

UNIDAD 7. XSLT – PARTE 2

TEORÍA

Lenguajes de Marcas
CFGS DAW 1

Autor: Pascual Ligeró
Revisado por: Vanessa Tarí – vanessatari@ceedcv.es
Dionisio García – dionisio.garcia@ceedcv.es
2020/2021
Versión: 180418.2030

Licencia





Reconocimiento – NoComercial – CompartirIgual (by-nc-sa): No se permite un uso comercial de la obra original ni de las posibles obras derivadas, la distribución de las cuales se debe hacer con una licencia igual a la que regula la obra original.

Nomenclatura

A lo largo de este tema se utilizarán distintos símbolos para distinguir elementos importantes dentro del contenido. Estos símbolos son:

 Importante

 Atención

 Interesante

ÍNDICE DE CONTENIDO

1. Introducción	4
1.1 El <xsl:template> elemento.....	4
1.2 El <xsl:value-of> Element	4
1.3 El <xsl:apply-templates> Element	4
2. El <xsl:for-each> Element	5
2.1 Filtrado de la salida.....	6
3. XSLT <xsl:sort> Element	7
3.1 ¿Dónde poner el tipo de información?	8
4. XSLT <xsl:if> Element	8
4.1 El <xsl:if> Element.....	9
5. XSLT <xsl:choose> Element.....	10
6. XSLT <xsl:ATTRIBUTE>	11
7. Bibliografía	12

UD07. XSLT PARTE 2

1. INTRODUCCIÓN

Por si no ha quedado claro en el bloque anterior el lenguaje XSL equivale a las hojas de estilo para XML. Aunque XSL es más que un lenguaje de hojas de estilo (CSS).

El lenguaje XSL consta de cuatro partes:

- XSLT - un lenguaje para transformar documentos XML
- XPath - un idioma para navegar en documentos XML
- XSL-FO - un lenguaje de formato para documentos XML (discontinued in 2013)
- XQuery - un lenguaje para la consulta de documentos XML

El XSLT se encarga de:

- Transforma un documento XML en otro documento XML
- Utiliza XPath para navegar en documentos XML

1.1 El <xsl:template> elemento

El <xsl:template> elemento se utiliza para construir las plantillas.

El **match atributo se utiliza para asociar una plantilla con un elemento XML**. El match atributo también se puede utilizar para definir una plantilla para todo el documento XML. El valor de la match atributo es una expresión XPath (por ejemplo match="/" define todo el documento) .

1.2 El <xsl:value-of> Element

El <xsl:value-of> elemento se utiliza para extraer el valor de un nodo seleccionado.

Ejemplo:

```
<table border="1">
<tr bgcolor="#9acd32">
<th>Title</th>
<th>Artist</th>
</tr>
<tr>
<td> <xsl:value-of select="catalog/cd/title"/> </td>
<td> <xsl:value-of select="catalog/cd/artist"/> </td>
</tr>
</table>
```

1.3 El <xsl:apply-templates> Element

El <xsl:apply-templates> elemento se aplica una plantilla al elemento actual o para nodos hijo del elemento actual.

2. EL <XSL:FOR-EACH> ELEMENT

El <xsl:for-each> elemento le permite hacer un bucle en XSLT.

El XSL <xsl:for-each> elemento puede ser usado para seleccionar cada elemento XML de un conjunto de nodos especificado:

Dado el siguiente XML:

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="bucle.xsl"?>
<catalog>
  <cd>
    <title>Empire Burlesque</title>
    <artist>Bob Dylan</artist>
    <country>USA</country>
    <company>Columbia</company>
    <price>10.90</price>
    <year>1985</year>
  </cd>
  <cd>
    <title>Hide your heart</title>
    <artist>Bonnie Tyler</artist>
    <country>UK</country>
    <company>CBS Records</company>
    <price>9.90</price>
    <year>1988</year>
  </cd>
  .....
</catalog>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="/">
  <html>
  <body>
  <h2>My CD Collection</h2>
  <table border="1">
    <tr bgcolor="#9acd32">
      <th>Title</th>
      <th>Artist</th>
    </tr>
    <xsl:for-each select="catalog/cd">
      <tr>
        <td><xsl:value-of select="title"/></td>
        <td><xsl:value-of select="artist"/></td>
      </tr>
    </xsl:for-each>
```

```
</table>
</body>
</html>
</xsl:template>

</xsl:stylesheet>
```

OJO, usando <xsl:apply-templates /> tal y como vimos en la primera parte nos **PERMITE OBTENER LO MISMO** sin bucles:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
<xsl:template match="/">
  <html>
  <body>
  <h2>My CD Collection</h2>
  <table border="1">
    <tr bgcolor="#9acd32">
      <th>Title</th>
      <th>Artist</th>
    </tr>
    <xsl:apply-templates />
  </table>
  </body>
  </html>
</xsl:template>
<xsl:template match="cd">
  <tr>
    <td><xsl:value-of select="title"/></td>
    <td><xsl:value-of select="artist"/></td>
  </tr>
</xsl:template>

</xsl:stylesheet>
```

2.1 Filtrado de la salida

También podemos filtrar la salida del archivo XML mediante la adición de un criterio para el select atributo en el <xsl:for-each> elemento.

```
<xsl:for-each select="catalog/cd[artist='Bob Dylan']">
```

De esta forma de todos los discos que hay en el catálogo sólo nos presentará los del artista Bob Dylan.

Operadores de filtro posibles:

- = (Igual)
- != (not equal)
- & lt; menos que
- & gt; mas grande que

Ejemplo XSL:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="/">
  <html>
  <body>
    <h2>My CD Collection</h2>
    <table border="1">
      <tr bgcolor="#9acd32">
        <th>Title</th>
        <th>Artist</th>
      </tr>
      <xsl:for-each select="catalog/cd[artist='Bob Dylan']">
        <tr>
          <td><xsl:value-of select="title"/></td>
          <td><xsl:value-of select="artist"/></td>
        </tr>
      </xsl:for-each>
    </table>
  </body>
</html>
</xsl:template>

</xsl:stylesheet>
```

3. XSLT <XSL:SORT> ELEMENT

El <xsl:sort> elemento se utiliza para ordenar la salida.

3.1 ¿Dónde poner el tipo de información?

Para ordenar la salida, basta con añadir un `<xsl:sort>` elemento dentro de la `<xsl:for-each>` elemento en el archivo XSL:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="/">
  <html>
  <body>
  <h2>My CD Collection</h2>
  <table border="1">
    <tr bgcolor="#9acd32">
      <th>Title</th>
      <th>Artist</th>
    </tr>
    <xsl:for-each select="catalog/cd">
      <xsl:sort select="artist" order="descending" />
      <tr>
        <td><xsl:value-of select="title"/></td>
        <td><xsl:value-of select="artist"/></td>
      </tr>
    </xsl:for-each>
  </table>
  </body>
  </html>
</xsl:template>

</xsl:stylesheet>
```

Parámetros de sort:

- lang: "language-code".
- data-type: "text | number | qname".
- order: "ascending | descending".
- case-order: "upper-first | lower-first".

4. XSLT <XSL:IF> ELEMENT

El `<xsl:if>` elemento se utiliza como condicional a la hora de obtener contenido del XML.

4.1 El <xsl:if> Element

Para agregar una prueba condicional hay que añadir el <xsl:if> dentro del bucle <xsl:for-each>

El siguiente XSL solo muestra los discos del catalogo que cuestan más de 10:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="/">
  <html>
  <body>
  <h2>My CD Collection</h2>
  <table border="1">
    <tr bgcolor="#9acd32">
      <th>Title</th>
      <th>Artist</th>
      <th>Price</th>
    </tr>
    <xsl:for-each select="catalog/cd">
      <xsl:if test="price > 10">
        <tr>
          <td><xsl:value-of select="title"/></td>
          <td><xsl:value-of select="artist"/></td>
          <td><xsl:value-of select="price"/></td>
        </tr>
      </xsl:if>
    </xsl:for-each>
  </table>
  </body>
  </html>
</xsl:template>

</xsl:stylesheet>
```

test admite:

- para mayor que: > test="price > 10"
- para menor que: < test="price < 10"
- para igual que: = 'cadena'; test="price='10'"

5. XSLT <XSL:CHOOSE> ELEMENT

El <xsl:choose> elemento se utiliza en conjunción con <xsl:when> y <xsl:otherwise> para expresar múltiples pruebas condicionales.

```
<xsl:choose>
  <xsl:when test="price > 10"> <!--Si se cumple la condición -->
    <td bgcolor="#ff00ff">
      <xsl:value-of select="artist"/></td>
    </xsl:when>
    <xsl:otherwise> <!--Si NO se cumple la condición -->
      <td><xsl:value-of select="artist"/></td>
    </xsl:otherwise>
</xsl:choose>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">
  <html>
  <body>
    <h2>My CD Collection</h2>
    <table border="1">
      <tr bgcolor="#9acd32">
        <th>Title</th>
        <th>Artist</th>
      </tr>
      <xsl:for-each select="catalog/cd">
        <tr>
          <td><xsl:value-of select="title"/></td>
          <xsl:choose>
            <xsl:when test="price > 10">
              <td bgcolor="#ff00ff">
                <xsl:value-of select="artist"/></td>
            </xsl:when>
            <xsl:otherwise>
              <td><xsl:value-of select="artist"/></td>
            </xsl:otherwise>
          </xsl:choose>
        </tr>
      </xsl:for-each>
    </table>
  </body>
</html>
</xsl:template>
</xsl:stylesheet>
```

6. XSLT <XSL:ATTRIBUTE>

```
<xsl:attribute name="nombre"><xsl:value-of select="nombre" />
</xsl:attribute>
```

Permite añadir atributos a una etiqueta.

```
<?xml version="1.0" encoding="UTF-8"?>
<museos>
  <museo>
    <nombre>Museo del Prado</nombre>
    <ciudad>Madrid</ciudad>
    <pais>España</pais>
  </museo>
  <museo>
    <nombre>British Museum</nombre>
    <ciudad>Londres</ciudad>
    <pais>Reino Unido</pais>
  </museo>
  <museo>
    <nombre>National Gallery</nombre>
    <ciudad>Londres</ciudad>
    <pais>Reino Unido</pais>
  </museo>
</museos>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
version="1.0">
  <xsl:template match="/">
    <museos>
      <xsl:apply-templates />
    </museos>
  </xsl:template>

  <xsl:template match="museo">
    <museo>
      <xsl:attribute name="nombre"><xsl:value-of select="nombre" />
    </xsl:attribute>
    <xsl:attribute name="ciudad"><xsl:value-of select="ciudad" />
    </xsl:attribute>
    <xsl:attribute name="pais"><xsl:value-of select="pais" />
    </xsl:attribute>
  </museo>
</xsl:template>
</xsl:stylesheet>
```

7. BIBLIOGRAFÍA

https://www.w3schools.com/xml/xsl_intro.asp