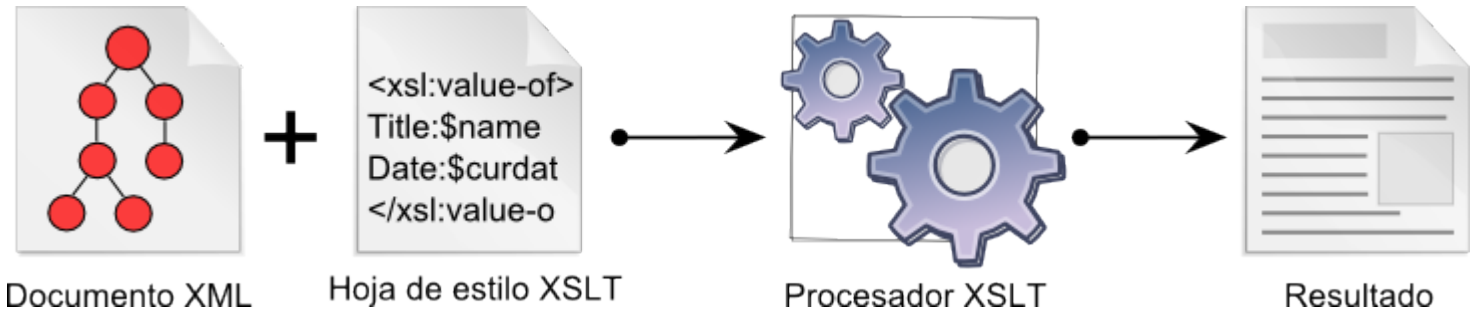
XSLT: Transformaciones XSL

El lenguaje de programación XSLT

XSLT (Transformaciones XSL) es un lenguaje de programación declarativo que permite generar documentos a partir de documentos XML, como ilustra la imagen siguiente:



- El documento XML es el documento inicial a partir del cual se va a generar el resultado.
- La hoja de estilo XSLT es el documento que contiene el código fuente del programa, es decir, las reglas de transformación que se van a aplicar al documento inicial.
- El procesador XSLT es el programa de ordenador que aplica al documento inicial las reglas de transformación incluidas en la hoja de estilo XSLT y genera el documento final.
- El resultado de la ejecución del programa es un nuevo documento (que puede ser un documento XML o no).

XSLT se utiliza para obtener a partir de un documento XML otros documentos (XML o no). A un documento XML se le pueden aplicar distintas hojas de estilo XSLT para obtener distintos resultados y una misma hoja de estilo XSLT se puede aplicar a distintos documentos XML.

Hojas de estilo XSLT

XSLT es un lenguaje declarativo. Por ello, las hojas de estilo XSLT no se escriben como una secuencia de instrucciones, sino como una colección de plantillas (template rules). Cada plantilla establece cómo se transforma un determinado elemento (definido mediante expresiones XPath). La transformación del documento se realiza de la siguiente manera:

- El procesador analiza el documento y construye el árbol del documento.
- El procesador va recorriendo todos los nodos desde el nodo raíz, aplicando a cada nodo una plantilla, sustituyendo el nodo por el resultado.
- Cuando el procesador ha recorrido todos los nodos, se ha terminado la transformación.

Una hoja de estilo XSLT es un documento XML que contiene al menos las etiquetas siguientes:

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

Estas etiquetas son:

- la declaración xml **<?xml>**, propia de cualquier documento XML.
- la instrucción **<xsl:stylesheet>** es la etiqueta raíz de la hoja de estilo, sus atributos indican la versión y el espacio de nombres correspondiente.

Dentro de la instrucción **<xsl:stylesheet>** se pueden encontrar los llamados elementos de alto nivel y las plantillas, como en el ejemplo siguiente:

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

  <xsl:output method="xml" indent="yes"/>

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

</xsl:stylesheet>
```

Estas etiquetas son

- el elemento de alto nivel **<xsl:output>** indica el tipo de salida producida.
- la instrucción **<xsl:template>** es una plantilla.
  - El atributo *match* indica los elementos afectados por la plantilla y contiene una expresión XPath.
  - El contenido de la instrucción define la transformación a aplicar (si la instrucción no contiene nada, como en el ejemplo anterior, sustituirá el nodo por nada).

Cuando se aplica una plantilla a un nodo, en principio la plantilla se aplica únicamente al nodo, pero se sustituye el nodo y todos sus descendientes por el resultado de la aplicación de la plantilla, lo que nos haría perder a los descendientes. Si se quiere que antes de sustituir el nodo y todos sus descendientes se apliquen también a los descendientes las plantillas que les correspondan, hay que utilizar la instrucción **<xsl:apply-templates />**, como en el ejemplo siguiente:

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

  <xsl:template match='/'>
    <xsl:apply-templates />
  </xsl:template>

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

</xsl:stylesheet>
```

Volver al principio de la página

Enlazar documentos XML con hojas de estilo XSLT

Se puede asociar de forma permanente una hoja de estilo XSLT a un documento XML mediante la instrucción de procesamiento **<?xml-stylesheet ?>**, la misma que permite asociar hojas de estilo CSS. La instrucción de procesamiento **<?xml-stylesheet ... ?>** va al principio del documento, después de la declaración XML.

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

Cuando se visualiza en un navegador web un documento XML enlazado con una hoja de estilo XSLT, los

navegadores muestran el resultado de la transformación, aunque si se muestra el código fuente de la página, los navegadores muestran el documento XML original.

Nota: Actualmente (abril de 2012) Google Chrome sólo muestra el documento transformado si se ha servido desde un servidor web (http://...). Si el documento se abre directamente (file://...), Google Chrome muestra un página en blanco (aunque sí puede mostrar el código fuente de la página).

Volver al principio de la página

Ejemplos de plantillas XSLT

Vamos a ver ejemplos de plantillas trabajando sobre el documento siguiente:

```
<?xml version="1.0" encoding="UTF-8"?>
<biblioteca>
  <libro>
    <titulo>La vida está en otra parte</titulo>
    <autor>Milan Kundera</autor>
    <fechaPublicacion año="1973"/>
  </libro>
  <libro>
    <titulo>Pantaleón y las visitadoras</titulo>
    <autor fechaNacimiento="28/03/1936">Mario Vargas Llosa</autor>
    <fechaPublicacion año="1973"/>
  </libro>
  <libro>
    <titulo>Conversación en la catedral</titulo>
    <autor fechaNacimiento="28/03/1936">Mario Vargas Llosa</autor>
    <fechaPublicacion año="1969"/>
  </libro>
</biblioteca>
```

Plantillas vacías o no existentes

Si no hay plantillas, el procesador simplemente extrae el texto contenido por los nodos:

Enlace a ejemplo 1-1 (xml)

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

</xsl:stylesheet>
```

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

    La vida está en otra parte
    Milan Kundera

    Pantaleón y las visitadoras
    Mario Vargas Llosa

    Conversación en la catedral
    Mario Vargas Llosa
```

Al abrir el ejemplo anterior en el navegador, el texto se muestra todo seguido ya que no hay etiquetas.

Si hay una plantilla vacía, el procesador sustituye el nodo por nada (y no extrae el texto contenido):

Enlace a ejemplo 1-2 (xml)

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

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

</xsl:stylesheet>
```

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

    La vida está en otra parte

    Pantaleón y las visitadoras

    Conversación en la catedral
```

En el ejemplo anterior, se obtienen los títulos contenidos por los nodos <titulo>, pero los de <autor> se pierden.

Si la plantilla vacía se aplica al nodo raíz, el procesador sustituye el nodo raíz y todos sus descendientes por nada:

Enlace a ejemplo 1-3 (xml)

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

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

</xsl:stylesheet>
```

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

La instrucción **<xsl:value-of>**

La instrucción **<xsl:value-of>** extrae el contenido del nodo seleccionado.

Enlace a ejemplo 2-1 (xml)

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

  <xsl:template match="libro">
     <xsl:value-of select="autor"/>
  </xsl:template>

</xsl:stylesheet>
```

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

  Milan Kundera
  Mario Vargas Llosa
  Mario Vargas Llosa
```

En el ejemplo anterior, los títulos de los libros se han perdido, porque el nodo <libro> (y sus subnodos <autor> y <titulo>) se sustituye por el contenido del nodo <autor>.

Enlace a ejemplo 2-2 (xml)

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

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

</xsl:stylesheet>
```

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

    La vida está en otra parte
    Milan Kundera

    Pantaleón y las visitadoras
    Mario Vargas Llosa

    Conversación en la catedral
    Mario Vargas Llosa
```

En el ejemplo anterior, los autores se obtienen gracias a la regla que extrae el contenido del nodo y los títulos se obtienen porque al no haber reglas para ese nodo se extrae el contenido. El resultado es el mismo que el del ejemplo 1-1, pero por motivos distintos.

También se pueden extraer los valores de los atributos, utilizando @.

Enlace a ejemplo 2-3 (xml)

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

  <xsl:template match="fechaPublicacion">
     <xsl:value-of select="@año"/>
  </xsl:template>

</xsl:stylesheet>
```

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

    La vida está en otra parte
    Milan Kundera
    1973

    Pantaleón y las visitadoras
    Mario Vargas Llosa
    1973

    Conversación en la catedral
```

```
    Mario Vargas Llosa
    1969
```

En el ejemplo anterior, las fechas de publicación se obtienen gracias a la regla que extraen el valor del atributo y los títulos y autores se obtienen porque al no haber reglas para ese nodo se extrae el contenido.

Generar texto adicional

Se puede generar texto escribiendolo en la regla, por ejemplo, código html:

Enlace a ejemplo 3-1 (xml)

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

  <xsl:template match="libro">
     <p><xsl:value-of select="autor"/></p>
  </xsl:template>

</xsl:stylesheet>
```

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

<p>Milan Kundera</p>
<p>Mario Vargas Llosa</p>
<p>Mario Vargas Llosa</p>
```

Sólo se ven los autores porque la regla selecciona el nodo <libro> como en el ejemplo, 2-1, pero además generamos las etiquetas <p>. El resultado sigue sin verse bien en el navegador, porque aunque hay etiquetas <p>, falta la etiqueta global <html>.

Dentro de la regla podemos hacer referencia a varios subnodos:

Enlace a ejemplo 3-2 (xml)

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

  <xsl:template match="libro">
     <p><xsl:value-of select="autor"/></p>
     <p><xsl:value-of select="titulo"/></p>
  </xsl:template>

</xsl:stylesheet>
```

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

<p>Milan Kundera</p>
<p>La vida está en otra parte</p>

<p>Mario Vargas Llosa</p>
<p>Pantaleón y las visitadoras</p>

<p>Mario Vargas Llosa</p>
<p>Conversación en la catedral</p>
```

Los siguientes ejemplos intentan conseguir el mismo resultado que el ejemplo anterior (3-2), pero utilizando dos reglas, y no lo consiguen:

Enlace a ejemplo 3-3 (xml)

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

  <xsl:template match="libro">
     <p><xsl:value-of select="autor"/></p>
 </xsl:template>

  <xsl:template match="libro">
     <p><xsl:value-of select="titulo"/></p>
  </xsl:template>

</xsl:stylesheet>
```

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

<p>La vida está en otra parte</p>

<p>Pantaleón y las visitadoras</p>

<p>Conversación en la catedral</p>
```

Enlace a ejemplo 3-4 (xml)

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

  <xsl:template match="libro">
     <p><xsl:value-of select="titulo"/></p>
  </xsl:template>

  <xsl:template match="libro">
     <p><xsl:value-of select="autor"/></p>
 </xsl:template>

</xsl:stylesheet>
```

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

<p>Milan Kundera</p>

<p>Mario Vargas Llosa</p>

<p>Mario Vargas Llosa</p>
```

¿Por qué en un caso se obtienen sólo los títulos y en el otro sólo los autores? Porque el procesador XSLT sólo aplica una regla a cada nodo. Si tenemos dos reglas para el mismo nodo, el procesador sólo aplica una de ellas (la última, en este caso).

Aplicar reglas a subnodos

Para generar la etiqueta *<html>* tenemos que añadir otra regla que se aplique al documento en general

Enlace a ejemplo 4-1 (xml)

```
<?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>
        <xsl:apply-templates />
    </html>
  </xsl:template>

  <xsl:template match="libro">
     <p><xsl:value-of select="autor"/></p>
  </xsl:template>

</xsl:stylesheet>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<html>
  <p>Milan Kundera</p>
  <p>Mario Vargas Llosa</p>
  <p>Mario Vargas Llosa</p>
</html>
```

Ahora el resultado ya se ve correctamente en el navegador como párrafos.

Volver al principio de la página

La instrucción **<xsl:text>**

La instrucción **<xsl:text>** permite generar texto que no puede generarse simplemente añadiéndolo como en los ejemplos anteriores. Por ejemplo, para generar un salto de línea en el resultado añadiendo la entidad de carácter
:

Enlace a ejemplo 5-1 (xml)

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

  <xsl:template match="/">
    <p>Prueba</p>
    <xsl:text>&#10;</xsl:text>
    <p>Prueba</p>
  </xsl:template>

</xsl:stylesheet>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<p>Prueba</p>


<p>Prueba</p>
```

Volver al principio de la página

La instrucción **<xsl:attribute>**

La instrucción **<xsl:attribute>** permite generar un atributo y su valor. Se utiliza cuando el valor del atributo se obtiene a su vez de algún nodo.

Por ejemplo, a partir del siguiente documento xml, se quiere generar la etiqueta *<img>*. en la que el valor del atributo *src* sea el contenido de la etiqueta <imagen>.

```
<?xml version="1.0" encoding="UTF-8"?>
<licencias>
  <licencia>
    <nombre>Creative Commons By - Share Alike</nombre>
    <imagen>cc_bysa_88x31.png</imagen>
  </licencia>
</licencias>
```

No se puede utilizar la instrucción **<xsl:value-of>** como en el ejemplo incorrecto siguiente:

Enlace a ejemplo 6-1 (xml)

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

  <xsl:template match="licencia">
    <p><img src="<xsl:value-of select="imagen"
/>" /></p>
  </xsl:template>

</xsl:stylesheet>
```

```
Error at line 5, column 19: not
well-formed (invalid  token)
```

En este caso el problema no es debido a la utilización de comillas dobles (también daría error si se hubieran utilizado comillas simples o entidades), sino que es necesario utilizar la instrucción **<xsl:attribute>** como en el ejemplo siguiente:

Enlace a ejemplo 6-2 (xml)

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

  <xsl:template match="licencia">
    <p><img>
    <xsl:attribute name="src">
      <xsl:value-of select="imagen" />
    </xsl:attribute>
    </img>
    </p>
  </xsl:template>

</xsl:stylesheet>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<p>
  <img src="cc_bysa_88x31.png"/>
</p>
```

En la hoja de estilo XSLT, la etiqueta <img> se escribe con apertura y cierre, aunque en el resultado aparece como etiqueta monoatómica.

Como en ejemplos anteriores, para que la imagen se muestre en el navegador sería necesario generar también la etiqueta *<html>*:

Enlace a ejemplo 6-3 (xml)

```
<?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>
     <xsl:apply-templates />
```

```
<?xml version="1.0" encoding="UTF-8"?>
<html>
  <p><img src="cc_bysa_88x31.png"/></p>
</html>
```

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

  <xsl:template match="licencia">
    <p><img>
    <xsl:attribute name="src">
      <xsl:value-of select="imagen" />
    </xsl:attribute>
    </img>
    </p>
  </xsl:template>

</xsl:stylesheet>
```

Volver al principio de la página

Esta página forma parte del curso "XML: Lenguaje de Marcas Extensible" disponible en http://www.mclibre.org
Autor: Bartolomé Sintes Marco
Última modificación: 3 de abril de 2012