Adlib Gebruikersdag 2012

adlibXML/XSLT in de praktijk

Breukelen – 2 mei 2013

Solino de Baay

s.debaay@adlibsoft.com

Programma

- XML en adlibXML
- Transformeren
- XSL
  - Elementen
  - Functies
- Xpath

- **De praktijk**

XML

- XML betekent E**X**tensible **M**arkup **L**anguage
- XML bevat **data** en **tags**
- **Tags** zijn te vergelijken met veld en groepnamen
- XML tags staan niet vast zoals in **HTML,** maar moeten **zelf gekozen** worden.
- In **Adlib** worden de XML tags bepaald door de **namen** van velden en groepen in de data dictionary (in te stellen in Adlib Designer)
- Veldnamen en groepsnamen moeten dus voldoen aan de regels voor namen in **XML** (geen spaties bijv.)

XML - Een voorbeeld : structured XML

```
<?xml version="1.0" encoding="UTF-8" ?>
<adlibXML xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <recordList>
    <record>
      <lead_word>De</lead_word>
      <title>regels van het <highlight>huis</highlight> : Novelle</title>
      <author.name linkref="1439" linkfield="name" linkreffield="l1"><name>Graaf, Hermine de</name>
       </author.name>
      <year_of_publication>1988</year_of_publication>
      <priref>859</priref>
    </record>
    <record>
      <lead_word>De</lead_word>
      <title>weg naar <highlight>huis</highlight></title>
      <author.name linkref="1295" linkfield="name" linkreffield="l1"><name>Brakman, Willem</name> </author.name>
      <year_of_publication>1981</year_of_publication>
      <priref>657</priref>
    </record>
  </recordList>
</adlibXML>
```

XML-Elementen

- Declaratie

  <?xml version="1.0" encoding="UTF-8" ?>

- Namespace

  <adlibXML xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

- Knooppunten (Nodes)

```
<adlibXML>
  <recordList>
    <record>
      <lead_word>Het</lead_word>
      <title>behouden huis</title>
      <author.name linkref="1457" linkfield="name" linkreffield="l1">
        <name>Hermans, W.F.</name>
      </author.name>
      <edition>20e dr</edition>
      ...
```

XML-Elementen

- Atribuut = extra informatie over de data in een XML tag

```
<creator linkref="29" linkfield="name" linkreffield="l1">
   <name>Turner, Joseph Mallord William</name>
</creator>
```

- Voorwaarden
  - Eén topelement, de root
  - Alle elementen moeten afgesloten worden (begin- en eindtag, of leeg)
  - Alle elementen moeten correct gestapeld ('genest') zijn
  - Namen van tags en attributen mogen alleen letters, getallen en een aantal speciale tekens bevatten (zoals _)
  - Namen mogen niet beginnen met getal, interpunctie
  - Namen mogen geen spaties bevatten

dlib

XSLT

- XSL betekent E**X**tensible **S**tylesheet **L**anguage
- XSLT betekent **XSL** for **T**ransformations
- XSL gebruikt **XPATH** om binnen een XML documente elementen te selecteren

XSL-Transformatie

- **Browser**
  - **XSLT Compatibele browser**
  - **Voeg XSL-stylesheet toe aan XML**
  - **<?xml-stylesheet type="text/xsl" href="transform.xsl"?>**
- **msxsl.exe**
  - **Command line**
  - **C:\>msxsl Adlib.xml transform.xsl -o output.html**

- **Adlib API**
- **Adlwin**
- **Adlib Office-Connect**
- **Adlib Internet Server**

dlib

XPATH

- XPATH is een **syntax** om in een XML-document te zoeken en elementen te selecteren

- XPATH bevat **standaard functies** om bewerkingen te doen (vergelijkbaar met die in adapl)

dlib

XPATH Syntax

- **naam selecteert elementen met de tag naam vanaf huidige node plus onderliggende**
- **/ selecteert vanaf de root node**
- **// selecteert willekeurig vanaf de huidige node**
- **. selecteert de huidige node**
- **.. selecteert de bovenliggende node**
- **@ selecteert een attribuut**
- **[] selecteert een bepaalde index van een node**

XSL - Een voorbeeld (HTML uitvoer)

```
<?xml version='1.0' ?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
  <xsl:output method="html" />

  <xsl:template match="/">
    <xsl:apply-templates select="adlibXML" />
  </xsl:template>

  <!-- adlibXML generates the HTML framework-->
  <xsl:template match="adlibXML">
    <html>
      <head>
        <link rel="stylesheet" type="text/css" href="detail.css"/>
      </head>
      <body>
        <xsl:apply-templates select="recordList" />
      </body>
    </html>
  </xsl:template>
```

XSL – Een voorbeeld (HTML uitvoer)

Vervolg

```
<!-- The recordlist is placed in a table -->
 <xsl:template match="recordList">
   <table class="briefResultsTable">
     <xsl:apply-templates select="record" />
   </table>
 </xsl:template>

 <!-- Each record gets a row -->
 <xsl:template match="record">
   <tr>
     <td>
       <xsl:value-of select="title" />
     </td>
     <td>
       <xsl:apply-templates select="author.name/name"  />
     </td>
     <td>
       <xsl:apply-templates select="keyword.contents" />
     </td>
   </tr>
 </xsl:template>
</xsl:stylesheet>
```

dlib

XSL-Elementen (1)

- **<xsl:apply-templates select="expression"/>**
- **<xsl:template match="node"/>**
- **<xsl:value-of select="expression"/>**
- **<xsl:call-template name="name"/>**
- **<xsl:template name="name"/>**

dlib

XSL-Besturingselementen (1)

- **<xsl:if test="expression">**
  ...
  </xsl:if>

- **<xsl:choose>**
  <xsl:when test="expression">...<xsl:when>
  <xsl:otherwise>...<xsl:otherwise>
  </xsl:choose>

- **<xsl:for-each select="node">**
  ...
  </xsl:for-each>

dlib

XSL-Besturingselementen (2)

- **<xsl:sort select="node">**

- **<xsl:for-each select="node">**
  <xsl:sort select="node">
  <xsl:value-of select="expression">
  </xsl:for-each>

- **<xsl:apply-templates select="expression">**
  <xsl:sort select="node">
  </xsl:apply-templates>

dlib

XSL-Functies (1)

- **<xsl:value-of select="position()"/>**
  Selecteert het occurrence nummer van een element, begint bij 1

- **<xsl:value-of select="count(node)"/>**
  Telt het aantal occurrences (adapl repcnt())

- **<xsl:value-of select="first()"/>**

- **<xsl:value-of select="last()"/>**

- **<xsl:value-of select="node[position()=last()]"/>**

dlib

XSL-Functies (2) - String functies

- **<xsl:value-of select="contains(string1, string2)"/>**
- **<xsl:value-of select="substring(string, start, [length])"/>**
- **<xsl:value-of select="starts-with(string1, string2)"/>**
- **<xsl:value-of select="ends-with(string1, string2)"/>**
- **<xsl:value-of select="substring-before(string1, string2)"/>**
- **<xsl:value-of select="substring-after(string1, string2)"/>**

XML/XSL Voorbeelden – Grouped (1)

- **XML met meerdere occurrences van een veldgroep**

```
<adlibXML>
  <recordList>
    <record>
      <Inscription>
        <inscription.type><term>signatuur</term></inscription.type>
        <inscription.position>voorzijde  rechtsonder</inscription.position>
        <inscription.content>E. den Herder</inscription.content>
        <inscription.method>geschilderd</inscription.method>
      </Inscription>
      <Inscription>
        <inscription.type><term>opschrift</term></inscription.type>
        <inscription.position>voorzijde  links en rechts boven</inscription.position>
        <inscription.content>Walvischvaarder  Zeilzee</inscription.content>
        <inscription.method>geschilderd</inscription.method>
      </Inscription>
      ...
```

XML/XSL Voorbeelden – Grouped (2)

XSL voor XML met meerdere occurrences in groepen

```
<xsl:template match="Inscription">
  <tr>
    <td>
      <xsl:if test="position() = 1">
        <xsl:text>inscription</xsl:text>
      </xsl:if>
    </td>
    <td>
      <xsl:apply-templates select="inscription.type"/>
      <xsl:apply-templates select="inscription.position"/>
      <xsl:apply-templates select="inscription.method"/>
      <xsl:apply-templates select="inscription.content"/>
      <xsl:value-of select="inscription.content"/>
    </td>
  </tr>
</xsl:template>

<xsl:template match="inscription.type">
  <xsl:apply-templates select="term"/>
</xsl:template>
```

XML/XSL Voorbeelden – Grouped (2)

Vervolg

```
<xsl:template match="inscription.position">
  <xsl:text>, </xsl:text>
  <xsl:value-of select="."/>
 </xsl:template>

 <xsl:template match="inscription.content">
  <xsl:value-of select="."/>
 </xsl:template>

 <xsl:template match="inscription.method">
  <xsl:text>: </xsl:text>
  <xsl:value-of select="."/>
 </xsl:template>

 <xsl:template match="term">
  <xsl:value-of select="."/>
 </xsl:template>
```

XML/XSL Voorbeelden – Unstructured

- **XML met meerdere occurrences niet in groepen**

```
<record>
  <object_number>0206-O-04139</object_number>
  <title>Schilderij, voorstellende de Keizerstraat te Harderwijk, door H. Donkelaar</title>
  <creator><name>Donkelaar, G.H.</name></creator>
  <dimension.part>geheel</dimension.part>
  <dimension.part>geheel</dimension.part>
  <dimension.part>geheel</dimension.part>
  <dimension.type><term>hoogte</term></dimension.type>
  <dimension.type><term>breedte</term></dimension.type>
  <dimension.type><term>diepte</term></dimension.type>
  <dimension.value>3.4</dimension.value>
  <dimension.value>45.5</dimension.value>
  <dimension.value>20</dimension.value>
  <dimension.unit><term>dm</term></dimension.unit>
  <dimension.unit><term>cm</term></dimension.unit>
  <dimension.unit><term>mm</term></dimension.unit>
```

XML/XSL Voorbeelden - Unstructured

- **XSL voor XML met meerdere occurrences niet in groepen**

```
<xsl:template name="dimension">
  <xsl:if test="dimension.part != "">
     <tr><td>
        <xsl:call-template name="fieldname">
          <xsl:with-param name="fieldname">dimension</xsl:with-param>
        </xsl:call-template>
     </td><td>
      <xsl:for-each select="dimension.part">
        <xsl:variable name="counter"><xsl:value-of select="position()" /></xsl:variable>
        <li>
          <xsl:value-of select="." /> 
          <xsl:value-of select="../dimension.type[position()=$counter]/term" /> 
          <xsl:value-of select="../dimension.value[position()=$counter]" /> 
          <xsl:value-of select="../dimension.unit[position()=$counter]" />
        </li>
      </xsl:for-each>
     </td></tr>
  </xsl:if>
</xsl:template>
```

De praktijk (1)

Uitvoerformaten

- In te stellen in Adlib Designer
- De basis is unstructured adlibXML
- XSL moet hier rekening mee houden
- Adlib 7.1 krijgt de optie om grouped XML te kiezen

De praktijk (2)

De Adlib API

- De API kan alle typen adlibXML uitvoeren
- De API kan direct een stylesheet toepassen
- Adlib adviseert om *grouped* te gebruiken

De praktijk (3)

Office Connect

- Stylesheets te configureren op de API server (AdlibConnectPreferences.xml).

- De basis is grouped XML

De praktijk (4)

Debugging XSLT

- MSXSL.exe (command line)
  - Voordeel: direct een URL aanroepen mogelijk
- XSLT Sheet koppelen aan een API (browserbased)

dlib

Websites / Tutorials

- W3C
  - http://www.w3.org/Style/XSL/
- W3Schools
  - http://www.w3schools.com/
- TOPXML
  - http://www.topxml.com/xsl/tutorials/intro/
- SRU/SRW
  - http://www.loc.gov/standards/sru/

Meer informatie?

- Cursus InternetServer
- sales@nl.adlibsoft.com
- Adlib Helpdesk
  - 030 - 247 50 70
  - helpdesk@nl.adlibsoft.com
- http://www.adlibsoft.com/

dlib

Vragen?