

1.- Documentos XML.

Hasta ahora hemos trabajado con documentos básicos XML. Sólo declarábamos el tipo de documento que iba a ser (el ejemplar) pero no definíamos que cualidades tenía ese tipo de documento.

Ya vimos que un documento XML básico estaba formado por un **prólogo** y por un **ejemplar**.

Prólogo

Informa al intérprete encargado de procesar el documento de todos aquellos datos que necesita para realizar su trabajo. Consta de dos partes:

- **Definición de XML.** Indicamos la versión de XML, la codificación y la autonomía. Hasta ahora todos los generados han sido independientes.

<?xml version="1.0" encoding="utf-8" standalone="yes" ?>

- **Declaración del tipo de documento.** Hasta ahora sólo indicábamos
<!DOCTYPE nombre_del_ejemplar>

<!DOCTYPE biblioteca>

Ejemplar

Contiene los datos del documento que se quiere procesar. **Es el elemento raíz y debe ser único.**

Está compuesto de elementos estructurados según una estructura de árbol. Los elementos pueden contener atributos.

```
<biblioteca>
  <libro>
    <titulo edición="2" pagina="540">La búsqueda</titulo>
    <autor>Sebastián Torres</autor>
    <editorial>Ediciones Plum</editorial>
    <isbn>978-2-7460-4344-1</isbn>
  </libro>
</biblioteca>
```

Documentos bien formados.

Son sintácticamente correctos. Cumplen las reglas de sintaxis del lenguaje.

Documentos válidos.

Además de bien formados cumplen los requisitos de la definición de estructura que se haya indicado en la definición del documento.

Es decir que el ejemplar debe ceñirse a un formato y estructura definido previamente. **Necesitamos entonces especificar este formato de comunicación en XML.**

Lenguaje de marcas XML

Si cumple la sintaxis decimos **está bien formado**.

Para definir la estructura de datos podemos usar DTD o XML Schema

Si el documento XML cumple lo especificado en el DTD o XML Schema decimos que **es válido**.

2.- Declaración de tipos de documentos XML.

El DTD (*Document Type Definition*) establece que elementos son aceptados y en qué posición deben estar dentro de un documento XML.

La declaración de una DTD puede establecerse de dos formas.

1. **Interna.** La DTD se ubica dentro del propio documento XML.
2. **Externa.** La DTD se ubica en un archivo externo al documento XML.

Para validar más de un documento XML con la misma DTD debemos usar la *externa* para no tener que repetir la DTD dentro de 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*.

Interna. Para referenciarlo como DTD interno, el atributo *standalone* en la declaración XML se debe marcar con un **Sí**. Esto significa que la declaración funciona al margen de la fuente externa.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
```

```
<!DOCTYPE autor[
```

```
  <!ELEMENT autor (nombre, empresa, telefono)>
```

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

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

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

```
<!--DOCTYPE root-element [element-declarations]-->
```



```
<autor>
```

```
  <nombre>Alejandro Romero</nombre>
```

```
  <empresa>Software Alcuza</empresa>
```

```
  <telefono>954119023</telefono>
```

```
</autor>
```

01interna.xml

Comprobación en XML Copy Editor

Bien formado



Válido

Externa

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
```

```
<!DOCTYPE autor SYSTEM "02autor.dtd">
```

```
<autor>
```

```
  <nombre>Alejandro Romero</nombre>
```

```
  <empresa>Software Alcuza</empresa>
```

```
  <telefono>954119023</telefono>
```

```
</autor>
```

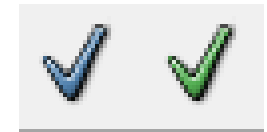
<!DOCTYPE root-element SYSTEM "file-name">

```
<!ELEMENT autor (nombre, empresa, telefono)>
```

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

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

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



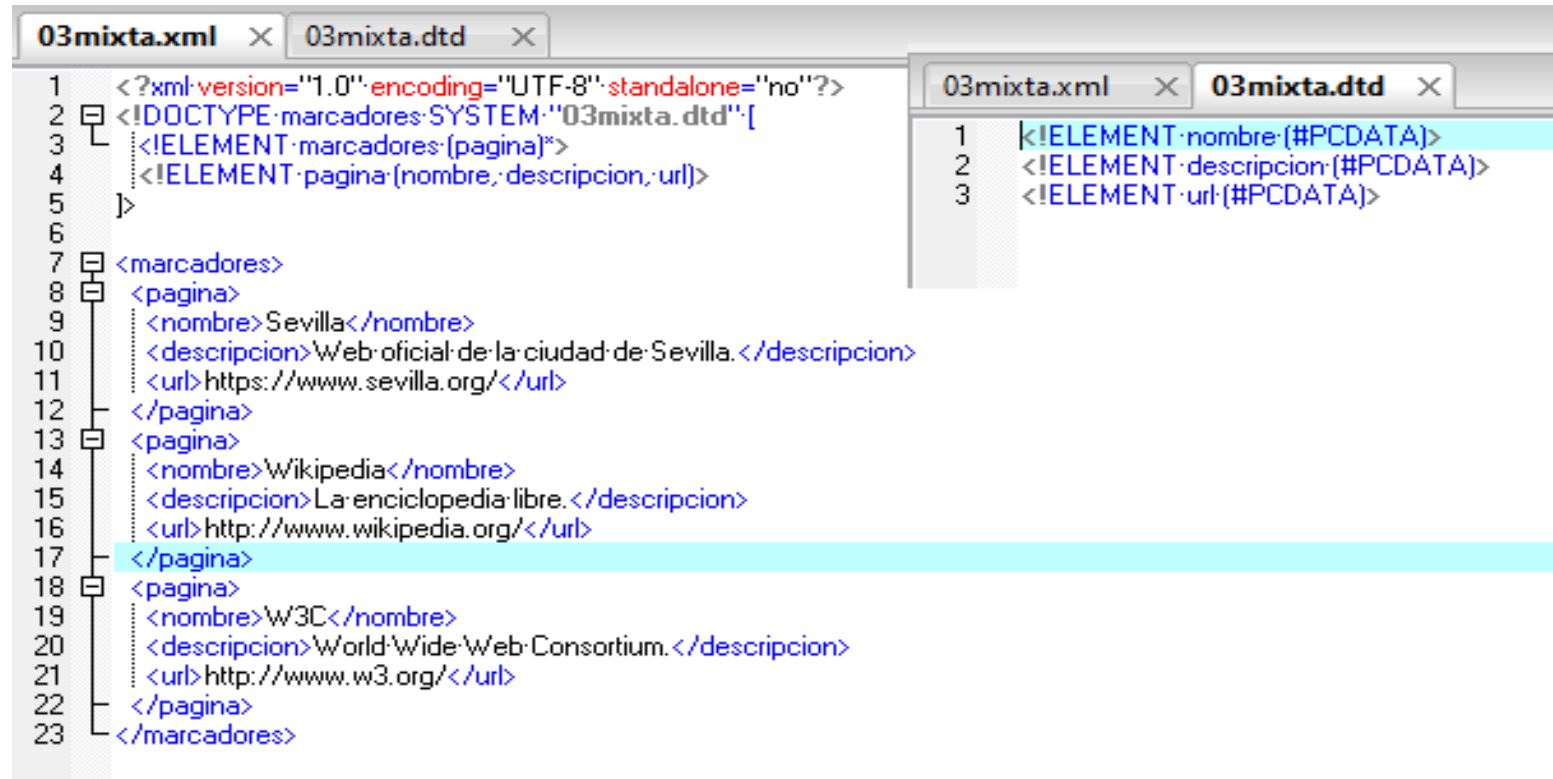
02externa.xml

02autor.dtd

.dtd no es un fichero XML

Combinando Interna / Externa

Existe la posibilidad de combinar ambas propuestas. Se concatenarían las líneas de código. Si una regla aparece en ambos lugares, las internas tienen prioridad sobre las externa.



The screenshot shows two windows of an XML editor. The left window, titled '03mixta.xml', contains the following XML code:

```
1 <?xml:version="1.0" encoding="UTF-8" standalone="no"?>
2 <!DOCTYPE marcadores SYSTEM "03mixta.dtd"[
3   <!ELEMENT marcadores (pagina)*>
4   <!ELEMENT pagina (nombre, descripcion, url)>
5 ]>
6
7 <marcadores>
8   <pagina>
9     <nombre>Sevilla</nombre>
10    <descripcion>Web oficial de la ciudad de Sevilla.</descripcion>
11    <url>https://www.sevilla.org/</url>
12  </pagina>
13  <pagina>
14    <nombre>Wikipedia</nombre>
15    <descripcion>La enciclopedia libre.</descripcion>
16    <url>http://www.wikipedia.org/</url>
17  </pagina>
18  <pagina>
19    <nombre>W3C</nombre>
20    <descripcion>World Wide Web Consortium.</descripcion>
21    <url>http://www.w3.org/</url>
22  </pagina>
23 </marcadores>
```

The right window, titled '03mixta.dtd', contains the following DTD code:

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

03mixta.xml

03mixta.dtd

Ejemplo. Documento XML que guarda *sms* (cero o más de uno)

04EjemploSMS.xml

04BSsms.dtd

<!ELEMENT BDsms (sms)>*

Ejemplo. Creación de un DTD y documento XML que recoja información (nombre y dirección) de diferentes alumnos de una clase.

05Ejemplo.xml

05alumno.dtd

Los bloques de un documento DTD:

- **Entidades.** Son caracteres especiales que permiten incluir datos que podrían confundirse con entidades propias del lenguaje de marcado. También se emplea para predefinir valores (constantes). **!ENTITY**
- **Elemento.** Es el bloque principal con el que se desarrollan los documentos. **!ELEMENT**
- **Atributos.** Es la manera de añadir más información a un elemento. **!ATTLIST**

Entidades (**!ENTITY**)

Espacio en blanco	
>	>
<	<
"	"
'	'
&	&

Para caracteres
especiales

06entidadesUTF.xml

06entidadesISO.xml

Entidades generales internas analizables

<!ENTITY pi "3,141592">	π
<!ENTITY alf "Alien Life Form">	&alf;

Para declarar constantes

Una **sección CDATA** es una etiqueta que comienza por <![CDATA[y termina por]]> y cuyo contenido **el procesador XML no interpreta como marcas sino como texto**. Es una alternativa en los textos con muchas entidades como < y & que dificultan la lectura del documento

Entidades (***!ENTITY***)

Uso de una entidad dentro de otra. En la DTD se han declarado dos entidades generales internas analizables (color y frase). La primera aparece como parte del valor de la segunda.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<!DOCTYPE frase [
  <!ELEMENT frase (#PCDATA)>
  <!ENTITY color "azul">
  <!ENTITY frase "El cielo es &color;.">
]>
<frase>&frase;
</frase>
```

06entidadesanidadas.xml

Entidades (**!ENTITY**)

Entidades paramétricas internas analizables

Nos permite dar nombres a partes de un DTD y hacer referencia a ellas a lo largo del mismo como una constante. Útiles cuando varios elementos del DTD comparten listas de atributos o especificaciones de contenidos. Se denotan por **%entidad**;

```
<!ENTITY % direccion "calle, ciudad, cp">
```

```
<!ELEMENT beca (alumno, ies)>
```

```
<!ELEMENT alumno (dni, %direccion;)>
```

```
<!ELEMENT ies (nombre, %direccion;)>
```

← Defino "dirección"

Uso "dirección"

06Entidadesparametricas.xml

06Entidadesparametricas.dtd


Elementos (**!ELEMENT**)

Elementos NO Terminales. Definimos elementos que están compuestos por otros elementos hijos.

<!ELEMENT A (B, C)>, el elemento A está formado por un elemento B seguido de uno C

<!ELEMENT clase (alumno)>

<!ELEMENT alumno (nombre, dirección)>

Operadores.	Opcional ?	<i><!ELEMENT teléfono (trabajo?, casa)></i>
	Uno o más +	<i><!ELEMENT provincia (nombre, (cp, ciudad)+)></i>
	Cero o más *	<i><!ELEMENT provincia (nombre, (cp, ciudad)*)></i> <i><!ELEMENT clase (alumno*)></i>
Lista de opciones  Elección		<i><!ELEMENT alumno (nombre, (teléfono correo))></i>

Ejemplo. Operadores.

```
<!ELEMENT clase (profesor+, alumno*)>
  <!ELEMENT profesor (#PCDATA)>
  <!ELEMENT alumno (nombre, (apellido1, apellido2?)?, dirección, correo,
teléfono?, (nif | nie), materias)>
    <!ELEMENT nombre (#PCDATA)>
      <!ELEMENT apellido1 (#PCDATA)>
      <!ELEMENT apellido2 (#PCDATA)>
    <!ELEMENT dirección (#PCDATA)>
    <!ELEMENT correo (#PCDATA)>
    <!ELEMENT teléfono (#PCDATA)>
    <!ELEMENT nif (#PCDATA)>
    <!ELEMENT nie (#PCDATA)>
    <!ELEMENT materias (asignatura+)>
      <!ELEMENT asignatura (#PCDATA)>
```

07operadores.xml

07operadores.dtd

Elementos (!ELEMENT)

Elementos Terminales. Se corresponden con hojas de la estructura del árbol formada por los datos del documento XML.

`<!ELEMENT nombre_del_elemento declaración_del_contenido>`

La declaración de contenido puede tener uno de los siguientes valores:

EMPTY. El elemento debe estar vacío. Por ejemplo

`<!ELEMENT jornadaCompleta EMPTY>`

ANY. Admite como elemento cualquiera de los elementos declarados.

No suele usarse salvo para pruebas

(#PCDATA). Almacena texto entre la etiqueta de apertura y la de cierre

Parser Carácter Data

Elementos (!ELEMENT)

En una lista de opciones se permite contenido mixto:

<!ELEMENT elemento1 (#PCDATA | elemento2)>*

08CDATAmixto.xml

08CDATAmixto.dtd

No se recomienda porque pierde rigidez la estructura.

Elementos (!ELEMENT)

09ediciones.xml

Empleando sólo !ELEMENT



09ediciones.dtd

Ediciones Informe regional del ventas Descripción: informe de ventas para las regiones Norte, Centro y Sur Fecha del informe: 30/12/2009		
Region	Trimestre	Libros Vendidos
Norte	1	24000
	2	38600
	3	NO_INFO
	4	NO_INFO
Centro	1	NO_INFO
	2	16080
	3	25000
	4	29000
Sur	1	27000
	2	31400
	3	40100
	4	30000

Atributos (!ATTLIST)

<!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 **país** **CDATA** **"España"**>

<!ELEMENT tenis (#PCDATA)>

10Atributos1.xml

Para el elemento F1, país es un atributo definido de tipo CDATA (*Character DATA*), es decir, su valor será una cadena de caracteres.

El valor por defecto es "España" si no se especifica.

<deportistas>

<F1 país="Alemania">Sebastian Vettel</F1>

<F1>Fernando Alonso</F1>

<tenis>Rafael Nadal</tenis>

</deportistas>

<deportistas>
 <F1 país="Alemania">Sebastian Vettel</F1>
 <F1 país="España">Fernando Alonso</F1>
 <tenis>Rafael Nadal</tenis>
 </deportistas>

Atributos (!ATTLIST)

Declaración de varios atributos en un elemento

```
<!ELEMENT deportistas (futbol | F1 | tenis)*>
<!ELEMENT futbol (#PCDATA)>
<!ELEMENT F1 (#PCDATA)>
  <!ATTLIST F1 pais CDATA #FIXED "España">
  <!ATTLIST F1 fecha_de_nacimiento CDATA #IMPLIED>
  <!ATTLIST F1 equipo CDATA #REQUIRED>
<!ELEMENT tenis (#PCDATA)>
```

10Atributos2.xml

El atributo equipo es obligatorio escribirlo, **#REQUIRED**.


El atributo fecha_de_nacimiento es opcional, **#IMPLIED**.

Por **#FIXED** todos los elementos F1 que aparezcan tendrán el atributo país con el valor "España"

```
<deportistas>
  <F1 fecha_de_nacimiento="03/07/1987" equipo="Ferrari">Sebastian
Vettel</F1>
  <F1 equipo="McLaren">Fernando Alonso</F1>
  <tenis>Rafael Nadal</tenis>
</deportistas>
```

```
<deportistas>
  <F1 fecha_de_nacimiento="03/07/1987" equipo="Ferrari" pais="España">Sebastian Vettel</F1>
  <F1 equipo="McLaren" pais="España">Fernando Alonso</F1>
  <tenis>Rafael Nadal</tenis>
</deportistas>
```

Tipos de atributos en DTD

	Tipo	Descripción
Enumerado 	CDATA	(<i>Character DATA</i>) El valor son datos de tipo carácter, es decir, texto.
	(a b c)	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: <i>xml:lang</i> y <i>xml:space</i> .

Atributos (!ATTLIST)

Declaración de varios atributos en un elemento

<!ATTLIST ciudad país **CDATA** #REQUIRED>

El valor del atributo país puede estar vacío.

10Atributos3.xml

<!ATTLIST F1 país (ESP | FRA | ITA | ALE) "ESP">

Enumerados. Se puede añadir uno por defecto y no sería necesario ponerlo.

10Atributos4.xml

<!ATTLIST F1 país (ESP | FRA | ITA | ALE) #REQUIRED>

Enumerados. Obligatorio. Hay que especificarlo.

<!ATTLIST F1 código **ID** #REQUIRED>

Cada elemento escrito en un documento XML sólo puede tener un atributo ID.
En un documento XML, no pueden escribirse dos elementos que tengan el mismo valor en un atributo ID, aunque dicho atributo sea distinto
Todo atributo declarado de tipo ID tiene que ser #IMPLIED (opcional) o #REQUIRED (obligatorio).

Atributos (!ATTLIST)

```
<!ATTLIST director coddir ID #REQUIRED>
```

```
<!ATTLIST película dirección IDREF>
```

Los elementos película que se escriban deben incluir el atributo dirección cuyo valor estará asignado a un atributo ID de otro elemento del documento. En este caso, el valor estará asignado a un atributo coddir de un elemento director.

10Atributos5.xml

10Atributos6.xml

```
<!ATTLIST pelicula codpel ID>
```

```
<!ATTLIST director filmografia IDREFS>
```

Similar a IDREF pero puede incluirse una lista de valores de atributos ID

Atributos (!ATTLIST)

Declaración de varios atributos en un elemento

```
<!ATTLIST pelicula IDpelicula ID #REQUIRED>  
<!ATTLIST pelicula valoracion CDATA "Pendiente">
```

Declaramos los atributos bajo
un solo !ATTLIST



11AtributosCompactados.xml

```
<!ATTLIST pelicula  
IDpelicula ID #REQUIRED  
valoracion CDATA "Pendiente">
```

Secciones condicionales

En DTD externas se pueden definir las secciones **IGNORE** e **INCLUDE**, para ignorar o incluir declaraciones. Tiene preferencia el IGNORE frente al INCLUDE

<!-- Mensaje largo -->

```
<![ INCLUDE [  
<!ELEMENT mensaje (titulo, emisor, receptor, contenido, palabras)>  
<!ELEMENT titulo (#PCDATA)>  
<!ELEMENT palabras (#PCDATA)>  
]]>
```

<!-- Mensaje corto -->

```
<![ IGNORE [  
<!ELEMENT mensaje (emisor, receptor, contenido)>  
]]>
```

<!-- Declaración de elementos comunes -->

```
<!ELEMENT emisor (#PCDATA)>  
<!ELEMENT receptor (#PCDATA)>  
<!ELEMENT contenido (#PCDATA)>
```

12seccionescondicionales.xml

12seccionescondicionales.dtd

Secciones condicionales

Pueden incluirse entidades constantes para cambiar de una opción a otra en el propio documento

```
<!ENTITY % largo "INCLUDE">
```

```
<!ENTITY % corto "IGNORE">
```

En el DTD haría referencia a

%largo;

%corto;

12bseccionescondicionales.xml

12bseccionescondicionales.dtd

Entidades NDATA

Entidades generales externas no analizables

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
```

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

Si es una URL pública se emplearía

PUBLIC "-//W3C//GIF logo//EN"

"http://www.abrirllave.com/dtd/logo.gif"

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

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


```

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

Con NDATA (*Notation Data*) se indica que la entidad no es analizable. Caso de archivos con formatos binarios.

Indicamos que el elemento "imagen" que se incluya tiene que incluir obligatoriamente el atributo fuente cuyo valor será una entidad.

La notación gif es una declaración del tipo *MIME image/gif*

13entidadNDATA.xml

COMENTARIOS

En una DTD se pueden escribir comentarios como en XML.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
```

```
<!DOCTYPE ciudades [<!-- Ejemplo de documento XML con comentarios en su DTD
interna-->
```

```
<!ELEMENT ciudades (ciudad*)>
```

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

```
<!-- país es atributo del elemento ciudad -->
```

```
<!ATTLIST ciudad pais CDATA #REQUIRED>
```



```
<ciudades>
```

```
<ciudad pais="Italia">Roma</ciudad> <!-- comentario -->
```

```
<ciudad pais="Francia">París</ciudad>
```

```
<ciudad pais="">Viena</ciudad>
```

```
</ciudades>
```

También en los DTD externos.

14Comentarios.xml

4.- Herramientas.

Estas son algunas de las herramientas que facilitan el trabajar con XML, DTD y Schema XML.

- **Editor, generador y validador XML COPY EDITOR**
- Editor <https://www.oxygenxml.com/> OXYGEN. Es de pago con período de 30 días de prueba.
- Extensión XML (Red Hat) para Visual Studio Code
- Validador Esquema <https://es.rakko.tools/tools/51/>

XML COPY EDITOR.

Algunas de las utilidades interesantes que aporta XML COPY como editor son:

- Al crear un documento nuevo nos facilita las líneas básicas (sintaxis) del documento dependiendo de la extensión: .xml, .dtd, .xsd, .xsl (transformación).
- Ver -> Wrap Words (para ajustar las líneas al tamaño de la pantalla, el ALT-Z de VSCode).
- Ver -> Esquema de color / Ver -> Tamaño de texto
- Ver -> Ocultar Atributos solo / Ver -> Etiquetas y Atributos
- XML -> Bloquear Etiquetas
- XML -> Formatear el fuente. Estructura el documento con tabulaciones.

XML COPY EDITOR.

Utilidades interesantes de XML COPY vinculadas:

- Editar -> Toggle Comment para des/convertir una línea en comentario.
- XML -> **Create Schema**, nos ofrecerá la posibilidad de **crear una estructura de datos DTD o Schema XML del documento XML activo.**
- XML -> **DTD → Schema** crea el Schema XML equivalente a un archivo **DTD dado.**
- XML -> Asociar (DTD, Schema XML, estilos). **Incluye la línea correspondiente asociando el fichero externo elegido.**
- XML -> Encoding