

# LENGUAJES DE MARCAS Y SISTEMAS DE GESTIÓN DE INFORMACIÓN

Bloque XML: UD2, Sintaxis XML

# Índice

2

- Introducción
- Estructura en árbol
- Reglas XML
- Elementos XML
- Atributos XML
- Reglas de nombrado y buenas prácticas
- Bien formado y válido

# Introducción

3

- Vamos a seguir la introducción en:  
[https://www.w3schools.com/xml/xml\\_what\\_is.asp](https://www.w3schools.com/xml/xml_what_is.asp)
- Ideas básicas:
  - XML : EXtensible Markup Language
  - Para organizar, almacenar y transportar información.
  - Las etiquetas XML (tags) no están predefinidas.
    - En HTML sí: solo puede usar las etiquetas definidas en el estándar (o definidas por el navegador). Si utiliza una que no conoce, la ignora.

# Introducción (II)

- Más ideas básicas:
  - XML permite definir nuestras propias etiquetas y la estructura del documento
  - Un documento XML NO hace nada.
  - XML está diseñado para ser autodescriptivo (si se usa bien, claro..)
  - Es una recomendación del W3C

# Fichero XML

5

- La extensión del archivo debe ser .xml
- Declaración inicial
- Elemento raíz, etiqueta de inicio y cierre

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

```
<note>
```

```
  <to>Tove</to>
```

```
  <from>Jani</from>
```

```
  <heading>Reminder</heading>
```

```
  <body>Don't forget me this week!</body>
```

```
</note>
```

# Estructura en árbol

6

- Deben contener un único elemento raíz (root), padre de todos los demás.
- El árbol comienza en este nodo raíz.
- Los elementos pueden contener
  - contenido textual
  - otros elementos (se convierten en elementos contenedores).
- Además un elemento pueden contener cero o más atributos
  - siempre en la etiqueta de inicio
  - formato con asignación y comillas
    - `nombreAtributo="valor Atributo"`

# Ejemplo bookstore

7

Atributo, dentro etiqueta elemento:  
Nombre\_atributo="valor\_atributo"

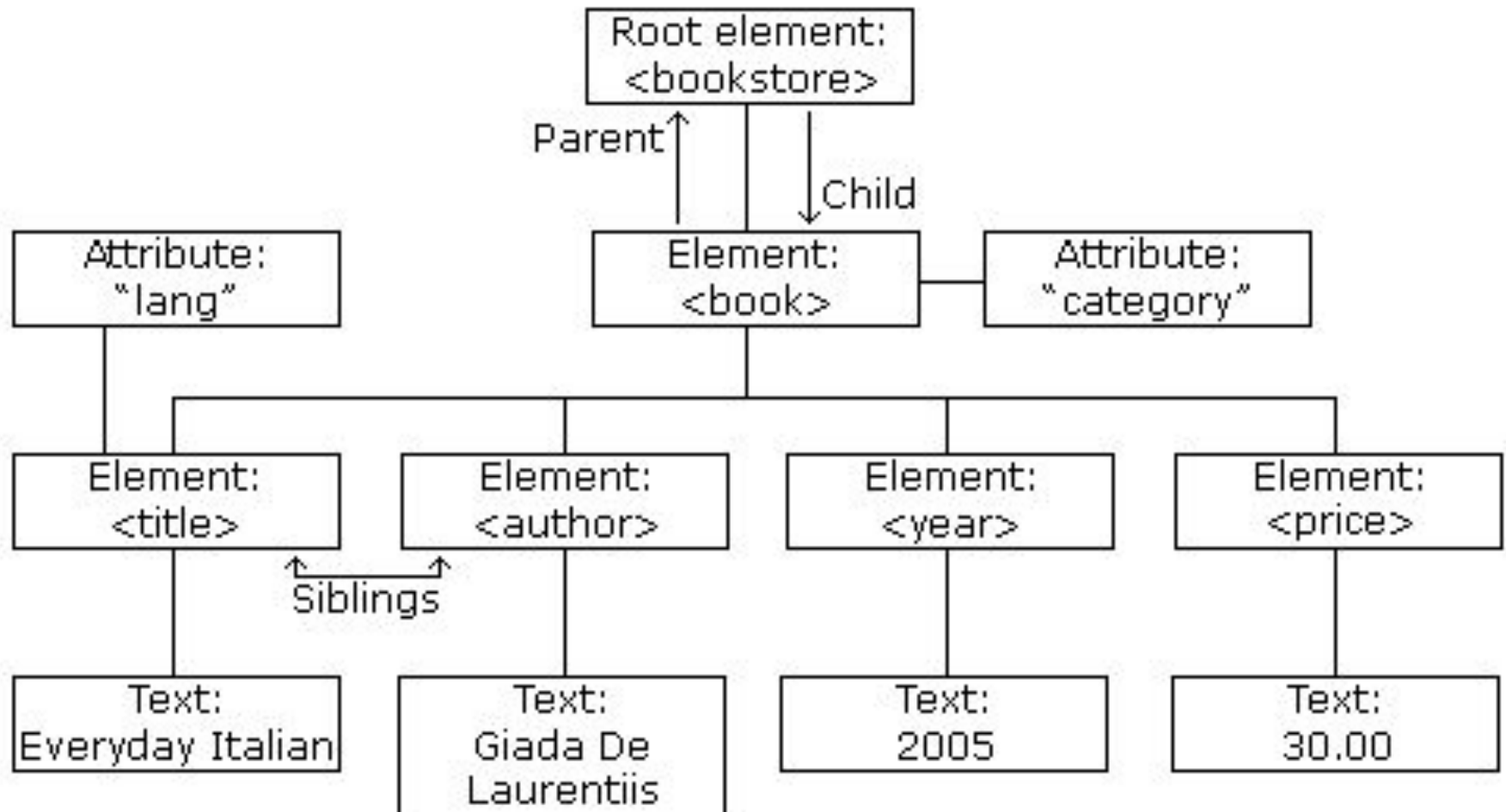
```
<bookstore>
  <book category="COOKING" priority="1">
    <title lang="en">Everyday Italian</title>
    <author>Giada De Laurentiis</author>
    <year>2005</year>
    <price>30.00</price>
  </book>
  <book category="CHILDREN">
    <title lang="en">Harry Potter</title>
    <author>J K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
</bookstore>
```

book contenedor de  
elementos

author solo contenido

# Árbol que le corresponde

8





# Reglas sintaxis XML

9

- **Elementos deben tener etiqueta de cierre**
  - `<p>This is another paragraph</p>`
- **Etiquetas distinguen mayúsculas/minúsculas: (Case Sensitive)**
  - `<Mmessage>This is incorrect</message>`  
`<message>This is correct</message>`
- **Elementos correctamente anidados**
  - `<b><i>This text is bold and italic</b></i>` Mal  
`<b><i>This text is bold and italic</i></b>` Bien
- **Sólo un único elemento raíz**
- **Los atributos deben estar delimitados**
  - `<note date=12/11/2007>` XML incorrecto
  - `<note date="12/11/2007">` XML correcto

# Reglas de nombrado XML

10

- Los nombre de los elementos
  - No pueden comenzar con número o carácter de puntuación
  - Pueden contener letras, números y otros caracteres..
  - Los nombres no pueden contener espacios
  - Cualquier nombre puede ser usado, no hay palabras reservadas.

# Buenas prácticas

11

- Los nombres deben ser significativos.
- Nombre compuestos: con subrayado o notación camel:  
    <primerr\_apellido> o <primerrApellido>.
- En lo posible cortos y simples.  
    <book\_title>      ~~<the\_title\_of\_the\_book>~~.
- Evita el guión "-" o el punto "." Se puede confundir con la resta, objetos, propiedades.
- Evita los dos puntos ":" reservados para espacios de nombrado.
- Si el XML se corresponden con una tabla (BD) => seguir las reglas de nombrado de la base de datos en XML.
- Los caracteres no ingleses son válidos, pero ten cuidado con los problemas que podría generar en SW de terceros.

# Atributos

12

- Proporcionan información adicional sobre el elemento.
- Los valores de los atributos deben estar delimitados.
- Puedes usar comilla simple o comilla doble.
- Si el valor del atributo contiene dobles comillas, usar las simples:
  - `<gangster name='George "Shotgun" Ziegler'>`
    - o usar las entidades carácter predefinidas:
  - `<gangster name="George &quot;Shotgun&quot; Ziegler">`

# ¿Atributos o elementos?

13

- No hay reglas fijas para decidir cuándo usar atributos o elementos
- Aunque en HTML se usan mucho, en XML es recomendable no abusar de ellos, y usar elementos en su lugar.
- Inconvenientes de los atributos:
  - No pueden contener valores múltiples
  - No pueden contener estructuras en árbol
  - No se pueden “expandir” o extender con facilidad en el futuro
- En general los atributos son difíciles de leer y mantener.
- Usa:
  - Elementos para los datos
  - Atributos para la información que no es relevante para los datos.
    - Es decir, los metadata (datos sobre los datos) deberían ser atributos, el resto elementos.

# Elemento vs atributo

14

- Ejemplo con atributo:

```
<note date="10/01/2008">  
  <to>Tove</to>  
  <from>Jani</from>  
  <heading>Reminder</heading>  
  <body>Don't forget me this week!</body>  
</note>
```

- Ejemplo con elemento:

```
<note>  
  <date>10/01/2008</date>  
  <to>Tove</to>  
  <from>Jani</from>  
  <heading>Reminder</heading>  
  <body>Don't forget me this week!</body>  
</note>
```

Incluso en el futuro  
puede que date  
tenga 3 elementos..

# Aclaraciones sobre atributos

15

- Siempre asociado a un elemento
- Se define dentro de la etiqueta de inicio de dicho elemento
  - `<book category="CHILDREN" > ..</book>`
- Elemento puede tener cero o más atributos  
`<book categ="CHILDREN" local="almacen1">`  
`..</book>`
- Se recomienda no abusar

# Editores XML

16

- VS Code
  - con plugin xml de Red Hat
- curso pasados: XMLCopyEditor:
  - liviano, libre orientado sólo a XML. Versiones para Windows y Linux  
<http://xml-copy-editor.sourceforge.net/>
  - Muy simple, sólo lo necesario para el curso. Comprueba bien formado, válido, etc.
- Otros de pago
  - Altova XMLSpy, <oXygen/> XML Editor



# Validación vs bien formado

17

- Bien formado: sintaxis correcta.
  - Las 5 reglas: root, closing tags, case sensitive, nested y quoted attributes
  - Reglas de nombrado
- Válido: contrastada el fichero XML con un documento externo donde se especifica nombres de los elementos, atributos posibles, repeticiones, estructura del documento, etc.
  - se usana ficheros DTD o schema XML => en tema posterior.
- Errores en sintaxis DETIENEN el procesamiento del documento.

# Bien formado: pruebas

18

[http://w3schools.com/xml/xml\\_validator.asp](http://w3schools.com/xml/xml_validator.asp)

- Añadir más de un nodo raíz.
- Olvidar cerrar un elemento
- Etiqueta de cierre con mayúsculas o minúsculas
- Anidar mal elementos
- No entrecomillar atributos
- Probar en validador o en local abriendo con navegador => siguiente..

# Bien formado: herramientas

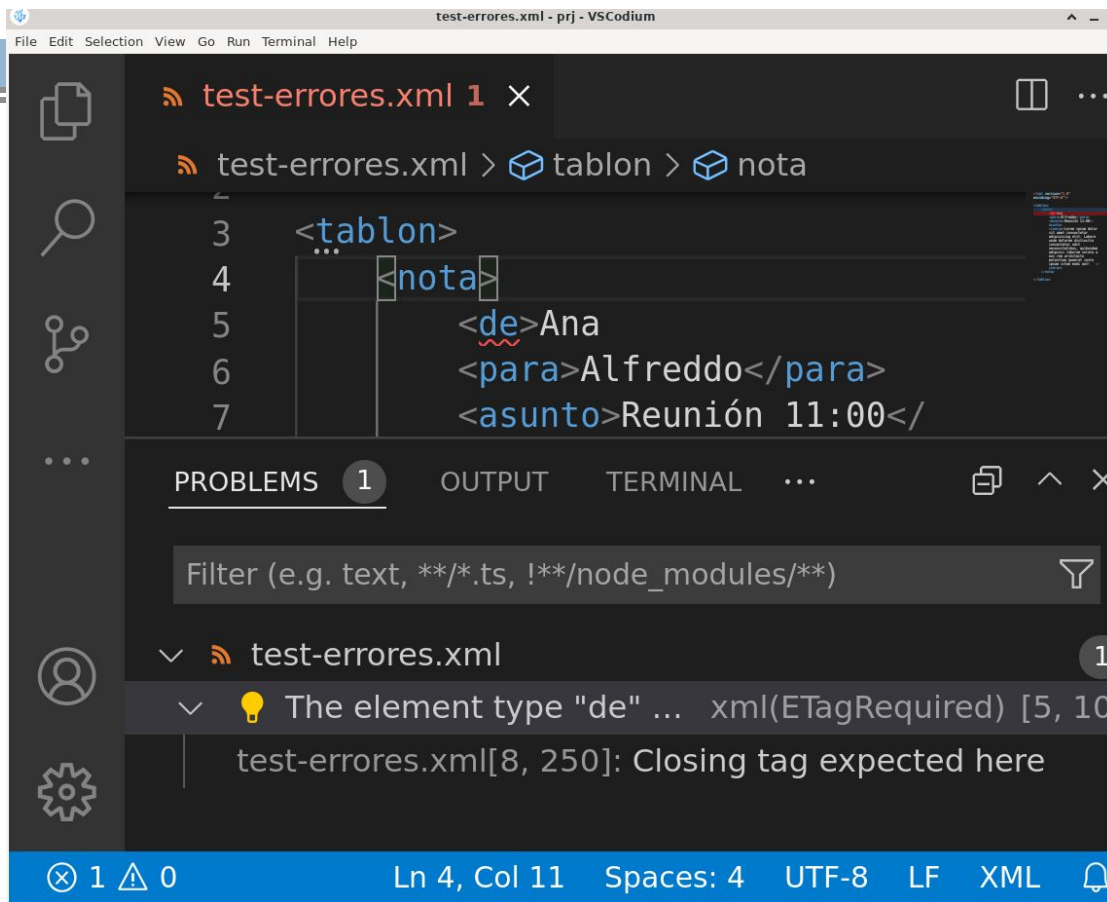
19

- El propio editor
  - VScode y complemento xml (Red Hat)
  - xmllcopyeditor (este curso no..)
    - al guardar los cambios
    - o al pulsar F2
- Navegador web:
  - al intentar visualizarlo
- Línea de comandos: xmlstarlet

```
$ xmlstarlet val -e fichero.xml
```

# Errores sintácticos

20



The screenshot shows the VS Code editor interface. The main editor window displays an XML file named `test-errores.xml` with the following content:

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <tablon>
3   <nota>
4     <de>Ana
5     <para>Alfredo</para>
6     <asunto>Reunión 11:00</
7
```

The cursor is positioned at the end of line 7. The `<de>` tag on line 5 is highlighted with a red squiggly line, indicating an error. The `PROBLEMS` panel at the bottom shows one error:

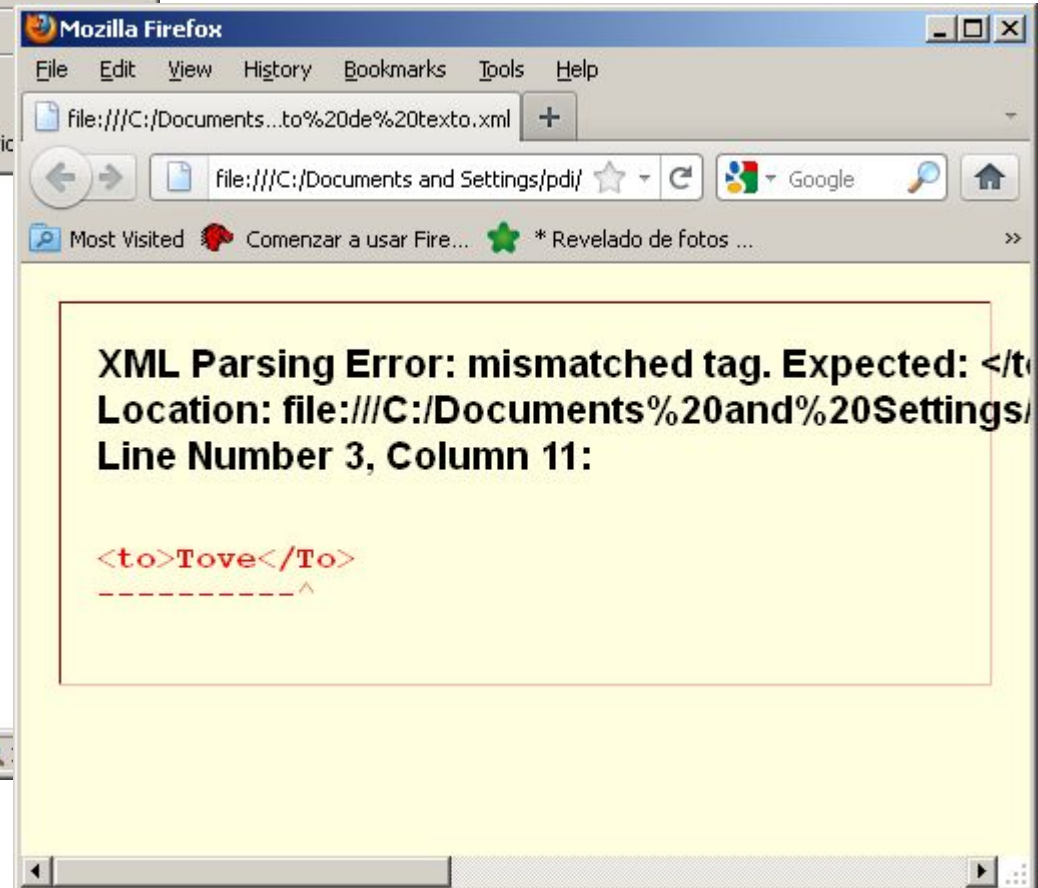
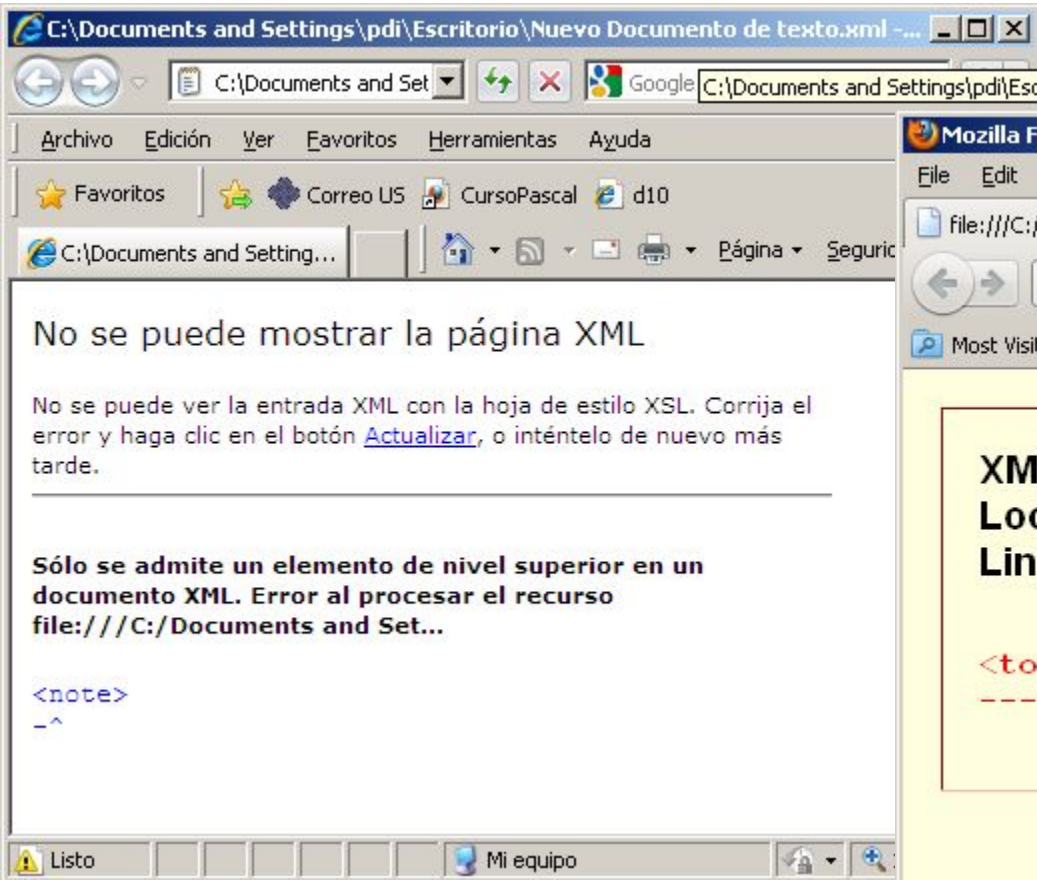
- test-errores.xml: The element type "de" ... xml(ETagRequired) [5, 10]

The status bar at the bottom indicates the current position is Ln 4, Col 11, with 4 spaces, UTF-8 encoding, LF line endings, and XML document type.

```
usuario@xdebian11:~/prj$ xmlstarlet val -e test-errores.xml
test-errores.xml:9.12: Opening and ending tag mismatch: de line 5 and nota
</nota>
^
test-errores.xml - invalid
usuario@xdebian11:~/prj$
```

# Errores sintácticos

21



# Viendo XML en navegador

22

- Con versiones o distintos navegadores => resultados similares..
  - El doc XML se mostrará con los elementos raíz e hijos "coloreados".
  - Símbolos para contraer/expandir la estructura.
  - Muestran o no la declaración de fichero xml
- Para ver el fuente XML: "View Page Source" or "View Source".
- Si hay errores, el navegador muestra el error
  - Ojo, algunos lo ocultan en la consola
- **Nota:** en algunos navegadores solo se muestran los elementos de texto: Si usamos el inspector de elementos (F12) vemos la transformación por defecto que hace el navegador a HTML y CSS para presentarlo.

# Viendo XML en navegador (II)

23

- Los docs XML no llevan información sobre presentación
  - los navegadores los muestran “tal cual” (algunos avisan de que no hay estilo asociado)
- Soluciones para presentación:
  - Ejemplo presentación XML **con CSS**:  
[https://www.w3schools.com/xml/cd\\_catalog\\_with\\_css.xml](https://www.w3schools.com/xml/cd_catalog_with_css.xml)
  - Ejemplo transformación de XML **con XSLT** a HTML:
    - <https://www.w3schools.com/xml/simplexsl.xml>
  - Desde lenguajes de programación: JavaScript, PHP, etc.
- Ejemplos ficheros XML
  - [http://w3schools.com/xml/cd\\_catalog.xml](http://w3schools.com/xml/cd_catalog.xml)
  - [http://w3schools.com/xml/plant\\_catalog.xml](http://w3schools.com/xml/plant_catalog.xml)
  - <http://w3schools.com/xml/simple.xml>

# Codificación

24

- La codificación por defecto es UTF-8.
  - Comprobaremos la codificación del editor
- Usar en declaración inicial atributo encoding  
`<?xml version="1.0" encoding="UTF-8" ?>`
- Ejemplo a comprobar  
`<aviso fecha='hoy'>`  
    `<to>Tove </to>`  
    `<head> Ñ, ñ, á, é, í, ó, ú </head>`  
    `<body> Asunto con tildes... ¿correcto?`  
    `¡bravo!</body>`  
`</aviso>`



# Codificación..

25

- Buscar editor soporte cambios codificación
  - code en preferences, buscar coding
- Escribir el mismo texto en dos ficheros con utf-8 e iso88591
- Ver el contenido de los ficheros con hd

```
usuario@mdebian10:~/prj/xml/dtd$ file texto-01-utf8.txt
texto-01-utf8.txt: UTF-8 Unicode text
usuario@mdebian10:~/prj/xml/dtd$ file texto-01-iso88591.txt
texto-01-iso88591.txt: ISO-8859 text
usuario@mdebian10:~/prj/xml/dtd$ cat texto-01-utf8.txt
123áéíóú
usuario@mdebian10:~/prj/xml/dtd$ hd texto-01-utf8.txt
00000000  31 32 33 c3 a1 c3 a9 c3  ad c3 b3 c3 ba 0a
0000000e
usuario@mdebian10:~/prj/xml/dtd$ hd texto-01-iso88591.txt
00000000  31 32 33 e1 e9 ed f3 fa  0a
00000009
usuario@mdebian10:~/prj/xml/dtd$
```

# Estructura lógica (I)

26

- El fichero XML es una mezcla de
  - marcas
  - datos (y metadatos)
- Marcado
  - En el prólogo:
    - declaración XML, Instrucciones de Proceso (IP), declaración de tipo de documento, comentarios y espacios.
  - En la instancia de datos (dentro del alcance del elemento raíz):
    - Etiqueta de inicio y cierre, elementos vacíos, atributos, entidades, etc, delimitadores de secciones CDATA..
    - Datos tipo carácter (CDATA): todo texto que no es marca.

# Estructura lógica (II): prólogo

27

- Declaración XML, obligatoria, al inicio

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

  - Atributo opcional: encoding

- Comentarios: mismo formato que HTML

```
<!-- filename: prueba.xml -->
```

Para legibilidad, o deshabilitar secciones

- Doctype: declaración tipo documento ("plantilla")

```
<!DOCTYPE note SYSTEM "note.dtd">
```


- Instrucciones de proceso (IP), destinadas aplicación, p.e. presentación

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

en temas posteriores

# Estructura lógica (III); datos

28

- Tipos de elementos según contenido
  - Contenedor: Contenido son sólo elementos
  - Datos: sólo datos textuales
  - Contenido mixto: texto y elementos (**evitar**)  
`<libro> encontrado en aula`  
    `<titulo> XML y XSLT </titulo>`  
    `<autor> García, J. </autor>`  
`</libro>`  

  - Vacíos:
    - Declarado vacío: `<reserved />`
    - Sin contenido en ese momento : `<cost></cost>`

# Ejemplos

29

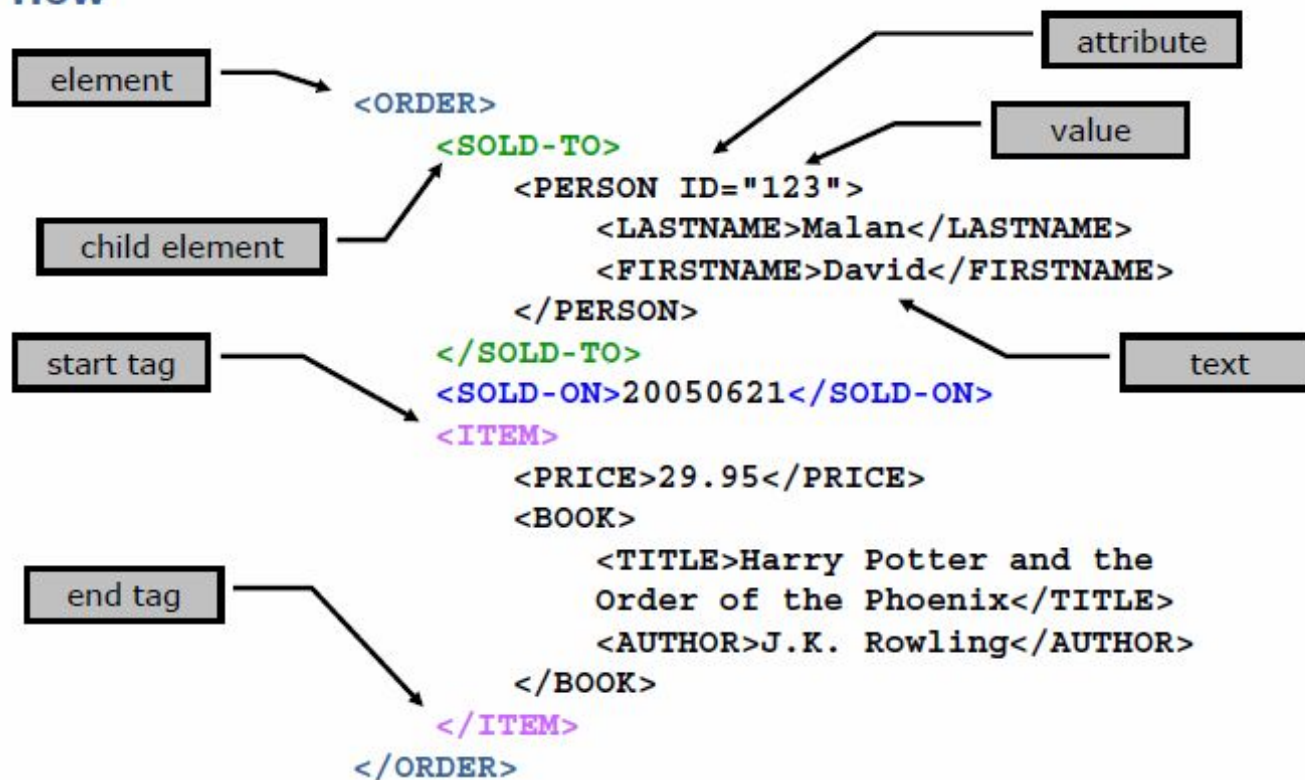
- Crear fichero xml con datos tablón de compra-venta.
  - Cada anuncio tendrá fecha, asunto, texto, precio de venta y contacto.
  - Cada anuncio tendrá un atributo que indique si se ha vendido ya o no, y el precio un atributo que indique la moneda de pago.
- Crear fichero xml con los datos de una clase.
  - Tendrá alumnos con nombre, fecha de nacimiento y mail.
  - Mail con atributo de nombre visible, con valores si o no.
- Comprobar que están bien formados mostrándolo en navegador
- Añadir a la clase las notas de cada módulo (basta poner un par de ellos) y de cada evaluación.

# XML... ¿extensible?

30

## XML

### How



# Si, con sentido común, claro

31

## XML

### How

```
<ORDER>
  <SOLD-TO>
    <PERSON ID="123">
      <LASTNAME>Malan</LASTNAME>
      <FIRSTNAME>David</FIRSTNAME>
      <INITIAL>J</INITIAL>
      <ADDRESS>
        <STREET>Oxford Street</STREET>
        <NUMBER>33</NUMBER>
        <CITY>Cambridge</CITY>
        <STATE>MA</STATE>
      </ADDRESS>
    </PERSON>
  </SOLD-TO>
  <SOLD-ON>20050621</SOLD-ON>
  <ITEM>
    ...
  </ITEM>
</ORDER>
```

extend structure  
without breaking  
existing data and  
applications

# Otras consideraciones

32

- Referencias de entidades:
  - Concepto + general, alias. 5 predefinidas

<code>&amp;lt;</code> <code>&amp;gt;</code> <code>&amp;amp;</code> <code>&amp;apos;</code> <code>&amp;quot;</code>	<code>&lt; less than</code> <code>&gt; greater than</code> <code>&amp; ampersand</code> <code>' apostrophe</code> <code>" quotation mark</code>	Nota: Sólo son ilegales <code>&lt;</code> y <code>&amp;</code> . Es recomendable sustituir <code>&gt;</code>
--	---	--

- Comentarios: delimitados por (como HTML):  
`<!--` *This is a comment* `-->`
- Espacios en blanco:
  - XML no trunca los espacios (HTML sí).
- Nueva línea: se almacena como LF.
  - En Windows “nueva línea” se almacena como dos caracteres, CR y LF. En Unix y Macintosh LF.



# Entidades predefinidas

33

- Hay algunos caracteres reservados porque el parser los procesa como parte del lenguaje XML:
  - Menor que (<)
  - Mayor que (>)
  - Comillas (“), Apostrofe (‘), Ampersand (&)
- Para insertar texto con estos caracteres XML tenemos tres opciones :
  - Entidades predefinidas, ya vistas.
  - Referencias numéricas (en hexadecimal..)
  - Secciones CDATA

# Referencias numéricas a carácter

34

- Para insertar símbolos de otras lenguas, o que no se pueden introducir por teclado.
- Dos tipos:
  - Decimal, `&#nnn;`; donde nnn es decimal asignado al carácter.
  - Hexadecimal, `&#xhhh;`; donde xhhh es valor hexadecimal asignado al carácter.

**Table 3.3** Common Decimal and Hexadecimal Character References

CHARACTER	DECIMAL CODE	HEXADECIMAL CODE	NAMED ENTITY	DISPLAY
Currency—Euro sign	<code>&amp;#8364;</code>	<code>&amp;#x20AC;</code>	<code>&amp;euro;</code>	€
Currency—Pound sterling sign	<code>&amp;#163;</code>	<code>&amp;#xA3;</code>	<code>&amp;pound;</code>	£
Currency—Yen (Yuan) sign	<code>&amp;#165;</code>	<code>&amp;#xA5;</code>	<code>&amp;yen;</code>	¥
Ampersand	<code>&amp;#38;</code>	<code>&amp;#x26;</code>	<code>&amp;amp;</code>	&
Less than sign (LH angle bracket)	<code>&amp;#60;</code>	<code>&amp;#x3C;</code>	<code>&amp;lt;</code>	<
Greater than sign (RH angle bracket)	<code>&amp;#62;</code>	<code>&amp;#x3E;</code>	<code>&amp;gt;</code>	>
Quotation mark	<code>&amp;#34;</code>	<code>&amp;#x22;</code>	<code>&amp;quot;</code>	"
Apostrophe	<code>&amp;#39;</code>	<code>&amp;#x27;</code>	<code>&amp;apos;</code>	'
Copyright symbol	<code>&amp;#169;</code>	<code>&amp;#xA9;</code>	<code>&amp;copy;</code>	©
Registered trademark symbol	<code>&amp;#174;</code>	<code>&amp;#xAE;</code>	<code>&amp;reg;</code>	®
En space (half as wide as it is tall)	<code>&amp;#8194;</code>	<code>&amp;#x2002;</code>	<code>&amp;ensp;</code>	n/a
Em space (as wide as it is tall)	<code>&amp;#8195;</code>	<code>&amp;#x2003;</code>	<code>&amp;emsp;</code>	n/a
Nonbreaking space	<code>&amp;#160;</code>	<code>&amp;#xA0;</code>	<code>&amp;nbsp;</code>	n/a
Horizontal tab	<code>&amp;#09;</code>	<code>&amp;#x09;</code>	<code>&amp;tab;</code>	n/a
Linefeed	<code>&amp;#10;</code>	<code>&amp;#xA;</code>	n/a	n/a
Carriage return	<code>&amp;#13;</code>	<code>&amp;#xD;</code>	n/a	n/a

# Secciones CData

35

- Los delimitadores de sección CData son
  - Inicio CData **<![CDATA[**
  - Final CData **]]>**
- Estas secciones le indican al parser que ignore el “marcado” de la sección, y pasar esos caracteres como texto a la aplicación.

- Ejemplo

```
<desc>
```

```
<![CDATA[
```

```
  <equipo> BMW & Lotus</equipo>
```

```
]]>
```

```
</desc>
```

# Que son ISO/W3C

36

- World Wide Web Consortium (W3C)
  - Fundado 1994 por [Tim Berners-Lee](#)
  - Sólo centrado en la web: HTML, CSS, DOM..
  - Y XML y toda su familia
- ISO: Organización Internacional para la Estandarización
  - Red de institutos de normas nacionales, por países
  - Normaliza casi de todo
  - 163 países, sede central en Ginebra
  - Normativa voluntaria. Libre acceso.. pagando.

# Ejemplos “libres”

37

- Crear un fichero series.xml. Libertad decisión.
  - Datos de cada serie: nombre, cadena que la emite, temporadas, nombre de los capítulos etc.
  - Para cada capítulo descripción y enlace de descarga.
  - En el enlace de descarga atributo indicando tipo de descarga: directa, emule, torrent..
- Visualizarlo en los navegadores
- Modificarlo para que no cumpla sintaxis
  - Más de un nodo raíz, No cerrar elementos, Anidar mal, Atributos no entrecomillados, Mayúsculas/minúsculas

# Ejemplos uso XML

38

- [SVG](#)
  - Ejemplo: [Tux](#)
- [Atom](#) (RSS)
  - Ejemplo: [blog de viajes](#) y Feedly
- [XSPF](#)
- [DocBook](#)
- Ficheros de configuración:
  - [Filezilla configuración](#)
  - [VmWare files](#)
- ¡¡ [Android](#) !!
- [GPX, or GPS Exchange Format, Recetas](#)

# Resumen

39

- Estructura de fichero xml
  - Prólogo
  - Instancia de datos: En árbol
- Reglas sintácticas
- Documento
  - bien formado
  - válido
- Buenas prácticas
- Herramientas: uso editor y navegador
- Organismos de estandarización: W3C, ISO