



# LENGUAJE DE MARCAS CREACIÓN Y VALIDACIÓN DE XML

## Introducción, evolución y estado actual

- XML (eXtensible Markup Language, “lenguaje de marcas extensible”, en castellano).
- XML es un metalenguaje (permite construir otros lenguajes y dialectos) de propósito general.
- XML no define etiquetas: define reglas.
- Tiene como orígenes los lenguajes GML y SGML.
- Características:
  - Sencillo.
  - Robusto.
  - Formato abierto (no tiene coste).

**Las versiones de XML son gestionadas por el W3C**

Año	Versión	Observaciones
1998	XML 1.0	Primera edición
2000	XML 1.0	Segunda edición
2004	XML 1.0	Tercera edición
2004	XML 1.1	Primera edición
2006	XML 1.0	Cuarta edición
2006	XML 1.1	Segunda edición
2008	XML 1.0	Quinta edición

## Aplicaciones basadas en XML

- Personalización de la WEB (lenguaje propio) Cada usuario (o grupo de usuarios) puede crear su propio lenguaje para el formato de datos y documentos, su propio vocabulario, según sus necesidades.
- Aplicaciones XML para el comercio electrónico (intercambio) Aunque inicialmente XML se definió para separar contenido de presentación, ha resultado esencial para el intercambio de información estructurada a través de Internet
- Gestión de la información / conocimiento (visual) Si etiquetamos la información y a cada usuario se le proporciona una serie de etiquetas de interés, se podría resaltar la información que le es interesante, frente a la que no es relevante.
- Descargar trabajo en el Servidor (desaturación) Por medio del Modelo de Objetos de Documentos (DOM), podemos evitarle trabajo al servidor, espera al cliente y no saturar tanto la red.
- Buscador WEB (velocidad en búsquedas) Si disponemos de un sitio donde toda la información se encuentre etiquetada en documentos XML, las búsquedas serían mucho más efectivas, ya que se conjuga la potencia de la búsqueda indexada junto la búsqueda semántica.
- Intercambio de información (seguridad, velocidad gestiones) Si contratamos a una empresa y nos facilitan la estructura de los datos que vamos a recibir (DTD/Schema), sabremos en todo momento qué tipos de documentos XML estamos recibiendo, y podremos tratarlos de la forma que deseemos.

# Aplicaciones basadas en XML



- Ejemplo Real de Método Antiguo Transmisión de información

# Aplicaciones basadas en XML

- Ejemplo real actual

```
<DSC>
  <codproducto>YNUA00385</codproducto>
  <FechaCreacionUsuario>2002-07-15 00:00:00.0</FechaCreacionUsuario>
  <VersionProducto>2</VersionProducto>
  <despachador>A54</despachador>
  <RutCliente>86223700</RutCliente>
  <DigitoVerificador>5</DigitoVerificador>

  <DescripcionDeclarada>
    ; SIERRA ALTERNATIVA; ALBER; KERF GANGSAW; DE 420MM X 13MM X 40MM X
  0,80MM., PARA MAQUINA PARA TRABAJAR LA MADERA
  </DescripcionDeclarada>
  <FechaCargaAduana>2002-08-14 16:27:58.0</FechaCargaAduana>
  <RESPUESTAS>
    <respuesta>
      ACEPTADO
    </respuesta>
  </RESPUESTAS>
</DSC>
```

# ESTRUCTURA DE DOCUMENTOS XML

Un documento XML tiene dos estructuras, una lógica y otra física.

- Físicamente, un documento XML puede consistir en una o más unidades de almacenamiento, llamadas entidades. Las entidades tienen contenido y están identificadas por un nombre. Cada documento XML contiene una entidad, llamada entidad documento, que sirve como punto de partida para el procesador XML y que puede contener el documento completo.
- Lógicamente, esta estructurado en forma de árbol, con una raíz a partir de la cual se organiza la información. El documento está compuesto de declaraciones, elementos, comentarios, referencias a caracteres e instrucciones de procesamiento, todos los cuales están indicados por una marca explícita.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE cv SYSTEM "2_cv_tunombre.dtd">
<cv>
    <nombre>Juan Pérez</nombre>
    <dirección>Calle Falsa 123</dirección>
    <teléfono>123456789</teléfono>
    <fax>987654321</fax>
    <email>juan.perez@example.com</email>
    <email>jperez@otraempresa.com</email>
    <idiomas>
        <idioma>Español</idioma>
        <idioma>Inglés</idioma>
    </idiomas>
</cv>
```

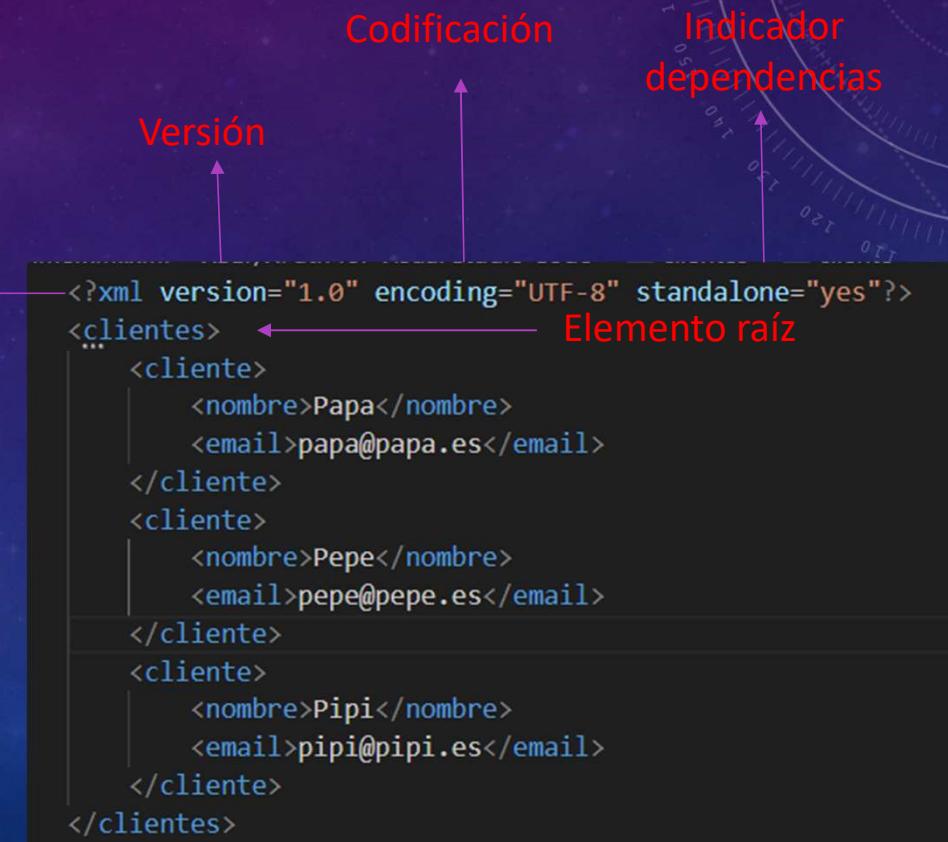
FÍSICA

LÓGICA

# Estructura y sintaxis de XML

- Un documento XML tiene la siguiente estructura:
- Declaración.
  - Versión.
  - Codificación.
  - Indicador de dependencias.
- Contenido.
  - Elemento raíz.
  - Resto de elementos.

```
This XML file does not appear to have any style information as
Declaración
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<clientes>
  <cliente>
    <nombre>Papa</nombre>
    <email>papa@papa.es</email>
  </cliente>
  <cliente>
    <nombre>Pepe</nombre>
    <email>pepe@pepe.es</email>
  </cliente>
  <cliente>
    <nombre>Pipi</nombre>
    <email>pipi@pipi.es</email>
  </cliente>
</clientes>
```



The diagram illustrates the components of an XML document structure. At the top, three red boxes represent the main components: 'Declaración' (Declaration), 'Versión' (Version), and 'Codificación' (Encoding). Below them, a large black box contains the XML code. A red bracket labeled 'Elemento raíz' (Root Element) spans the entire opening tag of the root element, '<clientes>'. Two purple arrows point upwards from the text 'Declaración' and 'Versión' to their respective components in the diagram. A blue arrow points upwards from the text 'Codificación' to its component in the diagram.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<clientes>
  <cliente>
    <nombre>Papa</nombre>
    <email>papa@papa.es</email>
  </cliente>
  <cliente>
    <nombre>Pepe</nombre>
    <email>pepe@pepe.es</email>
  </cliente>
  <cliente>
    <nombre>Pipi</nombre>
    <email>pipi@pipi.es</email>
  </cliente>
</clientes>
```

## XML – Declaración

- **version:** Indica la versión de XML usada en el documento. Es obligatorio ponerlo, a no ser que sea un documento externo a otro que ya lo incluía.
- **encoding:** La forma en que se ha codificado el documento. Se puede poner cualquiera, y depende del parser el entender o no la codificación. Por defecto es UTF-8, aunque podrían ponerse otras, como UTF-16, US-ASCII, ISO-8859-1, etc. No es obligatorio salvo que sea un documento externo a otro principal.
- **standalone:** Indica si el documento va acompañado de un DTD ("no"), o no lo necesita ("yes"); en principio no hay porqué ponerlo, porque luego se indica el DTD si se necesita

Los elementos pueden tener atributos, que son una manera de incorporar características o propiedades a los elementos de un documento.

Ej. un elemento "chiste" puede tener un atributo "tipo" y un atributo "calidad", con valores "vascos" y "bueno" respectivamente.

Cada documento XML contiene uno o más elementos, cuyos límites están delimitados por etiquetas de comienzo y de final, en el caso de elementos vacíos, por una etiqueta de elemento vacío.

Cada elemento tiene un tipo, identificado por un nombre, denominado identificador genérico, y puede tener un conjunto de especificaciones de atributos.

La "declaración de tipo de documento" define qué tipo de documento estamos creando para ser procesado. Es decir, definimos que declaración de tipo de documento (DTD) valida y define los datos que contiene nuestro documento XML.

Un identificador público (PUBLIC): que hace referencia a dicho DTD. El tipo de documento Identificador universal de recursos (URI): precedido de la palabra SYSTEM. Dónde encontrar la información sobre su Definición.

Un documento XML es una estructura en forma de árbol.

- Contiene una raíz.
- De la raíz descienden elementos que a su vez pueden ser contenedores de otros elementos.

Las relaciones entre los elementos de un documento XML son las siguientes:

1. Existe un único elemento raíz (root).
2. Cada uno de los elementos descendientes directos de un elemento se llama hijo (children).
3. El elemento ascendiente directo de un elemento se llama parent (parent).
4. Los elementos que tienen un parent común se denominan hermanos (sibling).

Los elementos tienen una etiqueta de apertura, una etiqueta de cierre y, opcionalmente, uno o más atributos.

- Los atributos están compuestos de pares parámetro-valor:

```
<nombre-elemento atributo-1="valor1" atributo-2="valor2" >  
</nombre-elemento>
```

- Los elementos pueden estar vacíos, contener texto o a otros elementos.

## Estructura de XML

- Un ejemplo de un documento XML bien formado:

```
<?xml version="1.0" standalone="yes"?>
<biblioteca>
    <libro call_no="PZ3.S8195Gr6">
        <cover href="grapes.gif" alt="Grapes of Wrath"/>
        <titulo>The Grapes of Wrath</titulo>
        <autor>
            <apellido>Steinbeck</apellido>
            <nombre>John</nombre>
        </autor>
        <publicación>Viking Press</publicación>
        <año_pub>1939</año_pub>
    </libro>
</biblioteca>
```

## **Reglas para asignar nombre a los elementos o etiquetas:**

1. Diferencian entre mayúsculas y minúsculas.
2. Deben comenzar con una letra o un guion bajo.
3. Los nombres de elementos tienen que ser idénticos en las etiquetas de apertura y de cierre.
4. Los nombres pueden estar formados por caracteres alfanuméricos, guiones, guiones bajos y puntos.
5. Los nombres no pueden contener espacios (interpreta como nombre del elemento la primera palabra y el resto como parámetros)

## **Reglas para la creación de atributos:**

1. Deben tener asignado un valor.
2. Los valores siempre van entrecomillados, admitiendo comillas simples y dobles (deben coincidir el tipo de comilla de cierre con el de apertura).
3. Diferencian entre mayúsculas y minúsculas.
4. Deben comenzar con una letra o un guion bajo.
5. Pueden estar formados por caracteres alfanuméricos, guiones, guiones bajos y puntos.

## COMENTARIOS

- Contienen información que no es procesada ni interpretada.
- Comienzan por <!--
- Terminan por -->  
    <!-- Este texto es un comentario -->

## ESPACIOS DE NOMBRES

- Permiten resolver las ambigüedades en caso de que varios elementos tengan el mismo nombre.
- Declaración:  
    xmlns:nombre-espacio-nombres=URL
- Uso:  
    <nombre-espacio-nombres:nombre-elemento>
- Ejemplo:  
    xmlns:autorestecnicos='http://unaurlcualquiera';  
    <autorestecnicos:autor>

## ENTIDADES

- Son un mecanismo para representar información dentro de un documento mediante referencias en lugar de incluirla directamente.
- Se usan por ejemplo, para definir una información común (datos de contacto) y poder incluirla en varios documentos sin copiar lo mismo varias veces.
- Tipos:
  - Generales
    - Internas. Contienen el valor en el propio documento.
    - Externas. Contienen el valor en un documento externo.

- **Entidades internas predefinidas:**

Entidad	Significado	Carácter
&lt;	Símbolo de "menor que"	<
&gt;	Símbolo de "mayor que"	>
&amp;	Símbolo del ampersand (en español et)	&
&apos;	Comilla simple	'
&quot;	Comilla doble	"

## Sección CDATA

- Contiene un conjunto de caracteres que no debe ser tratado por el analizador.
- Se utilizan habitualmente para almacenar código XML o HTML que no queremos que se valide y de error.
- Sintaxis:
  - Comienzo: <![CDATA[
  - Fin: ]]>

### Ejemplo:

```
<?xml version="1.0" encoding="UTF-8"?>
<ficha>
    <nombre>Lenguaje C</nombre>
    <info-html>
        <![CDATA[
            <h1>LENGUAJES DE PROGRAMACIÓN<h1>
            <hr>
            <a href="url-enlace">C</a>
        ]]>
    </info-html>
</ficha>
```

## VALIDACIONES DE XML

- XML dispone de mecanismos de validación.
- Dos perspectivas:
  1. XML bien formado. Es conforme a las reglas generales de XML.
  2. XML válido. Está bien formado y conforme a las reglas definidas por el creador o diseñador del documento.

Esas reglas se definen en un DTD (Document Type Definition) o XML Schema.

### XML bien formado

Reglas:

1. Debe haber uno y solo un elemento raíz.
2. Todos los elementos deben estar cerrados.
3. Los elementos deben estar anidados correctamente: no se pueden intercalar aperturas y cierres de elementos distintos.
4. Todos los valores de los atributos están entrecomillados.
5. Los nombres de elementos y atributos deben cumplir con sus respectivas reglas.

### XML válido

- Alternativa 1: Debe cumplir las reglas especificadas en un DTD.
- Alternativa 2: Debe cumplir las reglas especificadas en un XML-Schema.

## DTD

- Document Type Definition (Definición de Tipo de Documento).
- Contiene una serie de reglas que sirven para validar un documento XML.
- Alternativas de uso (como los CSS)
- DTD interno. Incluido en el propio documento XML.
- DTD externo. Almacenado en un fichero externo y referenciado desde el documento XML (el que usaremos).

- **DTD interno. Ejemplo:**

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE aviso [
  <!ELEMENT aviso (de,para,mensaje)>
  <!ELEMENT de (#PCDATA)>
  <!ELEMENT para (#PCDATA)>
  <!ELEMENT mensaje (#PCDATA)>
]>
<aviso>
  <de>David</de>
  <para>Rosalía</para>
  <mensaje>Mañana nos vemos en el estudio a las 10.</mensaje>
</aviso>
```

- **DTD externo. Ejemplo:**

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE aviso SYSTEM "dtdExterno.dtd">
<aviso>
  <de>David</de>
  <para>Rosalía</para>
  <mensaje>Mañana nos vemos en el estudio a las 10.</mensaje>
</aviso>
```

Entonces, un DTD son un conjunto de reglas que nos dicen lo que se puede o no se poner en un archivo xml concreto.

Ejemplo: supongamos que nos piden un xml para almacenar la siguiente información:

- El elemento raíz es <listaclientes>.
- Dentro de <listaclientes> deseamos permitir uno o más elementos <cliente>.
- Dentro de <cliente> todos deberán tener <cif> y <nombre> y en ese orden.
- Dentro de <cliente> puede aparecer o no un elemento <diasentrega> para indicar que ese cliente exige un máximo de plazos. Como no todo el mundo usa plazos, el <diasentrega> es optativo.

- **Un xml y dtd para este caso sería:**

```
<listaclientes>
  <cliente>
    <nombre>Mercasa</nombre>
    <cif>5676443</cif>
  </cliente>
  <cliente>
    <cif>5121554</cif>
    <nombre>Acer SL</nombre>
  </cliente>
</listaclientes>
```

```
<!ELEMENT listaclientes (cliente+)>
<!ELEMENT cliente (nombre, cif, diasentrega?)>
<!ELEMENT nombre (#PCDATA)>
<!ELEMENT cif (#PCDATA)>
<!ELEMENT diasentrega (#PCDATA)>
```

- Se establece el tipo de documento listaclientes que consta de una serie de elementos (dentro del corchete)
- Un elemento listaclientes consta de uno o más clientes. El signo + significa «uno o más».
- Un cliente tiene un nombre y un cif. También puede tener un elemento diasentrega que puede o no aparecer (el signo ? significa «0 o 1 veces»).
- Un nombre no tiene más elementos dentro, solo caracteres (#PCDATA)
- Un CIF solo consta de caracteres.
- Un elemento diasentrega consta solo de caracteres.

## XML - DTD

- Un "Document type definition", DTD, es una declaración en un documento de SGML o de XML que especifica apremios en la estructura del documento. Puede ser incluido dentro del archivo de documento, pero se almacena normalmente en un archivo separado de ASCII-text. La sintaxis del DTD de SGML y de DTD de XML son muy similares, pero no idénticos

### XML - DTD

Un ejemplo de un muy simple DTD de XML para describir un lista de personas es dado a continuación:

```
<!ELEMENT people_list (person*)> <!ELEMENT person  
(name, birthdate?, gender?, socialsecuritynumber?)> <!ELEMENT name (#PCDATA) >  
<!ELEMENT birthdate (#PCDATA) > <!ELEMENT gender (#PCDATA) > <!ELEMENT  
socialsecuritynumber (#PCDATA) >
```

Tomando esto línea por línea, dice:

- Una "people\_list" es un elemento que contiene muchos elemertos "person". El "\*" denota que pueden haber 0, 1 o muchos elementos "person".
- Un elemento "person" contiene los elementos "name", "birthdate", "gender" y "socialsecuritynumber". El "?" indica que un elemento es opcional. El elemento "name" no tiene "?", entonces "person" debe contener un elemento "name".
- Un elemento "name" contiene información.
- Un elemento "birthdate" contiene información.
- Un elemento "gender" contiene información.
- Un elemento "socialsecuritynumber" contiene información.

### XML - DTD

Un ejemplo de un archivo XML, el cual usa el DTD

```
<?xml version="1.0" encoding="UTF-8"?>  
 <!DOCTYPE people_list SYSTEM "example.dtd">  
<people_list>  
   <person>  
     <name>Fred Bloggs</name>  
     <birthdate>27/11/2008</birthdate>  
     <gender>Male</gender>  
   </person>  
</people list>
```

## DTD. Entidades.

- En un DTD, una entidad es el equivalente a una constante en programación. Un valor fijo que utilizamos en nuestro código.
- Declaración:  

```
<!ENTITY nombre-entidad "texto de reemplazo">
```
- Uso:  

```
&nombre-entidad;
```

- **Ejemplo:**

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE fichas [
    <!ENTITY poblacion "Tejeda de Tiétar, Cáceres">
]>
<fichas>
    <ficha>
        <empresa nombre="Tractores Toledo"></empresa>
        <direccion>&poblacion;</direccion>
    </ficha>
</fichas>
```

- Declaración entidad externa privada:  

```
<!ENTITY provincia SYSTEM "provincia.txt">
```
- Uso:  

```
<poblacion>&poblacion;</poblacion>
```

- **DTD. Elementos. Tipos de contenido.**

Tipo de contenido	Descripción
<b>EMPTY</b>	Indica que el elemento referenciado debe estar vacío.
<b>ANY</b>	Indica que el elemento referenciado puede contener cualquier contenido.
<b>(#PCDATA)</b>	Indica que el elemento referenciado puede tener datos de tipo carácter (Parsed Character Data).
<b>(nombreElemento)</b>	Indica que el elemento referenciado puede contener al elemento indicado.
<b>(nombreElemento1, nombreElemento2, ...)</b>	Indica que el elemento referenciado puede contener una sucesión de elementos indicados como una lista separada por comas.

- **DTD. Elementos. Cardinalidades.**

Notación	Descripción	Ejemplo
<b>(nombreElemento)</b>	Una única ocurrencia del elemento.	<!ELEMENT aviso (de)>
<b>(nombreElemento?)</b>	Cero o una única ocurrencia del elemento.	<!ELEMENT aviso (de?)>
<b>(nombreElemento+)</b>	Una o más ocurrencias del elemento.	<!ELEMENT aviso (mensaje+)>
<b>(nombreElemento*)</b>	Cero o más ocurrencias del elemento.	<!ELEMENT aviso (mensaje*)>
<b>(nombreElemento1, nombreElemento2,...)</b>	Debe contener todos los elementos de la lista.	<!ELEMENT aviso (de, para, mensaje)>
<b>(nombreElemento1   nombreElemento2)</b>	Debe contener uno u otro elemento	<!ELEMENTO aviso (de   para, mensaje)>

## DTD. Atributos.

- Los DTD permiten restringir las reglas y condiciones de los atributos usados en los documentos XML que validan.
- Sintaxis:

```
<!ATTLIST elemento nombre-atributo tipo-atributo valor-atributo>
```

Tipo	Descripción
<b>CDATA</b>	Cadena de caracteres.
<b>(valor1   valor2   ...)</b>	Una lista de posibles valores.
<b>ID</b>	Un identificador único
<b>IDREF</b>	Una referencia a un identificador único de otro elemento
<b>IDREFS</b>	Una lista de referencias a identificadores de otros elementos
<b>NMTOKEN</b>	Un nombre XML válido.
<b>NMTOKENS</b>	Una lista de nombres XML válidos separadas por espacios.
<b>ENTITY</b>	Una referencia a una entidad
<b>ENTITIES</b>	Una referencia a un conjunto de entidades
<b>NOTATION</b>	Un nombre de una notación
<b>xml:lang</b>	Indica el idioma del contenido.
<b>xml:space</b>	Indica qué hacer con espacios, tabulaciones y retornos de carro múltiples.

Valor	Descripción
<b>valor</b>	El valor por defecto del atributo
<b>#REQUIRED</b>	Indica que el atributo es obligatorio
<b>#IMPLIED</b>	Indica que el atributo es opcional
<b>#FIXED valor</b>	El valor del atributo

Si queremos incluir en el ejemplo del CV, en la etiqueta idioma un atributo que indique el nivel (bajo, medio, alto), incluiremos el elemento en el DTD:

```
<!ELEMENT idioma (#PCDATA)>
<!ATTLIST idioma nivel (bajo | medio | alto) "medio">
```