T1 XML

Lenguajes de Marcas

Lenguajes de Marcas

Definición:

Es la forma de codificar un documento de manera que, junto con el texto, se incorporan etiquetas, marcas con información adicional sobre la estructura del texto o su formato de presentación.

El lenguaje de marcas es el que especifica cuáles serán las etiquetas posibles, dónde deben colocarse y el significado que tendrá cada una de ellas.

XML y HTML

```
<contacto>
<nombre>Esther</nombre>
<apellidos>Lastra</apellidos>
<telefono>111222333</telefono>
<mail>esther@correo.es</mail>
</contacto>
```

```
<html>
    <head>
        <title>Ejemplo 1 HTML</title>
    </head>
    <body bgcolor='green'>
        <br/>
        <h1>Mi primer documento</h1>
        <p>Aquí escribo párrafos</p>
        <br/>
        <p><b>Vamos a ver</b></p>
            <ul>
                <li>XML </li>
                <li>HTML</li>
                <li>CSS</li>
            </ul>
    </body>
</html>
```

Diferencias entre lenguajes de marcas y lenguajes de programación.

Un lenguaje de marcado o lenguaje de marcas es una forma de **codificar un documento que, junto con el texto, incorpora etiquetas o marcas que contienen información adicional** acerca de la estructura del texto o su presentación.

Los lenguajes de marcas no disponen de los elementos típicos de programación como variables, arrays, sentencias de control, funciones, etc... Por lo tanto, **los lenguajes de marca no se programan.**

Históricamente, el marcado se usaba y se usa en la **industria editorial** y de la comunicación, así como entre autores, editores e impresores.

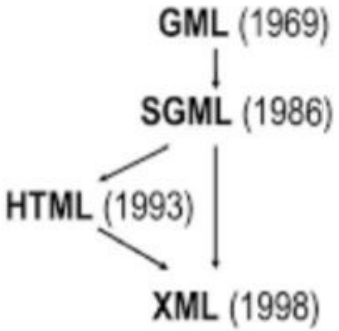
Busca información en Internet sobre XML, HTML, GML, SGML

¿Cual se creó primero?

¿Cómo evolucionaron los LM?

Busca ejemplos de cada uno de estos lenguajes

HISTORIA. INTRODUCCIÓN A LOS PRINCIPALES LENGUAJES DE MARCAS



GML (Generalized Markup Language): En los años 60, manejar documentos electrónicos tenía el problema de falta de compatibilidad entre aplicaciones, IBM intentó resolver estos problemas a través de un lenguaje de marcas denominado

SGML(Standard Generalized Markup Language): En 1986 GML pasó a manos de **ISO** (International Organization for Standardization) y se convirtió en SGML. Lenguaje de software libre y de código abierto (ISO 8879). SGML es un **metalenguaje**, es decir, un conjunto de normas que permiten crear otros lenguajes de marcas. Por ejemplo HTML, es uno de los lenguajes creados a partir de SGML.

HTML (Hyper Text Markup Language) : En 1989/90 Tim Berners-Lee creo HTML es una **versión simplificada de SGML**(+ASCII), de manera que rápidamente se convirtió en un estándar para la creación de páginas web. Pero tenía algunas desventajas:

- El lenguaje no es flexible, las etiquetas son limitadas.
- La estructura y el diseño están mezclados en el documento...

XML (eXtended Markup Language)

Finalmente, en el **1998, W3C (Word Wide Web Consortium)** hizo público un nuevo estándar que denominaron: XML (eXtended Markup Language)

Más sencillo que SGML y más potente que HTML. **Es una simplificación y adaptación de SGML, que permite definir lenguajes específicos.** Por lo tanto XML no es un lenguaje en particular, sino un **metalenguaje**. Utiliza reglas más estrictas que SGML (más formal). Se usa mucho para **base de datos y para exportación e importación de la información**. Algunos lenguajes basados en XML son (RSS, XHTML...).

EVOLUCIÓN DE LOS LENGUAJES DE MARCAS

Construye un ejemplo de XML y HTML con ellos en el aula. (XML-> catálogo de libros, HTML-> aparezca por pantalla Mi primera clase de LMSG).

¿Qué lenguaje presenta la información de forma legible para cualquiera?

¿Qué lenguaje muestra la estructura u organización jerárquica de la información?

¿Qué lenguaje estará más orientado a la presentación?

¿Qué lenguaje muestra mejor la estructura del documento?

Busca información en Internet sobre cómo la clasificación de los Lenguajes de Marcas.

T1_ Tarea1: Historia e introducción de los lenguajes de Marcas.

Lee los apuntes y responde a las preguntas.

1. XML.Elementos

Qué es XML

Lenguaje utilizado para el **almacenamiento e intercambio de datos estructurados** entre distintas plataformas.

XML es un **metalenguaje**, es decir, puede ser empleado para definir otros lenguajes, llamados dialectos XML. Por ejemplo, algunos lenguajes basados en XML son:

- RSS (Really Simple Syndication, Sindicación Realmente Simple).
- XHTML (eXtensible HyperText Markup Language, Lenguaje de Marcado de Hipertexto eXtensible).

Elementos en XML

Los documentos XML están formados por texto plano (**sin formato**) y contienen marcas (**etiquetas**) definidas por el desarrollador.

Dichas marcas, es recomendable que sean **lo más descriptivas posible** y, para escribirlas, se utilizan los caracteres menor que "<", mayor que ">" y barra inclinada "/".

<etiqueta>valor</etiqueta>

Elementos vacíos

En un documento XML, un elemento puede no contener ningún valor. En tal caso hay que escribir:

<etiqueta></etiqueta>

Se puede expresar lo mismo escribiendo:

<etiqueta/>

Relaciones padre-hijo entre elementos

Un elemento (padre) puede contener a otro u otros elementos (hijos):

En este ejemplo, el elemento **"persona"** contiene cuatro elementos (hijos): **"nombre", "mujer", "fecha de nacimiento" y "ciudad".**

A su vez, el elemento **"fecha de nacimiento"** contiene otros tres elementos (hijos): **"día", "mes" y "año"**.

Véase que, de todos los elementos que aparecen en este ejemplo, sólo el elemento "mujer" está vacío.

```
<persona>
  <nombre>Laura</nombre>
  <mujer/>
  <fecha-de-nacimiento>
    <día>10</día>
    <mes>6</mes>
    <año>1990</año>
  </fecha-de-nacimiento>
  <ciudad>Madrid</ciudad>
</persona>
```

Elemento raíz de un documento XML

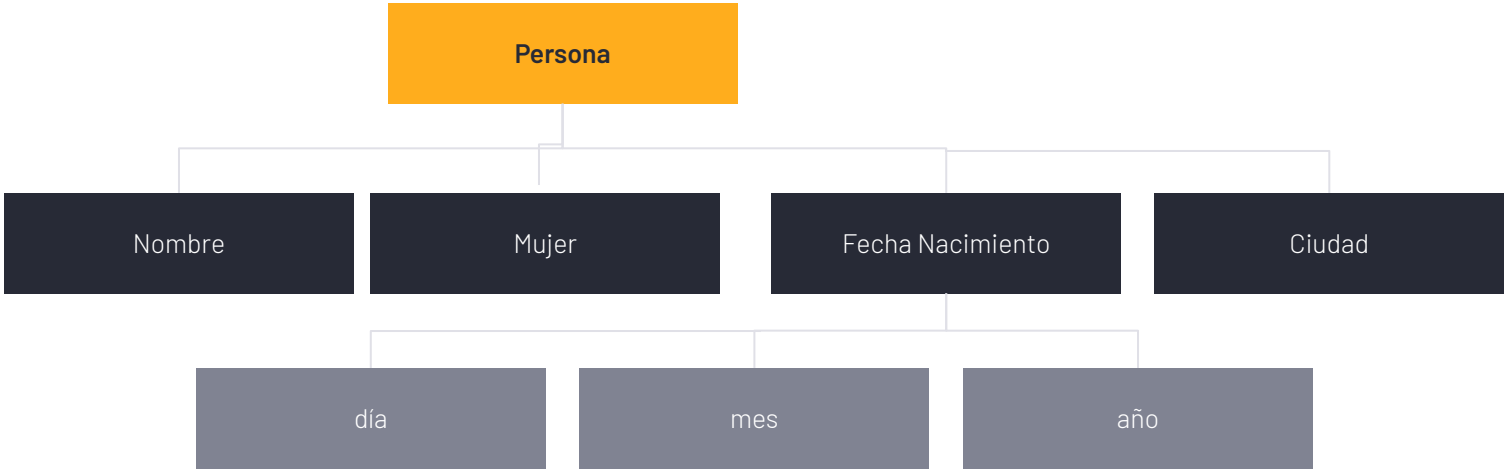
Todo documento XML tiene que tener **un único elemento raíz (padre)** del que descienden todos los demás.

En este caso, el elemento raíz es "persona".

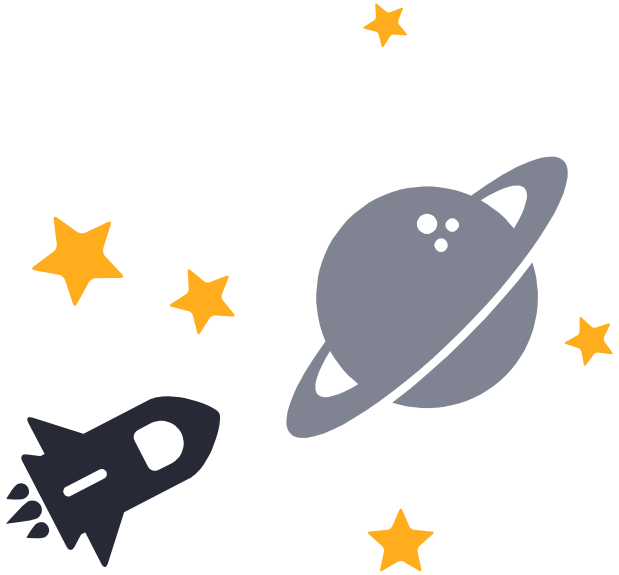
Gráficamente, la estructura de elementos de este documento se puede representar como se muestra a continuación:

```
<persona>
  <nombre>Laura</nombre>
  <mujer/>
  <fecha-de-nacimiento>
    <día>10</día>
    <mes>6</mes>
    <año>1990</año>
  </fecha-de-nacimiento>
  <ciudad>Madrid</ciudad>
</persona>
```

Estructura de árbol



Round 1

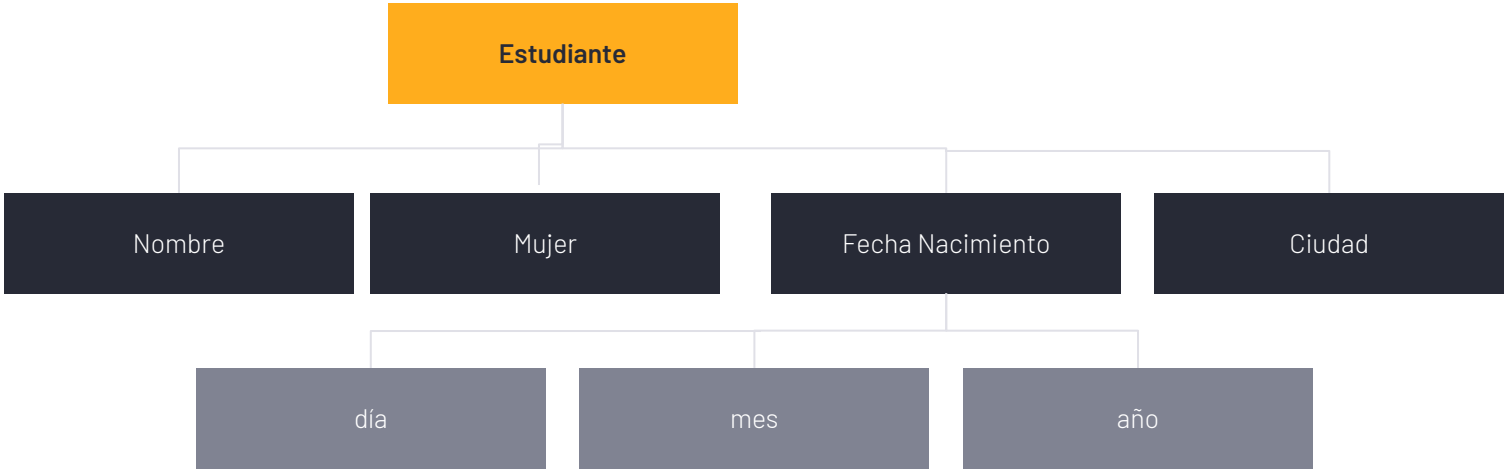


Elementos en XML

```
<estudiante>
        <nombre>Elisa</nombre>
        <mujer/>
        <fecha-de-nacimiento>
                <día>20</día>
                <mes>6</mes>
                <año>1996</año>
        </fecha-de-nacimiento>
        <ciudad>Santander</ciudad>
</estudiante>
```

Error:	No
Elemento raíz:	<estudiante>
Elementos vacios:	<mujer/>
Cuantos elementos hijos tiene el elemento estudiante:	4
Cuantos elementos hijos tiene el elemento fecha-de-nacimiento:	3
Representar en forma de árbol:	

Estructura de árbol

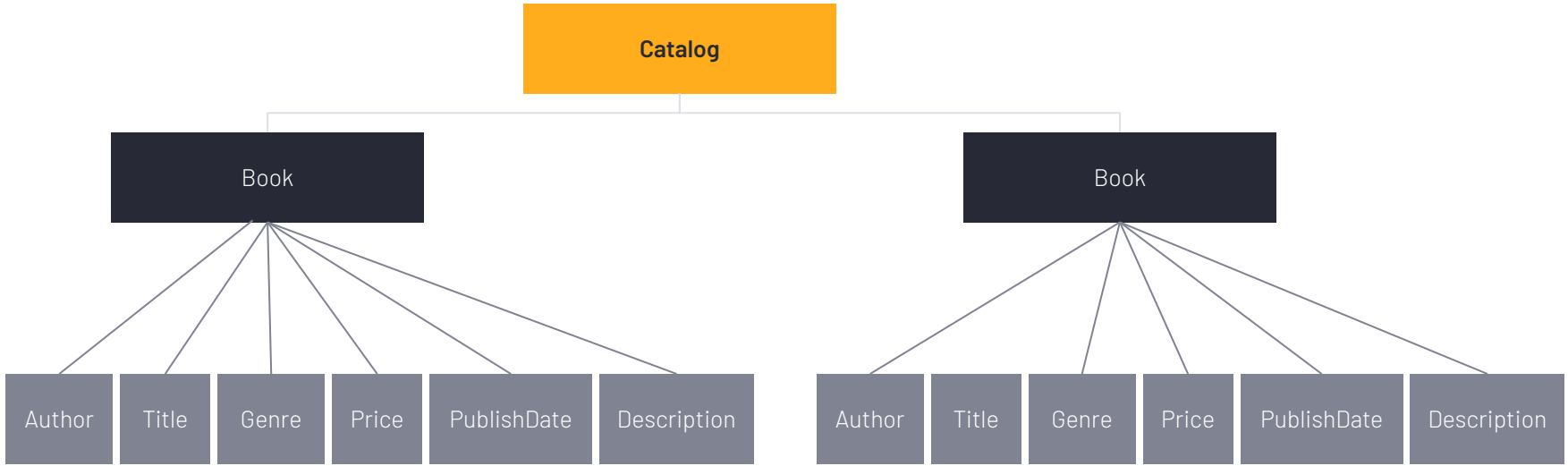


Elementos en XML

```
<Catalog>
 <Book id="bk101">
            <Author>Garghentini, Davide</Author>
            <Title>XML Developer's Guide</Title>
            <Genre>Computer</Genre>
            <Price>44.95</Price>
            <PublishDate>2000-10-01</PublishDate>
            <Description>An in-depth look at creating applications</Description>
 </Book>
 <Book id="bk102">
            <Author>Garcia, Debra</Author>
            <Title>Midnight Rain</Title>
            <Genre>Fantasy</Genre>
            <Price>5.95</Price>
            <PublishDate>2000-12-16</PublishDate>
            <Description>A former architect battles corporate
        zombies</Description>
 </Book>
</Catalog>
```

Error:	No
Elemento raíz:	<Catalog>
Elementos vacios:	No
Cuantos elementos hijos tiene el elemento Catalog:	2
Cuantos elementos hijos tiene el elemento Book:	6
Representar en forma de árbol:	

Estructura de árbol

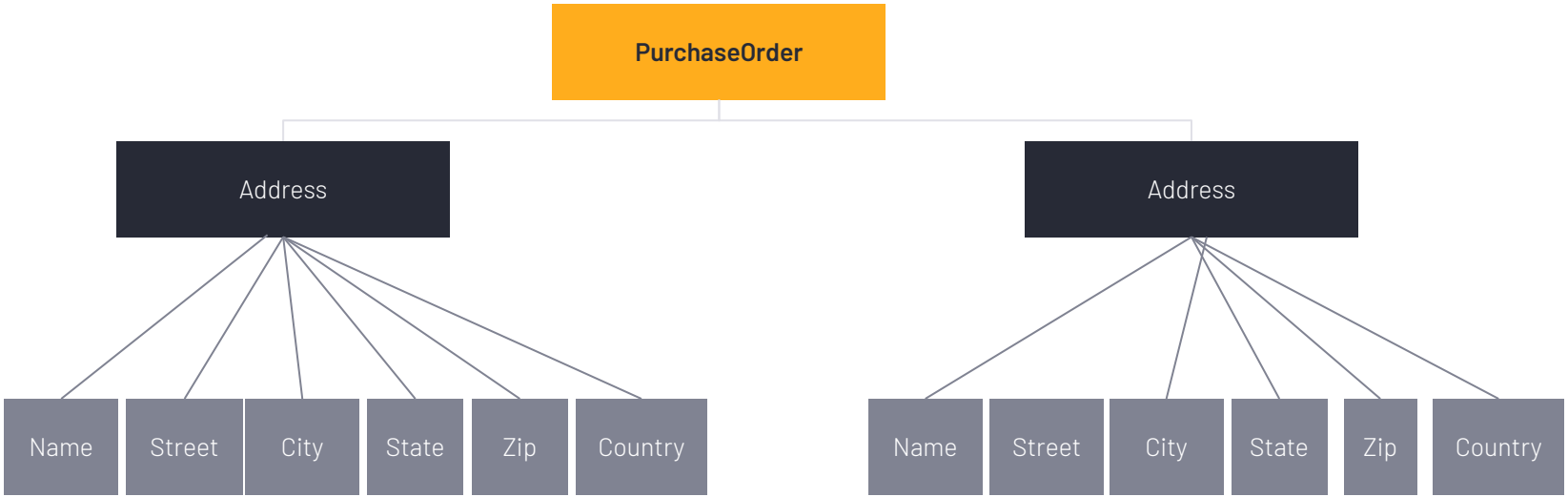


Elementos en XML

```
<PurchaseOrder PurchaseOrderNumber="99503" OrderDate="1999-10-20">
  <Address Type="Shipping">
            <Name>Ellen Adams</Name>
            <Street>123 Maple Street</Street>
            <City>Mill Valley</City>
            <State>CA</State>
            <Zip>10999</Zip>
            <Country>USA</Country>
  </Address>
  <Address Type="Billing">
            <Name>Tai Yee</Name>
            <Street>8 Oak Avenue</Street>
            <City>Old Town</City>
            <State>PA</State>
            <Zip\>
            <Country>USA</Country>
  </Address>
</PurchaseOrder>
```

Error:	<Zip/>
Elemento raíz:	<PurchaseOrder>
Elementos vacios:	Si
Cuantos elementos hijos tiene el elemento PurchaseOrder:	2
Cuantos elementos hijos tiene el elemento Address:	6
Representar en forma de árbol:	

Estructura de árbol



Ejemplo1(Contenido)

Escribe un documento xml para un catálogo de cine, con información sobre dos películas:

- La primera película se titula '**Tres anuncios en las afueras**' se estrenó en **2017**, fue dirigida por **Martin McDonagh** y la crítica la valoró con un **7**.

- La segunda película se titula '**La princesa prometida**' se estrenó en **1987**, fue dirigida por **Rob Reiner** y la crítica la valoró con un **8**.

Ejemplo1 (Sin Contenido)

Escribe un documento xml un catálogo de cine, con información sobre dos películas:

- El **catálogo** estará compuesto por **dos o más películas**.
- Para **cada película**, nos interesa conocer los siguientes datos:

  Nombre.
  Año de estreno.
  Nombre del director
  Puntuación de la crítica (0-10)
- Incluye dos películas de ejemplo.

2. Normas de sintaxis básicas en XML

Normas de sintaxis

Los nombres de los elementos son **case sensitive**, (sensibles a letras minúsculas y mayúsculas) teniendo que cumplir las siguientes normas:

- Pueden contener letras **minúsculas**, **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 "_"**.

Normas de sintaxis

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

>

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

<

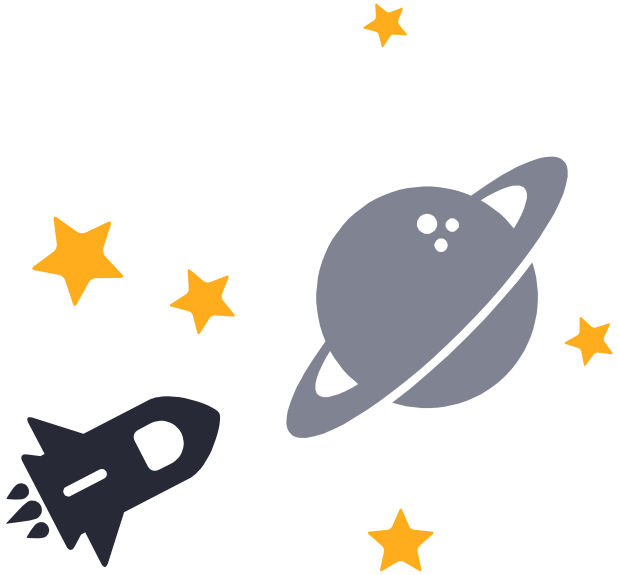
ciudad>Pamplona</ ciudad>

Normas de sintaxis

- Las letras no inglesas (**á, Á, ñ, Ñ...) están permitidas**. Sin embargo, e**s recomendable no utilizarlas** para reducir posibles incompatibilidades con programas que puedan no reconocerlas.

- En cuanto al carácter guion medio "-" y al punto ".", aunque también **están permitidos** para nombrar etiquetas, igualmente **se aconseja evitar su uso**.

Round 2



Sintaxis

Algunos de los siguientes elementos no están escritos correctamente por incumplir alguna regla de sintaxis:

Sintaxis

<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>

<Ciudad>Pamplona</Ciudad>

<día>18</día>

<mes>6</mes>

<ciudad>Pamplona</ciudad>

<_rojo/>

<colores2>Rojo y Naranja</colores2>

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

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

<color.favorito>azul</color.favorito>

<color-favorito>azul</color-favorito>

<color_favorito>azul</color_favorito>

Sintaxis

<address>C/Lealtad</Address>

<1 numero/>

<Nombre>6</Nombres>

<c.autonoma>Cantabria</c.autonoma>

<_estudio>DAM</_estudio>

<2curso>Bachillerato</2curso>

<_nombre.c>Nueva Delhi</_nombre.c>

<nombre>Elsa</nombre >

<color>azul</ color>

<address>C/Lealtad</address>

<numero1/>

<Nombre>6</Nombre>

Correcto

Correcto

<curso2>Bachillerato</curso2>

Correcto

Correcto

<color>azul</color>

Atributos en XML

Atributos en XML

- Los elementos de un documento XML pueden tener atributos definidos en la etiqueta de inicio.

- Un atributo sirve para proporcionar **información extra** sobre el elemento que lo contiene.

Atributos EJEMPLO

Dados los siguientes datos de un producto:

Código: F40

Nombre: Falda

Color: negro

Precio: 20

Su representación en un documento XML podría ser, por ejemplo:

```
<producto codigo="F40">
  <nombre color='Negro' precio="20">Falda</nombre>
</producto>
```

Atributos. Sintaxis

- Los nombres de los atributos deben cumplir las **mismas normas de sintaxis que los nombres de los elementos**.

- Todos los **atributos de un elemento tienen que ser únicos**.

Atributos. Sintaxis EJEMPLO

Son correctos los siguientes atributos. En caso de no serlo cuál es el error:

<nombre _color='Negro'>Falda</nombre>	Correcto
<producto 2codigo="F40"></producto>	<producto codigo2="F40"></producto>
<producto codigo_2="F40"></producto>	Correcto
<producto codigo.2="F40"></producto>	Correcto
<datos x="1" x="2" y="3"/>	<datos x="3" y="4" z="5"/>
<datos x="1" X="2" y="3"/>	Correcto

Elemento Vs Atributo:

Elemento Vs Atributo:

En ocasiones el diseñador puede elegir si cierta información va añadida en forma de atributo o como un nuevo elemento:

```
<paciente>
        <nombre>Ana</nombre>
        <apellido>Ríos</apellido>
        <edad>32</edad>
</paciente>
```

Ejemplo: Ficha médica de pacientes que contenga los siguientes datos: nombre, apellidos y edad.

```
<paciente edad="32">
        <nombre>Ana</nombre>
        <apellido>Ríos</apellido>
</paciente>
```

Elemento Vs Atributo:

Hay casos en los que debemos utilizar los atributos:

Secuenciación:

Diseña un documento XML que permita describir las instrucciones de montaje de un mueble. En las instrucciones de montaje debe describirse el **proceso a seguir secuenciado**, según el orden correspondiente:

1) desembalar, 2)asentar la base, 3) atornillar

```
<montaje>
    <paso>desembalar</paso>
    <paso>asentar la base</paso>
    <paso>atornillar</paso>
</montaje>
```

```
<montaje>
    <primero>desembalar</primero>
    <segundo>asentar la base</segundo>
    <tercero>atornillar</tercero>
</montaje>
```

```
<montaje>
    <paso orden="1">desembalar</paso>
    <paso  orden="2">>asentar la base</paso>
    <paso  orden="3">>atornillar</paso>
</montaje>
```

Elemento Vs Atributo:

Ejemplos:

Realiza un recetario con la siguiente información:

Receta: bizcocho de azafrán

Ingredientes: 250 ml leche

125 gr yogur

300 gr harina

200 gr de azúcar

Elemento Vs Atributo:

```
<recetario>
        <receta>
        <nombre>bizcocho de azafrán</nombre>
                <ingredientes>
                        <ingrediente unidad="mililitros" cantidad="250">leche</ingrediente>
                        <ingrediente unidad="gramos" cantidad="125">yogur</ingrediente>
                        <ingrediente unidad="gramos" cantidad="300">harina</ingrediente>
                        <ingrediente unidad="gramos" cantidad="200">azúcar</ingrediente>
                </ingredientes>
        </receta>
</recetario>
```

Elemento Vs Atributo:

Al crear un elemento debemos pensar en la naturaleza del dato, luego vamos a hacer un vocabulario para crear una estructura, no podemos crear 1000 etiquetas:

<aceite><tomate><arroz> → <ingredientes>

<fox_terrier><galgo><mastín> → <razas>

Si es el mismo tipo de información es un elemento.

Ejemplo XML

Escribe un documento xml para una herboristería, que cumpla las siguientes especificaciones:

- El catálogo estará compuesto por dos o más plantas.
- Para cada planta, nos interesa conocer los siguientes datos:
  - Nombre.
  - Color.
  - Precio.
- Además, cada planta tendrá como característica o propiedad, si es de tipo Interior o Exterior.

```
<catalogo>
    <planta tipo="exterior">
        <nombre>Rosa</nombre>
        <color>Rojo</color>
        <precio>20</precio>
    </planta>
    <planta tipo="interior">
        <nombre>Cactus</nombre>
        <color>Verde</color>
        <precio>5</precio>
    </planta>
</catalogo>
```

Ejemplo XML

Escribe un documento XML con información sobre un gimnasio.

La clase de yoga dura 2h, la imparte María Pérez, dos días a la semana.

La clase de esgrima dura 1h, la imparte Luis Fuentes, tres días a la semana.

OPCIÓN 1 :

```
<gimnasio>
        <clase>
                <profesor>María Perez</profesor>
                <duracion>2h</duracion>
                <d_semanales>2</d_semanales>
        </clase>
        <clase>
                <profesor>Luis Lopez</profesor>
                <duracion>1h</duracion>
                <d_semanales>3</d_semanales>
        </clase>
</gimnasio>
```

Opción 2 (Mejor):

```
<gimnasio>
    <clase>
        <profesor>
            <nombre>María</nombre>
            <apellido>Pérez</apellido>
        </profesor>
        <nombre>yoga</nombre>
        <duracion unidades="horas">2</duracion>
        <frecuencia unidades="días">2</frecuencia>
    </clase>
    <clase>
        <profesor>
            <nombre>Luis</nombre>
            <apellido>López</apellido>
        </profesor>
```

Ejemplo XML

Escribe un documento XML con información sobre una receta.

Receta:Pan de nueces

Ingredientes:
300 ml de agua.
300 gr harina.
20 gr levadura.
10 gr sal.
250 gr nueces.

Preparación:
Primer paso: tamizar la harina
Segundo paso:.mezclar harina, levadura, sal y agua.
Tercer paso: añadir nueces.
Cuarto paso: amasar dando forma al pan.
Quinto paso:Introducir en el horno 15 min.

```
<recetario>
	<receta>
		<nombre>Pan con nueces</nombre>
		<ingredientes>
			<ingrediente unidad="mililitros" cantidad="300">agua</ingrediente>
			<ingrediente unidad="gramos" cantidad="300">harina</ingrediente>
			<ingrediente unidad="gramos" cantidad="20">levadura</ingrediente>
			<ingrediente unidad="gramos" cantidad="10">sal</ingrediente>
			<ingrediente unidad="gramos" cantidad="250">nueces</ingrediente>
		</ingredientes>
		<preparacion>
			<paso orden="1">tamizar la harina</paso>
			<paso orden="2">mezclar harina, levadura, sal y agua</paso>
			<paso orden="3">añadir nueces</paso>
			<paso orden="4">amasar dando forma al pan</paso>
			<paso orden="5">Introducir en el horno 15 min</paso>
		</preparacion>
	</receta>
</recetario>
```

Kahoot XML

T1_Tarea1

Declaración XML

Declaración XML

Declaración XML contiene detalles que preparan al procesador XML para analizar el documento XML.

Es opcional, pero cuando se utiliza, debe aparecer en **primera línea** del documento XML.

El prólogo puede tener tres funciones:

- Declaración la versión de XML usada para elaborar el documento.

Para ello se utiliza la etiqueta:

<?xml version="1.0"?>

En este caso indica que el documento fue creado para la versión 1.0 de XML.

El prólogo puede tener tres funciones:

- Declaración de la codificación empleada para representar los caracteres.

  Determina el conjunto de caracteres que se utiliza en el documento. Para ello se escribe:

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

  En este caso se usa el código iso-8859-1 (Latin-1) que permite el uso de acentos o caracteres como la ñ.

Estándar ISO	Código de país
UTF-8 (Unicode)	Conjunto de caracteres universal
ISO -8859-1 (Latin-1)	Europa occidental, Latinoamérica
ISO -8859-2 (Latin-2)	Europa central y oriental

El prólogo puede tener tres funciones:

- Declaración de la autonomía del documento:

  Informa de si el documento necesita de otro para su interpretación.

  Para declararlo hay que definir el prólogo completo:

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

  En este caso, el documento es independiente, de no ser así el atributo standalone hubiese tomado el valor “no”.

declaración del tipo de documento

La **declaración del tipo de documento**, define qué tipo de documento estamos creando para ser procesado correctamente.

Toda declaración de tipo de documento comienza por la cadena:

<!DOCTYPE Nombre_tipo ...>

Si no hemos definido la DTD (Tema 4,) no incluimos la declaración del tipo de documento en el código XML.

Referencias a entidades en XML

Algunos caracteres tienen un significado especial en XML.

Si coloca un carácter como "<" dentro de un elemento XML, generará un error porque el analizador lo interpreta como el inicio de un nuevo elemento.

EJEMPLO

<empresa>m&m</empresa>

<empresa>m&m</empresa>

&lt;	<	less than
&gt;	>	greater than
&amp;	&	ampersand
&apos;	'	apostrophe
&quot;	"	quotation mark

Referencias de caracteres en XML

En un documento XML se pueden escribir referencias de caracteres Unicode con los símbolos **&#**, seguidos del valor decimal o hexadecimal del carácter Unicode que se quiera representar y, finalmente, añadiendo el carácter *punto y coma* "**;**".

Ejemplo:

Simbolo	valor decimal	valor hexadecimal
Euro (€)	&#8364;	&#x20AC;

Ejercicios

Escribe correctamente los siguientes programas y compruebalos en el navegador:

```
▼<Certificación>
  ▼<premio nombre="platino">
      <ventas>discos<200000</ventas>
    </premio>
  ▼<premio nombre="plata">
      <ventas>discos<500000</ventas>
    </premio>
  ▼<premio nombre="oro">
      <ventas>discos<800000</ventas>
    </premio>
  </Certificación>
```

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

<Certificación>

    <premio nombre="platino">
        <ventas>discos&lt;200000</ventas>
    </premio>

    <premio nombre="plata">
        <ventas>discos&lt;500000</ventas>
    </premio>

    <premio nombre="oro">
        <ventas>discos&lt;800000</ventas>
    </premio>


</Certificación>
```

```
▼<registro>
  ▼<marca>
      <nombre>H&M</nombre>
      <sector>textil</sector>
    </marca>
  ▼<marca>
      <nombre>Dunkin'</nombre>
      <sector>alimentación</sector>
    </marca>
  ▼<marca>
      <nombre>P&G</nombre>
      <sector>limpieza</sector>
    </marca>
  </registro>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<registro>
  <marca>
    <nombre>H&amp;amp;M</nombre>
    <sector>textil</sector>
  </marca>

  <marca>
    <nombre>Dunkin&amp;apos;</nombre>
    <sector>alimentación</sector>
  </marca>

  <marca>
    <nombre>P&amp;amp;G</nombre>
    <sector>limpieza</sector>
  </marca>
</registro>
```

```
<matematicas>
  <simbolo nombre="pi">π</simbolo>
  <simbolo nombre="infinito">∞</simbolo>
  <simbolo nombre="cuarto">¼</simbolo>
  <simbolo nombre="diametro">ø</simbolo>
</matematicas>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<matematicas>
  <simbolo nombre="pi">&#8508;</simbolo>
  <simbolo nombre="infinito">&#8734;</simbolo>
  <simbolo nombre="cuarto">&#188;</simbolo>
  <simbolo nombre="diametro">&#248;</simbolo>
</matematicas>
```

comentarios en XML

Para escribir comentarios en un documento XML, estos deben escribirse entre los caracteres "<!--" y "-->".

<!--Ejemplo uso de comentarios.-->

Espacios de nombres en XML

Los documentos XML se pueden mezclar, combinar, por lo que si dos elementos distintos tienen el mismo nombre pueden surgir conflictos.

Para resolver estos conflictos utilizaremos los espacios de nombres.

Definición espacio de nombres: utilizamos el atributo xmlns

xmlns:prefijo="URI"

Espacios de nombres en XML

EJEMPLO

Dos documentos XML podrían contener un elemento llamado "carta", pero con significados distintos.

```
<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>
```

Espacios de nombres en XML

De forma que, si se incluyen ambos elementos <carta> en un documento XML, se origina un conflicto de nombres. Para resolverlo, se pueden utilizar espacios de nombres (XML Namespaces). Por ejemplo, escribiendo:

Espacios de nombres en XML

```
<?xml version="1.0" encoding="UTF-8"?>
<ejemplo xmlns:e1="http://www.lmsgi.com/ejemplo1" xmlns:e2="http://www.lmasgi.com/ejemplo2">
    <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>

</ejemplo>
```

Espacios de nombres en XML

```
<?xml version="1.0" encoding="UTF-8"?>
<ejemplo >
    <e1:carta  xmlns:e1="http://www.lmsgi.com/ejemplo1" >
      <e1:palo>Corazones</e1:palo>
      <e1:numero>7</e1:numero>
   </e1:carta>

<e2:carta xmlns:e2="http://www.lmasgi.com/ejemplo2">
     <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>

</ejemplo>
```

Espacios de nombres en XML

Ejemplo:

Tengo un documento XML con una tabla de alimentos, donde figuran sus calorias:

```
<tabla>
    <tipo>fruta</tipo>
    <nombre>manzana</nombre>
    <calorias>64</calorias>
  </tabla>
```

Tengo otro documento XML con información sobre el material de una carpintería:

```
    <tabla>
      <material>madera</material>
      <color>blanco</color>
      <peso>3 </peso>
    </tabla>
```

Espacios de nombres en XML

Si estos fragmentos XML se agregan juntos, habría un conflicto de nombres. Ambos contienen un elemento <tabla>, pero los elementos tienen diferente contenido y significado.

Solución: Utilizar espacios de nombres.

Podemos resolver fácilmente el conflicto utilizando un prefijo de nombre, para especificar si nos referimos a la 'tabla' de alimentos o al mueble.

```
<a:tabla>
    <a:tipo>fruta</a:tipo>
    <a:nombre>manzana</a:nombre>
    <a:calorias>64</a:calorias>
  </a:tabla>
```

```
<m:tabla>
    <m:material>madera</m:material>
    <m:color>blanco</m:color>
    <m:peso>3 </m:peso>
  </m:tabla>
```

Espacios de nombres en XML

En el ejemplo anterior, no habrá conflicto porque los dos elementos <tabla> tienen nombres diferentes.

Cuando se usan prefijos en XML, se debe definir un espacio de nombres para el prefijo.

El espacio de nombres se puede definir mediante un atributo xmlns en la etiqueta de inicio de un elemento.

La declaración del espacio de nombres tiene la siguiente sintaxis. xmlns: prefix = "URI"

```
<a:tabla xmlns:a="http://www.lmsgi.com/alimentos">
    <a:tipo>fruta</a:tipo>
    <a:nombre>manzana</a:nombre>
    <a:calorias>64</a:calorias>
  </a:tabla>
```

```
<m:tabla xmlns:m="http://www.lmasgi.com/muebles">
      <m:material>madera</m:material>
      <m:color>blanco</m:color>
      <m:peso>3 </m:peso>
    </m:tabla>
```

Espacios de nombres en XML

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

    <a:tabla xmlns:a="http://www.lmsgi.com/alimentos">
     <a:tipo>fruta</a:tipo>
     <a:nombre>manzana</a:nombre>
     <a:calorias>64</a:calorias>
    </a:tabla>

     <m:tabla xmlns:m="http://www.lmasgi.com/muebles">
       <m:material>madera</m:material>
       <m:color>blanco</m:color>
       <m:peso>3 </m:peso>
     </m:tabla>

</ejemplo>
```

Espacios de nombres en XML

Los espacios de nombres también se pueden declarar en el elemento raíz XML:

```
<?xml version="1.0" encoding="UTF-8"?>
<ejemplo xmlns:a="http://www.lmsgi.com/alimentos" xmlns:m="http://www.lmasgi.com/muebles">

    <a:tabla >
     <a:tipo>fruta</a:tipo>
     <a:nombre>manzana</a:nombre>
     <a:calorias>64</a:calorias>
    </a:tabla>

     <m:tabla >
       <m:material>madera</m:material>
       <m:color>blanco</m:color>
       <m:peso>3 </m:peso>
     </m:tabla>

</ejemplo>
```

Ejercicios

Espacio de nombres: Ejemplo 1

Una empresa de catering realiza comidas para eventos, pero los postres los encarga a alguna de sus dos empresas asociadas (Dulcisimo o Choco). A partir de los documentos "dulcisimo.xml" y "choco.xml", crea el documento **"menus.xml"**. Utiliza espacios de nombres para hacer referencia a las etiquetas de "dulcisimo.xml" y "choco.xml" que vas a usar en el documento "menus.xml". dulcisimo.xml y choco.xml: contienen información sobre sus postres: el nombre, ingredientes, precio....

El documento "menus.xml" debe contener los menús servidos este mes. Para cada menú se necesita el **nombre del menú,** el **primer plato**, el **segundo plato**, el **postre** (del documentos "dulcisimo.xml" o "choco.xml" según corresponda) y el **precio** total del menú.

Incluye en el documento "menus.xml" los siguientes datos: Nombre del menú "Mar y tierra" primer plato “sopa de pescado”, segundo plato “chuletillas de cordero” y el nombre del postre de la pastelería dulcisimo “tarta de zanahoria”.

Nombre del menú "vegetariano", primer plato “crema de calabaza”, segundo plato “revuelto de setas” y el nombre del postre de la pastelería choco “mousse de chocolate”.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<menus xmlns:dulcisimo="https://dulcisimo.es/dulcisimo.xml" xmlns:choco="https://choco.es/choco.xml"
    <menu nombre="Mar y tierra">
        <plato orden="1">Sopa pescado</plato>
        <plato orden="2">Chuletillas cordero</plato>
        <dulcisimo:postre>Tarta zanahoria</dulcisimo:postre>
        <precio>18€</precio>
    </menu>

    <menu nombre="Vegetariano">
        <plato orden="1">Crema calabaza</plato>
        <plato orden="2">Revuelto setas</plato>
        <choco:postre>Mousse chocolate</choco:postre>
        <precio>17€</precio>
    </menu>

</menus>
```

Espacio de nombres: Ejemplo 2

Sean los documentos XML que organizan la información sobre los profesores y los alumnos del DAM respectivamente:

```xml
<?xml version="1.0" encoding="iso-8859-1" standalone="yes" ?>
<alumnos>
  <nombre>Ignacio Fernández González</nombre>
  <nombre>Marina González Fernández</nombre>
  <nombre>Javier Martínez López</nombre>
</alumnos>
```

```xml
<?xml version="1.0" encoding="iso-8859-1" standalone="yes" ?>
<profesores>
 <nombre>Ricardo Ruiz Pérez</nombre>
 <nombre>Marta Rodríguez Hernández</nombre>
</profesores>
```

Realiza un documento "**Charla.xml**" para recoger información sobre los asistentes a una charla divulgativa. Para ello contamos con el documento **"profesores.xml"** que contiene información relativa a los profesores de DAM y el documento "**alumnos.xml**" con todos los alumnos que se encuentran en el curso.

El documento "Charla.xml" debe contener los siguientes datos para cada asistente a la charla: **tipo** que puede ser alumno o profesor, **nombre** (referido a profesores.xml o alumnos.xml), **identificador** de asistente como propiedad del elemento asistente , **asiento** que ocupa y **fecha** de inscripción en la charla.

Al hacer un documento sobre los miembros del curso DAM no se distinguiría los profesores de los alumnos, para resolverlo definiremos un espacio de nombres para cada contexto:

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

<charla xmlns:alumno="http://DAM/alumnos.xml"
        xmlns:profesor="http://DAM/profesores.xml">
    <asistentes>
        <asistente id="718">
            <tipo>Alumno</tipo>
            <alumno:nombre>Juan López</alumno:nombre>
            <asiento>12B</asiento>
            <fecha>20/2/2021</fecha>
        </asistente>
        <asistente id="68">
            <tipo>Alumno</tipo>
            <alumno:nombre>Ana Dominguez</alumno:nombre>
            <asiento>33A</asiento>
            <fecha>1/4/2021</fecha>
        </asistente>
        <asistente id="21">
            <tipo>Profesor</tipo>
            <profesor:nombre>Ana Dominguez</profesor:nombre>
            <asiento>1A</asiento>
            <fecha>22/1/2021</fecha>
        </asistente>
    </asistentes>
</charla>
```

Espacio de nombres: Ejemplo 3

Realiza un documento “gala.xml” para recoger información sobre los asistentes a una entrega de premios. Para ello contamos con el documento **“musicos.xml”** que contiene toda la información relativa a los músicos (nombre, apellido, dirección, fecha nacimie a los productores de música.

El documento “gala.xml” musico o productor), nombre y apellido, hora

```xml
<?xml version="1.0" encoding="UTF-8"?>
<musicos>
    <musico>
        <nombre>Luis</nombre>
        <apellido>Lopez</apellido>
        <direccion>C/Herrera 3 Asturias</direccion>
        <f_nac>25/04/1999</f_nac>
    </musico>
    <musico>
        <nombre>Sara</nombre>
        <apellido>Diaz</apellido>
        <direccion>C/Bernesgas 11 Madrid</direccion>
        <f_nac>02/11/2000</f_nac>
    </musico>

</musicos>
```

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

<gala xmlns:musico="http://musicos.xml"
      xmlns:productor="http://productores.xml">
    <asistentes>
        <asistente>
            <trabajo>musico</trabajo>
            <musico:nombre>Juan</musico:nombre>
            <musico:apellido>Guerra</musico:apellido>
            <hora>12:00</hora>
            <asiento>10C</asiento>
            <premios>2</premios>
        </asistente>
        <asistente>
            <trabajo>productor</trabajo>
            <productor:nombre>Ernesto</productor:nombre>
            <productor:apellido>Aragon</productor:apellido>
            <hora>13:00</hora>
            <asiento>15A</asiento>
            <premios>0</premios>
        </asistente>

    </asistentes>
</gala>
```

Documentos XML bien formados y válidos

Se dice que un documento XML está **bien formado** (well-formed document) cuando no tiene errores de sintaxis:

- 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.

Se dice que un documento XML es **válido** (valid) cuando, además de no tener errores de sintaxis, no incumple ninguna de las normas establecidas en su estructura. Dicha estructura se puede definir utilizando distintos métodos, como:

DTD (Document Type Definition, Definición de Tipo de Documento).

XML Schema.

Ejercicios