1. Introducción

El objetivo de una DTD (Document Type Definition) es definir los bloques de construcción de un documento XML.Un DTD define la estructura del documento con una lista de elementos y atributos legales.

2. Declaración de un DTD

Un DTD puede ser declarado dentro de un documento XML o como referencia externa.

Declaración interna de un DTD

Si el DTD es declarado dentro del fichero XML debería estar dentro de una definición DOCTYPE con la siguiente sintaxis:

```
<!DOCTYPE raiz [elementos]>
```

Un ejemplo de un XML con su DTD incluído:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mensaje [
<!ELEMENT mensaje (para, de, encabezado, cuerpo)>
<!ELEMENT para  (#PCDATA)>
<!ELEMENT de  (#PCDATA)>
<!ELEMENT encabezado  (#PCDATA)>
<!ELEMENT cuerpo (#PCDATA)>
]>
<mensaje>
<para>Juan</para>
<de>Pablo</de>
<encabezado>Recordatorio</encabezado>
<cuerpo>¡No te olvides de llamarme este fin de semana!</body>
</mensaje>
```

Este DTD se interpreta así:

- DOCTYPE mensaje, define el **elemento raíz** del documento.
- ELEMENT mensaje, define uno de los elementos que **contiene cuatro descendientes (**para, de, encabezado y cuerpo**)**.
- ELEMENT para, define un elemento de tipo #PCDATA.
- ELEMENT de, define un elemento de tipo #PCDATA.
- ELEMENT encabezado, define un elemento de tipo #PCDATA.
- ELEMENT cuerpo, define un elemento de tipo #PCDATA.

Declaración externa de un DTD

Si el DTD es declarado en un fichero externo al fichero XML debería estar dentro de una definición DOCTYPE con la siguiente sintaxis:

```
<!DOCTYPE raiz SYSTEM “filename” [elementos]>
Un ejemplo de un XML con su DTD externo:
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mensaje SYSTEM “mensaje.dtd”>
<mensaje>
<para>Juan</para>
<de>Pablo</de>
<encabezado>Recordatorio</encabezado>
<cuerpo>¡No te olvides de llamarme este fin de semana!</body>
</mensaje>
```

3. Los elementos de los documentos XML

Para construir un XML los elementos que se definen en su DTD son:

- Elementos.
- Atributos.
- Entidades.
- PDDATA.
- CDATA.

ELEMENTOS

Son los bloques principales de documentos HTML, por ejemplo “body” o “table” y para XML, por ejemplo “mensaje”.

ATRIBUTOS

Proporcionan información extra sobre los elementos y van al lado de éstos, por ejemplo:

```
<img src="computer.gif" />
```

ENTIDADES

Algunos caracteres tiene un significado especial en XML como los siguientes:

- < equivale al carácter <.
- > equivale al carácter >.
- & equivale al carácter &.
- " equivale al carácter “.
- ' equivale al carácer '.
- ñ equivale al carácter ñ.

PCDATA

Significa datos de carácter parseados encontrados entre las etiquetas de los elementos. Es texto que será parseado por un analizador de entidades y marcado.
Las etiquetas dentro del texto serán tratadas como marcado y las entidades serán expandidas.

CDATA

Significa datos de carácter encontrados entre las etiquetas de los elementos. Es texto que no debería ser parseado por un analizador de entidades y marcado.
Las etiquetas dentro del texto **NO** serán tratadas como marcado y las entidades serán expandidas.

4. Declarando elementos

Los elementos son declarados con:

```
<!ELEMENT nombre categoria>
```

o

```
<!ELEMENT nombre (contenido)>
```

Elementos vacios

Son declarados con la palabra EMPTY:

```
<!ELEMENT nombre EMPTY>
```

Elementos con PCDATA

Son declarados con #PCDATA entre paréntesis:

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

Elementos con cualquier contenido

Son declarados con la palabra ANY y puede contener cualquier combinación de datos:

```
<!ELEMENT nombre ANY>
```

Elementos con hijos

Son declarados con los hijos entre paréntesis:

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

Cuando los hijos son declarados en una secuencia separados por comas, los hijos tienen que aparecer en el mismo orden:

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

Declarando sólo una ocurrencia de cualquier elemento

Crearemos un solo elemento mensaje:

```
<!ELEMENT nombre (hijo)>
```

Declarando un mínimo de una ocurrencia de un elemento

Con el signo + el hijo debe aparecer una o más veces:

```
<!ELEMENT nombre (hijo+)>
```

Declarando cero o más acurrencias de un elemento

Con el signo * el hijo puede aparecer varias veces o no aparecer:

```
<!ELEMENT nombre (hijo*)>
```

Declarando cero o una ocurrencia de un elemento

Con el signo ? el hijo puede aparecer una vez o ninguna:

```
<!ELEMENT nombre (hijo?)>
```

Declarando uno u otro contenido (elemento)

Con el signo | el hijo puede ser uno de varios mostrados:

```
<!ELEMENT nombre (hijo1|hijo2)>
```

5. Declarando atributos

Se declaran los atributos con <!ATTLIST nombre_etiqueta nombre atributo tipo valor>:

```
<!ATTLIST pago tipo CDATA "VISA">
        que será...
<pago tipo="VISA"/>
```

El tipo de atributo será:

CDATA: El valor es un carácter de datos.

(uno|dos|..): El valor es uno de la lista.

ID : Un identificador único. Se utiliza para identificar elementos, es decir, caracterizarlos de manera única. Dos elementos no pueden tener el mismo valor en atributos de tipo ID. Además un elemento puede tener a lo sumo un atributo de tipo ID. El valor asignado a un atributo de este tipo debe ser un nombre XML válido

IDREF : Representa el valor de un atributo ID de otro elemento, es decir, para que sea válido, debe existir otro elemento en el documento XML que tenga un atributo de tipo ID y cuyo valor sea el mismo que el del atributo de tipo IDREF del primer elemento.

Ejemplo:

Se requiere representar un elemento <empleado> que tenga dos atributos, ideEmpleado e idEmpleadoJefe. El primero será de tipo ID y carácter obligatorio y el segundo de tipo IDEREF y carácter optativo.

```
<!ELEMENT semaforo(nombre,apellido)>
<!ATTLIST empleado
        ideEmpleado ID#REQUIRED
        IdeEmpleadoJefe IDREF #IMPLIED>
```

```
<empleados>
    <empleado ideEmpleado="e_111">....</empleado>
    <empleado ideEmpleado="e_222" ideEmpleadoJefe="e_111">
    ......
    </empleado>
</empleados>
```

Cada elemento <empleado> tiene un atributo idEmpleado, con un valor válido(nombre XML válido) y el segundo elemento <empleado> tiene un atributo ideEmpleadoJeJefe, cuyo valor ha de ser el mismo que el del atributo de tipo ID de otro elemento existente en el documento. En este caso este atributo idEmpleadoJefe del segundo <empleado> vale igual que el atributo idEmpleado del primer <empleado>

Un fragmento de XML no válido con respecto a las reglas recién definidas sería aquél en el cual el atributo ideEmpleadoJefe del primer <empleado> tenga un valor que no exista para ningún atributo de tipo ID de otro elemento del documento:

```
<empleados>
    <empleado ideEmpleado="e_111" ideEmpleadoJefe="e_333">....</empleado>
    <empleado ideEmpleado="e_222" ideEmpleadoJefe="e_111">
    ......
    </empleado>
</empleados>
```

IDREFS :Representa mútiples Ids de otros elementos, separados por espacios.

NMTOKEN :Cualquier nombre sin espacios en blanco en su interior. Los espacios en blanco anteriores o posteriores se ignorarán.

Ejemplo:

Se quiere declarar un atributo de tipo NMTOKEN de carácter obligatorio.

En el siguiente DTD la declaración del elemento <rio> y su atributo pais es:

```
<!ELEMENT rio(nombre)>
```

```
<!ATTLIST rio pais NMTOKEN #REQUIRED>
```

En el siguiente fragmento XML el valor del atributo pais es válido con respecto a la regla anterior:

```
<rio pais="EEUU">
<nombre>Misisipi</nombre>
```

En el siguiente documento XML el valor del atributo pais no es válido con respecto a la regla anteriormente definida por contener espacios en su interior.

```
<rio pais="Estados Unidos">
<nombre>Misisipi</nombre>
</rio>
```

NMTOKENS : Una lista de nombres, sin espacios en blancos en su interior(los espacios en blanco anteriores o posteriores se ignorarán), separados por espacios.

ENTITY : El valor es una entidad. Hay diferentes tipos de entidades y en función del tipo de entidad su sintaxis varía.

Referida a entidades generales, se utilizarán dentro del documento XML.

Sintaxis general: <!ENTITY nombre_entidad definicion_entidad>

Ejemplo: <!ENTITY rsa "Republica Sudafricana">

A continuación se usará en el XML anteponiendo al nombre de la entidad el carácter ampersand(&) y a continuación un carácter de punto y coma (;). El programa analizador del documento realizará la sustitución.

```
<pais>
    <nombre>&rsa</nombre>
    …
</pais>
```

Referencia entidades generales externas, ubicadas en otros archivos:

Sintaxis general: <!ENTITY nombre_entidad tipo_uso url_archivo>

Siendo el tipo de uso privado (SYSTEM) O público (PUBLIC)

EJEMPLO:

Se dispone de un archivo de texto autores.txt que contiene texto plano "Juan Manuel y José Ramón".

Se crea un documento XML que hace referencia a ese archivo de texto en forma de entidad externa. Al visualizar el documento la referencia a la entidad general externa se sustituirá por el texto contenido en el archivo.

```
<?xml version="1.0"?>
<!DOCTYPE escritores[
<!ELEMENT escritores(#PCDATA)>
<!ENTITY autores SYSTEM "autores.txt">
]>
<escritores>&autores;</escritores>
```

……

ENTITIES: El valor es una lista de entidades.

NOTATION : Este es un elemento avanzado en el diseño de DTDs. Se usa para especificar un formato de datos que no se de XML. Se usa con frecuencia para describir tipos MIME,como image/gif o image/jpg. Se utiliza para indicar un tipo de atributo al que se le permite usar un valor que haya sido declarado como notación en el DTD.

Sintaxis general:

```
<!NOTATION nombre_notacion SYSTEM "identificador externo">
```

Sintaxis general del atributo que la usa:

```
<!ATTLIST nombre_elemento nombre_atributo NOTATION valor_defecto
```

xml: El valor es un valor XML predefinido.

Los valores por defecto pueden ser:

valor: El valor por defecto del atributo que aparece si no se especificase ninguno.

#REQUIRED El atributo es requerido.

#IMPLIED: El atributo no es requerido.

#FIXED valor: El valor es fijado.

Un valor de atributo por defecto

Un ejemplo donde creamos un elemento y su atributo con un valor por defecto:

```
<!ELEMENT caja EMPTY>
<!ATTLIST caja color CDATA "roja">
<caja color="azul" />
```

#REQUIRED

Se usa cuando no tenemos valores para el atributo pero queremos que aparezca:

```
<!ATTLIST nombre atributo tipo #REQUIRED>
<!ATTLIST persona NIF CDATA #REQUIRED>
```

BIEN: <persona NIF=”7887889C”/>

MAL: <persona/>

#IMPLIED

Se usa cuando no tenemos valores para el atributo y no queremos forzar a que aparezca:

```
<!ATTLIST nombre atributo tipo #IMPLIED>
<!ATTLIST empresa FAX CDATA #IMPLIED>
```

BIEN: <persona FAX=”923 444 555”/>

BIEN: <persona/>

#FIXED

Se usa cuando queremos que un atributo tenga un valor fijo sin poder cambiarlo sino, da error:

```
<!ATTLIST nombre atributo tipo #FIXED “valor”>
<!ATTLIST empresa Pais CDATA #FIXED “España”>
```

BIEN: <empresa Pais=”España”/>

MAL: <empresa Pais=”Spain”/>

- **Valores enumerados para atributos**

Para cuando queramos acotar los valores a usar:

```
<!ATTLIST nombre atributo tipo #FIXED “uno|dos|tres”>
<!ATTLIST pago TIPO (VISA|metalico|cheque)>
<!ATTLIST pago TIPO (VISA|metalico|cheque) “VISA”>
```

BIEN: <pago TIPO=”metalico”/>

BIEN: <pago TIPO=”VISA”/>

MAL: <pago TIPO=””/> Tomaría el valor VISA por defecto

6. Ejemplos

Ejemplo n.º 1:

Veamos un ejemplo muy sencillo de un pequeño lenguaje de etiquetas para definir documentos FAQ (Frequently Asked Questions):

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE FAQ SYSTEM "FAQ.DTD">
<FAQ>
<INFO>
<TITULO> WINDOWS98? SOLO SI ME OBLIGAN! </TITULO>
<AUTOR> Fernando Damián Lorenzo García </AUTOR>
<EMAIL> Fer@yahoo.es </EMAIL>
<VERSION> 1.0 </VERSION>
<FECHA> 20.MAYO.99 </FECHA>
</INFO>
<PART NO="1">
<Q NO="1">
<QTEXT>¿Por qué tengo que instalar W95?</QTEXT>
<A> A mi también me parece una buena pregunta.</A>
</Q>
<Q NO="2">
<QTEXT>Y de que me vale?</QTEXT>
<A>Otra ingeniosa pregunta.</A>
</Q>
</PART>
</FAQ>
```

Para el lenguaje de marcas mostrado arriba tenemos el siguiente DTD:

```
<!ELEMENT FAQ (INFO, PART+)>
<!ELEMENT INFO (TITULO, AUTOR, EMAIL?, VERSION?, FECHA?)>
<!ELEMENT TITULO (#PCDATA)>
<!ELEMENT AUTOR (#PCDATA)>
<!ELEMENT EMAIL (#PCDATA)>
<!ELEMENT VERSION (#PCDATA)>
<!ELEMENT FECHA (#PCDATA)>
<!ELEMENT PART (Q+)>
<!ELEMENT Q (QTEXT, A)>
<!ELEMENT QTEXT (#PCDATA)>
<!ELEMENT A (#PCDATA)>
<!ATTLIST PART NO CDATA #IMPLIED TITLE CDATA #IMPLIED>
<!ATTLIST Q NO CDATA #IMPLIED>
```

Ejemplo n.º 2:

Se quiere construir un DTD que valde el siguiente documento XML . Persona.xml.

```
<?xml version=”1.0” encoding=”iso-8859-1”?>
<!DOCTYPE persona SYSTEM “persona.dtd”>
<persona dni=”12345678-L” estadCivil=”casado”>
        <nombre>María Pilar</persona>
        <apellido>Sánchez</apellido>
        <edad>60</edad>
        <enActivo/>
</persona>
```

El esquema XML asociado se quiere que cumpla las siguientes restricciones:

El atributo dni es un identificador obligatorio

El estado civil puede ser soltero, casado o divorciado. Por defecto es soltero
El elemente <enActivo> es optativo
El DTD persona.dtd:

```
<!ELEMENT persona(nombre, apellido,edad,enActivo?)>
<!ATTLIST        persona dni ID #REQUIRED
                 estadoCivil (Soltero|Casado|Duvorciado) ¨Soltero¨>
<!ELEMENT nombre (#PCDATA)>
<!ELEMENT apellido (#PCDATA)>
<!ELEMENT edad (#PCDATA)>
<!ELEMENT enActivo (#PCDATA) EMPTY>
```