cuePoint` se produit, le gestionnaire d'événements vérifie si le point de repère `point3` est désactivé. Le cas échéant, il l'active.

```
import fl.video.*;
my_FLVPlybk.source = "http://www.helpexamples.com/flash/video/
  cuepoints.flv";
my_FLVPlybk.addEventListener(VideoEvent.READY, ready_listener);
function ready_listener(eventObject:VideoEvent):void {
  my_FLVPlybk.setFLVCuePointEnabled(false, "point2");
  my_FLVPlybk.setFLVCuePointEnabled(false, "point3");
}
my_FLVPlybk.addEventListener(MetadataEvent.CUE_POINT, cp_listener);
function cp_listener(eventObject:MetadataEvent):void {
    trace("Cue point time is: " + eventObject.info.time);
    trace("Cue point name is: " + eventObject.info.name);
    trace("Cue point type is: " + eventObject.info.type);
    if (my_FLVPlybk.isFLVCuePointEnabled("point2") == false) {
      my_FLVPlybk.setFLVCuePointEnabled(true, "point2");
    }
}
```

Pour plus d'informations, reportez-vous aux méthodes `FLVPlayback.isFLVCuePointEnabled()` et `FLVPlayback.setFLVCuePointEnabled()` dans le *Guide de référence du langage et des composants ActionScript 3.0*.

Suppression d'un point de repère ActionScript

La méthode `removeASCuePoint()` permet de supprimer un point de repère ActionScript. L'exemple suivant supprime le point de repère `ASpt2` lorsque le point de repère `ASpt1` se présente :

```
import fl.video.*;
my_FLVPlybk.source = "http://www.helpexamples.com/flash/video/
  cuepoints.flv"
my_FLVPlybk.addASCuePoint(2.02, "ASpt1");  //add AS cue point
my_FLVPlybk.addASCuePoint(3.4, "ASpt2");  //add 2nd Aspt
my_FLVPlybk.addEventListener(MetadataEvent.CUE_POINT, cp_listener);
function cp_listener(eventObject:MetadataEvent):void {
     trace("Cue point name is: " + eventObject.info.name);
     if (eventObject.info.name == "ASpt1") {
       my_FLVPlybk.removeASCuePoint("ASpt2");
       trace("Removed cue point ASpt2");
     }
}
```

Pour plus d'informations, reportez-vous à `FLVPlayback.removeASCuePoint()` dans le *Guide de référence du langage et des composants ActionScript 3.0*.

Lecture de plusieurs fichiers FLV

Vous pouvez lire des fichiers FLV de façon séquentielle dans une occurrence FLVPlayback en chargeant tout simplement une nouvelle URL dans la propriété `source` à la fin de la lecture du fichier FLV précédent. Par exemple, le code ActionScript suivant écoute l'événement `complete` qui se produit à la fin de la lecture d'un fichier FLV. Lorsque cet événement se produit, le code définit le nom et l'emplacement d'un nouveau fichier FLV dans la propriété `source`, puis appelle la méthode `play()` pour lire la nouvelle vidéo.

```
import fl.video.*;
my_FLVPlybk.source = "http://www.helpexamples.com/flash/video/clouds.flv";
my_FLVPlybk.addEventListener(VideoEvent.COMPLETE, complete_listener);
// listen for complete event; play new FLV
function complete_listener(eventObject:VideoEvent):void {
  if (my_FLVPlybk.source == "http://www.helpexamples.com/flash/video/
  clouds.flv") {
    my_FLVPlybk.play("http://www.helpexamples.com/flash/video/water.flv");
  }
};
```

Utilisation de plusieurs lecteurs vidéo

Vous pouvez également ouvrir plusieurs lecteurs vidéo dans une seule occurrence du composant FLVPlayback pour lire plusieurs vidéos et basculer entre eux en fonction de la lecture.

Le lecteur vidéo initial est créé lorsque vous faites glisser le composant FLVPlayback jusqu'à la scène. Le composant lui affecte automatiquement le numéro 0 et il devient lecteur par défaut. Pour créer un autre lecteur vidéo, définissez simplement la propriété `activeVideoPlayerIndex` sur un nouveau numéro. En définissant la propriété `activeVideoPlayerIndex`, vous *activez* également le lecteur vidéo spécifié, lequel sera affecté par les propriétés et les méthodes de la classe FLVPlayback. Toutefois la définition de la propriété `activeVideoPlayerIndex` ne permet pas d'afficher le lecteur vidéo. Pour le rendre visible, définissez la propriété `visibleVideoPlayerIndex` sur le numéro du lecteur vidéo. Pour plus d'informations sur la façon dont ces propriétés interagissent avec les méthodes et propriétés de la classe FLVPlayback, reportez-vous aux propriétés `FLVPlayback.activeVideoPlayerIndex` et `FLVPlayback.visibleVideoPlayerIndex` dans le *Guide de référence du langage et des composants ActionScript 3.0*.

Le code ActionScript suivant charge la propriété `source` pour lire un fichier FLV dans le lecteur vidéo par défaut, puis lui ajoute un point de repère. Lorsque l'événement `ready` se produit, le gestionnaire d'événements ouvre un second lecteur vidéo en définissant la propriété `activeVideoPlayerIndex` sur le numéro 1. Il indique un fichier FLV et un point de repère pour ce second lecteur, puis rend de nouveau le lecteur par défaut (0) actif.

```
/**
 Requires:
  - FLVPlayback component on the Stage with an instance name of my_FLVPlybk
*/
// add a cue point to the default player
import fl.video.*;
my_FLVPlybk.source = "http://www.helpexamples.com/flash/video/clouds.flv";
my_FLVPlybk.addASCuePoint(3, "1st_switch");
my_FLVPlybk.addEventListener(VideoEvent.READY, ready_listener);
function ready_listener(eventObject:VideoEvent):void {
  // add a second video player and create a cue point for it
  my_FLVPlybk.activeVideoPlayerIndex = 1;
  my_FLVPlybk.source = "http://www.helpexamples.com/flash/video/
  water.flv";
  my_FLVPlybk.addASCuePoint(3, "2nd_switch");
  my_FLVPlybk.activeVideoPlayerIndex = 0;
};
```

Pour basculer vers un autre fichier FLV lors de la lecture d'un premier fichier, vous devez passer dans le code ActionScript. Les points de repère vous permettent d'intervenir à des points spécifiques du fichier FLV à l'aide d'un événement `cuePoint`. Le code suivant crée un écouteur pour l'événement `cuePoint` et appelle une fonction de gestionnaire qui met en pause le lecteur vidéo actif (0), bascule vers le second lecteur (1), puis lit son fichier FLV :

```
import fl.video.*;
// add listener for a cuePoint event
my_FLVPlybk.addEventListener(MetadataEvent.CUE_POINT, cp_listener);
// add the handler function for the cuePoint event
function cp_listener(eventObject:MetadataEvent):void {
     // display the no. of the video player causing the event
     trace("Hit cuePoint event for player: " + eventObject.vp);
     // test for the video player and switch FLV files accordingly
     if (eventObject.vp == 0) {
       my_FLVPlybk.pause(); //pause the first FLV file
       my_FLVPlybk.activeVideoPlayerIndex = 1; // make the 2nd player active
       my_FLVPlybk.visibleVideoPlayerIndex = 1; // make the 2nd player
                                                // visible
       my_FLVPlybk.play(); // begin playing the new player/FLV
     } else if (eventObject.vp == 1) {
       my_FLVPlybk.pause(); // pause the 2nd FLV
       my_FLVPlybk.activeVideoPlayerIndex = 0; // make the 1st player active
       my_FLVPlybk.visibleVideoPlayerIndex = 0; // make the 1st player
                                                // visible
       my_FLVPlybk.play(); // begin playing the 1st player
     }
}
my_FLVPlybk.addEventListener(VideoEvent.COMPLETE, complete_listener);
function complete_listener(eventObject:VideoEvent):void {
     trace("Hit complete event for player: " + eventObject.vp);
     if (eventObject.vp == 0) {
       my_FLVPlybk.activeVideoPlayerIndex = 1;
       my_FLVPlybk.visibleVideoPlayerIndex = 1;
       my_FLVPlybk.play();
     } else {
       my_FLVPlybk.closeVideoPlayer(1);
     }
};
```

Lorsque vous créez un lecteur vidéo, l'occurrence FLVPlayback définit ses propriétés sur la valeur du lecteur vidéo par défaut, excepté les propriétés `source`, `totalTime` et `isLive` que cette occurrence définit toujours sur les valeurs par défaut : respectivement chaîne vide, 0 et `false`. Elle définit la propriété `autoPlay`, dont la valeur par défaut est `true` pour le lecteur vidéo par défaut, sur `false`. La propriété `cuePoints` n'a aucun effet, et n'a pas d'incidences sur un chargement ultérieur dans le lecteur vidéo par défaut.

Les méthodes et les propriétés qui contrôlent le volume, le positionnement, les dimensions, la visibilité et les commandes de l'interface utilisateur sont toujours globales, et leur comportement n'est pas affecté par la définition de la propriété `activeVideoPlayerIndex`. Pour plus d'informations sur ces méthodes et propriétés, et sur l'effet de la définition de la propriété `activeVideoPlayerIndex`, reportez-vous à la propriété `FLVPlayback.activeVideoPlayerIndex` dans le *Guide de référence du langage et des composants ActionScript 3.0*. Les autres propriétés et méthodes visent le lecteur vidéo identifié par la valeur de la propriété `activeVideoPlayerIndex`.

Néanmoins, les propriétés et les méthodes qui contrôlent les dimensions *interagissent* avec la propriété `visibleVideoPlayerIndex`. Pour plus d'informations, reportez-vous à la propriété `FLVPlayback.visibleVideoPlayerIndex` dans le *Guide de référence du langage et des composants ActionScript 3.0*.

Lecture de fichiers FLV en continu à partir de Flash Media Server

Les exigences en matière de lecture de fichiers FLV en continu à partir de Flash Media Server diffèrent selon que la détection de la bande passante native est disponible auprès de votre fournisseur FVSS (Flash Video Streaming Service). La détection de la bande passante native signifie que la détection de la bande passante est intégrée au serveur de diffusion, offrant ainsi de meilleures performances. Vérifiez auprès de votre fournisseur afin de déterminer si la détection de la bande passante native est disponible.

Pour accéder à vos fichiers FLV sur Flash Media Server, utilisez une URL du type rtmp://*mon_serveur/mon_application/flux_continu*.flv.

Pour lire un flux en direct avec Flash Media Server, vous devez définir la propriété FLVPlayback `isLive` sur `true`. Pour plus d'informations, reportez-vous à la propriété `FLVPlayback.isLive` dans le *Guide de référence du langage et des composants ActionScript 3.0*.

Pour plus d'informations sur l'administration de Flash Media Server, notamment la configuration d'un flux en direct, consultez la documentation Flash Media Server à l'adresse www.adobe.com/support/documentation/fr/flashmediaserver/.

Détection de la bande passante native ou absence de détection de la bande passante

La classe NCManagerNative est une sous-classe de NCManager qui prend en charge la détection de la bande passante native, pouvant être prise en charge par certains fournisseurs FVSS. Lorsque vous utilisez la classe NCManagerNative, aucun fichier spécial n'est requis sur Flash Media Server. NCManagerNative permet également de se connecter à une version quelconque de Flash Media Server, sans fichier main.asc, si la détection de la bande passante n'est pas requise.

Pour utiliser la classe NCManagerNative au lieu de la classe NCManager par défaut, ajoutez les lignes de code suivantes dans la première image de votre fichier FLA :

```
import fl.video*;
VideoPlayer.iNCManagerClass = fl.video.NCManagerNative;
```

Détection de la bande passante non native

Si la détection de la bande passante native n'est pas disponible auprès de votre fournisseur FVSS (Flash Video Streaming Service) et si vous en avez besoin, vous devez ajouter le fichier main.asc dans votre application FLV Flash Media Server. Le fichier main.asc se trouve dans le dossier applicatif Flash sous Adobe Flash CS3/Samples and Tutorials/Samples/Components/FLVPlayback/main.asc.

Pour configurer Flash Media Server afin de diffuser les fichiers FLV en continu :

1. Créez un dossier dans le dossier applicatif Flash Media Server, puis nommez-le par exemple **my_application**.
2. Copiez le fichier main.asc dans le dossier my_application.
3. Créez un dossier nommé **streams** dans le dossier my_application.
4. Créez un dossier nommé **_definst_** dans le dossier streams.
5. Placez vos fichiers FLV dans le dossier **_definst_**.

Personnalisation du composant FLVPlayback

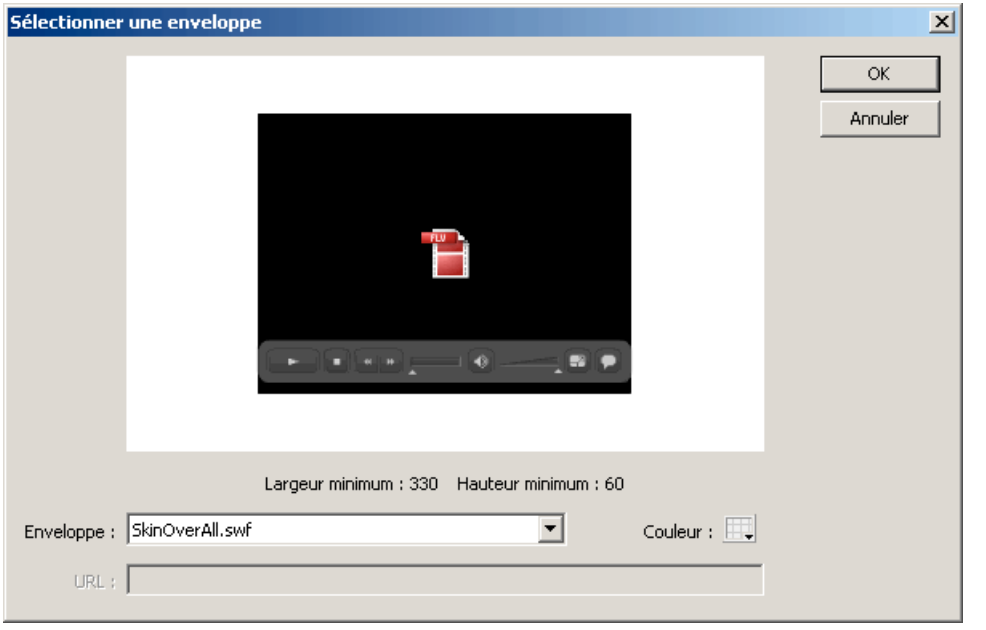
Cette section explique comment personnaliser le composant FLVPlayback. Cependant, la plupart des méthodes utilisées pour personnaliser les autres composants ne fonctionnent pas avec le composant FLVPlayback. Pour le personnaliser, utilisez uniquement les techniques décrites dans cette section.

Les méthodes suivantes permettent de personnaliser le composant FLVPlayback : sélection d'une enveloppe prédéfinie, application d'une enveloppe aux composants particuliers de l'interface utilisateur personnalisée Lecture de fichiers FLV ou création d'une enveloppe. Vous pouvez également utiliser les propriétés FLVPlayback pour modifier le comportement d'une enveloppe.

REMARQUE	Vous devez transférer votre fichier SWF d'enveloppe sur le serveur Web avec votre fichier SWF d'application correspondant à l'enveloppe à utiliser avec le composant FLVPlayback.

Sélection d'une enveloppe prédéfinie

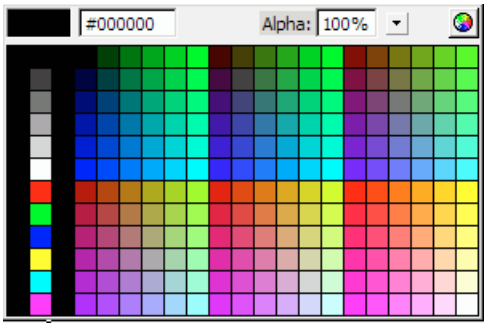
Vous pouvez choisir une enveloppe pour le composant FLVPlayback en cliquant sur la cellule `value` pour le paramètre `skin` dans l'Inspecteur des composants. Cliquez ensuite sur l'icône en forme de loupe pour ouvrir la boîte de dialogue Sélectionner une enveloppe qui vous permet de sélectionner une enveloppe ou de fournir une URL qui spécifie l'emplacement du fichier SWF d'enveloppe.



Boîte de dialogue Sélectionner une enveloppe de FLVPlayback

Les enveloppes répertoriées dans le menu contextuel Enveloppe se trouvent dans le dossier Flash Configuration/FLVPlayback Skins/ActionScript 3.0/. Vous pouvez créer des enveloppes et les faire apparaître dans cette boîte de dialogue ainsi que placer le fichier SWF dans le dossier. Le nom s'affiche dans le menu contextuel avec une extension .swf. Pour plus d'informations sur la création d'un ensemble d'enveloppes, reportez-vous à « Création d'une enveloppe », à la page 243.

Pour les enveloppes que vous affectez en définissant la propriété `skin`, en définissant le paramètre skin lors de la programmation ou avec ActionScript lors de l'exécution, vous pouvez affecter des valeurs de couleur et alpha (transparence) indépendamment du choix de l'enveloppe. Pour affecter des valeurs de couleur et alpha lors de la programmation, ouvrez le sélecteur de couleur dans la boîte de dialogue Sélectionner une enveloppe, comme illustré ci-dessous.



Pour choisir la couleur, cliquez sur une nuance dans le panneau ou entrez sa valeur numérique dans la zone de texte. Pour choisir la valeur alpha, faites glisser le curseur ou spécifiez un pourcentage dans la zone Alpha.

Pour affecter des valeurs de couleur et alpha lors de l'exécution, définissez les propriétés `skinBackgroundColor` et `skinBackgroundAlpha`. Définissez la propriété `skinBackgroundColor` sur une valeur 0xRRVVBB (rouge, vert, bleu). Définissez la propriété skinBackgroundAlpha sur un nombre compris entre 0,0 et 1,0. L'exemple suivant définit `skinBackgroundColor` sur 0xFF0000 (rouge) et skinBackgroundAlpha sur 0,5.

```
my_FLVPlybk.skinBackgroundColor = 0xFF0000;
my_FLVPlybk.skinBackgroundAlpha = .5;
```

Les valeurs par défaut sont celles choisies en dernier par l'utilisateur.

Si vous souhaitez appliquer une enveloppe au composant FLVPlayback à l'aide des composants de l'interface utilisateur personnalisée Lecture de fichiers FLV, sélectionnez Aucune dans le menu contextuel.

Application d'enveloppe aux composants particuliers de l'interface utilisateur personnalisée Lecture de fichiers FLV

Les composants de l'interface utilisateur personnalisée Lecture de fichiers FLV permettent de personnaliser l'apparence des commandes FLVPlayback dans le fichier FLA et d'observer les résultats lorsque vous affichez un aperçu de votre page Web. Cependant ces composants ne sont pas conçus pour être dimensionnés. Vous devez modifier un clip et son contenu pour qu'ils soient à une taille particulière. C'est pourquoi, il est généralement conseillé de placer le composant FLVPlayback sur la scène à la taille souhaitée, la propriété `scaleMode` étant définie sur `exactFit`.

Pour commencer, faites glisser simplement les composants de l'interface utilisateur personnalisée Lecture de fichiers FLV souhaités à partir du panneau Composants, placez-les où vous le souhaitez sur la scène et attribuez à chacun d'entre eux un nom d'occurrence.

Ces composants peuvent fonctionner sans code ActionScript. Si vous les placer dans le même scénario et sur la même image que le composant FLVPlayback, et qu'aucune enveloppe n'est définie au sein de ce dernier, le composant FLVPlayback s'y connectera automatiquement. Si vous disposez de plusieurs composants FLVPlayback sur la scène, ou si la commande personnalisée et l'occurrence FLVPlayback ne se trouvent pas sur le même scénario, vous devez alors intervenir.

Une fois que les composants sont sur la scène, modifiez-les comme vous le feriez avec tout autre symbole. Après avoir ouvert les composants, vous pouvez constater que la configuration de chacun diffère légèrement de celle des autres.

Composants Button

Les composants Button ont une structure similaire. Ces boutons sont les suivants : BackButton, ForwardButton, MuteButton, PauseButton, PlayButton, PlayPauseButton et StopButton. La plupart ont un seul clip sur l'image 1 avec l'occurrence placeholder_mc. Il s'agit généralement d'une occurrence de l'état normal du bouton, mais pas nécessairement. Sur l'image 2, il y a quatre clips sur la scène pour chaque état d'affichage : normal, survol, abaissé et désactivé. (A l'exécution, le composant n'accède jamais réellement à l'image2 ; ces clips sont placés ici pour faciliter les modifications et être forcés à être chargés dans le fichier SWF sans cocher la case Exporter dans la première image dans la boîte de dialogue Propriétés des symboles. Cependant, vous devez toujours sélectionner l'option Exporter pour ActionScript.)

Pour appliquer une enveloppe au bouton, vous devez simplement modifier chacun de ces clips. Vous pouvez modifier leur taille ainsi que leur apparence.

Un script ActionScript s'affiche en général sur l'image 1. Normalement, vous ne devez pas le modifier. Il arrête simplement la tête de lecture sur l'image 1 et indique quels clips utiliser pour quels états.

Boutons PlayPauseButton, MuteButton, FullScreenButton et CaptionButton

Les boutons PlayPauseButton, MuteButton, FullScreenButton et CaptionButton sont configurés différemment des autres ; ils possèdent une seule image avec deux calques, sans script. Cette image contient deux boutons situés l'un sur l'autre : dans le cas de PlayPauseButton, un bouton Lire et un bouton Pause ; dans le cas de MuteButton, un bouton de coupure du son et un bouton de restauration du son ; dans le cas de FullScreenButton, un bouton d'activation du mode plein écran et un bouton de désactivation du mode plein écran ; dans le cas de CaptionButton, un bouton d'affichage de la légende et un bouton de masquage de la légende. Pour appliquer une enveloppe à ces boutons, appliquez une enveloppe à chacun de ces deux boutons internes, comme décrit dans « Application d'enveloppe aux composants particuliers de l'interface utilisateur personnalisée Lecture de fichiers FLV », à la page 235; aucune action supplémentaire n'est requise.

Le bouton CaptionButton est dédié au composant FLVPlaybackCaptioning et doit y être exlusivement associé, et non pas au composant FLVPlayback.

Boutons BackButton et ForwardButton

La configuration des boutons BackButton et ForwardButton diffère également légèrement de celle des autres. Sur l'image 2, ils contiennent des clips supplémentaires que vous pouvez utiliser comme cadre autour d'un ou des deux boutons. Ces clips ne sont pas requis et n'ont pas de fonctionnalités spéciales ; ils sont uniquement fournis pour leur commodité. Pour les utiliser, faites-les simplement glisser sur la scène à partir du panneau Bibliothèque, puis placez-les là où vous le souhaitez. Si vous n'en voulez pas, ne les utilisez pas ou supprimez-les dans le panneau Bibliothèque.

La plupart de ces boutons, tels qu'ils sont fournis, sont basés sur un ensemble commun de clips pour que vous puissez modifier l'apparence de tous les boutons à la fois. Vous pouvez utiliser cette fonctionnalité ou remplacer ces clips communs et rendre l'apparence de chaque bouton différente.

Composant BufferingBar

Le composant BufferingBar est simple : il est constitué d'une animation qui s'affiche lorsqu'il entre en mémoire tampon, et ne nécessite pas de script ActionScript spécial pour être configuré. Par défaut, il s'agit d'une barre rayée déplacée de gauche à droite, sur laquelle est placé un masque rectangulaire pour lui donner un effet « bande zébrée », mais il n'y a rien de spécial sur cette configuration.

Bien que les barres de mise en mémoire tampon des fichiers SWF d'enveloppe utilisent l'échelle à 9 découpes car elles doivent être mises à l'échelle à l'exécution, le composant BufferingBar de l'interface utilisateur personnalisée FLV n'utilise pas et ne *peut pas* utiliser l'échelle à 9 découpes car il comporte des clips imbriqués. Si vous souhaitez modifier la largeur ou la longueur du composant BufferingBar, vous pouvez changer son contenu plutôt que sa dimension.

Composants SeekBar et VolumeBar

Les composants SeekBar et VolumeBar sont semblables, même si leurs fonctions sont différentes. Chacun possède des poignées, utilise les mêmes mécanismes de gestion des poignées et prend en charge les clips imbriqués qu'il contient pour suivre la progression.

Il existe de nombreux emplacements où le code ActionScript du composant FLVPlayback suppose que le point d'alignement (également appelé *point d'origine* ou *point zéro*) des composants SeekBar ou VolumeBar est situé dans le coin supérieur gauche du contenu. Il est donc important de conserver cette convention. Autrement, vous risquez de rencontrer des problèmes avec les poignées et avec les clips de progression et de fond.

Bien que les barres de recherche des fichiers SWF d'enveloppe utilisent l'échelle à 9 découpes car elles doivent être mises à l'échelle à l'exécution, le composant SeekBar de l'interface utilisateur personnalisée FLV n'utilise pas et ne *peut pas* utiliser l'échelle à 9 découpes car il comporte des clips imbriqués. Si vous souhaitez modifier la largeur ou la longueur du composant SeekBar, vous pouvez changer son contenu plutôt que sa dimension.

Poignée

Une occurrence du clip de poignée est située sur l'image 2. A l'instar des composants BackButton et ForwardButton, le composant n'accède jamais réellement à l'image 2 ; ces clips sont placés ici pour faciliter leur modification et pour être forcés à être chargés dans le fichier SWF sans cocher la case Exporter dans la première image de la boîte de dialogue Propriétés des symboles. Cependant, vous devez toujours sélectionner l'option Exporter pour ActionScript.

Vous pouvez remarquer que le clip de poignée est pourvu d'un rectangle à l'arrière-plan, la valeur alpha étant définie sur 0. Ce rectangle augmente la taille de la zone active de la poignée, ce qui facilite sa préhension sans changer son apparence, de la même façon que l'état actif d'un bouton. Comme la poignée est créée de façon dynamique à l'exécution, il doit s'agir d'un clip et non d'un bouton. Ce rectangle dont la valeur alpha est définie sur 0 n'est pas nécessaire et, en règle générale, vous pouvez remplacer l'intérieur de la poignée par l'image de votre choix. Cependant, il fonctionne mieux pour conserver le point d'alignement centré horizontalement au milieu du clip de poignée.

Le code ActionScript suivant est inséré sur l'image 1 du composant SeekBar afin de gérer la poignée :

```
stop();
handleLinkageID = "SeekBarHandle";
handleLeftMargin = 2;
handleRightMargin = 2;
handleY = 11;
```

L'appel de la fonction `stop()` est nécessaire en raison du contenu de l'image 2.

La deuxième ligne indique le symbole à utiliser comme poignée, et normalement vous n'avez pas besoin de la modifier si vous modifiez simplement l'occurrence du clip de poignée sur l'image 2. A l'exécution, le composant FLVPlayback crée une occurrence du clip spécifié sur la scène comme sœur de l'occurrence du composant Bar, ce qui signifie qu'elles ont le même clip parent. Ainsi, si votre barre se situe au niveau racine, votre poignée doit également se situer à ce niveau.

La variable `handleLeftMargin` détermine l'emplacement d'origine de la poignée (0 %) alors que la variable `handleRightMargin` détermine son emplacement final (100 %). Ces nombres indiquent les décalages par rapport aux extrémités gauche et droite de la barre : des nombres positifs marquent les limites à l'intérieur de la barre alors que des nombres négatifs marquent les limites à l'extérieur. Ces décalages indiquent où peut aller la poignée, en fonction de son point d'alignement. Si vous placez le point d'alignement au milieu de la poignée, les extrémités gauche et droite de celle-ci dépasseront les marges. Le point d'alignement d'un clip de barre de recherche doit être situé dans le coin supérieur gauche de son contenu pour fonctionner correctement.

La variable `handleY` détermine la position `y` de la poignée par rapport à l'occurrence de la barre. Elle est fonction des points d'alignement de chaque clip. Dans l'exemple de poignée, le point d'alignement est situé au niveau du sommet du triangle pour la placer par rapport à la partie visible, ignorant le rectangle d'état actif invisible. Le clip de barre doit donc conserver son point d'alignement dans le coin supérieur gauche de son contenu pour fonctionner correctement.

Par exemple, si un contrôle de barre est défini sur (100, 100), avec une largeur de 100 pixels, la poignée peut aller de 102 à 198 horizontalement et rester à 111 verticalement. Si vous modifiez les variables `handleLeftMargin` et `handleRightMargin` sur -2 et la variable `handleY` sur -11, la poignée peut aller de 98 à 202 horizontalement et rester à 89 verticalement.

Clips de progression et de fond

Le composant SeekBar contient un clip de *progression* et le composant VolumeBar contient un clip de *fond*, mais en pratique, ces deux composants possèdent chacun l'un ou l'autre de ces clips, n'en possèdent aucun ou possèdent les deux. Leur structure et leur comportement sont identiques, mais ils gèrent différentes valeurs. Un clip de progression se remplit à mesure du téléchargement du fichier FLV (ce qui est utile uniquement pour un téléchargement HTTP car il est toujours plein en cas de diffusion en flux continu à partir de FMS) et un clip de fond se remplit à mesure que la poignée se déplace de gauche à droite.

Le composant FLVPlayback recherche ces occurrences de clip en fonction d'un nom d'occurrence particulier. Ainsi votre occurrence de clip de progression doit posséder votre clip de barre comme parent et le nom d'occurrence progress_mc. L'occurrence de clip de fond doit être nommée fullness_mc.

Vous pouvez définir les clips de progression et de fond avec ou sans l'occurrence de clip fill_mc imbriquée dedans. Le clip fullness_mc du composant VolumeBar affiche la méthode *avec* le clip fill_mc, et le clip progress_mc du composant SeekBar affiche la méthode *sans* le clip fill_mc.

La méthode avec le clip fill_mc imbriqué est utile si vous souhaitez un remplissage qui ne peut pas être dimensionné sans déformer l'apparence.

Dans le clip fullness_mc du composant VolumeBar, l'occurrence du clip fill_mc imbriqué est masquée. Vous pouvez la masquer lorsque vous créez le clip ou créer un masque lors de l'exécution de façon dynamique. Si vous la masquez avec un clip, nommez cette occurrence **mask_mc**, puis configurez-la pour que fill_mc s'affiche comme si le pourcentage était de 100 %. Si vous ne masquez pas fill_mc, le masque créé de façon dynamique est rectangulaire et de même taille que fill_mc à 100 %.

Le clip fill_mc est révélé avec le masque d'une des deux façons, selon que fill_mc.slideReveal est défini sur `true` ou `false`.

Si fill_mc.slideReveal est défini sur `true`, alors fill_mc est déplacé de gauche à droite pour l'exposer à travers le masque. A 0 %, il est tout à fait à gauche, si bien que rien n'est affiché à travers le masque. A mesure que le pourcentage augmente, il se déplace vers la droite, jusqu'à 100 % ; il revient au pourcentage d'origine lors de sa création sur la scène.

Si fill_mc.slideReveal est défini sur `false` ou non défini (comportement pas défaut), le masque est redimensionné de gauche à droite pour révéler davantage de fill_mc. Lorsqu'il est sur 0 %, le masque est redimensionné horizontalement à 05, et à mesure que le pourcentage augmente, `scaleX` augmente, jusqu'à ce qu'il révèle l'intégralité de fill_mc lorsqu'il est sur 100 %. Cela ne correspond pas nécessairement à `scaleX` = 100 car mask_mc peut avoir été redimensionné au moment de sa création.

La méthode sans fill_mc est plus simple que la méthode avec fill_mc, mais elle déforme le remplissage horizontalement. Si vous ne souhaitez pas de déformation, vous devez utiliser fill_mc. Le clip progress_mc du composant SeekBar illuste cette méthode.

Le clip de progression ou de fond est redimensionné horizontalement en fonction du pourcentage. A 0 %, `scaleX` de l'occurrence est défini sur 0, la rendant invisible. A mesure que le pourcentage augmente, `scaleX` est réglé jusqu'à ce que le clip ait sa taille d'origine au moment de sa création sur la scène lorsqu'il est sur 100 %. De nouveau, cela ne correspond pas nécessairement à `scaleX` = 100 car cette occurrence de clip peut avoir été redimensionnée au moment de sa création.

Connexion aux composants de l'interface utilisateur personnalisée Lecture de fichiers FLV

Si vous placez vos composants de l'interface utilisateur personnalisée dans le même scénario et sur la même image que le composant FLVPlayback sans avoir défini la propriété `skin`, le composant FLVPlayback s'y connectera automatiquement sans nécessiter de code ActionScript.

Si vous disposez de plusieurs composants FLVPlayback sur la scène, ou si la commande personnalisée et le composant FLVPlayback ne se trouvent pas sur le même scénario, vous devez écrire du code ActionScript pour connecter vos composants de l'interface utilisateur personnalisée à l'occurrence du composant FLVPlayback. Vous devez d'abord nommer l'occurrence FLVPlayback, puis utiliser ActionScript pour affecter les occurrences des composants de l'interface utilisateur personnalisée Lecture de fichiers FLV aux propriétés FLVPlayback correspondantes. Dans l'exemple suivant, l'occurrence FLVPlayback est nommée my_FLVPlybk, les noms des propriétés FLVPlayback suivent les points (.) et les occurrences des commandes de l'interface utilisateur personnalisée Lecture de fichiers FLV figurent à droite du signe égal (=) :

```
//FLVPlayback instance = my_FLVPlybk
my_FLVPlybk.playButton = playbtn; // set playButton prop. to playbtn, etc.
my_FLVPlybk.pauseButton = pausebtn;
my_FLVPlybk.playPauseButton = playpausebtn;
my_FLVPlybk.stopButton = stopbtn;
my_FLVPlybk.muteButton = mutebtn;
my_FLVPlybk.backButton = backbtn;
```

```
my_FLVPlybk.forwardButton = forbtn;
my_FLVPlybk.volumeBar = volbar;
my_FLVPlybk.seekBar = seekbar;
my_FLVPlybk.bufferingBar = bufbar;
```

Exemple

Les procédures suivantes permettent de créer les commandes StopButton, PlayPauseButton, MuteButton et SeekBar personnalisées.

Pour créer les commandes StopButton, PlayPauseButton, MuteButton et SeekBar personnalisées :

1. Faites glisser le composant FLVPlayback sur la scène et nommez l'occurrence **my_FLVPlybk**.
2. Dans l'Inspecteur des composants, définissez le paramètre `source` sur **http://www.helpexamples.com/flash/video/cuepoints.flv**.
3. Définissez le paramètre Skin sur None.
4. Faites glisser un composant StopButton, PlayPauseButton et MuteButton sur la scène, puis placez-les au-dessus de l'occurrence FLVPlayback, en les empilant verticalement sur la gauche. Donnez à chaque bouton un nom d'occurrence dans l'Inspecteur des propriétés (par exemple **my_stopbttn**, **my_plypausbttn** et **my_mutebttn**).
5. Dans le panneau Bibliothèque, ouvrez le dossier Skins de FLVPlayback, puis son sous-dossier SquareButton.
6. Sélectionnez le clip SquareBgDown, puis double-cliquez dessus pour l'ouvrir sur la scène.
7. Cliquez avec le bouton droit (Windows) ou en appuyant sur la touche Contrôle (Macintosh), puis sélectionnez Sélectionner tout dans le menu, puis supprimez le symbole.
8. Sélectionnez l'outil Ovale, dessinez un ovale au même emplacement, puis définissez le remplissage sur bleu (**#0033FF**).
9. Dans l'Inspecteur des propriétés, définissez la largeur (L:) sur **40** et la hauteur (H:) sur **20**. Définissez la coordonnée x (X:) sur **0,0** et la coordonnée y (Y:) sur **0,0**.
10. Répétez les étapes 6 à 8 pour le clip SquareBgNormal, mais changez le remplissage en le définissant sur jaune (**#FFFF00**).
11. Répétez les étapes 6 à 8 pour le clip SquareBgOver, mais changez le remplissage en le définissant sur vert (**#006600**).

12. Modifiez les clips pour les différentes icônes de symboles dans les boutons (PauseIcon, PlayIcon, MuteOnIcon, MuteOffIcon et StopIcon). Vous pouvez trouver ces clips dans le panneau Bibliothèque sous FLV Playback Skins/*Etiquette* Button/Assets, où *Etiquette* correspond au nom du bouton, par exemple Lire, Pause, etc. Exécutez la procédure pour chacun :
    a. Cliquez sur l'option Sélectionner tout.
    b. Modifiez la couleur en la définissant sur rouge (**#FF0000**).
    c. Redimensionnez à 300 %.
    d. Modifiez l'emplacement X: du contenu sur **7.0** pour changer le placement horizontal de l'icône dans tous les états de bouton.

REMARQUE	En modifiant ainsi l'emplacement, vous n'avez pas à ouvrir chaque état de bouton et à déplacer l'occurrence de clip d'icône.

13. Cliquez sur la flèche Retour en bleu située au-dessus du scénario pour revenir à l'image 1 de la scène 1.
14. Faites glisser un composant SeekBar sur la scène, puis placez-le dans le coin inférieur droit de l'occurrence FLVPlayback.
15. Dans le panneau Bibliothèque, double-cliquez sur le composant SeekBar pour l'ouvrir sur la scène.
16. Redimensionnez-le à 400 %.
17. Sélectionnez le contour, puis définissez la couleur sur rouge (**#FF0000**).
18. Double-cliquez sur le composant SeekBarProgress dans le dossier FLVPlayback Skins/Seek Bar, puis définissez la couleur sur jaune (**#FFFF00**).
19. Double-cliquez sur le composant SeekBarHandle dans le dossier FLVPlayback Skins/Seek Bar, puis définissez la couleur sur rouge (**#FF0000**).
20. Cliquez sur la flèche Retour en bleu située au-dessus du scénario pour revenir à l'image 1 de la scène 1.
21. Sélectionnez l'occurrence SeekBar sur la scène et nommez-la **my_seekbar**.

22. Dans le panneau Actions situé sur l'image 1 du scénario, ajoutez une instruction import pour les classes Video, puis affectez les noms de bouton et de barre de recherche aux propriétés FLVPlayback correspondantes, comme indiqué dans l'exemple suivant :

```
import fl.video.*;
my_FLVPlybk.stopButton = my_stopbttn;
my_FLVPlybk.playPauseButton = my_plypausbttn;
my_FLVPlybk.muteButton = my_mutebttn;
my_FLVPlybk.seekBar = my_seekbar;
```

23. Appuyez sur Ctrl+Entrée pour tester l'animation.

Création d'une enveloppe

La meilleure façon de créer un fichier SWF d'enveloppe est de copier l'un des fichiers d'enveloppe fournis avec Flash, puis de l'utiliser comme point de départ. Vous pouvez trouver les fichiers FLA correspondant à ces enveloppes dans le dossier applicatif Flash, dans Configuration/FLVPlayback Skins/FLA/ActionScript 3.0/. Pour transformer votre fichier SWF d'enveloppe terminé en option dans la boîte de dialogue Sélectionner une enveloppe, placez-le dans le dossier Configuration/FLVPlayback Skins/ActionScript 3.0 du dossier applicatif Flash ou dans le dossier Configuration/FLVPlayback Skins/ActionScript 3.0 local d'un utilisateur.

Puisque vous pouvez définir la couleur d'une enveloppe indépendamment du choix de celle-ci, il n'est pas nécessaire de modifier le fichier FLA pour changer la couleur. Si vous créez une enveloppe de couleur spécifique qui ne doit pas être modifiable dans la boîte de dialogue Sélectionner une enveloppe, définissez `this.border_mc.colorMe = false;` dans le code ActionScript du fichier FLA d'enveloppe. Pour plus d'informations sur la définition d'une couleur d'enveloppe, reportez-vous à « Sélection d'une enveloppe prédéfinie », à la page 233.

Lorsque vous consultez les fichiers FLA d'enveloppe Flash installés, vous pouvez penser que certains éléments de la scène sont superflus, mais nombre de ces éléments sont insérés dans les calques de guide. La fonction Echelle 9 de l'aperçu en direct vous permet de voir rapidement ce qui apparaît réellement dans le fichier SWF.

Les sections ci-après présentent des personnalisations et des modifications plus complexes des clips SeekBar, BufferingBar et VolumeBar.

Utilisation de la disposition d'enveloppe

Lorsque vous ouvrez un fichier FLA d'enveloppe Flash, les clips de l'enveloppe sont disposés sur le scénario principal. Ces clips et le code ActionScript qui figurent sur la même image définissent la disposition des commandes à l'exécution.

Même si le calque de disposition ressemble beaucoup à l'apparence de l'enveloppe à l'exécution, son contenu n'est pas visible à l'exécution. Il est utilisé uniquement pour calculer l'emplacement des commandes. Les autres commandes sur la scène sont utilisées à l'exécution.

Le calque de disposition contient un espace réservé pour le composant FLVPlayback nommé video_mc. Toutes les autres commandes sont disposées par rapport à video_mc. Si vous commencez par l'un des fichiers FLA Flash et modifiez la taille des commandes, vous pouvez probablement corriger la disposition en déplaçant ces clips d'espace réservé.

Chacun de ces clips d'espace réservé a un nom d'occurrence particulier. Il s'agit des noms suivants : playpause_mc, play_mc, pause_mc, stop_mc, captionToggle_mc, fullScreenToggle_mc, back_mc, bufferingBar_mc, bufferingBarFill_mc, seekBar_mc, seekBarHandle_mc, seekBarProgress_mc, volumeMute_mc, volumeBar_mc et volumeBarHandle_mc. La partie recoloriée lorsque vous choisissez une couleur d'enveloppe s'appelle border_mc.

Le type de clip utilisé pour une commande n'est pas important. En règle générale, le clip d'état normal est utilisé pour les boutons. S'il s'agit des autres commandes, le clip correspondant à la commande en question est utilisé, mais c'est uniquement pour une raison de commodité. L'emplacement des coordonnées *x* (axe horizontal) et *y* (axe vertical) ainsi que la hauteur et la largeur de l'espace réservé sont les seuls élements qui importent.

Outre les commandes standard, vous pouvez posséder autant de clips supplémentaires que vous le souhaitez. Toutefois, l'option Exporter pour ActionScript correspondant aux symboles de bibliothèque de ces clips doit être cochée dans la boîte de dialogue Liaison. Les clips personnalisés du calque de disposition peuvent avoir un nom d'occurrence quelconque, autre que les noms d'occurrence réservés répertoriés ci-dessus. Un nom d'occurrence est nécessaire uniquement pour définir le code ActionScript sur les clips afin de déterminer la disposition.

Le clip border_mc est spécial. Si vous définissez la propriété `FlvPlayback.skinAutoHide` sur `true`, l'enveloppe s'affiche lorsque la souris est au-dessus du clip border_mc. Cela est important pour les enveloppes qui s'affichent hors des limites du lecteur vidéo. Pour plus d'informations sur la propriété `skinAutoHide`, reportez-vous à « Modification du comportement d'enveloppe », à la page 250.

Dans les fichiers FLA Flash, le clip border_mc est utilisé pour le chrome et pour la bordure entourant les boutons Avance et Retour.

Le clip border_mc est également la partie de l'enveloppe dont les valeurs alpha et de couleur sont modifiées par les propriétés `skinBackgroundAlpha` et `skinBackgroundColor`. Pour pouvoir personnaliser les valeurs de couleur et alpha, le code ActionScript du fichier FLA d'enveloppe doit inclure :

```
border_mc.colorMe = true;
```

ActionScript

En règle générale, le code ActionScript suivant s'applique à toutes les commandes. Certaines commandes contiennent du code ActionScript spécifique qui définit un comportement supplémentaire, et qui est expliqué dans la section correspondant à la commande en question.

Le code ActionScript initial constitue une section importante qui spécifie les noms de classe de l'état de chaque composant. Vous trouverez tous ces noms de classe dans le fichier SkinOverAll.fla. Le code a l'aspect suivant pour les boutons Pause et Lire, par exemple :

```
this.pauseButtonDisabledState = "fl.video.skin.PauseButtonDisabled";
this.pauseButtonDownState = "fl.video.skin.PauseButtonDown";
this.pauseButtonNormalState = "fl.video.skin.PauseButtonNormal";
this.pauseButtonOverState = "fl.video.skin.PauseButtonOver";
this.playButtonDisabledState = "fl.video.skin.PlayButtonDisabled";
this.playButtonDownState = "fl.video.skin.PlayButtonDown";
this.playButtonNormalState = "fl.video.skin.PlayButtonNormal";
this.playButtonOverState = "fl.video.skin.PlayButtonOver";
```

Les noms de classe n'ont pas de fichiers de classe externes ; ils sont simplement spécifiés dans la boîte de dialogue Liaison pour tous les clips de la bibliothèque.

Dans le composant ActionScript 2.0, certains clips présents sur la scène étaient utilisés lors de l'exécution. Dans le composant ActionScript 3.0, ces clips se trouvent toujours dans le fichier FLA, mais uniquement pour faciliter les modifications. Ils se trouvent tous désormais dans les calques de guide et ne sont pas exportés. Tous les éléments d'enveloppe de la bibliothèque sont définis pour être exportés dans la première image et sont créés de manière dynamique avec ce code, par exemple.

```
new fl.video.skin.PauseButtonDisabled()
```

La section suivante porte sur le code ActionScript qui définit les largeur et hauteur minimales de l'enveloppe. La boîte de dialogue Sélectionner une enveloppe affiche ces valeurs qui sont utilisées à l'exécution pour empêcher de redimensionner l'enveloppe en deça de sa taille minimale. Si vous ne souhaitez pas spécifier de taille minimale, laissez-les non définies ou affectez-leur une valeur inférieure ou égale à zéro.

```
// minimum width and height of video recommended to use this skin,
// leave as undefined or <= 0 if there is no minimum
this.minWidth = 270;
this.minHeight = 60;
```

Les propriétés suivantes peuvent être appliquées à chaque espace réservé :

Propriété	Description
`anchorLeft`	Boolean. Positionne la commande par rapport au côté gauche de l'occurrence FLVPlayback. Elle est définie par défaut sur `true` sauf si `anchorRight` est explicitement définie sur `true`, puis elle est définie par défaut sur `false`.
`anchorRight`	Boolean. Positionne la commande par rapport au côté droit de l'occurrence FLVPlayback. La valeur par défaut est `false`.
`anchorBottom`	Boolean. Positionne la commande par rapport au bas de l'occurrence FLVPlayback. Elle est définie par défaut sur `true` sauf si `anchorTop` est explicitement définie sur `true`, puis elle est définie par défaut sur `false`.
`anchorTop`	Boolean. Positionne la commande par rapport au haut de l'occurrence FLVPlayback. La valeur par défaut est `false`.

Si les deux propriétés `anchorLeft` et `anchorRight` sont définies sur `true`, la commande est redimensionnée à l'horizontale lors de l'exécution. Si les deux propriétés `anchorTop` et `anchorBottom` sont définies sur `true`, la commande est redimensionnée à la verticale lors de l'exécution.

Pour connaître les effets de ces propriétés, observez leur utilisation dans les enveloppes Flash. Les commandes BufferingBar et SeekBar sont les seules à pouvoir être redimensionnées, et elles sont disposées l'une au-dessus de l'autre. Les propriétés `anchorLeft` et `anchorRight` de ces deux commandes sont définies sur `true`. La propriété `anchorLeft` de toutes les commandes situées à gauche de BufferingBar et de SeekBar est définie sur `true`, et la propriété `anchorRight` de toutes les commandes situées à leur droite est définie sur `true`. La propriété `anchorBottom` de toutes les commandes est définie sur `true`.

Vous pouvez modifier les clips du calque de disposition pour créer une enveloppe dans laquelle les commandes sont situées en haut plutôt qu'en bas. Pour ce faire, vous devez simplement déplacer les commandes jusqu'en haut par rapport à `video_mc`, puis définir la propriété `anchorTop` de toutes les commandes sur `true`.

Barre de mise en mémoire tampon

La barre de mise en mémoire tampon possède deux clips : bufferingBar_mc et bufferingBarFill_mc. La position de chaque clip sur la scène par rapport à l'autre clip est importante car ce positionnement relatif est conservé. La barre de mise en mémoire tampon utilise deux clips distincts car ce composant redimensionne bufferingBar_mc, mais pas bufferingBarFill_mc.

L'échelle à 9 découpes est appliquée au clip bufferingBar_mc. De cette façon, les bordures ne sont pas déformées lors du redimensionnement. Le clip bufferingBarFill_mc est extrêmement large ; il le sera toujours suffisamment et vous n'aurez pas besoin de le redimensionner. A l'exécution, il est automatiquement masqué pour n'afficher que la partie située au-dessus du clip bufferingBar_mc étiré. Par défaut, les dimensions exactes du masque conservent une marge égale à gauche et à droite dans le clip bufferingBar_mc, basée sur la différence existant entre les positions *x* (axe horizontal) des clips bufferingBar_mc et bufferingBarFill_mc. Vous pouvez personnaliser le positionnement avec du code ActionScript.

Si votre barre de mise en mémoire tampon n'a pas besoin d'être redimensionnée ou n'utilise pas l'échelle à 9 découpes, vous pouvez la configurer comme le composant BufferingBar de l'interface utilisateur personnalisée Lecture de fichiers FLV. Pour plus d'informations, consultez la section « Composant BufferingBar », à la page 237.

La barre de mise en mémoire tampon dispose de la propriété suivante supplémentaire :

Propriété	Description
`fill_mc:MovieClip`	Indique le nom d'occurrence du remplissage de la barre de mise en mémoire tampon. Propriété par défaut de bufferingBarFill_mc.

Barre de recherche et barre de volume

La barre de recherche possède également deux clips : seekBar_mc et seekBarProgess_mc. La position de chaque clip sur le calque de disposition par rapport à l'autre clip est importante car ce positionnement relatif est conservé. Même si les deux clips sont redimensionnés, le clip seekBarProgress_mc ne peut pas être imbriqué dans le clip seekBar_mc car seekBar_mc utilise l'échelle à 9 découpes, qui ne fonctionne pas bien avec les clips imbriqués.

L'échelle à 9 découpes est appliquée au clip seekBar_mc. De cette façon, les bordures ne sont pas déformées lors du redimensionnement. Le clip seekBarProgress_mc peut également être redimensionné, mais il subit des déformations. Il n'utilise pas l'échelle à 9 découpes car il s'agit d'un remplissage dont l'apparence est correcte s'il est déformé.

Le clip seekBarProgress_mc fonctionne sans fill_mc, de la même façon qu'un clip progress_mc fonctionne dans les composants de l'interface utilisateur personnalisée Lecture de fichiers FLV. En d'autres mots, il n'est pas masqué et il est redimensionné à l'horizontale. Les dimensions exactes du clip seekBarProgress_mc à 100 % sont définies par les marges gauche et droite du clip seekBarProgress_mc. Ces dimensions sont, par défaut, égales et basées sur la différence entre les positions *x* (axe horizontal) des clips seekBar_mc et seekBarProgress_mc. Avec ActionScript, vous pouvez personnaliser les dimensions du clip de barre de recherche, comme indiqué dans l'exemple suivant :

```
this.seekBar_mc.progressLeftMargin = 2;
this.seekBar_mc.progressRightMargin = 2;
this.seekBar_mc.progressY = 11;
this.seekBar_mc.fullnessLeftMargin = 2;
this.seekBar_mc.fullnessRightMargin = 2;
this.seekBar_mc.fullnessY = 11;
```

Vous pouvez placer ce code dans le scénario du clip SeekBar ou avec l'autre code ActionScript sur le scénario principal. Si vous personnalisez avec du code au lieu de modifier la disposition, il n'est pas nécessaire de placer le remplissage sur la scène. Il doit simplement être dans la bibliothèque, défini sur Exporter pour ActionScript sur l'image 1 avec le nom de classe approprié.

Comme dans le composant SeekBar de l'interface utilisateur personnalisée Lecture de fichiers FLV, vous pouvez créer un clip de fond pour la barre de recherche. Si vous n'avez pas besoin de redimensionner votre barre de recherche, ou si elle est redimensionnée sans utiliser l'échelle à 9 découpes, vous pouvez configurer les clips progress_mc ou fullness_mc à l'aide des méthodes utilisées pour les composants de l'interface utilisateur personnalisée Lecture de fichiers FLV. Pour plus d'informations, consultez la section « Clips de progression et de fond », à la page 239.

Comme la barre de volume des enveloppes Flash ne peut pas être redimensionnée, elle est construite de la même façon que le composant VolumeBar de l'interface utilisateur personnalisée Lecture de fichiers FLV. Pour plus d'informations, consultez la section « Composants SeekBar et VolumeBar », à la page 237. L'exception concerne la poignée dont l'implémentation est différente. Pour plus d'informations à ce sujet, consultez la section suivante.

Poignée

Les poignées SeekBar et VolumeBar sont placées sur le calque de disposition en regard de la barre. Par défaut, les marges gauche et droite et les valeurs de l'axe *y* de la poignée sont définies par sa position par rapport au clip de barre. La marge gauche est définie par la différence entre l'emplacement des *x* (axe horizontal) de la poignée et l'emplacement des *x* (axe horizontal) de la barre, la marge droite étant égale à la marge gauche. Vous pouvez personnaliser ces valeurs via ActionScript dans le clip SeekBar ou VolumeBar. L'exemple suivant applique le même code ActionScript que celui utilisé dans les composants de l'interface utilisateur personnalisée Lecture de fichiers FLV :

```
this.seekBar_mc.handleLeftMargin = 2;
this.seekBar_mc.handleRightMargin = 2;
this.seekBar_mc.handleY = 11;
```

Vous pouvez placer ce code dans le scénario du clip SeekBar ou avec l'autre code ActionScript sur le scénario principal. Si vous personnalisez avec du code au lieu de modifier la disposition, il n'est pas nécessaire de placer la poignée sur la scène. Elle doit simplement être dans la bibliothèque, définie sur Exporter pour ActionScript sur l'image 1 avec le nom de classe approprié.

Au-delà de ces propriétés, les poignées sont de simples clips, configurés de la même façon que dans les composants de l'interface utilisateur personnalisée Lecture de fichiers FLV. Les deux poignées sont dotées d'arrière-plans rectangulaires, la propriété `alpha` étant définie sur 0. Elles sont uniquement destinées à augmenter la zone active et ne sont pas obligatoires.

Clips d'arrière-plan et de premier plan

Les clips chrome_mc et forwardBackBorder_mc sont implémentés comme clips d'arrière-plan.

Parmi les clips ForwardBackBorder, ForwardBorder et BackBorder situés sur la scène et les boutons d'espace réservé Avance et Retour, ForwardBackBorder est le seul à ne *pas* être sur un calque de guide. Ce clip figure uniquement dans les enveloppes qui utilisent réellement les boutons Avance et Retour.

Toutefois, ces clips doivent être exportés pour ActionScript sur l'image 1 de la bibliothèque.

Modification du comportement d'enveloppe

Les propriétés `bufferingBarHidesAndDisablesOthers` et `skinAutoHide` permettent de personnaliser le comportement de l'enveloppe FLVPlayback.

Si vous définissez la propriété `bufferingBarHidesAndDisablesOthers` sur `true`, le composant FLVPlayback masque SeekBar et sa poignée, et désactive les boutons Lire et Pause lorsqu'il entre dans l'état de mise en mémoire tampon. Cela peut être utile lorsqu'un fichier FLV est diffusé en flux continu à partir de FMS par le biais d'une connexion lente, la valeur de la propriété `bufferTime` étant élevée (10, par exemple). Dans cette situation, un utilisateur impatient peut essayer de lancer la recherche en cliquant sur les boutons Lire et Pause, ce qui peut retarder encore davantage la lecture du fichier. Pour empêcher cela, définissez `bufferingBarHidesAndDisablesOthers` sur `true`, puis désactivez l'élément SeekBar et les boutons Pause et Lire alors que le composant est en état de mise en mémoire tampon.

La propriété `skinAutoHide` affecte uniquement les fichiers SWF d'enveloppe prédéfinis, mais pas les commandes créées dans les composants de l'interface utilisateur personnalisée Lecture de fichiers FLV. Si cette propriété est définie sur `true`, le composant FLVPlayback masque l'enveloppe lorsque la souris n'est pas sur la zone d'affichage. La valeur par défaut de cette propriété est `false`.

Utilisation d'un fichier SMIL

Afin de gérer plusieurs flux continus pour plusieurs bandes passantes, la classe VideoPlayer utilise une classe d'assistance (NCManager) qui prend en charge un sous-ensemble de SMIL. Le langage SMIL permet d'identifier l'emplacement du flux vidéo, la disposition (largeur et hauteur) du fichier FLV ainsi que les fichiers FLV source qui correspondent aux différentes bandes passantes. Il peut également être utilisé pour indiquer la vitesse de transmission et la durée du fichier FLV.

Utilisez le paramètre `source` ou la propriété `FLVPlayback.source` (ActionScript) pour spécifier l'emplacement d'un fichier SMIL. Pour plus d'informations, reportez-vous à « La source », à la page 215 et à la propriété `FLVPlayback.source` dans le *Guide de référence du langage et des composants ActionScript 3.0*.

L'exemple suivant affiche un fichier SMIL qui diffuse en continu plusieurs fichiers FLV à partir d'un serveur FMS à l'aide du protocole :

```
<smil>
    <head>
        <meta base="rtmp://myserver/myapp/" />
        <layout>
            <root-layout width="240" height="180" />
        </layout>
    </head>
    <body>
       <switch>
           <ref src="myvideo_cable.flv" dur="3:00.1"/>
           <video src="myvideo_isdn.flv" system-bitrate="128000"
  dur="3:00.1"/>
           <video src="myvideo_mdm.flv" system-bitrate="56000"
  dur="3:00.1"/>
       </switch>
    </body>
</smil>
```

La balise `<head>` peut contenir les balises `<meta>` et `<layout>`. La balise `<meta>` ne prend en charge que l'attribut `base`, qui permet de spécifier l'URL de la vidéo en flux continu (RTMP à partir d'un serveur FMS).

La balise `<layout>` ne prend en charge que l'élément `root-layout` qui permet de définir les attributs `height` et `width` et donc de déterminer la taille de la fenêtre dans laquelle le fichier FLV est rendu. Ces attributs acceptent uniquement des valeurs en pixels et non des pourcentages.

Vous pouvez insérer dans le corps du fichier SMIL un lien vers un fichier source FLV ou, si vous diffusez en continu plusieurs fichiers pour plusieurs bandes passantes d'un serveur FMS (comme dans l'exemple précédent), vous pouvez utiliser la balise `<switch>` pour dresser la liste des fichiers source.

Les balises `video` et `ref` situées dans la balise `<switch>` sont synonymes : elles peuvent utiliser toutes deux l'attribut `src` pour spécifier des fichiers FLV. Par ailleurs, chacune peut utiliser les attributs `region`, `system-bitrate` et `dur` pour spécifier la région, la bande passante minimale requise et la durée du fichier FLV.

Dans la balise `<body>`, une seule occurrence des balises `<video>`, `<src>` ou `<switch>` est autorisée.

L'exemple suivant affiche le téléchargement progressif d'un seul fichier FLV qui n'utilise pas de détection de bande passante :

```
<smil>
     <head>
       <layout>
         <root-layout width="240" height="180" />
       </layout>
     </head>
     <body>
       <video src=""myvideo.flv" />
     </body>
</smil>
```

<smil>

Disponibilité

Flash Professional 8.

Utilisation

```
<smil>
...
child tags
...
</smil>
```

Attributs

Aucun.

Balises enfants

`<head>, <body>`

Balise parent

Aucune.

Description

Balise de niveau supérieur qui identifie un fichier SMIL.

Exemple

L'exemple suivant affiche un fichier SMIL spécifiant trois fichiers FLV :

```
<smil>
    <head>
        <meta base="rtmp://myserver/myapp/" />
        <layout>
            <root-layout width="240" height="180" />
        </layout>
    </head>
    <body>
        <switch>
         <ref src="myvideo_cable.flv" dur="3:00.1"/>
         <video src="myvideo_isdn.flv" system-bitrate="128000"
  dur="3:00.1"/>
         <video src="myvideo_mdm.flv" system-bitrate="56000"  dur="3:00.1"/
  >
      </switch>
    </body>
</smil>
```

<head>

Disponibilité

Flash Professional 8.

Utilisation

```
<head>
...
child tags
...
</head>
```

Attributs

Aucune.

Balises enfants

`<meta>`, `<layout>`

Balise parent

`<smil>`

Description

Prenant en charge les balises `<meta>` et `<layout>`, elle indique l'emplacement et la disposition par défaut (hauteur et largeur) des fichiers FLV source.

Exemple

L'exemple suivant définit la disposition racine sur 240 x 180 pixels :

```
<head>
  <meta base="rtmp://myserver/myapp/" />
  <layout>
    <root-layout width="240" height="180" />
  </layout>
</head>
```

<meta>

Disponibilité

Flash Professional 8.

Utilisation

```
<meta/>
```

Attributs

`base`

Balises enfants

`<layout>`

Balise parent

Aucune.

Description

Contient l'attribut `base` qui spécifie l'emplacement (URL RTMP) des fichiers FLV source.

Exemple

L'exemple suivant affiche une balise meta pour indiquer l'emplacement de base sur `myserver`:

```
<meta base="rtmp://myserver/myapp/" />
```

<layout>

Disponibilité

Flash Professional 8.

Utilisation

```
<layout>
...
child tags
...
</layout>
```

Attributs

Aucune.

Balises enfants

```
<root-layout>
```

Balise parent

```
<meta>
```

Description

Indique la largeur et la hauteur du fichier FLV.

Exemple

L'exemple suivant spécifie la disposition sur 240 x 180 pixels :

```
<layout>
  <root-layout width="240" height="180" />
</layout>
```

<root-layout>

Disponibilité

Flash Professional 8.

Utilisation

```
<root-layout...attributes.../>
```

Attributs

Largeur, hauteur

Balises enfants

Aucune.

Balise parent

`<layout>`

Description

Indique la largeur et la hauteur du fichier FLV.

Exemple

L'exemple suivant spécifie la disposition sur 240 x 180 pixels :

```
<root-layout width="240" height="180" />
```

<body>

Disponibilité

Flash Professional 8.

Utilisation

```
<body>
...
child tags
...
</body>
```

Attributs

Aucune.

Balises enfants

`<video>`, `<ref>`, `<switch>`

Balise parent

`<smil>`

Description

Contient les balises `<video>`, `<ref>` et `<switch>` qui indiquent le nom du fichier FLV source, la bande passante minimale et la durée du fichier FLV. L'attribut `system-bitrate` est uniquement pris en charge pour utiliser la balise `<switch>`. Dans la balise `<body>`, une seule occurrence des balises `<video>`, `<ref>` ou `<switch>` est autorisée.

Exemple

L'exemple suivant spécifie trois fichiers FLV, deux utilisant la balise `video` et un utilisant la balise `ref` :

```
<body>
  <switch>
    <ref src="myvideo_cable.flv" dur="3:00.1"/>
    <video src="myvideo_isdn.flv" system-bitrate="128000" dur="3:00.1"/>
    <video src="myvideo_mdm.flv" system-bitrate="56000"  dur="3:00.1"/>
  </switch>
</body>
```

<video>

Disponibilité

Flash Professional 8.

Utilisation

`<video...attributes.../>`

Attributs

`src, system-bitrate, dur`

Balises enfants

Aucune.

Balise parent

`<body>`

Description

Synonyme de la balise `<ref>`. Elle prend en charge les attributs `src` et `dur`, qui spécifient le nom du fichier FLV source ainsi que sa durée. L'attribut `dur` prend en charge les formats horaires complet (00:03:00:01) et partiel (03:00:01).

Exemple

L'exemple suivant définit la source et la durée d'une vidéo :

```
<video src="myvideo_mdm.flv" dur="3:00.1"/>
```

<ref>

Disponibilité

Flash Professional 8.

Utilisation

```
<ref...attributes.../>
```

Attributs

`src, system-bitrate, dur`

Balises enfants

Aucune.

Balise parent

`<body>`

Description

Synonyme de la balise `<video>`. Elle prend en charge les attributs `src` et `dur`, qui spécifient le nom du fichier FLV source ainsi que sa durée. L'attribut `dur` prend en charge les formats horaires complet (00:03:00:01) et partiel (03:00:01).

Exemple

L'exemple suivant définit la source et la durée d'une vidéo :

```
<ref src="myvideo_cable.flv" dur="3:00.1"/>
```

<switch>

Disponibilité

Flash Professional 8.

Utilisation

```
<switch>
...
child tags
...
<switch/>
```

Attributs

Aucune.

Balises enfants

`<video>`, `<ref>`

Balise parent

`<body>`

Description

Utilisée avec les balises enfants `<video>` ou `<ref>` pour répertorier les fichiers FLV destinés à la diffusion de vidéo en continu via plusieurs bandes passantes. La balise `<switch>` prend en charge l'attribut `system-bitrate`, qui spécifie la bande passante minimale ainsi que les attributs `src` et `dur`.

Exemple

L'exemple suivant spécifie trois fichiers FLV, deux utilisant la balise `video` et un utilisant la balise `ref` :

```
<switch>
  <ref src="myvideo_cable.flv" dur="3:00.1"/>
  <video src="myvideo_isdn.flv" system-bitrate="128000" dur="3:00.1"/>
  <video src="myvideo_mdm.flv" system-bitrate="56000"  dur="3:00.1" />
</switch>
```

CHAPITRE 6

Utilisation du composant FLVPlaybackCaptioning

6

Le composant FLVPlayback vous permet d'inclure un lecteur vidéo à votre application Adobe Flash CS3 Professional afin de lire des fichiers FLV (Vidéo Adobe Flash) téléchargés et des fichiers FLV diffusés en flux continu. Pour plus d'informations sur le composant FLVPlayback, consultez le Chapitre 5, « Utilisation du composant FLVPlayback », à la page 207.

Le composant FLVPlaybackCaptioning vous permet d'inclure la prise en charge du sous-titrage fermé pour votre vidéo. Le composant de sous-titrage prend en charge le format XML Timed Text standard W3C et inclut les fonctionnalités suivantes :

- **Sous-titrage avec les points de repère d'événement intégrés**. Vous permet d'associer les points de repère d'événement intégrés dans un fichier FLV au format XML de manière à fournir le sous-titrage plutôt que d'utiliser un fichier XML Timed Text.
- **Sous-titrage FLVPlayback multiple**. Vous permet de créer plusieurs occurrences de sous-titrage FLVPlayback pour plusieurs occurrences FLVPlayback.
- **Contrôle bouton bascule**. Permet l'interaction de l'utilisateur avec le sous-titrage par l'intermédiaire d'un bouton bascule de sous-titrage.

Utilisation du composant FLVPlaybackCaptioning

Vous pouvez utiliser le composant FLVPlaybackCaptioning avec un ou plusieurs composants FLVPlayback. Dans le scénario le plus simple, vous faites glisser un composant FLVPlayback sur la scène, un composant FLVPlaybackCaptioning sur la même scène, identifiez l'URL de votre légende et définissez des légendes à afficher. En outre, vous pouvez également définir divers paramètres pour personnaliser votre sous-titrage FLVPlayback.

Ajout de sous-titrage au composant FLVPlayback

Vous pouvez ajouter le composant FLVPlaybackCaptioning à un composant FLVPlayback quelconque. Pour plus d'informations sur l'ajout de composants FLVPlayback à votre application, consultez la section « Création d'une application avec le composant FLVPlayback », à la page 210.

Pour ajouter le composant FLVPlaybackCaptioning à partir du panneau Composants :

1. Dans le panneau Composants, ouvrez le dossier Video.
2. Faites glisser le composant FLVPlaybackCaptioning (ou double-cliquez sur celui-ci) et ajoutez-le sur la même scène que le composant FLVPlayback auquel vous souhaitez ajouter le sous-titrage.

REMARQUE

Adobe fournit deux fichiers d'exemple pour vous aider à découvrir le composant FLVPlaybackCaptioning rapidement : caption_video.flv (un exemple de composant FLVPlayback) et caption_video.xml (un exemple de sous-titrage). Accédez à ces fichiers à l'adresse http://www.helpexamples.com/flash/video_fr.

3. (Facultatif) Faites glisser le composant CaptionButton sur la même scène que les composants FLVPlayback et FLVPlaybackCaptioning. Le composant CaptionButton permet à l'utilisateur d'activer et de désactiver le sous-titrage.

REMARQUE

Pour activer le composant CaptionButton, vous devez le faire glisser sur la même scène que les composants FLVPlayback et FLVPlaybackCaptioning.

4. Dans l'onglet Paramètres de l'Inspecteur des propriétés, après avoir sélectionné le composant FLVPlaybackCaptioning sur la scène, spécifiez les informations suivantes requises :
   - Définissez `showCaptions` sur `true`.
   - Spécifiez la `source` du fichier XML Timed Text à télécharger.

CONSEIL	Lors du travail exécuté dans Flash pour tester vos légendes, vous devez définir la propriété `showCaptions` sur `true`. Cependant, si vous incluez le composant `CaptionButton` pour permettre aux utilisateurs d'activer et de désactiver le sous-titrage, vous devez définir la propriété `showCaptions` sur `false`.

   D'autres paramètres sont disponibles pour vous aider à personnaliser le composant FLVPlaybackCaptioning. Pour plus d'informations, consultez la section « Personnalisation du composant FLVPlaybackCaptioning », à la page 271 et le *Guide de référence du langage et des composants ActionScript 3.0*.
5. Choisissez Contrôle > Tester l'animation pour démarrer la vidéo.

Close collapsed procedure

Pour créer une occurrence de façon dynamique à l'aide d'ActionScript :

1. Faites glisser le composant FLVPlayback du panneau Composants vers le panneau Bibliothèque (Fenêtre> Bibliothèque).
2. Faites glisser le composant FLVPlaybackCaptioning du panneau Composants vers le panneau Bibliothèque.
3. Ajoutez le code suivant dans le panneau Actions, sur l'image 1 du scénario.

```
import fl.video.*;
```

REMARQUE	L'exemple suivant concerne Adobe Flash CS3 pour Windows. L'emplacement des enveloppes FLVPlayback sur Macintosh est le suivant : Disque dur Macintosh/ Applications/Adobe Flash CS3/ Configuration/FLVPlayback Skins/ActionScript 3.0/SkinUnderPlaySeekCaption.swf.

```
var my_FLVPlybk = new FLVPlayback();
my_FLVPlybk.x = 100;
my_FLVPlybk.y = 100;
addChild(my_FLVPlybk);
my_FLVPlybk.skin = "install_drive:/Program Files/Adobe/Adobe Flash CS3/
  en/Configuration/FLVPlayback Skins/ActionScript 3.0/
  SkinUnderPlaySeekCaption.swf";
my_FLVPlybk.source = "http://www.helpexamples.com/flash/video/
  caption_video.flv";
```

```
var my_FLVPlybkcap = new FLVPlaybackCaptioning();
addChild (my_FLVPlybkcap);
my_FLVPlybkcap.source = "http://www.helpexamples.com/flash/video/
  caption_video.xml";
my_FLVPlybkcap.showCaptions = true;
```

4. Remplacez *lecteur_installation* par le lecteur sur lequel vous avez installé Flash, puis modifiez le chemin pour refléter l'emplacement du dossier Skins de votre installation.

REMARQUE	Si vous créez une occurrence FLVPlayback avec ActionScript, vous devez également lui affecter une enveloppe de manière dynamique en définissant la propriété skin avec ActionScript. Lorsque vous appliquez une enveloppe à l'aide d'ActionScript, elle n'est pas automatiquement publiée avec le fichier SWF. Copiez le fichier SWF d'enveloppe et le fichier SWF d'application sur votre serveur ; si vous ne le faites pas, le fichier SWF d'enveloppe ne sera pas disponible lorsque l'utilisateur l'exécutera.

Définition des paramètres du composant FLVPlaybackCaptioning

Vous pouvez définir les paramètres suivants pour chaque occurrence du composant FLVPlaybackCaptioning dans l'Inspecteur des propriétés ou l'Inspecteur des composants afin de personnaliser le composant. La liste suivante identifie et explique brièvement les propriétés :

- `autoLayout`. Détermine si le composant FLVPlaybackCaptioning contrôle la taille de la zone de sous-titrage. La valeur par défaut est `true`.
- `captionTargetName`. Identifie le nom d'occurrence TextField ou MovieClip contenant des légendes. La valeur par défaut est auto.
- `flvPlaybackName`. Identifie le nom d'occurrence FLVPlayback qui doit recevoir une légende. La valeur par défaut est auto.
- `simpleFormatting`. Limite la mise en forme des instructions du fichier XML Timed Text lorsqu'elle est définie sur true. La valeur par défaut est `false`.
- `showCaptions`. Détermine si les légendes doivent être affichées. La valeur par défaut est `true`.
- `source`. Identifie l'emplacement du fichier XML Timed Text.

Pour plus d'informations sur tous les paramètres du composant FLVPlaybackCaptioning, consultez le *Guide de référence du langage et des composants ActionScript 3.0*.

Définition du paramètre source

Utilisez le paramètre `source` pour spécifier le nom et l'emplacement du fichier XML Timed Text qui contient les légendes de votre animation. Entrez le chemin de l'URL directement dans la cellule source de l'Inspecteur des composants.

Affichage des légendes

Pour afficher le sous-titrage, définissez le paramètre `showCaptions` sur `true`.

Pour plus d'informations sur tous les paramètres du composant FLVPlaybackCaptioning, consultez le *Guide de référence du langage et des composants ActionScript 3.0*.

Dans les exemples précédents, vous avez appris à créer et activer le composant FLVPlaybackCaptioning pour afficher les légendes. Vous pouvez utiliser deux sources pour vos légendes : (1) un fichier XML Timed Text contenant vos légendes ou (2) un fichier XML incluant le texte de sous-titrage que vous associez aux points de repère d'événement intégrés.

Utilisation des légendes Timed Text

Le composant FLVPlaybackCaptioning permet de créer des légendes pour le composant FLVPlayback associé en téléchargeant un fichier XML Timed Text (TT). Pour plus d'informations sur le format Timed Text, consultez les informations AudioVideo Timed Text disponibles sur le site http://www.w3.org.

Cette section présente une vue d'ensemble des balises Timed Text prises en charge, des balises de fichier de sous-titrage requises et un exemple de fichier XML Timed Text. Pour obtenir des informations détaillées sur toutes les balises Timed Text prises en charge, consultez la section Annexe A, « Balises Timed Text ».

Le composant FLVPlaybackCaptioning prend en charge les balises Timed Text suivantes :

- Prise en charge du formatage des paragraphes
  - Permet d'aligner un paragraphe à droite, à gauche ou au centre
- Prise en charge du formatage de texte
  - Permet de définir la taille du texte avec des tailles de pixel absolues ou le style delta (par exemple, +2, -4)
  - Permet de définir la couleur et la police du texte
  - Permet de définir le texte en gras et en italique
  - Permet de justifier le texte

- Autre prise en charge du formatage
  - Permet de définir la couleur d'arrière-plan du composant TextField pour les légendes
  - Permet de définir la couleur d'arrière-plan du composant TextField pour les légendes sur transparent (alpha 0)
  - Permet de définir le retour à la ligne du composant TextField pour les légendes (activé ou désactivé)

Le composant FLVPlaybackCaptioning correspond au code de temps du fichier FLV. Chaque légende doit disposer d'un attribut `begin` qui détermine à quel moment la légende doit apparaître. Si la légende ne dispose pas d'un attribut `dur` ou `end`, elle disparaît à l'apparition de la légende suivante ou lorsque le fichier FLV se termine.

L'exemple suivant illustre un fichier XML Timed Text. Ce fichier (caption_video.xml) contient les légendes du fichier caption_video.flv. Accédez à ces fichiers à l'adresse http://www.helpexamples.com/flash/video_fr.

```
<?xml version="1.0" encoding="UTF-8"?>
 <tt xml:lang="en" xmlns="http://www.w3.org/2006/04/ttaf1"
  xmlns:tts="http://www.w3.org/2006/04/ttaf1#styling">
  <head>
   <styling>
      <style id="1" tts:textAlign="right"/>
      <style id="2" tts:color="transparent"/>
      <style id="3" style="2" tts:backgroundColor="white"/>
      <style id="4" style="2 3" tts:fontSize="20"/>
   </styling>
  </head>
  <body>
   <div xml:lang="en">
    <p begin="00:00:00.00" dur="00:00:03.07">I had just joined <span
  tts:fontFamily="monospaceSansSerif,proportionalSerif,TheOther"tts:fontSi
  ze="+2">Macromedia</span> in 1996,</p>
    <p begin="00:00:03.07" dur="00:00:03.35">and we were trying to figure
  out what to do about the internet.</p>
    <p begin="00:00:06.42" dur="00:00:03.15">And the company was in dire
  straights at the time.</p>
    <p begin="00:00:09.57" dur="00:00:01.45">We were a CD-ROM authoring
  company,</p>
    <p begin="00:00:11.42" dur="00:00:02.00">and the CD-ROM business was
  going away.</p>
    <p begin="00:00:13.57" dur="00:00:02.50">One of the technologies I
  remember seeing was Flash.</p>
    <p begin="00:00:16.47" dur="00:00:02.00">At the time, it was called
  <span tts:fontWeight="bold" tts:color="#ccc333">FutureSplash</span>.</p>
    <p begin="00:00:18.50" dur="00:00:01.20">So this is where Flash got its
  start.</p>
    <p begin="00:00:20.10" dur="00:00:03.00">This is smart sketch running on
  the <span tts:fontStyle="italic">EU-pin computer</span>,</p>
```

```
    <p begin="00:00:23.52" dur="00:00:02.00">which was the first product
   that FutureWave did.</p>
    <p begin="00:00:25.52" dur="00:00:02.00">So our vision for this product
   was to</p>
    <p begin="00:00:27.52" dur="00:00:01.10">make drawing on the computer</
   p>
    <p begin="00:00:29.02" dur="00:00:01.30" style="1">as <span
   tts:color="#ccc333">easy</span> as drawing on paper.</p>
    </div>
  </body>
</tt>
```

Utilisation des points de repère avec le sous-titrage

Les points de repère vous permettent d'interagir avec une vidéo ; par exemple, vous pouvez affecter la lecture d'un fichier FLV ou l'affichage du texte à certaines périodes sur la vidéo. Si vous ne disposez pas d'un fichier XML Timed Text à utiliser avec un fichier FLV, vous pouvez intégrer des points de repère d'événement dans un fichier FLV, puis les associer au texte. Cette section fournit des informations sur les normes des points de repère du composant FLVPlaybackCaptioning et décrit rapidement comment associer ces points de repère au texte pour le sous-titrage. Pour plus d'informations sur la façon d'intégrer les points de repère d'événement à l'aide de l'Assistant d'importation vidéo ou de l'encodeur vidéo de Flash, consultez le chapitre 16, « Utilisation de la vidéo » du guide *Utilisation de Flash*.

Présentation des normes des points de repère du composant FLVPlaybackCaptioning

Dans les métadonnées du fichier FLV, un point de repère est représenté sous la forme d'un objet pourvu des propriétés suivantes : `name`, `time`, `type` et `parameters`. Les points de repère ActionScript du composant FLVPlaybackCaptioning possèdent les attributs suivants :

- **name**. La propriété `name` est une chaîne qui contient le nom affecté au point de repère. La propriété `name` doit commencer par le préfixe *`fl.video.caption.2.0.`*, suivi d'une chaîne. La chaîne est une série d'entiers positifs incrémentée à chaque création pour que chaque nom reste unique. Le préfixe inclut le numéro de version qui correspond également au numéro de version du composant FLVPlayback. Pour Adobe Flash CS3, vous devez définir le numéro de version sur `2.0`.
- **time**. La propriété `time` représente l'heure à laquelle la légende doit s'afficher.
- **type**. La propriété `type` est une chaîne dont la valeur est « `event` ».

- **parameters** La propriété `parameters` est un tableau qui prend en charge les paires nom/valeur suivantes :
  - `text:String`. Texte au format HTML de la légende. Ce texte est directement transmis à la propriété `TextField.htmlText`. Le composant FLVPlaybackCaptioning prend en charge une propriété `text:n` supplémentaire qui prend en charge l'utilisation de plusieurs pistes de langue. Pour plus d'informations, consultez la section « Prise en charge de plusieurs pistes de langue avec des points de repère intégrés », à la page 270.
  - `endTime:Number`. Heure à laquelle la légende doit disparaître. Si vous ne spécifiez pas cette propriété, le composant FLVPlaybackCaptioning suppose qu'il ne s'agit pas d'un nombre (NaN), et la légende s'affiche jusqu'à la fin de l'exécution du fichier FLV (l'occurrence FLVPlayback distribue l'événement `VideoEvent.COMPLETE`). Spécifiez la propriété `endTime:Number` en secondes.`backgroundColor:uint`.Ce paramètre définit la propriété `TextField.backgroundColor`. Cette propriété est facultative.
  - `backgroundColorAlpha:Boolean`. Si le paramètre backgroundColor possède un alpha de 0 %, il définit alors `TextField.background = !backgroundColor`. Cette propriété est facultative.
  - `wrapOption:Boolean`. Ce paramètre définit la propriété TextField.wordWrap. Cette propriété est facultative.

Présentation de la création du sous-titrage pour les points de repère d'événement intégrés

Si vous ne disposez pas d'un fichier XML Timed Text contenant les légendes de votre fichier FLV, vous pouvez créer le sous-titrage en associant le fichier XML qui contient les légendes aux points de repère d'événement intégrés. Le fichier d'exemple XML vous invite à effectuer les étapes suivantes pour créer des points de repère d'événement intégrés à votre vidéo :

- Ajoutez les points de repère d'événement (conformément aux normes FLVPlaybackCaptioning) et codez la vidéo.
- Dans Flash, faites glisser un composant FLVPlayback et un composant FLVPlaybackCaptioning sur la scène.
- Définissez les propriétés source des composants FLVPlayback et FLVPlaybackCaptioning (l'emplacement de votre fichier FLV et de votre fichier XML).
- Publiez.

L'exemple suivant importe le fichier XML dans le codeur.

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<FLVCoreCuePoints>

  <CuePoint>
    <Time>9136</Time>
    <Type>event</Type>
    <Name>fl.video.caption.2.0.index1</Name>
    <Parameters>
      <Parameter>
        <Name>text</Name>
        <Value><![CDATA[Captioning text for the first cue point]]></Value>
      </Parameter>
    </Parameters>
  </CuePoint>

  <CuePoint>
    <Time>19327</Time>
    <Type>event</Type>
    <Name>fl.video.caption.2.0.index2</Name>
    <Parameters>
      <Parameter>
        <Name>text</Name>
        <Value><![CDATA[Captioning text for the second cue point]]></
  Value>
      </Parameter>
    </Parameters>
  </CuePoint>

  <CuePoint>
    <Time>24247</Time>
    <Type>event</Type>
    <Name>fl.video.caption.2.0.index3</Name>
    <Parameters>
      <Parameter>
        <Name>text</Name>
        <Value><![CDATA[Captioning text for the third cue point]]></Value>
      </Parameter>
    </Parameters>
  </CuePoint>
```

```
  <CuePoint>
    <Time>36546</Time>
    <Type>event</Type>
    <Name>fl.video.caption.2.0.index4</Name>
    <Parameters>
      <Parameter>
        <Name>text</Name>
        <Value><![CDATA[Captioning text for the fourth cue point]]></
  Value>
      </Parameter>
    </Parameters>
</CuePoint>

</FLVCoreCuePoints>
```

Le composant FLVPlaybackCaptioning prend également en charge plusieurs pistes de langue avec un point de repère intégré. Pour plus d'informations, consultez la section « Prise en charge de plusieurs pistes de langue avec des points de repère intégrés », à la page 270.

Prise en charge de plusieurs pistes de langue avec des points de repère intégrés

La propriété `track` du composant FLVPlaybackCaptioning prend en charge plusieurs pistes de langue avec des points de repère intégrés, dans la mesure où le fichier XML Timed Text est en conformité avec les normes des points de repère du composant FLVPlaybackCaptioning. (Pour plus d'informations, consultez la section « Présentation des normes des points de repère du composant FLVPlaybackCaptioning », à la page 267.) Cependant, le composant FLVPlaybackCaptioning ne prend pas en charge plusieurs pistes de langue dans des fichiers XML distincts. Pour utiliser la propriété `track`, définissez-la sur une valeur qui n'est pas égale à 0. Par exemple, si vous définissez la propriété track sur 1 (`track == 1`), le composant FLVPlaybackCaptioning recherche les paramètres des points de repère. Si une correspondance est introuvable, la propriété de texte des paramètres des points de repère est utilisée. Pour plus d'informations, consultez la propriété `track` dans le *Guide de référence du langage et des composants ActionScript 3.0*.

Lecture de plusieurs fichiers FLV avec le sous-titrage

Vous pouvez ouvrir plusieurs lecteurs vidéo dans une seule occurrence du composant FLVPlayback pour lire plusieurs vidéos et basculer entre eux en fonction de la lecture. Vous pouvez également associer le sous-titrage à chaque lecteur vidéo dans le composant FLVPlayback. Pour plus d'informations sur la façon d'ouvrir plusieurs lecteurs vidéo, consultez la section « Utilisation de plusieurs lecteurs vidéo », à la page 228. Pour utiliser le sous-titrage dans plusieurs lecteurs vidéo, créez une occurrence du composant FLVPlaybackCaptioning pour chaque objet `VideoPlayer` et définissez l'index `videoPlayerIndex` du composant FLVPlaybackCaptioning sur l'index correspondant. L'index `VideoPlayer` prend la valeur 0 lorsqu'un seul objet `VideoPlayer` existe.

L'exemple de code suivant affecte un seul sous-titrage aux vidéos uniques. Pour exécuter cet exemple, vous devez remplacer les adresses fictives qu'il contient par des URL réelles.

```
captioner0.videoPlayerIndex = 0;
captioner0.captionURL = "http://www.[yourDomain].com/mytimedtext0.xml";
flvPlayback.play("http://www.[yourDomain].com/myvideo0.flv");
captioner1.videoPlayerIndex = 1;
captioner1.captionURL = "http://www.[yourDomain].com/mytimedtext1.xml";
flvPlayback.activeVideoIndex = 1;
flvPlayback.play ("http://www.[yourDomain].com/myvideo1.flv");
```

Personnalisation du composant FLVPlaybackCaptioning

Pour commencer à utiliser le composant FLVPlaybackCaptioning rapidement, vous pouvez choisir d'utiliser les valeurs par défaut du composant FLVPlaybackCaptioning qui placent le sous-titrage directement sur le composant FLVPlayback. Vous voudrez peut-être personnaliser le composant FLVPlaybackCaptioning pour éloigner le sous-titrage de la vidéo.

Le code suivant montre comment créer de manière dynamique un objet FLVPlayback à l'aide du bouton bascule de sous-titrage.

Pour créer de manière dynamique un objet FLVPlayback à l'aide du bouton bascule de sous-titrage :

1. Placez le composant FLVPlayback sur la scène à l'emplacement 0,0 et nommez l'occurrence player.
2. Placez le composant FLVPlaybackCaptioning sur la scène à l'emplacement 0,0 et nommez l'occurrence sous-titrage.

3. Placez le composant CaptionButton sur la scène.
4. Dans l'exemple de code suivant, définissez la `variable testVideoPath:String` sur un fichier FLV (à l'aide d'un chemin absolu ou relatif).

REMARQUE	L'exemple de code définit la variable `testVideoPath` sur l'exemple vidéo Flash, `caption_video.flv`. Modifiez cette variable et définissez-la sur le chemin du composant de sous-titrage vidéo auquel vous ajoutez un composant Button de sous-titrage.

5. Dans l'exemple de code suivant, définissez la variable `testCaptioningPath:String` sur un fichier XML Timed Text approprié (à l'aide d'un chemin absolu ou relatif).

REMARQUE	L'exemple de code définit la variable `testCaptioningPath` sur le fichier XML Timed Text, `caption_video.xml`. Modifiez cette variable et définissez-la sur le chemin du fichier XML Timed Text qui contient les légendes de votre vidéo.

6. Ajoutez les composants FLVPlayback et FLVPlaybackCaptioning dans votre bibliothèque.
7. Enregistrez le code suivant sous FLVPlaybackCaptioningExample.as dans le même répertoire que votre fichier FLA.
8. Définissez DocumentClass dans le fichier FLA sur FLVPlaybackCaptioningExample.

```
package
{
  import flash.display.Sprite;
  import flash.text.TextField;
  import fl.video.FLVPlayback;
  import fl.video.FLVPlaybackCaptioning;

  public class FLVPlaybackCaptioningExample extends Sprite {

    private var testVideoPath:String = "http://www.helpexamples.com/flash/
  video/caption_video.flv";
    private var testCaptioningPath:String = "http://www.helpexamples.com/
  flash/video/caption_video.xml";

    public function FLVPlaybackCaptioningExample() {
      player.source = testVideoPath;
      player.skin = "SkinOverAllNoCaption.swf";
      player.skinBackgroundColor = 0x666666;
      player.skinBackgroundAlpha = 0.5;
```

```
        captioning.flvPlayback = player;
        captioning.source = testCaptioningPath;
        captioning.autoLayout = false;
        captioning.addEventListener("captionChange",onCaptionChange);
      }
      private function onCaptionChange(e:*):void {
        var tf:* = e.target.captionTarget;
        var player:FLVPlayback = e.target.flvPlayback;

        // move the caption below the video
        tf.y = 210;
      }
    }
}
```

Pour plus d'informations sur tous les paramètres du composant FLVPlaybackCaptioning, consultez le *Guide de référence du langage et des composants ActionScript 3.0*.

ANNEXE A

Balises Timed Text

A

Le composant FLVPlaybackCaptioning prend en charge les balises Timed Text des fichiers XML de sous-titrage. Pour plus d'informations sur les balises Timed Text audio vidéo, consultez les informations disponibles sur le site http://www.w3.org. Le tableau suivant répertorie les balises prises en charge et non prises en charge.

Fonction	Balise/ Valeur	Utilisation/Description	Exemple
Balises ignorées	métadonnée	Ignorée / autorisée à tous les niveaux du document.	
	set	Ignorée / autorisée à tous les niveaux du document.	
	xml:lang	Ignorée	
	xml:space	Ignorée / Comportement remplacé par xml:space="default"	
	layout	Ignorée / y compris toutes les balises de la zone d'une section de balise de disposition.	
	balise br	Tous les attributs et le contenu sont ignorés.	

Fonction	Balise/ Valeur	Utilisation/Description	Exemple
Synchronisation des médias pour les légendes	attributs begin	Autorisés dans les balises p uniquement. Requis pour spécifier la durée de déploiement des médias des légendes.	‹p begin="3s"›
	attributs dur	Autorisés dans les balises p uniquement. Recommandés. S'ils ne sont pas inclus, la légende se termine avec le fichier FLV ou au démarrage d'une autre légende.	
	attributs end	Autorisés dans les balises p uniquement. Recommandés. S'ils ne sont pas inclus, la légende se termine avec le fichier FLV ou au démarrage d'une autre légende.	
Synchronisation de l'horloge pour les légendes	00:03:00.1	Format d'horloge complet	
	03:00.1	Format d'horloge partiel	
	10	Heures de décalage sans unité. Le décalage représente les secondes.	
	00:03:00:05 00:03:00:05.1 30f 30t	Pas de prise en charge. Les formats horaires qui incluent des images ou des graduations ne sont pas pris en charge.	
Balise body	body	Requise / Prise en charge pour une seule balise body.	‹body›‹div›...‹/div›‹/body›

Fonction	Balise/ Valeur	Utilisation/Description	Exemple
Balise de contenu	balise div	Valeur supérieure ou égale à zéro autorisée. La première balise est utilisée.	
	balise p	Valeur supérieure ou égale à zéro autorisée.	
	balise span	Conteneur logique pour une séquence d'unités de contenu textuel. Pas de prise en charge des plages imbriquées. Prise en charge des balises de style attribut.	
	balise br	Dénote un saut de ligne explicite.	
Application de styles aux balises (Toutes les balises de style sont utilisées dans la balise p)	style	Désigne un ou plusieurs éléments de style. Peut être utilisé en tant que balise et en tant qu'attribut. En tant que balise, un attribut d'ID est requis (le style peut être réutilisé dans le document). Prend en charge une ou plusieurs balises de style dans la balise de style.	
	tts:backgroun d Color	Permet de spécifier une propriété de style qui définit la couleur d'arrière-plan d'une zone. La valeur alpha est ignorée sauf si elle est définie sur zéro (alpha 0) afin de rendre l'arrière-plan transparent. Le format de couleur est #RRGGBBAA	

Fonction	Balise/ Valeur	Utilisation/Description	Exemple
	tts:color	Permet de spécifier une propriété de style qui définit la couleur d'avant-plan. La valeur alpha n'est prise en charge pour aucune couleur. La valeur `transparent` se traduit en noir.	<style id="3" style="2" tts:backgroundColor="white"/> "transparent" = #00000000 "black"=#000000FF "silver"=#C0C0C0FF "grey"=#808080FF "white"=#FFFFFFFF "maroon"=#800000FF "red"=#FF0000FF "purple"=#800080FF "fuchsia"("magenta")= #FF00FFFF "green"=#008000FF "lime"=#00FF00FF "olive"=#808000FF "yellow"=#FFFF00FF "navy"=#000080FF "blue"=#0000FFFF "teal"=#008080FF "aqua"("cyan")=#00FFFFFF

	tts:fontFamily	Permet de spécifier une propriété de style qui définit la famille de polices.	"default" = `_serif`
			"monospace" = `_typewriter`
			"sansSerif" = `_sans`
			"serif" = `_serif`
			"monospaceSansSerif" =
			`_typewriter`
			"monospaceSerif" =
			`_typewriter`
			"proportionalSansSerif" =
			`_sans`

Fonction	Balise/ Valeur	Utilisation/Description	Exemple
	tts:fontSize	Permet de spécifier une propriété de style qui définit la taille de police. Si deux valeurs sont fournies, seule la première valeur (verticale) est utilisée. Les valeurs et les unités de pourcentage sont ignorées. Prend en charge les tailles de pixel absolues (par exemple, 12) et de style relatives (par exemple +2).	
	tts: fontStyle	Permet de spécifier une propriété de style qui définit le style de police.	"normal" "italic" "inherit"* * Le comportement par défaut ; hérite du style de la balise englobante.
	tts: fontWeight	Permet de spécifier une propriété de style qui définit l'épaisseur de la police.	"normal" "bold" "inherit"* * Le comportement par défaut ; hérite du style de la balise englobante.
	tts: textAlign	Permet de spécifier une propriété de style qui définit la façon dont les zones sont alignées dans une zone qui contient un bloc.	"left" "right" "center" "start" (="left") "end" (="right") "inherit"* *Hérite du style de la balise englobante. Si aucune balise textAlign n'est définie, la valeur par défaut est "left".

Fonction	Balise/ Valeur	Utilisation/Description	Exemple
	tts: wrapOption	Permet de spécifier une propriété de style qui définit si le retour à la ligne automatique (saut) s'applique, ou non, dans le contexte de l'élément affecté. Ce paramètre affecte tous les paragraphes dans la légende.	"wrap" "noWrap" "inherit"* *Hérite du style de la balise englobante. Si aucune balise wrapOption n'est définie, la valeur par défaut est "wrap".
Attributs non pris en charge	tts: direction tts: display tts: displayAlign tts: dynamicFlow tts: extent tts: lineHeight tts: opacity tts: origin tts: overflow tts: padding tts: showBackground tts: textOutline tts: unicodeBidi tts: visibility tts: writingMode tts: zIndex		

Index

A

B

C

D

E

F

G

H

I

L

M

N

O

P

R

S

T

V