Présentation du langage XML

Le langage XML (Extensible Markup Language) est en passe de devenir la norme d'échange de données structurées dans les applications Internet. Vous pouvez intégrer les données de Flash avec des serveurs qui utilisent la technologie XML pour construire des applications sophistiquées telles que des services de discussion en ligne ou de courtage.

En XML, tout comme en HTML, vous utilisez des balises pour *marquer*, ou définir, une chaîne de texte. Dans le langage HTML, vous utilisez des balises prédéfinies pour indiquer la façon dont le texte doit apparaître dans un navigateur Web (par exemple, la balise `<b>` indique que le texte doit être en gras). Dans le langage XML, vous définissez des balises qui identifient le type d'une partie de données (par exemple, `<password>VerySecret</password>`). Le langage XML sépare la structure des informations de leur mode d'affichage, ce qui permet d'utiliser un même document XML dans des environnements différents.

Chaque balise XML est appelée *nœud* ou élément. Chaque nœud possède un type (1, qui indique un élément XML ou 3, qui indique un nœud texte) et les éléments peuvent également posséder des attributs. Un nœud imbriqué dans un autre est appelé *nœud enfant*. Cette structure hiérarchique de nœuds est appelée DOM XML, un peu comme le DOM JavaScript, qui correspond à la structure des éléments dans un navigateur Web.

Dans l'exemple suivant, `<portfolio>` est le nœud parent ; il ne comporte pas d'attributs et contient le nœud enfant `<holding>`, qui a les attributs `symbol`, `qty`, `price` et `value` :

```
<portfolio>
  <holding symbol="rich"
    qty="75"
    price="245.50"
    value="18412.50" />
</portfolio>
```

Pour plus d'informations, consultez les sections suivantes :

- « Utilisation de la classe XML », à la page 653
- « Utilisation de la classe XMLSocket », à la page 659

Pour plus d'informations sur XML, consultez www.w3.org/XML.

Pour un exemple de fichier source, xml_blogTracker.fla, qui décrit comment créer un traqueur de journaux du Web en chargeant, en analysant et en manipulant des données XML, consultez la page des exemples Flash à l'adresse www.adobe.com/go/learn_fl_samples_fr. Téléchargez le fichier zip Exemples et naviguez jusqu'au dossier ActionScript2.0/XML_BlogTracker afin d'accéder à cet exemple.

Pour un exemple de fichier source, xml_languagePicker.fla, qui décrit comment utiliser le langage XML et des tableaux imbriqués pour sélectionner des chaînes de langages différents afin de remplir les champs de texte, consultez la page des exemples Flash à l'adresse www.adobe.com/go/learn_fl_samples_fr. Téléchargez le fichier zip Exemples et naviguez jusqu'au dossier ActionScript2.0/XML_LanguagePicker afin d'accéder à cet exemple.

Pour un exemple de fichier source, xmlmenu.fla, qui décrit comment créer un menu dynamique avec des données XML, consultez la page des exemples Flash à l'adresse www.adobe.com/go/learn_fl_samples_fr. Téléchargez le fichier zip Exemples et naviguez jusqu'au dossier ActionScript2.0/XML_Menu afin d'accéder à cet exemple. Cet exemple appelle le constructeur ActionScript `XmlMenu()` et lui transmet deux paramètres : le chemin au fichier menu XML et une référence au scénario. Le reste de la fonctionnalité réside dans un fichier de classe personnalisée, XmlMenu.as.

Utilisation de la classe XML

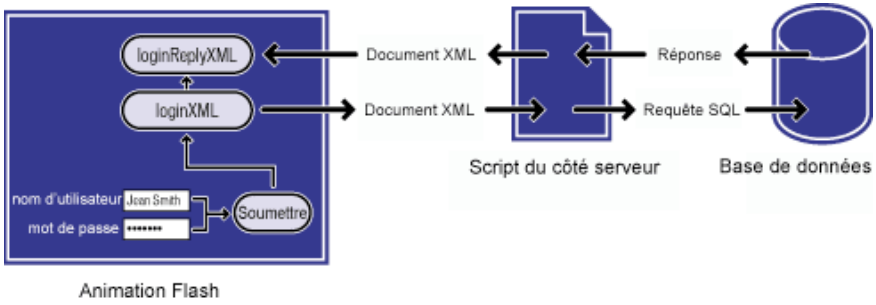
Les méthodes de la classe XML ActionScript (par exemple, `appendChild()`, `removeNode()` et `insertBefore()`) permettent de structurer les données XML dans Flash qui doivent être envoyées à un serveur et de manipuler et interpréter les données XML téléchargées.

Les méthodes de la classe XML suivantes permettent d'échanger des données XML avec un serveur à l'aide de la méthode HTTP `POST` :

- La méthode `load()` télécharge le code XML depuis une URL et le place dans un objet XML ActionScript.
- La méthode `send()` code l'objet XML dans le document XML et l'envoie à une URL spécifiée en utilisant la méthode `POST`. Si spécifié, les résultats s'affichent dans une fenêtre de navigateur.
- La méthode `sendAndLoad()` envoie un objet XML à une URL. Toutes les informations renvoyées sont placées dans un objet XML ActionScript.

Par exemple, vous pourriez créer un système de courtage qui stockerait toutes ses informations (noms d'utilisateur, mots de passe, identifiants de session, contenu des portefeuilles et informations de transaction) dans une base de données.

Le script côté serveur qui transmet les informations entre Flash et la base de données lit et écrit les données au format XML. Vous pouvez utiliser ActionScript pour convertir les informations récupérées dans le fichier SWF (par exemple, un nom d'utilisateur et un mot de passe) en un objet XML et envoyer ensuite les données au script côté serveur sous forme de document XML. Vous pouvez également utiliser ActionScript pour charger le document XML que le serveur renvoie dans un objet XML à utiliser dans le fichier SWF.



Flux et conversion des données entre un fichier SWF, un script côté serveur et une base de données

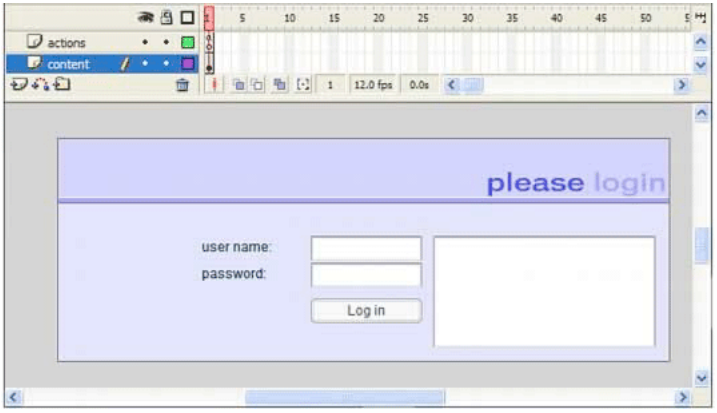
La validation du mot de passe pour le système de courtage nécessite deux scripts : une fonction définie dans l'image 1 et un script qui crée, puis envoie les objets XML créés dans le document.

Lorsqu'un utilisateur entre des informations dans les champs de texte du fichier SWF avec les variables `username` et `password`, les variables doivent être converties au format XML avant d'être transmises au serveur. La première section du script charge les variables dans un objet XML nouvellement créé et appelé `loginXML`. Quand un utilisateur clique sur un bouton pour se connecter, l'objet `loginXML` est converti en chaîne XML et envoyé au serveur.

Le code ActionScript suivant est placé sur le scénario et est utilisé pou envoyer les données formatées XML au serveur. Pour comprendre le script, vous pourrez vous aider des commentaires (indiqués par les caractères `//`) :

```
//ignorer les espaces blancs XML
XML.prototype.ignoreWhite = true;
// Construire un objet XML contenant la réponse du serveur
var loginReplyXML:XML = nouveau XML();
// cette fonction se déclenche quand un paquet XML est reçu du serveur.
loginReplyXML.onLoad = function(success:Boolean) {
  if (success) {
    //(facultatif) Crée deux champs texte pour l'état/débogage
    //status_txt.text = this.firstChild.attributes.status;
    //debug_txt.text = this.firstChild;
    switch (this.firstChild.attributes.STATUS) {
```

```
      case 'OK' :
        _global.session = this.firstChild.attributes.SESSION;
        trace(_global.session);
        gotoAndStop("welcome");
        break;
      case 'FAILURE' :
        gotoAndStop("loginfailure");
        break;
      default :
        // ceci ne devrait jamais arriver
        trace("Unexpected value received for STATUS.");
      }
    } else {
      trace("an error occurred.");
    }
};
// cette fonction se déclenche lorsque la connexion login_btn est cliquée
target_mc.onRelease = function() {
  var loginXML:XML = new XML();
  //crée des données au format XML à envoyer au serveur
  var loginElement:XMLNode = loginXML.createElement("login");
  loginElement.attributes.username = username_txt.text;
  loginElement.attributes.password = password_txt.text;
  loginXML.appendChild(loginElement);
  // Envoie les données mises au format XML au serveur
  loginXML.sendAndLoad("http://www.flash-mx.com/mm/main.cfm",
  loginReplyXML);
};
```



Vous pouvez tester ce code avec le nom d'utilisateur `JeanSmith` et le mot de passe `VerySecret`. La première section du script génère le code XML suivant lorsque l'utilisateur clique sur le bouton de connexion :

```
<login username="JeanSmith" password="VerySecret" />
```

Le serveur reçoit le code XML, génère une réponse XML et la renvoie au fichier SWF. Si le mot de passe est accepté, le serveur envoie la réponse suivante :

```
<LOGINREPLY STATUS="OK" SESSION="4D968511" />
```

Ce XML comprend un attribut `session` qui contient un ID unique, de session générée au hasard, qui est utilisé dans toutes les communications entre le client et le serveur pour le reste de la session. Si le mot de passe est rejeté, le serveur répond par le message suivant :

```
<LOGINREPLY STATUS="FAILURE" />
```

Le noeud XML`loginreply` doit charger dans un objet XML vide dans le fichier SWF. L'instruction suivante crée l'objet XML `loginreplyXML` pour recevoir le nœud XML :

```
// Construire un objet XML contenant la réponse du serveur
var loginReplyXML:XML = nouveau XML();
loginReplyXML.onLoad = function(success:Boolean) {
```

La deuxième déclaration dans ce code ActionScript définit une fonction anonyme (inline), qui est appelée quand l'événement `onLoad` se déclenche.

Le bouton de connexion (occurrence `login_btn`) permet d'envoyer le nom d'utilisateur et le mot de passe au format XML au serveur et de charger la réponse XML dans le fichier SWF. Vous pouvez utiliser la méthode `sendAndLoad()` pour ce faire, comme indiqué dans l'exemple suivant :

```
loginXML.sendAndLoad("http://www.flash-mx.com/mm/main.cfm", loginReplyXML);
```

Tout d'abord, les données au format XML sont créées, en utilisant les valeurs que l'utilisateur entre dans le fichier SWF. Cet objet XML est ensuite envoyé avec la méthode `sendAndLoad`. Comme pour les données provenant de la fonction `loadVariables()`, l'élément XML `loginreply` arrive de façon asynchrone (sans attendre les résultats avant d'être renvoyé) et se charge dans l'objet `loginReplyXML`. Lorsque les données arrivent, le gestionnaire `onLoad` de l'objet `loginReplyXML` est appelé. Vous devez définir la fonction `loginReplyXML`, qui est appelée lorsque le gestionnaire `onLoad` se déclenche, de façon à pouvoir traiter l'élément `loginreply`.

REMARQUE	Cette fonction doit toujours figurer sur l'image qui contient le code ActionScript relatif au bouton connexion.

Si la connexion aboutit, le fichier SWF passe à l'étiquette d'image `welcome`. Si la connexion ne réussit pas, la tête de lecture se place sur l'étiquette d'image `loginfailure`. Ceci a été traité en utilisant une condition et une instruction case.

Le fichier CFM contient le code suivant :

```
<cfif (Compare(Form.username, "Herbert") EQ 0) AND (Compare(Form.password,
  "glasses") EQ 0)>
  <cfoutput>&isValidLogin=1&session=#URLEncodedFormat(CreateUUID())#</
  cfoutput>
  <cfelse>
  <cfoutput>&isValidLogin=0</cfoutput>
</cfif>
```

Pour plus d'informations sur les instructions case et break, consultez les sections instruction case et instruction break dans le *Guide de référence du langage ActionScript 2.0*. Pour plus d'informations sur les conditions, consultez les sections instruction if et instruction else dans le *Guide de référence du langage ActionScript 2.0*.

REMARQUE	Ce design est seulement un exemple, et Adobe ne veut rien dire au sujet du niveau de sécurité qu'il fournit. Si vous mettez en œuvre un système sécurisé protégé par mot de passe, assurez-vous de bien connaître les concepts de la sécurité réseau.

Pour plus d' informations, consultez Intégration de XML et Flash dans une application Web à l'adresse http://www.adobe.com/support/flash/applications/xml/ et l'entrée XML dans le *Guide de référence du langage ActionScript 2.0*. Pour plus d'informations sur la sécurité de fichiers locaux, consultez « Sécurité des fichiers locaux et Flash Player », à la page 677.

Pour un exemple de fichier source, login.fla, qui décrit comment ajouter une fonctionnalité de connexion à vos sites Web à l'aide d'ActionScript 2.0, consultez la page des exemples Flash à l'adresse www.adobe.com/go/learn_fl_samples_fr. Téléchargez le fichier zip Exemples et naviguez jusqu'au dossier ActionScript2.0/Login afin d'accéder à cet exemple. Cet exemple utilise ActionScript et ses composants pour créer un petit formulaire dans lequel vous entrez un nom d'utilisateur et un mot de passe, puis cliquez un bouton pour entrer sur un site.

Flash Player 8 et versions ultérieures prennent en charge le gestionnnaire d'événement `onHTTPStatus` pour les classes XML, LoadVars et MovieClipLoader afin de permettre aux utilisateurs d'accéder au code de statut à partir d'une requête HTTP. Ceci permet aux développeurs de déterminer *pourquoi* une opération de chargement particulière peut échouer au lieu de pouvoir déterminer que l'opération de chargement a déjà échoué.

L'exemple suivant montre comment vous pouvez utiliser le gestionnaire d'événement `onHTTPStatus` de la classe XML pour vérifier si un fichier XML a été téléchargé avec succès à partir d'un serveur et ce qu'était le code de statut retourné à partir de la requête HTTP.

Vérification des codes d'état HTTP en utilisant la classe XML

1. Créez un document Flash, puis enregistrez-le sous le nom de **fileref.fla**.
2. Ajoutez le code ActionScript suivant à l'image 1 du scénario principal :

```
var my_xml:XML = new XML();
my_xml.ignoreWhite = true;
my_xml.onHTTPStatus = function(httpStatus:Number) {
  trace("HTTP status is: " + httpStatus);
};
my_xml.onLoad = function(success:Boolean) {
  if (success) {
    trace("XML successfully loaded");
    // 0 (Pas d'erreur : l'analyse est terminée.)
    trace("XML status is: " + my_xml.status);
  } else {
    trace("unable to load XML");
  }
};
my_xml.load("http://www.helpexamples.com/crossdomain.xml");
```

   Le code précédent définit un nouvel objet XML avec le nom variable `my_xml`, définit deux gestionnaires d'événement (`onHTTPStatus` et `onLoad`) et charge un fichier externe XML. Le gestionnaire d'événement `onLoad` s'assure que le fichier XML a été chargé.
   Dans l'affirmative, il envoie un message au panneau Sortie, ce qui permet de suivre la propriété d'état de l'objet XML. Il est important de se rappeler que le gestionnaire d'événement `onHTTPStatus` renvoie le code d'état retourné du serveur web, tandis que la propriété `XML.status` contient une valeur numérique qui indique si l'objet XML a été analysé correctement.
3. Choisissez Contrôle > Tester l'animation pour tester le document Flash.

AVERTISSEMENT

Ne pas confondre les codes httpStatus HTTP avec la propriété `status` de la classe XML. Le gestionnaire d'événement `onHTTPStatus` renvoie le code d'état du serveur d'une requête HTTP et la propriété `status` définit et renvoie automatiquement une valeur numerique qui indique si le document XML a été transformé correctement en objet XML.

ATTENTION

Si un serveur Web ne renvoie pas un code `status` à Flash Player, le nombre 0 est renvoyé au gestionnaire d'événement `onHTTPStatus`.

Pour un exemple de fichier source, xml_blogTracker.fla, qui décrit comment créer un traqueur de journaux du Web en chargeant, en analysant et en manipulant des données XML, consultez la page des exemples Flash à l'adresse www.adobe.com/go/learn_fl_samples_fr. Téléchargez le fichier zip Exemples et naviguez jusqu'au dossier ActionScript2.0/XML_BlogTracker afin d'accéder à cet exemple.

Pour un exemple de fichier source, xml_languagePicker.fla, qui décrit comment utiliser le langage XML et des tableaux imbriqués pour sélectionner des chaînes de langages différents afin de remplir les champs de texte, consultez la page des exemples Flash à l'adresse www.adobe.com/go/learn_fl_samples_fr. Téléchargez le fichier zip Exemples et naviguez jusqu'au dossier ActionScript2.0/XML_LanguagePicker afin d'accéder à cet exemple.

Pour un exemple de fichier source, xmlmenu.fla, qui décrit comment créer un menu dynamique avec des données XML, consultez la page des exemples Flash à l'adresse www.adobe.com/go/learn_fl_samples_fr. Téléchargez le fichier zip Exemples et naviguez jusqu'au dossier ActionScript2.0/XML_Menu afin d'accéder à cet exemple. Cet exemple appelle le constructeur ActionScript `XmlMenu()` et lui transmet deux paramètres : le chemin au fichier menu XML et une référence au scénario en cours. Le reste de la fonctionnalité réside dans un fichier de classe personnalisé, XmlMenu.as.

Utilisation de la classe XMLSocket

ActionScript fournit une classe XMLSocket intégrée qui vous permet d'établir une connexion continue avec un serveur. Une connexion socket permet au serveur de publier l'information sur le client ou de la *pousser* dès qu'elle est disponible. En l'absence de connexion continue, le serveur devra attendre une requête HTTP. Cette connexion ouverte supprime les périodes d'attente et est souvent utilisée dans des applications en temps réel telles que les dialogues en ligne. Les données sont envoyées sur la connexion socket sous forme d'une chaîne et doivent être au format XML. Vous pouvez utiliser la classe XML pour structurer les données.

Pour créer une connexion socket, vous devez créer une application côté serveur qui attendra la requête de connexion socket et enverra une réponse au fichier SWF. Ce type d'application côté serveur peut être écrit dans un langage tel que Java.

REMARQUE

La classe XMLSocket ne peut pas automatiquement tunneler à travers les pare-feux parce que, au contaire du Real-Time Messaging Protocol (RTMP), le XMLSocket n'a pas de capacité de tunneling HTTP. Si vous devez utiliser le tunneling HTTP, envisagez l'utilisation de Flash Remoting ou Flash Media Server (qui prend en charge RTMP).

Vous pouvez utiliser les méthodes `connect()` et `send()` de la classe XMLSocket pour transférer un objet XML vers et à partir d'un serveur sur une connexion socket. La méthode `connect()` établit une connexion socket avec le port d'un serveur Web. La méthode `send()` transmet un objet XML au serveur spécifié dans la connexion socket.

Lorsque vous invoquez la méthode `connect()`, Flash Player ouvre une connexion TCP/IP avec le serveur et garde cette connexion ouverte jusqu'à ce que l'un des événements suivants se produise :

- La méthode `close()` de la classe XMLSocket est appelée.
- Il n'existe plus aucune référence à l'objet XMLSocket.
- Flash Player se ferme.
- La connexion est interrompue (le modem est déconnecté, par exemple).

L'exemple suivant crée une connexion socket XML et envoie les données de l'objet XML `myXML`. Pour comprendre le script, vous pourrez vous aider des commentaires (indiqués par les caractères `//`) :

```
// Créer l'objet XMLSocket
var theSocket:XMLSocket = new XMLSocket();
// Se connecter à un site sur un port non utilisé au-dessus de 1024 en
// utilisant la methode connect().
// Entrer localhost ou 127.0.0.1 pour des essais locaux.
// Pour un serveur en direct, entrez votre nom de domaine www.yourdomain.com
theSocket.connect("localhost", 12345);
// affiche le texte relatif à la connexion
theSocket.onConnect = function(myStatus) {
  if (myStatus) {
    conn_txt.text = "connection successful";
  } else {
    conn_txt.text = "no connection made";
  }
};
//données à envoyer
function sendData() {
  var myXML:XML = new XML();
  var mySend = myXML.createElement("thenode");
  mySend.attributes.myData = "someData";
  myXML.appendChild(mySend);
  theSocket.send(myXML);
}
// le bouton envoie des données
sendButton.onRelease = function() {
  sendData();
};
// suit les données renvoyées par la connexion socket
theSocket.onData = function(msg:String):Void {
  trace(msg);
};
```