

Lenguajes de Marcas y Sistemas de Gestión de Información

Bloque XML: UD3, Validación con DTDs

Recordamos..

2

- XML y su sintaxis.
- El prólogo del doc. XML:
 - aparecía el DOCTYPE
- Uso de editores o línea de comandos
 - que comprueban sintaxis,
 - pero además también que el documento es VÁLIDO
- Ejemplo inicial:
 - nota.xml y nota.dtd
 - enlazarlos..

Document Type Definition (DTD)

3

- Define la estructura un de documento xml:
 - lista de elementos
 - atributos válidos, opcionales, obligatorios
 - orden, número de ocurrencias, etc.
- Puede ser declarado *dentro del documento XML (inline)* o **como una referencia externa** (o los dos)
- ¿Por qué usar una DTD?
 - Tener una descripción de formato
 - Grupos distintos pueden acordar usar una DTD estandarizada para intercambiar datos
 - Las aplicaciones que usen una DTD estándar pueden verificar qué datos recibidos sean válidos
 - Usarla para verificar los propios datos..

Resumimos

4

- Un documento válido
 - Es bien formado
 - (puede incluir referencia a DTD o schema)
 - y cumple lo especificado en esa DTD o schema.
- El fichero xml tiene
 - Prólogo
 - El DOCTYPE debe estar incluida en el prólogo
 - Declara el tipo de documento, si es interno o externo, y dónde o cómo localizarlo para comprobar que es válido.
 - Instancia de datos: contiene el elemento raíz.

DTDs externas

5

- DTDs Privadas

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

- Pueden estar

- Almacenadas en local o localizadas en la web

`<!DOCTYPE tablon SYSTEM "http://w.ae.es/tablon.dtd">`

- Pública, pero la “empresa” tiene control, la puede modificar.

- DTDs de acceso público

- La DTD se considera un estándar, para uso público

`<!DOCTYPE tablon PUBLIC fpi URL>`

- Fpi: identificador público formal

`<!DOCTYPE chapter PUBLIC "-//OASIS//DTD DocBook XML V4.1.2//EN"`

`"http://www.oasis-open.org/docbook/xml/4.1.2/docbookx.dtd">`

DTD Externa

6

- Fichero XML: referencia a DTD externa en DOCTYPE:

```
<?xml version="1.0"?>
```

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

```
<nota>
```

```
  <de>Tove</de>
```

```
  <a>Jani</a>
```

```
  <tema>Reminder</tema>
```

```
  <texto>Don't forget me this weekend!</texto>
```

```
</nota>
```

- Fichero DTD: fichero "nota.dtd" con la DTD sería:

```
<!ELEMENT nota (de,a,tema,texto)>
```

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

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

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

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

DTD interna (inline)

7

- No es la mejor idea...
- Sólo tenemos un fichero XML, y DTD definida en él:

```
<?xml version="1.0"?>
  <!DOCTYPE note [
    <!ELEMENT nota (de,a,tema,texto)>
    <!ELEMENT de (#PCDATA)>
    <!ELEMENT a (#PCDATA)>
    <!ELEMENT tema (#PCDATA)>
    <!ELEMENT texto (#PCDATA)>
  ]>
  <nota>
    <de>Tove</de>
    <a>Jani</a>
    <tema>Reminder</tema>
    <texto>Don't forget me this weekend</texto>
  </nota>
```

Analizando la DTD

8

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

- Indica que el elemento raíz es nota
 - Y que la DTD está en el fichero nota.dtd

```
<!ELEMENT nota (de,a,tema,texto)>
```

- Indica que nota tiene 4 elementos hijos que deben aparecer en ese orden.

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

- elemento a del tipo "#PCDATA", no tiene hijos
 - Atención a los espacios en blanco
- Más tarde veremos cómo definir atributos

Formato de una DTD

9

- Para la DTD, un documento XML está compuesto de:
 - Elementos
 - Atributos (y sus tipos)
 - Datos
 - PCDATA
 - CDATA
 - Entidades (definidas por el usuario)
- Comentarios en DTD: como en XML
`<!--` Comentario como en xml `-->`

PCDATA y CDATA

10

- PCDATA : *parsed character data*
 - Es el texto que se encuentra entre las etiquetas de inicio y cierre de un elemento XML.
 - PCDATA es texto que será procesado y analizado por el parser, en busca de entidades u otras etiquetas.
 - no debería contener &, <, > ,etc, que deberían ser representados por las entidades & < and >
- CDATA: *character data*.
 - Lo usaremos en los atributos
 - Es texto que no será analizado por el parser.

Un elemento puede contener

11

- Otros elementos (elemento contenedor):
`<!ELEMENT nota (de, a, tema, texto)>`
 - Hijos que deben aparecer, en ese mismo orden
 - Faltan definiciones de cada elementos hijo
- Datos finales: PCDATA
`<!ELEMENT tema (#PCDATA)>`
- Elemento vacío: no texto, puede tener atributos
`<!ELEMENT sinDatos EMPTY>`
- Elemento ANY: no se comprueba nada
`<!ELEMENT batiburrillo ANY>`

- Ya visto en tema anterior: evitar y rediseñar..
- Ejemplo:

<!ELEMENT estadoInv (#PCDATA|orderMsg)*>

- El elemento estadoInv puede contener el número de elementos (PCDATA), o un elemento del tipo orderMsg (definido luego) con el estado del pedido, que además se pueden repetir

<estadoInv>

10

<orderMsg> pedido impresora </orderMsg>

</estadoInv>

DTD: Número de ocurrencias

13

- Una:
`<!ELEMENT note (message)>`
- **,** especifica una secuencia obligatoria de elementos
`<!ELEMENT alumno (nombre, apellidos, nota) >`
- **+** Al menos una:
`<!ELEMENT tablon (aviso+) >`
- ***** Cero o más:
`<!ELEMENT clase (alumno*)>`
- **?** Cero o una, elemento opcional
`<!ELEMENT aviso (de, a, texto, prioridad?) >`
- **|**, pipe, o una u otra:
`<!ELEMENT nota (de, a, (tema | texto))>`

Validar con DTD: herramientas

14

- Obtener ficheros nota.dtd y nota.xml
- Validar desde línea comandos

```
$ xmlstartlet val -e -d nota.dtd nota.xml
```
- Validar desde editor: enlazar con DTD
 - VS Code: "<!" y "Insert System DOCTYPE"
 - xmllcopyeditor:
 - Menú XML/Asociar/System DTD y ojo a la ruta absoluta/relativa
- Introducir errores en fichero de datos
 - elementos repetidos, que falte un elemento, un atributo; ver errores en editor y en línea de comandos
 - ¡¡Ver errores en el navegador!!

Ejercicio (sin atributos)

15

- Hacer DTD para ejemplo tablón compraventa
- Elementos: tablon, anuncio, fecha, asunto, texto, precio, contacto..
- Incluir DOCTYPE en fichero xml
 - Probar si es válido con editor
- Probar si es válido en CLI
- Comprobar que navegadores NO validan
- Modificar el fichero xml para que falte o sobre algún elemento.
 - Comprobar “bien formado” pero NO válido

+ sobre ejercicio tablón

16

- Indicar si puede haber cero anuncios..
- Texto (descripción) opcional, pero si aparece, sólo una vez
 - Probar que valida si no aparece, pero no si aparece dos veces..
- Contacto obligatorio, pero puede aparecer más de uno.
 - Comprobar que no valida sin el contacto, pero que permite más de uno
- Añadir elemento opcional: fecha límite
- Modificar contacto para que sea contenedor con nombre y teléfono por ejemplo
- Añadir elemento localiza al tablón.

Ejercicio para casa: la clase

17

- Usar ejemplo clase.xml: `alumnado>alumno`, con nombre, apellidos, fechaNac, mail y notas. Notas con `modulo> nombre y notaFinal`
- Quitar atributos si los tiene
- Restricciones
 - La clase puede estar vacía.
 - Nombre y apellidos obligatorios. Fecha opcional.
 - Mail uno o más.
 - Notas: número de módulos no definido.

DTD: atributos

18

Pueden aparecer en la definición varios atributos asociados al mismo elemento

- Sintaxis:

```
<!ATTLIST nombreElemento nombreAtributo  
           tipoAtributo valorpordefecto >
```

- Ejemplo

- DTD contiene...

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

- XML contiene ...

```
<pago tipo="contado"> lavadora</pago>
```

```
<pago tipo="tarjeta"> ordenador</pago>
```

Valor por defecto: posibilidades

19

```
<!ATTLIST nombreElemento nombreAtributo  
          tipoAtributo valorpordefecto >
```

- Un valor
 - el valor por defecto si no se especifica ninguno en la instancia de datos
- **#REQUIRED** => obligatorio
- **#IMPLIED** => opcional
- **#FIXED** “**valor**” => valor constante
- **(valor1 | valor2 ...)** => enumerado, uno de los valores del conjunto

Atributos (I)

20

- **#REQUIRED** atributo obligatorio.

DTD:

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

Valid XML: <alumno numero="12" />

Invalid XML: <alumno />

- **#IMPLIED** opcional, ni obligatorio ni valor por defecto.

DTD:

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

Valid XML: <contacto fax="956345678" />

Valid XML: <contacto />

Atributos (II)

21

- **#FIXED valor** Fijo, si se incluye otro valor, error.
`<!ATTLIST envio empresa CDATA #FIXED "MS">`
Valid XML: `< envio empresa ="MS" />`
Invalid XML: `< envio empresa ="Unifert" />`
- **Enumerado:** un valor entre un conjunto
`<!ATTLIST pago tipo (CC | paypal) #REQUIRED>`
Valid XML `<pago tipo="paypal" />`
Invalid XML: `<pago tipo="tarjeta" />`
 - **Detrás** del enumerado se puede indicar #REQUIRED, #IMPLIED o un valor por defecto
`<!ATTLIST p t (CC|tar) #IMPLIED>`
`<!ATTLIST p t (CC|tar) "CC">`

Valor por defecto del atributo

22

- En el caso de que no se especifique un valor al atributo en la instancia de datos, la aplicación debería usar el valor por defecto.
- Pero, por ejemplo, presentación en navegadores, no lo usan..

DTD:

```
<!ELEMENT cuadrado EMPTY>
```

```
<!ATTLIST cuadrado lado CDATA "0">
```

Valid XML: <cuadrado lado ="100" />

Valid XML: <cuadrado/>

Ejercicio: + sobre tablon.xml

23

- Crear DTD para que el atributo :
 - vendido del elemento anuncio sea obligatorio
 - moneda_pago de precio sea opcional
 - lang (idioma) del anuncio sea fijo, de valor spa
 - visible del elemento contacto tenga los valores si o no.
- Resolver paso a paso: probarlos de uno en uno
 - Comprobar la validez de ficheros xml con/sin atributos.
 - Comprobar la validez o no de ficheros que cumplan o no dichas restricciones
- Validar desde la línea de comandos
- Presentar los resultados en Firefox/IE .. ¿diferencias?
- Modificar DTDs de ejercicios anteriores para incluir los atributos en la DTD

Tipos de atributos (I)

24

- **CDATA**
 - puede contener cualquier carácter si éste se atiene a las reglas de formación.
- **NMTOKEN**
 - sólo puede contener letras, dígitos y
 - punto [.],
 - guión [-],
 - subrayado [_] y
 - dos puntos [:].
- **NMTOKENS**
 - los mismos caracteres que NMTOKEN más espacios en blanco (uno o más espacios, retornos de carro, tabuladores).

Tipos de atributos (I): Ejemplo

25

Ni el espacio ni # están permitidos si el atributo es **NMTOKEN**

```
<!ATTLIST attributes aaa CDATA #IMPLIED  
                      bbb NMTOKEN #REQUIRED  
                      ccc NMTOKENS #REQUIRED>
```

```
<attributes bbb="AB C" aaa="#d ."  
ccc="C D"/>
```

Correcto,
NMTOKEN**S**

Incorrecto,
espacio

correcto, # Y
ESPACIO

Tipos de atributos (II)

26

- **ID:** más restricciones
 - debe comenzar con una letra
 - sólo caracteres válidos en NMTOKEN
 - un elemento sólo puede tener un atributo de tipo ID.
 - El valor del atributo ID de un elemento debe tener un valor que no se repita en la instancia de datos (incluso entre otros atributos del tipo ID de otros elementos).

Ejercicio con tablon.dtd

27

- Modificar para que el atributo `moneda_de_pago` sea del tipo `NMTOKEN`.
 - Comprobar que con espacios, o comenzado con `#`, o `/`, da error
- Modificar para añadir al elemento `anuncio` un atributo obligatorio de nombre `codigo` y de tipo `ID`
 - Comprobar que sigue sintaxis `NMTOKEN`
 - Comprobar que no se puede tener dos anuncios con el mismo código

Tipos de atributos (III)

28

- **IDREF**

- El valor de un atributo IDREF tiene que corresponder con el valor de algún atributo ID del documento.

- **IDREFS**

- El valor del atributo IDREFS puede contener varias referencias a elementos con atributos ID separados por espacios en blanco.

- Y algunos otros que ni nombramos

Recuerda: atributos:

29

- + de un atributo

```
<!ATTLIST anuncio vendido CDATA #REQUIRED  
                codigo ID #REQUIRED >
```

- Enumerados: poner o valor por defecto, o IMPLIED, o REQUIRED

```
<!ATTLIST contact visible ( si | no ) #REQUIRED>
```

```
<!ATTLIST contact visible ( si | no ) #IMPLIED>
```

```
<!ATTLIST contact visible ( si | no ) "no">
```

Entidades

30

- Variables para usar abreviaturas o atajos
 - en DTD:
`<!ENTITY writer "Donald Duck.">`
 - en XML:
`<author>&writer;</author>`
- Se dice externa si la referencia es una URL/URI
 - DTD Example:
`<!ENTITY writer SYSTEM
"http://www.w3schools.com/entities.dtd">`
 - XML example:
`<author>&writer;</author>`
- Ventaja: se puede compartir en muchas docs xml, pero sólo hay que actualizar en una localización.
- Comprobar..
 - XMLcopyEditor la valida correctamente, pero los navegadores no la visualizan o incluso dan error.

Ejercicio mensajes y DTD

31

- Generar DTD para sistema de mensajes o avisos. Cada mensaje:
 - Remitente
 - Uno o más destinatarios
 - Texto de la nota
 - Atributos:
 - mensaje atributo urgente o normal. Por defecto normal. Obligatorio.
 - Fecha nota. Del tipo NMTOKEN. Primero opcional, luego obligatorio.
 - Atributo del asunto: idioma, por defecto es. Valores posible: es, en, fr
 - Atributo id de la nota, del tipo ID.
- Crear un documento xml, sms.xml con varios mensajes
- Probar si está bien formado:
- Añadir la referencia a la DTD y ver si es válido.
- Modificar sms.xml para que no cumpla la DTD y comprobar.

Ejercicio clase.xml (II)

32

- Modificar DTD para validar el ejemplo de clase.xml
- Añadir al fichero clase.xml
 - Atributo repetidor al alumno. Obligatorio, con valores si y no.
 - Atributo opcional idioma en alumno.
 - Atributo visible al mail del alumno. Tipo NMTOKEN
 - Atributo código a la asignatura. Tipo ID. Obligatorio.
- Añadir antes de los alumnos los profesores. Con datos nombre y apellidos.
 - Añadir a los profesores un atributo tutor para indicar si es tutor si o no.

Resumen Validando xml

33

- Usar editor XML que valide:
 - XML Copy Editor
 - VS code con complemento
- Línea de comandos: **xmlstarlet**
 - Bien formado

```
$ xmlstarlet val -e e1.xml
```
 - Válido

```
$ xmlstarlet val -e -d e1.dtd e1.xml
```
- Vía web, poco práctico, p.e www.xmlvalidation.com/
- Navegadores: No, quizás con algún plugin

Resumen

34

- Necesidad comprobar la estructura, atributos, etc.
- Varias formas de hacerlo, una de ellas con DTD
- Debemos saber y comprender cómo
 - Definir elementos de los distintos tipos
 - Ocurrencias de los elementos
 - Atributos
 - REQUIRED, IMPLIED, etc.
 - Tipos de los atributos: CDATA, NMTOKEN, ID..

Elementos con mismo identificador

35

- En DTDs, si dos elementos tienen el mismo nombre..
 - => Sólo pueden ser de un único tipo, no se puede tener distintos tipos para ellos:
<alumno>
 <nombre> Ana Garrido</nombre>
 <tutor>
 <nombre> Alfredo Rivera </nombre>
- Si tienen distintos tipos deben tener distintos nombres
 - o si tienen mismo nombre deben tener mismo tipo
- No se puede definir dos veces el mismo elemento (re-declaración)

Ejercicio tipo examen

36

Ejercicio 1: El ejercicio consiste en realizar tareas relacionadas con ficheros XML que almacenan la información de un proyecto.

- 1) **examen.dtd**: fichero utilizado para validar el fichero `examen.xml` (que deberá ser un fichero válido), y que además deberá cumplir:
 - La colección puede tener uno o más de un propietario
 - El precio de venta es opcional.
 - El número de películas puede ser cero, aunque el elemento películas debe existir siempre.
 - El título original es opcional, y puede tener un atributo de nombre lang con los valores en, es o fr.
 - El cartel es un elemento sin contenido que tiene un atributo fuente obligatorio.
 - Cada película tiene un atributo código que debe ser un identificador único.
 - La duración se mide en minutos
 - El director es uno y sólo uno.
 - El reparto está compuesto de al menos un actor.
 - La sinopsis tiene al menos un párrafo.

Nota: No es suficiente con que los ficheros `.xml` proporcionados sean válidos. El alumno debería comprobar que ficheros que no cumplan las restricciones enunciadas no serían válidos.