

DTD: Definición de Tipo de Documento

- Qué es una DTD
 - Referencia a una DTD en un documento XML
 - Declaraciones
 - Declaración de entidades
 - Declaración de notaciones
 - Declaración de elementos
 - Declaración de atributos
-

Qué es una DTD

Una DTD es un documento que define la estructura de un documento XML: los elementos, atributos, entidades, notaciones, etc, que pueden aparecer, el orden y el número de veces que pueden aparecer, cuáles pueden ser hijos de cuáles, etc. El procesador XML utiliza la DTD para verificar si un documento es válido, es decir, si el documento cumple las reglas del DTD.

Referencia a una DTD en un documento XML

La DTD que debe utilizar el procesador XML para validar el documento XML se indica mediante la etiqueta DOCTYPE. La DTD puede estar incluida en el propio documento, ser un documento externo o combinarse ambas.

- La DTD puede incluirse en el propio documento, con la siguiente sintaxis:
- `<!DOCTYPE nombre [`
- `... declaraciones ...`
- `]>`
- La DTD puede estar en un documento externo y, si sólo va a ser utilizada por una única aplicación, la sintaxis es la siguiente:

```
<!DOCTYPE nombre SYSTEM "uri">
```

Se puede combinar una DTD externa con una DTD interna, con la siguiente sintaxis:

```
<!DOCTYPE nombre SYSTEM "uri" [  
... declaraciones ...  
>
```

- La DTD puede estar en un documento externo y, si va a ser utilizada por varias aplicaciones, la sintaxis es la siguiente:

```
<!DOCTYPE nombre PUBLIC "fpi" "uri">
```

Se puede combinar una DTD externa con una DTD interna, con la siguiente sintaxis:

```
<!DOCTYPE nombre PUBLIC "fpi" "uri" [  
  ... declaraciones ...  

```

En todos estos casos:

- "nombre" es el nombre del tipo de documento XML, que debe coincidir con el nombre del elemento raíz del documento XML.
- "uri" es el camino (absoluto o relativo) hasta la DTD.
- "fpi" es un indentificador público formal (Formal Public Identifier).

Declaraciones

Las DTDs describen la estructura de los documentos XML mediante declaraciones. Hay cuatro tipos de declaraciones:

- Declaraciones de entidades
- Declaraciones de notaciones
- Declaraciones de elementos, que indican los elementos permitidos en un documento y su contenido (que puede ser simplemente texto u otros elementos).
- Declaraciones de atributos, que indican los atributos permitidos en cada elemento y el tipo o valores permitidos de cada elemento.

Declaración de elementos

Las declaraciones de los elementos siguen la siguiente sintaxis:

```
<!ELEMENT nombreElemento (contenido)>
```

en la que "nombreElemento" es el nombre del elemento, y "(contenido)" una expresión que describe el contenido del elemento.

Para definir el contenido del elemento se pueden utilizar los términos EMPTY, (#PCDATA) o ANY o escribir expresiones más complejas:

- **EMPTY**: significa que el elemento es vacío, es decir, que no puede tener contenido. Los elementos vacíos pueden escribirse con etiquetas de apertura y cierre sin nada entre ellos, ni siquiera espacios, o con una etiqueta vacía. Por ejemplo:

```
<!DOCTYPE ejemplo [  
  <!ELEMENT ejemplo EMPTY>
```

]>



<ejemplo></ejemplo>



<ejemplo />



<ejemplo>Esto es un ejemplo</ejemplo>

<!-- ERROR: contiene texto -->



<ejemplo><a></ejemplo>

<!-- ERROR: contiene un elemento <a> -->

- **(#PCDATA)**: significa que el elemento puede contener texto (debe escribirse entre paréntesis). Por ejemplo:

```
<!DOCTYPE ejemplo [  
  <!ELEMENT ejemplo (#PCDATA)>  
>  
>
```



<ejemplo />



<ejemplo>Esto es un ejemplo</ejemplo>



<ejemplo><a></ejemplo>

<!-- ERROR: contiene un elemento <a> -->

- **ANY**: significa que el elemento puede contener cualquier cosa (texto y otros elementos). Por ejemplo:

```
<!DOCTYPE ejemplo [  
  <!ELEMENT ejemplo ANY>  
  <!ELEMENT a ANY>  
>  
>
```



<ejemplo />



<ejemplo>Esto es un ejemplo</ejemplo>



<ejemplo><a>Esto es un ejemplo</ejemplo>

Para indicar que un elemento puede o debe contener otros elementos se deben indicar los elementos, utilizando los conectores y modificadores siguientes:

- **, (coma)**: significa que el elemento contiene los elementos en el orden indicado. Por ejemplo:

```
<!DOCTYPE ejemplo [  
  <!ELEMENT ejemplo (a, b)>  
  <!ELEMENT a EMPTY>  
  <!ELEMENT b EMPTY>  
>  
>
```





<ejemplo><a /></ejemplo>



<ejemplo><a /><c /></ejemplo>


<!-- ERROR: contiene un elemento <c /> -->


 `<ejemplo><a /></ejemplo>`
`<!-- ERROR: falta el elemento -->`


 `<ejemplo><a /></ejemplo>`
`<!-- ERROR: el orden no es correcto -->`


- **| (o lógico):** significa que el elemento contiene uno de los dos elementos. Por ejemplo:

```
<!DOCTYPE ejemplo [
  <!ELEMENT ejemplo (a | b)>
  <!ELEMENT a EMPTY>
  <!ELEMENT b EMPTY>
]>
```

 `<ejemplo><a /></ejemplo>`


 `<ejemplo></ejemplo>`


 `<ejemplo><a /></ejemplo>`
`<!-- ERROR: están los dos elementos -->`


 `<ejemplo></ejemplo>`
`<!-- ERROR: no hay ningún elemento -->`


- **?:** significa que el elemento puede aparecer o no, pero sólo una vez. Por ejemplo:

```
<!DOCTYPE ejemplo [
  <!ELEMENT ejemplo (a, b?)>
  <!ELEMENT a EMPTY>
  <!ELEMENT b EMPTY>
]>
```

 `<ejemplo><a /></ejemplo>`


 `<ejemplo><a /></ejemplo>`


 `<ejemplo></ejemplo>`
`<!-- ERROR: falta el elemento <a /> -->`


 `<ejemplo></ejemplo>`
`<!-- ERROR: el elemento aparece dos veces -->`


- ***:** significa que el elemento puede no aparecer o aparecer una o más veces. Por ejemplo:

```
<!DOCTYPE ejemplo [
  <!ELEMENT ejemplo (a*, b)>
  <!ELEMENT a EMPTY>
  <!ELEMENT b EMPTY>
]>
```

 `<ejemplo></ejemplo>`

 `<ejemplo><a /></ejemplo>`

 `<ejemplo><a /><a /></ejemplo>`

 `<ejemplo><a /></ejemplo>`
`<!-- ERROR: el elemento <a /> aparece después de -->`

- `+`: significa que el elemento tiene que aparecer una o más veces (no puede no aparecer). Por ejemplo:

```
<!DOCTYPE ejemplo [  
  <!ELEMENT ejemplo (a+, b)>  
  <!ELEMENT a EMPTY>  
  <!ELEMENT b EMPTY>  

```



`<ejemplo><a /></ejemplo>`



`<ejemplo><a /><a /></ejemplo>`



`<ejemplo></ejemplo>`
`<!-- ERROR: falta el elemento <a /> -->`

- `()`: permite agrupar expresiones. Por ejemplo:

```
<!DOCTYPE ejemplo [  
  <!ELEMENT ejemplo (a, (a|b))>  
  <!ELEMENT a EMPTY>  
  <!ELEMENT b EMPTY>  

```



`<ejemplo><a /><a /></ejemplo>`



`<ejemplo><a /></ejemplo>`



`<ejemplo><a /></ejemplo>`
`<!-- ERROR: falta el elemento <a /> o -->`

```
<!DOCTYPE ejemplo [  
  <!ELEMENT ejemplo ((a, b)|(b, a))>  
  <!ELEMENT a EMPTY>  
  <!ELEMENT b EMPTY>  

```



`<ejemplo><a /></ejemplo>`



`<ejemplo><a /></ejemplo>`



`<ejemplo><a /><a /></ejemplo>`
`<!-- ERROR: sólo admite <a /> o <a /> -->`

Declaración de atributos

Una declaración de atributos sigue la siguiente sintaxis:

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

en la que:

- "nombreElemento" es el nombre del elemento para el que se define un atributo.
- "nombreAtributo" es el nombre del atributo.
- "tipoAtributo" es el tipo de datos .
- "valorInicialAtributo" es el valor predeterminado del atributo (aunque también puede indicar otras cosas).

Para definir varios atributos de un mismo elemento, se puede utilizar una o varias declaraciones de atributos. Los siguientes ejemplos son equivalentes:

```
<!ATTLIST nombreElemento nombreAtributo1 tipoAtributo1
valorInicialAtributo1>
<!ATTLIST nombreElemento nombreAtributo2 tipoAtributo2
valorInicialAtributo2>
<!ATTLIST nombreElemento
  nombreAtributo1 tipoAtributo1 valorInicialAtributo1
  nombreAtributo2 tipoAtributo2 valorInicialAtributo2
>
```


Los tipos de atributos son los siguientes:

- **CDATA:** el atributo contiene caracteres (sin restricciones). Por ejemplo:

```
<!DOCTYPE ejemplo [
  <!ELEMENT ejemplo EMPTY>
  <!ATTLIST ejemplo color CDATA #REQUIRED>
]>
<ejemplo></ejemplo>                                <!-- ERROR:
falta el atributo "color", obligatorio debido al #REQUIRED -
-->
<ejemplo color=""></ejemplo>
<ejemplo color="amarillo"></ejemplo>
<ejemplo color="azul marino #000080"></ejemplo>
```


- **NMTOKEN:** el atributo sólo contiene letras, dígitos, y los caracteres punto ".", guión "-", subrayado "_" y dos puntos ":". Por ejemplo:

```
<!DOCTYPE ejemplo [
  <!ELEMENT ejemplo EMPTY>
  <!ATTLIST ejemplo color NMTOKEN #REQUIRED>
]>
<ejemplo color=""></ejemplo>
<ejemplo color="azul-marino"></ejemplo>
<ejemplo color="1"></ejemplo>
<ejemplo color="azul marino"></ejemplo>
<!-- ERROR: hay un espacio en blanco -->
```


 `<ejemplo color="#F0F0F0"></ejemplo>`
`<!-- ERROR: contiene el carácter # -->`


- **NMTOKENS:** el atributo sólo contiene letras, dígitos, y los caracteres punto ".", guión "-", subrayado "_", dos puntos ":" (como el tipo NMTOKEN) y también espacios en blanco. Por ejemplo:

```
<!DOCTYPE ejemplo [
  <!ELEMENT ejemplo EMPTY>
  <!ATTLIST ejemplo color NMTOKENS #REQUIRED>
]>
```

 `<ejemplo color=""></ejemplo>`


 `<ejemplo color="1"></ejemplo>`


 `<ejemplo color="azul marino"></ejemplo>`


 `<ejemplo color="2*2"></ejemplo>`
`<!-- ERROR: hay un asterisco -->`

- **valores:** el atributo sólo puede contener uno de los términos de una lista. La lista se escribe entre paréntesis, con los términos separados por una barra vertical "|". Por ejemplo:

```
<!DOCTYPE ejemplo [
  <!ELEMENT ejemplo EMPTY>
  <!ATTLIST ejemplo color (azul|blanco|rojo) #REQUIRED>
]>
```


 `<ejemplo color=""></ejemplo>`


 `<ejemplo color="azul"></ejemplo>`

 `<ejemplo color="verde"></ejemplo>`
`<!-- ERROR: "verde" no está en la lista de valores -->`

- **ID:** el valor del atributo (no el nombre) debe ser único y no se puede repetir en otros elementos o atributos. Por ejemplo:

```
<!DOCTYPE ejemplo [
  <!ELEMENT ejemplo (libro*)>
  <!ELEMENT libro (#PCDATA) >
  <!ATTLIST libro codigo ID #REQUIRED>
]>
```

 `<ejemplo>`
`<libro codigo="L1">Poema de Gilgamesh</libro>`
`<libro codigo="L2">Los preceptos de Ptah-Hotep</libro>`
`</ejemplo>`

 `<ejemplo>`
`<libro codigo="1">Poema de Gilgamesh</libro>`
`<!-- ERROR: el valor de un atributo de tipo ID no puede`
`empezar con un número -->`
`<libro codigo="L2">Los preceptos de Ptah-Hotep</libro>`
`</ejemplo>`

```

<ejemplo>
  <libro codigo="L1">Poema de Gilgamesh</libro>
  <libro codigo="L1">Los preceptos de Ptah-Hotep</libro>
  <!-- ERROR: el valor "L1" ya se ha utilizado -->
</ejemplo>

```

- **IDREF:** el valor del atributo debe coincidir con el valor del atributo ID de otro elemento. Por ejemplo:

```

<!DOCTYPE ejemplo [
  <!ELEMENT ejemplo ((libro|prestamo)*)>
  <!ELEMENT libro (#PCDATA) >
  <!ATTLIST libro codigo ID #REQUIRED>
  <!ELEMENT prestamo (#PCDATA) >
  <!ATTLIST prestamo libro IDREF #REQUIRED>
]>
<ejemplo>
  <libro codigo="L1">Poema de Gilgamesh</libro>
  <prestamo libro="L1">Numa Nigerio</prestamo>
</ejemplo>
<ejemplo>
  <libro codigo="L1">Poema de Gilgamesh</libro>
  <prestamo libro="L2">Numa Nigerio</prestamo>
  <!-- ERROR: el valor "L2" no es ID de ningún elemento -->
</ejemplo>

```

- **IDREFS:** el valor del atributo es una serie de valores separados por espacios que coinciden con el valor del atributo ID de otros elementos.

```

<!DOCTYPE ejemplo [
  <!ELEMENT ejemplo ((libro|prestamo)*)>
  <!ELEMENT libro (#PCDATA) >
  <!ATTLIST libro codigo ID #REQUIRED>
  <!ELEMENT prestamo (#PCDATA) >
  <!ATTLIST prestamo libro IDREFS #REQUIRED>
]>
<ejemplo>
  <libro codigo="L1">Poema de Gilgamesh</libro>
  <libro codigo="L2">Los preceptos de Ptah-Hotep</libro>
  <prestamo libro="L1 L2">Numa Nigerio</prestamo>
</ejemplo>
<ejemplo>
  <libro codigo="L1">Poema de Gilgamesh</libro>
  <libro codigo="L2">Los preceptos de Ptah-Hotep</libro>
  <prestamo libro="L1">Numa Nigerio</prestamo>
</ejemplo>
<ejemplo>
  <libro codigo="L1">Poema de Gilgamesh</libro>
  <libro codigo="L2">Los preceptos de Ptah-Hotep</libro>
  <prestamo libro="L3">Numa Nigerio</prestamo>
  <!-- ERROR: el valor "L3" no es ID de ningún elemento -->
</ejemplo>

```

- **ENTITY:** el valor del atributo es alguna entidad definida en la DTD.
- **ENTITIES:** el valor del atributo es alguna de las entidades de una lista de entidades definida en la DTD.

- **NOTATION:** el valor del atributo es alguna notación definida en la DTD.

Los valores iniciales de los atributos son los siguientes:

- **#REQUIRED:** el atributo es obligatorio, aunque no se especifica ningún valor predeterminado. Por ejemplo:


```
<!DOCTYPE ejemplo [
  <!ELEMENT ejemplo EMPTY>
  <!ATTLIST ejemplo color CDATA #REQUIRED>
]>
<ejemplo></ejemplo>
<!-- ERROR: falta el atributo "color" -->
<ejemplo color=""></ejemplo>
<ejemplo color="amarillo"></ejemplo>
<ejemplo color="azul marino #000080"></ejemplo>
```

- **#IMPLIED:** el atributo no es obligatorio y no se especifica ningún valor predeterminado. Por ejemplo:

```
<!DOCTYPE ejemplo [
  <!ELEMENT ejemplo EMPTY>
  <!ATTLIST ejemplo color CDATA #IMPLIED>
]>
<ejemplo></ejemplo>
<ejemplo color=""></ejemplo>
<ejemplo color="amarillo"></ejemplo>
<ejemplo color="azul marino #000080"></ejemplo>
```

- **#FIXED valor:** el atributo tiene un valor fijo. Por ejemplo:


```
<!DOCTYPE ejemplo [
  <!ELEMENT ejemplo EMPTY>
  <!ATTLIST ejemplo color CDATA #FIXED "verde">
]>
<ejemplo></ejemplo>
<ejemplo color=""></ejemplo>
<!-- ERROR: el atributo "color" no tiene el valor "verde" -->
<ejemplo color="amarillo"></ejemplo>
<!-- ERROR: el atributo "color" no tiene el valor "verde" -->
```


 `<ejemplo color="verde"></ejemplo>`


- **valor:** el atributo tiene un valor predeterminado. Por ejemplo:


```
<!DOCTYPE ejemplo [  
  <!ELEMENT ejemplo EMPTY>  
  <!ATTLIST ejemplo color CDATA "verde">  

```

 `<ejemplo></ejemplo>`

 `<ejemplo color=""></ejemplo>`

 `<ejemplo color="amarillo"></ejemplo>`

 `<ejemplo color="verde"></ejemplo>`

Declaración de entidades

Una entidad consiste en un nombre y su valor (son similares a las constantes en los lenguajes de programación). Con algunas excepciones, el procesador XML sustituye las referencias a entidades por sus valores antes de procesar el documento. Una vez definida la entidad, se puede utilizar en el documento escribiendo una referencia a la entidad, que empieza con el caracter "&", sigue con el nombre de la entidad y termina con ";". (es decir, &nombreEntidad;)

Las entidades pueden ser internas o externas y tanto unas como otras pueden ser generales o paramétricas.

Las declaraciones de entidades internas (generales) siguen la siguiente sintaxis:

```
<!ENTITY nombreEntidad "valorEntidad">
```

En las declaraciones de entidades externas (generales) se distinguen dos casos:

- La entidad hace referencia a un fichero de texto y en ese caso la entidad se sustituye por el contenido del archivo.

La entidad puede ser una entidad de sistema, con la siguiente sintaxis:

```
<!ENTITY nombreEntidad SYSTEM "uri">
```

o puede ser una entidad pública, con la siguiente sintaxis:

```
<!ENTITY nombreEntidad PUBLIC "fpi" "uri">
```

- La entidad hace referencia a un fichero que no es de texto (por ejemplo, una imagen) y en ese caso la entidad no se sustituye por el contenido del archivo.

La entidad puede ser una entidad de sistema, con la siguiente sintaxis:

```
<!ENTITY nombreEntidad SYSTEM "uri" NDATA tipo>
```

o puede ser una entidad pública, con la siguiente sintaxis:

```
<!ENTITY nombreEntidad PUBLIC "fpi" "uri" NDATA tipo>
```

En todos estos casos:

- "nombreEntidad" es el nombre de la entidad.
- "valorEntidad" es el valor de la entidad.
- "uri" es el camino (absoluto o relativo) hasta un archivo.
- "tipo" es el tipo de archivo (gif, jpg, etc).
- "fpi" es un indentificador público formal (Formal Public Identifier).

Las declaraciones de entidades paramétricas siguen la mismas sintaxis que las generales, pero llevan el caracter "%" antes del nombre de la entidad. Por ejemplo:

```
<!ENTITY % nombreEntidad "valorEntidad">  
<!ENTITY % nombreEntidad SYSTEM "uri">  
<!ENTITY % nombreEntidad SYSTEM "uri" NDATA tipo>
```

La diferencia entre entidades generales y paramétricas es que las entidades paramétricas se sustituyen por su valor en todo el documento (incluso en la propia declaración de tipo de documento) mientras que las generales no se sustituyen en la declaración de tipo de documento.

Declaración de notaciones

Las notaciones se usan en XML para definir las entidades externas que no va a analizar el procesador XML (aunque sí lo hará la aplicación que trate un documento). Para hacer referencia estas entidades no se utiliza la notación habitual (&nombreEntidad;), sino que se utiliza el nombre de la entidad directamente.
