

Flash Article

Filtering XML data in Flash applications using ECMAScript for XML の日本語訳

本ドキュメントは、Adobe 社のサイトにある“Filtering XML data in Flash applications using ECMAScript for XML “をヒム・カンパニー 永井勝則が自主的に日本語に訳したものです。

<http://www.himco.jp>

knagai@himco.jp

(2007/9/17)

本ドキュメントの原文は

http://www.adobe.com/devnet/flash/articles/filtering_data_e4x.html

です。

Flash Article

ECMAScript for XML を使った Flash アプリケーションでの XML データのフィルタリング

Andrew Muller

webqem.com

ActionScript 3.0 と Adobe Flash Player 9 に導入された ECMAScript for XML (E4X) は、Adobe Flash アプリケーションに XML からデータを取得する新しい方法を提供します。この記事では、E4X を使って、アプリケーションを動作させるのに必要な情報だけを処理し、リアルタイムでアプリケーション内の XML オブジェクトからデータを取得する方法を見ていきます。この戦略は、不要な負荷やデータのロードを減らすことで Flash アプリケーションのパフォーマンスを向上させます。

E4X: Flash Player への待望の追加

Flash Player 9 には、プレイヤーのパフォーマンスを向上させる、ActionScript 仮想マシン 2 (AVM2) と呼ばれる新しい仮想マシンが含まれます。加えて、ActionScript 3.0 の ActionScript 言語の改良は、Flash アプリケーションを作成する Flash 開発者に新しい機能を提供します。

Note: ActionScript 仮想マシン 2 がサポートする命令や関連するデータ構造、ファイル形式に関する詳細は [AVM2 overview \(PDF, 401k\)](#) をご覧ください。

改良点の 1 つは ECMAScript for XML (E4X) の採用です。これは、JavaScript として広く実装されている標準的なスクリプティング言語である ECMAScript の一部です。ActionScript は ECMAScript に準拠し、Adobe は ECMAScript 標準への貢献者の 1 つです。

E4X は、XML をネイティブなデータタイプとして追加する、ECMAScript への機能拡張で、ドット (.) や属性識別子 (@) を追加してデータを検索しフィルタリングします。ActionScript 開発者にとってこれは、XML はいまや ActionScript 3.0 ではネイティブなデータタイプになったということです。XML は今やデータを管理しデータにアクセスするクラスのセットを備えたのです。これによって XML 構造内のデータにこれまでより簡単にアクセスすることができます。

E4X は、ActionScript 3.0 アプリケーションが使用する XML データへのアクセスをこれまでより高速に、論理的に行える機能を提供します。実行時にこれを使用することで、XML オブジェクトから直接データを取得できるので、特別な解析を記述する必要がなくなります。これまでの XML データは、アプリケーションでデータを使用するとき、XML パケットから、オブジェクトの配列などの構造化されたネイティブなデータタイプに変換する必要がありました。

この処理が今や直観的で分かりやすくなり、ActionScript 3.0 で XML データを扱いやすくなったのです。

Flash ヘルプ引用: E4X による XML 処理

ECMAScript for XML 仕様には、XML データを処理するためのクラス群や機能が定義されています。これらのクラスは総称して E4X と呼ばれています。ActionScript 3.0 は、E4X 準拠のクラスとして、XML、XMLList、 QName、および Namespace を備えています。

E4X の演算子

オブジェクトの配列内のデータへの角かっこ ([]) かドット記法を使った基本的なアクセスに習熟されている方なら、E4X を扱うときに使用するシンタックスはすぐに理解できます。E4X の演算の実行にはこれらの演算子に加えて属性識別子 (@) が使用されます。

以下のサンプルでは、文字のストリングは XML オブジェクトしてデータタイプ(型づけ)されています。

```
var myXML:XML =
<employees>
  <employee id="1">
    <fname>Frank</fname>
    <lname>Bacon</lname>

    <email>fbacon@company.com</email>
  </employee>
  <employee id="2">
    <fname>Chris</fname>
    <lname>Wren</lname>
    <email>cwren@company.com</email>
  </employee>
</employees>
```

以下のコードサンプルでは、前述した E4X 演算子を、XML オブジェクト内に含まれるデータにアクセスするために使用しています。

```
trace(myXML.employee[0].fname);
//出力: Frank
trace(myXML.employee.@id=="2").email);
```

```
//出力: cwren@company.com
trace(myXML.employee.(lname=="Bacon").email);
//出力: fbacon@company.com
```

1つめの trace ステートメントは、XML の1つめのノード内の employee のファーストネームを表示します。
2つめの trace ステートメントは、id 属性のデータでフィルタリングし、id 値が“2”である場合を検出します。
3つめの trace ステートメントは、データをフィルタリングして、値が“Bacon”であるノードを検出します。

Note: このフィルタは大文字小文字を区別します。ストリング“bacon”と比べると、同じ出力は得られません。

for ループを使って XML ファイルの全ノードを繰り返し処理できます。下記のコードサンプルでは、両方のノードのデータにアクセスするまでループを繰り返して、各 employee のファーストネーム、ラストネーム、メールアドレスを出力します。

```
for each ( var property:XML in myXML.employee ) {
    trace(property.fname + " " + property.lname);
    trace(property.email);
    trace("-----");
}
```

この記事ではこの後、XML データと E4X を使って、複数言語でビデオのキャプションを表示するアプリケーションの作成方法を見ていきます。

必要な環境

この記事にそって進めるには、以下のソフトウェアとファイルが必要です。

Flash CS3 Professional

サンプルファイル:

e4x_as3_sample.zip (ZIP、1.9MB)

必要な予備知識

この記事では、ActionScript 3.0 の中級程度の知識と、これまでに XML ファイルを扱った経験のあることを前提としています。

アプリケーション内の E4X

この記事では、XML データと E4X を使って、ビデオを再生するとき複数言語のキャプションを表示するアプリケーションの構築について述べていきます。これによって E4X の使用方法が理解できるようになります。単体の XML パッケージが、必要なすべてのキャプションデータを含むことになるアプリケーションにロードされます。みなさんは、そのデータをすべて複雑なデータ構造に解析するのではなく、XML オブジェクトから E4X でデータを抽出し、アプリケーション内の別の場所でそのデータを使用する方法を目にします(図1参照)。



私はちょうど1996年にMacromediaを結合した、

Japanese ▼

図1: 選択された言語にもとづいてキャプションをダイナミックに表示するビデオアプリケーション

アプリケーションのレイアウト

このビデオアプリケーションのレイアウトは単純です。まずアプリケーション内で表示する FLV ビデオを再生する FLVPlayback コンポーネントをステージの真ん中に置きます。その下にキャプションを表示するダイナミックテキストフィールドを置きます。そしてその下に、ユーザーが表示したい言語のキャプション文字を選択できる Combobox コンポーネントを置きます。

XML パッケージの1つめのノードはさまざまな言語に関する情報を含みます。この情報は残りのパッケージが保持します。言語データの情報はアプリケーションが ComboBox に値を入れるために使用し、ビデオのキャプションで使用できる言語のオプションを表示します。

この記事のために、キャプションのテキストとタイミングは、Flash CS3 Professional のドキュメンテーションから“借りて”います。FLVPlayback コンポーネントを使うときは、FLVPlaybackCaptioning コンポーネントを使ってビデオにキャプションを追加することもできます。FLVPlaybackCaptioning コンポーネントはタイミングとキャプションに Timed Text (TT) XML ファイルを使用します。これはキャプション表示に使用されるオープンスタンダードな

形式で、基本的には DTD (Data Type Definition、文書型定義) です。

複数言語でのしゃべり

複数言語でのキャプション表示は FLVPlaybackCaptioning コンポーネントを使って実装できますが、この記事ではサンプルとして複数言語のキャプション表示の概念を示し、E4X を使うと複雑な構造をした XML データへのアクセスが非常に簡単になるということをお見せします。

以下の XML コードはこのアプリケーションで使用する XML パケットの一部です。

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<movieCaptions>
  <languages>
    <language name="English" code="eng"/>
    <language name="French" code="fre"/>
    <language name="German" code="deu"/>
    <language name="Italian" code="ita"/>
    <language name="Japanese" code="jpn"/>
    <language name="Spanish" code="spa"/>
  </languages>
  <caption cuepoint="0.500">
    <eng>I had just joined Macromedia in 1996,</eng>
    <deu>Ich hatte gerade Macromedia 1996 verbunden,</deu>
    <fre>J'avais juste joint Macromedia en 1996,</fre>
    <ita>Avevo
unito appena Macromedia in 1996,</ita>
    <jpn>私はちょうど 1996 年に Macromedia を結合した、</jpn>
    <spa>Acababa de ensamblar Macromedia en 1996,</spa>
  </caption>
</movieCaptions>
```

Note: このサンプルで使用している翻訳文はオンラインの翻訳ソフトウェアを使って作成したものです。この文章がいくぶんでも正確なものであればよいのですが。

オリジナルの XML データは1つの<languages>ノードと複数の<caption>ノードで構成されます。XML を分析すると分かるように、<languages>ノードの項目は name と code 属性を含んでいます。<caption>ノードでは、

各国語のキャプションが、各言語の code 属性に対応するタグで囲まれています。

訳者注: 実際の translated_captions.xml の中身

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<movieCaptions>
  <languages>
    <language name="English" code="eng"/>
    <language name="French" code="fre"/>
    <language name="German" code="deu"/>
    <language name="Italian" code="ita"/>
    <language name="Japanese" code="jpn"/>
    <language name="Spanish" code="spa"/>
  </languages>
  <caption cuepoint="0.500">
    <eng>I had just joined Macromedia in 1996,</eng>
    <deu>Ich hatte gerade Macromedia 1996 verbunden,</deu>
    <fre>J'avais juste joint Macromedia en 1996,</fre>
    <ita>Avevo unito appena Macromedia in 1996,</ita>
    <jpn>私はちょうど 1996 年に Macromedia を結合した、</jpn>
    <spa>Acababa de ensamblar Macromedia en 1996,</spa>
  </caption>
  <caption cuepoint="3.350">
    <eng>and we were trying to figure out what to do about the internet.</eng>
    <deu>und wir versuchten, was heraus darzustellen, über das Internet zu tun.</deu>
    <fre>et nous essayions de figurer dehors quoi faire au sujet de l'Internet.</fre>
    <ita>e stavamo provando a calcolare fuori che cosa fare circa il Internet.</ita>
    <jpn>そして私達はインターネットについてすばいのか何を把握することを試みていた。</jpn>
    <spa>e intentábamos calcular hacia fuera qué hacer sobre el Internet.</spa>
  </caption>
  <caption cuepoint="6.420">
    <eng>And the company was in dire straights at the time.</eng>
    <deu>Und die Firma war in den entsetzlichen straights zu der Zeit.</deu>
    <fre>Et la compagnie était dans de grands straights alors.</fre>
    <ita>E l'azienda era allora negli straights dire. </ita>
    <jpn>そして会社はものすごい straights にその時にあった。</jpn>
    <spa>Y la compañía estaba en straights calamitosos en ese entonces.</spa>
```

</caption>

<caption cuepoint="9.550">

<eng>We were a CD-ROM authoring company,</eng>

<deu>Wir waren eine CD-ROM Firma schreibend,</deu>

<fre>Nous étions un CD-ROM écrivant la compagnie,</fre>

<ita>Eravamo un CD-ROM che creiamo l'azienda,</ita>

<jpn>会社を書いている私達は CD-ROM だった</jpn>

<spa>Éramos un CD-ROM que eran autor de la compañía,</spa>

</caption>

<caption cuepoint="11.000">

<eng>and the CD-ROM business was going away.</eng>

<deu>und das CD-ROM Geschäft ging weg.</deu>

<fre>et les affaires de CD-ROM partaient.</fre>

<ita>ed il commercio del CD-ROM stava andando via.</ita>

<jpn>そして CD-ROM ビジネスは立ち去っていた。</jpn>

<spa>y el negocio del CD-ROM salía.</spa>

</caption>

<caption cuepoint="13.200">

<eng>One of the technologies I remember seeing was Flash.</eng>

<deu>Eine an der Technologien, die ich mich erinnere, Flash daß sehend war.</deu>

<fre>Une des technologies que je me rappelle que voyant était le Flash.</fre>

<ita>Una delle tecnologie mi ricordo di che vedente era il Flash.</ita>

<jpn>フラッシュは見てことをだった私が覚えている技術の 1 つ。</jpn>

<spa>Una de las tecnologías que recuerdo que que consideraba era el Flash.</spa>

</caption>

<caption cuepoint="16.050">

<eng>At the time, it was called FutureSplash.</eng>

<deu>Zu der Zeit als, es FutureSplash genannt wurde.</deu>

<fre>Lorsque, ce s'est appelé FutureSplash.</fre>

<ita>Quando, è stato denominato FutureSplash.</ita>

<jpn>、それが FutureSplash と呼ばれた時。</jpn>

<spa>Cuando, fue llamado FutureSplash.</spa>

</caption>

<caption cuepoint="17.50">

<eng>So this is where Flash got its start.</eng>

<deu>So ist dieses, wohin Flash seinen Anfang erhielt.</deu>

<fre>Ainsi c'est où le Flash a obtenu son début.</fre>

<ita>Così questo è dove il Flash ha ottenuto il relativo inizio.</ita>

<jpn>そうこれはフラッシュが開始を得たところである。</jpn>

<spa>Aquí es tan adonde el Flash consiguió su comienzo.</spa>

</caption>

<caption cuepoint="19.100">

<eng>This is smart sketch running on the EU-pin computer,</eng>

<deu>Dieses ist die intelligente Skizze, die auf den Eu-Stift Computer läuft,</deu>

<fre>C'est croquis futé fonctionnant sur l'Eu-goupille ordinateur,</fre>

<ita>Ciò è abbozzo astuto che funziona sul Eu-perno calcolatore,</ita>

<jpn>これは EU ピンコンピュータで動くスマートなスケッチである</jpn>

<spa>Éste es bosquejo elegante que funciona en el EU-perno computadora,</spa>

</caption>

<caption cuepoint="23.520">

<eng>which was the first product that FutureWave did.</eng>

<deu>welches das erste Produkt war, das FutureWave.</deu>

<fre>ce qui était le premier produit que FutureWave.</fre>

<ita>quale era il primo prodotto che FutureWave.</ita>

<jpn>FutureWave が最初のプロダクトはだったかどれ。</jpn>

<spa>cuál era el primer producto que lo hizo FutureWave.</spa>

</caption>

<caption cuepoint="25.520">

<eng>So our vision for this product was to</eng>

<deu>So war unser Anblick für dieses Produkt zu</deu>

<fre>Ainsi notre vision pour ce produit était à</fre>

<ita>Così la nostra visione per questo prodotto era a</ita>

<jpn>このプロダクトのための私達の視野はにそうあった</jpn>

<spa>Nuestra visión para este producto estaba tan a</spa>

</caption>

<caption cuepoint="27.520">

<eng>make drawing on the computer</eng>

<deu>das Zeichnen auf dem Computer bilden</deu>

<fre>faire le dessin sur l'ordinateur</fre>

<ita>fare il disegno sul calcolatore</ita>

<jpn>コンピュータでデッサンを作りなさい</jpn>

<spa>hacer el dibujo en la computadora</spa>

```
</caption>
<caption cuepoint="29.020">
  <eng>as easy as drawing on paper.</eng>
  <deu>so einfach wie, zeichnend auf Papier.</deu>
  <fre>aussi facile que dessinant sur le papier.</fre>
  <ita>facile quanto disegnando sulla carta.</ita>
  <jpn>ペーパーのデッサン容易。</jpn>
  <spa>tan fácil como dibujando en el papel.</spa>
</caption>
</movieCaptions>
```

アプリケーションのコードの記述

このアプリケーションは XML パケットを取得すると、XMLList クラスのインスタンスを使用して言語とキャプションデータを保持します。これによって E4X 演算子を使って XML データを処理することが可能になります。まずいかに示すように、XMLList クラスの2つの変数を宣言し、それぞれ `captions` と `languages` という名前をつけます。

```
var captions:XMLList = new XMLList();
var languages:XMLList = new XMLList();
```

次に XML を取得する必要があります。これは URLLoader クラスのインスタンスを作成しその load メソッドを呼び出すことで行えます。load メソッドは、XML パケットを取得するために URLRequest クラスのインスタンスを使用します。

```
var captionsXMLLoader:URLLoader = new URLLoader();
captionsXMLLoader.load(new URLRequest("assets/translated_captions.xml"));
captionsXMLLoader.addEventListener(Event.COMPLETE,captionsXMLLoadedHandler);
```

上記サンプルのコードではローカルパスから XML をロードしていますが、ネットワーク上のファイルへの URL を使って XML をロードできない理由はありません。

アプリケーションでは、パケットのロードが完了するとその XML データを使用する必要があります。addEventListener メソッドは、パケットのロードの完了した時の関数呼び出しを行わせます。リスナーはサンプルの最後の行で追加しています。

次に関数を記述します。下記のコードサンプルでは captionsXMLLoadedHandler という名前の新しい関数を作成しています。この関数ではまず、新しい XML オブジェクトを取得したデータで作成して、captionsXML という名前のローカル変数に代入しています。取得したデータはここでは XML 型なので、XML クラスに含まれるメソッドを使ってデータを取得することができます。次の行では、XMLList の captions に、このローカル変数のチャイルドノードを使って値を入れています。

前述したように、取得した XML パケットの最初のノードはキャプションに使用する言語のリストです。そのパスは XMLList の languages のデータの取得にも使用しています。

```
function captionsXMLLoadedHandler(eventObj:Event):void {
    var captionsXML = new XML(eventObj.currentTarget.data);
    captions = captionsXML.children();
    languages = captionsXML.children()[0].children();
    setCuePoints();
    setLanguages();
}
```

Flash ヘルプ引用

XMLList.children()

各 XML オブジェクトの children() メソッドを呼び出し、その結果を含む XMLList オブジェクトを返します。
戻り値 XMLList – XML オブジェクト内の子 (複数) の XMLList オブジェクトです。

XMLList の captions と languages に保持したデータは、この後アプリケーションの初期値の設定に使用します。captions は FLVPlayback コンポーネントのキューポイントの設定に使用して、表示されるキャプションを変更するイベントを発生させます。languages は、特定の言語のリストをユーザーに提供する Combobox コンポーネントにその値を入れます。これらのアクションは、上記に詳しく説明した captionsXMLLoadedHandler 関数内から適切な関数を呼び出すことで処理されます。その最初の関数は setCuePoints 関数です。

以下に setCuePoints 関数のコードを示します。

```
function setCuePoints():void {
    var i:int = 0;
    for each (var prop:XML in captions) {
        var captionData:XML = prop;

        my_FLVPlayback.addASCuePoint({name:i.toString(),
```

```
        time:Number(captionData.@cuepoint));  
        i++;  
    }  
}
```

setCuePoints 関数は for ループを使って captions の繰り返し処理を行い、その中で FLVPLayback コンポーネントの addASCuePoint メソッドを使って各ノードの cuepoint 属性を追加します。i はキューポイントの name 属性の値として設定されます。同時に E4X が XMLList 内の現在の項目の cuePoint 属性を抽出するために使用されます。

Flash ヘルプ引用

```
my_FLVplybk.addASCuePoint(cuePoint:Object)
```

訳者注:

i と captionData.@cuepoint を for ループ内で trace() で出力すると、

```
i:1
```

```
0.500
```

```
i:2
```

```
3.350
```

```
i:3
```

```
6.420
```

```
i:4
```

```
9.550
```

```
i:5
```

```
11.000
```

...という結果が得られます。addASCuePoint メソッドではこの値をキューポイントオブジェクトとして追加していません。

次に新しい DataProvider オブジェクトを作成し、setLanguages 関数を詳しくみていきましょう。

```
import fl.data.DataProvider;  
var dp:DataProvider = new DataProvider();  
language.dataProvider = dp;  
var selectedLang:Object = new Object();  
  
function setLanguages():void {
```

```
var i:int = 0; //この i には何の意味があるのか？
for each(var prop:XML in languages) {
var languageData:XML = prop;
dp.addItem({
label:languageData.@name, data:languageData.@code });
i++; //この i には何の意味があるのか？
}
selectedLang = language.dataProvider.getItemAt(0);
}
```

setLanguages 関数は、Combobox コンポーネントの dataProvider に構造化されたデータを追加するので、追加的なコードに依存します。dataProvider (dp) は、上記コードの初めの2行で作成され、ComboBox コンポーネント (language) に割り当てられます。for ループ内のコードでは E4X のフィルタリングを使って、XMLList の languages から抽出した name と code 属性を、ループが繰り返されるたびに dataProvider の addItem メソッドを使って dataProvider に追加しています。

selectedLang という名前の変数には、languages の dataProvider の最初の項目が代入されます。この変数は、ComboBox コンポーネントの選択項目が変更されたとき、表示するキャプションを決めるために使用されます。選択が行われる前に ComboBox の初期値のデフォルト値を設定することは、アプリケーションが動作しているのに、ユーザーが ComboBox から言語を選択していない場合に備えるためにも重要です。

ユーザーインターフェイスに値を入れる

このセクションではアプリケーションのステージに視覚的な要素を追加する方法を示します。まずステージに FLVPlayback コンポーネントをドラッグし消去します。これによって FLVPlayback コンポーネントがライブラリに追加されます。ステージから消去したのは、この記事では ActionScript 3.0 を使ってコンポーネント表示をすべて制御するからです。

次にステージにダイナミックテキストフィールドを追加します。テキストフィールドには、選択したフォントサイズで2行のテキストを表示できるだけの十分な幅と高さを設定します。またテキストが折り返して表示されるように[複数行]オプションを選択します。インスタンス名は captionText にします。

最後にステージに ComboBox コンポーネントをドラッグします。ダイナミックテキストフィールドの下に移動しインスタンス名を language にします。整列ツールを使って両方の要素を真ん中に配置します (図2参照)

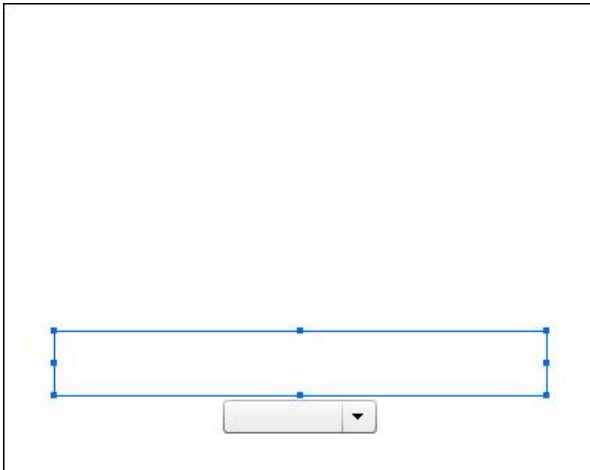


図2: ステージでテキストフィールドと ComboBox コンポーネントを整列させます。

次に FLVPlayback コンポーネントを制御するコードを追加します。

```
import fl.video.*;
var my_FLVPlayback = new FLVPlayback();
my_FLVPlayback.x = 115;
my_FLVPlayback.y = 50;
addChild(my_FLVPlayback);
my_FLVPlayback.source = "assets/caption_video.flv";
my_FLVPlayback.addEventListener(MetadataEvent.CUE_POINT, cp_listener);
```

このコードでは新しいFLVPlaybackコンポーネントをステージに追加し、テキストフィールドの上に配置しています。また FLVPlayback コンポーネントで表示する FLV ファイルをロードしています。それからキューポイントイベント用のイベントリスナーを追加し、cp_listener 関数をこのイベント発生時に呼び出すように設定しています。

以下が cp_listener 関数のコードです。変数 currentCuePoint は現在のキューポイントの名前を保持するために設定します。この変数の値はアプリケーション内のいくつかの関数が使用します。

```
var currentCuePoint:Number = 0;
function cp_listener(eventObject:MetadataEvent):void {
    currentCuePoint = eventObject.info.name;
    setCaption(currentCuePoint);
}
```

FLVPlayback コンポーネントのキューポイントイベントによって呼び出される cp_listener 関数では、

currentCuePoint の値を現在のキューポイント名に設定します。また setCaption 関数を、currentCuePoint の値を渡して呼び出します。

setCaption 関数では、角かっこ記法(配列アクセス演算子)と E4X フィルタを使用して、テキストフィールドのテキストの値を現在のキューポイントの言語に設定しています。

```
function setCaption(cp:Number):void {  
    captionText.text = captions[cp][selectedLang.data];  
}
```

最後の関数、setLanguage は ComboBox で選択が行われたとき ComboBox が呼び出します。setLanguage 関数は、以下に示すように、setCaption 関数を呼び出して現在のキャプションの言語を設定します。

```
function setLanguage(evt:Object):void {  
    selectedLang=language.selectedItem;  
    setCaption(currentCuePoint);  
}
```

このサンプルコードでは変数宣言と import ステートメントは、各部を論理的に示すために移動させています。

慣習として import ステートメントは ActionScript コードの最初に置き、変数もコードの初めで定義します。コードの最初に import ステートメントを置くと、その関数が呼び出される時、インポートされた項目はアプリケーションで確実に使用できるようになります。

次はどこへいく

E4X を使って実行時に積極的にデータをフィルタリングすると、アプリケーションのスタート時に不要なデータの変換を減らすことができます。これにはパフォーマンスの向上など多くのメリットがあります。各関数で詳しく見たように、このアプリケーションを構築するのに必要なコードはさほど多く、必要なデータだけを取得しているので非常にすぐれた方法になっています。通常の XML サンプルには RSS フィードの解析が含まれますが、わたしは、Flash Video で複数言語のキャプションを表示するサンプルで紹介した概念が、みなさんの今後のプロジェクトによりアイデアを与えることを願います。

覚えておいていただきたいのは、FLVPlaybackCaptioning コンポーネントは FLVPlayback コンポーネントの完璧なパートナーだということです。Flash オーサリング環境のオーサリングモードでこれらのコンポーネントを設定

する場合には、このサンプルで示したコードを記述せずにビデオのキャプションを作成できます。多くの操作と同様、同じような目的を達成できる方法はいくつもあります。みなさんは、ActionScript 3.0 を使って FLVPlayback コンポーネントを制御することで、ActionScript 3.0 アプリケーションに統合できる XML データを高速に取得する E4X を使った新しいテクニックを学んだのです。