

12

# XSLT

EXtensible Stylesheet Language Transformations

# XSLT: Introducción

13

- Se puede usar para transformar XML en HTML antes de ser presentado por el navegador.
- Ver ejemplo `rss_sin_xsl.xml`
  - ▣ Ver fuente
  - ▣ Ver en navegador
    - sin xsl asociado
    - Con xsl asociado
  - ▣ Otro ejemplo: `hamlet.xml`, con `willy.xsd` asociada
- La transformación se puede hacer en cliente (navegador) o en servidor.
  - ▣ Por ejemplo en javascript o en PHP

# XSLT

14

- La parte más importante de XSL.
- Se usa para transformar
  - ▣ Un documento XML en otro documento XML,
    - O en otro documento que pueda ser reconocido por un navegador (HTML o XHTML).
  - ▣ Normalmente XSLT hace esto transformando cada elemento XML en un elemento HTML.
- Con XSLT se pueden añadir o eliminar elementos y atributos.
- También ordenar, realizar test, tomar decisiones sobre que elementos ocultar o mostrar, etc.
- Se suele decir que XSLT transforma un árbol XML fuente en un árbol XML resultado.

# XSLT: incluyendo el estilo

15

cd.xml

- Modificamos el fichero de datos para hacer referencia al fichero xslt, un ÚNICO cambio:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
```

```
<?xml-stylesheet type='text/xsl'  
    href='cd_catalog_xslt.xsl' version='1.0'?>
```

cd.xsl

- Vemos el archivo de presentación asociado:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
```

```
<xsl:stylesheet version="1.0"  
    xmlns:xsl=http://www.w3.org/1999/XSL/Transform>
```

```
<xsl:template match="/CATALOG">
```

```
<html>
```

```
<body> <h2> My CD .. </h2>...
```

¡ Es XML!

“quedarnos” con el  
aspecto/estructura, luego se  
analizará en detalle

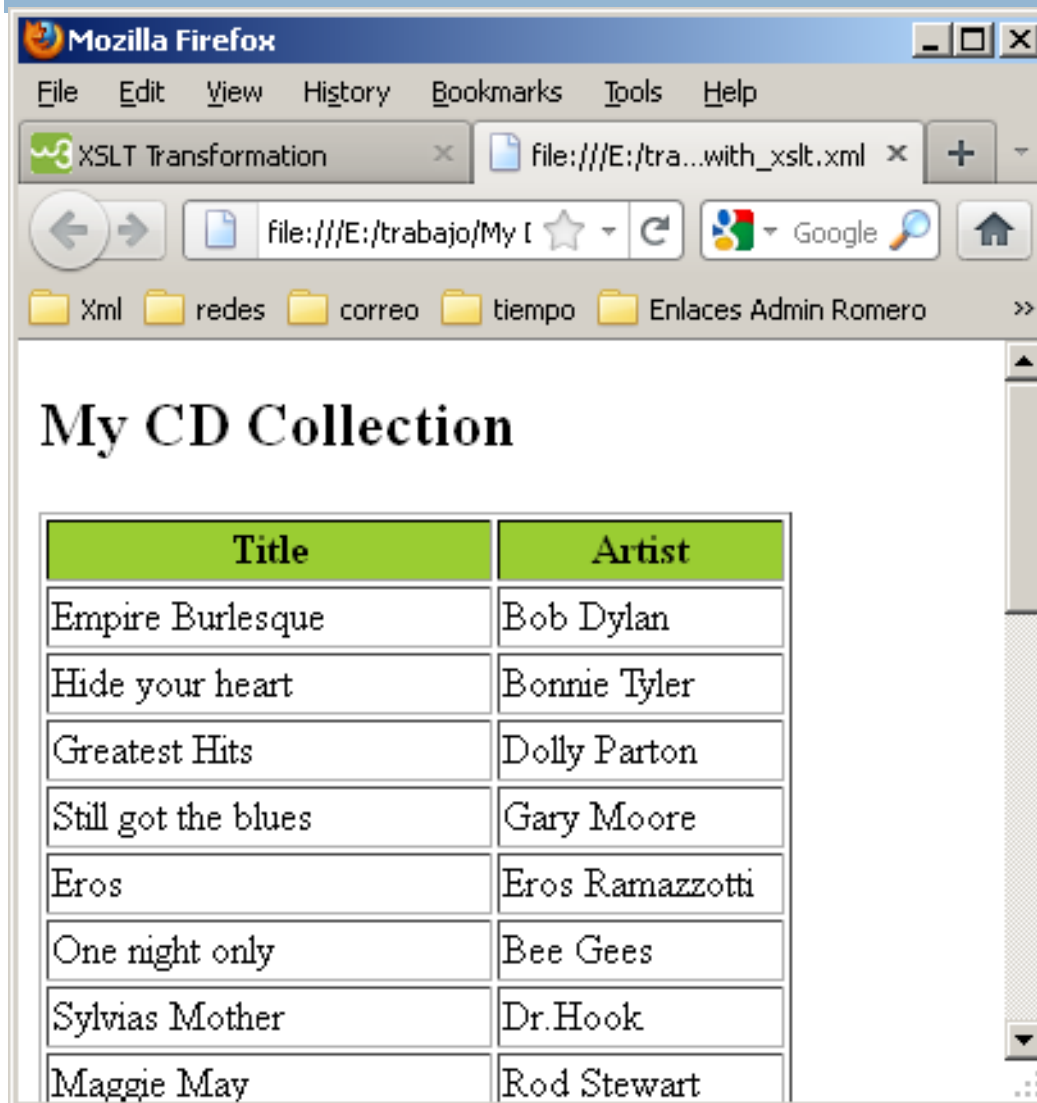
# XSLT: cd\_catalog\_xslt.xsl

16

```
<table border="1">
  <tr>
    <th align="left">Title</th>
    <th align="left">Artist</th>
  </tr>
  <xsl:for-each select="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>
```

# XSLT: Resultado en navegador

17



The screenshot shows a Mozilla Firefox browser window. The address bar displays the file path: file:///E:/trabajo/My I. The browser's menu bar includes File, Edit, View, History, Bookmarks, Tools, and Help. The browser's toolbar includes back, forward, home, and search buttons. The browser's status bar shows the file path: file:///E:/trabajo/My I. The browser's content area displays the title 'My CD Collection' and a table with 8 rows of CD data.

Title	Artist
Empire Burlesque	Bob Dylan
Hide your heart	Bonnie Tyler
Greatest Hits	Dolly Parton
Still got the blues	Gary Moore
Eros	Eros Ramazzotti
One night only	Bee Gees
Sylvias Mother	Dr.Hook
Maggie May	Rod Stewart

# XSLT: ¿Cómo funciona?

18

- Una hoja de estilos XSL consiste de uno o más conjunto de reglas llamadas “templates” o plantillas.
- Una plantilla contiene reglas que se aplican cuando un nodo coincide con un patrón.

# XSLT: Declaración hoja de estilo

19

- Es un documento XML
  - ▣ Para validar en XMLCopyEditor pulsar Ctrl+9
- El elemento raíz de una hoja de estilo

```
<xsl:stylesheet version="1.0"
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
```

o:

```
<xsl:transform version="1.0"
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
```
- Para acceder a los elementos, atributos, etc. debemos declarar el espacio de nombres XSLT al comienzo del documento.
  - ▣ `xmlns:xsl="http://www.w3.org/1999/XSL/Transform"`
  - ▣ Debemos incluir también el atributo `version="1.0"`.

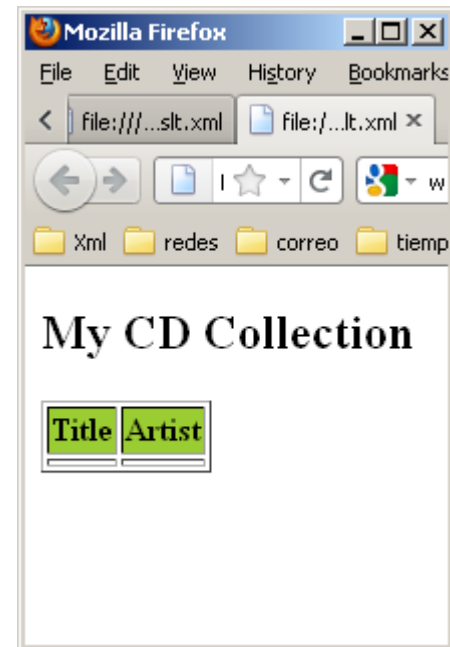


# XSLT: Poco a poco: version 01

20

```
<?xml version="1.0" encoding="ISO-8859-1"?>  
  <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>  
          <tr>  
            <td>.</td>  
            <td>.</td>  
          </tr>  
        </table>  
      </body>  
    </html>  
  </xsl:template>  
</xsl:stylesheet>
```



# XSLT: Primera aproximación

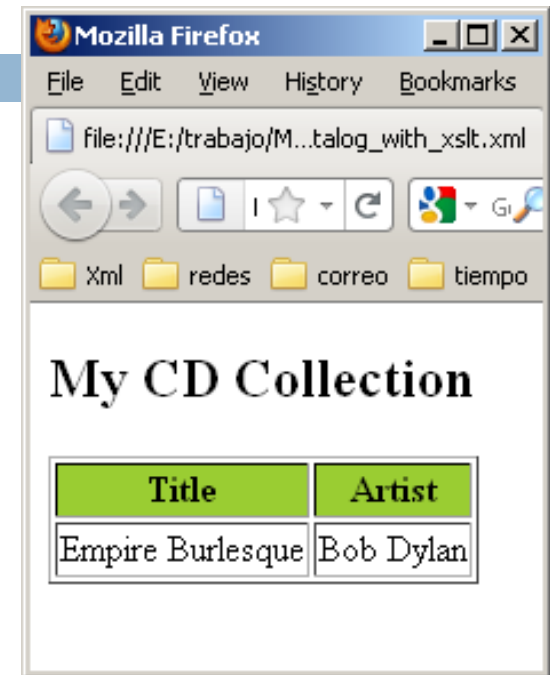
21

- La hoja de estilos es un doc XML
  - ▣ Comienza con una declaración XML  
`<?xml version="1.0" encoding="ISO-8859-1"?>.`
- El elemento `<xsl:stylesheet>` define que es una hoja de estilo XSLT, y añade como atributos la versión y el espacio de nombres.
- El elemento `<xsl:template>` define una plantilla:
  - ▣ El atributo `match="/"` asocia la plantilla con la raíz del documento XML fuente.
  - ▣ El contenido dentro de la plantilla define en este caso código HTML que formará parte del documento de salida o resultado.

# XSLT: Poco a poco: versión 02

22

```
□ <xsl:template match="/">
  <html>
  <body>
  <h2>My CD Collection</h2>
  <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>
  </body>
  </html>
</xsl:template>
```



# XSLT: Poco a poco: versión 02

23

- El elemento **<xsl:value-of>** se usa para extraer el valor del nodo seleccionado, y añadirlo al documento de salida resultante de la transformación.
- El atributo **select** contiene una expresión “Xpath”.
  - ▣ Una expresión Xpath funciona de forma similar a navegar en un sistema de ficheros: la (/) selecciona subdirectorios, en este caso nodos hijos..
- Resultado del ejemplo anterior: sólo una línea de datos en la salida..
  - ▣ Vamos a mejorarlo con el elemento **<xsl:for-each>**
- Con **<xsl:for-each>** tendremos un bucle para iterar sobre los elementos XML y poder mostrar todos los elementos

# XSLT: Poco a poco: version 03

24

```
<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>
```



# XSLT: Poco a poco: version 03

25

- El elemento `<xsl:for-each>` permite hacer un “bucle” en XSLT.
- Se usa para seleccionar cada uno de los elementos de un conjunto especificado de nodos del árbol XML.
- Observe con detenimiento
  - Atributo **match** del elemento **xsl:template**
  - Atributo **select** del elemento **xsl:for-each**
  - Atributo **select** del elemento **xsl:value-of**

# XSLT: Filtrando la salida

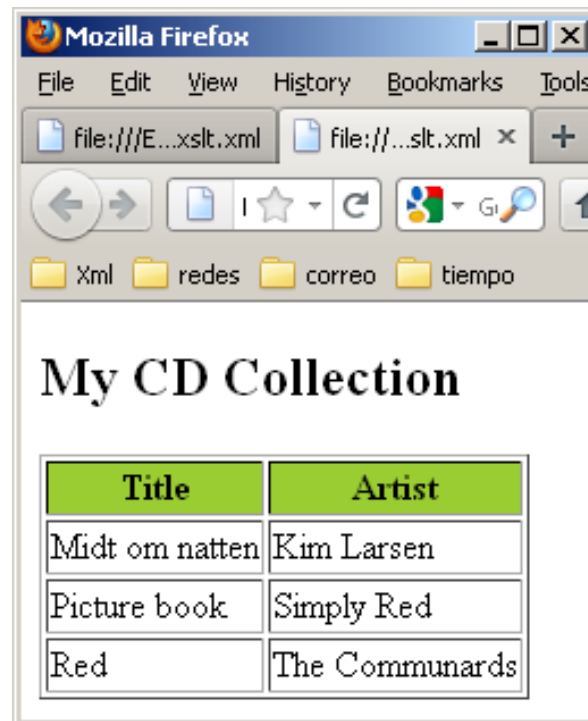
26

- Podemos filtrar la salida añadiendo un criterio al atributo select del elemento `<xsl:for-each>`:
- Operadores válidos son:
  - = (equal)
  - != (not equal)
  - &lt; less than
  - &gt; greater than
- Ejemplo:  
`<xsl:for-each select="catalog/cd[artist='Bob Dylan']">`

# XSLT: Ejemplos

27

- ❑ Filtrar discos de 1987
- ❑ Filtrar discos de precio inferior a 7.90
- ❑ Etc..





# XSLT:

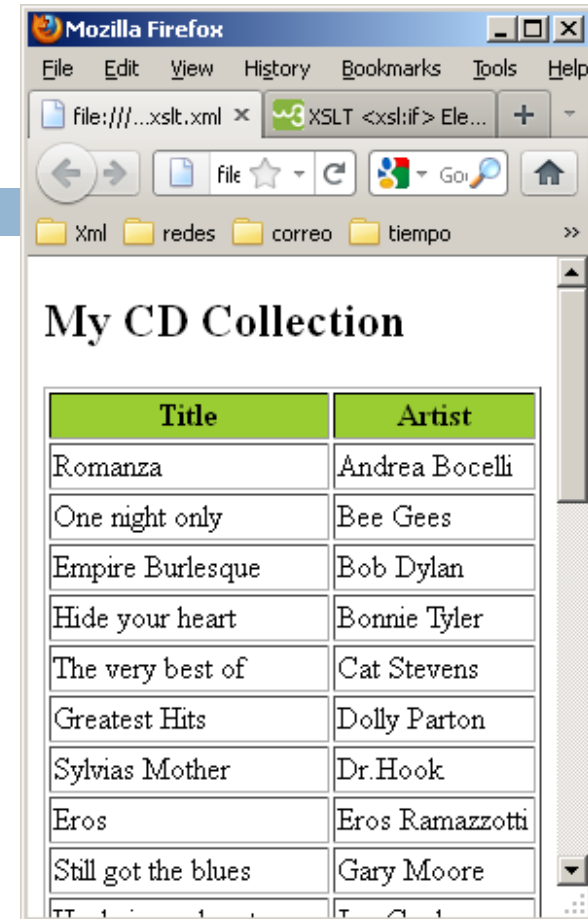
## Ordenando la salida

28

- Para ordenar, basta añadir el elemento **<xsl:sort>** dentro del elemento **<xsl:for-each>**  
**<xsl:for-each select="CATALOG/CD">**  
**<xsl:sort select="ARTIST" />**

### Attributes

Attribute	Value	Description
select	XPath-expression	Optional. Specifies which node/node-set to sort on
lang	language-code	Optional. Specifies which language is to be used by the sort
data-type	text number qname	Optional. Specifies the data-type of the data to be sorted. Default is "text"
order	ascending descending	Optional. Specifies the sort order. Default is "ascending"
case-order	upper-first lower-first	Optional. Specifies whether upper- or lowercase letters are to be ordered first



# XSLT: Selectivas: <xsl:if>

29

## □ Sintaxis

```
<xsl:if test="expresión">
```

salida si la expresión es verdadera...

```
</xsl:if>
```

## □ Se suele añadir dentro del elemento <xsl:for-each> :

```
<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>
```

```
</tr>
```

```
</xsl:if>
```

```
</xsl:for-each>
```

# XSLT: choose

30

- Como una selectiva múltiple

```
<xsl:choose>
```

```
  <xsl:when test="expression">
```

```
    ... salida...
```

```
  </xsl:when>
```

```
  ...
```

```
  <xsl:otherwise>
```

```
    ... Si no se cumple ninguna
```

```
  </xsl:otherwise>
```

```
</xsl:choose>
```

# XSLT: Ejemplo

31

```
<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:when test="price > 9">
<td bgcolor="#cccccc">
<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>
```



The screenshot shows a Mozilla Firefox browser window with the title "My CD Collection". The address bar shows "file:///E:/tra". The browser displays a table with two columns: "Title" and "Artist". The table contains 16 rows of data. The "Artist" column uses different background colors for each row, corresponding to the XSLT logic: magenta for prices greater than 10, light gray for prices greater than 9, and white for other cases.

Title	Artist
Empire Burlesque	Bob Dylan
Hide your heart	Bonnie Tyler
Greatest Hits	Dolly Parton
Still got the blues	Gary Moore
Eros	Eros Ramazzotti
One night only	Bee Gees
Sylvias Mother	Dr.Hook
Maggie May	Rod Stewart
Romanza	Andrea Bocelli
When a man loves a woman	Percy Sledge
Black angel	Savage Rose
1999 Grammy Nominees	Many
For the good times	Kenny Rogers
Big Willie style	Will Smith
Tupelo Honey	Van Morrison
Soulsville	Jorn Hoel
The very best of	Cat Stevens

# Incluso añadir estilos CSS

32

## □ Directamente en el fichero xsl

```
<xsl:template match="/">
<html>
<head>
  <style type="text/css">
    body {color: brown;}
  </style>
```

## □ Mediante referencia a un fichero CSS

```
<xsl:template match="/">
<html>
<head>
<link rel="stylesheet" type="text/css"
      href="cd.css" />
```

# XSLT: Diferencia XSLT y CSS

33

- CSS se concentran en cómo se presentan los datos.
- XSLT cambia la estructura y el tipo de los datos XML
  - ▣ Puede añadir, eliminar, duplicar, ordenar nodos..
  - ▣ XSLT puede transformar un XML en otro, usando incluso un vocabulario XML distinto del original
    - Crea páginas HTML.
- XSLT usa la sintaxis XML,
- CSS tiene su propia sintaxis

# Ejercicio sobre presentación (I)

34

- Modificar cd\_catalog.xml
  - ▣ => cd\_catalog\_xslt.xml
  - ▣ Hacer referencia a la hoja de estilos
    - cd\_catalog\_xslt.xsl
  - ▣ Probar presentación con esa hoja
- Modificar para incluir más datos de cada CD: año, precio, etc..
- Modificar para poner “bonita” la tabla: colores, fondos, etc..
- **¿Atributos?: @nombre\_atributo**

# Ejercicio sobre presentación

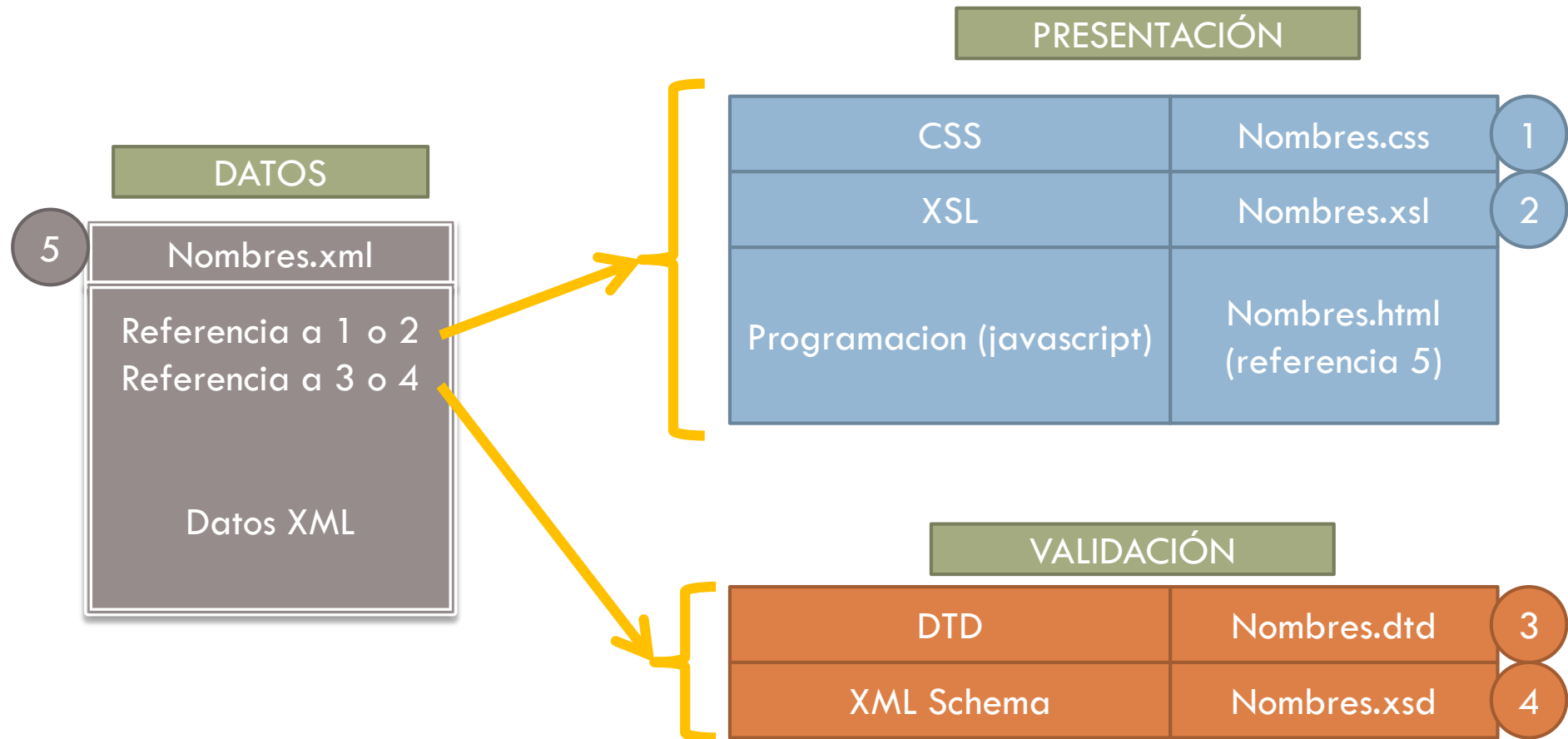
35

- Modificar cd\_catalog.css
  - ▣ cambiar tipo de letra o color, en alguno de los elementos.
- Modificar cd\_catalog.xml
  - ▣ => cd\_catalog\_xslt.xml
  - ▣ Hacer referencia a la hoja de estilos
    - cd\_catalog\_xslt.xsl
  - ▣ Probar presentación con esa hoja
- Modificar para incluir más datos de cada CD: año, precio, etc..
- Modificar para poner “bonita” la tabla: colores, fondos, etc..



# Esquema resumen

36



# Atributos

37

Ojo con XPATH

```
<xsl:template match="/catalog">
```

...

```
<h1>Discos de <xsl:value-of select="@propietario" /> </h1>
```

```
<table border="1">
```

```
<tr>    ...
```

```
<xsl:for-each select="cd" >
```

...

```
<tr class="caro">
```

```
<td><xsl:value-of select="title" /> </td>
```

```
<td> <xsl:value-of select="artist" />  
    ( <xsl:value-of select="artist/@pais" /> ) </td>
```

# Ordenar por más de un criterio

38

```
<xsl:template match="employees">  
  <xsl:apply-templates>  
    <xsl:sort select="last"/>  
    <xsl:sort select="first"/>  
  </xsl:apply-templates>  
</xsl:template>
```

# XSLT transforma.. a XML también..

39

```
<xsl:template match="/catalog">
  <coleccion>
    <xsl:for-each select="cd ">
      <xsl:sort select="price" data-type="number" order="descending" />
      <disco>
        <cancion>
          <xsl:value-of select="title" /> de <xsl:value-of select="artist" />
        </cancion>
      </disco>
    </xsl:for-each>
  </coleccion>
</xsl:template>
```

# Otra aproximación: apply-templates

40

- El elemento `<xsl:apply-templates>` aplica la plantilla al element actual o a sus hijos
  - ▣ Si le añadimos el atributo `select`, procesará solamente los hijos que cumplan la condición del `select`.
- Se puede utilizar el atributo `select` para especificar el orden en el que se procesan los nodos hijos.
- *xsl:apply-templates* which will check for template matches on all the children of *class*.

# <xsl:apply-templates>

[http://www.w3schools.com/xsl/xsl\\_apply\\_templates.asp](http://www.w3schools.com/xsl/xsl_apply_templates.asp)

41

```
<xsl:template match="/">
  <html>
  <body>
    <h2>My CD Collection</h2>
    <xsl:apply-templates/>
  </body>
</html>
</xsl:template>
```

```
<xsl:template match="cd">
  <p>
    <xsl:apply-templates select="title"/>
    <xsl:apply-templates select="artist"/>
  </p>
</xsl:template>
```

```
<xsl:template match="title">
  Title: <xsl:value-of select="."/>
  <br />
</xsl:template>
```

```
<xsl:template match="artist">
  Artist: <xsl:value-of select="."/>
  <br />
</xsl:template>
```

# Ejercicios

42

- Elecciones.xml
  - ▣ DTD
  - ▣ Presentación con CSS
  - ▣ Presentación con XSLT
- playlist
  - ▣ DTD
  - ▣ Presentación con CSS
  - ▣ Presentación con XSLT

# Presentación desde programación

Lado cliente

Lado servidor



# Presentación: con javascript

[http://www.w3schools.com/xml/tryit.asp?filename=tryxml\\_parsertest](http://www.w3schools.com/xml/tryit.asp?filename=tryxml_parsertest)

44

- Es un fichero XML, luego necesito:
  - ▣ Fichero html para incluir el javascript
  - ▣ Código JavaScript que procesa el fichero xml
- Nociones javascript
  - ▣ **Tomar de apuntes librosweb/w3schools**
  - ▣ **Modelo DOM**
  - ▣ **Noción de objeto XMLHttpRequest**
- Usar jQuery !!
  - ▣ .load()
  - ▣ .ajax
- Integrar con programación

# Con javascript: tres pasos

45

- El objeto **XMLHttpRequest**
  - ▣ Se usa para recuperar información del servidor
- Se abre la conexión, se envia petición, se espera respuesta,
  - ▣ Open, send, etc..
- Dos descriptors, uno para xml y otro para xslt
  - ▣ Se crea objeto XSLTProcessor
  - ▣ Se importa XSL y se transforma  
`xsltProcessor.importStylesheet(xsl);`
  - ▣ Se incorpora la transormación al html  
`document.getElementById("example").appendChild(resultDocument);`

# Desde JavaScript

46

□ [http://www.w3schools.com/xsl/xsl\\_client.asp](http://www.w3schools.com/xsl/xsl_client.asp)

```
function loadXMLDoc(filename) {  
    //...  
    xhttp = new XMLHttpRequest();  
    xhttp.open("GET", filename, false);  
    xhttp.send("");  
    return xhttp.responseXML;  
}
```

# Desde JavaScript (II)

47

□ [http://www.w3schools.com/xsl/xsl\\_client.asp](http://www.w3schools.com/xsl/xsl_client.asp)

```
function displayResult() {  
    xml = loadXMLDoc("cdcatalog.xml");  
    xsl = loadXMLDoc("cdcatalog.xsl");  
    // ...  
    xsltProcessor = new XSLTProcessor();  
    xsltProcessor.importStylesheet(xsl);  
    resultDocument =  
        xsltProcessor.transformToFragment(xml, document);  
  
    document.getElementById(  
        "example").appendChild(resultDocument);  
}
```

# Desde PHP

[http://www.w3schools.com/xsl/xsl\\_server.asp](http://www.w3schools.com/xsl/xsl_server.asp)

48

```
<?php
// Load XML file
$xml = new DOMDocument;
$xml->load('cdcatalog.xml');

// Load XSL file
$xsl = new DOMDocument;
$xsl->load('cdcatalog.xsl');

// Configure the transformer
$proc = new XSLTProcessor;

// Attach the xsl rules
$proc->importStyleSheet($xsl);

echo
    $proc->transformToXML($xml);
?>
```

# Generar etiqueta con atributo

49

## □ Ejemplos de uso

### ▣ Incluir un enlace en la página resultado

- atributo href en elemento <a>

### ▣ Incluir una imagen en la salida:

- atributo src en etiqueta <img>

18:00

	Duración	Localización
	755	<a href="http://media.ArtistServer.com/tracks/367/14107/1/1/5e3012625b1011367d4e/0/Planet%20Bliss%20-%20Rendered.mp3">http://media.ArtistServer.com/tracks/367/14107/1/1/5e3012625b1011367d4e/0/Planet Bliss - Rendered.mp3</a>
	567	<a href="http://media.ArtistServer.com/tracks/367/16018/1/1/5e3012625b1011367d4e/0/Planet%20Bliss%20-%20Cital%20Club.mp3">http://media.ArtistServer.com/tracks/367/16018/1/1/5e3012625b1011367d4e/0/Planet Bliss - Cital Club.mp3</a>
om	442	<a href="http://media.ArtistServer.com/tracks/367/17207/1/1/5e3012625b1011367d4e/0/Planet%20Bliss%20-%20Time%27s%20Kingdom.mp3">http://media.ArtistServer.com/tracks/367/17207/1/1/5e3012625b1011367d4e/0/Planet Bliss - Time%27s Kingdom.m</a>
	341	<a href="http://media.ArtistServer.com/tracks/367/14048/1/1/5e3012625b1011367d4e/0/Planet%20Bliss%20-%20Xeo.mp3">http://media.ArtistServer.com/tracks/367/14048/1/1/5e3012625b1011367d4e/0/Planet Bliss - Xeo.mp3</a>
	125	<a href="http://media.ArtistServer.com/tracks/367/16607/1/1/5e3012625b1011367d4e/0/Planet%20Bliss%20-%20Nomad%20Soul.mp3">http://media.ArtistServer.com/tracks/367/16607/1/1/5e3012625b1011367d4e/0/Planet Bliss - Nomad Soul.mp3</a>
	0123	<a href="http://media.ArtistServer.com/tracks/367/15648/1/1/5e3012625b1011367d4e/0/Planet%20Bliss%20-%20Willow.mp3">http://media.ArtistServer.com/tracks/367/15648/1/1/5e3012625b1011367d4e/0/Planet Bliss - Willow.mp3</a>
1	56	<a href="http://media.ArtistServer.com/tracks/367/17648/1/1/5e3012625b1011367d4e/0/Planet%20Bliss%20-%20Magnolia%20Bloom.mp3">http://media.ArtistServer.com/tracks/367/17648/1/1/5e3012625b1011367d4e/0/Planet Bliss - Magnolia Bloom.mp3</a>
ie	9	<a href="http://media.ArtistServer.com/tracks/367/16884/1/1/5e3012625b1011367d4e/0/Planet%20Bliss%20-%20Wishing%20Machine.mp3">http://media.ArtistServer.com/tracks/367/16884/1/1/5e3012625b1011367d4e/0/Planet Bliss - Wishing Machine.mp3</a>

# Generar etiqueta con atributo (II)

## (3 formas)

```
<td> <xsl:element name="a">
  <xsl:attribute name="href">
    <xsl:value-of select="location" />
  </xsl:attribute>
  <xsl:value-of select="title" />
</xsl:element> </td>
```

```
<td> <a>
  <xsl:attribute name="href">
    <xsl:value-of select="location" />
  </xsl:attribute>
  <xsl:value-of select="title" />
</a></td>
```

```
<td>
  <a href="{location}"> <xsl:value-of select="title" /> </a>
</td>
```

# Ejercicio

51

- Modificar playlist para que el nombre de la canción sea un enlace a la url location

**ArtistServer.com: Playlist, la lista musical de Jane Doe**

Versión de la playlist: 1

Fecha: 2013-05-25T14:52:48-08:00

Artista	Canción	Duración	Localización
Bliss Bliss	Rendered	755	<a href="#">Rendered</a>
Planet Bliss	Cital Club	567	<a href="#">Cital Club</a>
Planet	Time 27s Kingdom	442	<a href="#">Time 27s Kingdom</a>
Planet Bliss	Xeo	341	<a href="#">Xeo</a>
Bliss	Nomad Soul	125	<a href="#">Nomad Soul</a>
Planet Planet	Willow	0123	<a href="#">Willow</a>
Bliss	Magnolia Bloom	56	<a href="#">Magnolia Bloom</a>
Planet	Wishing Machine	9	<a href="#">Wishing Machine</a>