Document Type Definiton (DTD)

Crear una definición del tipo de documento (DTD) es como crear nuestro propio lenguaje de marcado, para una aplicación específica. Por ejemplo, podríamos crear un DTD que defina una tarjeta de visitas. A partir de ese DTD, tendríamos una serie de elemento XML que nos permitirían definir tarjetas de visita.

La DTD define los tipos de elementos, atributos y entidades permitidas, y puede expresar algunas limitaciones para combinarlos.

Los documentos que se ajustan a su DTD, se denominan "válidos". El concepto de "validez" no tiene nada que ver con el de estar "bien formado". Un documento "bien formado" simplemente respeta la estructura y sintaxis definida por la especificación de XML. Un documento "bien formado" puede además ser "válido" si cumple las reglas de una DTD determinada. También existen documentos XML sin una DTD asociada, en ese caso no son "válido", pero tampoco "inválido"... simplemente "bien formados"... o no.

Una DTD puede residir en un fichero externo, y quizás compartido por varios (puede que miles) de documentos. O bien, puede estar contenido en el propio documento XML, como parte de su declaración de tipo de documento.

Veamos un ejemplo

```
<! DOCTYPE etiqueta[
<!ELEMENT etiqueta (nombre, calle, ciudad, pais, codigo)>
<!ELEMENT nombre (#PCDATA)>
<!ELEMENT calle (#PCDATA)>
<!ELEMENT ciudad (#PCDATA)>
<!ELEMENT pais (#PCDATA)>
<!ELEMENT codigo (#PCDATA)>
]>

<etiqueta>
<nombre>Fulano Mengánez</nombre>
<calle>c/ Mayor, 27</calle>
<ciudad>Valderredible</ciudad>
<pais>España</pais>
<codigo>39343</codigo>
</etiqueta>
```

La declaración del tipo de documento empieza en la primera línea y termina con "]>". Las declaraciones DTD son las líneas que empiezan con "<!ELEMENT" y se

denominan declaraciones de tipo elemento. También se pueden declarar atributos, entidades y anotaciones para una DTD.

En el ejemplo anterior, todas las declaraciones DTD se definen "etiquetas" residen dentro del documento. Sin embargo, la DTD se puede definir parcial o completamente en otro ejemplo. Por ejemplo:

```
<?xml version="1.0"?>
<!DOCTYPE coche SYSTEM "http://www.sitio.com/dtd/coche.dtd">
<coche>
<modelo>...</modelo>
...
</coche>
```

Declaraciones Tipo Elemento

Los elementos son la base de las marcas XML, y deben ajustarse a un tipo de documento declarado en un DTD para que el documento XML sea considerado válido.

Las declaraciones de tipo de elemento deben empezar con "<!ELEMENT" seguidas por el identificador genérico del elemento que se declara. A continuación tienen una especificación de contenido.

Por ejemplo:

```
<!ELEMENT receta (titulo, ingredientes, procedimiento)>
```

En este ejemplo, el elemento <receta> puede contener dentro elementos <titulo>, <ingredientes> y <procedimiento>, que, a su vez, estarán definidos también en la DTD y podrán contener más elementos.

Siguiendo la definición de elemento anterior, este ejemplo de documento XML sería válido:

```
<receta>
<titulo>...</titulo>
<ingredientes>...</ingredientes>
<procedimiento>...</procedimiento>
</receta>
```

Pero no este:

```
<receta>
<parrafo>Esto es un párrafo</parrafo>
<titulo>...</titulo>
<ingredientes>...</ingredientes>
<procedimiento>...</procedimiento>
</receta>
```

La especificación de contenido puede ser de cuatro tipos:

EMPTY

Puede no tener contenido. Suele usarse para los atributos.

```
<!ELEMENT salto-de-pagina EMPTY>
```

ANY

Puede tener cualquier contenido. No se suele usar, ya que es conveniente estructurar adecuadamente nuestros documenteo XML.

```
<!ELEMENT batiburrillo ANY>
```

Mixed

Puede tener caracteres de tipo dato o una mezcla de caracteres y sub-elementos especificados en la especificación de contenido mixto.

```
<!ELEMENT enfasis (#PCDATA)>

<!ELEMENT parrafo (#PCDATA|enfasis)*>
```

Por ejemplo, el primer elemento definido en el ejemplo (<enfasis>) puede contener datos de carácter (#PCDATA). Y el segundo (<parrafo>) puede contener tanto datos de carácter (#PCDATA) como subelementos de tipo <enfasis>.

Element

Sólo puede contener sub-elementos especificados en la especificación de contenido.

```
<!ELEMENT mensaje (remite, destinatario, texto)>
```

Para declarar que un tipo de elemento tenga contenido de elementos se especifica un modelo de contenido en lugar de una especificación de contenido mixto o una de las claves ya descritas.

Modelos de contenido

Un modelo de contenido es un patrón que establece los sub-elementos aceptados, y el orden en que se acepta.

Un modelo sencillo puede tener un solo tipo de sub-elemento:

```
<!ELEMENT aviso (parrafo)>
```

Esto indica que <aviso> sólo puede contener un solo <parrafo>.

```
<!ELEMENT aviso (titulo, parrafo)>
```

La coma, en este caso, denota una secuencia. Es decir, el elemento <aviso> debe contener un <titulo> seguido de un <parrafo>.

```
<!ELEMENT aviso (parrafo | grafico)>
```

La barra vertical "|"indica una opción. Es decir, <aviso> puede contener o bien un <parrafo> o bien un <grafico>. El número de opciones no está limitado a dos, y se pueden agrupar usando paréntesis.

```
<!ELEMENT aviso (titulo, (parrafo | grafico))>
```

En este último caso, el <aviso> debe contener un <titulo> seguido de un <parrafo> o un <grafico>.

Además, cada partícula de contenido puede llevar un indicador de frecuencia, que siguen directamente a un identificador general, una secuencia o una opción, y no pueden ir precedidos por espacios en blanco.

Indicadores de frecuencia</B< caption> ?	Opcional (0 o 1 vez)
*	Opcional y repetible (0 o más veces)
+	
Necesarui y repetible (1 o más veces)	

Para entender esto, vamos a ver un ejemplo.

```
<!ELEMENT aviso (titulo?, (parrafo+, grafico)*)>
```

En este caso, <aviso> puede tener <titulo> o no (pero sólo uno), y puede tener cero o más conjuntos <parrafo><grafico>, <parrafo><parrafo><grafico>, etc.

Declaraciones de lista de atributos

Los atributos permiten añadir información adicional a los elementos de un documento. La principal diferencia entre los elementos y los atributos, es que los atributos no pueden contener sub-atributos. Se usan para añadir información corta, sencilla y desestructurada.

```
<mensaje prioridad="urgente">
<de>Alfredo Reino</de>
<a>Hans van Parijs</a>
<texto idioma="holandés">
Hallo Hans, hoe gaat het?
...
</texto>
</mensaje>
```

Otra diferencia entre los atributos y los elementos, es que cada uno de los atributos só se puede especificar una vez, y en cualquier orden.

En el ejemplo anterior, para declara la lista de atributo de los elementos <mensaje> y <texto> haríamos lo siguiente:

```
<!ELEMENT mensaje (de, a, texto)>
<!ATTLIST mensaje prioridad (normal | urgente) normal>
<!ELEMENT texto (#PCDATA)>
<!ATTLIST texto idioma CDATA #REQUIRED>
```

Las declaraciones de los atributos empiezan con "<!ATTLIST", y a continuación del espacio en blanco viene el identificador del elemento al que se aplica el atributo. Después viene el nombre del atributo, su tipo y su valor por defecto. En el ejemplo anterior, el atributo "prioridad" puede estar en el elemento <mensaje> y puede tener el valor "normal" o "urgente", siendo "normal" el valor por defecto si no especificamos el atributo.

El atributo "idioma", pertenece al atributo texto, y puede contener datos de carácter CDATA. Es más, la palabra #REQUIRED significa que no tiene valor por defecto, ya que es obligatoria especificar este atributo.

A menudo interesa que se pueda omitir un atributo, sin que se adopte automáticamente un valor por defecto. Para esto se usa la condición "#IMPLIED". Por ejemplo, en una supuesta DTD que define la etiqueta <IMG> de HTML:

```
<!ATTLIST IMG URL CDATA #REQUIRED>
<!ALT CDATE #IMPLIED>
```

Es decir, el atributo "URL" es obligatorio, mientras que el "ALT" es opcional (y si se omite, no toma ningún elemento por defecto).

Tipos de atributos

Atributos CDATA y NMTOKEN

Los atributos CDATA (Character DATA)sonlos más sencillos, y pueden contener casi cualquier cosa. Los atributos NMTOKEN (NaMe TOKEN) son parecidos, pero sólo aceptan los caracteres válidos para nombrar cosas (letras, números, puntos, guiones, subrayados y los dos puntos).

```
<!ATTLIST mensaje fecha CDATA #REQUIRED>
<mensaje fecha="15 de Diciembre de 1999"> <!ATTLIST mensaje fecha
NMTOKEN #REQUIRED>
<mensaje fecha="15-12-1999">
```

Atributos enumerados y notaciones

Los atributos enumerados son aquellos que sólo pueden contener un valor de entre un número reducido de opciones.

```
<!ATTLIST mensaje prioridad (normal | urgente) normal>
```

Existe otro tipo de atributo parecido, llamado de notación (NOTATION). Este tipo de atributo permite al autor declarar que su valor se ajusta a una notación declarada.

```
<!ATTLIST mensaje fecha NOTATION (ISO-DATE | EUROPEAN-DATE) #REQUIRED>
```

Para declarar las notaciones, se utiliza "<!NOTATION" con una definición externa de la notación. La definición externa puede ser pública o un identificador del sistema para para la documentación de la notación, una especificación formal o un asistente de la aplicación que contenga objetos representados en la notación

```
<!NOTATION HTML SYSTEM "http://www.w3.org/Markup">
```

<!NOTATION HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">

Atributos ID e IDREF

El tipo ID permite que un tipo determinado tenga un nombre único que podrá ser referenciado por un atributo de otro elemento que sea de tipo IDREF. Por ejemplo, para implementar un sencillo sistema de hipervínculos en un documento:

```
<!ELEMENT enlace EMPTY>
<!ATTLIST enlace destino IDREF #REQUIRED>
<!ELEMENT capitulo (parrafo)*>
<!ATTLIST capitulo referencia ID #IMPLIED>
```

En este caso, una etiqueta <enlace destino="seccion-3"> haría referencia a un <capitulo referencia="seccion-3">, de forma que el procesador XML lo podría convertir en un hipervínculom, u otra cosa.

Declaración de entidades

XML hace referencia a objetos (ficheros, páginas web, imágenes, cualquier cosa) que no deben ser analizados sintácticamente según las reglas de XML, mediante el uso de entidades. Se declaran en la DTD mediante el uso de "<!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.

Las entidades pueden ser:

- **Internas o Externas**
- **Analizadas o No analizadas**
- **Generales o Párametro**

Entidades generales internas

Son las más sencillas. 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 alf "Alien Life From">
```

```
]>

<texto><titulo>Un día en la vida de un &alf</titulo></texto>
```

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://www.miservidor.com/intro.xml">
```

Entidades no analizadas

Evidentemente, si el contenido de la entidad es un archibo MPG 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 siempre son generales y externas.

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

Entidades parámetro internas y externas

Se denominan entidades parámetro a aquellas que sólo pueden usarse en la DTD, y no en el documento XML. Se puede utilizar para agrupar ciertos elementos del DTD que se repitan mucho. Se diferencian las entidades parámetro de las generales, en que para hacer referencia a ellas, se usa el símbolo "%" en lugar de "&" tanto para declararlas como para usarlas.

```
<!DOCTYPE texto [
<!ENTITY % elemento-alf "<!ELEMENT ALF (#PCDATA)>">
...
%elemento-alf;
]>
```

También puede ser externa:

```
<!DOCTYPE texto [
<!ENTITY % elemento-alf SYSTEM "alf.ent">
...
%elemento-alf;
]>
```

Ejemplos de DTD

Un ejemplo de DTD que puede servir para resumir todo lo visto hasta ahora podría ser un DTD que nos defina un lenguaje de marcado para una base de datos de personas con direcciones e-mail.

El fichero LISTIN.DTD podría ser algo así:

```
<?xml encoding="UTF-8"?>
<!ELEMENT listin (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 #IMPLID>
```

Basándonos en este DTD, podríamos escribir nuestro primer listín en XML de la siguiente manera:

```
<?xml version="1.0"?>
<!DOCTYPE listin SYSTEM "listin.dtd">
<listin>

<persona sexo="hombre" id="ricky">
<nombre>Ricky Martin</nombre>
<email>ricky@puerto-rico.com</email>
<relacion amigo-de="leatitia">
</persona>

<persona sexo="mujer" id="leatitia">
<nombre>Leatitia Casta</nombre>
<email>castal@micasa.com</email>
</persona>

</listin>
```