Utilisation de ADOBE® FLASH® BUILDER™ 4.7

Informations juridiques

Pour consulter les informations juridiques, voir http://help.adobe.com/fr_FR/legalnotices/index.html.

Dernière mise à jour le 16/11/2012

Sommaire

Chapitre 1 : A propos de Flash Builder

Adobe® Flash® Builder™ est un environnement de développement intégré (IDE, integrated development environment) permettant de créer des applications Internet riches (RIA, rich Internet applications) utilisables sur plusieurs plateformes pour les ordinateurs de bureau et un grand nombre de périphériques mobiles. Il comporte également des outils de test, de débogage et de profilage favorisant productivité et efficacité.

Flash Builder repose sur Eclipse, un IDE à source libre, et fournit tous les outils requis pour développer des applications qui utilisent les structures Flex et ActionScript 3.0 à source libre.

Flash Builder prend intégralement en charge la création d'applications à l'aide du SDK Apache Flex. Lors de la création d'un projet Flex dans Flash Builder, vous pouvez spécifier l'utilisation du SDK Apache Flex. Pour plus d'informations sur le téléchargement et l'utilisation du SDK Apache Flex, voir www.adobe.com/go/apacheflex_download.

Flash Builder s'exécute sous Microsoft Windows et Apple Mac OS X et est disponible en plusieurs versions. Les options de configuration d'installation vous permettent d'installer Flash Builder en tant que série de plug-ins dans une installation existante du workbench Eclipse ou de créer une installation autonome comprenant le workbench Eclipse.

Applications que vous pouvez créer à l'aide de Flash Builder

Vous pouvez générer des applications qui utilisent la structure Flex, MXML, Adobe Flash Player, Adobe AIR, ActionScript 3.0 et LiveCycle Data Services :

- **Projets Flex** : créez des projets Flex compatibles avec toute technologie de serveur d'arrière-plan, notamment Adobe ColdFusion, LiveCycle Data Services et PHP.

  Voir « Projets Flex » à la page 61 et « Développement d'applications Web et d'ordinateur dans Flash Builder » à la page 171.

- **Projets ActionScript** : créez des projets ActionScript utilisant l'API Flash (et non la structure Flex).

  Voir « Projets ActionScript » à la page 64.

- **Projets mobiles** : créez des applications mobiles Flex pour la plateforme Google Android et des applications mobiles ActionScript pour la plateforme Apple iOS.

  Voir « Projets Flex Mobile » à la page 63 et « Création de projets ActionScript Mobile » à la page 64.

  Pour plus d'informations sur le développement d'applications mobiles avec Flex et Flash Builder, voir Developing mobile applications in Flash Builder (Développement d'applications mobiles dans Flash Builder).

- **Projets de bibliothèque** : créez des bibliothèques de code personnalisées que vous pourrez déployer en tant que fichiers de bibliothèque de composants (SWC) afin de les partager entre vos applications ou de les distribuer à d'autres développeurs.

  Voir « Utilisation de projets de bibliothèque Flex » à la page 200 et « Création de projets de bibliothèque ActionScript » à la page 204.

- **Projets Flash Professional** : créez des projets Flash Professional pour modifier et déboguer des fichiers Flash FLA ou XFL créés avec Flash Professional CS5.

  Voir « Utilisation de Flash Builder avec Flash Professional » à la page 254.

- **Composants MXML personnalisés** : créez des composants personnalisés, puis accédez-y facilement par le biais de la vue Composants Flash Builder.

  Voir « Création de composants MXML personnalisés » à la page 205.

  Vous pouvez aussi créer des composants ActionScript personnalisés. Voir « Création d'une classe ActionScript » à la page 70.

Fonctionnalités qui accélèrent le développement d'applications

Flash Builder dispose des outils nécessaires au développement d'applications qui utilisent la structure Flex et ActionScript 3.0. Vous pouvez :

- **Modification du code** : rédigez et modifiez le code source de l'application à l'aide des outils de développement de code de Flash Builder. Ceux-ci incluent la restructuration de code, les conseils de code, la navigation simplifiée dans le code et la vérification automatique de la syntaxe, entre autres.

  Voir « Outils de développement de code dans Flash Builder » à la page 16.

- **Publication du code source** : publiez le code source de votre application afin de permettre aux utilisateurs et aux autres développeurs d'y accéder.

  Voir « Publication du code source » à la page 103.

- **Gestion de projets, de dossiers, de fichiers et d'autres ressources** : créez, modifiez et supprimez des projets et des ressources, générez des liens vers des ressources extérieures à votre projet, etc.

  Voir « Utilisation des projets dans Flash Builder » à la page 59.

- **Personnalisation de workbench Flash Builder** : personnalisez le workbench en fonction de vos besoins personnels en termes de développement. Vous pouvez par exemple organiser l'interface de manière à inclure vos outils favoris dans une présentation particulière.

  Voir « Personnalisation du workbench » à la page 256.

Fonctionnalités d'exécution et de débogage de projets

Flash Builder intègre des outils de création, de test, de débogage et de profilage qui vous permettent d'améliorer votre productivité :

- **Création de projets** : Flash Builder compile et crée automatiquement vos applications de débogage ou de production. Vous pouvez aussi créer des scripts de création personnalisés à l'aide d'Apache Ant.

  Voir « Création de projets » à la page 85.

- **Exécution des applications et gestion des configurations de lancement** : exécutez vos applications dans un navigateur Web, Adobe AIR ou dans une instance autonome de Flash Player. Création de configurations de lancement personnalisées pour contrôler l'exécution des applications.

  Voir « Exécution et débogage des applications » à la page 105 et « Gestion des configurations de lancement » à la page 105.

- **Débogage des applications** : déboguez vos applications à l'aide des outils de débogage intégrés.

Voir « Débogage d'outils dans Flash Builder » à la page 121.

- **Exécution et débogage des applications mobiles** : exécutez et déboguez vos applications sur l'ordinateur ou sur un périphérique.

  Voir « Test et débogage » à la page 239.

- **Profilage des applications** : identifiez les problèmes de performance et les fuites de mémoire des applications à l'aide des outils de profilage de Flash Builder.

  Voir « Profilage d'outils dans Flash Builder » à la page 133.

- **Surveillance des applications qui accèdent aux services de données** : utilisez le Moniteur de réseau pour générer une piste d'audit détaillée de l'ensemble des données qui circulent entre vos applications et le serveur dorsal.

  Voir « Surveillance des applications qui accèdent aux services de données » à la page 196.

Versions de Flash Builder

Flash Builder est disponible en deux versions : Standard et Premium.

Flash Builder version Standard : cette version fournit un environnement IDE complet qui vous permet de créer des applications à l'aide de la structure Flex et de l'API Flash. Flash Builder version Standard inclut également des éditeurs MXML, ActionScript et CSS, ainsi que des outils de débogage. Il fournit une bibliothèque de diagrammes et de graphiques interactifs qui vous permettent de créer des tableaux de bord riches, des analyses de données interactives et des composants de visualisation des données.

Flash Builder version Premium : outre les fonctionnalités de la version Standard, Flash Builder Premium comporte des outils de profilage de la mémoire et des performances ainsi que des outils de tests automatisés. Utilisez le Moniteur de réseau pour consulter les données qui sont transmises entre une application client et un service de données. L'environnement de test FlexUnit permet de générer et de modifier des tests pouvant se répéter. Les tests pouvant se répéter s'exécutent à partir de scripts, directement dans Flash Builder ou en dehors de l'environnement Flash Builder. Les fonctionnalités de génération par ligne de commande vous permettent de synchroniser les paramètres de génération d'un développeur avec une génération nocturne.

Flash Builder pour PHP version Standard : Flash Builder pour PHP version Standard fournit un environnement IDE PHP complet qui inclut Flash Builder version Standard intégré avec Zend Studio 8. Vous pouvez créer des applications mobiles, Web et d'ordinateur utilisant PHP, Flex et ActionScript.

Flash Builder pour PHP version Premium : outre les fonctionnalités de la version Standard, Flash® Builder™ pour PHP version Premium inclut des outils de test professionnels, dont notamment des profileurs, la surveillance du réseau, une infrastructure de test automatisée, l'intégration avec les tests FlexUnit et la prise en charge de la génération par ligne de commande.

Pour plus d'informations sur Flash Builder pour PHP, voir Présentation de Flash Builder pour PHP.

Configurations Flash Builder

Flash Builder fournit un programme d'installation unique disponible au téléchargement, avec les deux options de configuration suivantes :

Configuration autonome : installe Flash Builder sous forme d'environnement de développement intégré (IDE) autonome. La configuration autonome est spécifiquement créée pour le développement d'applications qui utilisent

l'infrastructure Flex et ActionScript 3.0. Elle est idéale pour les nouveaux utilisateurs et pour ceux qui souhaitent développer des applications en utilisant uniquement l'infrastructure Flex et ActionScript 3.0.

Configuration plug-in : configure Flash Builder pour une exécution en tant que module externe au sein d'une installation Eclipse™ existante.

Pour exécuter la configuration plug-in, Eclipse 3.7 ou 4.2 doit être installé selon les conditions suivantes :

- Sous Windows, pour exécuter la configuration plug-in 32 bits de Flash Builder, installez Eclipse 3.7/4.2 32 bits.
- Sous Windows, pour exécuter la configuration plug-in 64 bits de Flash Builder, installez Eclipse 3.7/4.2 64 bits.
- Sous Mac, pour exécuter la configuration 64 bits de Flash Builder, installez Eclipse 3.7/4.2 64 bits.

Les configurations plug-in et autonome de Flash Builder fournissent les mêmes fonctionnalités. Si vous ne savez pas quelle configuration utiliser, suivez les recommandations suivantes :

- Si Eclipse 3.7 ou 4.2 est déjà installé, utilisez la configuration plug-in pour ajouter les fonctionnalités de Flash Builder à la copie existante d'Eclipse.
- Si aucune version d'Eclipse n'est installée et que votre objectif principal est de développer des applications Flex et ActionScript, utilisez la configuration autonome de Flash Builder. Cette configuration vous permet également d'installer d'autres plug-ins Eclipse afin d'étendre la portée de vos futurs travaux de développement.

Pour obtenir des informations détaillées sur l'installation de Flash Builder, voir les Notes de mise à jour sur Flash Builder 4.7.

Chapitre 2 : Initiation à Flash Builder

Flash Builder est un environnement de développement intégré (IDE) permettant de développer des applications utilisant la structure Flex et ActionScript 3.0. Vous pouvez développer des applications à déployer dans Adobe Flash Player, des applications de bureau à déployer dans Adobe AIR et des applications mobiles à déployer sur plusieurs types de périphériques mobiles.

Remarque : *Les informations nécessaires à l'utilisation de Flash Builder sont présentées dans la documentation de Flash Builder. Si vous utilisez d'autres plug-ins Eclipse (par exemple, CVS ou Java) avec Flash Builder, ou si vous souhaitez étendre les plug-ins Flash Builder, voir Guide de référence des extensibilités Adobe Flash Builder.*

Les environnements de développement Flash Builder comportent les composants suivants :

Workbench : le terme *workbench* se réfère à l'environnement de développement Flash Builder qui contient tous les outils permettant de développer des applications. Le workbench contient trois éléments principaux : les *perspectives*, les *éditeurs* et les *vues*. Vous utilisez ces trois composants dans différentes combinaisons à différents stades du processus de développement de l'application.

Voir « Personnalisation du workbench » à la page 256.

Remarque : *pour plus d'informations sur certaines fonctionnalités du workbench Eclipse, voir le guide de l'utilisateur du workbench Eclipse à l'adresse http://help.eclipse.org/help31/index.jsp.*

Perspective : une perspective représente un groupe de vues et d'éditeurs du workbench. Flash Builder contient deux perspectives : la perspective Développement Flash pour le développement des applications et la perspective Débogage Flash pour les applications de débogage. Flash Builder Premium contient également la perspective Profil Flash.

Si vous utilisez la configuration plug-in de Flash Builder (voir « Configurations Flash Builder » à la page 3), le workbench peut contenir d'autres perspectives telles qu'une perspective Java contenant des éditeurs et des vues permettant de développer des applications Java.

Voir « Perspectives et vues de Flash Builder » à la page 6.

Editeur : un éditeur vous permet d'éditer différents types de fichier. Les éditeurs dont vous disposez varient selon le nombre et le type de plug-ins Eclipse installés. Flash Builder contient des éditeurs permettant de rédiger du code MXML, ActionScript 3.0 et CSS (Cascading Style Sheets, feuilles de style en cascade).

Voir « Editeurs Flash Builder » à la page 12 et « Outils de développement de code dans Flash Builder » à la page 16.

Vue : une vue comporte généralement un éditeur. Par exemple, lorsque vous modifiez un fichier MXML ou ActionScript, les vues prises en charge s'affichent.

Voir « Utilisation de vues » à la page 11.

Remarque : *le terme vue est synonyme de panneau, ce dernier étant utilisé dans les versions précédentes de Flash Builder, dans Adobe Dreamweaver® et dans d'autres outils de développement Adobe.*

Espace de travail : à ne pas confondre avec le *workbench*, un *espace de travail* représente une zone définie d'un système de fichiers contenant les ressources (fichiers et dossiers) constituant vos projets d'application. Vous ne pouvez travailler que dans un seul espace de travail à la fois, mais vous pouvez cependant sélectionner un espace de travail différent à chaque lancement de Flash Builder.

Voir « Déplacement d'un projet d'un espace de travail vers un autre » à la page 79 et « Changement de l'espace de travail » à la page 81.

Ressource : le terme *ressource* est utilisé de façon générique pour désigner les fichiers et dossiers des projets d'un espace de travail.

Voir « Ressources des projets » à la page 68.

Projet : toutes les ressources constituant vos applications figurent dans des projets. Vous ne pouvez pas créer une application dans Flash Builder sans créer au préalable un projet. Flash Builder prend en charge les différents types de projets, selon le type d'application que vous créez.

Voir « Types de projets » à la page 59 et « Utilisation des projets dans Flash Builder » à la page 59.

Configuration de lancement : une *configuration de lancement* est créée pour chacun de vos projets. Elle définit les paramètres de projet utilisés lors de l'exécution et du débogage de vos applications. Par exemple, les noms et emplacements des fichiers SWF d'application compilés sont contenus dans la configuration de lancement, et vous pouvez modifier ces paramètres.

Voir « Gestion des configurations de lancement » à la page 105.

Perspectives et vues de Flash Builder

Pour prendre en charge une tâche particulière ou un groupe de tâches, les éditeurs et les vues correspondantes sont combinés dans une *perspective*. L'ouverture d'un fichier associé à une perspective particulière conduit à l'affichage automatique de cette perspective.

La configuration autonome de Flash Builder propose trois perspectives :

- Développement Flash

  Voir « Perspective Développement Flash » à la page 6.

- Débogage Flash

  Voir « Perspective Débogage Flash » à la page 121.

- Profil Flash

  Voir « Perspective Profil Flash et vues du profileur » à la page 136.

  Remarque : *la perspective Profil Flash est disponible dans Flash Builder Premium.*

Perspective Développement Flash

La perspective Développement Flash comporte les éditeurs et les vues dont vous avez besoin pour créer des applications pour la structure Flex. Lorsque vous créez un projet, Flash Builder affiche la perspective Développement pour que vous puissiez commencer à développer une application.

Le point central de la perspective (et du workbench en général) est la zone de l'éditeur. Elle affiche tous les documents actuellement ouverts dans une interface à plusieurs onglets

Lorsque vous créez un projet Flex, le fichier d'application MXML principal s'ouvre dans la zone de l'éditeur. Vous pouvez ensuite ouvrir et parcourir les documents MXML, ActionScript et CSS que vous utilisez. Pour plus d'informations, voir « Editeurs Flash Builder » à la page 12.

La perspective de développement contient les éditeurs et les vues de prise en charge suivantes :

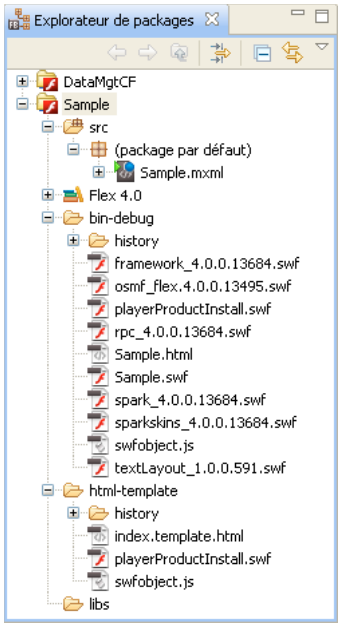
Vue Explorateur de packages

La vue Explorateur de packages contient l'ensemble des projets et des ressources de l'espace de travail et constitue par conséquent un élément essentiel du workbench Flash Builder. Elle apparaît toujours dans les perspectives Développement et Débogage.

Dans l'Explorateur de packages, vous pouvez sélectionner une ressource et en afficher les propriétés.

1 Dans l'Explorateur de packages, sélectionnez une ressource.

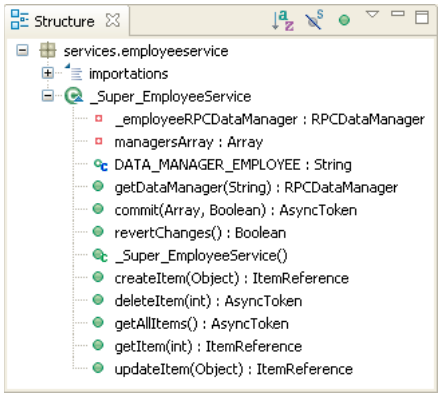
2 Sélectionnez Fichier > Propriétés.



Pour plus d'informations sur l'Explorateur de packages et sur l'utilisation de projets, voir « Gestion de projets » à la page 77.

Vue Structure

La vue Structure offre une présentation hiérarchique de la structure du code du document MXML ou ActionScript sélectionné, ce qui vous permet d'inspecter et de parcourir les sections ou lignes de code du document. La vue Structure affiche également les alertes d'erreur de syntaxe générées par le compilateur. Cette vue apparaît également lorsque vous utilisez l'éditeur ActionScript.

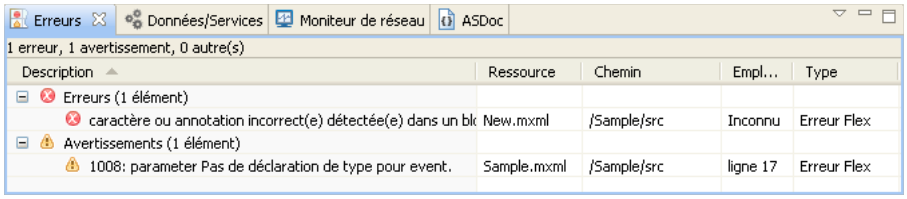


Pour plus d'informations sur l'utilisation de la vue Structure en mode Source, voir « Utilisation de la vue Structure » à la page 30.

Vue Erreurs

Lorsque vous saisissez du code, le compilateur Flash Builder détecte les erreurs de syntaxe et d'autres erreurs de compilation et les affiche dans la vue Erreurs. L'Explorateur de packages signale les nœuds contenant des erreurs.

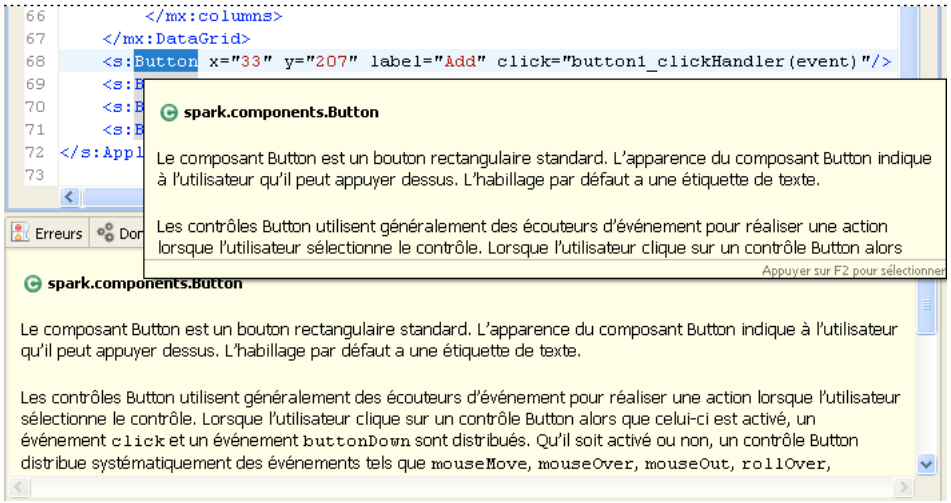
Lorsque vous déboguez vos applications, les erreurs, avertissements et autres informations apparaissent dans la vue Erreurs. Chaque erreur ou avertissement contient un message, le fichier et le dossier correspondants ainsi que le numéro de la ligne dans le fichier. Les erreurs demeurent affichées dans la vue Erreurs jusqu'à ce que vous les corrigiez ou qu'elles soient résolues.



Remarque : *vous pouvez également ajouter les vues Tâches et Signets. Ces vues offrent des raccourcis supplémentaires vous permettant de gérer et de parcourir le code. Pour plus d'informations sur ces vues, voir « Utilisation de marqueurs » à la page 34. Pour une présentation des vues optionnelles disponibles dans Flash Builder, voir « Affichage d'autres vues Workbench » à la page 257.*

Vue ASDoc

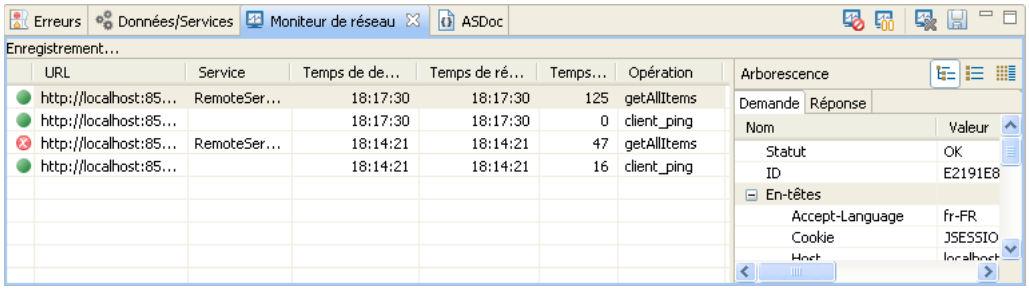
Flash Builder affiche le contenu ASDoc correspondant au code que vous saisissez ou aux éléments de code sur lesquels le curseur de la souris est positionné. Il affiche en outre dans la vue ASDoc le contenu ASDoc correspondant au code sélectionné.



Pour plus d'informations, voir « Affichage du résumé du guide de référence du langage ActionScript » à la page 19.

Vue Moniteur de réseau

Le Moniteur de réseau permet d'analyser les données transmises entre une application et le ou les services de données. Il est disponible avec Flash Builder Premium.



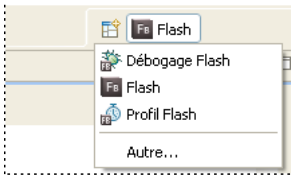
Pour plus d'informations, voir « Surveillance des applications qui accèdent aux services de données » à la page 196.

Utilisation de perspectives

Ouverture et changement de perspectives

Les perspectives changent automatiquement selon la tâche en cours. Par exemple, si vous créez un projet Flex, le workbench affiche la perspective Développement. Si vous lancez une session de débogage, la perspective Débogage Flash s'affiche lorsque le premier point d'arrêt est atteint.

Vous pouvez également changer manuellement de perspectives en sélectionnant Fenêtre > Ouvrir la perspective. Vous pouvez également utiliser la *barre des perspectives*, située dans la barre d'outils principale du workbench.



Le titre de la perspective que vous ouvrez remplace celui de la perspective précédemment affichée. Une icône apparaît en regard du titre. Elle permet de parcourir rapidement les perspectives dans la même fenêtre. Par défaut, les perspectives s'affichent dans la même fenêtre.

Si vous utilisez la configuration plug-in de Flash Builder et que vous avez installé d'autres plug-ins Eclipse, d'autres perspectives peuvent s'afficher. Chaque plug-in Eclipse propose des perspectives prédéfinies, mais vous pouvez les personnaliser ou créer vos propres perspectives. Pour afficher la liste complète des perspectives, sélectionnez Fenêtre > Ouvrir la perspective > Autre.

Définition d'une perspective par défaut

Le mot *par défaut* entre parenthèses qui suit le nom de la perspective indique qu'il s'agit de la perspective par défaut.

1 Ouvrez la boîte de dialogue Préférences et sélectionnez Général > Perspectives.

2 Sous Perspectives disponibles, sélectionnez la perspective à définir par défaut, puis cliquez sur Perspective par défaut.

3 Cliquez sur OK.

Ouverture d'une perspective dans une nouvelle fenêtre

Vous pouvez ouvrir les perspectives dans une nouvelle fenêtre.

1. Ouvrez la boîte de dialogue Préférences et sélectionnez Général > Perspectives.
2. Sous Ouverture d'une nouvelle perspective, sélectionnez Dans une nouvelle fenêtre.

   Pour rétablir les valeurs par défaut, sélectionnez Dans la même fenêtre.
3. Cliquez sur OK.

Personnalisation d'une perspective

Pour modifier la mise en forme d'une perspective, changez les éditeurs et les vues visibles dans une perspective donnée. Par exemple, vous pouvez afficher la vue Signets dans une perspective et la masquer dans une autre.

Pour créer une perspective, procédez comme suit :

1. Ouvrez une perspective existante.
2. Affichez les vues et les éditeurs de votre choix.
3. Sélectionnez Fenêtre > Perspective > Sauvegarder la perspective sous (Fenêtre > Sauvegarder la perspective sous dans la configuration de plug-in de Flash Builder).
4. Dans la boîte de dialogue Sauvegarder la perspective sous, entrez le nouveau nom de la perspective, puis cliquez sur OK.

Pour configurer une perspective, procédez comme suit :

1. Ouvrez la perspective à configurer.
2. Sélectionnez Fenêtre > Personnaliser la perspective.
3. Sélectionnez les onglets Raccourcis ou Commandes en fonction des éléments que vous souhaitez ajouter à la perspective personnalisée.
4. Cochez les cases correspondant aux éléments que vous souhaitez afficher dans les menus et dans les barres d'outils de la perspective sélectionnée.
5. Cliquez sur OK.
6. Sélectionnez Fenêtre > Sauvegarder la perspective sous.
7. Dans la boîte de dialogue Sauvegarder la perspective sous, entrez le nouveau nom de la perspective, puis cliquez sur OK.

   Lorsque vous enregistrez une perspective, Flash Builder ajoute le nom correspondant au menu Fenêtre > Ouvrir la perspective.

Suppression d'une perspective personnalisée

Vous pouvez supprimer des perspectives définies précédemment. Vous ne pouvez pas supprimer une perspective que vous n'avez pas créée.

1. Ouvrez la boîte de dialogue Préférences et sélectionnez Général > Perspectives.
2. Sous Perspectives disponibles, sélectionnez la perspective que vous souhaitez supprimer.
3. Cliquez sur Supprimer, puis sur OK.

Réinitialisation d'une perspective

Vous pouvez restaurer la présentation originale d'une perspective après l'avoir modifiée.

1. Ouvrez la boîte de dialogue Préférences et sélectionnez Général > Perspectives.
2. Sous Perspectives disponibles, sélectionnez la perspective à réinitialiser.

3 Cliquez sur Réinitialiser, puis sur OK.

Utilisation de vues

Déplacement et ancrage de vues

Vous pouvez déplacer les vues à l'intérieur du workbench, les ancrer à différents emplacements ou en annuler l'ancrage, en fonction de vos besoins.

1 Déplacez la vue vers l'emplacement souhaité en faisant glisser sa barre de titre.

Lorsque vous déplacez la vue dans le workbench, le curseur prend la forme d'une flèche noire pointant vers l'emplacement où la vue sera ancrée une fois le bouton de la souris relâché.

Vous pouvez faire glisser un groupe de vues empilées en cliquant sur l'espace vide situé à droite des onglets des vues et en déplaçant la souris.

Vous pouvez également déplacer une vue à l'aide de son menu contextuel. Ouvrez le menu contextuel à partir de l'onglet de la vue, sélectionnez Déplacer > Vue, déplacez la vue vers l'emplacement souhaité, puis cliquez de nouveau avec le bouton de la souris.

2 (Facultatif) Enregistrez les modifications en sélectionnant Fenêtre > Sauvegarder la perspective sous.

Réorganisation des onglets de vue

Outre l'ancrage des vues à différents emplacements du workbench, vous pouvez réorganiser les onglets des vues à l'intérieur d'un groupe de vues.

❖ Cliquez sur l'onglet de la vue à déplacer, faites glisser la vue vers l'emplacement souhaité, puis relâchez le bouton de la souris. Un symbole représentant une pile apparaît lorsque vous faites glisser l'onglet d'une vue sur d'autres onglets.

Changement de vues

Le passage d'une vue à l'autre peut se faire de différentes façons :

- Cliquez sur l'onglet de la vue à laquelle vous souhaitez accéder.
- Sélectionnez une vue dans le menu Fenêtre de Flash Builder.
- Utilisez un raccourci clavier.

  Utilisez Ctrl+F7 sous Windows ou Commande+F7 sous Macintosh. Appuyez sur la touche F7 pour sélectionner une vue.

  Pour obtenir la liste des raccourcis clavier, cliquez sur Aide > Assistant de touches.

Création et utilisation des vues rapides

Les vues rapides sont des vues masquées que vous pouvez ouvrir et fermer rapidement. Elles sont identiques aux autres vues à ceci près qu'elles n'encombrent pas le workbench.

Cliquez sur l'icône de vue rapide dans la barre de raccourcis pour afficher la vue. Cliquez à l'extérieur de la vue rapide (ou sur le bouton Réduire de sa barre d'outils) pour la masquer à nouveau.

Remarque : *si vous convertissez la vue Explorateur de packages en vue rapide, puis ouvrez un fichier à partir de la vue rapide Explorateur de packages, la vue est automatiquement masquée pour vous permettre d'utiliser ce fichier.*

Pour créer une vue rapide, procédez comme suit :

- Faites glisser la vue que vous souhaitez convertir en vue rapide vers la barre de raccourcis située dans l'angle inférieur gauche de la fenêtre du workbench.

  L'icône de la vue que vous avez déplacée apparaît dans la barre de raccourcis. Cliquez sur cette icône pour ouvrir la vue. Cliquez à l'extérieur de la vue pour la masquer.

Pour restaurer une vue rapide en vue normale, procédez comme suit :

- Dans le menu contextuel de la vue, désélectionnez Vue rapide.

Filtrage des vues Tâches et Erreurs

Vous pouvez filtrer les tâches ou les problèmes qui s'affichent dans les vues Tâches ou Erreurs. Par exemple, pour voir uniquement les erreurs que le workbench a consignées, ou les tâches que vous avez marquées comme rappel personnel, filtrez les éléments avec lesquels la ressource ou le groupe de ressources sont associés. Vous pouvez filtrer par chaîne de texte dans le champ Description, par gravité d'erreur, par priorité de tâche ou par état de tâche.

1. Dans la barre des tâches de la vue Tâches ou Erreurs, cliquez sur Filtrer.
2. Renseignez la boîte de dialogue Filtres, puis cliquez sur OK.

Pour plus d'informations sur les vues, voir « Utilisation de vues » à la page 11.

Editeurs Flash Builder

Le développement d'applications dans Flash Builder repose sur l'utilisation des éditeurs MXML, ActionScript 3.0 et CSS.

Les éditeurs sont associés à des types de ressource. Ainsi, lorsque vous ouvrez des ressources dans le workbench, l'éditeur correspondant s'affiche. Chaque éditeur intègre les fonctionnalités nécessaires au type de ressource donné.

Editeur MXML : Utilisez l'éditeur MXML pour modifier le code MXML et pour incorporer du code ActionScript et CSS dans les balises `<fx:Script>` et `<fx:Style>`.

Editeur ActionScript : utilisez l'éditeur ActionScript pour éditer des fichiers de classe et d'interface ActionScript. Vous pouvez incorporer des fonctions ActionScript dans un fichier MXML à l'aide d'une balise `<fx:Script>`. Il est toutefois de pratique courante de définir les classes dans des fichiers ActionScript distincts et d'importer ensuite les classes dans des fichiers MXML. Cette méthode vous permet de définir la plupart de vos applications dans ActionScript.

Editeur CSS : utilisez l'éditeur CSS pour afficher et éditer des feuilles de style en cascade. Vous pouvez ensuite appliquer des styles aux éléments visuels de vos applications. Pour plus d'informations, voir Styles et thèmes.

Voir aussi

« Outils de développement de code dans Flash Builder » à la page 16

Trucs et astuces concernant Flash Builder

Utilisation de l'éditeur MXML

Lors de l'utilisation de l'éditeur MXML, l'assistant de contenu vous aide lors de l'ajout de conteneurs et de contrôles Flex standard à l'interface utilisateur.

L'exemple suivant illustre comment utiliser l'assistant de contenu pour insérer un conteneur `<s:VGroup>` dans un conteneur `<s:HGroup>`. HGroup étant un composant Spark, l'assistant privilégie l'utilisation d'un conteneur VGroup à celle de `<mx:VBox>`.

1 Ouvrez le fichier MXML dans l'éditeur MXML.

Le fichier MXML peut être le fichier de l'application principale (fichier doté d'un conteneur Application) ou un fichier de composants MXML personnalisés.

2 Placez le point d'insertion dans la balise de conteneur parent.

Pour insérer par exemple un conteneur VGroup dans un conteneur parent HGroup, placez le point d'insertion après la balise d'ouverture `<s:HGroup>`.

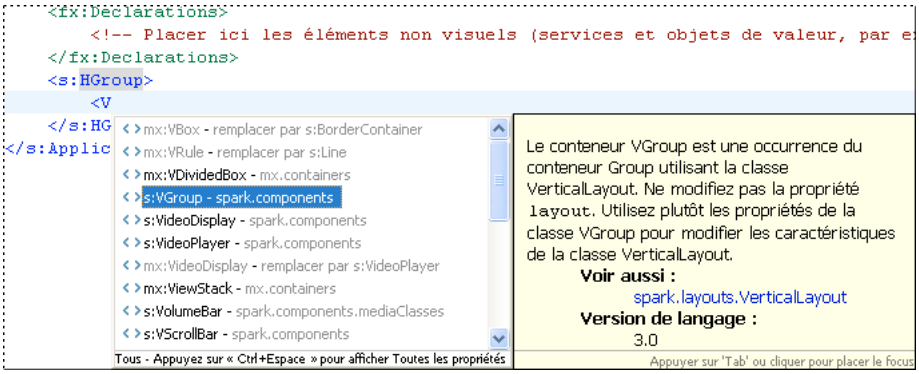
```
<s:HGroup>
    insertion point here
</s:HGroup>
```

3 Saisissez la balise du composant.

En cours de saisie, l'assistant de contenu s'affiche pour vous suggérer des entrées possibles. Les composants recommandés s'affichent en noir. Les composants non recommandés s'affichent en gris.

Dans cet exemple, VGroup est recommandé, contrairement à VBox qui ne l'est pas.

4 Appuyez le cas échéant sur les touches représentant des flèches pour sélectionner une balise dans le menu, puis appuyez sur Entrée.



Outre les composants standard, l'assistant de contenu répertorie les composants personnalisés que vous avez définis dans des fichiers MXML et ActionScript distincts et que vous avez enregistrés dans le projet en cours ou dans son chemin source.

L'assistant de contenu peut également suggérer des propriétés, des événements, des effets et des styles. Appuyez sur les touches Ctrl+Espace pour parcourir les recommandations de l'assistant de contenu.

Vous pouvez modifier le type et l'ordre des recommandations de l'assistant de contenu. Dans la boîte de dialogue Préférences, sélectionnez Flash Builder > Editeurs > Code MXML > Avancé.

Voir aussi

« Assistant de contenu, Assistant rapide et Correctif rapide » à la page 16

« Création de composants MXML personnalisés » à la page 205

Association d'éditeurs à des types de fichier

Le workbench permet d'associer les éditeurs avec différents types de fichiers.

1 Sélectionnez Fenêtre > Préférences.

2 Cliquez sur le signe plus pour développer la catégorie Général.

3 Cliquez sur le signe plus pour développer la catégorie Editeurs, puis sélectionnez Associations de fichiers.

4 Sélectionnez un type de fichier dans la liste Types de fichier.

Pour ajouter un type de fichier à la liste, cliquez sur Ajouter, entrez le nouveau type de fichier dans la boîte de dialogue Ajouter un type de fichier, puis cliquez sur OK.

5 Dans la liste Editeurs associés, sélectionnez l'éditeur à associer à ce type de fichier.

Pour ajouter à la liste un éditeur interne ou externe, cliquez sur Ajouter et renseignez la boîte de dialogue.

6 Cliquez sur OK.

Vous pouvez modifier les préférences par défaut de l'éditeur à partir du menu contextuel de toute ressource se trouvant dans l'une des vues de navigation. Dans le menu contextuel, sélectionnez Ouvrir avec.

Edition de fichiers en dehors du workbench

Vous pouvez éditer un fichier MXML ou ActionScript dans un éditeur externe puis l'utiliser dans Flash Builder. Le workbench effectue les opérations de création ou d'actualisation nécessaires afin d'appliquer les modifications que vous avez apportées au fichier en dehors du workbench.

Actualisation d'un fichier MXML ou ActionScript édité en dehors du workbench

1 Editez le fichier MXML ou ActionScript dans l'éditeur externe de votre choix.

2 Enregistrez et fermez le fichier.

3 Démarrez Flash Builder.

4 Dans l'une des vues de navigation du workbench, ouvrez le menu contextuel, puis sélectionnez Actualiser.

Si vous utilisez régulièrement des éditeurs externes, vous pouvez activer l'actualisation automatique. Pour ce faire, sélectionnez Fenêtre > Préférences, développez la catégorie Général, sélectionnez Espace de travail, puis cochez l'option Actualiser automatiquement. Lorsque vous activez cette option, le workbench consigne toutes les modifications externes apportées au fichier. La vitesse de l'actualisation automatique dépend de votre plateforme.

Mosaïque d'éditeurs

Le workbench vous permet d'ouvrir plusieurs fichiers dans différents éditeurs. Mais contrairement aux vues, les éditeurs ne peuvent pas être déplacés à l'extérieur du workbench pour créer des fenêtres. Vous pouvez cependant disposer les éditeurs en mosaïque dans la zone d'édition afin d'afficher les fichiers source côte à côte.

1 Ouvrez deux ou plusieurs fichiers dans la zone d'édition.

2 Cliquez sur l'onglet de l'un des éditeurs.

3 Faites-le glisser vers le bord gauche, droit, supérieur ou inférieur de la zone d'édition.

Le pointeur prend la forme d'une flèche noire pointant vers l'emplacement où la vue est ancrée une fois le bouton de la souris relâché.

4 (Facultatif) Faites glisser les bords des éditeurs pour les redimensionner, si besoin est.

Agrandissement d'une vue ou d'un éditeur

Vous pouvez procéder de différentes façons pour agrandir (restaurer) une vue ou un éditeur de manière à occuper entièrement la fenêtre du workbench.

- Accédez au menu contextuel de la barre de titre de la vue ou de l'éditeur et sélectionnez Agrandir (Restaurer).
- Cliquez deux fois sur l'onglet de la vue.
- Dans le menu Fenêtre de Flash Builder, sélectionnez Agrandir/Rétablir l'éditeur.
- Cliquez sur les icônes Agrandir/Restaurer situées dans l'angle supérieur droit de la vue ou de l'éditeur.

Chapitre 3 : Outils de développement de code dans Flash Builder

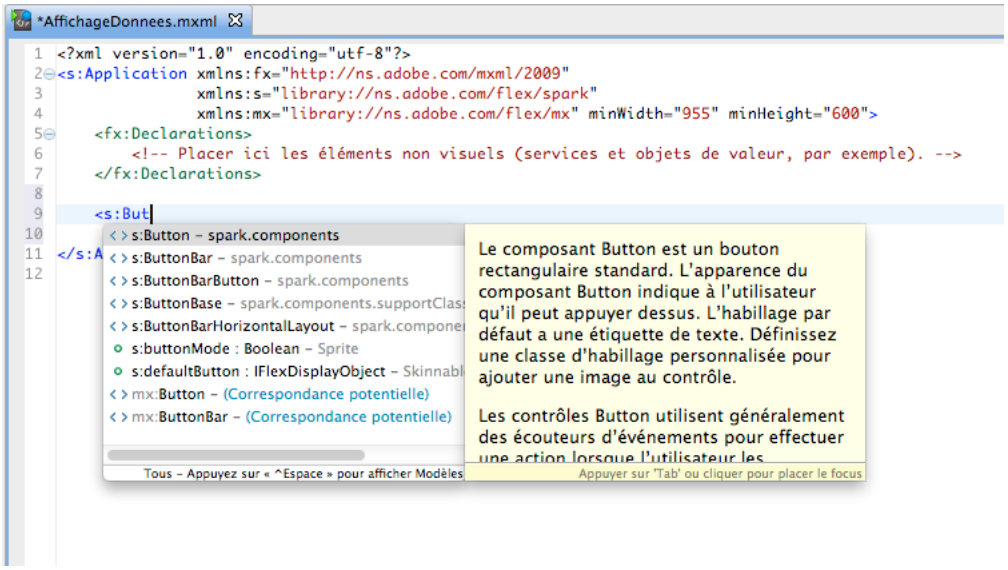
Dans Adobe® Flash® Builder™, la modification des codes MXML, ActionScript et CSS advient dans des éditeurs différents. Le workbench de Flash Builder intègre à la fois les projets et les documents, permettant ainsi l'ouverture automatique de l'éditeur correspondant au type de ressource. Les éditeurs de Flash Builder partagent les mêmes fonctionnalités, parmi lesquelles les conseils de code, la navigation, le formatage, la restructuration ainsi que d'autres fonctions d'amélioration de la productivité.

Assistant de contenu, Assistant rapide et Correctif rapide

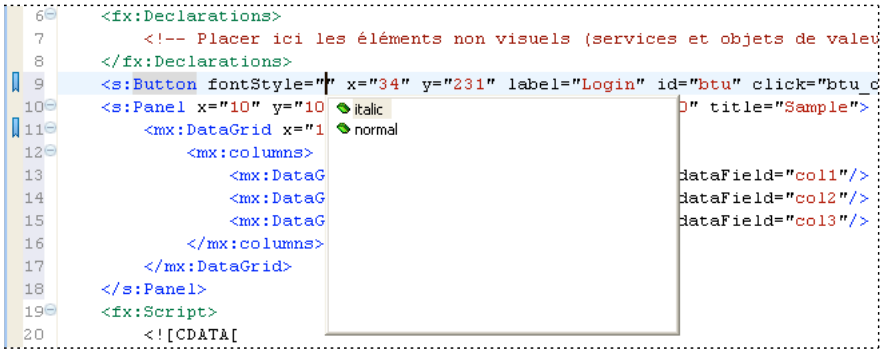
Assistant de contenu

En cours de rédaction de code MXML, ActionScript et CSS, des conseils et des documents de référence ASDoc s'affichent afin de faciliter la saisie. Cette fonction porte le nom d'*assistant de contenu*.

Dans l'éditeur MXML, la saisie dans le cadre d'un composant MXML conduit par exemple à l'affichage d'une liste contenant toutes les propriétés de ce composant. L'exemple suivant illustre les conseils de code pour les propriétés d'un composant MXML.



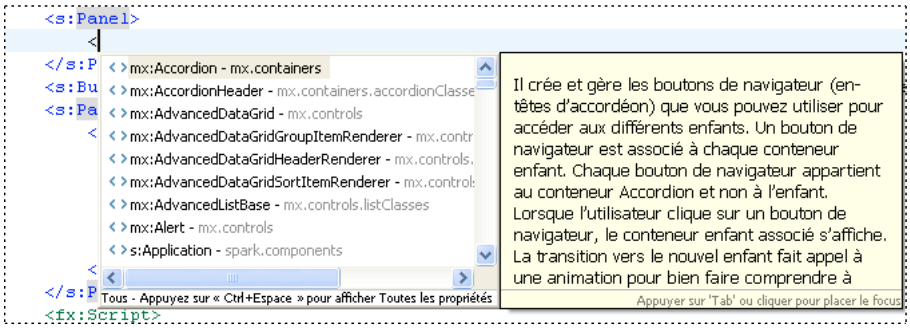
La sélection et la saisie d'une propriété entraînent l'affichage des valeurs possibles (à condition que des valeurs prédéfinies existent). L'exemple suivant illustre les conseils de code pour les valeurs de propriété.



Le fonctionnement de l'assistant de contenu dans les éditeurs ActionScript et CSS est analogue.

Assistant de contenu dans l'éditeur MXML

Dans l'éditeur MXML, les conseils de code s'affichent automatiquement à la saisie du code. L'exemple suivant illustre les conseils de code qui s'affichent lorsque vous ajoutez une balise à une balise Panel, ainsi que la documentation de référence ASDoc. Cliquez à l'intérieur du contenu ASDoc ou appuyez sur la touche F2 pour afficher le contenu dans une fenêtre distincte que vous pouvez faire défiler. Cliquez en dehors de la fenêtre ASDoc pour la fermer.



L'assistant de contenu classe les conseils de code par type, répertoriant les composants MXML visuels et non visuels, par propriétés, par événements et par styles.

Par défaut, l'assistant de contenu affiche uniquement les indicateurs correspondant aux *types recommandés*. Les types recommandés sont les composants disponibles dans les espaces de noms ou autres, suivant les balises englobantes. Les composants dont vous disposez dépendent de la déclaration d'espace de noms de l'application concernée ainsi que des balises englobant le point d'insertion dans l'éditeur.

Dans certains contextes par exemple, seuls les composants Spark sont autorisés. D'autres contextes autorisent les composants Spark et les composants Halo. L'assistant de contenu filtre les conseils de code en fonction du contenu.

Appuyez à plusieurs reprises sur les touches Ctrl+Espace pour faire défiler les filtres pour les conseils de code affichés. Les filtres sont les suivants :

- Affichage initial : Types recommandés
- Tous les composants
- Propriétés

- Evénements
- Effets
- Styles
- Retour aux types recommandés

Les conseils de code affichés et l'ordre dans lequel ils défilent dépendent des préférences définies par l'utilisateur. Pour modifier le paramètre par défaut dans la boîte de dialogue Préférences, voir « Code MXML » à la page 263.

Assistant de contenu dans l'éditeur ActionScript

Les conseils de code apparaissent automatiquement lorsque vous saisissez du code ActionScript dans l'éditeur ActionScript.

L'assistant de contenu filtre les conseils de code en fonction du contenu. Appuyez à plusieurs reprises sur les touches Ctrl+Espace pour faire défiler les filtres pour les conseils de code affichés. Les filtres sont les suivants :

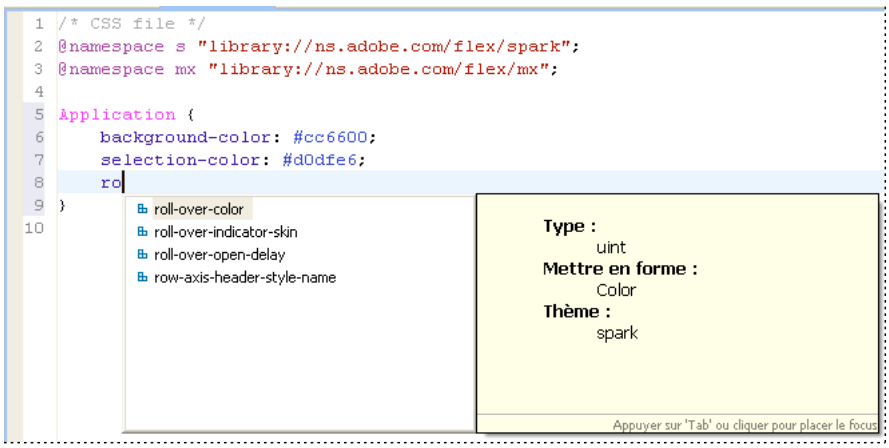
- Modèles
- Variables
- Fonctions
- Classes et interfaces
- Packages
- Espaces de noms

Les conseils de code affichés et l'ordre dans lequel ils défilent dépendent des préférences définies par l'utilisateur. Pour modifier le paramètre par défaut dans la boîte de dialogue Préférences, voir « Code ActionScript » à la page 262.

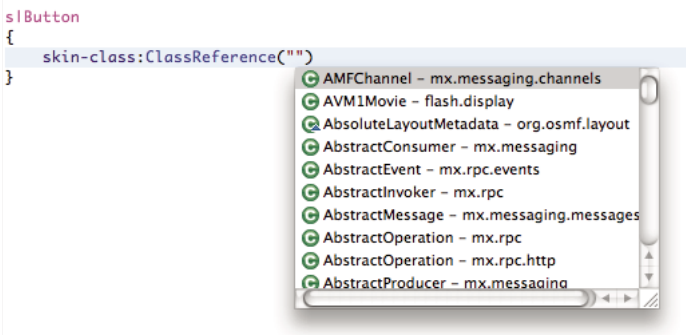
Assistant de contenu dans l'éditeur CSS

L'assistant de contenu affiche des indicateurs pour les styles CSS contenus dans des balises intégrées `<fx:Style>` ou dans des documents CSS autonomes, comme le montre l'exemple suivant.

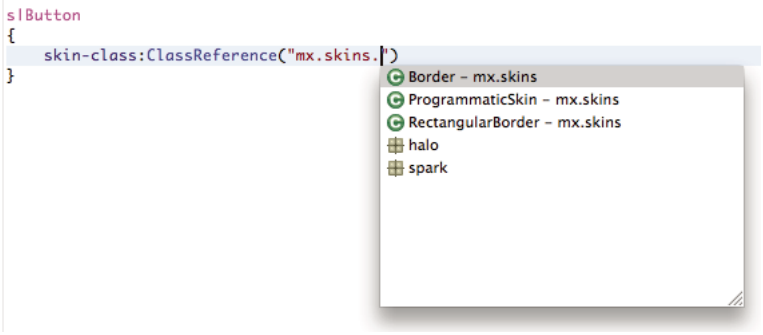
Remarque : *dans les documents CSS, les conseils de code s'affichent uniquement si vous appuyez sur Ctrl+Espace.*



L'assistant de contenu propose également des indicateurs pour les classes de composants contenues dans la balise ClassReference d'un document CSS, comme le montre l'exemple suivant :



Vous pouvez également saisir les noms complets des classes. Dans ce cas, les classes et packages disponibles s'affichent comme suit :



La balise ClassReference fournit une navigation par hyperliens qui vous permet d'accéder au composant référencé ou à la classe d'habillage. Pour ce faire, appuyez sur la touche Ctrl (touche Commande sur Mac) et déplacez le curseur sur la balise. Le nom de la classe se transforme en hyperlien. Pour plus d'informations sur la navigation par hyperliens, voir « Ouverture des définitions de code » à la page 29.

Affichage du résumé du guide de référence du langage ActionScript

1 Commencez à entrer une ligne de code qui contient une classe MXML ou ActionScript. Vous pouvez également survoler la classe avec le curseur de la souris.

   Au fur et à mesure que vous tapez, le résumé du guide de référence du langage ActionScript correspondant à la classe apparaît à côté des conseils de code. Si vous survolez une classe avec le curseur de la souris, seul le résumé du guide de référence du langage ActionScript s'affiche.

2 Cliquez à l'intérieur du résumé du guide de référence du langage ActionScript ou appuyez sur la touche F2 pour afficher le contenu du guide de référence du langage ActionScript dans une fenêtre distincte que vous pouvez faire défiler. Lorsque vous avez terminé la lecture de la documentation, cliquez hors de la fenêtre. La fenêtre ASDoc se ferme.

3 Pour afficher la vue ASDoc, appuyez sur Ctrl+3, puis saisissez asdoc et sélectionnez Vues.

Assistant rapide

La fonctionnalité Assistant rapide d'Adobe Flash Builder fournit une assistance contextuelle qui permet de réaliser rapidement une tâche. L'Assistant rapide vous permet de sélectionner une action dans une liste d'actions applicables à votre fragment de code actuel.

Pour appeler l'assistant rapide, effectuez l'une des opérations suivantes :

- Sélectionnez Assistant rapide dans le menu contextuel de l'éditeur et sélectionnez l'option requise.
- Utilisez le raccourci clavier Ctrl+1 (Windows) ou Commande+1 (Mac OS) et sélectionnez l'option requise.

Les options suivantes de l'Assistant rapide sont actuellement disponibles :

Renommer dans le fichier
Utilisez l'Assistant rapide pour renommer les éléments de code pour lesquels l'action Renommer/Restructurer est activée. Il peut s'agir de noms de variables, méthodes, classes, de paramètres, d'importations, d'états et du code ActionScript à l'intérieur des balises MXML.

Pour renommer toutes les instances d'une variable ou méthode dans votre fichier, placez le curseur sur le nom de la variable ou méthode sélectionnée, puis appelez l'assistant rapide. Sélectionnez ensuite l'option Renommer dans le fichier pour renommer la variable ou la méthode. De même, vous pouvez modifier la propriété d'ID d'un composant MXML.

Renommer dans l'espace de travail
Utilisez l'Assistant rapide pour renommer les éléments de code pour lesquels l'action Renommer/Restructure est activée dans tous les fichiers de votre espace de travail.

Pour renommer toutes les instances d'une variable ou méthode dans votre espace de travail, placez le curseur sur le nom de la variable ou méthode sélectionnée, puis appelez l'assistant rapide. Sélectionnez ensuite l'option Renommer dans l'espace de travail pour renommer la variable ou la méthode. Vous pouvez actualiser toutes les références à la variable ou méthode dans votre espace de travail.

Organiser les importations
Placez le curseur sur une instruction d'importation et appelez l'assistant rapide. Sélectionnez ensuite l'option d'organisation des importations. Pour plus d'informations sur l'organisation des instructions d'importation, voir « Organisation des instructions d'importation » à la page 36.

Générer l'instruction d'importation
Si vous disposez d'une variable non définie, placez votre curseur n'importe où sur la ligne de code, puis appelez l'assistant rapide. Une option permettant d'importer le composant s'affiche. Si le composant possède des équivalents MX et Spark, les deux options apparaissent.

Par exemple, si votre code comporte la variable non définie `btn` :

```
var btn:Button;
```

Placez votre curseur à n'importe quel endroit sur la ligne de code, puis appuyez sur Ctrl +1. Les options d'importation du composant Button s'affichent. Si vous sélectionnez le composant Spark Button, l'instruction d'importation est créée comme suit :

```
import spark.components.Button;
```

Fractionner la déclaration de variable
Utilisez l'Assistant rapide pour fractionner une variable en deux parties : la déclaration de la variable et l'initialisation de la variable.

L'Assistant rapide apparaît dans les contextes suivants.

Contexte	Exemple

Variables locales dans une fonction	Si vous possédez une fonction, comme suit :
	`public function TestAS()`
	`{`
	`var i:int=10;`
	`}`
	Pour fractionner la variable à l'aide de l'Assistant rapide, placez votre curseur à n'importe quel endroit dans la déclaration de la variable `var i:int=10;` et appuyez sur Ctrl+1. Ensuite, sélectionnez l'option permettant de fractionner la déclaration de la variable.
	La variable est fractionnée comme suit :
	`public function TestAS()`
	`{`
	`var i:int;`
	`i=10;`
	`}`

Plusieurs variables dans une fonction	Si vous possédez une fonction de ce type :
	`public function TestAS()`
	`{`
	`var i:int=10, j:int=20;`
	`}`
	Pour fractionner les variables à l'aide de l'Assistant rapide, placez votre curseur à n'importe quel endroit dans la déclaration de la variable `var i:int=10, j:int=20;` et appuyez sur Ctrl+1. Ensuite, sélectionnez l'option permettant de fractionner la déclaration de la variable.
	Vous pouvez fractionner la variable comme suit :
	`public function TestAS()`
	`{`
	`var i:int, j:int=20;`
	`i=10;`
	`}`
	Vous pouvez encore fractionner la variable j:int=20; en plaçant votre curseur à n'importe quel endroit au sein de la déclaration de la variable, puis en appuyant sur Ctrl+1. Ensuite, sélectionnez l'option permettant de fractionner la déclaration de la variable.
	La variable est fractionnée comme suit :
	`public function TestAS()`
	`{`
	`var i:int, j:int;`
	`j=20;`
	`i=10;`
	`}`

Attribution à une variable

Lorsque vous évaluez une expression ActionScript, si l'expression renvoie une valeur, vous pouvez utiliser l'assistant rapide afin de créer une variable pour cette expression. L'Assistant rapide n'est pas disponible pour les méthodes ne possédant pas de type de retour ou si le type de retour est `void`.

Le nom de la variable qui est créée découle du nom de la fonction ou de l'identifiant dans l'expression. Si le nom dérivé existe, il est incrémenté.

Par exemple, si votre code est le suivant :

```
var i:int;
i;
```

Placez votre curseur après `i` et appelez l'assistant rapide. Sélectionnez ensuite Attribuer l'instruction à une nouvelle variable locale. Une variable `"i2"` est créée. Comme la variable `i` est déjà présente, le nom de la nouvelle variable est incrémenté à `"i2"` comme suit :

```
var i:int;
var i2:int = i;
```

Par exemple, l'attribution à une variable apparaît dans les contextes suivants :

Contexte	**Exemple**

Expression littérale	L'Assistant rapide prend en charge une vaste gamme d'expressions simples et complexes.
	Par exemple, si vous avez l'expression suivante :
	`100+150`
	Placez votre curseur à n'importe quel endroit dans l'expression, puis appuyez sur Ctrl+1 pour obtenir un code ressemblant à celui-ci :
	`var number2:Number = 110 + 500;`

Appel de méthode	Si vous avez le code suivant dans une méthode :
	`var ac:ArrayCollection = new ArrayCollection();`
	`ac.createCursor();`
	Placez votre curseur dans `ac.createCursor();`, puis appuyez sur Ctrl+1. Sélectionnez ensuite l'option Attribuer l'instruction à une nouvelle variable locale. La variable locale est créée comme suit :
	`var createCursor:IViewCursor = ac.createCursor();`

Accès à la propriété	Si vous avez le code suivant dans une propriété :
	`var ac:ArrayCollection = new ArrayCollection();`
	`ac.source;`
	Placez votre curseur à n'importe quel endroit dans `ac.source`, puis appuyez sur Ctrl+1. Sélectionnez ensuite l'option Attribuer l'instruction à une nouvelle variable locale. La variable locale est créée comme suit :
	`var source:Array = ac.source;`

Conversion de la variable locale en champ

Lorsque vous disposez de variables locales dans une fonction, l'Assistant rapide vous permet de créer un champ dans la classe.

Par exemple, si vous avez une variable dans une fonction, comme suit :

```
var i:int = 10;
```

Placez votre curseur n'importe où dans la définition de la variable et appelez l'assistant rapide. Sélectionnez ensuite Convertir la variable locale en champ. Le champ de la classe est créé comme suit :

```
private var i:int;
```

Le nom du nouveau champ est le même que celui de la variable locale, à condition qu'il n'existe aucun conflit dans la portée de la classe. Si le nom existe, le nom du champ est incrémenté et vous pouvez renommer la variable ou la méthode dans votre fichier.

Conversion de la variable locale en paramètre

Si vous disposez d'une fonction qui comporte une variable locale, l'assistant rapide vous permet de convertir la variable locale en paramètre.

Par exemple, si vous disposez d'une fonction qui comporte une variable :

```
public function method():void {
var i:int = 10;
}
```

Placez votre curseur n'importe où dans la définition de la variable et appelez l'assistant rapide. Sélectionnez Convertir la variable locale en paramètre. Le paramètre est créé de la manière suivante :

```
public function method(i:int):void {
    i = 10;
}
```

Une fois le paramètre généré, vous pouvez renommer toutes les références du paramètre à l'aide du mode lié.

Conversion d'une fonction anonyme en fonction nommée

Si vous disposez d'une fonction anonyme, vous pouvez la convertir rapidement en fonction nommée à l'aide de l'assistant rapide.

Par exemple, si vous disposez de la fonction anonyme suivante :

```
public function method1():void;
{
    var foo:Function = function(x:int, y:int, z:int):void;
    {
        trace(x, y, z);
    }
}
```

Placez votre curseur dans `function()`, puis appuyez sur Ctrl+1. Sélectionnez ensuite Convertir une fonction anonyme en fonction nommée. La fonction nommée est créée de la manière suivante :

```
public function method1():void;
    {
        var foo:Function = fooFunction;
    }
    protected function fooFunction(x:int, y:int, z:int):void;
    {
        trace(x, y, z);
    }
```

Une fois la fonction générée, vous pouvez renommer toutes les références de la fonction à l'aide du mode lié.

Affectation d'un paramètre à une nouvelle variable ou une variable existante

Si vous disposez d'une fonction qui contient un paramètre, l'assistant rapide vous permet d'affecter le paramètre à une nouvelle variable d'instance.

Par exemple, si vous disposez d'une fonction qui contient un paramètre `arg` :

```
class A{
    function method(arg:String):void {
    }
}
```

Placez votre curseur dans le paramètre `arg` et appelez l'assistant rapide. Sélectionnez ensuite Affecter le paramètre à une nouvelle variable d'instance. L'instruction devient :

```
class A {
    private var arg:String;
    function method(arg:String):void {
    this.arg = arg;
    }
}
```

Si vous disposez de variables d'instance déjà définies, vous pouvez affecter le paramètre à l'une d'entre elles.

Par exemple, si vous disposez d'une variable myArg définie :

```
class A {
    private var myArg:String;
    function method(arg:String):void {
    }
}
```

Placez votre curseur dans le paramètre `arg` et appelez l'assistant rapide. L'assistant rapide détecte la variable `myArg` et affiche l'option Affecter le paramètre à la variable d'instance myArg. La sélection de cette option modifie l'instruction qui devient :

```
class A {
    private var myArg:String;
    function method(arg:String):void {
        myArg = arg;
    }
}
```

Une fois la variable générée, vous pouvez renommer toutes les références de la variable à l'aide du mode lié.

Création d'une variable locale avec le type de conversion

L'assistant rapide vous permet de créer rapidement une variable locale de type de conversion dans une expression conditionnelle.

Par exemple, si vous disposez d'une instruction `if` :

```
if(myObject is Button) {
}
```

Ou d'une instruction `while` :

```
while(myObject is Button) {
}
```

Placez votre curseur dans l'instruction `if` ou `why` et appelez l'assistant rapide. L'assistant rapide affiche l'option Créer une variable locale avec le type de conversion. La sélection de cette option génère l'expression conditionnelle suivante :

```
if(myObject is Button) {
    var button:Button = myObject as Button;
}
```

Une fois la variable locale générée, vous pouvez renommer toutes les références de la variable à l'aide du mode lié.

Remplacement d'une instruction conditionnelle par une instruction if-else

L'assistant rapide vous permet de remplacer une instruction conditionnelle par une instruction if-else.

Par exemple, si vous disposez de l'instruction conditionnelle suivante :

```
var foo:String = bool?"Yes":"No";
```

Placez votre curseur n'importe où dans l'instruction de condition `bool?"Yes":"No"` et appelez l'assistant rapide. Sélectionnez ensuite Remplacer la condition par if-else. L'instruction est générée de la manière suivante :

```
var foo:String;
    if (bool)
        {
            foo = "Yes";
        }
    else
        {
            foo = "No";
        }
```

Ajout d'instructions else, else-if, finally et catch

Si vous disposez d'une instruction if, vous pouvez utiliser l'assistant rapide pour ajouter des instructions else et else-if.

Par exemple, si vous disposez d'une instruction `if` :

```
var foo:Boolean = false;
if(foo) {
}
```

Placez votre curseur dans l'instruction `if` et appelez l'assistant rapide. L'assistant rapide vous propose des options permettant d'ajouter des instructions `else` et `else-if`. Si vous avez déjà défini une instruction `else`, l'assistant rapide affiche alors une option permettant d'ajouter une instruction `else-if`.

De la même manière, si vous avez déjà défini une instruction `try`, vous pouvez ajouter des instructions `finally` et `catch`.

Génération des méthodes get/set

L'Assistant rapide vous permet de générer des méthodes get/set (fonctions d'accesseur get et set) pour les variables de classes.

Par exemple, si votre code ressemble au suivant :

```
private var result:ResultEvent;
```

Placez votre curseur dans la variable `result` et appelez l'assistant rapide. Sélectionnez ensuite l'option permettant de créer les méthodes get/set destinées au résultat. La boîte de dialogue Générer les méthodes get/set apparaît. Cette boîte de dialogue vous permet de spécifier une nouvelle fonction get/set. Pour plus d'informations sur la génération de méthodes get/set, voir « Génération de fonctions d'accesseur get ou set » à la page 58.

Génération de fonctions

L'Assistant rapide vous permet de générer des fonctions spéciales, telles que labelFunction, iconFunction, etc.

Par exemple, pour créer une fonction labelFunction pour le code suivant :

```
<mx:DataGrid labelFunction="lblfunc" dataTipFunction="tipFunc" />
```

Placez votre curseur dans `"lblfunc"` et appelez l'assistant rapide. Sélectionnez ensuite l'option permettant de créer la fonction labelFunction. Le code de remplacement de la fonction est généré comme suit :

```
protected function lblfunc(item:Object, column:DataGridColumn):String
    {
    // TODO Auto-generated method stub
    }
```

Génération de gestionnaires d'événements à partir du code MXML

L'Assistant rapide vous permet de générer des gestionnaires d'événements avec des noms de gestionnaires personnalisés au sein du code MXML.

Par exemple, pour générer un gestionnaire d'événement en cliquant sur la balise <mx:Button>, entrez le code suivant :

```
<mx:Button click="clickHandler" />
```

Placez ensuite votre curseur dans `"clickHandler"`, appelez l'assistant rapide, puis sélectionnez Générer un gestionnaire d'événement. Le gestionnaire d'événement et le code de remplacement sont générés comme suit :

```
<mx:Button click="clickHandler(event)" />

protected function clickHandler(event:MouseEvent):void
    {
        // TODO Auto-generated method stub
    }
```

Vous pouvez également générer un gestionnaire d'événements pour le code suivant, en appuyant sur Ctrl+1 n'importe où dans `"clickHandler(event)"`.

```
<mx:Button click="clickHandler(event)" />
```

Pour personnaliser le code de remplacement prédéfini généré par Flash Builder, voir « Modèles de code » à la page 41.

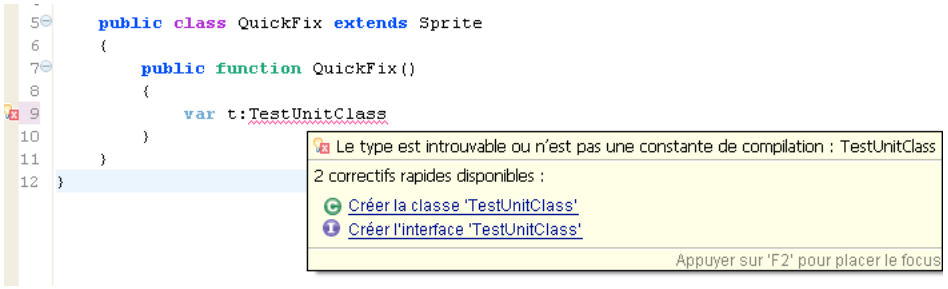
Correctif rapide

La fonction Correctif rapide de Flash Builder identifie les problèmes non résolus du code et vous invite à choisir une option pour les corriger. Cette fonction est prise en charge uniquement pour les fichiers ActionScript ou le code ActionScript des fichiers MXML.

Par exemple, si vous disposez du code suivant :

```
import flash.display.Sprite;
    public class QuickFix extends Sprite
    {
        public function QuickFix()
        {
            var t:TestUnitClass
        }
    }
```

Flash Builder identifie la variable non définie à l'aide d'un indicateur d'erreur en forme d'ampoule situé en regard de l'icône d'erreur . Lorsque vous passez votre curseur au-dessus de la variable non définie, les options de correctif rapide suivantes s'affichent :



Pour utiliser la fonction Correctif rapide, effectuez l'une des opérations suivantes :

- Sélectionnez Assistant/Correctif rapide dans le menu contextuel de l'éditeur et sélectionnez l'option requise.
- Utilisez le raccourci clavier Ctrl+1 (Windows) ou Commande+1 (Mac OS) et sélectionnez l'option requise.

Formatage, navigation et organisation du code

Les éditeurs Flash Builder offrent de nombreuses fonctions pour parcourir le code. Vous pouvez par exemple réduire ou développer des blocs de code, ouvrir les sources de définitions de code ainsi que rechercher et ouvrir des types. La navigation par code offre la possibilité de sélectionner un élément de code (une référence à un composant personnalisé d'un fichier d'application MXML, par exemple) et d'aller à la source de la définition du code, quel que soit son emplacement dans le projet, espace de travail ou chemin.

Des blocs de plusieurs lignes peuvent être réduits et développés afin de faciliter la navigation, l'affichage et la gestion de documents présentant un code complexe. Dans Flash Builder, ces deux opérations sont désignées par les termes de *développement* et de *réduction* d'instructions de code de plusieurs lignes.

Formatage, mise en retrait et commentaires de code

Lorsque vous saisissez du code, Flash Builder insère automatiquement des lignes de code pour améliorer la lisibilité, ajoute une couleur distinctive aux éléments de code et propose de nombreuses commandes de formatage rapide du code que vous saisissez (ajout d'un commentaire de bloc, par exemple).

Pour changer le formatage par défaut, dans la boîte de dialogue Préférences, sélectionnez Flash Builder > Code MXML > Formatage. Vous pouvez modifier l'ordre et le regroupement des attributs.

Lorsque vous collez du code MXML ou ActionScript dans l'éditeur de code, Flash Builder met automatiquement en retrait le code en fonction de vos préférences. Vous pouvez également spécifier une mise en retrait pour un bloc de code sélectionné.

Pour modifier les préférences de mise en retrait, dans la boîte de dialogue Préférences, sélectionnez Flash Builder > Editeurs. Vous pouvez spécifier le type et la taille de retrait.

Définition, réduction et développement des blocs de code

1 Dans l'éditeur, cliquez sur le symbole de réduction (-) ou sur celui de développement (+) dans la marge de gauche.

```
10 public function set cartItems(items:ShoppingCart):Void {
11     _cartItems = items;
12     dg.dataProvider = _cartItems.items;
13 }
```

La réduction d'un bloc de code en masque toutes les lignes à l'exception de la première.

```
10 public function set cartItems(items:ShoppingCart):Void {
14
```

Développez le bloc de code pour en afficher à nouveau le contenu. Placez la souris sur le symbole de développement (+) pour visionner l'ensemble du bloc dans une info-bulle.

```
10 public function set cartItems(items:ShoppingCart):Void {
14     _cartItems = items;
15     dg.dataProvider = _cartItems.items;
16 }
17
18     if (_cartItems.items.length == 0)
```

2 La réduction de code est activée par défaut dans Flash Builder. Pour la désactiver, ouvrez la boîte de dialogue Préférences et sélectionnez Flash Builder > Editeurs. Désélectionnez ensuite l'option Activer la réduction.

Application des préférences de couleurs pour la syntaxe

Les couleurs de la syntaxe peuvent être facilement personnalisées.

❖ Ouvrez la boîte de dialogue Préférences et sélectionnez Flash Builder > Editeurs > Couleurs pour la syntaxe.

Pour plus d'informations, voir « Coloration de la syntaxe » à la page 264.

Les couleurs de police par défaut peuvent également être définies dans les pages Editeurs de texte ainsi que Couleurs et polices des Préférences (voir Préférences > Général > Apparence > Couleurs et polices ; voir également Préférences > Général > Editeurs > Editeurs de texte).

Mise en retrait de blocs de commentaire

L'éditeur met automatiquement en forme les lignes de code en cours de saisie, améliorant ainsi la lisibilité et facilitant la rédaction. Vous pouvez également utiliser la touche de tabulation pour mettre manuellement en retrait certaines lignes de code.

Lorsque vous copiez et collez des blocs de code dans Flash Builder, ce dernier met automatiquement en retrait le code en fonction de vos préférences.

Pour mettre en retrait un bloc de code en une seule opération, vous pouvez utiliser les commandes Décaler vers la droite et Décaler vers la gauche de l'éditeur.

Déplacement d'un bloc de code vers la gauche ou vers la droite

1 Dans l'éditeur, sélectionnez un bloc de code.

2 Cliquez sur Source > Décaler vers la droite ou Source > Décaler vers la gauche.

3 Appuyez sur la touche de tabulation ou sur Maj et la touche de tabulation pour mettre en retrait ou annuler la mise en retrait de blocs de code.

Définition des préférences de la mise en retrait

1 Ouvrez la boîte de dialogue Préférences et sélectionnez Flash Builder > Mise en retrait.

2 Sélectionnez le type de mise en retrait (tabulations ou espaces) et spécifiez la taille de la mise en retrait ou des tabulations.

Ajout de commentaires et de blocs de commentaire

Vous pouvez ajouter ou supprimer des commentaires en utilisant les options dans le menu Source ou en faisant appel aux raccourcis clavier. Vous pouvez ajouter les types de commentaires suivants :

- Commentaires de source pour ActionScript (`//`)
- Commentaires de bloc pour ActionScript (`/* */`)
- Commentaires ASDoc pour ActionScript (`/** */`)
- Commentaires de bloc pour MXML (`<!---->`)
- Blocs CDATA pour MXML (`<![CDATA[ ]]>`)

Les commentaires ajoutés au code ActionScript peuvent être activés ou désactivés.

Paul Robertson, professionnel de la communauté Adobe, a publié sur son blog un article concernant l'utilisation du mode de sélection des blocs.

Activer/désactiver les commentaires du code ActionScript

1 Dans l'éditeur, sélectionnez au moins une ligne de code ActionScript.

2 Appuyez sur Ctrl+Maj+C (Windows) ou Commande+Maj+C (Mac OS) pour ajouter ou supprimer des commentaires de style C.

3 Appuyez sur Ctrl+/ (Windows) ou Commande+/ (Mac OS) pour ajouter ou supprimer des commentaires de style C++.

Ajout de commentaires XML à du code MXML

1 Dans l'éditeur, sélectionnez au moins une ligne de code MXML.

2 Appuyez sur Ctrl+Maj+C (Windows) ou Commande+Maj+C (Mac OS) pour ajouter un commentaire.

Ajout de blocs CDATA à du code MXML

1 Dans l'éditeur, sélectionnez au moins une ligne de code MXML.

2 Appuyez sur Ctrl+Maj+D (Windows) ou Commande+Maj+D (Mac OS) pour ajouter un commentaire.

Navigation et inspection de code

Quelle que soit la complexité des applications, les projets contiennent généralement de nombreuses ressources et lignes de code. Flash Builder fournit plusieurs fonctionnalités qui facilitent la navigation et l'inspection des différents éléments du code.

Ouverture des définitions de code

Flash Builder permet d'ouvrir la source d'une définition de code externe à partir de sa référence dans le code. Si vous créez par exemple un composant MXML personnalisé et l'importez dans l'application MXML, vous pouvez sélectionner la référence au composant MXML et ouvrir le fichier source dans l'éditeur.

Ouverture de la source d'une définition de code

1 Dans l'éditeur, sélectionnez la référence du code.

2 Dans le menu Naviguer, sélectionnez Accéder à la définition.

Vous pouvez utiliser le raccourci clavier F3.

Le fichier source contenant la définition du code s'ouvre dans l'éditeur.

Flash Builder prend également en charge la navigation dans le code par hyperliens.

Ouverture de la source d'une définition de code en utilisant la navigation par hyperliens

1 Localisez la référence du code dans l'éditeur.

2 Appuyez sur la touche Ctrl (Windows) ou la touche Commande (Mac OS) et maintenez-la enfoncée, puis positionnez le curseur de la souris sur la référence du code pour afficher l'hyperlien.

3 Pour atteindre la référence du code, cliquez sur l'hyperlien.

Utilisation de la vue Structure

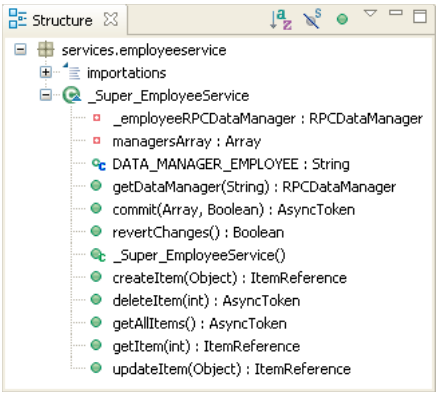
La vue Structure appartient à la perspective Développement Flash (voir « Perspective Développement Flash » à la page 6). Elle est donc disponible au cours de la modification du code. Cette vue permet d'examiner et de parcourir plus facilement la structure des documents MXML, ActionScript et CSS.

Elle présente trois modes : le mode Classe, le mode MXML et le mode CSS. En mode Classe, cette vue affiche la structure du code (classes, variables des membres, fonctions, etc.). En mode MXML, elle affiche la structure MXML (balises, composants, contrôles, etc.). En mode CSS, elle affiche les sélecteurs CSS et les propriétés imbriquées qu'ils contiennent.

La sélection d'un élément dans la vue Structure permet de le localiser et de le mettre en évidence dans l'éditeur, facilitant ainsi considérablement la navigation à l'intérieur du code.

Vue Structure en mode Classe

Lorsque vous éditez un document ActionScript (ou un code ActionScript contenu dans un document MXML), la vue Structure affiche la structure du code. Cette structure comporte les instructions d'importation, les packages, les classes, les interfaces, les variables non contenues dans les fonctions et les fonctions. En revanche, elle ne contient aucune métadonnée, aucun commentaire, aucune déclaration d'espace de noms et aucun contenu de fonction.



Les nœuds et les éléments de l'arborescence de la vue Structure représentent à la fois les différents types d'éléments de langage et leur visibilité. Les icônes rouges signalent par exemple des éléments privés ; les icônes vertes signalent des éléments publics ; les icônes jaunes indiquent que l'élément n'est ni privé, ni public.

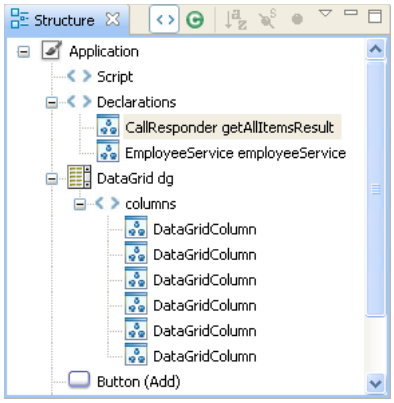
Barre d'outils de la vue Structure en mode Classe

En mode Classe, la barre d'outils de la vue Structure contient les commandes de tri et de filtrage, comme le montre l'exemple suivant :

Vue Structure en mode MXML

La vue Structure d'un document MXML en cours d'édition contenant du code MXML et du code ActionScript présente les modes Classe et MXML.

En mode MXML, chaque élément de la vue Structure représente une balise MXML. Les types de balise affichés sont les suivants : les composants, les contrôles, les balises non visuelles (`WebService` ou `State`), les propriétés des composants exprimées sous forme de balises enfant (contraintes de présentation, par exemple) et les balises de compilateur (`Model`, `Array` et `Script`).



Le mode MXML de la vue Structure n'affiche aucun commentaire, aucune règle ou propriété CSS et aucune propriété de composant exprimée sous forme d'attribut (par opposition aux balises enfant qui sont affichées).

Barre d'outils de la vue Structure en mode MXML

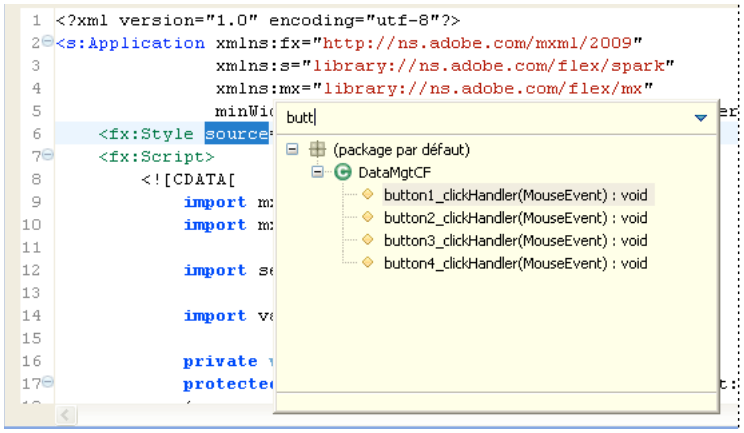
En mode MXML, la barre d'outils de la vue Structure contient des commandes supplémentaires permettant de basculer de la vue MXML à la vue des classes.

Pour basculer entre les deux vues, sélectionnez Afficher la vue MXML ou Afficher la vue des classes dans le menu de la barre d'outils.

Utilisation de la vue Structure rapide de l'éditeur

Vous pouvez ouvrir la vue Structure rapide à partir des éditeurs ActionScript et MXML pour afficher la vue Structure en mode Classe. La vue Structure rapide apparaît dans une fenêtre contextuelle à l'intérieur de l'éditeur et ne se présente donc pas sous forme de vue distincte. Elle permet de parcourir et d'examiner rapidement le code.

Son contenu est identique à celui du mode Classe, à ceci près que la Structure rapide présente une zone de saisie de texte permettant de filtrer les éléments affichés. Vous pouvez par exemple entrer le nom d'un élément dans la vue Structure rapide afin de limiter l'affichage aux éléments contenant les caractères saisis.



La vue Structure rapide ne contient pas les commandes permettant d'effectuer un tri alphabétique des éléments ou de les masquer.

De même que dans la vue Structure, vous pouvez sélectionner un élément afin de le localiser et de le mettre en évidence dans l'éditeur.

Ouverture de la vue Structure rapide

- Ouvrez un document ActionScript ou MXML dans l'éditeur. Dans le menu Naviguer, sélectionnez Structure rapide.

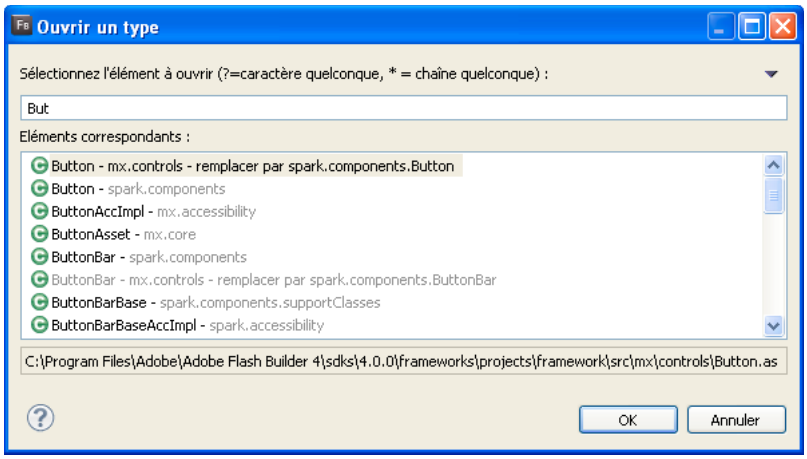
  Vous pouvez également utiliser le raccourci clavier Ctrl+O.

Fermeture de la vue Structure rapide

❖ L'accès à tout emplacement situé en dehors de la vue Structure rapide conduit à la fermeture de cette dernière. Elle peut également être fermée en appuyant sur la touche Echap.

Navigation et affichage des classes

La boîte de dialogue Ouvrir un type permet de parcourir toutes les classes disponibles (y compris les classes de la structure Flex) du projet. Dans la boîte de dialogue Ouvrir un type, sélectionnez une classe pour afficher l'implémentation.



Boîte de dialogue Ouvrir un type

La boîte de dialogue Ouvrir un type permet également de sélectionner des classes comme classe de base pour une nouvelle classe ActionScript ou un nouveau composant MXML.

La boîte de dialogue Ouvrir un type vous permet de filtrer les classes affichées en fonction du texte et des caractères génériques que vous spécifiez. La boîte de dialogue fait appel à différentes couleurs pour signaler les types recommandés et les types exclus. Les types recommandés sont affichés en gris. Les types exclus sont affichés en marron.

Les *types recommandés* sont les classes disponibles dans l'espace de noms par défaut d'un projet. Dans certains contextes par exemple, seuls les composants Spark sont autorisés. D'autres contextes autorisent les composants Spark et les composants Halo.

Les *types exclus* sont les classes qui ne sont pas disponibles dans l'espace de noms par défaut pour un projet.

Ouverture de la boîte de dialogue Ouvrir un type

- (Parcourir les classes) Pour parcourir les classes et visionner leur implémentation :
  1. Dans le menu Flash Builder, sélectionnez Navigation > Ouvrir un type.
  2. (Facultatif) Saisissez du texte ou sélectionnez des filtres pour modifier les classes visibles dans la liste.
  3. Sélectionnez une classe pour en afficher le code source.

     Vous ne pouvez pas modifier le code source des classes dans la structure Flex.
- (Nouvelles classes ActionScript) Pour sélectionner une classe de base pour une nouvelle classe ActionScript :
  1. Sélectionnez Fichier > Nouveau > Classe ActionScript.
  2. En regard du champ Superclasse, cliquez sur Parcourir.

3 (Facultatif) Saisissez du texte ou sélectionnez des filtres pour modifier les classes visibles dans la liste.

4 Sélectionnez une classe de base dans la liste.

- (Nouveaux composants MXML) Pour sélectionner un composant de base pour un nouveau composant MXML :

1 Sélectionnez Fichier > Nouveau > Composant MXML.

2 Dans la liste des projets de l'espace de travail, sélectionnez un projet pour un nouveau composant MXML et spécifiez un nom pour le fichier.

Les composants de base disponibles varient en fonction des espaces de noms configurés pour un projet.

3 En regard du champ Basé sur, cliquez sur Parcourir.

Remarque : *effacez ou modifiez la classe de base par défaut répertoriée dans le champ Basé sur pour élargir vos choix.*

4 (Facultatif) Saisissez du texte ou sélectionnez des filtres pour modifier les classes visibles dans la liste.

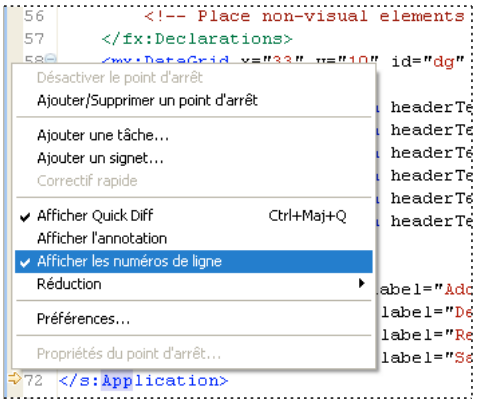
5 Sélectionnez un composant de base dans la liste.

Affichage des numéros de ligne

Vous pouvez ajouter des numéros de ligne dans l'éditeur afin de faciliter la lecture et la consultation du code.

❖ Dans le menu contextuel de la marge de l'éditeur, sélectionnez Afficher les numéros de ligne.

La marge de l'éditeur se situe entre la barre de repère et l'éditeur.



Utilisation de marqueurs

Les marqueurs sont des raccourcis vers les lignes de code d'un document, vers un document ou vers un dossier. Ils représentent des tâches, des signets et des problèmes. Ils sont affichés et peuvent être gérés. La sélection de marqueurs provoque l'ouverture du document associé dans l'éditeur et met éventuellement en évidence la ligne concernée du code.

Pour mettre à jour les marqueurs de problèmes dans Flash Builder, vous devez enregistrer le fichier. Seuls les fichiers auxquels l'application fait référence sont vérifiés. La syntaxe d'une classe isolée qui n'est pas utilisée dans le code n'est pas contrôlée.

Le workbench génère automatiquement les marqueurs de tâches et de problèmes suivants. Vous pouvez ajouter des tâches et des signets manuellement.

Tâches : les marqueurs de tâches représentent un élément de travail. Les éléments de travail sont générés automatiquement par le workbench. Vous pouvez ajouter manuellement une tâche à une ligne de code du document

ou au document même. Par exemple, afin de penser à définir une propriété pour un composant, vous pouvez créer une tâche nommée « Définir les propriétés de l'aspect ». Vous pouvez également ajouter des tâches générales qui ne s'appliquent pas directement aux ressources (« Créer un composant personnalisé pour l'invite de connexion de l'employé », par exemple). La vue Tâches permet de gérer tous les marqueurs de tâches. Pour plus d'informations, voir « Ajout de tâches » à la page 35.

Erreurs : les marqueurs d'erreurs sont générés par le compilateur et signalent différents types d'états non valides. Par exemple, les erreurs de syntaxe et les avertissements générés par le compilateur sont affichés comme marqueurs dans la vue Erreurs. Pour plus d'informations, voir « Filtrage des vues Tâches et Erreurs » à la page 12.

Signets : vous pouvez ajouter manuellement des signets à une ligne de code ou à une ressource (dossier ou document). Vous utilisez des signets pour leur côté pratique, afin de suivre les éléments de vos projets et de naviguer facilement jusqu'à eux. Ils sont gérés dans la vue Signets. Pour plus d'informations, voir « Ajout et suppression de signets » à la page 36.

***Remarque :** les vues Tâches et Signets ne sont pas affichées par défaut dans la perspective Développement Flash. Pour plus d'informations sur l'ajout de ces vues, voir « Utilisation de vues » à la page 11.*

Navigation dans les marqueurs

Les *marqueurs* sont à la fois des descriptions et des liens vers des éléments des ressources d'un projet. Les marqueurs sont générés automatiquement par l'ordinateur pour indiquer des problèmes au niveau du code ou sont ajoutés manuellement pour vous aider à suivre les tâches ou les bouts de code. Les marqueurs sont affichés et gérés dans les vues associées. Les marqueurs d'un projet sont facilement repérables dans les vues Signets, Erreurs et Tâches, à partir desquelles ils sont accessibles.

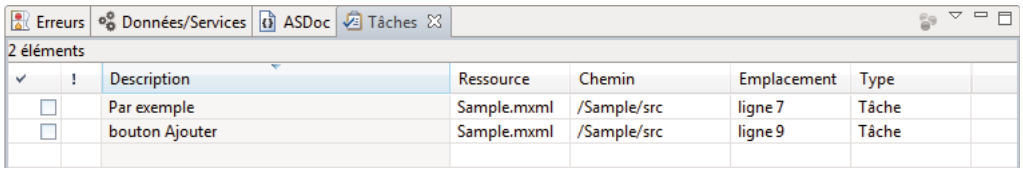
Accès à l'emplacement d'un marqueur

❖ Sélectionnez un marqueur dans les vues Signets, Erreurs ou Tâches.

Le fichier contenant le marqueur est localisé et affiché dans l'éditeur. Si l'emplacement du marqueur est une ligne de code, cette dernière est mise en surbrillance.

Ajout de tâches

Les tâches sont des éléments de l'espace de travail créés automatiquement ou manuellement. Toutes les tâches sont affichées et gérées dans la vue Tâches (Fenêtre > Autres vues > Généralités > Tâches), comme illustré dans l'exemple suivant :



Ajout d'une tâche à une ligne de code ou à une ressource

1 Ouvrez un fichier dans l'éditeur, puis repérez et sélectionnez la ligne de code à laquelle vous souhaitez ajouter une tâche. Vous pouvez également sélectionner une ressource dans l'Explorateur de packages.

2 Dans la vue Tâches, cliquez sur le bouton Ajouter une tâche dans la barre d'outils.

3 Entrez le nom de la tâche et sélectionnez une priorité (Elevée, Normale, Faible), puis cliquez sur OK.

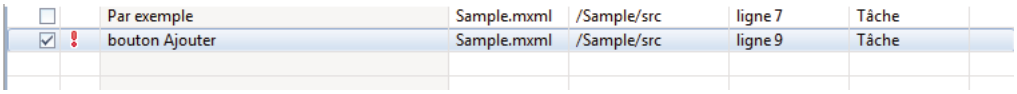
***Remarque :** comme vous pouvez le constater dans l'Explorateur de packages, la ressource ne signale pas qu'elle contient un marqueur. Vous pouvez visionner et gérer tous les marqueurs de tâches dans la vue Tâches.*

Achèvement et suppression de tâches

Une fois achevée, une tâche peut être marquée comme ayant été terminée et peut éventuellement être supprimée de la vue Tâches.

Définition d'une tâche comme étant terminée

❖ Dans la vue Tâches, sélectionnez une tâche de la colonne, comme l'illustre l'exemple suivant.



Suppression d'une tâche

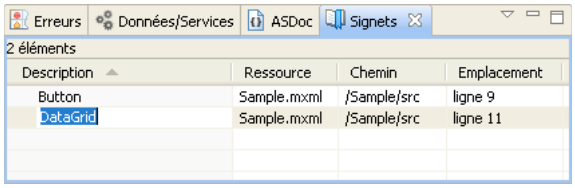
❖ Dans la vue Tâches, accédez au menu contextuel d'une tâche et sélectionnez Supprimer.

Suppression de toutes les tâches terminées

❖ Dans la vue Tâches, accédez au menu contextuel et sélectionnez Supprimer les tâches terminées.

Ajout et suppression de signets

Les signets permettent de repérer et rechercher facilement des éléments au sein des projets. Tous les signets sont affichés et gérés dans la vue Signets (Fenêtre > Autres vues > Généralités > Signets), comme illustré dans l'exemple suivant :



Ajout d'un signet à une ligne de code ou à une ressource

1 Ouvrez un fichier dans l'éditeur, puis localisez et sélectionnez la ligne de code à laquelle vous souhaitez ajouter un signet.

2 Dans le menu principal, sélectionnez Editer > Ajouter un signet.

3 Saisissez le nom du signet et cliquez sur OK.

Une icône de signet ( ) apparaît en regard de la ligne de code.

Remarque : *comme vous pouvez le constater dans l'Explorateur de packages Flex, la ressource ne signale pas qu'elle contient un marqueur. Vous pouvez visionner et gérer tous les signets dans la vue Signets.*

Suppression d'un signet

1 Dans la vue Signets, sélectionnez le signet à supprimer.

2 Cliquez sur le signet avec le bouton droit de la souris (Windows) ou en appuyant sur la touche Ctrl (Mac OS), puis sélectionnez Supprimer.

Organisation des instructions d'importation

Vous pouvez ajouter, trier et supprimer des instructions d'importation inutilisées dans les blocs de script MXML et ActionScript à l'aide de la fonction Organiser les importations.

Pour utiliser la fonction Organiser les importations avec un document ActionScript ou MXML qui contient les instructions d'importation ouvertes dans l'éditeur, effectuez l'une des opérations suivantes :

- Sélectionnez Organiser les importations dans le menu Source.
- Utilisez le raccourci clavier suivant : Ctrl+Maj+ O (Windows) ou Commande+Maj+O (Mac OS).
- Placez le curseur sur n'importe quelle instruction d'importation, puis appuyez sur Ctrl+1. Sélectionnez ensuite l'option d'organisation des importations.

Ajout d'instructions d'importation manquantes

Si votre script ActionScript ou bloc de script MXML comporte des variables non définies, vous pouvez ajouter toutes les instructions d'importation manquantes en une fois à l'aide de la fonction Organiser les importations.

Résolution d'importations ambiguës

La fonction Organiser les importations des éditeurs MXML et ActionScript entraîne l'importation automatique dans le document des packages dans lesquels les classes se situent.

Si un type ActionScript se trouve dans deux packages, vous pouvez choisir l'instruction d'importation du package requis. Par exemple, `Button` est présent dans les packages `spark.components` et `mx.controls`.

Si vous disposez de plusieurs instances d'instructions d'importation ambiguës, les importations non résolues s'affichent afin que vous puissiez les résoudre l'une après l'autre.

Suppression d'instructions d'importation inutilisées

Par défaut, Flash Builder place toutes les instructions d'importation en haut des documents ActionScript ou des blocs de script des documents MXML.

Pour supprimer des instructions d'importation non référencées dans votre document, gardez le document qui contient les instructions d'importation ouvert dans l'éditeur, puis utilisez la fonction Organiser les importations.

Tri des instructions d'importation

Vous pouvez rapidement trier les instructions d'importation dans votre bloc ActionScript ou MXML à l'aide de la fonction Organiser les importations.

Par défaut, Flash Builder trie les instructions d'importation par ordre alphabétique. Pour modifier l'ordre par défaut dans lequel Flash Builder ajoute les instructions d'importation, utilisez la boîte de dialogue Préférences. Pour ce faire, ouvrez la boîte de dialogue Préférences et sélectionnez Flash Builder > Editeurs > Code ActionScript > Organiser les importations. Pour plus d'informations, voir « Code ActionScript » à la page 262.

Instructions d'importation non prises en compte par l'importation automatique en cas de copier-coller du code ActionScript

Si vous copiez le code ActionScript d'un document ActionScript et le collez dans un autre document ActionScript, les instructions d'importation manquantes sont automatiquement ajoutées. Les instructions d'importation manquantes sont ajoutées au niveau du package ou du fichier, en fonction de l'emplacement où vous avez copié le code.

Remarque : *cette fonction est prise en charge uniquement pour le code ActionScript. Si vous copiez le code ActionScript à partir d'un fichier en dehors de Flash Builder ou d'un fichier qui n'est pas de type ActionScript, les instructions d'importation manquantes ne sont pas ajoutées.*

Consolidation des instructions d'importation des blocs de script MXML

Si vous disposez d'un document comportant plusieurs blocs de script MXML et plusieurs instructions d'importation définies pour chaque bloc de script, Flash Builder vous permet de consolider toutes les instructions d'importation.

L'option Consolider les instructions d'importation des blocs de script MXML de la boîte de dialogue de préférences d'organisation des importations est sélectionnée par défaut. Les instructions d'importation sont consolidées, triées et ajoutées une seule fois en haut du premier bloc de script.

Consolidation des instructions d'importation du même package à l'aide de caractères génériques
Si vous disposez d'un package comportant plusieurs instructions d'importation, vous pouvez utiliser des caractères génériques dans votre instruction d'importation au lieu de répéter les instructions plusieurs fois.

Vous pouvez spécifier le nombre d'instructions d'importation autorisées par type dans le même package avant que le caractère générique `<package>.*` ne soit utilisé. Par exemple, si vous disposez d'un package `flash.events` comportant plusieurs instructions d'importation, Flash Builder consolide toutes les instructions d'importation de la manière suivante :

```
import flash.events.*;
```

Par défaut, Flash Builder applique le caractère générique `<package>.*` si vous utilisez plus de 99 instructions d'importation du même package. Vous pouvez modifier la valeur par défaut à l'aide de la boîte de dialogue de préférences d'organisation des importations. Pour plus d'informations, voir « Code ActionScript » à la page 262.

Inspection, analyse et correction du code

Vérification et mise en surbrillance des erreurs de syntaxe

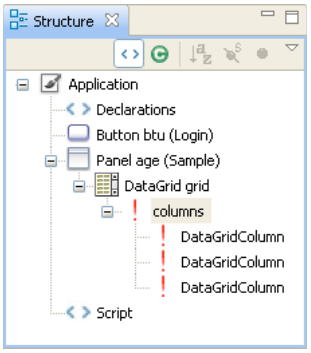
Flash Builder analyse le code ActionScript ou MXML à mesure que vous le saisissez et identifie les erreurs de syntaxe de codage et toutes les autres erreurs par défaut.

En fonction de la nature et de la gravité des erreurs, il est possible que l'application ne soit exécutée correctement qu'après correction de ces erreurs.

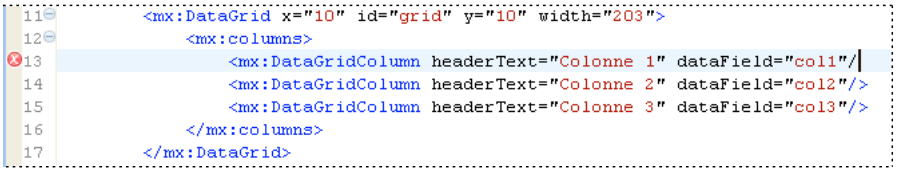
Vérification des erreurs de syntaxe dans les projets Flex

Lorsqu'un projet Flex présente des erreurs de syntaxe de code, vous en êtes informé des manières suivantes au moment de l'enregistrement du fichier :

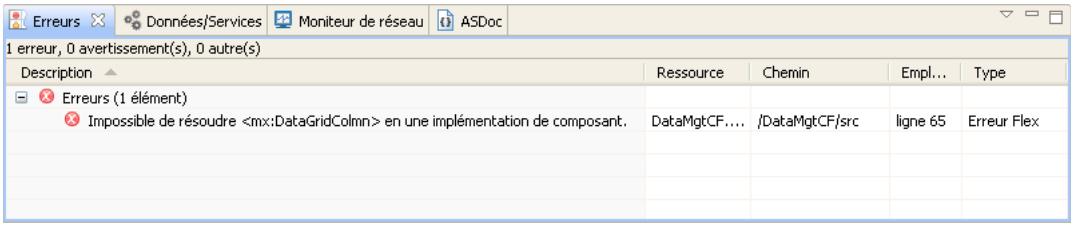
- Un indicateur d'erreur apparaît en regard de la ligne de code, comme dans l'exemple suivant.



- La vue Structure signale les erreurs par un point d'exclamation affiché dans les lignes de code concernées, comme dans l'exemple suivant.



- La vue Erreurs répertorie les erreurs en leur attribuant un symbole et en affichant le message correspondant. Cliquez deux fois sur le message d'erreur pour localiser et mettre en évidence la ligne de code dans l'éditeur, comme dans l'exemple suivant.



Mise en surbrillance des erreurs de référence non identifiées

Flash Builder génère des annotations d'erreur pour les identificateurs non définis dans votre code.

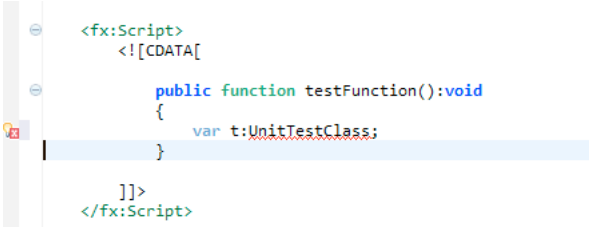
L'icône en regard de la ligne de code dans l'éditeur indique une annotation d'erreur.

La mise en surbrillance des erreurs de référence non identifiées vous permet également d'identifier les endroits dans le code où vous pouvez générer du code de remplacement pour une méthode, une variable ou une classe non définie. Pour plus d'informations, voir « Génération à partir de l'utilisation » à la page 55.

Mise en surbrillance d'erreurs en direct dans les projets ActionScript

Lorsque vous saisissez du code ActionScript dans un fichier ActionScript ou dans le bloc ActionScript d'un fichier MXML, Flash Builder détecte les erreurs à mesure de la saisie. Vous pouvez identifier ces erreurs rapidement avant d'enregistrer le fichier ou de compiler le code.

Un indicateur d'erreur en forme d'ampoule apparaît en regard de l'icône pour vous indiquer que vous pouvez utiliser Correctif rapide pour corriger le problème.



Pour plus d'informations sur l'utilisation de la fonction Correctif rapide, voir « Correctif rapide » à la page 26.

L'icône en regard de la ligne de code dans l'éditeur indique que l'option Correctif rapide n'est pas disponible pour cette erreur.

Pour désactiver la mise en surbrillance d'erreurs en direct, sélectionnez Flash Builder > Editeurs dans la boîte de dialogue Préférences, puis désélectionnez Signaler les erreurs au fur et à mesure de la saisie. Les erreurs de code sont reportées uniquement dans la vue Problèmes après l'enregistrement du fichier.

Vous pouvez également corriger les erreurs à l'aide de l'option Correctif rapide dans la vue Problèmes. Pour ce faire, cliquez avec le bouton droit de la souris sur l'erreur dans la vue Problèmes et sélectionnez Correctif rapide.

Recherche de références et restructuration du code

Flash Builder comporte des fonctions de recherche avancées plus puissantes que les fonctions de recherche/remplacement. Afin de faciliter la compréhension de la manière dont les fonctions, variables et autres identifiants sont utilisés, Flash Builder permet de rechercher et de repérer les références ou les déclarations vers les identifiants dans les fichiers ActionScript ou MXML, les projets ou les espaces de travail. Vous pouvez utiliser la restructuration pour renommer les identifiants suivants dans votre code et mettre à jour toutes les références à ces objets :

- Variables
- Fonctions
- Types (interface, classe)
- Accesseurs (get/set)
- Attributs
- Métadonnées en MXML (effets, événements, styles)

Marquage des références

1. En mode Source, cliquez sur le bouton Marquer les occurrences de la barre d'outils.
2. Dans l'éditeur, cliquez sur un identifiant. Toutes les instances sont marquées en fonction des paramètres des Préférences.

Pour modifier l'apparence des références marquées, dans la boîte de dialogue Préférences, sélectionnez Général > Editeurs > Editeurs de texte > Annotations. Pour plus d'informations sur les marqueurs, voir « Utilisation de marqueurs » à la page 34.

Recherche de toutes les références et de toutes les déclarations

1. En mode Source, cliquez sur un identifiant dans l'éditeur.
2. Dans le menu principal, sélectionnez Rechercher > Références ou Rechercher > Déclarations. Sélectionnez ensuite Fichier, Projet ou Espace de travail. Les résultats apparaissent dans la vue Recherche.

Restructuration du code

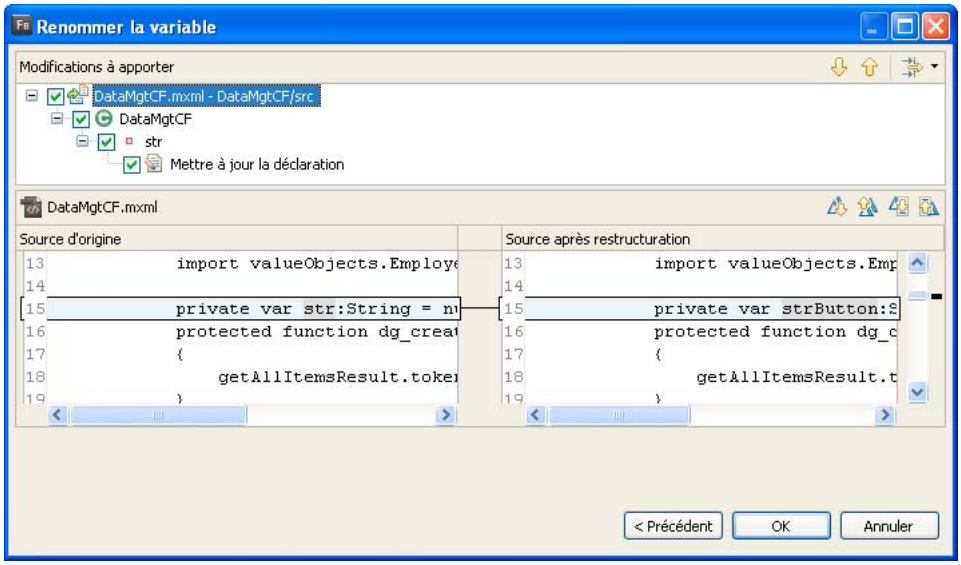
1. En mode Source, cliquez sur un identifiant dans l'éditeur.
2. Dans le menu principal, sélectionnez Source > Restructurer > Renommer.
3. Entrez un nouveau nom.

   Flash Builder vérifie que les conditions préalables à la modification du nom sont réunies et vous invite à confirmer les problèmes avant de procéder à la modification du nom. Les conditions préalables à la modification du nom sont les suivantes :

   - Les références situées dans des fichiers en lecture seule ne peuvent pas être renommées.
   - Tous les fichiers doivent avoir été sauvegardés.

- Les projets présentant des erreurs de génération provoquent l'affichage d'un avertissement.
- Le nouveau nom doit se trouver dans l'étendue, déterminée par le type de l'élément et son emplacement. Les erreurs d'occultation de noms sont également signalées.
- Le nouveau nom doit être un identifiant valide.
- La référence définie dans un fichier SWC doit comprendre une connexion de la source.

4 Pour vérifier la modification apportée, cliquez sur Aperçu pour visionner la source d'origine et la source restructurée, ou cliquez sur OK pour poursuivre la modification du code.



Restructuration dans les fichiers CSS

Lorsque vos fichiers CSS font référence à des fichiers ActionScript ou MXML et que vous renommez ou déplacez les fichiers ActionScript ou MXML, Flash Builder actualise automatiquement les fichiers CSS avec des références au nouveau nom ou au nouvel emplacement. Vous pouvez vérifier les modifications dans la boîte de dialogue Aperçu, puis cliquer sur OK pour appliquer la modification à votre code.

Modification avancée du code

Modèles de code

Les modèles de code permettent d'accélérer l'activité de codage grâce à l'insertion automatique de motifs de codage employés fréquemment.

Flash Builder inclut différents modèles de code prédéfinis. Vous pouvez également définir des modèles de code supplémentaires pour les motifs de code couramment utilisés. Pour voir tous les modèles de code disponibles, ouvrez la boîte de dialogue Préférences et sélectionnez Flash Builder > Editeurs > Modèles de code.

Paul Robertson, professionnel de la communauté Adobe, a publié sur son blog un article sur l'utilisation de modèles de code.

Modèles de code MXML, ActionScript et CSS

Les modèles de code ActionScript, CSS et MXML sont basés sur le contexte et peuvent être appelés en appuyant sur Ctrl+Espace. Vous pouvez utiliser les modèles prédéfinis fournis avec Flash Builder ou créer vos propres modèles.

Insertion des modèles de code

Pour insérer un modèle de code dans l'éditeur de code, saisissez le nom du modèle dans l'éditeur de code et appuyez sur Ctrl+Espace.

Par exemple, lors de l'écriture de code ActionScript, supposons que vous utilisez à plusieurs reprises la boucle `for`. Vous pouvez dans ce cas définir un modèle de code pour la boucle `for` de la manière suivante :

```
for (var i:int = 0; i < array.length; i++) { }
```

Lorsque vous utilisez un modèle de code, il n'est pas nécessaire de saisir le code complet pour la boucle `for`. A la place, dans la classe ActionScript, tapez `for` et appuyez sur Ctrl+Espace. Une option de modèle permettant de créer la boucle `for` s'affiche. Lorsque vous sélectionnez le modèle de code, le code défini dans le modèle est inséré.

Les modèles peuvent également contenir des variables de modèle. Les variables de modèle sont définies entre les symboles `${}`. Elles sont résolues en fonction de la définition de variable correspondante dans l'éditeur.

Par exemple, si vous définissez un modèle de code pour la boucle `for` de la manière suivante :

```
for (var ${index}:int = 0; ${index} < ${array}.length; ${index}++) { ${cursor} }
```

Puis, si vous appelez le modèle de code après avoir défini une variable `myArr` de la manière suivante :

```
{
    var myArr:ArrayCollection = null;
}
```

Dans ce cas, `${array}` dans le modèle de code est résolu comme `myArr` et le code résultant présente l'aspect suivant :

```
{
    var myArr:ArrayCollection = null;
    for (var ${index}:int = 0; ${index} < myArr.length; ${index}++) { ${cursor} }
}
```

Création et modification de modèles de code

1. Ouvrez la boîte de dialogue Préférences et sélectionnez Flash Builder > Editeurs > Modèles de code.
2. Les modèles de code sont classés dans les catégories ActionScript, MXML et CSS. Chaque catégorie contient un ensemble de modèles de code prédéfinis. Vous pouvez modifier un modèle existant ou en ajouter un nouveau.
3. Pour ajouter un nouveau modèle, sélectionnez la catégorie du modèle de code et cliquez sur Ajouter. Dans la boîte de dialogue Nouveau modèle, saisissez un nom et une description succincte pour le modèle de code. Spécifiez ensuite un contexte dans lequel le modèle de code doit être appelé.

   Vous pouvez spécifier des contextes pour les modèles de code ActionScript et MXML.

   Vous disposez des contextes ActionScript suivants :

   - **ActionScript** : insère le modèle de code à un endroit quelconque du document ActionScript.
   - **ActionScript statement** : insère le modèle de code dans les fonctions et dans les éléments d'une classe.
   - **ActionScript members** : insère le modèle de code uniquement dans les éléments d'une classe.
   - **ActionScript Package Scope** : insère le modèle de code dans un package, de la manière suivante :

```
Package
    {
    /* insert code template*/
    }
```

Vous disposez des contextes MXML suivants :

- **MXML** : insère le modèle de code à un endroit quelconque du document MXML.
- **MX Component** : insère le modèle de code dans les composants MX disponibles pour le SDK Flex 3.
- **Spark Components** : insère le modèle de code dans les composants Spark disponibles pour le SDK Flex 4 ou version supérieure.
- **MXML attributes** : insère le modèle de code pour les attributs MXML dans les composants MX et Spark.

4 Saisissez le code correspondant au modèle dans la section Motif. Pour insérer des variables dans le code, cliquez sur Insérer une variable et sélectionnez une variable prédéfinie dans la liste. Les variables sont liées à la catégorie du modèle.

Les modèles ActionScript contiennent des variables prédéfinies, notamment `array`, `enclosing_method`, `enclosing_package`, `enclosing_type`, `field`, `local_var` et `var`. Les modèles MXML contiennent des variables prédéfinies, notamment `fx`, `mx`, `s` et `tag`.

5 Si vous ne souhaitez pas que Flash Builder insère automatiquement le modèle de code dans votre code, désactivez l'option Insertion automatique dans le code.

6 Pour personnaliser un modèle de code existant, sélectionnez le modèle et cliquez sur Modifier. Après avoir modifié le modèle, cliquez sur OK.

Pour plus d'informations sur la personnalisation des modèles de fichier et des variables de modèle, voir « Personnalisation des modèles de fichier » à la page 48 et « Variables des modèles » à la page 50.

Vous pouvez à tout moment supprimer le modèle personnalisé et rétablir le modèle de code prédéfini en cliquant sur Rétablir la valeur par défaut.

Par défaut, vous pouvez appeler tous les modèles prédéfinis à l'aide de l'assistant de contenu. Si toutefois vous souhaitez exclure un modèle spécifique des options de l'assistant de contenu, désactivez ce modèle dans la section Modèles existants.

Vous pouvez également importer et exporter des modèles de code. Vous pouvez sélectionner un ou plusieurs modèles et les exporter. Les modèles sont exportés sous forme de fichier XML.


Paul Robertson, professionnel de la communauté Adobe, a publié sur son blog un article sur le partage de modèles de code.

Modèles de code Flash Builder

Flash Builder peut générer automatiquement un code prédéfini dans les scénarios suivants :

- « Génération de gestionnaires d'événement » à la page 185
- « Génération de fonctions d'accesseur get ou set » à la page 58
- « Génération à partir de l'utilisation » à la page 55 (code d'élément de remplacement de l'emplacement réservé pour une méthode non définie)
- « Remplacement ou implémentation de méthodes » à la page 46

Vous pouvez personnaliser le modèle de code prédéfini généré par Flash Builder.

Personnalisation du modèle de code

1 Ouvrez la boîte de dialogue Préférences, puis sélectionnez Flash Builder > Editeurs > Modèles de code > Flash Builder.

2 Sélectionnez le nom du modèle de code que vous souhaitez personnaliser, puis cliquez sur Editer. Par exemple, pour personnaliser le code généré lorsque vous générez un gestionnaire d'événement, sélectionnez le modèle de gestionnaire d'événement, puis cliquez sur Editer.

3 Vous pouvez personnaliser le nom du modèle, la description et le motif de code.

4 Pour insérer une variable dans le code, cliquez sur Insérer une variable, puis sélectionnez la variable. Pour plus d'informations à propos des variables de code disponibles, voir « Utilisation de variables de code » à la page 44.

5 Vous pouvez à tout moment annuler les modifications en cliquant sur Rétablir la valeur par défaut.

6 Vous pouvez aussi importer et exporter le modèle de code. Vous pouvez sélectionner un ou plusieurs modèles et les exporter. Les modèles sont exportés sous forme de fichier XML.

Utilisation de variables de code

Variables de code pour les gestionnaires d'événement

Variable	Description	Exemple
${component_id}	Correspond à l'ID unique du composant.	Si l'ID du composant du bouton est `test`, le gestionnaire d'événement généré est `test_clickHandler`. Si vous n'avez pas spécifié de valeurs d'ID, les valeurs générées automatiquement sont composant1, composant2, etc.
${component_name}	Correspond au nom de la balise.	`${namespace} ${modifiers}function ${:method_name('${component_id}_${event_name}Handler')}(${event}:${event_type}):${return_type} { // TODO Auto-generated method stub ${cursor} }`
${event_name}	Indique le nom de l'événement.	`clickEvent`, `onHover`
${event_type}	Correspond au type de gestionnaire d'événement.	Flash Builder désigne un type d'événement par défaut pour chaque composant de l'interface utilisateur. Lors de la génération d'un événement de clic pour un composant de bouton, un gestionnaire d'événement de type MouseEvent est généré comme suit : `button1_clickHandler(event:MouseEvent)`
${modifiers}	Indique les modificateurs de la fonction générée.	`static`
${method_name}	Correspond au nom du gestionnaire d'événement.	Pour un événement de clic du composant Bouton, le nom du gestionnaire d'événement peut être button1_clickHandler
${namespace}	Définit la valeur d'espace de noms pour la fonction générée.	La valeur d'espace de noms peut être l'une des suivantes : • `protected` • `public` • `private` La valeur par défaut est `protected`.

Exemple d'utilisations de variables de code pour les fonctions du gestionnaire d'événement :

```
${namespace} ${modifiers}function
${:method_name('${component_id}_${event_name}Handler')}
${event}:${event_type}):${return_type}
    {
        // TODO Auto-generated method stub
    ${cursor}
    }
```

Variables de code pour les fonctions d'accesseur get et set

Variable	Description	Exemple
${metadata}	Spécifie les balises de métadonnées qui sont générées	Génération de fonctions d'accesseur get and set pour une variable Bindable

${asdoc}	Spécifie les ASDoc qui sont générés pour les fonctions d'accesseur get and set	`${metadata}`
		`${asdoc}${namespace}`
		`${modifiers}function get`
		`${method_name}()${return_type}`
		`{`
		`return ${property};`
		`}`

${return_type}	Correspond au type de variable.
	Si le type de variable n'est pas spécifié, la fonction d'accesseur get générée n'a pas de type de retour.
		`${metadata}`
		`${asdoc}${namespace}`
		`${modifiers}function get`
		`${method_name}()${return_type}`
		`{`
		`return ${property};`
		`}`

${property}}	Correspond au nom de la propriété dans la boîte de dialogue des méthodes get/set.	Pour une variable `var i:int`, i est résolu en `_i`
${argument_type}	Correspond au type de données de la fonction set générée.	

Exemple d'utilisation de variables de code pour les fonctions d'accesseur get et set :

```
${asdoc}
${namespace} ${modifiers}function set ${method_name}(value${argument_type}):void
{
    if( ${property} !== value)
        {
            ${property} = value;
            dispatchEvent(new Event("${event_name}"));
        }
}
```

Variables de code pour les fonctions dans une classe non définie

Variable	Description	Exemple
${params}	Pour une fonction non définie qui accepte un nombre spécifié d'arguments, la fonction générée possède le même nombre d'arguments de ce type.	

Remplacement ou implémentation de méthodes

Flash Builder permet de sélectionner et de remplacer des méthodes d'une classe parent ou d'implémenter des méthodes d'interface.

1. Ouvrez la boîte de dialogue Remplacer/Implémenter des méthodes en sélectionnant l'option correspondante dans le menu Source. Vous pouvez également sélectionner Source > Remplacer/Implémenter des méthodes dans le menu contextuel de l'éditeur MXML ou ActionScript.
2. Les méthodes de chaque classe parent s'affichent sous forme d'arborescence. Pour chaque classe, vous pouvez sélectionner les méthodes à remplacer et les méthodes à implémenter.
3. Vous pouvez choisir le point d'insertion pour insérer les méthodes sélectionnées. L'option de point d'insertion par défaut dépend de l'emplacement du curseur dans l'éditeur lors de l'ouverture de la boîte de dialogue Remplacer/Implémenter des méthodes. La variable ou la méthode la plus proche de l'emplacement du curseur apparaît comme option de point d'insertion.

   Flash Builder génère le code de remplacement pour les méthodes sélectionnées.

Pour personnaliser le code de remplacement prédéfini généré par Flash Builder, voir « Modèles de code » à la page 41.

Remplissage du code de métadonnées

Flash Builder affiche des indicateurs de remplissage du code des métadonnées que vous utilisez dans vos documents MXML et ActionScript.

Dans un document MXML, les indicateurs de remplissage du code s'affichent dans les balises `<fx:Metadata>` et `<fx:Script>` intégrées. Dans un document ActionScript, les indicateurs de remplissage du code s'affichent également pour les éléments de langage ActionScript, tels que les noms de classe, les variables et les méthodes set/get.

Les conseils de code sont liés au contexte du document MXML et ActionScript, ainsi qu'au code dans lequel les métadonnées sont utilisées. Par exemple, lorsque vous appelez l'assistant de contenu au sein de deux lignes vierges d'une instruction ActionScript, les conseils de code applicables uniquement à cette instruction ActionScript apparaissent. Pour voir tous les conseils de code valides applicables au document ActionScript ou MXML, appuyez plusieurs fois sur Ctrl+Espace pour parcourir les conseils de code disponibles.

Utilisation du remplissage du code de métadonnées dans les documents MXML

Dans un document ou une classe MXML, vous pouvez utiliser le remplissage du code de métadonnées comme suit :

- Saisissez les balises « [» dans `<fx:Metadata>` comme suit :

```
<fx:Metadata>
    [
</fx:Metadata>
```

- Saisissez les balises « [» dans `<fx:Script>` comme suit :

```
<fx:Script>
    <![CDATA[
        [
    ]]>
</fx:Script>
```

Utilisation du remplissage du code de métadonnées dans les documents ActionScript

Dans un document ActionScript, vous pouvez utiliser le remplissage du code de métadonnées lorsque vous saisissez [avant un nom de classe, une variable, une méthode get ou une méthode set, comme suit :

```
[
    class GetSet
    {
        [
            private var privateProperty:String;
            [
            public function get publicAccess():String
            {
                return privateProperty;
            }
            [
            public function set publicAccess(setValue:String):void
            {
                privateProperty = setValue;
            }
}
```

Remplissage du code pour les balises de métadonnées personnalisées

Flash Builder prend en charge le remplissage du code pour les balises de métadonnées personnalisées qui sont introduites lors de l'utilisation de structures Flex tierces.

Pour activer les conseils de code pour les balises de métadonnées personnalisées dans votre projet, générez un fichier SWC contenant un fichier `metadata.xml`, comme suit :

1. Créez un projet de bibliothèque. L'assistant de création d'un projet de bibliothèque Flex vous guide à travers les différentes étapes et vous demande d'indiquer le nom du projet, son emplacement, ainsi que le chemin de génération. Pour plus d'informations, voir « Création de projets de bibliothèque Flex » à la page 200.
2. Ajoutez le fichier `metadata.xml` dans le dossier src, sous le dossier racine de votre projet de bibliothèque. Incluez toutes les balises de métadonnées que vous souhaitez dans le fichier `metadata.xml`.

   Ajoutez le fichier `metadata.properties` (le cas échéant) dans le dossier de paramètres régionaux approprié. Par exemple : locale/en_US ou locale/fr_FR.

   Pour plus d'informations sur les balises de métadonnées, voir About metadata tags dans la documentation Flex.
3. Incluez le fichier `metadata.xml` dans le fichier SWC de bibliothèque, comme suit :
   a. Sélectionnez Projet > Propriétés > Chemin de génération de la bibliothèque Flex.

      Le fichier `metadata.xml` que vous avez ajouté apparaît sous l'onglet Ressources.

   b. Sélectionnez le fichier `metadata.xml` à inclure dans le fichier SWC, puis cliquez sur OK.

      Le fichier SWC est compilé et généré dans le dossier de sortie (bin) du projet de bibliothèque.
4. Sélectionnez le dossier de paramètres régionaux auquel vous avez ajouté le fichier `metadata.properties` (le cas échéant).

5 Une fois le fichier SWC généré, ajoutez-le au chemin de génération de votre projet, comme suit :

1 Sélectionnez Projet > Propriétés > Chemin d'accès à la génération Flex.

2 Cliquez sur Ajouter un fichier SWC.

3 Accédez à l'emplacement du fichier SWC ou saisissez-le, puis cliquez sur OK.

Une fois le fichier SWC ajouté à votre chemin de génération, les indicateurs de remplissage du code de métadonnées apparaissent pour les balises de métadonnées définies dans le fichier `metadata.xml`. Vous pouvez partager le fichier SWC entre vos applications ou les distribuer à d'autres développeurs.

Personnalisation des modèles de fichier

Flash Builder vous permet de personnaliser les informations par défaut contenues dans les nouveaux fichiers MXML, ActionScript et CSS. Il peut s'agir par exemple des variables définissant l'auteur et la date, des variables pour les balises et les attributs d'ouverture et de fermeture ainsi que des variables pour différentes déclarations ActionScript, pour différents préfixes d'espace de noms et pour presque tout contenu que vous souhaitez inclure dans un modèle de fichier. Les modèles de fichier sont utiles notamment pour spécifier des commentaires d'introduction et des informations de copyright.

Le contenu d'un nouveau fichier est spécifié dans un modèle de fichier accessible à partir de Préférences > Flash Builder > Modèles de fichier. Des modèles sont disponibles pour les types de fichier suivants.

ActionScript	Fichier ActionScript Classe ActionScript Interface ActionScript Composant habillable ActionScript
MXML	Application Web MXML Application de bureau MXML Composant MXML Module MXML Habillage MXML Rendu d'élément pour composants Spark Rendu d'élément pour composants MX Rendu d'élément pour composant MX DataGrid Rendu d'élément pour composant AdvancedDataGrid Rendu d'élément pour composant MX Tree
FlexUnit	Classe TestCase FlexUnit Classe TestSuite FlexUnit Classe TestCase FlexUnit4 Classe TestSuite FlexUnit4
CSS	Fichier CSS

Une fois modifié, un modèle peut être exporté afin de pouvoir être partagé avec les autres membres de votre équipe.

Modification d'un modèle de fichier

1 Sélectionnez Préférences > Flash Builder > Modèles de fichier.

2 Développez les types de fichier et sélectionnez le modèle que vous souhaitez modifier.

3 Cliquez sur Editer et modifiez le modèle.

Vous pouvez saisir les données directement dans l'éditeur Modèle ou sélectionner Variables afin d'insérer des données prédéfinies dans le modèle.

4 Cliquez sur OK pour enregistrer les modifications.

Les modifications sont appliquées aux nouveaux fichiers.

Exportation et importation de modèles de fichier

1 Sélectionnez Préférences > Flash Builder > Modèles de fichier.

2 Développez les types de fichier et sélectionnez un modèle.

3 Cliquez sur Exporter pour exporter le modèle vers un système de fichiers ou Importer pour importer un modèle précédemment exporté.

Les modèles sont exportés dans des fichiers XML.

Restauration des valeurs par défaut

Remarque : *le bouton Restaurer les valeurs par défaut rétablit les valeurs par défaut de tous les modèles de fichier. Vous ne pouvez pas restaurer les valeurs par défaut d'un seul modèle.*

❖ Pour restaurer les modèles par défaut, ouvrez la fenêtre Préférences > Flash Builder > Modèles de fichier et sélectionnez Restaurer les valeurs par défaut.

Variables des modèles

Variables des modèles pour tous les types de fichiers

Variable	Description	Exemple
${date}	Date actuelle	15 février 2009
${year}	Année actuelle	2009
${time}	Heure actuelle	15h15
${file_name}	Nom du fichier créé	HelloWorld.mxml
${project_name}	Nom du projet Flex ou ActionScript	Hello_World_Project
${user}	Nom d'utilisateur de l'auteur	jdoe
$$ ${dollar}	Symbole du dollar	$

Variables des modèles pour les fichiers MXML

Variable	Description	Exemple

${application}
${component}
${module}
	Spécifient les noms des balises de l'application, du composant ou du module MXML.
	Pour une application Web, ${application} s'étend à « Application ».
	Pour une application de bureau, ${application} s'étend à « WindowedApplication ».
	${component} s'étend à « Component ».
	${module} s'étend à « Module ».
	Ces balises permettent généralement de positionner les balises d'ouverture et de fermeture d'un fichier.
		Les balises suivantes :
		`<${application} ${xmlns}${wizard_attributes}${min_size}> ${wizard_tags}`
		`</${application}>`
		s'étendent à :
		`<s:Application xmlns:fx="http://ns.adobe.com/mxml/2009" xmlns:s="library://ns.adobe.com/flex/spark " xmlns:mx="library://ns.adobe.com/flex/halo " minWidth="1024" minHeight="768"> <s:layout> <s:BasicLayout/> </s:layout>`
		`</s:Application>`

${xml_tag}	Version XML	`<?xml version="1.0" encoding="utf-8"?>`
${xmlns}	Correspond à la définition d'espace de noms en fonction du type de SDK Flex du projet et du préfixe d'espace de noms défini dans les Préférences.	Pour un projet SDK Flex 4 : xmlns="http://ns.adobe.com/mxml/2009"
${min_size}	Taille minimale d'une application Web MXML.	`minWidth="1024" minHeight="768"`
${ns_prefix}	Préfixe d'espace de noms pour le SDK Flex du projet. Vous ne pouvez pas modifier les valeurs par défaut de cette variable.	Pour Flex 3 : `mx:` Pour Flex 4 : `fx:`

Variable	Description	Exemple
${wizard_attributes}	Spécifie la position des attributs définis par l'assistant Nouveau *Fichier*.	Pour une nouvelle application Web : ${application} ${xmlns}${wizard_attributes}> s'étendent à : <Application xmlns="http://ns.adobe.com/mxml/2009" layout="vertical">
${wizard_tags}	Spécifie la propriété de présentation des conteneurs définis par l'assistant Nouveau *Fichier*.	Pour une nouvelle application basée sur le SDK Flex 4 : <s:layout> <s:BasicLayout/> </s:layout>

${fx}	Préfixe pour l'espace de noms du document de langage MXML 2009. Le préfixe est défini dans le document MXML.	Lorsque le modèle de balise de bibliothèque suivant apparaît dans un document MXML :
		`<${fx}Library>`
		`  <${fx}Definition id="${def}">`
		`  ${cursor}`
		`  </${fx}Definition>`
		`</${fx}Library>`
		Une balise de bibliothèque est créée comme suit :
		`<fxLibrary> <fxDefinition id="def">`
		`</fxDefinition> </fxLibrary>`

Variable	Description	Exemple
${mx}	Préfixe pour l'espace de noms du document MX. Le préfixe est défini dans le document MXML.	Lorsque vous utilisez le modèle Combobox suivant dans le contexte des composants MX : `<${mx}ComboBox id="${comboBox}" rowCount="${rowCount:values(5)}" dataProvider="${dataProvider}"/> ${cursor}` Un élément Combobox est créé comme suit : <mx:ComboBox id="comboBox" rowCount="5" dataProvider="dataProvider"/>

${s}	Préfixe pour l'espace de noms du document Spark.	Lorsque vous utilisez le modèle Spark Button suivant dans le contexte des composants Spark :
		`<${s}Button id="${btn}" label="${myButton}" click="${onClick}(${event})"/> ${cursor}`
		Un bouton Spark est créé comme suit :
		`<s:Button id="btn" label="myButton" click="onClick(event)"/>`

${tag}	Nom complet de balise pour les composants MX du projet.	Lorsque vous utilisez le modèle List dans le contexte MXML :
		`<${list:tag(mx.controls.List)} id="${myList}"> <${dp:tag(dataProvider)}>`
		`<${arraycollection:tag(mx.collections.ArrayCollection)}> ${cursor} </${arraycollection}> </${dp}> </${list}>`
		Une liste est créée comme suit :
		`<s:List id="myList"> <s:dataProvider> <s:ArrayCollection> </s:ArrayCollection> </s:dataProvider> </s:List>`

Variables des modèles pour les fichiers ActionScript

Variable	Description	Exemple
${package_declaration}	Génère la déclaration du package.	Pour un fichier du package com/samples, génère : package com.samples
${import_declaration}	Génère les déclarations d'importation requises pour une nouvelle classe ou interface ActionScript.	Pour une sous-classe de TextBox, génère : import flex.graphics.TextBox;
${interface_declaration}	Génère la déclaration d'interface pour une nouvelle interface ActionScript.	Pour une nouvelle interface qui étend l'interface IButton, génère : public interface IMyButton extends IButton
${class_declaration}	Génère la déclaration de classe pour une nouvelle classe ActionScript.	Pour une nouvelle sous-classe de CheckBox, génère : public class MyCheckBox extends CheckBox
${class_body}	Génère toutes les instructions requises pour une nouvelle classe.	Pour une nouvelle sous-classe de Button implémentant l'interface IBorder, génère ce qui suit pour le corps de la classe : `public function MyButton() { super(); } public function get borderMetrics():EdgeMetrics { return null; }`

${interface_name}
${class_name}
${package_name}
	Spécifie le nom de l'interface, de la classe ou du package.
	Cette variable est utilisée généralement lors de la création de commentaires.
		Par exemple, la spécification de modèle :
		`/* * ${class_name} implements. . . */`
		génère le code :
		`/* * MyButton implements. . . */`

${array}	Indique la valeur d'un tableau.	Le modèle `Fore` suivant
		`for each (var ${index}:${type} in ${array}) { ${line_selection} ${cursor} }`
		réalise une itération sur la valeur d'un tableau à l'aide d'une énumération comme suit :
		`for each (var i:type in array) { }`

${enclosing_method}	Indique le nom de la méthode englobante.	Le modèle `traceMethod` effectue le suivi de la méthode comme suit : `trace("${enclosing_type}.${enclosing_method} ${enclosing_method_arguments})");`
${enclosing_package}	Indique le nom du package, tel que « xx.yy » de « xx.yy.class ».	Le modèle `package` crée un package comme suit : `package ${enclosing_package} { /** * @author ${user} */ class ${enclosing_type} { ${cursor} } }`

Variable	Description	Exemple

${enclosing_type}	Indique le nom du type, tel que « class » de « xx.yy.class ».	Le modèle `package` crée un package indiquant le nom de classe comme suit :
		`package ${enclosing_package}`
		`{`
		`/**`
		`* @author ${user}`
		`*/`
		`class ${enclosing_type}`
		`{`
		`${cursor}`
		`}`
		`}`

${field}	Indique les variables de classe.	Le modèle `do` crée la boucle « do-while » comme suit :
		`do`
		`{`
		`${line_selection}`
		`${cursor}`
		`} while`
		`(${condition:local_var(Boolean)});`

${local_var}	Spécifie la variable locale qui est visible dans le bloc.	Le modèle `if` crée une instruction if comme suit :
		`if (${condition:local_var(Boolean)})`
		`{`
		`${cursor}`
		`}`

${var}	Spécifie toutes les variables visibles.	Le modèle `fori` réalise une itération sur la valeur d'un tableau comme suit :
		`for (var ${index}:int = 0; ${index} < ${array}.length; ${index}++)`
		`{`
		`${cursor}`
		`}`

Variables des modèles pour les fichiers CSS

Variable	Description	Exemple

${css_namespaces}	Définit les espaces de noms pour les sélecteurs de style Spark et Halo.	Valeurs par défaut pour Flex 3 :
		`""`
		(Dans Flex 3, les déclarations d'espace de noms ne sont pas requises dans les fichiers CSS).
		Valeurs par défaut pour Flex 4 :
		`@namespace s "library://ns.adobe.com/flex/spark";`
		`@namespace mx "library://ns.adobe.com/flex/halo";`

Exemples de fichiers de modèle

Les codes suivants illustrent un exemple de modèle de fichier de composant MXML, suivi par un nouveau fichier de composant MXML généré à partir du modèle.

Exemple de modèle pour un fichier de composant MXML

```
${xml_tag}
<!--
* ADOBE SYSTEMS Confidential
*
* Copyright ${year}. All rights reserved.
*
* ${user}
* ${project_name}
* Created ${date}
*
-->
<${component} ${xmlns}${wizard_attributes}>
    ${wizard_tags}

    <${ns_prefix}Script>
    <![CDATA[

    ]]>
    </${ns_prefix}Script>
</${component}>
```

Nouveau fichier de composant MXML généré à partir du modèle d'exemple

```
<?xml version="1.0" encoding="utf-8"?>
<!--
* ADOBE SYSTEMS Confidential
*
* Copyright 2009. All rights reserved.
*
* jdoe
* FileTemplates
* Created Jul 13, 2009
*
-->

<s:Group xmlns:fx="http://ns.adobe.com/mxml/2009"
xmlns:s="library://ns.adobe.com/flex/spark"
xmlns:mx="library://ns.adobe.com/flex/halo" width="400" height="300">
    <s:layout>
        <s:BasicLayout/>
    </s:layout>

    <fx:Script>
        <![CDATA[

        ]]>
    </fx:Script>
</s:Group>
```

Génération à partir de l'utilisation

L'Assistant rapide permet de générer du code de remplacement pour une méthode, une variable ou une classe non définie dans le code. Le code de remplacement généré peut être utilisé comme espace réservé pour le code que vous souhaitez implémenter plus tard sans rendre votre code incompatible. Pour personnaliser le code de remplacement prédéfini généré par Flash Builder, voir « Modèles de code » à la page 41.

Pour appeler l'Assistant rapide, vous pouvez utiliser le raccourci clavier Ctrl+1(Windows) ou Commande+1 (Mac OS).

Utilisez l'Assistant rapide pour générer un code de remplacement dans la classe appropriée ou un bloc de script MXML en sélectionnant une des actions suivantes :

Générer la méthode : crée une méthode

Par exemple, si vous avez le code suivant :

```
private function genFunc():void
{
bar();
}
```

Placez votre curseur à n'importe quel endroit sur la ligne de code contenant `bar();`, puis appuyez sur Ctrl+1. Une option permettant de créer une fonction apparaît. Lorsque vous sélectionnez cette option, une nouvelle fonction est générée comme suit :

```
private function bar():void{}
```

Vous pouvez également générer une fonction pour une fonction non définie dans une classe référencée. Par exemple, si vous avez une fonction non définie « setValue() » dans la classe référencée « MyClass », comme suit :

```
MyClass cls = new MyClass();
cls.setValue(5);
```

Placez votre curseur à n'importe quel endroit sur la ligne de code contenant « setValue », puis appuyez sur Ctrl+1. Une option permettant de créer une fonction s'affiche. Lorsque vous sélectionnez cette option, une nouvelle fonction `setValue(int)` est générée dans la classe « MyClass » référencée, comme suit :

```
 private function setValue(param0:int):void
{
    // TODO Auto Generated method stub
}
```

Générer une variable : crée une propriété

Par exemple, si dans votre code, `i` est une variable non définie, comme suit :

```
public function MyClass
{
        i;
}
```

Placez votre curseur à n'importe quel endroit sur la ligne de code contenant `i;`, puis appuyez sur Ctrl+1. Des options permettant de créer une variable locale ou un champ apparaissent.

Sélectionner l'option qui permet de créer une variable locale a pour effet de créer la variable comme suit :

```
var i:Object;
```

Sélectionner l'option qui permet de créer un champ a pour effet de créer une variable au niveau de la classe, comme suit :

```
private var i:Object;
```

Vous pouvez également générer une propriété pour une variable non définie dans une classe référencée. Par exemple, si vous avez une variable non définie « aValue » dans la classe référencée « MyClass », comme suit :

```
MyClass cls = new MyClass();
cls.aValue = "str";
```

Placez votre curseur à n'importe quel endroit sur la ligne de code contenant `aValue`, appuyez sur Ctrl+1, puis sélectionnez Créer la variable de champ. Une propriété `aValue` de type chaîne est générée dans la classe « MyClass » référencée, comme suit :

```
private var aValue:String;
```

Générer une classe/interface : crée une classe ou une interface

Par exemple, si dans votre code, `Foo` est une variable non définie, comme suit :

```
public function myFunction():Foo;
    {
    }
```

Placez votre curseur à n'importe quel endroit sur la ligne de code contenant `Foo;`, puis appuyez sur Ctrl+1. Des options permettant de créer une classe ou une interface nommée `Foo` apparaissent. Sélectionnez une de ces options pour ouvrir l'assistant de création d'une classe ActionScript ou l'assistant de création d'une interface ActionScript. Entrez les détails nécessaires et cliquez sur Terminer. Après avoir cliqué sur Terminer, une classe ou interface nommée `Foo` est créée.

Lorsque vous générez une nouvelle classe, vous pouvez créer une classe ActionScript avec un constructeur paramétré.

Par exemple, si votre code ressemble au suivant :

```
Private function func(): void {
    New NewClass("str1");
}
```

Placez votre curseur à n'importe quel endroit sur la ligne de code contenant NewClass("str1"), appuyez sur Ctrl+1, puis sélectionnez Créer la classe. Une classe ActionScript avec un constructeur paramétré est créée. Toutefois, si vous spécifiez une superclasse pour la classe ActionScript, aucun constructeur paramétré n'est généré.

Paul Robertson, professionnel de la communauté Adobe, a publié sur son blog un article concernant l'utilisation de Quick Assist pour les classes externes et les interfaces.

Générer un gestionnaire d'événement : génère des fonctions de gestionnaire d'événement

Par exemple, si votre code ressemble au suivant :

```
public function MyClass
    {
        Var button:Button = new Button();
        button.addEventListener(DragEvent.DRAG,dragHandlerFunction);
    }
```

Placez votre curseur à n'importe quel endroit sur la ligne de code contenant `dragHandlerFunction`, puis appuyez sur Ctrl+1. Sélectionnez ensuite l'option de l'Assistant rapide permettant de créer le gestionnaire d'événement. La fonction du gestionnaire d'événement est créée comme suit :

```
protected function dragHandlerFunction (event:DragEvent):void
{
}
```

Générer l'instruction d'importation de l'utilisation : créer une instruction d'importation

Par exemple, si vous avez un code dans lequel le type de variable `Button` n'est pas défini, comme suit :

```
<fx:Script>
    <![CDATA[
        var btn:Button;
    ]]>
</fx:Script>
```

Placez votre curseur à n'importe quel endroit sur la ligne de code contenant `var btn:Button`, puis appuyez sur Ctrl+1. Une option d'importation de `Button` apparaît si une classe nommée `Button` est disponible dans le projet. L'instruction d'importation est créée comme suit :

```
import spark.components.Button;
```

Vous pouvez générer des instructions d'importation pour les arguments de fonction, les types de retour de fonction, etc.

Génération de fonctions d'accesseur get ou set

Les fonctions des accesseurs get et set permettent de faire en sorte que les propriétés d'une classe restent privées pour cette classe. Les utilisateurs peuvent accéder à ces propriétés de la même manière qu'ils accèdent à une variable de classe (au lieu d'appeler une méthode de classe).

Flash Builder peut générer des fonctions d'accesseur get et set ActionScript pour les variables de classe. Vous pouvez sélectionner une propriété liable et générer des fonctions d'accesseur get et set pour cette propriété. Vous pouvez également spécifier un nom d'événement personnalisé pendant la génération du code.

Méthode de génération de fonctions d'accesseur get ou set

1. Ouvrez un fichier ActionScript dans l'éditeur de source, puis placez le curseur sur une variable de classe.
2. Sélectionnez Source > Générer les méthodes get/set dans le menu Flash Builder ou le menu contextuel.
3. Dans la boîte de dialogue Générer les méthodes get/set, spécifiez les informations concernant les fonctions d'accesseur puis cliquez sur OK.

   Remarque : *pour afficher le code qui est généré, sélectionnez Aperçu avant de cliquer sur OK.*

Pour personnaliser le code prédéfini généré par Flash Builder, voir « Modèles de code » à la page 41.

Lorsque vous générez des accesseurs get et set, Flash Builder propose les options suivantes :

- Rendre la variable de classe privée.

  En règle générale, les variables de classe ont un accès privé.

- Renommer la variable de classe, en suggérant un trait de soulignement au début du nom.

  Par convention, les variables de classe privée ont un trait de soulignement au début du nom.

- Renommer les fonctions d'accesseur.
- Spécifier une propriété liable et un nom d'événement personnalisé.

  Lorsque vous spécifiez une propriété liable, une balise `[Bindable]` est définie au-dessus de la fonction d'accesseur générée dans le code généré.

- Préciser si vous voulez générer des fonctions d'accesseur get et set.
- Spécifier la valeur de l'espace de noms pour la fonction d'accesseur.
- Spécifier le placement de la fonction d'accesseur dans l'un des emplacements suivants :
  - Avant la première méthode
  - Après la dernière méthode
  - Avant les déclarations de variable
- Afficher un aperçu du code qui est généré.

Pour plus d'informations sur les fonctions d'accesseur get et set, voir Méthodes d'accesseur get et set dans le *Guide de référence du langage ActionScript 3.0 pour la plateforme Adobe Flash*.

Chapitre 4 : Utilisation des projets dans Flash Builder

Adobe Flash Builder regroupe les ressources (dossiers et fichiers) qui constituent une application dans un conteneur appelé *projet*.

Utilisez la vue Explorateur de packages pour ajouter, modifier et supprimer des ressources de projet. Vous pouvez également fermer les projets à l'intérieur d'un espace de travail, importer des ressources et créer des liens vers des ressources externes.

Types de projets

Flash Builder prend en charge les différents types de projets, selon le type d'application que vous créez.

Projets Flex
Utilisez l'assistant Projet Flex pour créer une application Web ou d'ordinateur basée sur la structure Flex. Une application Web s'exécute dans Adobe Flash Player, tandis qu'une application de bureau s'exécute dans Adobe AIR. Lorsque vous créez le projet, vous indiquez s'il est destiné aux applications Web ou d'ordinateur.

Un projet Flex contient un ensemble de propriétés qui contrôlent la façon dont l'application est générée, l'endroit où l'application construite est hébergée, la manière dont le débogage est géré et les relations avec les autres projets de l'espace de travail.

Un projet Flex contient un fichier d'application MXML par défaut. Les autres fichiers MXML du projet peuvent également servir de fichiers d'application.

Si un fichier ActionScript se trouve dans votre projet Flex, vous pouvez définir le fichier comme application par défaut en sélectionnant Définir comme application par défaut dans le menu contextuel du projet.

Voir « Projets Flex » à la page 61 et « Développement d'applications Web et d'ordinateur dans Flash Builder » à la page 171.

Projets Flex Mobile
Utilisez l'assistant Projet Flex Mobile pour créer une application Adobe AIR destinée à une plateforme mobile. L'application repose sur la structure Flex. Vous pouvez utiliser Flash Builder pour afficher un aperçu, déboguer et profiler les applications mobiles à partir de l'ordinateur ou sur un périphérique.

Un projet Flex Mobile comporte un seul fichier d'application MXML par défaut. En général, une application mobile comporte un jeu de composants Vue qui affichent le contenu d'un périphérique. Le fichier d'application MXML par défaut lance le composant Vue par défaut.

Flash Builder utilise AIR Debug Launcher (ADL) pour obtenir un aperçu des applications mobiles sur l'ordinateur. Bien qu'il ne s'agisse pas d'une vraie émulation, ADL vous permet de voir la mise en forme et le comportement de l'application, en proposant des options pour faire tourner l'application.

Vous pouvez afficher un aperçu de l'application mobile sur un périphérique connecté au port USB de l'ordinateur de développement. Lorsque vous affichez un aperçu sur un périphérique, Flash Builder exporte l'application sur le périphérique.

Voir « Projets Flex Mobile » à la page 63 et Developing mobile applications in Flash Builder (Développement d'applications mobiles dans Flash Builder).

Projets de bibliothèque Flex

Utilisez l'assistant Bibliothèque de projet Flex pour créer des bibliothèques de code personnalisées à partager entre les applications ou à distribuer à d'autres développeurs. En règle générale, vous utilisez les projets de bibliothèque pour assembler et distribuer les composants et ressources à d'autres développeurs.

Un projet de bibliothèque génère un fichier SWC, c'est-à-dire un fichier archive pour les composants Flex ainsi que pour d'autres ressources.

Voir « Utilisation de projets de bibliothèque Flex » à la page 200.

Projets ActionScript

Utilisez l'assistant Projet ActionScript pour créer des applications Web ou de bureau qui reposent sur les API Flash ou Adobe AIR. Lorsque vous créez le projet, vous indiquez s'il est destiné à une application Web ou d'ordinateur.

Vous travaillez exclusivement dans l'éditeur ActionScript, et dans les outils de débogage selon les besoins, puis créez le projet dans des fichiers SWF pour afficher un aperçu de votre application et la tester.

Lorsque vous créez un projet ActionScript ou un fichier ActionScript autonome destiné à contenir des fonctions, une classe ou une interface, la perspective de développement Flex change pour prendre en charge l'éditeur ActionScript. Les principales vues associées à l'éditeur ActionScript sont les vues Structure et Erreurs.

Voir « Projets ActionScript » à la page 64.

Projets ActionScript Mobile

Utilisez l'assistant Projet ActionScript Mobile pour créer des applications mobiles qui reposent sur l'API Adobe AIR. Lorsque vous créez le projet, vous spécifiez une plateforme mobile cible et certains paramètres d'applications mobiles. Vous pouvez utiliser Flash Builder pour afficher un aperçu de l'application mobile à partir de l'ordinateur ou sur un périphérique.

Flash Builder utilise Adobe AIR Debug Launcher (ADL) pour afficher un aperçu, déboguer et profiler des applications mobiles sur l'ordinateur. Bien qu'il ne s'agisse pas d'une vraie émulation, ADL vous permet de voir la mise en forme et le comportement de l'application, en proposant des options pour faire tourner l'application.

Vous pouvez afficher un aperçu de l'application mobile sur un périphérique connecté au port USB de l'ordinateur de développement. Lorsque vous affichez un aperçu sur un périphérique, Flash Builder exporte l'application sur le périphérique. Vous pouvez faire appel à Flash Builder pour déboguer l'application exportée sur un périphérique.

Voir « Création de projets ActionScript Mobile » à la page 64.

Projets de bibliothèque ActionScript

Utilisez l'assistant de bibliothèque de projet ActionScript pour créer des bibliothèques de code personnalisées. En général, vous créez le projet de bibliothèque ActionScript, ajoutez des classes ActionScript spécifiques et compilez le projet. Flash Builder génère ensuite un fichier SWC à distribuer aux autres développeurs ou à partager entre les applications.

Voir « Création de projets de bibliothèque ActionScript » à la page 204.

Projets Flash Professional

Utilisez l'assistant Projet Flash Professional pour modifier, créer ou déboguer des fichiers FLA ou XFL créés dans Adobe Flash Professional. Cette fonctionnalité permet aux développeurs Flash Professional de profiter de l'environnement de modification et de débogage que fournit Flash Builder. Les projets Flash Professional sont disponibles dans Flash Builder uniquement si vous avez installé Flash Professional.

Voir « Utilisation de Flash Builder avec Flash Professional » à la page 254.

Création de projets dans Flash Builder

Flash Builder permet de créer différents types de projets tels que Flex, Flex Mobile, ActionScript et Adobe AIR. Flash Builder fournit un assistant de création de projet qui vous guide à travers les différentes étapes en vous demandant le type de projet que vous souhaitez créer, le nom du projet, son emplacement, le type d'application (Web ou d'ordinateur), les version du SDK ainsi que d'autres options.

Pour plus d'informations sur la création d'un projet ActionScript, voir « Projets ActionScript » à la page 64.

Pour plus d'informations sur la création de projets de bibliothèque, voir « Utilisation de projets de bibliothèque Flex » à la page 200.

Pour plus d'informations sur la création de projets Flash Professional, voir « Utilisation de Flash Builder avec Flash Professional » à la page 254.

Projets Flex

Vous pouvez utiliser les projets Flex afin de créer des applications Web (dans Flash Player) ou des applications de bureau (dans Adobe AIR). Des options permettent de créer des projets Flex MX uniquement ou des projets Flex Spark uniquement, n'utilisant que des composants Spark.

Création de projets Flex

Utilisez cette procédure pour créer des applications Web ou d'ordinateur.

1. Sélectionnez Fichier > Nouveau > Projet Flex.
2. Entrez un nom de projet et un emplacement.

   L'emplacement par défaut est l'espace de travail actuel.
3. Sélectionnez Web ou Bureau comme type d'application.
4. Utilisez le SDK Flex par défaut ou accédez à un autre SDK installé. Cliquez sur Suivant.
5. (Facultatif) Indiquez les paramètres du serveur.

   Voir « Accès aux services de données » à la page 186.
6. Spécifiez un dossier de sortie.

   Si vous ne spécifiez aucun serveur d'applications, l'emplacement se trouve dans votre dossier du projet.

   Si vous spécifiez un serveur d'applications, le serveur de sortie se trouve hors du dossier du projet. En général, vous placez le dossier de sortie avec vos fichiers de service.
7. Cliquez sur Terminer ou sur Suivant pour indiquer d'autres options de configuration.
8. (Facultatif) Spécifiez des chemins d'accès à la génération et d'autres options de configuration.

   Voir « Chemins de génération, extension natives et autres options de configuration du projet » à la page 65.

9 Cliquez sur Terminer.

Regardez la vidéo Développement de votre première application de bureau avec Flash Builder (en anglais) de l'expert Adobe James Ward.

Création d'un projet Flex utilisant uniquement des composants MX

L'option MX uniquement intervient lors de la création d'applications dont la conception est similaire aux applications créées avec une version antérieure de Flex (c'est-à-dire, Flex 3), mais ayant malgré tout accès aux dernières fonctions de Flex et de Flash Builder (telles que la syntaxe d'états, les fonctionnalités CSS avancées, des fonctions de compilation améliorées ainsi que d'autres fonctionnalités de langage).

L'option MX uniquement ne met pas à la disposition des applications du projet les composants Spark.

Vous pouvez convertir un projet Flex en projet MX uniquement. Gardez cependant à l'esprit que Flash Builder ne réécrit pas le code du projet. Mettez manuellement à jour le code afin d'en supprimer toute référence aux composants Spark.

Création d'un projet Flex MX uniquement

1 Sélectionnez Fichier > Nouveau > Projet Flex.

2 Spécifiez l'emplacement du projet et les paramètres du serveur, comme décrit dans « Chemins de génération, extension natives et autres options de configuration du projet » à la page 65.

3 Sur la page des chemins de génération de l'assistant Nouveau projet Flex, spécifiez MX uniquement.

4 Indiquez d'autres paramètres de génération de chemin, comme le décrit la section « Chemins de génération, extension natives et autres options de configuration du projet » à la page 65. Cliquez sur Terminer.

Conversion d'un projet Flex en projet Flex MX uniquement

1 Rendez le projet actif dans Flash Builder :

Pour ce faire, vous devez généralement ouvrir un fichier source du projet.

2 Sélectionnez Projet > Propriétés > Chemin d'accès à la génération Flex.

3 Pour le paramètre Jeu de composants, sélectionnez l'option MX uniquement. Cliquez sur OK.

4 Modifiez tout code d'application du projet accédant à des composants Spark.

Il ne peut y avoir de références à des composants Spark dans un projet MX uniquement.

Création d'un projet Flex utilisant uniquement des composants Spark

L'option Spark Only est utile pour créer des applications qui utilisent les fonctionnalités Flex et Flash Builder, telles que la nouvelle syntaxe d'états, les fonctionnalités CSS avancées, des fonctions de compilation améliorées ainsi que d'autres fonctionnalités de langage.

L'option Spark uniquement ne met pas à la disposition des applications du projet les composants MX fournis avec Flex 3.

1 Sélectionnez Fichier > Nouveau > Projet Flex.

2 Spécifiez l'emplacement du projet et les paramètres du serveur, comme le décrit la section « Projets Flex » à la page 61.

3 Sur la page des chemins de génération de l'assistant Nouveau projet Flex, spécifiez Spark uniquement.

4 Indiquez d'autres paramètres de génération de chemin, comme le décrit la section « Chemins de génération, extension natives et autres options de configuration du projet » à la page 65. Cliquez sur Terminer.

Projets Flex Mobile

Flash Builder permet de créer des projets Flex adaptés aux périphériques mobiles. Un projet Flex Mobile crée une application Adobe AIR pouvant s'exécuter sur les périphériques Apple iOS, BlackBerry Tablet OS et Google Android.

Création de projets Flex Mobile

Utilisez cette procédure pour créer un projet Flex Mobile pour les périphériques Apple iOS, BlackBerry Tablet OS et Google Android.

1 Sélectionnez Fichier > Nouveau > Projet Flex Mobile.

2 Entrez un nom de projet et un emplacement.

L'emplacement par défaut est l'espace de travail actuel.

3 Utilisez le SDK Flex 4.6 par défaut, qui prend en charge le développement des applications mobiles, ou un SDK Apache Flex. Cliquez sur Suivant.

4 Spécifiez les paramètres mobiles :

- Sélectionnez la plateforme que votre application doit cibler. Pour plus d'informations, voir « Choix des plateformes cibles » à la page 226.
- Spécifiez un modèle d'application.

  Pour plus d'informations, voir « Choix d'un modèle d'application » à la page 226.
- Spécifiez les autorisations.

  Sélectionnez la plateforme cible et définissez les autorisations pour chaque plateforme, le cas échéant. Vous pourrez modifier les autorisations par la suite dans le fichier XML descripteur de l'application.

  Pour plus d'informations, voir « Choix des autorisations d'une application mobile » à la page 227.
- Spécifiez les paramètres de plateforme.

  Les paramètres de plateforme vous permettent de sélectionner une gamme de périphériques cibles. Selon la plateforme sélectionnée, vous pouvez choisir le périphérique cible ou une gamme de périphériques cibles.

  Remarque : *il n'existe aucun paramètre spécifique aux plateformes Google Android et BlackBerry Tablet OS.*

  Pour plus d'informations, voir « Choix des paramètres de plateforme » à la page 228.
- Spécifiez les paramètres d'application.

  Pour plus d'informations, voir « Choix des paramètres d'application » à la page 228.

5 Cliquez sur Terminer ou Suivant pour spécifier les paramètres du serveur.

6 (Facultatif) Indiquez les paramètres du serveur.

Pour plus d'informations, voir « Accès aux services de données » à la page 186.

7 Spécifiez un dossier de sortie.

Si vous ne spécifiez aucun serveur d'applications, l'emplacement se trouve dans votre dossier du projet.

Si vous spécifiez un serveur d'applications, le serveur de sortie se trouve hors du dossier du projet. En général, vous placez le dossier de sortie avec vos fichiers de service.

8 Cliquez sur Terminer ou sur Suivant pour indiquer d'autres options de configuration.

9 (Facultatif) Spécifiez des chemins d'accès à la génération et d'autres options de configuration.

Pour plus d'informations, voir « Chemins de génération, extension natives et autres options de configuration du projet » à la page 65.

10 Cliquez sur Terminer.

Brian Telintelo, consultant Flex, a publié un article concernant la création d'une application mobile pour la plateforme Android.

Pour plus d'informations sur le développement d'applications mobiles avec Flex et Flash Builder, voir Developing mobile applications in Flash Builder (Développement d'applications mobiles dans Flash Builder).

Voir aussi

« Projets Flex Mobile » à la page 63

Projets ActionScript

Utilisez des projets ActionScript pour créer des applications Web ou de bureau basées sur le SDK AIR. Ce SDK contient le SDK d'Adobe AIR, le compilateur ActionScript (ASC) et d'autres composants et fichiers requis.

Création de projets ActionScript

Créez une application ActionScript pour le Web ou de bureau en effectuant les étapes suivantes :

1 Sélectionnez Fichier > Nouveau > Projet ActionScript.

2 Entrez un nom de projet et un emplacement.

L'emplacement par défaut est l'espace de travail actuel.

3 Sélectionnez Web ou Bureau comme type d'application.

4 Ce projet utilise le SDK AIR 3.4 par défaut. Cliquez sur Rechercher des mises à jour pour rechercher et télécharger toute version mise à jour disponible du SDK AIR.

Cliquez sur Suivant.

5 Spécifiez les chemins de génération.

Pour plus d'informations, voir « Chemins de génération, extension natives et autres options de configuration du projet » à la page 65.

6 Cliquez sur Terminer.

Pour plus d'informations sur le codage dans ActionScript, voir le Guide du développeur ActionScript.

Création de projets ActionScript Mobile

Créez des applications ActionScript basées sur AIR et compatibles avec les périphériques Apple iOS, BlackBerry Tablet OS et Google Android en effectuant les étapes suivantes :

1 Sélectionnez Fichier > Nouveau > Projet ActionScript Mobile.

2 Entrez un nom de projet et un emplacement.

L'emplacement par défaut est l'espace de travail actuel.

3 Ce projet utilise le SDK AIR 3.4 par défaut. Cliquez sur Rechercher des mises à jour pour rechercher et télécharger toute version mise à jour disponible du SDK AIR.

Cliquez sur Suivant.

4 Spécifiez les paramètres mobiles :

- Sélectionnez les plateformes que votre application doit cibler.

  Pour plus d'informations, voir « Choix des plateformes cibles » à la page 226.

- Spécifiez les autorisations de plateforme.

  Sélectionnez la plateforme cible et définissez les autorisations pour chaque plateforme, le cas échéant. Vous pourrez modifier les autorisations par la suite dans le fichier XML descripteur de l'application.

  Pour plus d'informations, voir « Choix des autorisations d'une application mobile » à la page 227.

- Spécifiez les paramètres de plateforme.

  Pour plus d'informations, voir « Choix des paramètres de plateforme » à la page 228.

- Spécifiez les paramètres d'application.

  Sélectionnez Réorientation automatique pour que l'application pivote lorsque le périphérique subit une rotation.

  Sélectionnez Plein écran pour que l'application s'affiche en mode plein écran sur le périphérique.

5 Cliquez sur Terminer ou sur Suivant pour indiquer d'autres options de configuration.

6 (Facultatif) Indiquez les chemins de génération.

Voir « Chemins de génération, extension natives et autres options de configuration du projet » à la page 65.

7 Cliquez sur Terminer.

Voir aussi

« Gestion des configurations de lancement » à la page 105

Chemins de génération, extension natives et autres options de configuration du projet

Lorsque vous créez un projet Flex ou ActionScript, vous pouvez personnaliser sa configuration. Toutes les étapes de configuration supplémentaires sont facultatives.

Remarque : *vous pouvez également modifier la configuration d'un projet après sa création en sélectionnant Projet > Propriétés.*

Chemin source : cliquez sur l'onglet Chemin source pour ajouter des dossiers source supplémentaires à un projet. Vous pouvez réorganiser les dossiers source, en modifier l'emplacement ou les supprimer du chemin source.

Dossier source principal, Fichier de l'application principale et URL du dossier de sortie : par défaut, Flash Builder place les fichiers source dans le dossier src du projet.

Pour les projets Flex, le nom par défaut du fichier de l'application MXML principale est le nom du projet. Ces valeurs par défaut peuvent être modifiées à la création du projet.

Lorsque vous créez un projet, Flash Builder exécute les fichiers d'application à partir d'une URL par défaut, en fonction des paramètres du projet. Spécifiez une URL de dossier de sortie pour remplacer les paramètres par défaut. L'URL de dossier de sortie est utile lorsque vous créez et déployez l'application sur le serveur de votre projet, mais que vous déboguez l'application sur un serveur Web. Vous spécifiez ensuite l'URL du serveur Web comme URL de dossier de sortie. Par exemple, si vous indiquez http://myserver/test.swf comme URL de dossier de sortie, une configuration de lancement est créée avec cette URL.

Voir « Définition d'un dossier de sortie de projet » à la page 91 et « Exécution et débogage des applications » à la page 105.

Chemin d'accès à la bibliothèque : utilisez le chemin d'accès à la bibliothèque afin de spécifier le lien du cadre et les bibliothèques de chemins d'accès à la génération.

- Jeu de composants

  Le jeu de composants est disponible pour les projets Flex uniquement. En règle générale, tous les composants sont disponibles. Dans certains cas, vous pouvez ne spécifier que les composants MX. Voir « Jeu de composants (MX + Spark, Spark uniquement ou MX uniquement) » à la page 90.

- Liaison de la structure

  Pour les projets Flex, les classes d'application pour Flex 4.5 ou versions ultérieures de la structure Flex utilisent par défaut la liaison dynamique. Pour plus d'informations, voir « Liaison de la structure des applications » à la page 90.

  Les options suivantes sont disponibles uniquement pour les projets Flex et sont activées par défaut :

  - Vérifier les condensés RSL (recommandé en production)
  - Supprimer les RSL non utilisées

    Remarque : *cette option n'est pas disponible dans les structures Flex antérieures à Flex 4.5.*
  - Utiliser les bibliothèques RSL locales de débogage lors du débogage
  - Déterminer automatiquement l'ordre de la bibliothèque, en fonction des dépendances

- Bibliothèques du chemin de génération

  Vous pouvez ajouter au chemin de génération des bibliothèques de projet, des dossiers de bibliothèque SWC ou des fichiers SWC ; vous pouvez également les en supprimer. Vous pouvez en outre modifier l'ordre du chemin de génération.

  Cliquez sur le bouton Editer pour modifier l'emplacement des bibliothèques ou des dossiers ajoutés.

  Utilisez le bouton Ajouter un SDK Flex pour rétablir le SDK par défaut d'un projet dont vous avez supprimé le SDK Flex du chemin de génération.

Extensions natives : l'onglet Extensions natives est disponible uniquement pour les projets mobiles. utilisez l'onglet Extensions natives pour inclure les fichiers ANE (ActionScript Native Extension) et intégrer des fonctions de plateforme native dans vos applications.

Important : *vous pouvez créer des extensions ActionScript seulement pour les projets mobiles prenant en charge Adobe AIR 3.0 ou version ultérieure.*

Pour plus d'informations, voir « Ajout d'extensions natives à un projet » à la page 223.

Définition des propriétés de projet

Chaque projet possède son propre jeu de propriétés. Pour définir ces propriétés, sélectionnez le projet dans la vue Explorateur de packages. Dans le menu principal, sélectionnez ensuite Projet > Propriétés. Vous pouvez également accéder aux propriétés à partir du menu contextuel du projet.

Propriétés des projets Flex

Pour les projets Flex, vous pouvez définir les préférences de projet suivantes :

Ressource : présente des informations générales concernant le projet, les paramètres pour le codage de texte et les délimiteurs de lignes du système d'exploitation.

Opérateurs ActionScript : affiche les informations relatives aux opérateurs ActionScript utilisés dans le projet.

Voir « Gestion des fichiers d'opérateur ActionScript » à la page 73.

Générateurs : précise l'outil de génération à utiliser. Flash Builder contient un outil de génération standard. Vous pouvez faire appel à Apache Ant (outil de génération « open source ») pour créer des scripts de génération ou importer des scripts de génération Ant existants

Voir « Personnalisation des générations avec Apache Ant » à la page 95.

Modèle de données : disponible uniquement avec LiveCycle Data Services. Il spécifie l'emplacement du fichier du modèle de données contenant les informations sur le type de service et de données pour LiveCycle Data Services ES.

Données/Services : pour les projets accédant à des services de données, cette page définit si le générateur de code par défaut doit être utilisé pour accéder aux services. Vous pouvez également indiquer si vous souhaitez utiliser une seule instance de serveur pour accéder aux services.

Pour plus d'informations sur l'extension de Flash Builder de manière à prendre en charge la génération de code personnalisé, voir Extension de la prise en charge des services dans Flash Builder.

Pour plus d'informations sur l'utilisation d'une seule instance de serveur lors de l'accès aux services, voir Utilisation d'une instance de serveur unique.

Applications Flex : affiche le nom des fichiers de projet définis comme fichiers d'application pouvant être compilés, débogués et exécutés en tant qu'applications distinctes

Voir « Gestion des fichiers d'application du projet » à la page 80.

Chemin d'accès à la génération Flex : spécifie le chemin d'accès à la génération, qui précise l'emplacement des fichiers source externes et des fichiers de bibliothèque. Vous pouvez modifier ce chemin et changer le nom du dossier de sortie

(voir « Définition d'un dossier de sortie de projet » à la page 91 et « Création manuelle de projets » à la page 93).

Compilateur Flex : spécifie d'une part les préférences optionnelles pour le compilateur (génération d'un fichier SWF accessible, activation des avertissements de compilation et de la vérification des types, spécification d'arguments de compilation supplémentaires et définition de la version du SDK Flex) et définit d'autre part les paramètres de l'enveloppe HTML

(voir « Options de génération avancées » à la page 93).

Modules Flex : précise les modules à générer et à optimiser pour le projet. Pour plus d'informations sur l'utilisation des modules dans Flash Builder, voir « Création d'un projet distinct pour les modules dans Flash Builder » à la page 208.

Serveur Flex : spécifie le type de serveurs d'applications pour le projet. A la création d'un projet, vous avez la possibilité de spécifier un type de serveurs d'applications. Vous pouvez modifier ce type ici. Si vous modifiez le type de serveurs d'applications d'un projet, vous risquez de ne plus pouvoir accéder aux services de données précédemment configurés.

Voir « Projets Flex » à la page 61 et Création d'un projet Flex pour l'accès à des services de données.

Thème Flex : spécifie le thème à utiliser pour toutes les applications du projet. Vous pouvez spécifier l'un des thèmes disponibles dans Flash Builder ou en importer un.

Voir « Utilisation de thèmes » à la page 172.

Références de projet : répertorie les projets auxquels le projet en cours fait référence.

Exécuter/déboguer les paramètres : gère les paramètres de configuration de lancement.

Voir « Gestion des configurations de lancement » à la page 105.

Propriétés des projets ActionScript

Pour les projets ActionScript, vous pouvez définir les préférences de projet suivantes :

Ressource : présente des informations générales concernant le projet, les paramètres pour le codage de texte et les délimiteurs de lignes du système d'exploitation.

Applications ActionScript : affiche le nom des fichiers de projet définis comme fichiers d'application pouvant être compilés, débogués et exécutés en tant qu'applications distinctes

Voir « Gestion des fichiers d'application du projet » à la page 80.

Chemin d'accès à la génération ActionScript : spécifie le chemin d'accès à la génération, qui précise l'emplacement des fichiers source externes et des fichiers de bibliothèque. Vous pouvez modifier ce chemin et changer le nom du dossier de sortie

(voir « Définition d'un dossier de sortie de projet » à la page 91 et « Création manuelle de projets » à la page 93).

Compilateur ActionScript : spécifie les préférences optionnelles pour le compilateur (génération d'un fichier SWF accessible, activation des avertissements de compilation et de la vérification des types, spécification d'arguments de compilation supplémentaires et définition de la version du SDK AIR) et définit les paramètres de l'enveloppe HTML

(voir « Options de génération avancées » à la page 93).

Modules ActionScript : précise les modules à générer et à optimiser pour le projet. Pour plus d'informations sur l'utilisation des modules dans Flash Builder, voir « Création d'un projet distinct pour les modules dans Flash Builder » à la page 208.

Opérateurs ActionScript affiche les informations relatives aux opérateurs ActionScript utilisés dans le projet.

Voir « Gestion des fichiers d'opérateur ActionScript » à la page 73.

Générateurs : précise l'outil de génération à utiliser. Flash Builder contient un outil de génération standard. Vous pouvez faire appel à Apache Ant (outil de génération « open source ») pour créer des scripts de génération ou importer des scripts de génération Ant existants

Voir « Personnalisation des générations avec Apache Ant » à la page 95.

Références de projet : répertorie les projets auxquels le projet en cours fait référence.

Exécuter/déboguer les paramètres : gère les paramètres de configuration de lancement.

Voir « Gestion des configurations de lancement » à la page 105.

Ressources des projets

Les projets sont constitués de ressources (dossiers et fichiers) que vous pouvez gérer depuis l'Explorateur de packages. Les projets sont contenus dans un espace de travail. L'Explorateur de packages propose une représentation logique de l'espace de travail dans le système de fichiers. Il est actualisé à chaque fois que vous ajoutez, supprimez ou modifiez une ressource.

Vous pouvez également modifier les ressources du projet directement dans le système de fichiers, sans passer par Flash Builder ni par l'Explorateur de packages.

Création ou suppression de fichiers et de dossiers dans un projet

Une application dans Flex est généralement composée d'un fichier d'application MXML, d'une ou plusieurs vues (projets mobiles uniquement) et d'un ou plusieurs composants Flex standard. Vous pouvez également avoir un ou plusieurs composants définis dans divers fichiers MXML, ActionScript ou de composant Flash (SWC). En scindant l'application en blocs de dimensions maniables, vous pouvez rédiger et tester chaque composant individuellement. Dans un souci d'efficacité, les composants peuvent être réutilisés, dans la même application ou dans une application différente.

Vous pouvez utiliser Flash Builder pour créer des composants MXML et ActionScript personnalisés, puis les intégrer à vos applications. Pour plus d'informations sur la création de composants ActionScript, voir « Création d'une classe ActionScript » à la page 70.

Vous pouvez également créer un composant MXML directement à partir de code.

Création de fichiers et de dossiers dans un projet

1. Dans l'Explorateur de packages, sélectionnez Fichier > Nouveau > Fichier.
2. Si l'espace de travail contient plusieurs projets, sélectionnez le projet auquel vous voulez ajouter le fichier.
3. Entrez le nom du fichier, puis cliquez sur Terminer.

Vous pouvez également ajouter des dossiers et des fichiers se trouvant en dehors du projet en cours. Pour plus d'informations, voir « Liaison à des ressources situées en dehors de l'espace de travail du projet » à la page 75.

Vous pouvez ajouter des dossiers à un projet selon vos besoins. Par exemple, vous pouvez créer un dossier pour stocker tous vos modèles de données. Vous pouvez également organiser tous les éléments qui constituent l'aspect visuel de votre application.

1. Dans l'Explorateur de packages, sélectionnez Fichier > Nouveau > Dossier.
2. Si l'espace de travail contient plusieurs projets, sélectionnez le projet à ajouter au dossier autonome.

   Si vous créez le dossier dans le dossier du chemin d'accès source, il est considéré comme un nom de package. Vous pouvez ensuite placer les fichiers source dans le dossier reconnu par le compilateur.

   Si vous le créez en dehors du dossier du chemin d'accès source, vous pourrez l'ajouter ultérieurement à ce chemin afin de le transformer en racine d'une structure de packages. Une fois que vous avez terminé cette procédure, sélectionnez Projet > Propriétés, puis Chemin d'accès à la génération Flex. Cliquez sur Ajouter un dossier et accédez au dossier que vous venez de créer.
3. Entrez le nom du dossier, puis cliquez sur Terminer.

Suppression de dossiers et de fichiers

La suppression de dossiers et de fichiers du projet les élimine de l'espace de travail et, par conséquent, du système de fichiers.

Remarque : *la suppression d'une ressource liée ne supprime du projet que le lien, pas la ressource (voir « Liaison à des ressources situées en dehors de l'espace de travail du projet » à la page 75). Cependant, si vous avez créé un lien à un dossier et que vous supprimez un ou plusieurs de ses fichiers, ceux-ci sont supprimés du système de fichiers.*

1. Dans l'Explorateur de packages, sélectionnez la ressource à supprimer.
2. Sélectionnez Edition > Supprimer ou appuyez sur la touche Supprimer, puis cliquez sur Oui.

La ressource est supprimée du système de fichiers.

Création de fichiers d'application MXML

1 Dans l'Explorateur de packages, sélectionnez le projet auquel vous souhaitez ajouter le fichier d'application MXML.

2 Dans le menu contextuel de l'Explorateur de packages, sélectionnez Nouveau > Application MXML.

3 Le fichier d'application est créé dans le dossier src par défaut. Vous pouvez sélectionner un autre dossier du projet, le cas échéant.

4 Indiquez le nom du fichier d'application et sélectionnez une présentation Spark, si nécessaire. Pour plus d'informations sur les présentations Spark, voir About Spark layouts (A propos des présentations Spark).

5 Les options suivantes s'appliquent uniquement aux projets mobiles :

a Spécifiez un ID d'application.

b Spécifiez le modèle d'application.

Voir « Choix d'un modèle d'application » à la page 226.

c Spécifiez les droits des applications mobiles pour la plateforme Google Android.

Voir « Choix des autorisations d'une application mobile » à la page 227.

d Spécifiez les paramètres de plateforme.

Voir « Choix des paramètres de plateforme » à la page 228.

e Spécifiez les paramètres d'application.

Voir « Choix des paramètres d'application » à la page 228.

6 Cliquez sur Terminer.

Création d'un fichier ActionScript

Vous pouvez faire appel à un assistant Flash Builder afin de créer rapidement un fichier ActionScript pour les projets Flex et ActionScript.

1 Sélectionnez Fichier > Nouveau > Fichier ActionScript.

2 Spécifiez un nom de package pour votre fichier ActionScript. Si vous ne spécifiez aucun nom de package, le package par défaut est sélectionné. Si vous spécifiez un dossier de package inexistant, l'assistant le crée.

3 Nommez le fichier ActionScript.

4 Cliquez sur Terminer.

Création d'une classe ActionScript

Vous pouvez faire appel à un assistant Flash Builder pour créer rapidement des classes ActionScript pour les projets Flex et ActionScript. Cet assistant vous permet également de générer facilement des stubs pour des fonctions devant être implémentées.

1 Sélectionnez Fichier > Nouveau > Classe ActionScript.

2 Dans la boîte de dialogue, spécifiez les propriétés de base de la nouvelle classe, puis cliquez sur Terminer.

Une fois que vous avez cliqué sur Terminer, Flash Builder enregistre le fichier dans le package spécifié et le rouvre dans l'éditeur de code.

3 Rédigez la définition de la classe ActionScript.

Pour plus d'informations, voir Create simple visual components in ActionScript (Création de composants visuels simples dans ActionScript).

Création d'une interface ActionScript

Vous pouvez faire appel à un assistant Flash Builder pour créer rapidement des interfaces ActionScript pour les projets Flex et ActionScript. Une *interface* est un ensemble de constantes et de méthodes que différentes classes peuvent partager.

1 Sélectionnez Fichier > Nouveau > Interface ActionScript.

2 Spécifiez les propriétés de base de la nouvelle interface dans la boîte de dialogue, puis cliquez sur Terminer.

3 Ajoutez à l'interface ActionScript les éventuelles constantes ou méthodes que les différentes classes partagent.

Création d'un composant ActionScript habillable

Vous pouvez faire appel à un assistant Flash Builder afin de créer rapidement des composants ActionScript pour les projets Flex.

1 Cliquez sur Fichier > Nouveau > Composant habillable ActionScript.

2 Spécifiez les propriétés de base du nouveau composant habillable, puis cliquez sur Terminer.

Flash Builder crée une classe ActionScript étendue depuis le composant de base ou le composant habillable. Par défaut, le composant de base est `spark.components.supportClasses.SkinnableComponent`. Vous pouvez toutefois le modifier sur tout autre composant correspondant à une sous-classe du composant habillable.

Vous pouvez également ajouter toutes les interfaces ActionScript que le nouveau composant habillable peut implémenter. Pour plus d'informations sur la création d'interfaces ActionScript, voir « Création d'une interface ActionScript » à la page 71.

Le composant `SkinnableComponent` est créé dans l'emplacement et le package sélectionné. Par défaut, le logiciel ajoute automatiquement une référence au composant dans le fichier MXML principal.

Utilisation d'opérateurs ActionScript

Flash Builder prend en charge la programmation ActionScript multithread ; vous pouvez ainsi exécuter un programme ActionScript en tant qu'opérateur. Les opérateurs s'exécutent en parallèle du programme ActionScript principal et peuvent communiquer avec ce dernier et avec d'autres opérateurs ActionScript.

Les opérateurs ActionScript vous permettent de créer des jeux et applications plus réactifs en envoyant des tâches et calculs sur les opérateurs en arrière-plan. Les opérateurs s'exécutent simultanément, de sorte que vos jeux sont plus réactifs. Pour en savoir plus sur les opérateurs ActionScript, reportez-vous à l'article Utilisation de programmes de travail à des fins de simultanéité.

Pour une présentation animée des opérateurs ActionScript, regardez cette vidéo du spécialiste d'Adobe Lee Brimelow.

Création d'un opérateur ActionScript

Vous pouvez utiliser un assistant dans Flash Builder pour créer un opérateur ActionScript pour vos projets Flex et ActionScript en suivant les étapes suivantes :

1 Sélectionnez Fichier > Nouveau > Opérateur ActionScript.

2 Spécifiez un nom de package pour votre opérateur ActionScript. Si vous ne spécifiez aucun package, la classe est déclarée dans le package par défaut. Si vous spécifiez un dossier de package inexistant, l'assistant le crée.

3 Nommez le fichier de l'opérateur ActionScript.

4 Les opérateurs ActionScript s'étendent depuis la classe `flash.display.Sprite` par défaut. Vous pouvez donc sélectionner uniquement les classes étendues depuis Sprite.

5 Ajoutez toute interface contenant des constantes et des méthodes que vous souhaitez utiliser dans l'opérateur ActionScript.

6 Spécifiez l'une des options de génération de code suivantes :

Générer un constructeur à partir de la superclasse Génère un constructeur avec un appel super(). Si le constructeur de superclasse accepte des arguments, ils sont inclus dans le constructeur généré et transmis dans l'appel super().

Générer des fonctions héritées d'interfaces Génère des stubs de fonction pour chaque méthode d'interface. Les stubs comprennent une instruction return qui renvoie une valeur null (ou 0 ou false pour les types primitifs) pour permettre la compilation des stubs.

Générer des commentaires Insère "`//TODO: implement function`" dans les constructeurs ou fonctions générés.

7 Spécifiez l'utilisation de l'opérateur ActionScript. Pour plus d'informations, voir « Spécification de l'utilisation de l'opérateur ActionScript » à la page 72.

8 Cliquez sur Terminer pour enregistrer l'opérateur en tant qu'extension de nom de fichier dans le package spécifié et l'ouvrir dans l'éditeur de code.

9 Ecrivez le code de votre opérateur ActionScript.

Flash Builder fournit des modèles de code d'opérateur ActionScript qui vous permettent d'ajouter rapidement du code à un fichier d'opérateur. Pour plus d'informations sur les modèles de code, voir « Modèles de code » à la page 41.

Pour convertir une classe ActionScript existante en opérateur, suivez ces étapes :

1 Dans l'Explorateur de packages, sélectionnez la classe ActionScript à convertir en opérateur ActionScript.

2 Dans le menu contextuel de l'Explorateur de packages, sélectionnez Définir en tant qu'opérateur ActionScript.

Lorsque vous créez un opérateur ActionScript, le fichier d'opérateur est ajouté à la liste d'opérateurs ActionScript dans la boîte de dialogue des propriétés du projet.

Spécification de l'utilisation de l'opérateur ActionScript

Lorsque vous créez et modifiez un opérateur ActionScript ou convertissez une classe ActionScript en opérateur, vous pouvez spécifier l'utilisation de l'opérateur.

Vous pouvez choisir d'intégrer l'opérateur ou de le charger de manière externe en sélectionnant l'une des options suivantes :

Opérateur intégré : génère un fichier SWF pour la classe d'opérateur ActionScript à intégrer. Le fichier SWF est généré dans le dossier workerswfs de votre projet, au lieu d'un dossier de sortie sélectionné par l'utilisateur.

Si vous choisissez d'intégrer l'opérateur, Flash Builder génère la classe Workers.as. La classe générée contient du code permettant d'intégrer le fichier SWF et de créer une méthode get. Vous pouvez utiliser la méthode get pour accéder à la classe ByteArray de l'opérateur ActionScript intégré.

Important : *ne modifiez pas le contenu de la classe générée automatiquement.*

Par exemple, si vous créez un opérateur ActionScript intégré que vous nommez MyWorker, le code suivant est généré :

```
public class Workers
{
    [Embed(source="../workerswfs/MyWorker.swf", mimeType="application/octet-stream")]
    private static var MyWorker_ByteClass:Class;
    public static function get MyWorker():ByteArray
    {
        return new MyWorker_ByteClass();
    }
}
```

Dès que vous ajoutez, renommez, déplacez ou supprimez un opérateur ActionScript, Flash Builder modifie le code généré dans le fichier Workers.as.

Si le fichier Workers.as existe dans le projet, Flash Builder génère la classe avec les noms de fichier suivants dans l'ordre spécifié :

1 WorkersManager.as

2 WorkersHolder.as

Si des fichiers portant le même nom existent, Flash Builder génère la classe avec un nom de fichier unique en ajoutant des numéros aux noms de fichiers de manière séquentielle. Par exemple, Workers1.as, WorkersManager1.as, WorkersHolder1.as, Workers2.as, etc. Vous pouvez renommer ces fichiers si vous le souhaitez, Flash Builder enregistre et utilise le nouveau nom de fichier pour générer le code.

Si vous exécutez ou déboguez un projet contenant un opérateur ActionScript intégré, le dossier workerswfs contient les fichiers SWF de débogage. Si vous exportez une version validée, les fichiers SWF de version remplacent les fichiers SWF de débogage dans le dossier workerswfs.

Opérateur chargé à l'extérieur : génère le fichier SWF de la classe d'opérateur ActionScript dans un dossier de sortie sélectionné par l'utilisateur.

Si vous exécutez ou déboguez un projet avec un opérateur ActionScript chargé à l'extérieur, les fichiers SWF de débogage sont générés dans le dossier bin-debug et les fichiers SWF de version dans le dossier release.

Gestion des fichiers d'opérateur ActionScript

La boîte de dialogue Propriétés du projet vous permet de gérer les fichiers d'opérateur de votre projet Flex ou ActionScript.

1 Dans l'Explorateur de packages, sélectionnez un projet.

2 Sélectionnez Projet > Propriétés dans le menu principal ou sélectionnez Propriétés dans le menu contextuel. La boîte de dialogue Propriétés du projet s'ouvre.

3 Dans le panneau de gauche, sélectionnez les opérateurs ActionScript.

4 Ajoutez, modifiez ou supprimez les fichiers d'opérateur selon vos besoins et cliquez sur OK.

Ajout d'un opérateur ActionScript ou conversion d'une classe ActionScript en opérateur

Flash Builder fournit une option permettant de convertir les classes ActionScript en opérateurs dans un projet.

1 Accédez à la boîte de dialogue Propriétés du projet et sélectionnez des opérateurs ActionScript.

2 Cliquez sur Ajouter.

3 Cliquez sur Parcourir pour sélectionner la classe à définir en tant qu'opérateur ActionScript. Flash Builder affiche uniquement les classes qui s'étendent depuis les classes Sprite et Public.

4 Spécifiez l'utilisation de l'opérateur. Pour plus d'informations, voir « Spécification de l'utilisation de l'opérateur ActionScript » à la page 72.

Vous pouvez également sélectionner l'option Définir en tant qu'opérateur ActionScript du menu contextuel des projets de l'Explorateur de packages, et spécifier les préférences d'utilisation de l'opérateur.

Recherche de ressources dans le workbench

Flash Builder intègre un outil de recherche qui vous permet de localiser rapidement des ressources.

Pour rechercher le document qui est actuellement ouvert dans l'éditeur, procédez comme suit :

1. Ouvrez le document sur lequel la recherche doit porter.
2. Utilisez l'une des méthodes suivantes :
   - Appuyez sur Ctrl+F (Windows) ou Commande+F (Mac OS).
   - Sélectionnez Editer > Rechercher/Remplacer.
3. Saisissez la chaîne de texte à localiser.
4. (Facultatif) Entrez la chaîne de texte de remplacement.
5. (Facultatif) Définissez les critères de recherche avancés.
6. Cliquez sur Rechercher, Remplacer, Remplacer tout ou Remplacer/Rechercher.

   Une fois localisée, la chaîne de texte est mise en surbrillance et éventuellement remplacée.

Remarque : *pour procéder à une recherche incrémentielle, appuyez sur Ctrl+J (Windows) ou Commande+J (Mac OS).*

Pour rechercher toutes les ressources dans l'espace de travail, Flash Builder comporte des fonctions de recherche avancées plus puissantes que les fonctions de recherche/remplacement. Flash Builder permet de rechercher et de repérer les références ou les déclarations vers les identifiants dans les fichiers ActionScript ou MXML, les projets ou les espaces de travail. Pour plus d'informations, voir « Recherche de références et restructuration du code » à la page 40.

Recherche de fichiers

❖ Sélectionnez Rechercher > Rechercher pour effectuer des recherches de fichiers complexes.

Remarque : *cliquez sur Personnaliser pour définir les types d'onglet de recherche disponibles dans la boîte de dialogue Rechercher.*

Utilisation de la vue Recherche

La vue Recherche affiche les résultats de votre recherche.

Ouverture d'un fichier de la liste

❖ Cliquez deux fois sur le fichier.

Suppression d'un fichier de la liste

❖ Sélectionnez le fichier à supprimer, puis cliquez sur Supprimer les occurrences sélectionnées.

Suppression de tous les fichiers de la liste

❖ Cliquez sur Supprimer toutes les occurrences.

Navigation entre les fichiers correspondants

❖ Cliquez sur Afficher l'occurrence suivante ou Afficher l'occurrence précédente.

Affichage des recherches précédentes

❖ Cliquez sur la flèche en regard de l'option Afficher les recherches précédentes et sélectionnez une recherche dans la liste déroulante.

Retour à la vue Recherche après sa fermeture

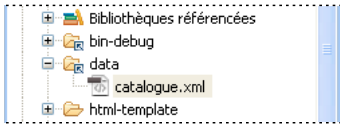
1 Sélectionnez Fenêtre > Affichage d'une vue > Autres.

2 Développez la catégorie Généralités, sélectionnez Recherche, puis cliquez sur OK.

Liaison à des ressources situées en dehors de l'espace de travail du projet

Vous pouvez créer des liens avec des ressources situées en dehors du projet et de l'emplacement de l'espace de travail ainsi qu'avec des dossiers et des fichiers situés n'importe où dans le système de fichiers. Cette option est utile en cas de partage des ressources entre différents projets. Vous pouvez par exemple partager une bibliothèque de composants Flex ou ActionScript personnalisés entre plusieurs projets Flex.

Les dossiers contenant des ressources liées sont signalés dans l'Explorateur de packages (comme l'illustre l'exemple suivant). Vous pouvez ainsi distinguer les ressources normales des ressources liées.



Parmi d'autres exemples de liaison de ressources, citons les dossiers de fichiers d'images ou encore le cas des dossiers de sortie ne se trouvant pas dans le dossier racine du projet.

Lorsque les ressources sont partagées, les modifications que vous apportez aux dossiers et fichiers source affectent tous les projets qui leur sont liés. Soyez prudent lorsque vous supprimez de vos projets des ressources liées. Dans certains cas, vous supprimez simplement la référence au lien ; dans d'autres, vous supprimez la source même. Pour plus d'informations, voir « Suppression de dossiers et de fichiers » à la page 69.

Remarque : *il est recommandé de lier d'autres projets aux projets de bibliothèque. Les ressources liées sont recommandées seulement pour les bibliothèques tierces avec un fichier SWC.*

Liaison à des ressources situées en dehors de l'espace de travail du projet

1 Dans l'Explorateur de packages, sélectionnez le projet auquel vous souhaitez ajouter des ressources liées.

2 Sélectionnez Fichier > Nouveau > Dossier (ou Fichier).

3 Sélectionnez le projet ou le dossier du projet auquel vous souhaitez ajouter les ressources liées.

4 Saisissez le nom du dossier ou du fichier. Il peut être différent du nom du dossier ou du fichier avec lequel vous créez un lien.

5 Cliquez sur le bouton Options avancées.

6 Sélectionnez Lier au dossier dans le système de fichiers. Saisissez ou recherchez l'emplacement de la ressource.

7 Cliquez sur Terminer pour lier la ressource au projet.

Utilisation d'une variable de chemin à relier aux ressources

Vous pouvez établir des liens vers des ressources en définissant des variables de chemin. Vous évitez ainsi de devoir saisir le chemin complet d'un dossier local ou d'un dossier réseau dans lequel vous enregistrez les fichiers. Pour plus d'informations, voir « Création d'une variable de chemin » à la page 93.

1 Dans l'Explorateur de packages, sélectionnez le projet auquel vous souhaitez ajouter des ressources liées.

Les variables de chemin peuvent également intervenir dans certains paramètres de projet, tels que le chemin d'accès à la bibliothèque et le chemin d'accès à la source.

2 Sélectionnez Fichier > Nouveau > Dossier (ou Fichier, si vous voulez ajouter des fichiers).

3 Sélectionnez le projet ou le dossier du projet auquel vous souhaitez ajouter les ressources liées.

4 Cliquez sur le bouton Options avancées.

5 Sélectionnez Lier au dossier dans le système de fichiers. Cliquez sur le bouton Variables.

6 Sélectionnez une variable de chemin définie ou cliquez sur Nouveau pour créer une variable de chemin.

Si vous avez sélectionné une variable de chemin définie, passez à l'étape 9. Si vous avez cliqué sur Nouveau, la boîte de dialogue Nouvelle variable s'affiche.

7 Saisissez le nom de la variable de chemin et saisissez ou recherchez l'emplacement du fichier ou du dossier.

Cliquez sur OK pour créer la variable de chemin.

8 Sélectionnez la nouvelle variable de chemin dans la boîte de dialogue Sélection d'une variable de chemin, puis cliquez sur OK.

9 Cliquez sur Terminer pour terminer la création du lien vers la ressource.

Vous pouvez également définir et gérer des variables de chemin dans les préférences du workbench de Flash Builder (dans la boîte de dialogue Préférences, sélectionnez Général > Espace de travail > Ressources liées).

Déplacement ou partage de ressources entre les projets d'un espace de travail

Déplacement de ressources

Lorsque vous travaillez avec plusieurs projets dans un espace de travail, vous pouvez déplacer des ressources d'un projet vers un autre.

1 Dans l'Explorateur de packages, sélectionnez la ressource à déplacer.

2 Effectuez l'une des opérations suivantes :

- Faites glisser la ressource vers un nouveau projet.
- Coupez et collez la ressource dans un autre projet.

Lors du déplacement de ressources entre les projets, vous pouvez choisir de mettre à jour les références.

Remarque : *vous pouvez déplacer des ressources normales et des ressources liées. Pour plus d'informations sur la liaison de ressources, voir « Liaison à des ressources situées en dehors de l'espace de travail du projet » à la page 75.*

Partage de ressources

Pour partager des ressources entre plusieurs projets, placez-les dans des dossiers qui pourront ensuite être reliés aux différents projets à l'aide du chemin d'accès source de ces derniers. Il s'agit de la meilleure méthode de partage de ressources, telles que les classes, les composants MXML et les images. Les mises à jour de ces ressources sont immédiatement disponibles pour tous les projets qui les utilisent. Lorsque les projets sont compilés, les ressources partagées sont ajoutées au fichier SWC.

Ajout d'un dossier de ressources externes au chemin d'accès source

1 Sélectionnez un projet dans l'Explorateur de packages.

2 Sélectionnez Projet > Propriétés > Chemin d'accès à la génération Flex (ou Chemin d'accès à la génération ActionScript si vous utilisez un projet ActionScript).

3 Sur la page des propriétés du chemin de génération, cliquez sur l'onglet Chemin source.

4 Cliquez sur le bouton Ajouter un dossier.

5 Saisissez ou recherchez le chemin du dossier, puis cliquez sur OK.

Le dossier est ajouté au chemin source.

L'onglet de propriétés Chemin source vous permet également de modifier, supprimer ou changer l'ordre des éléments du chemin source.

Les dossiers qui sont ajoutés au chemin source sont signalés dans l'Explorateur de packages.

Actualisation des ressources dans l'espace de travail

A mesure que vous modifiez, ajoutez ou supprimez les ressources d'un projet, le workbench actualise automatiquement les différentes vues qui affichent ces ressources. La suppression d'un fichier du projet par exemple est immédiatement répercutée dans l'Explorateur de packages.

Vous pouvez également modifier des ressources situées en dehors de Flash Builder, directement dans le système de fichiers. Ces modifications ne sont visibles dans Flash Builder qu'après actualisation de l'espace de travail.

Par défaut, l'espace de travail de la configuration autonome de Flash Builder est actualisé automatiquement. Cette option peut être modifiée dans les préférences de Flash Builder. Ouvrez la boîte de dialogue Préférences et sélectionnez Général > Espace de travail. Vous pouvez également modifier le comportement par défaut de Flash Builder et désactiver l'actualisation automatique de l'espace de travail.

Actualisation manuelle de l'espace de travail

❖ Dans le menu contextuel de l'Explorateur de packages, sélectionnez Actualiser. Toutes les ressources de projet dans l'espace de travail sont actualisées.

Désactivation de l'actualisation automatique

1 Ouvrez la boîte de dialogue Préférences et sélectionnez Général > Espace de travail.

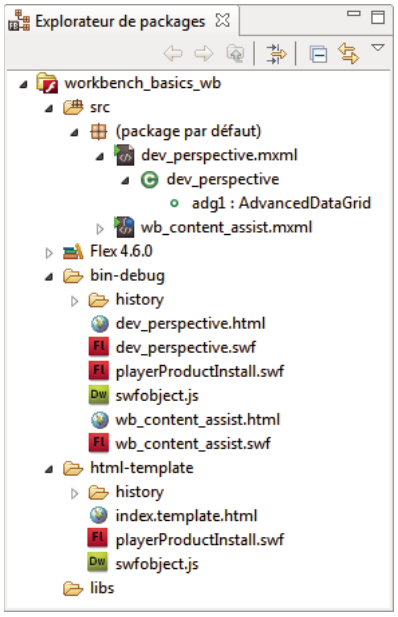
2 Désélectionnez l'option Actualiser automatiquement.

Gestion de projets

L'Explorateur de packages permet d'ajouter des ressources à un projet, d'importer des ressources, d'exporter des projets ainsi que de déplacer et de supprimer des ressources.

Projets dans l'Explorateur de packages

Tous les projets d'un espace de travail s'affichent dans l'Explorateur de packages, comme l'illustre l'exemple suivant. L'Explorateur de packages présente une arborescence des projets dans une vue à la fois physique et logique (plate). Cette vue vous permet de gérer les projets en ajoutant et en supprimant des ressources (dossier et fichiers), en important des ressources externes et en créant des liens vers ces ressources ainsi qu'en déplaçant les ressources vers d'autres projets dans l'espace de travail.



Principales caractéristiques de l'Explorateur de packages :

- L'affichage des packages ActionScript peut se présenter de manière hiérarchique ou plate.

  Utilisez le menu de l'Explorateur de packages pour spécifier la présentation des packages.

- Les bibliothèques des projets Flex et les bibliothèques ActionScript sont représentées en deux nœuds de niveau supérieur.

  Les bibliothèques de projets Flex présentent un nœud pour le SDK Flex et un autre pour les bibliothèques référencées. De la même façon, les bibliothèques de projets ActionScript présentent un nœud pour le SDK AIR et un autre pour les bibliothèques référencées.

  Vous pouvez développer le contenu d'une bibliothèque et ouvrir les éditeurs pour afficher les pièces jointes.

- Les symboles d'erreur et d'avertissement apparaissant au niveau des nœuds de l'Explorateur de packages signalent d'éventuels problèmes dans un package.

- Vous pouvez restreindre les projets et ressources visibles.

  Vous pouvez créer un jeu de documents (collection de ressources), définir des filtres d'affichage et trier les ressources par nom et type. Ces options sont disponibles dans les menus de l'Explorateur de packages. Pour plus d'informations sur la modification des vues, voir « Personnalisation du workbench » à la page 256.

- Vous pouvez développer les fichiers ActionScript, MXML et CSS ainsi qu'afficher leur contenu sous forme d'arborescence.

Dans l'Explorateur de packages, vous pouvez ouvrir les ressources du projet pour les modifier. Par exemple, vous pouvez modifier des fichiers MXML et ActionScript dans les blocs `<fx:Script>` et des fichiers CSS dans les blocs `<fx:Style>`. Pour plus d'informations sur l'utilisation des éditeurs Flash Builder, voir « Editeurs Flash Builder » à la page 12 et « Création d'interfaces utilisateur » à la page 172.

Ensuite, vous ajoutez des projets, des fichiers et des dossiers, puis vous les organisez et les gérez selon vos besoins.

La plupart des commandes de menu que vous utilisez dans la vue Explorateur de packages sont également disponibles dans le menu contextuel de la vue.

Voir aussi

« Ressources des projets » à la page 68

Déplacement d'un projet d'un espace de travail vers un autre

Vous pouvez faire appel à une association d'opérations de suppression et d'importation pour déplacer un projet d'un espace de travail vers un autre. Lorsque vous supprimez un projet d'un espace de travail, vous pouvez le supprimer de l'espace de travail mais le laisser dans le système de fichiers. Après avoir supprimé un projet d'un espace de travail, vous pouvez l'importer dans un autre.

Spécification du SDK d'un projet

Projets Flex

Lors de la création d'un projet Flex, Flash Builder utilise le SDK par défaut disponible dans l'installation de Flash Builder. Vous pouvez toutefois choisir d'utiliser un SDK Flex non disponible dans l'installation de Flash Builder.

Par exemple, si vous souhaitez compiler votre projet à l'aide d'un SDK Apache Flex, effectuez les étapes suivantes :

1. Téléchargez le SDK Apache Flex à l'adresse http://www.adobe.com/go/apacheflex_download.
2. Ajoutez le SDK téléchargé à votre installation Flash Builder en accédant à Projet > Propriétés > Compilateur Flex et en cliquant sur Configurer les SDK Flex.
3. Dans la boîte de dialogue des propriétés du projet, sélectionnez Compilateur Flex > Utiliser un SDK spécifique et sélectionnez le SDK à utiliser.

Pour plus d'informations sur l'installation et l'utilisation du SDK Flex, voir « SDK Flex installés » à la page 266.

Projets ActionScript

Les projets ActionScript utilisent le SDK AIR. Pour plus d'informations sur le SDK AIR, voir « SDK AIR installé » à la page 266.

Remarque : *vous ne pouvez pas utiliser un SDK Flex avec un projet ActionScript.*

Suppression d'un projet

Lorsque vous supprimez un projet, vous le retirez de l'espace de travail en cours. Vous pouvez également le supprimer simultanément du système de fichiers.

Au lieu de supprimer le projet de l'espace de travail, vous pouvez le fermer. La fermeture du projet vous permet de conserver une référence à ce projet dans l'espace de travail tout en libérant des ressources système.

1. Dans l'Explorateur de packages, sélectionnez le projet à supprimer.
2. Dans le menu principal, sélectionnez Editer > Supprimer.

3 Sélectionnez une option.

Supprimer également le contenu sous le répertoire : supprime définitivement le projet de l'espace de travail et du système de fichiers.

Ne pas supprimer le contenu : supprime le projet de l'espace de travail sans le supprimer du système de fichiers.

Fermeture et ouverture de projets

Pour économiser de la mémoire et accélérer la génération sans supprimer un projet, vous pouvez le fermer. Lorsque vous fermez un projet, vous réduisez le projet et ses ressources, mais son nom reste visible dans l'Explorateur de packages. Un projet fermé nécessite moins de mémoire qu'un projet ouvert et est exclu des générations. Vous pouvez facilement rouvrir un projet fermé.

1 Dans l'Explorateur de packages, sélectionnez le projet à fermer ou rouvrir.

2 Dans le menu contextuel de l'Explorateur de packages, sélectionnez Fermer le projet ou Ouvrir le projet.

Changement du fichier de l'application principale

Lorsque vous créez un projet Flex, un fichier de l'application principale est créé. Par défaut, il est nommé d'après le projet. Le fichier de l'application principale est le point d'accès à vos applications et est la base du fichier SWF de l'application. Cependant, à mesure que vous ajoutez des fichiers à l'application, vous pouvez désigner un fichier différent comme fichier de l'application principale.

Si vous préférez définir plusieurs fichiers d'application de manière à ce que chaque fichier d'application soit généré dans un fichier SWF distinct, voir « Gestion des fichiers d'application du projet » à la page 80.

1 Dans l'Explorateur de packages, sélectionnez le fichier d'application MXML que vous voulez définir comme fichier de l'application principale.

2 Dans le menu contextuel de l'Explorateur de packages, sélectionnez Définir comme application par défaut.

Vous pouvez gérer les fichiers d'application du projet en sélectionnant Projet > Propriétés > Applications Flex (ou Applications ActionScript si vous utilisez un projet ActionScript).

Gestion des fichiers d'application du projet

Un projet possède généralement un seul fichier de l'application principale, qui sert de point d'accès à l'application. Le compilateur Flash Builder utilise ce fichier pour générer le fichier d'application SWF.

Une application complexe peut par exemple être dotée de nombreux composants MXML personnalisés qui représentent des éléments d'application distincts mais interdépendants. Vous pouvez créer un fichier d'application contenant un composant personnalisé et ensuite le générer, l'exécuter et le tester séparément.

Par défaut, vous pouvez exécuter l'application à chaque ajout d'un fichier d'application MXML au projet Flex. Ce fichier est alors ajouté à la liste des fichiers d'application du projet. Tous les fichiers définis comme fichiers d'application doivent se trouver dans le dossier source du projet.

Vous pouvez gérer la liste des fichiers d'application en sélectionnant un projet et en affichant ses propriétés.

1 Dans l'Explorateur de packages, sélectionnez un projet.

2 Sélectionnez Projet > Propriétés dans le menu principal ou sélectionnez Propriétés dans le menu contextuel.

3 Dans la boîte de dialogue Propriétés du projet, sélectionnez Applications Flex (ou Applications ActionScript si vous utilisez un projet ActionScript).

4 Ajoutez et supprimez les fichiers d'applications selon vos besoins. Cliquez sur OK.

Changement de l'espace de travail

Vous ne pouvez travailler que dans un seul espace de travail à la fois. Lorsque vous installez et exécutez Flash Builder pour la première fois, un message vous invite à créer un espace de travail qui devient l'espace de travail par défaut. Vous pouvez créer d'autres espaces de travail et passer de l'un à l'autre en sélectionnant l'espace de travail voulu au lancement de Flash Builder ou en choisissant Fichier > Commuter l'espace de travail.

Exportation et importation de projets

Flash Builder permet d'exporter et importer des projects aux formats FXP et ZIP et depuis des répertoires de fichier ouverts. Le format FXP est un format d'archive contenant les dossiers, les fichiers et les métadonnées d'un projet. Flash Builder exporte les projets Flex au format FXP et les projets de bibliothèque Flex au format FXPL. Les projets ActionScript peuvent être exportés uniquement vers des fichiers archive, généralement au format ZIP.

L'importation et l'exportation de projets permet de transférer des projets de façon sûre et sécurisée entre différents ordinateurs et utilisateurs.

Remarque : *vous pouvez également faire appel à l'assistant d'exportation Eclipse pour exporter des projets Flex et des projets de bibliothèque Flex au format ZIP (utilisés dans Flex Builder 3) ou vers un autre format d'archive.*

Exportation de projets

Flash Builder peut exporter de nombreux projets en plus des fonctions d'exportation Eclipse.

Exportation d'un projet Flex ou d'un projet de bibliothèque Flex au format FXP

Certains projets Flex devront faire l'objet d'un traitement spécial à l'importation. Voir « Projets nécessitant un traitement spécial » à la page 84

1 Dans Flash Builder, sélectionnez Fichier > Exporter un projet Flash Builder.

   Vous pouvez également faire appel au menu contextuel du projet dans l'Explorateur de packages. Sélectionnez Exporter > Flash Builder > Projet Flash Builder.

2 Dans l'assistant Exportation d'un projet Flex, sélectionnez le projet à exporter.

   La liste déroulante Projet répertorie tous les projets disponibles.

3 Recherchez l'emplacement vers lequel vous souhaitez exporter le fichier FXP dans votre système de fichiers.

4 (Facultatif) Activez l'option Valider la compilation du projet.

   Sélectionnez cette option pour obtenir confirmation que le projet sera compilé sans erreurs. La présence d'erreurs ne vous empêchera pas de poursuivre l'exportation.

5 Cliquez sur Terminer.

   Pour les projets sur serveur, les chemins absolus vers les ressources du serveur sont enregistrés sous forme de variables de chemin. Lorsque vous importerez ultérieurement le projet, vous devrez spécifier des valeurs pour les variables de chemin.

Exportation d'un projet ActionScript au format ZIP (ou vers tout autre format d'archive)

1 Dans Flash Builder, sélectionnez Fichier > Exporter > Autre.

2 Dans l'assistant d'exportation, sélectionnez Général > Fichier archive et cliquez sur Suivant.

3 Sélectionnez le projet et les fichiers à exporter :

- Dans le volet de gauche, développez le projet pour en sélectionner les dossiers à inclure dans l'exportation.
- Dans le volet de droite, indiquez pour chaque dossier sélectionné les fichiers à exporter.

4 Recherchez un emplacement dans lequel enregistrer le projet exporté et attribuez un nom au fichier.

5 Définissez les options pour le fichier archive, puis cliquez sur Terminer.

Importation de projets

Flash Builder peut importer des projets Flex, des projets de bibliothèque Flex ainsi que des projets ActionScript précédemment exportés de Flash Builder. Vous pouvez importer plusieurs versions d'un même projet Flex ou d'un même projet de bibliothèque Flex.

Importation d'un projet Flex ou d'un projet de bibliothèque Flex

Vous pouvez importer un projet à partir d'un fichier FXP exporté ou en accédant au dossier contenant le projet.

1 Dans le menu de Flash Builder, sélectionnez Fichier > Importer.

Les fonctions d'importation d'un projet sont également disponibles dans le menu contextuel de l'Explorateur de packages.

2 (Dossier du projet) Si l'importation est effectuée à partir d'un dossier de projet existant, activez l'option Dossier du projet et recherchez le dossier contenant le projet.

3 (Fichier FXP) Si vous effectuez une importation à partir d'un fichier FXP, sélectionnez Fichier et naviguez jusqu'à l'emplacement du fichier.

Si le fichier FXP contient plusieurs projets, vous pouvez sélectionnez individuellement les projets à importer.

4 (Projet de bibliothèque ou projet FXPL) Si vous importez un projet de bibliothèque, vous avez la possibilité d'importer le contenu dans un projet existant.

5 (Fichier FXP) Si un projet du même nom existe dans l'espace de travail, spécifiez la méthode d'importation.

- Importer en tant que nouveau projet : Flash Builder identifie le projet en ajoutant un chiffre à son nom. Les versions antérieures du projet sont conservées.

  Dans le champ Extraire le nouveau projet vers, spécifiez l'emplacement dans lequel vous souhaitez extraire le fichier. Cet emplacement est généralement un répertoire de l'espace de travail de Flash Builder représentant un dossier du projet. Vous pouvez spécifier un nouveau dossier de projet ou remplacer un projet existant.

- Remplacer un projet existant : sélectionnez le projet que vous souhaitez remplacer. La version antérieure du projet est définitivement supprimée.

6 (Variables de chemin) Si vous importez un projet définissant des variables de chemin, mettez à jour les variables de chemin pour le projet.

Les projets compilés pour ColdFusion, PHP, LiveCycle Data Services ou d'autres technologies de serveur font appel aux variables de chemin pour accéder au serveur Web et aux ressources du serveur. D'autres projets peuvent présenter des variables de chemin personnalisées.

Sélectionnez chaque variable de chemin et attribuez-lui une valeur valide.

7 Cliquez sur Terminer.

8 (Projets pour serveurs PHP) Si vous importez un projet pour serveur d'applications de type PHP, installez ou mettez à jour l'installation Zend.

La boîte de dialogue Zend vous guide au long du processus.

Remarque : *si vous annulez le processus dans la boîte de dialogue Zend, installez ou mettez à jour manuellement Zend Framework. L'accès aux services PHP est impossible tant que Zend Framework n'est pas installé et configuré correctement. Pour plus d'informations sur l'installation, la configuration et le dépannage de votre installation Zend Framework, voir Installation de Zend Framework.*

9 (Projets pour serveurs) Déployez des services.

   a Placez manuellement des services sous la racine Web du serveur. Utilisez la structure de répertoire utilisée dans le projet d'origine.

   b Dans la vue Données/Services, sélectionnez Actualiser dans le menu contextuel d'un service.

Importation d'un projet Flex 3

Vous pouvez importer un projet Flex 3 dans Flash Builder 4.6 en utilisant le mode de compatibilité Flex 3. Dans ce cas, les espaces de noms et les composants Flex 3 demeureront inchangés. Vous pourrez cependant profiter du compilateur disponible avec Flex 4.6.

Les documents créés en mode de compatibilité Flex 3 utilisent les composants MX et les espaces de noms suivants :

```
mx="http://www.adobe.com/2006/mxml"
```

1 Dans Flash Builder, sélectionnez Fichier > Importer un projet Flex.

2 Recherchez le fichier ZIP du projet Flex 3 précédemment exporté ou le dossier du projet Flex 3.

3 Cliquez sur Terminer.

4 Dans la boîte de dialogue Choisir la version du SDK Flex, vérifiez que le SDK Flex 4 a été activé. Sélectionnez Utiliser le mode de compatibilité Flex 3.

5 Cliquez sur OK.

Pour plus d'informations sur l'utilisation de Flash Builder 4 pour vos projets Flex 3 existants, voir l'article du pôle de développement Adobe : *Migration de projets Flex existants de Flex Builder 3 vers Flash Builder 4*.

Importation d'un projet ActionScript

Utilisez l'assistant Eclipse pour importer un projet ActionScript.

1 Dans Flash Builder, sélectionnez Fichier > Importer > Autre > Général > Fichier archive.

   Vous pouvez également faire appel au menu contextuel de l'Explorateur de packages.

2 Dans la boîte de dialogue d'importation d'un projet Flex, sélectionnez le fichier ZIP à importer.

3 Cliquez sur Terminer.

Importation de projets exportés avec l'assistant d'exportation Eclipse

Un projet exporté à l'aide de l'assistant d'exportation Eclipse devra être importé avec l'assistant d'importation Eclipse. Sélectionnez Fichier > Importer > Général. Recherchez le format correspondant à votre projet.

Pour plus d'informations, voir la documentation Eclipse sur l'importation de projets. Cette documentation est disponible dans l'aide des assistants d'importation et d'exportation Eclipse.

Les services d'un projet créés avec les outils Flash Builder d'accès aux services de données devront être ajoutés manuellement. Copiez les fichiers de serveur dans le dossier services du serveur approprié. Dans la vue Données/Services, utilisez les propriétés du service pour un serveur afin de déterminer l'emplacement du service.

Si vous avez exporté un projet PHP basé sur la structure Zend Framework, il vous faut installer cette dernière sur le serveur cible. Modifiez le fichier `amf-config.ini` qui configure la structure Zend Framework. Pour `zend_path`, spécifiez le chemin absolu du répertoire d'installation de Zend Framework.

Pour plus d'informations sur l'installation, la configuration et le dépannage de l'installation de la structure Zend Framework, voir Installation de Zend Framework.

Importation d'un projet dans plusieurs espaces de travail

Lorsque vous importez un projet, il est importé dans un espace de travail Flash Builder. Un projet peut être importé dans plusieurs espaces de travail. Dans ce cas, les fichiers du projet se trouvent dans un seul emplacement sur le disque, mais sont référencés par chaque espace de travail. Les modifications apportées au projet se répercutent dans tous les espaces de travail.

Importation de fichiers source dans un nouveau projet

Vous pouvez créer un projet dans lequel importer les fichiers source et les ressources situés sur votre système de fichiers mais n'appartenant à aucun projet.

1 Dans le menu Flash Builder, sélectionnez Fichier > Nouveau > *Projet*.

Il peut s'agir d'un projet Flex, d'un projet de bibliothèque Flex ou d'un projet ActionScript.

2 Dans l'assistant Nouveau projet Flex, spécifiez la source et le dossier de sortie.

Remarque : *vous pouvez également accepter les emplacements par défaut proposés par l'assistant et y déplacer les fichiers source.*

Comparaison des modifications apportées à un projet

Si vous importez plusieurs versions d'un projet, vous pouvez comparer, copier ou fusionner les contenus des versions. Vous pouvez uniquement comparer des versions différentes d'un même projet.

1 Dans l'Explorateur de packages, sélectionnez l'un des projets que vous souhaitez comparer.

2 Ouvrez le menu contextuel et sélectionnez Comparer le projet avec la version.

L'affichage de comparaison s'ouvre, vous permettant de comparer le projet à d'autres versions de ce même projet.

3 Sélectionnez la version avec laquelle vous souhaitez effectuer la comparaison. L'éditeur de comparaison Eclipse s'ouvre.

4 Dans l'éditeur de comparaison, accédez au fichier que vous souhaitez comparer et sélectionnez Afficher la comparaison du contenu dans le menu contextuel.

L'éditeur de comparaison affiche les deux versions du fichier et en met en évidence les différences.

Vous pouvez utiliser les options de l'éditeur de comparaison pour copier ou fusionner les différences dans le fichier. Pour plus d'informations, voir la documentation Eclipse sur l'éditeur de comparaison.

Projets nécessitant un traitement spécial

Certains projets Flex devront faire l'objet d'un traitement spécial lors de l'importation et de l'exportation. Par exemple :

- le projet fait référence à une version antérieure du SDK Flex ;
- le projet fait référence à des fichiers de service pour l'accès aux données côté serveur ;
- la configuration Zend Framework pour l'accès aux services PHP doit être mise à jour ;
- le projet utilise des liens de LiveCycle Data Services vers un fichier de modèle de données.

Lors de l'exportation ou de l'importation d'un projet Flex, certains contenus du projet nécessiteront un traitement spécial.

- Versions différentes du SDK Flex

  Vous pouvez importer un projet Flex faisant référence à une version du SDK Flex qui n'a pas été installée avec Flash Builder. Pour plus d'informations sur le téléchargement et l'installation de versions supplémentaires du SDK Flex, voir « SDK Flex installés » à la page 266. Si Flash Builder ne peut pas trouver de version spécifique du SDK, recherchez l'emplacement du SDK.

- Fichiers de services

  Les projets de serveur Flex qui se connectent aux services de données, tels que ColdFusion ou BlazeDS, contiennent un dossier services dont la référence de classe ActionScript a déployé les fichiers de serveur. Lors de l'exportation du projet, Flash Builder exporte le dossier de services, mais vous devez vous assurer qu'un serveur et les fichiers côté serveur correspondants existent lors de l'importation. Au moment de l'importation, vous devrez peut-être déployer les fichiers côté serveur et mettre à jour les adresses du serveur dans les classes des fichiers côté serveur manuellement. Pour les projets qui se connectent aux services à l'aide de LiveCycle Data Services ou BlazeDS, assurez-vous que les destinations de service sont disponibles sur le serveur cible.

- Structure Zend Framework

  Les projets Flex qui se connectent à des services de données en utilisant PHP et la structure Zend Framework contiennent deux fichiers de configuration. A l'importation, examinez ces fichiers afin de vérifier qu'ils ont été correctement configurés pour votre système :

  ```
  amf-config.ini
  ```

  ```
  gateway.php
  ```

  Voir Installation de Zend Framework

pour des informations sur l'installation, la configuration et le dépannage de votre installation Zend Framework.

- Fichiers de modèles de données (LiveCycle Data Services)

  Un projet Flex utilisant LiveCycle Data Services est relié à un fichier de modèle de données. Après exportation puis importation, Flash Builder référence le fichier de modèle de données actuel et non le lien qui y renvoie. Si vous souhaitez utiliser un fichier lié (et non celui inclus dans le projet exporté), modifiez le fichier de modèle de données en accédant aux propriétés du projet. Sélectionnez Projet > Propriétés > Modèle de données pour apporter les modifications.

Création de projets

Adobe® Flash® Builder™ génère et exporte automatiquement les projets vers des applications. Cette opération comporte la création de fichiers d'application et de bibliothèque, le placement des fichiers de sortie dans les emplacements appropriés et la signalisation de toute erreur survenue en cours de compilation.

Différentes options permettent d'ajuster les paramètres de génération. Vous pouvez ainsi déterminer la manière dont les projets sont intégrés aux applications. Par exemple, vous pouvez définir des préférences de génération sur des projets individuels ou sur tous les projets de l'espace de travail. Vous pouvez également modifier le chemin de résultat de la création, modifier l'ordre de la génération, etc. Vous pouvez en outre créer des instructions de génération personnalisées à l'aide d'outils tiers tels qu'Apache Ant.

Lorsque les applications sont prêtes à être publiées, vous pouvez en publier l'ensemble du code source ou uniquement certaines parties. A l'instar du code source HTML, le code source de l'application peut être visualisé dans un navigateur Web.

Présentation de la création et de l'exportation des projets

Un flux de travail se compose de la génération de projets Flex et ActionScript, l'option Générer automatiquement étant activée. Au cours du développement, Flash Builder affiche les erreurs et les avertissements dans la vue Erreurs. A l'exécution de l'application, une version de débogage du fichier SWF est placée dans le dossier de sortie (bin) du projet avec les ressources requises et une enveloppe HTML. Cette génération contient des informations de débogage et n'est destinée qu'à l'usage des développeurs. Pour plus d'informations sur l'exportation de projets, voir « Exportation et importation de projets » à la page 81.

Une fois l'application prête à être déployée, l'assistant Exporter vers une version validée vous permet de créer une version optimisée de qualité finale de l'application. Cette opération enregistre le fichier SWF dans le dossier bin-release. Les informations de débogage étant supprimées, la taille du fichier diminue. Cette version est destinée à être consultée par l'utilisateur final. Pour les projets Adobe AIR, les applications AIR sont exportées dans un fichier AIR. La fonction Exporter vers une version validée permet de créer un fichier AIR signé numériquement que les utilisateurs doivent installer avant d'exécuter une application (à l'instar d'un fichier install.exe).

Aucune exportation n'est requise pour les projets de bibliothèque. Le fichier SWC créé par un projet de bibliothèque Flex se prête à la fois au développement et à la production. Pour plus d'informations, voir « Utilisation de projets de bibliothèque Flex » à la page 200.

Notions de base sur la génération

MXML et ActionScript 3.0 sont des langages *compilés*. Les langages compilés sont différents des langages *interprétés*, tels que JavaScript, qui peuvent être exécutés par leur propre environnement d'exécution. C'est-à-dire que les fichiers MXML et ActionScript 3.0 doivent disposer d'un format compilé pour que Flash Builder puisse les exécuter. Ce processus, ainsi que la création des fichiers de sortie correspondants, est désigné par le terme de *génération*.

Flash Builder génère automatiquement un projet à chaque modification ou enregistrement de l'un de ses fichiers. Vous avez par ailleurs la possibilité de générer manuellement vos applications. Comprendre le processus de génération et les fichiers de sortie qui sont générés vous aide à diagnostiquer et résoudre les erreurs de configuration du projet, lorsqu'elles se présentent.

Projets Flex : les fichiers source et les éléments incorporés (tels que les images) sont compilés dans un seul fichier SWF de sortie. Le fichier SWF peut s'exécuter directement dans le lecteur autonome Flash Player ou dans un navigateur Web via un fichier d'*enveloppe HTML* qui est également issu de la génération. Ces fichiers sont générés dans le dossier de sortie du projet. Le dossier de sortie s'appelle bin par défaut, mais vous pouvez le renommer.

Projets ActionScript 3.0 : à l'instar des projets Flex, les projets ActionScript 3.0 compilent les fichiers source et les ressources intégrées dans un fichier SWF.

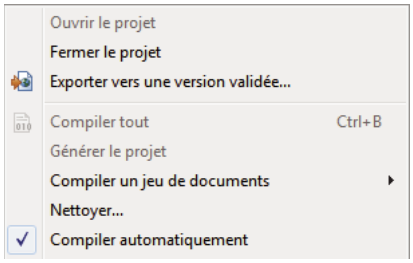
Projets de bibliothèque Flex : pour les projets de bibliothèque, les fichiers source sont des composants et des ressources apparentées. Lors de la génération de projets de bibliothèque, un fichier SWC est créé dans le dossier de sortie. Un fichier SWF est archivé dans un fichier SWC contenant des composants, des ressources et le fichier catalog.xml, fichier manifeste des éléments contenus dans le fichier SWF.

Générations automatiques

Dans une configuration autonome de Flash Builder, l'option Générer automatiquement est sélectionnée par défaut et vos applications sont générées automatiquement. En configuration plug-in, sélectionnez l'option Générer automatiquement. Le fait de désélectionner l'option Générer automatiquement empêche le compilateur d'identifier les erreurs de syntaxe. La vue Erreurs n'affiche alors pas de messages d'avertissement et d'erreur lorsque vous saisissez du code. La vue Erreurs affiche des messages d'avertissement et d'erreur uniquement lorsque vous compilez le projet. Il est par conséquent recommandé de définir Flash Builder sur la génération automatique.

Options avancées pour la génération de projets

Les options avancées permettent de définir la durée et l'étendue des générations. Vous pouvez par exemple générer un seul projet ou un ensemble de projets dans l'espace de travail ou encore définir un jeu de documents (une collection) de projets à générer. Toutes les commandes de génération se situent dans le menu Projet, comme le montre l'exemple suivant.



Le compilateur Flash Builder est incrémentiel. Il ne génère que les ressources qui ont été ajoutées ou concernées par une mise à jour et ignore toutes les autres, ce qui se traduit par un gain de temps et une utilisation parcimonieuse des ressources du système. Vous disposez toutefois de la possibilité de régénérer l'ensemble des ressources du projet en procédant à une *génération nettoyée*. Ce type de génération est indiqué lorsque l'application présente un comportement imprévisible au cours des tests effectués. Vous pouvez dans ce cas éliminer toute source potentielle de problèmes en ignorant et régénérant tous les fichiers du projet. Pour plus d'informations, voir « Options de génération avancées » à la page 93.

En présence de dépendances entre des projets distincts de l'espace de travail, le compilateur détermine automatiquement l'ordre dans lequel les projets ont été générés afin de résoudre correctement les dépendances. Vous pouvez toutefois remplacer l'ordre de génération par défaut en définissant manuellement l'ordre dans lequel les projets de l'espace de travail sont générés.

Vous pouvez en outre modifier le chemin de génération, la liste des applications et les paramètres du compilateur pour chaque projet de l'espace de travail.

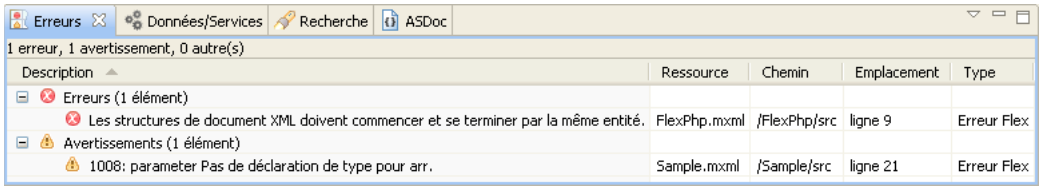
Voir aussi

« Création manuelle de projets » à la page 93

« Options de génération avancées » à la page 93

Affichage des erreurs de génération dans la vue Erreurs

Les erreurs détectées par le compilateur au cours de la génération s'affichent dans la vue Erreurs des perspectives Développement et Débogage, ainsi que dans l'éditeur de code, dans lequel les lignes contenant des erreurs sont signalées par un x, comme l'illustre l'exemple suivant.



Voir aussi

« Filtrage des vues Tâches et Erreurs » à la page 12

Erreurs d'environnement Eclipse dans le fichier journal

Des erreurs générées par l'environnement Eclipse sont susceptibles de se produire. Elles surviennent généralement lorsque des ressources (des fichiers SWC, par exemple) ne sont pas trouvées à l'exécution. Dans ce cas, les messages d'erreur apparaissent dans le fichier journal des erreurs Eclipse. L'emplacement par défaut de ce fichier journal sous Windows est le suivant : \Documents and Settings*nom_utilisateur*\workspace\.metadata\.log. Sous Mac, l'emplacement par défaut se trouve également dans le répertoire de l'espace de travail. Les fichiers et dossiers commençant par un point y sont toutefois masqués par défaut.

Personnalisation des scripts de génération avec Apache Ant

Vous pouvez modifier et étendre le processus de génération standard en faisant appel à Apache Ant, outil de génération Java à code source libre. Pour plus d'informations sur la création de générateurs personnalisés, voir « Personnalisation des générations avec Apache Ant » à la page 95.

Options du compilateur

Vous pouvez modifier les paramètres par défaut du compilateur Flex ou ActionScript utilisé par Flash Builder.

Pour afficher ou modifier les paramètres par défaut, dans le menu de Flash Builder, sélectionnez Projet > Propriétés, puis sélectionnez Compilateur Flex ou Compilateur ActionScript selon le besoin.

Version du SDK Flex

Pour les projets Flex, Flash Builder utilise Flex 4.6 en tant que SDK par défaut. Toutefois, si votre projet utilise une version particulière, telle que Flex 3.5, Flash Builder compile les applications du projet à l'aide du SDK Flex spécifié.

Vous pouvez définir comme paramètre par défaut un SDK Flex spécifique ou vous pouvez définir que la compilation soit effectuée en utilisant la compatibilité avec Flex 3. La spécification de la compatibilité ascendante affecte certains comportements, tels que les règles de présentation, les marges intérieures et les espaces, les habillages ainsi que d'autres paramètres de style. Elle a également une incidence sur les règles d'analyse des fichiers de propriétés. La définition de la version de compatibilité n'applique pas toutes les différences existant entre les versions. Pour plus d'informations, voir Compatibilité ascendante.

SDK AIR

Pour les projets ActionScript, Flash Builder utilise le SDK AIR contenant le compilateur ActionScript. Pour plus d'informations, voir « SDK AIR installé » à la page 266.

Options d'Adobe Flash Player

La version par défaut de Flash Player utilisée par le compilateur est la version minimale requise par le SDK utilisé pour la compilation.

Vous pouvez définir une version spécifique de Flash Player pour l'application. Les fonctionnalités nécessitant une version ultérieure de Flash Player ne seront pas compilées dans l'application.

Options du compilateur

Flash Builder fournit des cases à cocher pour les options suivantes du compilateur :

- Utiliser le moteur de texte Flash dans les composants MX

  Cette option n'est disponible que pour les projets Flex. Le moteur de texte Flash (Flash Text Engine, FTE) est une bibliothèque qui fournit des contrôles de texte dotés d'un ensemble complet d'options de formatage. Tous les composants Spark du package spark.components prennent en charge FTE. Voir

  Certains contrôles MX fournissent une prise en charge de FTE. Les contrôles MX prenant en charge FTE utilisent les mêmes polices intégrées que les composants Spark utilisant FTE. Voir Using FTE in MX controls.

- Copier les fichiers non imbriqués dans le dossier de sortie
- Générer un fichier SWF accessible

  Cette option n'est disponible que pour les projets Flex. La sélection de cette option active les fonctionnalités d'accessibilité lors de la compilation de l'application ou du fichier SWC. Pour plus d'informations sur l'utilisation des fonctionnalités d'accessibilité avec Flex, voir Accessible applications.

- Activer la vérification stricte

  Lorsque la vérification stricte est activée, le compilateur imprime des appels de propriété et de fonction non définis. Il effectue également une vérification de type au moment de la compilation pour les affectations et les options fournies aux appels de méthode.

- Activer les avertissements

  Cette option active des avertissements spécifiques. Pour plus d'informations, voir Viewing warnings and errors.

Vous pouvez aussi spécifier des arguments de compilateur à plusieurs lignes qui sont disponibles avec la ligne de commande sur le compilateur mxmlc. Vous pouvez définir les valeurs de la plupart des options du champ Arguments de compilateur supplémentaires en utilisant la syntaxe de la ligne de commande. Pour plus d'informations sur la syntaxe pour la définition des options dans la boîte de dialogue Compilateur Flex, voir About the command-line compilers.

Dans le champ Arguments de compilateur supplémentaires, vous pouvez remplacer un chemin vers le répertoire du SDK par le jeton `${flexlib}`, comme l'illustre l'exemple suivant :

```
-include-libraries "${flexlib}/libs/automation.swc" "${flexlib}/libs/automation_agent.swc"
```

Enveloppe HTML

Outre les fichiers SWF pour les applications Web, le compilateur Flash Builder génère également une enveloppe HTML que vous pouvez utiliser lors du déploiement de l'application. Les options suivantes sont disponibles :

- Générer le fichier d'enveloppe HTML
- Vérifier la version du lecteur cible

  Lorsque cette option est activée, l'application compilée vérifie que la version de Flash Player est correcte.

  Si l'option d'utilisation de l'installation rapide est activée, l'application exécute un fichier SWF dans le lecteur Flash Player existant pour mettre à niveau les utilisateurs vers la dernière version du lecteur.

- Activer l'intégration avec le navigateur

  Cette option active la liaison profonde. La liaison profonde permet aux utilisateurs de parcourir leurs interactions avec l'application en utilisant les boutons Précédent et Suivant de leur navigateur.

Accès par ligne de commande aux compilateurs de structure Flex

Vous pouvez accéder directement par ligne de commande aux compilateurs de structure Flex (mxmlc et compc). Pour plus d'informations, voir About the command-line compilers dans Using Adobe Flex 4.6.

Personnalisation des générations de projet

Flash Builder vous permet de générer des applications automatiquement en utilisant les paramètres de projet par défaut. L'utilisation des paramètres de projet par défaut est l'approche recommandée pour générer vos applications. Vous pouvez toutefois personnaliser la génération de projets en fonction de vos besoins. Vous pouvez ainsi par exemple sélectionner un autre dossier de sortie par défaut ou modifier les options du compilateur.

Jeu de composants (MX + Spark, Spark uniquement ou MX uniquement)

Par défaut, les projets Flex mettent à la disposition des applications l'ensemble des composants, c'est-à-dire les composants Spark fournis avec Flex 4 et les composants MX livrés avec Flex 3.

Dans certains cas, vous pouvez utiliser uniquement les composants MX rendus disponibles avec Flex 3, ou uniquement les composants Spark livrés avec Flex 4. Vous pouvez par exemple disposez d'un projet Flex 3 dans lequel vous ne voulez pas introduire les nouveaux composants Spark mais pour lequel vous souhaitez toutefois bénéficier des avantages des fonctionnalités introduites avec Flex 4 et Flash Builder 4 (nouvelle syntaxe d'états, fonctions de compilation améliorées et autres fonctionnalités de langage). Dans ce cas, activez l'option MX uniquement pour le jeu de composants. L'activation de cette option supprime du chemin de génération toutes les bibliothèques associées à Spark. Lors de la conversion d'un projet Flex 4 en projet MX uniquement, Flash Builder n'apporte aucune modification au code du projet. Vous devrez mettre à jour le code manuellement afin d'en supprimer toute référence aux composants et aux bibliothèques Spark.

De même, si vous sélectionnez le jeu de composants Spark uniquement, seuls les composants Spark sont utilisés et les bibliothèques de type MX sont supprimées du chemin de génération.

Liaison de la structure des applications

Par défaut, les classes d'application pour Flex 4 ou version supérieure de la structure Flex utilisent la liaison dynamique. Au lieu de compiler toutes les classes dans le fichier SWF de l'application (liaison statique), certaines classes sont chargées à partir de la bibliothèque partagée à l'exécution (Runtime Shared Library, RSL). Les applications créées avec une liaison dynamique sont dotées de fichiers SWF plus petits, ce qui permet de les télécharger plus rapidement. En revanche, ces applications utilisent plus de mémoire, toutes les classes de la structure étant chargées, y compris celles dont vous n'avez pas besoin. Pour plus d'informations, voir Bibliothèque partagée à l'exécution.

Vous pouvez modifier les propriétés d'un projet afin de personnaliser ce comportement pour toutes les applications d'un projet. Après avoir sélectionné un projet, dans le menu de Flash Builder, choisissez Projet > Propriétés > Chemin de génération Flex > Chemin d'accès à la bibliothèque.

Par défaut, Flash Builder utilise le comportement par défaut du SDK Flex pour la liaison de la structure. Pour Flex 4 et versions supérieures, le comportement par défaut est la liaison dynamique des RSL. pour Flex 3, de la liaison statique. Utilisez la liste déroulante Liaison de structure pour remplacer le comportement par défaut.

Pour le SDK Flex 4.5 et versions supérieures de la structure Flex, les options suivantes sont activées par défaut :

- Vérifier les condensés RSL (recommandé en production)

  Vérifie que le condensé de la bibliothèque RSL correspond à celui stocké dans l'application au moment de la compilation lorsque l'application a été liée à la bibliothèque RSL interdomaines. Pour plus d'informations, voir About RSL digests.

- Supprimer les RSL non utilisées

Supprime les RSL qui ne sont pas utilisées au moment de la compilation. Seules les RSL utilisées sont copiées dans le dossier de sortie. Cette option est uniquement disponible pour le SDK Flex 4.5 et versions supérieures de la structure Flex, et non pour les structures Flex antérieures.

Vous pouvez toutefois charger de force un fichier RSL inutilisé au moment de la compilation en cliquant deux fois sur le chemin d'accès à la bibliothèque du projet. Sélectionnez ensuite Charger de force le fichier RSL dans la boîte de dialogue Options de l'élément de chemin d'accès à la bibliothèque.

Vous pouvez spécifier le domaine d'application dans lequel les bibliothèques RSL interdomaines doivent être chargées. Vous pouvez sélectionner Par défaut, Actif, Parent ou Supérieur comme domaine d'application.

Par exemple, si un module utilise une bibliothèque RSL spécifique, vous pouvez sélectionner Actif comme domaine d'application. Ensuite, la bibliothèque RSL spécifiée est accessible seulement à ce module et pas à l'application qui charge le module. Le déchargement du module décharge automatiquement également la bibliothèque RSL associée.

Remarque : *les RSL interdomaines et les domaines d'application sont uniquement pris en charge dans le SDK Flex 4.5 et versions supérieures de la structure Flex, et non dans les structures Flex antérieures. Si vous importez un projet qui a été créé avec une structure Flex faisant appel à des fichiers RSL standard, Flash Builder convertit automatiquement ces fichiers RSL standard en fichiers RSL interdomaines.*

Pour plus d'informations, voir Compilation avec des bibliothèques RSL standard ou interdomaines dans Flash Builder.

- Utilisez des bibliothèques SWF RSL locales de débogage lors du débogage.

  Utilise les bibliothèques RSL locales lors du débogage de l'application. L'utilisation des bibliothèques RSL locales vous permet d'intervenir dans les fichiers RSL de débogage. Cette option est ignorée lors de l'exportation d'une version validée.

- Déterminer automatiquement l'ordre de la bibliothèque, en fonction des dépendances

  Si cette option est activée, Flash Builder détermine l'ordre des bibliothèques en fonction des dépendances au sein des bibliothèques. Pour personnaliser l'ordre des bibliothèques, désactivez cette option et indiquez l'ordre voulu à l'aide des boutons Haut et Bas.

Activation et désactivation des générations automatiques

En configuration autonome, Flash Builder génère vos projets automatiquement. Dans la configuration du plug-in, sélectionnez vous-même cette option. Flash Builder est conçu pour générer les projets automatiquement. La désactivation de cette option empêche le compilateur de détecter les erreurs de syntaxe et d'afficher les avertissements et les messages d'erreur au fur et à mesure de la saisie du code. Pour plus d'informations sur la génération manuelle de projets, voir « Création manuelle de projets » à la page 93.

Effectuez l'une des opérations suivantes :

- Sélectionnez Projet > Générer automatiquement.
- Ouvrez la boîte de dialogue Préférences et sélectionnez Général > Espace de travail. Sélectionnez ou désélectionnez l'option Générer automatiquement.

Cette option a une incidence sur tous les projets de l'espace de travail.

Définition d'un dossier de sortie de projet

Lorsque vous créez un projet dans Flash Builder, le résultat de la création est généré par défaut dans le dossier de sortie.

Vous pouvez modifier le nom de ce dossier à la création du projet ou une fois le projet créé. Vous avez en outre la possibilité de créer un dossier ou de sélectionner un dossier existant de l'espace de travail.

1 Dans l'Explorateur de packages Flex, sélectionnez un projet.

2 Cliquez avec le bouton droit de la souris (Windows) ou en maintenant la touche Ctrl enfoncée (Macintosh), puis sélectionnez Propriétés dans le menu contextuel.

La boîte de dialogue des propriétés du projet s'affiche.

3 Sélectionnez la page Chemin de génération Flex.

4 Modifiez le dossier de sortie spécifié en saisissant un nouveau nom ou en recherchant et sélectionnant le dossier existant du projet.

Remarque : *le dossier de sortie d'une application LiveCycle Data Services ES ne peut pas être modifié de cette manière. Son emplacement est en effet contrôlé par le serveur et il est accessible uniquement par le biais du fichier Flex-config.xml du projet.*

5 Cliquez sur OK.

Le nouveau dossier de sortie remplace le dossier de sortie existant.

Important : *la modification du nom du dossier de sortie entraîne la suppression du dossier de sortie d'origine ainsi que de l'ensemble de son contenu. Régénérez le projet afin de recréer le fichier d'application SWF et les fichiers d'enveloppe HTML.*

Modification d'un chemin de génération de projet

Chaque projet possède son propre chemin de génération composé du chemin source et du chemin de bibliothèque (les chemins de génération de projet de bibliothèque sont légèrement plus complexes. Pour plus d'informations, voir « Création de projets de bibliothèque Flex » à la page 200). Le chemin source correspond à l'emplacement des fichiers source MXML et ActionScript du projet. Le chemin de bibliothèque correspond à l'emplacement des classes de la structure Flex de base ainsi que des composants Flex personnalisés que vous avez créés (sous la forme de fichiers SWC).

Modification du chemin source

1 Sélectionnez un projet dans l'Explorateur de packages.

2 Cliquez avec le bouton droit de la souris (Windows) ou en maintenant la touche Ctrl enfoncée (Macintosh), puis sélectionnez Propriétés dans le menu contextuel. La boîte de dialogue des propriétés du projet apparaît.

3 Sélectionnez la page Chemin de génération Flex (dans le cadre d'un projet ActionScript, sélectionnez la page Chemin de génération ActionScript).

4 Cliquez sur le bouton Ajouter un dossier pour ajouter un dossier au chemin source.

5 Saisissez le nom du dossier ou cliquez sur le bouton Parcourir pour rechercher l'emplacement des classes personnalisées.

Vous pouvez également utiliser des variables de chemin afin d'éviter de saisir le chemin complet du système de fichiers. Vous pouvez saisir le nom d'une variable existante ou en créer une. Pour plus d'informations, voir « Création d'une variable de chemin » à la page 93.

6 Apportez les modifications nécessaires au chemin source, puis cliquez sur OK.

Modification du chemin de la bibliothèque

1 Répétez les étapes 1 à 3 de la procédure précédente afin d'accéder à la page des propriétés Chemin de génération Flex.

2 Cliquez sur l'onglet Chemin d'accès à la bibliothèque.

Le chemin de la bibliothèque contient des références aux classes de la structure Flex contenues dans des fichiers SWC. Un fichier SWC est un fichier archive pour les composants Flex et les autres ressources. Pour plus d'informations, voir « Utilisation des fichiers SWC dans le cadre des projets » à la page 203.

Vous pouvez modifier le chemin vers la structure ou, si vous avez créé des composants Flex personnalisés, ajouter de nouveaux dossiers ou fichiers SWC au chemin de la bibliothèque. Vous avez également la possibilité de supprimer des éléments du chemin.

3 Apportez les modifications nécessaires au chemin de la bibliothèque, puis cliquez sur OK.

Création d'une variable de chemin

Vous pouvez établir des liens vers des ressources en définissant des variables de chemin. Vous évitez ainsi de devoir saisir le chemin complet d'un dossier local ou d'un dossier réseau dans lequel vous enregistrez les fichiers. Définissez par exemple une variable de chemin nommée Classes, puis spécifiez le chemin vers un dossier du système de fichiers. La variable Classes peut ensuite être sélectionnée comme étant l'emplacement du nouveau dossier auquel la relation a été établie. Si le dossier est déplacé, il vous suffit de mettre à jour l'emplacement dans la variable de chemin définie pour que tous les projets reliés à la variable Classes puissent continuer à accéder aux ressources.

Définition ou création d'une variable de chemin

1 Sélectionnez un projet dans l'Explorateur de packages.

2 Cliquez avec le bouton droit de la souris (Windows) ou en maintenant la touche Ctrl enfoncée (Macintosh), puis sélectionnez Propriétés dans le menu contextuel. La boîte de dialogue des propriétés du projet apparaît.

3 Sélectionnez la page Chemin de génération Flex (dans le cadre d'un projet ActionScript, sélectionnez la page Chemin de génération ActionScript).

4 Vous pouvez créer une variable de chemin pour chaque élément du chemin (dont les dossiers du chemin source et les dossiers SWC, les projets et les fichiers SWC du chemin de la bibliothèque). Par exemple, cliquez sur Ajouter un dossier sous l'onglet Chemin source. La boîte de dialogue d'ajout d'un dossier apparaît.

5 Saisissez une variable de chemin conformément au format suivant : `${nom_de_la_variable_de_chemin}`.

Remarque : *la variable doit exister, sinon la saisie du chemin échoue. Pour obtenir la liste des variables de ressource existantes, sélectionnez Fenêtre > Préférences dans le menu principal, puis Général > Espace de travail > Ressources liées. Les variables de ressources liées peuvent également être gérées à partir de cette page de propriétés.*

6 Cliquez sur OK pour ajouter la variable au chemin.

Options de génération avancées

Flash Builder offre des options avancées pour la personnalisation des générations de projet. Vous pouvez par exemple générer des projets manuellement, modifier l'ordre par défaut de génération des projets dans l'espace de travail et créer des générateurs personnalisés avec l'utilitaire Apache Ant.

Création manuelle de projets

La génération manuelle de projets vous permet de contrôler la durée et l'étendue de la génération. Vous pouvez par exemple générer un seul projet ou un ensemble de projets dans l'espace de travail. Vous pouvez également définir un jeu de documents de projets ou de ressources de projet et ne générer que ces projets et ces ressources. Un jeu de documents est une collection de ressources d'espace de travail (projets, fichiers et dossiers) pouvant être sélectionnées et groupées en fonction de l'utilisation qui en est faite. Pour plus d'informations sur les jeux de documents, voir « Création de jeux de documents » à la page 120.

Génération d'un projet unique

1 Dans l'Explorateur de packages, sélectionnez le projet que vous souhaitez générer.

2 Sélectionnez Projet > Générer le projet dans le menu principal.

Le projet sélectionné est généré. Des fichiers d'application validés ou débogués (nouveaux ou mis à jour) sont ajoutés au dossier de sortie du projet.

Remarque : *le programme vous invite à enregistrer les fichiers de projet qui ne l'ont pas encore été avant le début de la génération. Pour éviter l'affichage de cette invite, activez l'enregistrement automatique des fichiers avant le début de la génération dans les préférences de l'espace de travail.*

Génération de tous les projets dans l'espace de travail

❖ Sélectionnez Projet > Générer tout dans le menu principal.

Tous les projets de l'espace de travail sont générés. Les fichiers d'application sont ajoutés aux dossiers de sortie des projets respectifs. Dans la mesure où vous n'avez pas défini l'enregistrement automatique des fichiers avant le début de la génération, le programme vous invite à les enregistrer.

Génération d'un jeu de documents

Utilisez l'une des méthodes suivantes :

- Sélectionnez Projet > Générer un jeu de documents > Sélectionner un jeu de documents dans le menu principal. Cliquez sur Nouveau pour créer un jeu de documents. Pour plus d'informations sur la création d'un jeu de documents, voir « Création de jeux de documents » à la page 120.
- Choisissez un jeu de documents existant en sélectionnant Projet > Générer un jeu de documents > Sélectionner un jeu de documents dans le menu principal.

Tous les projets du jeu de documents sont générés. Les fichiers d'application sont ajoutés au dossier de sortie du projet.

Enregistrement automatique des ressources du projet

Lorsque vous générez des projets manuellement, le programme vous invite à enregistrer toutes les ressources avant de débuter la génération. Pour éviter l'affichage de cette invite, activez l'enregistrement automatique des ressources du projet dans les préférences de l'espace de travail.

1 Ouvrez la boîte de dialogue Préférences et sélectionnez Général > Espace de travail.

2 Sélectionnez l'option Sauvegarder automatiquement avant la génération.

3 (Facultatif) Vous pouvez modifier la fréquence d'enregistrement des ressources en saisissant une valeur (en minutes) pour Intervalle de sauvegarde du plan de travail.

Exécution d'une génération nettoyée

Une fois un projet créé, les générations ultérieures n'impliqueront que les ressources ajoutées ou modifiées. Une génération nettoyée permet de forcer le compilateur Flash Builder à régénérer toutes les ressources d'un projet. Vous pouvez y recourir par exemple pour éliminer toutes les sources potentielles d'un problème survenu au cours du test de l'application.

1 Sélectionnez Projet > Nettoyer dans le menu principal.

2 Choisissez le ou les projets dont vous souhaitez éliminer les fichiers de génération pour les régénérer.

3 Cliquez sur OK.

Modification de l'ordre de génération d'un projet

Flash Builder permet de créer des relations entre les projets ouverts dans l'espace de travail. Vous pouvez par exemple importer les classes ActionScript d'un projet à l'autre. La création de relations entre les projets a une incidence sur l'ordre dans lequel les projets sont générés.

Par défaut, le compilateur génère des projets reliés dans l'ordre garantissant la génération correcte de tous les projets. Prenons l'exemple de deux projets, dont le premier fait référence à des classes contenues dans le second : le second projet est généré en premier. Dans la plupart des cas, le compilateur génère les projets dans l'ordre requis et les applications sont créées correctement sans qu'aucune intervention ne soit nécessaire.

Vous avez cependant la possibilité de modifier l'ordre de génération. Cette modification peut s'avérer nécessaire par exemple si vous avez créé un générateur personnalisé Ant et que vous l'avez associé à un projet de l'espace de travail que vous devez générer avant les autres projets. Pour plus d'informations sur la création de générateurs personnalisés, voir « Personnalisation des générations avec Apache Ant » à la page 95.

1 Ouvrez la boîte de dialogue Préférences et sélectionnez Général > Espace de travail > Ordre de génération.

   La boîte de dialogue Ordre de génération présente les options suivantes.

   Utiliser l'ordre de génération par défaut : l'ordre de génération par défaut dépend des dépendances entre les projets. Il est géré par le compilateur.

   Ordre de génération du projet : vous pouvez définir manuellement l'ordre dans lequel tous les projets de l'espace de travail doivent être générés. Il est également possible de supprimer un projet de la liste définissant l'ordre de génération. Dans ce cas, le projet sera généré après tous les autres projets de la liste.

   Itérations maximales lors de la génération avec des cycles : en présence de références cycliques dans un projet (ce que nous vous conseillons d'éviter), vous pouvez définir le nombre de tentatives de génération, afin de permettre au compilateur de générer tous les projets correctement. Par défaut, le nombre maximal d'itérations est 10.

2 Modifiez si nécessaire l'ordre de génération, puis cliquez sur OK.

Personnalisation des générations avec Apache Ant

La création d'un générateur personnalisé vous permet de modifier et d'étendre le processus de génération standard. Flash Builder comporte un script de génération standard intervenant dans la compilation des applications. Vous pouvez, le cas échéant, créer des scripts de génération personnalisés grâce à Apache Ant, outil de génération Java à code source libre.

Vous pouvez appliquer les générateurs personnalisés à tous les types de projet Flash Builder.

Création d'un générateur

1 Dans l'Explorateur de packages, sélectionnez un projet. Cliquez avec le bouton droit de la souris (Windows) ou en maintenant la touche Ctrl enfoncée (Macintosh) pour afficher le menu contextuel et sélectionnez Propriétés.

2 Sélectionnez la page de propriétés Générateurs. La liste sera plus ou moins longue en fonction du nombre de plug-ins que vous utilisez. Flash Builder fournit un générateur nommé Flex qui ne peut pas être modifié.

3 Sélectionnez Nouveau.

4 Dans la boîte de dialogue Sélection d'un type de configuration, choisissez le type de configuration approprié. Flash Builder prend en charge le type Programme. Sélectionnez-le, puis cliquez sur OK pour poursuivre. Définissez les propriétés du nouveau générateur dans la page des propriétés et référencez le script Ant (fichier XML).

5 Cliquez sur OK pour appliquer les valeurs définies au projet.

Pour plus d'informations sur l'utilisation des scripts de génération Ant, voir la documentation Eclipse disponible à l'adresse help.eclipse.org/help31/index.jsp.

Utilisation de plusieurs SDK dans Flash Builder

Flash Builder permet de modifier la version du SDK utilisé pour la compilation des projets. Vous pouvez sélectionner le SDK à la création ou en cours de développement du projet.

Le SDK se compose d'une structure et du compilateur. Le SDK Flex 4 signifie que vous utilisez la version 4 des fichiers SWC de la structure Flex et la version 4 du compilateur Flex. Vous ne pouvez pas utiliser le compilateur Flex 4 avec les fichiers SWC de la structure Flex 3, par exemple.

La possibilité de modifier le SDK est utile en présence d'un projet développé avec Flex Builder 3 (qui utilise le SDK Flex 3) alors que vous exécutez la version 4 de Flash Builder (qui utilise par défaut le SDK Flex 4). En sélectionnant un SDK antérieur pour la génération, vous pouvez préserver la compatibilité avec la dernière version du SDK des projets qui n'ont pas été mis à jour. En outre, si vous travaillez actuellement sur un projet destiné au SDK Flex 3, mais souhaitez utiliser les fonctionnalités de Flash Builder 4, vous pouvez mettre à niveau votre version de Flash Builder, mais sélectionnez dans ce cas le SDK antérieur comme SDK par défaut.

La modification du SDK après le développement d'un projet entraîne une régénération complète, et non une génération incrémentielle. Flash Builder signale alors toute différence qui aurait généré des erreurs de compilateur, tout comme si le projet avait été développé avec le SDK d'origine.

Flash Builder régénère également tous les fichiers de prise en charge des projets, dont les fichiers de gestion de l'historique et de liaisons profondes utilisés par l'enveloppe HTML. Pour les projets SDK Flex 4, Flash Builder crée les fichiers compatibles de gestion des historiques history.swf, history.html et history.js dans le répertoire des modèles HTML. Pour les projets SDK Flex 3, Flash Builder crée les fichiers compatibles de liaisons profondes history.htm et history.js et historyFrame.html dans le répertoire html-templates/history.

Les options présentes dans Flash Builder varient en outre en fonction du SDK sélectionné. Si vous ajoutez par exemple un module à un projet utilisant le SDK Flex 3, Flash Builder ne vous propose pas d'optimiser le module. Cette opération devra être effectuée manuellement.

Pour plus d'informations sur les différences entre les SDK Flex 4 et Flex 3, voir Compatibilité ascendante.

Flash Builder utilise le SDK par défaut pour les projets Flex créés. Le SDK par défaut est la dernière version fournie avec Flash Builder. Vous pouvez toutefois sélectionner tout SDK répertorié dans la liste des SDK disponibles dans Flash Builder.

Lorsque vous créez un projet de bibliothèque Flex ou un projet ActionScript, vous pouvez sélectionner le SDK à utiliser dans les boîtes de dialogue Nouveau projet de bibliothèque Flex et Nouveau projet ActionScript.

Ajout d'un SDK Flex à la liste des SDK disponibles

1. Ouvrez la boîte de dialogue Préférences et sélectionnez Flash Builder > SDK Flex installés.

   Elle répertorie les SDK installés. Le nom du SDK par défaut est coché.

2. Cliquez sur Ajouter.
3. Saisissez l'emplacement du SDK dans le champ Emplacement du SDK Flex.
4. Attribuez-lui un nom dans le champ Nom du SDK Flex. Ne saisissez pas le nom d'un SDK existant.
5. Cliquez sur OK pour enregistrer vos modifications.
6. Cliquez de nouveau sur OK pour ajouter le nouveau SDK à la liste des SDK disponibles. Cette liste est gérée dans l'espace de travail Flash Builder pour tous les projets Flex. Au prochain projet créé, la liste des SDK disponibles comportera également ce nouveau SDK.

Modification de la version de SDK du projet en cours

1. Sélectionnez Projet > Propriétés.

2 Choisissez Compilateur Flex.

3 Cliquez sur Utiliser un SDK spécifique.

4 Sélectionnez un SDK dans la liste déroulante. Si elle ne contient pas le SDK recherché, cliquez sur le lien Configurer les SDK Flex.

5 Cliquez sur OK.

Flash Builder applique le nouveau SDK au projet en cours. Des erreurs et des avertissements peuvent survenir si le projet utilise un code qui n'est pas compatible avec le nouveau SDK.

Sélection d'un nouveau SDK par défaut

1 Ouvrez la boîte de dialogue Préférences et sélectionnez Flash Builder > SDK Flex installés.

Le nom du SDK par défaut est coché.

2 Cochez la case correspondant à un autre SDK pour le définir comme SDK par défaut. Ce SDK sera appliqué à tout projet, y compris celui en cours, pour lequel l'option Utiliser le SDK par défaut a été sélectionnée dans la boîte de dialogue Compilateur Flex. Un projet paramétré pour utiliser un SDK spécifique n'est pas concerné par cette modification, et ce même si vous modifiez le SDK par défaut de l'espace de travail.

3 Cliquez sur OK pour enregistrer vos modifications.

4 Cliquez à nouveau sur le bouton OK.

Alternatives à l'utilisation de références de projets

Les références de projet pouvant avoir une incidence sur l'ordre de génération, Flash Builder propose une alternative à leur utilisation.

Projets de bibliothèque Flex : méthode privilégiée de création d'une bibliothèque réutilisable. Flash Builder crée une référence de projet pour garantir que le projet SWC est généré avant le projet principal qui le contient sur le chemin d'accès à la bibliothèque. En outre, puisque Flash Builder l'ajoute au chemin de la bibliothèque, des conseils de code apparaissent dans le projet principal pour les classes du projet SWC.

Chemin source : méthode recommandée pour inclure du code dans le projet ne se trouvant pas sous la même structure de dossier. Elle active les conseils de code dans les fichiers du projet et dans les classes situées dans des fichiers liés. Le compilateur sait où trouver le code source. Vous pouvez ajouter au projet un nombre quelconque de chemins source. Ils s'affichent en tant que dossiers liés dans l'Explorateur de packages.

Génération par ligne de commande Flash Builder à l'aide d'Apache Ant

Flash Builder fournit la tâche Ant <fb.exportReleaseBuild>. Utilisez-la pour implémenter des générations par ligne de commande qui synchronisent les paramètres de génération d'un développeur avec une génération nocturne. Vous pouvez également utiliser la tâche <mxmlc> dans les scripts personnalisés pour la génération nocturne.

Remarque : *la prise en charge de la génération par ligne de commande n'est disponible que sur Flash Builder Premium.*

Tâche <fb.exportReleaseBuild>

La tâche <fb.exportReleaseBuild> garantit que les paramètres de génération nocturne correspondent exactement à ceux utilisés par les développeurs pendant leur travail de la journée.

Par exemple, si un développeur modifie le chemin de la bibliothèque d'un projet Flex, le nouveau chemin est écrit dans le projet Flash Builder. A son exécution par l'ordinateur de génération nocturne, la tâche <fb.exportReleaseBuild> charge le projet Flash Builder et tous ses paramètres.

La tâche <fb.exportReleaseBuild> présente un autre avantage : elle prend automatiquement en charge les tâches de maintenance supplémentaires qui font normalement partie d'une génération Flash Builder. Par exemple :

- La compilation automatique des projets de bibliothèque associés
- La copie de ressources, telles que les fichiers JPEG, dans le répertoire de sortie
- La copie du modèle HTML, y compris le remplacement des macros en fonction des résultats de compilation (comme la largeur et la hauteur)

Remarque : *la tâche <fb.exportReleaseBuild> nécessite l'installation de Flash Builder sur l'ordinateur de génération nocturne.*

Tâche <mxmlc>

Si vous écrivez un script personnalisé qui fait appel à la tâche <mxmlc> (un script Ant, par exemple), vous n'avez pas besoin d'installer Flash Builder sur l'ordinateur de génération. Le SDK Flex doit toutefois s'y trouver. L'ordinateur de génération peut par conséquent se trouver sur une plateforme Linux.

Toutefois, l'inconvénient de cette approche réside dans le fait que vous avez deux jeux de paramètres de génération à synchroniser. En d'autres termes, vous avez un jeu dans Flash Builder, utilisé par les développeurs dans leur travail quotidien, et un autre jeu sur votre ordinateur de génération nocturne.

Utilisation de la tâche <fb.exportReleaseBuild>

1 Installez Flash Builder sur un ordinateur de génération.

2 Ecrivez `build.xml` avec fb.exportReleaseBuild comme cible. Par exemple :

```
<?xml version="1.0"?>
<project default="main">
    <target name="main">
        <fb.exportReleaseBuild project="MyProject" />
    </target>
</project>
```

`build.xml` indique qu'une génération par ligne de commande du projet Flex doit être exécutée avec les paramètres enregistrés dans les fichiers du projet. Pour plus d'informations sur les paramètres disponibles, voir « Paramètres de la tâche fb.exportReleaseBuild » à la page 99.

3 Créez un script de génération nocturne qui demande à Eclipse de rechercher un fichier de génération et d'exécuter sa cible.

Les exemples suivants indiquent `build.xml` comme fichier de génération, qui exécute MyTarget.

Si le script de la génération nocturne se trouve sur une plateforme Macintosh, vous pouvez exécuter le script suivant :

```
WORKSPACE="$HOME/Documents/Adobe Flash Builder"

# works with either FlashBuilder.app or Eclipse.app
"/Applications/Adobe Flash Builder/FlashBuilder.app/Contents/MacOS/FlashBuilder" \
    --launcher.suppressErrors \
    -noSplash \
    -application org.eclipse.ant.core.antRunner \
    -data "$WORKSPACE" \
    -file "$(pwd)/build.xml" MyTarget
```

Si la génération nocturne est effectuée sur une plateforme Windows, vous pouvez exécuter le fichier de commandes suivant :

```
set WORKSPACE=%HOMEPATH%\Adobe Flash Builder

REM works with either FlashBuilderC.exe or eclipsec.exe
"C:\Program Files\Adobe\Adobe Flash Builder 4.6\FlashBuilderC.exe" ^
    --launcher.suppressErrors ^
    -noSplash ^
    -application org.eclipse.ant.core.antRunner ^
    -data "%WORKSPACE%" ^
    -file "%cd%\build.xml" MyTarget
```

Propriété Ant fb.running

La propriété Ant fb.running a la valeur true lorsque Flash Builder est en cours d'exécution. Vous pouvez utiliser cette propriété lorsque vous exécutez des scripts dans Flash Builder. Par exemple :

```
<target name="myFlashBuilderTasks" if="fb.running">
    <fb.exportReleaseBuild ... />
</target>
```

Tâches Ant d'Eclipse

Eclipse propose plusieurs tâches Ant que vous pouvez incorporer comme cibles dans le script de génération. Par exemple :

- eclipse.incrementalBuild
- elicpse.refreshLocal
- eclipse.convertpath

Pour plus d'informations sur ces scripts, voir la documentation Eclipse.

Paramètres de la tâche fb.exportReleaseBuild

Attribut	Description	Obligatoire ?	Valeur par défaut
application	Nom de l'application à compiler. Vous pouvez spécifier simplement le nom de l'application, sans chemin d'accès ou extension (app1, par exemple). Pour éviter toute ambiguïté de nom, vous pouvez spécifier un chemin d'accès complet par rapport à la racine du projet (src/app1.mxml, par exemple). Pour compiler toutes les applications, spécifiez « * » ou omettez cet attribut. Lorsque vous exécutez un projet AIR, vous ne pouvez spécifier qu'une seule application. La valeur « * » n'est pas autorisée.	Non	Application par défaut du projet.
basefilename	Pour les projets mobiles et de bureau AIR : nom du package à exporter. Spécifiez le nom du package sans préciser l'extension de fichier.	Oui	Nom de l'application sélectionnée.
certificate	Pour les projets mobiles et AIR : lors de l'export d'un fichier AIR ou d'un fichier de programme d'installation natif, chemin du certificat utilisé pour signer le package. Remarque : en cas d'omission, la valeur de l'attribut de certificat présente dans le fichier .actionScriptProperties du projet est utilisée. Si l'attribut de certificat n'existe pas dans le fichier .actionScriptProperties, un message d'erreur s'affiche.	Non	Sans objet

Attribut	Description	Obligatoire ?	Valeur par défaut
destdir	Spécifie le dossier de sortie dans lequel exporter le package validé. Le dossier peut être un chemin relatif ou absolu. Si vous spécifiez un chemin relatif, il est relatif par rapport à la racine du projet.	Non	« bin-release » pour les projets Web et «» (indique la racine du projet) pour tous les autres types de projets
failonerror	Indique si des erreurs de compilation provoquent l'échec de la génération.	Non	true
locale	Indique la langue, par exemple fr-FR. Cette valeur est transmise au compilateur à l'aide de l'identifiant -locale du compilateur. Si spécifiée, cette valeur remplace la langue indiquée dans le champ Arguments de compilateur supplémentaires de Flash Builder.	Non	Sans objet
packagetype	Pour les projets mobiles et de bureau AIR : spécifie le type de package à utiliser lors de l'export du package validé. Pour les projets mobiles, vous pouvez spécifier les types « air », « airi » ou « platform ». Si vous spécifiez « air », un seul fichier AIR commun à toutes les plateformes est généré. Si vous spécifiez « airi », un fichier airi est généré pour chaque plateforme. Si vous spécifiez « platform », un fichier spécifique est généré pour chaque plateforme. Par exemple, un fichier APK est généré pour la plateforme Android et un fichier IPA est généré pour la plateforme iOS. Dans le cas de projets de bureau AIR, vous pouvez spécifier les types de package « air », « airi » ou « native ». Chaque option génère un seul fichier.	Oui	Sans objet
password	Pour les projets AIR uniquement : le mot de passe du certificat qui est utilisé pour signer le fichier AIR. Si cet argument est omis, un message d'erreur s'affiche. **Avertissement** : l'utilisation d'une valeur littérale pour un mot de passe risque de compromettre la sécurité.	Non	Sans objet
platform	Pour les projets mobiles uniquement : spécifie la plateforme cible vers laquelle exporter la version validée. Vous pouvez spécifier la plateforme en tant que « android », « ios » ou « qnx ». Utilisez une liste séparée par des virgules pour spécifier plusieurs plateformes. **Remarque :** dans le cas d'un projet mobile avec le type de package « air », cet attribut n'est pas requis.	Oui	Sans objet
project	Projet à générer. Indiquez le nom d'un projet dans l'espace de travail Flash Builder, sans chemin d'accès. Par exemple : « MonProjetFlex ».	Oui	Sans objet

Attribut	Description	Obligatoire ?	Valeur par défaut
publishsource	Indique si la source de l'application doit être publiée, ce qui permet à l'utilisateur d'afficher les fichiers source à l'aide du menu contextuel Afficher la source.	Non	false
timestamp	Pour les projets AIR uniquement : indique si le fichier AIR généré inclut un horodatage. Remarque : en cas d'omission, la valeur de l'attribut d'horodatage présente dans le fichier .actionScriptProperties du projet est utilisée. Si l'attribut d'horodatage n'existe pas dans le fichier .actionScriptProperties du projet, la valeur par défaut « false » est utilisée.	Non	false
verbose	La tâche <fb.exportReleaseBuild> génère des informations supplémentaires. Par exemple, elle répertorie les fichiers qui ont été assemblés dans le fichier AIR et la durée de chaque étape du processus.	Non	false

Spécification des attributs spécifiques à la plateforme pour les projets mobiles

Vous pouvez exporter les packages validés simultanément vers plusieurs plateformes en spécifiant des attributs spécifiques à la plateforme. Vous pouvez utiliser les attributs spécifiques à la plateforme en ajoutant le préfixe spécifique à la plateforme. Par exemple, pour utiliser l'attribut « certificate » pour la plateforme Android, spécifiez « android.certificate » et pour la plateforme iOS, spécifiez « ios.certificate ».

Plateforme Android

Attribut	Description	Obligatoire ?
android.airDownloadURL	URL de téléchargement de l'environnement d'exécution Adobe AIR si celui-ci n'est pas installé sur un périphérique de l'utilisateur au lancement de l'application. **Remarque :** si vous ne spécifiez pas cet attribut, l'URL Android Market par défaut est utilisée.	Non
android.certificate	Chemin du certificat utilisé pour signer le fichier APK. Vous spécifiez cet attribut lorsque vous définissez l'attribut « packagetype » sur « platform » et que l'une des plateformes cibles est Android.	Oui, si vous n'avez pas défini le chemin de certificat correspondant à la plateforme Android dans le fichier .actionScriptProperties.
android.password	Mot de passe du certificat utilisé pour signer le fichier APK. Si cet attribut est omis, un message d'erreur s'affiche.	Oui

Plateforme iOS

Attribut	Description	Obligatoire ?
ios.certificate	Chemin du certificat (extension du fichier .p12) utilisé pour signer le fichier IPA. Vous spécifiez cet attribut lorsque vous définissez l'attribut « packagetype » sur « platform » et que l'une des plateformes cibles est Apple iOS.	Oui, si vous n'avez pas défini le chemin de certificat dans le fichier .actionScriptProperties.
ios.packagetype	Type de package utilisé lors de l'export d'un fichier IPA. Spécifiez le type de package en tant que « adhoc » pour la création de packages ad hoc et « appstore » pour la création de packages destinés à l'App Store d'Apple.	Oui, si vous n'avez pas défini le type de package dans le fichier .actionScriptProperties.
ios.password	Mot de passe du certificat utilisé pour signer le fichier IPA. Si cet argument est omis, un message d'erreur s'affiche.	Oui
ios.provisioning	Chemin du fichier de configuration (extension de fichier .mobileprovision) utilisé pour signer le fichier IPA.	Oui, si vous n'avez pas défini le fichier de configuration dans le fichier .actionScriptProperties.

Assistant Exporter vers une version validée

Lorsque vous exécutez l'assistant Exporter vers une version validée (Projet > Exporter vers une version validée), les paramètres que vous définissez dans l'assistant sont enregistrés dans le fichier .actionScriptProperties. Une génération par ligne de commande qui utilise la tâche fb.exportReleaseBuild récupère les paramètres de l'assistant. L'assistant Exporter vers une version validée enregistre les paramètres suivants :

- Afficher la source

  Les fichiers source que vous spécifiez pour l'option Afficher la source sont enregistrés. Si vous définissez le paramètre publishsource sur fb.exportReleaseBuild, l'assistant inclut ces fichiers comme fichiers source pouvant être affichés.

  Important : *pour les projets pour serveurs, vous pouvez sélectionner le dossier des services à l'exportation des fichiers source. L'exportation de fichiers implémentant des services a des implications en termes de sécurité. Ces fichiers peuvent donner accès à votre base de données, noms d'utilisateurs et mots de passe inclus. Voir Exportation des fichiers source vers la version validée d'une application.*

- Pour les projets Adobe AIR, tous les fichiers de sortie supplémentaires que vous indiquez à l'assistant d'inclure avec le fichier AIR ou AIRI.

Exécution des générations par ligne de commande sur Linux et d'autres plateformes

La tâche <fb.exportReleaseBuild> est uniquement prise en charge sur des plateformes Windows et Mac.

Si vous rédigez toutefois un script de génération pour une autre plateforme, utilisez l'option `-dump-config` dans le compilateur mxmlc ou compc pour écrire les paramètres de configuration du compilateur dans un fichier. L'option -load-config vous permettra ensuite de lire les options de configuration.

Modifiez les paramètres de configuration du fichier en fonction de vos besoins. Par exemple, changez <debug>true</debug> en <debug>false</debug> pour que la génération nocturne génère une version validée.

Exécution d'une génération par ligne de commande à l'aide des paramètres du compilateur Flash Builder

1. Dans Flash Builder, sélectionnez Projet > Propriétés > Compilateur Flex.
2. Dans le champ Arguments de compilateur supplémentaires, indiquez l'argument suivant :

   -dump-config *nom_du_chemin*, où *nom_du_chemin* spécifie le chemin d'accès absolu à un fichier du système.
3. Appliquez les modifications dans la fenêtre du projet.

   Les paramètres du compilateur sont écrits dans le fichier spécifié. Supprimez l'argument -dump-config après avoir vérifié que le fichier a été écrit.
4. Modifiez les paramètres de configuration comme il convient.
5. Dans le script de génération, exécutez le compilateur afin qu'il intègre les paramètres enregistrés du compilateur :

   mxmlc -load-config *nom_du_chemin*

Restrictions pour l'exécution des générations par ligne de commande

Certaines restrictions s'appliquent à l'exécution des générations par ligne de commande à l'aide de la tâche <fb.exportReleaseBuild>.

Exécution des générations par ligne de commande sur des plateformes 64 bits

Flash Builder s'exécute sur des plateformes qui implémentent Java 32 bits. Pour exécuter une génération par ligne de commande sur des plateformes prenant en charge Java 64 bits (Mac OS X 10.6, par exemple), ajoutez `-d32` aux options de ligne de commande qui sont transmises à Java. Par exemple :

```
java -d32 ...
```

Publication du code source

Une fois qu'une application est prête à être publiée, Flash Builder vous permet de définir si les utilisateurs peuvent visionner le code source ainsi que les ressources de l'application. A l'instar du langage HTML, les utilisateurs peuvent accéder à la source et la visionner dans un navigateur Web en sélectionnant Afficher la source dans le menu contextuel. Le visualiseur de source met en forme et colore le code afin d'en faciliter la lecture. Il constitue également un moyen pratique de partager le code avec d'autres développeurs Flex et ActionScript 3.0.

Activation de l'option d'affichage de la source

1. Ouvrez l'ensemble du projet d'application dans l'éditeur, puis sélectionnez Projet > Exporter vers une version validée.
2. Sélectionnez Activer l'affichage de la source ou, pour les projets ActionScript, Inclure la source.
3. Cliquez sur Choisir des fichiers source.
4. Dans la boîte de dialogue Publier la source de l'application, sélectionnez le ou les fichiers de l'application à inclure dans le menu Afficher la source. Par défaut, le fichier de l'application principale est sélectionné.

Important : *pour les projets pour serveurs, vous pouvez sélectionner le dossier des services à l'exportation des fichiers source. L'exportation de fichiers implémentant des services a des implications en termes de sécurité. Ces fichiers peuvent donner accès à votre base de données, noms d'utilisateurs et mots de passe inclus. Voir Exportation des fichiers source vers la version validée d'une application.*

5 (Facultatif) Modifiez le dossier de sortie de la source. Par défaut, un dossier d'affichage de la source est ajouté au dossier de sortie du projet.

6 Cliquez sur OK.

Les utilisateurs exécutant l'application peuvent accéder au code source en sélectionnant Afficher la source dans le menu contextuel. Le code source est affiché dans le navigateur Web par défaut sous forme d'arborescence reflétant la structure des ressources (package, dossiers et fichiers) contenues dans l'application (que vous avez décidé de publier). Sélectionnez un élément source pour en afficher le code dans le navigateur. Les utilisateurs peuvent également télécharger tous les fichiers source en cliquant sur le lien vers le fichier Download.zip.

Remarque : *les restrictions de sécurité d'Internet Explorer risquent d'entraver l'affichage du code source sur votre ordinateur local. Dans ce cas, déployez l'application sur un serveur Web pour afficher la source.*

Ajout du menu Afficher la source aux projets ActionScript

Dans les projets Flex, l'ajout de l'option Afficher la source à l'application s'effectue par l'intermédiaire de l'assistant Exporter vers une version validée. Dans les applications ActionScript, ajoutez cette option manuellement.

La fonction suivante, contenue dans la structure Flex, peut être utilisée dans un constructeur d'application ActionScript pour activer le menu d'affichage de la source :

```
com.adobe.viewsource.ViewSource.addMenuItem(obj:InteractiveObject, url:String,
hideBuiltins:Boolean = true)
```

L'exemple suivant illustre l'utilisation du code dans les applications ActionScript :

```
package {
    import flash.display.MovieClip;
    import com.adobe.viewsource.ViewSource;

    public class MyASApp extends MovieClip
    {
        public function MyASApp()
        {
            ViewSource.addMenuItem(this, "srcview/index.html");

            // ... additional application code here
        }
    }

}
```

Cet exemple se base sur l'emplacement par défaut du dossier source (srcview). Si vous modifiez l'emplacement du dossier source, vérifiez que le code utilise l'emplacement approprié.

Exécution et débogage des applications

Les applications sont exécutées (et déboguées) en fonction d'une configuration de lancement. La configuration de lancement d'une nouvelle application Flex ou ActionScript spécifie l'emplacement des fichiers d'application générés et du fichier de l'application principale. Les configurations de lancement peuvent être modifiées ou personnalisées. Pour plus d'informations, voir « Création de configurations de lancement personnalisées » à la page 108.

L'exécution des projets dans Flash Builder peut être effectuée de différentes manières. Vous pouvez par exemple utiliser la commande d'exécution à partir de la barre d'outils et du menu principal du workbench, de la vue Explorateur de packages et des menus contextuels de l'éditeur de code.

Remarque : *le bouton Exécuter comporte deux éléments : le bouton d'action principal et la liste déroulante affichant les fichiers d'application du projet pouvant être exécutés ou débogués. Cliquez sur le bouton d'action principal pour exécuter le fichier d'application par défaut. Vous pouvez également activer la liste déroulante et sélectionner l'un des fichiers d'application du projet ainsi que créer ou modifier une configuration de lancement dans la boîte de dialogue Création, gestion et exécution d'une configuration.*

Gestion des configurations de lancement

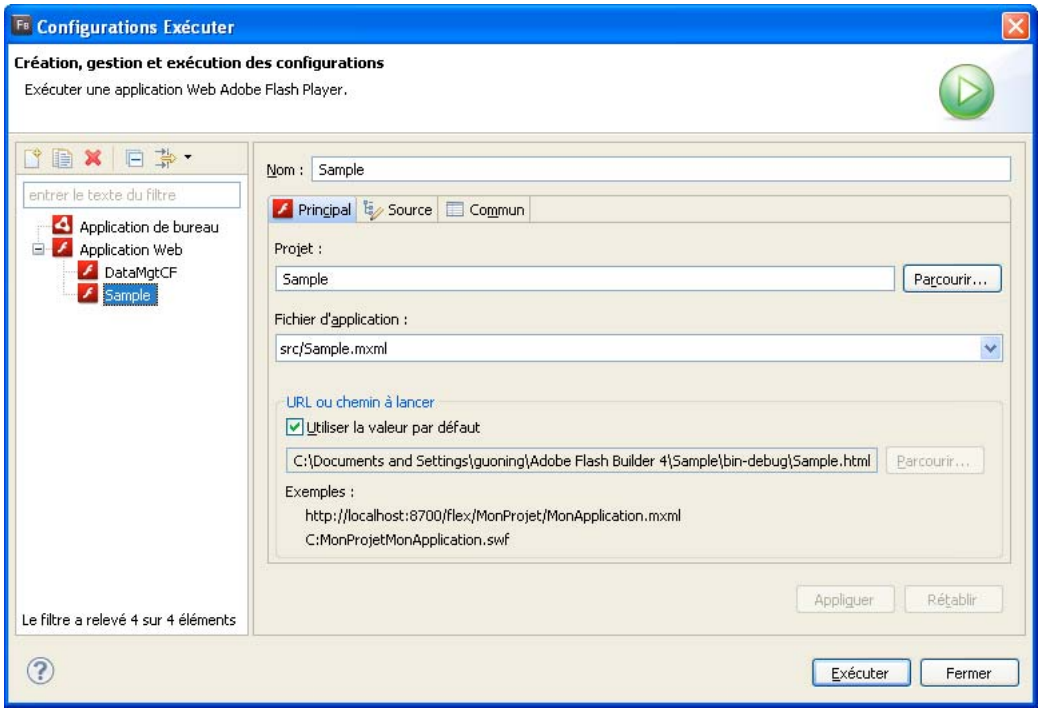
Les configurations de lancement permettent d'exécuter et de déboguer des applications. Flash Builder comporte une configuration de lancement par défaut pour les applications Flex et ActionScript.

Une configuration de lancement définit le nom du projet, le fichier de l'application principale et le chemin des versions d'exécution et de débogage de l'application. Flash Builder contient une configuration de lancement d'application par défaut. Cette configuration est utilisée pour créer automatiquement des configurations de lancement pour chacun de vos projets.

Création ou modification des configurations de lancement pour l'exécution ou le débogage des applications Web

Un projet créé et généré est prêt à être exécuté ou débogué. L'exécution et le débogage des applications du projet sont contrôlés par une configuration de lancement. Par défaut, Flash Builder crée une configuration de lancement pour chaque fichier d'application du projet exécuté ou débogué pour la première fois. Les configurations se basent sur la configuration d'application par défaut et peuvent être modifiées.

Les configurations de lancement sont gérées dans la boîte de dialogue Création, gestion et exécution des configurations.



1 Dans l'Explorateur de packages Flex, sélectionnez un projet.

2 Ouvrez un fichier de projet dans l'éditeur de code, puis sélectionnez Exécuter > Configurations Exécuter.

3 Sélectionnez la configuration de lancement que vous souhaitez modifier.

4 Adaptez les paramètres de la configuration, puis cliquez sur Exécuter ou Déboguer.

L'exécution des projets Web conduit à l'ouverture du fichier principal SWF de l'application dans le navigateur Web par défaut ou directement dans la version autonome de Flash Player.

Création ou modification d'une configuration de lancement sous Windows 8

Pour exécuter ou déboguer une application Web sous Windows 8, reportez-vous à cet article technique.

Voir aussi

« Modification du navigateur Web par défaut » à la page 111

« Exécution du fichier SWF de l'application dans la version autonome de Flash Player » à la page 111

Création ou modification des configurations de lancement pour l'exécution ou le débogage des applications de bureau

Lors de l'exécution ou du débogage des applications de bureau, vous pouvez créer des configurations de lancement pour remplacer les configurations par défaut fournies par Flash Builder.

Lors de la création ou de la modification d'une configuration de lancement pour une application de bureau, vous devez spécifier des arguments de ligne de commande, un ID d'éditeur, un profil et des paramètres de format d'écran. Pour les applications de bureau, la valeur de Profil peut être soit desktop, soit extendedDesktop.

Pour créer une configuration de lancement :

1 Sélectionnez Exécuter > Exécuter en tant que.

Pour les configurations de lancement de débogage, sélectionnez Exécuter > Déboguer en tant que. Pour les configurations de lancement de profil, spécifiez Exécuter > Profiler en tant que.

2 Sélectionnez le projet de bureau. Cliquez sur le bouton de création d'une configuration de lancement.

Accédez aux onglets Principal, Source et Commun pour spécifier les paramètres de la configuration.

3 Sélectionnez Exécuter, Déboguer ou Profiler pour exécuter l'application avec les paramètres définis.

Cliquez sur Fermer pour enregistrer les paramètres.

Création ou modification des configurations de lancement pour l'exécution ou le débogage des applications mobiles

Avant d'exécuter ou de déboguer une application mobile pour la première fois, ouvrez la fenêtre Configurations de lancement pour définir une configuration de lancement. Vous pouvez créer une ou plusieurs configurations de lancement pour une application. Le jeu de configurations de lancement que vous définissez est partagé entre l'exécution et le débogage de l'application.

1 Sélectionnez Exécuter > Configurations Exécuter pour ouvrir la boîte de dialogue d'exécution des configurations.

Pour ouvrir la boîte de dialogue Configurations Déboguer, sélectionner Exécuter > Configurations Déboguer. Voir « Test et débogage d'une application mobile sur un périphérique » à la page 241.

Vous pouvez également accéder au menu Configurations Exécuter ou Déboguer dans la liste déroulante du bouton Exécuter ou Déboguer de la barre d'outils Flash Builder.

2 Développez le nœud des applications mobiles. Cliquez sur le bouton Crée une configuration de lancement dans la barre d'outils de la boîte de dialogue.

3 Sélectionnez une plateforme cible dans la liste déroulante.

4 Spécifiez une méthode de lancement :

- Dans le simulateur AIR

  exécute l'application sur votre ordinateur à l'aide du AIR Debug Launcher, selon une configuration de périphérique spécifiée. Cette méthode de lancement n'est pas une vraie émulation d'exécution de l'application sur un périphérique. Elle vous permet toutefois de voir la mise en forme de l'application et d'interagir avec l'application.

  Voir « Prévisualisation des applications avec ADL » à la page 239.

  Cliquez sur Configurer pour modifier les configurations du périphérique. Voir « Définition des configurations de périphériques » à la page 225.

- Sur le périphérique :

  installez et exécutez l'application sur votre périphérique.

  Flash Builder installe l'application sur votre périphérique et lance l'application. Flash Builder accède au périphérique connecté au port USB de l'ordinateur. Pour plus d'informations, voir « Test et débogage d'une application mobile sur un périphérique » à la page 241.

  Les plateformes Windows nécessitent un pilote USB pour connecter un périphérique Android à l'ordinateur. Pour plus d'informations, voir « Installation des pilotes de périphérique USB pour les périphériques Android (Windows) » à la page 234.

- Dans un simulateur iOS (plateforme Apple iOS uniquement)

  Important : *le simulateur iOS est pris en charge uniquement sous Mac et non sous Windows.*

  Le simulateur iOS constitue un moyen rapide d'exécuter et de déboguer les applications iOS sans utiliser de périphérique. Pour plus d'informations, voir « Test et débogage d'une application iOS à l'aide d'un simulateur » à la page 240.

5 Sélectionnez la cible de génération et cliquez sur Appliquer.

Lorsque vous spécifiez une configuration de lancement pour l'exécution ou le débogage d'une application, Flash Builder sélectionne par défaut les paramètres de cible de génération applicables. Par exemple, si vous créez une configuration de lancement pour effectuer un débogage sur un périphérique, périphérique est sélectionné comme cible de génération par défaut. Toutefois, vous pouvez modifier la sélection par défaut et sélectionner la cible de génération personnalisée que vous avez créée. Pour plus d'informations sur la création de cibles de génération, voir « Propriétés des projets pour la création de packages de projets mobiles » à la page 118.

Recherche de périphériques connectés

Lorsque vous exécutez ou déboguez une application mobile sur un périphérique, Flash Builder vérifie les périphériques connectés. Si Flash Builder détecte un seul périphérique connecté en ligne, il déploie et lance l'application. Dans le cas contraire, Flash Builder lance la boîte de dialogue Choisir un périphérique pour les scénarios suivants :

- Aucun périphérique n'est trouvé
- Un seul périphérique déconnecté est trouvé
- Plusieurs périphériques connectés trouvés

  La boîte de dialogue Choisir un périphérique répertorie les périphériques et leur état (en ligne ou déconnecté). Sélectionnez le périphérique à lancer.

Débogage des applications mobiles

Vous pouvez déboguer une application mobile à partir de votre ordinateur de développement ou d'un périphérique. Avant de déboguer une application mobile, créez une configuration de lancement. Voir « Gestion des configurations de lancement » à la page 105.

Le débogage d'un périphérique Android nécessite l'installation d'Android 2.2 ou d'une version ultérieure. Lorsque vous exécutez une application sur un périphérique à partir de Flash Builder, Flash Builder installe une version déboguée de l'application sur le périphérique.

Pour plus d'informations, voir « Débogage d'une application sur un périphérique Google Android » à la page 241.

Le débogage d'une application sur un périphérique Apple iOS nécessite de grouper l'application sous forme de fichier IPA (iOS Package) de débogage et d'installer manuellement l'application sur le périphérique connecté.

Pour plus d'informations, voir « Débogage d'une application sur un périphérique Apple iOS » à la page 243.

Remarque : *si vous exportez une version validée vers un périphérique, vous installez une version non déboguée de l'application. Toutefois, la version non déboguée ne convient pas pour le débogage.*

Création de configurations de lancement personnalisées

Les configurations de lancement diffèrent selon que vous exécutez une application Web, d'ordinateur ou mobile. Vous pouvez personnaliser les configurations de lancement que Flash Builder crée automatiquement pour vous.

1 Dans le menu Flash Builder, sélectionnez Exécuter > Configurations Exécuter.

2 Dans la boîte de dialogue Création, gestion et exécution des configurations, sélectionnez le type d'application à configurer. Cliquez sur le bouton Nouveau de la barre d'outils.

3 Spécifiez un nom, un projet et un fichier d'application pour la configuration de lancement.

4 Modifiez les propriétés de configuration requises pour le type d'application que vous configurez.

5 Cliquez sur Appliquer pour enregistrer les paramètres ou sur Exécuter pour lancer l'application.

Personnalisation de la configuration de lancement pour lancer manuellement le navigateur

Lorsque vous exécutez l'application, Flash Builder lance automatiquement le navigateur. Vous pouvez, toutefois, décider de lancer manuellement le navigateur à la place de Flash Builder. Pour ce faire :

1 Dans la liste des configurations, sélectionnez la configuration de lancement du projet.

2 Sous l'onglet Principal, désélectionnez l'option Utiliser la valeur par défaut.

3 Modifiez l'URL ou le chemin par défaut pour effectuer le lancement sur about:blank.

Exécution de la dernière configuration lancée

❖ Cliquez sur le bouton Exécuter situé dans la barre d'outils principale.

Si une configuration de lancement doit être créée, Flash Builder ouvre la boîte de dialogue Configurations de lancement.

Voir aussi

« Création ou modification des configurations de lancement pour l'exécution ou le débogage des applications Web » à la page 105

« Création ou modification des configurations de lancement pour l'exécution ou le débogage des applications de bureau » à la page 106

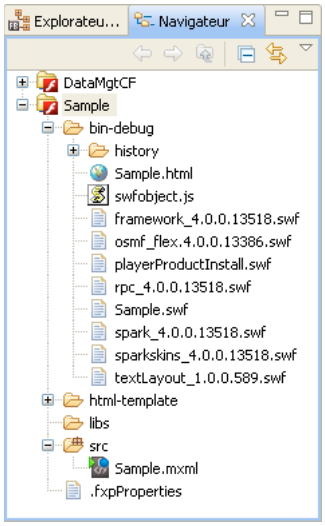
« Création ou modification des configurations de lancement pour l'exécution ou le débogage des applications mobiles » à la page 107

Version déboguée de l'application

La version déboguée de l'application contient des informations de débogage qui servent au débogage de l'application. La version validée résultant de l'exportation ne contient aucune information de débogage. La taille de son fichier est donc inférieure à celle du fichier de la version déboguée. Un fichier d'enveloppe HTML contient un lien vers le fichier SWF de l'application. Il intervient dans l'exécution ou le débogage de l'application dans un navigateur Web.

Remarque : *les commandes Exécuter et Déboguer lancent la version de développement dans le dossier bin-debug (et non dans le dossier de version validée, bin-release).*

Dans une application standard, le dossier de sortie ressemble à l'exemple suivant.



Vous pouvez exécuter ou déboguer vos applications Flex et ActionScript dans un navigateur, dans Adobe AIR ou sur votre périphérique mobile. Pour définir la manière dont les applications sont exécutées ou déboguées, modifiez la configuration de lancement du projet (voir « Gestion des configurations de lancement » à la page 105).

Voir aussi

« Débogage de votre application » à la page 124

Utilisation de la version de débogage de Flash Player

Par défaut, Flash Builder exécute la version de débogage de Flash Player. Cette version est disponible sous forme de plug-in ou de contrôle ActiveX pour navigateur ou en tant que version autonome. Elle est installée avec Flash Builder mais peut également être téléchargée à partir du site Web d'Adobe.

Les programmes d'installation de la version de débogage de Flash Player se trouvent dans le répertoire *Installation d'Adobe Flash Builder*/player de votre ordinateur.

Vous pouvez déterminer par programmation la version de Flash Player exécutée en utilisant la méthode `Capabilities.isDebugger()`. Pour plus d'informations, voir Determining Flash Player version in Flex.

Lorsque vous lancez ou déboguez une application à partir de Flash Builder, vous pouvez spécifier la version de Flash Player à utiliser.

Voir aussi

« Gestion de la version du lecteur Flash Player autonome » à la page 260

Exécution et débogage des applications dans un navigateur

Par défaut, les chemins d'exécution et de débogage de la configuration de lancement des applications Web dirigent vers les fichiers d'enveloppe HTML situés dans le dossier de sortie du projet. Les applications sont donc exécutées et déboguées dans la version Flash Player intégrée dans le navigateur Web utilisé. Vous avez également la possibilité d'exécuter et de déboguer les applications dans la version autonome de Flash Player (voir « Exécution du fichier SWF de l'application dans la version autonome de Flash Player » à la page 111). Vous pouvez en outre remplacer le navigateur Web standard par tout autre navigateur installé sur votre ordinateur (voir « Modification du navigateur Web par défaut » à la page 111).

Modification du navigateur Web par défaut

L'exécution et le débogage des applications sont effectués dans le navigateur Web défini par défaut. Vous ne pouvez pas définir un navigateur Web différent pour chaque configuration de lancement. Vous pouvez toutefois modifier le navigateur Web défini pour le workbench, ce qui aura une incidence sur l'exécution et le débogage de toutes les applications.

1 Ouvrez la boîte de dialogue Préférences et sélectionnez Général > Navigateur Web.

2 Sélectionnez un navigateur dans la liste de ceux installés sur le système.

Remarque : *l'option Utiliser le navigateur Web interne ne concerne pas l'exécution et le débogage des applications, ces opérations étant toujours effectuées dans un navigateur Web externe.*

Vous pouvez également ajouter des navigateurs à la liste, en supprimer ou éditer ceux qui y sont répertoriés.

3 Cliquez sur OK pour appliquer les modifications.

Exécution du fichier SWF de l'application dans la version autonome de Flash Player

Les applications peuvent être exécutées ou déboguées dans la version autonome de Flash Player. Il vous suffit pour ce faire de modifier légèrement la configuration de lancement. Vous pouvez spécifier la version de Flash Player à utiliser lorsque vous exécutez et déboguez vos applications. Pour plus d'informations, voir « Gestion de la version du lecteur Flash Player autonome » à la page 260.

Exécution et débogage des applications dans la version autonome de Flash Player

1 Sélectionnez Exécuter > Configurations Exécuter dans le menu de Flash Builder.

2 Dans la boîte de dialogue Création, gestion et exécution des configurations, sélectionnez la configuration de lancement que vous souhaitez modifier.

3 Sous l'onglet Principal, désélectionnez l'option Utiliser la valeur par défaut.

4 Sous URL ou chemin à lancer, cliquez sur Parcourir.

La boîte de dialogue de sélection d'un fichier s'ouvre, répertoriant le contenu du dossier de sortie de la génération.

5 Sélectionnez le fichier d'application SWF du répertoire bin-debug. Ne sélectionnez pas le fichier SWF de l'éventuel répertoire bin-release, étant donné qu'il ne contient aucune information de débogage.

6 Cliquez sur Ouvrir pour sélectionner le fichier et retourner à la boîte de dialogue de configuration.

7 Validez les modifications apportées et exécutez ou déboguez l'application en utilisant la configuration modifiée.

Comparaison des versions déboguées et non déboguées d'une application

Par défaut, Flash Builder génère des versions déboguées du fichier SWF de l'application Flex et les enregistre dans le répertoire bin-debug du projet. Le volume de cette application est supérieur à celui d'une version non déboguée car il contient du code et des métadonnées supplémentaires utilisées par le débogueur.

Pour générer une version non déboguée de l'application, effectuez l'une des opérations suivantes :

- Sélectionnez Projet > Exporter vers une version validée. Cette opération crée un fichier SWF ou un fichier AIR non débogué dans le répertoire bin-release.
- Ajoutez `-debug=false` dans le champ Arguments de compilateur supplémentaires. Ceci faisant, un fichier SWF non débogué est généré où que vous exportiez le projet.

Exportation d'une application vers une version validée

Après avoir terminé la création de votre application, si vous souhaitez la publier, vous pouvez exporter une version validée de votre application. L'assistant Exporter vers une version validée de Flash Builder crée une version validée optimisée (fichier SWF ou AIR sans débogage) de votre application.

En fonction du type d'application, l'assistant vous guide à travers les étapes de personnalisation et d'optimisation de la création de package.

Une fois l'assistant exécuté, vous devrez procéder à des opérations supplémentaires pour déployer l'application sur un serveur.

Exportation d'une application Web (exécution dans Adobe Flash Player)

1. Sélectionnez Projet > Exporter vers une version validée pour ouvrir l'assistant correspondant.
2. Sélectionnez le projet et l'application que vous souhaitez exporter.
3. (Facultatif) Sélectionnez Activer l'affichage de la source pour permettre l'accès aux fichiers source à partir de l'application exportée.

   Cliquez sur Choisir des fichiers source pour spécifier les fichiers source à inclure. Outre les fichiers source spécifiés, l'assistant génère un fichier archive ZIP contenant les fichiers source.

   Important : *pour les projets pour serveurs, vous pouvez sélectionner le dossier des services à l'exportation des fichiers source. L'exportation de fichiers implémentant des services a des implications en termes de sécurité. Ces fichiers peuvent donner accès à votre base de données, noms d'utilisateurs et mots de passe inclus. Voir Exportation des fichiers source vers la version validée d'une application.*
4. Cliquez sur Terminer.
5. Copiez le dossier contenant la version exportée vers la racine Web du serveur hébergeant l'application.
6. (Projets pour serveurs) Si vous exportez la version d'un projet spécifiant un type de serveur d'applications, déployez les services et les autres fichiers côté serveur à la racine Web du serveur cible.

   Conservez la structure de répertoire utilisé lors du développement.

   Cette étape s'applique aux services ColdFusion, PHP, BlazeDS et Data Services. Le type de serveurs d'applications est défini à la création du projet dans Flash Builder.

Si le serveur d'applications se trouve sur un hôte différent que l'application déployée, un fichier de régulation interdomaines est nécessaire pour accéder à ces services. Cette manière de procéder s'applique aux projets accédant à des fichiers de service XML statiques ou à des fichiers locaux pour des services HTTP ou Web. Voir Utilisation de fichiers de régulation interdomaines.

7 (Projets pour serveurs PHP uniquement) Exécutez les étapes supplémentaires suivantes :

a Installez la structure Zend Framework sur le serveur. Voir Installation de Zend Framework.

b Modifiez le fichier `amf-config.ini` qui se trouve dans le dossier de sortie de la version exportée.

Pour `zend_path`, spécifiez le chemin absolu du répertoire d'installation de Zend Framework.

Définissez `amf.production` sur **true**.

Saisissez pour `webroot` le chemin absolu vers la racine Web du serveur.

Exportation d'une application de bureau (exécution dans Adobe AIR)

1 (Facultatif) Modifiez les paramètres du serveur dans les propriétés du projet.

Une application de bureau exportée peut accéder uniquement aux services utilisés au cours du développement. Si vous souhaitez modifier le serveur pour l'application de bureau exportée, modifiez les paramètres du projet.

2 Sélectionnez Projet > Exporter vers une version validée pour ouvrir l'assistant correspondant.

3 Sélectionnez le projet et l'application que vous souhaitez exporter.

Remarque : *pour les applications AIR, Flash Builder ne peut pas rendre disponibles les fichiers sources de l'application à partir de l'application exportée.*

4 Indiquez l'emplacement vers lequel exporter le projet. L'emplacement par défaut est le dossier de votre projet.

5 Pour exporter votre application Adobe AIR, ajoutez une signature numérique à votre génération en sélectionnant l'un des éléments suivants :

- Package AIR signé pour exporter un package AIR à signature numérique.
- Signed Native Installer pour créer un programme d'installation à signature numérique pour la plateforme cible, à savoir Windows ou Mac OS X
- Application signée avec environnement d'exécution intégré pour exporter l'application avec l'environnement d'application Adobe AIR fourni.
- Package AIRI intermédiaire qui pourra être signé plus tard.

Cliquez sur Suivant.

Pour plus d'informations sur la signature numérique, voir la documentation d'Adobe Flash Builder « Signature numérique des applications Adobe AIR » à la page 119.

6 A la page Signature numérique, procédez comme suit :

Spécifiez le certificat numérique représentant l'identité de l'éditeur de l'application. Pour générer un certificat auto-signé, cliquez sur Créer et saisissez les données dans les champs obligatoires.

7 Dans la page Contenu du package, sélectionnez les fichiers de sortie à inclure dans le fichier AIR ou AIRI.

8 Cliquez sur Terminer.

9 Copiez le projet `.air` vers l'ordinateur cible.

10 (Projets pour serveurs) Si vous exportez la version d'un projet spécifiant un type de serveurs d'applications, déployez les services sur le serveur cible.

Cette étape s'applique aux services ColdFusion, PHP, BlazeDS et LCDS. Le type de serveurs d'applications est défini à la création du projet dans Flash Builder.

11 (Projets pour serveurs PHP uniquement) Exécutez les étapes supplémentaires suivantes :

a Installez la structure Zend Framework sur le serveur. Voir Installation de Zend Framework.

b Modifiez le fichier `amf-config.ini` qui se trouve dans le dossier de sortie de la version exportée.

Pour `zend_path`, spécifiez le chemin absolu du répertoire d'installation de Zend Framework.

Définissez `amf.production` sur `true`.

Saisissez pour `webroot` le chemin absolu vers la racine Web du serveur.

Création d'un fichier AIR intermédiaire

Vous pouvez créer un fichier AIR intermédiaire et le signer ultérieurement. Utilisez cette option uniquement lorsque vous procédez à des tests.

1 Sélectionnez Projet > Exporter vers une version validée.

Sélectionnez le projet Adobe AIR à exporter ainsi que le fichier vers lequel vous souhaitez exporter le projet. Cliquez sur Suivant.

2 Sélectionnez l'option Exporter un fichier AIRI intermédiaire qui sera signé ultérieurement.

3 (Facultatif) Cliquez sur Suivant. Sélectionnez les fichiers de sortie qui devront être inclus dans le fichier AIRI exporté.

Par défaut, tous les fichiers sont inclus.

4 Cliquez sur Terminer.

Une fois généré, le fichier AIR intermédiaire peut être signé en utilisant l'outil ADT (AIR Developer Tool). Pour plus d'informations sur l'outil de ligne de commande ADT, voir Signature d'un fichier intermédiaire AIR à l'aide de l'outil ADT dans la documentation d'Adobe AIR.

Pour plus d'informations à propos de l'utilisation de l'outil AIR Developer Tool (ADT), voir la documentation Adobe AIR à l'adresse www.adobe.com/go/learn_fbairdevelopertool_fr.

Exécution d'une application avec un moteur d'exécution captif

Les procédures d'exécution de l'application à l'issue du déploiement final diffèrent selon le système d'exploitation vers lequel le package est exporté avec l'environnement d'exécution Adobe AIR (Windows ou Mac).

Sous Windows

❖ Le package de l'application contient les fichiers d'application requis ainsi que l'environnement d'exécution pour exécuter l'application. Les utilisateurs peuvent exécuter l'application immédiatement après la création du package. Vous pouvez également utiliser des outils tiers pour créer un Windows Installer (MSI) à partir du package exporté de l'application.

Sous Mac

❖ Pour exécuter l'application, faites glisser le package d'application dans le dossier Applications.

Conversion de projets Flex en projets Adobe AIR

Vous pouvez convertir le type d'applications d'un projet Flex d'application Web (exécutée dans Flash Player) en application de bureau (exécutée dans Adobe AIR). La conversion entraîne les modifications suivantes :

- Un fichier descripteur AIR est créé pour chaque application du projet.
- Les configurations de lancement du projet sont mises à jour de manière à être lancées correctement dans Adobe AIR.
- Les paramètres des enveloppes HTML sont supprimés.
- Les paramètres personnalisés de Flash Player sont supprimés.
- Le chemin d'accès à la bibliothèque est modifié pour inclure le fichier airglobal.swc au lieu du fichier playerglobal.swc.

Au cours de la conversion, vous pouvez définir pour chaque application du projet si vous souhaitez modifier les balises Application de base en balises WindowedApplication. Si vous décidez de convertir ces balises, cette modification sera la seule à intervenir dans le code d'application au cours de la conversion. Après la conversion, examinez les attributs des balises de base pour vérifier que l'application est exécutée correctement dans Adobe AIR.

Modification d'un projet d'application Web en projet d'application de bureau

1. Sélectionnez le projet que vous souhaitez convertir.

   Il doit s'agir d'un projet Flex avec type d'application Web (s'exécutant dans Flash Player).

2. Dans le menu contextuel du projet, sélectionnez Ajouter/Modifier le type de projet > Convertir en projet Bureau/Adobe AIR.

3. Dans la boîte de dialogue Convertir en projet Bureau/Adobe AIR, spécifiez si vous souhaitez apporter des modifications au code :

   - Convertir les balises Application en WindowedApplication

     Les balises Application de toutes les applications existantes du projet sont converties en balises WindowedApplication. Aucune autre modification ne sera apportée au code. Examinez les attributs des balises de base pour vérifier que l'application est exécutée correctement dans Adobe AIR.

     Les nouvelles applications créées dans le projet sont des applications de bureau pouvant être exécutées dans Adobe AIR.

   - Ne réécrire aucun code

     Aucune modification n'est apportée au code. Modifiez les applications dans le projet avant de pouvoir les exécuter dans Adobe AIR.

     Les nouvelles applications créées dans le projet sont des applications de bureau pouvant être exécutées dans Adobe AIR.

Remarque : *cette procédure ne peut pas être annulée.*

Exportation d'une application mobile (exécution dans Adobe AIR)

1. Sélectionnez Projet > Exporter vers une version validée pour ouvrir l'assistant correspondant.
2. Sélectionnez le projet et l'application que vous souhaitez exporter.
3. Sélectionnez les plateformes cibles.
4. Indiquez l'emplacement vers lequel exporter le projet. Vous pouvez exporter le projet vers un ordinateur cible ou un périphérique connecté.

Sélectionnez Conserver le dossier bin-release-temp pour conserver le dossier bin-release-temp dans l'espace de travail du projet. Pour plus d'informations, voir « Conservation des dossiers bin-release-temp et dSYM » à la page 253.

5 Par défaut, le nom du fichier de base est le nom du projet. Vous pouvez modifier le nom du fichier, le cas échéant.

6 Pour grouper votre application avec une signature numérique pour chaque plateforme cible, sélectionnez Packages signés pour chaque plateforme cible.

Pour créer un package de votre application en tant qu'application Adobe AIR à signature numérique pour ordinateur, sélectionnez Package AIR signé pour l'installation sur un bureau.

Pour exporter un fichier dont la signature peut être effectuée ultérieurement, exportez un fichier intermédiaire AIRI.

Pour générer un certificat auto-signé, cliquez sur Créer et saisissez les données dans les champs obligatoires.

Cliquez sur Suivant.

Pour plus d'informations sur la signature numérique, voir la documentation de Adobe Flash Builder « Signature numérique des applications Adobe AIR » à la page 119.

7 Dans la page Paramètres de packages, vous pouvez spécifier le certificat numérique et le contenu du package. Selon la plateforme cible, les paramètres varient. Ces paramètres sont utilisés lors de la création du package d'une application à l'aide des flux de travail Exécution/Débogage ou Exporter vers une version validée.

Dans la page Signature numérique, spécifiez le certificat numérique P12 qui représente l'identité de l'éditeur de l'application. Vous pouvez aussi spécifier un mot de passe pour le certificat sélectionné.

Pour la plateforme Apple iOS, convertissez le certificat Apple iPhone au format de fichier P12 et sélectionnez un fichier de configuration. Pour ce faire, vous devez d'abord obtenir un certificat de développement iPhone et un profil de configuration mobile auprès d'Apple.

Pour plus d'informations, voir « Préparation à la génération, au débogage et au déploiement d'une application iOS » à la page 236.

Vous pouvez ensuite choisir de déployer le package de la version finale dans le magasin Apple App ou en tant que package ad-hoc pour une distribution limitée.

Dans la page Contenu du package, sélectionnez les fichiers de sortie que vous souhaitez inclure dans la version finale de votre application.

Remarque : *vous ne pouvez pas exclure le fichier SWF et le fichier descripteur de l'application, car ils sont requis pour exporter un package valide.*

Les fichiers suivants ne sont pas nécessaires et sont par conséquent automatiquement désélectionnés ou masqués :

- Fichiers .ipa
- Fichiers de certificat (.p12, .pfx, .provisioning)
- Fichiers .air, .airi

Pour la plateforme Google Android, sélectionnez Installer et lancer l'application sur tout périphérique connecté pour exporter le projet vers un périphérique connecté. L'application exportée s'installe sur le périphérique.

Si Adobe AIR n'est pas déjà installé sur le périphérique d'un utilisateur, vous pouvez sélectionner ou spécifier une URL de façon à télécharger Adobe AIR pour le package d'application. L'URL par défaut pointe vers un emplacement au sein de l'App Store Android. Vous pouvez toutefois remplacer l'URL par défaut en saisissant votre propre URL ou en sélectionnant l'URL qui pointe vers un emplacement de l'Amazon Appstore.

8 Cliquez sur Terminer.

Pour plus d'informations, voir « Création de packages et exportation » à la page 247.

Création d'un package d'applications Adobe AIR

Une fois l'application achevée et prête à être distribuée (ou testée à partir de l'ordinateur), elle doit être mise en package dans un fichier AIR. La création de packages se compose des étapes suivantes :

- Sélection de l'application AIR à publier
- Affichage du code source (en présence de l'autorisation requise) et sélection des fichiers d'application à inclure
- Sélection d'un fichier AIR générique ou d'un programme d'installation natif pour l'installation de l'application
- Signature numérique de l'application AIR par un certificat commercial de signature de code ou par la création et l'application d'une signature autosignée
- Création (facultative) d'un fichier AIR intermédiaire pouvant être signé ultérieurement

Création de package pour une application Adobe AIR

1. Ouvrez le projet et vérifiez que l'application ne présente aucune erreur de compilation ni d'exécution.
2. Sélectionnez Projet > Exporter vers une version validée.
3. Si plusieurs projets et applications sont ouverts dans Flash Builder, sélectionnez le projet Adobe AIR à assembler.
4. Vous pouvez également sélectionner Activer l'affichage de la source pour permettre aux utilisateurs de visionner le code source à l'exécution de l'application. Cliquez sur Choisir des fichiers source pour sélectionner les fichiers à exclure. Par défaut, tous les fichiers source sont sélectionnés. Pour plus d'informations sur la publication des fichiers source dans Flash Builder, voir l'Aide de Flash Builder.

   Important : *pour les projets pour serveurs, vous pouvez sélectionner le dossier des services à l'exportation des fichiers source. L'exportation de fichiers implémentant des services a des implications en termes de sécurité. Ces fichiers peuvent donner accès à votre base de données, noms d'utilisateurs et mots de passe inclus. Voir Exportation des fichiers source vers la version validée d'une application.*
5. Sélectionnez l'option Exporter vers un fichier AIR ou l'option Exporter vers le programme d'installation natif. Cliquez sur Suivant.

   L'option Exporter vers un fichier AIR crée un fichier de programme d'installation générique permettant l'installation de l'application sur les plateformes Windows ou Mac OS X.

   L'option Exporter vers le programme d'installation natif crée un programme d'installation pour la plateforme cible (Windows ou Mac OS X).
6. Vous avez également la possibilité de renommer le fichier AIR généré. Dès que vous êtes prêt à poursuivre, cliquez sur Suivant pour apposer une signature numérique sur l'application.

Propriétés des projets pour la création de packages de projets de bureau

Vous pouvez gérer le certificat numérique du package et son contenu en sélectionnant un projet et en visualisant ses propriétés.

1. Dans l'Explorateur de packages, sélectionnez un projet.
2. Sélectionnez Projet > Propriétés dans le menu principal ou sélectionnez Propriétés dans le menu contextuel.
3. Dans la boîte de dialogue Propriétés du projet, sélectionnez Paquet de génération Flex.

4 A la page Signature numérique, procédez comme suit :

Sélectionnez le certificat numérique à utiliser pour créer un package de l'application Adobe AIR et la signer. Pour générer un certificat auto-signé, cliquez sur Créer et saisissez les données dans les champs obligatoires.

5 Sur la page Contenu du groupement :

Désélectionnez les fichiers que vous ne souhaitez pas inclure dans la version finale de votre application.

6 Cliquez sur OK.

Propriétés des projets pour la création de packages de projets mobiles

Vous pouvez spécifier les propriétés de projet, les certificats numériques, le contenu du package et les extensions natives d'une cible de génération.

Lors du développement d'applications mobiles pour plusieurs plateformes et périphériques, vous pouvez spécifier différentes configurations de génération pour chaque cible de lancement. Vous pouvez créer une cible de génération personnalisée avec le contenu du package et les propriétés de projet requis. Pour plus d'informations, voir « Prise en charge de plusieurs cibles mobiles » à la page 229.

1 Dans l'Explorateur de packages, sélectionnez un projet.

2 Sélectionnez Projet > Propriétés dans le menu principal ou sélectionnez Propriétés dans le menu contextuel.

3 Dans la boîte de dialogue des propriétés du projet, sélectionnez Paquet de génération Flex et sélectionnez la plateforme cible dont les paramètres doivent être affichés ou modifiés.

Cibles : cliquez sur Cibles afin de sélectionner une cible pour votre plateforme d'application.

Pour la plateforme Apple iOS, la cible peut être périphérique, bureau ou simulateur. Pour la plateforme Google Android, la cible peut être bureau ou périphérique.

Cibles personnalisées : vous pouvez également créer une plateforme cible personnalisée à l'aide d'une cible existante d'une plateforme. Pour ce faire, cliquez sur Créer une cible personnalisée et spécifiez les détails de la nouvelle cible. Cliquez ensuite sur Créer. La cible créée apparaît dans la liste des cibles disponibles.

Signature numérique : sélectionnez le certificat numérique à utiliser pour créer un package de l'application mobile et la signer.

Si vous disposez d'une plateforme Apple iOS, utilisez le certificat de développeur correct émis par Apple pour signer votre application. Pour ce faire, convertissez votre certificat de développeur iOS au format P12 et sélectionnez un fichier de configuration. Pour plus d'informations, voir « Processus de développement Apple iOS à l'aide de Flash Builder » à la page 235.

Pour la plateforme Google Android, vous pouvez sélectionner un certificat numérique ou générer un certificat auto-signé. Pour générer un certificat auto-signé, cliquez sur Créer et saisissez les données dans les champs obligatoires.

Contenu du package : désélectionnez les fichiers à exclure de la version finale de votre application.

Habilitations : ce paramètre est disponible uniquement pour la plateforme Apple iOS.

Pour ajouter des habilitations à votre application, modifiez le fichier XML descripteur de l'application. Les habilitations permettent aux applications d'accéder à des ressources et fonctions spéciales d'iOS. Pour plus d'informations, voir cet article de blog.

Autorisations : ce paramètre est disponible uniquement pour la plateforme Google Android.

Pour modifier les autorisations liées aux applications mobiles pour Android, modifiez le fichier descripteur de l'application (*nom_app*-app.xml). Pour plus d'informations, voir « Choix des autorisations d'une application mobile » à la page 227.

Extensions natives : sélectionnez les extensions natives ActionScript (ANE) à utiliser dans votre application. Pour plus d'informations sur l'utilisation des ANE, voir « Utilisation d'extensions natives » à la page 222.

Sélectionnez Masquer les symboles de bibliothèque ANE pour résoudre les éventuels conflits de symboles. Ces conflits de symboles peuvent être dus à l'utilisation de plusieurs extensions natives dans votre application iOS. Pour plus d'informations, voir « Masquage des symboles de bibliothèque ANE » à la page 225.

4 Cliquez sur OK.

Signature numérique des applications Adobe AIR

Lors de l'exportation de la version validée de votre application, vous devez ajouter une signature numérique à votre application Adobe AIR. Les possibilités suivantes s'offrent à vous :

- Vous pouvez signer l'application à l'aide d'un certificat de signature de code commercial.
- Vous pouvez créer et utiliser un certificat numérique autosigné.
- Vous pouvez assembler l'application et la signer plus tard.

Les certificats numériques émis par les autorités de certification, telles que VeriSign, Thawte, GlobalSign et ChosenSecurity garantissent aux utilisateurs votre identité en tant qu'éditeur. Les certificats numériques vérifient en outre que le fichier d'installation n'a pas été modifié depuis que vous l'avez signé. Les certificats numériques autosignés sont utilisés dans le même but, mais se distinguent par le fait qu'ils ne présentent aucune validation par un tiers.

Vous pouvez également créer un package de l'application Adobe AIR sans signature numérique en créant un fichier AIR intermédiaire (.air). La validité de ce type de fichier est limitée. Il ne peut en effet pas être installé. Il est utilisé dans les procédures de test (effectuées par les développeurs) et peut être lancé avec l'outil de ligne de commande AIR ADT. Adobe AIR fournit cette fonctionnalité pour les environnements de développement au sein desquels la signature est prise en charge par un développeur ou une équipe spécifique. Cette manière de procéder augmente le degré de sécurité de gestion des certificats numériques.

Pour plus d'informations sur la signature d'applications, voir Signature d'applications AIR dans la documentation d'Adobe AIR.

Signature numérique d'une application Adobe AIR

Vous pouvez ajouter une signature numérique à une application Adobe AIR en sélectionnant le certificat numérique existant ou en créant un certificat autosigné.

1 Sélectionnez Projet > Exporter vers une version validée.

Sélectionnez le projet Adobe AIR à exporter ainsi que le fichier vers lequel vous souhaitez exporter le projet. Cliquez sur Suivant.

2 Sélectionnez l'option Exporter et signer un fichier AIR avec un certificat numérique.

3 Si vous disposez déjà d'un certificat numérique, cliquez sur Parcourir pour y accéder, puis sélectionnez-le.

4 Pour créer un certificat numérique autosigné, cliquez sur Créer.

5 Spécifiez les informations requises et cliquez sur OK.

6 (Facultatif) Cliquez sur Suivant. Sélectionnez les fichiers de sortie qui devront être inclus dans le fichier AIRI exporté.

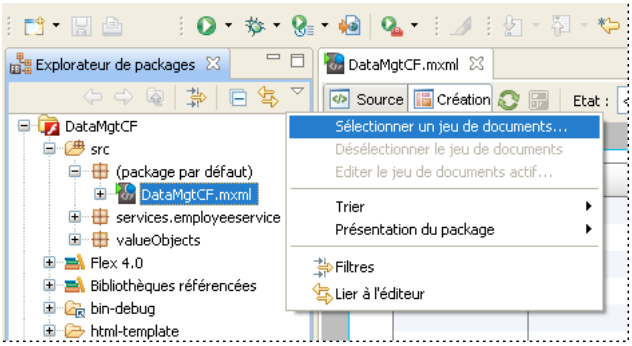
Par défaut, tous les fichiers sont inclus.

7 Cliquez sur Terminer pour créer le fichier AIR.

Création de jeux de documents

Si votre espace de travail contient de nombreux projets, vous pouvez créer un jeu de documents afin de regrouper un certain nombre de projets. Vous pouvez alors afficher des jeux de documents différents dans les vues Explorateur de packages et Tâches de même que limiter la portée de vos recherches à des jeux de documents (et ainsi éviter les recherches portant sur l'ensemble de l'espace de travail).

1 Dans la vue Explorateur de packages, ouvrez le menu de la barre d'outils et choisissez Sélectionner un jeu de documents.



2 Sélectionnez Nouveau.

Flash Builder propose deux types de jeu : les points d'arrêt (utilisés lors du débogage) et les ressources.

3 Sélectionnez le type Ressource, puis cliquez sur Suivant.

4 Entrez le nom du jeu de documents, puis choisissez les projets de l'espace de travail que vous souhaitez inclure dans le jeu.

5 Cliquez sur Terminer.

Le jeu de documents est aussitôt appliqué à la vue Explorateur de packages et seuls les projets et les ressources qui figurent dans le jeu s'affichent.

Pour afficher tous les projets dans l'espace de travail, dans la vue Explorateur de packages, ouvrez le menu de la barre d'outils et choisissez Désélectionner le jeu de documents.

Chapitre 5 : Débogage d'outils dans Flash Builder

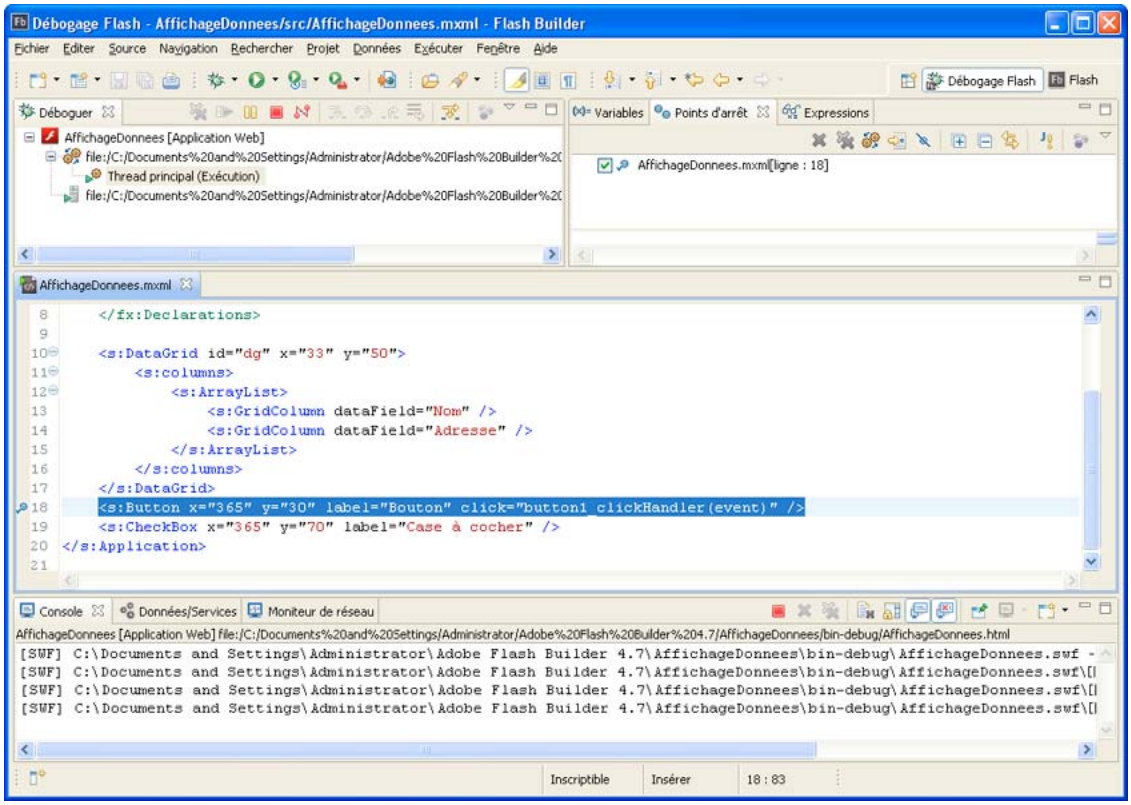
Une fois les projets générés en applications, vous pouvez les déboguer dans Flash Builder.

Le débogage est similaire à l'exécution des applications. Toutefois, lorsque vous déboguez, vous déterminez le moment où l'application s'arrête à des points spécifiques du code. Vous pouvez également définir si vous souhaitez contrôler les variables importantes et vous pouvez tester des corrections sur votre code.

Perspective Débogage Flash

La perspective Débogage Flash fournit un jeu d'outils de débogage très complet qui vous permet :

- de définir et de gérer des points d'arrêt ;
- de déterminer comment interrompre, reprendre ou quitter l'application ;
- d'avancer d'un pas avec et sans entrée dans le code ;
- de contrôler des variables ;
- d'évaluer des expressions.



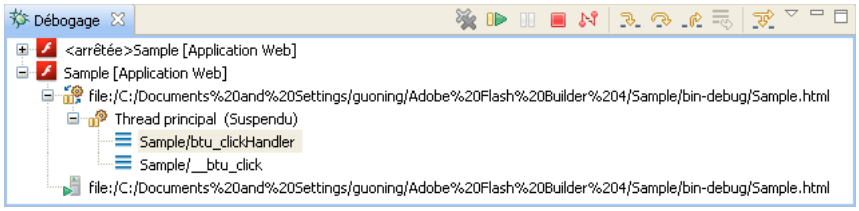
Des points d'arrêt peuvent être ajoutés à des lignes de code exécutables dans l'éditeur de code. La perspective Débogage apparaît automatiquement lorsque le premier point d'arrêt est atteint. A ce stade, la perspective Débogage Flash est activée. Elle vous permet d'analyser l'état de l'application et de la gérer en faisant appel aux outils de débogage. Pour plus d'informations, voir « Lancement d'une session de débogage » à la page 125.

Vous pouvez également ouvrir la perspective Débogage manuellement en la sélectionnant dans la barre des perspectives. La barre des perspectives se trouve sur le côté droit de la barre d'outils principale du workbench. La perspective Débogage contient les vues Débogage, Points d'arrêt, Console, Variables et Expressions.

Vue Débogage

La vue Débogage est le centre de contrôle de la perspective Débogage Flash. Elle permet de contrôler l'exécution d'une application, d'interrompre, de reprendre ou de terminer l'application ainsi que d'avancer d'un pas dans le code avec ou sans entrée.

La vue Débogage (dans d'autres débogueurs, cette option porte le nom *pile des appels*) affiche l'élément de pile du thread suspendu de l'application que vous déboguez.

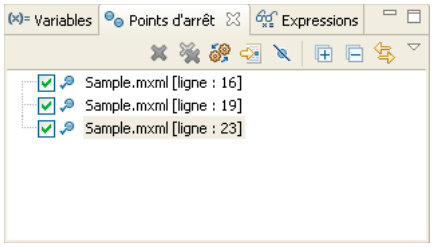


Les applications créées avec Flex ne comportent qu'un seul thread (ce qui n'est pas le cas de Java, par exemple). Vous ne pouvez déboguer qu'une seule application à la fois. Au cours du débogage d'une application, vous ne pouvez donc afficher les processus et la vue Débogage que d'un seul thread d'exécution.

La vue Débogage affiche la liste des fonctions appelées jusqu'à ce point, dans l'ordre de leur appel. Par exemple, la première fonction appelée apparaît en bas de la liste. Vous pouvez cliquer deux fois sur une fonction pour la déplacer dans le script ; Flash Builder met à jour les informations de la vue Variables conformément au nouvel emplacement de la fonction dans le script.

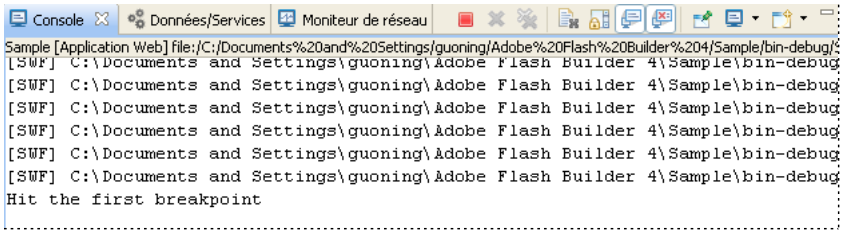
Vue Points d'arrêt

La vue Points d'arrêt répertorie tous les points d'arrêt définis dans le projet. Cliquez deux fois sur un point d'arrêt pour afficher son emplacement dans l'éditeur. Vous pouvez également désactiver, ignorer et supprimer des points d'arrêt.



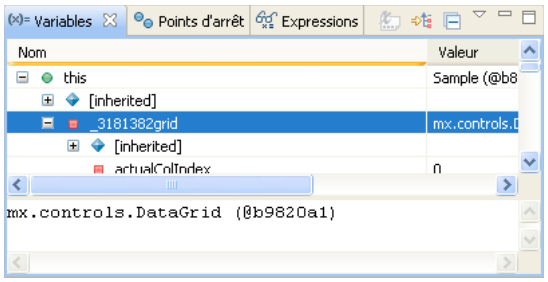
Vue Console

La vue Console affiche le résultat des instructions trace placées dans le code ActionScript ainsi que les informations provenant du débogueur lui-même (statut, avertissements et erreurs).

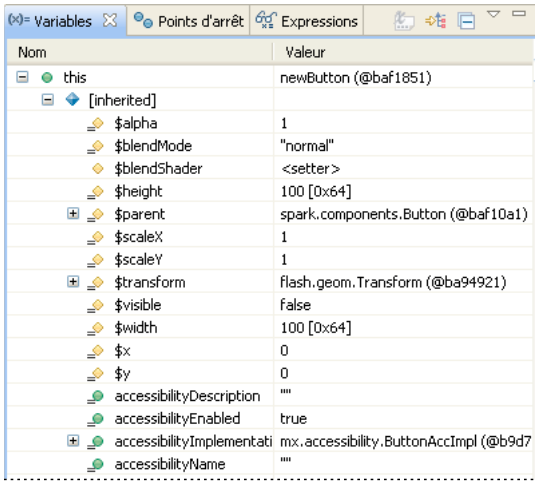


Vue Variables

La vue Variables affiche des informations sur les variables dans un cadre de pile sélectionné. Vous pouvez sélectionner les variables à surveiller (dans la vue Expressions) et modifier leurs valeurs dans la session de débogage. Au cours de la session de débogage, vous pouvez visualiser les modifications apportées au fichier SWF en cours d'exécution et essayer des correctifs pour le problème que vous devez résoudre.



La vue Variables fait appel à des icônes et des superpositions pour donner des repères visuels sur le type de variable.



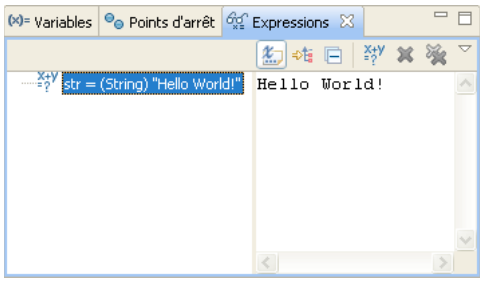
Vue Variables

Les variables complexes peuvent être développées pour en afficher les membres. La vue Variables permet de contrôler les variables en les ajoutant à la vue Expressions. Elle permet également de modifier la valeur des variables au cours de la session de débogage. Vous pouvez également y définir des points de contrôle (voir « Utilisation de points de contrôle » à la page 131).

Tous les membres de super-classes sont groupés dans un nœud d'arborescence distinct. Par défaut, seuls les membres de la classe actuelle sont affichés. Ce regroupement permet de limiter le nombre de variables affichées dans la vue.

Vue Expressions

La vue Expressions permet de surveiller un ensemble de variables critiques. Vous pouvez choisir les variables que vous jugez critiques dans la vue Variables et les ajouter à la vue Expressions afin de les surveiller. Vous pouvez également ajouter et évaluer des expressions de contrôle. Il s'agit d'expressions de code évaluées à chaque interruption du débogage.



Voir aussi

« Exécution et débogage des applications » à la page 105

« Débogage de votre application » à la page 124

Débogage de votre application

L'exécution et le débogage font appel à une configuration pour contrôler la manière dont les applications sont lancées. Le débogage d'une application consiste à exécuter la version déboguée du fichier d'application. Voir « Version déboguée de l'application » à la page 109.

Vous pouvez modifier le fichier de l'application principale par défaut spécifié dans la configuration de lancement de même que le chemin de lancement par défaut, afin que l'application soit exécutée ou déboguée dans la version autonome de Flash Player plutôt que dans un navigateur Web. Voir « Exécution du fichier SWF de l'application dans la version autonome de Flash Player » à la page 111.

Lors du débogage d'applications mobiles, utilisez la configuration de lancement pour spécifier s'il faut lancer l'application sur l'ordinateur ou sur un périphérique connecté à votre ordinateur.

Voir aussi

« Perspective Débogage Flash » à la page 121

« Exécution et débogage des applications » à la page 105

« Erreurs d'environnement Eclipse dans le fichier journal » à la page 88

Lancement d'une session de débogage

Pour lancer une session de débogage, exécutez la configuration de lancement de l'application dans la perspective Débogage Flash.

Débogage d'une application

1 Dans l'Explorateur de packages, sélectionnez le projet que vous souhaitez déboguer.

2 Cliquez sur le bouton situé dans la barre d'outils principale.

Remarque : *le bouton Déboguer comporte deux éléments : le bouton d'action principal et une liste déroulante. La liste déroulante présente les fichiers d'application du projet qui peuvent être exécutés ou débogués. Cliquez sur le bouton d'action principal pour exécuter le fichier d'application par défaut du projet. Vous pouvez également cliquer sur la liste déroulante et sélectionner l'un des fichiers d'application du projet que vous voulez déboguer. Vous pouvez en outre ouvrir la boîte de dialogue des configurations de lancement et créer ou éditer une configuration en cliquant sur la commande Déboguer.*

Vous pouvez également sélectionner Exécuter > Déboguer.

Si le projet n'a pas encore été généré, Adobe® Flash® Builder™ le génère et l'exécute en mode de débogage.

3 L'application est affichée dans le navigateur Web par défaut ou dans la version autonome de Flash Player. Vous pouvez ensuite faire appel au débogueur de Flash Builder pour interagir avec elle.

4 La perspective Débogage Flash est activée dans le workbench à chaque point d'arrêt rencontré.

Lancement d'une session de débogage dans la configuration plug-in

L'action de la commande Déboguer de la version plug-in de Flash Builder est légèrement différente. Elle n'exécute pas le projet sélectionné, mais débogue la dernière configuration à avoir été lancée. Vous pouvez également sélectionner une configuration parmi la liste de celles à avoir été lancées en dernier.

Débogage d'une application Adobe AIR

Flash Builder prend entièrement en charge le débogage des applications Adobe AIR.

1 Dans Flash Builder, ouvrez un fichier source pour l'application (un fichier MXML, par exemple).

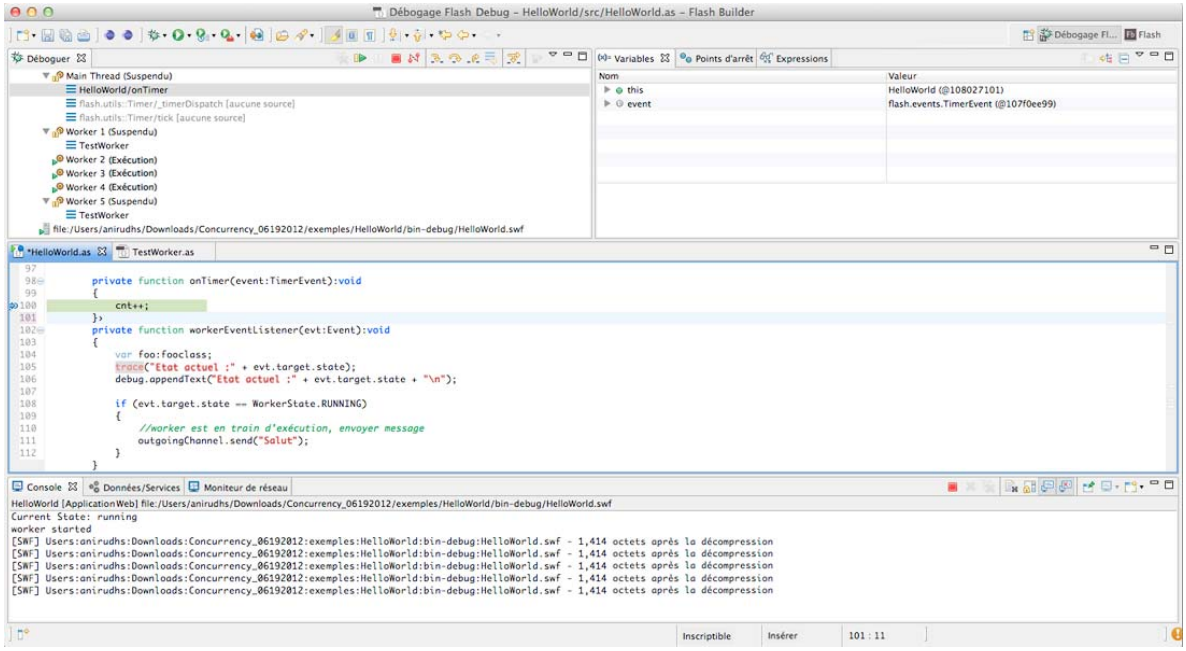
2 Cliquez sur le bouton situé dans la barre d'outils principale.

L'application est lancée. Elle est exécutée dans l'application ADL (AIR Debugger Launcher, application de débogage du lanceur AIR). L'application de débogage de Flash Builder capture les éventuels points d'interruption ou erreurs d'exécution et vous permet de déboguer l'application comme toute autre application développée avec Flex.

Vous pouvez également déboguer une application à partir de la ligne de commande en utilisant l'outil de ligne de commande de l'application de débogage du lanceur AIR. Pour plus d'informations, voir *Utilisation de l'application de débogage du lanceur AIR (ADL)* dans la documentation AIR.

Débogage d'une application à l'aide d'opérateurs ActionScript

Flash Builder prend en charge le débogage d'applications à l'aide de plusieurs opérateurs ActionScript. Le thread principal, ainsi que l'ensemble des threads d'opérateur pris en charge, sont affichés dans la vue Débogage de la façon suivante :



Vous pouvez définir des points d'arrêt dans vos fichiers d'opérateur. Lors du débogage, Flash Builder vous permet d'exécuter les instructions pas à pas pour un opérateur spécifique.

Vous pouvez interrompre ou reprendre un opérateur spécifique en le sélectionnant et en utilisant les boutons appropriés dans la barre d'outils de la vue Débogage.

Important : *vous ne pouvez pas mettre fin à un thread d'opérateur spécifique. Si vous choisissez d'en arrêter un, la session de débogage est arrêtée et l'application en cours d'exécution sur votre ordinateur ou périphérique est fermée.*

Chaque thread d'opérateur ActionScript dispose de son état d'exécution, de sa pile d'appels et de ses variables propres. Sélectionnez la pile d'appels pour afficher les variables de cet opérateur. La vue Expressions affiche seulement les expressions pertinentes pour la pile d'appels sélectionnée.

Toutes les autres fonctionnalités de débogage fonctionnent également selon la pile d'appels sélectionnée. Par exemple, lorsque vous évaluez une expression, la première partie de l'expression est résolue à l'aide de la pile d'appels sélectionnée. Une fois cette opération effectuée, les actions de débogage suivantes concernant l'expression, telles que l'accès membre, les appels de fonction et autres actions similaires, se basent sur l'ID d'opérateur défini dans la première partie de l'expression.

Ajout et suppression de points d'arrêt

Les points d'arrêt vous permettent d'interrompre l'exécution de votre application. En procédant ainsi, vous pouvez inspecter votre code et utiliser les outils de débogage de Flash Builder pour explorer les options de résolution des erreurs. Les points d'arrêt sont ajoutés dans l'éditeur de code et gérés dans la vue Points d'arrêt lorsque vous déboguez les applications.

Ils sont placés au niveau des lignes de code exécutables. Le débogueur ne s'arrête qu'aux points d'arrêt situés sur les lignes présentant les éléments suivants :

- Balises MXML contenant un gestionnaire d'événement ActionScript, par exemple `<mx:Button click="`*dofunction*`()" ...>`
- Lignes ActionScript, telles que celles contenues dans une balise `<mx:Script>` ou dans un fichier ActionScript
- Toute ligne de code exécutable dans un fichier ActionScript

Vous pouvez définir des points d'arrêt en cours de rédaction du code ou pendant le débogage.

Positionnement d'un point d'arrêt dans l'éditeur de code

1 Ouvrez un fichier de projet contenant du code ActionScript.

2 Localisez la ligne de code sur laquelle vous souhaitez placer un point d'arrêt et cliquez deux fois sur la barre de repère pour en ajouter un.

La barre de repère longe le côté gauche de l'éditeur de code.

Un point d'arrêt est ajouté à la barre de repère ainsi qu'à la liste de la vue Points d'arrêt de la perspective Débogage Flash.

A chaque fois que le débogueur rencontre un point d'arrêt, l'application est interrompue et la perspective Débogage Flash est affichée. La ligne de code est signalée par un point d'arrêt et mise en surbrillance dans l'éditeur de code. Vous pouvez ensuite utiliser les commandes de débogage de la barre d'outils de la vue Points d'arrêt pour interagir avec le code. Voir « Vue Points d'arrêt » à la page 122.

Suppression d'un point d'arrêt dans l'éditeur de code

❖ Dans la barre de repère, cliquez deux fois sur un point d'arrêt.

Le point d'arrêt est supprimé de la barre de repère ainsi que de la vue Points d'arrêt de la perspective Débogage Flash.

Suppression de points d'arrêt de la vue Points d'arrêt

Vous pouvez supprimer un seul, plusieurs ou tous les points d'arrêt de la vue en utilisant les commandes de la barre d'outils.

- Sélectionnez un ou plusieurs points d'arrêt dans la liste, puis cliquez sur Supprimer les points d'arrêt sélectionnés.
- Pour supprimer tous les points d'arrêt en une seule action, cliquez sur Supprimer tous les points d'arrêt.

Vous pouvez également désactiver les points d'arrêt et les réactiver ultérieurement.

Définition de points d'arrêt conditionnels

Vous pouvez spécifier les conditions régissant l'interruption de l'exécution du débogueur par les points d'arrêt. La définition d'un point d'arrêt conditionnel consiste à spécifier une expression ActionScript qui sera évaluée au cours de la session de débogage. Les points d'arrêt conditionnels interrompent l'exécution lorsque l'une des conditions suivantes est remplie :

- l'expression prend la valeur « true » ;
- la valeur de l'expression change ;
- le nombre d'instances spécifié a été atteint.

Définition d'un point d'arrêt conditionnel

1 Dans le menu contextuel d'un point d'arrêt, cliquez sur Propriétés du point d'arrêt.

2 Dans la boîte de dialogue qui s'ouvre, spécifiez les paramètres suivants :

- Activé

  Activez ou désactivez le point d'arrêt.

- Nombre d'occurrences

  Sélectionnez l'option Nombre d'occurrences afin de définir un compteur pour le point d'arrêt. Saisissez un nombre dans le champ Nombre d'occurrences.

  Si vous activez les deux options, Nombre d'occurrences et Activer la condition, le nombre d'occurrences représente le nombre de fois que la condition spécifiée est remplie (elle prend la valeur « true » ou elle change).

  Si vous n'activez que l'option Nombre d'occurrences, le nombre d'occurrences représente le nombre de fois que le point d'arrêt est atteint.

- Activer la condition

  Sélectionnez l'option Activer la condition et saisissez une expression ActionScript à évaluer. Pour plus d'informations sur les types d'expressions dont l'évaluation est prise en charge, voir « Exemples d'expressions » à la page 129.

  Remarque : *Flash Builder vérifie la syntaxe de l'expression et vous en signale les erreurs. La présence d'un opérateur d'affectation dans l'expression déclenche l'affichage d'un avertissement.*

- Suspendre lorsque

  Spécifiez à quel moment l'exécution doit être interrompue : lorsque la condition a la valeur « true » ou lorsque la valeur de la condition change.

Gestion des variables dans la vue Variables

Modification de la valeur d'une variable

1 Sélectionnez la variable que vous souhaitez modifier.

2 Cliquez avec le bouton droit de la souris (Windows) ou en maintenant la touche Ctrl enfoncée (Mac OS) pour afficher le menu contextuel et sélectionnez Modifier la valeur.

3 Saisissez la nouvelle valeur et cliquez sur OK.

La variable contient la nouvelle valeur.

Les variables modifiées sont affichées en rouge.

Recherche de variables

❖ Pour localiser une variable ou un membre de variable dans la vue Variables, sélectionnez la vue, puis commencez à saisir le nom de la variable recherchée. Vous pouvez utiliser le caractère générique (*) pour rechercher les mots figurant dans le nom d'une variable (« *couleur », par exemple).

Utilisation de la vue Expressions

En cours de débogage, vous pouvez analyser et modifier la valeur des variables contrôlées. Vous pouvez également ajouter des expressions de contrôle. Il s'agit d'expressions de code évaluées à chaque interruption du débogage. Les expressions de contrôle sont utiles pour contrôler les variables susceptibles de dépasser l'étendue et de ne pas être visibles dans la vue lorsque vous avancez d'un pas avec entrée dans une fonction différente.

Vous pouvez aussi placer le pointeur sur une expression ou une variable dans l'éditeur source afin d'afficher la valeur de cette expression ou de cette variable dans une infobulle. Vous pouvez ajouter l'expression à la vue correspondante en cliquant avec le bouton droit de la souris, puis en sélectionnant la commande de contrôle dans le menu contextuel.

Exemples d'expressions

Le débogueur de Flash Builder prend en charge une large gamme d'expressions simples et complexes. Le tableau suivant répertorie des exemples d'expressions pouvant être évaluées en cours de session de débogage. Cette liste d'expressions prises en charge n'est pas exhaustive et est fournie uniquement à titre indicatif.

Exemples d'expressions prises en charge

Expression	Description
myString.length	Retourne la longueur d'une chaîne.
myString.indexOf('@')	Effectue le suivi de l'index du caractère « @ ».
"constant string".charAt(0)	Effectue le suivi du caractère à un emplacement spécifique de la chaîne. Les constantes de chaîne sont prises en charge.
employees.employee.@nom	« employees » est une variable XML. Ce type d'expression est utile pour déboguer les applications E4X.
x == null	Mots réservés représentant des valeurs dans les expressions.
user1 === user2	La plupart des opérateurs ActionScript sont pris en charge.
MyClass.myStaticFunc()	Fonctions interprétées comme une classe.
this.myMemberFunc()	Fonctions interprétées en utilisant le mot-clé `this`.
String.fromCharCode(33)	String est en fait une fonction, et non une classe, dont String.fromCharCode est un membre dynamique.
myStaticFunc()	Ne peut être évalué que si myStaticFunc est visible à partir de la chaîne d'étendue actuelle.
myMemberFunc()	Ne peut être évalué que si myMemberFunc est visible à partir de la chaîne d'étendue actuelle.
Math.max(1,2,3)	Les fonctions mathématiques sont prises en charge.
mystring.search(/myregex/i)	Les expressions régulières sont prises en charge.
["my", "literal", "array"]	Création de tableaux.
new MyClass()	Instanciation de classes.
"string" + 3	Traitement correct d'une chaîne plus un nombre entier.
x >>> 2	Les opérations de décalage logique sont prises en charge.
3.5 + 2	Exécution correcte des opérations arithmétiques.

Limitations de l'évaluation des expressions

Il existe quelques limitations à l'évaluation des expressions.

- Les espaces de noms ne sont pas pris en charge.
- Les objets insérés ne sont pas pris en charge.
- Le mot-clé `super` n'est pas pris en charge.
- Les noms de classe complets ne sont pas pris en charge.

  Par exemple, vous ne pouvez pas évaluer mx.controls.Button.

  Vous pouvez faire référence au nom de classe non qualifié. Par exemple, vous pouvez spécifier que Button fait référence à mx.controls.Button.

  En présence d'un nom de classe ambigu (deux classes portant le même nom dans deux packages différents), il vous est impossible de contrôler quelle classe est évaluée. Vous pouvez cependant spécifier :

  ```
  getDefinitionByName("mx.controls.Button")
  ```

- La plupart des expressions E4X peuvent être évaluées, mais les expressions de filtre E4X ne sont pas prises en charge.

Par exemple, vous ne pouvez pas évaluer `myxml.(@id=='3')`.

- Vous ne pouvez pas appeler des fonctions définies en tant que variables.

Utilisation de points de contrôle

Au cours du débogage d'une application, vous pouvez définir des points de contrôle pour des instances spécifiques de variables. Ils auront pour tâche d'interrompre l'exécution à tout changement de valeur de la variable contrôlée. Les points de contrôle étant définis pour l'instance spécifique d'une variable, ils ne peuvent pas être placés dans l'éditeur de code. Ils sont donc définis à partir de la vue Variables au cours de la session de débogage.

Pour la définition des points de contrôle, veillez aux points suivants :

- A la fin d'une session de débogage, tous les points de contrôle sont supprimés.
- Vous ne pouvez pas définir des points de contrôle sur les getter, mais vous pouvez en définir sur les champs des getter.

  Par exemple, vous ne pouvez pas définir un point de contrôle sur `width`, mais vous pouvez en définir un sur `_width`.

- Vous ne pouvez pas définir des points de contrôle sur des variables locales, mais vous pouvez en définir sur les membres des variables locales, comme dans le fragment de code suivant :

```
public class MyClass
{
   // These are fields of a class, so you can set a watchpoint on
   // 'memberInt', and on 'memberButton', and on 'memberButton._width':
   private var memberInt:int = 0;
   private var memberButton:Button = new Button();

   public function myFunction():void {
      // You CANNOT set a watchpoint on 'i', because it is local:
      var i:int = 0;

      // You CANNOT set a watchpoint on 'someButton', because it is local;
      // but you CAN set a watchpoint on 'someButton._width':
      var someButton:Button = new Button();

...
}
```

- L'exécution s'interrompt pour un point de contrôle lorsque la valeur d'origine d'une instance d'objet change.

  Ce comportement diffère de l'utilisation d'une expression dans un point d'arrêt conditionnel pour arrêter l'exécution lorsque la valeur d'une variable change.

Définition de points de contrôle

❖ Deux méthodes permettent de définir un point de contrôle au cours d'une session de débogage :

- Dans la vue Variables, ouvrez le menu contextuel d'une variable et sélectionnez Ajouter/Supprimer un point de contrôle.
- Dans le menu Exécuter de Flash Builder, sélectionnez Ajouter un point de contrôle.

  Dans la boîte de dialogue d'ajout d'un point de contrôle, sélectionnez la variable que vous souhaitez contrôler.

La vue Variables affiche une icône représentant un crayon en regard de la variable pour laquelle un point de contrôle a été défini.

Remarque : *si vous tentez de définir un point de contrôle pour un getter, une boîte de dialogue s'ouvre. Elle suggère une variable valide pour le point d'arrêt. Si vous supprimez la variable suggérée, la boîte de dialogue répertorie toutes les variables valides de l'objet.*

Utilisation de la commande Exécuter jusqu'à la ligne

La commande Exécuter jusqu'à la ligne permet de quitter une boucle au cours d'une session de débogage.

Au cours du débogage, vous constaterez parfois que votre code exécute une boucle répétitive. Pour quitter cette boucle, faites appel à la commande Exécuter jusqu'à la ligne située dans le menu Exécuter.

Chapitre 6 : Profilage d'outils dans Flash Builder

Adobe Flash Builder Premium contient une perspective supplémentaire, la perspective Profil Flash, qui vous aide à identifier les problèmes de performance et les fuites de mémoire de vos applications.

Présentation du profilage et fonctionnement du profileur

Le profileur permet d'identifier les problèmes de performance et les fuites de mémoire des applications. Vous le lancez dans Adobe Flash Builder, et comme vous interagissez avec votre application, le profileur enregistre les données concernant l'état de l'application. Les données enregistrées incluent le nombre d'objets, la taille de ces objets, le nombre d'appels de méthodes et le temps passé dans ces appels de méthodes.

Le profilage d'une application peut vous aider à comprendre les éléments suivants :

Fréquence des appels : dans certains cas, les méthodes faisant appel à un grand nombre de ressources sont appelées plusieurs fois, alors qu'un seul appel est nécessaire. En identifiant les méthodes les plus fréquemment appelées, vous pouvez concentrer le temps que vous passez à ajuster les performances sur une zone plus réduite de l'application, où l'impact sur les performances peut être plus important.

Durée de la méthode : le profileur peut vous indiquer le temps passé dans une méthode particulière ou, si la méthode est appelée plusieurs fois, le laps de temps moyen passé dans cette méthode au cours d'une session de profilage. Si vous trouvez que certaines méthodes entraînent des goulets d'étranglement des performances, vous pouvez tenter de les optimiser.

Piles d'appels : en suivant la pile d'appels d'une méthode, vous pouvez voir l'intégralité du chemin que poursuit l'application lorsqu'elle appelle des méthodes successives. Vous constatez parfois alors que des méthodes sont appelées alors que cela n'est pas nécessaire.

Nombre d'occurrences (allocation d'objet) : si vous constatez que le même objet est créé plusieurs fois alors que seul un nombre spécifique d'occurrences est requis, vous pouvez implémenter un motif Singleton si vous n'avez réellement besoin que d'un seul de ces objets. Vous pouvez également appliquer d'autres techniques permettant de réduire l'allocation excessive d'objets. Si le nombre d'objets est important, mais nécessaire, vous pouvez optimiser l'objet même afin de réduire sa ressource Agrégat et l'utilisation qu'il fait de la mémoire.

Taille de l'objet : si vous remarquez que certains objets sont de taille disproportionnée, vous pouvez tenter de les optimiser pour réduire leur encombrement mémoire. Cette opération s'avère particulièrement utile si vous optimisez des objets créés de nombreuses fois dans l'application.

Récupération de place : lorsque vous comparez des instantanés de profilage, vous pouvez constater que certains objets dont l'application n'a plus besoin sont toujours présents et sont toujours stockés en mémoire. Pour prévenir ces pertes de mémoire, vous pouvez ajouter une logique qui supprime toute référence restante à ces objets.

Le profilage est un processus d'itération et constitue une partie de chaque étape du développement de l'application. Pour identifier rapidement les zones d'erreur et tirer le meilleur profit du profilage, il est conseillé de profiler l'application aussi tôt et aussi souvent que possible dans le cycle de développement.

Fonctionnement du profileur

Le profileur est un agent qui communique avec l'application s'exécutant dans Flash Player. Il se connecte à l'application par l'intermédiaire d'une connexion à un socket local. Vous devrez donc éventuellement désactiver votre logiciel antivirus si ce dernier entrave la communication avec les sockets.

Lorsque le profileur est en cours d'exécution, il prend un instantané des données à intervalles courts et enregistre les actions d'Adobe Flash Player à ces moments. C'est ce qu'on appelle l'*échantillonnage*. Le profileur enregistre par exemple une méthode exécutée par l'application au moment de la prise de l'instantané. Si au moment de l'instantané suivant, l'application exécute toujours la même méthode, le profileur continue d'enregistrer l'heure. Lorsque le profileur effectue l'instantané suivant et que l'application est passée à l'opération suivante, le profileur peut consigner le temps qu'il a fallu à la méthode pour s'exécuter.

L'échantillonnage vous permet de profiler sans ralentir notablement l'application. L'intervalle écoulé s'appelle la *fréquence d'échantillonnage*, qui survient environ toutes les millisecondes pendant la période de profilage. Cela signifie que les opérations ne sont pas toutes enregistrées et que les instantanés ne sont pas tous précis à une fraction de milliseconde près. Cela vous permet toutefois d'avoir une idée plus précise des opérations qui prennent plus de temps que les autres.

En analysant les données d'échantillonnage, le profileur peut afficher toutes les opérations effectuées dans l'application et enregistrer leur délai d'exécution. Il consigne également l'utilisation de la mémoire et les traces de pile et affiche les données dans une série de vues, ou panneaux. Les appels de méthode sont classés par durée d'exécution et nombre d'appels, ainsi que par nombre d'objets créés dans la méthode.

Le profileur calcule également les valeurs cumulées des données. Par exemple, si vous consultez des statistiques liées aux méthodes, les données cumulées comprennent le temps et la mémoire alloués au cours de cette méthode, ainsi que le temps et la mémoire alloués au cours de toutes les méthodes appelées à partir de cette méthode. Vous pouvez effectuer une analyse plus poussée des appels de méthode suivants, jusqu'à trouver la source des problèmes de performances.

Types de profilage

Avant de faire appel au profileur, décidez du type de profilage que vous allez effectuer : profilage des performances ou profilage de la mémoire.

Le *profilage des performances* est le processus de recherche de méthodes de l'application qui s'exécutent lentement et peuvent être améliorées. Une fois identifiées, ces zones à problèmes peuvent être optimisées pour accélérer l'exécution de l'application et pour que celle-ci réponde plus rapidement aux interactions de la part de l'utilisateur. Lors du profilage des performances, vous cherchez généralement deux choses : une méthode appelée une seule fois mais qui met plus de temps à s'exécuter que des méthodes similaires, ou une méthode qui ne met pas autant de temps à s'exécuter, mais qui est appelée de nombreuses fois. Les données de profilage des performances permettent d'identifier les méthodes que vous optimisez par la suite. Vous trouverez peut-être que le fait de réduire le nombre d'appels à une méthode est plus efficace que la restructuration du code au sein de la méthode.

Le *profilage de la mémoire* est l'examen de la quantité de mémoire qu'utilise chaque objet ou type d'objet de l'application. Les données de profilage de la mémoire s'utilisent de diverses façons : pour voir si des objets sont plus volumineux que nécessaire, pour voir si trop d'objets d'un même type existent et pour identifier des objets qui ne sont pas nettoyés dans la mémoire (fuites de mémoire). Grâce aux données de profilage de la mémoire, vous pouvez tenter de réduire la taille des objets, le nombre des objets créés ou permettre le nettoyage des objets dans la mémoire en supprimant les références à ces objets.

Le profilage de la mémoire peut ralentir les performances de l'application car il fait appel à une plus grande quantité de mémoire que le profilage des performances. Ne faites appel au profilage de la mémoire qu'en cas de nécessité.

Il convient souvent d'effectuer à la fois un profilage des performances et un profilage de la mémoire pour trouver la source des problèmes de performances. Le profilage des performances permet d'identifier les méthodes qui entraînent une allocation de mémoire excessive et un allongement des temps d'exécution. Vous pouvez ensuite recourir au profilage de la mémoire afin d'identifier les fuites de mémoire dans ces méthodes.

Lorsque vous savez quel profilage vous allez effectuer, vous pouvez démarrer le profileur.

Profilage d'API

Le profileur utilise les API ActionScript définies dans le packageflash.sampler.*. Ce package contient la Sample, StackFrame, NewObjectSample et DeleteObjectSample. Vous pouvez faire appel aux méthodes et aux classes contenues dans ce package pour écrire votre propre application de profileur ou pour inclure un sous-ensemble de fonctionnalités de profilage dans vos applications.

Outre les classes du package flash.sampler.*, le profileur utilise les méthodes contenues dans la classe System.

Actions internes de Flash Player

Le profileur enregistre généralement des données sur les méthodes de la classe dont l'exécution était en cours au moment de l'instantané d'échantillonnage. Il arrive toutefois que le profileur enregistre également des actions internes de Flash Player. Ces actions, consignées entre crochets, sont notamment `[keyboardEvent]`, `[mark]` et `[sweep]`.

Par exemple, si `[keyboardEvent]` se trouve dans la liste des méthodes avec une valeur de 100, vous savez que le lecteur a effectué une action interne liée à cet événement au moins 100 fois au cours de la période d'interaction.

Le tableau suivant décrit les actions internes de Flash Player qui apparaissent dans les données de profilage :

Action	Description
`[generate]`	Le compilateur juste à temps (JIT) génère du code machine AS3.
`[mark]`	Flash Player marque les objets actifs pour la récupération de place.
`[newclass]`	Flash Player définit une classe. Cette opération a généralement lieu au démarrage. Le chargement d'une nouvelle classe peut avoir cependant lieu à tout moment.
`[pre-render]`	Flash Player se prépare à rendre les objets (notamment les calculs de géométrie et le parcours de la liste d'affichage qui se produisent avant l'affichage).
`[reap]`	Flash Player récupère les objets de décompte différé des références (DRC, deferred reference counting).
`[render]`	Flash Player rend les objets dans la liste d'affichage (pixel par pixel).
`[sweep]`	Flash Player récupère la mémoire des objets non marqués.
`[verify]`	Le compilateur JIT effectue la vérification du bytecode ActionScript 3.0.
`[`*event_type*`Event]`	Flash Player distribue l'événement spécifié.

Ces informations peuvent vous aider à identifier d'éventuels problèmes de performances. Par exemple, si vous voyez plusieurs entrées pour `[mark]` et `[sweep]`, vous pouvez partir du principe que plusieurs objets sont en cours de création, puis sont marqués pour la récupération de place. En comparant ces nombres dans différents profils de performances, vous pouvez constater par vous-même si les modifications que vous apportez sont efficaces.

Pour consulter les données correspondant à ces actions internes, affichez un profil de performances dans la vue Profil de performance ou un profil de mémoire dans la vue Suivi des allocations. Pour plus d'informations, voir « Vue Profil de performance » à la page 145 et « Vue Suivi des allocations » à la page 142.

Ressources supplémentaires

Le profileur seul n'améliore ni la taille, ni la vitesse, ni les performances perçues d'une application. Après avoir utilisé le profileur pour identifier les méthodes et classes qui posent problèmes, consultez les ressources suivantes dans la documentation Flex, qui vous aideront à améliorer l'application :

- Optimisation d'applications
- Amélioration des performances de démarrage

Perspective Profil Flash et vues du profileur

La perspective Profil affiche plusieurs panneaux (ou vues) présentant les données de profilage sous différentes formes. Lorsque vous interagissez avec l'application, le profileur enregistre les données concernant le statut de l'application, notamment le nombre d'objets, leur taille, le nombre d'appels de la méthode et le temps consacré à ces appels.

Vues du profileur

Le profileur se compose de plusieurs vues (ou panneaux) présentant de différentes manières les données de profilage. Le tableau suivant présente chacune de ces vues :

Vue	Description
Profiler	Affiche les applications actuellement connectées, leur état et tous les instantanés de mémoire et de performances associés. Les sessions de profilage démarrent sans instantanés de performances ou de mémoire.
Données de profilage enregistrées	Affiche une liste d'instantanés enregistrés, classés par application. Vous pouvez charger des données de profilage enregistrées en cliquant deux fois sur l'instantané enregistré dans cette liste. Pour plus d'informations, voir « Enregistrement et chargement des données de profilage » à la page 155.
Objets actifs	Affiche des informations concernant les classes utilisées par l'application en cours. Cette vue montre les classes qui sont instanciées, le nombre créé, combien se trouvent dans le segment mémoire et la quantité de mémoire que les objets actifs occupent. Pour plus d'informations, voir « Affichage des informations dans la vue Objets actifs » à la page 137.
Instantané de la mémoire	Affiche l'état de l'application à un moment donné. Cette vue contraste avec la vue Objets actifs, mise à jour en continu. La vue Instantané de la mémoire affiche le nombre d'objets référencés et utilisés dans l'application et la quantité de mémoire que chaque type d'objets utilisait à ce moment-là. Deux instantanés de la mémoire pris à des moments différents sont comparés en vue de déterminer les fuites de mémoire qui existent entre les deux moments. Vous pouvez afficher la vue Instantané de la mémoire en cliquant sur le bouton Prendre un instantané de la mémoire puis en cliquant deux fois sur l'instantané de mémoire dans la vue Profiler. Pour plus d'informations, voir « Vue Instantané de la mémoire » à la page 138.
Objets vagabonds	Affiche les objets créés entre deux instantanés de mémoire qui existent toujours dans la mémoire ou qui n'ont pas été nettoyés. Vous pouvez cliquer deux fois sur le nom d'une classe dans la table pour ouvrir la vue Références de l'objet. Celle-ci vous permet d'examiner la relation entre les objets sélectionnés et d'autres objets. Vous pouvez afficher la vue Objets vagabonds en sélectionnant deux instantanés de mémoire et en cliquant sur le bouton Objets vagabonds. Pour plus d'informations, voir « Vue Objets vagabonds » à la page 148.

Vue	Description
Suivi des allocations	Affiche des statistiques liées aux méthodes lors de la comparaison de deux instantanés de mémoire. Vous pouvez afficher la vue Suivi des allocations en sélectionnant deux instantanés de mémoire et en cliquant sur le bouton Afficher le suivi des allocations. Pour plus d'informations, voir « Vue Suivi des allocations » à la page 142.
Références de l'objet	Affiche les objets et les objets qui les référencent. Vous pouvez afficher la vue Références de l'objet en cliquant deux fois sur le nom d'une classe dans la vue Instantané de la mémoire ou Objets vagabonds. Pour plus d'informations, voir « Vue Références de l'objet » à la page 140.
Statistiques de l'objet	Affiche des informations concernant l'appelant et l'appelé du groupe d'objets sélectionné. Vous pouvez afficher la vue Statistiques de l'objet en cliquant deux fois sur une entrée de la vue Suivi des allocations. Pour plus d'informations, voir « Vue Statistiques de l'objet » à la page 143.
Profil de performance	Affiche les performances des méthodes de l'application au cours d'une période donnée. Cliquez ensuite sur le nom d'une méthode dans la table pour ouvrir la vue Statistiques de la méthode, qui vous permet de détecter les goulets d'étranglement des performances. Vous pouvez afficher la vue Profil de performance en cliquant deux fois sur l'un des instantanés des performances dans la vue Profiler. Pour plus d'informations, voir « Vue Profil de performance » à la page 145.
Statistiques de la méthode	Affiche des statistiques concernant les performances du groupe de méthodes sélectionné. Vous pouvez afficher la vue Statistiques de la méthode en cliquant deux fois sur une ligne de la vue Profil de performance ou en y sélectionnant une méthode et en cliquant sur le bouton Ouvrir les statistiques de la méthode. Pour plus d'informations, voir « Identification des caractéristiques des performances des méthodes » à la page 147.
Utilisation de la mémoire	Affiche sous forme de graphique les pics d'utilisation de la mémoire et l'utilisation actuelle de la mémoire dans le temps. Pour plus d'informations, voir « Graphique Utilisation de la mémoire » à la page 149.

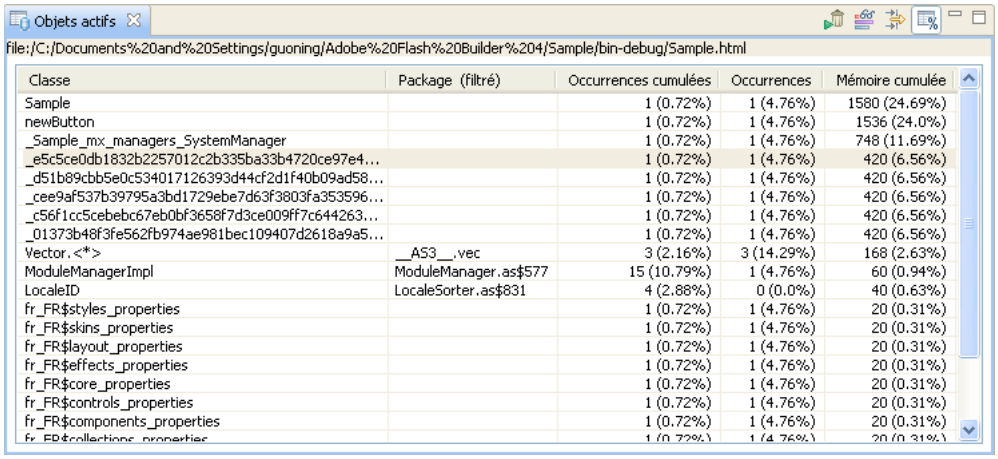
Affichage des informations dans la vue Objets actifs

La vue Objets actifs affiche des informations concernant les classes utilisées par l'application en cours. Cette vue montre les classes qui sont instanciées, le nombre créé, combien se trouvent dans la mémoire et la quantité de mémoire que les objets actifs occupent.

Le profileur met à jour en continu les données dans la vue Objets actifs à mesure que vous profilez l'application. Vous n'avez pas besoin d'actualiser la vue ou de garder le focus dessus pour mettre à jour les données.

Pour pouvoir utiliser la vue Objets actifs, activez le profilage de la mémoire au démarrage du profileur. C'est la valeur par défaut. Si vous fermez la vue Objets actifs et que vous voulez la rouvrir, déployez la liste déroulante dans la vue Profiler et sélectionnez Surveiller les objets actifs.

L'exemple suivant illustre la vue Objets actifs :



Le tableau suivant répertorie les colonnes de la vue Objets actifs :

Colonne	**Description**
Classe	Classes possédant des occurrences dans l'application en cours d'exécution.
Package	Package dans lequel se trouve chaque classe. Si la classe n'est pas dans un package, la valeur de ce champ est le nom du fichier dans lequel se trouve la classe. Le nombre suivant le signe du dollar est un identifiant unique de cette classe. Si le champ Package est vide, la classe se trouve dans le package global ou le package non nommé.
Occurrences cumulées	Nombre total d'occurrences de chaque classe créées depuis le démarrage de l'application.
Occurrences	Nombre d'occurrences de chaque classe actuellement dans la mémoire. Cette valeur est toujours inférieure ou égale à la valeur contenue dans la colonne Occurrences cumulées.
Mémoire cumulée	Quantité totale de mémoire, exprimée en octets, que toutes les occurrences de chaque classe ont utilisée, y compris les classes n'étant plus dans la mémoire.
Mémoire	Quantité totale de mémoire, exprimée en octets, que toutes les occurrences de chaque classe utilisent actuellement. Cette valeur est toujours inférieure ou égale à la valeur contenue dans la colonne Mémoire cumulée.

Vous utilisez généralement les données contenues dans la vue Objets actifs pour voir la quantité de mémoire que les objets utilisent. A mesure que les objets sont nettoyés de la mémoire, l'utilisation du nombre d'occurrences et de la mémoire diminue, mais l'utilisation des occurrences cumulées et de la mémoire cumulée augmente. Cette vue indique également la façon dont la mémoire est utilisée lors de l'exécution de l'application.

Pour plus d'informations sur l'exécution et l'analyse des résultats du nettoyage de la mémoire, voir « Récupération de place » à la page 157.

Vous pouvez limiter les données affichées dans la vue Objets actifs grâce aux filtres du profileur. Pour plus d'informations, voir « Filtres du profileur » à la page 150.

Vue Instantané de la mémoire

La vue Instantané de la mémoire propose des informations concernant l'utilisation des objets et de la mémoire de l'application à un moment donné. A la différence de la vue Objets actifs, les données de la vue Instantané de la mémoire ne sont pas mises à jour en continu.

Pour pouvoir utiliser la vue Instantané de la mémoire, activez le profilage de la mémoire au démarrage du profileur. Il s'agit de la valeur par défaut.

Création et affichage d'un instantané de la mémoire

1 Démarrez une session de profilage.

2 Interagissez avec votre application jusqu'à atteindre un point dans l'état de l'application où vous voulez prendre un instantané de la mémoire.

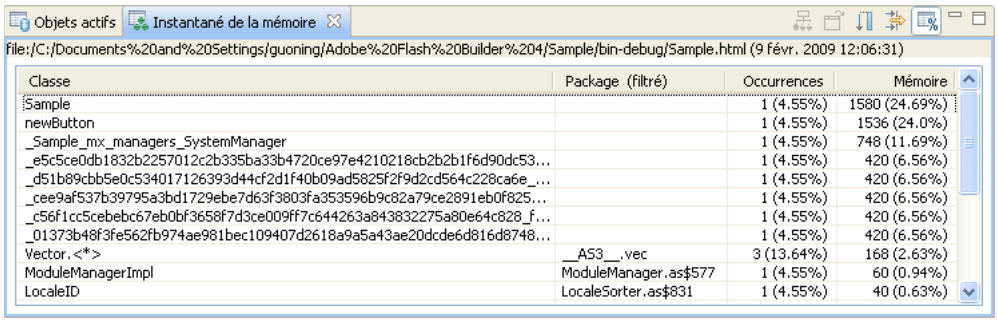
3 Sélectionnez l'application dans la vue Profiler.

4 Cliquez sur le bouton Prendre un instantané de la mémoire.

Le profileur crée un instantané de la mémoire et y appose un cachet de date. Il déclenche également implicitement la récupération de place avant l'enregistrement de l'instantané de mémoire.

5 Pour afficher les données dans l'instantané de mémoire, cliquez deux fois sur l'instantané de mémoire dans la vue Profiler.

L'exemple suivant illustre la vue Instantané de la mémoire :



Le tableau suivant répertorie les colonnes de la vue Instantané de la mémoire :

Colonne	Description
Classe	Classes qui avaient des occurrences dans la mémoire au moment où vous avez enregistré l'instantané de la mémoire.
Package	Package dans lequel se trouve chaque classe. Si la classe n'est pas dans un package, la valeur de ce champ est le nom du fichier dans lequel se trouve la classe. Le nombre suivant le signe du dollar est un identifiant unique de cette classe. Si le champ Package est vide, la classe se trouve dans le package global ou le package non nommé.
Occurrences	Nombre d'occurrences de chaque classe dans la mémoire au moment où vous avez enregistré l'instantané de la mémoire.
Mémoire	Quantité de mémoire, exprimée en octets, que toutes les occurrences de chaque classe utilisaient au moment de la prise de l'instantané de mémoire.

L'instantané de mémoire est généralement utilisé comme point de départ pour déterminer sur quelles classes vous concentrer pour optimiser la mémoire ou trouver des pertes de mémoire. Pour ce faire, créez plusieurs instantanés de mémoire à différents moments, puis comparez les différences dans les vues Objets vagabonds ou Suivi des allocations.

Vous pouvez enregistrer des instantanés de la mémoire pour comparer l'état d'une application au cours d'une session de profilage différente. Pour plus d'informations, voir « Enregistrement et chargement des données de profilage » à la page 155.

Lorsque vous cliquez deux fois sur une ligne de la vue Instantané de la mémoire, le profileur affiche la vue Références de l'objet. Cette vue affiche les traces de pile correspondant aux occurrences de la classe en cours. Vous pouvez consulter les traces de pile pour les occurrences de la classe en cours dans la vue Références de l'objet. Pour plus d'informations sur la vue Références de l'objet, voir « Vue Références de l'objet » à la page 140.

Vous pouvez également spécifier quelles données afficher dans la vue Instantané de la mémoire grâce aux filtres du profileur. Pour plus d'informations, voir « Filtres du profileur » à la page 150.

Vue Références de l'objet

La vue Références de l'objet affiche les traces de pile pour les classes qui ont été instanciées dans l'application.

La vue Références de l'objet affiche des informations sur les occurrences de la classe sélectionnée sous forme d'arborescence. La vue Références de l'objet affiche également les chemins d'accès aux références arrière de l'objet, menant à la racine de la RP. S'il existe plusieurs occurrences d'un chemin d'accès à une référence d'objet, le chemin d'accès n'est pas répété.

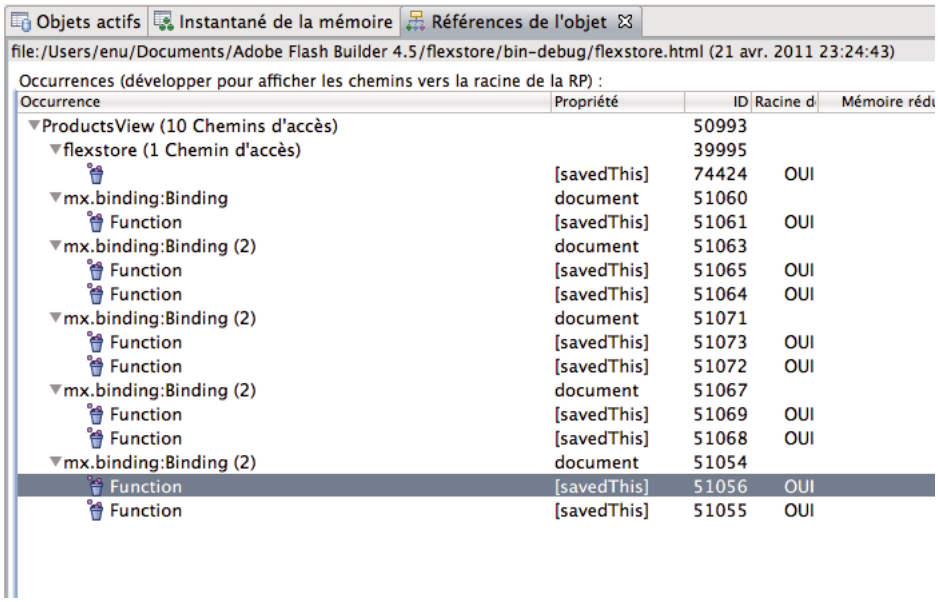
Pour ouvrir la vue Références de l'objet, cliquez deux fois sur le nom d'une classe dans les vues Instantané de la mémoire ou Objets vagabonds.

La vue Références de l'objet affiche les données dans deux tables : Occurrences et Suivi des allocations.

La table Occurrences répertorie tous les objets contenant des références à l'objet actuel. Le numéro entre parenthèses figurant après chaque nom de classe est représenté en tant que nœud dans l'arborescence. Il représente le nombre de chemins d'accès menant à la racine de la RP à partir de ce nœud.

Vous ne pouvez pas voir le nombre de références futures d'un objet. Un objet sans références n'est pas listé dans cette table. Cela ne devrait pas se produire, car l'objet en question doit être nettoyé de la mémoire s'il ne possède aucune référence.

L'exemple suivant illustre la table Occurrences de la vue Références de l'objet :

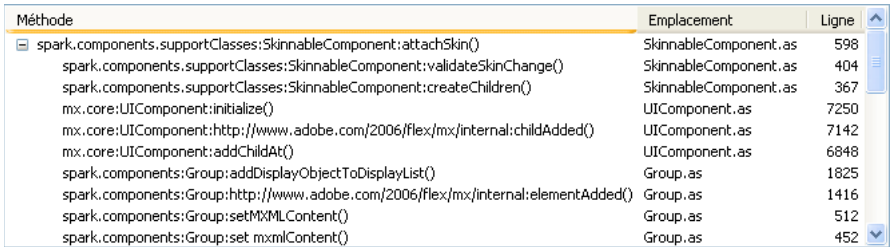


Le tableau suivant répertorie les colonnes de la table Occurrences :

Colonne	Description
Occurrence	Classe de l'objet contenant une référence à l'objet spécifié. Développez l'occurrence d'une classe pour afficher les chemins menant à l'objet. Le nombre de chemins peut être configuré en accédant à l'option Filtres de la vue Instantané de la mémoire.
Propriété	Propriété de l'objet contenant une référence à l'objet spécifié. Par exemple, si vous avez un objet o avec une propriété `i`, et que vous attribuez cette propriété pour qu'elle pointe sur l'étiquette du bouton : `o.i = myButton.label;` Une référence est créée à l'étiquette `monBouton.label` à partir de la propriété `i`.
ID	Identifiant de référence de l'objet contenant la référence à l'objet sélectionné.
Racine de la RP	Indique si un objet possède une référence arrière à la racine de la RP (récupération de place). Développez un nœud d'objet dans l'arborescence pour afficher s'il existe une référence arrière à la racine de la RP.
Mémoire superficielle	Indique la taille de l'objet.
Mémoire conservée	Indique la taille de l'objet et celle de tous les autres objets référencés.

La table Suivi des allocations affiche la trace de la pile de l'occurrence sélectionnée dans la table Occurrences. Lorsque vous sélectionnez une occurrence dans la table Occurrences, le profileur affiche la pile d'appels de cette occurrence dans la table Suivi des allocations.

L'exemple suivant illustre la table Suivi des allocations de la vue Références de l'objet :



Le tableau suivant répertorie les colonnes de la table Suivi des allocations :

Colonne	Description
Méthode	La méthode de niveau supérieur de cette table est celle qui a créé l'occurrence de la classe répertoriée dans la table Occurrences. Vous pouvez développer la méthode pour en afficher la trace de pile. Cela peut vous aider à déterminer l'endroit où la pile d'appels a commencé.
Emplacement	Fichier dans lequel la méthode est définie.
Ligne	Numéro de la ligne dans le fichier.

Vous pouvez afficher des données dans cette table uniquement si vous activez le suivi d'allocations au démarrage du profileur.

Vous pouvez ouvrir le code source de la classe sélectionnée en cliquant deux fois sur une classe dans cette table.

Vue Suivi des allocations

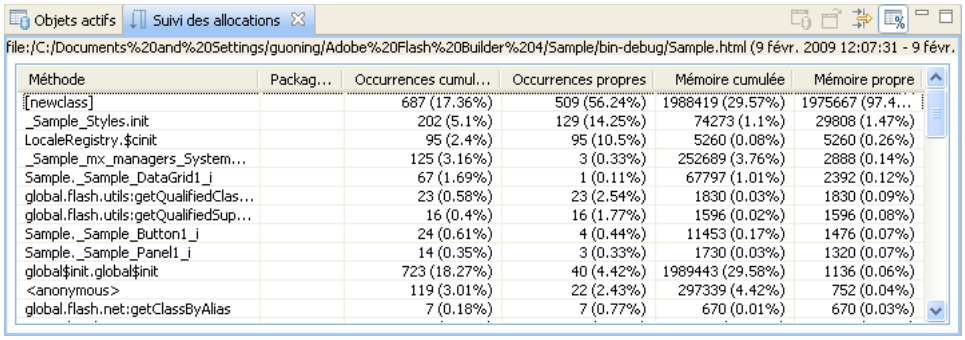
La vue Suivi des allocations indique les méthodes qui ont été appelées entre deux instantanés de mémoire et la quantité de mémoire consommée lors de ces appels de méthode. Pour ouvrir la vue Suivi des allocations, sélectionnez deux instantanés de mémoire puis cliquez sur le bouton Afficher le suivi des allocations. Pour plus d'informations sur l'enregistrement d'un instantané de la mémoire, voir « Vue Instantané de la mémoire » à la page 138.

Le résultat de la comparaison des instantanés de la mémoire est une liste de méthodes que Flash Player a exécutées entre les deux instantanés de mémoire. Pour chacune de ces méthodes, le profileur indique le nombre d'objets créés dans cette méthode.

Ces informations peuvent vous aider à optimiser les performances. Une fois que vous avez identifié des méthodes qui créent un nombre excessif d'objets, vous pouvez optimiser ces zones à problème.

Pour pouvoir utiliser la vue Suivi des allocations, activez le suivi des allocations au démarrage du profileur. Cette option est désactivée par défaut.

L'exemple suivant illustre la vue Suivi des allocations :



Le tableau suivant répertorie les colonnes de la vue Suivi des allocations :

Colonne	Description
Méthode	Méthode appelée pendant l'intervalle de l'instantané. Cette colonne contient aussi la classe dont l'occurrence a appelé cette méthode.
Package	Package dans lequel se trouve chaque classe. Si la classe n'est pas dans un package, la valeur de ce champ est le nom du fichier dans lequel se trouve la classe. Le nombre suivant le signe du dollar est un identifiant unique de cette classe. Si le champ Package est vide, la classe se trouve dans le package global ou le package non nommé.
Occurrences cumulées	Nombre d'objets instanciés dans cette méthode et toutes les méthodes appelées à partir de cette méthode.
Occurrences propres	Nombre d'objets instanciés dans cette méthode, à l'exception des objets instanciés dans des appels de méthode suivants à partir de cette méthode.
Mémoire cumulée	Quantité de mémoire, exprimée en octets, utilisée par les objets instanciés dans cette méthode et toutes les méthodes appelées à partir de cette méthode.
Mémoire propre	Quantité de mémoire, exprimée en octets, utilisée par les objets instanciés dans cette méthode, à l'exception de la mémoire utilisée par les objets instanciés dans des appels de méthode suivants à partir de cette méthode.

Lors de l'enregistrement de méthodes au cours d'intervalles d'échantillonnage, le profileur consigne aussi les actions internes de Flash Player. Ces actions apparaissent dans la liste de méthodes entre crochets sous la forme `[mouseEvent]` ou `[newclass]` ou avec des noms similaires. Pour plus d'informations sur les actions internes de Flash Player, voir « Présentation du profilage et fonctionnement du profileur » à la page 133.

Pour ouvrir la vue Statistiques de l'objet, cliquez deux fois sur une ligne de la table Suivi des allocations. Cette vue fournit des informations concernant les objets créés dans la méthode que vous avez sélectionnée. Elle permet aussi de consulter en détail les objets créés dans des méthodes appelées à partir de cette méthode. Pour plus d'informations, voir « Vue Statistiques de l'objet » à la page 143.

Vous pouvez limiter les données affichées dans la vue Suivi des allocations grâce aux filtres du profileur. Pour plus d'informations, voir « Filtres du profileur » à la page 150.

Vue Statistiques de l'objet

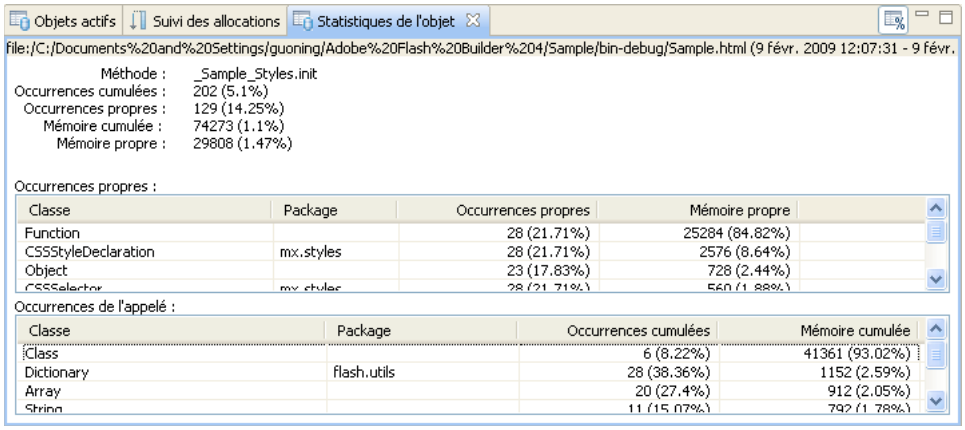
La vue Statistiques de l'objet propose des statistiques concernant les performances du groupe d'objets sélectionné. Cette vue permet d'identifier quelles méthodes appellent un nombre disproportionné d'autres méthodes. Elle indique également la quantité de mémoire que les objets instanciés dans ces appels de méthode consomment. Vous pouvez faire appel à la vue Statistiques de l'objet pour identifier d'éventuelles fuites de mémoire et d'autres sources de problèmes de performances dans votre application.

Pour accéder à la vue Statistiques de l'objet, sélectionnez deux instantanés de mémoire dans la vue Profiler et affichez la comparaison dans la vue Suivi des allocations. Cliquez ensuit deux fois sur une ligne pour afficher les informations dans la vue Statistiques de l'objet.

La vue comporte trois sections :

- Un récapitulatif des statistiques de l'objet sélectionné, notamment le nombre d'occurrences et la quantité de mémoire utilisée.
- La table Occurrences propres : liste d'objets instanciés dans la méthode sélectionnée dans la vue Suivi des allocations. Elle ne répertorie pas les objets instanciés dans les appels de méthode suivants à partir de cette méthode. Le nombre d'objets dans cette liste correspond au nombre d'objets spécifié dans la colonne Occurrences propres de la vue Suivi des allocations.
- La table Occurrences de l'appelé : liste d'objets instanciés dans les méthodes appelées par la méthode sélectionnée dans la vue Suivi des allocations. Elle ne comprend pas les objets instanciés dans la méthode même. Le nombre d'objets dans cette liste correspond au nombre d'objets spécifié dans la colonne Occurrences cumulées de la vue Suivi des allocations.

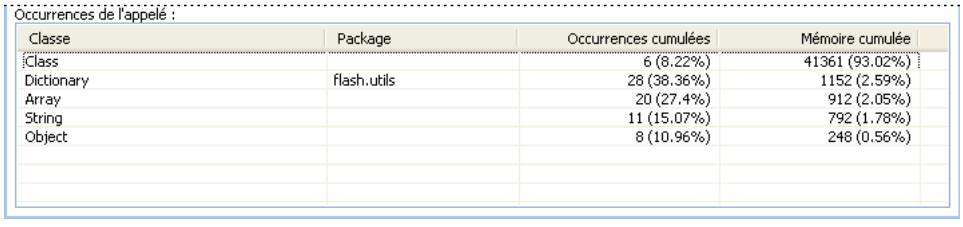
L'exemple suivant illustre le récapitulatif de méthodes et les tables Occurrences propres et Occurrences de l'appelé de la vue Statistiques de l'objet :



Le tableau suivant répertorie les champs de la table Occurrences propres de la vue Statistiques de l'objet :

Colonne	**Description**
Classe	Classes qui ont été instanciées uniquement dans la méthode sélectionnée, à l'exception des classes instanciées dans des appels suivants à partir de cette méthode.
Package	Package dans lequel se trouve chaque classe. Si la classe n'est pas dans un package, la valeur de ce champ est le nom du fichier dans lequel se trouve la classe. Le nombre suivant le signe du dollar est un identifiant unique de cette classe. Si le champ Package est vide, la classe se trouve dans le package global ou le package non nommé.
Occurrences propres	Nombre d'occurrences de cette classe créées uniquement dans la méthode sélectionnée, à l'exception des occurrences créées dans des appels suivants à partir de cette méthode.
Mémoire propre	Quantité de mémoire, exprimée en octets, utilisée par les occurrences créées uniquement dans la méthode sélectionnée, à l'exception de la mémoire utilisée par les occurrences créées dans des appels suivants à partir de cette méthode.

L'exemple suivant illustre la table Occurrences de l'appelé de la vue Statistiques de l'objet :



Le tableau suivant répertorie les champs de la table Occurrences de l'appelé de la vue Statistiques de l'objet :

Colonne	Description
Classe	Classes qui ont été instanciées dans la méthode sélectionnée, y compris les classes instanciées dans des appels suivants à partir de cette méthode.
Package	Package dans lequel se trouve chaque classe. Si la classe n'est pas dans un package, la valeur de ce champ est le nom du fichier dans lequel se trouve la classe. Le nombre suivant le signe du dollar est un identifiant unique de cette classe. Si le champ Package est vide, la classe se trouve dans le package global ou le package non nommé.
Occurrences cumulées	Nombre d'occurrences de cette classe créées dans la méthode sélectionnée et dans des appels suivants à partir de cette méthode.
Mémoire cumulée	Quantité de mémoire, exprimée en octets, utilisée par les occurrences créées dans la méthode sélectionnée et dans des appels suivants à partir de cette méthode.

Vue Profil de performance

La vue Profil de performance est la principale vue à utiliser lors du profilage des performances. Elle propose des statistiques telles que le nombre d'appels, la durée propre et la durée cumulée pour les méthodes appelées au cours d'un intervalle d'échantillonnage particulier. Ces données permettent d'identifier les goulets d'étranglement des performances.

Le processus de profilage des performances conserve une liste des méthodes et des informations concernant ces méthodes qui ont été appelées entre le moment où vous supprimez les données de performances et le moment où vous capturez les nouvelles données. Cet écart de temps est désigné par le terme de *période d'interaction*.

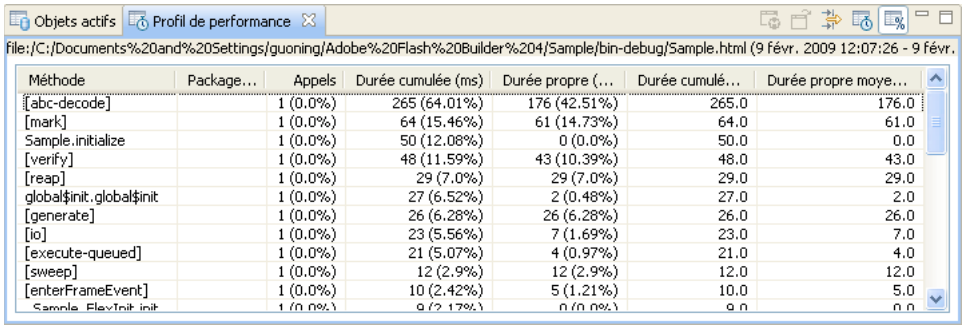
Pour pouvoir utiliser la vue Profil de performance, activez le profilage des performances au démarrage du profileur. C'est la valeur par défaut.

Génération d'un profil de performance

1. Activez le profilage des performances et démarrez une session de profilage.
2. Interagissez avec l'application jusqu'à atteindre le point où vous voulez commencer le profilage.
3. Cliquez sur le bouton Réinitialiser les données de performance.
4. Interagissez avec l'application et effectuez les actions à profiler.
5. Cliquez sur le bouton Capturer le profil de performance.

   L'écart de temps entre le moment où vous cliquez sur Réinitialiser les données de performance et le moment où vous cliquez sur Capturer le profil de performance est la période d'interaction. Si vous ne cliquez pas sur Réinitialiser les données de performance, le profil de performance contiendra toutes les données capturées depuis le premier démarrage de l'application.
6. Cliquez deux fois sur le profil de performance dans la vue Profiler.

L'exemple suivant illustre la vue Profil de performance :



Le tableau suivant répertorie les colonnes de la vue Profil de performance :

Colonne	Description
Méthode	Nom de la méthode et classe à laquelle appartient la méthode. Les actions internes exécutées par Flash Player s'affichent sous la forme d'entrées entre crochets : par exemple, `[mark]` et `[sweep]`. Vous ne pouvez pas changer le comportement de ces actions internes, mais vous pouvez utiliser les informations les concernant pour aider au profilage et à l'optimisation. Pour plus d'informations sur ces actions, voir « Présentation du profilage et fonctionnement du profileur » à la page 133.
Package	Package dans lequel se trouve la classe. Si la classe n'est pas dans un package, la valeur de ce champ est le nom du fichier dans lequel se trouve la classe. Le nombre suivant le signe du dollar est un identifiant unique de cette classe. Si le champ Package est vide, la classe se trouve dans le package global ou le package non nommé.
Appels	Nombre de fois où la méthode a été appelée au cours de la période d'interaction. Si une méthode est appelée un nombre disproportionné de fois par rapport aux autres méthodes, vous pouvez tenter d'optimiser cette méthode en vue de réduire le temps d'exécution.
Durée cumulée	Laps de temps total, exprimé en millisecondes, qu'ont mis à s'exécuter tous les appels à cette méthode, et tous les appels à des méthodes suivantes, au cours de la période d'interaction.
Durée propre	Laps de temps, exprimé en millisecondes, qu'ont mis à s'exécuter tous les appels à cette méthode au cours de la période d'interaction.
Durée cumulée moyenne	Laps de temps moyen, exprimé en millisecondes, qu'ont mis à s'exécuter tous les appels à cette méthode, et tous les appels à des méthodes suivantes, au cours de la période d'interaction.
Durée propre moyenne	Laps de temps moyen, exprimé en millisecondes, qu'a mis à s'exécuter cette méthode au cours de la période d'interaction.

Si vous cliquez deux fois sur une méthode dans la vue Profil de performance, Flex affiche des informations concernant cette méthode dans la vue Statistiques de la méthode. Cela vous permet de consulter le détail de la pile d'appels d'une méthode donnée. Pour plus d'informations, voir « Identification des caractéristiques des performances des méthodes » à la page 147.

Vous pouvez limiter les données affichées dans la vue Profil de performance grâce aux filtres du profileur. Pour plus d'informations, voir « Filtres du profileur » à la page 150.

Vous pouvez enregistrer les profils de performances pour une utilisation ultérieure. Pour plus d'informations, voir « Enregistrement et chargement des données de profilage » à la page 155.

Identification des caractéristiques des performances des méthodes

La vue Statistiques de la méthode présente les caractéristiques des performances de la méthode sélectionnée. Vous pouvez faire appel à la vue Statistiques de la méthode pour identifier des goulets d'étranglement dans l'application. En affichant, par exemple, les temps d'exécution d'une méthode, vous pouvez voir les méthodes qui prennent un temps disproportionné pour s'exécuter. Vous pouvez ensuite optimiser de manière sélective ces méthodes.

Pour plus d'informations, voir « Vue Profil de performance » à la page 145.

Affichage des informations concernant les méthodes dans la vue Statistiques de la méthode

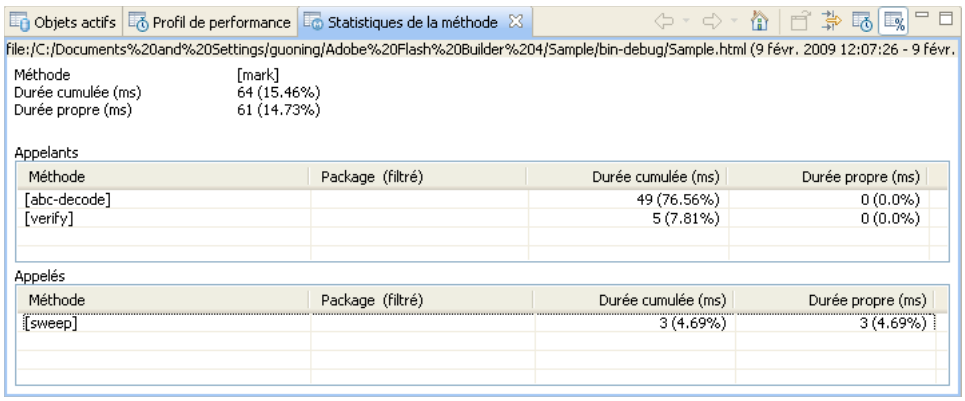
1 Cliquez deux fois sur une ligne dans la vue Profil de performance ou sélectionnez une méthode dans cette même vue.

2 Cliquez sur le bouton Ouvrir les statistiques de la méthode.

La vue comporte trois sections :

- Un récapitulatif des performances de la méthode sélectionnée, notamment le nombre d'appels, la durée cumulée et la durée propre.
- La table Appelants : informations concernant les méthodes qui ont appelé la méthode sélectionnée. Dans certains cas, il est important de savoir si la méthode sélectionnée est trop appelée, la façon dont elle est utilisée et si elle est utilisée correctement.
- La table Appelés : informations concernant les méthodes appelées par la méthode sélectionnée.

L'exemple suivant illustre le récapitulatif de méthodes et les tables Appelants et Appelés de la vue Statistiques de la méthode :



Le tableau suivant répertorie les champs de la table Appelants de la vue Statistiques de la méthode :

Colonne	Description
Méthode	Méthodes qui ont appelé la méthode apparaissant dans le récapitulatif de la partie supérieure de cette vue. Si cette liste est vide, la méthode cible n'a été appelée par aucune méthode qui n'est pas filtrée.

Colonne	Description
Package	Package dans lequel se trouve chaque classe. Si la classe n'est pas dans un package, la valeur de ce champ est le nom du fichier dans lequel se trouve la classe. Le nombre suivant le signe du dollar est un identifiant unique de cette classe. Si le champ Package est vide, la classe se trouve dans le package global ou le package non nommé.
Durée cumulée	Laps de temps, exprimé en millisecondes, qu'a mis à s'exécuter chaque méthode appelante, et toutes les méthodes suivantes.
Durée propre	Laps de temps, exprimé en millisecondes, qu'a mis à s'exécuter chaque méthode appelante, à l'exception des méthodes appelées par les méthodes suivantes.

Le tableau suivant répertorie les champs de la table Appelés de la vue Statistiques de la méthode :

Colonne	Description
Méthode	Méthodes appelées par la méthode apparaissant dans le récapitulatif de la partie supérieure de cette vue. Si cette liste est vide, la méthode cible n'a été appelée par aucune méthode qui n'est pas filtrée.
Package	Package dans lequel se trouve chaque classe. Si la classe n'est pas dans un package, la valeur de ce champ est le nom du fichier dans lequel se trouve la classe. Le nombre suivant le signe du dollar est un identifiant unique de cette classe. Si le champ Package est vide, la classe se trouve dans le package global ou le package non nommé.
Durée cumulée	Laps de temps (exprimé en millisecondes) qu'a mis à s'exécuter chaque méthode appelée et toutes les méthodes suivantes.
Durée propre	Laps de temps (exprimé en millisecondes) qu'a mis à s'exécuter chaque méthode appelée, à l'exception des méthodes appelées par les méthodes suivantes.

Vous pouvez parcourir la pile d'appels pour rechercher des goulets d'étranglement des performances en cliquant sur les méthodes dans la table Appelants ou Appelés. Si vous cliquez deux fois sur une méthode dans ces tables, le profileur affiche le récapitulatif de cette méthode dans la partie supérieure de la vue Statistiques de la méthode puis affiche les appelants et les appelés pour la méthode nouvellement sélectionnée dans les deux tables.

Remarque : *la durée cumulée moins la durée propre dans cette vue n'équivaut pas toujours à la durée cumulée des appelants. La raison en est que si vous remontez à un appelant, la durée cumulée correspond à la durée propre de cet appelant plus toutes les chaînes à partir desquelles la méthode de départ a été appelée, mais pas les autres appelés.*

Vous pouvez aussi utiliser les boutons Précédent, Suivant et Valeur par défaut du profileur pour parcourir la pile d'appels.

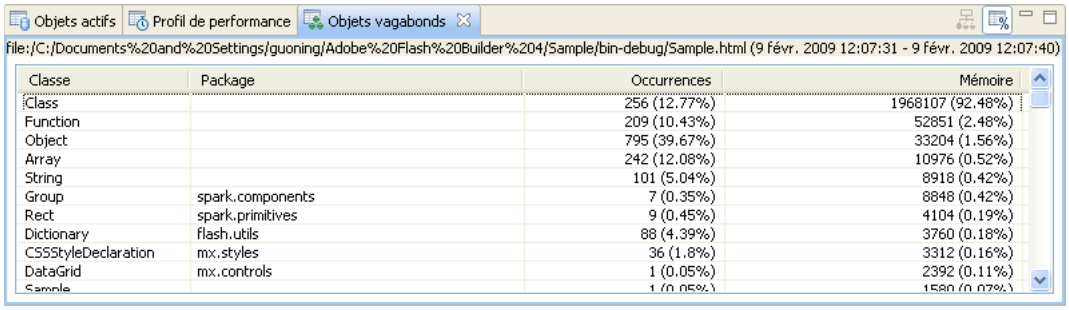
Vous pouvez limiter les données affichées dans la vue Statistiques de la méthode grâce aux filtres du profileur. Pour plus d'informations, voir « Filtres du profileur » à la page 150.

Vue Objets vagabonds

La vue Objets vagabonds présente les différences entre deux instantanés de mémoire de l'application que vous profilez. Les différences que cette vue illustre sont le nombre d'occurrences d'objets dans la mémoire et la quantité de mémoire que ces objets utilisent. Ces données sont utiles pour identifier les fuites de mémoire. Le laps de temps écoulé entre deux instantanés de la mémoire est l'*intervalle d'instantanés*.

Pour ouvrir la vue Objets vagabonds, sélectionnez deux instantanés de mémoire et cliquez sur le bouton Objets vagabonds. Pour plus d'informations sur l'enregistrement d'un instantané de la mémoire, voir « Vue Instantané de la mémoire » à la page 138.

L'exemple suivant illustre la vue Objets vagabonds :



Le tableau suivant répertorie les colonnes de la vue Objets vagabonds :

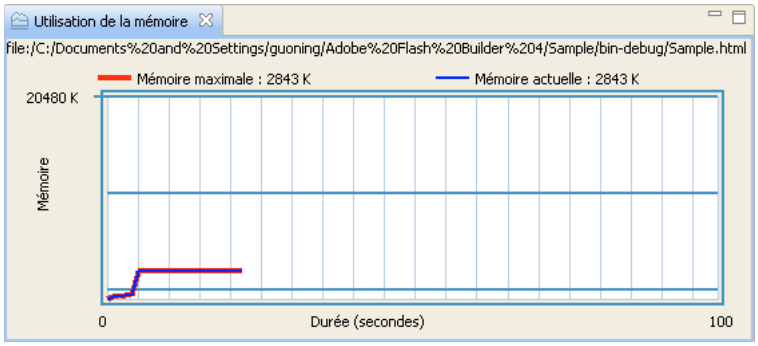
Colonne	**Description**
Classe	Classes créées mais pas détruites pendant l'intervalle d'instantanés.
Package	Package dans lequel se trouve chaque classe. Si la classe n'est pas dans un package, la valeur de ce champ est le nom du fichier dans lequel se trouve la classe. Le nombre suivant le signe du dollar est un identifiant unique de cette classe. Si le champ Package est vide, l'objet se trouve dans le package global ou le package non nommé.
Occurrences	Nombre d'occurrences créées pendant l'intervalle d'instantanés. Différence entre le nombre d'occurrences de chaque classe dans le premier intervalle et dans le second intervalle. Par exemple, si le premier instantané contient 22 567 chaînes et le second 22 861 chaînes, la valeur de ce champ est 294.
Mémoire	Quantité de mémoire allouée pendant l'intervalle d'instantanés. Différence entre la quantité de mémoire que les instances de chaque classe ont utilisée lors de la prise du premier instantané et lors de la prise du second instantané. Par exemple, si la classe Strings occupe 2 031 053 octets dans le premier instantané et 2 029 899 dans le second, la valeur de ce champ est 1 154 octets.

Pour plus d'informations sur l'identification de fuites de mémoire, voir « Localisation des fuites de mémoire » à la page 158.

Graphique Utilisation de la mémoire

Le graphique Utilisation de la mémoire présente la mémoire utilisée par l'application que vous profilez. Cette valeur est différente de l'utilisation de la mémoire de Flash Player et son navigateur. La raison en est que cette valeur ne comprend pas l'utilisation de la mémoire de l'agent du profileur ou du navigateur. Elle correspond uniquement à la somme des objets actifs de l'application profilée. En conséquence, si vous comparez la valeur de l'utilisation de la mémoire dans ce graphique à la quantité de mémoire que le navigateur utilise telle qu'indiquée, par exemple, dans le Gestionnaire des tâches de Windows, vous obtenez des résultats différents.

L'image suivante présente le graphique de la vue Utilisation de la mémoire :



La valeur de Mémoire actuelle est égale à la somme des totaux de la colonne Mémoire de la vue Objets actifs, en supposant que tous les filtres soient désactivés.

La valeur de Mémoire maximale correspond à la quantité la plus élevée de mémoire que cette application a utilisée au cours de la session de profilage en cours.

Le graphique Utilisation de la mémoire montre l'état de la mémoire de l'application au cours des 100 dernières secondes. Vous ne pouvez pas configurer ce nombre et vous ne pouvez pas enregistrer de données historiques pour le diagramme.

Si vous fermez le graphique Utilisation de la mémoire et que vous voulez le rouvrir, cliquez sur la liste déroulante dans la vue Profiler et sélectionnez Utilisation de la mémoire.

Filtres du profileur

La quantité de données contenue dans une vue du profileur peut quelquefois être considérable et le niveau de détail trop important. Les actions internes de Flash Player peuvent occulter les données qui vous intéressent véritablement, telles que vos propres méthodes et classes. En outre, Flash Player crée et détruit de nombreux objets sans votre intervention directe. Vous pouvez par conséquent constater que des milliers de chaînes ou de tableaux sont utilisés dans votre application.

Vous pouvez définir des filtres dans les vues suivantes :

- Objets actifs
- Instantané de la mémoire
- Profil de performance
- Statistiques de la méthode
- Suivi des allocations

Les filtres du profileur permettent de définir quels packages apparaissent dans les vues du profileur. Il existe deux types de filtres :

Filtres d'exclusion : les filtres d'exclusion indiquent au profileur qu'il doit exclure de ses vues les packages qui correspondent aux motifs de sa liste de motifs. Si vous faites appel, par exemple, à des commandes de création de graphiques, mais que vous ne voulez pas les profiler, vous pouvez ajouter le motif mx.charts.* au filtre d'exclusion. Vous pouvez également exclure des éléments intégrés globaux, ce qui comprend les classes globales String et Array.

Filtres d'inclusion : les filtres d'exclusion indiquent au profileur qu'il doit inclure dans ses vues uniquement les packages qui correspondent aux motifs de sa liste de motifs. Si vous possédez, par exemple, un package personnalisé

nommé com.mycompany.*, vous pouvez afficher des informations uniquement sur les classes contenues dans ce package en l'ajoutant au filtre d'inclusion.

Les exclusions par défaut sont les classes flash.*.*, spark.*.*, mx.*.*, ainsi que les classes d'infrastructure Flex dans le package global ou non nommé, ce qui comprend les classes globales String et Array. Cela signifie que les inclusions par défaut sont des classes personnalisées se trouvant dans le package non nommé et des classes personnalisées dans des packages hors infrastructure (telles que com.mycompany.MyClass).

Vous pouvez exclure les classes personnalisées se trouvant dans le package non nommé à partir des données de profilage. Pour ce faire, ajoutez « * » à la liste d'exclusion.

Nombre maximal de lignes visibles : l'option Nombre maximal de lignes visibles définit le nombre de rangées de données qui peuvent être affichées dans une vue. Augmentez cette valeur si les données que vous recherchez ne s'affichent pas dans la vue. Réduisez cette valeur pour améliorer les performances du profileur. Utilisez d'autres filtres pour vérifier que vous affichez les données qui vous intéressent.

Nombre maximal de chemins de références arrière à rechercher : l'option Nombre maximal de chemins de références arrière à rechercher définit le nombre de chemins vers un objet référencé à afficher lorsque vous examinez les références d'un objet. Les chemins s'affichent selon le chemin le plus court. Par défaut, les dix chemins les plus courts s'affichent. Augmentez cette valeur pour afficher d'autres chemins ou sélectionnez Afficher tous les chemins de références arrière. L'affichage de chemins supplémentaires peut vous aider à trouver les objets qui ne sont plus référencés. Voir « Localisation des fuites de mémoire » à la page 158.

Définition de préférences de filtres par défaut

- Ouvrez la boîte de dialogue Préférences et sélectionnez Flash Builder > Profileur > Filtres d'inclusion ou Filtres d'exclusion.

Lors de l'affichage des données du profileur, celui-ci applique d'abord les filtres d'exclusion, puis les filtres d'inclusion. Supposons que vous définissiez le filtre d'exclusion sur mx.controls.*, mais que vous définissiez le filtre d'inclusion sur mx.*.*. Le profileur n'affiche aucune information concernant les classes du package mx.controls, même si leur motif correspond à la liste de motifs d'inclusion, car ce package est exclu. Supposons également que vous définissiez le filtre d'exclusion sur mx.*.* et le filtre d'inclusion sur mx.controls.*. Le profileur n'affiche aucune information concernant les classes du package mx.controls.* car elles ont été exclues avant l'inclusion du package.

Lorsque vous filtrez certains points de données, les valeurs de pourcentage des colonnes sont ajustées afin de refléter uniquement le pourcentage de données non filtrées.

Le profileur conserve les filtres d'une session de profilage à l'autre pour la même application.

Les paramètres de filtres ne sont pas hérités par les vues secondaires. Par exemple, si vous appliquez un filtre aux données contenues dans la vue Instantané de la mémoire, puis accédez à la vue Références de l'objet en cliquant deux fois sur une méthode, la vue Références de l'objet n'applique pas le même filtre.

Déterminer si les données sont filtrées

1. Cliquez sur le bouton Filtres ou examinez le titre des tables de données. Si des filtres sont appliqués, l'en-tête de la colonne Package est Package (filtré).
2. (Facultatif) Réinitialisez les filtres aux valeurs par défaut en cliquant sur le bouton Restaurer les valeurs par défaut.

Utilisation du profileur Flash Builder

Le profileur nécessite Flash Player version 9.0.124 ou une version ultérieure. Vous pouvez profiler des applications qui ont été compilées pour Flex 2, Flex 2.0.1 et Flex 3. Vous pouvez utiliser le profileur pour profiler des applications ActionScript 3.0 créées avec des outils de programmation Flash, ainsi que des applications de bureau exécutées sur Adobe AIR.

Le profileur a besoin d'informations de débogage dans l'application que vous êtes en train de profiler. Lorsque vous compilez une application et que vous lancez le profileur, Flash Builder inclut par défaut les informations de débogage dans l'application. Vous pouvez inclure explicitement les informations de débogage dans n'importe quelle application en définissant l'option `débug` du compilateur sur `true`. Si vous exportez une application par le biais de l'option Exporter vers une version validée, l'application ne contient aucune information de débogage.

Barre d'outils du profileur

Le tableau suivant présente les boutons de la barre d'outils du profileur :

Bouton	Nom	Description
	Reprendre	Reprend la session de profilage. Cette option est active uniquement si le nom d'une application est sélectionné et qu'elle est interrompue.
	Interrompre	Interrompt la session de profilage. Cette option est active uniquement si le nom d'une application est sélectionné et qu'elle est en cours d'exécution.
	Arrêter	Met fin à la session de profilage. Cette option est active uniquement si le nom d'une application est sélectionné et qu'elle n'a pas déjà été fermée.
	Lancer la récupération de place	Demande à Flash Player de récupérer de la place. Cette option est active uniquement si le nom d'une application est sélectionné et qu'elle est en cours d'exécution. Pour plus d'informations sur la récupération de place, voir « Récupération de place » à la page 157.
	Prendre un instantané de la mémoire	Enregistre l'utilisation que fait une application de la mémoire pour que vous puissiez la consulter ou la comparer à d'autres instantanés. Cette option est active uniquement si le nom d'une application est sélectionné, que celle-ci est en cours d'exécution et si vous sélectionnez Activer le profilage de la mémoire dans la boîte de dialogue de lancement. Le profileur ajoute de nouveaux instantanés de la mémoire en tant qu'enfants de l'application sélectionnée dans la vue Profiler. Pour ouvrir le nouvel instantané de mémoire dans la vue Instantané de la mémoire, cliquez deux fois sur l'entrée d'instantané de la mémoire. La récupération de place se fait implicitement avant l'enregistrement d'instantanés de la mémoire. En d'autres termes, un clic sur le bouton Prendre un instantané de la mémoire équivaut à un clic sur le bouton Lancer la récupération de place puis à un clic sur le bouton Prendre un instantané de la mémoire. Pour plus d'informations sur l'utilisation des instantanés de mémoire, voir « Vue Instantané de la mémoire » à la page 138.
	Rechercher les objets vagabonds	Compare deux instantanés de la mémoire dans la vue Objets vagabonds. Cette option est active uniquement si deux instantanés de mémoire sont sélectionnés et que l'option Activer le profilage de la mémoire est sélectionnée dans la boîte de dialogue de lancement. Pour plus d'informations sur la vue Objets vagabonds, voir « Vue Objets vagabonds » à la page 148.

Bouton	Nom	Description
	Afficher le suivi des allocations	Compare les méthodes entre deux instantanés de la mémoire dans la vue Suivi des allocations. Cette option est active uniquement si deux instantanés de mémoire sont sélectionnés et que l'option Activer le profilage de la mémoire est sélectionnée dans la boîte de dialogue de lancement. Pour plus d'informations sur la vue Suivi des allocations, voir « Vue Suivi des allocations » à la page 142.
	Réinitialiser les données de performance	Efface les données de profilage des performances. Cette option est active uniquement si le nom d'une application est sélectionné, que celle-ci est en cours d'exécution, et si vous sélectionnez Activer le profilage de la mémoire dans la boîte de dialogue de lancement. Généralement, vous cliquez sur ce bouton, vous interagissez avec l'application, puis vous cliquez sur le bouton Capturer le profil de performance pour obtenir un instantané des performances de l'application à partir du moment où vous réinitialisez les données. Pour plus d'informations sur la vue Profil de performance, voir « Vue Profil de performance » à la page 145.
	Capturer le profil de performance	Enregistre un nouvel instantané des performances en tant qu'enfant de l'application sélectionnée. Cette option est active uniquement si le nom d'une application est sélectionné, que celle-ci est en cours d'exécution, et si vous sélectionnez Activer le profilage des performances dans la boîte de dialogue de lancement. Pour ouvrir la vue Profil de performance, cliquez deux fois sur l'entrée d'instantané des performances. Pour plus d'informations sur la vue Profil de performance, voir « Vue Profil de performance » à la page 145.
	Supprimer	Supprime de la mémoire les données de l'instantané sélectionné. Un clic sur ce bouton supprime aussi l'application de la vue du profil, si l'application a été fermée. Cette option est activée uniquement si un instantané de performances ou de mémoire est sélectionné.
Sans objet	Enregistrer	Enregistre les données de profilage sur le disque. L'option Enregistrer est également disponible à partir du menu de la vue Profiler. Cette option est active uniquement si le nom d'une application est sélectionné.

Certaines vues du profileur, telles que Statistiques de la méthode et Statistiques de l'objet, possèdent des boutons de navigation qui permettent de parcourir la pile ou de modifier la vue. Le tableau suivant présente les boutons de navigation dans ces vues du profileur :

Bouton	Nom	Description
	Précédent	Affiche toutes les méthodes que vous avez parcourues à partir de la première méthode sélectionnée, jusqu'à la méthode en cours d'affichage.
	Suivant	Affiche la méthode en cours d'affichage et les méthodes qui mènent à la méthode sélectionnée. Cet élément est actif une fois que vous êtes revenu en arrière.
	Accueil	Affiche la méthode sélectionnée.

Bouton	Nom	Description
	Ouvrir le fichier source	Ouvre un éditeur de source qui affiche le code source de la méthode sélectionnée.
	Filtres	Vous permet de contrôler les méthodes à inclure dans la table. Pour plus d'informations, voir « Filtres du profileur » à la page 150.
	Afficher/Masquer les méthodes de durée nulle	Affiche ou masque les méthodes comportant un temps égal à 0,00 dans la colonne de temps moyen, car elles n'apparaissent dans aucun échantillon.

Démarrage, arrêt et reprise du profileur

Vous pouvez profiler des applications que vous êtes en train de créer dans Flash Builder. Flash Builder inclut des informations de débogage lors de la compilation et de l'exécution d'une application lors d'une session de débogage. Vous pouvez également profiler dans Flash Builder des applications externes que vous n'êtes pas en train de développer, mais dont le fichier SWF est disponible avec une URL ou dans le système de fichiers. Le fichier SWF d'une application doit contenir des informations de débogage pour que vous puissiez la profiler. Pour plus d'informations, voir « Profilage des applications externes » à la page 157.

Début du profilage d'une application

1 Fermez toutes les occurrences du navigateur.

2 Ouvrez l'application dans Flash Builder.

3 Cliquez sur le bouton Profiler *nom_application* dans la barre d'outils principale. Flash Builder vous invite à fermer toutes les occurrences de vos navigateurs si ce n'est pas déjà fait.

4 Cliquez sur OK. Flash Builder compile l'application et la lance dans une fenêtre de navigateur distincte. Flash Builder ouvre également la boîte de dialogue Configurer le profileur.

5 Sélectionnez les options de la boîte de dialogue Configurer le profileur puis cliquez sur Reprendre. Pour profiler une application, sélectionnez l'option Activer le profilage de la mémoire ou l'option Activer le profilage des performances. Vous pouvez sélectionner ces deux options lors du profilage d'une application.

Le tableau suivant présente les options :

Paramètre	Description
Connecté à partir de	Indique le serveur depuis lequel vous lancez l'application. Si l'application s'exécute sur le même ordinateur que le profileur, cette valeur est localhost. Vous ne pouvez pas modifier cette valeur. Vous pouvez toutefois profiler une application s'exécutant sur un ordinateur distinct. Voir « Profilage des applications externes » à la page 157.
Application	Précise l'application que vous êtes sur le point de profiler. Vous ne pouvez pas modifier cette valeur.
Activer le profilage de la mémoire	Demande au profileur de recueillir des données concernant la mémoire Utilisez cette option pour détecter des fuites de mémoire ou un nombre excessif de créations d'objets. Si vous êtes en train d'effectuer le profilage des performances, vous pouvez désélectionner cette option. L'option Activer le profilage de la mémoire est sélectionnée par défaut.

Paramètre	Description
Surveiller les données de mémoire active	Demande au profileur d'afficher des données concernant la mémoire dans la vue Objets actifs lors du profilage. Cela n'est pas obligatoire pour le profilage de la mémoire ni le profilage des performances. Vous pouvez sélectionner cette option uniquement si vous avez sélectionné Activer le profilage de la mémoire. L'option Surveiller les données de mémoire active est sélectionnée par défaut.
Générer des traces de pile d'allocation d'objets	Demande au profileur de capturer une trace de pile à chaque création d'objet. L'activation de cette option ralentit le profilage. Cette option n'est généralement sélectionnée qu'en cas de nécessité. Elle n'est disponible que si l'option Activer le profilage de la mémoire a été activée. L'option Générer des traces de pile d'allocation d'objets est désactivée par défaut. Elle doit être activée pour pouvoir afficher les informations liées au suivi des allocations dans la vue Références de l'objet ou Suivi des allocations.
Activer le profilage des performances	Demande au profileur de recueillir les données liées à la trace de la pile aux intervalles d'échantillonnage. Utilisez ces échantillons pour déterminer les processus auxquels la majeure partie du temps d'exécution de l'application est employée. Si vous êtes en train d'effectuer le profilage de la mémoire, vous pouvez désélectionner cette option. L'option Activer le profilage des performances est sélectionnée par défaut.

Vous pouvez changer les valeurs par défaut de ces options en modifiant les préférences de profilage. Pour plus d'informations, voir « Définition des préférences du profileur » à la page 156.

6 Vous pouvez maintenant commencer à interagir avec l'application et à examiner les données du profileur.

Profilage d'une application sous Windows 8

Pour profiler une application sous Windows 8, reportez-vous à cet article technique.

Interruption et reprise du profilage d'une application

Une fois que vous avez démarré le profileur, vous pouvez interrompre et redémarrer les applications dans la vue Profiler. Sélectionnez une application puis sélectionnez l'action à effectuer sur cette application. L'exemple suivant illustre la vue Profiler avec plusieurs applications. Une application est en cours d'exécution ; toutes les autres sont fermées.

Arrêt du profilage d'une application

1 Sélectionnez l'application dans la vue Profiler.

2 Cliquez sur le bouton Arrêter pour mettre fin manuellement à la session de profilage. Cette opération ne ferme pas le navigateur ni ne termine le processus Flash Player.

3 Pour retourner à la perspective Développement Flex, sélectionnez Développement Flex dans la liste déroulante de perspective. Vous pouvez également changer les perspectives en appuyant sur les touches Ctrl+F8 sous Windows.

Enregistrement et chargement des données de profilage

Une fois le profileur exécuté, vous pouvez enregistrer les données afin de comparer un instantané de la session de profilage en cours avec un instantané pris après avoir modifié le code. Vous pouvez ainsi déterminer si vous avez identifié les causes responsables des problèmes et si les modifications apportées améliorent les performances et l'utilisation de la mémoire par l'application.

Lors de l'enregistrement des données de profilage, vous enregistrez toutes les données de l'application dans ce profil, à savoir tous les profils de performance, les instantanés de la mémoire et le suivi des allocations. Flash Builder enregistre ces informations dans un groupe de fichiers binaires à l'emplacement que vous spécifiez.

Enregistrement des données de profilage

1. Sélectionnez l'application dans la vue Profiler.
2. Ouvrez la liste déroulante dans la vue Profiler et sélectionnez Enregistrer. La boîte de dialogue de sélection de dossier apparaît.
3. Choisissez un emplacement pour enregistrer les données de profil et cliquez sur OK. Créez un dossier pour chaque jeu de données de profilage à enregistrer. Si vous choisissez le même dossier, les nouvelles données remplacent les anciennes.

Récupération des données de profilage enregistrées

1. Sélectionnez la vue Données de profilage enregistrées.
2. Cliquez sur le bouton Ouvrir. La boîte de dialogue Rechercher un dossier apparaît.
3. Recherchez le dossier contenant les données de profilage de l'application, puis cliquez sur OK. Flash Builder affiche les données de profilage disponibles dans la vue Données de profilage enregistrées. Vous ne pouvez pas reprendre l'application dans cette vue, mais vous pouvez afficher les instantanés de la mémoire ou les autres données que vous avez enregistrées.

Vous ne pouvez pas supprimer les données d'application enregistrées depuis Flash Builder.

Suppression des données de profilage

1. Sélectionnez l'instantané de l'application dans la vue Profiler.
2. Cliquez sur le bouton Supprimer.

Définition des préférences du profileur

Vous pouvez définir certaines préférences du profileur de manière telle à ce que les paramètres s'appliquent à toutes les sessions de profilage. Ces paramètres vous permettent de définir le Flash Player/navigateur dans lequel vous profilez l'application, ainsi que les filtres par défaut et le numéro de port sur lequel l'application est disponible, si l'application profilée s'exécute sur un serveur.

Définition des préférences de Flash Builder pour plusieurs sessions de profilage

- Ouvrez la boîte de dialogue Préférences et sélectionnez Flash Builder > Profileur.

Sélectionnez les options du menu Profileur pour y accéder. Le tableau suivant répertorie les préférences que vous pouvez définir :

Option de menu	Description
Profileur	Sélectionnez la méthode de profilage par défaut. Vous pouvez activer ou désactiver le profilage de la mémoire et le profilage des performances.
Connexions	Définissez le numéro du port sur lequel Flash Builder écoute l'application profilée. Le numéro de port par défaut est 9999. Vous ne pouvez pas changer le port en 7935, car c'est le port utilisé par le débogueur.

Option de menu	Description
Filtres d'exclusion	Définissez les packages par défaut qui sont exclus des vues du profileur. Pour plus d'informations sur l'utilisation des filtres, voir « Filtres du profileur » à la page 150.
Filtres d'inclusion	Définissez les packages par défaut qui sont inclus dans les vues du profileur. Tous les autres packages sont exclus. Pour plus d'informations sur l'utilisation des filtres, voir « Filtres du profileur » à la page 150.
Lecteur/Navigateur	Définissez l'emplacement du fichier exécutable de Flash Player et du navigateur que Flash Builder utilise pour exécuter votre application profilée.

Profilage des applications externes

Outre les applications que vous développez dans Flash Builder, vous pouvez profiler des applications externes. Il peut s'agir de fichiers SWF situés dans n'importe quel emplacement accessible, notamment des applications situées sur un serveur Web distant ou d'une application se trouvant dans votre système de fichiers local.

Dans le cas d'un fichier SWF, vous pouvez spécifier une URL ou un emplacement de système de fichiers. Si vous indiquez une URL, Flash Builder lance le fichier SWF de l'application dans le navigateur par défaut. Le navigateur doit utiliser la version débogueur de Flash Player pour pouvoir profiler l'application correctement.

Si vous spécifiez un emplacement de système de fichiers pour le fichier SWF, Flash Builder ouvre l'application dans la version débogueur du Flash Player autonome. En règle générale, utilisez une URL pour demander le fichier. L'exécution des applications dans la version autonome de Flash Player peut produire des résultats inattendus, plus particulièrement si l'application utilise des services distants ou des appels réseau.

Profilage d'une application externe

1. Placez-vous dans la perspective Profil Flash.
2. Sélectionnez Profiler > Profiler une application externe. La boîte de dialogue Profiler une application externe apparaît.
3. Sélectionnez l'option (par défaut) Lancer l'application sélectionnée puis cliquez sur le bouton Nouveau. La boîte de dialogue Ajouter une application apparaît.

   Vous pouvez aussi lancer l'application manuellement en sélectionnant l'option Lancer l'application manuellement hors Flash Builder.
4. Saisissez l'emplacement du fichier SWF puis cliquez sur OK ou sur le bouton Parcourir et localisez votre application dans votre système de fichiers.
5. Cliquez sur le bouton Lancer. Si vous avez indiqué une URL comme emplacement de l'application, Flash Builder lance l'application à partir du navigateur par défaut. Si vous avez spécifié un emplacement de système de fichiers pour l'application, Flash Builder ouvre l'application dans la version autonome de Flash Player.

   Si vous avez spécifié un fichier SWF qui a été compilé sans informations de débogage, Flash Builder retourne une erreur. Recompilez l'application avec l'option `debug` du compilateur définie sur `true`, puis relancez-la.

Récupération de place

La *récupération de place* est l'action de supprimer de la mémoire des objets qui ne sont plus nécessités. La mémoire utilisée par des occurrences n'ayant plus de références à ces objets doit être désallouée pendant ce processus.

Flash Player nettoie la mémoire en fonction des besoins au cours du cycle de vie d'une application. Le déréférencement d'un objet ne déclenche pas le nettoyage de la mémoire. Ainsi, lorsque vous supprimez toutes les références à un objet, le nettoyage de la mémoire ne désalloue pas nécessairement la mémoire pour cet objet. Cet objet devient une cible potentielle du nettoyage de la mémoire.

Le fait de cliquer sur le bouton Lancer la récupération de place ne garantit pas que tous les objets susceptibles d'être nettoyés le soient. La récupération de place est généralement déclenchée par l'allocation de mémoire à de nouvelles ressources. En effet, lorsque de nouvelles ressources nécessitent de la mémoire qui n'est pas disponible dans l'allocation actuelle, la fonction de récupération de place s'exécute et libère de la mémoire signalée pour la désallocation. Par conséquent, la suppression de toutes les références à la mémoire ne signifie pas qu'elle sera nettoyée immédiatement, mais bien qu'elle le sera probablement lors de la création et de l'utilisation d'autres occurrences. Si une application est inactive, vous pouvez consulter l'allocation de sa mémoire. Même si certains objets sont marqués pour le nettoyage, l'utilisation de la mémoire d'une application inactive ne change pas tant que vous ne commencez pas à interagir avec elle.

Flash Player alloue la mémoire en blocs de plusieurs octets, et non octet par octet. Si une partie d'un bloc est marquée pour le nettoyage sans que d'autres parties le soient, le bloc n'est pas libéré. La récupération de place tente d'associer les parties de la mémoire non utilisées dans des blocs plus volumineux pouvant être nettoyés, mais cela ne se produit pas systématiquement à chaque tentative de récupération.

La récupération de place se fait implicitement avant l'enregistrement d'instantanés de la mémoire. En d'autres termes, un clic sur le bouton Prendre un instantané de la mémoire équivaut à un clic sur le bouton Lancer la récupération de place puis à un clic sur le bouton Prendre un instantané de la mémoire.

Exécution de la récupération de place pendant le profilage d'une application

❖ Sélectionnez l'application dans la vue Profiler, puis cliquez sur le bouton Lancer la récupération de place.

Vous pouvez analyser l'efficacité de ce processus en comparant deux instantanés de mémoire, avant et après le nettoyage de la mémoire.

Identification des zones à problèmes

Le profileur propose différentes techniques d'identification des zones à problèmes des applications.

Localisation des fuites de mémoire

L'un des problèmes les plus courants auxquels les développeurs d'applications sont confrontés est celui des fuites de mémoire. Celles-ci prennent souvent la forme d'objets qui ont été créés dans un laps de temps donné mais qui n'ont pas été récupérés.

L'une des façons de les identifier consiste à observer le nombre de références à un objet dans la table Occurrences de la vue Références de l'objet. Vous pouvez généralement ignorer les références provenant de propriétés et de liaisons d'un document et rechercher des références inattendues ou inhabituelles, plus particulièrement des objets qui ne sont pas des enfants de l'objet. Pour plus d'informations, voir « Vue Références de l'objet » à la page 140.

Vous pouvez également examiner les chemins menant aux occurrences d'un objet pour déterminer si un chemin possède une référence arrière à la récupération de place (racine de la RP). Une occurrence dont la libération était attendue mais qui possède des références à la racine de la RP indique une perte de mémoire. Dans ce cas, modifiez le code de l'application de manière à supprimer toute référence à la racine de la RP. Une occurrence ne possédant plus aucune référence à la racine de la RP est prête à faire l'objet d'une récupération de place. Flash Player libère finalement cette mémoire.

Une autre façon de localiser les pertes de mémoire consiste à comparer deux instantanés de mémoire dans la vue Objets vagabonds, afin de déterminer quels objets se trouvent toujours dans la mémoire après une série particulière d'événements.

Il est courant, pour nettoyer les liens de la mémoire, de faire appel aux méthodes `disconnect()`, `clearInterval()` et `removeEventListener()`.

Recherche d'une référence arrière à la racine de la RP pour une occurrence

1 Créez un instantané de la mémoire.

Voir « Création et affichage d'un instantané de la mémoire » à la page 139.

2 Spécifiez le nombre de chemins de référence arrière à rechercher.

Dans la vue Instantané de la mémoire, indiquez le nombre maximal de chemins à rechercher ou sélectionnez Afficher tous les chemins de références arrière.

3 Cliquez deux fois sur une classe dans la vue Instantané de la mémoire pour ouvrir la vue Références de l'objet.

4 Développez les chemins répertoriés et regardez s'il existe une référence arrière à la racine de la RP.

Rechercher les objets vagabonds

1 Créez deux instantanés de la mémoire.

Voir « Création et affichage d'un instantané de la mémoire » à la page 139.

2 Sélectionnez les deux instantanés de mémoire à comparer.

Remarque : *si vous avez plus de deux instantanés de la mémoire, vous ne pouvez pas en sélectionner un troisième. Vous pouvez seulement comparer deux instantanés de mémoire en même temps.*

3 Cliquez sur le bouton Rechercher les objets vagabonds.

La vue Objets vagabonds matérialise d'éventuelles fuites de mémoire. Les colonnes Occurrences et Mémoire indiquent les différences entre le nombre d'occurrences d'une classe et la quantité de mémoire consommée par ces occurrences au cours de l'intervalle entre deux instantanés. Si vous voyez une classe que vous ne vous attendiez pas à voir créée pendant ce laps de temps ou une classe que vous pensiez voir détruite pendant ce laps de temps, examinez l'application pour détecter si ces éléments peuvent être à l'origine d'une fuite de mémoire.

4 Pour déterminer la façon dont un objet de la vue Rechercher les objets vagabonds a été instancié, cliquez deux fois sur l'objet dans la vue. La vue Références de l'objet présente la trace de pile de chaque occurrence que vous avez sélectionnée.

Un moyen d'identifier une fuite de mémoire consiste dans un premier temps à déterminer un ensemble d'étapes que vous pouvez répéter de nombreuses fois avec l'application, là où l'utilisation de la mémoire ne cesse d'augmenter. Il est important d'effectuer ces étapes au moins une fois dans l'application avant de prendre le premier instantané de la mémoire, de manière que tous les objets mis en mémoire cache ou que toute autre occurrence soient inclus dans cet instantané.

Répétez ces étapes un certain nombre de fois (3, 7 ou un autre nombre premier) dans l'application. Prenez un second instantané de mémoire pour le comparer à l'instantané de départ. Dans la vue Rechercher les objets vagabonds, vous trouverez peut-être des objets vagabonds dotés d'occurrences au nombre de multiples de 3 ou de 7. Ces objets présentent probablement des fuites. Cliquez deux fois sur les classes pour consulter les traces de pile de chacune des occurrences.

Un autre moyen consiste à répéter la séquence d'étapes sur une longue période et d'attendre que l'utilisation de la mémoire atteigne son maximum. Si elle n'augmente pas après cela, il n'y a aucune fuite de mémoire pour cet ensemble d'étapes.

Parmi d'autres causes, les écouteurs d'événement en attente sont souvent à l'origine de fuites de mémoire. La méthode `removeEventListener()` permet de supprimer un écouteur d'événement qui n'est plus utilisé. Pour plus d'informations, voir Création et destruction d'objets dans Utilisation de Flex 4.6.

Analyse des temps d'exécution

En analysant les temps d'exécution des méthodes et la quantité de mémoire allouée pendant ces appels de méthode, vous pouvez déterminer à quels niveaux les ralentissements des performances surviennent.

Cette opération est utile notamment pour identifier le temps d'exécution des méthodes appelées fréquemment, plutôt que celui des méthodes qui le sont rarement.

Détermination de la fréquence des appels de méthodes

1. Démarrez une session de profilage et veillez à activer le profilage des performances lors de la configuration du profileur sur l'écran de démarrage.
2. Sélectionnez l'application dans la vue Profiler.
3. Interagissez avec l'application jusqu'à atteindre le point où vous voulez commencer à analyser le nombre d'appels de méthode. Pour voir le nombre de fois qu'une méthode a été appelée lors du démarrage de l'application, n'interagissez pas avec l'application.
4. Cliquez sur le bouton Réinitialiser les données de performance. Toutes les données de performance sont alors effacées, de manière telle à ce que le profil de performance suivant n'inclue que les données saisies à compter de la réinitialisation.
5. Interagissez avec l'application jusqu'à atteindre le point où vous voulez vérifier le nombre d'appels de méthode depuis la réinitialisation des données.
6. Cliquez sur le bouton Capturer le profil de performance.
7. Cliquez deux fois sur le profil de performance dans la vue Profiler.
8. Dans la vue Profil de performance, triez les données selon la colonne Méthode et localisez votre méthode dans la liste.

   La valeur figurant dans la colonne Appels correspond au nombre de fois que cette méthode a été appelée au cours de cet intervalle d'échantillonnage. Il s'agit de l'écart de temps qui sépare le moment où vous avez cliqué sur Réinitialiser les données de performance et le moment où vous avez cliqué sur Capturer le profil de performance.

Examinez les valeurs contenues dans les colonnes Durée cumulée, Durée propre, Durée cumulée moyenne et Durée propre moyenne de la vue Profil de performance. Elles indiquent le temps d'exécution des méthodes.

Comparez le temps mis par chaque méthode pour s'exécuter au temps que mettent à s'exécuter toutes les méthodes appelées par une méthode particulière. En règle générale, si la durée propre ou la durée propre moyenne d'une méthode est élevée ou élevée par rapport à celle d'autres méthodes, observez de plus près la façon dont la méthode est implémentée et tentez de réduire le temps d'exécution.

De la même façon, si la durée propre ou la durée propre moyenne d'une méthode est faible, mais que la durée cumulée ou la durée cumulée moyenne est élevé, observez les méthodes que cette méthode appelle pour localiser les goulets d'étranglement.

Localisation d'une allocation excessive d'objets

Un moyen d'identifier les zones à problèmes d'une application consiste à déterminer à quels niveaux un nombre excessif d'objets est créé. La création d'une occurrence d'un objet peut consommer un grand nombre de ressources, plus particulièrement si cet objet se trouve dans la liste d'affichage. L'ajout d'un objet à la liste d'affichage peut engendrer un grand nombre d'appels à des méthodes de style et de mise en forme, ce qui peut ralentir une application. Dans certains cas, vous pouvez restructurer le code afin de réduire le nombre d'objets créés.

Une fois que vous avez déterminé si la création de certains objets est superflue, vous pouvez décider s'il est raisonnable ou utile de réduire le nombre d'occurrences de cette classe. Vous pouvez par exemple déterminer la taille des objets, car les objets volumineux présentent généralement le plus grand potentiel d'optimisation.

Pour savoir quels objets sont créés en nombre important, vous pouvez comparer les instantanés de mémoire de l'application à deux moments.

Affichage du nombre d'occurrences d'une classe spécifique

1. Démarrez une session de profilage et veillez à activer le profilage de la mémoire lors de la configuration du profileur sur l'écran de démarrage.
2. Interagissez avec l'application jusqu'à atteindre l'endroit dont vous souhaitez prendre un instantané de la mémoire.
3. Cliquez sur le bouton Prendre un instantané de la mémoire. Le profileur ajoute un nouvel instantané de la mémoire à la liste d'applications dans la vue Profiler.
4. Dans la vue Profiler, cliquez deux fois sur l'instantané de la mémoire pour l'ouvrir.
5. Pour afficher le nombre d'occurrences d'une classe particulière, ainsi que la quantité de mémoire que ces occurrences utilisent, effectuez un tri selon la colonne Classe et localisez la classe dans cette colonne. Vous pouvez aussi effectuer un tri selon les autres colonnes afin d'identifier rapidement les objets qui occupent le plus de mémoire ou les objets comptant le plus grand nombre d'occurrences. Dans la plupart des cas, la classe Strings est celle qui compte le plus d'occurrences et qui utilise le plus de mémoire.

Pour plus d'informations sur la vue Instantané de la mémoire, voir « Vue Instantané de la mémoire » à la page 138.

Localisation d'occurrences d'allocation excessive d'objets

1. Démarrez une session de profilage et veillez à activer le profilage de la mémoire lors de la configuration du profileur sur l'écran de démarrage.
2. Interagissez avec l'application jusqu'à atteindre le premier endroit dont vous souhaitez prendre un instantané de la mémoire.
3. Cliquez sur le bouton Prendre un instantané de la mémoire.

   Le profileur enregistre l'instantané de la mémoire dans la vue Profiler en indiquant la date et l'heure.
4. Interagissez avec l'application jusqu'à atteindre le deuxième endroit dont vous souhaitez prendre un instantané de la mémoire.
5. Cliquez à nouveau sur le bouton Prendre un instantané de la mémoire.

   Le profileur enregistre le deuxième instantané de la mémoire dans la vue Profiler en indiquant la date et l'heure.
6. Sélectionnez les deux instantanés de mémoire à comparer.

   Remarque : *si vous avez plus de deux instantanés de la mémoire, vous ne pouvez pas en sélectionner un troisième. Vous ne pouvez comparer que deux instantanés de mémoire à la fois.*
7. Cliquez sur le bouton Afficher le suivi des allocations.

   La vue Suivi des allocations indique les méthodes qui ont été appelées entre les deux instantanés et la quantité de mémoire consommée lors de ces appels de méthode. Voir « Vue Suivi des allocations » à la page 142.

Utilisation d'Adobe Scout avec Flash Builder

Adobe® Scout (nom de code précédent : « Project Monocle ») est un outil de profilage de la mémoire permettant de profiler les applications ActionScript (Flash Player sur le bureau), ainsi que les applications mobiles exécutées sur Adobe AIR.

Adobe Scout utilise la fonction de télémétrie pour vous aider à profiler votre contenu Flash. Adobe Scout fournit de nombreuses options de télémétrie avancées que vous pouvez utiliser lors du profilage de votre application. Par exemple, la fonction de profilage à distance dans Adobe Scout vous permet d'exécuter le contenu à profiler sur un ordinateur ou un périphérique différent de l'ordinateur exécutant Adobe Scout.

Pour en savoir plus sur Adobe Scout et la fonction de télémétrie, reportez-vous à l'article www.adobe.com/go/learn_telemetry_fr.

Différences entre Adobe Scout et le profileur Flash Builder

Pour utiliser le profileur Flash Builder, vous devez disposer d'une version de débogage de Flash Player. Le profileur Flash Builder vous permet d'identifier les goulots d'étranglement des performances et les fuites de mémoire au niveau ActionScript dans la version de débogage de Flash player.

Adobe Scout est basé sur la fonction de télémétrie, qui s'exécute sur la version validée de Flash Player. La télémétrie fonctionne dans l'environnement d'exécution Flash, au-delà du niveau ActionScript, et envoie les données à Adobe Scout. Adobe Scout analyse ensuite les données et les affiche de manière claire et concise.

La version de débogage de Flash Player n'est pas nécessaire pour utiliser Adobe Scout ; vous pouvez profiler votre contenu même dans une version validée de l'application. Cette fonctionnalité est particulièrement utile pour détecter les fuites de mémoire ou affiner les performances d'une application validée.

Configuration d'Adobe Scout avec Flash Builder

Avant de configurer Adobe Scout avec Flash Builder, assurez-vous d'avoir installé les éléments suivants :

- Flash Player version 11.3 ou ultérieure (autonome ou plug-in) et Adobe AIR version 3.3 ou ultérieure
- Adobe Scout

Pour activer la télémétrie dans Flash Player et Adobe AIR, vous devez modifier le fichier de configuration de Flash Player (telemetry.cfg).

Pour plus d'informations sur l'installation d'Adobe Scout et la modification du fichier de configuration, reportez-vous à l'article www.adobe.com/go/learn_telemetry_fr.

Activation de la télémétrie avancée dans Flash Builder

Important : *Les options de télémétrie avancée sont uniquement disponibles pour les projets ActionScript et ActionScript Mobile qui utilisent ActionScript Compiler 2.0 (ASC 2.0).*

Vous pouvez activer la télémétrie avancée lors du débogage ou du test de votre application.

Pour utiliser les fonctionnalités de télémétrie avancées dans Flash Builder, procédez comme suit :

1. Dans la boîte de dialogue Propriétés du projet de votre projet ActionScript, sélectionnez Compilateur ActionScript.
2. Sous Options de télémétrie, sélectionnez Activer la télémétrie détaillée.

Lorsque vous sélectionnez cette option, Flash Builder ajoute l'argument `-advanced-telemetry` au compilateur et met à jour les paramètres du compilateur.

Protection des données de télémétrie avancée pour une application validée

Vous pouvez protéger vos données de télémétrie avancée à l'aide d'un mot de passe, si nécessaire. Lorsque vous exportez une version validée de votre application, vous pouvez spécifier un mot de passe afin de garantir l'accès authentifié à vos données de télémétrie.

Suivez la procédure suivante pour spécifier un mot de passe :

1 Dans l'assistant d'exportation vers une version validée, assurez-vous que l'option Activer la télémétrie détaillée est sélectionnée.

2 Saisissez un mot de passe, puis confirmez-le.

Remarque : *Le mot de passe est uniquement applicable pour la session actuelle. Si vous modifiez le mot de passe, vous devrez recompiler le fichier SWF.*

Désactivation de la valeur -sampler pour les applications iOS

L'échantillonnage est une fonctionnalité importante de la télémétrie avancée. Une fois que vous avez activé la télémétrie avancée pour votre projet, la valeur de paramètre `-sampler` est définie par défaut.

Pour les applications iOS, l'activation de l'option `-sampler` peut entraîner des problèmes de performance. Si vous ne souhaitez pas vous connecter à Adobe Scout pour afficher les données de télémétrie, Flash Builder vous permet de désactiver la valeur `-sampler`.

Pour désactiver la valeur `-sampler` dans Flash Builder, procédez comme suit :

1 Cliquez sur Personnaliser le lancement dans la boîte de dialogue Configurations Exécuter/Déboguer ou dans la boîte de dialogue Exporter vers une version validée.

2 Dans la boîte de dialogue Personnaliser les paramètres de lancement, cliquez sur la croix (X) en regard du nom du paramètre `-sampler`.

3 Cliquez sur OK. Sous Paramètres modifiés, la valeur du paramètre `-sample` apparaît comme `<value not set>`.

Chapitre 7 : Outils de test unitaire dans Flash Builder

Environnement de test FlexUnit

L'environnement de test FlexUnit permet de générer et de modifier des tests reproductibles pouvant être exécutés à partir de scripts ou directement dans Flash Builder. Flash Builder prend en charge les structures « open source » FlexUnit 4 et FlexUnit 1.

Procédez comme suit :

- Créez des scripts et des suites de tests unitaires.

  Les assistants de Flash Builder vous guident tout au long de la création de scripts et de suites de tests en générant du code stub pour les tests.

- Exécutez les scripts et les suites de tests.

  Vous pouvez exécuter des scripts et des suites de tests de différentes manières à partir de Flash Builder ou en dehors de l'environnement Flash Builder. Les résultats des tests sont affichés dans une application de test. Flash Builder ouvre la vue Résultat des tests unitaires FlexUnit permettant d'analyser l'exécution du test.

- A partir de la vue Résultats des tests unitaires FlexUnit, accédez au code source.

  Dans le panneau Résultats des tests, cliquez deux fois sur un test pour en ouvrir l'implémentation.

  Le panneau Détails des tests ayant échoué répertorie la source et le numéro de la ligne correspondant à l'échec. S la source répertoriée se trouve dans l'espace de travail en cours d'utilisation, cliquez deux fois sur son nom pour accéder directement à l'échec.

Pour comprendre les principes de base de Test Driven Development (TDD) et utiliser l'environnement de test de FlexUnit, voir l'article publié dans l'Adobe Dev Center par Elad Elrom, professionnel de la communauté Adobe.

Création de tests FlexUnit

Vous pouvez créer des classes de scripts de tests et des suites de scripts de tests unitaires FlexUnit pour les types de projets suivants :

- Projet Flex
- Projet Flex Mobile
- Projet de bibliothèque Flex
- Projet ActionScript
- Projet Adobe AIR

Lorsque vous créez une classe de scripts de tests, vous pouvez spécifier les options suivantes :

- Le dossier `src` d'un projet Flex pour la classe
- Un package pour la classe

- Les classes à tester
- Les méthodes à tester pour chaque classe spécifiée

Une suite de scripts de tests unitaires FlexUnit est une série de tests basés sur des classes de scripts déjà créées, des méthodes spécifiées dans ces classes et d'autres suites de scripts de tests.

Création d'une classe de scripts de tests unitaires FlexUnit

Lors de la création d'une classe de scripts de tests unitaires FlexUnit, Flash Builder génère un fichier ActionScript pour la classe de scripts de tests et le place dans un package de scripts de tests.

La procédure suivante part du principe que vous avez créé un projet Flash Builder, dans lequel vous souhaitez générer et exécuter des tests unitaires FlexUnit.

1 Sélectionnez le projet Flex, puis Nouveau > Classe de script de tests dans le menu contextuel.

Si le fichier du projet sélectionné correspond à une classe ActionScript, cette classe sera automatiquement sélectionnée pour le script de test FlexUnit dans l'assistant Nouvelle classe de script de tests.

2 Dans l'assistant de création d'une classe de script de tests, indiquez si vous souhaitez créer une classe de style FlexUnit 4 ou FlexUnit 1.

3 Attribuez-lui un nom.

4 (Facultatif) Spécifiez un dossier source et un package pour la classe de script de tests ou validez les paramètres par défaut.

Le dossier source par défaut est le dossier `src` du projet en cours. Le package par défaut est `flexUnitTests`, qui correspond au niveau supérieur de la structure de packages par défaut du projet.

5 (Facultatif) Activez la case Sélectionner la classe à tester et recherchez la classe voulue. Cliquez sur Suivant.

6 (Facultatif) Sélectionnez les méthodes que vous souhaitez tester.

7 Cliquez sur Terminer.

Codez le script de test créé en vous basant sur les stubs de code générés.

Création d'une suite de scripts de tests unitaires FlexUnit

Cette procédure part du principe que vous avez déjà créé des classes de script de tests.

1 Sélectionnez le projet Flex, puis créez une suite de script de tests à partir des menus contextuels en sélectionnant Nouveau > Classe de suite de tests.

2 Dans l'assistant de création d'une classe de suite de tests, indiquez si vous souhaitez créer une classe de style FlexUnit 4 ou FlexUnit 1.

3 Nommez la suite de tests.

4 Recherchez ensuite, dans les suites et les scripts de tests, les classes et les méthodes à inclure dans la suite de tests. Cliquez sur Terminer.

Personnalisation des classes de script et de suite de tests FlexUnit par défaut

Vous pouvez personnaliser les classes et les scripts de test FlexUnit créés par Flash Builder. Flash Builder fait appel à des modèles de fichiers pour créer les versions par défaut de ces fichiers.

Les modèles de fichiers pour les tests FlexUnit se trouvent dans la boîte de dialogue Préférences, sous Flash Builder > Modèles de fichier > FlexUnit. Les classes de suite et de script de tests FlexUnit 1 et FlexUnit 4 correspondent à des modèles distincts.

Pour plus d'informations sur la modification des modèles de fichiers par défaut, voir « Personnalisation des modèles de fichier » à la page 48.

Remarque : *`FlexUnitCompilerApplication.mxml` et `FlexUnitApplication.mxml` dérivent du modèle pour applications MXML Web ou d'ordinateur. Le modèle utilisé dépend de la configuration du projet Flex : pour une application Web (s'exécute dans Adobe® Flash® Player) ou une application de bureau (s'exécute dans Adobe AIR).*

Voir aussi

Référence de langage « open source » pour FlexUnit

Documentation « open source » pour FlexUnit

Exécution des tests FlexUnit

Les tests unitaires FlexUnit peuvent s'exécuter à l'intérieur comme à l'extérieur de Flash Builder à l'aide des fichiers SWF générés pour le test unitaire FlexUnit. Dans les deux cas, les résultats des tests s'affichent dans la vue Résultat des tests unitaires FlexUnit.

Vous pouvez également configurer et enregistrer un test unitaire FlexUnit avant de l'exécuter.

Par défaut, les tests FlexUnit sont exécutés dans la perspective Débogage Flash. Vous pouvez les lancer à partir des perspectives de développement et de profilage Flash. Dans ce cas, Flash Builder basculera toutefois vers la perspective de débogage à l'exécution du test.

Vous pouvez modifier la perspective par défaut des tests FlexUnit. Ouvrez la fenêtre des préférences et accédez à Flash Builder > FlexUnit.

Application de compilation FlexUnit et application FlexUnit

Lorsque vous créez un script de test FlexUnit, Flash Builder crée l'application de compilation FlexUnit suivante et une application FlexUnit :

- FlexUnitCompilerApplication.mxml
- FlexUnitApplication.mxml

Flash Builder utilise ces applications lors de la compilation et de l'exécution des tests FlexUnit. Flash Builder place les applications dans le répertoire `src` du projet.

Cette application contient des références à tous les scripts et à toutes les suites de tests FlexUnit générés par Flash Builder. Ce dernier place tous les tests FlexUnit à l'intérieur des balises `<fx:Declarations>` de cette application. Vous ne serez généralement pas amené à éditer ou à modifier ce fichier directement.

Actualisez l'application de compilation FlexUnit dans les circonstances suivantes :

- Vous ajoutez manuellement un script de tests.

  Si vous créez une classe de script de tests sans utiliser l'assistant de création de script de tests, actualisez `FlexUnitCompilerApplication.mxml`. Placez le nouveau script de tests dans le package avec les autres scripts.

- Vous renommez un script de tests.

- Vous supprimez un script de tests.

Actualisez `FlexUnitCompilerApplication.mxml` :

1 Si cette vue n'est pas ouverte, vous pouvez y accéder en sélectionnant Fenêtre > Affichage d'une vue > Autres > Flash Builder > Résultat des tests unitaires FlexUnit. Cliquez sur OK.

2 Dans la vue Résultat des tests unitaires FlexUnit, cliquez sur le bouton Actualiser.

Exécution d'un test unitaire FlexUnit dans Flash Builder

Vous pouvez exécuter des tests FlexUnit pour un projet dans son ensemble ou pour des scripts de tests individuels.

Vous lancerez généralement l'exécution à partir du menu contextuel d'un projet ou d'un script de test individuel.

Vous pouvez toutefois également lancer les tests en accédant au menu Exécuter, en cliquant sur le bouton Exécuter ou en sélectionnant le bouton Exécuter des tests unitaires FlexUnit dans la vue Résultat des tests unitaires FlexUnit.

L'exécution à partir du menu Exécuter de Flash Builder entraîne l'ouverture d'une boîte de dialogue de configuration du test, dans laquelle vous pouvez sélectionner les classes de test et les méthodes à exécuter. Les scripts de tests pour les projets de bibliothèque ne peuvent pas être exécutés à partir du menu Exécuter de Flash Builder.

Flash Builder fournit les raccourcis clavier suivants pour faciliter le lancement des tests FlexUnit :

- `Alt+Maj+A,F`

  Exécute tous les tests FlexUnit du projet.

- `Alt+Maj+E,F`

  Exécute le test FlexUnit sélectionné.

Exécutez les tests FlexUnit de la sélection actuelle dans l'éditeur Voir « Configuration des tests FlexUnit » à la page 168.

1 Sélectionnez un projet et exécutez les tests :

  Dans le menu contextuel du projet, sélectionnez Exécuter des tests unitaires FlexUnit.

  Dans le menu Exécuter ou dans la liste déroulante du bouton Exécuter de Flash Builder, sélectionnez Exécuter > Tests FlexUnit.

2 (Menu Exécuter de Flash Builder) Dans la boîte de dialogue Exécution d'une configuration de test unitaire FlexUnit, sélectionnez les scripts de tests et les méthodes à exécuter dans le cadre des tests. Cliquez sur OK pour exécuter les tests.

3 Visionnez les résultats des tests.

  Flash Builder génère un fichier SWF dans le dossier bin-debug du projet.

  Une application s'ouvre pour afficher des informations sur les tests et en indiquer la fin.

  Le panneau Résultat des tests unitaires FlexUnit s'ouvre et affiche les résultats des tests. Voir « Affichage des résultats de l'exécution d'un test FlexUnit » à la page 169.

Exécution de tests FlexUnit individuels :

1 Dans l'Explorateur de projets, accédez au package `flexUnitTest` :

  Dans le menu contextuel du fichier de test FlexUnit, sélectionnez Exécuter des tests unitaires FlexUnit.

2 Visionnez les résultats des tests.

  Flash Builder génère un fichier SWF dans le dossier bin-debug du projet.

Une application s'ouvre pour afficher des informations sur les tests et en indiquer la fin.

Le panneau Résultat des tests unitaires FlexUnit s'ouvre et affiche les résultats des tests. Voir « Affichage des résultats de l'exécution d'un test FlexUnit » à la page 169.

Exécution d'un test unitaire FlexUnit en dehors de l'environnement Flash Builder

Cette procédure part du principe que vous avez déjà exécuté un test unitaire FlexUnit dans Flash Builder et que Flash Builder est en cours d'exécution.

1 Copiez le fichier SWF généré pour un test du dossier bin-debug du projet vers un dossier situé en dehors de votre environnement de développement.

Vous pouvez copier le fichier SWF généré automatiquement ou un fichier SWF provenant d'un test unitaire FlexUnit que vous avez précédemment enregistré et configuré.

2 Exécutez la copie du fichier SWF.

Flash Player apparaît pour afficher des informations sur le test et indiquer la fin du test.

La vue Résultat des tests unitaires FlexUnit s'ouvre dans Flash Builder et affiche les résultats du test.

Configuration des tests FlexUnit

1 Ouvrez la boîte de dialogue Exécution d'une configuration de test unitaire FlexUnit.

Vous pouvez y accéder à partir du menu Exécuter ou à partir de la vue Résultat des tests unitaires FlexUnit :

- Sélectionnez un projet. Dans le menu Flash Builder, sélectionnez Exécuter > Exécuter des tests unitaires FlexUnit.
- Dans la vue Résultat des tests unitaires FlexUnit, cliquez sur le bouton Exécuter des tests unitaires FlexUnit.

  Si cette vue n'est pas ouverte, vous pouvez y accéder en sélectionnant Fenêtre > Affichage d'une vue > Autres > Flash Builder > Résultat des tests unitaires FlexUnit.

2 Dans la boîte de dialogue Configuration de test, sélectionnez les scripts et les méthodes à enregistrer comme configuration de test.

Remarque : *la boîte de dialogue Configuration de test n'est pas disponible si vous exécutez un test à partir du menu contextuel de l'Explorateur de packages.*

3 (Facultatif) Sélectionnez Charger pour importer une configuration précédemment sauvegardée dans un fichier XML.

4 Cliquez sur Enregistrer.

Flash Builder enregistre un fichier XML et un fichier MXML dans le dossier `.FlexUnitSettings` du projet.

Vous pouvez utiliser le fichier XML dans les scripts de génération pour exécuter le test.

Vous pouvez générer un fichier SWF à partir du fichier MXML. Ce fichier SWF peut servir à effectuer des tests en dehors de l'environnement Flash Builder. Pour générer le fichier SWF, le fichier MXML est généralement copié dans le dossier `src` du projet.

Affichage des résultats de l'exécution d'un test FlexUnit

Le panneau de la vue Résultat des tests unitaires FlexUnit affiche les résultats d'un test unitaire FlexUnit, en détaillant les échecs. Vous pouvez parcourir les résultats, filtrer l'affichage, écrire les résultats dans un fichier et charger les résultats à partir d'un fichier.

Vous pouvez également relancer les tests, annuler un test en cours d'exécution et effacer les résultats de la vue.

Si cette vue n'est pas ouverte, vous pouvez y accéder en sélectionnant Fenêtre > Affichage d'une vue > Autres > Flash Builder > Résultat des tests unitaires FlexUnit.

Panneau Résultats des tests

Ce panneau répertorie tous les tests de la série, en indiquant si le test a échoué ou réussi.

Cliquez deux fois sur un test de la liste pour y accéder dans l'éditeur ActionScript.

Panneau Détails des tests ayant échoué

Sélectionnez un test dans le panneau Résultats des tests pour afficher les détails de l'échec.

Chaque détail indique le fichier source et la méthode, y compris le numéro de la ligne correspondant à l'échec.

Si le fichier source appartient à l'espace de travail, cliquez deux fois dans la liste pour accéder à l'échec dans l'éditeur ActionScript.

Menu de la vue Résultats des tests unitaires FlexUnit

Dans le menu de la vue Résultats des tests unitaires FlexUnit, vous pouvez procéder comme suit :

- Filtrez les résultats affichés.

  Masquage du panneau Détails des tests ayant échoué.

  Limitation de l'affichage aux tests ayant échoué
- Défilement des tests affichés dans le panneau Résultat des tests
- Annulation d'un test en cours d'exécution
- Enregistrement des résultats ou de la configuration d'un test
- Chargement des résultats précédemment enregistrés dans un fichier
- Suppression des résultats du panneau
- Réexécution du test actuel. Vous pouvez choisir :
  - d'exécuter tous les tests ;.
  - d'exécuter uniquement les échecs ;
  - d'exécuter le test sélectionné.
- Actualisation de la configuration FlexUnit

  Si vous avez modifié un test ou avez ajouté ou supprimé des tests, cliquez sur le bouton d'actualisation pour charger la nouvelle configuration FlexUnit.
- Configuration et exécution de tests FlexUnit

  Utilisez le bouton Exécuter des tests unitaires FlexUnit pour configurer et exécuter des tests FlexUnit.

Prise en charge de FlexUnit pour les projets mobiles

Il vous est maintenant possible d'exécuter un test FlexUnit sur un périphérique mobile. Dans le cas où vous ne disposez pas de périphérique mobile, vous pouvez en simuler un sur votre ordinateur et ensuite exécuter le test FlexUnit.

Exécution d'un test FlexUnit sur un périphérique mobile

1 Sélectionnez un projet mobile et exécutez le test en sélectionnant Exécuter > Tests FlexUnit ou sélectionnez Exécuter des tests unitaires FlexUnit dans le menu contextuel.

2 Spécifiez une méthode de lancement :

- **Sur le bureau** Exécute l'application sur votre ordinateur à l'aide de AIR Debug Launcher (ADL), selon une configuration de périphérique que vous avez spécifiée.
- **Sur le périphérique** exécute l'application FlexUnit sur un périphérique mobile.

  Flash Builder peut accéder au périphérique connecté au port USB de l'ordinateur ou au réseau via une connexion Wi-Fi. Une connexion socket et l'adresse IP de l'ordinateur permettent la communication entre Flash Builder et l'application FlexUnit exécutée sur le périphérique.

  Remarque : *le test FlexUnit peut être exécuté sur un périphérique connecté au port USB de votre ordinateur, mais les résultats du test pourront uniquement être affichés si la connexion Wi-Fi est activée sur le périphérique.*

  Le numéro de port utilisé par défaut pour établir une connexion socket est 8765. Vous pouvez modifier ce numéro en accédant à la boîte de dialogue Préférences et en sélectionnant Flash Builder > FlexUnit.

  Si l'application FlexUnit ne parvient pas à établir une connexion avec le périphérique, Flash Builder affiche parfois une boîte de dialogue vous demandant d'indiquer l'adresse IP de l'ordinateur. Assurez-vous que le périphérique est correctement connecté au réseau sans fil et que l'ordinateur qui exécute Flash Builder est accessible depuis ce réseau.

3 Dans la fenêtre de configuration de l'option Exécuter les tests unitaires FlexUnit, sélectionnez les scripts et les méthodes à exécuter au cours du test. Cliquez sur OK pour exécuter les tests.

4 Visionnez les résultats des tests.

  Flash Builder génère un fichier SWF dans le dossier bin-debug du projet.

  Une application s'ouvre pour afficher des informations sur les tests et en indiquer la fin.

  Le panneau Résultat des tests unitaires FlexUnit s'ouvre et affiche les résultats des tests. Voir « Affichage des résultats de l'exécution d'un test FlexUnit » à la page 169.

Chapitre 8 : Développement d'applications Web et d'ordinateur dans Flash Builder

Flash Builder fournit des outils permettant de développer des applications Web basées sur Flex, ou des applications de bureau Adobe AIR, et d'en créer des packages.

Vous sélectionnez le type d'application requis lors de la création du projet Flex :

Web : déploiement de l'application en tant que fichier SWF à utiliser dans Flash Player s'exécutant dans un navigateur.

De bureau : déploiement d'une application Adobe AIR autonome pour un ordinateur de bureau, tel qu'un ordinateur exécutant Windows ou un Macintosh.

Flux de travail de base permettant de développer une application de bureau ou basée sur navigateur

1 (Facultatif) Créez ou importez un projet.

Dans Flash Builder, sélectionnez Fichier > Nouveau > Projet Flex.

Sélectionnez le type d'application (Web ou de bureau).

Dans certains cas, votre projet accède aux données d'un serveur d'applications, tel qu'un serveur PHP, ColdFusion ou Java. Précisez les informations d'accès au serveur lors de la création du projet.

Pour plus d'informations, voir « Création de projets dans Flash Builder » à la page 61.

Vous pouvez importer des projets qui ont déjà été créés ou exportés depuis Flash Builder. Pour plus d'informations, voir « Exportation et importation de projets » à la page 81.

2 Créez ou modifiez les fichiers source d'un projet.

Utilisez les éditeurs Flash Builder pour modifier les fichiers source d'un projet.

Pour plus d'informations, voir « Outils de développement de code dans Flash Builder » à la page 16.

3 Créez le projet.

Par défaut, Flash Builder crée un projet à chaque fois que vous enregistrez un fichier dans le projet. Lorsque vous créez le projet, Flash Builder place les fichiers de sortie à l'emplacement approprié au projet. Flash Builder vous informe des erreurs ou avertissements rencontrés lors de la création du projet.

Pour plus d'informations, voir « Création de projets » à la page 85.

4 Exécutez les applications du projet.

Chaque projet, à l'exception des projets de bibliothèque, peut contenir des fichiers d'application susceptibles d'être lancés dans les perspectives d'exécution, de débogage ou de profil de Flash Builder. Flash Builder définit une perspective de lancement qui lui permet de lancer l'application. Vous pouvez modifier la perspective de lancement par défaut ou créer d'autres perspectives de lancement.

Pour plus d'informations, voir « Exécution et débogage des applications » à la page 105.

5 Testez le projet.

Utilisez les outils de débogage, de profilage et d'analyse de Flash Builder pour tester et optimiser vos projets. Vous pouvez également utiliser l'environnement de test FlexUnit pour générer et modifier des tests reproductibles pouvant être exécutés à partir de scripts ou directement dans Flash Builder.

Pour plus d'informations, voir « Débogage d'outils dans Flash Builder » à la page 121 et « Outils de test unitaire dans Flash Builder » à la page 164.

Si votre application accède aux données distantes, utilisez le moniteur de réseau pour examiner les données transmises entre une application et un service de données. Pour plus d'informations, voir « Surveillance des applications qui accèdent aux services de données » à la page 196.

Utilisez le profileur Adobe Flex pour identifier les problèmes de performance et les pertes de mémoire d'une application. Pour plus d'informations, voir « Profilage d'outils dans Flash Builder » à la page 133.

6 Déploiement d'une application.

Utilisez Flash Builder pour exporter une version validée d'une application, que vous pouvez utiliser pour distribuer et déployer l'application. Le processus d'exportation d'une version validée d'une application varie selon le type de projet.

Pour plus d'informations, voir « Exportation d'une application vers une version validée » à la page 112.

Création d'interfaces utilisateur

Vous pouvez créer votre interface utilisateur dans Flash Builder à l'aide de composants Flex. Vous pouvez également ajouter des propriétés et méthodes à des composants existants et créer des composants requis par votre application.

Par exemple, le formulaire Spark vous permet de créer des formulaires complexes. Le formulaire Spark est un conteneur hautement personnalisable qui prend en charge plusieurs conceptions de formulaire. Vous pouvez utiliser les composants graphiques de Flex pour créer quelques-uns des types de graphique les plus courants et en déterminer entièrement l'aspect.

Pour plus d'informations sur l'utilisation des composants Flex pour créer des interfaces utilisateur, voir Building the user interface (Création de l'interface utilisateur).

Utilisation de thèmes

Flash Builder fournit plusieurs thèmes permettant de personnaliser l'apparence de vos applications.

Il comporte des thèmes Spark et Halo. Le thème par défaut de Flex 4.*x* est Spark et celui de Flex 3 est Halo. Vous pouvez également importer d'autres thèmes ou créer les vôtres.

Pour plus d'informations sur la prise en charge des thèmes dans Flex, voir About themes (A propos des thèmes).

Spécification d'un thème

Spécifiez des thèmes par projet. Une fois que vous avez spécifié un thème pour un projet, toutes les applications du projet le partagent.

1 Dans le menu Flash Builder, sélectionnez Projet > Propriétés > Thème Flex pour ouvrir la boîte de dialogue Sélectionner le thème du projet.

2 Sélectionnez un thème et cliquez sur OK.

Importation de thèmes

Flash Builder vous permet d'importer des thèmes. Les fichiers des thèmes doivent se trouver dans un dossier et tous les fichiers requis du thème Flex doivent être présents.

Le nom du thème est déterminé par l'élément de nom du fichier `metadata.xml` situé dans le dossier des thèmes. Si l'élément de nom n'est pas précisé ou si `metadata.xml` est absent, le thème prend le nom du dossier des thèmes.

Les thèmes de Flash Builder peuvent prendre les formats suivants :

- Fichier de thème ZIP

  Extrayez le contenu du fichier ZIP avant d'importer le thème. Le contenu extrait doit englober tous les fichiers requis.

- Fichier CSS ou SWC pour un thème

  Le fichier CSS ou SWC doit se trouver dans un dossier contenant tous les fichiers requis pour un thème Flex. Lorsque vous importez un thème à l'aide de Flash Builder, vous sélectionnez le fichier CSS ou SWC correspondant à ce thème.

- Fichier MXP

  Adobe Extension Manager CS5 vous permet d'assembler des fichiers correspondant à des thèmes Flex dans un fichier MXP. Vous pouvez ensuite importer le thème dans Flash Builder par le biais d'Adobe Extension Manager.

  Pour plus d'informations sur l'assemblage de thèmes dans un fichier MXP, voir « Création d'un fichier d'extension (fichier MXP) pour un thème Flex » à la page 175.

Importation de thèmes Flex à l'aide de Flash Builder

1 Dans le menu Flash Builder, sélectionnez Projet > Propriétés > Thème Flex pour ouvrir la boîte de dialogue Sélectionner le thème du projet.

2 Sélectionnez Importer un thème, accédez au dossier contenant le thème à importer, sélectionnez le fichier CSS ou SWC, puis cliquez sur OK.

Importation de thèmes Flex assemblés dans un fichier MXP

1 Si vous ne l'avez pas encore fait, importez Adobe Flash® Builder™ 4.7 dans Adobe Extension Manager CS6 :

  Dans Adobe Extension Manager, sélectionnez Fichier > Importer produit.

2 Dans Adobe Extension Manager, sélectionnez Flash Builder 4.7.

3 Sélectionnez Fichier > Installer l'extension, accédez au fichier MXP correspondant au thème, puis cliquez sur Ouvrir.

  Une fois que vous avez accepté la licence, Adobe Extension Manager installe le thème dans Flash Builder. Il est désormais disponible dans la boîte de dialogue Sélectionner le thème du projet de Flash Builder.

  Remarque : *vous pouvez également cliquer deux fois sur le fichier MXP pour lancer Adobe Extension Manager, qui installe automatiquement le thème.*

Téléchargement de thèmes

Vous pouvez télécharger des thèmes que vous pourrez ensuite importer dans Flash Builder.

Téléchargement de thèmes Flex

1 Dans le menu Flash Builder, sélectionnez Projet > Propriétés > Thème Flex pour ouvrir la boîte de dialogue Sélectionner le thème du projet :

2 Sélectionnez Découvrir d'autres thèmes.

Flash Builder ouvre votre navigateur Web par défaut à une page contenant des thèmes à télécharger. Vous pouvez également accéder à tout autre site contenant des thèmes Flex téléchargeables.

3 Sélectionnez un thème Flex à télécharger.

Une fois le thème téléchargé, vous pouvez l'importer, comme décrit dans « Importation de thèmes » à la page 173.

Création de thèmes

Vous pouvez créer vos propres thèmes et les importer dans Flash Builder. Un thème Flex contient généralement les fichiers suivants :

- Des fichiers SWC, SWF, CSS, PNG et JPEG ainsi que d'autres fichiers composant le thème.

  Les fichiers composant le thème peuvent varier, mais ils doivent comprendre un fichier SWC ou CSS.

- Fichier `preview.jpg`

  Il s'agit du fichier image d'aperçu pour le thème. Si le dossier de thème ne contient pas `preview.jpg`, Flash Builder fait appel à une image d'aperçu par défaut pour le thème.

- Fichier `metadata.xml`

  Ce fichier contient des informations relatives au thème, notamment les versions du SDK avec lesquelles le thème est compatible. Si le dossier de thème ne contient pas ce fichier, Flash Builder en crée un lors de l'importation du thème.

Un thème est généralement compressé dans un fichier ZIP mais celui-ci doit être extrait avant l'importation du thème dans Flash Builder. Vous pouvez aussi assembler les fichiers de thèmes dans un fichier Adobe Extension Manager (MXP) et utiliser Adobe Extension Manager pour importer le thème dans Flash Builder.

Pour plus d'informations, voir About themes.

Fichier metadata.xml

Le tableau suivant répertorie les éléments pouvant être inclus dans le fichier `metadata.xml`.

Nom d'élément	Description
Name	Nom du thème tel qu'il apparaît dans Flash Builder. Lors de l'importation d'un thème avec Flash Builder, vous pouvez remplacer le nom spécifié dans le fichier `metadata.xml`.
Category	Auteur du thème. Catégorie sous laquelle le thème est affiché dans Flash Builder.
sdks	Précise les versions du Flex SDK avec lesquelles le thème est compatible. C'est un élément parent de minVersionInclusive et maxVersionExclusive. Si l'élément sdks est absent, le thème est valide pour tous les SDK.

Nom d'élément	Description
minVersionInclusive	Version la plus ancienne du SDK de Flex avec laquelle ce thème est compatible. Si cette information est absente, ce thème est compatible avec toutes les versions antérieures du SDK de Flex.
maxVersionExclusive	Version la plus récente du SDK de Flex avec laquelle ce thème est compatible. Si cette information est absente, ce thème est compatible avec toutes les versions récentes du SDK de Flex.
mainFile	Fichier de niveau supérieur pour l'implémentation du thème. Ce fichier peut référencer d'autres fichiers dans le thème. Par exemple, un fichier CSS pourrait référencer un fichier SWC ou SWF. L'argument -theme du compilateur référence le fichier spécifié.

L'exemple suivant illustre un fichier `metadata.xml` standard correspondant à un thème créé par la société ABC.

```
<theme>
    <name>WindowsLookAlike</name>
    <category>ABC</category>
        <sdks>
            <minVersionInclusive>2.0.1</minVersionInclusive>
            <maxVersionExclusive>4.0.0</maxVersionExclusive>
        </sdks>
    <mainFile>WindowsLookAlike.css</mainFile>
</theme>
```

Selon le fichier `metadata.xml`, le thème est compatible avec le SDK de Flex 2.0.1. Il est aussi compatible avec des SDK jusqu'à la version Flex 4.0.0 (non incluse). Lorsque ce thème est sélectionné, `WindowsLookAlike.css` est le fichier ajouté à l'argument de compilateur `-themes`.

Création d'un fichier d'extension (fichier MXP) pour un thème Flex

Adobe Extension Manager vous permet de créer un fichier d'extension (fichier MXP) pour un thème Flex. Vous pouvez ensuite importer le fichier MXP dans Flash Builder par le biais d'Adobe Extension Manager.

Placez tous vos fichiers de thème dans un dossier hiérarchique, puis créez un fichier d'installation d'extensions (fichier MXI). Adobe Extension Manager utilise ce fichier pour créer le fichier MXP. Pour plus d'informations sur le format d'un fichier MXI, voir le document Format du fichier d'extension.

Lors de la création du fichier MXI, vous devez préciser un chemin de destination pour chacun des fichiers du thème. Les chemins sont au format suivant :

```
$flexbuilder/<Theme Name>
```

- `$flexbuilder` est défini dans le fichier de configuration Flash Builder, `XManConfig.xml`. Adobe Extension Manager développe `$flexbuilder` selon cette définition. `XManConfig.xml` se trouve à l'emplacement suivant dans votre système de fichiers :

  `/<Install Dir>/Flash Builder 4/configuration/XManConfig.xml`

- *<Theme Name>* est le nom du dossier qui contient le thème Flex.

Création d'un fichier d'extension MXP pour un thème Flex

1. Placez tous les fichiers correspondant au thème, notamment le fichier MXI, dans un dossier hiérarchique.
2. Dans Adobe Extension Manager, sélectionnez Fichier > Empaqueter l'extension.
3. Accédez au fichier d'installation d'extension souhaité et sélectionnez-le.
4. Recherchez un emplacement pour le fichier de package et nommez-le avec l'extension .mxp.

   Vous pouvez ensuite tester le fichier d'extension en l'installant par le biais d'Adobe Extension Manager.

Ajout d'autres thèmes

Vous pouvez définir plusieurs fichiers de thèmes pour une application. En l'absence de chevauchement de styles, les deux thèmes sont appliqués dans leur intégralité. D'autres aspects doivent également être pris en considération lors de l'ajout de thèmes. L'un d'eux est l'ordre des fichiers de thèmes.

Pour ajouter des thèmes, utilisez le compilateur de ligne de commande `mxmlc` avec l'option de compilateur `theme` afin de spécifier le chemin d'accès aux fichiers de thèmes.

Using themes fournit des informations sur la définition des arguments du compilateur et sur l'ordre des fichiers de thèmes.

Utilisation de styles

Les styles modifient l'aspect de l'application par la définition de valeurs pour les paramètres visuels des composants. Vous pouvez définir des styles s'appliquant à tous les composants d'une application, à des composants individuels ou à des groupes de composants référencés par un sélecteur de style.

Vous pouvez par exemple définir les styles suivants :

- Texte

  Famille, taille, couleur et épaisseur de la police ainsi que d'autres paramètres d'affichage (gras, italique et souligné, par exemple).

- Bordure

  Epaisseur, couleur, couleur au survol, style (pleine, incrusté, sortant, aucune), rayon de l'angle et autres.

- Couleur

  Couleur de remplissage et alpha.

Remarque : *les styles disponibles varient en fonction des composants.*

Vous pouvez définir des propriétés de style de manière intraligne sur une balise MXML ou séparément à l'aide de code CSS. Le code CSS peut être placé à l'intérieur de balises `<fx:Style>` dans une application ou dans un fichier CSS distinct.

Lorsque vous appliquez des styles intralignes à des composants, vous pouvez convertir les styles des composants en règle CSS dans une feuille de style externe. Vous pouvez utiliser l'éditeur CSS pour modifier les fichiers CSS.

Application d'un style externe ou intégré

Vous pouvez intégrer des styles CSS dans un fichier d'application MXML ou référencer un fichier CSS externe. L'exemple suivant illustre du code CSS appliquant des styles à tous les composants Spark Button d'une application. Ce code génère également le sélecteur`myStyle` qui peut être appliqué à n'importe quel composant :

```
@namespace s "library://ns.adobe.com/flex/spark";
@namespace mx "library://ns.adobe.com/flex/mx";

<s:Button { fontSize: 16pt; color: Red } /* type selector */
.myStyle { color: Red } /* class selector */ />
```

Pour les styles appliqués à des composants (par exemple, `s:Button`), le sélecteur de composants doit spécifier un espace de nom. Dans cet exemple, `s:Button` définit un style qui est automatiquement appliqué à tous les composants Spark Button.

Utilisez la notation à points CSS pour créer un sélecteur pouvant être appliqué à n'importe quel composant. Dans cet exemple, `.myStyle` ne doit pas déclarer d'espace de noms et peut être appliqué à n'importe quel composant.

Flex définit un cadre précis pour la création et l'application de styles. Pour plus d'informations, voir Using styles in Flex.

Création de fichiers CSS

Utilisez l'assistant Nouveau fichier CSS pour créer des fichiers CSS pour un projet. Cet assistant crée un fichier vide que vous pouvez utiliser pour définir vos styles CSS.

Par défaut, Flash Builder ajoute les espaces de noms par défaut pour les styles Spark et MX.

Pour créer un fichier CSS vide :

1 Dans le menu Flash Builder, sélectionnez Fichier > Nouveau > Fichier CSS.

2 Spécifiez un dossier source.

Le dossier source peut se trouver dans le projet actuel ou dans un autre projet.

3 Spécifiez un package pour le fichier.

Sélectionnez-le dans les packages disponibles dans le projet. Si vous souhaitez placer le fichier dans un nouveau package, créez d'abord ce dernier. Sélectionnez Fichier > Nouveau package.

4 Spécifiez un nom pour le fichier.

5 Cliquez sur Terminer.

Flash Builder utilise des modèles qui définissent le contenu des nouveaux fichiers. Vous avez la possibilité de personnaliser ces modèles. Voir « Personnalisation des modèles de fichier » à la page 48.

Utilisation de l'éditeur CSS

Flash Builder comporte un éditeur CSS que vous pouvez utiliser pour créer et modifier des feuilles de style pour votre application. L'éditeur CSS est disponible uniquement en mode Source.

A la création d'une feuille de style, Flash Builder fournit les déclarations suivantes pour les espaces de noms Spark et MX :

```
@namespace s "library://ns.adobe.com/flex/spark";
@namespace mx "library://ns.adobe.com/flex/mx";
```

Certains composants Spark et MX partagent le même nom local. Prenons l'exemple d'un composant Spark Button (dans le package spark.components.*)et d'un composant MX Button (dans le package mx.controls.*). Pour les distinguer l'un de l'autre, vous pouvez spécifier des espaces de noms s'appliquant aux types dans le fichier CSS.

Si vous n'utilisez pas de sélecteurs de type dans vos feuilles de style, vous n'avez pas besoin de déclarer des espaces de noms. Pour plus d'informations (illustrées par des exemples), voir About namespaces in CSS.

Remarque : *les styles sont traités différemment pour les applications utilisant le SDK Flex 3. Si votre projet Flex repose sur le SDK Flex 3, l'éditeur CSS applique le comportement implémenté avec Flex Builder 3. Pour plus d'informations sur l'utilisation de l'éditeur CSS avec les applications reposant sur le SDK Flex 3, voir la documentation de Flex Builder 3.*

Ajout d'états d'affichage et de transitions

Flash Builder permet de créer des applications dont l'apparence varie selon les tâches exécutées par l'utilisateur. Par exemple, l'état de base de l'application peut représenter la page d'accueil et inclure un logo, une barre latérale et un message de bienvenue. Lorsque l'utilisateur clique sur un bouton de la barre latérale, l'apparence (l'*état*) de l'application change de façon dynamique et la zone de contenu principale est remplacée par un formulaire de bon de commande sans modifier la position du logo et de la barre latérale.

Flex vous permet d'appliquer cette interaction aux états d'affichage et aux transitions. Un *état d'affichage* représente l'un des différents affichages que vous définissez pour une application ou un composant personnalisé. Une *transition* consiste en un ou plusieurs effets regroupés pour être lancés lorsqu'un état d'affichage est modifié. Une transition vise à améliorer le changement visuel d'un état à un autre.

Prise en charge des états d'affichage Flex 3

Flash Builder prend en charge les états d'affichage implémentés dans Flex 3. Si vous créez un projet qui utilise le SDK Flex 3, l'éditeur MXML retourne à l'implémentation Flex Builder 3. Pour plus d'informations sur la modification des états pour le SDK Flex 3, voir la documentation de Flex Builder 3.

Création et modification d'états d'affichage

L'éditeur MXML de Flash Builder contient plusieurs fonctions permettant de modifier le code source des états d'affichage.

Les propriétés `includeIn` et `excludeFrom` des composants MXML spécifient l'état d'affichage ou le groupe d'états dans lequel un composant s'affiche. Les conseils de code de l'éditeur MXML facilitent la sélection d'un état d'affichage ou d'un groupe d'états pour ces propriétés.

Vous pouvez également utiliser la notation à points pour les attributs de composants. Elle vous permet de spécifier l'état d'affichage dans lequel l'attribut est appliqué. Pour afficher par exemple un composant Button dans deux états d'affichage et obtenir la modification de l'étiquette en fonction de l'état, associez l'opérateur point à la propriété `label`. Les conseils de code de l'éditeur MXML facilitent la sélection de l'état d'affichage. Par exemple :

```
<s:Button label.State1="Button in State 1" label.State2="Same Button in State 2">
```

Pour plus d'informations sur la création et la modification des états d'affichage, voir Create and apply view states (Création et application d'états d'affichage).

Changement d'état d'affichage au moment de l'exécution

Lors de l'exécution de l'application, les utilisateurs doivent passer d'un état d'affichage à un autre. Vous pouvez définir les gestionnaires d'événement pour les commandes utilisateur afin de permettre aux utilisateurs de changer d'état au moment de l'exécution.

La méthode la plus simple consiste à affecter la propriété `currentState` à l'événement de clic d'un contrôle tel qu'un bouton ou un lien. La propriété `currentState` sélectionne le nom de l'état d'affichage que vous souhaitez consulter lorsque l'événement de clic se produit. Dans le code, spécifiez la propriété `currentState` de la façon suivante :

```
click="currentState='viewstatename'"
```

Si l'état d'affichage est défini pour un composant spécifique, vous devez également préciser le nom du composant, comme illustré ci-dessous :

```
click="currentState='componentID.viewstatename'"
```

Pour plus d'informations, voir View states.

Création de groupes d'états d'affichage

Flex prend en charge les groupes d'états d'affichage. L'attribut `stateGroups` de la balise `<States>` vous permet de regrouper un ou plusieurs états. Par exemple, si plusieurs composants apparaissent dans le même jeu d'états d'affichage, vous pouvez créer un groupe d'états d'affichage les contenant tous. L'attribution à la propriété `currentState` de l'un des états d'affichage du groupe conduira ensuite à l'affichage des composants. Pour plus d'informations, illustrées d'exemples, voir Defining view state groups.

Flash Builder fournit des conseils de code afin de vous assister lors de la création et de la modification de groupes d'états.

Création d'une transition

Lorsque vous modifiez les états d'affichage d'une application, les composants semblent sauter d'un état d'affichage à un autre. Vous pouvez rendre cet effet plus fluide visuellement en ajoutant des transitions.

Une transition consiste en un ou plusieurs effets regroupés pour être lancés lorsqu'un état d'affichage est modifié. Par exemple, vous pouvez définir une transition utilisant un effet Resize de redimensionnement afin de réduire progressivement un composant dans l'état d'affichage d'origine et un effet Fade de fondu pour afficher progressivement un composant dans le nouvel état d'affichage.

1 Assurez-vous de créer au moins un état d'affichage en plus de l'état de base.

2 Dans l'éditeur MXML, définissez un objet Transition en ajoutant une balise `<s:transitions>`, puis une balise enfant `<s:Transition>`, comme l'illustre l'exemple suivant :

```
<s:transitions>
    <mx:Transition id="myTransition">
    </mx:Transition>
</s:transitions>
```

Pour définir plusieurs transitions, insérez d'autres balises enfant `<s:Transition>` dans la balise `<s:transitions>`.

3 Dans la balise `<s:Transition>`, définissez la modification de l'état d'affichage qui déclenche la transition en définissant les propriétés `fromState` et `toState`, comme l'illustre l'exemple suivant (en gras) :

```
<s:transitions>
    <mx:Transition id="myTransition" fromState="*" toState="checkout">
    </mx:Transition>
</s:transitions>
```

Dans l'exemple, vous spécifiez que vous souhaitez appliquer la transition lorsque l'application change d'un état d'affichage (`fromState="*"`) à l'état d'affichage intitulé checkout (`toState="checkout"`). La valeur `"*"` est un caractère générique représentant n'importe quel état d'affichage.

4 Dans la balise `<mx:Transition>`, indiquez si vous souhaitez que les effets s'exécutent de façon parallèle ou séquentielle en ajoutant une balise enfant `<mx:Parallel>` ou `<mx:Sequence>`, comme l'illustre l'exemple suivant (en gras) :

```
<mx:Transition id="myTransition" fromState="*" toState="checkout">
    <mx:Parallel>
    </mx:Parallel>
</mx:Transition>
```

Si vous souhaitez appliquer simultanément les effets, utilisez la balise `<mx:Parallel>`. Si vous souhaitez les appliquer les uns après les autres, utilisez la balise `<mx:Sequence>`.

5 Dans la balise `<mx:Parallel>` ou `<mx:Sequence>`, spécifiez le ou les composants cible de la transition en définissant la propriété `target` (pour un composant cible) ou `targets` (pour plusieurs composants cible) selon l'ID du ou des composants cible, comme l'illustre l'exemple suivant :

```
<mx:Parallel targets="{[myVGroup1,myVGroup2,myVGroup3]}">
</mx:Parallel>
```

Dans cet exemple, trois conteneurs VGroup sont ciblés. La propriété `targets` sélectionne un tableau de plusieurs ID.

6 Dans la balise `<mx:Parallel>` ou `<mx:Sequence>`, spécifiez les effets à appliquer lors d'un changement d'état d'affichage en ajoutant des balises enfant d'effet, comme l'illustre l'exemple suivant (en gras) :

```
<mx:Parallel targets="{[myVBox1,myVBox2,myVBox3]}">
    <mx:Move duration="400"/>
    <mx:Resize duration="400"/>
</mx:Parallel>
```

Pour connaître la liste des effets possibles et savoir comment définir leurs propriétés, voir Introduction to effects.

7 Pour tester la transition, cliquez sur le bouton Exécuter de la barre d'outils de Flash Builder, puis changez d'état après le lancement de l'application.

Modification de l'interface utilisateur à l'aide des habillages

Les classes d'habillage modifient l'aspect des contrôles dans une interface utilisateur. La création, la modification et l'importation d'habillages varient en fonction des composants (Spark ou MX).

A propos des habillages Spark

Les habillages Spark contrôlent tous les éléments visuels d'un composant, dont sa présentation. Ils peuvent contenir plusieurs éléments, tels que des éléments graphiques, du texte, des images et des transitions. Les habillages Spark prennent en charge les états. Vous pouvez utiliser un habillage pour définir l'aspect d'un composant pour chacun de ses états. Les habillages spécifient généralement une taille minimale pour le composant. Pour plus d'informations sur l'impémentation d'habillages Spark dans Flex, voir A propos des habillages Spark.

A propos des habillages pour les composants MX

Les habillages des composants MX sont soit des graphiques bitmap, soit des graphiques vectoriels. Le graphique bitmap (ou habillage graphique) se compose de pixels individuels formant une image. Le graphique vectoriel (ou habillage par programmation) se compose d'une série de définitions de ligne définissant les points de début et de fin, l'épaisseur, la couleur et d'autres caractéristiques de la ligne requises par Adobe® Flash® Player pour la dessiner. Pour plus d'informations sur l'implémentation des habillages pour les composants MX dans Flex, voir About MX component skinning.

Flash Builder vous permet d'importer des illustrations d'habillage pour des composants MX. Voir « Importation d'illustrations d'habillage pour les composants MX » à la page 181.

Le package `mx.skins.spark` définit les habillages Spark pour les composants MX.

Création et modification d'habillages pour les composants Spark

Vous pouvez ouvrir la boîte de dialogue de nouvel habillage MXML en sélectionnant Fichier > Nouveau > Habillage MXML.

Vous pouvez également l'ouvrir directement à partir de l'éditeur. Par exemple, pour créer une classe skinClass pour un composant Spark Button, procédez comme suit.

1 En mode Source de l'éditeur, placez votre curseur dans une balise `<s:Button>` et saisissez ce qui suit :

```
<s:Button skinClass="
```

Après avoir saisi le premier guillemet pour le nom de la classe skinClass, un menu contextuel s'affiche.

2 Avec Create Skin en surbrillance dans les conseils de code, la touche Entrée ouvre la boîte de dialogue de création d'un habillage MXML.

3 Dans la boîte de dialogue de création d'un habillage MXML, spécifiez les éléments suivants :

- Dossier source et package de la déclaration Skin générée

- Nom

  Nom de la classe Skin créée.

- Composant hôte

  Pour modifier le composant par défaut, cliquez sur Parcourir et sélectionnez un composant hôte.

- (Recommandé) Sélectionnez Créer en tant que copie de et ne supprimez pas le code de style ActionScript.

  Si vous êtes novice en matière d'habillage, utilisez une copie pour vous lancer dans la création d'une classe Skin. Modifiez le code de style ActionScript.

- (Utilisateurs avancés) Si vous savez comment créer des classes Skin, procédez de l'une des manières suivantes :

  Supprimez le code de style ActionScript ou ne créez pas de copie d'une classe existante.

  Si vous ne créez pas de copie d'une classe existante, Flash Builder génère un fichier de classe d'habillage vide comportant des commentaires pour vous guider.

  Les étapes restantes de cette procédure supposent que vous avez suivi l'option recommandée pour la génération d'une classe Skin.

4 Cliquez sur Terminer.

  Flash Builder génère une nouvelle classe skinClass en fonction des sélections effectuées dans la boîte de dialogue de création d'un habillage MXML. L'éditeur passe en mode Source pour la nouvelle classe générée.

5 Modifiez la classe skinClass.

  Enregistrez le fichier de classe et le fichier d'application.

Importation d'illustrations d'habillage pour les composants MX

L'assistant Importation d'illustrations d'habillage permet l'importation d'illustrations graphiques vectorielles ainsi que d'illustrations bitmap à partir des versions CS5 ou ultérieures de Flash Professional, Fireworks, Illustrator et Photoshop pour les illustrations bitmap, tout fichier PNG, JPG ou GIF peut être utilisé. L'illustration peut alors servir d'habillage pour les composants Flex.

Remarque : *Adobe propose un ensemble de modèles d'habillage destinés à faciliter la création d'habillages pour les composants Flex intégrés. Ces modèles, utilisés avec Flash, Fireworks, Illustrator ou Photoshop, permettent de créer l'illustration. Grâce à Flash, vous pouvez aussi créer des composants Flex personnalisés totalement fonctionnels. Pour plus d'informations, voir les articles sous Importation de ressources Flash Professional dans Flex.*

1 Sélectionnez Fichier > Importer > Flash Builder > Illustrations d'habillage MX.

2 Dans la boîte de dialogue Importation d'illustrations d'habillage :

- Choisissez un dossier de bitmaps ou un fichier SWC ou SWF à partir duquel importer des habillages ou cliquez sur Parcourir pour en localiser un. Les types de fichiers pris en charge sont les suivants :
  - Les fichiers SWF AS3 et SWC AS3 créés dans Adobe Flash Professional CS5
  - Les fichiers de graphiques vectoriels créés dans Adobe Illustrator® et exportés en tant que fichiers SWF pour Flash Player 8
  - Les fichiers graphiques bitmap aux formats PNG, GIF et JPG
- Sélectionnez un dossier dans lequel importer les habillages. Il doit s'agir du dossier source d'un projet Flex (vous pouvez également indiquer un sous-dossier du dossier source). Le dossier du projet Flex ouvert est sélectionné par défaut.

- Dans le champ Copier les illustrations dans le sous-dossier, le nom du dossier par défaut est dérivé du dossier ou des ressources en cours d'importation. Cliquez sur Parcourir pour choisir un emplacement différent.
- Dans le champ Créer des règles de style d'habillage dans, précisez un nom pour le fichier CSS qui contient les règles de style. Le nom par défaut est dérivé du nom du dossier de l'illustration ou du fichier FLA en cours d'importation.
- Cochez la case Supprimer toutes les règles existantes dans le fichier si vous voulez que le fichier CSS spécifié soit remplacé lors de l'importation (plutôt que d'importer les habillages et de conserver d'autres définitions existantes dans le fichier CSS). Par défaut, la case est désélectionnée. Si le fichier CSS n'existe pas, elle est désactivée.
- Dans le champ Appliquer les styles à l'application, la valeur par défaut est le fichier sélectionné dans le Navigateur Flex ou dans la vue d'éditeur active, ou le fichier de l'application principale du projet.
- Cliquez sur Suivant.

3 Dans la boîte de dialogue Importation d'illustrations d'habillage suivante, sélectionnez les habillages à importer et précisez quel type de style CSS et propriété de partie d'habillage seront utilisés. Vous pouvez cocher les éléments un par un ou cliquer sur Sélectionner tout ou Désélectionner tout.

- Si les éléments n'ont pas de style valide ou de nom de propriété de partie d'habillage, ils ne sont pas cochés par défaut. Les exemples suivants illustrent la convention d'attribution de noms dans Flash Builder :
  - Button_upSkin
  - Button_glow_downSkin (correspond à la propriété downSkin de la règle de style Button.glow)
  - TabBar-tab_upSkin (la propriété upSkin correspond à la propriété tabStyleName de la règle de style TabBar)
  - MyCustomComponent_borderSkin

  Pour les composants personnalisés, l'élément est coché si le composant a été défini dans le projet vers lequel vous effectuez l'importation.
- Choisissez au besoin un style et une partie d'habillage dans les listes déroulantes de chaque colonne.
- Cliquez sur Terminer.

  Un fichier CSS est créé et affiché dans la vue Source. Le fichier CSS est joint à l'application spécifiée dans l'assistant. Si vous importez un fichier SWC, il est ajouté automatiquement au chemin d'accès à la bibliothèque du projet.

Ajout d'interactivité à l'aide d'effets

Un *effet* est la modification visible ou audible du composant cible se produisant sur une période exprimée en millisecondes. Le fondu, le redimensionnement ou le déplacement d'un composant constituent des exemples d'effets.

Les effets sont générés en réponse à un événement, alors que l'événement est souvent déclenché par une action de l'utilisateur, par exemple un clic sur un bouton. Les effets peuvent toutefois être programmés ou déclenchés en réponse à des événements qui ne sont pas déclenchés par l'utilisateur.

Vous pouvez par exemple créer un effet faisant rebondir légèrement un composant TextInput auquel l'utilisateur accède par tabulation. Vous pouvez également appliquer un effet de fondu en sortie à un composant Label sur lequel l'utilisateur fait glisser le curseur de la souris.

Dans Flash Builder, les effets peuvent être définis en tant que propriétés des composants MXML.

L'implémentation des effets varie suivant qu'il s'agit de composants Spark ou MX. Pour plus d'informations sur la création d'effets en code MXML et ActionScript, voir Introduction to effects.

Génération de rendus d'élément personnalisés

Vous pouvez générer des rendus d'élément destinés à des contrôles basés sur une liste pour des applications de bureau et mobiles.

Pour les applications de bureau, vous pouvez générer des rendus d'élément personnalisés destinés à des contrôles basés sur une liste Spark, tels que List et ComboBox. Vous pouvez également utiliser les rendus d'élément Spark avec certains contrôles MX, tels que MX DataGrid et MX Tree.

Pour les applications mobiles, vous pouvez générer des rendus d'élément personnalisés destinés à des contrôles basés sur une liste mobile.

Utilisez des rendus d'élément personnalisés pour contrôler l'affichage d'un élément de données dans un conteneur DataGroup, SkinnableDataContainer ou dans une sous-classe de ces conteneurs. L'aspect défini par un rendu d'élément peut inclure la police, la couleur d'arrière-plan, la bordure et tout autre aspect visuel de l'élément de données. Un rendu d'élément peut également définir l'aspect d'un élément de données au moment où l'utilisateur interagit avec celui-ci. Par exemple, le rendu d'élément peut afficher l'élément de données d'une certaine manière lorsque l'utilisateur déplace la souris sur ce dernier. Il l'affiche différemment lorsque l'utilisateur clique sur l'élément de données.

Flash Builder permet de générer et de modifier des rendus d'élément. Il utilise l'un des modèles suivants pour générer les rendus d'élément pour une application de bureau :

- Composants Spark

  Utilisez ce modèle pour les contrôles basés sur une liste Spark, tels que List et ComboBox.

- MX Advanced DataGrid
- MX DataGrid
- MX Tree

Il utilise l'un des modèles suivants pour générer les rendus d'élément destinés à des contrôles basés sur une liste dans une application mobile :

- Elément de rendu d'icône

  Utilisez ce modèle pour créer un rendu d'élément personnalisé destiné à des contrôles basés sur une liste en indiquant le libellé, le message et les propriétés de l'icône. Le fichier de rendu d'élément est créé au format MXML.

- Rendu d'élément ActionScript personnalisé

  Utilisez ce modèle pour créer un rendu d'élément standard destiné à des contrôles basés sur une liste. Vous pouvez ensuite personnaliser le rendu d'élément afin de définir l'aspect de la liste, le cas échéant. Le fichier de rendu d'élément est créé au format ActionScript.

Création et modification de rendus d'élément

Vous pouvez ouvrir l'assistant Nouvel ItemRenderer depuis l'éditeur MXML. Dans l'assistant de création d'un rendu d'élément, spécifiez un nom et un modèle pour le rendu d'élément. Flash Builder génère un fichier MXML ou ActionScript qui implémente le rendu d'élément.

Les composants de l'application référencent le rendu d'élément généré à l'aide de la propriété `itemRenderer`.

Par exemple, procédez comme suit pour créer un rendu d'élément pour un composant Spark List.

1 Dans l'éditeur MXML, placez votre curseur dans une balise `<s:List>` et saisissez ce qui suit :

```
<s:List itemRender="
```

Après avoir saisi le premier guillemet du nom de la classe du rendu d'élément, un menu contextuel s'affiche.

2 Cliquez deux fois sur Créer un rendu d'élément pour ouvrir la boîte de dialogue de création d'un rendu d'élément

Pour une application de bureau, spécifiez ce qui suit dans la boîte de dialogue Nouvel ItemRenderer :

- Dossier source et package pour la déclaration du rendu d'élément généré
- Nom

  Nom de la classe de rendu d'élément créée.
- Modèle

  Sélectionnez le modèle à utiliser lors de la génération du rendu d'élément.

Pour une application mobile, spécifiez ce qui suit dans la boîte de dialogue Nouvel ItemRenderer :

- Dossier source et package pour la déclaration du rendu d'élément généré
- Nom

  Nom de la classe de rendu d'élément créée.
- Modèle

  Sélectionnez le modèle de liste des rendus d'éléments d'icône pour mobile.
- Champ du libellé

  Nom du champ dans les données que vous souhaitez définir comme libellé.
- Champ du message

  Nom du champ dans les données que vous souhaitez définir comme contenu du message.
- Champ de l'icône

  Nom du champ dans les données contenant le chemin d'accès à l'icône.
- Largeur de l'icône

  Largeur de l'icône. La valeur par défaut est 64 pixels.
- Hauteur de l'icône

  Hauteur de l'icône. La valeur par défaut est 64 pixels.
- Classe du décorateur

  Sélectionnez une image au format GIF, JPEG ou PNG. L'image sélectionnée est intégrée à l'application. Par exemple, `decoratorClass="@Embed('/foo/myfoo.png')"`

  Vous pouvez également sélectionner un fichier FXP. Ajoutez le chemin d'accès au fichier FXP pour référencer le fichier FXG. Par exemple, `decoratorClass="{assets.Chevron}"`

3 Cliquez sur Terminer.

Flash Builder génère un nouveau rendu d'élément en fonction des sélections effectuées dans la boîte de dialogue de création d'un rendu d'élément L'éditeur passe en mode Source pour la nouvelle classe générée.

4 Modifiez la classe du rendu d'élément.

Enregistrez le fichier de classe et le fichier d'application.

Pour plus d'informations sur la création et l'utilisation des rendus d'élément personnalisés, voir Custom Spark Item Renderers (Rendus d'éléments Spark personnalisés).

Génération de gestionnaires d'événement

Les applications conçues avec Flex sont gérées par les événements. Les composants de l'interface utilisateur réagissent à divers événements, par exemple lorsqu'un utilisateur clique sur un bouton ou lorsque l'initialisation d'un objet est terminée. Les gestionnaires d'événement, rédigés en code ActionScript, définissent la manière dont les composants réagissent à l'événement.

Remarque : *vous pouvez également générer des gestionnaires d'événement pour des éléments non visibles, tels que RemoteObject et HTTPService.*

Flash Builder assiste l'utilisateur par la génération des fonctions des gestionnaires d'événement pour un composant. Dans la fonction générée, rédigez le code qui définit le comportement du composant en réponse à l'événement.

L'assistant de contenu des éléments de l'éditeur MXML vous permet d'accéder à l'assistance du gestionnaire d'événements.

A propos des gestionnaires d'événement générés

Lorsque Flash Builder génère une fonction de gestionnaire d'événement, il place le gestionnaire d'événement dans le premier bloc Script du fichier. La fonction est placée à la fin du bloc Script. Le gestionnaire d'événement généré a un accès protégé et accepte la sous-classe appropriée de la classe Event comme seul paramètre.

Flash Builder génère soit un nom unique pour le gestionnaire d'événement en fonction du nom de la classe du composant, soit un nom personnalisé pour le gestionnaire d'événement spécifié. Si vous ne précisez pas de nom personnalisé, le nom est généré selon le processus suivant :

- Si une propriété ID est définie, Flash Builder en dérive le nom.
- Si aucune propriété ID n'est définie pour le composant, Flash Builder génère un nom unique, basé sur le nom de la classe du composant.

Vous devez fournir le corps du gestionnaire d'événement. Le bloc de code suivant contient un gestionnaire d'événement généré pour un contrôle Button.

```
. . .
<fx:Script>
    <![CDATA[
        protected function myButton_clickHandler(event:MouseEvent):void
        {
            // TODO Auto-generated method stub
        }

    ]]>

</fx:Script>
<s:Button label="Button" id="myButton" click="myButton_clickHandler(event)"/>
. . .
```

Flash Builder désigne un événement par défaut pour chaque composant de l'interface utilisateur. Par exemple, l'événement par défaut d'un contrôle Button est l'événement click.

Création de gestionnaires d'événement pour les composants

1 Dans un bloc MXML de l'éditeur, créez un composant, mais sans préciser d'événements.

2 Activez l'assistant de contenu pour les propriétés d'un composant en insérant un espace après le nom de la classe.

3 Dans la liste de propriétés sélectionnées, sélectionnez un événement (doubleClick, par exemple).

4 Appuyez sur Ctrl+Espace et sélectionnez Générer un gestionnaire d'événement.

Flash Builder génère un nom unique pour le gestionnaire d'événement et place ce dernier dans le bloc Script.

Remarque : *si vous avez spécifié un nom personnalisé pour le gestionnaire d'événement, Flash Builder ne peut pas générer le gestionnaire. Pour utiliser un nom personnalisé, générez d'abord un gestionnaire d'événement, puis modifiez-en le nom aussi bien dans la propriété event que dans le gestionnaire généré.*

Ajoutez le code pour l'implémentation du gestionnaire d'événement à la fonction de gestionnaire d'événement générée.

Accès aux services de données

Dans Flash Builder, vous interagissez avec des données et des contrôles gérés par des données directement dans les codes ActionScript et MXML. Vous pouvez travailler avec des données, générer automatiquement des applications de bases de données et utiliser un code proxy pour les services Web, ainsi que générer et utiliser le code fonctionnant avec Flex Ajax Bridge. Vous pouvez également gérer les problèmes de sécurité d'accès aux données d'Adobe Flash Player et utiliser Flash Builder avec un service proxy.

Utilisation de données dans Flash Builder

Dans Flash Builder, les données sont manipulées en modifiant directement les codes des applications ActionScript et MXML.

Conteneurs et contrôles gérés par les données

fournit des composants de type conteneur et de type contrôle à partir desquels vous pouvez créer les interfaces utilisateur de vos applications Flex. Certains composants présentent des données que les utilisateurs peuvent sélectionner et avec lesquelles ils peuvent interagir lors de l'utilisation de l'application. Voici quelques exemples d'utilisation des contrôles gérés par les données :

- Dans un formulaire pour la saisie d'une adresse, vous pouvez donner aux utilisateurs la possibilité de sélectionner leur pays de résidence (par exemple) en utilisant les contrôles ComboBox ou List.
- Dans une application de panier d'achats, vous pouvez utiliser un composant Spark List pour présenter les données des produits comportant des images. Pour le composant List, la présentation peut prendre les valeurs suivantes : VerticalLayout, HorizontalLayout ou TileLayout.
- Vous pouvez fournir des options standard de navigation en utilisant des conteneurs tels que les contrôles TabBar et ButtonBar.

Pour tous les contrôles gérés par des données, l'entrée de données est effectuée par un *fournisseur de données*.

Pour plus d'informations sur l'utilisation de contrôles gérés par des données, voir Spark list-based controls.

Fournisseurs de données et collections

Un objet *collection* contient un objet de données, par exemple Array ou XMLList, et fournit un ensemble de méthodes permettant de consulter, trier, filtrer et modifier les éléments de données de cet objet. Plusieurs contrôles Adobe Flex, désignés par l'expression *contrôles de fournisseurs de données*, ont une propriété `dataProvider` renseignée par une collection.

Les exemples suivants illustrent comment un fournisseur de données est défini (en tant qu'ArrayCollection ActionScript) et utilisé par un contrôle :

```
<!-- Simple example to demonstrate the Spark ComboBox control -->
<s:Application xmlns:fx="http://ns.adobe.com/mxml/2009"
               xmlns:s="library://ns.adobe.com/flex/spark"
               xmlns:mx="library://ns.adobe.com/flex/halo">

    <fx:Script>
        <![CDATA[
            import mx.collections.ArrayCollection;

            [Bindable]
            public var complexDP:ArrayCollection = new ArrayCollection(
                [    {ingredient:"Salmon", category:"Meat"},
                    {ingredient:"Potato", category:"Starch"},
                    {ingredient:"Cucumber", category:"Vegetable"},
                    {ingredient:"Steak", category:"Meat"},
                    {ingredient:"Rice", category:"Starch"},
                    {ingredient:"Cumin", category:"Spice"}
                ]
            );

            <!-- Function to handel custom input strings -->
            private function myLabelToItemFunction(input:String):*
            {
                <!-- Returns object that matches items in dataProvider -->
                return {ingredient:input, category:"mystery"};
            }
        ]]>
    </fx:Script>
```

```
    <s:Panel title="Spark ComboBox Example" width="75%" height="75%">
        <s:layout>
            <s:VerticalLayout paddingTop="10" paddingLeft="10"/>
        </s:layout>

        <!-- Label that displayed current property values -->
        <s:Label text="Index : {cb.selectedIndex}
                       Item : {cb.selectedItem.ingredient}
                       Type : {cb.selectedItem.category}"/>

        <!-- ComboBox with custom labelToItem function -->
        <s:ComboBox
            id="cb"
            dataProvider="{complexDP}"
            width="150"
            labelToItemFunction="{myLabelToItemFunction}"
            selectedIndex="0"
            labelField="ingredient"/>
    </s:Panel>
</s:Application>
```

Pour plus d'informations sur les fournisseurs de données et les collections, voir Data providers and collections.

Accès aux données distantes

Flex contient des composants d'accès aux données basés sur une architecture orientée services (Service-oriented architecture, SOA). Ces composants utilisent des appels de procédure distants pour interagir avec des environnements de serveur (PHP, Adobe ColdFusion et Microsoft ASP.NET, par exemple), afin de fournir des données aux applications et d'envoyer des données aux sources de données d'arrière-plan.

Selon le type d'interface dont vous disposez sur une application côté serveur particulière, vous pouvez vous connecter à une application via l'une des méthodes suivantes :

- HTTP GET ou POST avec le composant HTTPService
- Services Web conformes au protocole SOAP avec le composant WebService
- Services distants AMF (Adobe Action Message Format) avec le composant RemoteObject

Remarque : *lorsque vous utilisez Flash Builder pour développer des applications accédant aux données côté serveur, utilisez un fichier* `cross-domain.xml` *ou un proxy si vous accédez aux données à partir d'un domaine différent du domaine à partir duquel l'application a été chargée.*

Flash Builder permet également de créer des applications utilisant Adobe LiveCycle Data Services, un produit distinct offrant des fonctionnalités de service de données avancées. LiveCycle Data Services fournit des connexions proxy pour les applications de service d'appel de procédure distante (RPC), ainsi qu'une configuration de sécurité avancée. Il offre également les services de données suivants :

Service de gestion de données : ce service permet de créer des applications qui fonctionnent avec les données distribuées. Le service de gestion de données vous permet en outre de gérer de vastes collections de données et des relations de données imbriquées, telles que des relations un-à-un et un-à-plusieurs.

Service de message : ce service permet de créer des applications pouvant envoyer des messages à d'autres applications et recevoir des messages d'autres applications, dont les applications créées dans Flex et JMS (Java Message Service).

Liaison de données

Comme vous avez pu le constater dans l'exemple de code fourni dans « Fournisseurs de données et collections » à la page 187, la valeur de la propriété `dataProvider` du contrôle ComboBox est `"{complexDP}"`. Il s'agit d'un exemple de liaison de données.

La liaison de données copie la valeur d'un objet (la source) dans un autre objet (la destination). Une fois qu'un objet est lié à un autre objet, les modifications apportées à la source sont automatiquement répercutées dans la destination.

L'exemple suivant lie la propriété de texte d'un contrôle TextInput (la source) à la propriété de texte d'un contrôle Label (la destination), de sorte à ce que le texte saisi dans le contrôle TextInput s'affiche dans le contrôle Label :

```
<?xml version="1.0" encoding="utf-8"?>
<s:Application xmlns:fx="http://ns.adobe.com/mxml/2009"
               xmlns:s="library://ns.adobe.com/flex/spark"
               xmlns:mx="library://ns.adobe.com/flex/mx" minWidth="955" minHeight="600">
    <fx:Declarations>
        <!-- Place non-visual elements (e.g., services, value objects) here -->
    </fx:Declarations>

    <s:TextInput id="LNameInput"  x="10" y="10"/>
    <s:Label text="{LNameInput.text}"  x="10" y="50"/>
</s:Application>
```

Pour lier des données d'un objet à un autre, vous pouvez faire appel à la syntaxe d'accolades ({ }) (comme l'illustre l'exemple) ou à la balise `<fx:Binding>`. Pour plus d'informations, voir Utilisation de liaisons avec des modèles de données et Liaison de données.

Modèles de données

Un *modèle de données* est un objet que vous pouvez utiliser afin de stocker de manière temporaire des données dans la mémoire pour faciliter leur manipulation. Vous pouvez définir un modèle de données dans ActionScript et dans MXML en utilisant une balise telle que `<fx:Model>` ou en utilisant tout objet contenant des propriétés. Par exemple, le modèle de données suivant affiche des informations telles que le nom d'une personne, son âge et son numéro de téléphone :

```
<fx:Declarations>
    <fx:Model id="reg">
        <registration>
            <name>{nme.text}</name>
            <email>{email.text}</email>
            <phone>{phone.text}</phone>
            <zip>{zip.text}</zip>
            <ssn>{ssn.text}</ssn>
        </registration>
    </fx:Model>
</fx:Declarations>
```

Les champs d'un modèle de données peuvent contenir des données statiques (comme l'illustre l'exemple). Vous pouvez également avoir recours aux liaisons de données pour transmettre des données au modèle de données ou en provenance de ce dernier.

Vous pouvez en outre définir le modèle de données dans un fichier XML. Vous référencez alors le fichier XML par le biais du système de fichiers ou d'une URL à l'aide de la propriété `source` de la balise `<fx:Model>`, comme l'illustre l'exemple suivant :

```
<fx:Model source="content.xml" id="Contacts"/>
<fx:Model source="http://www.somesite.com/companyinfo.xml" id="myCompany"/>
```

Pour plus d'informations sur les modèles de données, voir Storing Data.

Validation de données

La validation de données vérifie que les données saisies par l'utilisateur dans l'application sont valides. Vous pouvez par exemple utiliser un *validateur de données* de code postal pour vérifier que le code postal indiqué par l'utilisateur est valide.

Flex propose des validateurs prédéfinis pour les types de données suivants : carte de crédit, devise, date, adresse électronique, numéro, numéro de téléphone, expression régulière, sécurité sociale, chaîne et code postal.

Les validateurs de données sont des composants Flex non visuels, ce qui signifie qu'ils ne sont pas accessibles à partir de la vue Composants. Vous devez donc les insérer dans le code, comme l'illustre l'exemple MXML suivant :

```
<!-- Define the ZipCodeValidator. -->
<mx:ZipCodeValidator id="zcV" source="{zipcodeInput}" property="text"/>
<!-- Define the TextInput control for entering the zip code. -->
<s:TextInput id="zipcodeInput"/>
```

Dans cet exemple MXML, le validateur est défini avec la balise MXML appropriée et est lié à la propriété ID d'un contrôle TextInput. Au moment de l'exécution, lorsque l'utilisateur saisit le numéro de téléphone dans le contrôle TextInput, le numéro est immédiatement validé.

Vous pouvez utiliser les validateurs de données dans ActionScript en définissant une variable en tant qu'instance d'une classe de validateur, puis en créant une fonction pour la lier au contrôle de saisie.

Les validateurs de données sont souvent utilisés avec les modèles de données. Pour plus d'informations, voir Validating Data.

Mise en forme des données

La mise en forme correcte de certains types de données dans l'application est garantie par l'utilisation de *formateurs de données*. Flex fournit des formateurs de données prédéfinis pour les types de données suivants : devise, date, numéro, téléphone et code postal.

Les formateurs de données sont liés aux contrôles de saisie. Ils mettent correctement en forme les données saisies par l'utilisateur. Par exemple, un utilisateur peut saisir une date au format suivant :

120105

Un formateur, lié au contrôle de saisie de texte, stocke et affiche la date au format suivant :

12/01/05

A l'instar des validateurs de données, les formateurs de données sont des composants Flex non visuels que vous pouvez utiliser avec des balises MXML ou des classes ActionScript.

Pour plus d'informations, voir Formatting Data.

Configuration de l'accès aux services de données

L'assistant Nouveau projet pour les projets Flex et les projets Flex Mobile présente des options de configuration d'accès aux services de données. Les étapes de l'assistant sont identiques pour les projets Flex et les projets Flex Mobile. Vous pouvez configurer l'accès aux types de services suivants :

Services PHP
Services ColdFusion

Services Java
ASP.NET

Configuration de l'accès aux services PHP

L'accès aux données de services PHP présuppose l'existence d'un serveur hébergeant les services. Il peut s'agir d'un serveur local ou d'un serveur appartenant à un réseau local.

1 Dans l'assistant Nouveau projet, indiquez l'emplacement du projet. Pour les projets mobiles, spécifiez les paramètres mobiles.

Voir « Projets Flex » à la page 61 et « Projets Flex Mobile » à la page 63.

2 Sélectionnez PHP comme type de serveurs d'applications.

3 Spécifiez la racine Web et l'URL racine pour le service. Cliquez sur Valider la configuration.

La racine Web et l'URL racine spécifiées appartiennent généralement à votre environnement local. Vous pouvez toutefois également accéder à des serveurs réseau. Vérifiez que le répertoire est partagé et que le compte sous lequel Flash Builder est exécuté possède des droits d'accès en écriture.

Vérifiez que vous avez mappé ou monté un lecteur pour le serveur réseau. Spécifiez ensuite le chemin vers le serveur. Le chemin d'accès est spécifique à la plateforme. Par exemple :

(Windows) `\\10.192.18.12\serveur\racine Web`

(Windows) `Z:\racine Web`

(Mac) `/Volumes/racine Web`

4 (Facultatif) Spécifiez le dossier de sortie de l'application.

5 Cliquez sur Terminer ou sur Suivant pour accéder à d'autres options de configuration.

Voir « Chemins de génération, extension natives et autres options de configuration du projet » à la page 65.

Configuration de l'accès aux services ColdFusion

Pour accéder aux données qui utilisent ColdFusion, vous devez disposer d'Adobe ColdFusion 8 ou d'une version ultérieure de ColdFusion. Pour plus d'informations, consulter la page produit de ColdFusion.

1 Dans l'assistant Nouveau projet, indiquez l'emplacement du projet. Pour les projets mobiles, spécifiez les paramètres mobiles.

Voir « Projets Flex » à la page 61 et « Projets Flex Mobile » à la page 63.

2 Pour le type de serveurs d'applications, sélectionnez ColdFusion, puis choisissez l'une des options suivantes.

Utiliser le service d'accès aux objets distants : si vous désactivez l'option Utiliser le service d'accès aux objets distants, spécifiez la racine Web et l'URL racine à l'étape suivante.

Si vous activez l'option Utiliser le service d'accès aux objets distants, vous pouvez effectuer les choix suivants.

- Data Services

  Spécifiez LiveCycle Data Services en tant que type d'application ColdFusion uniquement si votre installation ColdFusion est configurée pour LiveCycle Data Services. Voir Integrating Adobe LiveCycle Data Services ES 2.6 with Adobe ColdFusion 8 (Intégration d'Adobe LiveCycle Data Services ES 2.6 à Adobe ColdFusion 8).

  Généralement, le type de serveurs d'applications pour LiveCycle Data Services n'est pas ColdFusion, mais Java. Voir « Configuration de l'accès aux services Java » à la page 193.

- BlazeDS

  Spécifiez BlazeDS en tant que type d'application ColdFusion uniquement si votre installation ColdFusion est configurée pour Adobe BlazeDS. Voir Integrating BlazeDS with a ColdFusion 8 Installation (Intégration de BlazeDS à une installation ColdFusion 8).

  Généralement, le type de serveurs d'applications pour BlazeDS n'est pas ColdFusion, mais Java. Voir « Configuration de l'accès aux services Java » à la page 193.

- ColdFusion Flash Remoting

  Activez cette option si vous prévoyez d'utiliser les outils de développement centrés sur les données disponibles avec Flash Builder. Cette option s'applique également si vous utilisez Flash Remoting pour invoquer des méthodes dans les composants ColdFusion (CFC). Voir Création d'applications centrées sur les données avec Flash Builder.

3 Spécifiez un emplacement de serveur, une racine Web, une URL racine et une racine du contexte.

Pour l'accès à un service d'objets distants, vous pouvez définir une configuration ColdFusion autonome ou une configuration ColdFusion déployée sur un serveur d'applications Java.

- Autonome

  Activez l'option Autonome si l'installation ColdFusion utilise la configuration du serveur.

  Spécifiez l'emplacement du serveur ColdFusion, de la racine Web et de l'URL racine.

- Déployé(e) vers le serveur d'applications Java

  Utilisez l'option Déployé(e) vers le serveur d'applications Java si votre installation ColdFusion fait appel aux configurations multiserveurs ou Java.

  Spécifiez la racine Web, l'URL racine et la racine du contexte. Si vous faites appel à la configuration multiserveurs ColdFusion, vous ne devrez généralement pas spécifier la racine du contexte.

  La racine du contexte correspond en règle générale au dernier segment du chemin racine de l'URL lorsque vous déployez ColdFusion en tant qu'application Web dans la configuration Java ColdFusion.

Pour spécifier l'emplacement du serveur ou de la racine Web, recherchez un répertoire local ou saisissez le chemin vers un répertoire situé sur un serveur réseau. Vérifiez que le répertoire est partagé et que le compte sous lequel Flash Builder est exécuté possède des droits d'accès en écriture.

Vérifiez que vous avez mappé ou monté un lecteur réseau pour le serveur réseau. Le chemin au serveur réseau est fonction de la plateforme. Par exemple :

(Windows) `\\10.192.18.12\serveur\racine Web`

(Windows) `Z:\racine Web`

(Mac) `/Volumes/racine Web`

4 Cliquez sur Valider la configuration pour vérifier l'exactitude des paramètres spécifiés.

Si le répertoire de la racine Web n'est pas accessible en écriture, Flash Builder affiche un avertissement.

5 Sélectionnez un dossier de sortie pour l'application compilée.

6 Cliquez sur Terminer ou sur Suivant pour accéder à d'autres options de configuration.

Voir « Chemins de génération, extension natives et autres options de configuration du projet » à la page 65.

Configuration de l'accès aux services Java

Cette configuration de projet vous permet de créer des projets Flex utilisant des classes de service Java avec l'option de service d'accès aux objets distants. Si aucune option n'est sélectionnée et qu'un serveur Java est utilisé, un dossier de sortie est créé sous la racine du serveur. Si le plug-in Eclipse Web Tools Project (WTP) est installé, vous pouvez créer des projets mixtes Java et Flex avec ou sans service d'accès aux objets distants.

Remarque : *LiveCycle Data Services et BlazeDS prennent en charge des versions spécifiques du SDK Flex. Consultez le Tableau de compatibilité LiveCycle Data Services pour connaître les versions du SDK Flex prises en charge par votre version de LiveCycle Data Service. Ce tableau de compatibilité répertorie également les versions du SDK Flex prises en charge par BlazeDS.*

1 Dans l'assistant Nouveau projet, indiquez l'emplacement du projet. Pour les projets mobiles, spécifiez les paramètres mobiles.

Voir « Projets Flex » à la page 61 et « Projets Flex Mobile » à la page 63.

2 Sélectionnez Java comme type de serveurs d'applications.

3 (Facultatif) Cochez l'option Utiliser le service d'accès aux objets distants.

Data Services ES est sélectionné par défaut. Vous pouvez sélectionner BlazeDS. Si vous avez installé WTP, vous pouvez aussi choisir de créer un projet mixte Java et Flex faisant appel à WTP (le dossier source Java est sélectionné pour vous).

4 Configurez le serveur d'applications Java.

- Si vous sélectionnez les options Utiliser le service d'accès aux objets distants et Data Services ES ou BlazeDS, spécifiez les paramètres racine :

  Dossier racine : emplacement physique du serveur d'applications Web qui affiche les données de votre application (par exemple, `C:\Program Files\Apache Software Foundation\Tomcat 5.5\webapps\myservices`). Si vous utilisez un serveur distant, il doit se trouve sur un lecteur mappé ou accessible par chemin d'accès UNC.

  URL racine : URL racine de l'application Web. Cette URL correspond à l'emplacement physique nommé dans Dossier racine ci-dessus. Pour BlazeDS sur Tomcat, il peut s'agir de l'URL :

  ```
  http://localhost:8080/myservices
  ```

  Pour Data Services, l'URL racine par défaut est :

  ```
  http://localhost:8400/lcds/
  ```

  Si vous utilisez un serveur distant, il peut s'agir de l'URL suivante :

  ```
  http://myserver.com:8400/lcds/.
  ```

  Racine du contexte : la racine du contexte correspond généralement au dernier segment du chemin de l'URL racine. Pour les exemples de l'URL racine donnés ci-dessus, la racine contextuelle serait `/myservices` pour BlazeDS et `/lcds` pour Data Service.

- Si vous avez sélectionné l'option Créer un projet mixte Java/Flex à l'aide de WTP (avec ou sans LiveCycle Data Services) :
  - Indiquez le nom des dossiers source et des exécutions cible Java et Flex.

    Lorsque vous créez un projet Flex avec LiveCycle Data Services, Flash Builder crée un répertoire portant le même nom que le projet ou utilise un répertoire existant portant ce nom. Ce répertoire est un sous-répertoire du dossier racine que vous avez spécifié pour le projet.

  - Avec Data Services ES, précisez un fichier flex.war qui se trouve dans le dossier d'installation du serveur.

Remarque : *quelle que soit l'option choisie pour un projet LiveCycle Data Services dans Flash Builder, spécifiez un dossier racine et une URL racine. Ces valeurs mappent la racine d'une application Web LiveCycle Data Services. Si vous désélectionnez les options, saisissez uniquement la racine Web et l'URL racine.*

5 Spécifiez l'emplacement auquel le projet sera compilé.

6 Cliquez sur Terminer ou sur Suivant pour accéder à d'autres options de configuration.

Voir « Chemins de génération, extension natives et autres options de configuration du projet » à la page 65.

Création d'un projet Flex avec ASP.NET

Si Microsoft Windows et Microsoft Visual Web Developer sont installés, vous pouvez créer des projets Flex déployés avec ASP.NET. De plus, si vous avez accès à un serveur de développement Internet Information Service (IIS), vous pouvez créer des projets Flex avec un dossier de sortie Flex sous IIS.

1 Dans l'assistant Nouveau projet, indiquez l'emplacement du projet. Pour les projets mobiles, spécifiez les paramètres mobiles.

Voir « Projets Flex » à la page 61 et « Projets Flex Mobile » à la page 63.

2 Sélectionnez ASP.NET comme type de serveurs d'applications.

3 Sélectionnez le serveur ASP.NET :

- Si vous utilisez un serveur de développement ASP.NET, vous n'avez pas besoin de préciser l'emplacement du serveur.
- Si vous utilisez IIS, entrez la racine de l'application Web et l'URL de l'application Web.
- Spécifiez le dossier de sortie de l'application

4 Cliquez sur Terminer ou sur Suivant pour accéder à d'autres options de configuration.

Voir « Chemins de génération, extension natives et autres options de configuration du projet » à la page 65.

Modification des options de serveur des projets existants

Vous serez probablement amené dans certains cas à modifier la configuration d'origine du serveur d'un projet. Pour une application Web ou d'ordinateur, accédez à la fenêtre des propriétés du projet concerné.

Sélectionnez l'option Serveur Flex pour ajouter ou modifier les options du serveur du projet.

- Sélectionnez Aucun pour supprimer la configuration serveur d'un projet.

  La suppression de la configuration serveur d'un projet conduit également à la suppression de tout fichier SWC ajouté au chemin d'accès à la bibliothèque pour ce type de serveur.

- Sélectionnez un type de serveur à modifier ou ajoutez la configuration serveur d'un projet.

  Toutes les options de serveur pour la configuration serveur sélectionnée sont disponibles. Pour plus d'informations sur les paramètres de configuration du serveur, voir « Projets Flex » à la page 61.

  La modification du type de serveur d'un projet peut provoquer des erreurs dans le code existant reposant sur le type de serveur d'origine. Recherchez et corrigez toutes les erreurs résultantes dans votre code.

Gestion de la sécurité de Flash Player

Flash Player n'autorise pas une application à recevoir des données d'un domaine autre que celui à partir duquel il a été chargé, à moins d'une autorisation explicite donnée. Si vous chargez le fichier SWF d'application depuis http://mydomain.com, il ne peut pas charger de données depuis http://yourdomain.com. Ce sandbox de sécurité empêche une utilisation malveillante des fonctions de Flash Player (JavaScript utilise un modèle de sécurité similaire pour empêcher toute utilisation malveillante de JavaScript).

Pour accéder aux données à partir d'une application développée dans Flex, placez votre fichier SWF d'application sur le serveur qui héberge le service de données ou effectuez une des opérations suivantes :

Utilisation des fichiers de régulation interdomaines

Un fichier de régulation interdomaines est un simple fichier XML qui accorde à Flash Player l'autorisation d'accéder aux données à partir d'un domaine autre que le domaine sur lequel réside l'application. Sans ce fichier de régulation, l'utilisateur est invité à fournir une autorisation d'accès via une boîte de dialogue. Cette situation est à éviter.

Le fichier de régulation interdomaines (intitulé crossdomain.xml) est placé à la racine du serveur (ou des serveurs) contenant les données auxquelles vous souhaitez accéder. L'exemple suivant illustre un fichier de régulation interdomaines :

```
<?xml version="1.0"?>
<!DOCTYPE cross-domain-policy SYSTEM "http://www.adobe.com/xml/dtds/cross-domain-policy.dtd">
<cross-domain-policy>
    <allow-access-from domain="www.yourdomain.com" />
</cross-domain-policy>
```

Pour plus d'informations sur la configuration des fichiers de régulation interdomaines, voir Sécurité.

Configuration de Flash Builder pour l'accès par proxy aux données distantes

Sur le serveur contenant le fichier SWF d'application, créez un proxy qui appelle un service de données hébergé sur un autre serveur.

Le service proxy traite les requêtes de l'application vers le service distant et répond du service distant vers l'application (Flash Player).

Pendant le développement des applications, le proxy est souvent hébergé sur l'ordinateur local. Pour ce faire, exécutez un serveur Web et un langage de script sur votre ordinateur de développement local.

Une fois que vous avez configuré un proxy pour accéder aux données d'un service distant, vous placez les fichiers d'application dans le même domaine que le proxy. Dans Flash Builder, vous pouvez modifier les paramètres de version du projet et la configuration de lancement pour gérer l'utilisation d'un proxy.

Si vous utilisez Flash Builder pour compiler vos applications et que le serveur proxy est également configuré sur votre ordinateur de développement local, vous pouvez modifier les paramètres de version de projet de sorte à copier automatiquement les fichiers d'application compilés à l'emplacement approprié sur votre serveur Web.

Modification d'un chemin de génération de projet

1. Dans l'Explorateur de packages Flex, sélectionnez un projet.
2. Cliquez avec le bouton droit de la souris et sélectionnez Propriétés. La boîte de dialogue des propriétés du projet s'affiche.
3. Sélectionnez la page Chemin de génération Flex.
4. Modifiez le dossier de sortie existant en saisissant un nouveau chemin ou en naviguant jusqu'au dossier approprié du serveur Web (C:\inetpub\wwwroot\myApp\, par exemple).

5 Cliquez sur OK.

Pour exécuter et déboguer l'application à partir du serveur Web, modifiez la configuration de lancement du projet.

Remarque : *si le serveur proxy n'est pas votre ordinateur local, copiez le contenu du répertoire bin sur le serveur avant l'exécution ou le débogage du programme.*

Surveillance des applications qui accèdent aux services de données

Le Moniteur de réseau est un outil ingénieux permettant de surveiller et de déboguer les applications accédant aux services de données. Le Moniteur de réseau permet d'analyser les données transmises entre une application et un service de données. Il examine également les données XML, AMF et JSON envoyées avec les protocoles SOAP, AMF, HTTP et HTTPS.

Le Moniteur de réseau est disponible dans les vues Flash de développement et de débogage.

Activation de la surveillance du réseau

Le Moniteur de réseau peut être activé individuellement pour chaque projet Flex. L'état du moniteur (activé ou désactivé) s'applique à toutes les applications de ce projet. Il est impossible d'activer ou de désactiver le Moniteur de réseau pour des applications individuelles.

Par défaut, le Moniteur de réseau n'est pas activé. Activez le Moniteur de réseau en sélectionnant l'icône Activer le moniteur dans la barre d'outils de la vue du Moniteur.

Cette procédure suppose que vous êtes dans la perspective Développement Flex ou Débogage Flex.

1 Si la vue Moniteur de réseau n'est pas ouverte, sélectionnez Fenêtre dans le menu Flash Builder, puis Affichage d'une vue > Flash Builder > Moniteur de réseau.

2 Si le Moniteur de réseau n'est pas activé, cliquez sur le bouton Activer le moniteur de réseau dans la barre d'outils de la vue Moniteur de réseau. Cet bouton permet de l'activer et de le désactiver.

Surveillance des services distants

Pour surveiller l'application, activez le Moniteur de réseau et exécutez la version de développement ou de débogage de l'application.

En général, le Moniteur de réseau capture et stocke toutes les données d'événements jusqu'à ce que vous quittiez l'application ou effaciez les données de manière explicite. Les événements s'affichent dans l'ordre chronologique.

Démarrage d'une session de surveillance

1 Exécutez la version de développement ou de débogage de l'application qui accède aux services distants.

2 Pour chaque accès à un service distant, le Moniteur de réseau répertorie les éléments suivants :

- Heure de la requête
- Service
- Demande d'un service
- Heure de la réponse
- Temps écoulé

3 Sélectionnez un en-tête de colonne pour trier les données renvoyées en fonction des valeurs de ladite colonne.

Cliquez de nouveau sur la colonne pour inverser l'ordre des données.

4 Sélectionnez les onglets de requête et de paramètres situés au bas du moniteur pour afficher les détails de l'opération.

Les données envoyées dans la requête, ainsi que les autres informations concernant la requête, peuvent être visualisées à partir de ces onglets.

5 Sélectionnez les onglets de réponse et de résultats situés au bas du moniteur pour afficher les détails de la réponse.

Les données envoyées dans la requête, ainsi que les autres informations concernant la réponse, peuvent être visualisées à partir de ces onglets.

6 Cliquez deux fois sur une entrée pour accéder au code source de l'opération.

L'éditeur de source de Flash Builder s'ouvre ; la ligne correspondant au code source est mise en évidence.

Remarque : *dans la plupart des cas, le Moniteur de réseau peut faire correspondre un événement au code source Flex. Certains événements toutefois sont déclenchés en dehors de la portée du Moniteur de réseau. Dans ce cas, le Moniteur ne trouvera pas le code source Flex.*

7 Cliquez sur le bouton Enregistrer de la barre d'outils du Moniteur de réseau pour inscrire toutes les informations saisies dans un fichier XML.

Remarque : *utilisez le fichier XML généré pour étudier les données hors ligne. Il est impossible d'importer des données de ce fichier vers le Moniteur de réseau.*

8 Cliquez sur l'icône Effacer dans la barre d'outils du Moniteur de réseau pour supprimer toutes les informations du moniteur.

Interruption d'une session de surveillance

Vous pouvez interrompre, puis reprendre, la surveillance du réseau. L'interruption et la reprise d'une session s'appliquent à toutes les applications du projet Flex. Par exemple, il est impossible d'interrompre l'application d'un projet et de continuer la surveillance d'une autre.

1 Cliquez sur le bouton Interrompre dans la barre d'outils du Moniteur de réseau pour interrompre la surveillance d'une session.

2 Cliquez sur le bouton Reprendre pour continuer la surveillance de la session.

Arrêt d'une session de surveillance

Pour arrêter la surveillance d'une session, désactivez le Moniteur de réseau.

1 (Facultatif) Fermez le Moniteur de réseau.

Remarque : *la fermeture du Moniteur de réseau ne suffit pas à mettre un terme à la session de surveillance. La surveillance reste active, même si le Moniteur de réseau est fermé.*

2 Cliquez sur le bouton Activer le moniteur.

Ce bouton permet de l'activer et de le désactiver.

Remarque : *la désactivation du Moniteur de réseau s'applique à toutes les applications du projet Flex.*

Prise en charge du protocole HTTPS

Le Moniteur de réseau prend en charge la surveillance des appels HTTPS dirigés vers un serveur certifié par une autorité de certification ou par un certificat autosigné.

Pour surveiller les appels passés par protocole HTTPS, modifiez la préférence par défaut du Moniteur de réseau en activant l'option Ignorer les vérifications de sécurité SSL. Ouvrez la boîte de dialogue Préférences et accédez à Flash Builder > Moniteur de réseau.

Affichage des données du Moniteur de réseau

Le volet de gauche du Moniteur de réseau fournit des informations sur la source des données. Il affiche les informations suivantes :

- L'URL source du service de données
- Le type de service affiché :

  Par exemple RemoteService, HTTPService ou WebService.
- Le temps de demande, le temps de réponse et le temps écoulé pour la demande de données
- Le nom de l'opération appelée à partir du service de données

Le Moniteur de réseau comporte deux onglets, l'un pour l'affichage des données de demande et l'autre pour l'affichage des données de réponse.

Les données des demandes et réponses peuvent être visionnées en structure arborescente, dans une vue brute ou en affichage hexadécimal. Passez d'un affichage à l'autre en sélectionnant l'icône correspondante.

- Arborescence

  Affiche les données XML, JSON et AMF en structure arborescente. L'arborescence est la vue par défaut.
- Vue brute

  Affiche les données en cours de transfert.
- Affichage hexadécimal

  Affiche les données au format hexadécimal. Cet affichage est indiqué en particulier pour le débogage des données binaires envoyées par l'intermédiaire d'un réseau.

Par défaut, le Moniteur de réseau efface toutes les données enregistrées à chaque lancement d'une application. Vous pouvez cependant modifier ce comportement par défaut et conserver toutes les données de surveillance. Les données conservées sont celles issues de toutes les applications de tous les projets. Ouvrez la boîte de dialogue Préférences et accédez à Flash Builder > Moniteur de réseau. Désactivez l'option Effacer les entrées au démarrage.

Enregistrement des données du Moniteur de réseau

Vous pouvez enregistrer les données du Moniteur de réseau dans un fichier XML. Pour ce faire, cliquez sur le bouton Enregistrer de la vue Moniteur de réseau.

Surveillance de plusieurs applications

Vous pouvez surveiller plusieurs applications en même temps. Il existe deux scénarios possibles pour surveiller plusieurs applications :

- Surveillance de plusieurs applications d'un même projet

  Vous pouvez uniquement disposer d'un Moniteur de réseau par projet Flex. Lors de la surveillance de plusieurs applications d'un même projet, les événements de toutes les applications s'affichent dans le moniteur en fonction de l'heure à laquelle ils surviennent.

  Le moniteur ne permet pas de filtrer les événements selon une application spécifique.

- Surveillance de plusieurs applications dans différents projets

  Vous pouvez ouvrir un Moniteur de réseau pour chaque projet Flex actif. Chaque Moniteur de réseau est indépendant : il affiche uniquement les événements de son projet spécifique.

  L'interruption ou la désactivation d'un Moniteur de réseau dans un projet ne s'applique pas aux moniteurs d'autres projets.

Surveillance des applications mobiles

Le Moniteur de réseau permet d'analyser les données transmises entre une application mobile et un service de données.

Vous pouvez paramétrer le Moniteur de réseau de manière à ce qu'il surveille les applications mobiles, suivez pour cela la procédure suivante :

1 Exécutez la configuration de débogage de l'application avec le Moniteur de réseau activé.

2 Spécifiez une méthode de lancement.

- **Sur ordinateur** : sélectionnez cette méthode de lancement si vous ne disposez pas de périphérique mobile et souhaitez surveiller l'application sur votre ordinateur.

  Sélectionnez Dans le simulateur AIR afin de surveiller l'application sur un périphérique simulé qui est créé à l'aide d'ADL (AIR Debug Launcher). Le périphérique est simulé en prenant compte de la configuration de périphérique que vous sélectionnez.

- **Sur le périphérique** : sélectionnez cette méthode de lancement pour surveiller l'application sur un périphérique mobile.

  Cette méthode de lancement est habituellement utilisée afin de surveiller des applications qui accèdent au code natif du périphérique. Flash Builder peut accéder au périphérique connecté au port USB de l'ordinateur ou au réseau via une connexion Wi-Fi.

  Que vous utilisiez une connexion USB ou Wi-Fi, le Moniteur de réseau enregistre uniquement les appels si le périphérique et l'ordinateur hôte sont connectés au même réseau via Wi-Fi.

  Dans le cas où le Moniteur de réseau ne parvient pas à établir une connexion Wi-Fi, Flash Builder affiche une boîte de dialogue qui vous demande d'indiquer l'adresse IP de l'ordinateur hôte. Une fois la connexion établie, les appels réseau de l'application sont alors redirigés vers Flash Builder via le réseau Wi-Fi du périphérique.

3 Surveillez l'application mobile de la même façon que vous surveillez une application Web ou d'ordinateur qui accède aux services de données. Pour plus d'informations, voir Surveillance des applications qui accèdent aux services de données.

Limites du Moniteur de réseau

La surveillance des données du réseau présente les deux restrictions suivantes :

- Le Moniteur de réseau ne prend pas en charge les applications créées dans le cadre de projets purement ActionScript ou de bibliothèque.
- De même, il ne prend pas en charge le protocole Real Time Messaging Protocol (RTMP). Vous ne pouvez pas, par exemple, surveiller de la vidéo en flux continu.

Utilisation de projets de bibliothèque Flex

Les projets de bibliothèque vous permettent de créer des bibliothèques de code personnalisé que vous pouvez partager entre vos applications ou distribuer à d'autres développeurs. Un projet de bibliothèque génère un fichier SWC, c'est-à-dire un fichier archive pour les composants Flex ainsi que pour d'autres ressources. Par exemple, la structure Flex est contenue dans des fichiers SWC.

Lorsque vous créez un projet Flex, les fichiers SWC de la structure Flex sont ajoutés au chemin de la bibliothèque du projet. L'ajout de la bibliothèque à un projet vous permet d'utiliser ces composants dans votre application et en active les conseils de code.

Vous pouvez afficher et modifier le chemin de la bibliothèque dans la page de propriétés du chemin de génération du projet. Pour les projets Flex, sélectionnez Projet > Propriétés > Chemin d'accès à la génération Flex.

Les bibliothèques SWC constituent une méthode pratique d'assemblage et de distribution des composants, mais servent également de thèmes, l'aspect visuel des applications créées dans Flex. Un fichier de thème SWC contient un fichier CSS ainsi que toutes les ressources graphiques associées. Pour plus d'informations sur la création et l'utilisation des thèmes, voir About themes.

Configuration de bibliothèques pour vos applications

Vous pouvez utiliser les bibliothèques SWC dans vos projets des façons suivantes.

Fusionnées dans l'application : lorsque vous ajoutez un fichier SWC au chemin d'accès à la bibliothèque du projet, les composants contenus dans la bibliothèque sont disponibles pour être utilisés dans l'application. Lorsque vous générez l'application, seuls les composants de bibliothèque que vous avez utilisés dans les faits sont compilés dans le fichier SWF de l'application. En d'autres termes, tout le code de votre application est fusionné dans un seul fichier SWF. C'est la façon la plus courante et la plus simple d'utiliser les composants de bibliothèques.

Externes à l'application : vous pouvez conserver les composants de bibliothèque séparément du fichier SWF compilé pour qu'ils ne soient pas fusionnés dans le fichier. Le compilateur résout tout le code contenu dans la bibliothèque utilisé par l'application, mais ne le fusionne pas dans le fichier SWF de l'application. L'avantage de cette méthode est de réduire la taille du fichier SWF de l'application. Les composants contenus dans le fichier SWC sont extraits et chargés dans la mémoire selon les besoins, au moment de l'exécution.

Bibliothèque partagée à l'exécution : dans les projets Flex uniquement, vous pouvez également utiliser des fichiers SWC en tant que bibliothèque partagée à l'exécution (Runtime Shared Library, RSL), qui s'apparente à une bibliothèque à liaison dynamique sur d'autres plateformes. Vous pouvez utiliser des fichiers SWC en tant que bibliothèque RSL si vous disposez d'un ensemble de composants qui sont utilisés par plusieurs applications.

Le partage de composants entre applications à l'aide de ce type de bibliothèque présente plusieurs avantages. Dans un premier temps, la bibliothèque est chargée en mémoire une fois, placée dans le cache, puis mise à disposition de toutes les applications faisant appel à ces composants. Dans un second temps, les composants se trouvant dans la bibliothèque sont chargés uniquement selon les besoins, ce qui réduit le temps de démarrage de l'application, car la taille de l'application est moindre. Le problème potentiel de cette méthode est que la bibliothèque partagée à l'exécution est chargée en mémoire dans son intégralité, plutôt que les composants individuels que les applications utilisent. Pour plus d'informations sur l'utilisation des fichiers SWC en tant que RSL, voir Runtime Shared Libraries.

Création de projets de bibliothèque Flex

Lorsque vous créez un projet de bibliothèque, l'assistant Nouveau projet de bibliothèque Flex vous guide à travers les différentes étapes et vous demande d'indiquer le nom du projet, son emplacement, ainsi que le chemin de génération.

La première étape de la création d'un fichier SWC dans Flash Builder est la création d'une projet de bibliothèque Flex. Une fois le projet de bibliothèque créé, vous ajoutez des composants, spécifiez les éléments du projet de bibliothèque à inclure dans le fichier SWC, puis générez le projet afin de créer le fichier SWC.

1 Sélectionnez Fichier > Nouveau > Projet de bibliothèque Flex.

2 Saisissez un nom de projet, puis précisez les éléments suivants.

Emplacement du projet : l'emplacement par défaut est l'espace de travail actuel. Sous Windows, l'emplacement de l'espace de travail par défaut est C:\Documents and Settings*nom d'utilisateur*Adobe Flash Builder\. Sous Macintosh, l'emplacement de l'espace de travail par défaut est /Users/*nom d'utilisateur*Adobe Flash Builder/. Pour choisir un emplacement de projet différent, désactivez l'option Utiliser l'emplacement par défaut.

Configuration : vous pouvez préciser si le projet de bibliothèque Flex utilise des bibliothèques génériques ou mobiles. Les bibliothèques génériques sont utilisées dans les projets Web, d'ordinateur et mobiles.

Vous pouvez également spécifier si le projet de bibliothèque Flex doit être compatible avec Flash Catalyst. Dans ce cas, les bibliothèques génériques sont utilisées. Les bibliothèques mobiles ne sont pas prises en charge pour les projets compatibles avec Flash Catalyst.

Version SDK Flex : choisissez la version par défaut ou spécifiez-en une. Vous pouvez aussi cliquer sur le lien Configurer les SDK Flex pour ajouter, modifier ou supprimer des SDK sur la page principale Préférences.

Inclure les bibliothèques Adobe AIR : sélectionnez cette option si votre bibliothèque doit utiliser des fonctions AIR, telles que l'accès aux API AIR. Flash Builder change alors le chemin d'accès à la bibliothèque de ce nouveau projet de bibliothèque Flex de manière qu'il contienne les fichiers airglobal.swc et airframework.swc. Les projets Web Flex ne peuvent pas utiliser cette bibliothèque.

Ne sélectionnez pas cette option si vous créez une bibliothèque générique destinée à une utilisation exclusive dans une application Web, ou dans une application Web ou AIR.

3 Cliquez sur Suivant.

4 (Facultatif) Définissez les informations concernant le chemin de génération. Vous pouvez, par exemple, ajouter des dossiers au chemin source du projet qui contient les composants à inclure dans le fichier SWC. Vous pouvez également ajouter d'autres projets, dossiers ou fichiers SWC de bibliothèque à inclure dans votre projet de bibliothèque. Voir « Utilisation des fichiers SWC dans le cadre des projets » à la page 203.

5 Une fois les paramètres du projet saisis, cliquez sur Terminer.

Pour connaître les meilleures pratiques lors de la création de bibliothèques Flex, voir l'article Three points to remember when creating a library de Xavi Beumala, expert de la communauté Adobe Flex.

Création d'un projet de bibliothèque AIR

Afin de créer une bibliothèque de code AIR pour plusieurs projets Adobe AIR, générez un projet de bibliothèque Adobe AIR en faisant appel à l'assistant standard de projet de bibliothèque Flex.

1 Sélectionnez Fichier > Nouveau > Projet de bibliothèque Flex.

2 Entrez un nom de projet.

3 Activez l'option Inclure les bibliothèques Adobe AIR, puis cliquez sur Suivant.

4 Modifiez si nécessaire le chemin de génération, puis cliquez sur Terminer. Pour plus d'informations sur la création de projets de bibliothèque, voir « A propos des projets de bibliothèque » dans l'Aide de Flash Builder.

Ajout de composants à un projet de bibliothèque

Il existe plusieurs façons d'ajouter des composants à un projet de bibliothèque :

- Vous pouvez ajouter des composants, nouveaux ou existants personnalisés, des classes ActionScript et d'autres ressources au projet.
- Vous pouvez créer des liens avec des composants existant dans d'autres projets de l'espace de travail (voir « Liaison à des ressources situées en dehors de l'espace de travail du projet » à la page 75).
- Vous pouvez ajouter un dossier lié qui contient des composants au chemin source du projet de bibliothèque (voir « Ajout d'un dossier de ressources externes au chemin d'accès source » à la page 77).

Remarque : *tous les composants que vous incluez dans le projet de bibliothèque doivent être associés au projet de bibliothèque (directement ou en tant que ressources liées).*

Sélection d'éléments du projet de bibliothèque à inclure dans le fichier SWC

Pour sélectionner les éléments (composants et ressources) à inclure dans le fichier SWC, procédez comme suit :

1 Sélectionnez Projet > Propriétés > Chemin de génération de la bibliothèque Flex.

Les composants que vous avez ajoutés au projet (soit directement soit en les reliant) s'affichent sous l'onglet Classes.

2 Sélectionnez les classes de composants à inclure dans le fichier SWC.

3 (Facultatif) Sous l'onglet Ressources, sélectionnez les ressources à inclure dans le fichier SWC.

4 Une fois que vous avez effectué vos sélections, cliquez sur OK.

Création d'un projet de bibliothèque

Une fois que vous avez sélectionné les éléments à inclure dans le fichier SWC et si vous avez sélectionné l'option de génération automatique, le fichier SWC est compilé immédiatement et généré dans le dossier de sortie du projet. Si vous générez vos projets manuellement, vous pouvez créer le projet de bibliothèque quand vous le souhaitez en sélectionnant Projet > Générer le projet ou Générer tout.

La création du projet de bibliothèque génère un fichier SWC que vous pouvez partager avec d'autres applications ou utilisateurs.

Un fichier SWC est un fichier archive. Vous pouvez l'ouvrir grâce à n'importe quel utilitaire d'archivage, tel que WinZip. Les fichiers library.swf et catalog.xml sont contenus dans le fichier SWC, ainsi que les fichiers de propriétés et autre ressources incorporées.

Vous pouvez exporter la bibliothèque sous la forme d'un répertoire ouvert plutôt qu'en tant que fichier SWC. Vous exportez généralement une bibliothèque en tant que répertoire ouvert si vous prévoyez de faire appel au fichier library.swf dans le fichier SWC en tant que bibliothèque RSL.

Pour ce faire, vous devez définir les options `directory` et `output` du compilateur. Définissez l'option `output` comme le nom du répertoire à créer et l'option `directory` sur `true` pour indiquer que vous voulez ouvrir le répertoire et non un fichier SWC lorsque vous générez la bibliothèque. Pour modifier les options du compilateur, sélectionnez Projet > Propriétés > Compilateur de bibliothèque Flex et ajoutez les options au champ « Arguments supplémentaires de compilateur », par exemple :

```
-directory=true -output=myOpenDir
```

Flash Builder crée un répertoire dans le projet nommé myOpenDir et y enregistre le contenu du fichier SWC.

Utilisation des fichiers SWC dans le cadre des projets

Pour pouvoir utiliser des fichiers SWC dans vos projets Flex, vous devez les ajouter au chemin d'accès à la bibliothèque du projet. Les fichiers SWC peuvent se trouver dans le projet, dans un projet de bibliothèque Flex, dans un dossier partagé dans l'espace de travail ou dans tout autre emplacement lié au projet (en utilisant un dossier partagé qui a été ajouté au chemin source du projet, par exemple).

Lorsque vous utilisez des fichiers SWC dans des applications, certaines options de configuration déterminent s'ils sont liés de manière statique ou dynamique à l'application, s'ils sont fusionnés dans le fichier SWF de l'application ou s'ils sont externes et si leur accès se fait séparément au moment de l'exécution.

Ajout d'un fichier SWC au chemin d'accès à la bibliothèque

1. Sélectionnez un projet dans l'Explorateur de packages, puis cliquez sur Projet > Propriétés > Chemin d'accès à la génération Flex.
2. Cliquez sur l'onglet Chemin d'accès à la bibliothèque.
3. Sélectionnez l'une des options suivantes pour ajouter des fichiers SWC.

   Ajouter un projet : ajoute un projet de bibliothèque Flex.

   Ajouter un dossier SWC : permet d'ajouter un dossier contenant des fichiers SWC.

   Ajouter un fichier SWC : ajoute un fichier SWC compilé.

   Ajouter un SDK Flex : permet d'ajouter d'autres SDK Flex. Ce bouton est désactivé si le chemin d'accès à la bibliothèque du projet contient déjà un SDK Flex. Si vous supprimez le SDK Flex existant du chemin d'accès à la bibliothèque, le bouton est activé. Lorsque vous cliquez sur ce bouton, un nœud SDK Flex est ajouté, mais aucune invite ne précise lequel. Pour préciser quel SDK Flex utiliser, sélectionnez Projet > Propriétés > Compilateur Flex.
4. Saisissez ou recherchez l'emplacement du ficher SWC, du projet ou du dossier, puis sélectionnez-le. Cliquez sur OK.

   Le fichier SWC, projet de bibliothèque ou dossier est ajouté au chemin d'accès à la bibliothèque.

Fusion du fichier SWC dans le fichier SWF de l'application au moment de la compilation

1. Sélectionnez un projet dans l'Explorateur de packages, puis cliquez sur Projet > Propriétés > Chemin d'accès à la génération Flex.
2. Sous l'onglet Chemin d'accès à la bibliothèque, sélectionnez et développez l'entrée du fichier SWC pour en afficher les options.
3. Cliquez deux fois sur l'option Type de lien. La boîte de dialogue Options de l'élément de chemin d'accès à la bibliothèque apparaît.
4. Sélectionnez l'option Fusionné dans le code, puis cliquez sur OK.

Cette procédure équivaut à utiliser l'option `library-path` du compilateur.

Définition du fichier SWC en tant que fichier de bibliothèque externe

1. Sélectionnez un projet dans l'Explorateur de packages, puis cliquez sur Projet > Propriétés > Chemin d'accès à la génération Flex.
2. Sous l'onglet Chemin d'accès à la bibliothèque, sélectionnez et développez l'entrée du fichier SWC pour en afficher les options.
3. Cliquez deux fois sur l'option Type de lien. La boîte de dialogue Options de l'élément de chemin d'accès à la bibliothèque apparaît.
4. Sélectionnez l'option Externe, puis cliquez sur OK.

Cette procédure équivaut à utiliser l'option `external-library-path` du compilateur.

Utilisation du fichier SWC en tant que bibliothèque RSL

Vous pouvez externaliser les éléments partagés des fichiers SWF de vos applications en fichiers autonomes. Ces fichiers autonomes peuvent se télécharger séparément et être mis en cache sur le client. Les éléments partagés sont transférés et chargés une seule fois sur le client, mais un nombre illimité d'applications peut les utiliser au moment de l'exécution. Ces fichiers partagés sont connus sous le nom de *bibliothèques partagées à l'exécution* ou *RSL (Runtime Shared Libraries)*.

1. Sélectionnez un projet dans l'Explorateur de packages, puis cliquez sur Projet > Propriétés > Chemin d'accès à la génération Flex.
2. Sous l'onglet Chemin d'accès à la bibliothèque, sélectionnez et développez l'entrée du fichier SWC pour en afficher les options.
3. Cliquez deux fois sur l'option Type de lien ou cliquez sur Modifier. La boîte de dialogue Options de l'élément de chemin d'accès à la bibliothèque s'affiche.
4. Sélectionnez Bibliothèque RSL (Runtime Shared Library) comme type de lien.
5. Vous pouvez ajouter une URL pour spécifier où la bibliothèque SWC réside lorsque l'application est déployée. Vous pouvez également modifier une URL existante, si nécessaire.
6. Sélectionnez Charger de force le fichier RSL pour charger de force une bibliothèque RSL inutilisée au cours de la compilation.
7. Vous pouvez spécifier le domaine d'application dans lequel les bibliothèques RSL interdomaines doivent être chargées.

L'utilisation des fichiers SWC en tant que bibliothèque RSL simplifie le processus d'utilisation manuelle de bibliothèques RSL. Pour ce faire, extrayez le fichier SWF du fichier SWC et définissez les valeurs de l'option `runtime-shared-library-path` du compilateur.

Pour plus d'informations sur l'utilisation des fichiers SWC en tant que bibliothèque RSL, voir Runtime Shared Libraries dans *Utilisation d'Adobe Flex*.

Création de projets de bibliothèque ActionScript

L'assistant Nouveau projet de bibliothèque ActionScript vous guide à travers les étapes de création d'un projet de bibliothèque ActionScript et vous demande d'indiquer le nom du projet, son emplacement, ainsi que le chemin de génération.

1. Sélectionnez Fichier > Nouveau > Projet de bibliothèque ActionScript.
2. Saisissez un nom de projet, puis précisez les éléments suivants.

   Emplacement du projet : l'emplacement par défaut est l'espace de travail actuel. Sous Windows, l'emplacement de l'espace de travail par défaut est C:\Documents and Settings*nom d'utilisateur*Adobe Flash Builder\. Sous Macintosh, l'emplacement de l'espace de travail par défaut est /Users/*nom d'utilisateur*Adobe Flash Builder/. Pour choisir un emplacement de projet différent, désactivez l'option Utiliser l'emplacement par défaut.

   Version SDK Flex : choisissez la version par défaut ou spécifiez-en une. Vous pouvez aussi cliquer sur le lien Configurer les SDK Flex pour ajouter, modifier ou supprimer des SDK sur la page principale Préférences.

Utilisation d'un compilateur hérité Pour les projets ActionScript, Flash Builder utilise le compilateur ActionScript (ASC) par défaut. Si vous utilisez le SDK Adobe Flex 4.6 ou une version antérieure du SDK Flex, sélectionnez Utiliser le compilateur hérité pour utiliser l'ancien compilateur.

Inclure les bibliothèques Adobe AIR : sélectionnez cette option si votre bibliothèque doit utiliser des fonctions Adobe AIR, telles que l'accès aux API Adobe AIR. Flash Builder modifie alors le chemin d'accès à la bibliothèque de ce nouveau projet de bibliothèque ActionScript afin qu'il contienne les fichiers airglobal.swc et airframework.swc.

Les projets Web Flex ne peuvent pas utiliser cette bibliothèque. Ne sélectionnez pas cette option si vous créez une bibliothèque générique destinée à une utilisation exclusive dans une application Web, ou dans une application Web ou AIR.

3 Cliquez sur Suivant.

4 (Facultatif) Définissez les informations concernant le chemin de génération. Vous pouvez, par exemple, ajouter des dossiers au chemin source du projet qui contient les composants à inclure dans le fichier SWC. Vous pouvez également ajouter d'autres projets, dossiers ou fichiers SWC de bibliothèque à inclure dans votre projet de bibliothèque. Voir « Utilisation des fichiers SWC dans le cadre des projets » à la page 203.

5 Une fois les paramètres du projet saisis, cliquez sur Terminer.

Création de composants MXML personnalisés

Vous pouvez créer des composants personnalisés pour ajouter certaines fonctions à un composant existant ou pour créer un composant réutilisable, tel qu'une zone de recherche ou l'affichage d'un élément dans une grille de données. Vous pouvez également rédiger un nouveau composant, qui n'existe pas dans la structure Flex.

Vous pouvez distribuer les composants à l'aide des fichiers SWC. Pour plus d'informations, voir « Utilisation des fichiers SWC dans le cadre des projets » à la page 203.

Si le composant est créé principalement à partir de composants existants, l'utilisation de MXML en facilitera la définition. Toutefois, s'il s'agit d'un nouveau type de composant, définissez-le en tant que composant ActionScript. Pour plus d'informations, voir « Création d'une classe ActionScript » à la page 70.

1 Sélectionnez Fichier > Nouveau > Composant MXML.

La boîte de dialogue pour la création d'un nouveau composant s'affiche :

2 Spécifiez le dossier parent pour le fichier du composant personnalisé.

Enregistrez le fichier dans un dossier situé dans le répertoire du projet actuel ou dans le chemin source du projet actuel. Le composant s'affichera ainsi dans la vue Composants.

3 Spécifiez le nom de fichier du composant.

Il définit le nom du composant. Par exemple, le nom du composant dont le fichier est nommé `LoginBox.mxml` est LoginBox.

4 Dans le champ Basé sur, sélectionnez le composant de base de votre composant personnalisé.

Les composants personnalisés sont généralement dérivés de composants existants. Les composants de présentation personnalisés se basent généralement sur des composants Container.

A partir de Flex 4, Flash Builder suggère **spark.components.Group** comme composant de base.

Sélectionnez Parcourir pour ouvrir la boîte de dialogue Ouvrir un type, puis choisissez un composant.

Modifiez ou effacez le composant suggéré afin d'élargir la sélection dans la boîte de dialogue Ouvrir un type. Par exemple, spécifiez **spark.components.** avant de cliquer sur Parcourir.

Vous pouvez filtrer la sélection dans la boîte de dialogue Ouvrir un type en fonction de vos besoins. Pour plus d'informations sur l'utilisation de la boîte de dialogue Ouvrir un type, voir « Navigation et affichage des classes » à la page 33.

5 (Facultatif) Si vous basez le composant sur un conteneur, vous avez la possibilité de définir la largeur et la hauteur du composant.

Vous pouvez saisir des valeurs fixes ou des pourcentages. Vous pouvez également vider les deux champs. Lorsque vous créez une instance du composant, vous pouvez remplacer la largeur et la hauteur du composant dans l'instance.

6 Cliquez sur Terminer.

Flash Builder enregistre le fichier dans le dossier parent et l'ouvre dans l'éditeur.

7 Créez un composant personnalisé.

Pour plus d'informations, voir Simple MXML Components.

Vous pouvez distribuer les composants personnalisés en créant des projets de bibliothèque. Pour plus d'informations, voir « Utilisation de projets de bibliothèque Flex » à la page 200.

Définition de commentaires ASDoc pour les composants personnalisés

Vous pouvez documenter les composants personnalisés en ajoutant des commentaires ASDoc au code qui les implémente. Les commentaires ASDoc sont ensuite disponibles dans l'assistant de contenu des éditeurs MXML et ActionScript. Pour plus d'informations, voir « Assistant de contenu, Assistant rapide et Correctif rapide » à la page 16.

Ajoutez des commentaires ASDoc aux fichiers source ActionScript pour fournir la documentation de référence de l'API. Vous pouvez également ajouter des commentaires ASDoc à des éléments MXML du document. Voir ASDoc pour plus d'informations sur la création de commentaires ASDoc pour vos fichiers source.

Création de modules

Flash Builder permet de créer, d'ajouter, d'optimiser et de déboguer des modules. Pour plus d'informations sur la rédaction de code de module, voir Applications modulaires.

Les étapes suivantes décrivent la procédure de création d'un module dans Flash Builder. Après avoir créé un module, vous pouvez le compiler.

1 Dans Flash Builder, sélectionnez Fichier > Nouveau > Module MXML. La boîte de dialogue Création d'un(e) module MXML s'affiche.

2 Sélectionnez un répertoire parent pour le module. Les modules sont habituellement stockés dans le même répertoire que l'application principale, pour que les chemins relatifs des ressources partagées soient identiques.

Les modules étant exécutables, ils doivent se trouver dans le dossier source du projet.

3 Entrez un nom de fichier pour le module, par exemple MyModule.

4 Entrez les propriétés Largeur, Hauteur et Présentation.

5 Indiquez si vous souhaitez ou non optimiser le module.

L'optimisation d'un module pour une application signifie que les classes utilisées par l'application sont exclues du module, ce qui peut contribuer à réduire le volume de téléchargement des fichiers SWF. Pour plus d'informations, voir « Optimisation de modules dans Flash Builder » à la page 211.

- Optimiser pour l'application

  Si vous activez cette option, spécifiez l'application pour laquelle le module doit être optimisé.

- Ne pas optimiser

  Si vous activez cette option, toutes les classes sont incluses dans le module, qu'elles soient définies ou non dans l'application principale, améliorant ainsi les performances de la compilation d'incrémentation. Vous pouvez en outre charger le module dans n'importe quelle application, toutes ses dépendances y étant compilées.

6 Cliquez sur Terminer.

Flash Builder ajoute un nouveau fichier de module MXML dans le projet.

Compilation des modules

Dans Flash Builder, vous pouvez exécuter le module comme une application ou générer le projet du module. Si les modules figurent dans le même projet que l'application, alors Flash Builder compile automatiquement les fichiers SWF pour tous les modules du projet lorsque vous exécutez l'application. Les fichiers SWF sont ensuite chargés dans l'application lors de l'exécution.

Il est impossible d'exécuter le fichier SWF basé sur un module comme un fichier SWF autonome ou de le charger dans une fenêtre de navigateur. Il doit être chargé par une application en tant que module. Si vous exécutez le module dans Flash Builder pour le compiler, fermez Adobe Flash Player ou la fenêtre du navigateur et ignorez les erreurs éventuelles. Les modules ne doivent pas être sollicités directement par Flash Player ou par l'intermédiaire d'un navigateur.

Les fichiers SWF du module et le fichier SWF de l'application principale se trouvent généralement dans le même répertoire. Flash Builder compile toutefois les modules en même temps que l'application, quel que soit leur emplacement. Les modules peuvent se situer dans le même répertoire que l'application ou dans des sous-répertoires.

Vous pouvez également créer un projet Flex ou ActionScript distinct pour chaque module ou groupe de modules. Il en résulte une meilleure maîtrise de la compilation des modules, les options de compilateur de chaque projet pouvant différer de celles de l'application ou des autres modules. Cela permet également de compiler le ou les projets d'un module sans compiler l'application. Cette approche requiert toutefois la compilation manuelle de chaque module avant de compiler l'application. Une manière de procéder consiste à compiler simultanément tous les projets ouverts dans Flash Builder.

Si vous compilez les modules indépendamment de l'application principale, veillez à inclure ou exclure les informations de débogage, selon que vous souhaitez déboguer l'application et les modules. Pour plus d'informations, voir « Débogage de modules dans Flash Builder » à la page 211.

Le flux de travail dans Flash Builder est conçu pour l'association de modules avec une même application. Pour utiliser des modules dans plusieurs applications, encapsulez éventuellement le code dans une classe ou un composant de bibliothèque à inclure dans un module simple pour chaque application. Contrairement aux bibliothèques, les modules ne sont pas destinés à une utilisation dans le code commun à plusieurs applications.

Utilisation de plusieurs projets pour des modules

Lors de la définition de l'architecture de votre projet, vous pouvez décider d'inclure des modules dans votre projet d'application, de créer un projet distinct pour chaque module ou de créer un projet distinct pour l'ensemble des modules.

Utilisation d'un projet pour chaque module

L'utilisation d'un projet pour chaque module présente les avantages suivants :

- Les projets de module peuvent figurer à un emplacement quelconque de l'espace de travail.
- Les projets de module peuvent disposer de paramètres de compilateur distincts, tel qu'un chemin d'accès à la bibliothèque personnalisé.

  Ils peuvent utiliser l'option de compilateur `load-externs` pour supprimer les dépendances qui se recoupent.

L'utilisation d'un projet pour chaque module présente les inconvénients suivants :

- La mémoire nécessaire augmente avec le nombre de projets.
- L'existence de nombreux projets dans un même espace de travail peut augmenter son encombrement.
- Par défaut, lorsque vous compilez l'application, tous les projets de module ne sont pas compilés, même s'ils ont subi des modifications.
- Pour optimiser la taille du fichier du module, appliquez manuellement les options de compilateur `load-externs` et `link-report`.

Utilisation d'un projet pour l'ensemble des modules

Une approche semblable consiste à utiliser un même projet pour l'ensemble des modules, en conservant l'application dans son propre projet distinct. Certains inconvénients, liés à l'utilisation d'un même projet à la fois pour l'application et les modules, subsistent. Cette approche apporte toutefois de nombreux avantages découlant de l'utilisation d'un projet distinct pour chaque module.

L'utilisation d'un projet pour l'ensemble des modules présente les avantages suivants :

- Le projet de module peut figurer à un emplacement quelconque de l'espace de travail.
- Vous pouvez compiler séparément les modules ou l'application, sans nécessité de compiler les deux simultanément.
- Le projet de module peut utiliser l'option de compilateur `load-externs` pour supprimer les dépendances qui se recoupent.

L'utilisation d'un projet de module pour l'ensemble des modules présente les inconvénients suivants :

- Tous les modules du projet de module doivent utiliser les mêmes paramètres de compilateur, tel que le chemin d'accès à la bibliothèque.
- Par défaut, lorsque vous compilez l'application, le projet de module n'est pas compilé, même s'il a subi des modifications.
- Pour optimiser la taille du fichier du module, appliquez manuellement les options de compilateur `load-externs` et `link-report`.

Création de projets pour des modules

Lors de la création d'un projet séparé pour les modules, vous modifiez le dossier de sortie du projet de module sur un répertoire utilisé par l'application. Vous supprimez également la création de fichiers d'enveloppe.

Création d'un projet distinct pour les modules dans Flash Builder

1. Créez un projet principal.
2. Créez un projet pour le ou les modules.
3. Dans le menu contextuel du projet du module, sélectionnez Propriétés. La boîte de dialogue Propriétés s'ouvre.

4 Sélectionnez l'option Chemin de génération Flex.

5 Modifiez le dossier de sortie de manière telle à ce qu'il pointe vers le répertoire des modules MainProject, comme illustré dans l'exemple ci-dessous :

```
${DOCUMENTS}\MainProject\assets
```

Cette modification redirige la sortie de compilation du module vers le répertoire des ressources du projet d'application (MainProject). Dans l'application principale, vous pouvez définir la propriété `url` de ModuleLoader sur les fichiers SWF du répertoire des ressources. La valeur de cette propriété est relative au dossier de sortie.

6 Cliquez sur OK pour enregistrer vos modifications.

7 Ouvrez à nouveau les propriétés du projet et sélectionnez l'option Compilateur Flex.

8 Désélectionnez l'option Générer le fichier d'enveloppe HTML afin d'éviter que le projet du module ne génère les fichiers d'enveloppe HTML. Ces fichiers ne sont généralement utilisés que pour l'application et sont inutiles pour les modules.

9 Cliquez sur OK pour appliquer les modifications.

Compilation de projets pour les modules

La compilation de plusieurs projets dans Flash Builder est une opération courante. Il vous faut tout d'abord choisir l'ordre dans lequel vous souhaitez compiler les projets. Vous pouvez ensuite compiler tous les projets en même temps.

Compilation simultanée de tous les projets dans Flash Builder

❖ Dans le menu principal, sélectionnez Projets > Générer tout.

Flex génère alors l'ensemble des projets de l'espace de travail. Les fichiers d'application sont ajoutés au dossier de sortie de chaque projet. Si vous n'avez pas choisi l'enregistrement automatique des fichiers avant le début d'une génération, vous êtes invité à enregistrer les fichiers.

Pour modifier l'ordre de génération, utilisez la boîte de dialogue Ordre de génération. La modification de l'ordre de génération n'est pas toujours nécessaire. Les projets utilisant des modules ne doivent être compilés qu'avant l'exécution de l'application de projet principale, et non lors de sa compilation. Dans la plupart des cas, l'ordre de génération par défaut ne doit pas être modifié.

Si toutefois vous souhaitez éliminer les dépendances qui se recoupent, vous pouvez modifier l'ordre de génération afin de compiler d'abord l'application principale. L'option de compilateur `link-report` permet alors de générer le rapport de liaison. Lorsque vous compilez les modules, l'option de compilateur `load-externs` permet d'utiliser le rapport de liaison généré par l'application principale. Pour plus d'informations sur la réduction de la taille des modules, voir « Optimisation de modules dans Flash Builder » à la page 211.

Modification de l'ordre de génération des projets

1 Ouvrez la boîte de dialogue Préférences et sélectionnez Général > Espace de travail > Ordre de génération.

La boîte de dialogue Ordre de génération apparaît.

2 Désélectionnez la case à cocher Utiliser l'ordre de génération par défaut.

3 Utilisez les boutons Haut et Bas pour reclasser les projets de la liste Ordre de génération. Vous pouvez également utilisez le bouton Supprimer un projet pour supprimer les projets qui ne font pas partie de l'application principale ou qui ne sont pas des modules utilisés par l'application. Dans ce cas, le projet supprimé sera généré mais uniquement après tous les autres projets de la liste.

4 Cliquez sur OK.

5 Modifiez si nécessaire l'ordre de génération, puis cliquez sur OK.

En présence de dépendances entre des projets distincts de l'espace de travail, le compilateur détermine automatiquement l'ordre dans lequel les projets sont générés afin de résoudre correctement les dépendances.

Si vous utilisez un projet distinct pour chaque module, vous pouvez compiler un seul module à la fois. Cela permet de gagner du temps par rapport à la compilation simultanée de tous les projets, ou par rapport à la compilation d'un seul projet contenant tous les fichiers de modules et d'application.

Compilation du projet d'un seul module

1. Cliquez avec le bouton droit de la souris sur le fichier MXML du module dans le projet du module.
2. Sélectionnez Exécuter l'application. La fenêtre du lecteur ou du navigateur tente d'exécuter le module après sa compilation. Vous pouvez fermer la fenêtre du lecteur ou du navigateur et ignorer les messages d'erreur éventuels qui s'affichent lors de l'exécution. Les modules ne sont pas destinés à une exécution directe dans le lecteur ou dans un navigateur.

Ajout de modules au projet

Vous serez amené, dans certaines situations, à utiliser des modules ne figurant pas dans le projet de l'application principale. Le module peut provenir d'un projet distinct, vous pouvez alors utiliser des options de configuration personnalisées. Vous pouvez également vouloir partager le module entre plusieurs applications. Ajoutez le code source du module au chemin source de l'application, puis ajoutez le module à la liste de modules de l'application afin de pouvoir l'utiliser dans le projet.

Ajout d'un module préalablement compilé au projet

1. Sélectionnez l'application principale dans l'Explorateur de packages Flex.
2. Sélectionnez Projet > Propriétés > Chemin d'accès à la génération Flex pour ajouter la source du module au chemin source du projet d'application.
3. Cliquez sur le bouton Ajouter un dossier et accédez au chemin source du module. Cliquez sur OK pour sélectionner le module. Répétez cette procédure pour chaque module externe à ajouter au projet de l'application.
4. Cliquez à nouveau sur OK pour enregistrer vos modifications.
5. Sélectionnez Projet > Propriétés > Modules Flex pour ajouter le module à la liste de modules de l'application. La boîte de dialogue Modules Flex liste l'ensemble des modules ajoutés au projet actif ou figurant dans ce projet. Lors de la création initiale d'un projet, cette boîte de dialogue est vide.
6. Cliquez sur le bouton Ajouter. La boîte de dialogue Ajouter un module s'affiche.
7. Utilisez le bouton Parcourir ou entrez l'emplacement du fichier MXML du module dans le champ Source. Tous les modules figurant dans le chemin source du projet peuvent être ajoutés à l'aide de cette boîte de dialogue.
8. Sélectionnez l'une des options Taille du fichier SWF du module pour activer ou désactiver l'optimisation de module. Si vous sélectionnez Optimiser pour l'application, Flash Builder compile le module pour l'application sélectionnée en excluant toutes les classes définies dans l'application principale. Il peut s'agir de classes de structure ou de classes personnalisées. Lorsque vous sélectionnez cette option, le module ne peut pas être utilisé dans une autre application car la liste de classes exclues peut être différente. Pour plus d'informations, voir « Optimisation de modules dans Flash Builder » à la page 211.
9. Cliquez sur OK pour enregistrer vos modifications. Flash Builder ajoute le module à la liste des modules disponibles dans le projet de l'application.

Optimisation de modules dans Flash Builder

Dans Flash Builder, vous ne sélectionnez généralement qu'une seule application par rapport à laquelle le module doit être optimisé lors de sa création initiale ou de son ajout au projet. Si vous décidez par la suite de changer l'application visée par l'optimisation du module, ou de ne pas optimiser le module, vous pouvez modifier les propriétés du module au sein du projet. Pour plus d'informations, voir Reducing module size.

Cette procédure part du principe que le module et l'application se trouvent dans le même projet Flash Builder. Si les modules se situent dans un projet distinct, ajoutez manuellement les options de compilateur `load-externs` et `link-report`.

1. Cliquez avec le bouton droit de la souris sur le projet de l'application dans l'Explorateur de package Flex et sélectionnez Propriétés. La boîte de dialogue Propriétés du projet s'ouvre.
2. Dans le volet de gauche, sélectionnez Modules Flex.
3. Sélectionnez le module dans la liste et cliquez sur le bouton Modifier. La boîte de dialogue Modifier le module s'affiche.
4. Pour supprimer l'optimisation, sélectionnez l'option Ne pas optimiser sous Taille du fichier SWF du module.
5. Afin d'optimiser le module pour une autre application, sélectionnez la nouvelle application dans la liste déroulante Optimiser pour l'application.
6. Cliquez sur OK.

Pour optimiser davantage la taille du fichier d'un module, vous pouvez supprimer les informations de débogage. Lors de la génération d'un module dans Flash Builder, les informations de débogage y sont incluses par défaut. En les supprimant, vous pouvez réduire la taille du module. Pour obtenir des instructions sur la procédure de suppression des informations de débogage des modules, voir « Débogage de modules dans Flash Builder » à la page 211.

Débogage de modules dans Flash Builder

Pour déboguer des modules et des applications, intégrez des informations de débogage dans les fichiers SWF lors de leur compilation. Dans Flash Builder, exécutez l'application, puisque les informations de débogage sont incluses par défaut. A la ligne de commande, définissez l'option de compilateur `debug` sur `true`. La valeur par défaut est `true`, mais si vous l'avez désactivée dans un fichier de configuration, veillez à le remplacer.

Par défaut, Flash Builder génère un seul fichier SWF comprenant des symboles de débogage. Les instructions Déboguer et Exécuter fonctionnent donc toutes les deux lors de l'exécution d'une application utilisant des modules dans Flash Builder. Toutefois, l'inclusion de symboles de débogage dans le fichier SWF d'un module augmente sa taille. Pour exclure les symboles de débogage avant le déploiement, désactivez le débogage des modules de l'application. Pour ce faire, exportez la version validée des modules en sélectionnant Projet > Exporter vers une version validée.

Pour exclure les informations de débogage des fichiers SWF dans Flash Builder, vous pouvez définir l'option `debug` sur `false` dans la zone de texte des arguments de compilateur supplémentaires, ou générer des fichiers SWF sans débogage en les exportant à l'aide de la fonctionnalité Exporter vers une version validée. Cela inclut les modules, s'ils figurent dans le projet actif.

Si vous créez un projet distinct pour les modules, vous pouvez activer ou désactiver le débogage pour l'ensemble du projet sans modifier les paramètres de l'application principale.

Pour déboguer un module, vous devez également déboguer son application de chargement. Le débogueur Flex ne se connecte pas aux applications dépourvues d'informations de débogage, même si les modules chargés par l'application les contiennent. En d'autres termes, vous ne pouvez pas exclure les informations de débogage de l'application si vous souhaitez déboguer le module qu'elle charge.

Lors de l'utilisation de modules dans une application Adobe AIR, le fichier SWF du module doit figurer dans le même répertoire que le fichier SWF de l'application, ou dans l'un de ses sous-répertoires.

Intégration de Flex avec les applications HTML

La fonction Créer un composant Ajax Bridge permet de générer un code JavaScript et un fichier d'enveloppe HTML facilitant l'utilisation d'une application créée dans Flex en JavaScript dans une page HTML. Elle peut être utilisée avec la bibliothèque Flex Ajax Bridge JavaScript qui permet d'exposer une application au script dans le navigateur Web. Le code JavaScript généré est léger puisqu'il est destiné à exposer la fonctionnalité déjà fournie par Flex Ajax Bridge. Pour plus d'informations sur Flex Ajax Bridge, voir Flex Ajax Bridge.

La fonction Créer un composant Ajax Bridge génère un code proxy JavaScript spécifique aux API d'application que vous voulez appeler à partir de JavaScript. Vous pouvez générer du code pour toute classe ActionScript ou application MXML dans un projet Flash Builder.

Pour les fichiers d'application MXML, vous pouvez générer un code pour tous les éléments suivants ou une partie d'entre eux dans le code MXML :

- Liste des éléments hérités pouvant s'étendre de façon non récursive
- Propriétés publiques, incluant les balises avec les propriétés `id`
- Constantes publiques
- Fonctions publiques incluant les classes définies dans la ligne

Pour les classes ActionScript, vous pouvez générer un code pour tous les éléments suivants ou une partie d'entre eux :

- Liste des éléments hérités
- Propriétés publiques (pour chaque propriété, affichage d'une méthode get et set)
- Constantes publiques
- Méthodes publiques

Dans un répertoire que vous spécifiez, la fonction Créer un composant Ajax Bridge génère des fichiers *.js et *.html correspondant aux applications MXML et aux classes ActionScript sélectionnées pour la génération. Elle place une copie de la bibliothèque Flex Ajax Bridge (fabridge.js) dans un sous-répertoire du répertoire de génération de code. Cette fonction génère également des fichiers d'aide MXML dans le répertoire `src` du projet ; ces fichiers sont utilisés pour finaliser la génération du code JavaScript.

Génération de code Ajax Bridge

1. Cliquez avec le bouton droit sur un projet dans l'Explorateur de packages Flex et sélectionnez Créer un composant Ajax Bridge.
2. Dans la boîte de dialogue Création d'un composant Ajax Bridge, sélectionnez les applications MXML et les classes ActionScript pour lesquelles vous souhaitez générer un code JavaScript. Vous pouvez sélectionner la case à cocher de niveau supérieur pour inclure l'intégralité du projet ou vous pouvez sélectionner des membres spécifiques.
3. Spécifiez le répertoire dans lequel générer les classes de proxy.
4. Cliquez sur OK pour générer le code. L'exemple suivant illustre un fichier .js généré pour une application affichant des images :

```
*
    * You should keep your JavaScript code inside this file as light as possible,
    * and keep the body of your Ajax application in separate *.js files.
    *
    * Do make a backup of your changes before regenerating this file. (Ajax Bridge
    * display a warning message.)
    *
    * For help in using this file, refer to the built-in documentation in the Ajax Bridge
application.
    *
    */


/**
    * Class "DisplayShelfList"
    * Fully qualified class name: "DisplayShelfList"
    */
function DisplayShelfList(obj) {
    if (arguments.length > 0) {
        this.obj = arguments[0];
    } else {
        this.obj = FABridge["b_DisplayShelfList"].
            create("DisplayShelfList");
    }
}

// CLASS BODY
// Selected class properties and methods
DisplayShelfList.prototype = {

    // Fields form class "DisplayShelfList" (translated to getters/setters):

    // Methods form class "DisplayShelfList":

    getAngle : function() {
        return this.obj.getAngle();
    },

    setAngle : function(argNumber) {
        this.obj.setAngle(argNumber);
    },

    setCurrentPosition : function(argNumber) {
        this.obj.setCurrentPosition(argNumber);
    },

    setSelectedIndex : function(argNumber) {
        this.obj.setSelectedIndex(argNumber);
    },

    setPercentHeight : function(argNumber) {
        this.obj.setPercentHeight(argNumber);
    },

    setPercentWidth : function(argNumber) {
        this.obj.setPercentWidth(argNumber);
    },
```

```
    DisplayShelfList : function() {
        return this.obj.DisplayShelfList();
    },

    setFirst : function(argNumber) {
        this.obj.setFirst(argNumber);
    },

    setFormat : function(argString) {
        this.obj.setFormat(argString);
    },

    setLast : function(argNumber) {
        this.obj.setLast(argNumber);
    }
}


/**
    * Listen for the instantiation of the Flex application over the bridge.
    */
FABridge.addInitializationCallback("b_DisplayShelfList", DisplayShelfListReady);


/**
    * Hook here all of the code that must run as soon as the DisplayShelfList class
    * finishes its instantiation over the bridge.
    *
    * For basic tasks, such as running a Flex method on the click of a JavaScript
    * button, chances are that both Ajax and Flex have loaded before the
    * user actually clicks the button.
    *
    * However, using DisplayShelfListReady() is the safest way, because it lets
    * Ajax know that involved Flex classes are available for use.
    */
function DisplayShelfListReady() {

    // Initialize the root object. This represents the actual
    // DisplayShelfListHelper.mxml Flex application.
    b_DisplayShelfList_root = FABridge["b_DisplayShelfList"].root();


    // YOUR CODE HERE
    // var DisplayShelfListObj = new DisplayShelfList();
    // Example:
    // var myVar = DisplayShelfListObj.getAngle ();
    // b_DisplayShelfList_root.addChild(DisplayShelfListObj);

}
```

5 Modifiez les fichiers .js générés. Dans la fonction `xxxReady()` des fichiers .js générés, ajoutez le code à exécuter dès que la classe correspondante termine son instanciation sur Ajax Bridge. Selon l'application, le code par défaut peut être généré dans cette méthode. Le code en gras de l'exemple suivant illustre le code d'initialisation personnalisé pour l'exemple d'application d'image.

```
...
function DisplayShelfListReady() {

    // Initialize the root object. This represents the actual
    // DisplayShelfListHelper.mxml Flex application.
    b_DisplayShelfList_root = FABridge["b_DisplayShelfList"].root();


       // Create a new object.
       DisplayShelfListObj = new DisplayShelfList();
       // Make it as big as the application.
       DisplayShelfListObj.setPercentWidth(100);
       DisplayShelfListObj.setPercentHeight(100);
       //Set specific attributes.
        DisplayShelfListObj.setFirst(1);
        DisplayShelfListObj.setLast(49);
        DisplayShelfListObj.setFormat("./photos400/photo%02d.jpg");
        //Add the object to the DisplayList hierarchy.
        b_DisplayShelfList_root.addChild(DisplayShelfListObj.obj);
}
```

6 Modifiez les fichiers .html générés. Dans la partie des pages HTML contenant le texte d'indication d'emplacement de la description, remplacez le texte par le code HTML à utiliser pour accéder à l'application à partir de la page HTML. Par exemple, ce code ajoute des boutons pour contrôler l'exemple d'application d'image :

```
<h2>Test controls</h2>
<ul>
<li><input type="button" onclick="DisplayShelfListObj.setCurrentPosition(0)"
value="Go to first item"/>
</li>
<li><input type="button" onclick="DisplayShelfListObj.setCurrentPosition(3)"
value="Go to fourth item"/>
</li>
<li><input type="button" onclick="DisplayShelfListObj.setSelectedIndex(0)"
value="Go to first item (with animation)"/>
</li>
<li><input type="button" onclick="DisplayShelfListObj.setSelectedIndex(3)"
value="Go to fourth item (with animation)"/>
</li>
<li><input type="button" onclick="alert(DisplayShelfListObj.getAngle())"
value="Get photo angle"/>
</li>
<li><input type="button" onclick="DisplayShelfListObj.setAngle(15);"
value="Set photo angle to 15&deg;"/>
</li>
</ul>
```

7 Ouvrez la page HTML dans un navigateur Web pour exécuter l'application.

Chapitre 9 : Développement d'applications mobiles dans Flash Builder

Flash Builder 4.7 fournit des fonctions qui prennent en charge le développement d'applications pour les périphériques mobiles. Les fonctionnalités mobiles dans Flash Builder rendent le développement d'une application mobile basée sur ActionScript ou Flex aussi simple que le développement d'une application de bureau ou Web.

Important : *Flash Builder 4.7 vous permet d'écrire le code d'applications mobiles et ensuite de le tester et déboguer. Pour concevoir votre propre interface mobile à l'aide des flux de conception de Flash Builder, utilisez Flash Builder 4.6. Pour plus d'informations sur l'utilisation des flux de conception dans Flash Builder 4.6, voir Développement d'applications mobiles avec Flex 4.6 et Flash Builder 4.6.*

Différences entre le développement d'applications mobiles, de bureau et de navigateur

Navigateur : déploie l'application en tant que fichier SWF à utiliser dans Flash Player s'exécutant dans un navigateur.

De bureau : déploie une application Adobe AIR autonome pour un ordinateur de bureau, tel qu'un ordinateur exécutant Windows ou un Macintosh.

Mobile : déploie une application Adobe AIR autonome pour un périphérique mobile, tel qu'un téléphone ou une tablette.

Les environnements d'exécution de Flash Player et Adobe AIR sont similaires. Vous pouvez effectuer la plupart des opérations dans l'un ou l'autre des environnements d'exécution. Adobe AIR vous permet non seulement de déployer des applications autonomes en dehors d'un navigateur, mais vous fournit en outre une intégration étroite à la plateforme hôte. Cette intégration permet par exemple d'accéder au système de fichiers du périphérique, de créer et d'utiliser des bases de données SQL locales, etc.

Restrictions liées à l'utilisation de Spark et des composants MX dans une application mobile

Utilisez le composant Spark défini lors de la création des applications mobiles dans Flex. Les composants Spark sont définis dans les packages `spark.components.*`. Toutefois, pour des raisons de performance ou parce que tous les composants Spark n'ont pas d'habillage pour le thème Mobile, les applications mobiles ne prennent pas en charge la totalité des composants Spark.

A l'exception des contrôles graphiques MX et du contrôle MX Spacer, les applications mobiles ne prennent pas en charge l'ensemble de composants MX défini dans les `packages mx.*`.

Le tableau ci-dessous répertorie les composants que vous pouvez utiliser, que vous ne pouvez pas utiliser ou qui requièrent un soin particulier pour être utilisés dans une application mobile :

Composant	Composant	Utilisation dans une application mobile ?	Remarques
Spark ActionBar Spark BusyIndicator Spark Callout Spark CalloutButton Spark DateSpinner Spark SpinnerList Spark SpinnerListContainer	Spark TabbedViewNavigator Spark TabbedViewNavigatorApplication Spark ToggleSwitch Spark View Spark ViewMenu Spark ViewNavigator Spark ViewNavigatorApplication	Oui	Ces nouveaux composants prennent en charge les applications mobiles.
Spark Button Spark CheckBox Spark DataGroup Spark Group/HGroup/VGroup/TileGroup Spark Image/BitmapImage Spark Label	Spark List Spark RadioButton/RadioButtonGroup Spark SkinnableContainer Spark Scroller Spark TextArea Spark TextInput	Oui	La plupart de ces composants disposent d'habillages pour le thème Mobile. Label, Image et BitmapImage peuvent être utilisés, même s'ils ne disposent pas d'un habillage mobile. Certains conteneurs de présentation Spark, comme Group et ses sous-classes, n'ont pas d'habillage. Par conséquent, vous pouvez les utiliser dans une application mobile.
Autres composants Spark habillables		Déconseillé	Les composants Spark habillables autres que ceux figurant dans la liste ci-dessus sont déconseillés car ils n'ont pas d'habillage pour le thème Mobile. Si le composant n'a pas d'habillage pour le thème Mobile, vous pouvez en créer un pour votre application.
Spark DataGrid	Spark RichEditableText Spark RichText	Déconseillé	Ces composants sont déconseillés pour des raisons de performance. Même si vous pouvez les utiliser dans une application mobile, cela risque d'affecter les performances. En ce qui concerne le contrôle DataGrid, les performances sont basées sur la quantité de données dont vous effectuez le rendu. Quant aux contrôles RichEditableText et RichText, leur performance est basée sur la quantité de texte et sur le nombre de contrôles présents dans l'application.

Composant	Composant	Utilisation dans une application mobile ?	Remarques
Composants MX autres que Spacer et les graphiques		Non	Les applications mobiles ne prennent pas en charge les composants MX tels que MX Button, CheckBox, List ou DataGrid. Ces composants correspondent aux composants Flex 3 dans les packages mx.controls.* et mx.containers.*.
MX Spacer		Oui	Spacer n'utilise pas d'habillage, de sorte qu'il peut être utilisé dans une application mobile.
Composants graphiques MX		Oui, mais avec des implications pour les performances .	Vous pouvez utiliser les contrôles graphiques MX, tels que AreaChart et BarChart, dans une application mobile. Les contrôles graphiques MX sont dans les packages mx.charts.*. Toutefois, leurs performances sur un périphérique mobile peuvent être amoindries selon la taille et le type des données graphiques. Par défaut, Flash Builder n'inclut pas les composants MX dans le chemin d'accès aux bibliothèques des projets mobiles. Pour utiliser les composants graphiques MX dans une application, ajoutez les fichiers mx.swc et charts.swc à votre chemin d'accès à la bibliothèque.

Les fonctionnalités Flex ci-dessous ne sont pas prises en charge dans les applications mobiles :

- Pas de prise en charge des opérations de glisser-déposer.
- Pas de prise en charge du contrôle ToolTip.
- Pas de prise de charge des bibliothèques RSL.

Considérations liées aux performances avec les applications mobiles

En raison des contraintes de performances qui pèsent sur les périphériques mobiles, certains aspects du développement d'applications mobiles diffèrent de ceux du développement d'applications de navigateur et d'ordinateur. Les considérations liées aux performances comprennent :

- **Rédaction des rendus d'élément en ActionScript**

  Pour les applications mobiles, le défilement dans les listes doit être aussi performant que possible. Rédigez des rendus d'élément en ActionScript pour bénéficier de performances optimales. Il est possible de rédiger les rendus d'élément en MXML, mais les performances de vos applications peuvent en souffrir.

  Flex fournit deux rendus d'élément qui sont optimisés à utiliser dans une application mobile : spark.components.LabelItemRenderer et spark.components.IconItemRenderer. Pour plus d'informations sur ces rendus d'élément, voir Using a mobile item renderer with a Spark list-based control.

  Pour plus d'informations sur la création de rendus d'élément personnalisés en ActionScript, voir Custom Spark item renderers. Pour plus d'informations sur les différences entre les rendus d'élément mobiles et de bureau, voir Differences between mobile and desktop item renderers.

- **Utilisation d'ActionScript et de graphiques FXG compilés ou de bitmaps pour développer des habillages personnalisés**

  Les habillages mobiles livrés avec Flex 4.6 sont écrits en ActionScript avec des graphiques FXG compilés afin d'offrir des performances optimales. Vous pouvez rédiger les habillages en MXML, mais les performances de votre application peuvent en souffrir, selon le nombre de composants qui utilisent les habillages MXML. Pour des performances optimales, rédigez les habillages en ActionScript et utilisez des graphiques FXG compilés. Pour plus d'informations, voir Habillage Spark et Graphiques FXG et MXML.

- **Utilisation de composants d'entrée de texte utilisant StageText**

  Utilisez les paramètres par défaut lorsque vous utilisez des composants d'entrée de texte du type de TextInput ou TextArea. Ces contrôles utilisent StageText comme mécanisme sous-jacent à la saisie de texte, il vient se connecter aux classes d'entrée de texte natives. Vous profitez ainsi de performances accrues et d'un accès aux fonctions natives, telles que la correction automatique, la mise en majuscules automatique, la restriction du texte ainsi que les claviers logiciels personnalisés.

  L'utilisation de StageText présente certains inconvénients, comme l'impossibilité de faire défiler la vue dans laquelle se trouve les contrôles. Il ne vous est pas non plus possible d'utiliser des polices intégrées ou d'attribuer des dimensions personnalisées aux contrôles basés sur StageText. Si nécessaire, vous pouvez utiliser des contrôles d'entrée de texte basés sur la classe TextField.

- **Soyez prudent lorsque vous utilisez des composants graphiques MX dans une application mobile.**

  Vous pouvez utiliser les contrôles graphiques MX, tels que les contrôles AreaChart et BarChart, dans une application mobile. Toutefois, ils peuvent nuire aux performances selon la taille et le type des données graphiques.

Le blogueur Nahuel Foronda a rédigé une série d'articles sur Mobile ItemRenderer en ActionScript.

Le blogueur Rich Tretola a créé une entrée de cookbook sur la création d'une liste avec un élément ItemRenderer pour une application mobile.

Flux de travail pour la création d'applications mobiles

Création d'une application Android dans Flash Builder

Voici un flux de travail général permettant de créer une application Flex mobile pour la plateforme Google Android. Ce flux de travail suppose que vous avez déjà conçu votre application mobile.

L'expert Adobe, Mike Jones, partage certains des enseignements qu'il a pu tirer du développement de son jeu multi-plateforme, Mode, et il vous donne 10 conseils à utiliser dans le développement multi-plateforme.

Configuration requise pour Adobe AIR

Les projets Flex Mobile et ActionScript Mobile requièrent Adobe AIR 2.6 ou version supérieure. Les projets mobiles peuvent être exécutés sur des périphériques physiques prenant en charge Adobe AIR 2.6 ou version supérieure.

Vous pouvez installer Adobe AIR 2.6 ou version supérieure uniquement sur les périphériques Android pris en charge exécutant Android 2.2 ou version supérieure. Pour obtenir la liste complète des périphériques Android pris en charge, voir Certified Devices. Consultez également la configuration minimale requise pour exécuter Adobe AIR sur les périphériques Android à la page Configuration requise.

Remarque : *si vous ne disposez pas d'un périphérique prenant en charge Adobe AIR 2.6 ou version supérieure, vous pouvez utiliser Flash Builder pour lancer et déboguer des applications mobiles sur votre ordinateur.*

Chaque version du SDK Flex inclut la version requise d'Adobe AIR. Si vous avez installé des applications mobiles sur un périphérique exécutant une version antérieure du SDK Flex, désinstallez Adobe AIR du périphérique. Flash Builder installe la version correcte d'Adobe AIR lorsque vous exécutez ou déboguez une application mobile sur un périphérique.

Création d'une application

1 Dans Flash Builder, sélectionnez Fichier > Nouveau > Projet Flex Mobile.

Un projet Flex Mobile est un type particulier de projet Adobe AIR. Suivez les invites de l'assistant de nouveau projet comme vous le feriez pour tout autre projet Adobe AIR dans Flash Builder. Pour plus d'informations, voir « Projets Flex Mobile » à la page 63.

Pour définir des préférences mobiles spécifiques à Android, voir « Définition des préférences de projet mobile » à la page 225.

Lorsque vous créez un projet Flex Mobile, Flash Builder génère les fichiers suivants pour le projet :

- *ProjectName*.mxml

  Le fichier d'application par défaut pour le projet.

  Par défaut, Flash Builder attribue à ce fichier le même nom qu'au projet. Si le nom du projet contient des caractères non autorisés par ActionScript, Flash Builder nomme ce fichier Main.mxml. Le fichier MXML contient la balise Spark Application de base du projet. La balise Spark Application peut être ViewNavigatorApplication ou TabbedViewNavigatorApplication.

  Généralement, vous n'ajoutez pas directement de contenu au fichier d'application par défaut, à l'exception du contenu ActionBar affiché dans toutes les vues. Pour ajouter du contenu au contrôle ActionBar, définissez les propriétés `navigatorContent`, `titleContent` ou `actionContent`.

- *ProjectName*HomeView.mxml

  Le fichier représentant la vue initiale du projet. Flash Builder place le fichier dans un package de vues. L'attribut `firstView` de la balise ViewNavigatorApplication dans *ProjectName*.mxml spécifie ce fichier comme vue d'ouverture par défaut de l'application.

Vous pouvez aussi créer un projet mobile uniquement en ActionScript. Voir « Création de projets ActionScript Mobile » à la page 64.

2 Disposez le contenu de la vue initiale de votre application.

Vous pouvez également ajouter toutes les vues que vous souhaitez inclure dans votre application.

Utilisez l'éditeur de code dans Flash Builder pour ajouter des composants à une vue.

Utilisez uniquement des composants pris en charge par Flash pour le développement mobile. Pour plus d'informations, voir « Restrictions liées à l'utilisation de Spark et des composants MX dans une application mobile » à la page 216.

3 (Facultatif) Ajoutez des rendus d'élément optimisés pour les applications mobiles pour les composants List.

Adobe fournit IconItemRenderer, un rendu d'élément basé sur ActionScript à utiliser avec les applications mobiles.

4 Configurez les configurations de lancement pour exécuter et déboguer l'application.

Vous pouvez exécuter ou déboguer l'application sur l'ordinateur ou sur un périphérique.

Une configuration de lancement est requise pour exécuter ou déboguer une application à partir de Flash Builder. La première fois que vous exécutez ou déboguez une application mobile, Flash Builder vous invite à configurer une configuration de lancement.

Lors de l'exécution ou du débogage d'une application mobile sur un périphérique, Flash Builder installe l'application sur le périphérique.

Voir « Test et débogage d'une application mobile sur un périphérique » à la page 241.

5 Exportez l'application en tant que package d'installation.

Utilisez Exporter vers une version validée pour créer des packages qui pourront être installés sur des périphériques mobiles. Flash Builder crée des packages pour la plateforme sélectionnée pour l'exportation. Voir « Exportation de packages Android APK pour publication » à la page 247.

Création d'une application iOS dans Flash Builder

Voici un flux de travail général permettant de créer une application mobile destinée à la plateforme Apple iOS.

1 Avant de commencer à créer l'application, veillez à suivre la procédure indiquée dans « Processus de développement Apple iOS à l'aide de Flash Builder » à la page 235.

2 Dans Flash Builder, sélectionnez Fichier > Nouveau > Projet Flex Mobile.

Sélectionnez la plateforme cible Apple iOS et définissez les paramètres du projet mobile.

Suivez les invites de l'assistant de nouveau projet comme vous le feriez dans tout autre assistant de création de projets dans Flash Builder. Pour plus d'informations, voir « Création d'une application » à la page 220.

Vous pouvez aussi créer un projet mobile uniquement en ActionScript. Pour plus d'informations, voir « Création de projets ActionScript Mobile » à la page 64.

3 Configurez les configurations de lancement pour exécuter et déboguer l'application. Vous pouvez exécuter ou déboguer l'application sur l'ordinateur ou sur un périphérique connecté.

Pour plus d'informations, voir « Débogage d'une application sur un périphérique Apple iOS » à la page 243.

4 Exportez l'application vers l'App Store d'Apple ou déployez l'application de package iOS (IPA) sur un périphérique.

Pour plus d'informations, voir « Exportation de packages Apple iOS pour publication » à la page 249 et « Installation d'une application sur un périphérique Apple iOS » à la page 246.

Création d'une application BlackBerry Tablet OS dans Flash Builder

Installation du plug-in BlackBerry pour Flash Builder

Research In Motion (RIM) fournit un plug-in permettant d'utiliser Flash Builder afin de créer des applications et des packages d'applications Flex et ActionScript pour le système d'exploitation BlackBerry® Tablet OS.

Ce plug-in n'est pas inclus dans Flash Builder. Vous pouvez l'installer depuis le site de mises à jour RIM préconfiguré à l'aide de l'assistant d'installation de logiciel de la façon suivante :

1 Dans le menu de Flash Builder, sélectionnez Aide > Installer le nouveau logiciel.

2 Cliquez sur la liste déroulante Utiliser avec et sélectionnez BlackBerry plugin for Flash Builder 4.7.

3 Suivez les étapes de l'assistant d'installation de logiciel pour effectuer l'installation.

Création, déploiement et signature de l'application pour BlackBerry Tablet OS

Le flux de travail général suivant permet de créer des applications pour BlackBerry Tablet OS.

1 Avant de commencer la création d'une application mobile, installez le SDK BlackBerry Tablet OS pour Adobe AIR depuis le site de développement d'applications BlackBerry Tablet OS (en anglais).

   Le SDK BlackBerry Tablet OS pour Adobe AIR fournit des API qui permettent de créer des applications Flex et ActionScript basées sur AIR.

   Pour plus d'informations sur l'installation du SDK BlackBerry Tablet OS, voir le guide de prise en main de BlackBerry Tablet OS (en anglais).

2 Pour créer une application Adobe AIR basée sur Flex, dans Flash Builder, sélectionnez Fichier > Nouveau > Projet Flex Mobile.

   Suivez les invites de l'assistant de nouveau projet comme vous le feriez pour tout autre projet Adobe AIR dans Flash Builder. Veillez à sélectionner BlackBerry Tablet OS comme plateforme cible.

3 Pour créer une application Adobe AIR basée sur ActionScript, dans Flash Builder, sélectionnez Fichier > Nouveau > Projet ActionScript Mobile.

   Suivez les invites de l'assistant de nouveau projet comme vous le feriez pour tout autre projet Adobe AIR dans Flash Builder. Veillez à sélectionner BlackBerry Tablet OS comme plateforme cible.

Pour plus d'informations sur la signature, la création de package et le déploiement de l'application, voir le guide de développement du SDK BlackBerry Tablet OS pour Adobe AIR publié par RIM.

Vous trouverez des ressources supplémentaires sur le développement BlackBerry Tablet OS provenant d'Adobe et de RIM sur la page Adobe Developer Connection.

Environnement de développement

La création d'une application pour un périphérique mobile ne se résume pas au redimensionnement d'une application d'ordinateur pour un écran de plus petite taille. Vous pouvez créer différentes interfaces utilisateur appropriées pour chaque facteur de forme, tout en partageant le code d'accès au modèle et aux données sous-jacent entre les projets mobiles, de navigateur et d'ordinateur.

Utilisation d'extensions natives

Les extensions natives vous permettent d'intégrer des fonctions de plateforme native dans votre application mobile.

Une extension native contient des classes ActionScript ainsi que du code natif. L'implémentation de code natif vous permet d'accéder aux fonctions spécifiques au périphérique qui sont inaccessibles au travers de classes ActionScript pures. Par exemple, l'accès à la fonction de vibration du périphérique.

L'implémentation de code natif peut être définie comme le code s'exécutant à l'extérieur de l'environnement d'exécution Adobe AIR. Vous définissez l'implémentation de code natif et les classes ActionScript spécifiques à la plateforme dans l'extension. Les classes de l'extension ActionScript accèdent aux données et en échangent avec le code natif à l'aide de la classe ActionScript ExtensionContext.

Les extensions sont spécifiques à la plateforme matérielle d'un périphérique. Vous avez la possibilité de créer des extensions spécifiques pour chaque plateforme ou de créer une extension unique ciblant plusieurs plateformes. Vous pouvez, par exemple, créer une extension native ciblant les plateformes Android et iOS. Les périphériques mobiles suivants prennent en charge les extensions natives :

- Périphériques Android exécutant Android 2.2 ou version ultérieure
- Périphériques iOS exécutant iOS 4.0 ou version ultérieure

Pour plus d'informations sur la création des extensions natives multi-plateformes, voir Developing Native Extensions for Adobe AIR.

Pour consulter un ensemble d'exemples d'extensions natives, partagés par Adobe et la communauté, voir Extensions natives pour Adobe AIR.

Création de package d'extension native

Pour fournir votre extension native aux développeurs d'applications, il vous faut créer un package de tous les fichiers nécessaires sous la forme d'un fichier ANE (ActionScript Native Extension) en suivant la procédure suivante :

1. Générez la bibliothèque ActionScript de l'extension dans un fichier SWC.
2. Générez les bibliothèques natives de l'extension. Dans le cas où l'extension doit prendre en charge différentes plateformes, générez une bibliothèque pour chacune des plateformes cibles.
3. Créez un certificat signé pour votre extension. Si celle-ci n'est pas signée, Flash Builder affiche un avertissement lorsque vous ajoutez l'extension à votre projet.
4. Créez un fichier descripteur d'extension.
5. Incluez éventuellement des ressources externes pour l'extension, telles que des images.
6. Créez le package d'extension à l'aide de l'outil Air Developer (ADT). Pour plus d'informations, voir la documentation d'Adobe AIR.

Pour plus d'informations sur la création de package d'extensions natives multi-plateformes, voir Developing Native Extensions for Adobe AIR.

Ajout d'extensions natives à un projet

Vous pouvez inclure un fichier ANE (ActionScript Native Extension) dans le chemin d'accès à la génération du projet de la même façon que vous incluez un fichier SWC.

1. Lors de la création d'un projet Flex Mobile dans Flash Builder, sélectionnez l'onglet Extensions natives au niveau de la page de paramètres Chemins d'accès à la génération.

   Il vous est également possible d'ajouter des extensions à partir de la boîte de dialogue Propriétés du projet en sélectionnant Chemin d'accès à la génération Flex.

2. Accédez au fichier ANE ou au dossier contenant les fichiers ANE à ajouter au projet. Lors de l'ajout d'un fichier ANE, l'ID d'extension est ajouté au fichier descripteur de l'application du projet (*Nom du projet*-app.xml) par défaut.

Flash Builder affiche un symbole d'erreur pour l'extension ajoutée dans les cas suivants :

- La version de l'environnement d'exécution Adobe AIR est postérieure à la version de l'environnement d'exécution de l'application.
- L'extension n'inclut pas toutes les plateformes sélectionnées qui sont ciblées par l'application.

Remarque : *vous pouvez créer une extension native ActionScript ciblant plusieurs plateformes. Pour tester une application incluant ce fichier ANE sur votre ordinateur de développement à l'aide du simulateur Adobe AIR, assurez-vous que le fichier ANE prend en charge la plateforme de l'ordinateur. Par exemple, pour tester l'application sous Windows à l'aide du simulateur Adobe AIR, assurez-vous que le fichier ANE prend en charge Windows.*

Inclusion des extensions natives ActionScript dans un package d'application

Lorsque vous utilisez la fonction Exporter vers une version validée pour exporter l'application mobile, par défaut les extensions utilisées dans le projet sont incluses.

Suivez la procédure suivante pour modifier la sélection par défaut :

1. Dans la boîte de dialogue Exporter vers une version validée, sélectionnez l'onglet Extensions natives sous Paramètres de groupement.
2. Les fichiers d'extensions natives ActionScript référencés dans votre projet sont répertoriés, indiquant si le fichier ANE est utilisé dans le projet ou non.

   Si le fichier ANE est utilisé dans le projet, il est sélectionné par défaut dans le package de l'application.

   S'il est inclus dans le projet sans être utilisé, le compilateur ne reconnaît alors pas le fichier ANE. Il n'est dans ce cas pas inclus dans le package de l'application. Pour inclure le fichier ANE dans le package de l'application, procédez comme suit :

   a. Dans la boîte de dialogue Propriétés du projet, sélectionnez Paquet de génération Flex, puis sélectionnez la plateforme requise.

   b. Sélectionnez les extensions que vous souhaitez inclure dans le package de l'application.

Configuration externe d'un SDK iOS

Le SDK Adobe AIR contient une version intégrée du SDK iOS, spécifique à la version Adobe AIR. Par exemple, Adobe AIR 3.1 et Adobe AIR 3.4 contiennent respectivement le SDK iOS4 et le SDK iOS5.

Si vous souhaitez utiliser les fonctions d'une version du SDK iOS supérieure à la version intégrée, vous devez configurer la version prise en charge du SDK iOS de manière externe.

Les versions suivantes du SDK iOS sont prises en charge :

Version Adobe AIR	Version intégrée du SDK iOS	Version prise en charge du SDK iOS pour la configuration externe
AIR 3.1	iOS4	iOS5
AIR 3.4	iOS5	IOS6

Vous pouvez configurer un SDK iOS à l'aide de l'outil ADT (AIR Developer Tool). Flash Builder vous permet de sélectionner l'emplacement du SDK iOS externe pris en charge dans la boîte de dialogue Paramètres de groupement. Une fois l'emplacement du SDK iOS sélectionné, il est ensuite transmis à l'aide de la commande `-platformsdk` de l'outil ADT.

Remarque : *Avec Adobe AIR 3.1, vous pouvez configurer un SDK iOS externe sous Mac uniquement. Avec Adobe AIR 3.4, vous pouvez le configurer sous Windows et Mac.*

Masquage des symboles de bibliothèque ANE

Lors de l'utilisation de plusieurs extensions natives dans votre application iOS, il se peut que des noms de symbole communs soient utilisés dans les différentes extensions natives. Ces conflits de noms de symbole peuvent engendrer des problèmes lors de la création de packages (un message d'erreur s'affiche). Dans certains cas, l'application peut même se bloquer en cours d'exécution.

Pour éviter cette erreur, effectuez les étapes suivantes :

1 Dans la boîte de dialogue des propriétés du projet, sélectionnez Paquet de génération > Apple iOS.

2 Sous l'onglet Extensions natives, sélectionnez Masquer les symboles de bibliothèque ANE.

Lorsque vous sélectionnez cette option, la commande ADT package `-hideAneLibSymbols` est définie sur `yes`, et les problèmes de conflits non désirés sont résolus. Pour plus d'informations, voir Commande ADT package.

Vérifier si les conflits de symboles engendrent un blocage de l'application

Les blocages d'application ne sont pas tous dus à des conflits de symboles. Pour déterminer si un blocage a été engendré uniquement par un symbole dupliqué dans les fichiers d'objet, créez un package de l'application à partir de la ligne de commande à l'aide de la commande ADT -package. Vérifiez ensuite si l'avertissement s'affiche à nouveau.

Remarque : *si l'IPA est créé, Flash Builder n'affiche pas d'avertissement même si le créateur de packages a identifié des avertissements. Pour vérifier les avertissements qui s'affichent pendant la création de packages, vous devez créer un package de l'application à partir de la ligne de commande.*

Pour obtenir les détails de la ligne de commande, effectuez les étapes suivantes :

1 Cliquez sur Personnaliser le lancement dans la boîte de dialogue Configurations Exécuter/Déboguer, ou dans la section des paramètres de package de la boîte de dialogue Exporter vers une version validée.

2 Dans la boîte de dialogue Personnaliser les paramètres de lancement, sélectionnez Afficher la commande.

Définition des préférences de projet mobile

Définition des configurations de périphériques

Flash Builder utilise des configurations de périphérique pour afficher les aperçus des tailles d'écran de périphérique sur ordinateur à l'aide d'AIR Debug Launcher (ADL). Voir « Test et débogage d'une application mobile sur l'ordinateur » à la page 239.

Pour définir des configurations de périphériques, ouvrez la fenêtre Préférences et sélectionnez Flash Builder > Configuration des périphériques.

Flash Builder fournit plusieurs configurations de périphérique par défaut. Vous pouvez ajouter, modifier ou supprimer des configurations de périphérique supplémentaires. Vous ne pouvez pas modifier les configurations par défaut fournies par Flash Builder.

Cliquez sur le bouton Restaurer les valeurs par défaut pour rétablir les configurations de périphériques par défaut, sans supprimer les configurations que vous avez ajoutées. De plus, si vous avez ajouté une configuration de périphérique avec un nom correspondant à celui d'une configuration par défaut, Flash Builder remplace la configuration ajoutée par les paramètres par défaut.

Les configurations de périphérique contiennent les propriétés suivantes :

Propriété	Description
Nom du périphérique	Nom unique du périphérique.
Plateforme	Plateforme du périphérique. Sélectionnez une plateforme dans la liste des plateformes prises en charge.
Taille plein écran	Largeur et hauteur de l'écran du périphérique.
Taille d'écran utilisable	Taille standard d'une application sur le périphérique. Il s'agit de la taille prévue d'une application lancée en mode non-plein écran, en tenant compte du dispositif chrome du système, comme la barre d'état.
Pixels par pouce	Pixels par pouce sur l'écran du périphérique.

Choix des plateformes cibles

Flash Builder prend en charge les plateformes cibles en fonction du type d'application.

Pour sélectionner une plateforme, ouvrez la fenêtre Préférences et sélectionnez Flash Builder > Plateformes cibles.

Pour tous les plug-ins tiers, voir la documentation correspondante.

Choix d'un modèle d'application

Lorsque vous créez une application mobile, vous pouvez faire un choix parmi les modèles d'application suivants :

Vide : utilise la balise Spark Application en tant qu'élément d'application de base.

Utilisez cette option pour créer une application personnalisée sans utiliser la navigation standard dans les vues.

Application basée sur une vue : utilise la balise Spark ViewNavigatorApplication en tant qu'élément d'application de base pour créer une application dotée d'une vue unique.

Vous pouvez spécifier le nom de la vue initiale.

Application à onglets : utilise la balise Spark TabbedViewNavigatorApplication comme élément d'application de base pour créer une application basée sur des onglets.

Pour ajouter un onglet, saisissez le nom de l'onglet et cliquez sur Ajouter. Vous pouvez modifier l'ordre des onglets en cliquant sur Haut et Bas. Pour supprimer un onglet d'une application, sélectionnez un onglet et cliquez sur Supprimer.

Le nom de la vue correspond au nom de l'onglet auquel est ajouté le mot « View ». Par exemple, si vous nommez un onglet FirstTab, Flash Builder génère une vue nommée FirstTabView.

Pour chaque onglet que vous créez, un nouveau fichier MXML est généré dans le package « Views ».

Remarque : *le nom du package n'est pas configurable à l'aide de l'assistant Projet Flex Mobile.*

Les fichiers MXML sont générés conformément aux règles suivantes :

- Si le nom de l'onglet est un nom de classe ActionScript valide, Flash Builder génère le fichier MXML en ajoutant le suffixe « View » au nom de l'onglet.
- Si le nom de l'onglet n'est pas un nom de classe valide, Flash Builder modifie le nom de l'onglet en supprimant les caractères non valides et en insérant des caractères initiaux valides. Si le nom modifié n'est pas acceptable, Flash Builder modifie le nom de fichier MXML en le remplaçant par « ViewN », où N correspond à la position de la vue, en commençant par N=1.

Brent Arnold, expert Flex certifié par Adobe, a créé un didacticiel vidéo sur l'utilisation du modèle d'application à onglets.

Choix des autorisations d'une application mobile

Lorsque vous créez une application mobile, vous pouvez spécifier ou modifier les droits par défaut d'une plateforme cible. Les autorisations sont spécifiées au moment de la compilation et ne peuvent pas être modifiées dans l'environnement d'exécution.

Commencez par sélectionner la plateforme cible, puis définissez les autorisations pour chaque plateforme, le cas échéant. Vous pourrez modifier les autorisations par la suite dans le fichier XML descripteur de l'application.

Des plug-ins tiers fournissent une prise en charge de plateformes supplémentaires pour les projets Flex et ActionScript. Pour les autorisations spécifiques aux plateformes, voir la documentation du périphérique.

Autorisations pour la plateforme Google Android

Pour la plateforme Google Android, vous pouvez définir les autorisations suivantes :

INTERNET : autorise les demandes réseau et le débogage à distance.

L'autorisation INTERNET est sélectionnée par défaut. Si vous désélectionnez ce droit, vous ne pouvez pas déboguer votre application sur un périphérique.

WRITE_EXTERNAL_STORAGE : autorise l'écriture sur un périphérique externe.

Sélectionnez ce droit pour permettre à l'application d'écrire sur une carte mémoire externe du périphérique.

READ_PHONE_STATE : coupe le son au cours d'un appel entrant.

Sélectionnez ce droit pour permettre à l'application de couper le son au cours des appels téléphoniques. Par exemple, vous pouvez sélectionner ce droit si votre application lit les sons en arrière-plan.

ACCESS_FINE_LOCATION : autorise l'accès à un emplacement GPS.

Sélectionnez ce droit pour permettre à l'application d'accéder aux données GPS à l'aide de la classe Geolocation.

DISABLE_KEYGUARD and WAKE_LOCK : interdit la mise en veille du périphérique.

Sélectionnez ce droit pour empêcher le périphérique de se mettre en veille à l'aide des paramètres de la classe SystemIdleMode.

CAMERA : permet d'accéder à une caméra.

Sélectionnez ce droit pour permettre à l'application d'accéder à une caméra.

RECORD_AUDIO : permet d'accéder à un microphone.

Sélectionnez ce droit pour permettre à l'application d'accéder à un microphone.

ACCESS_NETWORK_STATE and ACCESS_WIFI_STATE : autorise l'accès aux informations concernant les interfaces réseau associées au périphérique.

Sélectionnez ce droit pour permettre à l'application d'accéder aux informations réseau à l'aide de la classe NetworkInfo.

Pour plus d'informations sur la définition des propriétés des applications mobiles, voir la Documentation Adobe AIR.

Autorisations pour la plateforme Apple iOS

La plateforme Apple iOS utilise la validation dans l'environnement d'exécution pour les autorisations au lieu des autorisations prédéfinies. En d'autres termes, si une application souhaite accéder à une fonction spécifique de la plateforme Apple iOS qui requiert des droits d'utilisateur, une fenêtre contextuelle apparaît, demandant l'autorisation.

Choix des paramètres de plateforme

Les paramètres de plateforme vous permettent de sélectionner une gamme de périphériques cibles. Selon la plateforme sélectionnée, vous pouvez choisir le périphérique cible ou une gamme de périphériques cibles. Vous pouvez sélectionner un périphérique spécifique ou tous les périphériques pris en charge par la plateforme.

Des plug-ins tiers fournissent une prise en charge de plateformes supplémentaires pour les projets Flex et ActionScript. Pour les paramètres spécifiques aux plateformes, voir la documentation du périphérique.

Paramètres de plateforme pour la plateforme Google Android

il n'existe pas de paramètres spécifiques à la plateforme Google Android.

Paramètres de plateforme pour la plateforme Apple iOS

Pour un projet Flex Mobile ou un projet ActionScript Mobile, vous pouvez spécifier les périphériques cibles suivants pour la plateforme Apple iOS :

iPhone/iPod Touch : les applications utilisant cette gamme cible sont répertoriées comme compatibles uniquement avec les périphériques iPhone et iPod Touch dans l'App Store d'Apple.

iPad : les applications utilisant cette gamme cible sont répertoriées comme compatibles uniquement avec les périphériques iPad dans l'App Store d'Apple.

Tous : les applications utilisant cette gamme cible sont répertoriées comme compatibles avec les périphériques iPhone/iPod Touch et iPad dans l'App Store d'Apple. Il s'agit de l'option par défaut.

Choix des paramètres d'application

Réorientation automatique : fait pivoter l'application lorsque l'utilisateur tourne le périphérique. Lorsque ce paramètre n'est pas activé, votre application apparaît toujours dans la même orientation.

Plein écran : affiche l'application en mode plein écran sur le périphérique. Lorsque ce paramètre est activé, la barre d'état du périphérique n'apparaît pas au-dessus de l'application. Votre application remplit tout l'écran.

Si vous souhaitez cibler votre application vers plusieurs types de périphériques aux densités d'écran variables, sélectionnez Redimensionner automatiquement l'application pour différentes densités d'écran. La sélection de cette option redimensionne automatiquement l'application et gère les changements de densité des périphériques, le cas échéant. Voir « Définition du redimensionnement d'application » à la page 228.

Définition du redimensionnement d'application

Vous utilisez le redimensionnement d'application mobile pour créer une application mobile unique compatible avec des périphériques de taille d'écran et de densité différentes.

Les écrans des périphériques mobiles possèdent des densités d'écran, ou valeurs PPP(points par pouce), variables. Vous pouvez spécifier une valeur de PPP de 160, 240 ou 320, selon la densité d'écran du périphérique cible. Lorsque vous activez le redimensionnement automatique, Flex optimise la façon dont il affiche l'application pour la densité d'écran de chaque périphérique.

Par exemple, supposez que vous spécifiez une valeur PPP cible de 160 et activez le redimensionnement automatique. Lorsque vous exécutez l'application sur un périphérique doté d'une valeur PPP de 320, Flex redimensionne automatiquement l'application par un facteur 2. En d'autres termes, Flex agrandit tout de 200 %.

Pour spécifier la valeur PPP cible, définissez-la comme propriété `applicationDPI` de la balise `<s:ViewNavigatorApplication>` ou `<s:TabbedViewNavigatorApplication>` dans le fichier de l'application principale :

```
<s:ViewNavigatorApplication xmlns:fx="http://ns.adobe.com/mxml/2009"
    xmlns:s="library://ns.adobe.com/flex/spark"
    firstView="views.HomeView"
    applicationDPI="160">
```

Si vous choisissez de ne pas utiliser le redimensionnement automatique de votre application, vous devez gérer manuellement les changements de densité pour votre présentation, selon les besoins. Toutefois, Flex adapte les habillages à la densité de chaque périphérique.

Prise en charge de plusieurs cibles mobiles

Lors du développement d'applications mobiles pour plusieurs plateformes et périphériques, Flash Builder vous permet de spécifier plusieurs configurations de génération pour chaque cible de lancement.

Pour les projets mobiles, vous pouvez créer une cible de génération personnalisée avec le contenu du package spécifique, tel que les extensions natives, les configurations de certificat et les SDK de différentes plateformes.

Pour plus d'informations sur la spécification de propriétés pour une cible de génération, voir « Propriétés des projets pour la création de packages de projets mobiles » à la page 118.

Création d'une cible de génération personnalisée

1. Dans l'Explorateur de packages, sélectionnez le projet.
2. Sélectionnez Projet > Propriétés dans le menu principal ou sélectionnez Propriétés dans le menu contextuel.
3. Dans la boîte de dialogue des propriétés du projet, sélectionnez Paquet de génération Flex et sélectionnez la plateforme cible dont les paramètres doivent être affichés ou modifiés.
4. Pour créer une cible personnalisée, cliquez sur Créer une cible personnalisée et indiquez les détails de la nouvelle cible. Cliquez ensuite sur Créer. La cible créée apparaît dans la liste des cibles disponibles.
5. Spécifiez le certificat numérique à utiliser pour signer l'application mobile, le contenu du package, les extensions natives, les autorisations et les habilitations (le cas échéant).
6. Cliquez sur OK.

Sélection d'une cible de génération personnalisée dans la configuration de lancement

Lorsque vous spécifiez une configuration de lancement pour l'exécution ou le débogage d'une application, Flash Builder sélectionne par défaut les paramètres de cible de génération applicables. Par exemple, supposons que vous créez une cible de génération d1 pour exécuter l'application sur un périphérique iOS. Lorsque vous créez une configuration de lancement pour déboguer l'application sur un simulateur, la cible de génération sélectionnée par défaut est la cible d1. Toutefois, vous pouvez modifier la sélection par défaut et sélectionner une autre cible de génération.

Exportation de l'application vers une cible personnalisée

Vous pouvez exporter une application mobile vers une cible personnalisée prise en charge en plus des plateformes standard prises en charge.

Important : *vous pouvez utiliser la fonction Exporter vers une version validée pour lancer une cible personnalisée uniquement sur un périphérique.*

Lorsque vous sélectionnez Projet > Exporter vers une version validée, toutes les cibles basées sur périphérique configurées pour le projet apparaissent.

S'il existe plusieurs cibles de génération pour un périphérique, les noms des cibles s'affichent au format `nomplateforme-nomcible`. Par exemple, si deux cibles de génération sont configurées pour lancer l'application sur un périphérique iOS, les noms des cibles s'affichent de la manière suivante : `AppleiOS-nomcible1` et `AppleiOS-nomcible2`. Le nom du package d'application créé comporte le nom de la cible correspondante.

Exemple de création et d'exportation d'une application vers plusieurs cibles de génération
Vous développez une application pour la plateforme Apple iOS et disposez de plusieurs fichiers ActionScript Native Extension (ANE), comme indiqué ci-dessous :

- Extension native n1 pour un périphérique iOS (iPhone-ARM)
- Extension native n2 pour le simulateur iOS (iPhone-x86)

L'extension native n1 utilise les fonctionnalités du SDK iOS4 et l'extension native n2 utilise les fonctionnalités du SDK iOS5.

Lors de la création du package de votre application mobile, il vous faut créer différents packages de l'application pour le périphérique et le simulateur en sélectionnant des ANE différents pour chaque package :

- Cible de génération d1 pour lancer l'application sur un périphérique iOS avec l'extension native n1 sélectionnée
- Cible de génération s1 pour lancer l'application sur le simulateur iOS avec l'extension native n2 sélectionnée

Pour ce faire, procédez comme suit :

1. Dans la page des propriétés du projet, sélectionnez Paquet de génération Flex > Apple iOS.
2. Cliquez sur Personnaliser pour spécifier la configuration de la première génération.
3. Définissez le nom de la cible de génération comme étant d1, sélectionnez la base de la cible de génération Périphérique et cliquez sur Créer.
4. Cliquez sur Appliquer.
5. Cliquez à nouveau sur Personnaliser pour spécifier la configuration de la deuxième génération.
6. Définissez le nom de la cible de génération comme étant s1, sélectionnez la base de la cible de génération Simulateur et cliquez sur Créer.
7. Les cibles personnalisées créées apparaissent dans la liste des cibles disponibles.
8. Pour sélectionner les extensions natives requises lors de l'exportation de l'application, effectuez les étapes suivantes :
   - Sélectionnez la cible d1 (périphérique) et l'extension native n1 dans la section des extensions natives. Cliquez sur Appliquer.
   - Sélectionnez la cible s1 (simulateur) et l'extension native n2 dans la section des extensions natives. Cliquez sur Appliquer.
9. Cliquez sur OK.

Vous pouvez exporter l'application vers plusieurs cibles de génération en suivant les étapes ci-dessous :

1. Sélectionnez Projet > Exporter vers une version validée.
2. Dans la boîte de dialogue Exporter vers une version validée, les cibles personnalisées que vous avez créées apparaissent en tant que Apple iOS-d1 et Apple iOS-s1. Vous pouvez sélectionner plusieurs cibles pour l'exportation de votre application.
3. Pour chaque plateforme cible sélectionnée, vous pouvez spécifier l'extension native à utiliser, le certificat numérique pour la signature de l'application et le contenu du package.

Remarque : *les extensions natives utilisées dans le projet sont incluses par défaut dans le package de l'application. Vous pouvez toutefois modifier la sélection par défaut si nécessaire. Pour plus d'informations, voir «* *Inclusion des extensions natives ActionScript dans un package d'application » à la page 224.*

4 Avant d'exporter l'application, vous pouvez ajouter ou modifier des valeurs de paramètre de lancement, telles que -target, -keystore et autres. Cliquez sur Personnaliser le lancement pour personnaliser les valeurs des paramètres de lancement.

5 Cliquez sur Terminer.

Personnalisation des paramètres de lancement

Flash Builder permet d'afficher et de modifier les paramètres de lancement d'une application avant de la lancer, de la déboguer ou de l'exporter.

Flash Builder calcule les valeurs des paramètres de lancement de façon interne et permet, selon le besoin, d'ajouter, de supprimer ou de modifier des valeurs par l'intermédiaire de la boîte de dialogue Personnaliser les paramètres de lancement.

Flash Builder permet également d'afficher les détails de commande des paramètres de lancement pour la création de packages d'application mobile en cliquant sur Afficher la commande dans la boîte de dialogue Personnaliser les paramètres de lancement.

Vous pouvez accéder à la boîte de dialogue Personnaliser les paramètres de lancement à partir des boîtes de dialogue Configurations Exécuter/Déboguer, Exporter vers une version validée ou Propriétés de création de package de génération.

Remarque : *dans la boîte de dialogue Propriétés de création de package de génération, vous pouvez afficher les paramètres mais pas leurs valeurs. Vous pouvez toutefois modifier les valeurs des paramètres.*

Personnalisation des valeurs des paramètres ADL et ADT

Vous pouvez personnaliser les valeurs des paramètres avant de créer le package de l'application à l'aide d'AIR Developer Tool (ADT), ou avant de lancer l'application à l'aide d'AIR Debug Launcher (ADT).

Lors du lancement ou du débogage d'une application, vous pouvez :

- modifier des valeurs de paramètres personnalisables ;
- ajouter du contenu pour la création de package ;
- ajouter un paramètre ;
- supprimer un paramètre existant en lui assignant une valeur vide.

Les valeurs de paramètres personnalisables sont des valeurs que vous avez configurées ou des valeurs que Flash Builder calcule de façon interne.

Remarque : *pour ADL et pour les exportations de versions validées de packages de type autre que mobile, l'ajout de paramètres n'est pas pris en charge.*

Pour modifier la valeur d'un paramètre, procédez comme suit :

1 Cliquez sur Personnaliser le lancement dans la boîte de dialogue Configurations Exécuter/Déboguer.

2 Dans la boîte de dialogue Personnaliser les paramètres de lancement, cliquez sur Modifier.

3 Sélectionnez les valeurs des paramètres à modifier. Vous pouvez choisir d'ajouter une valeur personnalisée à la fin ou au début d'un argument particulier. Flash Builder accepte les valeurs personnalisées après leur validation uniquement. Une valeur de paramètre vide, par exemple, n'est pas permise.

Il est recommandé de créer des cibles de génération différentes pour une utilisation sur ordinateur de bureau et sur tablette, et d'en modifier les paramètres en conséquence. Les valeurs personnalisées sont conservées pour la cible et le nom de génération sélectionnés.

Exemple

```
<buildTarget airDownloadURL="" androidSettingsVersion="2"
    buildTargetName="device" isCurrent="true"
    platformId="com.adobe.flexide.multiplatform.android.platform">
    <newPackagingParams>
        <Parameter location="placeAtEnd" name="-sfsfgfd" value="sdfdgdg"/>
    </newPackagingParams>.
</buildTarget>
```

Remarque : *lors de la création d'un projet, ces valeurs ne sont pas spécifiées dans les paramètres du projet. Elles ne sont ajoutées aux paramètres du projet que si elles sont applicables.*

Il existe plusieurs méthodes de personnalisation des valeurs de paramètres de lancement :

Activation du profilage avec Adobe Scout pour les packages iOS

Pour configurer le profilage par Adobe® Scout (nom de code précédent : « Project Monocle ») de votre package IPA, spécifiez un nouveau paramètre ADT lors de la création du fichier IPA.

Pour ce faire, procédez comme suit :

1 Cliquez sur Personnaliser le lancement dans la boîte de dialogue Configurations Exécuter/Déboguer ou dans la boîte de dialogue Exporter vers une version validée.

   Remarque : *lors du lancement ou du débogage de l'application, assurez-vous de sélectionner la méthode de lancement sur le périphérique.*

2 Dans la boîte de dialogue Personnaliser les paramètres de lancement, cliquez sur Ajouter un paramètre.

3 Spécifiez le nom de paramètre `-sampler.` et l'emplacement Placer avant `-provisioning-profile`.

Les valeurs spécifiées pour une cible de génération sont utilisées dans les différentes méthodes de création de package. Si vous souhaitez spécifier des valeurs de paramètres différentes pour la boîte de dialogue Configurations Exécuter/Déboguer et la boîte de dialogue Exporter vers une version validée, créez plusieurs cibles de génération. Pour plus d'informations, voir « Prise en charge de plusieurs cibles mobiles » à la page 229.

Lancement d'ANE avec des liens symboliques à l'aide d'ADL

Lors de l'utilisation d'AIR Debug Launcher (ADL) pour le lancement d'une application avec des ANE, Flash Builder extrait automatiquement les ANE vers un répertoire à l'aide de la commande -extDir. Cette extraction est effectuée en arrière-plan.

Si toutefois l'ANE contient des liens symboliques, Flash Builder ne peut pas extraire les ANE de façon automatique et le lancement d'ADL échoue. Afin d'éviter cet échec de lancement, procédez manuellement à l'extraction des fichiers ANE vers un dossier de votre ordinateur et spécifiez le chemin complet du dossier comme valeur pour le paramètre -extDir.

Par exemple, si vous extrayez les fichiers ANE vers le dossier /User/monnom/moncheminane, modifiez le paramètre -extDir de la façon suivante :

1 Cliquez sur Personnaliser le lancement dans la boîte de dialogue Configurations Exécuter/Déboguer.

2 Dans la boîte de dialogue Personnaliser les paramètres de lancement, cliquez sur Modifier.

3 Sélectionnez -extDir et spécifiez la valeur du paramètre à l'aide du chemin complet du dossier : /User/monnom/moncheminane.

La valeur que vous assignez à -extDir remplace toute valeur calculée par Flash Builder de façon interne.

Ajout de contenu pour la création de package

Lors de l'exportation d'un package d'application, vous pouvez ajouter un fichier au package en suivant les étapes ci-dessous :

1 Dans la section des paramètres de package de la boîte de dialogue Exporter vers une version validée, cliquez sur Personnaliser le lancement.

2 Cliquez sur Ajouter un fichier au package et spécifiez le chemin complet du fichier à ajouter au package.

Connexion des périphériques Google Android

Vous pouvez connecter un périphérique Google Android à votre ordinateur de développement pour visualiser ou déboguer l'application sur le périphérique Android.

Périphériques Android pris en charge

Les projets Flex et ActionScript Mobile requièrent Adobe AIR 2.6 ou version supérieure. Il vous est possible d'exécuter et de déboguer les projets mobiles uniquement sur des périphériques physiques prenant en charge Adobe AIR 2.6 ou version supérieure.

Adobe AIR 2.6 peut être installé sur les périphériques Android pris en charge exécutant Android 2.2 ou version supérieure. Pour obtenir la liste des périphériques pris en charge, voir http://www.adobe.com/flashplatform/certified_devices/. Consultez également la configuration minimale requise pour exécuter Adobe AIR sur les périphériques Android à la page Configuration requise.

Configuration de périphériques Android

Pour exécuter et déboguer les applications Flex Mobile à partir d'un périphérique Android, activez le débogage USB comme indiqué ci-dessous :

1 Sur le périphérique, procédez comme suit pour vous assurer que le débogage USB est activé :

   a Appuyez sur le bouton Accueil pour afficher l'écran d'accueil.

   b Accédez à Paramètres, puis sélectionnez Applications > Développement.

   c Activez le débogage USB.

2 Connectez le périphérique à votre ordinateur à l'aide d'un câble USB.

3 Développez la zone de notification située en haut de l'écran. Vous devez voir une indication de type USB connecté ou Connexion USB.

   a Appuyez sur USB connecté ou sur Connexion USB.

   b Si un ensemble d'options apparaît, parmi lesquelles le mode Charge seulement, sélectionnez le mode Charge seulement et appuyez sur OK.

   c Si vous voyez un bouton permettant de désactiver le mode stockage de masse, cliquez dessus pour désactiver le stockage de masse.

4 (Windows uniquement) Installez le pilote USB approprié à votre périphérique. Voir « Installation des pilotes de périphérique USB pour les périphériques Android (Windows) » à la page 234.

5 Développez la zone de notification située en haut de l'écran.

Si le débogage USB n'apparaît pas parmi les entrées, vérifiez le mode USB comme décrit à l'étape 3 ci-dessus. Assurez-vous que le mode USB n'est pas défini sur Mode PC.

Remarque : *une configuration supplémentaire est nécessaire pour le débogage. Voir « Test et débogage d'une application mobile sur un périphérique » à la page 241.*

Installation des pilotes de périphérique USB pour les périphériques Android (Windows)

Pilotes de périphérique et configurations

Les plateformes Windows requièrent l'installation d'un pilote USB pour connecter un périphérique Android à votre ordinateur de développement. Flash Builder fournit un pilote de périphérique et une configuration pour plusieurs périphériques Android.

Ces configurations de pilotes de périphérique sont répertoriées dans le fichier `android_winusb.inf`. Windows Device Manager accède à ce fichier au cours de l'installation du pilote de périphérique. Flash Builder installe `android_winusb.inf` à l'emplacement suivant :

```
<Adobe Flash Builder 4.7 Home>\utilities\drivers\android\android_winusb.inf
```

Pour obtenir la liste complète des périphériques pris en charge, voir Périphériques certifiés. Dans le cas de périphériques Android ne figurant pas dans la liste, vous pouvez mettre à jour le fichier `android_winusb.inf` afin d'y ajouter les pilotes USB. Voir « Ajout de configurations de pilotes de périphérique USB Android » à la page 234.

Installation d'un pilote de périphérique USB

1. Connectez votre périphérique Android au port USB de votre ordinateur.
2. Accédez à l'emplacement suivant :

   ```
   <Flash Builder>/utilities/drivers/android/
   ```

   Installez le pilote USB à l'aide de l'Assistant Ajout de nouveau matériel détecté de Windows ou du Gestionnaire de périphériques Windows.

Important : *si Windows ne reconnaît toujours pas le périphérique, vous devez installer le pilote USB approprié disponible auprès du fabricant du périphérique. Voir la page relative aux fabricants de pilotes USB pour obtenir des liens vers les sites Web de plusieurs fabricants de périphériques depuis lesquels vous pouvez télécharger le pilote USB approprié pour votre périphérique.*

Ajout de configurations de pilotes de périphérique USB Android

Si vous avez un périphérique Android pris en charge qui ne figure pas dans la section « Installation des pilotes de périphérique USB pour les périphériques Android (Windows) » à la page 234, mettez à jour le fichier `android_winusb.inf` pour inclure le périphérique.

1. Branchez le périphérique à un port USB de votre ordinateur. Windows vous indique que le pilote est introuvable.
2. A l'aide du Gestionnaire de périphériques Windows, ouvrez l'onglet Détails des propriétés du périphérique.
3. Sélectionnez la propriété ID de matériel pour afficher l'ID du matériel.
4. Ouvrez le fichier `android_winusb.inf` dans un éditeur de texte. Recherchez `android_winusb.inf` à l'emplacement suivant :

   ```
   <Adobe Flash Builder 4.7 Home>\utilities\drivers\android\android_winusb.inf
   ```

5. Notez les éléments qui apparaissent dans le fichier relatif à votre architecture : `[Google.NTx86]` ou `[Google.NTamd64]`. Les listes contiennent un commentaire descriptif et une ou plusieurs lignes dotées de l'ID de matériel, comme cela est illustré ici :

```
. . .
[Google.NTx86]
; HTC Dream
%CompositeAdbInterface% = USB_Install, USB\VID_0BB4&PID_0C02&MI_01
. . .
```

6 Copiez et collez un commentaire et une description de matériel. Pour le pilote de périphérique que vous souhaitez ajouter, modifiez la liste comme suit :

a Pour le commentaire, indiquez le nom du périphérique.

b Remplacez l'ID de matériel par celui qui a été identifié à l'étape 3 ci-dessus.

Par exemple :

```
. . .
[Google.NTx86]
; NEW ANDROID DEVICE
%CompositeAdbInterface%     = USB_Install, NEW HARDWARE ID
. . .
```

7 Utilisez le Gestionnaire de périphériques Windows pour installer le périphérique, de la manière décrite dans « Installation des pilotes de périphérique USB pour les périphériques Android (Windows) » à la page 234 ci-dessus.

Au cours de l'installation, Windows affiche un avertissement indiquant que le pilote provient d'un éditeur inconnu. Toutefois, le pilote permet à Flash Builder d'accéder à votre périphérique.

Processus de développement Apple iOS à l'aide de Flash Builder

Avant de développer une application iOS dans Flash Builder, il est essentiel de comprendre le processus de développement iOS et de savoir comment obtenir les certificats requis auprès d'Apple.

Vue d'ensemble du processus de développement et de déploiement pour iOS

Le tableau suivant fournit la liste des étapes constituant le processus de développement pour iOS, ainsi que des informations sur la manière d'obtenir les certificats requis et les prérequis de chaque étape.

Pour plus d'informations sur chacune de ces étapes, voir « Préparation à la génération, au débogage et au déploiement d'une application iOS » à la page 236.

N° d'étape	Etape	Emplacement	Prérequis
1.	Inscription au programme de développement d'Apple	Site Apple Developer	Aucun
2.	Enregistrement de l'identifiant de périphérique unique (UDID) de votre périphérique iOS	iOS Provisioning Portal	ID de développeur Apple (étape 1)
3.	Génération d'un fichier CSR de demande de signature de certificat (*.certSigningRequest)	• Sous Mac OS, utilisez le programme Trousseau d'accès. • Sous Windows, utilisez OpenSSL.	Aucun
4.	Génération d'un certificat de distribution/développeur iOS (*.cer)	iOS Provisioning Portal	• ID de développeur Apple (étape 1) • Fichier CSR (étape 3)

N° d'étape	Etape	Emplacement	Prérequis
5.	Conversion du certificat de distribution/développeur iOS au format P12	• Sous Mac OS, utilisez le programme Trousseau d'accès. • Sous Windows, utilisez OpenSSL.	• ID de développeur Apple (étape 1) • Certificat de distribution/développeur iOS (étape 4)
6.	Génération de l'ID d'application	iOS Provisioning Portal	ID de développeur Apple (étape 1)
7.	Génération d'un profil de configuration (*.mobileprovision)	iOS Provisioning Portal	• ID de développeur Apple (étape 1) • UDID de votre périphérique iOS (étape 2) • ID de l'application (étape 6)
8.	Développement de l'application	Flash Builder	• ID de développeur Apple (étape 1) • Certificat de distribution/développeur P12 (étape 5) • ID de l'application (étape 6)
9.	Déploiement de l'application	Périphérique	• Profil de configuration (étape 7) • Package de l'application (étape 8)

Préparation à la génération, au débogage et au déploiement d'une application iOS

Avant de générer une application iOS à l'aide de Flash Builder et de la déployer sur un périphérique iOS ou de la soumettre à l'App Store d'Apple, suivez la procédure suivante :

1 Inscrivez-vous au programme Apple iOS Developer Program.

Vous pouvez vous connecter à l'aide de votre ID Apple existant ou en créer un nouveau. L'assistant Apple Developer Registration vous guide au cours des étapes requises.

2 Enregistrez l'identifiant de périphérique unique (UDID) de votre périphérique.

Cette étape s'applique uniquement si vous déployez votre application sur un périphérique iOS et non vers l'App Store d'Apple. Dans le cas où vous voulez déployer votre application sur différents périphériques iOS, enregistrez l'UDID de chacun de ces périphériques.

Obtention de l'UDID de votre périphérique iOS

a Connectez le périphérique iOS à votre ordinateur de développement et lancez ensuite iTunes. Le périphérique iOS connecté apparaît alors sous la section Appareils dans iTunes.

b Cliquez sur le nom du périphérique iOS de façon à afficher son résumé.

c Sous l'onglet Résumé, cliquez sur Numéro de série afin d'afficher l'UDID à 40 caractères du périphérique iOS.

Vous pouvez copier l'UDID dans iTunes à l'aide du raccourci clavier Ctrl+C (Windows) ou Commande+C (Mac).

Enregistrement de l'UDID de votre périphérique

Connectez-vous au portail iOS Provisioning Portal à l'aide de votre ID Apple et enregistrez l'UDID du périphérique.

3 Générez un fichier CSR de demande de signature de certificat (*.certSigningRequest).

Il vous faut générer un fichier CSR pour pouvoir obtenir un certificat de distribution/développeur iOS. Vous pouvez générer un fichier CSR à l'aide de Trousseau d'accès sous Mac ou d'OpenSSL sous Windows. Lors de la génération d'un fichier CSR, vous fournissez uniquement votre nom d'utilisateur et adresse électronique, aucune autre information n'est nécessaire concernant l'application ou le périphérique.

Générer un CSR crée deux clés, une publique et une privée, ainsi qu'un fichier *.certSigningRequest. La clé publique est incluse dans le fichier CSR, tandis que la clé privée permet de signer la demande.

Pour plus d'informations sur la génération d'un fichier CSR, voir Génération d'une demande de signature de certificat.

4 Génération d'un certificat de développeur iOS ou d'un certificat de distribution iOS (*.cer), en fonction de vos besoins.

Remarque : *pour déployer une application sur un périphérique, il vous faut un certificat de développeur. Alors qu'un certificat de distribution est requis pour déployer l'application vers l'App Store d'Apple.*

Génération d'un certificat de développeur iOS

a Connectez-vous au portail iOS Provisioning Portal à l'aide de votre ID Apple et sélectionnez l'onglet Development.

b Cliquez sur Request Certificate et accédez au fichier CSR que vous avez généré et enregistré sur votre ordinateur (étape 3).

c Sélectionnez le fichier CSR et cliquez sur Submit.

d Cliquez sur Download sur la page Certificates.

e Enregistrez le fichier téléchargé (*.developer_identity.cer).

Génération d'un certificat de distribution iOS

f Connectez-vous au portail iOS Provisioning Portal à l'aide de votre ID Apple et sélectionnez l'onglet Distribution.

g Cliquez sur Request Certificate et accédez au fichier CSR que vous avez généré et enregistré sur votre ordinateur (étape 3).

h Sélectionnez le fichier CSR et cliquez sur Submit.

i Cliquez sur Download sur la page Certificates.

j Enregistrez le fichier téléchargé (*.distribution_identity.cer).

5 Convertissez le certificat de développeur iOS ou le certificat de distribution iOS au format de fichier P12 (*.p12).

Il est nécessaire de convertir le certificat de distribution ou de développeur iOS au format P12 de manière à ce que Flash Builder puisse signer numériquement votre application iOS. La conversion au format P12 regroupe dans un même fichier votre certificat de distribution/développeur iOS et la clé privée associée.

Remarque : *dans le cas où vous testez l'application sur l'ordinateur à l'aide d'AIR Debug Launcher (ADL), il n'est pas nécessaire de convertir le certificat de distribution/développeur iOS au format P12.*

Utilisez Trousseau d'accès sous Mac ou OpenSSL sous Windows pour générer un fichier d'échange d'informations personnelles (*.p12). Pour plus d'informations, voir Conversion d'un certificat de développement en fichier P12.

6 Générez l'ID de l'application en suivant la procédure suivante :

a Connectez-vous au portail iOS Provisioning Portal à l'aide de votre ID Apple.

b Accédez à la page App IDs et cliquez sur New App ID.

c Sous l'onglet Manage, saisissez une description de l'application, générez un nouvel identifiant initial de bundle (Bundle Seed ID) ainsi qu'un nouvel identifiant de bundle (Bundle Identifier).

Chaque application possède un ID d'application unique que vous spécifiez dans le fichier XML descripteur de l'application. Un ID d'application fourni est constitué d'un Bundle Seed ID de dix caractères qui est fourni par Apple et d'un suffixe Bundle Identifier que vous spécifiez vous-même. L'identifiant de bundle (Bundle Identifier) spécifié doit correspondre à l'ID d'application dans le fichier descripteur de l'application. Par exemple, si votre ID d'application est com.myDomain.*, l'ID dans le fichier descripteur de l'application doit commencer par com.myDomain.

Important : *les identifiants de bundle génériques (Wildcard Bundle Identifier) peuvent être utilisés dans le développement et le test d'applications iOS, mais pas dans le cadre de leur déploiement vers l'App Store d'Apple.*

7 Générez un fichier de configuration de développeur ou un fichier de configuration de distribution (*.mobileprovision).

Remarque : *pour déployer une application sur un périphérique, il vous faut un profil de configuration de développeur. Tandis que pour la déployer vers l'App Store d'Apple, vous avez besoin d'un profil de configuration de distribution. Celui-ci permet de signer l'application.*

Génération d'un profil de configuration de développeur

a Connectez-vous au portail iOS Provisioning Portal à l'aide de votre ID Apple.

b Accédez à Certificate > Provisioning et cliquez sur New Profile.

c Saisissez un nom de profil, sélectionnez le certificat de développeur iOS, l'ID de l'application ainsi que les UDID des périphériques sur lesquels vous souhaitez installer l'application.

d Cliquez sur Submit.

e Téléchargez le fichier de configuration de développeur généré (*.mobileprovision) et enregistrez-le sur l'ordinateur.

Génération d'un profil de configuration de distribution

f Connectez-vous au portail iOS Provisioning Portal à l'aide de votre ID Apple.

g Accédez à Certificate > Provisioning et cliquez sur New Profile.

h Saisissez un nom de profil et sélectionnez le certificat de distribution iOS ainsi que l'ID de l'application. Si vous souhaitez tester l'application avant de la déployer, spécifiez les UDID des périphériques sur lesquels vous voulez la tester.

i Cliquez sur Submit.

j Téléchargez le profil de configuration généré (*.mobileprovision) et enregistrez-le sur l'ordinateur.

Voir aussi

« Création d'une application iOS dans Flash Builder » à la page 221

Fichiers à sélectionner lors du test, du débogage ou de l'installation d'une application iOS

Pour exécuter, déboguer ou installer une application à des fins de test sur un périphérique iOS, sélectionnez les fichiers suivants dans la boîte de dialogue Exécuter > Configurations Déboguer :

- Certificat de développeur iOS au format P12 (étape 5)
- Fichier XML descripteur de l'application contenant l'ID de l'application (étape 6)
- Profil de configuration de développeur (étape 7)

Pour plus d'informations, voir « Débogage d'une application sur un périphérique Apple iOS » à la page 243 et « Installation d'une application sur un périphérique Apple iOS » à la page 246.

Fichiers à sélectionner lors du déploiement d'une application vers l'App Store d'Apple

Pour déployer une application vers l'App Store d'Apple, sélectionnez Final Release Package For Apple App Store (Package de version finale pour l'App Store d'Apple) comme type de package dans la boîte de dialogue Exporter vers une version validée et sélectionnez les fichiers suivants :

- Certificat de distribution iOS au format P12 (étape 5)
- Fichier XML descripteur de l'application contenant l'ID de l'application (étape 6)

  Remarque : *vous ne pouvez pas utiliser d'ID d'application générique lors de l'envoi d'une application à l'App Store d'Apple.*

- Profil de configuration de distribution (étape 7)

Pour plus d'informations, voir « Exportation de packages Apple iOS pour publication » à la page 249.

Test et débogage

Flash Builder propose des options de test et de débogage de votre application mobile sur le périphérique ou l'ordinateur à l'aide d'AIR Debug Launcher (ADL). Dans les deux cas, vous pouvez utiliser les capacités de débogage de Flash Builder, y compris la définition de points d'arrêt et l'examen de l'état de l'application à l'aide des volets Variables et Expressions.

Test et débogage d'une application mobile sur l'ordinateur

Pour des tâches initiales de test ou de débogage, ou si vous ne possédez pas de périphérique mobile, Flash Builder vous permet de tester et de déboguer les applications sur l'ordinateur en utilisant AIR Debug Launcher (ADL).

Avant de tester ou de déboguer une application mobile pour la première fois, vous devez définir une configuration de lancement. Spécifiez la plateforme cible et Sur le bureau comme méthode de lancement. Voir « Gestion des configurations de lancement » à la page 105.

Configuration des informations du périphérique pour un aperçu sur l'ordinateur

Les propriétés d'une configuration de périphérique déterminent la manière dont l'application apparaît dans ADL.

« Définition des configurations de périphériques » à la page 225 répertorie les configurations prises en charge. Les configurations des périphériques n'affectent pas l'aspect de l'application sur le périphérique.

Densité d'écran

Flash Builder utilise une densité d'écran de 240 PPP lorsque vous affichez l'aperçu de votre application sur votre ordinateur de développement. L'aspect d'une application lors de sa prévisualisation diffère parfois de son aspect sur un périphérique qui prend en charge une densité de pixels différente.

Prévisualisation des applications avec ADL

Lors de la prévisualisation d'une application sur l'ordinateur, Flash Builder lance l'application à l'aide d'ADL. ADL fournit un menu Périphérique avec les raccourcis correspondants pour émuler les boutons du périphérique.

Par exemple, pour émuler le bouton Retour d'un périphérique, sélectionnez Périphérique > Retour. Sélectionnez Périphérique > Rotation gauche ou Périphérique > Rotation droite pour émuler la rotation du périphérique. Les options de rotation sont désactivées si vous n'avez pas sélectionné l'orientation automatique.

Faites glisser le curseur dans une liste pour émuler le défilement de la liste sur un périphérique.

Brent Arnold, expert Flex certifié par Adobe, a créé un didacticiel vidéo sur l'utilisation d'ADL pour obtenir un aperçu d'une application mobile sur l'ordinateur.

Test et débogage d'une application iOS à l'aide d'un simulateur

Le simulateur iOS constitue un moyen rapide d'exécuter et de déboguer les applications iOS sans utiliser de périphérique. Il est capable de simuler les périphériques iOS, tels que l'iPhone ou l'iPad, ainsi que différentes versions d'iOS.

Important : *vous pouvez utiliser le simulateur iOS pour exécuter votre application iOS sur Mac uniquement.Il n'est pas pris en charge sous Windows.*

Aucun certificat de développeur ou profil de configuration n'est nécessaire pour les tests effectués avec le simulateur iOS. Toutefois, vous devez tout de même créer un certificat p12.

Préparation au débogage sur le simulateur iOS

Avant de tester ou de déboguer votre application sur le simulateur iOS, assurez-vous que :

- l'application utilise un SDK comprenant Adobe AIR 3.4 ;
- vous avez téléchargé le dernier jeu d'outils de développeur Xcode sur le site des développeurs d'Apple. Le jeu d'outils inclut le package Xcode, le simulateur iOS, ainsi que la structure et tous les outils requis pour exécuter le simulateur.

Débogage d'une application à l'aide du simulateur iOS

Remarque : *Cette fonctionnalité nécessite AIR 3.4 ou une version ultérieure*

Vous pouvez déboguer une application iOS à l'aide de la boîte de dialogue Configurations Déboguer.

1. Dans Flash Builder, sélectionnez Déboguer > Configurations Déboguer.
2. Dans la boîte de dialogue Configurations Profiler, sélectionnez Apple iOS pour le paramètre Plateforme cible et l'option Dans le simulateur iOS pour le paramètre Méthode de lancement.
3. Sélectionnez l'emplacement et la version du SDK du simulateur.

   Pour les ordinateurs Mac OS 10.6 ou version antérieure, le simulateur iOS est installé par défaut à l'emplacement suivant : `/Developer/Platforms/iPhoneSimulator.platform/Developer/SDKs`

   Pour les ordinateurs Mac OS 10.7 ou version supérieure, le simulateur iOS est installé par défaut à l'emplacement suivant :

   `/Applications/Xcode.app/Contents/Developer/Platforms/iPhoneSimulator.platform/Developer/SDKs`
4. Cliquez sur la liste déroulante Périphérique par défaut et sélectionnez le périphérique à simuler pour le débogage de l'application. Toute modification au niveau du périphérique par défaut effectuée lors de l'exécution du simulateur ne prend effet qu'au prochain lancement du simulateur.

   Remarque : *pour sélectionner un outil différent alors que le simulateur est en cours d'exécution, sélectionnez Hardware (Matériel) > Device (Appareil) dans le simulateur iOS et choisissez le périphérique souhaité.*
5. Cliquez sur Configurer les paramètres de groupement et sélectionnez le certificat p12.

Si vous ne possédez pas de certificat p12 émis par Apple, vous pouvez créer un certificat signé par l'utilisateur. Toutefois, les certificats signés par les utilisateurs peuvent être utilisés pour lancer des applications uniquement dans le simulateur et non sur des périphériques Apple iOS.

6 Cliquez sur Déboguer pour lancer le simulateur iOS et déboguer l'application. L'application est automatiquement lancée dans le simulateur iOS par défaut.

Pour plus d'informations sur l'utilisation du simulateur iOS, voir Using iOS Simulator (Utilisation du simulateur iOS).

Test et débogage d'une application mobile sur un périphérique

Vous pouvez utiliser Flash Builder pour tester ou déboguer une application mobile à partir de votre ordinateur de développement ou d'un périphérique.

Vous testez et déboguez les applications en fonction d'une configuration de lancement que vous définissez. Flash Builder partage la configuration de lancement entre l'exécution et le débogage de l'application. Lorsque vous utilisez Flash Builder pour déboguer une application sur un périphérique, Flash Builder installe une version de débogage de l'application sur le périphérique.

Remarque : *si vous exportez une version validée vers un périphérique, vous installez une version non déboguée de l'application. La version non déboguée ne convient pas au débogage.*

Pour plus d'informations, voir « Création ou modification des configurations de lancement pour l'exécution ou le débogage des applications mobiles » à la page 107.

Débogage d'une application sur un périphérique Google Android

Sur un périphérique Android, le débogage requiert Android 2.2 ou version ultérieure.

Vous pouvez effectuer le débogage dans l'un des scénarios suivants :

Débogage via USB : pour déboguer une application via une connexion USB, connectez le périphérique à l'ordinateur hôte par le biais d'un port USB. Lors d'un débogage via USB, Flash Builder groupe toujours l'application, puis l'installe et la lance sur le périphérique avant le démarrage du débogage. Assurez-vous que le périphérique est connecté au port USB de l'ordinateur hôte au cours de la session de débogage complète.

Débogage via un réseau : lorsque vous déboguez une application via le réseau, le périphérique et l'ordinateur hôte doivent utiliser le même réseau. L'ordinateur hôte et le périphérique peuvent être connectés au réseau via Wi-Fi, Ethernet ou Bluetooth.

Lors d'un débogage via un réseau, Flash Builder vous permet de déboguer une application qui est déjà installée sur un périphérique connecté, sans réinstaller l'application. Connectez le périphérique à l'ordinateur hôte via un port USB seulement au cours de la création de package et au cours de l'installation de l'application sur le périphérique. Vous pouvez déconnecter le périphérique du port USB au cours du débogage. Toutefois, assurez-vous qu'il existe une connexion réseau entre le périphérique et l'ordinateur hôte tout au long de la session de débogage.

Préparation au débogage de l'application

Avant de commencer le débogage via USB ou via un réseau, procédez comme suit :

1 (Windows) Vérifiez que le pilote USB approprié est installé.

Dans Windows, installez le pilote Android USB. Voir la documentation accompagnant la version du SDK Android pour plus d'informations. Pour plus d'informations, voir « Installation des pilotes de périphérique USB pour les périphériques Android (Windows) » à la page 234.

2 Veillez à ce que le débogage USB soit activé sur le périphérique.

Dans les paramètres du périphérique, accédez à Applications > Développement, et activez le débogage USB.

Recherche de périphériques connectés

Lorsque vous exécutez ou déboguez une application mobile sur un périphérique, Flash Builder vérifie les périphériques connectés. Si Flash Builder détecte un seul périphérique connecté en ligne, il déploie et lance l'application. Dans le cas contraire, Flash Builder lance la boîte de dialogue Choisir un périphérique pour les scénarios suivants :

- Aucun périphérique n'est trouvé
- Un seul périphérique déconnecté est trouvé ou sa version du système d'exploitation n'est pas prise en charge
- Plusieurs périphériques connectés trouvés

Si plusieurs périphériques sont détectés, la boîte de dialogue Choisir un périphérique répertorie les périphériques et leur état (en ligne ou hors connexion). Sélectionnez le périphérique à lancer.

La boîte de dialogue Choisir un périphérique répertorie la version du système d'exploitation et la version d'Adobe AIR. Si Adobe AIR n'est pas installé sur le périphérique, Flash Builder l'installe automatiquement.

Configuration du débogage réseau

Effectuez cette procédure seulement si vous déboguez une application via un réseau.

Préparation au débogage par le biais du réseau

Avant de déboguer une application via le réseau, procédez comme suit :

1. Dans Windows, ouvrez le port 7935 (port du débogueur Flash Player) et le port 7 (port echo/ping).

   Pour obtenir des instructions détaillées, voir cet article Microsoft TechNet.

   Sous Windows Vista, désélectionnez Connexion réseau sans fil dans Pare-feu Windows > Modifier les paramètres > Avancés.

2. Sur le périphérique, configurez les paramètres sans fil dans Paramètres > Sans fil et Réseau.

Sélection d'une interface réseau principale

Votre ordinateur hôte peut être connecté à plusieurs interfaces réseau simultanément. Toutefois, vous pouvez sélectionner une interface réseau principale à utiliser pour le débogage. Vous sélectionnez cette interface en ajoutant une adresse d'hôte dans le fichier de package Android APK.

1. Dans Flash Builder, ouvrez la fenêtre Préférences.
2. Sélectionnez Flash Builder > Plateformes cibles.

   La boîte de dialogue répertorie toutes les interfaces réseau disponibles sur l'ordinateur hôte.

3. Sélectionnez l'interface réseau que vous souhaitez incorporer dans le package Android APK.

Assurez-vous que l'interface réseau sélectionnée est accessible à partir du périphérique. Si le périphérique ne peut pas accéder à l'interface réseau sélectionnée pendant qu'il établit une connexion, Flash Builder affiche une boîte de dialogue vous invitant à fournir l'adresse IP de l'ordinateur hôte.

Débogage de l'application

1. Connectez le périphérique par le biais d'un port USB ou via une connexion réseau.
2. Sélectionnez Exécuter > Configurations Déboguer afin de paramétrer une configuration de lancement destinée au débogage.
   - Pour la Méthode de lancement, choisissez Sur le périphérique.

- Sélectionnez Déboguer via USB ou Déboguer via le réseau.

  Lors du premier débogage de l'application via un réseau, vous pouvez installer l'application sur le périphérique via USB. Pour déboguer une application via connexion USB, connectez le périphérique à l'ordinateur hôte par le biais d'un port USB.

  Une fois l'application installé, si vous ne voulez pas vous connecter via USB pour des sessions de débogage suivantes, désélectionnez l'option Installer l'application sur le périphérique via USB.

- (Facultatif) Effacez les données de l'application à chaque lancement.

  Sélectionnez cette option pour conserver l'état de l'application pour chaque session de débogage. Cette option s'applique uniquement si l'occurrence sessionCachingEnabled est définie sur True dans votre application.

3 Sélectionnez Déboguer pour lancer une session de débogage.

Le débogueur se lance et attend le démarrage de l'application. La session de débogage commence lorsque le débogueur établit une connexion avec le périphérique.

Lorsque vous essayez de déboguer sur un périphérique via un réseau, l'application affiche parfois une boîte de dialogue qui vous invite à saisir une adresse IP. Cette boîte de dialogue indique que le débogueur n'a pas pu se connecter. Assurez-vous que le périphérique est correctement connecté au réseau sans fil et que l'ordinateur qui exécute Flash Builder est accessible depuis ce réseau.

Remarque : *sur un réseau d'entreprise, le réseau d'un hôtel ou un autre réseau invité, parfois le périphérique ne peut pas se connecter à l'ordinateur, même si les deux sont sur le même réseau.*

Si vous effectuez un débogage via un réseau et que l'application a été installée auparavant sur le périphérique, démarrez le débogage en tapant l'adresse IP de l'ordinateur hôte.


Brent Arnold, expert Flex certifié par Adobe, a conçu un didacticiel vidéo traitant du débogage d'une application sur USB pour un périphérique Android.

Débogage d'une application sur un périphérique Apple iOS

Vous pouvez déboguer une application iOS sur un périphérique Apple iOS des manières suivantes :

Remarque : *vous pouvez déboguer via USB uniquement si vous disposez du SDK Adobe AIR version 3.4 ou ultérieure.*

Débogage via USB : pour déboguer une application via une connexion USB, connectez le périphérique iOS à l'ordinateur hôte par le biais d'un port USB. Lorsque vous effectuez un débogage via une connexion USB, Flash Builder crée le package iOS de débogage (fichier IPA) et installe le fichier IPA sur le périphérique. Vous devez lancer l'application manuellement sur le périphérique iOS pour démarrer le débogage. Assurez-vous que le périphérique est connecté au port USB de l'ordinateur hôte au cours de la session de débogage complète.

Débogage via un réseau : lorsque vous déboguez une application via le réseau, le périphérique et l'ordinateur hôte doivent utiliser le même réseau. L'ordinateur hôte et le périphérique peuvent être connectés au réseau via Wi-Fi, Ethernet ou Bluetooth.

Préparation au débogage de l'application

Avant de déboguer une application sur un périphérique iOS, assurez-vous de disposer d'un fichier de configuration et d'un certificat de développeur fournis par Apple, nécessaires pour la conversion au format P12. Pour plus d'informations, suivez la procédure décrite dans « Préparation à la génération, au débogage et au déploiement d'une application iOS » à la page 236.

Pour déboguer une application iOS par USB, assurez-vous que :

- l'application utilise un SDK comprenant Adobe AIR 3.4 ;

- iTunes est installé sur l'ordinateur utilisé pour le développement. iTunes installe les pilotes de périphérique nécessaires au débogage par USB.

Débogage de l'application

1 Connectez le périphérique Apple iOS à votre ordinateur de développement.

2 Dans Flash Builder, sélectionnez Exécuter > Configurations Déboguer.

3 Dans la boîte de dialogue Configurations Déboguer, procédez comme suit :

   a Sélectionnez l'application que vous souhaitez déboguer.

   b Sélectionnez la plateforme cible Apple iOS.

   c Sélectionnez la méthode de lancement Sur le périphérique.

   d Sélectionnez l'une des méthodes de création de package suivantes :

   Standard : utilisez cette méthode pour créer le package d'une version de l'application de qualité analogue à une version validée qui peut s'exécuter sur des périphériques Apple iOS. L'application dont le package a été effectué de cette façon offre des performances semblables à celles du package final de la version validée et peut être soumise à l'App Store d'Apple.

   Toutefois, notez que cette méthode de création d'un fichier iOS de débogage (IPA) prend plusieurs minutes.

   Rapide : utilisez cette méthode pour créer rapidement un fichier IPA, puis exécutez et déboguez le fichier sur le périphérique. Cette méthode est appropriée à des fins de test d'application. L'application traitée de cette façon présente des performances inférieures à celles d'une version validée et ne peut donc pas être soumise à l'App Store d'Apple.

   e Cliquez sur Configurer pour sélectionner le certificat de signature de code, le fichier de configuration et le contenu du package appropriés.

   f Sélectionnez Déboguer via USB ou Déboguer via le réseau.

   Lorsque vous effectuez un débogage via une connexion USB, un port de débogage disponible est sélectionné et ajouté au package iOS de débogage.

   Lors du débogage du réseau, cliquez sur Configurer le débogage réseau pour sélectionner l'interface réseau que vous souhaitez incorporer dans le package iOS de débogage.

   Remarque : *votre ordinateur hôte peut être connecté à plusieurs interfaces réseau simultanément. Toutefois, vous pouvez sélectionner une interface réseau principale à utiliser pour le débogage.*

   g Avant de commencer à déboguer l'application, vérifiez qu'elle n'est pas déjà installée sur votre périphérique. Si elle est installée, désinstallez-la.

   h Cliquez sur Déboguer. Flash Builder affiche une boîte de dialogue vous invitant à fournir un mot de passe. Saisissez le mot de passe de votre certificat P12.

   Flash Builder génère le fichier IPA de débogage et l'installe sur le périphérique.

   Remarque : *Flash Builder installe le fichier IPA sur le périphérique uniquement si vous utilisez le SDK Adobe AIR version 3.4 ou ultérieure. Si vous utilisez une version du SDK Adobe AIR antérieure à 3.4, vous devez utiliser iTunes pour installer l'application sur le périphérique. Pour plus d'informations, reportez-vous à cette rubrique d'aide.*

4 Lancez l'application sur le périphérique iOS.

5 Flash Builder essaie de se connecter à l'adresse d'hôte spécifiée dans le fichier IPA de débogage. Si l'application ne peut pas se connecter à l'adresse d'hôte, Flash Builder affiche une boîte de dialogue vous invitant à fournir l'adresse IP de l'ordinateur hôte.

Remarque : *si vous n'avez pas modifié votre code ni vos ressources depuis la création du dernier package IPA de débogage, Flash Builder ignore la création de package et effectue le débogage de l'application. Cela signifie que vous pouvez lancer l'application installée sur le périphérique et cliquer sur Déboguer pour vous connecter au débogueur Flash Builder. De cette manière, vous pouvez effectuer plusieurs débogages successifs sans chaque fois créer le package de l'application.*

Modification de l'ID d'application lors du test ou du débogage

Chaque application possède un ID d'application unique à spécifier dans le fichier XML descripteur de l'application.

Lors de l'exécution ou du débogage d'une application iOS, vous générez un ID d'application dans le portail iOS Provisioning Portal. L'ID d'application généré est basé sur l'ID d'application spécifié dans le fichier descripteur de l'application. Par exemple, si votre ID d'application est com.myDomain.*, l'ID dans le fichier descripteur de l'application doit commencer par com.myDomain. Cet ID d'application est utilisé dans le fichier de configuration pour le test et le débogage de l'application. Pour plus d'informations, voir « Processus de développement Apple iOS à l'aide de Flash Builder » à la page 235.

Généralement, lorsque vous exécutez ou déboguez une application, Flash Builder crée une copie du fichier descripteur de l'application et modifie de façon interne l'ID d'application en y ajoutant une extension .debug. Si, par exemple, l'ID de votre application est com.myDomain, il est modifié en com.myDomain.debug. Flash Builder effectue cette modification pour vous éviter de remplacer une application portant le même nom lors du test ou du débogage de votre application.

Cette modification interne peut toutefois résulter en une non-correspondance entre l'ID d'application spécifié dans le fichier XML descripteur de l'application et l'ID d'application spécifié dans le fichier de configuration. Pour résoudre ce problème, Flash Builder permet de modifier l'ID d'application dans la boîte de dialogue Configurations Exécuter/Déboguer. Lorsque vous remplacez l'ID d'application par un fichier d'application dans le projet, Flash Builder conserve les informations de mappage et les utilise pour d'autres sessions de débogage.

Remarque : *l'ID d'application apparaît vide dans la boîte de dialogue Configurations Exécuter/Déboguer si aucun dossier bin-debug n'a été créé pour le projet. La valeur d'ID d'application saisie est mise à jour dans le fichier descripteur de l'application au prochain lancement de l'application lorsque vous cliquez sur Exécuter ou Déboguer.*

Installation sur des périphériques

Installation d'une application sur un périphérique Google Android

Au cours des phases de développement, de test et de déploiement de votre projet, vous avez la possibilité d'installer votre application directement sur un périphérique.

Vous pouvez utiliser Flash Builder pour installer une application directement sur un périphérique Android. Lorsque vous installez un package sur un périphérique sur lequel Adobe AIR n'est pas installé, Flash Builder installe automatiquement Adobe AIR.

1 Connectez le périphérique Google Android à votre ordinateur de développement.

   Flash Builder accède au périphérique connecté au port USB de l'ordinateur. Assurez-vous d'avoir configuré les pilotes de périphérique USB requis. Voir « Connexion des périphériques Google Android » à la page 233.

2 Dans Flash Builder, sélectionnez Exécuter > Configurations Exécuter. Dans la boîte de dialogue Configurations Exécuter, sélectionnez l'application mobile que vous voulez déployer.

3 Sélectionnez la méthode de configuration de lancement Sur le périphérique.

4 (Facultatif) Indiquez si vous souhaitez effacer les données de l'application à chaque lancement.

5 Cliquez sur Appliquer.

Flash Builder installe et lance l'application sur le périphérique Android. Si vous installez un package sur un périphérique sur lequel Adobe AIR n'est pas installé, Flash Builder l'installe automatiquement.

Brent Arnold, expert Flex certifié par Adobe, a conçu un didacticiel vidéo traitant de la configuration et de l'exécution de votre application sur un périphérique Android.

Installation d'une application sur un périphérique Apple iOS

Important : *avant d'installer une application sur périphérique iOS, il vous faut un certificat de développement Apple iOS (au format P12) ainsi qu'un profil de configuration de développeur. Veillez à suivre la procédure décrite dans* « *Préparation à la génération, au débogage et au déploiement d'une application iOS* » *à la page 236.*

1 Connectez le périphérique Apple iOS à votre ordinateur de développement.

2 Dans Flash Builder, sélectionnez Exécuter > Configurations Exécuter.

3 Dans la boîte de dialogue Configurations Exécuter, procédez comme suit :

a Sélectionnez l'application que vous souhaitez installer.

b Sélectionnez la plateforme cible Apple iOS.

c Sélectionnez la méthode de lancement Sur le périphérique.

d Sélectionnez l'une des méthodes de création de package suivantes :

Standard : utilisez cette méthode pour créer le package d'une version de l'application de qualité analogue à une version validée qui peut s'exécuter sur des périphériques Apple iOS.

La méthode standard de création de packages traduit le bytecode du fichier SWF de l'application en instructions ARM avant de créer le package. En raison de cette étape de traduction supplémentaire, cette méthode de création d'un fichier d'application (IPA) prend plusieurs minutes. La méthode standard prend plus de temps que la méthode rapide. Toutefois, l'application traitée de cette façon présente des performances dignes d'une version validée et peut donc être soumise à l'App Store d'Apple.

Rapide : utilisez cette méthode pour créer rapidement un fichier IPA.

La méthode rapide de création de packages ne convertit pas le bytecode et regroupe simplement le fichier SWF et les ressources de l'application avec l'environnement d'exécution Adobe AIR précompilé. La méthode rapide de création de packages est plus rapide que la méthode standard. Toutefois, l'application traitée de cette façon présente des performances inférieures à celle d'une version validée et ne peut donc pas être soumise à l'App Store d'Apple.

Remarque : *entre les méthodes de création de packages standard et rapide, la différence réside dans l'environnement d'exécution ou le fonctionnement.*

e Cliquez sur Configurer pour sélectionner le certificat de signature de code, le fichier de configuration et le contenu du package appropriés.

f Cliquez sur Exécuter. Flash Builder affiche une boîte de dialogue vous invitant à fournir un mot de passe. Entrez votre mot de passe de certificat P12.

Flash Builder génère le fichier IPA et l'installe sur votre périphérique iOS.

Remarque : *Flash Builder installe le fichier IPA sur le périphérique uniquement si vous utilisez le SDK Adobe AIR version 3.4 ou ultérieure. Si vous utilisez une version du SDK Adobe AIR antérieure à 3.4, vous devez utiliser iTunes pour installer l'application sur le périphérique. Pour plus d'informations, reportez-vous à cette rubrique d'aide.*

4 Lancez l'application déployée sur votre périphérique iOS.

Création de packages et exportation

Lorsque votre application est prête à être déployée, utilisez le processus d'exportation d'une version validée, tout comme vous le feriez pour préparer une application de bureau ou Web. La principale différence tient au fait que lorsque vous exportez une version validée d'un projet mobile, Flash Builder groupe la version en tant que programme d'installation natif et non pas en tant que fichier .air. Par exemple, sur Android, Flash Builder produit un fichier .apk qui ressemble à un package d'application Android natif. Le programme d'installation natif permet de distribuer les applications AIR de la même façon que les applications natives sur chaque plateforme.

Création du package et exportation d'une application mobile vers une boutique en ligne

Utilisez la fonction Flash Builder Exporter vers une version validée pour créer un package de la version validée d'une application mobile et l'exporter. La version validée correspond généralement à la version finale de l'application que vous souhaitez télécharger vers une boutique en ligne, telle que Android Market, Amazon Appstore ou l'App Store d'Apple.

Lors de l'exportation d'une application, vous pouvez choisir d'installer l'application sur un périphérique. Si le périphérique est connecté à votre ordinateur au cours de l'exportation, Flash Builder installe l'application sur le périphérique. Vous pouvez également choisir d'exporter un package d'application spécifique à une plateforme pour l'installer ultérieurement sur un périphérique. Le package ainsi obtenu peut être déployé et installé de la même manière qu'une application native.

Pour plus d'informations sur l'exportation d'une application vers Android Market ou Amazon Appstore, voir « Exportation de packages Android APK pour publication » à la page 247.

Pour plus d'informations sur l'exportation d'une application iOS vers l'App Store d'Apple, voir « Exportation de packages Apple iOS pour publication » à la page 249.

Exportation de l'application avec un moteur d'exécution captif

Lorsque vous utilisez la fonction Exporter vers une version validée pour exporter une application mobile, vous pouvez choisir d'incorporer l'environnement d'exécution Adobe AIR dans le package de l'application.

Les utilisateurs peuvent ensuite exécuter l'application sur un périphérique ne disposant pas d'Adobe AIR. Selon la plateforme vers laquelle le package est exporté, il vous est possible de choisir un moteur d'exécution captif ou un environnement d'exécution partagé.

Exportation de packages Android APK pour publication

Avant d'exporter une application mobile, vous pouvez personnaliser les autorisations Android. Personnalisez manuellement les paramètres dans le fichier descripteur de l'application. Ces paramètres figurent dans le bloc `<android>` du fichier bin-debug/*nom_application*-app.xml. Pour plus d'informations, voir Setting AIR application properties (Configuration des propriétés d'application AIR).

Si vous exportez l'application pour l'installer ultérieurement sur un périphérique, installez le package d'application à l'aide des outils fournis par le fournisseur du système d'exploitation du périphérique.

1 Dans Flash Builder, sélectionnez Projet > Exporter vers une version validée.

2 Sélectionnez le projet et l'application que vous souhaitez exporter.

3 Sélectionnez les plateformes cibles et l'emplacement où exporter le projet.

4 Exportez et signez un package d'application spécifique à la plateforme.

Vous pouvez créer un package de votre application avec une signature numérique pour chaque plateforme cible ou en tant qu'application AIR signée numériquement pour l'ordinateur.

Vous pouvez également exporter l'application en tant que fichier AIRI intermédiaire qui peut être signé ultérieurement. Si vous sélectionnez cette option, utilisez ultérieurement l'outil de ligne de commande AIR adt pour créer un package du fichier AIRI sous la forme d'un fichier APK. Installez ensuite le fichier APK sur le périphérique à l'aide d'outils spécifiques à la plateforme (par exemple, avec le SDK Android, utilisez adb). Pour plus d'informations sur l'utilisation des outils de ligne de commande pour créer un package de votre application, voir « Signature numérique des applications Adobe AIR » à la page 119.

5 Sur la page des paramètres de package, vous pouvez sélectionner le certificat numérique ainsi que le contenu du package et les éventuelles extensions natives.

Déploiement : si vous voulez également installer l'application sur un périphérique, cliquez sur la page Déploiement et sélectionnez Installer et lancer l'application sur tout périphérique connecté. Assurez-vous que vous avez connecté un ou plusieurs périphériques aux ports USB de votre ordinateur.

- **Exportation d'une application avec moteur d'exécution captif**

  Sélectionnez cette option si vous souhaitez intégrer l'environnement d'exécution AIR au fichier APK lors de l'exportation du package de l'application. Les utilisateurs peuvent ensuite exécuter l'application sur un périphérique ne disposant pas d'AIR.

- **Exporter l'application qui utilise un environnement d'exécution partagé**

  Sélectionnez cette option si vous ne souhaitez pas intégrer l'environnement d'exécution AIR au fichier APK lors de l'exportation du package de l'application. Vous pouvez sélectionner ou spécifier une URL afin de télécharger Adobe AIR pour le package de l'application si AIR n'est pas déjà installé sur le périphérique de l'utilisateur.

  L'URL par défaut pointe vers Android Market. Vous pouvez toutefois remplacer l'URL par défaut en saisissant votre propre URL ou en sélectionnant l'URL qui pointe vers un emplacement de l'Amazon Appstore.

Signature numérique : cliquez sur l'onglet Signature numérique pour créer ou localiser un certificat numérique qui représente l'identité de l'éditeur de l'application. Vous pouvez aussi spécifier un mot de passe pour le certificat sélectionné.

Si vous créez un certificat, il est *auto-signé*. Vous pouvez obtenir un certificat commercial signé auprès d'un fournisseur de certificats. Voir « Signature numérique des applications Adobe AIR » à la page 119.

Contenu du package : (facultatif) cliquez sur l'onglet Contenu du package pour spécifier les fichiers à inclure dans le package.

Extensions natives : (facultatif) sélectionnez les extensions que vous souhaitez inclure dans le package de l'application.

Pour plus d'informations sur les extensions natives, voir « Utilisation d'extensions natives » à la page 222.

6 Cliquez sur Terminer.

Flash Builder crée `ApplicationName.apk` dans le répertoire spécifié dans le premier volet (par défaut, au niveau supérieur de votre projet). Si le périphérique a été connecté à votre ordinateur au cours de l'exportation, Flash Builder installe l'application sur le périphérique.

Exportation de packages Apple iOS pour publication

Vous pouvez créer et exporter un package iOS pour une distribution ponctuelle ou pour une soumission à l'App Store d'Apple.

Important : *avant d'exporter un package iOS, veillez à obtenir les certificats requis ainsi qu'un profil de configuration de distribution auprès d'Apple. Pour ce faire, suivez la procédure décrite dans « Préparation à la génération, au débogage et au déploiement d'une application iOS » à la page 236.*

1 Dans Flash Builder, sélectionnez Projet > Exporter vers une version validée.

2 Sélectionnez Apple iOS en tant que plateforme cible pour exporter et signer un package IPA.

Cliquez sur Suivant.

3 Sur la page des paramètres de package, vous pouvez sélectionner le certificat de configuration, le certificat numérique ainsi que le contenu du package et les éventuelles extensions natives.

Déploiement : lorsque vous exportez un package iOS, l'environnement d'exécution AIR est intégré par défaut au fichier IPA.

Signature numérique : sélectionnez le certificat P12 ainsi que le profil de configuration de distribution obtenu auprès d'Apple.

Vous pouvez sélectionner l'un des types de packages suivants :

- **Ad Hoc Distribution For Limited Distribution** (Package ad hoc destiné à une distribution limitée du package) dans le cas d'une distribution limitée de l'application
- **Final Release Package For Apple App Store** (Package de version finale pour l'App Store d'Apple) pour envoyer l'application à l'App Store d'Apple

Contenu du package : (facultatif) cliquez sur l'onglet Contenu du package pour spécifier les fichiers à inclure dans le package.

Extensions natives : (facultatif) sélectionnez les extensions que vous souhaitez inclure dans le package de l'application.

Dans le cas où l'extension native utilise les fonctionnalités du SDK iOS5, sélectionnez l'emplacement de ce SDK. Pour plus d'informations, voir « Configuration externe d'un SDK iOS » à la page 224.

4 Cliquez sur Terminer.

Flash Builder valide la configuration des paramètres du package, puis compile l'application. Une fois le package créé, vous pouvez installer le fichier IPA sur un périphérique Apple iOS connecté ou l'envoyer à l'App Store d'Apple.

Pour créer un package du fichier IPA à l'aide de l'outil ADT (AIR Developer Tool), voir Packages iOS dans *Création d'applications Adobe AIR*.

Création, test et déploiement à l'aide de la ligne de commande

Vous pouvez créer une application mobile sans Flash Builder. Vous pouvez utiliser à la place les outils de ligne de commande mxmlc, adl et adt.

Le processus général de développement et d'installation d'une application mobile sur un périphérique à l'aide des outils de ligne de commande est détaillé ci-dessous. Chacune de ces étapes sera décrite ultérieurement de façon plus détaillée :

1 Compilez l'application à l'aide de l'outil mxmlc.

```
mxmlc +configname=airmobile MyMobileApp.mxml
```

Cette étape requiert que vous transmettiez le paramètre `configname` défini sur airmobile.

2 Testez l'application dans AIR Debug Launcher (ADL) à l'aide de l'outil adl.

```
adl MyMobileApp-app.xml -profile mobileDevice
```

Cette étape nécessite la création d'un fichier descripteur de l'application et sa transmission en tant qu'argument à l'outil adl. Vous devez également spécifier le profil `mobileDevice`.

3 Groupez l'application à l'aide de l'outil adt.

```
adt -package -target apk SIGN_OPTIONS MyMobileApp.apk MyMobileApp-app.xml MyMobileApp.swf
```

Cette étape requiert la création préalable d'un certificat.

4 Installez l'application sur votre périphérique mobile. Pour installer l'application sur un périphérique Android, utilisez l'outil adb.

```
adb install -r MyMobileApp.apk
```

Cette étape requiert la connexion préalable de votre périphérique mobile à votre ordinateur via USB.

5 Déployez l'application mobile dans des magasins en ligne.

Compilation d'une application mobile à l'aide de mxmlc

Vous pouvez compiler des applications mobiles avec le compilateur de ligne de commande mxmlc. Pour utiliser mxmlc, affectez au paramètre `configname` la valeur `airmobile` ; par exemple :

```
mxmlc +configname=airmobile MyMobileApp.mxml
```

En affectant `+configname=airmobile`, vous invitez le compilateur à utiliser le fichier airmobile-config.xml. Ce fichier se trouve dans le répertoire sdk/frameworks. Ce fichier effectue les tâches suivantes :

- S'applique au thème mobile.swc.
- Effectuez les modifications suivantes dans le chemin d'accès à la bibliothèque :
  - Supprime libs/air du chemin d'accès à la bibliothèque. Les applications mobiles ne prennent pas en charge les classes Window et WindowedApplication.
  - Supprime libs/mx du chemin d'accès à la bibliothèque. Les applications mobiles ne prennent pas en charge les composants MX (autres que les graphiques).
  - Ajoute libs/mobile au chemin d'accès à la bibliothèque.
- Supprime les espaces de nom ns.adobe.com/flex/mx et www.adobe.com/2006/mxml. Les applications mobiles ne prennent pas en charge les composants MX (autres que les graphiques).
- Désactive l'accessibilité.
- Supprime les entrées RSL ; les applications mobiles ne prennent pas en charge les RSL.

Le compilateur mxmlc génère un fichier SWF.

Test d'une application mobile à l'aide d'adl

Vous utilisez AIR Debug Launcher (ADL) pour tester une application mobile. Vous utilisez ADL pour exécuter et tester une application sans être tenu de la grouper ni de l'installer auparavant sur un périphérique.

Débogage à l'aide de l'outil adl

ADL imprime les instructions trace et les erreurs d'exécution au format de sortie standard, mais ne prend pas en charge les points d'arrêt ni d'autres fonctions de débogage. Vous pouvez utiliser un environnement de développement intégré tel que Flash Builder pour les problèmes complexes de débogage.

Lancement de l'outil adl

Pour lancer l'outil adl à partir de la ligne de commande, passez votre fichier descripteur d'application de l'application mobile et définissez le paramètre `profile` sur `mobileDevice`, comme le montre l'exemple suivant :

```
adl MyMobileApp-app.xml -profile mobileDevice
```

Le profil `mobileDevice` définit un ensemble de fonctionnalités pour les applications qui sont installées sur des périphériques mobiles. Pour obtenir des informations spécifiques sur le profil `mobileDevice`, voir Fonctionnalités des différents profils.

Création d'un descripteur d'application

Si vous n'avez pas utilisé Flash Builder pour compiler votre application, vous devez créer manuellement le fichier descripteur de l'application. Vous pouvez utiliser le fichier /sdk/samples/descriptor-sample.xml en tant que base. En général, au minimum, effectuez les modifications suivantes :

- Pointez l'élément `<initialWindow><content>` sur le nom du fichier SWF de votre application mobile :

```
<initialWindow>
    <content>MyMobileApp.swf</content>
    ...
</initialWindow>
```

- Modifiez le titre de l'application, car c'est ainsi qu'il apparaît sous l'icône de l'application sur votre périphérique mobile. Pour modifier le titre, éditez l'élément `<name><text>` :

```
<name>
    <text xml:lang="en">MyMobileApp by Nick Danger</text>
</name>
```

- Ajoutez un bloc `<android>` pour définir les autorisations spécifiques à Android pour l'application. Selon les services utilisés par votre périphérique, vous pouvez souvent utiliser l'autorisation suivante :

```
<application>
    ...
    <android>
        <manifestAdditions>
            <![CDATA[<manifest>
                <uses-permission android:name="android.permission.INTERNET"/>
            </manifest>]]>
        </manifestAdditions>
    </android>
</application>
```

Vous pouvez aussi utiliser le fichier descripteur pour définir la hauteur et la largeur de l'application, l'emplacement des fichiers d'icône, les informations relatives à la version et d'autres détails concernant l'emplacement d'installation.

Pour plus d'informations sur la création et la modification des fichiers descripteurs d'applications, voir Fichiers descripteurs d'applications AIR.

Groupement d'une application mobile à l'aide d'adt

Vous utilisez AIR Developer Tool (ADT) pour grouper les applications mobiles sur la ligne de commande. L'outil adt peut créer un fichier APK que vous pouvez déployer sur un périphérique mobile Android.

Création d'un certificat

Avant de créer un fichier APK, créez un certificat. A des fins de développement, vous pouvez utiliser un certificat auto-signé. Vous pouvez créer un certificat auto-signé avec l'outil adt, comme le montre l'exemple suivant :

```
adt -certificate -cn SelfSign -ou QE -o "Example" -c US 2048-RSA newcert.p12 password
```

L'outil adt crée le fichier newcert.p12 dans le répertoire en cours. Vous passez ce certificat à adt lorsque vous groupez votre application. N'utilisez pas de certificats auto-signés pour les applications de production. Ils fournissent uniquement une assurance restreinte aux utilisateurs. Pour obtenir des informations sur la signature de vos fichiers d'installation AIR avec un certificat émis par une autorité de certification reconnue, voir Signature d'applications AIR.

Création du fichier de package

Pour créer le fichier APK pour Android, passez les détails sur l'application à l'outil adt, y compris le certificat, comme l'illustre l'exemple suivant :

```
adt -package -target apk -storetype pkcs12 -keystore newcert.p12 -keypass password
MyMobileApp.apk MyMobileApp-app.xml MyMobileApp.swf
```

La sortie de l'outil adt est un fichier *appname*.apk.

Groupement pour iOS

Pour grouper des applications mobiles pour iOS, vous devez obtenir un certificat de développement d'Apple, ainsi qu'un fichier de configuration. Cela exige que vous rejoigniez le programme des développeurs Apple. Pour plus d'informations, voir « Préparation à la génération, au débogage et au déploiement d'une application iOS » à la page 236.

Piotr Walczyszyn, expert en programmation Flex, explique comment grouper l'application avec ADT en utilisant Ant pour les périphériques iOS.

Dans son blog, Valentin Simonov fournit des informations supplémentaires sur la façon de publier votre application sur iOS.

Installation d'une application mobile sur un périphérique à l'aide d'adb

Vous utilisez Android Debug Bridge (adb) pour installer l'application (fichier APK) sur un périphérique mobile qui exécute Android. L'outil adb fait partie du SDK Android.

Connexion du périphérique à un ordinateur

Avant d'exécuter adb pour installer le fichier APK sur votre périphérique mobile, connectez le périphérique à votre ordinateur. Sur les systèmes Windows et Linux, la connexion d'un périphérique requiert les pilotes USB.

Pour plus d'informations sur l'installation des pilotes USB pour votre périphérique, voir Utilisation des périphériques matériels.

Installation de l'application sur un périphérique connecté

Après avoir connecté le périphérique à l'ordinateur, vous pouvez installer l'application sur le périphérique. Pour installer l'application à l'aide de l'outil adb, utilisez l'option `install` et affectez le nom de votre fichier APK, comme le montre l'exemple suivant :

```
adb install -r MyMobileApp.apk
```

Utilisez l'option `-r` pour remplacer l'application si vous l'avez installée au préalable. Dans le cas contraire, vous devez désinstaller l'application chaque fois que vous souhaitez installer une version plus récente sur le périphérique mobile.

Voir aussi

Android Debug Bridge

Déploiement de l'application dans des magasins en ligne

Vous pouvez déployer l'application sur des magasins d'applications en ligne, tels qu'Android Market, Amazon Appstore ou l'App Store d'Apple.


Lee Brimlow montre comment déployer une nouvelle application AIR pour Android dans Android Market.


Christian Cantrell explique comment déployer l'application sur Amazon Appstore pour Android.

Conservation des dossiers bin-release-temp et dSYM

Lors de l'exportation d'un package pour une la publication d'une version, Flash Builder crée le fichier SWF adapté à une version validée, le fichier XML descripteur de l'application et le contenu de package requis dans le dossier bin-release-temp de votre projet. Une fois l'exportation du package effectuée, Flash Builder supprime par défaut le contenu du dossier bin-release-temp.

Si vous souhaitez conserver le contenu du dossier et empêcher Flash Builder de le supprimer, sélectionnez Conserver le dossier bin-release-temp dans la boîte de dialogue Exporter vers une version validée. Vous devez sélectionner cette option chaque fois que vous souhaitez conserver le contenu du dossier bin-release-temp lors de l'exportation.

Conservation du dossier *.dSYM

Lors de l'exportation d'un package Apple iOS (IPA) avec des extensions natives sur Mac, le compilateur AOT (ahead-of-time, en avance) crée un dossier (avec une extension *.dSYM). Le dossier *.dSYM est créé dans AOT/*.dSYM, dans le dossier bin-release-temp. Pour conserver le dossier *.dSYM et empêcher Flash Builder de le supprimer (configuration par défaut), sélectionnez Conserver le dossier bin-release-temp dans la boîte de dialogue Exporter vers une version validée.

Remarque : *lors de l'exécution ou du débogage d'une application de package iOS, un dossier *.dSYM est créé dans AOT/*.dSYM, dans le dossier bin-debug. Ce dossier contient les fichiers nécessaires au débogage des extensions natives sur Mac. Par défaut, Flash Builder conserve ce dossier.*

Chapitre 10 : Utilisation de Flash Builder avec Flash Professional

Utilisez des projets Flash Professional pour accéder au code de fichiers FLA ou XFL créés avec Flash Professional CS5.5 ou version supérieure. En général, vous créez un projet et des fichiers dans Flash Professional, puis vous créez un projet correspondant dans Flash Builder pour modifier et déboguer les fichiers. Cette fonctionnalité permet aux développeurs Flash Professional d'utiliser l'environnement de modification et de débogage que fournit Flash Builder.

Remarque : *pour accéder aux fonctions de Flash Professional dans Flash Builder, vous devez installer Flash Professional CS5.5 ou une version supérieure.*

Création d'un projet Flash Professional

Pour créer un projet Flash Professional dans Flash Builder, vous devez avoir déjà commencé le projet dans Flash Professional. C'est-à-dire que le fichier FLA ou XFL doit exister.

1. Sélectionnez Fichier > Nouveau > Projet Flash Professional.
2. Recherchez le fichier FLA ou XFL pour le projet.

   Le nom du fichier est utilisé comme nom du projet.
3. Spécifiez l'emplacement du projet.

   Vous pouvez utiliser l'emplacement par défaut dans l'espace de travail ou rechercher un nouvel emplacement.
4. Cliquez sur Terminer.

   Flash Builder ouvre le nouveau projet dans l'Explorateur de packages. Le dossier contenant le fichier FLA cible est accessible. Le fichier FLA sélectionné devient le fichier cible du projet. Les fichiers ActionScript qui dépendent des fichiers cible peuvent être modifiés.

   S'il n'est pas déjà en cours d'exécution, Flash Professional démarre.

Définition des propriétés des projets Flash Professional

1. Sélectionnez Projet > Propriétés > Flash Professional.
2. Sélectionnez Ajouter pour ajouter des fichiers supplémentaires au projet.

   Un projet ne peut comporter qu'un seul fichier FLA ou XFL cible comme fichier cible par défaut. Utilisez le bouton Définir par défaut pour spécifier le fichier cible par défaut du projet.
3. Cliquez sur OK.

Opérations possibles dans un projet Flash Professional

Vous pouvez appliquer les opérations suivantes aux fichiers source d'un projet Flash Professional :

- Modifier les fichiers ActionScript qui dépendent du fichier FLA cible
- Déboguer le fichier dans le débogueur Flash Builder ou Flash Professional :

  Lors de la modification des fichiers dans Flash Builder, vous pouvez définir des points d'arrêt dans les fichiers ActionScript du projet.

  Pour déboguer le fichier dans Flash Builder, sélectionnez Exécuter > Déboguer *fichier* ou cliquez sur le bouton Déboguer dans la barre d'outils.

  Pour déboguer le fichier dans Flash Professional, sélectionnez Exécuter > Déboguer la séquence ou cliquez sur le bouton Déboguer la séquence dans Flash Builder. Le débogueur Flash Professional reconnaît les points d'arrêt définis dans Flash Builder.
- Publiez le fichier dans Flash Professional CS5.5 ou une version supérieure :

  Sélectionnez Projet > Publier la séquence ou cliquez sur Publier dans le bouton Flash Professional de la barre d'outils.
- Exécuter le fichier dans Flash Builder ou Flash Professional :

  Pour exécuter le fichier dans Flash Builder, sélectionnez Exécuter > Exécuter *fichier* ou cliquez sur le bouton Exécuter de la barre d'outils.

  Pour exécuter le fichier dans Flash Professional, sélectionnez Exécuter > Tester la séquence ou cliquez sur le bouton Tester la séquence de la barre d'outils.

Chapitre 11 : Personnalisation de Flash Builder

Adobe ® Flash® Builder™ est un plug-in pour la plateforme de développement Eclipse. Outre les préférences générales que vous avez spécifiées pour Eclipse, spécifiez les préférences de Flash Builder. Les paramètres de certaines fonctions devront être définis à la fois dans les nœuds Eclipse et Flash Builder de la boîte de dialogue Préférences. Par exemple, lorsque vous définissez vos préférences pour le débogueur Flash Builder, vous devez spécifier un comportement personnalisé aussi bien sous Exécution/Débogage d'Eclipse que sous Flash Builder > Déboguer.

L'accès à la boîte de dialogue Préférences varie en fonction de la plateforme et de la version de Flash Builder (autonome ou plug-in) utilisées.

Définition des préférences de Flash Builder

1 Ouvrez la boîte de dialogue Préférences :

- (Windows) Sélectionnez Fenêtre > Préférences.
- (Macintosh, autonome) Sélectionnez Flash Builder > Préférences.
- (Macintosh, plug-in) Sélectionnez Eclipse > Préférences.

2 Développez le nœud Flash Builder pour afficher et spécifier les préférences de l'utilisateur.

Préférences Adobe

Définissez les préférences pour les modules plug-in d'Adobe.

Configuration RDS

La configuration RDS (Remote Development Server) s'applique aux utilisateurs de LiveCycle Data Services ou d'Adobe BlazeDS. RDS donne accès aux destinations Data Services et BlazeDS.

La configuration RDS par défaut est le point de départ pour l'établissement d'une connexion aux services de données. Modifiez la configuration par défaut afin de permettre l'accès à votre serveur ou base de données.

Pour plus d'informations sur la configuration RDS, voir la documentation de LiveCycle Data Services.

Important : *la configuration RDS présente des implications en termes de sécurité. Pour plus d'informations sur les implications sur la sécurité du serveur d'applications lors de la spécification d'une configuration RDS, voir la documentation sur LiveCycle Data Services.*

Personnalisation du workbench

Vous pouvez personnaliser le workbench en fonction de vos besoins. Il vous est par exemple possible de définir l'affichage des éléments dans la barre d'outils principale, de créer des raccourcis clavier ou encore de modifier les polices et les couleurs de l'interface utilisateur.

Affichage d'autres vues Workbench

Outre les éditeurs et les vues associées aux perspectives Développement, Débogage et Profil par défaut de Flash Builder, le workbench contient des vues supplémentaires. Ces vues optionnelles sont classées par type et associées à des fonctionnalités distinctes du workbench ou à des plug-ins Eclipse spécifiques.

Pour accéder aux vues qui n'apparaissent pas déjà dans une perspective et les ajouter au workbench, sélectionnez Fenêtre > Affichage d'une vue > Autre.

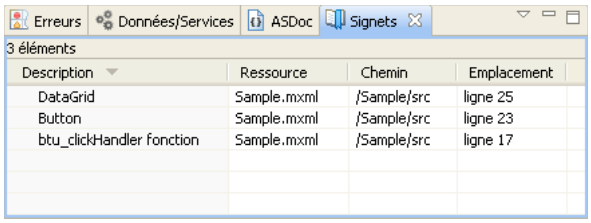
Ces vues incluent les vues Tâches, Signets et Recherche, lesquelles vous aident à simplifier le processus de développement de l'application.

Après avoir ajouté une vue à la perspective active, vous pouvez l'enregistrer en tant que partie de la perspective. Pour plus d'informations, voir « Personnalisation d'une perspective » à la page 10.

Vous pouvez également créer des vues rapides qui facilitent l'accès aux vues fréquemment utilisées. Pour plus d'informations, voir « Création et utilisation des vues rapides » à la page 11.

Vue Signets

La vue Signets sert à gérer les signets que vous ajoutez à des lignes de code spécifiques ou à des ressources. Tout comme dans un navigateur Web, les signets permettent d'effectuer le suivi d'éléments importants. La sélection d'un signet recherche et affiche ce signet dans le workbench.

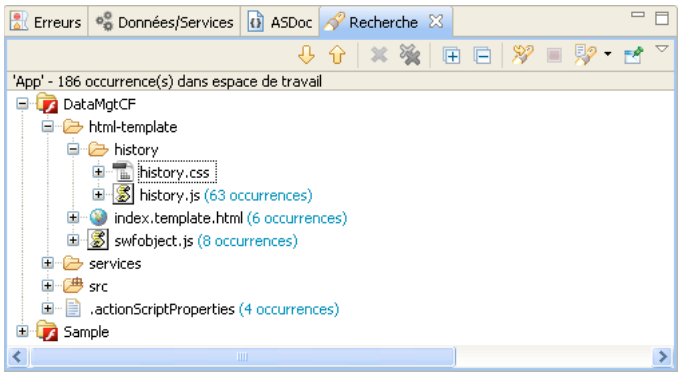


Voir aussi

« Utilisation de marqueurs » à la page 34

Vue Recherche

La vue Recherche s'affiche automatiquement lorsque vous recherchez des ressources dans l'espace de travail. Elle permet de définir des recherches, accéder à des recherches précédentes et filtrer la liste des résultats de la recherche.



Voir aussi

« Recherche de ressources dans le workbench » à la page 74

Réorganisation de la barre d'outils principale

Flash Builder vous permet de réorganiser les sections de la barre d'outils principale. Les sections de la barre d'outils principale sont séparées par un espace.

1 Vérifiez que la barre d'outils est déverrouillée. Dans le menu contextuel de la barre d'outils, désélectionnez l'option Verrouiller les barres d'outils.

2 Positionnez le pointeur sur la ligne verticale qui délimite sur la gauche la section à déplacer.

3 Cliquez sur cette ligne et déplacez la section vers la gauche, la droite, le haut ou le bas. Relâchez le bouton de la souris pour positionner la section à son nouvel emplacement.


Afin d'éviter tout déplacement non intentionnel des barres d'outils, verrouillez-les de nouveau en activant l'option correspondante de leur menu contextuel.

Modification des raccourcis clavier

1 Ouvrez la boîte de dialogue Préférences et sélectionnez Général > Touches.

2 Dans l'écran Vue de la boîte de dialogue Touches, sélectionnez la commande à modifier.

3 Dans le champ Liaison, saisissez le nouveau raccourci clavier à lier à la commande.

4 Dans la liste déroulante Quand, sélectionnez à quel moment le raccourci clavier doit être activé.

5 Cliquez sur Appliquer ou sur OK.


Pour obtenir la liste de tous les raccourcis clavier, cliquez sur Aide > Assistant de touches.

Modification des polices et des couleurs

Par défaut, le workbench utilise les polices et les couleurs du système d'exploitation de l'ordinateur. Vous pouvez cependant personnaliser les polices et les couleurs de différentes manières.

1 Ouvrez la boîte de dialogue Préférences et sélectionnez Général > Apparence > Couleurs et polices.

2 Développez les catégories Comparaison de texte, Couleurs et polices de base, CVS, Déboguer ainsi que Dossiers des éditeurs et des vues pour rechercher et sélectionner la police et les couleurs à modifier.

***Remarque :** vous pouvez également cliquer sur Utiliser la police du système au lieu de Modifier pour laisser le système d'exploitation choisir une valeur de police appropriée. Sous Windows par exemple, si vous sélectionnez cette option, Flash Builder utilise la police sélectionnée dans les propriétés d'affichage du Panneau de configuration de Windows.*

3 Définissez vos préférences en matière de police et de couleur.

Contrôle du clic et du double-clic

Vous pouvez définir la réaction du workbench à un clic ou à un double-clic.

1 Ouvrez la boîte de dialogue Préférences et sélectionnez Général.

2 Dans la section Mode d'ouverture, sélectionnez les options voulues et cliquez sur OK.

Définition des préférences du workbench

Vous pouvez définir les préférences de nombreux aspects du workbench. Vous pouvez par exemple configurer Flash Builder de manière à ce qu'il vous invite à sélectionner l'espace de travail au démarrage, définir l'éditeur à utiliser à l'ouverture de certains types de ressource ainsi que paramétrer différentes options pour l'exécution et le débogage de vos applications.

Vos préférences Flash Builder s'appliquent uniquement à l'espace de travail actuel. Vous pouvez cependant exporter les préférences du workbench et les importer dans un autre espace de travail. L'exportation des préférences est utile si vous gérez plusieurs espaces de travail ou si vous souhaitez partager les préférences du workbench avec d'autres membres de votre équipe de développement.

Vous pouvez également définir des préférences individuelles pour chaque projet d'un espace de travail, par exemple définir des options de compilation ou de débogage distinctes pour chacun de vos projets Flex.

Définition des préférences du workbench Flash Builder

1 Ouvrez la fenêtre Préférences.

2 Développez Général, choisissez l'une des catégories de préférences du workbench et modifiez-la si nécessaire.

3 Cliquez sur OK.

Préférences de Flash Builder

Flash Builder

Avertissement avant la mise à niveau d'anciens projets Flash Builder

Flash Builder met à jour les projets qui ont été créés dans l'une de ses versions antérieures. La mise à jour englobe la mise en conformité des fichiers et de la structure du projet avec la structure actuelle des projets Flash Builder. Un projet converti ne pourra plus être ouvert avec une version antérieure de Flash Builder.

Par défaut, Flash Builder vous avertit avant de procéder à la conversion. Désactivez cette option pour que les conversions s'effectuent sans avertissement.

Données/Services

Les préférences disponibles pour Données/Services sont décrites ci-dessous. Elles s'appliquent à tous les projets se trouvant dans votre environnement de développement Flash Builder.

Vous pouvez les remplacer pour chaque projet individuellement. Sélectionnez Projet > Propriétés > Données/Services pour remplacer les préférences qui y sont spécifiées.

Générateur de code

Flash Builder fournit un utilitaire de génération de code par défaut pour générer l'accès aux services de données. Faites appel aux fonctions d'extensibilité de Flash Builder pour créer vos propres utilitaires de génération de code et les ajouter à Flash Builder en tant que plug-in. Pour plus d'informations, voir Guide de référence des API d'extensibilité de Flash Builder.

Toute extension de génération de code ajoutée à Flash Builder en tant que plug-in est répertoriée dans la liste déroulante Générateur de code.

Activer l'utilisation de l'occurrence de service simple (singleton) lors de la génération de code pour le gestionnaire de services

Par défaut, cette option n'est pas activée. Au cours de la génération de code, chaque application client d'un projet crée sa propre occurrence d'un service de données.

Pour que les applications client du projet partagent une seule occurrence du service, activez cette option.

Cette option est disponible uniquement si vous sélectionnez l'utilitaire de génération de code par défaut de Flash Builder.

Débogage

Les options de débogage décrites ci-dessous sont activées par défaut. Pour les autres options affectant une session de débogage, voir :

- Préférences > Général > Navigateur Web
- Préférences > Exécution/Débogage

Avertissement en cas de tentative d'ouverture de plusieurs sessions de débogage

Certaines combinaisons de navigateurs et plateformes n'autorisent pas l'exécution en parallèle de plusieurs sessions de débogage. Le démarrage d'une seconde session de débogage entraîne la fermeture ou la déconnexion de la première session.

Ne désactivez pas cette option si vous souhaitez que Flash Builder vous avertisse de la tentative de démarrage d'une seconde session de débogage.

Signalement des erreurs de sécurité après le lancement

Dans certains cas, le navigateur Web émet une erreur de sécurité lorsqu'il ne peut pas lire le fichier de confiance de sécurité de Flash Player. Le redémarrage du navigateur Web suffit généralement à résoudre ce problème.

Ne désactivez pas cette option si vous souhaitez que Flash Builder vous avertisse de ce type d'erreur de sécurité.

Voir la note technique pour l'avertissement de sécurité de Flash Player.

Appel automatique des fonctions getter

Au cours des étapes d'une session de débogage, les variables représentant les fonctions d'accesseur (getter) sont automatiquement évaluées. Ce comportement est généralement utile au cours d'une session de débogage.

Désactivez cette option si vous ne souhaitez pas évaluer automatiquement les variables représentant des fonctions d'accesseur lors des étapes d'une session de débogage.

Gestion de la version du lecteur Flash Player autonome

Lorsque vous lancez ou déboguez une application à partir de Flash Builder, vous pouvez spécifier l'exécutable de Flash Player à utiliser.

Pour ce faire, sélectionnez Flash Builder > Déboguer dans la boîte de dialogue Préférences. Définissez ensuite l'emplacement du fichier exécutable de Flash Player que Flash Builder utilise pour déboguer ou lancer votre application.

Configurations de périphériques

Flash Builder utilise des configurations de périphérique lors de l'affichage de l'aperçu des tailles d'écran de périphérique sur l'ordinateur à l'aide d'AIR Debug Launcher. Voir « Gestion des configurations de lancement » à la page 105.

Flash Builder fournit des configurations par défaut pour les téléphones Nexus One, Droid, Apple iPad, Apple iPhone, HTC et Samsung Android. Vous ne pouvez pas modifier les configurations par défaut.

Editeurs Flash Builder

Général

Flash Builder fournit des options utilisateur pour la réduction du code, en signalant les problèmes à mesure que vous saisissez le code et en utilisant le positionnement du caret au niveau du sous-mot.

Accolades

Flash Builder comporte des options pour la mise en retrait, l'insertion et la mise en évidence d'accolades représentant des blocs de code.

Assistant de contenu

L'éditeur de source MXML ou ActionScript de Flash Builder fournit des conseils de code qui vous aident à compléter vos expressions de code. Il facilite par exemple la sélection des classes recommandées, des propriétés et des événements disponibles.

- **Activer l'activation automatique**

  Désactivez l'option Activer l'activation automatique si vous ne souhaitez pas que les conseils de code soient automatiquement affichés en cours de frappe. Si cette option est désactivée, vous pouvez accéder aux conseils de code en appuyant sur les touches Ctrl+Espace.

- **Activer après**

  Spécifiez le délai en millisecondes devant s'écouler entre le moment auquel vous commencez à saisir du code et le moment auquel les conseils de code doivent s'afficher. Par défaut, ce délai est de 100 millisecondes.

- **Utiliser des raccourcis personnalisés supplémentaires**

  Vous pouvez non seulement accéder aux conseils de code en appuyant sur Ctrl + Espace, mais aussi spécifier vos propres touches de raccourci.

  Saisissez les touches que vous souhaitez utiliser comme raccourcis pour accéder aux conseils de code dans le code MXML ou ActionScript. Si vous tapez sur une seule touche du raccourci, des conseils de code sont appelés. Par exemple, si vous spécifiez comme touches de raccourci `ab`. Une pression sur `a` ou sur `b` appelle l'assistant de code.

Voir aussi

« Outils de développement de code dans Flash Builder » à la page 16

Mise en retrait

Flash Builder fournit des options utilisateur permettant la mise en retrait à l'aide de tabulations au lieu d'espaces et la mise en retrait automatique des nouvelles lignes de saisie.

Préférences générales de l'éditeur pour Flash Builder

Par défaut, Flash Builder réduit le code et met automatiquement les lignes en retrait en cours de saisie ou de collage de texte dans l'éditeur de source. Vous pouvez désactiver les préférences par défaut de ces fonctions.

Code ActionScript

Lorsque vous modifiez des fichiers ActionScript dans l'éditeur de source, Flash Builder fournit des fonctions par défaut pour le renvoi à la ligne du code.

Par défaut, la documentation de référence du langage ActionScript disponible s'affiche avec les conseils de code ou en survolant un élément de langage avec le curseur de la souris. Chacune de ces fonctions par défaut peut être désactivée.

Lorsque vous saisissez un code ActionScript dans les éditeurs de Flash Builder, l'assistant de contenu vous propose une liste d'options pour terminer votre expression de code. Vous pouvez compléter automatiquement votre code ActionScript partiellement saisi en utilisant des touches de raccourci de remplissage. Pour ce faire, Flash Builder fournit des touches de raccourci par défaut. Vous pouvez choisir de remplacer les touches de raccourci par défaut par vos propres raccourcis. Vous pouvez aussi choisir d'insérer le code en appuyant sur la barre d'espacement.

Cycle d'assistance au contenu

Lors de la saisie de code ActionScript, appuyez plusieurs fois sur les touches Ctrl+Espace pour faire défiler les filtres pour les conseils de code affichés.

Vous pouvez spécifier les conseils de code à afficher, ainsi que l'ordre de défilement des conseils de code. Pour modifier le paramètre par défaut, procédez comme suit :

1. Ouvrez la boîte de dialogue Préférences et sélectionnez Flash Builder > Editeurs > Code ActionScript > Cycle d'assistance au contenu.
2. Sélectionnez un type d'indicateur et modifiez-en la position dans le cycle de défilement en cliquant sur les boutons Haut et Bas.

   Le premier type d'indicateur de la liste sera affiché en premier.

Mise en retrait

Par défaut, Flash Builder met en retrait le contenu du package, les déclarations de fonction et les instructions switch. Chacune de ces fonctions par défaut peut être désactivée.

Organisation des importations

Vous pouvez gérer l'organisation des instructions d'importation dans les blocs de script MXML et ActionScript à l'aide de la boîte de dialogue Préférences.

Ouvrez la boîte de dialogue Préférences, sélectionnez Flash Builder > Editeurs > ActionScript > Organiser les importations et spécifiez les préférences suivantes :

- Par défaut, Flash Builder organise les instructions d'importation par ordre alphabétique. Pour modifier l'ordre par défaut, utilisez les boutons Vers le haut, Vers le bas et Supprimer. Vérifiez que l'option Garder les importations organisées est sélectionnée afin que l'ordre de tri spécifié soit appliqué.
- Pour ajouter un package au groupe d'importation, cliquez sur Nouveau et saisissez un préfixe de nom de package. Par exemple, saisissez `monpackage`.

  Lorsque vous référencez une classe d'un package non répertorié, l'instruction d'importation de cette classe est ajoutée après toutes les instructions d'importation triées.

- Sélectionnez l'option Consolider les instructions d'importation des blocs de script MXML pour consolider toutes les instructions d'importation des blocs de script MXML. Les instructions d'importation sont consolidées, triées et ajoutées une seule fois en haut du premier bloc de script.
- Vous pouvez spécifier le nombre d'instructions d'importation autorisées par type dans le même package avant que le caractère générique `<package>.*` ne soit utilisé. La valeur par défaut est 99.

Voir aussi

« Assistant de contenu, Assistant rapide et Correctif rapide » à la page 16

« Organisation des instructions d'importation » à la page 36

Modèles de code

Vous pouvez personnaliser le code de remplacement prédéfini généré par Flash Builder lorsque vous créez des méthodes ou des fonctions d'accesseur get et set, générez des gestionnaires d'événement ou remplacez des méthodes.

Flash Builder fournit également des modèles de code prédéfinis qui vous permettront de créer rapidement du code MXML, ActionScript et CSS. Vous pouvez créer et modifier des modèles de code. Vous pouvez également importer et exporter les fichiers de modèles de code.

Pour plus d'informations à propos de l'utilisation, de la création et de la modification des modèles de code, voir « Création et modification de modèles de code » à la page 42.

Code MXML

Lorsque vous éditez des fichiers MXML dans l'éditeur de source, Flash Builder fournit des fonctions par défaut de suggestion de code. Chacune de ces fonctions peut être désactivée.

Cycle d'assistance au contenu

Flash Builder définit un comportement par défaut pour l'assistant de contenu lors de l'édition de fichiers MXML. Lorsqu'il est disponible, l'assistant de contenu affiche des indicateurs sur les éléments de langage pouvant être insérés dans le code. Pour parcourir les éléments affichés dans la fenêtre de l'assistant de contenu, appuyez plusieurs fois sur les touches Ctrl+Espace.

Vous pouvez spécifier les conseils de code à afficher, ainsi que l'ordre de défilement des conseils de code. Pour modifier le paramètre par défaut, procédez comme suit :

1. Ouvrez la boîte de dialogue Préférences et sélectionnez Flash Builder > Editeurs > Code MXML > Cycle d'assistance au contenu.
2. Spécifiez les types d'indicateurs que vous souhaitez afficher.
3. Sélectionnez un type d'indicateur et modifiez-en la position dans le cycle de défilement en cliquant sur les boutons Haut et Bas.

   Le premier type d'indicateur de la liste sera affiché en premier.

Voir aussi

« Assistant de contenu, Assistant rapide et Correctif rapide » à la page 16

« Outils de développement de code dans Flash Builder » à la page 16

Formatage

Par défaut, Flash Builder formate le code MXML généré automatiquement. Par exemple, Flash Builder formate le code MXML autogénéré lorsque vous utilisez des outils Flash Builder pour vous connecter à un service de données et accéder à des données.

Vous pouvez modifier l'ordre et le regroupement des attributs dans la boîte de dialogue des préférences de formatage.

Le formatage ne s'applique pas au code rédigé à la main. Pour formater un fragment de code rédigé à la main, sélectionnez Source > Appliquer le formatage (Ctrl+Maj+F). Le code est ensuite formaté, que vous ayez ou non sélectionné l'option Maintenir les attributs MXML organisés.

Mise en retrait

Par défaut, la mise en retrait du code s'effectue par tabulations. Ce paramètre peut être modifié en faveur de l'utilisation d'espaces. La taille de la tabulation et de la mise en retrait par défaut équivaut à quatre espaces.

Sélectionnez ActionScript ou MXML sous Mise en retrait dans la boîte de dialogue Préférences afin d'afficher les paramètres correspondants. Vous pouvez également personnaliser les méthodes heuristiques de mise en retrait.

Coloration de la syntaxe

La coloration de la syntaxe et la mise en évidence de texte dans les fichiers ActionScript, CSS et MXML sont définies par défaut. Ces paramètres par défaut peuvent être remplacés. Développez le nœud du langage et sélectionnez une fonction de langage pour remplacer la fonction par défaut.

Les préférences générales de l'éditeur pour Eclipse sont également disponibles. Voir :

- Préférences > Général > Editeurs de texte
- Préférences > Général > Couleurs et polices

Préférences générales de l'éditeur pour Eclipse

Pour définir d'autres préférences générales de l'éditeur, sélectionnez Préférences > Général > Editeurs.

Associations de fichiers

Vous pouvez sélectionner Flash Builder comme application par défaut pour l'ouverture des fichiers MXML et ActionScript. La spécification d'une association de fichiers remplace les paramètres du système d'exploitation.

1. Ouvrez la boîte de dialogue Préférences et sélectionnez Flash Builder > Associations de fichiers.
2. Sélectionnez les options Fichiers ActionScript (*.as) et Fichiers MXML (*.mxml) pour définir les associations de fichiers.

Exclusions de fichiers

Vous pouvez spécifier les types de fichier que vous souhaitez que Flash Builder ne copie pas dans le dossier de sortie au cours de la compilation du projet.

1. Ouvrez la boîte de dialogue Préférences, puis sélectionnez Flash Builder > Exclusions de fichiers.
2. Vous pouvez ajouter des extensions de nom de fichier et des noms de fichier, ou supprimer des éléments existants de la liste.
3. Pour ajouter des extensions de nom de fichier ou des noms de fichier, cliquez sur Nouveau, puis indiquez une extension de nom de fichier ou un nom de fichier valide.

4 Pour restaurer la liste par défaut des extensions à n'importe quel moment, cliquez sur Restaurer les valeurs par défaut.

Modèles de fichiers

Les modèles de fichiers interviennent lors de la création de nouveaux fichiers MXML, ActionScript et CSS et peuvent être personnalisés. Vous pouvez également importer et exporter des fichiers de modèles.

Dans la zone Types de fichier, sélectionnez le modèle d'un type de fichier. Cliquez sur Editer pour le modifier. Désactivez l'option Mise en retrait automatique des nouveaux fichiers si vous ne souhaitez pas faire appel aux préférences de mise en retrait de Flash Builder lors de la création de fichiers.

Vous pouvez modifier les préférences de mise en retrait de Flash Builder. Accédez à Préférences > Flash Builder > Mise en retrait.

Voir aussi

« Personnalisation des modèles de fichier » à la page 48

« Mise en retrait » à la page 261

FlexUnit

Mode de lancement par défaut
Lorsque vous exécutez un test FlexUnit, Flash Builder est lancé par défaut en mode de débogage.

Cette préférence vous permet de modifier le mode de lancement de Flash Builder pour les tests FlexUnit.

Les modes de lancement par défaut de Flash Builder sont les suivants :

- Mode Exécuter

  Par défaut, ce mode lance la perspective Flash.

- Mode Déboguer

  Par défaut, ce mode lance la perspective Débogage Flash.

- Mode Profiler

  Par défaut, ce mode lance la perspective Profil Flash.

Les préférences Eclipse pour le lancement et les perspectives définissent les modes de lancement. Voir :

- Préférences > Exécution/Débogage > Lancement
- Préférences > Exécution/Débogage > Perspectives

Nom de l'application personnalisée
Spécifiez le nom complet du fichier de l'application FlexUnit personnalisée. Si le fichier de l'application FlexUnit personnalisée ne se trouve pas dans le projet Flex sélectionné, Flash Builder génère automatiquement un fichier d'application FlexUnit par défaut.

Numéro de port
Spécifie le numéro de port à utiliser pour connecter Flash Builder à l'application FlexUnit sur le périphérique mobile. Le numéro de port présente une valeur de 8765 codée en dur.

Utilisation d'une structure alternative

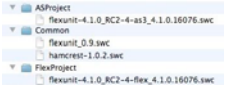
Par défaut, les fichiers SWC de la structure qui sont disponibles avec Flash Builder sont ajoutés au chemin de génération FlexUnit.

Pour indiquer une structure FlexUnit différente, sélectionnez Use Alternative Framework. Accédez à l'emplacement du répertoire de la structure que vous souhaitez sélectionner.

Lorsque vous créez une classe de cas de test ou une suite de cas de test FlexUnit pour le projet sélectionné, Flash Builder ajoute automatiquement les fichiers SWC figurant dans le répertoire de la structure alternative au chemin de génération de FlexUnit.

Lorsque vous créez un répertoire de structure FlexUnit alternative, il est recommandé d'utiliser la structure de répertoire suivante pour les fichiers SWC :

- Fichiers `flexunit-x.x-as3_x.x.swc` dans le dossier « ASProject ». Les fichiers SWC présents dans ce dossier sont automatiquement ajoutés au chemin de génération d'un projet ActionScript.
- Fichiers `flexunit-x.x-flex_x.x.swc` dans le dossier « FlexProject ». Les fichiers SWC présents dans ce dossier sont automatiquement ajoutés au chemin de génération d'un projet Flex.
- Les éventuels autres fichiers SWC requis par votre projet ou par rapport auxquels votre code FlexUnit présente des dépendances (qui ne figurent pas encore dans le chemin de génération) dans le dossier "Commun" Les fichiers SWC présents dans ce dossier sont automatiquement ajoutés au chemin de génération des projets ActionScript et Flex.



Si Flash Builder ne détecte pas la structure de répertoire recommandée, le répertoire alternatif sélectionné est ajouté, qu'il s'agisse d'un projet Flex ou ActionScript.

Paul Robertson, professionnel de la communauté Adobe, a publié sur son blog un article concernant la création d'un framework FlexUnit personnalisé.

Voir aussi

« Environnement de test FlexUnit » à la page 164

« Perspectives et vues de Flash Builder » à la page 6

SDK AIR installé

Flash Builder 4.7 installe le SDK AIR à utiliser pour l'ensemble des projets ActionScript. Ce SDK contient le SDK AIR d'Adobe, le compilateur ActionScript ainsi que d'autres composants et fichiers requis.

Par défaut, Flash Builder installe le SDK Adobe AIR 3.4. Cliquez sur Rechercher des mises à jour pour rechercher et télécharger toute version mise à jour disponible du SDK AIR.

Lors de l'importation des projets ActionScript créés dans une version antérieure de Flash Builder, les projets sont mis à niveau automatiquement en vue de l'utilisation du SDK AIR. Vous pouvez choisir d'être averti par Flash Builder en cas d'une telle mise à niveau.

SDK Flex installés

Flash Builder utilise le SDK Flex pour tous les projets Flex. Flash Builder installe les SDK Flex 4.6 et 3.6A. Par défaut, Flash Builder utilise le SDK Flex 4.6 pour les projets qui ne requièrent pas de SDK spécifique.

Vous pouvez ajouter et supprimer des SDK Flex ainsi que modifier le SDK Flex utilisé par défaut pour les projets qui ne requièrent pas de SDK spécifique.

Les types de SDK Flex disponibles sont les suivants :

SDK Adobe Flex : versions publiées de produits Adobe officiels. Ces SDK contiennent différents composants source ouverts et fermés.

Pour ajouter un SDK Adobe Flex, effectuez les étapes suivantes :

1 Téléchargez le SDK Flex requis vers un emplacement temporaire de votre système depuis la page de téléchargements de SDK Flex (en anglais).

Consultez la page de téléchargement pour obtenir des informations sur les différentes version, les conditions requises et les licences.

Remarque : *si vous souhaitez utiliser une version stable et testée, téléchargez une version d'étape. Cette version a été déclarée comme étant une version majeure par l'équipe de développement. Elle comporte une version signée des RSL de la structure Flex (requis pour utiliser la mise en cache RSL).*

2 Les SDK Flex téléchargés sont au format ZIP. Décompressez le fichier ZIP dans un nouveau dossier sur votre système.

Pour plus de clarté, vous pouvez attribuer au dossier de niveau supérieur le numéro de version du SDK.

Les SDK sont généralement placés dans le répertoire d'installation de Flash Builder à l'emplacement suivant :

`<Flash Builder Install Directory>`/sdks/

3 Dans les préférences de Flash Builder, sélectionnez Flash Builder > SDK installés.

4 Cliquez sur Ajouter, puis recherchez le SDK Flex décompressé. Cliquez sur Appliquer.

5 Pour chaque projet utilisant le SDK Flex, accédez à cette boîte de dialogue et sélectionnez le SDK qui sera utilisé par défaut. Cliquez sur OK.

Lors de l'importation de projets créés dans une version antérieure de Flash Builder, vous êtes invité à sélectionner le SDK à utiliser. Vous pouvez désactiver l'affichage de cette invite à l'importation de projets.

SDK Apache Flex : destiné aux utilisateurs souhaitant télécharger et utiliser le SDK Apache Flex en source libre.

Pour plus d'informations sur le téléchargement et l'utilisation du SDK Apache Flex, voir www.adobe.com/go/apacheflex_download.

Moniteur de réseau

La page de préférences Moniteur de réseau répertorie les ports sur lesquels le Moniteur de réseau intercepte des événements et écoute les requêtes HTTP.

Par défaut, le Moniteur de réseau supprime au démarrage toutes les entrées. Cette préférence peut être désactivée.

Vous pouvez également activer les préférences suivantes :

- Suspendre la surveillance au démarrage
- Ignorer les vérifications de sécurité SSL

  L'activation de cette préférence est utile pour surveiller le trafic réseau à partir d'un serveur autosigné.

Profileur

Par défaut, le profilage de la mémoire et le profilage des performances sont tous les deux activés.

- Profilage de la mémoire

  Le profilage de la mémoire permet de déterminer la quantité de mémoire utilisée par un objet ou un type d'objet dans l'application. Faites appel au profilage de la mémoire pour diminuer la taille des objets, réduire le nombre d'objets créés ou permettre le nettoyage des objets dans la mémoire en supprimant les références à ces objets.

  La quantité de mémoire utilisée par le profilage de la mémoire est bien supérieure à celle utilisée par le profilage des performances et peut donc ralentir votre application.

- Profilage des performances

  Le profilage des performances recherche les méthodes de l'application dont l'exécution est lente et qui sont susceptibles de pouvoir être améliorées ou optimisées.

Vous pouvez définir les préférences suivantes pour le profileur :

- Connexions

  Spécifiez le numéro de port que Flash Builder écoute lors du profilage d'une application. Le numéro de port par défaut est 9999. Ce port ne peut pas être défini sur 7935, ce dernier étant utilisé par le débogueur Flash.

- Filtres d'exclusion

  Définit les packages par défaut qui sont exclus des vues du profileur.

- Filtres d'inclusion

  Définit les packages par défaut qui sont inclus dans les vues du profileur. Tous les autres packages sont exclus.

- Références d'objet

  Spécifie le nombre de chemins de références arrière à des occurrences d'un objet à afficher. Les chemins de références arrière permettent de déterminer si un chemin possède une référence arrière à la récupération de place (racine de la RP). Une occurrence dont la libération est attendue mais qui possède des références à la racine de la RP indique une perte de mémoire.

  Par défaut, le profileur affiche dix chemins de références arrière. Vous pouvez augmenter cette valeur ou activer l'option d'affichage de tous les chemins de références arrière.

- Lecteur/Navigateur

  Vous pouvez spécifier quel lecteur Adobe Flash Player autonome et quel navigateur Web vous voulez utiliser pour le profilage des applications externes. Utilisez une version de débogage de Flash Player pour profiler une application.

  Lors du profilage d'applications externes, Flash Builder utilise par défaut les lecteurs Flash Player suivants :

  - URL vers un fichier SWF

    Flash Builder lance le fichier SWF de l'application en utilisant le navigateur par défaut du système.

  - Emplacement du fichier SWF sur le système de fichiers

    Flash Builder ouvre l'application avec la version par défaut du débogueur du lecteur Flash Player autonome.

Voir aussi

« Utilisation du profileur Flash Builder » à la page 152

« Vues du profileur » à la page 136

Plateformes cible

Pour le développement d'applications mobiles destinées à la plateforme Android, Flash Builder installe les parties nécessaires du SDK Android.

Sélection d'une interface réseau principale

Lorsque vous déboguez votre application via le réseau, votre machine hôte peut être connectée à plusieurs interfaces réseau simultanément. Toutefois, vous pouvez sélectionner une interface réseau principale à utiliser pour le débogage dans le package APK Android ou le package iOS de débogage.

Voir aussi

« Test et débogage » à la page 239Test et débogage