cpi fp
Los Enlaces

Lenguajes de Marcas y Sistemas de Información

UD3.2 Validación de documentos. DTD

Contenido

1. La importancia de la validación
2. Documento Válido
3. Bien formado VS Válido
4. Validación de documentos con DTD
   - 4.1. Elementos
   - 4.2. Atributos
   - 4.3. Entidades
   - 4.4. Notaciones
   - 4.5. Ejemplo completo
   - 4.6. Conclusiones

1. La importancia de la validación

- Hasta ahora, hemos aprendido cómo crear documentos XML bien formados, es decir, documentos que respetan las normales establecidas por la W3C en la especificación del estándar.

- Sin embargo, **un documento bien formado no es necesariamente válido**. Haciendo una analogía podríamos decir que **una frase ortográficamente correcta no puede no tener sentido.**

1. La importancia de la validación

- Por ejemplo, si tuviéramos una aplicación para manejar información de facturas en el que apareciera los siguientes datos, **resultaría informativamente inútil**.

- Ya que estando el documento bien formado, presenta evidentes errores de sentido común, como que una factura debe pertenecer a un único cliente y que una fecha debe especificarse en algún tipo de formato del tipo día/mes/año.

```
<factura>
    <importe>5000</importe>
    <cliente>Vicente Sánchez</cliente>
    <cliente>María del Carmen Pérez</cliente>
    <cliente>Juan Estévez</cliente>
    <fecha>mediados del mes pasado</fecha>
</factura>
```

1. La importancia de la validación

- Por ello, hay que ir más allá de lo bien formado y validar la estructura usada, respecto a unas reglas, con el objeto de **tener asegurada tanto su integridad como su formato**.

- Para ello **se recurre a ficheros de definición de esquemas que especifican la distribución de los datos y los contenidos que están permitidos en un documento**.

- **En XML si un documento está de acuerdo con un esquema se dice que es válido respecto a él, y en caso contrario se declara como no válido**.

2. Documento válido

- Son aquellos documentos que **además de estar bien formados, cumplen las especificaciones** de la DTD (Document Type Definition), del Schema, o del elemento que lo valide, y siguen las pautas marcadas por sus modelos de contenido. No hay esquemas de documentos idénticos, por eso es difícil concretar las reglas que deben cumplir.

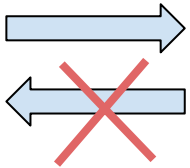
2. Documento válido

- Para que un documento XML además de bien formado también sea válido, durante el proceso de validación **se comprueba**:
  - **Qué elementos o atributos se permiten** en un documento del tipo definido en el esquema.
  - **La estructura de los elementos y atributos** (elementos anidados, atributos obligatorios u opcionales, etc.)
  - **El orden de los elementos**.
  - **Los valores de los datos de atributos y elementos** (según enumeraciones, rangos de valores delimitados, tipo de dato correcto (ej. formato correcto de una fecha, utilizar un entero para expresar un número), etc.
  - **La unicidad de valores dentro de un documento** (ej. Referencias de productos que no pueden repetirse).

3. Bien formado VS Válido

- Hay que distinguir entre un documento bien formado y un documento válido.
  - Un documento bien formado es simplemente un documento que respeta las normas del estándar XML

  - Sin embargo, un documento válido debe estar bien formado y, además, debe respetar las restricciones definidas por el esquema que se pretende respetar.

Documento XML válido  Documento XML Bien Formado

4. Validación de documentos con DTD

- **DTD (Document Type Definition)** es una descripción de estructura de un documento XML.

- Especifica que elementos deben aparecer, en que orden, cuales son opcionales, cuales son obligatorios, que atributos tienen los elementos, etc.

- DTD es un mecanismo de validación de documentos que ya existía antes de la publicación de XML. Se usaba para validar documentos SGML. Cuando apareció XML se integró en su especificación como modelo de validación gramatical.

4. Validación de documentos con DTD

- Al igual que sucedía en HTML con la hojas de estilos (CSS), **podemos declarar un DTD integrado en el propio documento (Interno) o en un archivo independiente con extensión DTD (Externo)**

- La opción más habitual es la de declarar un documento externo ya que nos permitirá reutilizar un mismo DTD para diferentes documentos, facilitando posibles modificaciones de una manera centralizada.

- Siempre que se quiera **declarar un DTD se hará al comienzo del documento XML**. Las reglas que lo constituyen podrán aparecer a continuación de la declaración (En el caso de un DTD interno) o en un archivo independiente.

4. Validación de documentos con DTD

Declaración de un DTD

- **<!DOCTYPE> es la introducción donde se indica qué DTD validará nuestro XML.** Aparece al comienzo del documento XML.
- Existen diferentes tipos de declaraciones dependiendo del tipo de DTD que queramos usar.

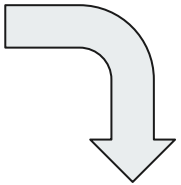
Sintaxis	Tipo de DTD
`<!DOCTYPE elementoRaíz [reglas]>`	Interno
`<!DOCTYPE elementoRaíz SYSTEM URL>`	Externo y privado
`<!DOCTYPE elementoRaíz PUBLIC FPI URL>`	Externo y público

FPI es un identificador público que puede ser utilizado por el procesador XML para intentar generar una URI alternativo

4. Validación de documentos con DTD

Declaración de un DTD

```
<?xml version="1.0"?>
<!DOCTYPE note SYSTEM "note.dtd">
<note>
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
```



note.dtd

```
<!ELEMENT note (to,from,heading,body)>
<!ELEMENT to (#PCDATA)>
<!ELEMENT from (#PCDATA)>
<!ELEMENT heading (#PCDATA)>
<!ELEMENT body (#PCDATA)>
```

4. Validación de documentos con DTD

Ejemplo de un DTD interno

```
<?xml version="1.0"?>
<!DOCTYPE note [
<!ELEMENT note (to,from,heading,body)>
<!ELEMENT to (#PCDATA)>
<!ELEMENT from (#PCDATA)>
<!ELEMENT heading (#PCDATA)>
<!ELEMENT body (#PCDATA)>
]>
<note>
<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Don't forget me this weekend</body>
</note>
```

4. Validación de documentos con DTD

Componentes de un DTD

- Existen cuatro tipos posibles de componentes que podemos declarar en un DTD:
  - **Elementos**
  - **Atributos**
  - **Entidades**
  - **Notaciones**

4. Validación de documentos con DTD

cpi fp Los Enlaces

Componentes de un DTD: Elementos

- Es una declaración de tipo de elemento. Indica la existencia de un elemento dentro del documento XML.
- Sintaxis general: **<!ELEMENT nombreElemento modeloContenido>**
  - El nombre del elemento debe ser el mismo que usemos en el documento XML.
  - El modelo de contenido puede indicar:
    - Una regla
      - ANY→ Indica que nuestro elemento puede contener cualquier cosa. Es un comodín que *no debe aparecer en un DTD definitivo*.
      - EMPTY → Indica que el elemento no contiene nada.

```
<!ELEMENT planetas ANY>
```

```
<!ELEMENT planetas EMPTY>
```

4. Validación de documentos con DTD

cpi fp Los Enlaces

Componentes de un DTD: Elementos

- Datos (Caracteres) → Sean textuales, numéricos o cualquier otro formato que no contenga marcas. Se describe como **#PCDATA** y debe aparecer entre paréntesis.

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

- Elementos hijos → Su descripción debe aparecer entre paréntesis y se basa en las siguientes reglas:
  - Cardinalidad de los elementos

4. Validación de documentos con DTD

Componentes de un DTD: Elementos

- Elementos hijos → Su descripción debe aparecer entre paréntesis y se basa en las siguientes reglas:
  - **Cardinalidad** de los elementos: Indica las veces que puede aparecer un elemento. (por defecto la cardinalidad será 1)
    - ? → Puede aparecer 0 o 1 vez.
    - * → Puede aparecer 0 o N veces.
    - + → El elemento puede aparecer 1 o N veces
  - Secuencias de elementos: Indica el orden en que los elementos deben aparecer o bien si un elemento puede aparecer como alternativa a otro:
    - A,B → El elemento B aparecerá a continuación del elemento A.
    - A|B → Aparecerá el elemento A o el elemento B pero no ambos.

4. Validación de documentos con DTD

cpifp Los Enlaces

Componentes de un DTD: Elementos

- Se pueden combinar los símbolos de cardinalidad con los de secuencia de elementos.
- En el siguiente ejemplo se pretende crear un correo electrónico descrito por un elemento raíz <email> que contiene la secuencia de elementos <para>, <cc> (opcional), <cco> (opcional>), <asunto> y <cuerpo>.

```
<!ELEMENT email (para, cc?, cco?, asunto, cuerpo)>
<!ELEMENT para (#PCDATA)>
<!ELEMENT cc (#PCDATA)>
<!ELEMENT cco (#PCDATA)>
<!ELEMENT asunto (#PCDATA)>
<!ELEMENT cuerpo (#PCDATA)>
```

4. Validación de documentos con DTD

cpi fp Los Enlaces

Componentes de un DTD: Elementos

- En el siguiente ejemplo se describe un contrato definido por una lista de cláusulas. Cada una de las cláusulas está compuesta por varios epígrafes y sus desarrollos asociados y concluye por un único epílogo.

```
<!ELEMENT clausulas (clausula)+>
<!ELEMENT clausula ((epigrafe, desarrollo)+, epilogo)>
<!ELEMENT epigrafe (#PCDATA)>
<!ELEMENT desarrollo (#PCDATA)>
<!ELEMENT epilogo (#PCDATA)>
```

4. Validación de documentos con DTD

Componentes de un DTD: Atributos

- DTD permite definir que atributos se asocian a un elemento concreto mediante la etiqueta <!ATTLIST>. En general se utiliza `<!ATTLIST>` para declarar todos los atributos de un elemento, aunque podríamos utilizar un `<!ATTLIST>` por cada atributo.

```
<!ATTLIST nombreElemento
                nombreAtributo1 tipoAtributo1 caracter
                nombreAtributo2 tipoAtributo2 caracter
>
```

4. Validación de documentos con DTD

Componentes de un DTD: Atributos

- **NombreAtributo** → tiene que ser un nombre XML válido.
- **Carácter**:
    - Un valor textual entre comillas que representa el valor por defecto.
    - #IMPLIED → **El atributo es opcional** y no se le asigna ningún valor por defecto.
    - #REQUIRED → **El atributo es obligatorio**, no se le asigna valor por defecto.
    - #FIXED → **El atributo es obligatorio y se le asigna un valor por defecto** que será el único valor que el atributo pueda adoptar.

4. Validación de documentos con DTD

cpifp Los Enlaces

Componentes de un DTD: Atributos

- **TipoAtributo →**
  - CDATA → **caracteres textuales**
  - ENTITY → **nombre de una entidad** (Que debe declararse en el DTD)
  - ENTITIES → una **lista de nombres de entidades separadas por espacios**
  - Enumerado → Una **lista de valores**, entre los cuales, el atributo debe tomar uno. En este caso no se pondría nada en el DTD
  - NOTATION
  - ID → **Un identificador único**
  - IDREF → **representa el valor de un atributo ID de otro elemento**

4. Validación de documentos con DTD

cpi fp Los Enlaces

Componentes de un DTD: Atributos

- Atributo opcional de tipo CDATA sin ningún valor por defecto.

```
<!ATTLIST persona signoparticular CDATA #IMPLIED>
```

- Atributo opcional de tipo CDATA con valor por defecto

```
<!ATTLIST persona signoparticular CDATA "nada">
```

- Atributo obligatorio

```
<!ATTLIST persona sexo CDATA #REQUIRED>
```

4. Validación de documentos con DTD

cpifp Los Enlaces

Componentes de un DTD: Atributos

- Atributo con valor prefijado #FIXED

```
<!-- El valor del atributo formato es siempre "mp3" -->
<!ATTLIST archivoaudio formato CDATA #FIXED "mp3">
```

- Atributo de tipo textual

```
<!ATTLIST apartamento tipo CDATA #IMPLIED>
```

- Atributo de tipo enumerado.

```
<!ATTLIST apartamento tipo (ubicación | venta) #IMPLIED
```

4. Validación de documentos con DTD

Componentes de un DTD: Atributos

- Atributo de tipo ID

```
<!ATTLIST receta id ID #REQUIRED>
```

- Atributo de tipo IDREF.

```
<!-- El atributo ref señala al identificador de otro
elemento -->
<!ATTLIST ingrediente ref IDREF #IMPLIED>
```

4. Validación de documentos con DTD

cpi fp Los Enlaces

Componentes de un DTD: Atributos

- El siguiente ejemplo muestra el DTD de un XML que describe una receta de cocina.

```
<!ELEMENT librodecocina (receta | ingrediente)*>
<!ELEMENT receta #PCDATA>
<!ATTLIST receta id ID #REQUIRED>
<!ELEMENT ingrediente #PCDATA>
<!ATTLIST ingrediente ref IDREFS #IMPLIED>
```

4. Validación de documentos con DTD

cpifp Los Enlaces

Componentes de un DTD: Atributos

- XML a partir del esquema DTD de la anterior diapositiva.

```
<librodecocina>
  <receta id="rec_1"> Mousse de chocolate </receta>
  <receta id="rec_2"> Fondue savoyana </receta>
  <receta id="rec_3"> Fondue de chocolate </receta>
  <ingrediente ref="rec_1 rec_3"> chocolate </ingrediente>

  <ingrediente ref="rec_2"> Queso Comté </ingrediente>
</librodecocina>
```

4. Validación de documentos con DTD

Componentes de un DTD: Atributos

```
<!-- El atributo a señala a una simple entidad -->
<!ATTLIST A a ENTITY #IMPLIED>
<!-- El atributo b señala a varias entidades -->
<!ATTLIST A b ENTITIES #IMPLIED>
```

4. Validación de documentos con DTD

Componentes de un DTD: Entidades

- Las entidades **permiten definir constantes para los documentos XML**.
- Existen dos tipos de entidades:
  - Entidades internas→ Permite declarar una variable cuyo alcance se limita al documento XML, se limitan por “&” y “;”, por ejemplo &entidad;
  - Entidades externas→ Permiten vincular el documento XML a otros documentos (de cualquier tipo).
  - Insertamos &nombredelaentidad; en el documento XML para utilizar el valor de la entidad llamado nombredelaentidad.

4. Validación de documentos con DTD

cpifp Los Enlaces

Componentes de un DTD: Entidades

- XML define una serie de entidades por defecto:

Entidad	Valor
<	<
>	>
&	carácter de espacio
'	carácter ' (apóstrofe)
"	carácter " (comillas)

4. Validación de documentos con DTD

cpifp Los Enlaces

Componentes de un DTD: Entidades

- Las entidades internas solamente se pueden utilizar en el documento en el que se han declarado.

```
<!ENTITY nombredelaentidad "valor de la entidad">
```

4. Validación de documentos con DTD

Componentes de un DTD: Entidades

- Ejemplo de entidad interna

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE tarjetadevisita [<!ENTITY avstrb "Avenida
de Estrasburgo">]>
<tarjetadevisita>
  <apellido>Guevara</apellido>
  <nombre>Ernesto</nombre>
<puesto>Doctor</puesto>
  <dirección>
    <número>172</número>
    <avenida>&avstrb;</avenida>
    <códigopostal>75455</códigopostal>
    <ciudad>Buenos Aires</ciudad>
  </dirección>
</tarjetadevisita>
```

4. Validación de documentos con DTD

cpifp Los Enlaces

Componentes de un DTD: Entidades

- Las entidades externas permiten vincular el documento XML actual a otro documento al que se hace referencia a través de su URL.
- Este documento puede ser de tipo XML o de cualquier otro tipo.

```
<!ENTITY otrodoc SYSTEM "http://localhost/docsxml/otrodoc.xml">
```

4. Validación de documentos con DTD

Componentes de un DTD: Notaciones

- Es un elemento avanzado del diseño de DTD.
- Se trata de la declaración de un atributo de tipo NOTATION.
- Permiten determinar cuál es la aplicación que ha de procesar un fichero binario asociado a un fichero XML
- Una notación se usa para especificar un formato de datos que no sea XML, como ficheros binarios tipo imagen. Con frecuencia un tipo MIME como
  - image/gif
  - 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.

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

4. Validación de documentos con DTD

Componentes de un DTD: Notaciones

```
<?xml version='1.0' encoding='UTF-8'?>
<!DOCTYPE mares [
<!NOTATION GIF SYSTEM "image/gif" >
<!NOTATION PNG SYSTEM "image/png" >
<!NOTATION JPG SYSTEM "image/jpg" >
<!ELEMENT mares (mar)+>
<!ELEMENT mar (nombre)>
<!ELEMENT nombre (#PCDATA)>
<!ATTLIST mar imagen ENTITY #IMPLIED>
<!ATTLIST mar formato_imagen NOTATION (GIF | JPG | PNG) #IMPLIED>
]>
<mares>
  <mar imagen="mediterraneo" formato_imagen="JPG">
    <nombre>Mar Mediterráneo</nombre>
  </mar>
</mares>
```

Aclaración

4. Validación de documentos con DTD

Componentes de un DTD: Declaraciones ocultas

Podemos dejar no visibles una serie de declaraciones con el comando **Ignore**. Es similar a comentar esa parte del código

<!IGNORE [declaraciones ocultas]>

```
<![IGNORE [<!ELEMENT clave (#PCDATA)>] ] >
```

Aclaración

4. Validación de documentos con DTD

Ejemplo completo.

Se quiere construir un DTD que valide el siguiente esquema XML.

```
<?xml version='1.0' encoding='UTF-8'?>
<!DOCTYPE persona SYSTEM "persona.dtd">
<persona cod="ml12345678P" estadoCivil="Casado">
  <nombre>Maria</nombre>
  <apellido>López</apellido>
  <edad>60</edad>
  <enActivo/>
</persona>
```

- El atributo cod es un identificador obligatorio.
- El estado civil puede ser Soltero, Casado o Divorciado. Por defecto es Soltero
- El elemento enActivo es opcional

4. Validación de documentos con DTD

cpi fp Los Enlaces

Ejemplo completo.

El siguiente DTD cumple con las especificaciones dadas.

```
<!ELEMENT persona (nombre, apellido, edad, enActivo?)>
<!ATTLIST persona
          cod ID  #REQUIRED
          estadoCivil (Soltero | Casado | Divorciado) "Soltero">
<!ELEMENT nombre (#PCDATA)>
<!ELEMENT apellido (#PCDATA)>
<!ELEMENT edad (#PCDATA)>
<!ELEMENT enActivo (#PCDATA)>
```

4. Validación de documentos con DTD

cpifp Los Enlaces

Conclusiones

- Hemos estudiado una potente herramienta de validación de documentos XML, sin embargo DTD cuenta con algunas limitaciones importantes:
  - Un DTD no deja de ser un documento XML, por lo que no podemos garantizar su validez.
  - No podemos fijar restricciones sobre los valores de los elementos y atributos como su tipo de dato, longitud, etc.
  - No soporta espacios de nombres.
  - Solo se pueden crear enumerados en cuanto a valores de atributos pero no de elementos.
  - Solo se pueden definir valores por defecto para atributos pero no para elementos.
  - Existe un control limitado sobre las cardinalidades de los elementos.

4. Validación de documentos con DTD

Validador

- Para documentos con DTD interna:
  - **http://xmlvalidator.new-studio.org/**
  - **https://www.xmlvalidation.com/**

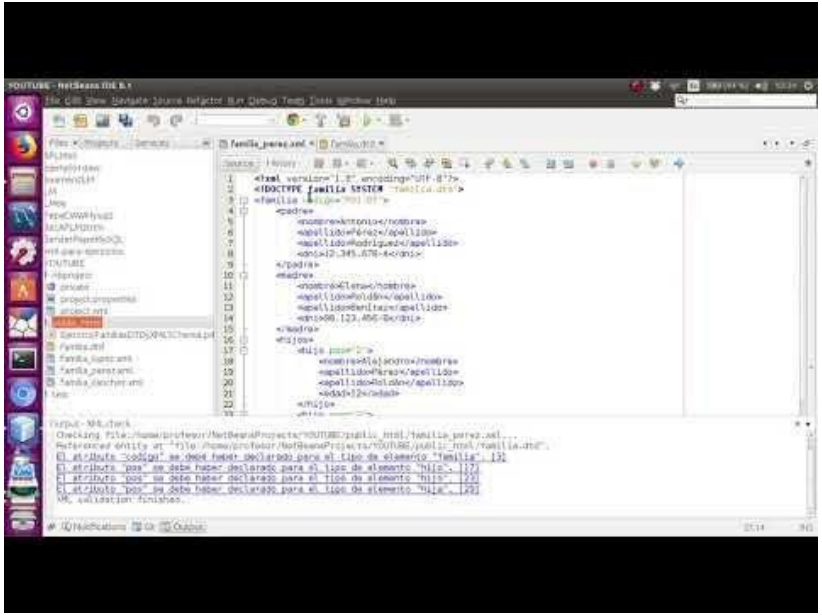
- Para documentos con DTD externa:
  - **https://www.truugo.com/xml_validator/**
  - **XmlCopy Editor**

4. Validación de documentos con DTD

Inconvenientes de los DTD

- Su sintaxis no es XML
- No soportan espacios de nombres
- No definen tipos para los datos. Solo hay un tipo de elementos terminales, que son los datos textuales.
- No permite las secuencias no ordenadas
- No es posible formar claves a partir de varios atributos o elementos
- Una vez que se define un DTD no es posible añadir nuevos vocabularios

Ejemplo completo



Ejercicios con soluciones

Dudas y preguntas