

FORMATION A ACTIONSCRIPT™ FLASH® LITE™ 1.x

© 2007 Adobe Systems Incorporated. Tous droits réservés.

Formation à ActionScript Flash® Lite™ 1.x

S'il est distribué avec un logiciel comprenant un contrat de licence, ce manuel, ainsi que le logiciel qui y est décrit, sont cédés sous licence et ne peuvent être utilisés ou copiés que conformément à la présente licence. Sous réserve des clauses de cette licence, aucune partie de ce guide ne peut être reproduite, enregistrée dans un système de recherche automatique ou transmise sous une forme ou par un moyen quelconque, électronique, mécanique ou autre, sans l'autorisation écrite préalable d'Adobe Systems Incorporated. Veuillez noter que le contenu de ce manuel est protégé par des droits d'auteur, même s'il n'est pas distribué avec un logiciel comprenant un contrat de licence.

Le contenu de ce guide est fourni à titre purement informatif, est susceptible d'être modifié sans avertissement et ne doit pas être considéré comme un engagement de la part d'Adobe Systems Incorporated. Adobe Systems Incorporated décline toute responsabilité quant aux éventuelles erreurs ou inexactitudes pouvant apparaître dans le contenu informatif de ce guide.

Veuillez noter que les dessins ou images que vous pouvez avoir l'intention d'insérer dans vos projets sont susceptibles d'être protégés par la loi sur les droits de reproduction. L'utilisation non autorisée de ces éléments peut constituer une violation des droits des détenteurs de leurs copyrights. Il est impératif de demander toute autorisation nécessaire au détenteur des droits de reproduction.

Les éventuels noms de sociétés dans les exemples de modèles n'existent qu'à fins de démonstration et ne font aucunement référence à des organisations existantes.

Adobe, le logo Adobe, Flash Lite, et Flash sont des marques commerciales ou des marques déposées d'Adobe Systems Incorporated aux Etats-Unis et/ou dans d'autres pays.

Informations au sujet des parties tierces

Ce guide contient des liens vers des sites Web qui ne sont pas sous le contrôle d'Adobe Systems Incorporated, qui n'est aucunement responsable de leur contenu. Si vous accédez à un site Web tiers mentionné dans ce guide, vous le faites à vos propres risques. Adobe Systems Incorporated fournit ces liens à des fins pratiques et l'inclusion de ces liens n'implique pas qu'Adobe Systems Incorporated parraine ou accepte la moindre responsabilité pour le contenu de ces sites Web tiers.



Technologie de compression et décompression vidéo Sorenson™ Spark™ utilisée sous licence de Sorenson Media, Inc.

Fraunhofer-IIS/Thomson Multimedia : technologie MPEG Layer-3 de compression audio utilisée sous licence de Fraunhofer IIS et Thomson Multimedia (<http://www.iis.fhg.de/amm/>).

Independent JPEG Group : ce logiciel repose en partie sur les travaux de l'« Independent JPEG Group ».

Nellymoser, Inc. : Technologie de compression et décompression de la parole utilisée sous licence de Nellymoser, Inc. (<http://www.nelly-moser.com>).

Navigateur Opera ® Copyright © 1995-2002 Opera Software ASA et ses fournisseurs. Tous droits réservés.

La vidéo de Macromedia Flash 8 est optimisée par la technologie vidéo On2 TrueMotion. © 1992-2005 On2 Technologies, Inc. Tous droits réservés. <http://www.on2.com>.

Visual SourceSafe est une marque commerciale ou une marque déposée de Microsoft Corporation aux Etats-Unis et/ou dans d'autres pays.

Des informations récentes et des renseignements sur le code supplémentaire créé par des tiers sont disponibles à l'adresse suivante http://www.adobe.com/go/thirdparty_fr/.

Adobe Systems France Tour Maine Montparnasse, 33 Avenue du Maine, BP 14, 75755 PARIS Cedex 15

A l'attention des utilisateurs du Gouvernement des Etats-Unis. Ce logiciel et sa documentation sont des « articles commerciaux », conformément à la définition de ce terme dans le document 48 C.F.R. §2.101, comprenant d'une part un « logiciel informatique commercial » et d'autre part une « documentation de logiciel informatique commercial », conformément à la définition de ces termes dans le document 48 C.F.R. §12.212 ou 48 C.F.R. §227.7202, si approprié. Conformément à l'article 48 C.F.R., alinéa 12.212 ou 48 C.F.R., alinéas 227.7202-1 à 227.7202-4, selon le cas, la licence applicable aux « Commercial Computer Software » et « Commercial Computer Software Documentation » est accordée aux utilisateurs finaux faisant partie du gouvernement des Etats-Unis (a) en tant que Commercial Items et (b) uniquement selon les droits accordés aux autres utilisateurs finaux ayant accepté les termes et les conditions dudit contrat. Droits non publiés réservés aux termes de la loi sur les droits de reproduction des USA. Adobe Systems Incorporated, 345 Park Ave, San Jose, CA 95110-2704, Etats-Unis. A l'attention des utilisateurs du Gouvernement des Etats-Unis, Adobe s'engage à respecter la législation relative à l'égalité des chances y compris, le cas échéant, les dispositions du décret 11246, tel qu'amendé, à la section 402 de la loi sur l'assistance aux vétérans du Vietnam (Vietnam Era Veterans Readjustment Assistance Act) de 1974 (38 USC 4212), et à la section 503 de la loi sur la réadaptation (Rehabilitation Act) de 1973, telle qu'amendée, et la réglementation des articles 41 CFR, alinéas 60-1 à 60-60, 60-250 et 60-741. La clause relative à l'égalité des chances et les règlements énoncés dans la phrase précédente doivent être compris comme tels lorsqu'il y est fait référence.

Table des matières

Chapitre 1 : Formation à ActionScript Flash Lite 1.x	7
Présentation d'ActionScript Flash Lite 1.x	7
Différences entre ActionScript Flash Lite 1.0 et Flash Lite 1.1	8
Code ActionScript de Flash 4 non pris en charge par le code ActionScript de Flash Lite 1.x	9
Fonctionnalités non disponibles dans le code ActionScript Flash Lite 1.x	9
Chapitre 2 : Guide d'introduction au code ActionScript Flash 4	11
Récupération et définition des propriétés de clip	11
Contrôle d'autres scénarios	12
Utilisation des variables	13
Emulation des tableaux	13
Utilisation de texte et de chaînes	14
Utilisation de la fonction call() pour créer des fonctions	15
Utilisation de la fonction eval()	19
Chapitre 3 : Tâches de script courantes	21
Définition des fonctionnalités de la plate-forme et du périphérique	21
Ouverture d'une page Web	22
Lancement d'un appel téléphonique	23
Envoi d'un message texte ou multimédia	23
Envoi d'un message électronique	24
Chargement de fichiers SWF externes	24
Chargement de données externes	25
Index	29

Formation à ActionScript Flash Lite 1.x

Vous pouvez utiliser ActionScript pour ajouter une logique de programmation et de l'interactivité à vos applications Adobe® Flash® Lite™. La version d'ActionScript des logiciels Macromedia® Flash® Lite™ 1.0 et Macromedia® Flash® Lite™ 1.1 d'Adobe, communément appelée ActionScript de Flash Lite 1.x, est un code hybride de Macromedia® Flash® 4 ActionScript d'Adobe, doté de commandes et propriétés supplémentaires spécifiques au lecteur Flash Lite, telles que la possibilité d'initialiser des appels téléphoniques ou d'envoyer des messages textuels, ou encore de récupérer les informations relatives à l'heure et à la date à partir du périphérique.

Ce chapitre contient les sections suivantes :

Présentation d'ActionScript Flash Lite 1.x	7
Différences entre ActionScript Flash Lite 1.0 et Flash Lite 1.1	8
Code ActionScript de Flash 4 non pris en charge par le code ActionScript de Flash Lite 1.x	9
Fonctionnalités non disponibles dans le code ActionScript Flash Lite 1.x	9

Présentation d'ActionScript Flash Lite 1.x

ActionScript Flash Lite 1.x est composé des éléments suivants :

ActionScript Flash Player 4 Cet élément inclut les opérateurs (par exemple, opérateurs de comparaison et d'affectation), les propriétés de clip (par exemple, `_height`, `_x` et `_y`), les fonctions de contrôle du scénario (par exemple, `gotoAndPlay()` ou `stop()`) et les fonctions réseau, telles que les fonctions `loadVariables()` et `loadMovie()` (Flash Lite 1.1 uniquement). Pour obtenir une liste du code ActionScript Flash 4 non pris en charge, consultez la section « [Code ActionScript de Flash 4 non pris en charge par le code ActionScript de Flash Lite 1.x](#) », à la page 9.

Propriétés et commandes d'intégration téléphonique Flash Lite comprend des commandes permettant, par exemple, de connaître l'heure et la date du périphérique, de lancer un appel téléphonique ou d'envoyer un SMS, ou encore de démarrer des applications externes installées sur le périphérique.

Variables des fonctionnalités de la plate-forme (Flash Lite 1.1 uniquement)

Ces propriétés fournissent des informations relatives aux fonctionnalités du périphérique ou de l'environnement d'exécution Flash Lite. Par exemple, la variable `_capLoadData` indique si votre application peut charger des données via le réseau.

Fonction `fscommand2()` Comme la fonction `fscommand()`, vous pouvez utiliser la fonction `fscommand2()` pour communiquer avec l'environnement ou le système hôte (dans le cas présent, le téléphone ou le périphérique portable). La fonction `fscommand2()` a été améliorée par rapport à la fonction `fscommand()` ; elle est désormais en mesure de transmettre un nombre arbitraire d'arguments et d'en récupérer les valeurs renvoyées immédiatement (plutôt que d'avoir à attendre jusqu'à l'image suivante, comme c'est le cas avec la fonction `fscommand()`).

Différences entre ActionScript Flash Lite 1.0 et Flash Lite 1.1

Les fonctionnalités ActionScript suivantes de Flash Lite 1.1 ne sont pas disponibles dans Flash Lite 1.0 :

- Accès réseau ou informations relatives à l'état du réseau. Par exemple, dans Flash Lite 1.0, vous ne pouvez pas utiliser les fonctions `loadVariables()` ou `loadMovie()` pour charger des données ou fichiers SWF externes, ni les commandes `fscommand2()` pour déterminer l'intensité du signal de connexion d'un périphérique ou l'état d'une demande de connexion réseau.
- Récupération des informations relatives à l'heure et à la date depuis le périphérique.
- Variables des fonctionnalités de la plate-forme, qui fournissent des informations sur les fonctionnalités de la plate-forme Flash Lite et du périphérique.
- La fonction `fscommand2()` et ses commandes associées, telles que `SetSoftKeys` et `FullScreen`.
- Les propriétés de champ texte `scroll` et `maxscroll`.

Code ActionScript de Flash 4 non pris en charge par le code ActionScript de Flash Lite 1.x

Les fonctionnalités ActionScript de Flash 4 suivantes ne sont pas prises en charge, ou seulement partiellement, dans le code ActionScript de Flash Lite 1.x :

- Les fonctions `startDrag()` et `stopDrag()`.
- Le code ActionScript Flash Lite 1.x ne prend en charge qu'un sous-ensemble des événements de bouton pris en charge dans Flash Player 4. Pour plus d'informations sur la gestion des événements de bouton, consultez le Chapitre 1, « Création d'interactivité et d'éléments de navigation » dans le manuel *Développement d'applications Flash Lite*.
- Le code ActionScript Flash Lite 1.x ne prend en charge qu'un sous-ensemble des événements de touche pris en charge dans Flash Player 4. Pour plus d'informations sur les événements de touche pris en charge dans Flash Lite, consultez le Chapitre 1, « Création d'interactivité et d'éléments de navigation » dans le manuel *Développement d'applications Flash Lite*.
- La propriété `_dropTarget`.
- La propriété `_soundBufTime`.
- La propriété `_url`.
- La fonction de conversion `String()`.

Fonctionnalités non disponibles dans le code ActionScript Flash Lite 1.x

Le lecteur Flash Lite étant basé sur une ancienne version de Flash Player, il ne prend pas en charge toutes les fonctionnalités de programmation disponibles dans les versions plus récentes de Flash Player ou d'autres langages de programmation de votre connaissance. Cette section présente les fonctionnalités de programmation non disponibles dans le code ActionScript de Flash Lite 1.x ainsi que les solutions de remplacement disponibles.

Fonctions définies par l'utilisateur Flash Lite 1.x ne prend pas en charge la fonctionnalité de définition et d'appel de fonctions personnalisées. Cependant, vous pouvez utiliser la fonction `call()` pour exécuter du code se trouvant sur une image quelconque dans le scénario. Pour plus d'informations, consultez la section « [Utilisation de la fonction call\(\) pour créer des fonctions](#) », à la page 15.

Tableaux, objets ou autres types de données complexes natifs Flash Lite 1.x ne prend pas en charge les structures de données de tableau natives ni tout autre type de données complexes. Cependant, vous pouvez émuler des tableaux à l'aide de pseudo-tableaux ; cette technique implique l'utilisation de la fonction `eval()` permettant d'évaluer de façon dynamique les chaînes concaténées. Pour plus d'informations, consultez la section « [Emulation des tableaux](#) », à la page 13.

Chargement à l'exécution de fichiers image et audio externes Contrairement à la version pour ordinateurs de bureau de Flash Player, ActionScript Flash Lite 1.x ne peut pas charger de fichiers JPEG ou MP3 externes. Dans Flash Lite 1.1, la fonction `loadMovie()` vous permet de charger des fichiers SWF externes. Pour plus d'informations, consultez la section « [Chargement de fichiers SWF externes](#) », à la page 24.

Guide d'introduction au code ActionScript Flash 4

Le langage ActionScript de Macromedia Flash Lite 1.x d'Adobe est basé sur la version d'ActionScript, disponible pour la première fois dans Macromedia Flash Player 4 d'Adobe. En conséquence, plusieurs fonctionnalités de programmation proposées dans des versions ultérieures de Flash Player (version pour ordinateurs de bureau) ne sont pas disponibles dans les applications Flash Lite 1.x.

Si vous ne maîtrisez pas les fonctionnalités et la syntaxe du code ActionScript Flash 4 ou si vous avez oublié certaines informations depuis votre dernier projet de développement Flash, ce chapitre propose un guide d'introduction à l'utilisation du code ActionScript Flash 4 dans vos applications Flash Lite.

Ce chapitre contient les sections suivantes :

Récupération et définition des propriétés de clip	11
Contrôle d'autres scénarios	12
Utilisation des variables	13
Emulation des tableaux	13
Utilisation de texte et de chaînes	14
Utilisation de la fonction <code>call()</code> pour créer des fonctions	15
Utilisation de la fonction <code>eval()</code>	19

Récupération et définition des propriétés de clip

Pour récupérer ou définir une propriété de clip (si définissable), vous pouvez utiliser une syntaxe à point ou bien les fonctions `setProperty()` et `getProperty()`. Vous pouvez également utiliser la fonction `tellTarget()`.

Pour utiliser une syntaxe à point, indiquez le nom d'occurrence du clip suivi d'un point (.) puis du nom de la propriété. Par exemple, le code suivant renvoie la coordonnée d'écran *x* (représentée par la propriété de clip `_x`) du clip intitulé `cartoonArea` et attribue le résultat à une variable nommée `x_pos`.

```
x_pos = cartoonArea._x;
```

L'exemple suivant est identique à l'exemple précédent, mais il utilise la fonction `getProperty()` pour récupérer la position *x* du clip :

```
x_pos = getProperty(cartoonArea, _x);
```

La fonction `setProperty()` vous permet de définir une propriété d'occurrence de clip, comme indiqué dans l'exemple suivant :

```
setProperty(cartoonArea, _x, 100);
```

L'exemple suivant est identique au précédent mais il utilise une syntaxe à point :

```
cartoonArea._x = 100;
```

Vous pouvez également récupérer ou définir des propriétés de clip à partir d'une instruction `tellTarget()`. Le code suivant est identique à l'exemple `setProperty()` précédent :

```
tellTarget("/cartoonArea") {  
    _x = 100;  
}
```

Pour plus d'informations sur la fonction `tellTarget()`, consultez la section « [Contrôle d'autres scénarios](#) », à la page 12.

Contrôle d'autres scénarios

Pour spécifier le chemin d'un scénario, utilisez une syntaxe à barre oblique (/) associée à deux points (..) pour créer la référence du chemin. Vous pouvez également utiliser les notations `_levelN`, `_root` ou `_parent` de Flash 5 pour faire référence, respectivement, à un niveau d'animation spécifique, au scénario racine de l'application ou au scénario parent.

Supposons par exemple que le scénario principal de votre fichier SWF contienne une occurrence de clip appelée `box`. Cette occurrence `box` contient elle-même une autre occurrence de clip appelée `cards`. Les exemples suivants ciblent le clip `cards` depuis le scénario principal :

```
tellTarget("/box/cards")  
tellTarget("_level0/box/cards")
```

L'exemple suivant cible le scénario principal depuis le clip `cards` :

```
tellTarget(".././cards")  
tellTarget("_root")
```

L'exemple suivant cible le clip parent `cards` :

```
tellTarget("../cards")
tellTarget("_parent/cards")
```

Utilisation des variables

Pour spécifier une variable dans un scénario, utilisez une syntaxe à barre oblique (/) associée à des points (..) suivi de deux-points (:). Vous pouvez également utiliser la notation par point.

Le code suivant fait référence à la variable `car` dans le scénario principal :

```
/:car
_root.car
```

L'exemple suivant illustre la variable `car` dans une occurrence de clip se trouvant dans le scénario principal :

```
/mc1/mc2/:car
_root.mc1.mc2.car
```

L'exemple suivant illustre la variable `car` dans une occurrence de clip se trouvant dans le scénario actuel :

```
mc2/:car
mc2.car
```

Emulation des tableaux

Les tableaux permettent de créer et manipuler des listes triées d'informations telles que des variables et des valeurs. Cependant, Flash Lite 1.1 ne prend pas en charge les structures de tableau natives. Une technique courante de programmation Flash Lite (et Flash 4) consiste à émuler des tableaux à l'aide du traitement de chaîne. Un tableau émulé est également appelé un pseudo-tableau. Pour traiter les pseudo-tableaux, c'est la fonction ActionScript `eval()` qui vous permet d'accéder aux variables, propriétés ou clips en fonction de leur nom. Pour plus d'informations, consultez la section « [Utilisation de la fonction eval\(\)](#) », à la page 19.

Un pseudo-tableau comprend généralement deux variables minimum qui partagent le même nom de base suivi d'un suffixe numérique. Le suffixe représente l'indice de chaque élément du tableau.

Par exemple, vous créez les variables ActionScript suivantes :

```
color_1 = "orange";
color_2 = "green";
color_3 = "blue";
color_4 = "red";
```

Vous pouvez alors utiliser le code suivant pour passer en boucle sur les éléments du pseudo-tableau :

```
for (i = 1; i <=4; i++) {  
    trace (eval ("color_" add i));  
}
```

En plus de référencer les variables existantes, vous pouvez également utiliser la fonction `eval()` située à gauche d'une affectation de variable pour créer des variables lors de l'exécution. Supposons par exemple que vous souhaitiez conserver une liste des meilleurs scores d'une partie jouée par un utilisateur. Chaque fois que l'utilisateur termine une partie, vous ajoutez son score à la liste :

```
eval("highScore" add scoreIndex) = currentScore;  
scoreIndex++;
```

Chaque fois que ce code est exécuté, il ajoute un nouvel élément à la liste des meilleurs scores et incrémente alors la variable `scoreIndex` qui détermine les indices de tous les éléments de la liste. Vous pouvez par exemple terminer par les variables suivantes :

```
highScore1 = 2000  
highScore2 = 1500  
highScore3 = 3000
```

Utilisation de texte et de chaînes

Flash Lite contient des propriétés et commandes ActionScript de base permettant d'utiliser du texte. Vous pouvez récupérer et définir les valeurs des champs texte, concaténer des chaînes, coder ou décoder des chaînes de texte pour l'URL et créer des champs texte défilants.

Cette section contient les rubriques suivantes :

- « Concaténation de chaînes », à la page 14
- « Texte défilant », à la page 15

Concaténation de chaînes

Pour concaténer des chaînes dans Flash Lite, vous pouvez utiliser l'opérateur `add`, comme indiqué dans l'exemple suivant :

```
city = "Boston";  
team = "Red Sox";  
fullName = city add " " add team;  
// Résultat :  
// fullName = "Boston Red Sox"
```

Texte défilant

Vous pouvez utiliser la propriété `scroll` des champs texte dynamique et de saisie pour récupérer ou définir la position de défilement courante du champ. Vous pouvez également utiliser la position `maxscroll` pour déterminer la position de défilement courante d'un champ par rapport à la position de défilement maximale. Pour savoir comment créer un champ texte défilant, consultez la section « Création de texte défilant » dans le manuel *Développement d'applications Flash Lite*.

Utilisation de la fonction `call()` pour créer des fonctions

Vous ne pouvez pas définir ou appeler des fonctions personnalisées dans Flash Lite comme c'est le cas dans Flash Player 5 et les versions ultérieures. Cependant, vous pouvez utiliser la fonction ActionScript `call()` pour exécuter du code se trouvant sur une image quelconque dans le scénario. Cette technique vous permet d'encapsuler le code le plus utilisé dans un emplacement unique pour en faciliter la gestion.

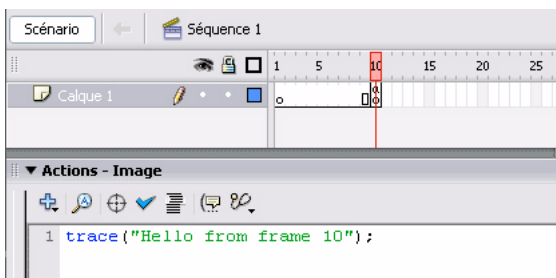
La fonction `call()` prend comme paramètre un numéro d'image ou d'étiquette. Par exemple, le code ActionScript suivant appelle le code situé sur l'image intitulée `moveUp` :

```
call("moveUp");
```

La fonction `call()` fonctionne de façon synchronisée ; tout code ActionScript suivant une fonction `call()` ne s'exécute que lorsque l'exécution de l'intégralité du code ActionScript est terminée au niveau de l'image spécifiée.

Pour appeler du code ActionScript sur une autre image :

1. Dans un nouveau document Flash, insérez une image-clé dans l'image 10.



2. Avec la nouvelle image-clé sélectionnée, ouvrez le panneau Actions (Fenêtre > Actions) et saisissez le code suivant :

```
trace("Hello from frame 10");
```

- Sélectionnez l'image-clé de l'image 1, puis tapez le code suivant dans le panneau Actions :

```
stop();  
call(10);
```

Ce code arrête la tête de lecture sur l'image 1, puis appelle le code de l'image 10.

- Testez l'application dans Adobe® Device Central™.

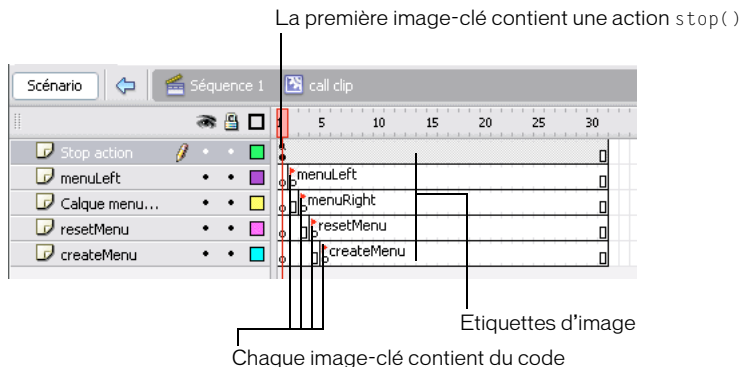
Le message « Hello from frame 10 » doit s'afficher dans l'émulateur.

Vous pouvez également appeler du code résidant dans un autre scénario, par exemple un scénario de clip. Pour exécuter le code, indiquez le nom d'occurrence du clip suivi de deux-points, puis du numéro d'image ou d'étiquette. Par exemple, le code ActionScript suivant appelle le code situé sur l'image intitulée `moveUp` dans l'occurrence de clip nommée `callClip` :

```
call("callClip:moveUp");
```

Cette technique est souvent utilisée pour créer des *call clips* (*clips d'appel*) ou *function clips* (*clips de fonction*) ; ces clips ont pour seule fonction d'encapsuler du code fréquemment utilisé. Un clip d'appel contient une image-clé de chaque fonction que vous souhaitez créer. Vous étiquetez généralement chaque image-clé en fonction de son rôle. Adobe recommande également de créer un nouveau calque pour chaque image-clé et de donner à chaque calque le même nom que celui de l'étiquette d'image assigné à l'image-clé.

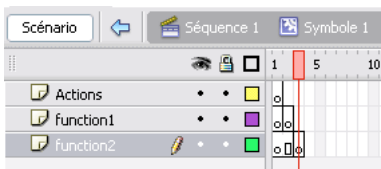
L'illustration suivante présente le scénario d'un exemple de clip d'appel. La première image-clé d'un clip d'appel contient toujours une action `stop()` permettant de garantir que la tête de lecture ne passe pas continuellement en boucle sur les images du scénario. Les images-clés suivantes contiennent du code pour chaque « fonction ». Chaque image-clé de fonction est étiquetée de manière à identifier sa fonction. Pour faciliter la modification et l'affichage du clip d'appel, chaque image-clé de fonction est généralement insérée sur un calque distinct.



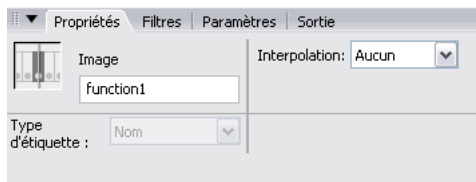
La procédure suivante explique comment créer et utiliser un clip d'appel.

Pour créer et utiliser un clip d'appel :

1. Dans Adobe® Flash® CS3, créez un nouveau document à partir du modèle de document Symbian Series 60 de Flash Lite 1.1.
2. Sélectionnez Insertion > Nouveau symbole.
3. Dans la boîte de dialogue Créer un nouveau symbole, tapez **Call Clip** dans le champ texte Nom, puis cliquez sur OK.
Le clip s'ouvre en mode d'édition.
4. Cliquez deux fois sur le bouton Insérez un calque dans la fenêtre Scénario pour insérer deux nouveaux calques.
Nommez le calque supérieur **Actions**, le deuxième calque **function1** et le troisième **function2**.
5. Insérez une image-clé dans l'image 2 du calque function1, et une autre image-clé dans l'image 3 du calque function2, comme indiqué ci-dessous :



6. Sélectionnez l'image-clé dans le calque Actions et ouvrez le panneau Actions.
7. Ajoutez une action `stop()` dans le panneau Actions.
8. Sélectionnez l'image-clé de l'image 2 dans le calque function1 puis effectuez l'une des opérations suivantes :
 - a. Dans l'inspecteur des propriétés, tapez **function1** dans la zone de texte Etiquette de l'image.

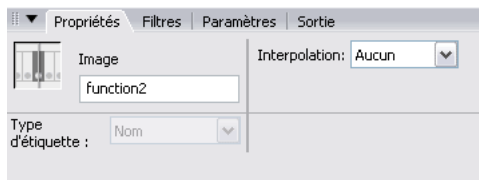


- b. Dans le panneau Actions (Fenêtre > Actions), tapez le code suivant :

```
trace("function1 was called.");
```

9. Sélectionnez l'image-clé de l'image 3 dans le calque fonction2, puis effectuez l'une des opérations suivantes :

- a. Dans l'inspecteur des propriétés, tapez **fonction2** dans la zone de texte Etiquette de l'image.



- b. Dans le panneau Actions (Fenêtre > Actions), tapez le code suivant :

```
trace("fonction2 was called.");
```

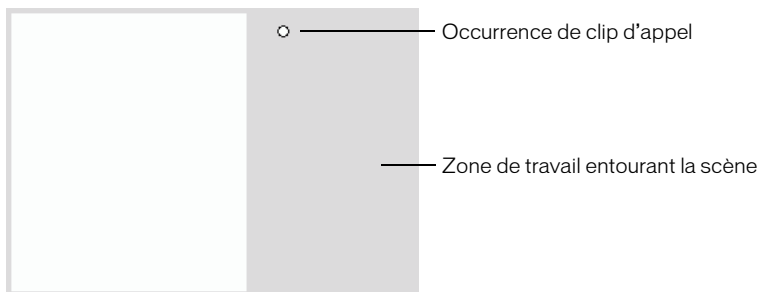
10. Appuyez sur Ctrl+E (Windows) ou sur Commande+E (Macintosh) pour retourner dans le scénario principal.

11. Paramétrez l'affichage de votre document de manière à inclure l'espace de travail entourant la scène en choisissant Affichage > Zone de travail.

L'utilisateur n'ayant pas besoin de consulter le clip d'appel, vous pouvez le placer dans la zone de travail.

12. Ouvrez le panneau Bibliothèque (Fenêtre > Bibliothèque) et faites glisser le symbole du clip d'appel dans la zone de travail entourant la scène.

Le clip d'appel ne contenant aucun élément visuel, il apparaît sur la scène sous la forme d'un petit cercle représentant le point d'alignement du clip.



CONSEIL

Afin de faciliter l'identification de votre clip d'appel sur la scène, ajoutez du texte ou tout autre élément visuel à la première image-clé dans le scénario du clip.

13. Dans l'inspecteur des propriétés, tapez **callClip** dans la zone de texte Nom de l'occurrence.

14. Dans le scénario, sélectionnez l'image 1 dans le calque intitulé ActionScript.

15. Dans le panneau Actions, entrez le code suivant :

```
call("callClip:function1");  
call("callClip:function2");
```

16. Testez votre application dans l'émulateur (Contrôle > Tester l'animation).

Vous pouvez utiliser le texte suivant :

```
function1 was called.  
function2 was called.
```

Utilisation de la fonction eval()

La fonction `eval()` vous permet de référencer de façon dynamique les variables et occurrences de clip lors de l'exécution. La fonction `eval()` prend comme paramètre une expression de chaîne et renvoie soit la valeur de la variable représentée par cette expression soit une référence au clip.

Par exemple, le code suivant évalue la valeur de la variable ActionScript `name` et attribue le résultat à `nameValue` :

```
name = "Jack";  
nameValue = eval("name");  
// résultat : nameValue = "Jack"
```

La fonction `eval()` est souvent utilisée en association avec des boucles `for()` et l'opérateur `add` (concaténation de chaîne) pour créer des tableaux basés sur des chaînes, Flash Lite ne prenant pas en charge les structures de données de tableau natives. Pour plus d'informations, consultez la section « [Emulation des tableaux](#) », à la page 13.

Vous pouvez également utiliser la fonction `eval()` pour référencer les occurrences de clip en fonction du nom. Supposons par exemple que vous disposiez de trois clips intitulés `clip1`, `clip2` et `clip3`. La boucle `for()` suivante incrémente la position `x` de chaque clip de 10 pixels :

```
for(index = 1; index <= 3; index++) {  
    eval("clip" + index)._x += 10  
}
```


Ce chapitre décrit les tâches de script courantes de Flash Lite permettant d'utiliser le périphérique de l'utilisateur. Elles incluent, par exemple, la récupération des informations relatives aux fonctionnalités du périphérique, le lancement d'appels téléphoniques et l'envoi de messages texte ainsi que la définition de l'état du réseau.

Ce chapitre contient les sections suivantes :

Définition des fonctionnalités de la plate-forme et du périphérique	21
Ouverture d'une page Web	22
Lancement d'un appel téléphonique	23
Envoi d'un message texte ou multimédia	23
Envoi d'un message électronique	24
Chargement de fichiers SWF externes	24
Chargement de données externes	25

Définition des fonctionnalités de la plate-forme et du périphérique

Macromedia Flash Lite 1.1 d'Adobe comporte de nombreuses variables ActionScript fournissant des informations sur les fonctionnalités disponibles dans les applications Flash Lite qui s'exécutent sur un périphérique spécifique. Par exemple, la variable `_capLoadData` indique si le périphérique prend en charge le chargement de données externes, et la variable `_capSMS` indique si le périphérique prend en charge l'envoi de messages SMS. Pour obtenir une liste complète des variables de fonctionnalité, consultez la section « Fonctionnalités » du *Guide de référence du langage ActionScript Flash Lite 1.x*.

Les variables de fonctionnalité sont généralement utilisées pour déterminer si un périphérique prend en charge une fonctionnalité spécifique avant de tenter d'utiliser cette dernière. Supposons par exemple que vous souhaitiez développer une application téléchargeant des données depuis un serveur Web à l'aide de la fonction `loadVariables()`. Avant de tenter de télécharger les données, vous pouvez d'abord vérifier la valeur de la variable `_capLoadData` pour déterminer si le périphérique prend en charge cette fonctionnalité en procédant comme suit :

```
if(_capLoadData == 1) {
    loadVariables("http://foo.com/data.txt");
} else {
    status_message = "Sorry, unable to load external data."
}
```

Flash Lite définit les variables de fonctionnalité dans le scénario racine du fichier SWF principal. Par conséquent, pour accéder à ces variables depuis un autre scénario (par exemple, depuis le scénario d'un clip), vous devez spécifier le chemin d'accès à la variable. L'exemple suivant utilise une barre oblique (/) pour fournir le chemin qualifié de la variable `_capSMS`.

```
canSendSMS = /:_capSMS
```

Ouverture d'une page Web

Utilisez la commande `getURL()` pour ouvrir une page Web dans le navigateur Web du périphérique. La procédure est la même que pour ouvrir une page Web depuis une application Flash pour ordinateurs de bureau. Par exemple, la procédure suivante permet d'ouvrir la page Web d'Adobe :

```
getURL("http:www.adobe.com");
```

Flash Lite ne traite qu'une seule action `getURL()` par image ou par gestionnaire d'événements. Certains téléphones limitent la fonction `getURL()` aux événements de touche uniquement, auquel cas l'appel `getURL()` n'est traité que s'il est déclenché dans un gestionnaire d'événements de touche. Même dans ces conditions, une seule fonction `getURL()` est traitée par gestionnaire d'événements de touche. Le code suivant, associé à une occurrence de bouton sur la scène, ouvre une page Web lorsque l'utilisateur appuie sur le bouton de sélection du périphérique :

```
on (keyPress "<Enter>"){
    getURL("http://www.adobe.com");
}
```

Lancement d'un appel téléphonique

Pour lancer un appel téléphonique depuis une application Flash Lite, utilisez la fonction `getURL()`. Vous utilisez généralement cette fonction pour ouvrir une page Web, mais dans le cas présent, vous devez spécifier `tel:` comme protocole (au lieu de `http`), puis fournir le numéro de téléphone à composer. Lorsque vous appelez cette fonction, Flash Lite affiche une boîte de dialogue de confirmation demandant à l'utilisateur l'autorisation de composer le numéro spécifié.

Le code suivant tente de lancer un appel vers le 555-1212 :

```
getURL("tel:555-1212");
```

Flash Lite ne traite qu'une seule action `getURL()` par image ou par gestionnaire d'événements. Certains téléphones limitent la fonction `getURL()` aux événements de touche uniquement, auquel cas l'appel `getURL()` n'est traité que s'il est déclenché dans un gestionnaire d'événements de touche. Même dans ces conditions, une seule fonction `getURL()` est traitée par gestionnaire d'événements de touche. L'exemple suivant lance un appel téléphonique lorsque l'utilisateur appuie sur le bouton de sélection du périphérique :

```
on (keyPress "<Enter>"){  
    getURL("tel:555-1212");  
}
```

Envoi d'un message texte ou multimédia

Vous pouvez utiliser Flash Lite pour envoyer des messages SMS ou MMS. Pour envoyer un message SMS ou MMS depuis une application Flash Lite, utilisez la commande `getURL()` en lui transmettant les protocoles `sms:` ou `mms:` au lieu du protocole standard `http`, puis indiquez le numéro de téléphone auquel envoyer le message.

```
getURL("sms:555-1212");
```

Vous pouvez également spécifier le corps du message dans la chaîne de requête de l'URL, comme le montre le code ci-dessous :

```
getURL("sms:555-1212?body=More info please");
```

Pour envoyer un message MMS, préférez le protocole `mms:` au protocole `sms:`, comme suit :

```
getURL("mms:555-1212");
```

REMARQUE

Vous ne pouvez pas spécifier de pièce jointe pour le message MMS depuis Flash Lite.

Flash Lite ne traite qu'une seule action `getURL()` par image ou par gestionnaire d'événements. Certains téléphones limitent la fonction `getURL()` aux événements de touche uniquement, auquel cas l'appel `getURL()` n'est traité que s'il est déclenché dans un gestionnaire d'événements de touche. Même dans ces conditions, une seule fonction `getURL()` est traitée par gestionnaire d'événements de touche. L'exemple suivant envoie un message SMS lorsque l'utilisateur appuie sur le bouton de sélection du périphérique :

```
on (keyPress "<Enter>"){
    getURL("sms:555-1212");
}
```

Envoi d'un message électronique

Vous pouvez utiliser Flash Lite pour envoyer un message électronique. Pour envoyer un message électronique, utilisez la commande `getURL()` en lui transmettant le protocole `mailto:` suivi de l'adresse électronique du destinataire. Vous pouvez également spécifier l'objet et le corps du message dans la chaîne de requête de l'URL, comme indiqué ci-dessous :

```
getURL("mailto:mobile-developer@adobe.com?subject=Flash Lite");
```

Pour spécifier simplement le corps du message dans la chaîne de requête, utilisez le code suivant :

```
getURL("mailto:mobile-developer@adobe.com?body=More+info+please");
```

Chargement de fichiers SWF externes

La fonction `loadMovie()` vous permet de charger des fichiers SWF à partir d'un réseau ou d'un fichier local. Cette fonctionnalité est disponible uniquement dans Flash Lite 1.1 et versions ultérieures. Les avertissements suivants s'appliquent lorsque vous chargez des fichiers SWF externes :

- Flash Lite peut charger d'autres fichiers SWF de Flash Lite 1.0 ou Flash Lite 1.1, ou bien des fichiers SWF au format Flash 4 ou antérieur. Si vous tentez de charger un fichier SWF dans un autre format (par exemple, un fichier SWF de Flash Player 6), Flash Lite génère une erreur d'exécution.

- Flash Lite ne peut pas charger directement des fichiers d'image externes, telles que des images JPEG ou GIF. Pour charger ces types de média, vous devez convertir les données image au format de fichier SWF. Vous pouvez effectuer cette opération « manuellement » à l'aide de l'outil de programmation de Flash en important le fichier d'image dans un nouveau document, puis en l'exportant dans un fichier SWF de Flash Lite ou de Flash 4. Il existe également des utilitaires tiers qui vous permettent d'effectuer ce type de conversion automatiquement.

Pour plus d'informations sur le chargement des fichiers SWF, consultez la section `loadMovie()` dans le *Guide de référence du langage ActionScript Flash Lite 1.x*.

Chargement de données externes

Pour charger des données externes dans une application Flash Lite, utilisez la fonction `loadVariables()`. Vous pouvez charger des données via le réseau (à partir d'une adresse HTTP) ou le système de fichiers local. Cette fonctionnalité est disponible uniquement dans Flash Lite 1.1 et versions ultérieures.

Cette section décrit l'utilisation de la fonction `loadVariables()` qui permet de charger des données à partir d'un fichier externe et de les afficher dans des champs texte dynamique. Vous allez d'abord créer le fichier de données, un fichier texte contenant cinq paires nom-valeur séparées par des esperluettes (&). Vous allez ensuite créer l'application Flash Lite qui charge et affiche les données contenues dans le fichier texte.

Cet exemple présume que le fichier de données et le fichier SWF se trouvent tous les deux dans le même dossier, soit sur votre ordinateur (si vous testez dans l'émulateur) soit sur la carte mémoire du périphérique (si vous testez sur un véritable périphérique). Pour tester l'application sur le périphérique, vous devez effectuer l'une des opérations suivantes :

- Transférez le fichier texte sur votre périphérique et placez-le dans le même dossier que le fichier SWF.
- Postez le fichier texte sur une URL d'un serveur Web (par exemple, `www.your-server.com/data.txt`). Modifiez ensuite l'appel `loadVariables()` dans l'exemple d'application de manière à pointer vers cette URL, comme indiqué ci-dessous :

```
loadVariables("http://www.your-server.com/data.txt", "data_clip");
```

Pour un exemple d'application chargeant des données via le réseau, consultez la section « Interface de Flash Lite pour la lecture de nouvelles » dans le guide *Exemples Flash*.

Pour créer le fichier de données :

1. À l'aide d'un éditeur de texte (par exemple, Notepad ou SimpleText), créez un fichier contenant le texte suivant :

```
item_1=Hello&item_2=Bonjour&item_3=Hola&item_4=Buon+giorno&item_5=G'day
```

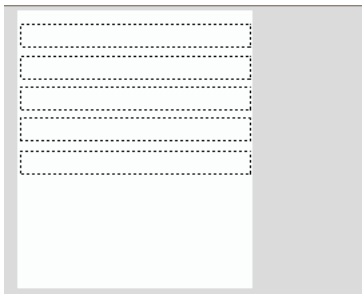
2. Enregistrez le fichier sous data.txt.

Pour créer l'application Flash Lite devant charger les données :

1. Créez un nouveau document à partir du modèle de document Symbian Series 60 de Flash Lite 1.1.

Pour plus d'informations sur l'utilisation de modèles de document Flash Lite, consultez la section « Utilisation des modèles de document Flash Lite » dans le guide *Prise en main de Flash Lite 1.x*.

2. Enregistrez le fichier sous le nom dataloading fla dans le dossier contenant le fichier texte (data.txt) que vous avez créé précédemment.
3. Dans le scénario, sélectionnez l'image 1 du calque nommé Content.
4. Avec l'outil Texte, créez cinq champs texte dynamique sur la scène, comme le montre l'illustration suivante :



5. Sélectionnez le premier champ texte (celui du haut), dans l'inspecteur de propriétés, tapez `item_1` dans la zone de texte Var.

Ce nom de variable correspond au nom de la première variable définie dans le fichier data.txt que vous avez précédemment créé (`item_1=Hello`).

6. De la même manière que celle décrite dans les deux étapes précédentes, affectez aux quatre champs texte restants les noms de variable `item_2`, `item_3`, `item_4` et `item_5`.
7. Appuyez sur la touche Maj et sélectionnez tous les champs, puis choisissez Modifier > Convertir en symbole.
8. Dans la boîte de dialogue Convertir en symbole, sélectionnez Clip comme type de symbole puis cliquez sur OK.

9. Sélectionnez le clip que vous venez de créer, et dans l'inspecteur de propriétés, tapez **data_clip** dans la zone de texte Nom de l'occurrence.
10. Dans le scénario, cliquez sur l'image 1 dans le calque Actions et ouvrez le panneau Actions (Fenêtre > Actions).
11. Tapez le code suivant dans le panneau Actions :

```
loadVariables("data.txt", "data_clip");
```
12. Enregistrez vos modifications (Fichier > Enregistrer), puis testez l'application dans l'émulateur (Contrôle > Tester l'animation).

Tous les champs texte devraient afficher les données du fichier texte, comme le montre l'illustration suivante :



Index

A

- ActionScript Flash Lite 1.x
 - ActionScript Flash 4 non pris en charge 9
 - différences entre 1.0 et 1.1 8
 - fonctionnalités non disponibles dans 9
 - présentation 7
- appel téléphonique, lancement 23

C

- chaînes, concaténation 14
- charger des données externes 25
- charger des fichiers SWF externes 24
- clips
 - récupération et définition des propriétés 11
 - référence dynamique 19
- clips de fonction, création 15
- concaténer des chaînes 14

E

- envoyer des messages 23

F

- fonction call(), utilisation 15
- fonction eval(), utilisation 19
- fonction getURL()
 - envoi d'un message électronique avec 24
 - envoi d'un message multimédia avec 23
 - envoi d'un message texte avec 23
 - lancement d'un appel téléphonique avec 23
 - ouverture de pages Web avec 22
- fonction loadMovie(), utilisation 24
- fonction loadVariables(), utilisation 25
- fonctions, émulation avec la fonction call() 15

L

- lancer des appels téléphoniques 23

M

- message électronique, envoi 24
- message multimédia, envoi 23
- message texte, envoi 23

O

- ouvrir une page Web 22

P

- page Web, ouverture 22

S

- scénarios, contrôle avec ActionScript 12

T

- tableaux, émulation avec des chaînes 13
- texte défilant, création 15

V

- variables
 - référence 13
 - référence dynamique 19
 - syntaxe à point et syntaxe à barre oblique 13
- variables de fonctionnalité de la plate-forme,
 - présentation 21

