

XML: eXtensible
Markup Language
(Parte I)
DTDs

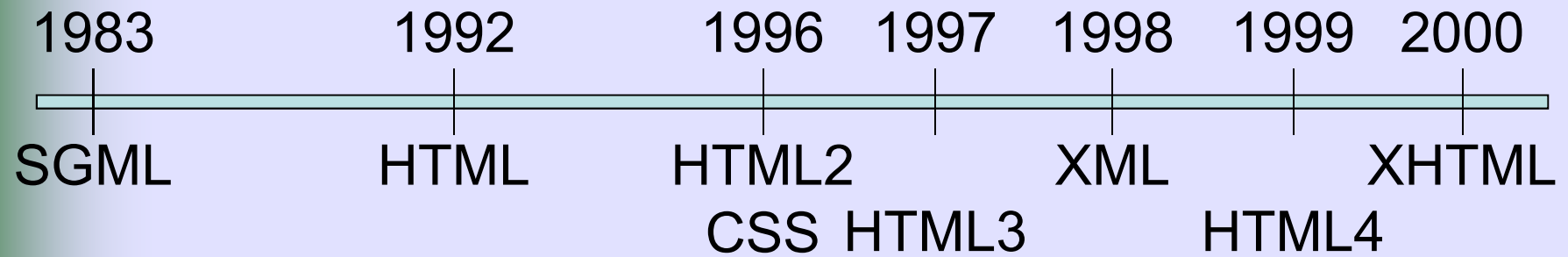
Tabla de Contenidos

- Introducción histórica
- XML vs. HTML
- Características de XML
- Mi primer documento XML
- El lenguaje XML
- Entidades
- Espacios de nombres: *Namespaces*
- La familia de XML
- Dominios de utilización de XML

Introducción histórica (i)

- ✎ XML se constituyó como estándar de la W3C en el año 1998. En 2000 se aprueba su versión 1.0.
- ✎ Se trata de un lenguaje de marcas, igual que HTML o su precursor SGML.
- ✎ Se diferencia de SGML por su sencillez.
- ✎ Se diferencia de HTML por su flexibilidad: el número de etiquetas que puede incluir un documento XML es ilimitado.
- ✎ Al igual que HTML, es portable a cualquier plataforma.

Introducción histórica (ii)



Introducción histórica (iii)

Objetivos principales:

- Directamente utilizable en Internet.
- Soporte para una amplia variedad de aplicaciones.
- Compatible con SGML.
- Posibilidad de crear sencillos procesadores de XML.
- Número de opciones mínimo (lo óptimo son cero).
- Documentos XML legibles y medianamente claros.
- Diseño rápido del lenguaje.
- Simple, pero perfectamente formalizado.
- Documentos XML fáciles de crear.

XML vs. HTML

- HTML carece de un chequeo sintáctico. Páginas con errores son mostradas en los navegadores.
- HTML carece de estructura.
- HTML mezcla contenido y representación.
- Por todo esto:
 - HTML no puede ser fácilmente leído por una máquina.
 - HTML nunca será un estándar de intercambio de datos.
- XML cubre todo esto con un lenguaje de sencillez extrema.

Características de XML (i)

- Es un subconjunto del lenguaje SGML.
- Al igual que él, se utiliza para representar datos de forma estructurada.
- Se basa en una gramática de obligado cumplimiento. Esto facilita el desarrollo de ***parsers*** y, por lo tanto, su utilización masiva.
- La estructura interna de un documento XML puede reflejarse en:
 - DTD (*Document Type Definition*)
 - XML Schema
- A diferencia de HTML, separa radicalmente la semántica del documento, de su representación gráfica.

Características de XML (ii)

- ❏ XML, *eXtensible Markup Language*, se ha convertido en el estándar para intercambio de datos no sólo en el WWW.
- ❏ Fácil de usar por estar basado en un conjunto extensible de etiquetas semánticas entendibles por humanos y máquinas.
- ❏ Ya se encuentra en una fase de madurez y expansión absoluta.
- ❏ Gracias al soporte de *Unicode* se soportan los alfabetos de todo el mundo.

Mi primer documento XML. HTML (i)

<HTML>

<HEAD><TITLE>Libros de mi infancia</TITLE></HEAD>

<BODY>

<P><I>Don Quijote de la Mancha</I>

<P><I>Miguel de Cervantes</I>

<HR>

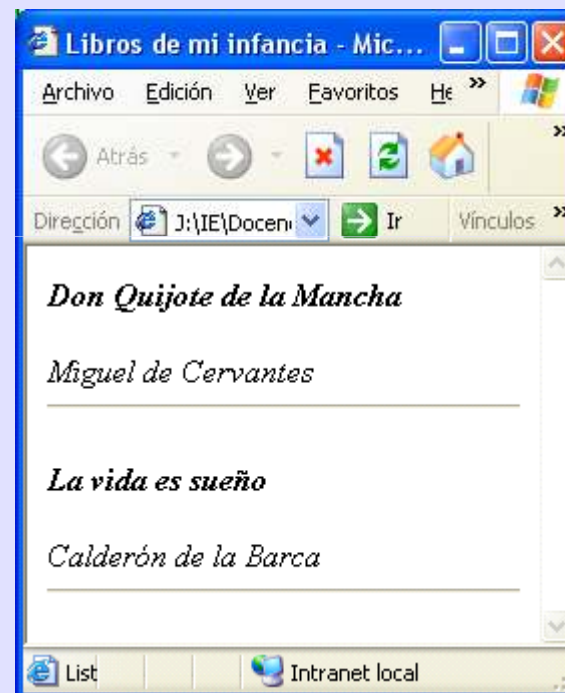
<P>La vida es sueño

<P><I>Calderón de la Barca</I>

<HR>

</BODY>

</HTML>



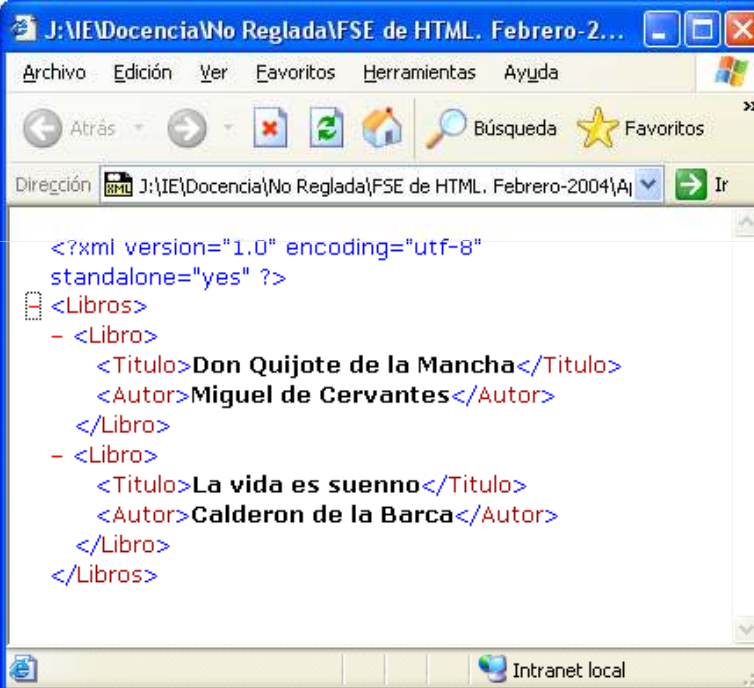
Mi primer documento XML. HTML (ii)

- ❏ En apariencia, el documento HTML anterior es correcto, sin embargo:
 - Existen etiquetas que nunca se cierran: <P>
 - Algunas etiquetas no están bien anidadas: el primer <I> nunca se cierra.
 - Para un lector no humano, no se sabe qué es un libro y qué es un autor.

iiXML erradica todos estos problemas!!

Mi primer documento XML (iii)

```
<?xml version="1.0" encoding="utf-8"
  standalone="yes" ?>
<!-- Inventario.xml -->
<Libros>
  <Libro>
    <Titulo>Don Quijote de la
    Mancha</Titulo>
    <Autor>Miguel de
    Cervantes</Autor>
  </Libro>
  <Libro>
    <Titulo>La vida es
    sueño</Titulo>
    <Autor>Calderon de la
    Barca</Autor>
  </Libro>
</Libros>
```



```
<?xml version="1.0" encoding="utf-8"
standalone="yes" ?>
<Libros>
- <Libro>
  <Titulo>Don Quijote de la Mancha</Titulo>
  <Autor>Miguel de Cervantes</Autor>
</Libro>
- <Libro>
  <Titulo>La vida es sueño</Titulo>
  <Autor>Calderon de la Barca</Autor>
</Libro>
</Libros>
```

El lenguaje XML. Reglas generales

- ❏ Un único elemento raíz (elemento documento).
- ❏ Todo elemento debe tener etiquetas de apertura y cierre.
- ❏ Distinción entre mayúsculas/minúsculas.
- ❏ Anidamiento perfecto entre elementos.
- ❏ Los valores de atributos van siempre entre comillas.
- ❏ Los espacios en blanco se conservan.
- ❏ Los caracteres CR/LF se transforman en LF.

El lenguaje XML.

Documentos válidos

- Se dice que un documento es **bien formado** cuando:
 - Cumple con todas las reglas anteriormente expuestas.
 - Contiene uno o más elementos.
 - Hay un único elemento documento.
 - Si el documento consta de más de una parte, todas están bien formadas.
 - No se encuentran caracteres prohibidos en el texto.
- Un documento es **válido** cuando, además de ser 'bien formado', cumple con las especificaciones semánticas expuestas en su plantilla (DTD o XML Schema).

El lenguaje XML. Elementos (i)

Comentarios:

- `<! -- Esto es un comentario, y no puedo incluir un doble guión-->`

Instrucciones de procesamiento:

- `<? Instrucción ?>`
- La instrucción no puede incluir los caracteres `?>`

Secciones CDATA:

- `<![CDATA[Este texto no será tratado, puede incluir "cualquier" &carácter < >]]>`
- No son tratadas por el *parser*.
- Pueden incluir cualquier carácter prohibido (`"`, `'`, `&`, `>`, `<`), pero no puede incluir el carácter `'ñ'`, acentos, ni la cadena `]]>`

El lenguaje XML. Elementos (ii)

Prólogo:

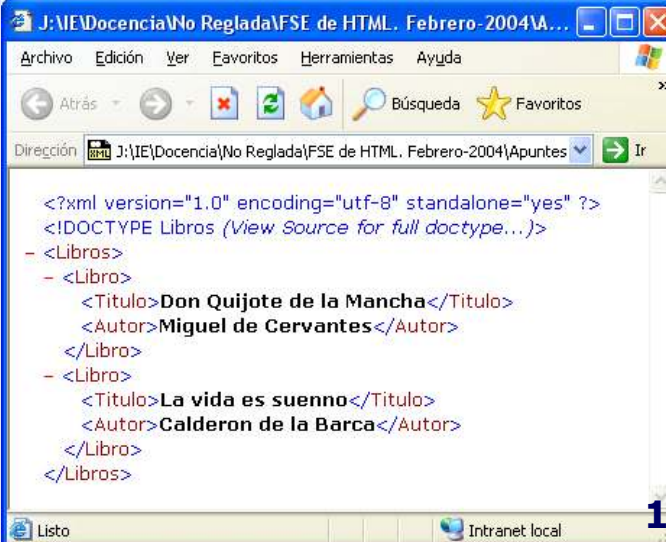
- `<?xml version="1.0" encoding="utf-8" standalone="yes" ?>`
- Es una instrucción de procesamiento obligatoria.
- Version: indica la versión de XML que se está utilizando (1.0 en la actualidad). Es obligatoria.
- Encoding: indica cómo se codificó el documento, y no es obligatoria (por defecto UTF-8). Válido para otros juegos de caracteres.
- Standalone: "yes" indica que el documento no va acompañado de DTD ni de XML Schema; "no" indica que requiere una DTD o XML Schema. No es un atributo obligatorio.

El lenguaje XML. Elementos (iii)

DOCTYPE:

- `<!DOCTYPE MiDTD SYSTEM "C:\MiDTD.dtd">`
- Indica la referencia (URI) a la DTD, así como el nombre (MiDTD) del elemento raíz de la misma.
- La DTD podría ir incorporada en el propio documento XML, sin requerir otro fichero aparte.
- El documento XML deberá cumplir con el contenido de la DTD.
- También se puede utilizar para definir entidades.

```
<?xml version="1.0" encoding="utf-8" ?>  
<!DOCTYPE Libros SYSTEM "Libros1.dtd">  
<Libros>  
  <Libro>  
    <Titulo>Don Quijote de la Mancha</Titulo>  
    <Autor>Miguel de Cervantes</Autor>  
  </Libro>  
  <Libro>  
    <Titulo>La vida es sueno</Titulo>  
    <Autor>Calderon de la Barca</Autor>  
  </Libro>  
</Libros>
```



```
<?xml version="1.0" encoding="utf-8" standalone="yes" ?>  
<!DOCTYPE Libros (View Source for full doctype...)>  
- <Libros>  
  - <Libro>  
    <Titulo>Don Quijote de la Mancha</Titulo>  
    <Autor>Miguel de Cervantes</Autor>  
  </Libro>  
  - <Libro>  
    <Titulo>La vida es sueno</Titulo>  
    <Autor>Calderon de la Barca</Autor>  
  </Libro>  
</Libros>
```


El lenguaje XML. Elementos (iv)

🔗 Para referenciar un XML Schema:

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

```
<note xmlns="http://www.w3schools.com"  
xmlns:xsi="http://www.w3.org/2001/XMLSchema-  
instance"  
xsi:schemaLocation="http://www.w3schools.com  
note.xsd">
```

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

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

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

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

```
</note>
```

El lenguaje XML. Elementos (v)

Etiquetas:

- Deben ir correctamente anidadas: apertura y cierre.
- Etiqueta de apertura: comienza por <, más el nombre de la etiqueta y terminan por >. Ejemplo <Libro>.
- Etiqueta de cierre: </Libro>
- Etiqueta vacía: <Libro />
- Debe comenzar por una letra o un “_”.
- No puede comenzar por “xml”.

El lenguaje XML. Elementos (vi)

Elemento:

- Es el conjunto de la etiqueta (marcador) de apertura, su contenido y la de cierre.
- Por ejemplo: `<Libro>Don Quijote de la Mancha</Libro>`
- Hay algunos caracteres reservados (prohibidos):
 - Signo de mayor: `>`
 - Signo de menor: `<`
 - Ampersand: `&`
 - Apóstrofe: `'`
 - Comilla: `"`
- Estos caracteres prohibidos se reemplazan por entidades o se incluyen en secciones **CDATA**.

El lenguaje XML. Elementos (vii)

Atributos:

- Especificación **nombre-valor** asociada con el elemento.
- Cada elemento puede contener 0 ó más atributos
- Su valor debe ir siempre entrecomillado.
- Sólo pueden aparecer en etiquetas de apertura o vacías.
- El mismo atributo no puede aparecer repetido en la misma etiqueta.
- No puede contener ninguna referencia a entidad externa.
- Son siempre tratados como cadenas de texto.

El lenguaje XML. Elementos (viii)

Atributos:

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

```
<Libros>
```

```
<Libro Precio = "1.123 euros" Editorial = "Santillana">
```

```
<Titulo>Don Quijote de la Mancha</Titulo>
```

```
<Autor>Miguel de Cervantes</Autor>
```

```
</Libro>
```

```
<Libro Precio = "658 euros" Editorial = "Anaya">
```

```
<Titulo>La vida es sueño</Titulo>
```

```
<Autor>Calderón de la Barca</Autor>
```

```
</Libro>
```

```
</Libros>
```

El lenguaje XML.

Referencias a caracteres

- ▶ Permiten incluir cualquier carácter dentro de un documento XML.
- ▶ Basado en el conjunto de caracteres ISO/IEC 10646 (<http://xml.coverpages.org/xml-ISOents.txt>).
- ▶ Dos formatos:
 - `&#valor;` Valor representado en decimal
 - `&#xvalor;` Valor representado en hexadecimal

```
<?xml version="1.0" encoding="utf-8" standalone="yes" ?>
<Libros>
  <Libro Precio = "658 euros" Editorial = "Anaya">
    <Titulo>La vida es sue&#241;o </Titulo>
    <Autor>Calder&#243;n de la Barca </Autor>
  </Libro>
</Libros>
```

El lenguaje XML. Entidades (i)

- Las entidades permiten:
 - Dar modularidad al texto evitando tener que escribir algo de forma repetitiva.
 - Incluir caracteres prohibidos &, >, <, ", `
 - Incluir caracteres de otros idiomas: ñ, acentos...
- Comienzan por & y terminan en ;
 - Por ejemplo ***&***; para representar ***&***

El lenguaje XML. Entidades (ii)

Entidades predefinidas:

- | | | | |
|-----------------|------|---|--------|
| • Signo menor | lt | < | < |
| • Signo mayor | gt | > | > |
| • Ampersand | amp | & | & |
| • Apóstrofe | apos | ' | ' |
| • Comilla doble | quot | " | " |

El lenguaje XML. Entidades (iii)

Tipos de entidades:

- **General y de Parámetro:**

- General: contiene texto XML u otros caracteres.
- De Parámetro: contiene texto XML que puede insertarse dentro de una DTD.

- **Interna y Externa:**

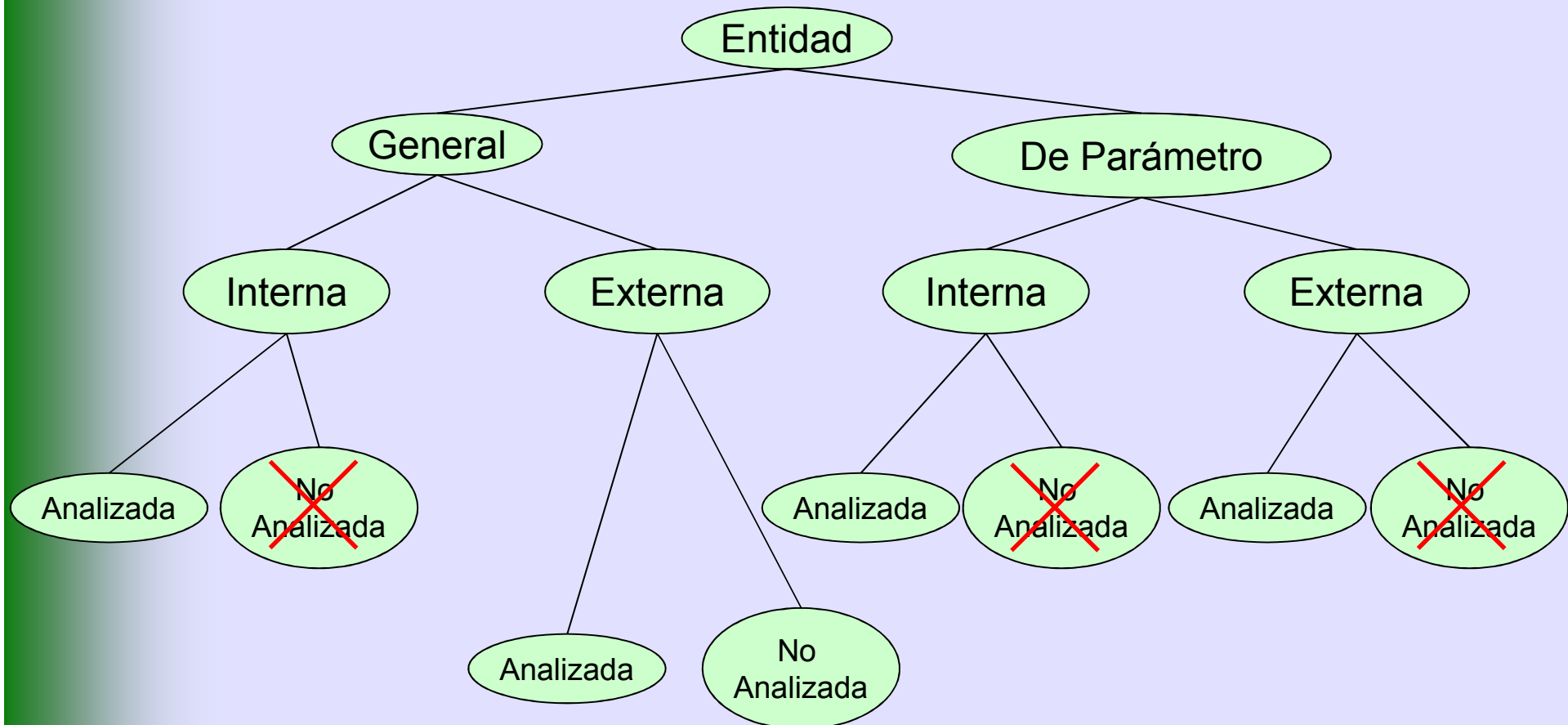
- Interna: contiene el texto dentro de una cadena entrecomillada.
- Externa: hace referencia a un archivo externo.

- **Analizada y no Analizada:**

- Analizada: texto XML que será *procesado* en su punto de inserción.
- No Analizada: no será *procesada*.

El lenguaje XML. Entidades (iv)

Tipos de entidades:



Entidades generales internas analizadas (i)

- ▶ Permiten que una determinada cadena se repita fácilmente a lo largo de un documento XML
- ▶ Se definen dentro del fichero XML o de la DTD .

```
<?xml version="1.0" encoding="utf-8" ?>
<!DOCTYPE Libros [
  <!ENTITY Es "Espa&#241;ol">
  <!ENTITY It "Italiano">
]>
<Libros>
...
</Libros>
```

```
<!ELEMENT Libros (Libro)+>
<!ELEMENT Libro (Titulo, Autor)>
<!ELEMENT Titulo (#PCDATA)>
<!ELEMENT Autor (#PCDATA)>
<!ATTLIST Autor Nacionalidad CDATA #IMPLIED>
<!ENTITY Es "Espa&#241;ol">
<!ENTITY It "Italiano">
```

Entidades generales internas analizadas (ii)

```
<?xml version="1.0" encoding="utf-8" ?>
<!DOCTYPE Libros SYSTEM "Libros5.dtd">
<Libros>
  <Libro Precio = "1.123 euros" Editorial = "Santillana">
    <Titulo>Don Quijote de la Mancha</Titulo>
    <Autor Nacionalidad = "&Es;">Miguel de Cervantes</Autor>
  </Libro>
  <Libro Precio = "658 euros" Editorial = "Anaya">
    <Titulo>La vida es sue&#241;o</Titulo>
    <Autor Nacionalidad = "&Es;">Calder&#243;n de la Barca</Autor>
  </Libro>
  <Libro Precio = "88 euros" Editorial = "Anaya">
    <Titulo>El nombre de la rosa</Titulo>
    <Autor Nacionalidad = "&It;">Umberto Eco</Autor>
  </Libro>
  <Libro Precio = "83 euros" Editorial = "Santillana">
    <Titulo>El alquimista</Titulo>
    <Autor Nacionalidad = "Brasile&#241;o">Paulo Coelho</Autor>
  </Libro>
</Libros>
```

Entidades generales internas analizadas (iii)

```
<!ELEMENT Libros (Libro)+>
<!ELEMENT Libro (Titulo, Autor)>
<!ELEMENT Titulo (#PCDATA)>
<!ELEMENT Autor (#PCDATA)>
<!ATTLIST Autor Nacionalidad CDATA #IMPLIED>
<!ENTITY Es "Espa&#241;o">
<!ENTITY It "Italiano">
<!ENTITY Coelho '<Autor Nacionalidad = "Brasile&#241;o">Paulo Coelho</Autor>''>
<?xml version="1.0" encoding="utf-8" ?>
<!DOCTYPE Libros SYSTEM "Libros6.dtd">
<Libros>
  <Libro Precio = "83 euros" Editorial = "Santillana">
    <Titulo>El alquimista</Titulo>
    <Autor Nacionalidad = "Brasile&#241;o">Paulo Coelho</Autor>
  </Libro>
  <Libro Precio = "85 euros" Editorial = "Santillana">
    <Titulo>A orillas del r&#237;o Piedra</Titulo>
    &Coelho;
  </Libro>
</Libros>
```

Entidades generales externas analizadas

- Su misión es idéntica a las *Entidades Internas Analizadas*.
- No definen su contenido entre comillas, sino en un fichero externo.
- El fichero externo puede incluir entidades. Si éstas son externas ojo a las referencias circulares.
- Sólo pueden ser incluidas dentro de un elemento.
- Declaración:
 - `<!ENTITY NombreEntidad SYSTEM "URI">`

Entidades generales externas no analizadas

- ▶ Permiten incluir dentro de un fichero XML información, procedente de otro fichero externo, que no tiene porqué ser XML.
 - Por ejemplo: imágenes, documentos Office...
- ▶ Se basa en la definición de notaciones: Ubicación del programa que reconoce al fichero externo, y de atributos de tipo ENTITY.
 - `<!NOTATION NombreNotacion SYSTEM "URI">`

Entidades generales externas no analizadas (ii)

```
<!ELEMENT Libros (Libro)+>  
<!NOTATION GIF SYSTEM "C:\Archivos de programa\IrfanView\i_view32.exe">  
<!ELEMENT Libro (Titulo, Autor, Portada)>  
<!ELEMENT Titulo (#PCDATA)>  
<!ELEMENT Autor (#PCDATA)>  
<!ELEMENT Portada EMPTY>  
<!ATTLIST Libro Precio CDATA #REQUIRED Editorial CDATA #REQUIRED>  
<!ATTLIST Portada Imagen ENTITY #IMPLIED>  
<!ATTLIST Autor Nacionalidad CDATA #IMPLIED>  
<!ENTITY Coelho '<Autor Nacionalidad = "Brasile&#241;o">Paolo Coelho</Autor>'  
<!ENTITY Alquimista SYSTEM "Alquimista.gif" NDATA GIF>
```

```
<?xml version="1.0" encoding="utf-8" ?>  
<!DOCTYPE Libros SYSTEM "Libros(Notaciones).dtd">  
<Libros>  
  <Libro Precio = "83 euros" Editorial = "Santillana">  
    <Titulo>El alquimista</Titulo>  
    &Coelho;  
    <Portada Imagen = "Alquimista" />  
  </Libro>  
</Libros>
```


Entidades de parámetro internas analizadas

- Al igual que las generales, permiten la sustitución de cadenas de caracteres, pero esta vez dentro de documentos DTD.
- No pueden usarse dentro de una etiqueta, sino para sustituir una etiqueta completa.
- Declaración:

- `<!ENTITY % NombreEntidad "Valor">`

```
<!ELEMENT Biblioteca (Libro)+>  
<!ELEMENT Libro (Titulo, Autor)>  
<!ELEMENT Titulo (#PCDATA)>  
<!ELEMENT Autor (#PCDATA)>  
<!ATTLIST Libro  
    Precio CDATA #IMPLIED  
    Editorial CDATA #IMPLIED  
>
```

```
<!ENTITY % Nacion '  
<!ATTLIST Autor Nacionalidad CDATA #IMPLIED>'  
%Nacion;
```

Entidades de parámetro externas analizadas

- Su uso es idéntico a las *Entidades de Parámetros Internas Analizadas*.
- Hacen referencia a un fichero externo que contiene un trozo de DTD.

```
<!ELEMENT Biblioteca (Libro)+>
<!ELEMENT Libro (Titulo, Autor)>
<!ELEMENT Titulo (#PCDATA)>
<!ELEMENT Autor (#PCDATA)>
<!ATTLIST Libro
    Precio CDATA #IMPLIED
    Editorial CDATA #IMPLIED
>
<!ENTITY % Nacion SYSTEM "Libros7.dtd">
%Nacion;
```

Entidades con referencias a caracteres

ISO-10646.dtd {
<!ENTITY Agrave "À"><!-- capital A, grave accent -->
<!ENTITY Aacute "Á"><!-- capital A, acute accent -->
<!ENTITY Acirc "Â"><!-- capital A, circumflex accent -->
<!ENTITY Atilde "Ã"><!-- capital A, tilde -->
....

ISOLat.dtd {
<!ENTITY % isolat1 SYSTEM "ISO-10646.dtd">
%isolat1;
<!ELEMENT Libros ANY>

```
<?xml version="1.0" encoding="utf-8" ?>  
<!DOCTYPE Libros SYSTEM "ISOLat.dtd">  
<Libros>  
    <LoQueQuiera>Y tambi&eacute;n lo que &lt;quiera> </LoQueQuiera>  
</Libros>
```

Namespaces

- ❏ XML permite crear etiquetas 'casi' sin ninguna limitación en sus nombres.
- ❏ Esto implica que, mezclar dos documentos, con diferentes etiquetas, podría resultar en una duplicidad de etiquetas.
- ❏ Mediante la definición de espacios de nombres, se pueden evitar estas colisiones.
- ❏ Tecnologías como XSL y otras muchas hacen uso de *Namespaces*.

Namespaces. Definición

- Un *namespace* se identifica por su prefijo.

Ejemplo:

```
<xsl:stylesheet  
  xmlns:xsl="http://www.w3.org/XSL/Transform/1.0">
```

- Donde:

- xsl es el prefijo del *namespace*.
- Stylesheet es el nombre completo del *namespace*.
- <http://www....> es la URI donde se puede encontrar más información sobre el estándar.
- Puede incluir otros atributos como *version...*
- Como todo elemento XML, ha de cerrarse.

```
</xsl:stylesheet>
```

Namespaces. Utilización

- Tras la definición del espacio de nombres, los elementos que pertenezcan a dicho espacio deberán prefijarse.

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

```
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
```

```
  <xsl:template match = "/">
```

```
    ...
```

```
  </xsl:template>
```

```
  <xsl:template match = "Libro">
```

```
    ...
```

```
  </xsl:template>
```

```
</xsl:stylesheet>
```

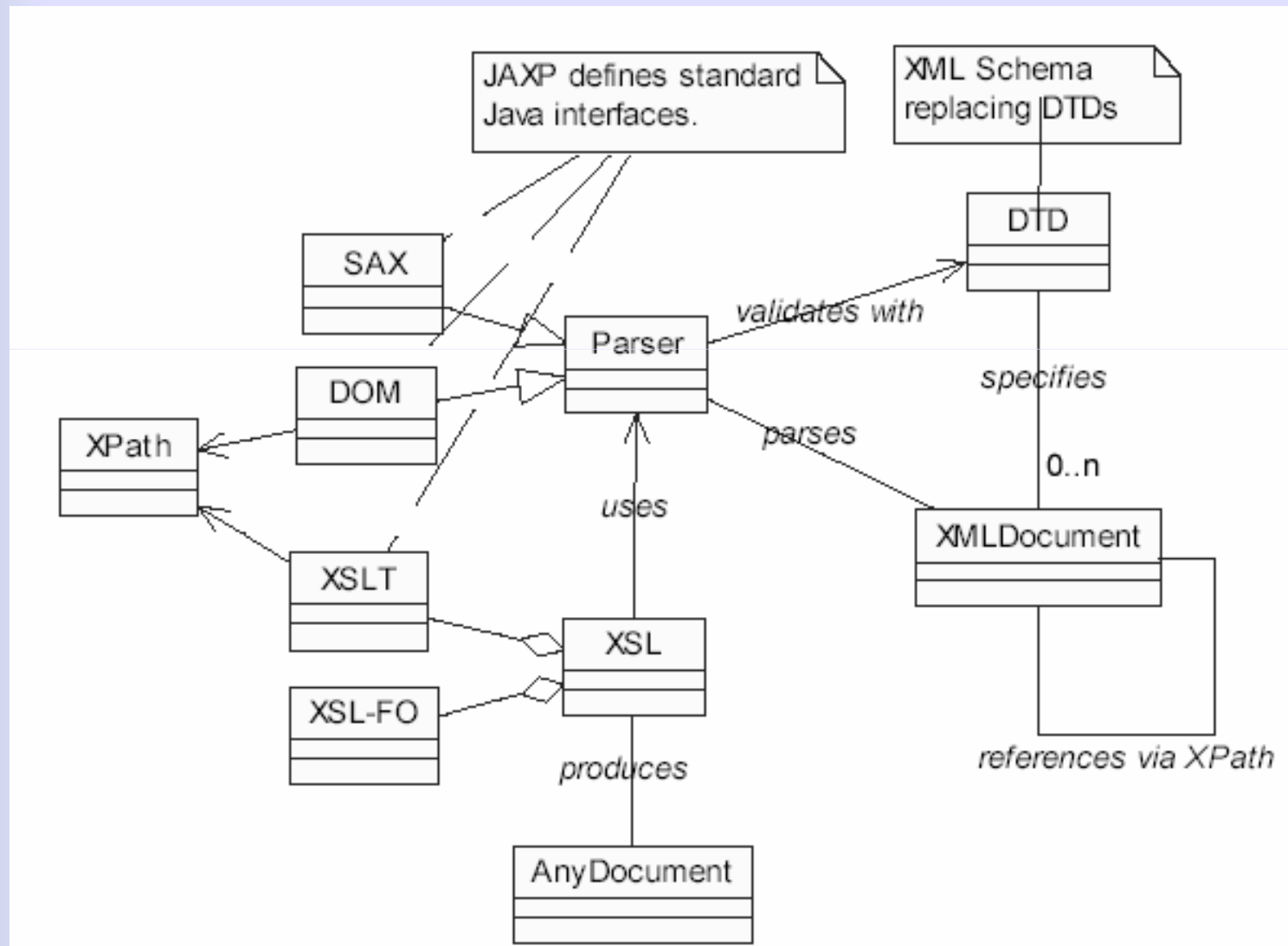
La familia de XML (i)

- Para sacar todo el partido a XML, éste viene acompañado de un conjunto de lenguajes hermanos:
 - XSL (eXtensible StyleSheet Language): junto con CSS es utilizado para modificar el aspecto de los documentos XML.
 - XSLT (eXtensible StyleSheet Language Translation): extensión de XSL que permite transformar un documento XML en otro diferente.
 - XSL-FO: objetos de formateo que gobiernan cómo los datos transformados se presentan al usuario.
 - DOM (Document Object Model): componente creado por Microsoft para el *procesamiento* de documentos XML.
 - SAX: *procesador* XML orientado a eventos.

La familia de XML (ii)

- XLink/XPointer: permiten referenciar a diferentes recursos, dentro o fuera del documento XML.
- XQL (XML Query Language): útil para localizar y extraer elementos de un documento XML.
- XPath: lenguaje de consulta para recorrer ficheros XML.

La familia de XML (iii)



Dominios de utilización de XML

- Intercambio de datos sobre fármacos.
- Tratamiento de información matemática (XMath).
- Intercambio de información entre programas ejecutables (SOAP).
- Intercambio de información entre herramientas CASE (XMI).
- Intercambio de información sobre RRHH (HR-XML).
- Intercambio de información sobre bolsa y financiera (IFX).
- Amplia utilización en el sector de EDI (*Electronic Data Exchange*).
- Estándares 'Web' como WML y XHTML.
- ...