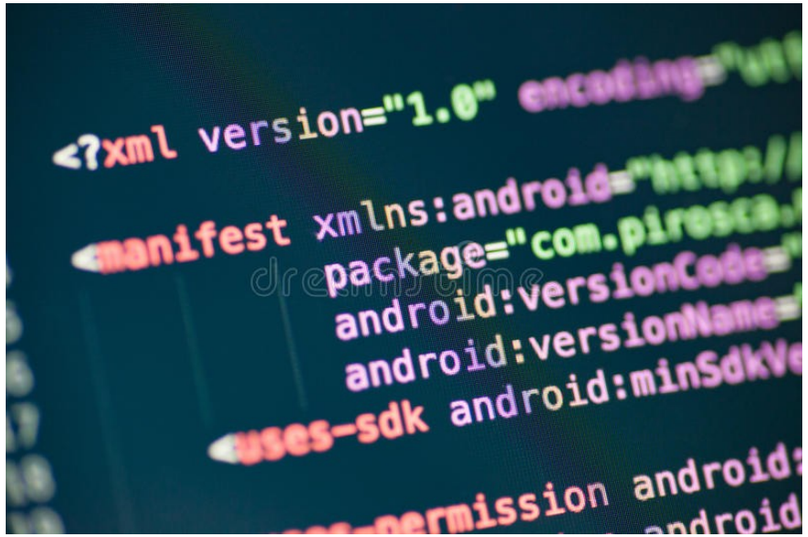
Lenguajes de marcas y sistemas de gestión de la información



UT06 – Conversión de documentos

1 – XSLT – Introducción

IES Clara del Rey – Madrid

XSL

XSL: eXtensible Stylesheet Language.

CSS es una forma de definir como presentar HTML.

XSL es un sistema para definir cómo presentar un documento XML.

XSL está compuesto de:

- XSLT: lenguaje para transformar XML
- XPath: lenguaje para navegar / seleccionar elementos o atributs en documentos XML
- XSL-FO: lenguaje para formatear XML
- XQuery: lenguaje para realizar consultas en un documento XML.

XSLT

XSLT: XSL Transformations.

Es la parte más importante de XSL.

Permite transformar XML en otros formatos:

- XML > XML
- XML > HTML
- XML > Texto
- Etc.

Usa XPath para navegar y seleccionar elementos dentro de los documentos XML que transforma.

XSLT

XSLT permite transformar un documento XML en otro documento, y para hacerlo:

- Se basa en un sistema de plantillas que podemos aplicar a diferentes elementos XML.
- Usa XPath para definir los elementos/atributos del XML a los que debemos aplicar cada plantilla.
- Permite evaluar condiciones, iterar por conjuntos de elementos/atributos, ordenar y recolocar elementos/atributos, realizar cálculos con los elementos/atributos, etc.

XSLT

Para poder realizar transformaciones XSLT es necesario conocer:

- XML
- Elementos XSLT
- XPath
- Funciones XSLT
- El lenguaje o lenguajes del documento que queremos obtener con la transformación.

Lo más habitual es transformar XML en un HTML al que se aplican luego estilos con CSS.

XSLT

XSLT funciona, a rasgos generales, de la siguiente forma:

- Usa XPath para definir los elementos del XML original que queremos transformar.
- A cada uno de los elementos que queremos transformar se le aplica una plantilla específica.
- El resultado de aplicar las distintas plantillas a los elementos especificados será el documento final o de salida.

XSLT – Ejemplo – Documento XML

```xml
<?xml version="1.0" encoding="UTF-8"?>
<bibioteca>
    <libro>
        <titulo>La mujer del viajero en el tiempo</titulo>
        <autor>Audry Niffeneger</autor>
        <anio>2003</anio>
        <genero>Novela</genero>
        <subgeneros>
            <subgenero>Fantástica</subgenero>
            <subgenero>Romántica</subgenero>
            <subgenero>Ciencia Ficción</subgenero>
        </subgeneros>
        <tituloOriginal>
            The Time Traveler's Wife
        </tituloOriginal>
    </libro>
    <libro>…
```

XSLT – Ejemplo – Transformación

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
    <xsl:output method="html" omit-xml-declaration="yes" media-type="text/html" />
    <xsl:template match="/">
        <xsl:text disable-output-escaping='yes'>&lt;!DOCTYPE html&gt;</xsl:text>
        <html lang="es">
            <head>
                <meta charset="UTF-8" />
```

```
  <xsl:for-each select="biblioteca/libro">
      <article>
          <h2>
              <xsl:value-of select="titulo" />
          </h2>
```

Enlazar XML con XSLT

Se utiliza una instrucción de procesamiento XML: xml-stylesheet

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet href="biblioteca.xslt" type="text/xsl"?>
<biblioteca>
    <libro>
```

En los navegadores abrir directamente el XML no suele funcionar por restricciones de seguridad.

A veces sirve instalar un servidor web ligero (Live Server Extension en VS Code) o usar otra aplicación.

Transformación de XML a HTML

Declaración XSLT

XSLT es XML. Debe cumplir con las normas XML:

- Bien formado
- Válido (según normas de XSLT)

Tras la declaración XML debe aparecer la declaración de la XSLT, que se puede hacer de dos formas, completamente equivalentes, se puede usar cualquiera de las dos:

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

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

Elemento xsl:output

Elemento de nivel superior.

Sólo puede ser hijo directo de xsl:stylesheet o de xsl:transform.

Define tipo de salida producida.

Atributos importantes:

- method: xml, html, text, name. Por defecto es XML.
- omit-xml-declaration: en el documento generado no se incluirá la declaración XML.
- indent: determina si la salida debe o no indentarse.
- media-type: define el tipo MIME de la salida. Por defecto es text/xml

Elemento xsl:template

Permite crear plantillas.

El atributo “match” especifica a qué elemento(s) del XML se aplicará la plantilla. El valor de este atributo es una expresión XPath.

Por ejemplo, en:

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

"/" se refiere a la raíz del documento.

Dentro de la plantilla aparecerán otros elementos XSLT con atributos “match”, “select” o “test”, que también son XPath, para ir especificando qué transformación debe hacerse con cada elemento del XML original.

XSLT sin plantillas o con plantillas vacías

Si no hay plantillas, el procesador XSLT extrae el texto de todos los nodos (no atributos).

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:transform version="1.0"
               xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
    <xsl:output method="html" omit-xml-declaration="yes" media-type="text/html" />
</xsl:transform>
```

Produce:

La mujer del viajero en el tiempo Audry Niffeneger 2003 Novela Fantástica Romántica Ciencia Ficción The Time Traveler's Wife El aprendiz de guerrero Lois McMaster Bujold 1986 Novela Ficción especulativa Ciencia Ficción The warrior's Apprentice El Juego de Ender Orson Scott Card 1985 Novela Ciencia Ficción Ender's Game

Si hay plantilla, pero está vacía, no se produce resultado.

Elemento xsl:value-of

Permite obtener el valor de un elemento o atributo, y mostrarlo en el fichero de salida.

El atributo “select” es una expresión XPath que especifica el elemento o atributo del que queremos obtener el valor.

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

Muestra el título del libro que estemos presentando.

Las expresiones XPath en el atributo select son relativas al contexto en el que nos encontremos, y se pueden hacer absolutas (volver a comenzar desde la raíz del documento XML) si comienza por /.

También se puede usar "//" para buscar en todo el documento, sin importar jerarquías.

Elemento xsl:for-each

Permite hacer un bucle en XSLT.

El atributo “select” es una expresión XPath que especifica el grupo de elementos o atributos que queremos iterar.

Por ejemplo:

```
<xsl:for-each select="biblioteca/libro">
```

Permite recorrer todos los libros de la biblioteca.

Para cada libro, se procesan los elementos que haya entre el inicio del <xsl:for-each …> y su cierre (</xsl:for-each>).

Otra vez, el valor del select es relativo al contexto. Similar al path en un sistema de archivos.

Elemento xsl:for-each – Filtrar la salida

Se pueden usar condiciones para filtrar los resultados de un atributo select en xls:for-each y otros elementos XSLT.

La sintaxis es muy parecida a los filtros que ya hemos visto en Xpath.

Por ejemplo:

```
<xsl:for-each select="biblioteca/libro[anio=2003]">
```

Sólo iterará los libros del año 2003

Se puede usar:

- Igual y distinto: “=“, “!=“.
- Mayor que y menor que (escapado): “>”, “<”

Elemento xsl:for-each – Ordenar la salida

Se puede indicar el orden en que se presentan los elementos de un xls:for-each, añadiendo un xsl:sort dentro:

```
<xsl:for-each select="biblioteca/libro">
     <xsl:sort select="anio"></xsl:sort>
</xsl:for-each>
```

El atributo “select” en xsl:sort, indica el elemento o atributo por el que queremos ordenar.

Normalmente es relativa al elemento ordenado. En el ejemplo, se ordena por el valor del elemento "anio" de cada elemento "libro.

Elemento xsl:for-each – Ordenar la salida

Se puede personalizar el orden de distintas formas:

- Orden ascendente o desdendente (ascending, descending).

```
<xsl:sort select="anio" order="descending" />
```

- Tipo de dato del elemento / atributo por el que se ordena (number, text, qname). Por defecto "text".

```
<xsl:sort select="anio" data-type="number" />
```

- Para texto, se puede indicar primero mayúsculas o minúsculas:

```
<xsl:sort select="nombre" case-order="upper-first|lower-first"
/>
```

- Idioma usado en la comparación:

```
<xsl:sort select="nombre" lang="es" />
```

Elemento xsl:if

Sirve para tomar decisiones en función de un criterio que se especifica en el atributo “test”.

```
<xsl:if test="anio &lt; 2000">
     <p>Libro del siglo pasado.</p>
</xsl:if>
```

En este caso, si el libro está publicado antes del año 2000, se añade un párrafo a la salida HTML.

Se ha escapado el carácter para comparar (menor que), porque si se pusiera sin escapar la XSLT no sería XML bien formado.

Elemento xsl:choose

Permite elegir entre varias condiciones, y un valor por defecto.

Equivalente al “switch” de Java u otros lenguajes de programación.

Funciona junto a:

- xsl:when ("case” Java)
- xsl:otherwise (“default” Java)

```
<xsl:choose>
    <xsl:when test="anio &lt; 1950">
        <p>Primera mitad del sigo XX.</p>
    </xsl:when>
    <xsl:when test="anio &gt; 1950 and anio &lt; 2000 ">
        <p>Segunda mitad del sigo XX.</p>
    </xsl:when>
    <xsl:otherwise>
        <p>Siglo XXI</p>
    </xsl:otherwise>
</xsl:choose>
```

Elemento xsl:attribute

Permite añadir atributos a un elemento de la salida.

Ejemplo: añadir atributos “src” y “alt” al elemento “img”:

```
<img>
     <xsl:attribute name="src">
          <xsl:value-of select="imagen"/>
     </xsl:attribute>
     <xsl:attribute name="alt">
          Texto alternativo
     </xsl:attribute>
</img>
```

Alternativa a xsl:attribute – {}

Para atributos hay una notación más abreviada basada en llaves.

Ejemplo: Este atributo "src":

```
<img alt="">
     <xsl:attribute name="src">
          <xsl:value-of select="imagen"/>
     </xsl:attribute>
</img>
```

Se puede escribir más abreviadamente:

```
<img src="{imagen}" alt="">
```

Y se puede hacer interpolación (insertar en el texto):

```
<img src="{imagen}" alt="Imagen del producto {nombre}">
```

Elemento xsl:element

Permite crear un elemento XML en la salida.

Es una alternativa a escribir directamente el XML de salida.

Ejemplo: crear un elemento "p" y dentro escribir el autor del libro.

```
<xsl:element name="p">
     Autor: <xsl:value-of select="autor" />
</xsl:element>
```

El ejemplo anterior es lo mismo que esto:

```
<p>
     Autor: <xsl:value-of select="autor" />
</p>
```