



DTD

Lenguajes de Marcas y Sistemas de Gestión de la Información

CIFP La Laboral

Profesora: Inés Menéndez

¿QUÉ ES UN DTD?

- DTD significa Document Type Definition : Definición del Tipo de Documento
- ¿Qué define?
 - Los tipos de elementos, atributos y entidades permitidas que pueden aparecer en el documento XML
- Un DTD puede ser declarado en línea dentro de un documento XML, o como una referencia externa

¿QUÉ ES UN DTD?

- Los documentos XML que se ajustan a su DTD, se denominan “válidos”.
- Un documento XML “bien formado” simplemente respeta la estructura y sintaxis definidas por la especificación de XML. Un documento “bien formado” puede además ser “válido” si cumple las reglas de una DTD determinada.

DECLARACIÓN DE DTD INTERNA

- ▶ Debe seguir la siguiente sintaxis:

<!DOCTYPE elemento raíz [declaración de elementos]>

- ▶ Ejemplo de documento XML con una DTD interna:

```
<?xml version="1.0"?>
<!DOCTYPE mensaje[
  <!ELEMENT mensaje (para, de, encabezado, cuerpo)>
  <!ELEMENT para (#PCDATA)>
  <!ELEMENT de (#PCDATA)>
  <!ELEMENT encabezado (#PCDATA)>
  <!ELEMENT cuerpo (#PCDATA)>
]>
<mensaje>
  <para>Jaime</para>
  <de>Margarita</de>
  <encabezado>Recordatorio</encabezado>
  <cuerpo>NO olvides comprar la comida del perro</cuerpo>
</mensaje>
```

DTD EXTERNA

- ▶ Debe seguir la siguiente sintaxis:

<!DOCTYPE elemento raíz SYSTEM “archivo”>

```
<?xml version="1.0"?>
<!DOCTYPE mensaje SYSTEM "nota.dtd">
<mensaje>
<para>Jaime</para>
<de>Margarita</de>
<encabezado>Recordatorio</encabezado>
<cuerpo>NO olvides comprar la comida del perro</cuerpo>
</mensaje>
```

mensaje.xml

nota.dtd

```
<!ELEMENT mensaje (para, de, encabezado, cuerpo)>
<!ELEMENT para (#PCDATA)>
<!ELEMENT de (#PCDATA)>
<!ELEMENT encabezado (#PCDATA)>
<!ELEMENT cuerpo (#PCDATA)>
```

¿POR QUÉ UTILIZAR UNA DTD?

- Con una DTD, cada uno de los archivos XML puede llevar una descripción de su propio formato.
- Con una DTD, grupos independientes de personas se ponen de acuerdo para utilizar una DTD estándar para intercambiar datos.
- Su aplicación puede utilizar una norma DTD para verificar que los datos que recibimos del mundo exterior es válida.
- También puede utilizar un DTD para verificar sus propios datos.

¿POR QUÉ NO? – LIMITACIONES DTD

- La sintaxis no es XML (son difíciles de manipular).
- En las DTDs no se diferencia entre los distintos tipos de datos posibles: No existe el concepto de tipo de datos
- No son la mejor opción para las aplicaciones orientadas al intercambio de datos en las que el tipo de los datos es crucial.
- No es posible establecer restricciones, por ejemplo no se puede definir que un atributo 'edad' no pueda tomar un valor menor que cero.

DECLARACIÓN DE TIPO DE DOCUMENTO: <!DOCTYPE>

- Es la instrucción donde se indica qué DTD validará el XML. Aparece al comienzo del documento XML. El primer dato que aparece es el nombre del elemento raíz del documento XML.
- En función del tipo de DTD la sintaxis varía. Las características que definen el tipo son:
 - Ubicación: dónde se localizan las reglas del DTD.
 - Interno: las reglas aparecen en el propio documento XML.
 - Externo: las reglas aparecen en un archivo independiente.
 - Mixto: mezcla de los anteriores , las reglas aparecen en ambos lugares. Las reglas internas tienen prioridad sobre las externas.
 - Carácter: si es un DTD para uso privado o público.
 - Para uso privado se identifica por la palabra SYSTEM.
 - Para uso público: se identifica por la palabra PUBLIC. Debe ir acompañado del FPI (Identificador Público Formal), una etiqueta que identifica al DTD de manera 'universal'.

DECLARACIÓN DE TIPO DE DOCUMENTO: <!DOCTYPE>

○ Ejemplo 1:

Se declara un DTD respecto a un documento XML cuyo elemento raíz se llama *html* y que es de carácter público:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional/"EN"  
http://www.w3.org/TR/xhtml11/DTD/xhtml-transitional.dtd>
```

Este ejemplo sigue la sintaxis:

```
<!DOCTYPE elemento-raíz PUBLIC FPI URL>
```

La url sólo existe cuando el DTD se encuentra declarado en un archivo externo, del que se da su ubicación.

○ Ejemplo 2:

Se declara el tipo de un documento XML cuyo elemento raíz se llama *<planetas>*. Se trata de un DTD de uso privado (SYSTEM) y la ubicación de las reglas del DTD es externa, en un archivo llamado '*planetas.dtd*', que se encuentra en la misma carpeta que el archivo XML.

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

Este ejemplo sigue la sintaxis:

```
<!DOCTYPE elemento-raíz SYSTEM URL>
```

DECLARACIONES DTD

○ Tipos de declaraciones:

- ELEMENT
 - Declara un nombre de tipo de elemento y sus posibles subelementos
- ATTLIST
 - Lista de atributos de un elemento
- ENTITY
 - Declara referencias a caracteres especiales o a bloques de texto (similar a un #define) o también a contenido repetitivo que puede estar contenido en un recurso externo (similar a un #include).
- NOTATION
 - Declara contenido no-XML externo (p.e. Imágenes) y la aplicación externa que gestiona dicho contenido
 - Facilitan la inclusión de formatos binarios (imágenes...)

DECLARACIÓN DE ELEMENTO

- ▶ Los elementos son la base de las marcas XML, y deben ajustarse a un tipo de documento declarado en una DTD para que el documento XML sea declarado válido.
- ▶ Las declaraciones de tipo de elemento deben empezar con `<!ELEMENT` seguidas por el identificador genérico del elemento que se declara:

`<!ELEMENT nombre-elemento categoría>`

O

`<!ELEMENT nombre-elemento (elemento)>`

DECLARACIÓN DE ELEMENTO

- La declaración de un elemento en la DTD indica:
 - El nombre del elemento
 - El contenido que puede tener, también llamada ‘declaración de contenido’.
 - La ‘declaración de contenido’ se escribe entre paréntesis, y en ella se puede indicar:
 - El nombre de otros elementos. No es necesario haber declarado un elemento para poder utilizarlo en la declaración de contenido de otro elemento en la misma DTD.
 - La palabra reservada #PCDATA, que indica que el elemento puede contener datos de tipo carácter.

ELEMENTO VACÍO: EMPTY

- ▶ Los elementos vacíos se declaran con la palabra clave EMPTY

`<!ELEMENT nombre-elemento EMPTY>`

- ▶ Ejemplo DTD:

`<!ELEMENT br EMPTY>`

- ▶ Ejemplo XML:

`
`

ELEMENTO CON CUALQUIER CONTENIDO: ANY

- ▶ Los elementos que pueden tener cualquier contenido son declarados con ANY. Puede contener cualquier combinación de los datos apta para su procesamiento:

`<!ELEMENT nombre-elemento ANY>`

- ▶ Ejemplo DTD:

`<!ELEMENT mensaje ANY>`

- ▶ Ejemplo XML:

```
<mensaje>
<para>Jaime</para>
<de>Margarita</de>
<encabezado>Recordatorio</encabezado>
<cuerpo>NO olvides comprar la comida del perro</cuerpo>
</mensaje>
```

ELEMENTOS DE DATOS: PCDATA

- ▶ Los elementos con datos de caracteres se analizan con la declaración (#PCDATA):

`<!ELEMENT nombre-elemento (#PCDATA)>`

- ▶ Esta declaración de contenido indica que el elemento puede contener cualquier tipo de texto que no sea 'mark up', esto es cualquier letra menos `<` `>` `&`
- ▶ En lugar de estos caracteres se usarán las entidades `<` `>` ó `&`;
- ▶ Las comillas simples y doble pueden sustituirse por las entidades `"` y `'`;
- ▶ Ejemplo DTD:
`<!ELEMENT para (#PCDATA)>`
- ▶ Ejemplo XML:
`<para>Jaime</para>`

ELEMENTO CON HIJOS

- Los elementos con uno o más hijos se declaran con el nombre de los elementos hijo entre paréntesis:

`<!ELEMENT nombre-elemento (hijo1)>`

`<!ELEMENT nombre-elemento (hijo1, hijo2, ...)>`

Ejemplo DTD:

```
<!ELEMENT mensaje (para, de, encabezado, cuerpo)>
<!ELEMENT para (#PCDATA)>
<!ELEMENT de (#PCDATA)>
<!ELEMENT encabezado (#PCDATA)>
<!ELEMENT cuerpo (#PCDATA)>
```

Hijos

Ejemplo XML:

```
<mensaje>
<para>Jaime</para>
<de>Margarita</de>
<encabezado>Recordatorio</encabezado>
<cuerpo>NO olvides comprar la comida del perro</cuerpo>
</mensaje>
```


OCURRENCIA DE ELEMENTOS HIJO

- ▶ El elemento hijo aparece una y sólo una vez:
`<!ELEMENT nombre-elemento (hijo)>`
- ▶ El elemento hijo puede aparecer 0 o 1 vez:
`<!ELEMENT nombre-elemento (hijo?)>`
- ▶ El elemento hijo puede aparecer 0 o más veces:
`<!ELEMENT nombre-elemento (hijo*)>`
- ▶ El elemento hijo puede aparecer 1 o más veces:
`<!ELEMENT nombre-elemento (hijo+)>`
- ▶ Lista de elección, sólo puede aparecer uno de los elementos:
`<!ELEMENT nombre-elemento (hijo1 | hijo2)>`

DECLARACIÓN DE ATRIBUTOS

- ▶ En una DTD los atributos se declaran con una declaración ATTLIST:

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

- ▶ Ejemplo DTD:

`<!ATTLIST pago tipo CDATA "cheque">`



- ▶ Ejemplo XML:

`<pago tipo="cheque" />`

TIPOS DE ATRIBUTO

Tipo	Descripción
CDATA	El valor es un dato de carácter
(en1 en2 ...)	El valor debe ser uno de una lista enumerada
ID	El valor es un identificador único. No puede empezar por un número y no puede tener espacios en blanco.
IDREF	El valor es el identificador de otro elemento: apunta a un ID
IDREFS	El valor es una lista de identificadores separados por espacios
NMTOKEN	El valor es una palabra sin espacios: letras, números, puntos, guiones, subrayados y los dos puntos
NMTOKENS	El valor es una lista de múltiples NMTOKEN
ENTITY	El valor es una entidad
ENTITIES	El valor es una lista de entidades
NOTATION	El valor es un nombre de una notación
xml:	El valor es un valor predefinido xml

VALOR DE LOS ATRIBUTOS

<i>Valor</i>	Explicación
<i>Valor</i>	El valor predeterminado del atributo
<i>#REQUIRED</i>	El atributo es obligatorio y por tanto no tiene valor por defecto
<i>#IMPLIED</i>	El atributo es opcional y no se adopta automáticamente un valor por defecto
<i>#FIXED valor</i>	El valor del atributo es fijo

EJEMPLOS DE ATRIBUTOS

DTD:

```
<!ATTLIST contacto fax CDATA #IMPLIED>
```

XML:

```
<contacto fax="985-111111" />
```

DTD:

```
<!ATTLIST persona numero CDATA #REQUIRED>
```

XML:

```
<persona numero="5666" />
```

DTD:

```
<!ATTLIST novela fecha CDATA #IMPLIED lugar CDATA "Gijón">
```

XML:

```
<novela fecha="1885" lugar "Oviedo">
```

¿USAR ELEMENTOS O ATRIBUTOS?

- Se recomienda usar los atributos lo menos posible.
- Algunos de los problemas con los atributos son:
 - No pueden contener varios valores
 - No son fácilmente extensibles, para futuros cambios
 - Son más difíciles de manipular por el código del programa
 - Los valores de los atributos no son fáciles de probar con una DTD
- Sólo utilizar los atributos como identificador único

ENTIDADES

- XML hace referencia a objetos (ficheros, páginas web, imágenes, ...) que no deben ser analizados sintácticamente según las reglas de XML mediante el uso de entidades. Se declaran en la DTD con `<!ENTITY`
- Una entidad puede no ser más que una abreviatura que se utiliza como una forma corta de algunos textos. Al usar una referencia a esta entidad, el analizador sintáctico reemplaza la referencia con su contenido. En otras ocasiones es una referencia a un objeto externo o local.

TIPOS DE ENTIDADES

- Las entidades pueden ser:

- | | | |
|--------------|---|---------------|
| • Internas | O | Externas |
| • Analizadas | O | No analizadas |
| • Generales | O | Parámetro |

TIPOS DE ENTIDADES

- Entidades generales internas:
 - Son básicamente abreviaturas definidas en la sección de la DTD del documento XML. Son siempre entidades analizadas, es decir, una vez reemplazada la referencia a la entidad por su contenido, pasa a ser parte del documento XML y como tal, es analizada por el procesador XML.

```
<!DOCTYPE texto[  
  <!ENTITY ovni "Objeto Volante No  
    Identificado">  
>  
<texto><titulo>Un día en la vida de un &ovni;  
</titulo></texto>
```

TIPOS DE ENTIDADES

- Entidades generales externas analizadas:
 - Las entidades externas obtienen su contenido en cualquier otro sitio del sistema, ya sea otro archivo del disco duro, una página web o un objeto de una base de datos. Se hace referencia al contenido de una entidad así mediante la palabra SYSTEM seguida de un URI (Universal Resource Identifier)

```
<!ENTITY intro SYSTEM  
  "http://server.com/intro.xml">
```

TIPOS DE ENTIDADES

- Entidades no analizadas:

- Si el contenido de la entidad es un archivo MPEG o una imagen GIF o un fichero ejecutable EXE, el procesador XML no debería intentar interpretarlo como si fuera texto XML. Este tipo de entidades son siempre generales y externas.

```
<!ENTITY logo SYSTEM  
  "http://server.com/logo.gif">
```

EJEMPLO DE DTD Y XML

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE lista[
<!ELEMENT lista (persona)+>
<!ELEMENT persona (nombre, email*, relacion?)>
<!ATTLIST persona id ID #REQUIRED>
<!ATTLIST persona sexo (hombre | mujer) #IMPLIED>
<!ELEMENT nombre (#PCDATA)>
<!ELEMENT email (#PCDATA)>
<!ELEMENT relacion EMPTY>
<!ATTLIST relacion amigo-de IDREFS #IMPLIED
                    enemigo-de IDREFS #IMPLIED>
]>

<lista>
  <persona sexo="hombre" id="alvaro">
    <nombre>Álvaro Álvarez</nombre>
    <email>alvaroa@alvarez.com</email>
    <relacion amigo-de="beatriz"/>
  </persona>
  <persona sexo="mujer" id="beatriz">
    <nombre>Beatriz Bea</nombre>
    <email>beatrizb@bez.com</email>
  </persona>
</lista>
```