Índice

Validación de un documento XML

Mediante DTD

1. a) ¿Qué es un DTD? b) ¿Para qué sirve?

a) Una DTD (**D**ocument **T**ype **D**efinition) es un conjunto de reglas para definir un documento XML y etiquetarlo adecuadamente. En una DTD definimos los "componentes" de un documento XML y cómo se relacionan y estructuran.

b) Sirve al programador para conocer cómo redactar el documento XML en cuestión, conociendo los elementos y atributos requeridos por dicha sintaxis.

2. a) ¿Qué es un documento bien formado? b) ¿Y un documento válido?

a) Documento bien formado es aquel que cumple con las reglas generales de los documentos XML.

b) Documento válido es el que además se adapta a las especificaciones de validación —por ejemplo mediante DTD— concretas de dicho documento.

3. Algunos fallos a evitar, para que un documento sea "bien formado"

No se considerará un documento XML como "bien formado" si tiene fallos como los siguientes:

- El documento no contiene un prólogo válido.
- Algunas etiquetas no se han cerrado.
- Los elementos no están conectados correctamente.
- Los valores de los atributos no se han colocado entre comillas.
- Los nombres de los elementos contienen caracteres no permitidos.
- No existe ningún elemento raíz en el documento

4. a) ¿Para qué sirve el término *standalone* del prólogo? b) ¿Para qué caso se pone "yes" y para cuál *"no"*?

a) El término ***standalone*** (del Inglés To Stay Alone: Estar solo) nos permite especificar si el documento es autosuficiente (se puede procesar sin necesidad de otro documento) o no.

b) El valor *"no"* de *standalone* significa que el documento depende de un documento DTD que especifique la gramática que le describe. El valor "yes" indica que puede procesarse por sí mismo, bien porque lleva el DTD incrustado o porque sólo depende de la gramática estándar XML.

5. ¿Cómo comienza un DTD?

Comienza con la etiqueta **<!DOCTYPE>** seguida del nombre de un elemento, que debe corresponder con el elemento raíz (Elemento documento)

```
<!DOCTYPE nombreDeLaRaíz [ declaraciones ]>
```

6. De qué dos maneras se vinculan los datos XML a un documento DTD?

1. Mediante la inserción del DTD dentro del archivo XML: En este caso, la sintaxis de la etiqueta **<!DOCTYPE>** es:

   ```
   <!DOCTYPE nombreDeLaRaíz [ declaraciones ]>
   ```

2. Mediante la integración de una referencia a un archivo DTD distinto. Para vincular un documento a un archivo DTD específico, basta con utilizar la siguiente instrucción justo después del prólogo —en el que hay que poner **standalone="no"**— y antes de los datos XML:

   ```
   <!DOCTYPE nombreDeLaRaíz SYSTEM "nombrefichero.dtd">
   ```

7. ¿Cómo se definen los elementos en un DTD?

Los elementos son los nodos del árbol de un documento XML. Se declaran mediante la etiqueta <**!ELEMENT**> .

8. ¿De qué puede estar compuesto un elemento?

Un elemento puede contener, a su vez...

- Otros elementos: `<!ELEMENT A (B, C)>`
- Datos textuales: `<!ELEMENT elem (#PCDATA)>`
- Nada: `<!ELEMENT elem (EMPTY)>`
- Cualquier cosa de las anteriores: `<!ELEMENT elem (ANY)>`

9. ¿Qué es un grupo? ¿Cómo se construye?

El primer caso de los presentados en la pregunta anterior es un grupo. Se trata de una composición de referencias a otros elementos o a subgrupos. Se construye en la declaración de un elemento, presentándolo entre paréntesis. Los elementos que declaran un grupo son elementos no terminales. Sin embargo, los elementos que contienen **EMPTY**, **ANY** o datos textuales, son elementos terminales (las hojas del árbol XML).

10.¿Para qué sirve la palabra clave EMPTY?

Impide a un elemento tener elementos hijos o datos textuales.

11.¿Para qué sirve la palabra clave ANY?

Permite mezclar todos los tipos de elementos declarados. Además puede contener cero o varios elementos hijos declarados o datos textuales

12.¿Para qué sirve el operador ... a) ? b) + c) * d) |

a) El operador **(?)** define un componente opcional.

```
<!—El sub-elemento fijo es opcional-->
<!ELEMENT numeroTel (fijo?, movil)>
```

b) El operador (**+**) define un componente que aparece al menos una vez.

```
<!—El subgrupo (codPost, ciudad) al menos aparece una vez-->
<!ELEMENT provincia (nombreProv, (codPost, población)+)>
```

c) El operador **(*)** define un componente que aparece cero o más veces.

```
<!—El grupo (dirIP, maquina) aparece cero o más veces-->
<!ELEMENT maquina (dirIP, maquina)*>
```

d) El operador **| (Alt Gr + 1)** significa que se pone sólo uno de los **n** elementos entre los que se encuentra.

Ejemplo: si tenemos la siguiente declaración

```
<!ELEMENT numeroTel (fijo | movil)>
<!ELEMENT fijo (#PCDATA)>
<!ELEMENT movil(#PCDATA)>
```

..., sería incorrecto el siguiente fragmento de documento XML:

```
<numeroTel>
  <movil>654654654</movil>
  <fijo>954959654</fijo>
</numeroTel>
```

…porque por fuerza sólo puede existir uno de los dos.

13.¿Para qué sirve (#PCDATA) ?

Cuando un elemento lleva la palabra reservada #PCDATA, quiere decir que su contenido sólo puede estar formado por datos de caracteres alfanuméricos. No admite otro tipo de datos que no sean los definidos.

```
<!ELEMENT producto (#PCDATA)>
...

<producto>DELL Vostro 7200</ producto >
```

Nota: El término PCDATA significa *Parsed Character DATA.*

14. ¿Cómo podemos combinar datos textuales con otros elementos? Sintaxis y condiciones.

Deben tener una estructura como la siguiente:

```
<!ELEMENT elem1 (#PCDATA | elem2 | elem3)*>
```

Estas declaraciones combinadas deben respetar los siguientes criterios:

- Los datos textuales **#PCDATA** aparecen en primer lugar de la declaración.
- El grupo será una elección (separaciones con el carácter **|**).
- El grupo debe aparecer cero o más veces (operador *) .

Ejemplo:

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

Para esta declaración valdrían cualquiera de estos dos ejemplos de datos XML:

```
<parrafo>
    Esto es una <negrita>frase</negrita>
</parrafo>
ó
<parrafo>Esto es una frase</parrafo>
```

En el primero de estos dos ejemplos se está repitiendo el bloque **(#PCDATA | negrita)*** dos veces: la primera con un elemento simple (#PCDATA) y la segunda con un elemento de tipo negrita

En el segundo ejemplo aparece el bloque sólo una vez con un elemento simple (#PCDATA).

15.¿Cómo se definen los atributos en un DTD?

Los elementos pueden tener cero o más atributos. Se declaran mediante la etiqueta `<!ATTLIST ... >`

16.¿Cómo se asocian los atributos a su elemento?

Después de abrir la etiqueta del atributo se pone el nombre del elemento al que va asociado, luego el nombre del atributo y su tipo. Por ejemplo:

```
<!ELEMENT capitulo (#PCDATA)>
<!ATTLIST capitulo numero ID #REQUIRED>
```

Estos son datos XML conformes a la anterior declaración:

```
<capitulo numero="id15">
  La Musaraña de la montaña
</capitulo>
```

17.Si un elemento tiene más de un atributo ¿De qué dos formas se asocian a su elemento?

1 Repitiendo la etiqueta y el nombre del elemento al que se asocia:

```
<!ELEMENT libro (titulo, capitulo+)>
<!ATTLIST libro autor CDATA #REQUIRED>
<!ATTLIST libro editorial CDATA #IMPLIED>
```

2 Sin repetir la etiqueta del elemento:

```
<!ELEMENT libro (titulo, capitulo+)>
<!ATTLIST libro
          autor CDATA #REQUIRED
          editorial CDATA #IMPLIED>
```

18.¿Qué significa que un atributo tenga la cláusula ... a) (#IMPLIED) b) (#REQUIRED) c) (#FIXED).

a) `#IMPLIED` Significa “es opcional”

b) `#REQUIRED` Significa “es obligatorio”

c) `#FIXED` Significa “su valor es fijo” y hay que indicar cuál va a ser.

En las preguntas anteriores vemos atributos obligatorios y opcionales. Un ejemplo de atributo fijo:

```
<!ATTLIST imagen formato CDATA #FIXED "jpge">
```

19.¿Cómo se asocia a un atributo opcional un valor por defecto?

Basta poner directamente el valor por defecto, en lugar de **#IMPLIED**, después de declarar el tipo de atributo:

```
<!ELEMENT empleado (nombre, apellidos, ...)>
<!ATTLIST empleado cargo CDATA "peón">
```

20. Tipos de atributos

CDATA, NMTOKEN, enumerados, NOTATION, ENTITY, ENTITIES, ID, IDREF.

21. El tipo CDATA

Los atributos CDATA (character data) son los más sencillos, y pueden contener casi cualquier cosa:

```
<!ATTLIST elemento1 fecha CDATA #REQUIRED>
<mensaje fecha="15 de Julio de 1999">
```

22. Las Enumeraciones

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

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

23.El tipo NOTATION

Este tipo de atributo permite asociar una aplicación a un tipo de información. Este tipo de atributo permite al autor declarar que su valor se ajusta a una notación declarada.

```
<!NOTATION jpg PUBLIC "JPEG">
```

Si queremos seleccionar una aplicación en particular hay que sustituir la palabra clave PUBLIC por SYSTEM:

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

Los atributos pueden ser del tipo de una notación previamente declarada. También es posible declarar una enumeración de notaciones declaradas para un atributo.

```
<!ATTLIST foto formato NOTATION (gif | jpg) #IMPLIED>
```

24. Los atributos ID o IDREF

Los atributos ID se comportan como identificador de un elemento, por tanto tienen que ser único en el documento y su nombre tiene que empezar por letra o guion bajo.

```
<!--id será el identificador del elemento receta-->
<!ATTLIST receta id ID #REQUIRED>
```

Los atributos IDREF son referencias a los identificadores, es decir, el valor de un IDREF tiene que ser el valor de uno de los ID existentes.

```
<!-- referencia apunta al identificador de otro elemento-->
<!ATTLIST ingrediente referencia IDREFS #IMPLIED>
```

Estos son datos XML conformes a la anterior declaración:

```
<receta id="rec_1"> Mousse de chocolate </receta>
<ingrediente ref="rec_1"> chocolate </ingrediente>
```

25.Los atributos de tipo NMTOKEN

El tipo *NMTOKEN* especifica los caracteres permitidos por XML. Los atributos pueden utilizar el tipo *NMTOKEN* o *NMTOKENS* para una lista de *NMTOKEN* separados por caracteres de espacio:

```
<!--Elatributo aa debe contener una palabra-->
<!--utilizando únicamente caracteres permitidos por XML-->
  <!ATTLIST A aa NMTOKEN #IMPLIED>

<!--El atributo bb debe contener una o varias palabras-->
<!--utilizando únicamente caracteres permitidos por-->
<!--XML y separados por el caracter espacio-->
  <!ATTLIST A bb NMTOKENS #IMPLIED>
```

Sería correcto:

```
<A aa="rec_1"> Mousse de chocolate </A>
<A bb="rec_1 rec_2"> Mousse de chocolate </A>
```

Sería INcorrecto

```
<A aa="rec?_1"> Mousse de chocolate </A>
<A aa="rec_1 rec_2"> Mousse de chocolate </A>
```

```
<ingrediente ref="rec_1"> chocolate </ingrediente>
```