*SEGUNDO TRIMESTRE*

# LENGUAJES DE MARCAS

PARA EL ALMACENAMIENTO Y TRANSMISIÓN DE LA INFORMACIÓN

# TIPOS DE LENGUAJES

- LENGUAJES ORIENTADOS A LA PRESENTACIÓN:
  - Este tipo de lenguaje son los usados por los procesadores de texto.
- LENGUAJES PROCEDURALES
  - En el que se mezclan dentro del lenguaje macros o pequeños programas para ser ejecutados dentro del documento. Latex, Tex, muy utilizados en la comunidad científica.
- LENGUAJES DESCRIPTIVOS:
  - Su función es describir como son los datos y la estructura de los mismos.SGML, HTML, XML…

# TIPOS DE LENGUAJES- Otra clasificación

- EXISTEN 2 TIPOS DE LENGUAJES:
  - de Marcas XML: Extended Markup Language: Es un metalenguaje extensible de etiquetas, desarrollado por el W3C, que permite definir la gramática de lenguajes específicos. Es decir, XML no es un lenguaje en particular sino la manera de definir lenguajes para diferentes necesidades.
  - de Listas JSON: JavaScript Object Notation: Es un formato ligero para el intercambio de datos.

# EVOLUCIÓN

- Los lenguajes de Marcas comenzaron a utilizarse a finales de los 60, para poder introducir anotaciones dentro de documentos electrónicos. Es en estas fechas cuando nace el  SGML, descendiente directo de GML (IBM) , pero era extenso y complejo.
- A finales de los 80, el CERN investiga otra manera de compartir información acerca de sus investigaciones, y crea el HTML, que se basaba en el SGML,y que posteriormente se popularizó muchísimo.
- A mediados de los 90, el consorcio W3, investigó acerca de dotar al HTML de una estructura semántica para darle más potencia, y crearon el XML, basado también en el antiguo SGML.

# XML

- Se puede ver como un subconjunto de SGML (Standard Generalized Markup Language), de hecho en base a XML se ha definido HTML.
- Su propósito general es el de describir datos y no mostrarlos ( como HTML).
- Juega un papel fundamental en intercambio de datos.

```
<!DOCTYPE motd [ <!EL
<motd>
<!-- created: 2003-12-12-->
 <sentence>Do not throw
 out the <keep>baby</>
 with the
 <refuse>dirty</>,
 <refuse>stinky</>,
 <refuse>bathwater</>.
 </>
 <!-- finish this later-->
</motd>
```

SGML

```
<?xml version="1.0"?>
<quiz>
 <qanda seq="1">
  <question>
   Who was the forty-second
   president of the U.S.A.?
  </question>
  <answer>
   William Jefferson Clinton
  </answer>
 </qanda>
 <!-- Note: We need to add
  more questions later.-->
</quiz>
```

XML

# ESTRUCTURA de XML

- Es un lenguaje de texto plano, en el que no existen caracteres ocultos (solo tabulador, salto de línea y espacio).
- Se utilizan las marcas <> , </> para separar y estructurar la información, y así poder almacenarla. De esta manera construimos los elementos.  Existen diferentes tipos de elementos:
  - Etiquetas, se forman con <> y definen los elementos
  - Elementos, es la información que requiere  ser almacenada
  - Atributos, proporcionan información adicional sobre un elemento en concreto.

# ESTRUCTURA de XML

- Para almacenar un valor dentro de un "campo" se utiliza la siguiente sintaxis:

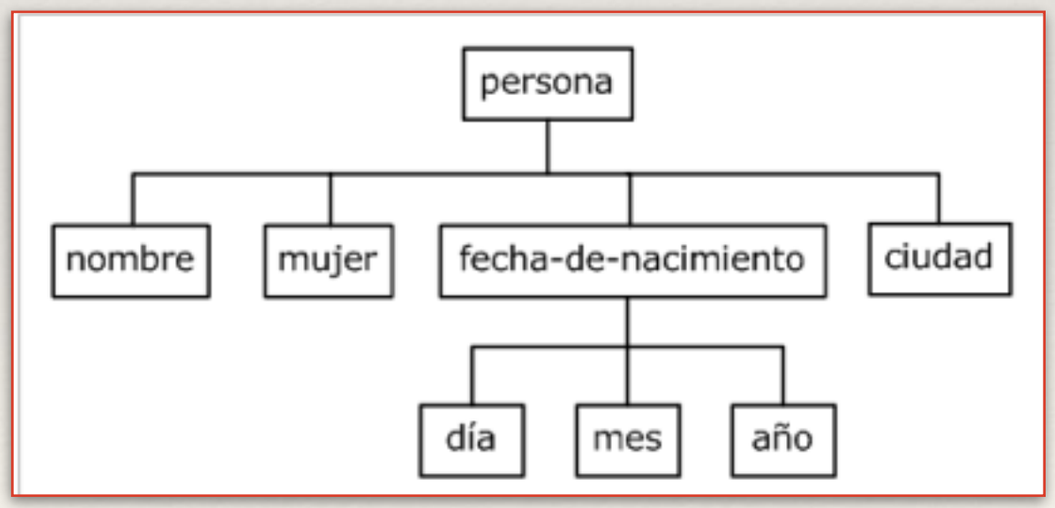
<etiqueta> valor</etiqueta>

- Puede haber elementos vacíos:

<etiqueta> </etiqueta> ó <etiqueta/>

- Relaciones padre/hijos: Todo documento XML tiene que tener un único elemento raíz del que dependan todos los demás.

```
<persona>
   <nombre>Elsa</nombre>
   <mujer/>
   <fecha-de-nacimiento>
      <día>18</día>
      <mes>6</mes>
      <año>1996</año>
   </fecha-de-nacimiento>
   <ciudad>Pamplona</ciudad>
</persona>
```




- Elementos con contenido mixto:

```
<persona>
  <nombre>Elsa</nombre> vive en <ciudad>Pamplona</ciudad>.
</persona>
```

# ESTRUCTURA de XML

- Los elementos de un documento XML pueden contener mas información , estos son los:
  - Atributos, proporcionan información adicional sobre un elemento en concreto y se especifica con nombre="xxx" en la etiqueta de inicio del elemento.

```
<alumno sexo="varon" fechaNacimiento="5/6/1990">
        <nombre>Pablo</nombre>
        <apellido>Pérez</apellido>
        <telefono tipo="movil">9155555</telefono>
        <direccion>Ronda de Segovia 111</direccion>
</alumno>
```

# ¡recomendación!

- Se observa que se pueden utilizar elementos o atributos para representar la información, pero…
- Un uso excesivo de atributos hace que el documento sea menos legible, y difícil de mantener. Y además no pueden estar estructurados en forma de árbol.
- Se recomienda simplificar y convertir algunos atributos en elementos:

```
<alumno sexo="varon" fechaNacimiento="5/6/1990">
        <nombre>Pablo</nombre>
        <apellido>Pérez</apellido>
        <telefono tipo="movil">9155555</telefono>
        <direccion>Ronda de Segovia 111</direccion>
</alumno>
```

```
<alumno id="532">
        <nombre>Pablo</nombre>
        <apellido>Pérez</apellido>
        <fechaNacimiento>
                <dia>5</dia>
                <mes>6</mes>
                <año>1990</año>
        </fechaNacimiento>
        <sexo>varón</sexo>
        <telefono tipo="movil">9155555</telefono>
        <direccion>Ronda de Segovia 111</direccion>
```

# Otros aspectos

- **Normas para  los nombres de las etiquetas**:
  - Pueden contener letras minúsculas, letras mayúsculas, números, puntos ".", guiones medios "-" y guiones bajos “_".
  - Asimismo, pueden contener el carácter dos puntos ":". No obstante, su uso se reserva para cuando se definan espacios de nombres.
  - El primer carácter tiene que ser una letra o un guion bajo “_".
  - detrás del nombre de una etiqueta se permite escribir un espacio en blanco o un salto de línea. Por ejemplo, sintácticamente es correcto escribir:

```
<ciudad >Pamplona</ciudad

>
```

  - Pero,no puede haber un salto de línea o un espacio en blanco antes del nombre de una etiqueta:

```
<

ciudad>Pamplona</ ciudad>
```

# Actividad:

Las siguientes etiquetas y elementos están mal expresados, determinar cómo se escriben correctamente:

```
<Ciudad>Pamplona</ciudad>

<día>18</dia>

<mes>6<mes/>

<ciudad>Pamplona</finciudad>

<_rojo>

<2colores>Rojo y Naranja</2colores>

< Aficiones >Cine, Bailar, Nadar</ Aficiones >

<persona><nombre>Elsa</persona></nombre>

<color favorito>azul</color favorito>
```

# Otros aspectos

- **Caracteres especiales**: existen caracteres reservados, que para poder representarlos utilizamos el carácter &palabraclave;

| Código | Carácter |
|---|---|
| &quot; | " |
| &amp; | & |
| &apos; | ' |
| &lt; | < |
| &gt; | > |

- **Instrucciones de procesamiento**: Son instrucciones especiales, que comienzan por <? ?> , la más habitual es la que especifica la versión de xml y la codificación de caracteres.

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

  - También puede incluirse la instrucción STANDALONE, que indica si el documento depende de otros:

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

  - O puede incluirse referencias a documentos CSS

```
<?xml-stylesheet type="text/css" href="estilo-animales.css"?>
```

# Otros aspectos

- **Comentarios y secciones CData**:
  - Los comentarios no son interpretados como datos, sino que van al margen de la información contenida.

```
<! comentario  - ->
```

  - Las secciones CData son secciones para que no sean interpretadas por el parser:

```
<![CDATA[ ……..]]>
```

# Espacios de nombres en XML

- De forma general los documentos XML se suelen combinar con otros documentos XML, para así poder reutilizar código. Es aquí donde pueden entrar en conflictos los nombres de los elementos.
- Para solucionar este problema de colisión de nombres se utiliza los Espacios de nombres:
  - Un espacio de nombres se define como una referencia URI (Uniform Resource Identifier), es decir, que un elemento puede tener dos partes:
    - Nombre del elemento
    - Nombre del espacio
- Para definir un espacio de nombres se utiliza la siguiente codificación:

```
xmlns:prefijo="URI"
```

  - Y se puede definir o bien en la etiqueta de inicio o bien en cada etiqueta cuando se hace referencia

# Espacios de nombres en XML: Ejemplo

Tenemos dos documentos: ejemplo1.xml y ejemplo2.xml

```
<carta>
   <palo>Corazones</palo>
   <numero>7</numero>
</carta>
```

```
<carta>
   <carnes>
      <filete_de_ternera precio="12.95"/>
      <solomillo_a_la_pimienta precio="13.60"/>
   </carnes>
   <pescados>
      <lenguado_al_horno precio="16.20"/>
      <merluza_en_salsa_verde precio="15.85"/>
   </pescados>
</carta>
```

Si uniéramos estos documentos en uno solo puede originar conflictos, para evitarlo definimos los espacios de nombres:

Ejemplo de definición de espacio de nombres en la etiqueta raíz

```
<?xml version="1.0" encoding="UTF-8"?>
<e1:ejemplo xmlns:e1="file:///Users/mfhoyuela/Desktop/ejemplo1.xml"
  xmlns:e2=""file:///Users/mfhoyuela/Desktop/ejemplo2.xml">

  <e1:carta>
    <e1:palo>Corazones</e1:palo>
    <e1:numero>7</e1:numero>
  </e1:carta>

  <e2:carta>
    <e2:carnes>
      <e2:filete_de_ternera precio="12.95"/>
      <e2:solomillo_a_la_pimienta precio="13.60"/>
    </e2:carnes>
    <e2:pescados>
      <e2:lenguado_al_horno precio="16.20"/>
      <e2:merluza_en_salsa_verde precio="15.85"/>
    </e2:pescados>
  </e2:carta>

</e1:ejemplo>
```

# Espacios de nombres en XML: Ejemplo

Tenemos dos documentos: ejemplo1.xml y ejemplo2.xml

```
<carta>
   <palo>Corazones</palo>
   <numero>7</numero>
</carta>
```

```
<carta>
   <carnes>
      <filete_de_ternera precio="12.95"/>
      <solomillo_a_la_pimienta precio="13.60"/>
   </carnes>
   <pescados>
      <lenguado_al_horno precio="16.20"/>
      <merluza_en_salsa_verde precio="15.85"/>
   </pescados>
</carta>
```

Si uniéramos estos documentos en uno solo puede originar conflictos, para evitarlo definimos los espacios de nombres:

Ejemplo de definición de espacio de nombres en cada etiqueta

```
<?xml version="1.0" encoding="UTF-8"?>
<e1:ejemplo xmlns:e1="file:///Users/mfhoyuela/Desktop/ejemplo1.xml”>

  <e1:carta>
    <e1:palo>Corazones</e1:palo>
    <e1:numero>7</e1:numero>
  </e1:carta>

  <e2:carta xmlns:e2="file:///Users/mfhoyuela/Desktop/ejemplo1.xml">
    <e2:carnes>
      <e2:filete_de_ternera precio="12.95"/>
      <e2:solomillo_a_la_pimienta precio="13.60"/>
    </e2:carnes>
    <e2:pescados>
      <e2:lenguado_al_horno precio="16.20"/>
      <e2:merluza_en_salsa_verde precio="15.85"/>
    </e2:pescados>
  </e2:carta>

</e1:ejemplo>
```

# Definición de espacio de nombre por defecto

Tenemos dos documentos: ejemplo1.xml y ejemplo2.xml

```
<carta>
   <palo>Corazones</palo>
   <numero>7</numero>
</carta>
```

```
<carta>
   <carnes>
      <filete_de_ternera precio="12.95"/>
      <solomillo_a_la_pimienta precio="13.60"/>
   </carnes>
   <pescados>
      <lenguado_al_horno precio="16.20"/>
      <merluza_en_salsa_verde precio="15.85"/>
   </pescados>
</carta>
```

Si uniéramos estos documentos en uno solo puede originar conflictos, para evitarlo definimos los espacios de nombres:

Ejemplo de definición de espacio de nombres en cada etiqueta

```
<?xml version="1.0" encoding="UTF-8"?>
<e1:ejemplo xmlns:e1="file:///Users/mfhoyuela/Desktop/ejemplo1.xml”>

  <e1:carta>
    <e1:palo>Corazones</e1:palo>
    <e1:numero>7</e1:numero>
  </e1:carta>

  <e2:carta xmlns:e2="file:///Users/mfhoyuela/Desktop/ejemplo1.xml">
    <e2:carnes>
      <e2:filete_de_ternera precio="12.95"/>
      <e2:solomillo_a_la_pimienta precio="13.60"/>
    </e2:carnes>
    <e2:pescados>
      <e2:lenguado_al_horno precio="16.20"/>
      <e2:merluza_en_salsa_verde precio="15.85"/>
    </e2:pescados>
  </e2:carta>

</e1:ejemplo>
```

# Documentos XML bien formados

- Una vez que sabemos todos los elementos y estructuras que pueden formar parte de un documento XML , estableceremos cuando un documento está bien especificado:
  - Documento bien formado: aquellos que cumplen las normas anteriormente expuestas, es decir que son sintácticamente correctos.
  - Documento válido: aquellos que además de ser bien formado, además cumplen los requisitos de una definición de estructura.

# Documentos XML bien formados

- Los nombres de los elementos y sus atributos deben estar escritos correctamente.
- Los valores de los atributos deben estar escritos entre comillas dobles o simples.
- Los atributos de un elemento deben separarse con espacios en blanco.
- Se tienen que utilizar referencias a entidades donde sea necesario.
- Tiene que existir un único elemento raíz.
- Todo elemento debe tener un elemento padre, excepto el elemento raíz.
- Todos los elementos deben tener una etiqueta de apertura y otra de cierre.
- Las etiquetas deben estar correctamente anidadas.
- Las instrucciones de proceso deben estar escritas de forma correcta.
- La declaración XML debe estar en la primera línea escrita correctamente.
- Las secciones CDATA y los comentarios deben estar correctamente escritos.

# Documentos XML bien formados

## Actividad:

- Especificar cuales son los fallos de los siguientes documentos:

**DOCUMENTO 1:**

```
<?xml version="1.0"?>
<documento>
      <p>Mi Primer <destacar
      importancia=1>documento
      XML</destacar></p]
      <p>Comienza con la etiqueta
      <documento&gt;</p>
      <p>A continuacion colocamos un
      elemento sin contenido</p>
      <imagen fichero="imagen.gif">
</documento>
```

**DOCUMENTO 2:**

```
<?xml version="1.0"?>
<libros>
      <libro id="quijote">
      <titulo>El Quijote</titulo>
      <autor nombre=cervantes nombre=cervantes>
      <descripcion>Es el m<ejor libro de cervantes.</descripcion>
      </libro]
</Libros>
```

https://www.xmlvalidation.com/