Pasos a seguir:

- Guardar este documento en nuestro equipo.
- Renombrar según el siguiente ejemplo: ApellidoNombre_ActividadesUD5
- En el transcurso de las clases, el alumnado irá completando el documento según se indique en el enunciado de las actividades.

Actividad 1.....2
Actividad 2.....3
Actividad 3.....4

Actividad	Puntuación	Puntuación obtenida
Act 1. XSL básico	**3**	
5,1	0,6	
5,2	0,6	
5,3	0,6	
5,4	0,6	
5,5	0,6	
Act 2. Lista CDs	**3**	
1	0,75	
2	0,75	
3	0,75	
4	0,75	
Act 3. Noticias	**4**	
	0,8	
	0,8	
	0,8	
	0,8	
	0,8	
Calificación		**0**

Actividad 1.

Ejercicios del PDF ‘UD5. Conversión y transformación de documentos XML’:

- Actividades 5.1 (0,6 puntos)
- Actividades 5.2 (0,6 puntos)
- Actividades 5.3 (0,6 puntos)
- Actividades 5.4 (0,6 puntos)
- Actividades 5.5 (0,6 puntos)

Actividad 2.

Crear un documento XML que almacene una lista de CD de música. Cada CD deberá almacenar la siguiente información:

- Título del álbum.
- Artista.
- Títulos de las canciones con el tiempo por canción.
- Sello discográfico.
- Año de publicación.

Al menos insertar 10 CDs.

Con el documento XML anterior, se pide:

- (0,75 puntos)Generar un fichero XSL (“cd_p1.xsl”) en el que se muestre una tabla con todos los datos de los discos de música.
- (0,75 puntos)Elegir un artista cualquiera y generar un fichero XSL (“cd_p2.xsl”) en el que se muestre una tabla con todas las canciones de ese artista.
- (0,75 puntos)Elegir un sello cualquiera y generar un fichero XSL (“cd_p3.xsl”) en el que se muestre una tabla con todas las canciones de ese sello discográfico.
- (0,75 puntos)Elegir una duración máxima de canción y generar un fichero XSL (“cd_p4.xsl”) en el que se muestre una tabla con todas las canciones que tienen una duración inferior a la elegida.

Actividad 3.

La carpeta ***scissorsandpaste-master*** contiene tres documentos principales:

- **TEISAP.XML**: el archivo XML
- **Transformers** (Transformadores): una colección de hojas de estilo XSL
- **Outputs**: archivos derivados de la base de datos utilizando las hojas de estilo XSL

También encontrarás los siguientes documentos:

- Un archivo Template (Plantilla) para los investigadores que quieran contribuir con más noticias
- Un archivo README (Léeme) con información sobre la base de datos
- Un archivo Cite (Cita), que explica cómo citar la base de datos
- Un archivo License (Licencia) con los términos de uso

La información contenida en el archivo TEISAP.XML ha sido codificada según las recomendaciones de la Text-Encoding Initiative (TEI), gran parte de la cual corresponde a los metadatos.

Ve a la carpeta Outputs y continúa hasta la carpeta XML. Dentro encontrarás un directorio llamado Simplified; copia o traslada el archivo SAPsimple_es.xml a tu escritorio.

Abre el archivo SAPsimple_es.xml con tu navegador favorito y examina su contenido. Puedes abrirlo eligiendo la opción Abrir como, arrastrando el documento al icono del navegador de tu escritorio o con un editor de texto sencillo como Notepad o TextEdit.

Dentro de cada registro hay varios elementos hijos. La Text Encoding Initiative permite anidar centenares de elementos para modelar datos de muy distinta naturaleza. Además, la gracia de XML es que puedes dar nombre a tus elementos nuevos con bastante libertad. En la base de datos Scissors and Paste los registros contienen los siguientes:

- **identificador**: número de identificación del registro
- **título**: título del periódico
- **ciudad**: ciudad del periódico

- **provincia**: provincia o región del periódico
- **país**: país del periódico
- **fecha**: fecha del artículo en formato ISO[6]
- **año**: año de la publicación
- **mes**: mes de la publicación
- **día**: día de la publicación
- **secciónPalabrasClave**: sección que contiene las palabras claves
- **palabraClave**: palabra clave que describe el artículo
- **titular**: titular del artículo (opcional)
- **texto**: sección que contiene el artículo
- **p**: párrafo de texto

Abre el archivo SAPsimple_es.xml con tu navegador favorito y examina su contenido.

Añade una línea nueva debajo de <?xml version="1.0" encoding="UTF-8"?>. En esta nueva línea, escribe

<?xml-stylesheet type="text/xsl" href="miestilo.xsl"?>

y luego guarda el archivo.

Crea un archivo nuevo y guárdalo con el nombre miestilo.xsl (o bien con el nombre que hayas escogido en el paso anterior). Antes de continuar, asegúrate de que el archivo se ha guardado en el mismo directorio que contiene el archivo XML (por ejemplo, en tu Escritorio o bien en la carpeta Simplified).

Añade las dos líneas siguientes al inicio del archivo XSL:

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

Dentro de la instrucción que acabamos de crear, escribe <xsl:value-of select="raíz"/>. No es necesario introducir una línea nueva, pero si lo haces será más fácil de leer. Te habrás dado cuenta de que no hemos incluido una etiqueta de cierre; esto se debe a que la instrucción <xsl:value-of select="raíz"/> no tiene contenido y ya está cerrada gracias a la barra lateral / situada al final.

Tras guardar el archivo, ábrelo con un navegador (Internet Explorer o Firefox) y utilízalo para transformar el archivo XML. La manera más sencilla de hacer esto es arrastrando el archivo XML (SAPsimple_es.xml) a la ventana del navegador; también es posible abrirlo mediante la pestaña Archivo/Abrir Archivo…

El resultado debería ser el texto con los saltos de línea existentes, peor sin los elementos XML

Cómo poblar un output

La línea de código <xsl:value-of select="raíz"/> imprime la base de datos entera en formato de texto sencillo. Si examinas los componentes de la línea, sabrás por qué:

- xsl:value-of (literalmente, valor-de): es una instrucción que sirve para imprimir el valor de un elemento; es decir, el texto contenido entre la etiqueta de inicio y de cierre.

- select=”raíz” (en español, selecciona=”raíz”): esta instrucción indica el elemento que contiene el valor que debería imprimirse. A menos que declares lo contrario, si apuntas hacia un elemento padre (parent) el procesador también imprimirá el valor de los elementos contenidos (children). Por tanto, al apuntar a raíz obtenemos el valor de identificador, título, etc.

Cómo imprimir valores

Si quieres imprimir el valor de un elemento en concreto, simplemente hay que sustituir “raíz” por el nombre del elemento. Por ejemplo, en nuestra hoja de estilo, reemplaza raíz por título. Guarda el archivo y refresca el navegador (normalmente, con ctrl+F5 or cmd+r) para ver los cambios.

No ha funcionado, ¿verdad? No debería porque no hemos dado al procesador todas las instrucciones que necesitaba.

Padres e hijos

El elemento título no está situado en el nivel más alto de la jerarquía, así que debemos explicarle al procesador cómo llegar hasta él. El lenguaje con que se hace esto se conoce como XPATH y funciona de una manera similar al modo en que se estructuran las rutas de un ordenador. Así pues, sustituye título por raíz/registro/título:

<xsl:value-of select="raíz/registro/título"/>
Guarda y refresca el navegador.

Ahora deberías obtener “Caledonian Mercury”, es decir, el primer título del documento XML. Sin embargo, tenemos más de 300 elementos título. ¿Qué ha ocurrido? Es muy sencillo: como no hemos especificado cuál título queríamos imprimir, el procesador ha asumido que solo nos interesaba el primero.

Realizar repeticiones con for-each

Para un ser humano quizás parezca normal querer el valor contenido en todos los elementos título, pero el procesador no sabe esto por defecto. Para remediar la situación, debemos repetir la operación en forma de bucle.

Una repetición en forma de bucle indica al procesador que debería procesar todo el archivo y llevar a cabo la transformación indicada cada vez que la condición sea satisfactoria.

Así pues, crea una nueva línea después de <xsl:template match="/"> e inserta <xsl:for-each select="raíz/registro">. Esta instrucción le indica al procesador que para cada registro situado dentro del elemento raíz debe realizar una determinada acción.

A continuación, elimina raíz/registro de la instrucción <xsl:value-of>. Es decir, debería quedar solamente título porque esta operación ya está contextualizada mediante raíz/registro. Tras <xsl:value-of>, hay que terminar la operación con la etiqueta de cierre </xsl:for-each>

En resumen, el archivo resultante debiera ser:

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

<xsl:template match="/">

   <xsl:for-each select="raíz/registro">
         <xsl:value-of select="título"/>
   </xsl:for-each>
```

</xsl:template>

</xsl:stylesheet>

Ahora la instrucción template tiene tres líneas de código:

1. Una etiqueta de inicio para la repetición for-each
2. Una instrucción para que el valor de título se imprima
3. Una etiqueta de cierre para for-each

Guarda el archivo y actualiza la ventana del navegador. Deberías obtener una masa de líneas de texto con el valor de cada uno de los elementos title. Puedes arreglar esto indicando al procesador que añada una nueva línea tras cada entrada.

Al final de la línea que contiene value-of, hay que añadir <xsl:text>
</xsl:text> para crear un salto de línea.
 es el código ISO 10646 con el que se representa un salto de línea; con el elemento <xsl:text> se declara que queremos imprimir el valor como texto sencillo.

En función del output que debamos generar, algunos caracteres especiales, el número de espacios o bien los saltos de línea a veces no se mantienen en el archivo resultante. Es por eso que se recomienda utilizar el elemento <text> para asegurarse de que el valor impreso no se ve alterado durante la transformación.

Guarda el archivo y refresca el navegador para ver los cambios. Ahora deberías ver impreso el valor de los títulos de todos los registros contenidos en el documento.

A(0,8 puntos). Genera un inventario de los registros que contenga el identificador, el título y la fecha.

B(0,8 puntos). Genera un documento que contenga el texto de todos los artículos precedido por el número de identificador entre paréntesis cuadrados.

Atributos

No todos los datos corresponden a los valores de los elementos. Algunos datos, en cambio, se almacenan como valores de los atributos de elementos. Por ejemplo, el elemento fecha tiene un atributo llamado cuándo que contiene el valor de la fecha del artículo.

<fecha cuándo="1789-01-05">

Para obtener el valor contenido en cuándo hay que hacer referencia a este atributo utilizando la expresión @cuándo.

<xsl:value-of select="fecha/@cuándo"/>

C(0,8 puntos). Crea un inventario de registros en el que se liste el título del periódico seguido de la fecha de publicación.

Cómo ordenar los resultados

El archivo XML fue escrito según la información se iba recolectando, sin organizar los registros por fecha o título. A fin de organizarlos, podemos añadir la instrucción <xsl:sort> (literalmente, ordena o clasifica) al inicio de la repetición en bucle, es decir, inmediatamente después de <xsl:for-each>. Esta instrucción tiene varios atributos opcionales que modifican cómo los datos se ordenan en el documento resultante:

select (selecciona): el nombre del elemento que sirve para ordenar los datos
order (ordena): define si los datos se ordenan de manera ascendiente (el valor del atributo debe ser ascending, en inglés) o descendiente (el valor debe ser descending, en inglés)
data-type (tipo-de-dato): informa al procesador si los datos son textuales (textual) o numéricos (number)
Por ejemplo, podemos escribir la siguiente instrucción para ordenar los datos a partir del elemento identificador en orden inverso:

<xsl:sort select="identificador" order="descending" data-type="number"/>

Es decir, a modo de aclaración, se puede ordenar los resultados utilizando un elemento que no se desea imprimir en el output.

D(0,8 puntos). Genera un documento con el texto de los artículos ordenados de más a menos recientes. Para ello, utiliza la función <xsl:sort> y trata las fechas como si fueran texto (*text*).

Cómo filtrar los resultados

Hasta el momento hemos impreso todos los registros contenidos en el documento XML. Ahora bien, si solo queremos seleccionar unos cuantos, necesitaremos filtrar los resultados mediante condiciones. Esto se consigue utilizando el elemento <xsl:if> (literalmente, si) y añadiendo la condición deseada en el atributo @test. Si el registro cumple con la condición, el procesador llevará a cabo la instrucción contenida en <xsl:if>. Si no la cumple, lo ignorará y seguirá adelante.

Así, para imprimir los identificadores de los registros de 1815 podemos escribir la siguiente plantilla

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

   <xsl:template match="/">
      <xsl:for-each select="raíz/registro">
         <xsl:if test="fecha/año='1815'">
            <xsl:value-of select="identificador"/>
            <xsl:text>&#xA;</xsl:text>
         </xsl:if>
      </xsl:for-each>
   </xsl:template>

</xsl:stylesheet>
```

Si queremos excluir el año 1815, en cambio, utilizaremos la expresión fecha/año != '1815' donde != significa que no es igual a.

E(0,8 puntos). Crea una lista de registros fechados a partir de 1789 ordenada del más reciente al más antiguo y que contenga el identificador, el título y la fecha separados por comas; cada registro debiera presentarse tras un salto de línea.

Actividad 4.

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="yes"?>
<!DOCTYPE notas>
<?xml-stylesheet type="text/xsl" href="notas2.xsl"?>
<notas>
  <alumno convocatoria="Septiembre">
    <nombre>Carlos</nombre>
    <apellidos>Amaya Arozamena</apellidos>
    <matricula>m019843</matricula>
    <cuestionarios>8.0</cuestionarios>
    <tareas>8.0</tareas>
    <examen>6.0</examen>
    <final>8.0</final>
  </alumno>
  <alumno convocatoria="Junio">
    <nombre>Jose</nombre>
    <apellidos>Merlo Soto</apellidos>
    <matricula>m019872</matricula>
    <cuestionarios>7.0</cuestionarios>
    <tareas>9.0</tareas>
    <examen>7.0</examen>
    <final>8.5</final>
  </alumno>
  <alumno convocatoria="Junio">
    <nombre>Ana</nombre>
    <apellidos>Martinez de la Fuente</apellidos>
    <matricula>m097215</matricula>
    <cuestionarios>8.0</cuestionarios>
    <tareas>9.0</tareas>
    <examen>9.0</examen>
    <final>8.5</final>
  </alumno>
  <alumno convocatoria="Septiembre">
    <nombre>Roberto</nombre>
    <apellidos>Carrera Fernandez</apellidos>
    <matricula>m059312</matricula>
    <cuestionarios>6.0</cuestionarios>
    <tareas>7.0</tareas>
    <examen>6.0</examen>
    <final>6.5</final>
  </alumno>
  <alumno convocatoria="Septiembre">
    <nombre>Concepcion</nombre>
    <apellidos>Lalinde Priego</apellidos>
    <matricula>m034093</matricula>
```

```
    <cuestionarios>4.0</cuestionarios>
    <tareas>3.0</tareas>
    <examen>2.0</examen>
    <final>3.0</final>
  </alumno>
  <alumno convocatoria="Junio">
    <nombre>Esther</nombre>
    <apellidos>Pereda</apellidos>
    <matricula>m938762</matricula>
    <cuestionarios>2.0</cuestionarios>
    <tareas>3.0</tareas>
    <examen>2.0</examen>
    <final>2.5</final>
  </alumno>
</notas>
```

Crear un documento XSLT de versión 2.0 en el que decimos que la salida será para un documento HTML, y que tendrá la siguiente estructura:

Asignamos una plantilla para los datos contenidos en el nodo principal “notas”, cuya información será la utilizada para componer el documento xhtml que estamos creando.

Dicho documento tendrá un título en el que necesitamos utilizar caracteres especiales por lo que hacemos uso de su correspondiente código hexadecimal. También utilizo varios estilos diferentes a lo largo del documento, por lo que los define utilizando css insertado en el mismo escrito (cuando son muchos estilos los que debemos de definir utilizamos un css externo, pero en esta ocasión los defino internamente ya que son pocos estilos los que hemos de crear).

Terminada la parte de la cabecera, pasamos a teclear la parte del cuerpo, y como de lo que se trata es de sacar por pantalla una tabla de contenidos, decido incluirlo en una capa para poder moverla libremente por la pantalla, y en este caso, que aparezca centrada. Dentro de dicha capa insertamos una tabla y creamos la cabecera según lo especificado para este ejercicio:

✓ Primera línea de cabecera compuesta únicamente de dos celdas (Datos y notas) por lo que hemos de unir tres columnas para cada una de las celdas. ✓ Segunda línea de cabecera compuesta por seis celdas que servirán de título para los datos representados

✓ Todos los datos que compondrán la información solicitada para el ejercicio, por lo cual insertaremos en este punto una plantilla que recoja dicha información.

✓ Cerramos la tabla, la capa, el cuerpo y el documento xhtml, ya que el resto de código será para definir la plantilla que usaremos para el cuerpo de la tabla.

La plantilla que definimos a continuación, recogerá los datos de cada alumno, y utilizamos el condicional “if” para fijarnos únicamente en aquellos alumnos que en el atributo de “convocatoria” tengan el dato de “Junio” (lo solicitado para el ejercicio).

Si se encuentran alumnos de dicha convocatoria se abre una nueva fila para la tabla y se muestran los datos de nombre, apellidos, cuestionarios, tareas y examen, y el último dato de “final” lo usamos para comprobar entre qué valores se encuentra, para lo que nos apoyamos en la orden “choose” y cuando (when) el dato de “final” sea:

✓ mayor o igual que 9 pondremos la palabra “Sobresaliente” en color azul
✓ mayor o igual que 7 pondremos la palabra “Notable” en color azul algo más claro
✓ mayor o igual que 6 pondremos la palabra “Bien” en color negro
✓ mayor o igual que 5 pondremos la palabra “Suficiente” en color naranja
✓ en cualquier otro caso pondremos la palabra “Suspenso” en color rojo

Cuando terminamos con la plantilla para los datos que hemos de mostrar creamos una plantilla vacía para el dato “matricula” con lo que conseguimos que dicho dato no aparezca.