Unidad de Trabajo 6:
XSLT

XSLT

Versión 1.0: http://www.w3.org/TR/xslt
Versión 2.0: http://www.w3.org/TR/xslt20/

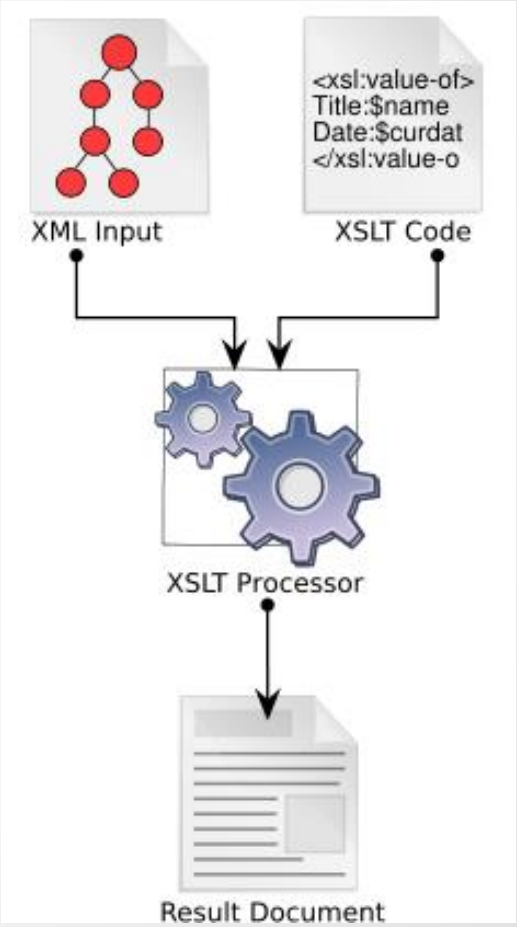
Aplicación XML para especificar reglas mediante las cuales se transforma un documento XML en otro documento XML.

Un documento XSLT es una hoja de estilos que contiene reglas de plantillas.

Cada regla de plantilla tiene un patrón y una plantilla.

Un procesador XSLT compara los elementos y otros nodos en un documento XML de entrada con los patrones de las reglas de plantilla en una hoja de estilos.

Cuando uno coincide, escribe la plantilla de dicha regla en el árbol de salida.



```
<?xml version="1.0" encoding="UTF-8"?>
<alumnos>
        <alumno>
                <nombre>Francisco</nombre>
                <apellido>Rodríguez</apellido>
                <direccion>
                        <poblacion>Benavente</poblacion>
                        <cp>49600</cp>
                </direccion>
        </alumno>
        <alumno>
                <nombre>María</nombre>
                <apellido>Pérez</apellido>
                <direccion>
                        <poblacion>Valladolid</poblacion>
                        <cp>47010</cp>
                </direccion>
        </alumno>
</alumnos>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="2.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
…
</xsl:stylesheet>
```

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

        Francisco
        Rodríguez

                Benavente
                49600

        María
        Pérez

                Valladolid
                47010
```

Todos los elementos estándar de XSLT se encuentran en el espacio de nombre http://www.w3.org/1999/XSL/Transform.

Este espacio de nombre se asigna normalmente al prefijo xsl. Para ello se habrá de utilizar una declaración xmlns:xsl.

El elemento raíz es **xsl:stylesheet** o **xsl:transform.**

El elemento raíz tendrá el atributo **version** y la declaración **xmlns:xsl.**

El documento tendrá extensión **xsl**.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="2.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
...
</xsl:stylesheet>
```

Un procesador XSLT lee una hoja de estilos XSLT, lee un documento XML y crea un documento de salida aplicando las instrucciones de la hoja de estilos a la información del documento de entrada.

Un procesador XSLT puede integrarse en un explorador Web.

Los documentos XML que se van a servir directamente a los exploradores Web pueden tener una instrucción de procesamiento xml-stylesheet en el prólogo indicando al explorador dónde buscar la hoja de estilos asociada al documento.

Tipos: V1 (application/xml o text/xsl), V2 (application/xslt+xml).

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

XSLT

Para controlar qué salida se crea desde qué entrada añadimos reglas de plantilla a la hoja de estilos XSLT.

Cada regla de plantilla se representa mediante un elemento **xsl:template**.

Este elemento tiene un atributo **match** que contiene un patrón (expresión XPath) identificando la entrada con la que coincide; también contiene una plantilla que sirve como salida cuando coincide con el patrón.

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

<xsl:template match=”alumno”>Alumno</xsl:template>

</xsl:stylesheet>
```

En el documento de entrada existen dos elementos alumno.
El procesador reemplaza todo lo que se encuentra dentro del elemento alumno por el contenido de la regla de plantilla, que se denomina plantilla.

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

También podemos añadir etiquetas.

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

<xsl:template match=”alumno”><p>Alumno</p></xsl:template>

</xsl:stylesheet>
```

XSLT puede seleccionar un determinado contenido del documento de entrada e insertarlo en el documento de salida.

Para ello podemos utilizar el elemento **xsl:value-of**.

Este elemento calcula el valor de una determinada cadena de una expresión XPath y lo inserta en la salida.

El elemento cuyo valor se recoge se identifica mediante un atributo **select** que contiene una expresión XPath.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match=”alumno”>
        <p><xsl:value-of  select=”nombre” /></p>
</xsl:template>
</xsl:stylesheet>
```

Un procesador XSLT lee el documento XML de entrada de arriba abajo, empezando por la raíz del documento.

Las reglas de plantilla se activan en el orden en el que se encuentra la coincidencia de elementos.

Una plantilla puede especificar qué elemento o elementos deben procesarse a continuación.

El elemento **xsl:apply-template** consigue ordenar el procesamiento.

Su atributo **select** contiene una expresión XPath que le indica al procesador XSLT qué nodos debe procesar y el punto del árbol de salida en el que debe hacerlo.

El orden de las reglas de plantilla en la hoja de estilos no es importante. Sólo importa el orden de los elementos en el documento de entrada.

alumnos.xml

1

```
<?xml version="1.0" encoding="UTF-8"?>
<alumnos>
	<alumno>
		<nombre>Francisco</nombre>
		<apellido>Rodríguez</apellido>
		<direccion>
			<poblacion>Benavente</poblacion>
			<cp>49600</cp>
		</direccion>
	</alumno>
	<alumno>
		<nombre>María</nombre>
		<apellido>Pérez</apellido>
		<direccion>
			<poblacion>Valladolid</poblacion>
			<cp>47010</cp>
		</direccion>
	</alumno>
</alumnos>
```

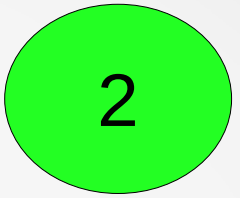
alumnos.html

```
<?xml version="1.0" encoding="UTF-8"?>
<html>
        <head>
                <title>Alumnos</title>
        </head>
        <body>
                <p>FranciscoRodríguez</p>
                <p>MaríaPérez</p>
        </body>
</html>
```

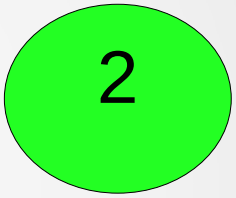
```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform" >

<xsl:template match="alumnos">
<html>
  <head><title>Alumnos</title></head>
  <body>
    <xsl:apply-templates />
  </body>
 </html>
</xsl:template>
<xsl:template match="alumno">
            <p>
                        <xsl:value-of select="nombre"/>
                        <xsl:value-of select="apellido"/>
            </p>
</xsl:template>
</xsl:stylesheet>
```

alumnos.html

```
<?xml version="1.0" encoding="UTF-8"?>
<html>
        <head>
                <title>Alumnos</title>
        </head>
        <body>
                <p>FranciscoRodríguez
                49600
                Benavente</p>
                <p>MaríaPérez
                47010
                Valladolid</p>
        </body>
</html>
```



```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0"
   xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
        <xsl:template match="alumnos">
                <html><head><title>Alumnos</title></head>
                        <body>
                                <xsl:apply-templates />
                        </body>
                </html>
        </xsl:template>
  <xsl:template match="alumno">
                <p>
                        <xsl:value-of select="nombre" />
                        <xsl:value-of select="apellido" />
                        <xsl:value-of select="direccion/cp" />
                        <xsl:value-of select="direccion/poblacion" />
                </p>
        </xsl:template>
</xsl:stylesheet>
```



XSLT proporciona una regla de plantilla integrada para cada uno de los siete tipos de nodos de XPath, indicando lo que hacer con el nodo si no se proporcionan instrucciones específicas.

Regla de plantilla predeterminada para nodos de elemento y raíz.

Garantiza que los hijos se procesan.

La regla se aplica a todos los nodos excepto a los nodos de atributo y de espacio de nombre (porque no son hijos).

```
<xsl:template match="* | /" >
    <xsl:apply-templates />
</xsl:template>
```

Regla de plantilla predeterminada para los nodos de texto y atributos.

Copia el valor de los nodos de texto y atributos en el documento de salida.

Para los atributos se copia el valor del atributo pero no el nombre.

```
<xsl:template match="text() | @*" />
    <xsl:value-of select="." />
</xsl:template>
```

Regla de plantilla predeterminada para nodos de comentarios e instrucciones de procesamiento.

No proporciona ninguna salida en el árbol de resultados.

<xsl:template match="processing-instruction() | comment()" />

Regla de plantilla predeterminada para nodos de espacios de nombre.

No copia nada del nodo del espacio de nombre en la salida.

Algunas veces, el mismo contenido de entrada necesita aparecer múltiples veces en el documento de salida, formateado de acuerdo a una plantilla diferente cada vez.

Tanto los elementos **xsl:apply-templates** como **xsl:template** pueden tener atributos **mode** opcionales que conecten diferentes reglas de plantillas con distintas posiciones.

1. Rodriguez, Francisco
2. Pérez, Maria

Francisco Rodriguez 49600 Benavente

Maria Pérez 47010 Valladolid

```
<?xml version="1.0" encoding="UTF-8"?>
<html>
        <head>
                <title>Alumnos</title>
        </head>
        <body>
        <ol>
                <li>Rodríguez,Francisco </li>
                <li>Pérez,María </li>
        </ol>
<p>
        Francisco
        Rodríguez
        49600
        Benavente
</p>
<p>
        María
        Pérez
        47010
        Valladolid
</p>
</body>
</html>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
<xsl:template match="alumnos">
        <html><head><title>Alumnos</title></head>
        <body>
                <ol>
                <xsl:apply-templates select="alumno" mode="lista"/>
                </ol>
                <xsl:apply-templates select="alumno"/>
        </body>
        </html>
</xsl:template>
<xsl:template match="alumno" mode="lista">
        <li>
                <xsl:value-of select="apellido"/>,
                <xsl:value-of select="nombre"/>
         </li>
        </xsl:template>
        <xsl:template match="alumno">
                <p><xsl:apply-templates/></p>
        </xsl:template>
</xsl:stylesheet>
```

<xsl:for-each select=”expresión”> </xsl:for-each>

Itera por los nodos identificados por su atributo **select** y aplica plantillas a cada uno.

El atributo **select** contendrá una expresión de conjunto de nodos de XPath que identifica los nodos sobre los que tiene que iterar.

Los nodos seleccionados se procesan en el orden en el que aparecen en el documento.

Los nodos se pueden ordenar usando elementos hijos **xsl:sort**. El primero de estos elementos es la clave de ordenación principal, el segundo es la clave de ordenación secundaria y así sucesivamente.

También tiene que contener una plantilla que se instancia una vez para cada miembro del conjunto de nodos devuelto por la expresión del conjunto de nodos en el atributo **select**.

Carrera	Créditos
I.T. Informática	250
Dipl. Empresariales	275
Dipl. Relaciones Laborales	280
Lic. Quimica	175
Lic. Biologia	175
Lic. Humanidades	475

```
<?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>
  <head><title>Carreras</title></head>
  <body>
  <table border="1">
  <tr><td>Carrera</td><td>Créditos</td></tr>
  <xsl:for-each select="//carreras/carrera">
  <tr>
        <td><xsl:value-of select="nombre"/></td>
        <td><xsl:value-of select="creditos"/></td>
    </tr>
  </xsl:for-each>
  </table>
  </body>
  </html>
  </xsl:template>
</xsl:stylesheet>
```

<xsl:sort />

Aparece como hijo de **xsl:apply-templates** o **xsl:for-each**.

Cambia el orden el que se aplican las plantillas a la lista de nodos de contexto.

El atributo opcional **select** especifica la clave por la que se va a realizar la ordenación. Si se omite, la clave de orden se establece en el valor del nodo actual.

El atributo opcional **data-type** podrá contener los valores “text” o “number”. Por defecto la ordenación es alfabética.

El atributo opcional **order** podrá contener los valores “descending” o “ascending”. Por defecto la ordenación es ascendente.

El atributo opcional **case-order** podrá contener los valores “upper-first” o “lower-first”.

Carrera	Créditos
Lic. Humanidades	475
Dipl. Relaciones Laborales	280
Dipl. Empresariales	275
I.T. Informática	250
Lic. Quimica	175
Lic. Biologia	175

```
<?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>
  <head><title>Carreras</title></head>
  <body>
  <table border="1">
  <tr><td>Carrera</td><td>Créditos</td></tr>
  <xsl:for-each select="//carreras/carrera">
  <xsl:sort select="creditos" data-type="number" order="descending"/>
  <tr>
        <td><xsl:value-of select="nombre"/></td>
        <td><xsl:value-of select="creditos"/></td>
   </tr>
  </xsl:for-each>
  </table>
  </body>
  </html>
  </xsl:template>
</xsl:stylesheet>
```

Para filtrar elementos utilizaremos en el elemento xsl:for-each sintaxis XPath.

Los operadores de filtrado son:
= (igual) != (distinto) < menor que > mayor que

Carrera	Créditos
Lic. Humanidades	475
Dipl. Relaciones Laborales	280
Dipl. Empresariales	275

```
<?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>
  <head><title>Carreras</title></head>
  <body>
  <table border="1">
  <tr><td>Carrera</td><td>Créditos</td></tr>
  <xsl:for-each select="//carreras/carrera[creditos&gt;250]">
  <xsl:sort select="creditos" data-type="number" order="descending"/>
  <tr>
        <td><xsl:value-of select="nombre"/></td>
        <td><xsl:value-of select="creditos"/></td>
        </tr>
  </xsl:for-each>
  </table>
  </body>
  </html>
  </xsl:template>
</xsl:stylesheet>
```

<xsl:if test=”expresión booleana” > </xsl:if>

Contiene una plantilla que se instancia sólo si la expresión XPath contenida en el atributo **test** es true.

Carrera	Créditos
I.T. Informática	250
Lic. Quimica	175
Lic. Biologia	175

```
<?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>
  <head><title>Carreras</title></head>
  <body>
  <table border="1">
  <tr><td>Carrera</td><td>Créditos</td></tr>
  <xsl:for-each select="//carreras/carrera">
  <xsl:sort select="creditos" data-type="number" order="descending"/>
  <xsl:if test="creditos&lt;=250">
  <tr>
        <td><xsl:value-of select="nombre"/></td>
        <td><xsl:value-of select="creditos"/></td>
        </tr>
        </xsl:if>
  </xsl:for-each>
  </table>
  </body>
  </html>
  </xsl:template>
</xsl:stylesheet>
```

<xsl:choose > </xsl:choose>

Selecciona una o ninguna secuencia de las alternativas.

Contiene uno o más elementos **xsl:when**, cada uno de los cuales con una condición de prueba.

El contenido del primer hijo **xsl:when** cuya condición de prueba es verdadera es la salida.

Puede tener un elemento **xsl:otherwise** opcional cuyo contenido es la salida sólo si ninguna de las condiciones de prueba de ningún elemento **xsl:when** es verdadera.

Si no tiene un elemento **xsl:otherwise** y ninguna de las condiciones de prueba de ningún elemento hijo **xsl:when** es verdadera, este elemento no produce ninguna salida.

<xsl:when test=”expresión booleana” > </xsl:when>

Solamente aparece como hijo de un elemento **xsl:choose**.

El atributo **test** es requerido. Será una expresión XPath.

El contenido de **xsl:when** se inserta en el árbol de resultados sólo si es el primer elemento **xsl:when** en el elemento **xsl:choose** cuyo atributo test resulta verdadero.

La plantilla se crea e inserta en el árbol de resultados si el atributo **test** es true.

<xsl:otherwise > </xsl:otherwise>

Sólo aparece como último elemento hijo de un elemento **xsl:choose**.

Sirve como resultado predeterminado si ningún elemento **xsl:when** se instancia en el mismo elemento **xsl:choose**.

Carrera	Créditos
Dipl. Empresariales	**275**
Dipl. Relaciones Laborales	**280**
I.T. Informática	250
Lic. Biologia	175
Lic. Humanidades	**475**
Lic. Quimica	175

```
<?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>
                <head><title>Carreras</title></head>
                <body>
                  <table border="1">
                  <tr><td>Carrera</td><td>Créditos</td></tr>
                  <xsl:for-each select="//carreras/carrera">
                  <xsl:sort select="nombre"/>
                  <tr>
                   <xsl:choose>
                         <xsl:when test="creditos&lt;=250">
                            <td><xsl:value-of select="nombre"/></td>
                            <td><xsl:value-of select="creditos"/></td>
                          </xsl:when>
                          <xsl:otherwise>
                            <td><strong><xsl:value-of select="nombre"/></strong></td>
                            <td><strong><xsl:value-of select="creditos"/></strong></td>
                         </xsl:otherwise>
                     </xsl:choose>
                </tr>
                </xsl:for-each>
                </table>
              </body>
        </html>
  </xsl:template>
</xsl:stylesheet>
```

Elementos XSLT

Elementos raíz

xsl:stylesheet
xsl:transform

Elementos de nivel superior (hijos inmediatos de la raíz)

xsl:template
xsl:output
xsl:import
xsl:include
xsl:variable
xsl:param
xsl:attribute-set
xsl:key
xsl:decimal-format
xsl:preserve-space
xsl:strip-space
xsl:namespace-alias

Elementos de instrucción

xsl:apply-templates
xsl:value-of
xsl:for-each
xsl:if
xsl:choose
xsl:when
xsl:apply-imports
xsl:otherwise
xsl:sort
xsl:text
xsl:number
xsl:copy
xsl:copy-of
xsl:message
xsl:element
xsl:attribute
xsl:with-param
xsl:variable
xsl:call-template
xsl:fallback
xsl:processing-instruction

<xsl:output> </xsl:output>

Define el formato del documento de salida.

Elemento hijo de **xsl:stylesheet**.

Atributos opcionales:

method		xml, html, text
version		usado con xml y html
encoding		
omit-xml-declaration	yes, no	
doctype-public		
doctype-system		
indent		yes, no

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
 <xsl:output method="html" />
 <xsl:template match="/">
        <html>
            <head>
                <title>Carreras</title>
             </head>
            <body>
                <xsl:apply-templates select="//carreras/carrera"/>
            </body>
         </html>
 </xsl:template>
 <xsl:template match="carrera">
        <p><xsl:value-of select="nombre"/></p>
 </xsl:template>
</xsl:stylesheet>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html>
 <head>
  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
  <title>Carreras</title>
 </head>
 <body>
  <p>I.T. Informática</p>
  <p>Dipl. Empresariales</p>
  <p>Dipl. Relaciones Laborales</p>
  <p>Lic. Quimica</p>
  <p>Lic. Biología</p>
  <p>Lic. Humanidades</p>
 </body>
</html>
```

<xsl:import href="URI" />

Debe ser hijo de **xsl:stylesheet** y aparecer en primer lugar.

Importa la hoja de estilos que se encuentra en el URI.

En caso de conflicto entre plantillas importadas y plantillas del documento, las importadas tienen menor prioridad.

<xsl:include href="URI" />

Debe ser hijo de xsl:stylesheet.

Copia el contenido de la hoja de estilos que se encuentra en el URI.

Las plantillas incluidas tienen la misma prioridad que las del documento.

<xsl:variable>

Se utiliza para declarar una variable global (si es declarada como elemento de nivel superior) o local (si es declarada dentro de una plantilla).

Una vez establecido el valor de la variable, no se puede cambiar.

Se puede establecer el valor de la variable mediante el contenido o mediante el atributo opcional **select**.

El atributo obligatorio **name** tiene como valor el nombre de la variable.

El atributo opcional **select** tendrá como valor el de la variable. Será una expresión XPath.

Para utilizar el contenido de la variable, deberemos anteponer al nombre de la misma el operador $.

<xsl:param>

Se utiliza para declarar un parámetro global (si es declarado como elemento de nivel superior) o local (si es declarado dentro de una plantilla).

Se puede establecer el valor del parámetro mediante el contenido o mediante el atributo opcional **select**.

El atributo obligatorio **name** tiene como valor el nombre del parámetro.

El atributo opcional **select** tendrá como valor el del parámetro. Será una expresión XPath.

Para utilizar el contenido del parámetro, deberemos anteponer al nombre del mismo el operador $.

<xsl:param>

Como elemento de nivel superior puede establecerse un valor desde el exte

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
 <xsl:param name="numCre"/>
 <xsl:template match="/">
 <html>
 <head><title>Carreras</title></head>
 <body>
 <table border="1">
 <tr><td>Carrera</td><td>Créditos</td></tr>
 <xsl:for-each select="//carreras/carrera">
 <xsl:sort select="creditos" data-type="number" order="descending"/>
 <xsl:if test="creditos&lt;=$numCre">
 <tr>
      <td><xsl:value-of select="nombre"/></td>
      <td><xsl:value-of select="creditos"/></td>
      </tr>
      </xsl:if>
 </xsl:for-each>
 </table>
 </body>
 </html>
 </xsl:template>
</xsl:stylesheet>
```

<xsl:param>

Como elemento de instrucción puede recibir valores si **xsl:apply-templates** o **xsl:call-template** usan **xsl:with-param** cuando se llama a la plantilla.

```
<?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>
 <head>
 <title>Carreras</title>
 </head>
 <body>
 <xsl:apply-templates select="//carreras/carrera">
 <xsl:with-param name="numCre" select="200"/>
 </xsl:apply-templates>
 </body>
 </html>
 </xsl:template>

 <xsl:template match="carrera">
 <xsl:param name="numCre"/>
 <xsl:if test="creditos&lt;$numCre">
            <p><xsl:value-of select="nombre"/></p>
            </xsl:if>
 </xsl:template>
</xsl:stylesheet>
```

<xsl:with-param>

Pasa un parámetro de nombre a una plantilla que lo espera.

Puede ser hijo de **xsl:apply-templates** o de **xsl:call-template**.

Un elemento **xsl:template** recibe el parámetro a través de un elemento **xsl:param** con el mismo nombre.

Si una plantilla espera recibir un parámetro determinado y no lo obtiene, puede recoger en su lugar el valor predeterminado del elemento **xsl:param**.

El atributo **name** es obligatorio e indica el nombre del parámetro.

Se puede establecer el valor del parámetro mediante el contenido o mediante el atributo opcional **select**.

El atributo opcional **select** tendrá como valor el del parámetro. Será una expresión XPath.

<xsl:call-template>

Llama a una plantilla por su nombre. El elemento **xsl:template** al que llama debe tener atributo **name**.

El nodo actual y la lista de nodos de contexto son iguales para la plantilla llamada y para la plantilla que llama.

Puede incluir varios elementos **xsl:with-param**.

El atributo **name** es obligatorio e indica el nombre de la plantilla.

```
<?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>
 <head>
 <title>Carreras</title>
 </head>
 <body>
 <xsl:for-each select="//carreras/carrera">
 <xsl:call-template name="carrera">
 <xsl:with-param name="numCre" select="200"/>
 </xsl:call-template>
 </xsl:for-each>
 </body>
 </html>
 </xsl:template>

 <xsl:template name="carrera">
 <xsl:param name="numCre"/>
 <xsl:if test="creditos&lt;$numCre">
            <p><xsl:value-of select="nombre"/></p>
            </xsl:if>
 </xsl:template>

</xsl:stylesheet>
```

<xsl:element> </xsl:element>

Inserta un elemento en el árbol de resultados.

El nombre del elemento lo proporciona el atributo **name**.

El URI del espacio de nombre del elemento, si existe, lo proporciona el atributo opcional **namespace**.

Los atributos se pueden añadir a través de los hijos **xsl:attribute** o mediante una lista de nombres de elementos **xsl:attribute-set** separados por espacios en blanco como valor del atributo **use-attribute-sets**.

El contenido del elemento es una plantilla.

<xsl:copy-of select=”expresión XPath” />

Copia los nodos especificados identificados por la expresión así como todos los hijos, atributos, espacios de nombre y descendientes de los nodos.

<xsl:copy > </xsl:copy>

Copia el nodo actual del documento de origen en el documento de salida. No copia los hijos o atributos del nodo.

Atributo opcional: **use-attribute-sets**. Contendrá una lista de nombres **xsl:attribute-set** separados por espacios en blanco. Los atributos se añaden al elemento copiado.

Se pueden añadir atributos a través del hijo **xsl:attribute**.

Puede contener una plantilla para especificar el contenido del elemento insertado en el árbol.

<xsl:attribute name=”nombre” /> </xsl:attribute>

Añade un atributo a un elemento en el árbol de resultados.

Puede ser un hijo del elemento **xsl:attribute-set**, una instrucción **xsl:element** o un elemento de resultado literal.

Todos los elementos **xsl:attribute** deben preceder a todos los elementos resultantes.

El atributo **name** es obligatorio.

El atributo opcional **namespace** contiene el URI del espacio de nombre.

El contenido del elemento es una plantilla.

<xsl:attribute-set name=”nombre” /> </xsl:attribute-set>

Define una colección de atributos que se pueden aplicar a elementos que se encuentran en cualquier parte de la hoja de estilos.

El atributo **name** es obligatorio y proporciona un nombre para el conjunto, mediante el cual **xsl:element** y otros elementos **xsl:attribute-set** pueden cargar este conjunto de atributos.

El atributo opcional **use-attribute-sets** añade atributos de otro conjunto de atributos diferente.

Contendrá cero o más elementos **xsl:attribute**.

<xsl:text> </xsl:text>

Se usa dentro de las plantillas para indicar que su contenido tiene que tener su salida como texto.

El atributo opcional **disable-output-escaping** con el valor **yes** indica que los caracteres especiales no son sustituidos por referencias de caracteres.

Operador { }

Fuerza la evaluación de un elemento dentro de una cadena de caracteres.

Funciones XSLT

Admite todas las funciones definidas en XPath y define 10 funciones adicionales.

current()

Devuelve un conjunto de nodos que contiene un solo nodo, el nodo actual.

Fuera de un predicado XPath, el nodo actual y el nodo de contexto son idénticos. En el predicado de un paso de localización, el nodo de contexto cambia según la ruta de localización, mientras que el nodo sigue siendo el mismo.

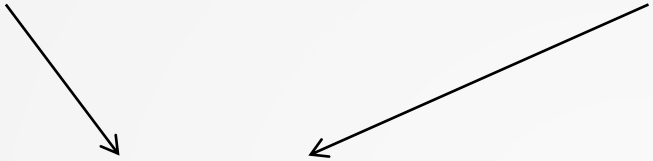
document()

Permite acceder a los nodos de un documento XML externo.

Devuelve un conjunto de nodos que contiene el nodo raíz del documento.

Recibe el URI del documento como una cadena.

```
<?xml version="1.0" encoding="UTF-8"?>
<trimestres>
<trimestre id="1">Primer trimestre</trimestre>
<trimestre id="2">Segundo trimestre</trimestre>
<trimestre id="3">Tercer trimestre</trimestre>
</trimestres>
```



```
<?xml version="1.0" encoding="UTF-8"?>
<asignaturas>
<asignatura>
<nombre>Ofimática</nombre>
<trimestre>1</trimestre>
</asignatura>
<asignatura>
<nombre>Ingeniería del Software</nombre>
<trimestre>2</trimestre>
</asignatura>
.....
</asignaturas>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<asignaturas>
<asignatura>
<nombre>Ofimática</nombre>
<trimestre>Primer trimestre</trimestre>
</asignatura>
<asignatura>
<nombre>Ingeniería del Software</nombre>
<trimestre>Segundo trimestre</trimestre>
</asignatura>
.....
</asignaturas>
```

Desarrollo de Aplicaciones web: LMSGI

XSLT

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
  <xsl:template match="universidad">
    <asignaturas>
    <xsl:apply-templates select="asignaturas/asignatura"/>
    </asignaturas>
  </xsl:template>
  <xsl:template match="asignatura">
    <xsl:copy>
      <xsl:copy-of select="nombre"/>
      <xsl:element name="trimestre">
        <xsl:value-of select="document('trimestres.xml')//trimestre[@id=current()/trimestre]"/>
      </xsl:element>
    </xsl:copy>
  </xsl:template>
</xsl:stylesheet>
```