



# **UD1: INTRODUCCIÓN Y RECONOCIMIENTO DE LAS CARACTERÍSTICAS DE LOS LENGUAJES DE MARCAS**

LENGUAJES DE MARCAS Y SISTEMAS DE GESTIÓN DE INFORMACIÓN



**Arangoya**  
1974

Ander González Ortiz  
C.E. Arangoya  
[ander.gonzalez@arangoya.net](mailto:ander.gonzalez@arangoya.net)

# QUÉ ES XML

- **XML** (*eXtensible Markup Language*).
- Desarrollado por **W3C** (*World Wide Web Consortium*).
- Basado en **SGML** (*Standard Generalized Markup Language*).
- Utilizado para el almacenamiento e intercambio de datos estructurados entre distintas plataformas.
- Es un metalenguaje empleado para definir otros lenguajes, llamados dialectos XML: GML, MathML, RSS, SVG, XHTML...

# ELEMENTOS

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

```
<nombre>Elsa</nombre>
```

- Sintaxis:

```
<etiqueta>valor</etiqueta>
```

# ELEMENTOS VACÍOS

- Un elemento puede no contener ningún valor.

```
<etiqueta></etiqueta>  
<etiqueta/>
```

- **EJEMPLO**

```
<nombre></nombre>  
<nombre/>
```

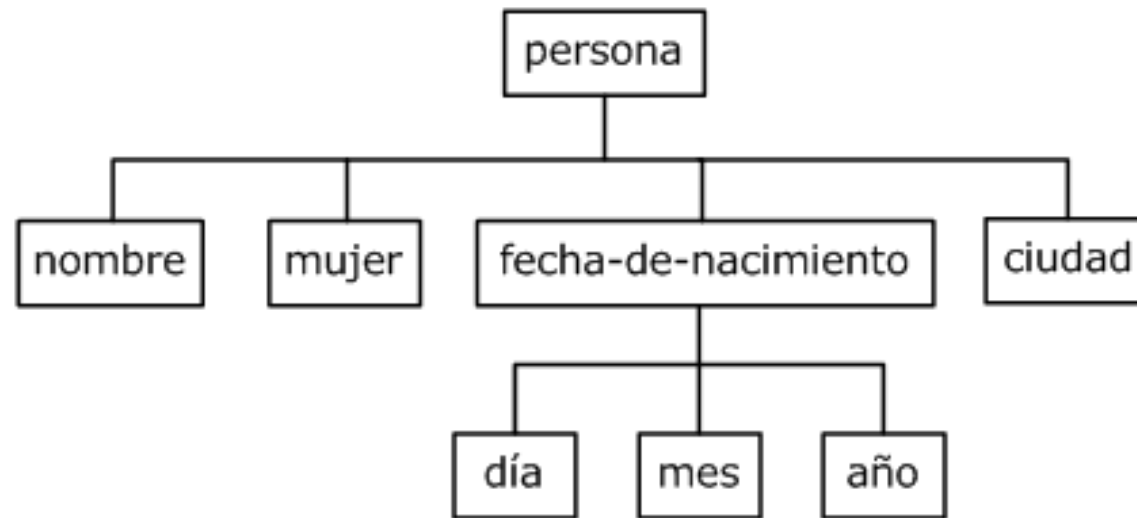
# RELACIONES PADRE-HIJO ENTRE ELEMENTOS

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

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

# ELEMENTO RAÍZ DE UN DOCUMENTO XML

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



- Los elementos son los que dan estructura semántica al documento.

# ELEMENTOS CON CONTENIDO MIXTO

- Un elemento puede contener contenido mixto, es decir, texto y otros elementos.

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

- El elemento “persona” contiene los elementos “nombre” y “ciudad”, además de los textos " **vive en** " y "."

# NORMAS DE SINTAXIS BÁSICAS

- Todos los nombres de los elementos son *case sensitive*.
- Pueden contener letras minúsculas, letras mayúsculas, números, *puntos “.”*, *guiones medios “-”* y *guiones bajos “\_”*.
- 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 BÁSICAS

- Detrás del nombre de una etiqueta se permite escribir un espacio en blanco o un salto de línea.

```
<ciudad >Pamplona</ciudad>
```

- No puede haber un salto de línea o un espacio en blanco antes del nombre de una etiqueta.

```
<  
ciudad>Pamplona</ ciudad>
```

# EJEMPLOS

## ELEMENTOS ESCRITOS INCORRECTAMENTE

<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>

# EJEMPLOS

## ELEMENTOS ESCRITOS CORRECTAMENTE

<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>

# NORMAS DE SINTAXIS BÁSICAS

- Las letras no inglesas (á, Á, ñ, Ñ...) están permitidas.
- Sin embargo, es recomendable no utilizarlas para reducir posibles incompatibilidades con programas que puedan no reconocerlas.
- Igualmente, se aconseja evitar el uso del carácter guion medio “–” y punto “.”

# ATRIBUTOS

- Un atributo proporciona información extra del elemento que lo contiene.

```
<producto codigo="G45">  
  <nombre color="negro" precio="12.56">Gorro de lana</nombre>  
</producto>
```

- Los valores de los atributos pueden escribirse entre comillas dobles (") o simples (').

# NORMAS DE SINTAXIS (ATRIBUTOS)

- Los nombres de los atributos deben cumplir las mismas normas de sintaxis que los nombres de los elementos.
- Además, todos los atributos de un elemento tienen que ser únicos. Por ejemplo, es incorrecto escribir:

```
<datos x="3" x="4" y="5"/>
```

- Sí es correcto escribir:

```
<datos x="3" X="4" y="5"/>
```

# DECLARACIÓN XML

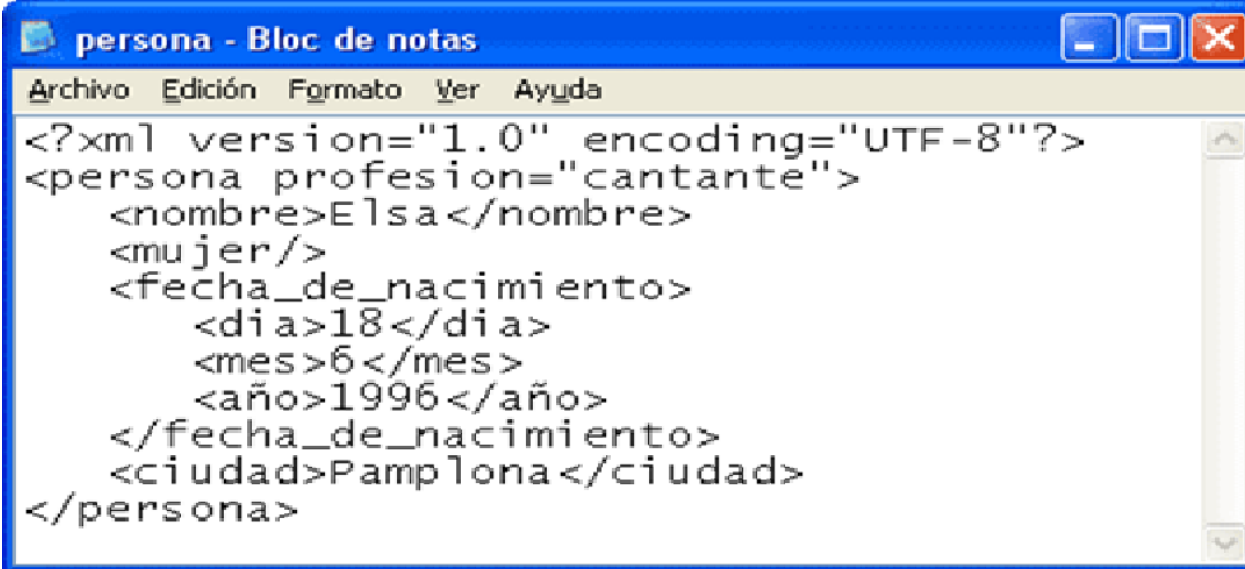
- La declaración XML no es una instrucción de procesamiento (o proceso).

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

- En un documento XML no es obligatorio que aparezca la declaración XML.
- Si se incluye, tiene que aparecer en la primera línea del documento, y el carácter “<” debe ser el primero de dicha línea.

# CÓMO CREAR UN DOCUMENTO XML

- **EJEMPLO** en el *Bloc de notas* de *Microsoft Windows* (codificado en UTF-8).

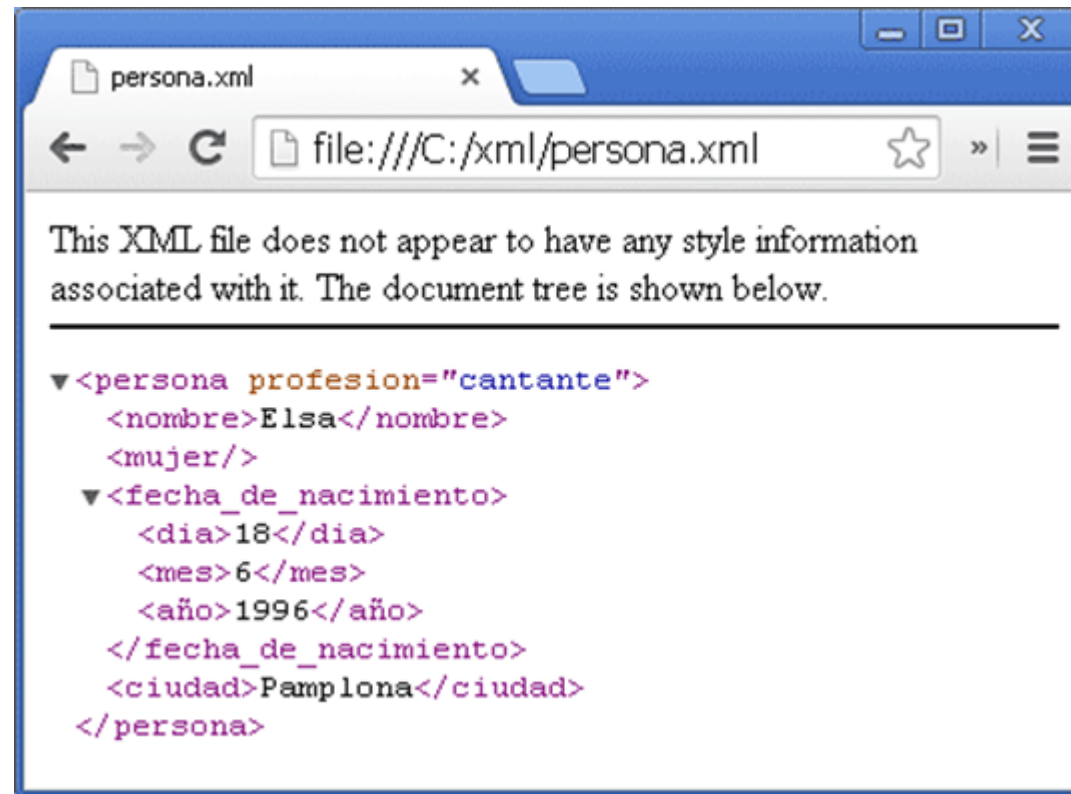


```
<?xml version="1.0" encoding="UTF-8"?>
<persona profesion="cantante">
  <nombre>Elsa</nombre>
  <mujer/>
  <fecha_de_nacimiento>
    <dia>18</dia>
    <mes>6</mes>
    <año>1996</año>
  </fecha_de_nacimiento>
  <ciudad>Pamplona</ciudad>
</persona>
```



# VISUALIZAR UN DOCUMENTO XML

- EJEMPLO en *Google Chrome*.



# DECLARACIÓN XML

- El atributo **standalone** puede tomar dos valores ("**yes**" o "**no**").

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

- "**yes**" indica que el documento es independiente de otros.
- Si se escribe la declaración XML, el atributo **version** es obligatorio. Sin embargo, los atributos **encoding** y **standalone** son opcionales y, por defecto, sus valores son "**UTF-8**" y "**no**", respectivamente.

# INSTRUCCIONES D E PROCESAMIENTO

- Una instrucción de procesamiento sirve para indicar cierta información al programa que procese dicho documento.
- **EJEMPLO** Asociar un archivo CSS a un documento XML:

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

- **EJEMPLO** Contenido del archivo "*estilo-animales.css*":

```
nombre{color:blue;font-size:40px}  
patas{color:red;font-size:22px}
```

# EJEMPLO "ANIMALES.XML"

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/css" href="estilo_animales.css"?>
<animales>
  <animal>
    <nombre>perro</nombre>
    <patas>4</patas>
  </animal>
  <animal>
    <nombre>pato</nombre>
    <patas>2</patas>
  </animal>
  <animal>
    <nombre>ballena</nombre>
    <patas>0</patas>
  </animal>
</animales>
```

# "ANIMALES.XML"



# REFERENCIAS A ENTIDADES

Referencias a entidades en XML		
<i>Carácter</i>	<i>Entidad</i>	<i>Referencia a entidad</i>
< ( <i>menor que</i> )	<b>lt</b> ( <i>less than</i> )	<b>&amp;lt;</b>
> ( <i>mayor que</i> )	<b>gt</b> ( <i>greater than</i> )	<b>&amp;gt;</b>
" ( <i>comilla doble</i> )	<b>quot</b> ( <i>quotation mark</i> )	<b>&amp;quot;</b>
' ( <i>comilla simple</i> )	<b>apos</b> ( <i>apostrophe</i> )	<b>&amp;apos;</b>
& ( <i>ampersand</i> )	<b>amp</b> ( <i>ampersand</i> )	<b>&amp;amp;</b>

# REFERENCIAS A ENTIDADES

- EJEMPLO “*entidades.xml*”

```
<?xml version="1.0" encoding="UTF-8"?>
<entidades>
  <menor_que>&lt;</menor_que>
  <mayor_que>&gt;</mayor_que>
  <comilla_doble>&quot;</comilla_doble>
  <comilla_simple>&apos;</comilla_simple>
  <ampersand>&amp;</ampersand>
</entidades>
```

# "ENTIDADES.XML"





# CARACTERES PROBLEMÁTICOS EN XML: MENOR QUE ( < ) Y AMPERSAND (&)

- No es correcto:

```
<condicion>a<b</condicion>
```

```
<condicion>a=1 && b=2</condicion>
```

- Sí es correcto:

```
<condicion>a< b</condicion>
```

```
<condicion>a=1 & & b=2</condicion>
```

```
<condicion>a>b</condicion>
```

# USO DE LA COMILLA DOBLE (") Y DE LA COMILLA SIMPLE (') EN ATRIBUTOS

- No es correcto:

```
<dato character="comilla doble(")"/>  
<dato character='comilla simple(')'/>
```

- Sí es correcto:

```
<dato character="comilla doble(&quot;)/>  
<dato character='comilla simple(&apos;)/>
```

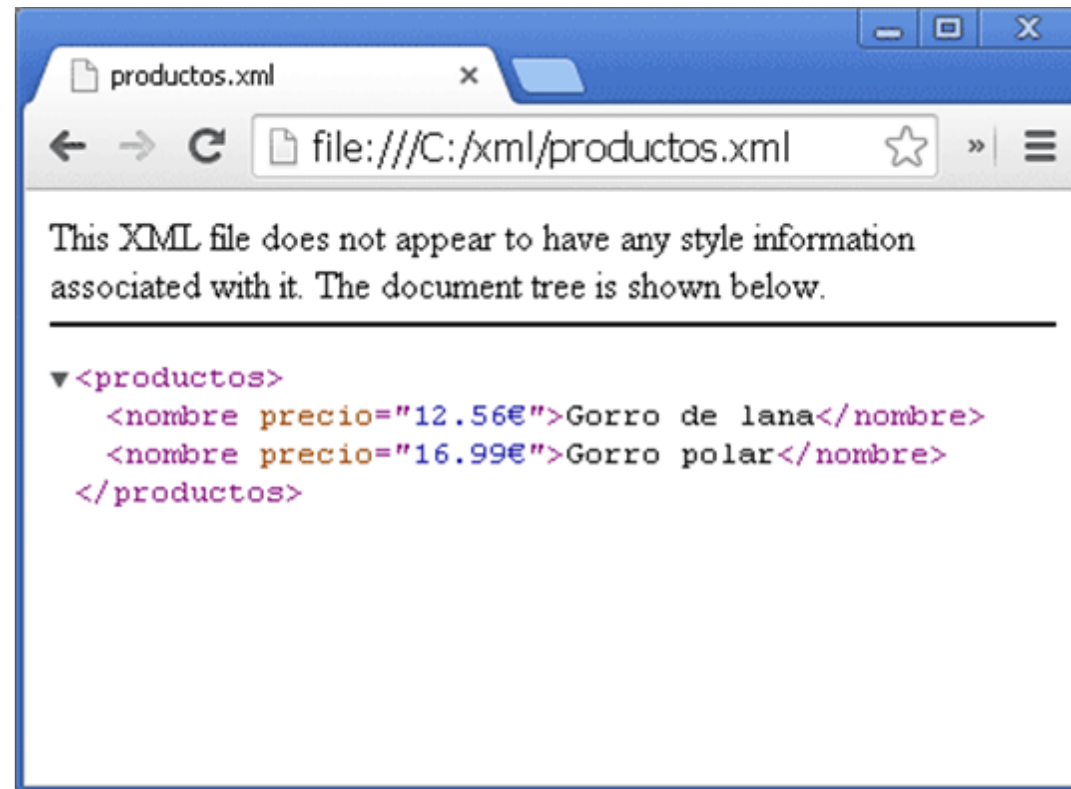
```
<dato character="comilla simple(')"/>  
<dato character='comilla doble(")'/>
```

# REFERENCIAS D E CARACTERES

- EJEMPLO “*productos.xml*”

```
<?xml version="1.0" encoding="UTF-8"?>
<productos>
<nombre precio="12.56&#8364;">Gorro de lana</nombre>
<nombre precio="16.99&#x20AC;">Gorro polar</nombre>
</productos>
```

# "PRODUCTOS.XML"



# COMENTARIOS. EJEMPLO "LETRAS.XML"

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

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

```
<a>
```

```
  <b>
```

```
    <c cantidad="4">cccc</c>
```

```
    <d cantidad="2">dd</d>
```

```
  </b>
```

```
  <e>
```

```
    <f cantidad="8">fffffffff</f>
```

```
    <!--g puede aparecer varias veces.-->
```

```
    <g cantidad="5">ggggg</g>
```

```
    <g cantidad="2">gg</g>
```

```
  </e>
```

```
</a>
```

# "LETRAS.XML"



# COMENTARIOS

- No se pueden escribir comentarios dentro de las etiquetas.

```
<mujer <!-- elemento vacío --> />
```

- En los comentarios no está permitido usar dos guiones seguidos:

```
<!-- Dos guiones seguidos -- en un comentario da error -->
```

- No es posible anidar comentarios en un documento XML.

# SECCIONES CDATA

- Dentro de una sección CDATA no se puede escribir la cadena “]]>”. En consecuencia, no se pueden anidar secciones CDATA.
- No está permitido escribir espacios en blanco o saltos de línea en las cadenas de inicio “<![CDATA[” o fin “]]>” de una sección CDATA.



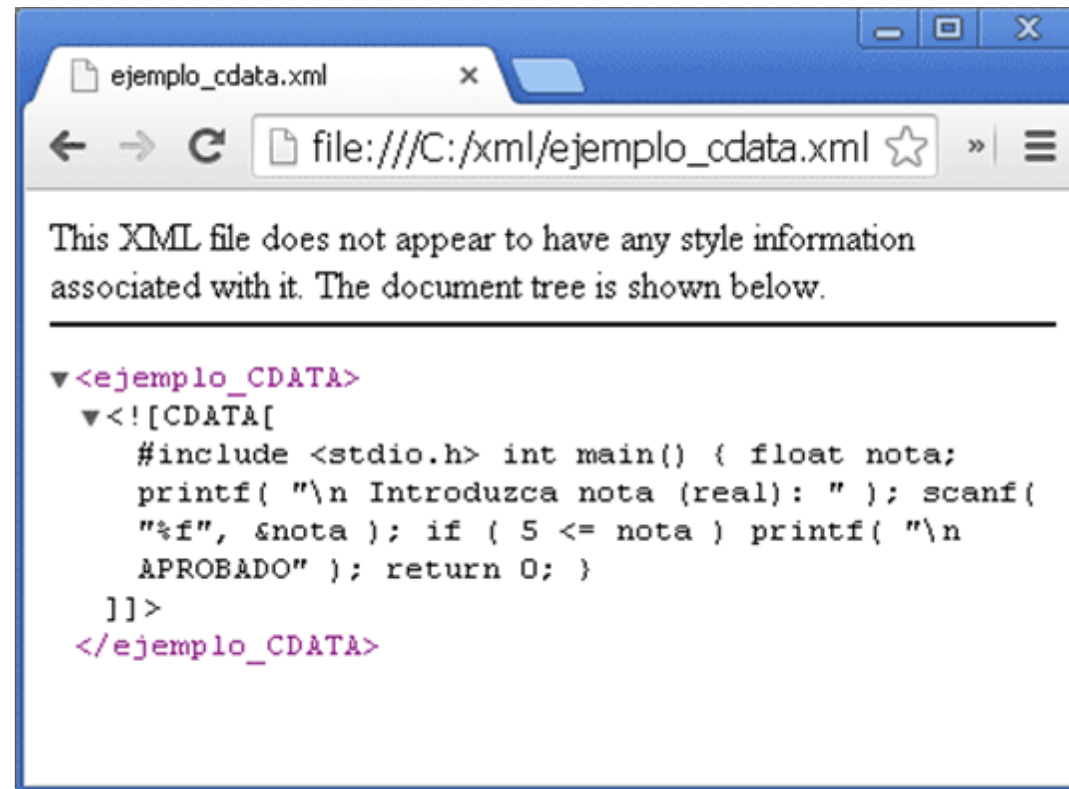
# SECCIONES CDATA. "EJEMPLO\_CDATA"

```
<?xml version="1.0" encoding="UTF-8"?>  
<ejemplo_CDATA>  
  <![CDATA[
```

En un lugar de la Mancha<sup>2</sup>, de cuyo nombre no quiero acordarme<sup>3</sup>, no ha mucho tiempo que vivía un hidalgo de los de lanza en astillero, adarga antigua, rocín flaco y galgo corredor<sup>4</sup>.

```
  ]]>  
</ejemplo_CDATA>
```

# "EJEMPLO\_CDATA"



# ESPACIOS DE NOMBRES

- **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_tenera 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>
```

# USO DE ESPACIOS DE NOMBRES

```
<?xml version="1.0" encoding="UTF-8"?>
<e1:ejemplo xmlns:e1="http://www.arangoya.com/ejemplo1"
  xmlns:e2="http://www.arangoya.com/ejemplo2">
  <e1:carta>
    <e1:palo>Corazones</e1:palo>
    <e1:numero>7</e1:numero>
  </e1:carta>
  <e2:carta>
    <e2:carnes>
      <e2:filete_de_tenera 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>
</e2:ejemplo>
```

# SINTAXIS PARA DEFINIR UN ESPACIO DE NOMBRES

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

```
xmlns:e1="http://www.arangoya.com/ejemplo1"
```

```
xmlns:e2="http://www.arangoya.com/ejemplo2"
```

- Los URI no tienen porqué contener nada, su función es ser únicos. No obstante, en un URI se puede mostrar información si se considera oportuno:
  - <http://www.w3.org/1999/xhtml/>
  - <http://www.w3.org/1999/XSL/Transform>
  - <http://www.w3.org/2000/svg>

# DEFINICIÓN DE ESPACIOS DE NOMBRES EN ELEMENTOS DISTINTOS A LA RAÍZ

```
<?xml version="1.0" encoding="UTF-8"?>
<e1:ejemplo xmlns:e1="http://www.arangoya.com/ejemplo1">
  <e1:carta>
    <e1:palo>Corazones</e1:palo>
    <e1:numero>7</e1:numero>
  </e1:carta>
  <e2:carta xmlns:e2="http://www.arangoya.com/ejemplo2">
    <e2:carnes>
      <e2:filete_de_tenera 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 UN ESPACIO DE NOMBRES POR DEFECTO

- Sintaxis:

```
xmlns="URI"
```

- **EJEMPLO**

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

- **EJEMPLO**

```
<?xml version="1.0" encoding="UTF-8"?>
<ejemplo xmlns="http://www.arangoya.com/ejemplo1">
  <carta>
    <palo>Corazones</pallo>
    <numero>7</numero>
  </carta>
  <carta xmlns="http://www.arangoya.com/ejemplo2">
    <carnes>
      <filete_de_tenera 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>
</ejemplo>
```



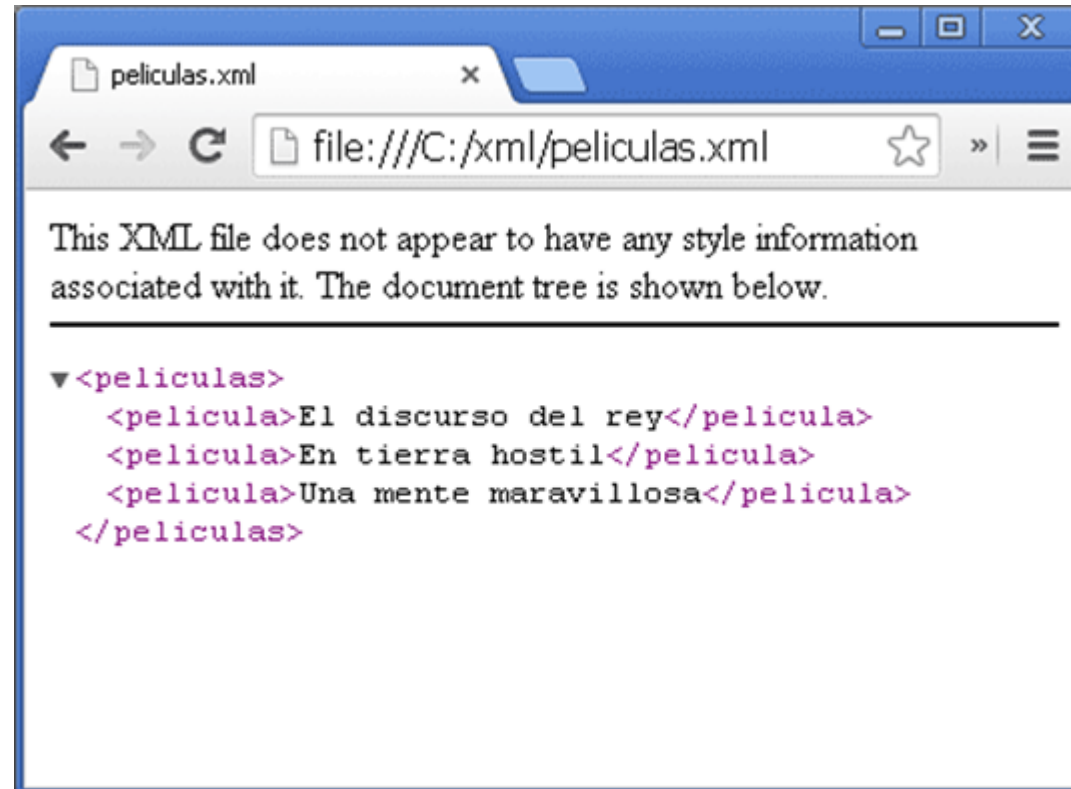
# CÓMO INDICAR QUE UN ELEMENTO NO PERTENECE A NINGÚN ESPACIO DE NOMBRES

```
<?xml version="1.0" encoding="UTF-8"?>
<ejemplo xmlns="http://www.arangoya.com/ejemplo1">
  <carta>
    <palo>Corazones</palo>
    <numero>7</numero>
  </carta>
  <carta xmlns="http://www.arangoya.com/ejemplo2">
    <carnes>
      <filete_de_tenera precio="12.95"/>
      <solomillo_a_la_pimienta precio="13.60"/>
    </carnes>
    <pescados xmlns="">
      <lenguado_al_horno precio="16.20"/>
      <merluza_en_salsa_verde precio="15.85"/>
    </pescados>
  </carta>
</ejemplo>
```

# ESPACIOS EN BLANCO EN EL CONTENIDO (TEXTO) DE UN ELEMENTO. EJEMPLO "PELICULAS.XML"

```
<?xml version="1.0" encoding="UTF-8"?>
<peliculas>
  <pelicula>El discurso del rey</pelicula>
  <pelicula>En      tierra      hostil</pelicula>
  <pelicula>Una
    mente
maravillosa</pelicula>
</peliculas>
```

# "PELICULAS.XML"



# ESPACIOS EN BLANCO EN ATRIBUTOS.

## EJEMPLO "SERIES.XML"

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

```
<series>
```

```
  <serie numeros="2 4 6 8"/>
```

```
  <serie numeros="3
```

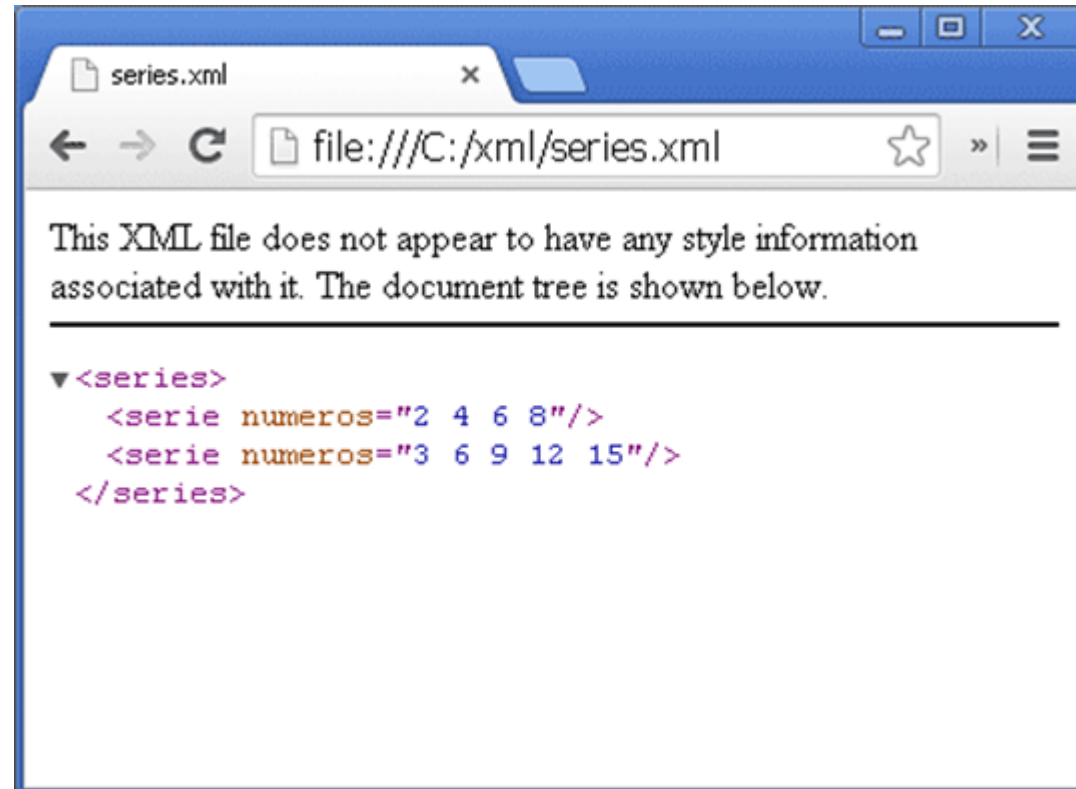
```
6
```

```
9
```

```
12 15"/>
```

```
</series>
```

# "SERIES.XML"



# ESPACIOS E N BLANCO ENTRE ELEMENTOS.

## EJEMPLO "DATOS.XML"

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

```
<datos>
```

```
  <dato>1</dato>
```

```
  <dato>2</dato>
```

```
  <dato>3</dato>
```

```
</datos>
```

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

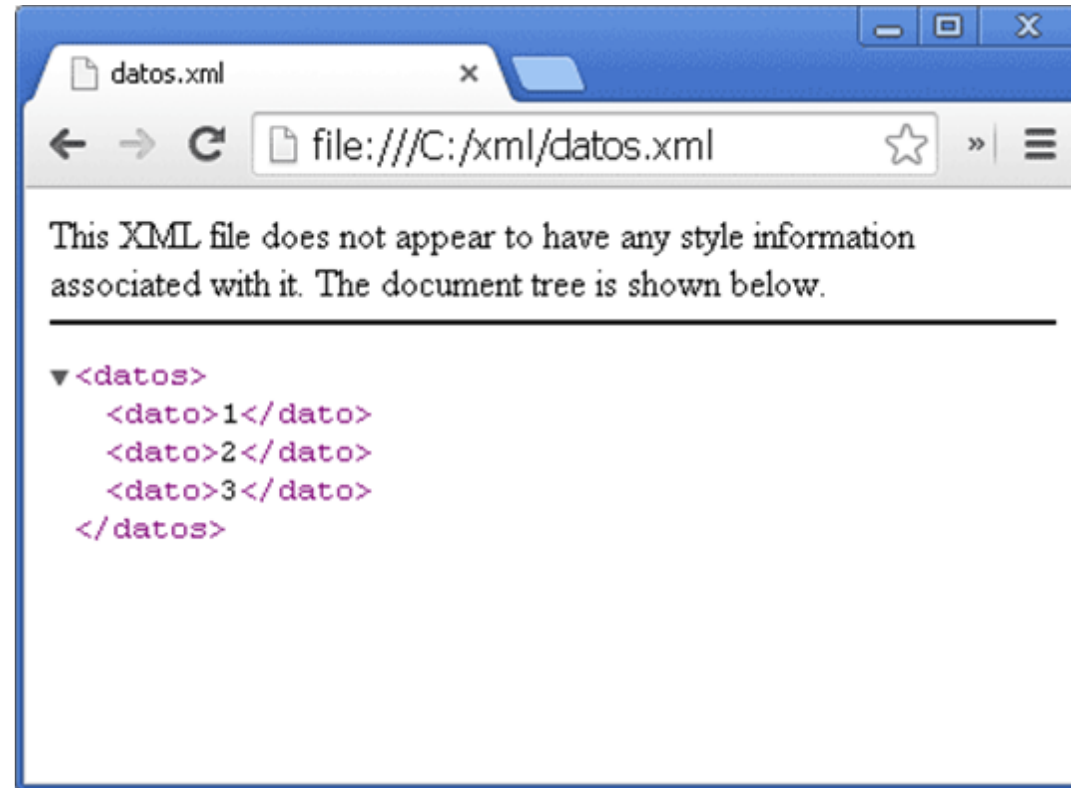
```
<datos><dato>1</dato><dato>2</dato><dato>3</dato></datos>
```

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

```
<datos><dato>1</dato>  <dato>2</dato>
```

```
<dato>3</dato></datos>
```

# "DATOS.XML"



# USO DEL ATRIBUTO XML:SPACE

```
<clasificacion xml:space="preserve">
1          Fernando Alonso          1:55.341
2          Lewis Hamilton            1:55.729
3          Sebastian Vettel          1:56.122
</clasificacion>
```

- Los únicos valores que admite el atributo **xml:space** son "**preserve**" y "**default**", siendo este último su valor por defecto cuando no se escribe dicho atributo.
- El valor "**default**" indica que la aplicación que haga uso del documento XML es la encargada de decidir cómo tratar los espacios en blanco.
- No todos los programas reconocen este atributo.



# DOCUMENTOS XML BIEN FORMADOS (SIN 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.
- Las secciones CDATA y los comentarios deben estar correctamente escritos.

# DOCUMENTOS XML VÁLIDOS

- Un documento XML es válido 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:
  - **DTD** (*Document Type Definition*).
  - **XML Schema**.
  - **RELAX NG** (*REgular LAnguage for XML Next Generation*).