DEFINICIÓN DE TIPOS. DTD

LIMA UD3 – Tema 1

IES Plurilingüe Antón Losada Diéguez

Tabla de contenido

Definición de tipos de datos. DTD

1. Introducción

Cuando se diseña un documento XML para un propósito en concreto, se define la estructura que ha de respetar para que el sistema lo reconozca. Aunque un documento esté bien formado siguiendo las reglas de XML, eso no implica que sea válido para el sistema.

Para establecer las reglas que un documento XML ha de seguir para ser válido para un sistema en concreto se usan las definiciones de tipos de datos o DTD.

2. DTD

Los DTD permiten definir la estructura del documento, qué elementos están dentro de cuales, cuales solo contienen texto, cuales nada, el orden de los mismos y los atributos de cada uno.

Para ello se usa la etiqueta *DOCTYPE*.

```
<!DOCTYPE nombre_raiz [

]>
```

Esta se estipula con el nombre del elemento raíz que representa. Entre los corchetes se estipularán todos los elementos y atributos que contiene, incluyendo el propio elemento raíz.

2.1. Archivo externo

El DTD también puede añadirse mediante un archivo externo indicando la ruta del archivo .dtd.

```
<!DOCTYPE raiz SYSTEM "archivo.dtd">
```

En este ejemplo, raíz es el nombre que le etiqueta base en la que empezará a afectar el dtd.

Pueden incluirse múltiples archivos e incluso combinar archivos con definición interna.

```
<!DOCTYPE raiz SYSTEM "archivo.dtd" [

]>
```

3. Declaraciones

Existen tres tipos de declaraciones en DTD, los elementos, los atributos y las entidades.

Estas se estipulan dentro de los corchetes si son internos o directamente en el archivo DTD luego importado.

Todas las declaraciones empiezan por <! Y terminan por >, sin / de cierre.

3.1. Elemento

Las declaraciones *ELEMENT* estipulan las etiquetas permitidas, su nombre y el contenido que pueden tener en su interior.

```
<!ELEMENT element-name (element-content)>
```

En este ejemplo, el *element-name* representa el nombre de la etiqueta, mientras que *element-content*, el tipo de contenido que admite en su interior.

3.1.1. Contenido de texto

Para el contenido textual, se estipula el contenido como *#PCDATA*.

```
<!ELEMENT element-name (#PCDATA)>
```

Con esta definición, el elemento puede contener texto o nada.

3.1.2. Contenido de otro elemento

Para anidar elementos, se estipula con el nombre de cada elemento que puede contener.

```
<!ELEMENT element-name (child1)>
```

Con esta definición, el elemento *element-name* contiene, obligatoriamente el elemento *child1* una única vez.

```
<!ELEMENT element-name (child1,child2,...)>
```

Para estipular varios hijos, se separan con comas. Hay que tener en cuenta que el elemento tiene que contener la secuencia de hijos por completo y en el mismo orden.

Todos los elementos hijo han de estar definidos en un DTD, ya sea interno o externo, antes o después de esta declaración.

Mediante el uso de los mismos caracteres que en las expresiones regulares, se puede establecer otra ocurrencia de los hijos.

```
<!ELEMENT element-name (child-name+)>
```

Con el + se estipula que el elemento aparece una o más veces.

```
<!ELEMENT element-name (child-name*)>
```

Con el *, se indica que el hijo aparece múltiples veces o ninguna.

```
<!ELEMENT element-name (child-name?)>
```

Con el ?, se estipula que el elemento aparece una vez o ninguna.

```
<!ELEMENT note (to,from,header,(message|body))>
```

Con la | se estipula un *OR* lógico, o uno u otro.

En este caso, los paréntesis agrupan esos dos como uno, indicando que aparece uno de esos dos obligatoriamente

```
<!ELEMENT note (#PCDATA|to|from|header|message)*>
```

Como se puede observar en este ejemplo, se puede combinar el uso del *OR* con otros caracteres. En este caso, se indica que aparecerá cualquiera de esos elementos o texto cero o más veces.

3.1.3. Contenido vacío

Si un elemento no ha de tener contenido, se estipula con *EMPTY*.

```
<!ELEMENT element-name EMPTY>
```

Con esta definición, el elemento no puede contener nada.

3.1.4. Cualquier contenido

Para cualquier contenido, se estipula el contenido como *ANY*.

```
<!ELEMENT element-name ANY>
```

Con esta definición, el elemento puede estar vacío o contener, texto, otros elementos o una combinación de texto y elementos.

3.2. Atributos

Las declaraciones *ATTLIST* estipulan los atributos de un elemento en concreto. Dicho elemento ha de estar declarado en el DTD.

```
<!ATTLIST element-name attribute-name attribute-type attribute-value>
```

En este ejemplo, *element-name* representa el nombre del elemento al que se le aplica el atributo, *attribute-name* es el nombre del atributo, *attribute-type* es el tipo de valor que admite el atributo y *attribute-value* es el valor que toma por defecto o la obligatoriedad del mismo dependiendo del tipo establecido.

A diferencia de los elementos, los atributos se estipulan uno a continuación de otro, habitualmente con un salto de línea entre ellos.

```
<!ATTLIST datos
    fecha CDATA #IMPLIED
    id ID #REQUIRED
>
```

En este ejemplo, el elemento datos tiene dos atributos distintos.

3.2.1. Valor textual

Con *CDATA* se estipula que el valor del atributo es texto. Si se estipula un valor, este será el valor por defecto para aquellos elementos sin atributo establecido.

```
<!ATTLIST payment type CDATA "check">
```

En este ejemplo, el elemento *payment* tiene un atributo textual *type* cuyo valor por defecto es *check*.

3.2.2. Valor enumerado

Mediante el uso de paréntesis y el *OR* lógico | se puede establecer una lista de posibles valores. A mayores, se puede incluir un valor por defecto.

```
<!ATTLIST payment type (check|cash) "cash">
```

En este caso, el elemento *payment* tiene un atributo *type* cuyo valor puede ser o *check* o *cash* y toma el valor *cash* si no se estipula ninguno.

3.2.3. Valor identificativo y de referencia

Con *ID* se estipula que el valor va a ser un identificador y, por tanto, no se puede repetir en ningún otro elemento. Este valor tiene que empezar por una letra, ":" o "_" y puede contener alfanuméricos ":", "_", "-" y ".".

Con *IDREF* se establece que el valor es un id de otro elemento, es decir, una referencia a otro elemento del documento, por lo que tiene que ser un valor que tenga otro elemento como atributo de tipo ID.

```
<!DOCTYPE elements [
    <!ELEMENT elements (elem1|elem2)*>
    <!ELEMENT elem1 EMPTY>
    <!ATTLIST elem1 id ID #IMPLIED>
    <!ELEMENT elem2 EMPTY>
    <!ATTLIST elem2 link IDREF #IMPLIED>
]>
```

```
<elements>
    <elem1 id="a1435"/>
    <elem2 link="a1435"/>
</elements>
```

En este ejemplo, el elemento *elem2* referencia con su atributo *link* al elemento *elem1* mediante su *id* establecido en el atributo *id*.

Hay que tener en cuenta que este tipo de atributo requiere estipular si es obligatorio o no y no admite valor por defecto.

Con *IDREFS* se estipula que el valor es una lista de *ids* de otros elementos separados por espacios.

3.2.4. Valor token XML

Con *NMTOKEN* se estipula como tipo *XML TOKEN*. Este tipo es similar al de texto normal, con la salvedad de que solo admite minúsculas, mayúsculas, números, puntos ".", guiones "-", guiones bajos "_" o dos puntos ":".

Con *NMTOKENS* se estipula una lista de *XML TOKEN* separados por espacios.

3.2.5. Valor de entidad

Con *ENTITY* se estipula que el valor tiene que ser una entidad previamente definida.

Con *ENTITIES* se define el valor como una lista de entidades separadas por espacios.

3.2.6. Valor requerido

Si un atributo es necesario, se establece como obligatorio con *#REQUIRED* en el lugar del valor por defecto.

```
<!ATTLIST person number CDATA #REQUIRED>
```

En este ejemplo, el elemento *person* tiene un atributo obligatorio *number* de tipo textual.

3.2.7. Valor opcional

Si un atributo no permite un valor por defecto, no se quiere incluir o no es necesario, se establece como opcional con *#IMPLIED* en el lugar del valor por defecto.

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

En este ejemplo, el elemento *contact* tiene un atributo opcional *fax* de tipo texto.

3.2.8. Valor fijo

Si un atributo ha de aparecer con un valor fijo, se establece con *#FIXED*, seguido del valor preestablecido.

```
<!ATTLIST sender company CDATA #FIXED "Microsoft">
```

En este caso, el elemento *sender* tiene un atributo *company* cuyo valor es siempre *Microsoft*.

3.3. Entidades

Las entidades permiten establecer atajos de escritura o sustitución de elementos. Estas se establecen con *ENTITY* y se utilizan en el XML con &nombre;.

```
<!ENTITY copyright "Copyright W3Schools.">
```

En este ejemplo, se establece como entidad copyright, que obtiene el valor de *Copyright W3Schools.* Cuando se escribe.

Otro ejemplo de entidades son las que tiene XML por defecto declaradas para sustituir los caracteres especiales. (> como >, por ejemplo)