DTD

Document Type Definition

Profesor: Alonso Rebolo

Curso: 2019-2020

¿Qué es DTD?

DTD (Document Type Definition) es un documento en el que se elabora la estructura de un documento XML.

El documento XML se confronta con el DTD y si el XML sigue la estructura creada en el DTD y además, el documento está libre de errores sintácticos (Bien formado), entonces el documento XML será VÁLIDO.

Declaración de tipo de documento

Un DTD se puede escribir tanto interna o externamente al XML. En nuestro curso, y lo aconsejable, es crear el DTD en un archivo externo al XML. Por lo expuesto, solo se mencionará la declaración externa.

Documento XML asociado a una DTD externa

Existen dos tipos: Privada y Pública.

Privada: **<!DOCTYPE elemento-raíz SYSTEM “URI”>**

Pública: **<!DOCTYPE elemento-raíz PUBLIC “identificador-público” “URI”>**

Ejemplos:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE marcadores SYSTEM "marcadores.dtd">
<marcadores>
    <pagina>
        <nombre>Abrirllave</nombre>
```

```
<?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>Titulo</title>
   </head>
```

Estructura de un documento XML

En una DTD se pueden declarar:

- **Elementos**
- **Atributos**
- **Entidades**
- **Notaciones**

Cuando un documento XML no tiene errores sintácticos y además cumple con lo indicado en las declaraciones de elementos, atributos, entidades y notaciones que están codificadas en la DTD, entonces el XML será un documento válido.

Declaración de elementos

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

Tipo-de-contenido podrá ser:

- **Texto (#PCDATA) «Parsed Character Data»**
- **Otros elementos (hijos)**
- **Estar vacío (EMPTY)**
- **Mixto -texto y otros elementos- (ANY)**

#PCDATA (Parsed Character DATA)

<!ELEMENT ciudad (#PCDATA)>

En este caso se ha indicado que el elemento “ciudad” tendrá como valor una cadena de caracteres analizable por el parser.

En una primera fase, un analizador léxico convierte al texto entrante en tokens (componentes léxicos que tienen significado propio) que pasan a ser analizados por el parser.

Un parser es un programa informático que lleva a cabo un anális sintáctico analizando una cadena de acuerdo a las reglas de una gramática formal.

El parser convierte el texto entrante en otras estructuras, por lo general árboles, que son más útiles para el posterior análisis.

En XML:

```
<ciudad>Roma</ciudad>
```

Elemento que contiene otros elementos

<!ELEMENT ciudad (nombre, pais)>

El elemento ciudad está compuesto por los elementos nombre y país.

También se puede decir que nombre y país son hijos de ciudad.

El orden de colocación de los hijos en el documento XML tiene que ser el Mismo que el expresado en el DTD

```
<!ELEMENT ciudad (nombre, pais)>
<!ELEMENT nombre (#PCDATA)>
<!ELEMENT pais (#PCDATA)>
```

En XML:

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

Elementos vacíos

<!ELEMENT mayor_de_edad EMPTY>

El elemento mayor_de_edad no contiene ningún valor aunque sí puede contener atributos.

```
<!ELEMENT mayor_de_edad EMPTY>
```

En XML:

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

También:
<mayor_de_edad></mayor_de_edad>

El elemento vacío y con atributo

```
<!ELEMENT br EMPTY>
<!ATTLIST br descripcion CDATA #REQUIRED>
```

En XML:

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

Elementos con contenido mixto

<!ELEMENT persona ANY>

El elemento persona puede contener texto y otros elementos

```
<!ELEMENT persona ANY>
<!ELEMENT nombre (#PCDATA)>
<!ELEMENT ciudad (#PCDATA)>
```

En XML:

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

Cardinalidad de los elementos

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.

```
<!ELEMENT personas (nombre+)>
```

```
<personas>
    <nombre>Ana</nombre>
    <nombre>Iker</nombre>
    <nombre>Elsa</nombre>
</personas>
```

```
<!ELEMENT receta_de_cocina (nombre, ingrediente*)>
```

```
<receta_de_cocina>
    <nombre>Tortilla de patatas</nombre>
    <ingrediente>Huevo</ingrediente>
    <ingrediente>Patata</ingrediente>
    <ingrediente>Aceite</ingrediente>
    <ingrediente>Sal</ingrediente>
</receta_de_cocina>
```

Elementos opcionales

Se utiliza la barra vertical | que puede interpretarse como OR Exclusivo.
Uno de los elementos será obligatorio pero solo uno.

```
<!ELEMENT articulo (codigo | id)>
<!ELEMENT codigo (#PCDATA)>
<!ELEMENT id (#PCDATA)>
```

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

Elementos opcionales

Si se utiliza la barra vertical | con el operador * se indica que habrá 0 o más elementos dentro de artículo en cualquier orden y cualquier combinación, entre 0 y muchos elementos <código> y elementos <id>.

```
<!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>
```

Elementos opcionales

Se puede utilizar la barra vertical | en combinación con otros elementos.

```
<!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>
```

Elementos opcionales

Otra combinación posible de elementos con la barra vertical | es:

```
<!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>
```

Elementos opcionales

Se puede combinar la barra vertical | con #PCDATA

```
<!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>
```

Declaración de Atributos

La sintaxis para declarar un atributo es:

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

```
<!ELEMENT deportistas (futbol | f1 | tenis)*>
<!ELEMENT futbol (#PCDATA)>
<!ELEMENT f1 (#PCDATA)>
   <!ATTLIST f1 pais CDATA "España">
<!ELEMENT tenis (#PCDATA)>
```

Valor por omisión

```
<deportistas>
   <f1 pais="Alemania">Sebastian Vettel</f1>
   <f1>Fernando Alonso</f1>
   <tenis>Rafael Nadal</tenis>
</deportistas>
```

CDATA: Character DATA

Declaración de Atributos

Si se declaran varios atributos para un elemento

```
<!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)>
```

#IMPLIED: Opcional

#REQUIRED: Obligatorio

O de forma más reducida

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

```
<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>
```

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.

#FIXED hace referencia a que el valor es una constante por lo que no se puede modificar.

```
<!ELEMENT f1 (#PCDATA)>
   <!ATTLIST f1 pais CDATA #FIXED "España">
```

Tipos de Atributos

Tipo	*Descripción*
CDATA	(*Character DATA*) El valor son datos de tipo carácter, es decir, texto.
Enumerado	El valor puede ser uno de los pertenecientes a una lista de valores escritos entre *paréntesis* "**()**" y separados por el carácter "**|**".
ID	El valor es un identificador único.
IDREF	El valor es un identificador que tiene que existir en otro atributo **ID** del documento XML.
IDREFS	El valor es una lista de valores que existan en otros atributos **ID** del documento XML, separados por espacios en blanco.
NMTOKEN	El valor es una cadena de caracteres, pudiendo contener letras minúsculas, letras mayúsculas, números, puntos " . ", guiones medios "**-**", guiones bajos "**_**" o el carácter dos puntos "**:**".
NMTOKENS	El valor puede contener uno o varios valores de tipo **NMTOKEN** separados por espacios en blanco.
NOTATION	El valor es el nombre de una notación.
ENTITY	El valor es el nombre de una entidad.
ENTITIES	El valor puede contener uno o varios valores de tipo **ENTITY** separados por espacios en blanco.
Especiales	Existen dos atributos especiales: **xml:lang** y **xml:space**.

Ejemplos Tipos de Atributos

<!ATTLIST ciudad país CDATA #REQUIRED> Cadenas de caracteres Tipo CDATA

<!ATTLIST ciudad país (ESP | FRA | ITA | ALE) “ESP”> Tipo ENUMERADO

<!ATTLIST ciudad país ID #REQUIRED> Tipo ID. El valor de este tipo de atributos es único

Ejemplos Tipos de Atributos

Tipo IDREF

Este tipo de atributo tienen un valor que tiene que existir en otro atributo ID del XML

```
<!ELEMENT cine (directores, peliculas)>
<!ELEMENT directores (director)*>
   <!ELEMENT director (#PCDATA)>
      <!ATTLIST director coddir ID #REQUIRED>
<!ELEMENT peliculas (pelicula)*>
   <!ELEMENT pelicula (#PCDATA)>
      <!ATTLIST pelicula direccion IDREF #REQUIRED>
```

```
<cine>
   <directores>
      <director coddir="CE">Clint Eastwood</director>
      <director coddir="JC">James Cameron</director>
   </directores>
   <peliculas>
      <pelicula direccion="JC">Avatar</pelicula>
      <pelicula direccion="CE">Mystic River</pelicula>
      <pelicula direccion="JC">Titanic</pelicula>
   </peliculas>
</cine>
```

Ejemplos Tipos de Atributos

Tipo IDREFS

Este tipo de atributo tienen un valor o una lista de valores que tienen que existir en otros atributos ID del XML

```
<!ELEMENT cine (peliculas, directores)>
<!ELEMENT peliculas (pelicula)*>
   <!ELEMENT pelicula (#PCDATA)>
      <!ATTLIST pelicula codpel ID #REQUIRED>
<!ELEMENT directores (director)*>
   <!ELEMENT director (#PCDATA)>
      <!ATTLIST director filmografia IDREFS #REQUIRED>
```

```
<cine>
   <peliculas>
      <pelicula codpel="P1">Avatar</pelicula>
      <pelicula codpel="P2">Mystic River</pelicula>
      <pelicula codpel="P3">The Terminator</pelicula>
      <pelicula codpel="P4">Titanic</pelicula>
   </peliculas>
   <directores>
      <director filmografia="P2">Clint Eastwood</director>
      <director filmografia="P1 P3 P4">James Cameron</director>
   </directores>
</cine>
```

Tipo de NMTOKEN (NaMe TOKEN)

Estos atributos tendrán de valor una cadena de caracteres incluyendo letras mayúsculas y minúsculas, números, puntos, guiones medios y bajos y el caracteres de dos puntos “:”

No podrán tener espacios en blanco INTERNOS. Si los espacios están al principio o al final no se tienen en cuenta

```
<!ELEMENT usuarios (usuario)*>
<!ELEMENT usuario (#PCDATA)>
<!ATTLIST usuario clave NMTOKEN #REQUIRED>
```

```
<usuarios>
  <usuario clave="123456789">Ana</usuario>
  <usuario clave="ab-c-d-fg">Iker</usuario>
  <usuario clave="A1_B2..C3">Elsa</usuario>
</usuarios>
```

Tipo de NMTOKENS (NaMe TOKENS)

Estos atributos tendrán de valor una o más cadenas de caracteres incluyendo letras mayúsculas y minúsculas, números, puntos, guiones medios y bajos y el caracteres de dos puntos “:”

```
<!ELEMENT usuarios (usuario)*>
<!ELEMENT usuario (#PCDATA)>
<!ATTLIST usuario codigos NMTOKENS #REQUIRED>
```

```
<usuarios>
  <usuario codigos="1234 567 89">Ana</usuario>
  <usuario codigos="ab c-d fg">Iker</usuario>
  <usuario codigos="A1:B2">Elsa</usuario>
</usuarios>
```

Tipo NOTATION

El valor del atributo puede ser el nombre de una notación.

```
<!ELEMENT animales (animal)*>
<!ELEMENT animal (nombre)>
<!ELEMENT nombre (#PCDATA)>
<!ATTLIST animal
    imagen CDATA #IMPLIED
    tipo_de_imagen NOTATION (jpg | gif | png) #IMPLIED>

<!NOTATION gif SYSTEM "image/gif">
<!NOTATION jpg SYSTEM "image/jpeg">
<!NOTATION png SYSTEM "image/png">
```

En este ejemplo, las notaciones **gif**, **jpg** y **png** son declaraciones de los siguientes tipos MIME (*Multipurpose Internet Mail Extensions*, Extensiones Multipropósito de Correo de Internet): *image/gif*, *image/jpeg* e *image/png*.

```
<animales>
    <animal imagen="ballena-azul.gif" tipo_de_imagen="gif">
        <nombre>Ballena</nombre>
    </animal>
    <animal imagen="leon-dormido.png" tipo_de_imagen="png">
        <nombre>Leon</nombre>
    </animal>
</animales>
```

Tipo ENTITY

El valor del atributo puede ser el nombre de una ENTIDAD.

```
<!ELEMENT animales (animal)*>
<!ELEMENT animal EMPTY>
<!ATTLIST animal imagen ENTITY #REQUIRED>

<!ENTITY ballena SYSTEM "ballena.gif" NDATA gif>
<!ENTITY delfin SYSTEM "delfin.gif" NDATA gif>

<!NOTATION gif SYSTEM "image/gif">
```

```
<animales>
    <animal imagen="ballena"/>
    <animal imagen="delfin"/>
</animales>
```

- En la DTD de este ejemplo se está indicando que los valores –datos– de las entidades (**ballena** y **delfin**) van a ser cargados desde una URI (*Uniform Resource Identifier*, Identificador Uniforme de Recurso). En este caso, se hace referencia a los archivos externos *"ballena.gif"* y *"delfin.gif"*.
- Con **NDATA** (*Notation Data*) se ha asociado a las entidades **ballena** y **delfin** con la notación **gif**.
- La notación **gif** es una declaración del tipo *MIME image/gif*.

Tipo ENTITIES

El valor del atributo puede ser el nombre de una o varias ENTIDADES.

```
<!ELEMENT animales (grupos)*>
<!ELEMENT grupos EMPTY>
<!ATTLIST grupos imagenes ENTITIES #REQUIRED>

<!ENTITY ballena SYSTEM "ballena.gif" NDATA gif>
<!ENTITY delfin SYSTEM "delfin.gif" NDATA gif>
<!ENTITY elefante SYSTEM "elefante.gif" NDATA gif>
<!ENTITY leon SYSTEM "leon.gif" NDATA gif>
<!ENTITY oso SYSTEM "oso.gif" NDATA gif>

<!NOTATION gif SYSTEM "image/gif">
```

```
<animales>
   <grupos imagenes="ballena"/>
   <grupos imagenes="ballena delfin"/>
   <grupos imagenes="elefante leon oso"/>
   <grupos imagenes="ballena elefante"/>
</animales>
```

ATRIBUTOS ESPECIALES

xml:lang

Permite indicar el idioma del contenido y de los valores de los atributos de un elemento declarado.

```
<!ELEMENT siglas (sigla)*>
<!ELEMENT sigla (significado, traduccion)>
   <!ATTLIST sigla letras CDATA #REQUIRED>
   <!ATTLIST sigla xml:lang CDATA "en">
<!ELEMENT significado (#PCDATA)>
<!ELEMENT traduccion (#PCDATA)>
   <!ATTLIST traduccion xml:lang CDATA #FIXED "es">
```

- Inicialmente, para el elemento "sigla" se ha indicado el idioma inglés, "**en**", por defecto.
- No obstante, después se ha fijado el valor "**es**", del español, para el atributo **xml:lang** del elemento "traduccion".
- Por otra parte, para el CERN se ha especificado que el idioma es el francés, "**fr**".

```
<siglas>
   <sigla letras="ANSI">
      <significado>American National Standards Institute</significado>
      <traduccion>Instituto Nacional Estadounidense de Estándares</traduccion>
   </sigla>
   <sigla letras="ISO">
      <significado>International Organization for Standardization</significado>
      <traduccion>Organización Internacional de Normalización</traduccion>
   </sigla>
   <sigla letras="CERN" xml:lang="fr">
      <significado>Conseil Européen pour la Recherche Nucléaire</significado>
      <traduccion>Organización Europea para la Investigación Nuclear</traduccion>
   </sigla>
</siglas>
```

ATRIBUTOS ESPECIALES

xml:space

Determina como tratat los espacios en blanco, las tabulaciones y los retornos de carro que aparezcan en el contenido de un elemento.
Es de tipo enumerado.

```
<!ELEMENT programas (programa)*>
<!ELEMENT programa (#PCDATA)>
<!ATTLIST programa xml:space (default|preserve) "preserve">
```

```
<programas>
  <programa>/* Programa: Hola mundo */

#include &lt;conio.h&gt;
#include &lt;stdio.h&gt;

int main()
{
    printf( &quot;Hola mundo.&quot; );

    getch(); /* Pausa */

    return 0;
}</programa>
  <programa>/* Programa: Calificación según nota */

#include &lt;conio.h&gt;
#include &lt;stdio.h&gt;

int main()
{
    float nota;

    printf( &quot;\n   Introduzca nota (real): &quot; );
    scanf( &quot;%f&quot;, &amp;nota );

    if ( nota &amp;gt;= 5 )
        printf( &quot;\n   APROBADO&quot; );
    else
        printf( &quot;\n   SUSPENDIDO&quot; );

    getch(); /* Pausa */

    return 0;
}</programa>
</programas>
```

DECLARACIÓN DE ENTIDADES

Se pueden declarar entidades generales y paramétricas (de parámetro).

Entidades Generales, se pueden utilizar en un documento XML y en su DTD.
Son de tipo:

- **Internas analizables (parsed)** ✘
- **Externas analizables (parsed)**
- **Externas no analizables (unparsed)**

Entidades Paramétricas solo se pueden utilizar en la DTD.
Son de tipo:

- **Internas analizables (parsed)** ✘
- **Externas analizables (parsed)**

ENTIDADES Generales Externas Analizables

Hay dos tipos:

Privadas:
<!ENTITY nombre-de-la-entidad SYSTEM “URI”>

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

<!ENTITY escritor SYSTEM "escritor.txt">
```

```
<textos>
   <texto>El Quijote fue escrito por &escritor;.</texto>
</textos>
```

Pública:
<!ENTITY nombre-de-la-entidad PUBLIC “identificador-público” “URI”>

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

<!ENTITY escritor PUBLIC "-//W3C//TEXT escritor//EN"
"http://www.abrirllave.com/dtd/escritor.txt">
```

ENTIDADES Generales Externas NO Analizables

Hay dos tipos:

Privadas:
<!ENTITY nombre-de-la-entidad SYSTEM “URI” NDATA notación>

```
<!ELEMENT imagen EMPTY>
   <!ATTLIST imagen fuente ENTITY #REQUIRED>

<!ENTITY logo SYSTEM "logo.gif" NDATA gif>

<!NOTATION gif SYSTEM "image/gif">
```

```
<imagen fuente="logo"/>
```

Pública:
<!ENTITY nombre-de-la-entidad PUBLIC “identificador-público” “URI” NDATA notación>

```
<!ELEMENT imagen EMPTY>
   <!ATTLIST imagen fuente ENTITY #REQUIRED>

<!ENTITY logo PUBLIC "-//W3C//GIF logo//EN"
"http://www.abrirllave.com/dtd/logo.gif" NDATA gif>

<!NOTATION gif SYSTEM "image/gif">
```

ENTIDADES Paramétricas Externas Analizables

Hay dos tipos:

Privadas:

Declaración de la entidad persona

<!ENTITY % nombre-de-la-entidad SYSTEM “URI”>
%nombre-de-la-entidad;

```
<!ENTITY % persona SYSTEM "persona.dtd">
%persona;
```

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

Suponiendo que el archivo *“persona.dtd”* contenga:

```
<!ELEMENT persona (nombre, mayor_de_edad?, ciudad)>
<!ELEMENT nombre (#PCDATA)>
<!ELEMENT mayor_de_edad EMPTY>
<!ELEMENT ciudad (#PCDATA)>
```

ENTIDADES Paramétricas Externas Analizables

Hay dos tipos:

Públicas:

Declaración de la entidad persona

<!ENTITY % nombre-de-la-entidad PUBLIC
“identificador-público” “URI”>
%nombre-de-la-entidad;

```
<!ENTITY % persona PUBLIC "-//W3C//TEXT persona//EN"
"http://www.abrirllave.com/dtd/persona.dtd">
%persona;
```

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

ENTIDADES dentro de otra Entidad

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

<!ENTITY color "azul">
<!ENTITY frase "El cielo es &color;.">
```

```
<frase>&frase;</frase>
```

Resultado en el navegador:

```
<frase>El cielo es azul.</frase>
```

Referencia circular o recursiva a ENTIDADES

Hay que tener cuidado con las referencias circulares o recursivas porque la DTD que se genera es incorrecta.

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

<!ENTITY frase1 "Esta frase incluye a la &frase2;.">
<!ENTITY frase2 "Esta frase incluye a la &frase1;.">
```

```
<frase>&frase1;</frase>
```

Para que la DTD sea correcta hay que quitar una de las referencias:

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

<!ENTITY frase1 "Esta frase incluye a la &frase2;.">
<!ENTITY frase2 "segunda frase">
```

Declaración de Notaciones

Se pueden declarar NOTACIONES (NOTATION) privadas (SYSTEM) y Públicas (PUBLIC)

Sintaxis:

```
<!NOTATION nombre-de-la-notación SYSTEM "identificador-del-sistema">
```

```
<!NOTATION nombre-de-la-notación PUBLIC "identificador-público">
```

```
<!NOTATION nombre-de-la-notación PUBLIC "identificador-público"
"identificador-del-sistema">
```

Ejemplos de Notaciones Privadas

Ejemplo 1

EJEMPLO En la DTD del siguiente documento XML, se indica que los elementos "fruta" que se escriban, tienen que incluir obligatoriamente el atributo **foto**, cuyo valor será una entidad y, para indicar el formato de dicha entidad, se usa la notación **gif**:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE frutas [
   <!ELEMENT frutas (fruta)*>
   <!ELEMENT fruta EMPTY>
   <!ATTLIST fruta foto ENTITY #REQUIRED>

   <!ENTITY manzana SYSTEM "manzana.gif" NDATA gif>
   <!ENTITY naranja SYSTEM "naranja.gif" NDATA gif>

   <!NOTATION gif SYSTEM "image/gif">
]>

<frutas>
   <fruta foto="manzana"/>
   <fruta foto="naranja"/>
</frutas>
```

- En la DTD de este ejemplo se está indicando que los valores –datos– de las entidades (**manzana** y **naranaja**) van a ser cargados desde una URI (*Uniform Resource Identifier*, Identificador Uniforme de Recurso). En este caso, se hace referencia a los archivos externos *"manzana.gif"* y *"naranja.gif"*.
- Con **NDATA** (*Notation Data*) se ha asociado a las entidades **manzana** y **naranaja** con la notación **gif**.
- La notación **gif** es una declaración del tipo *MIME image/gif*.

```
<frutas>
   <fruta foto="manzana"/>
   <fruta foto="naranja"/>
</frutas>
```

Ejemplos de Notaciones Privadas

Ejemplo 2

EJEMPLO Si en el sistema existe, por ejemplo, un programa llamado *"procesadorGIF.exe"* en la carpeta *"aplicaciones"* capaz de procesar imágenes GIF (*Graphics Interchange Format*, Formato de Intercambio de Gráficos), también se podría escribir:

```
<!ELEMENT frutas (fruta)*>
<!ELEMENT fruta EMPTY>
<!ATTLIST fruta foto ENTITY #REQUIRED>

<!ENTITY manzana SYSTEM "manzana.gif" NDATA gif>
<!ENTITY naranja SYSTEM "naranja.gif" NDATA gif>

<!NOTATION gif SYSTEM "aplicaciones/procesadorGIF.exe">
```

```
<frutas>
    <fruta foto="manzana"/>
    <fruta foto="naranja"/>
</frutas>
```

Ejemplos de Notaciones Públicas

Ejemplo 1

EJEMPLO En la declaración de una notación se puede indicar un identificador público estándar, como por ejemplo, *GIF 1.0*:

```
<!ELEMENT frutas (fruta)*>
<!ELEMENT fruta EMPTY>
<!ATTLIST fruta foto ENTITY #REQUIRED>

<!ENTITY manzana SYSTEM "manzana.gif" NDATA gif>
<!ENTITY naranja SYSTEM "naranja.gif" NDATA gif>

<!NOTATION gif PUBLIC "GIF 1.0">
```

```
<frutas>
    <fruta foto="manzana"/>
    <fruta foto="naranja"/>
</frutas>
```

Ejemplos de Notaciones Públicas

Ejemplo 2

EJEMPLO En la notación escrita en la DTD del siguiente documento XML, se ha declarado el tipo *MIME imagen/gif* e indicado el identificador público estándar *GIF 1.0*:

```
<!ELEMENT frutas (fruta)*>
<!ELEMENT fruta EMPTY>
<!ATTLIST fruta foto ENTITY #REQUIRED>

<!ENTITY manzana SYSTEM "manzana.gif" NDATA gif>
<!ENTITY naranja SYSTEM "naranja.gif" NDATA gif>

<!NOTATION gif PUBLIC "GIF 1.0" "image/gif">
```

```
<frutas>
    <fruta foto="manzana"/>
    <fruta foto="naranja"/>
</frutas>
```

Atributos con valor = Nombre de una Notación

EJEMPLO En la DTD del siguiente documento XML, se indica que los elementos "documento" que se escriban, tienen que incluir obligatoriamente el atributo **version**, cuyo valor será una notación (**h4** o **h5**):

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE documentos [
   <!ELEMENT documentos (documento)*>
   <!ELEMENT documento (#PCDATA)>
   <!ATTLIST documento version NOTATION (h4|h5) #REQUIRED>

   <!NOTATION h5 PUBLIC "HTML 5">
   <!NOTATION h4 PUBLIC "HTML 4.01">
]>

<documentos>
   <documento version="h4"><!-- Código del documento 1. --></documento>
   <documento version="h5"><!-- Código del documento 2. --></documento>
   <documento version="h5"><!-- Código del documento 3. --></documento>
   <documento version="h4"><!-- Código del documento 4. --></documento>
</documentos>
```

Secciones condicionales

En DTD externas se pueden definir las secciones **IGNORE** e **INCLUDE**, para ignorar o incluir declaraciones. Las sintaxis empleadas para ello son:

```
<![ IGNORE [ declaraciones ]]>
```

```
<![ INCLUDE [ declaraciones ]]>
```

Ejemplos Secciones condicionales

EJEMPLO Si en un archivo llamado ***"persona.dtd"*** se ha escrito:

```
<![ %datos_basicos; [
   <!ELEMENT persona (nombre, edad)>
]]>

<![ %datos_ampliados; [
   <!ELEMENT persona (nombre, apellidos, edad, ciudad)>
]]>

<!ELEMENT nombre (#PCDATA)>
<!ELEMENT apellidos (#PCDATA)>
<!ELEMENT edad (#PCDATA)>
<!ELEMENT ciudad (#PCDATA)>
```

El siguiente documento XML sería válido:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE persona SYSTEM "persona.dtd" [
   <!ENTITY % datos_basicos "INCLUDE">
   <!ENTITY % datos_ampliados "IGNORE">
]>

<persona>
   <nombre>Elsa</nombre>
   <edad>23</edad>
</persona>
```

Espacios de nombres en DTD

```xml
<?xml version="1.0" encoding="UTF-8"?>
<e1:ejemplo xmlns:e1="http://www.abrirllave.com/ejemplo1">

   <e1:carta>
      <e1:palo>Corazones</e1:palo>
      <e1:numero>7</e1:numero>
   </e1:carta>
   <e2:carta xmlns:e2="http://www.abrirllave.com/ejemplo2">
      <e2:carnes>
         <e2:filete_de_ternera precio="12.95"/>
         <e2:solomillo_a_la_pimienta precio="13.60"/>
      </e2:carnes>
      <e2:pescados>
         <e2:lenguado_al_horno precio="16.20"/>
         <e2:merluza_en_salsa_verde precio="15.85"/>
      </e2:pescados>
   </e2:carta>

</e1:ejemplo>
```

```dtd
<!ELEMENT e1:ejemplo (e1:carta, e2:carta)>
   <!ATTLIST e1:ejemplo xmlns:e1 CDATA #FIXED "http://www.abrirllave.com/ejemplo1">
<!ELEMENT e1:carta (e1:palo, e1:numero)>
<!ELEMENT e1:palo (#PCDATA)>
<!ELEMENT e1:numero (#PCDATA)>

<!ELEMENT e2:carta (e2:carnes, e2:pescados)>
   <!ATTLIST e2:carta xmlns:e2 CDATA #FIXED "http://www.abrirllave.com/ejemplo2">
<!ELEMENT e2:carnes (e2:filete_de_ternera, e2:solomillo_a_la_pimienta)>
<!ELEMENT e2:pescados (e2:lenguado_al_horno, e2:merluza_en_salsa_verde)>
<!ELEMENT e2:filete_de_ternera EMPTY>
   <!ATTLIST e2:filete_de_ternera precio CDATA #REQUIRED>
<!ELEMENT e2:solomillo_a_la_pimienta EMPTY>
   <!ATTLIST e2:solomillo_a_la_pimienta precio CDATA #REQUIRED>
<!ELEMENT e2:lenguado_al_horno EMPTY>
   <!ATTLIST e2:lenguado_al_horno precio CDATA #REQUIRED>
<!ELEMENT e2:merluza_en_salsa_verde EMPTY>
   <!ATTLIST e2:merluza_en_salsa_verde precio CDATA #REQUIRED>
```

Comentarios en DTD

<!-- Mi comentario -->

```
<!ELEMENT ciudades (ciudad*)>
<!ELEMENT ciudad (#PCDATA)>
    <!-- pais es atributo del elemento ciudad -->
    <!ATTLIST ciudad pais CDATA #REQUIRED>
```

Comprobación de XML Bien-Formado y válido

Utilizando XML Copy Editor

Comprobar que el documento está Bien Formado

Visualizando el documento XML: <Menú> XML / Comprobar Bien-Formado (F2)

En la parte inferior de la pantalla dará el resultado:

nombre-fichero.xml es bien-formado

Comprobar que el documento es Válido

Visualizando el documento XML: <Menú> XML / Validar / (DTD/XML Schema) (F5)

En la parte inferior de la pantalla dará el resultado:

nombre-fichero.xml is valid