Identifying an XML Document's Type

By David Adams

Technical Note 07-02

Abstract

このテクニカルノートは、4th DimensionのDOMコマンドを使用して、XMLドキュメントのタイプを確認する方法を説明します。サンプルデータベースには、このテクニカルノートで説明するコードが含まれています。

Overview

XMLはエンコーディングの標準システムやデータの構造を定義します。今日、さまざまなアプリケーションで何百万もの異なるタイプのXMLドキュメントが使用されていることでしょう。これらのドキュメントはXML標準にしたがっているので、4th DimensionのDOMパーサのような標準XMLパーサであれば、どんなXMLファイルでも読むことができます。しかし、ドキュメントを解析することと、その内容を読み取ることとの間には大きな違いがあります。例えばWSDL (Web Service Description Language)ドキュメント、SVG (Scalable Vector Graphics)イメージ、そしてXHTML Webページはそれぞれがドキュメントタイプです。WSDLファイルを編集するために設計されたプログラムが、SVGイメージ描画のルールも解釈できるなどということはほとんど起こりません。同様に、ドキュメントのフォーマットも時間を経て変化したり、関連するいくつかのバージョンができたりします。例えば、現時点で4つのXHTMLのバージョンがあります。そのため、特定のXMLフォーマットを解釈するプログラムは、自身が解析可能なXMLドキュメントやバージョンと、そうでないものとを見分ける方法が必要です。

このテクニカルノートでは、4DのDOMコマンドを使用して、ドキュメントのタイプを見分ける方法について説明します。サンプルデータベースには、このテクニカルノートで説明されるコードが含まれています。

How XML Documents Are Distinguished

XMLはプラットフォームに依存せず、将来にわたって使用可能なように設計されています。そのため、ファイルタイプや拡張子、MIMEタイプなどは、ファイルタイプを定義するのに十分ではありません。これらに加えて、プログラムやオープンスタンダードでは、いくつかのテクニックを使用します:

ドキュメントの先頭に記述するドキュメントタイプ宣言

特定のルート要素名

特定の名前空間宣言

例として、以下のような単純なXHTMLを見てみましょう:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
        "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
 <head>
  <title>Hello world!</title>
 </head>
 <body>
  <p>This is an XHTML document.</p>
 </body>
</html>
```

XHTMLドキュメントはルート要素として<html>を持たなければなりません。また名前空間としてhttp://www.w3.org/1999/xhtmlを使用します:

```
<html xmlns="http://www.w3.org/1999/xhtml">
```

ドキュメントの先頭で指定されたDTDは、XHTMLの特定のバージョンをプログラムに伝えます。

Tip *独自のXMLフォーマットを作成する場合、ユニークなルート要素名と、名前空間宣言を含めることにより、ドキュメントを確認する際に役立ちます。*

Reading the Identifying Elements with 4th Dimension

XMLドキュメントのタイプを確認する正しいルールは、ドキュメントタイプ自身により異なります。ここではXHTMLを例にとり、ドキュメントタイプを確認する典型的なルールについて見ていくことにします。確認に使用する要素は、以下で太字で示した部分です:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
      "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
```

以下の表は、確認に使用する情報をXMLから取り出す方法を示します:

-//W3C//DTD XHTML 1.1//EN	**DOM Get XML information** (xmlref;PUBLIC ID)
http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd	**DOM Get XML information** (xmlref;SYSTEM ID)
html	**DOM GET XML ELEMENT NAME**を使用して、ルート要素の名前を取得する。
http://www.w3.org/1999/xhtml	**DOM GET XML ATTRIBUTE BY NAME.**を使用して、ルート要素のxmlns属性値を取得する。

Note *XMLパス、要素名、そして属性名は大文字小文字を区別します。大文字小文字を区別する文字列の比較については、4D Technical Note 05-41,* Case-Sensitive Operations in 4th Dimension *を参照してください。*

Demo_ReadXHTMLDocument

ここで説明するコードは、XHTMLドキュメントを受け入れ、バージョンを識別する方法を示します。*DOM_GetRootElementReference*は、ルート要素の検出を簡略化するように実装されています。

```
C_STRING(16;$document_xmlref)
$document_xmlref:=DOM Parse XML source("")  ` Balance with a call to DOM CLOSE XML.
If (OK=1)
    ` Assume the following DOCTYPE in the example below.
    ` <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
    ` "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">

  C_TEXT($publicID_text)
  C_TEXT($systemID_text)

    ` -//W3C//DTD XHTML 1.1//EN
  $publicID_text:=DOM Get XML information($document_xmlref;PUBLIC ID )
    ` http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd
  $systemID_text:=DOM Get XML information($document_xmlref;SYSTEM ID )

  C_TEXT($documentVersionDescription_text)
  $documentVersionDescription_text:=""

  C_BOOLEAN($documentIsXhtml_b)
  $documentIsXhtml_b:=False  ` Set to True below if all tests are satisfied.

    ` A document was parsed, but is it an XHTML document?
  C_STRING(16;$root_xmlref)
  $root_xmlref:=DOM_GetRootElementReference ($document_xmlref)

  If ($document_xmlref#"0000000000000000")  ` The reference looks valid

    C_TEXT($name_text)
    DOM GET XML ELEMENT NAME($root_xmlref;$name_text)

    If (CS_AlphasAreEqual ($name_text;"html"))
        ` The root element is <html>.
        ` Note: Element names are case-sensitive in XML.
      DOM_StartCustomErrorHandling

      C_TEXT($namespace_text)
      $namespace_text:=""
      DOM GET XML ATTRIBUTE BY NAME($root_xmlref;"xmlns";$namespace_text)
      DOM_StopCustomErrorHandling


      If ($namespace_text="http://www.w3.org/1999/xhtml")
          ` The root element includes the required namespace declaration.
        $documentIsXhtml_b:=True
          ` Now try to distinguish the XHTML type. There are only
          ` four defined, at the moment.
          ` Note: The statements below use wildcards to ignore the language code.
          ` For example, a source document might include the following public ID:
          ` -//W3C//DTD XHTML 1.0 Strict//EN
          ` For matching purposes, we're only interested in this part of the string:
          ` -//W3C//DTD XHTML 1.0 Strict//

        Case of
          ¥ (($publicID_text="-//W3C//DTD XHTML 1.0 Strict//@") &
                ($systemID_text="http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd"))
            $documentVersionDescription_text:="XHTML 1.0 Strict."

          ¥ (($publicID_text="-//W3C//DTD XHTML 1.0 Transitional//@") &
                  ($systemID_text="http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd"))
            $documentVersionDescription_text:="XHTML 1.0 Traditional."

          ¥ (($publicID_text="-//W3C//DTD XHTML 1.0 Frameset//@") &
                  ($systemID_text="http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd"))
            $documentVersionDescription_text:="XHTML 1.0 Frameset."

          ¥ (($publicID_text="-//W3C//DTD XHTML 1.1//@") &
                  ($systemID_text="http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd"))
            $documentVersionDescription_text:="XHTML 1.1."
```

```
            Else
               $documentVersionDescription_text:="Unrecognized or unsupported XHTML type."

         End case
      End if
   End if
End if

If ($documentIsXhtml_b)
   ALERT("The document appears to be an XHTML document."+
   Char(Carriage return )+Char(Carriage return )+$documentVersionDescription_text)
Else
   BEEP
   ALERT("Warning: "+Char(Carriage return )+Char(Carriage return )+
         "The document does not appear to be XHTML.")
End if

DOM CLOSE XML($document_xmlref)` Balances earlier call to DOM Parse XML source.
End if
```

以上のメソッドは長く複雑なように見えますが、単にテストを繰り返しているにすぎません:

1) ドキュメントは妥当なXMLか? そうであれば、
2) ルート要素の名前は`html`か? そうであれば、
3) XMLルート要素に`xmlns`属性が含まれ、`http://www.w3.org/1999/xhtml`という値を持っているか? そうであれば、
4) ドキュメントはXHTMLである。systemとpublic IDを、XHTML標準で指定された定数と比較して、バージョンを特定する。

ドキュメントを特定するルールは、ドキュメントのタイプにより異なります。XHTMLドキュメントはこのノートに示した方法で特定することができますが、他のXMLドキュメントタイプを取得する際には、ここで示した方法は必要なステップを示した例題として参考にしてください。

The Routines

以下、サンプルデータベースに含まれる他のルーチンを、参考のために記載します。

DOM_GetRootElementReference

```
C_STRING(16;$0;$rootNode_xmlref)
C_STRING(16;$1;$noderef)

$noderef:=$1
$rootNode_xmlref:="0000000000000000"    ` Default to a null reference.

   ` Store existing error/error handling state.
If (Undefined(Error))
   Error:=0
End if

C_LONGINT($previousValueOfErrorVariable_l)
$previousValueOfErrorVariable_l:=Error

C_STRING(31;$previousErrorMethodName_s)
$previousErrorMethodName_s:=Method called on error

ON ERR CALL("DOM_ErrorTrappingRoutine")

C_TEXT($currentNodeName_text)
$currentNodeName_text:=""

   ` Default to starting element name.
DOM GET XML ELEMENT NAME($noderef;$currentNodeName_text)

While (OK=1)
   $rootNode_xmlref:=$noderef

   Case of
      ¥ ($currentNodeName_text="")
         OK:=0` We're done scanning.

      ¥ ($currentNodeName_text="#document")
            ` An artificial node above the tree. We're not treating this as a valid ancestor.
         $noderef:=DOM Get first child XML element($noderef;$currentNodeName_text)
            ` We're above the root, try moving down one.
         If (OK=1)    ` The first child of#document is the root.
            $rootNode_xmlref:=$noderef` This is the root reference.
         End if
         OK:=0    ` Stop the loop before it moves back up to #document.
      Else
            ` Try to get another parent.
         $noderef:=DOM Get parent XML element($noderef;$currentNodeName_text)
   End case
End while

   ` Restore previous error/error handling state.
Error:=$previousValueOfErrorVariable_l    ` Restore original error value.
ON ERR CALL($previousErrorMethodName_s)    ` Restore original error handler.

$0:=$rootNode_xmlref
```

DOM_StartCustomErrorHandling

*DOM_StartCustomErrorHandling*ルーチンは、実行時のエラー情報を記憶し、エラーハンドラをインストールします。

```
If (Undefined(Error))
   Error:=0
End if

C_STRING(80;DOM_PreviousErrorHandler_s)
C_LONGINT(DOM_PreviousValueOfError_l)
C_LONGINT(DOM_Error)

DOM_PreviousErrorHandler_s:=Method called on error
DOM_PreviousValueOfError_l:=Error
DOM_Error:=0   ` Assign a value in error method, if run.
Error:=0

ON ERR CALL("DOM_ErrorTrappingRoutine")
```

DOM_StopCustomErrorHandling

*DOM_StopCustomErrorHandling*ルーチンは、カスタムエラーハンドラを削除し、エラー情報を*DOM_StartCustomErrorHandling*で記憶した状態に戻します。

```
ON ERR CALL(DOM_PreviousErrorHandler_s)   ` Restore original error handler.
Error:=DOM_PreviousValueOfError_l   ` Restore original value of Error.
```

DOM_ErrorTrappingRoutine

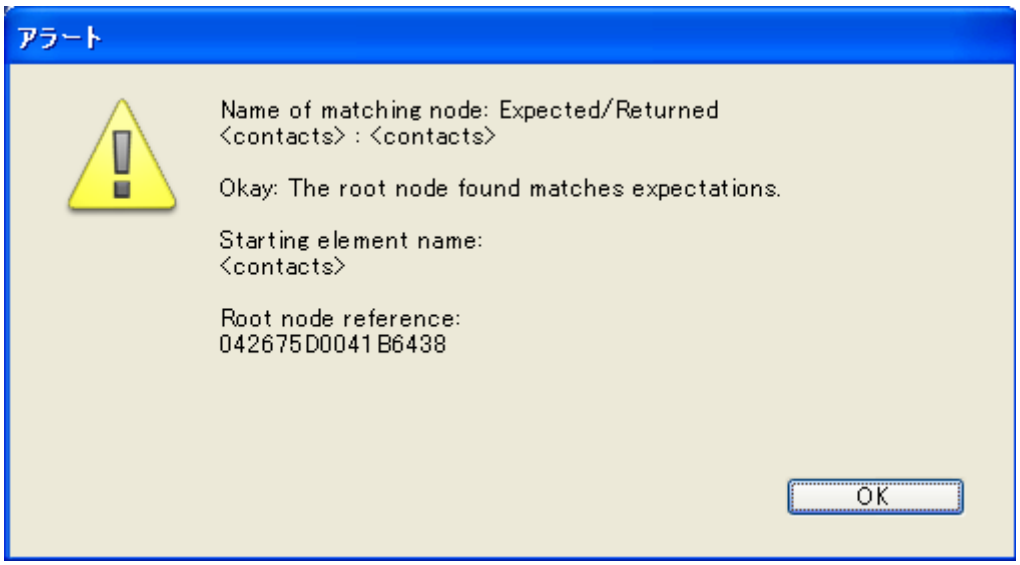
*DOM_ErrorTrappingRoutine*は*DOM_GetRootElementReference*によってインストールされ、無効なノード参照を読み込もうとしたときに生成されるエラーをトラップします。エラーハンドラには以下の一行が記述されています:

```
DOM_Error:=Error
```

Note *サンプルデータベースには、テクニカルノート05-41で紹介した大文字小文字を区別するルーチン*CS_AlphasAreEqual*も含まれています。*

The Sample Database

サンプルデータベースには、これまでに説明したコードと、簡単なテストルーチン*Test_GetRootElementReference*が含まれています。テストコードは、有効なノード、無効なノードなどを含むいろいろな条件下で*DOM_GetRootElementReference*メソッドを実行できるよう設計されています。テストごとにコードは、以下のようなテスト結果を表示します:



Summary

XMLドキュメントのタイプはDTDや要素名、要素属性の組み合わせで決定できる可能性があります。正確なルールは特定のドキュメントタイプにより異なりますが、4th DimensionのDOMコマンドを使用すれば必要な情報にいつでもアクセスすることができます。このテクニカルノートとサンプルデータベースでは、XHTMLドキュメントを特定し、4つのXHTML定義のうちどれが使われているのかを知る方法について説明しました。同様の基本的な方法は、他のXMLドキュメントタイプを検出するために使用することができます。