Cómo validar documentos XML utilizando DTD.

Qué es DTD

- **DTD** (***D**ocument **T**ype **D**efinition*, Definición de Tipo de Documento) sirve para definir la estructura de un documento SGML o XML, permitiendo su validación.
  - **SGML** (***S**tandard **G**eneralized **M**arkup **L**anguage*, Lenguaje de Marcado Generalizado Estándar).
  - **XML** (*e**X**tensible **M**arkup **L**anguage*, Lenguaje de Marcado eXtensible) es un lenguaje desarrollado por **W3C** (*World Wide Web Consortium*) que está basado en SGML.
- Un documento XML es válido (*valid*) cuando, además de estar bien formado, no incumple ninguna de las normas establecidas en su estructura.

Declaración de tipo de documento

- Una DTD se puede escribir tanto interna como externamente a un archivo XML.

- En ambos casos hay que escribir una definición **`DOCTYPE`** (*Document Type Declaration*, Declaración de Tipo de Documento) para asociar el documento XML a la DTD.

- Asimismo, un archivo XML se puede asociar simultáneamente a una DTD interna y externa.

Documento XML asociado a una DTD interna

- Sintaxis:

```
<!DOCTYPE elemento-raíz [ declaraciones ]>
```

- **EJEMPLO**

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE marcadores [
   <!ELEMENT marcadores (pagina)*>
   <!ELEMENT pagina (nombre, descripcion, url)>
   <!ELEMENT nombre (#PCDATA)>
   <!ELEMENT descripcion (#PCDATA)>
   <!ELEMENT url (#PCDATA)>
]>
```

Documento XML asociado a una DTD interna

- En la DTD se ha indicado que **`marcadores`** es el elemento raíz del documento XML, el cual puede contener cero o más páginas. Para indicar esto último, se ha escrito: **`(pagina)*`**

- Escribiendo **`pagina (nombre, descripcion, url)`** se especifica que, cada elemento "pagina" tiene que contener tres elementos (hijos): "nombre", "descripcion" y "url".

- Con **`#PCDATA`** (*Parsed Character Data*) escrito entre paréntesis " **`()`** " se indica que los elementos "nombre", "descripcion" y "url" pueden contener texto (cadenas de caracteres) analizable por un procesador XML.

Documento XML asociado a una DTD externa

Existen dos tipos de DTD externa: privada y pública.

- Sintaxis DTD externa privada:

```
<!DOCTYPE elemento-raíz SYSTEM "URI">
```

- Sintaxis DTD externa pública:

```
<!DOCTYPE elemento-raíz PUBLIC "identificador-
público" "URI">
```

DTD externa privada - SYSTEM

- **EJEMPLO** Si en un archivo llamado ***"marcadores.dtd"*** se escribiese la siguiente DTD:

```
<!ELEMENT marcadores (pagina)*>
<!ELEMENT pagina (nombre, descripcion, url)>
<!ELEMENT nombre (#PCDATA)>
<!ELEMENT descripcion (#PCDATA)>
<!ELEMENT url (#PCDATA)>
```

- El siguiente documento XML llamado ***"marcadores-con-dtd-externa.xml"***, sería válido:

DTD externa privada - SYSTEM

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE marcadores SYSTEM "marcadores.dtd">
<marcadores>
   <pagina>
      <nombre>Abrirllave</nombre>
      <descripcion>Tutoriales de informática.</descripcion>
      <url>http://www.abrirllave.com/</url>
   </pagina>
   <pagina>
      <nombre>Wikipedia</nombre>
      <descripcion>La enciclopedia libre.</descripcion>
      <url>http://www.wikipedia.org/</url>
   </pagina>
   <pagina>
      <nombre>W3C</nombre>
      <descripcion>World Wide Web Consortium.</descripcion>
      <url>http://www.w3.org/</url>
   </pagina>
</marcadores>
```

DTD externa pública - PUBLIC

- **EJEMPLO** El siguiente documento XML está asociado a una DTD externa pública:

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
   <head>
      <title>Título</title>
   </head>
   <body>
      <p>Párrafo</p>
   </body>
</html>
```

Cuándo utilizar una DTD interna o una DTD externa

- Para validar más de un documento XML con la misma DTD, escribir esta en un archivo externo proporciona la ventaja de no tener que repetir la DTD internamente en cada documento XML.

- En el caso de que la DTD solo se utilice para validar un único documento XML, la DTD es habitual escribirla internamente.

Uso combinado de DTD interna y externa en un documento XML

- Para asociar un documento XML a una DTD interna y externa simultáneamente, se pueden utilizar las siguientes sintaxis:

```
<!DOCTYPE elemento-raíz SYSTEM "URI" [ declaraciones ]>
```

```
<!DOCTYPE elemento-raíz PUBLIC "identificador-público"
"URI" [ declaraciones ]>
```

- **EJEMPLO** Si en un documento XML llamado ***"marcadores-con-dtd-interna-y-externa.xml"*** se quiere almacenar una lista de marcadores de páginas web, guardando de cada uno de ellos su nombre, una descripción y su URL. En dicho documento se podría escribir:

Uso combinado de DTD interna y externa en un documento XML

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE marcadores SYSTEM "marcadores.dtd" [
   <!ELEMENT marcadores (pagina)*>
   <!ELEMENT pagina (nombre, descripcion, url)>
]>
<marcadores>
   <pagina>
      <nombre>Abrirllave</nombre>
      <descripcion>Tutoriales de informática.</descripcion>
      <url>http://www.abrirllave.com/</url>
   </pagina>
   <pagina>
      <nombre>Wikipedia</nombre>
      <descripcion>La enciclopedia libre.</descripcion>
      <url>http://www.wikipedia.org/</url>
   </pagina>
   <pagina>
      <nombre>W3C</nombre>
      <descripcion>World Wide Web Consortium.</descripcion>
      <url>http://www.w3.org/</url>
   </pagina>
</marcadores>
```

Uso combinado de DTD interna y externa en un documento XML

- El contenido del archivo *"marcadores.dtd"* podría ser:

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

Estructura de un documento XML

- En una DTD se pueden declarar:
  - Elementos
  - Atributos
  - Entidades
  - Notaciones

- Por tanto, **un documento XML será válido si** –además de no tener errores de sintaxis– **cumple lo indicado en las declaraciones de elementos, atributos, entidades y notaciones, de la DTD a la que esté asociado.**

Declaración de elementos en una DTD

- Sintaxis:

```
<!ELEMENT nombre-del-elemento tipo-de-contenido>
```

- En el **`tipo-de-contenido`** se especifica el contenido permitido en el elemento, pudiendo ser:
  - Texto, **`(#PCDATA)`**.
  - Otros elementos (hijos).
  - Estar vacío, **`EMPTY`**.
  - Mixto (texto y otros elementos), **`ANY`**.

El contenido de un elemento puede ser texto - (#PCDATA)

- **EJEMPLO** En el siguiente documento XML, el elemento "ciudad" puede contener cualquier texto (cadena de caracteres):

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE ciudad [
   <!ELEMENT ciudad (#PCDATA)>
]>
<ciudad>Roma</ciudad>
```

- Escribiendo **#PCDATA** (*Parsed Character Data*) entre paréntesis " **()** ", se ha indicado que el elemento "ciudad" puede contener una cadena de caracteres analizable.

Un elemento puede contener a otros elementos

- **EJEMPLO** En el siguiente ejemplo, el elemento “ciudad” contiene a los elementos “nombre” y “pais”:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE ciudad [
   <!ELEMENT ciudad (nombre, pais)>
   <!ELEMENT nombre (#PCDATA)>
   <!ELEMENT pais (#PCDATA)>
]>

<ciudad>
   <nombre>Roma</nombre>
   <pais>Italia</pais>
</ciudad>
```

Un elemento puede no contener contenido (estar vacío) - EMPTY

- **EJEMPLO** En la DTD interna del siguiente documento XML, se ha declarado el elemento “mayor_de_edad” como vacío, **EMPTY**:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE persona [
   <!ELEMENT persona (nombre, mayor_de_edad, ciudad)>
   <!ELEMENT nombre (#PCDATA)>
   <!ELEMENT mayor_de_edad EMPTY>
   <!ELEMENT ciudad (#PCDATA)>
]>

<persona>
   <nombre>Elsa</nombre>
   <mayor_de_edad/>
   <ciudad>Pamplona</ciudad>
</persona>
```

- Los elementos vacíos no pueden tener contenido, pero sí pueden tener atributos.

Un elemento puede definirse para contener contenido mixto - ANY

- **EJEMPLO** En la DTD interna del siguiente documento XML, se ha indicado que el elemento “persona” puede contener texto y otros elementos, es decir, contenido mixto, **ANY**:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE persona [
   <!ELEMENT persona ANY>
   <!ELEMENT nombre (#PCDATA)>
   <!ELEMENT ciudad (#PCDATA)>
]>

<persona>
   <nombre>Elsa</nombre> vive en <ciudad>Pamplona</ciudad>.
</persona>
```

Un elemento puede definirse para contener contenido mixto - ANY

- Obsérvese que, por ejemplo, también sería válido el siguiente documento XML:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE persona [
   <!ELEMENT persona ANY>
   <!ELEMENT nombre (#PCDATA)>
   <!ELEMENT ciudad (#PCDATA)>
]>

<persona>
   <nombre>Elsa</nombre> vive en Pamplona.
</persona>
```

Un elemento puede definirse para contener contenido mixto - ANY

- O el siguiente:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE persona [
   <!ELEMENT persona ANY>
   <!ELEMENT nombre (#PCDATA)>
   <!ELEMENT ciudad (#PCDATA)>
]>

<persona>
   <nombre>Elsa</nombre>
</persona>
```

Un elemento puede definirse para contener contenido mixto - ANY

- Incluso, si el elemento “persona” estuviese vacío, el documento también sería válido:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE persona [
   <!ELEMENT persona ANY>
   <!ELEMENT nombre (#PCDATA)>
   <!ELEMENT ciudad (#PCDATA)>
]>

<persona/>
```

Elementos vacíos - EMPTY

- Para declarar un elemento vacío en una DTD, hay que indicar que su contenido es **EMPTY**.

- Un ejemplo de ello podría ser el elemento “br” del HTML, el cual sirve para hacer un salto de línea y no tiene contenido:

```
<!ELEMENT br EMPTY>
```

- Dada la declaración anterior, en un documento XML el elemento “br” podría escribirse como:

```
<br/>
```

- O también:

```
<br></br>
```

Elementos vacíos - EMPTY

- Por ejemplo, el siguiente documento XML sería válido:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE etiquetas_html [
   <!ELEMENT etiquetas_html (br)>
   <!ELEMENT br EMPTY>
]>

<etiquetas_html>
   <br/>
</etiquetas_html>
```

Un elemento vacío puede tener atributos

- **EJEMPLO** Aunque un elemento se declare vacío, no pudiendo contener texto ni otros elementos, sí puede tener atributos:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE etiquetas_html [
   <!ELEMENT etiquetas_html (br)>
   <!ELEMENT br EMPTY>
   <!ATTLIST br descripcion CDATA #REQUIRED>
]>

<etiquetas_html>
   <br descripcion="Salto de línea"/>
</etiquetas_html>
```

- En este ejemplo, para el elemento “br” se ha declarado el atributo **descripcion** de tipo **CDATA** (*Character DATA*), es decir, su valor puede ser una cadena de caracteres. Además, se ha indicado que el atributo es obligatorio escribirlo, **#REQUIRED**.

Elementos con cualquier contenido - ANY

- Cuando en una DTD se quiere declarar un elemento que pueda contener cualquier contenido –bien sea texto, otros elementos o una mezcla de ambos– esto se puede hacer indicando que su contenido es de tipo **ANY**:

```
<!ELEMENT cualquier_contenido ANY>
```

- **EJEMPLO** En el siguiente documento XML, el elemento "cualquier_contenido" contiene tres elementos "texto":

Elementos con cualquier contenido - ANY

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE ejemplo [
   <!ELEMENT ejemplo (cualquier_contenido)>
   <!ELEMENT cualquier_contenido ANY>
   <!ELEMENT texto (#PCDATA)>
]>

<ejemplo>
   <cualquier_contenido>
      <texto>Texto1</texto>
      <texto>Texto2</texto>
      <texto>Texto3</texto>
   </cualquier_contenido>
</ejemplo>
```

Elementos con cualquier contenido - ANY

- Definiendo la misma DTD, también sería válido el siguiente documento XML donde el elemento “cualquier_contenido” solo contiene texto:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE ejemplo [
   <!ELEMENT ejemplo (cualquier_contenido)>
   <!ELEMENT cualquier_contenido ANY>
   <!ELEMENT texto (#PCDATA)>
]>

<ejemplo>
   <cualquier_contenido>Texto1. Texto2. Texto3</cualquier_contenido>
</ejemplo>
```

Elementos con cualquier contenido - ANY

- Asimismo, el elemento “cualquier_contenido” podría contener una mezcla de texto y uno o más elementos.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE ejemplo [
   <!ELEMENT ejemplo (cualquier_contenido)>
   <!ELEMENT cualquier_contenido ANY>
   <!ELEMENT texto (#PCDATA)>
]>

<ejemplo>
   <cualquier_contenido>Texto1<texto>Texto2</texto>Texto3</cualquier_contenido>
</ejemplo>
```

Elementos con cualquier contenido - ANY

- Por otra parte, si el elemento “cualquier_contenido” estuviese vacío, el documento XML seguiría siendo válido:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE ejemplo [
   <!ELEMENT ejemplo (cualquier_contenido)>
   <!ELEMENT cualquier_contenido ANY>
   <!ELEMENT texto (#PCDATA)>
]>

<ejemplo>
   <cualquier_contenido></cualquier_contenido>
</ejemplo>
```

Elementos con contenido de tipo texto - (#PCDATA)

- Para declarar en una DTD un elemento que pueda contener texto analizable, se tiene que indicar que su contenido es **(#PCDATA)**, (*Parsed Character Data*):

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

- **EJEMPLO** En el siguiente documento XML, el elemento "texto" contiene caracteres:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE ejemplo [
   <!ELEMENT ejemplo (texto)>
   <!ELEMENT texto (#PCDATA)>
]>

<ejemplo>
   <texto>Este elemento solo contiene caracteres.</texto>
</ejemplo>
```

Elementos con contenido de tipo texto - (#PCDATA)

- Ahora bien, el elemento “texto” podría estar vacío y el documento XML seguiría siendo válido:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE ejemplo [
   <!ELEMENT ejemplo (texto)>
   <!ELEMENT texto (#PCDATA)>
]>

<ejemplo>
   <texto/>
</ejemplo>
```

Secuencias de elementos

- En una DTD, un elemento (padre) puede ser declarado para contener a otro u otros elementos (hijos).

- En la sintaxis, los hijos –también llamados sucesores– tienen que escribirse entre paréntesis “ **( )** ” y separados por comas “ **,** ”.

Elemento con varios hijos

- **EJEMPLO** Para declarar un elemento (padre) que contenga tres elementos (hijos), se puede escribir:

```
<!ELEMENT padre (hijo1, hijo2, hijo3)>
```

- **EJEMPLO** En el siguiente documento XML, el elemento “persona” contiene a los elementos “nombre”, “fecha_de_nacimiento” y “ciudad”:

Elemento con varios hijos

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE persona [
   <!ELEMENT persona (nombre, fecha_de_nacimiento, ciudad)>
   <!ELEMENT nombre (#PCDATA)>
   <!ELEMENT fecha_de_nacimiento (#PCDATA)>
   <!ELEMENT ciudad (#PCDATA)>
]>

<persona>
   <nombre>Iker</nombre>
   <fecha_de_nacimiento>26-12-1997</fecha_de_nacimiento>
   <ciudad>Valencia</ciudad>
</persona>
```

Elemento con varios hijos

- A su vez, los hijos también pueden tener sus propios hijos. Así, el elemento “fecha_de_nacimiento” puede contener, por ejemplo, a los elementos “dia”, “mes” y “anio”:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE persona [
   <!ELEMENT persona (nombre, fecha_de_nacimiento, ciudad)>
   <!ELEMENT nombre (#PCDATA)>
   <!ELEMENT fecha_de_nacimiento (dia, mes, anio)>
   <!ELEMENT dia (#PCDATA)>
   <!ELEMENT mes (#PCDATA)>
   <!ELEMENT anio (#PCDATA)>
   <!ELEMENT ciudad (#PCDATA)>
]>

<persona>
   <nombre>Iker</nombre>
   <fecha_de_nacimiento>
      <dia>26</dia>
      <mes>12</mes>
      <anio>1997</anio>
   </fecha_de_nacimiento>
   <ciudad>Valencia</ciudad>
</persona>
```

Orden de los hijos de un elemento

- En un documento XML, los elementos (hijos) de un elemento (padre), deben escribirse en el mismo orden en el que han sido declarados en la DTD.
- **EJEMPLO** El siguiente documento XML no es válido:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE persona [
   <!ELEMENT persona (nombre, fecha_de_nacimiento, ciudad)>
   <!ELEMENT nombre (#PCDATA)>
   <!ELEMENT fecha_de_nacimiento (dia, mes, anio)>
   <!ELEMENT dia (#PCDATA)>
   <!ELEMENT mes (#PCDATA)>
   <!ELEMENT anio (#PCDATA)>
   <!ELEMENT ciudad (#PCDATA)>
]>
<persona>
   <nombre>Iker</nombre>
   <fecha_de_nacimiento>
      <anio>1997</anio>
      <mes>12</mes>
      <dia>26</dia>
   </fecha_de_nacimiento>
   <ciudad>Valencia</ciudad>
</persona>
```

Cardinalidad de los elementos

- En una DTD, para definir el número de veces que pueden aparecer los elementos de un documento XML, se pueden utilizar los *operadores de cardinalidad* mostrados en la siguiente tabla:

Operadores de cardinalidad en DTD		
Operador	***Cardinalidad***	***Significado***
? (*interrogación*)	0-1	El elemento es opcional, pudiendo aparecer una sola vez o ninguna.
***** (*asterisco*)	0-n	El elemento puede aparecer cero, una o más veces.
+ (*signo más*)	1-n	El elemento tiene que aparecer, obligatoriamente, una o más veces.

- Los elementos declarados en una DTD sobre los que no actúe ningún operador de cardinalidad, tendrán que aparecer obligatoriamente una única vez, en el o los documentos XML a los que se asocie.

Operador de cardinalidad “+” (signo más)

- **EJEMPLO** En el siguiente documento XML, el elemento “nombre” tiene que aparecer una o más veces. En este caso, aparece tres veces:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE personas [
   <!ELEMENT personas (nombre+)>
   <!ELEMENT nombre (#PCDATA)>
]>
<personas>
   <nombre>Ana</nombre>
   <nombre>Iker</nombre>
   <nombre>Elsa</nombre>
</personas>
```

- Si sobre nombre no actuase el operador (**+**) el documento no sería válido, ya que, el elemento “personas” solo tendría que contener un elemento “nombre”.

- En vez de **(nombre+)**, también se puede escribir **(nombre)+**.

Operador de cardinalidad “*” (asterisco)

- **EJEMPLO** En la DTD interna del siguiente documento XML, se ha indicado que el elemento “nombre” tiene que aparecer una única vez. Ahora bien, el elemento “ingrediente” tiene cardinalidad (0-n), por tanto, puede aparecer cero, una o más veces:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE receta_de_cocina [
   <!ELEMENT receta_de_cocina (nombre, ingrediente*)>
   <!ELEMENT nombre (#PCDATA)>
   <!ELEMENT ingrediente (#PCDATA)>
]>
<receta_de_cocina>
   <nombre>Tortilla de patatas</nombre>
   <ingrediente>Huevo</ingrediente>
   <ingrediente>Patata</ingrediente>
   <ingrediente>Aceite</ingrediente>
   <ingrediente>Sal</ingrediente>
</receta_de_cocina>
```

Operador de cardinalidad “?” (interrogación)

- **EJEMPLO** En la DTD del siguiente documento XML, la cardinalidad del elemento “mayor_de_edad” es (0-1), siendo opcional su aparición:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE persona [
   <!ELEMENT persona (nombre, mayor_de_edad?)>
   <!ELEMENT nombre (#PCDATA)>
   <!ELEMENT mayor_de_edad EMPTY>
]>

<persona>
   <nombre>Iker</nombre>
   <mayor_de_edad/>
</persona>
```

Operador de cardinalidad “?” (interrogación)

- El siguiente documento también es válido:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE persona [
   <!ELEMENT persona (nombre, mayor_de_edad?)>
   <!ELEMENT nombre (#PCDATA)>
   <!ELEMENT mayor_de_edad EMPTY>
]>

<persona>
   <nombre>Pedro</nombre>
</persona>
```

Elementos opcionales

- En la DTD asociada a un documento XML, se pueden declarar elementos que contengan elementos opcionales. Para ello, se utiliza el *operador de elección*, representado por una barra vertical (|).

- **EJEMPLO** En el siguiente documento XML el elemento “articulo” puede contener un elemento “codigo" o un elemento “id”; obligatoriamente uno de ellos, pero no ambos:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE articulo [
   <!ELEMENT articulo (codigo | id)>
   <!ELEMENT codigo (#PCDATA)>
   <!ELEMENT id (#PCDATA)>
]>

<articulo>
   <codigo>AF-33</codigo>
</articulo>
```

Operador de elección “ | ” y operador “*”

- **EJEMPLO** En la DTD del siguiente documento XML se indica que el elemento “articulos” puede contener varios elementos “codigo" e “id”:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE articulos [
   <!ELEMENT articulos (codigo | id)*>
   <!ELEMENT codigo (#PCDATA)>
   <!ELEMENT id (#PCDATA)>
]>

<articulos>
   <codigo>AF-32</codigo>
   <id>3891</id>
   <codigo>AF-50</codigo>
   <codigo>AF-89</codigo>
</articulos>
```

Operador de elección “ | ” y operador “*”

- Con el operador “*****” se ha indicado que el contenido del elemento “articulos” tiene cardinalidad (0-n). Por tanto, el elemento “articulos” puede:
  - Estar vacío.
  - Contener un elemento “codigo”.
  - Contener un elemento “id”.
  - Contener un elemento “codigo” y un elemento “id”.
  - Contener un elemento “codigo” y varios elementos “id”.
  - Contener un elemento “id” y varios elementos “codigo”.
  - Contener varios elementos “codigo” y varios elementos “id”.

- Dentro del elemento “articulos” pueden aparecer elementos “codigo” e “id” en cualquier orden.

Operador de elección “ | ” en una secuencia de elementos

- **EJEMPLO** En el siguiente documento XML, pueden aparecer cero o más elementos “articulo” que contengan un elemento “codigo” o un elemento “id”, y obligatoriamente un elemento “nombre”:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE articulos [
   <!ELEMENT articulos (articulo)*>
   <!ELEMENT articulo ((codigo | id), nombre)>
   <!ELEMENT codigo (#PCDATA)>
   <!ELEMENT id (#PCDATA)>
   <!ELEMENT nombre (#PCDATA)>
]>
<articulos>
   <articulo>
      <codigo>AF-47</codigo>
      <nombre>Martillo</nombre>
   </articulo>
   <articulo>
      <id>2056</id>
      <nombre>Destornillador</nombre>
   </articulo>
</articulos>
```

Secuencia de elementos en una lista de opciones

- **EJEMPLO** En la DTD del siguiente documento XML se ha indicado que pueden aparecer cero o más elementos “localidad”. En el caso de aparecer, cada uno de ellos contendrá los elementos “pais” y “ciudad”, o alternativamente un elemento “codigo_postal”:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE localidades [
   <!ELEMENT localidades (localidad)*>
   <!ELEMENT localidad ((pais, ciudad) | codigo_postal)>
   <!ELEMENT pais (#PCDATA)>
   <!ELEMENT ciudad (#PCDATA)>
   <!ELEMENT codigo_postal (#PCDATA)>
]>

<localidades>
   <localidad>
      <pais>España</pais>
      <ciudad>Valencia</ciudad>
   </localidad>
   <localidad>
      <codigo_postal>31015</codigo_postal>
   </localidad>
</localidades>
```

#PCDATA en una lista de opciones permite contenido mixto

- **EJEMPLO** Al utilizar el *operador de elección* (|) en una DTD, si una de las opciones es **#PCDATA**, esta debe escribirse en primer lugar:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE articulos [
   <!ELEMENT articulos (#PCDATA | codigo | id)*>
   <!ELEMENT codigo (#PCDATA)>
   <!ELEMENT id (#PCDATA)>
]>
<articulos>
   <id>8608</id>
   Teclado
   <codigo>AF-18</codigo>
   <codigo>AF-45</codigo>
   Disquetera
   <id>7552</id>
   <id>4602</id>
</articulos>
```

- Fíjese que, el elemento “articulos” de este documento, puede contener contenido mixto, es decir, texto y otros elementos.

#PCDATA en una lista de opciones permite contenido mixto

- **EJEMPLO** Véase, en este ejemplo, que el elemento “provincia” puede aparecer cero o más veces, pudiendo contener contenido mixto:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE localidades [
   <!ELEMENT localidades (provincia*)>
   <!ELEMENT provincia (#PCDATA | ciudad | codigo_postal)*>
   <!ELEMENT ciudad (#PCDATA)>
   <!ELEMENT codigo_postal (#PCDATA)>
]>
<localidades>
   <provincia>
      Navarra
      <ciudad>Estella</ciudad>
      <codigo_postal>31015</codigo_postal>
      <ciudad>Tafalla</ciudad>
   </provincia>
   <provincia>
      Valencia
      <codigo_postal>46520</codigo_postal>
   </provincia>
</localidades>
```

Declaración de atributos

- La sintaxis básica para declarar un atributo en una DTD es:

```
<!ATTLIST nombre-del-elemento nombre-del-atributo
tipo-de-atributo valor-del-atributo>
```

Declaración de un atributo indicando un `valor` por defecto

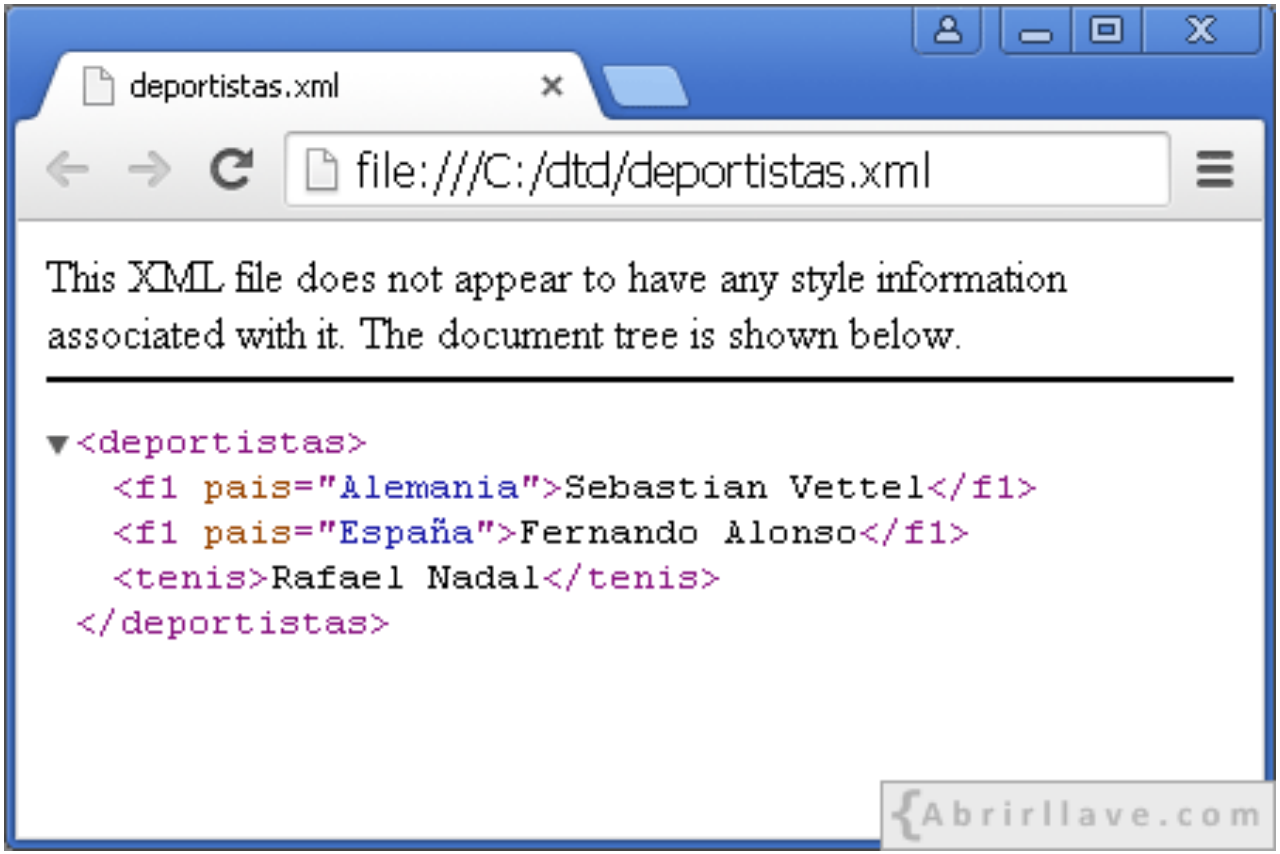
- **EJEMPLO** En la DTD del siguiente documento XML se ha indicado que el elemento “f1” puede tener el atributo “pais”:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE deportistas [
   <!ELEMENT deportistas (futbol | f1 | tenis)*>
   <!ELEMENT futbol (#PCDATA)>
   <!ELEMENT f1 (#PCDATA)>
      <!ATTLIST f1 pais CDATA "España">
   <!ELEMENT tenis (#PCDATA)>
]>
<deportistas>
   <f1 pais="Alemania">Sebastian Vettel</f1>
   <f1>Fernando Alonso</f1>
   <tenis>Rafael Nadal</tenis>
</deportistas>
```

- Para el elemento “f1”, **`pais`** es un atributo definido de tipo **`CDATA`** (*Character DATA*), es decir, su valor será una cadena de caracteres.
- Al no indicarse el país de Fernando Alonso, por defecto es **`"España"`**.
- Para Sebastian Vettel, al atributo **`pais`** se le ha asignado **`"Alemania"`**, que es un valor distinto al **`valor-del-atributo`**, que por defecto es **`"España"`**.

Declaración de un atributo indicando un `valor` por defecto

- Al visualizar el documento ***"deportistas.xml"*** en un navegador web, se verá algo parecido a:



Declaración de varios atributos en un elemento

- En la DTD del siguiente documento XML se ha indicado que el elemento “f1” puede tener tres atributos (**pais**, **fecha_de_nacimiento** y **equipo**):

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE deportistas [
   <!ELEMENT deportistas (futbol | f1 | tenis)*>
   <!ELEMENT futbol (#PCDATA)>
   <!ELEMENT f1 (#PCDATA)>
      <!ATTLIST f1 pais CDATA "España">
      <!ATTLIST f1 fecha_de_nacimiento CDATA #IMPLIED>
      <!ATTLIST f1 equipo CDATA #REQUIRED>
   <!ELEMENT tenis (#PCDATA)>
]>
<deportistas>
   <f1 pais="Alemania" fecha_de_nacimiento="03/07/1987"
       equipo="Ferrari">Sebastian Vettel</f1>
   <f1 equipo="McLaren">Fernando Alonso</f1>
   <tenis>Rafael Nadal</tenis>
</deportistas>
```

- Obsérvese que, en este ejemplo, el atributo **equipo** es obligatorio escribirlo, **#REQUIRED**. Mientras que, el atributo **fecha_de_nacimiento** es opcional, **#IMPLIED**.

Declaración de varios atributos en un elemento

- En una DTD, cuando se declara más de un atributo para un elemento –como se ha hecho en este caso– no es necesario escribir varias veces **<!ATTLIST**, pudiéndose escribir, por ejemplo:

```
<!ATTLIST f1 pais CDATA "España"
          fecha_de_nacimiento CDATA #IMPLIED
          equipo CDATA #REQUIRED>
```

Tipos de declaración de atributos

- En DTD, existen los siguientes tipos de declaración de atributos:

Tipos de declaración de atributos en DTD	
Valor	***Significado***
`valor` entre comillas dobles (") o simples (').	El atributo tiene un valor por defecto.
`#REQUIRED`	El atributo es obligatorio escribirlo.
`#IMPLIED`	El atributo es opcional escribirlo.
`#FIXED valor` entre comillas dobles (") o simples (').	El valor del atributo es fijo.

Atributo obligatorio - #REQUIRED

- **EJEMPLO** En la DTD interna del siguiente documento XML se ha declarado un atributo indicando que es obligatorio, es decir, **#REQUIRED**:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE deportistas [
   <!ELEMENT deportistas (futbol | f1 | tenis)*>
   <!ELEMENT futbol (#PCDATA)>
   <!ELEMENT f1 (#PCDATA)>
      <!ATTLIST f1 pais CDATA #REQUIRED>
   <!ELEMENT tenis (#PCDATA)>
]>
<deportistas>
   <f1 pais="Alemania">Sebastian Vettel</f1>
   <f1>Fernando Alonso</f1>
   <tenis>Rafael Nadal</tenis>
</deportistas>
```

- En este ejemplo, es obligatorio escribir el atributo **pais** en los elementos "f1". Por tanto, aunque el documento XML está bien formado, habría que indicar el **pais** de Fernando Alonso para que fuese válido.

```
<f1 pais="España">Fernando Alonso</f1>
```

- De Rafa Nadal no es obligatorio indicar su país, ni se puede hacer.

Atributo opcional - #IMPLIED

- **EJEMPLO** En una DTD, para especificar que un atributo es opcional escribirlo o no, hay que indicarlo mediante **#IMPLIED**:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE deportistas [
   <!ELEMENT deportistas (futbol | f1 | tenis)*>
   <!ELEMENT futbol (#PCDATA)>
   <!ELEMENT f1 (#PCDATA)>
      <!ATTLIST f1 pais CDATA #IMPLIED>
   <!ELEMENT tenis (#PCDATA)>
]>
<deportistas>
   <f1 pais="Alemania">Sebastian Vettel</f1>
   <f1>Fernando Alonso</f1>
   <tenis>Rafael Nadal</tenis>
</deportistas>
```

- En este caso, el atributo **pais** es opcional para los elementos “f1” que aparezcan en el documento XML. Así pues, obsérvese que, aunque no se ha indicado el país de Fernando Alonso, el documento es válido.

Atributo con valor fijo - #FIXED valor

- **EJEMPLO** Cuando en una DTD, se quiere declarar un atributo que tome un valor fijo, esto se puede hacer con **#FIXED valor**:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE deportistas [
   <!ELEMENT deportistas (futbol | f1 | tenis)*>
   <!ELEMENT futbol (#PCDATA)>
   <!ELEMENT f1 (#PCDATA)>
      <!ATTLIST f1 pais CDATA #FIXED "España">
   <!ELEMENT tenis (#PCDATA)>
]>
<deportistas>
   <f1 pais="España">Carlos Sainz</f1>
   <f1>Fernando Alonso</f1>
   <tenis>Rafael Nadal</tenis>
</deportistas>
```

- Según la DTD de este documento XML, todos los elementos “f1” que aparezcan tendrán el atributo **pais** con el valor **"España"**. Por tanto, no es necesario haberlo escrito para Carlos Sainz. De hecho, si se hubiese escrito otro valor, el documento no sería válido.

Atributo con valor fijo - #FIXED valor

- De modo que, para este caso, al visualizar el documento XML en un navegador web, se mostrará algo parecido a: