14 La communication Flash/HTML

Introduction

Interprété la plupart du temps depuis une fenêtre de navigateur, Flash offre de nombreuses fonctionnalités qui favorisent l'échange entre des contenus HTML et Flash. Vous pouvez gérer l'historique de navigation de Flash depuis le navigateur. Vous pouvez aussi appeler une configuration particulière d'un document Flash, en fonction de la page HTML qui y fait référence, en invoquant par exemple directement une image-clé du scénario. Naturellement, vous pouvez également lancer des pages HTML à partir de Flash. Enfin, pour offrir encore plus de fonctionnalités dans l'interface de Flash, et rappeler certaines options de la fenêtre de navigateur, le menu contextuel de Flash est personnalisable et vous pouvez y attacher les instructions de votre choix. Enfin, vous pouvez, en ajoutant un paramètre dans la page HTML, configurer une animation Flash pour rendre la scène transparente.

Dans ce chapitre, nous allons aborder l'ensemble de ces mécanismes pour vous permettre d'intégrer efficacement vos réalisations au sein d'un environnement web HTML. À l'issue de ce chapitre, vous serez capable d'élaborer facilement des liaisons entre différents types de contenus, réalisés en HTML et en Flash.

Menu contextuel du lecteur Flash

Un menu contextuel est accessible par le clic-droit de la souris sur la fenêtre de l'animation. Si votre souris ne dispose pas du clic-droit, le menu contextuel peut apparaître en effectuant simultanément un clic de souris et en appuyant sur la touche Contrôle du clavier.

La classe `contextMenu` de Flash permet de personnaliser une partie du menu contextuel. L'option d'affichage de la version du lecteur reste cependant affichée. Vous pouvez aussi choisir de conserver certaines des options disponibles par défaut, comme l'option Imprimer, par exemple.

Dans cette section, nous allons voir comment personnaliser un menu contextuel sur un symbole de type `MovieClip`. Nous allons permettre à l'utilisateur d'afficher une nouvelle image, en sélectionnant une option du menu contextuel à l'emplacement d'une photo, pour ce faire (voir Figure 14.1).

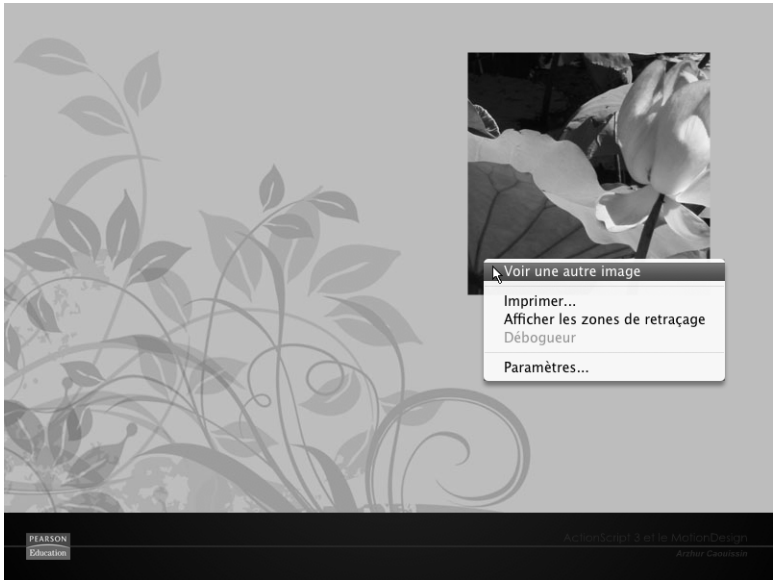


Figure 14.1
Aperçu de la scène principale.

Exemples > ch14_communicationFlashHTML_1.fla

Dans le document "ch14_communicationFlashHTML_1.fla", dans la scène principale, un symbole `vegetal_mc` décore la scène. Au-dessus, un MovieClip nommé `galerie_mc` affiche une suite d'images.

Dans le document SWF si l'utilisateur effectue un clic-droit sur l'image affichée, une option propose de voir une autre image. Si l'option est activée, une nouvelle image apparaît aussitôt.

Dans la fenêtre de scénario de la scène principale, le calque `Actions` affiche le code suivant :

```
//---------------------- actions
galerie_mc.stop();

//---------------------- Personnalisation du menu contextuel
// attacher le menu à un objet
var menuContextuel:ContextMenu= new ContextMenu();
galerie_mc.contextMenu=menuContextuel;

// purger le contenu par défaut
menuContextuel.hideBuiltInItems();
var entreesParDefaut:ContextMenuBuiltInItems=menuContextuel.builtInItems;
entreesParDefaut.print=true;

// ajouter les entrées du menu
var entree1:ContextMenuItem=new ContextMenuItem("Voir une autre image");
menuContextuel.customItems.push(entree1);
entree1.addEventListener(ContextMenuEvent.MENU_ITEM_SELECT, fonctionEntree1);

// attacher les fonctions aux entrées
```

```
function fonctionEntree1(evt:ContextMenuEvent) {
   if (galerie_mc.currentFrame==galerie_mc.totalFrames) {
      galerie_mc.gotoAndStop(1);
   } else {
      galerie_mc.nextFrame();
   }
}
```

Pour créer un menu contextuel, nous devons d'abord purger les entrées existantes. C'est alors que nous ajoutons de nouveaux éléments auxquels nous associerons des fonctions.

Dans notre exemple, la première instruction interrompt la tête de lecture sur la première image du symbole `galerie_mc` :

```
//---------------------- actions
galerie_mc.stop();
```

Nous attachons ensuite un nouvel objet `menuContextuel` à notre galerie. Nous pouvons, ainsi, spécifier un menu différent pour chaque conteneur, si nous le souhaitons :

```
// attacher le menu à un objet
var menuContextuel:ContextMenu= new ContextMenu();
galerie_mc.contextMenu=menuContextuel;
```

Puis, nous purgeons le menu actuel de ses entrées affichées par défaut.

```
// purger le contenu par défaut
menuContextuel.hideBuiltInItems();
```

Nous utilisons ici plusieurs méthodes. La première, `hideBuiltInItem()` supprime tous les éléments sauf les paramètres du lecteur Flash. Il est possible de conserver néanmoins certaines options. Pour cela, il suffit de les ajouter à la liste vidée. Pour ajouter une option par défaut, nous typons un nouvel objet ContextMenuBuiltInItem auquel nous spécifions l'option à intégrer :

```
var entreesParDefaut:ContextMenuBuiltInItems=menuContextuel.builtInItems;
entreesParDefaut.print=true;
```

Les options disponibles par défaut sont : `forwardAndBack` (affiche une image en arrière ou en avant), `loop` (boucle), `play` (lire), `print` (imprimer), `quality` (qualité), `rewind` (rembobiner), `save` (enregistrer) et `zoom` (zoom). Pour les activer, il suffit de les attacher au menu désigné et de les passer en true, comme dans notre exemple avec l'option d'impression print.

Une fois la structure de base définie, nous pouvons personnaliser le menu en y intégrant nos propres entrées :

```
// ajouter les entrées du menu
var entree1:ContextMenuItem=new ContextMenuItem("Voir une autre image");
menuContextuel.customItems.push(entree1);
```

Pour ajouter une entrée, nous créons un nouvel objet `ContextMenuItem()` et renseignons, en paramètre, une chaîne de caractères qui correspond au texte à afficher dans le menu.

Pour ajouter cet objet au menu, nous l'implémentons comme pour l'ajouter à un tableau, à l'aide de la méthode `push()`. Attention, l'objet doit être implémenté en tant qu'entrée – élément d'une liste. Nous devons pour cela préciser la propriété customItems, juste avant d'appliquer la méthode `push()`.

Vous pouvez ajouter autant d'entrées que voulu. Créez alors autant de nouveaux objets `ContenuMenuItem`.

Lorsque les entrées ont été ajoutées, il ne reste plus qu'à leur affecter une action. Nous utilisons, pour établir cette relation, un écouteur avec l'événement `MENU_ITEM_SELECT` qui désigne le moment où l'utilisateur sélectionne l'entrée du menu :

```
entree1.addEventListener(ContextMenuEvent.MENU_ITEM_SELECT, fonctionEntree1);
```

Puis, nous développons les actions appelées. Dans notre exemple, la fonction affiche une nouvelle image de la galerie, avec la méthode `nextFrame()`. Si cette image est la dernière de la séquence animée, alors, la tête de lecture revient à la première image. Elle boucle :

```
// attacher les fonctions aux entrées
function fonctionEntree1(evt:ContextMenuEvent) {
   if (galerie_mc.currentFrame==galerie_mc.totalFrames) {
      galerie_mc.gotoAndStop(1);
   } else {
      galerie_mc.nextFrame();
   }
}
```

À retenir

- Pour créer un menu contextuel personnalisé, nous utilisons la classe `contextMenu`.
- Les options du menu contextuel de Flash, affichées par défaut, peuvent être supprimées ou partiellement conservées. Seule la version du lecteur ne peut être supprimée du menu contextuel.

Navigation Flash vers le HTML

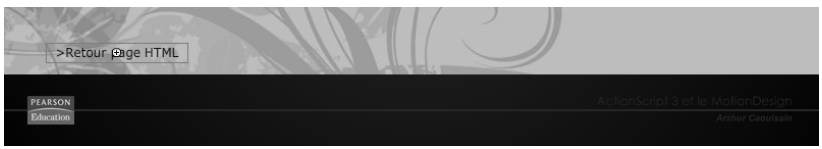
La base de la communication entre un contenu en Flash et une page HTML est la création de liens. Ces liens appellent une page HTML, un site distant ou activent l'ouverture d'un logiciel de messagerie, par exemple.

Dans cette section, nous présentons la syntaxe de création de lien dans Flash pour appeler une page HTML et d'autres types de références hypertextes (sites, messagerie, PDF).

Exemples > ch14_communicationFlashHTML_2.fla

Sur la scène principale du document "ch14_communicationFlashHTML_2.fla" figure un bouton affichant un texte simple : ">Retour à la page HTML" (voir Figure 14.3).

Figure 14.3
Aperçu de la scène principale.



Dans le fichier SWF, en cliquant sur le bouton, une page HTML est bien appelée ; page dont nous détaillons d'ailleurs le contenu dans la section suivante.

Dans la fenêtre de scénario de la scène principale, sur le calque `Actions`, nous pouvons lire le code suivant :

```
// lien HTML
lienHTML_btn.addEventListener(MouseEvent.CLICK,lienHTML);
function lienHTML (evt:MouseEvent) {
   navigateToURL(new URLRequest("ch14_communicationFlashHTML_3b.html"));
}
```

En ActionScript 3, pour appeler une page HTML, nous utilisons la méthode `navigateToURL` et spécifions, en paramètre, un objet `URLRequest` qui fait référence à une adresse définie sous la forme d'une chaîne de caractères.

L'adresse utilisée peut appeler une page web locale, au sein du même site que la page qui contient le document Flash. Les exemples suivants invoquent différents éléments :

- une page de contact au format PHP :

```
navigateToURL(new URLRequest("contact.php"));
```

- une adresse de site web distant :

```
navigateToURL(new URLRequest("http://www.pearson.fr/"));
```

- l'ouverture d'une fenêtre de messagerie :

```
navigateToURL(new URLRequest("mailto:contact@pearson.fr?subject=demande de
➥ renseignement"));
```

- l'ouverture d'un PDF :

```
navigateToURL(new URLRequest("nomDuDocument.pdf"));
```

- Pour appeler un site dans une nouvelle fenêtre, nous ajoutons la cible :

```
navigateToURL(new URLRequest("http://www.pearson.fr", "_blank"));
```

Les cibles d'affichage. *Pour chaque lien hypertexte créé, une cible permet de définir la manière dont le navigateur affiche la page appelée. Lorsque les sites web usaient encore, dans le passé, de structures composées de jeux de cadres (framesets), nous pouvions utiliser les valeurs* `_self`, `_blank`, `_parent` *ou* `_top` *afin de déterminer si la page appelée devait se substituer ou non aux jeux de cadres. Les liens demeurant désormais indépendants de ce type de structure, nous n'utilisons plus que* `_self` *ou* `_blank`.

- `_self` *: la cible* `_self` *équivaut à ne rien indiquer et signifie que la page appelée doit s'afficher dans le même cadre que celui où l'on a cliqué.*
- `_blank` *: la cible* `_blank` *appelle l'affichage dans une nouvelle fenêtre de navigateur. Le terme anglais Blank désigne un espace vierge, représenté ici par l'ouverture d'une nouvelle fenêtre. Ne confondez pas ce mécanisme avec le comportement JavaScript permettant d'enclencher l'ouverture d'une fenêtre pop-up, que les usages ont également proscris. L'affichage d'une page avec* `_blank` *établie en outre une rupture sémantique avec la page initiale. Cette technique*

d'affichage ne doit donc pas être abusée, surtout dans le contexte de création de sites qui doivent répondre aux règles élémentaires d'accessibilité (voir aussi les sections en fin d'ouvrage sur l'accessibilité).

La plupart des contenus appelés s'affichent dans la même fenêtre que le document Flash, à la place du Flash lui-même, à l'exception des scripts JavaScript ou PHP (ou autres) éventuellement invoqués. Les pages pour lesquelles une cible de type `_blank` est également associée s'affichent dans une nouvelle fenêtre de navigateur et préservent l'affichage de votre document dans la fenêtre initiale.

Attention, cette option n'est recommandée que pour les liens faisant appel à des éléments situés à une autre URL, car elle entraîne une rupture sémantique du flux qui peut perturber certains utilisateurs.

Lorsque vous désignez une adresse d'un site ou d'une page distante, veillez à toujours commencer l'adresse par `http://`.

Si les liens créés appellent des instructions localisées dans la page HTML qui contient le Flash, ou lorsque des actions ne peuvent être lancées que par un navigateur, pour tester le bon fonctionnement des liens hypertextes, il est recommandé de projeter la page dans un navigateur et non de vérifier les animations seulement depuis Flash en publiant le fichier SWF. Le lecteur Flash, activé à la publication du document, n'est pas un navigateur. Ainsi, il ne peut exécuter tous les types de liens (notamment pour l'ouverture de la fenêtre de messagerie ou lancer les scripts exécutés localement ou depuis un serveur distant).

Créer un lien HTML sans ActionScript dans Flash. *Si vous cherchez à réaliser un lien dans le flux d'un texte, sans style spécifique, sans avoir à créer de symbole Bouton ou MovieClip et sans avoir à coder en ActionScript, vous pouvez utiliser les propriétés de texte. Sélectionnez d'abord la portion de texte à rendre cliquable. Et depuis l'inspecteur de propriétés, dans le champ Lien (catégorie Option), saisissez directement le lien ou le type de référence hypertexte à appeler. Puis, appuyez sur Entrée pour confirmer la saisie.*

Il est également possible de générer un lien HTML dynamiquement si le champ de texte dynamique affiche un rendu HTML. En ActionScript, modifiez la valeur du champ de texte en spécifiant, dans la chaine de caractères du texte à afficher, la balise HTML qui permet de créer un lien. Le texte dynamique de nom d'occurrence monTexte_txt utilise, dans l'exemple suivant, cette propriété :

```
monTexte_txt.htmlText = '<a href="http://www.pearson.fr">Lien</a>'
```

À retenir

- *Vous pouvez appeler tout type de contenu extérieur à Flash et l'afficher dans la fenêtre de navigateur à la place du Flash en utilisant la méthode* `navigateToURL` *à laquelle vous associez un nouvel objet* `URLRequest` *qui affiche la référence hypertexte mentionnée.*
- *Il est préférable de tester les liens dans le contexte du navigateur, voire en ligne, lorsque ces liens se réfèrent à des comportements dynamiques exécutés côté serveur ou par le navigateur.*

Navigation HTML vers Flash

Si vous réalisez un site presqu'entièrement en Flash et que, ponctuellement, vous appelez une page HTML, vous aimeriez certainement pouvoir faire un lien de retour depuis cette page, vers un endroit précis de votre document Flash, et pas forcément vers la première image du scénario de votre document Flash. Cela est possible en adoptant une syntaxe particulière pour les liens placés dans la page HTML et une structure adaptée pour la construction du document Flash.

Dans ce contexte, le lien HTML fait référence, en fait, à une image clé du scénario du document Flash en ajoutant, au nom de la page qui contient le Flash, un dièse (#), suivi du nom d'étiquette de l'image cible :

```
nomDeLaPage.html#nomDeLetiquette
```

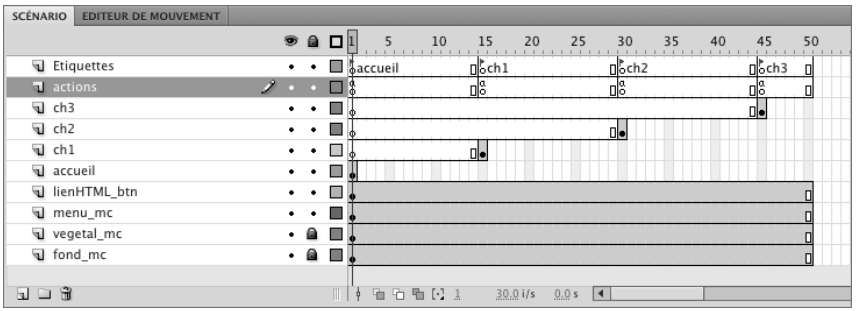
Dans cette section, nous présentons un document SWF structuré en vue d'être appelé par un lien HTML. Le lien HTML contient, de son côté, des liens qui activent distinctement chacune des rubriques du Flash.

Cette technique impose une conception de site dans le scénario, pour lequel chaque type de contenu est réparti, à un endroit distinct des autres.

Exemples > ch14_communicationFlashHTML_3.fla
Exemples > ch14_communicationFlashHTML_3.html
Exemples > ch14_communicationFlashHTML_3b.html

Dans notre exemple, nous disposons d'un document Flash dont l'interface est distribuée sur le scénario en mode séquentiel (voir Figure 14.4). Chaque emplacement d'étiquette correspond à une nouvelle rubrique.

Figure 14.4
Aperçu du scénario de la scène principale.



Ce document est placé dans la page HTML "ch4_communicationFlashHTML_3.html". En affichant ce fichier dans le navigateur, nous constatons dans un premier temps que la page affiche bien l'animation (voir Figure 14.5).

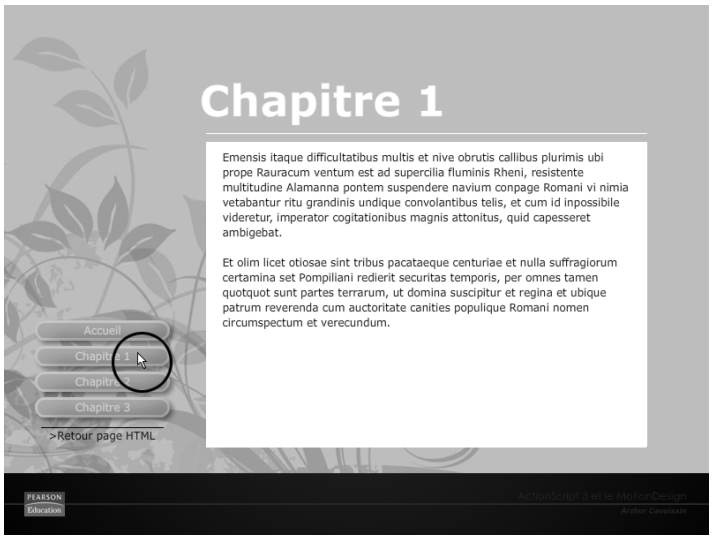
Attention

Pour cet exemple, ne publiez pas le document SWF en faisant F12 ou Fichier > Publier. Vous risqueriez d'écraser la page HTML déjà mise en forme et qui porte le même nom que le document Flash. Pour publier le document SWF, faites simplement Cmd+Entrée (Mac) ou Ctrl+Entrée (Windows). Pour projeter la page HTML directement dans le navigateur, lancez-la depuis le système d'exploitation.

Figure 14.5
Aperçu dans le navigateur.

En cliquant sur les liens du menu situés à gauche, vous accédez aux différentes rubriques mises en place dans le scénario de Flash (voir Figures 14.6 à 14.8).

Figure 14.6



En cliquant sur le lien HTML situé en bas du document Flash, vous accédez à une page HTML (vue à la section précédente). Cette page HTML affiche une série de liens codés en HTML.

Figure 14.7

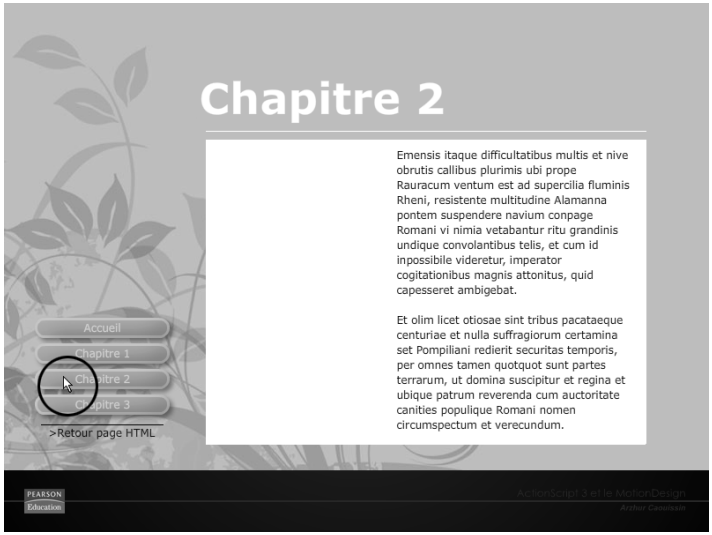
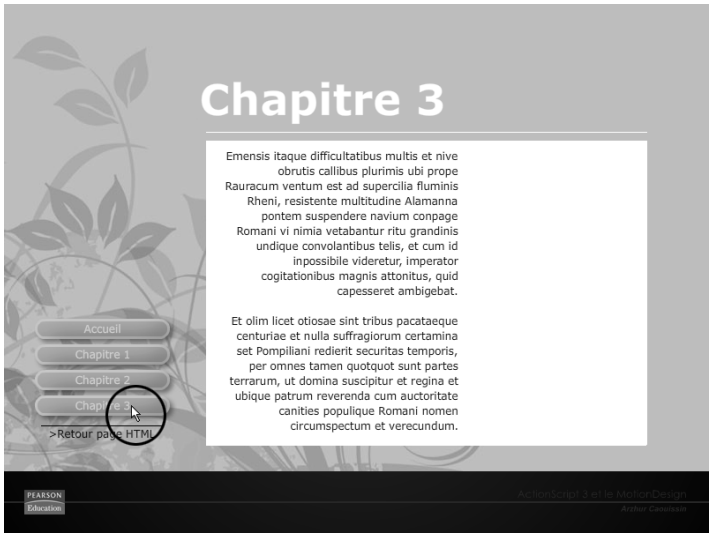


Figure 14.8



En cliquant sur chaque lien, vous atteignez directement l'image du scénario qui correspond à la rubrique activée (voir Figure 14.9).

Dans le document Flash, sur la scène principale, se trouve un symbole nommé `menu_mc`. Dans ce symbole, nous pouvons distinguer quatre boutons respectivement nommés `accueil`, `ch1`, `ch2` et `ch3` (voir Figure 14.10).

Dans le scénario de la scène principale, chaque rubrique est placée précisément sous une étiquette. L'accueil est sous l'étiquette accueil. Le Chapitre 1 est sous l'étiquette ch1, et ainsi de suite pour les Chapitres 2 et 3 (voir Figure 14.4). Vous relevez que les noms d'étiquettes sont identiques aux noms d'occurrence empruntés pour les boutons du menu.

Figure 14.9
Aperçu de la page HTML appelée.

Figure 14.10
Boutons de navigation dans le Flash.

Cela n'a pas d'incidence directe sur la navigation depuis la page HTML, mais simplifie le codage au sein du document Flash pour une navigation interne.

Info

Placer un nom d'étiquette sur une image du scénario. *Pour placer un nom d'étiquette sur une image du scénario, vous devez d'abord ajouter une image clé (de préférence vide), sur un calque dédié à cela dans le scénario, à l'endroit ciblé. Puis, cliquez sur cette image du scénario pour en afficher les propriétés. Dans l'inspecteur de propriétés, attribuez un identifiant sans caractères spéciaux, sans espace, et ne commençant pas par un chiffre. Appuyez sur la touche Entrée pour confirmer la saisie.*

Dans le scénario de la scène principale, sur le calque `Actions`, à l'image 1, le code affiché est le suivant :

```
stop();

menu_mc.addEventListener(MouseEvent.CLICK,afficherRubrique);
function afficherRubrique (evt:MouseEvent) {
   gotoAndStop(evt.target.name);
}

lienHTML_btn.addEventListener(MouseEvent.CLICK,lienHTML);
function lienHTML (evt:MouseEvent) {
   navigateToURL(new URLRequest("ch14_communicationFlashHTML_2b.html"));
}
```

Tout d'abord, une fonction permet de naviguer à l'intérieur du document Flash en cliquant sur les liens contenus dans le symbole `menu_mc`. Cette fonction utilise la propriété target qui établit une relation directe entre le nom de l'occurrence activée et le nom de l'étiquette à atteindre. Si nous cliquons sur le bouton ch1, son nom est donc directement passé en paramètre de la méthode gotoAndStop() et nous conduit à cette image du scénario. Cette technique nous épargne l'ajout de structures conditionnelles, pour simplement organiser une navigation dans le scénario.

En dessous, figure le lien qui pointe vers la page HTML – nous l'avons utilisé dans la section précédente.

Ce document Flash est donc autonome. Aucune relation ne l'attache à la page HTML qui s'y réfère. Mais, ce qui permet aux liens de la page HTML de cibler la bonne rubrique, c'est qu'il est structuré avec des étiquettes dont les noms sont repris dans les liens de la page HTML.

Si nous revenons dans la page HTML qui porte le nom "ch14_communicationFlashHTML_3b.html", dans Dreamweaver par exemple ou directement dans le code source du document, en sélectionnant chaque lien individuellement, vous remarquez la valeur inscrite dans l'attribut href .

Pour appeler la rubrique accueil, le lien HTML fait référence d'abord à la page qui contient le Flash (ch14_communicationFlashHTML_3.html), suivi directement d'un dièse (#) et de l'étiquette (accueil). Ce qui donne :

```
<a href="ch14_communicationFlashHTML_3.html#accueil">texte du lien</a>
```

Le principe est décliné pour chaque lien de la page HTML.

Mais, dans ce mécanisme relativement simple, nous devons souligner quelques contraintes.

Lorsque le lien de la page HTML atteint directement une image du scénario, les écouteurs, placés sur la première image du scénario, ne sont plus actifs. Vous devez recopier chaque écouteur placé sur la première image au niveau de chaque étiquette. Les fonctions, elles, restent actives. Il n'est pas nécessaire de les dupliquer. D'autant qu'il ne peut y avoir deux fonctions de même nom. Dans le calque Actions, à l'image ch1, ch2 et ch3, nous pouvons lire le code suivant :

```
stop();
menu_mc.addEventListener(MouseEvent.CLICK,afficherRubrique);
```

Nous avons un stop qui empêche la tête de lecture de poursuivre l'animation. Nous pouvons également lire une nouvelle instance de l'écouteur déjà placé à l'image 1.

Tester les liens HTML sur un serveur distant. *Avec certaines configurations, il est possible que les liens HTML qui appellent une étiquette Flash ne soient pas directement actifs sur votre poste de travail. Placez dans ce cas vos fichiers sur un serveur distant, et exécutez à nouveau les pages. Les liens sont bien actifs.*

À retenir

- *Il est possible de cibler directement une image du scénario dans Flash, à partir d'un lien codé en HTML. Pour cela, nous invoquons un nom d'étiquette (`#nomEtiquette`) directement depuis le lien HTML.*

Historique de l'animation Flash dans le navigateur

Le bouton Suivant et Retour du navigateur peuvent contrôler l'affichage des contenus dans un document Flash. Pour cela, le site **www.asual.com** met à disposition les classes SWFAddress et SWFObject, gratuitement et en téléchargement libre, à l'adresse suivante : **http://www.asual.com/swfaddress/**.

Installer une classe importée. *Pour voir comment installer la classe nativement ou ponctuellement, reportez-vous au début du Chapitre La 3D et PaperVision. Nous nous contentons, pour cet exemple, de citer le site référent. Une documentation détaillée et un forum sont disponibles sur leur site Internet.*

Ces classes fonctionnent sur le même principe que le mécanisme des liens d'une page HTML vers un contenu Flash, évoqué dans la section précédente. À la différence que les liens ne sont pas activés par la page HTML, mais par un script JavaScript interprété par le navigateur.

Pour les utiliser, vous devez donc créer un document Flash qui sera contenu dans une page HTML, laquelle fait référence à plusieurs fichiers JavaScript. Le code proposé par le site **www.asual.com** pour intégrer le contenu Flash est le suivant :

```
<script type="text/JavaScript" src="swfobject/swfobject.js"></script>
<script type="text/JavaScript" src="swfaddress/swfaddress.js"></script>
<script type="text/JavaScript">swfobject.embedSWF('website.swf', 'website',
'100%', '100%', '9','expressInstall.swf', {}, {menu: 'false'}, {id: 'website'});
</script>
```

Dans Flash, vous devez ensuite placer les commandes dans le scénario. Elles seront automatiquement interprétées par le même JavaScript utilisé dans la page HTML. Dans le document Flash, sur la première image du scénario de la scène principale, le site **www.asual.com** propose les actions suivantes :

```
// SWFAddress actions
function btnClick(e:MouseEvent) {
   SWFAddress.setValue(e.target.deepLink);
}
function btnRollOver(e:MouseEvent) {
   SWFAddress.setStatus(e.target.deepLink);
}
function btnRollOut(e:MouseEvent) {
   SWFAddress.resetStatus();
}

// SWFAddress handling
function handleSWFAddress(e:SWFAddressEvent) {
   try {
      if (currentFrame == 2 && e.value == '/') {
         play();
      } else {
         gotoAndStop('$' + e.value);
      }
   } catch(err) {
      gotoAndStop('$/error/');
   }
```

```
        var title:String = 'SWFAddress Website';
        for (var i = 0; i < e.pathNames.length; i++) {
            title += ' / ' + e.pathNames[i].substr(0,1).toUpperCase() + e.path-
            ➥ Names[i].substr(1);
        }
        SWFAddress.setTitle(title);
    }
    SWFAddress.addEventListener(SWFAddressEvent.CHANGE, handleSWFAddress);
    stop();
```

Nous distinguons les différentes fonctions faisant référence à des occurrences de MovieClip distribuées dans le scénario à chaque niveau d'étiquette. Ces fonctions sont donc valides pour toute la durée du scénario. La variable `deepLink` est simplement mise à jour en fonction de l'objet activé (`target`). C'est donc l'étiquette invoquée qui est différente en fonction de l'objet activé.

Dans le scénario, à chaque nouvelle rubrique, utilisez un nom d'étiquette qui reprend cette syntaxe:

```
$/portfolio/
```

Ce nom commence avec le caractère dollar. Puis, un nom, sans caractères spéciaux ni accent est placé entre deux slashs.

À l'intérieur de chaque MovieClip – qui est utilisé comme bouton et dont la navigation doit être contrôlée par JavaScript –, **www.asual.com** propose d'utiliser le code suivant :

```
this.deepLink = '/about/';
this.buttonMode = true;
this.addEventListener(MouseEvent.CLICK, (parent as MovieClip).btnClick);
this.addEventListener(MouseEvent.ROLL_OVER, (parent as MovieClip).btnRollOver);
this.addEventListener(MouseEvent.ROLL_OUT, (parent as MovieClip).btnRollOut);
```

Dans ce code, nous relevons principalement l'affectation d'une nouvelle valeur pour la variable `deepLink`, qui véhicule la cible à atteindre par le bouton cliqué. Cette valeur, enregistrée par JavaScript, va permettre au navigateur de contrôler toute la navigation à l'intérieur du document Flash.

Enfin, dans le dossier de votre document Flash, vous devez placer les éléments suivants :

- la classe intitulée `SWFAddress.as` ;
- la classe intitulée `SWFAddressEvent.as` ;
- le dossier de scripts `swfaddress/` ;
- et le dossier de scripts `swfobject/`.

Publiez le document SWF. En activant les liens du menu, le document Flash défile normalement. Mais les commandes de l'historique du navigateur enregistrent votre navigation. En activant les boutons Suivant et Retour, vous pouvez revenir dans le document Flash comme si vous naviguiez de page HTML en page HTML.

Pour utiliser et tester un fichier Flash dans un environnement déjà mis en forme, vous pouvez télécharger le dossier compressé de présentation déjà structuré, et disponible directement en ligne. Pour des raisons liées à la protection des droits d'auteurs, nous ne

pouvons pas inclure ces ressources dans notre ouvrage. Vous pouvez en revanche les télécharger librement à l'adresse suivante : **http://sourceforge.net/projects/swfaddress/files/swfaddress/SWFAddress%202.4/swfaddress-2.4.zip/download**. Dans le répertoire téléchargé, différentes configurations sont proposées. Pour un développement en ActionScript 3, utilisez les éléments du répertoire CS3 mis à disposition dans le dossier Samples.

À retenir

- Il est possible d'activer les boutons Retour et Suivant du navigateur pour créer un historique de la navigation dans Flash. Pour cela, nous utilisons la classe `SWFAddress` et `SWFObject` disponible gratuitement sur le site **www.asual.com**.

Importer du HTML dans un SWF

Plus qu'une simple fonctionnalité, l'intégration de contenus formatés avec des balises HTML est une solution presque incontournable quand il s'agit d'intégrer des contenus éditoriaux. Sans avoir à recourir à des systèmes de gestion de contenu, il demeure assez facile d'importer et de mettre en forme des textes et des images avec des feuilles de style. Pour cela, nous utilisons un champ de texte dynamique et, éventuellement, un composant `ScrollBar` (ou un ascenseur fait maison, comme vu au Chapitre 2).

Dans cette section, nous utilisons une interface en Flash qui charge un fichier texte et une feuille de style au format CSS (voir Figure 14.11).

Figure 14.11
Aperçu du document publié.

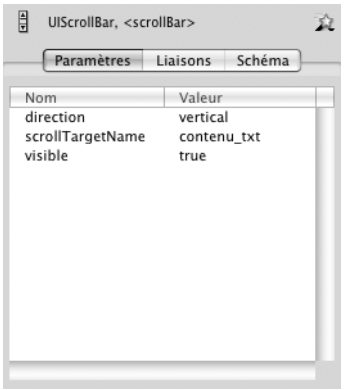
Exemples > ch14_communicationFlashHTML_4.fla
Exemples > html > styles.css
Exemples > html > texte.txt

Dans le document "ch14_communicationFlashHTML_4.fla", à droite de la scène principale, est placé un aplat de couleur (forme graphique). Au-dessus, un champ de texte dynamique porte le nom d'occurrence `contenu_txt`. À sa droite, un composant `ScrollBar` porte le nom d'occurrence `ScrollBar` (voir Figure 14.12).

Figure 14.12
Aperçu du document.

Dans l'inspecteur de composants (Fenêtre > Inspecteur de composants), la propriété `scrolltargetName`, qui définit le champ de texte à contrôler, fait référence au nom de notre champ de texte dynamique (voir Figure 14.13).

Figure 14.13
Fenêtre Inspecteur de composants.



En dehors du document Flash, dans le dossier html/, placé dans le répertoire des exemples du livre, nous disposons d'un fichier nommé `texte.txt` qui affiche en partie le code suivant :

```
content=<p class="titre">A la une</p><p><i>de Arzhur CAOUISSIN</i><br><li><b>
Source :</b> Les marais salant de Guérande</li><li><b>Site Web :</b>  <a href=
"http://www.ot-guerande.fr/" target="_blank"><span class="lien">http://www.ot-
guerande.fr</span></a></li></p><p><img src="images/galerie/vignettes/photo2-
vignette.jpg" vspace="0" hspace="5">Lorem ipsum dolor sit amet, consectetuer
adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore
magna aliquam erat volutpat. Ut wisi enim ad minim veniam, …/… </p>
```

Une variable du nom de content ouvre sur une série de balises HTML qui organisent le flux d'un texte et contiennent, parfois, des références à des classes CSS (`.titre`, `.lien`) ou à des balises d'apparence modifiée, dans la feuille de styles (`p`). Les images importées sont, elles, déjà placées localement dans le répertoire images du dossier des exemples du livre.

Le dossier `html/` des exemples du livre accueille cette feuille de style qui adopte la forme suivante :

```
p {
font-family: Arial,Helvetica,sans-serif;
font-size: 12px;
color: #001622;
}
.titre {
font-family: Arial,Helvetica,sans-serif;
font-size: 24px;
color: #001622;
}
.lien {
font-family: Arial,Helvetica,sans-serif;
font-size: 12px;
color: #001622;
text-decoration: underline;
}
```

Dans le scénario du document Flash, le composant `Scrollbar` et le texte dynamique sont répartis entre les formes d'habillage graphique et le calque `Actions` (voir Figure 14.14).

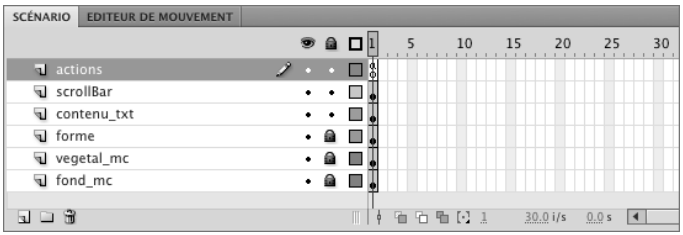


Figure 14.14
Aperçu du scénario de la scène principale.

Le calque `Actions` affiche le code suivant :

```
// Charge les CSS
var chargeurCSS:URLLoader = new URLLoader();
chargeurCSS.load(new URLRequest("html/styles.css"));
chargeurCSS.addEventListener(Event.COMPLETE, CSSchargees);
```

```
// Applique les CSS
function CSSchargees(evt:Event) {
   var stylesCSS:StyleSheet = new StyleSheet();
   stylesCSS.parseCSS(evt.currentTarget.data);
   contenu_txt.styleSheet=stylesCSS;
   chargerLeTexte();
}

// Charge le texte
function chargerLeTexte() {
   var chargeurTexte:URLLoader = new URLLoader();
   chargeurTexte.load(new URLRequest("html/texte.txt"));
   chargeurTexte.addEventListener(Event.COMPLETE, placerLeTexte);
}

// Place le texte
function placerLeTexte(evt:Event) {
   var cheminVariable:URLVariables=new URLVariables(evt.currentTarget.data);
   contenu_txt.condenseWhite=true;
   contenu_txt.htmlText=String(cheminVariable.content);
}
```

Dans cette application, nous activons le chargement des contenus textes et CSS en quatre temps. D'abord, nous chargeons les styles. Puis, nous les appliquons au conteneur du futur texte. Une fois les styles appliqués, nous pouvons importer le texte dans ce champ dynamique.

La première étape charge la feuille de style :

```
// Charge les CSS
var chargeurCSS:URLLoader = new URLLoader();
chargeurCSS.load(new URLRequest("html/styles.css"));
chargeurCSS.addEventListener(Event.COMPLETE, CSSchargees);
```

Il s'agit d'un chargeur tout simple qui exécute une fonction une fois le chargement complété. La fonction appelée applique ensuite les styles, sur le champ de texte dynamique :

```
// Applique les CSS
function CSSchargees(evt:Event) {
   var stylesCSS:StyleSheet = new StyleSheet();
   stylesCSS.parseCSS(evt.currentTarget.data);
   contenu_txt.styleSheet=stylesCSS;
   chargerLeTexte();
}
```

Dans cette fonction, nous définissons, dans un premier temps, un objet feuille de style qui va stocker les styles avant de les appliquer. Nous compilons ensuite les balises CSS avec un parseur CSS interne à Flash, pour que le code soit correctement interprété par Flash. Les données chargées (`evt.currentTarget.data`) sont passées en paramètre du parseur CSS. Le parseur affecte les styles à l'objet `StyleSheet` que nous venons d'instancier.

Puis, nous appliquons la feuille de style sur le champ de texte dynamique, à l'aide de la propriété `styleSheet`.

La fonction qui charge le texte est ensuite appelée :

```
// Charge le texte
function chargerLeTexte() {
   var chargeurTexte:URLLoader = new URLLoader();
   chargeurTexte.load(new URLRequest("html/texte.txt"));
   chargeurTexte.addEventListener(Event.COMPLETE, placerLeTexte);
}
```

Une fois le texte complété, la fonction appelée place le texte formaté dans le champ de texte dynamique :

```
// Place le texte
function placerLeTexte(evt:Event) {
   var cheminVariable:URLVariables=new URLVariables(evt.currentTarget.data);
   contenu_txt.condenseWhite=true;
   contenu_txt.htmlText=String(cheminVariable.content);
}
```

Dans cette dernière fonction, nous créons un objet de traitement de variable `URLVariables`, afin de contenir l'ensemble du code qui se situe, à l'intérieur du fichier `texte.txt`, après le signe égal. Cette variable appelle en paramètre le texte que nous venons de charger pour s'en approprier naturellement le contenu.

La propriété `condenseWhite` permet de supprimer les espaces blancs redondants, comme le fait tout interpréteur HTML.

La dernière instruction ajoute le texte chargé, dans le champ dynamique situé sur la scène, à l'aide de la propriété `htmlText`. Mais il le convertit au passage sous la forme d'une chaîne de caractères (transtypage avec la méthode `String()`), afin d'éviter que les balises de formatage soient également affichées dans le flux du texte.

À retenir

- Le HTML peut être importé dans un document Flash et associé à une feuille de style CSS. Pour cela, nous utilisons la méthode `parseCSS()`.
- Il n'est pas nécessaire de stocker les balises dans un flux XML. Un simple fichier texte peut suffire à contenir les éléments importés. Nous utilisons alors un objet `URLVariable` pour traiter les données.

Gérer le Flash transparent

Un document Flash affiche, par défaut, une couleur d'arrière-plan. Vous pouvez indiquer au document HTML de ne pas interpréter cette donnée en vue de rendre transparent l'arrière-plan du document Flash.

L'objectif de cette modification est de permettre d'entrevoir le contenu de la page HTML généralement placée sous l'animation Flash. Cette technique est employée aussi bien pour agrémenter des contenus qui doivent se coordonner avec une image d'arrière-plan placée dans une page HTML beaucoup plus large, ou avec un motif répété à l'arrière-plan de la page HTML, par exemple. Mais elle est aussi employée dans des sites éditoriaux pour placer des annonces commerciales au premier plan sur des articles de fond, ou encore

permettre à des menus HTML / CSS de passer au devant (`index-z`) d'animations SWF. Sans la transparence de celui-ci cela ne serait effectivement pas possible.

Pour appliquer un fond transparent à un document SWF, dans la page HTML qui le contient, ajoutez un paramètre `wmode` et affectez la valeur transparent :

```
<param name="wmode" value="transparent">
```

Assurez-vous que cette option est également renseignée dans les balises `embed` et dans les paramètres du JavaScript qui contrôle éventuellement l'affichage du document Flash (voir Chapitre 15).

Attention, cette option peut perturber l'affichage des caractères dans les champs de texte dynamiques, surtout s'ils sont isolés par un masque, même en ajoutant les caractères au champ de texte dynamique, particulièrement à partir de Firefox 2 (voir aussi la section Gérer la typographie au Chapitre 15).

À retenir

- *Afin de convertir l'arrière-plan d'un document SWF en fond transparent, nous définissons, dans la page HTML qui contient l'animation Flash, le paramètre `wmode` à la valeur transparent.*

Synthèse

Dans ce chapitre, vous avez appris à établir des liaisons entre un contenu réalisé en Flash et la page HTML qui le contient, et donc, à personnaliser l'affichage d'un document Flash selon la page HTML qui s'y réfère. Vous avez également appris à organiser un historique de navigation, à personnaliser le menu contextuel de Flash et à importer du HTML dans une zone Flash. Vous savez désormais facilement intégrer vos développements dans un site web et favoriser les échanges entre contenus HTML et Flash. Des techniques complémentaires propres à l'implémentation du Flash dans le navigateur sont abordées dans le chapitre suivant.