

## Lenguajes para el almacenamiento y la transmisión de la información



1

## Tipos de Lenguajes

Los lenguajes para el almacenamiento y la transmisión de información se pueden dividir en dos tipos:

- De marcas: **XML** (*eXtended Markup Language*) es un metalenguaje extensible de etiquetas desarrollado por el W3C que permite definir la gramática de lenguajes específicos.
- De listas: **JSON** (*JavaScript Object Notation*) es un formato ligero de intercambio de datos cuya simplicidad le ha permitido convertirse en alternativa a XML en AJAX.

2

## Definición de XML (1)

**XML** (*eXtended Markup Language* o Lenguaje de Marcado Extensible) proviene de SGML y puede considerarse un subconjunto de éste mucho más simple.

Se puede definir XML como metalenguaje que define las reglas para crear otros lenguajes. Un ejemplo de ello es XHTML que es una definición de HTML utilizando las reglas de XML.

La importancia de XML reside en que diversos equipos y aplicaciones lo pueden generar y leer fácilmente, de manera que puede utilizarse para compartir datos entre diferentes sistemas.

3

## Definición de XML (2)

### Ejemplo:

Una editorial suministra libros a diversas librerías, tanto la editorial como las librerías deberán disponer de su catálogo de libros, como cada una utiliza su propio sistema deberán introducir la misma información en cada uno de los sistemas.

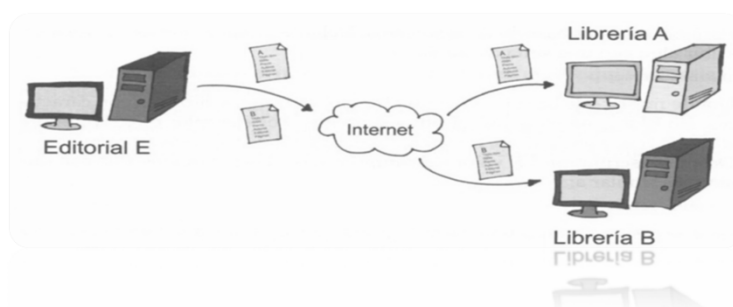


4

## Definición de XML (3)

### Ejemplo:

Dado que la editorial ya dispone de la información, lo ideal sería que pudiera facilitar esa información a las librerías en un archivo o que éstas pudieran descargarla de la web de la editorial.



5

## Definición de XML (4)

### Ejemplo:

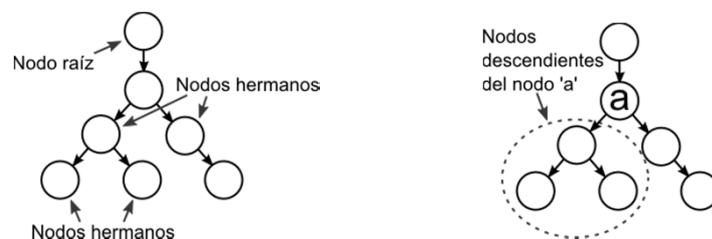
Para ello es necesario almacenar la información utilizando un lenguaje sencillo que cualquier aplicación pueda leer: XML.

```
<catalogo>
  <libro>
    <titulo>Lenguajes de marcas</titulo>
    <isbn>9788415452171</isbn>
    <autor>Juan Manuel Castro Ramos</autor>
    <autor>Jose Ramón Rodríguez Sánchez</autor>
    <editorial>Garceta</editorial>
  </libro>
  ...
</catalogo>
```

6

## Estructura y sintaxis de XML (1)

Los documentos XML se estructuran de forma jerárquica en la que los elementos se relacionan entre sí mediante relaciones de padres, hijos, hermanos, ascendentes, descendentes, etc. es lo que se conoce como el árbol del documento XML.



7

## Estructura y sintaxis de XML (2)

### **Declaración XML**

Los documentos XML deben comenzar con una declaración que permita indicar que el documento es XML, su versión y otras características como codificación de caracteres, si se procesa independientemente o requiere de archivos externos, etc.

Se trata de una instrucción de procesamiento que no forma parte del contenido XML pero proporciona información a las aplicaciones sobre cómo procesar el documento. Estas instrucciones van encerradas entre **<? y ?>**.

8

## Estructura y sintaxis de XML (3)

### Declaración XML

La forma más simple de esta declaración es:

```
<?xml version="1.0"?>
```

Además puede establecer otras propiedades:

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

- **version:** indica la versión de XML. Aunque existe la versión 1.1, la más extendida sigue siendo la 1.0.
- **encoding:** establece la codificación de caracteres, siendo utf-8 la más extendida.
- **standalone:** indica si el documento está listo para procesarse independientemente o requiere de archivos externos, por defecto es "no".

9

## Estructura y sintaxis de XML (4)

### Elementos

El elemento es la unidad básica de un documento XML y se compone de:

- Etiqueta de apertura: `<etiqueta>`
- Contenido: textual, otro elemento o nulo.
- Etiqueta de cierre: `</etiqueta>`

Ejemplos:

```
<modulo>Lenguajes de marcas</modulo>
```

```
<libro>  
  <titulo>Lenguajes de marcas</titulo>  
  <autor>Juan Manuel Castro Ramos</autor>  
</libro>
```

10

## Estructura y sintaxis de XML (5)

### **Elementos: nombres XML**

Todos los elementos deben tener un nombre que ha de cumplir unas normas:

- Puede contener letras, dígitos, guiones, guiones bajos, comas y dos puntos.
- Aunque puede empezar por letra, guión bajo o dos puntos, se aconseja que sea por letra.
- Son *case-sensitive*, así que se recomienda utilizar minúsculas para evitar confusión.
- No puede contener caracteres de espaciado.
- Los nombres que comienzan por las letras XML en cualquier combinación de mayúsculas y minúsculas se reservan para estandarización.

11

## Estructura y sintaxis de XML (6)

### **Elementos: atributos**

Los elementos pueden contener atributos que proporcionen información adicional.

Los atributos son pares nombre-valor que se sitúan en la etiqueta de apertura del elemento:

```
<etiqueta atributo1="valor1">Contenido</etiqueta>
```

Para los nombres se siguen las mismas normas que para los nombres de los elementos, y los valores deben ir encerrados entre comillas simples o dobles.

```
<distancia unidades="km">100</distancia>
```

12

## Estructura y sintaxis de XML (7)

### **Elementos: elementos vs atributos**

En muchas ocasiones la información se podrá almacenar como elemento o como atributo.

Se recomienda utilizar elementos siempre que sea posible teniendo en cuenta las siguientes propiedades:

#### ■ **Elementos:**

- Se emplean para representar jerarquías.
- Se pueden extender con otros elementos.
- El orden en que aparecen es representativo.
- Pueden tener atributos.
- Puede haber múltiples ocurrencias de un atributo.

13

## Estructura y sintaxis de XML (8)

### **Elementos: elementos vs atributos**

#### ■ **Atributos:**

- Son modificadores de la información.
- Van asociados a los elementos.
- No se pueden extender con otros elementos o atributos.
- El orden en que aparecen dentro del elemento no es representativo.
- No puede haber múltiples ocurrencias de un atributo dentro del mismo elemento.

14

## Estructura y sintaxis de XML (9)

### Elementos: contenido

Los elementos pueden contener texto, otro elemento o estar vacíos.

- **Otro elemento:** los elementos se anidan unos dentro de otros formando una estructura jerárquica. Como ocurre en HTML las etiquetas deberán cerrarse en orden inverso al que se abrieron.
- **Texto:** representa información de un elemento y puede contener cualquier carácter aunque para algunos hay que utilizar su entidad:

Carácter	Entidad	Carácter	Entidad
&	&amp;	'	&apos;
<	&lt;	"	&quot;
>	&gt;		

## Estructura y sintaxis de XML (10)

### Elementos: contenido

- **Secciones CDATA:** son conjuntos de caracteres que el procesador no debe analizar permitiendo introducir libremente cualquier tipo de carácter dentro del documento.

Para crear una sección CDATA se utiliza el formato:

**<![CDATA[contenido]]>** .

```
<codigo>
  <![CDATA[
    <html><body><h2>Título de la página</h2></body></html>
  ]]>
</codigo>
```

De esta forma podemos introducir cualquier carácter sin necesidad de utilizar entidades.



## Estructura y sintaxis de XML (11)

### Elementos: contenido

- **Elementos sin contenido:** existe la posibilidad de tener elementos sin contenido.

Los elementos sin contenido pueden tener atributos y se abren y cierran en una sola etiqueta.

```
<separador/>
```

```
<separador ancho="80"/>
```

17

## Estructura y sintaxis de XML (12)

### Elementos: elemento raíz

Todo documento XML debe contener un único elemento raíz, también llamado elemento documento, que contiene a todos los demás elementos.

Para añadir el elemento raíz basta con identificar los elementos principales y buscar un nombre genérico para ellos.

```
<catalogo>  
  <libro> ... </libro>  
  <libro> ... </libro>  
</catalogo>
```

18

## Estructura y sintaxis de XML (13)

### Comentarios

Los documentos XML pueden contener comentarios que servirán para aclarar el contenido pero serán ignorados por el procesador y las aplicaciones.

Los comentarios pueden situarse en cualquier lugar del documento excepto dentro de las etiquetas.

**<!-- comentario -->**

Dentro del comentario se puede utilizar cualquier carácter excepto doble guión (--) ya que podría ser confundido con el cierre.

19

## Estructura y sintaxis de XML (14)

### Instrucciones de procesamiento

Las instrucciones de procesamiento no forman parte del contenido del documento pero proporcionan información a las aplicaciones que procesan el XML.

- **Declaración:** es obligatoria. Se utiliza para indicar la versión de XML y otras características.
- **Visualización:** XML no dispone de una visualización concreta en el navegador puesto que refleja datos y no presentación, pero podemos establecer la forma de representar el documento mediante una hoja de estilos (CSS) o una hoja de transformaciones (XSLT).

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

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

## Espacios de nombres (1)

La especificación del espacio de nombres es un mecanismo que permite evitar los conflictos de nombres cuando en un documento XML existen elementos o atributos con el mismo nombre pero diferente definición.

### ■ Declaración

Se declaran como atributo de un elemento:

<elemento xmlns:prefijo="URI\_espacio\_nombres">

- Prefijo: sirve como referencia. Puede ser cualquier texto pero se recomienda que sea descriptivo.
- URI: debe ser único aunque no es más que un identificador que no se comprueba mediante conexión alguna.

21

## Espacios de nombres (2)

### ■ Utilización

Se antepone el prefijo definido en la declaración y dos puntos (:) al nombre del elemento o atributo.

<prefijo:elemento prefijo:atributo="valor">

### ■ Ámbito de aplicación y uso

El espacio de nombres puede ser referenciado anteponiendo su prefijo en el elemento en que se ha declarado y en sus descendientes.

También se puede declarar un espacio de nombres nuevo para un elemento descendiente de otro en el que ya se había declarado un espacio de nombres.

22

## Espacios de nombres (3)

```
<h:html xmlns:xdc="http://www.xml.com/libros"
        xmlns:h="http://www.w3.org/TR/html5/">
  <h:head> <h:title>Ficha del libro</h:title> </h:head>
  <h:body>
    <xdc:ficha_libro>
      <xdc:titulo>XML Edición 2012</xdc:titulo>
      <h:table>
        <h:tr> <h:td>Autor</h:td><h:td>Precio</h:td> </h:tr>
        <h:tr>
          <h:td> <xdc:autor>M.A. Acera</xdc:author> </h:td>
          <h:td> <xdc:precio>39.95</xdc:precio></h:td>
        </h:tr>
      </h:table>
    </xdc:ficha_libro>
  </h:body>
</h:html>
```

23

## Espacios de nombres (4)

Los atributos pueden pertenecer al mismo espacio de nombres al que pertenece el elemento al que están asociados o a otro diferente.

Los atributos que no tienen ningún prefijo no pertenecen a ningún espacio de nombres.

Dos atributos con el mismo nombre pero distinto prefijo son diferentes y pueden estar asociados al mismo elemento.

```
<emp:empleado xmlns:emp="empresa:espacios:emp">
  <emp:datosPersonales trabajo:empno="1234"
    xmlns:trabajo="empresa:espacios:trabajo">
    <emp:departamento depto="10">Ventas</emp:departamento>
    <emp:nombre>Juan</emp:nombre>
    <emp:apellido>Pérez</emp:apellido>
  </emp:datosPersonales>
</emp:empleado>
```

## Espacios de nombres (5)

### ■ Espacio de nombres por defecto

Cuando en la declaración de un espacio de nombres no se especifica un prefijo estamos definiendo un espacio de nombres por defecto y su ámbito de aplicación se extiende al elemento en que se ha declarado y a todos sus descendientes, pero no a sus atributos.

```
<empleado xmlns="empresa:espacios:emp"
           xmlns:trabajo="empresa:espacios:trabajo">
  <datosPersonales trabajo:empno="1234" empno="e_1234">
    <departamento depto="10">Ventas</departamento>
    <nombre>Juan</nombre>
    <apellido>Pérez</apellido>
  </datosPersonales>
</empleado>
```

25

## Espacios de nombres (6)

### ■ Espacio de nombres por defecto

Se puede desasignar un elemento de un espacio de nombres por defecto asignando el valor vacío al atributo xmlns:

<elemento xmlns="">

```
<empleado xmlns="empresa:espacios:emp"
           xmlns:trabajo="empresa:espacios:trabajo">
  <datosPersonales trabajo:empno="1234" empno="e_1234">
    <departamento xmlns="" depto="10">Ventas</departamento>
    <nombre>Juan</nombre>
    <apellido>Pérez</apellido>
  </datosPersonales>
</empleado>
```

26

## Documentos XML bien formados (1)

La especificación de XML define la sintaxis que debe seguir todo documento:

- Cómo se delimitan los elementos con etiquetas.
- Qué formato puede tener una etiqueta.
- Qué nombres son aceptables para los elementos.
- Dónde se colocan los atributos.
- Etc.

Se dice que un documento XML está **bien formado** cuando cumple las normas establecidas por el estándar del W3C para el XML.

27


## Documentos XML bien formados (2)

Las reglas más importantes son las siguientes:

- El documento debe empezar por una instrucción de procesamiento que indique la versión del XML y opcionalmente la codificación de caracteres (por defecto UTF-8) y si está listo para ser procesado de forma independiente (por defecto NO).  
`<?xml version="1.0" encoding="utf-8" standalone="yes"?>`
- Debe existir un único elemento raíz que tendrá como descendientes a todos los demás elementos.
- Los elementos que no sean vacíos deben tener una etiqueta de apertura y otra de cierre, que tienen el mismo nombre pero a la de cierre se le antepone /.  
`<etiqueta>....</etiqueta>`

28

### Documentos XML bien formados (3)

- Los elementos vacíos se cierran en la apertura.  
`<etiqueta [atributos]/>`
- Los elementos deben aparecer bien anidados de manera que su cierre se produzca en orden inverso a su apertura.  
`<alfa> <beta> <gamma> ... </gamma> </beta> </alfa>`
- Los nombres de elementos y atributos son sensibles a mayúsculas/minúsculas.
- Los valores de los atributos deben aparecer entre comillas, simples o dobles pero del mismo tipo.  
`<etiqueta atributo="valor"/>` ó `<etiqueta atributo='valor'/>`
- No puede haber dos atributos con el mismo nombre asociados al mismo elemento.

29

### Documentos XML bien formados (4)

- No se pueden introducir instrucciones de procesamiento ni comentarios dentro de las etiquetas de apertura y cierre de un elemento.
- No puede haber nada antes de la instrucción de procesamiento de declaración `<?xml ... ?>`.
- No puede haber texto antes ni después del elemento documento.
- No pueden aparecer los símbolos (`<`, `>`, `&`) en el contenido de los elementos ni en el valor de los atributos.

Una manera de verificar un documento XML es abrirlo en el navegador y comprobar que se muestra correctamente el árbol de nodos.

30

## Herramientas XML (1)

### ■ Editor

Podemos utilizar cualquier editor de texto para editar archivos XML: notepad, notepad++, etc. Si bien existen editores especializados que pueden facilitarnos el trabajo:

- Altova XMLSpy: editor XML, XSLT, Xquery, etc. Dispone de representaciones gráficas para los distintos documentos directamente editables.
- <Oxygen/>: también dispone de editor para las diferentes tecnologías (XML, XSD, XSLT, XQuery) y una interfaz gráfica de desarrollo muy cuidada.
- XML Copy Editor: sencillo editor XML, XSD, XSLT, etc. libre bajo licencia GNU GPL.

31

## Herramientas XML (2)

### ■ Analizador

Un analizador o *parser* es un procesador que lee un documento XML, determina la estructura y propiedades de los datos que contiene comprobando que esté bien formado y genera el árbol jerárquico asociado, lo que nos permite ver los datos en un navegador o ser tratados por cualquier aplicación.

### ■ Validador

Analiza el documento y comprueba que cumple las reglas establecidas en el DTD o esquema correspondiente comprobando su semántica e informando de los errores existentes.

Ejemplo: <http://www.xmlvalidation.com/>

32