

## Lenguaje de marcas

Tarea Tema 13 y 14 :  
Transformaciones XSLT y  
XSL-FO



## Temática

- Se ha creado un xml basado en un jardín, donde cada planta es representada por un atributo “tipo”. Tenemos diferentes características según su tipo, por ejemplo el nombre, el color, la altura, el uso y el origen

```
planta.xml X
jardin  planta  uso
1      <?xml version="1.0" encoding="UTF-8"?>
2      <?xml-stylesheet type="text/xsl" href="planta.xsl"?>
3      <jardin>
4      <planta tipo="flor">
5          <nombre>Rosa</nombre>
6          <color>Rojo</color>
7          <origen>Asia</origen>
8      </planta>
9
10     <planta tipo="flor">
11         <nombre>Tulipán</nombre>
12         <color>Amarillo</color>
13         <origen>Países Bajos</origen>
14     </planta>
15
16     <planta tipo="arbol">
17         <nombre>Roble</nombre>
18         <altura>20 metros</altura>
19         <origen>Europa</origen>
20     </planta>
21
22     <planta tipo="suculenta">
23         <nombre>Aloe Vera</nombre>
24         <uso>Medicinal</uso>
25         <origen>África</origen>
26     </planta>
27
28     <planta tipo="hierba">
29         <nombre>Menta</nombre>
30         <uso>Infusiones</uso>
31         <origen>Mediterráneo</origen>
32     </planta>
33 </jardin>
34
```

## Creación de página web mediante transformaciones XSLT

Con el xml anterior se ha creado el siguiente xsl:

- xmlns:xsl → especifica el espacio de nombres xslt
- xsl:template → define una plantilla que se le aplica al elemento raíz jardín



- empezamos a especificar el html y dentro del head vamos a aplicar los estilos que va a tener nuestra tabla una vez hayamos creado las etiquetas para la misma
- dentro de style, podemos ver que según el tipo, son de un color u otro
- en el body, creamos las etiquetas de la tabla
- `xsl:for-each select="planta"` → recorre por cada planta dentro de jardin
- `<td><xsl:value-of select="nombre"/></td>` → muestra el nombre de la planta en una celda
- para aplicar un color según el tipo de planta usamos choose:
  - `xsl:choose`, → se puede traducir como "si..."
  - `xsl:when test="tipo = 'flor'"` → cuando sea tipo flor (o árbol, etc)
  - `<td id="tipo-flor"><xsl:value-of select="@tipo"/></td>` → con el id vamos a poder darle estilo y con valueof seleccionamos el tipo

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!-- Enlazamos el xsl con el xml de plantas -->
3 <xsl:stylesheet
4   xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
5   version="1.0">
6
7   <!-- Se aplica la transformación al elemento raíz que es jardin -->
8   <xsl:template match="/jardin">
9     <html>
10      <head>
11        <!-- Estilos que le vamos a aplicar a la tabla -->
12        <style>
13          table { border: solid 2px black; box-shadow: 10px 10px 10px grey; }
14          th { color: white; background-color: #159dcc; padding: 10px; }
15          td { border: solid 2px black; padding: 8px; text-align: center; }
16          #tipo-flor { background-color: pink; }
17          #tipo-arbol { background-color: lightgreen; }
18          #tipo-suculenta { background-color: lightblue; }
19          #tipo-hierba { background-color: lightgray; }
20        </style>
21      </head>
```



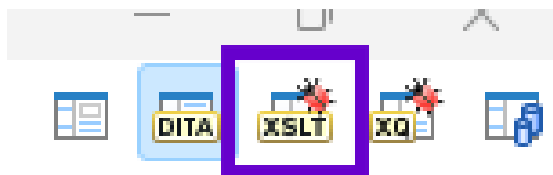
```
<body>
  <h2>Listado de Plantas</h2>

  <!-- Creamos la tabla -->
  <table>
    <!-- Encabezados de la tabla -->
    <tr>
      <th>Nombre</th>
      <th>Tipo</th>
      <th>Origen</th>
    </tr>
    <!-- Recorre cada elemento "planta" y crea una fila por cada elemento -->
    <xsl:for-each select="planta">
      <tr>
        <!-- Crea una celda con el nombre de la planta -->
        <td><xsl:value-of select="nombre"/></td>
        <!-- Asigna el color según el tipo de planta -->
        <xsl:choose>
          <xsl:when test="@tipo = 'flor'">
            <td id="tipo-flor"><xsl:value-of select="@tipo"/></td>
          </xsl:when>
          <xsl:when test="@tipo = 'arbol'">
            <td id="tipo-arbol"><xsl:value-of select="@tipo"/></td>
          </xsl:when>
          <xsl:when test="@tipo = 'suculenta'">
            <td id="tipo-suculenta"><xsl:value-of select="@tipo"/></td>
          </xsl:when>
          <xsl:otherwise>
            <td id="tipo-hierba"><xsl:value-of select="@tipo"/></td>
          </xsl:otherwise>
        </xsl:choose>
        <td><xsl:value-of select="origen"/></td>
      </tr>
    </xsl:for-each>
  </table>
</body>
</html>
</xsl:template>
</xsl:stylesheet>
```

### Depuración XML aplicando hoja de estilo XSLT

Con la herramienta Oxygen XML Editor, realizaremos la depuración paso a paso para crear un html y poder ver la tabla en una página web

- Seleccionamos de los botones ubicado en la parte superior derecha, el XSLT

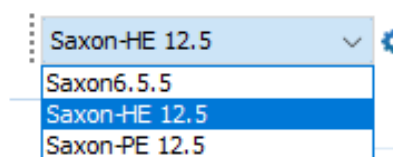


- Ubicamos el xml a la izquierda y el xsl a la derecha





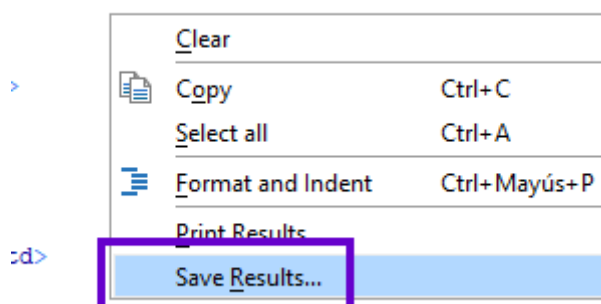
- Seleccionamos Saxon-HE 12.5



- Ahora de los controles, hacemos clic en el primer botón y vamos viendo como línea por línea según vamos haciendo clic, se va depurando



- Una vez tenemos los resultados (Output), hacemos clic derecho en los resultados y lo guardamos como .html



- Si lo abrimos con el navegador podemos ver la tabla que hemos creado:

The screenshot shows a web browser window with the address bar displaying 'planta.html' and the file path 'C:/Users/molin/Documents/GitHub/LLMM/Leng'. The page title is 'Listado de Plantas'. Below the title is a table with three columns: 'Nombre', 'Tipo', and 'Origen'. The table contains five rows of plant data, with the 'Tipo' column cells highlighted in different colors.

Nombre	Tipo	Origen
Rosa	flor	Asia
Tulipán	flor	Países Bajos
Roble	arbol	Europa
Aloe Vera	suculenta	África
Menta	hierba	Mediterráneo

Creación pdf mediante transformaciones XSL-FO:

Modificaremos el xsl que hemos creado anteriormente con etiquetas fo para poder transformarlo a pdf

```
planta.xml x | planta.xml x | planta.xsl x
xslstylesheet
1 <?xml version="1.0" encoding="UTF-8"?>
2 <xsl:stylesheet version="1.0"
3   xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
4   xmlns:fo="http://www.w3.org/1999/XSL/Format">
5
6   <xsl:template match="/jardin">
7     <fo:root>
8       <fo:layout-master-set>
9         <fo:simple-page-master master-name="A4">
10           <fo:region-body margin="20mm"/>
11         </fo:simple-page-master>
12       </fo:layout-master-set>
13
14       <fo:page-sequence master-reference="A4">
15         <fo:flow flow-name="xsl-region-body">
16           <fo:block font-size="18pt" font-family="Arial" font-weight="bold"
17             text-align="center" margin-bottom="10mm">Listado de Plantas</fo:block>
18
19           <!-- Tabla -->
20           <fo:table border="2pt solid black" border-collapse="collapse"
21             width="100%">
22             <fo:table-body>
23
24               <!-- Encabezados -->
25               <fo:table-row background-color="#159dcc" color="white">
26                 <fo:table-cell border="2pt solid black" padding="6pt">
27                   <fo:block font-weight="bold" text-align="center">Nombre</fo:block>
28                 </fo:table-cell>
29                 <fo:table-cell border="2pt solid black" padding="6pt">
30                   <fo:block font-weight="bold" text-align="center">Tipo</fo:block>
31                 </fo:table-cell>
32                 <fo:table-cell border="2pt solid black" padding="6pt">
33                   <fo:block font-weight="bold" text-align="center">Origen</fo:block>
34                 </fo:table-cell>
35               </fo:table-row>
36
37               <!-- Filas de la tabla -->
38               <xsl:for-each select="planta">
39                 <fo:table-row>
40                   <fo:table-cell border="2pt solid black" padding="6pt">
41                     <fo:block text-align="center">
42                       <xsl:value-of select="nombre"/>
43                     </fo:block>
44                   </fo:table-cell>
```



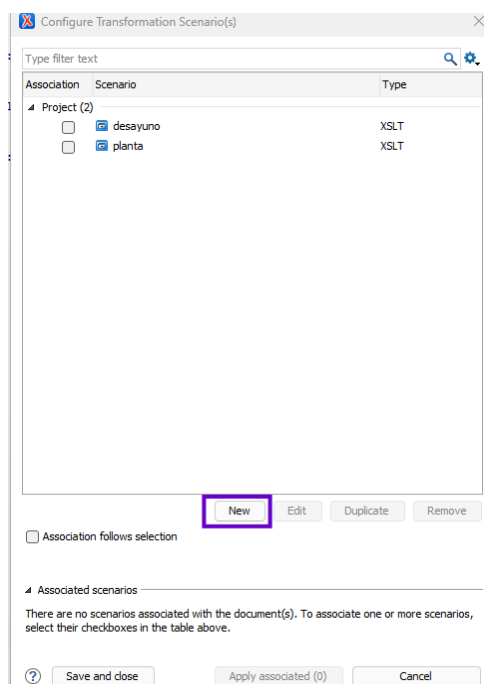
- fo:root → es el elemento raíz del documento XSL-FO
- fo:layout-master-set → define las plantillas de página
- fo:simple-page-master → crea el diseño de página con márgenes
- fo:region-body → define el área donde irá el contenido (las medidas de lo que va a ser la página de pdf)
- fo:page-sequence → es la estructura de cada página
- fo:blow → contiene el contenido de la página
- fo:block → es como un div, una “caja” que agrupa elementos
- fo:table → crea una tabla
- fo:table-body / row / cell → es para representar el cuerpo, la fila y la celda de la tabla

Una vez hecho el xsl, lo guardamos en el mismo sitio donde tenemos el xml de plantas

- Seleccionamos la siguiente herramienta

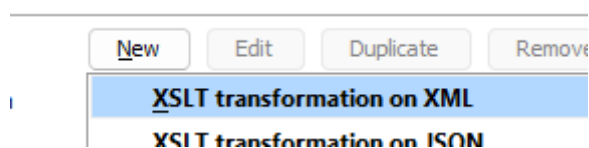


- En la siguiente ventana, hacemos clic en new

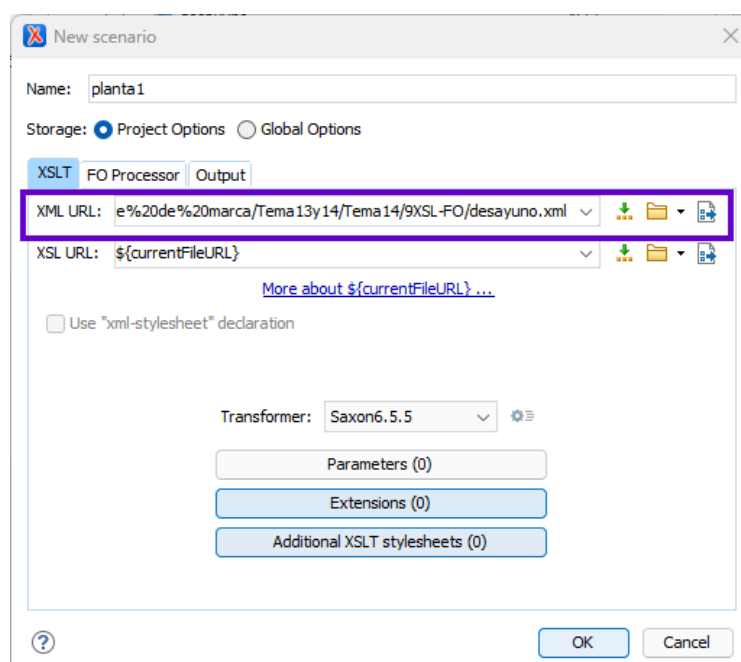




- Seleccionamos la primera opción



- En xslt, vamos a buscar el xml que usaremos



- En la pestaña de FO Processor hacemos clic en Perform FO Processing y en XSLT result as input





New scenario

Name:

Storage: ☒ Project Options ☐ Global Options

XSLT **FO Processor** Output

☒ Perform FO Processing

Input: ☒ XSLT result as input ☐ XML URL as input

Method:

Processor:

? OK Cancel

- En la siguiente pestaña de Output vamos a seleccionar Prompt for file y a Ok

New scenario

Name:

Storage: ☒ Project Options ☐ Global Options

XSLT FO Processor **Output**

Output file

☒ Prompt for file

☐ Save as

☐ Open in Browser/System Application

☒ Saved file

☐ Other location

☐ Open in Editor

Show in results view as

☐ XML

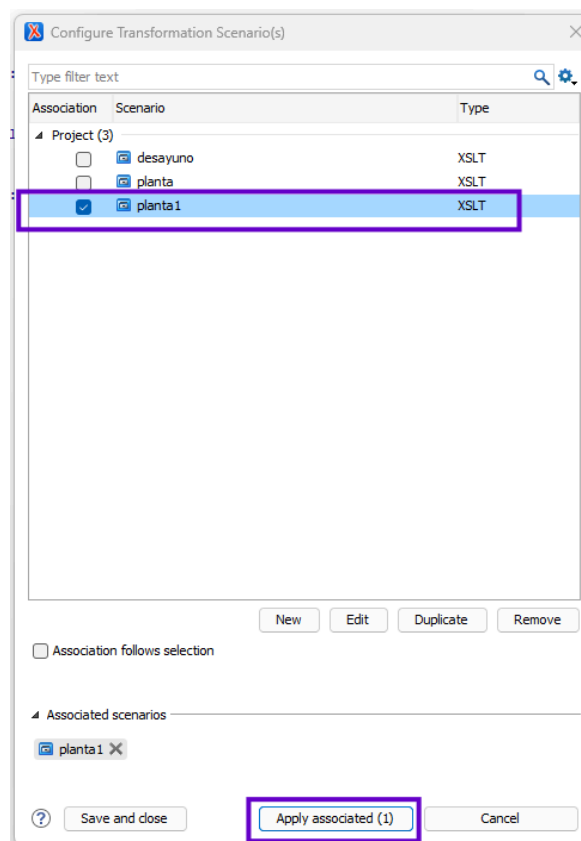
☐ SVG


☐ XHTML

Image URLs are relative to:

? OK Cancel

- Aplicamos y al guardarlo, lo haremos como pdf



Nombre	Fecha de modificación	Tipo	Tamaño
 planta.pdf	05/03/2025 18:51	Microsoft Edge P...	20 KB

