3. Documentos XML

> XML son las siglas de e**X**tensible **M**arkup **L**anguage (lenguaje de etiquetado extensible). Es un lenguaje estándar, una recomendación del World Wide Web Consortium (W3C, www.W3.Org/TR/REC-xml). Está basado en el estándar ISO SGML.

El XML está tan basado en SGML que cualquier documento XML es un tipo de documento SGML correcto (aunque al revés no pasa).

El XML surgió para intentar superar los problemas que tenía el HTML a la hora de procesar automáticamente la información que contienen las páginas web manteniendo la forma de funcionar que utiliza el HTML. Además, se añadieron otros requisitos, como que:

- Fuera fácil crear documentos XML y que los humanos pudieran leer y entenderlo fácilmente.
- No sea complicado hacer programas de ordenador que trabajaran con XML.
- El XML se pudiera utilizar en el máximo posible de campos de aplicación y mantuviera la compatibilidad con SGML.

El resultado ha sido que XML:

- **Define la sintaxis genérica** para marcar los datos con valores comprensibles para los humanos.
- Es una manera de **dar formato** a los documentos siendo lo suficientemente **flexible** para ser personalizada en diferentes destinos: web, impresoras, bases de datos, etc.
- Está pensado para que **todos lo puedan utilizar** cualquiera que sea su área de interés.

3.1. Metalenguaje

En realidad, XML no es un lenguaje de marcas, sino que es un lenguaje que nos permitirá definir nuestros lenguajes de marcas propios. Esto quiere decir que podremos definir lenguajes de marcas específicos para cada uno de los campos de interés. Si trabajamos en el mundo de los gráficos podemos definir nuestro lenguaje específico para definir estos gráficos, y si trabajamos en el mundo de la prensa podremos definir nuestro lenguaje para representar las noticias.

Por este motivo a menudo se dice que **XML es un metalenguaje**, ya que nos permite definir la estructura y el vocabulario de otros lenguajes de marcas.

```
El XML es un lenguaje para crear lenguajes de marcas.
```

Esta libertad a la hora de definir las marcas que nos interesen, es uno de los puntos fuertes de XML, haciéndolo mucho más potente y adaptable a las diferentes complejidades de los entornos en los que pueda ser necesario.

3.2. Elementos

La base de XML son los elementos. Un elemento normalmente estará formado por la *apertura de una etiqueta* (con atributos o sin), un *contenido* (que también puede ser un grupo de etiquetas), y el *cierre de la etiqueta*.

En este ejemplo podemos ver lo que acabamos de comentar. Definimos una etiqueta (nombre), luego el contenido (Pedro Martín) y posteriormente cerramos la etiqueta:

```
<nombre > Pedro Martín </ nombre >
```

Este es uno de los puntos fuertes del XML: queda bastante claro que el contenido de dentro de la etiqueta es un nombre.

Se podría hacer el ejemplo algo más complejo añadiendo un atributo.

Los atributos se añaden siempre en la apertura de la etiqueta.

```
<nombre  cargo = "director" > Pedro Martín </ nombre >
```

Otro de los aspectos básicos dl XML es que no se preocupa de la presentación; parte de la idea de que lo importante es el contenido de los datos y no la forma en que se visualizarán. Esta característica lo hace ideal para presentar la información y posteriormente por medio de algún proceso convertir la información en un formato de presentación específico como HTML, PDF, PostScript, etc. Además, aporta una característica muy interesante desde el punto de vista de la informática: permite tener los datos separados de la manera de representarlos.

XML permite separar el contenido y la forma en que se presentará ese contenido a los usuarios.

1-Etiquetas

Las etiquetas se definen dentro del documento XML; se define un formato para separarlas claramente del contenido de datos. Estrictamente hablando hay dos tipos de etiquetas:

- **Las etiquetas de apertura**
- **Las etiquetas de cierre**

La información se distribuye entre los dos tipos de etiquetas. Así se consigue una manera sencilla de definir qué partes del documento son datos y cuáles son estructura.

Las etiquetas de apertura se definen con los símbolos de menor "**<**" y mayor "**>**" con un nombre de etiqueta en medio.

```
<etiqueta >
```

Y las etiquetas de cierre se definen igual que las de apertura, pero a la hora de empezar la etiqueta se especifican dos símbolos: un símbolo de más pequeño y una barra "**</**". De

esta manera se pueden distinguir fácilmente los dos tipos de etiquetas y queda claro en qué lugar se debe añadir el contenido.

```
<etiqueta > Contenido </ etiqueta >
```

XML no define ninguna etiqueta, sino que permite que las defina cada uno en función de lo que necesite representar. Esto hace más fácil crear documentos XML, ya que no es necesario conocer ninguna etiqueta para poder empezar a trabajar y porque permite adaptarse a cualquier tarea.

A la hora de definir las etiquetas se puede dejar un espacio o un salto de línea detrás del nombre de la etiqueta, pero nunca antes ni en medio. Por lo tanto, estas etiquetas serían correctas:

```
<etiqueta1 >
<etiqueta >
</etiqueta >
</etiqueta1 >
</Etiqueta2
>
```

Y en cambio estas otras no serían correctas:

```
< Etiqueta>
<etiqueta 1 >
</ Etiqueta2 >
```

Como se acaba de ver, la especificación XML define claramente cómo se deben crear las etiquetas en los documentos XML, pero en cambio no define ningún tipo ni significado asociado a ninguna de las etiquetas.
Por ejemplo, si usamos la etiqueta <b> de HTML, se nos está indicando que el contenido de la etiqueta deberá representarse en negrita. Esto obliga a que para hacer documentos XML se deban conocer las etiquetas.

Para XML las etiquetas sólo son una manera de separar el contenido y definir la estructura de los datos que contiene. La interpretación de qué significan los datos y cómo se deben representar se deja a quien lea el documento (independientemente de que éste sea un humano o un programa).

> El tratamiento de excepciones para interpretar HTML es una de las cosas que complica la tarea de programar los navegadores web actuales.

A pesar de la libertad a la hora de dejar que los usuarios elijan sus propias etiquetas, el XML es mucho más riguroso que otros lenguajes a la hora de definir cómo se deben escribir. El objetivo de tener reglas de escritura está determinado por la necesidad de que los documentos XML, en un momento u otro, serán procesados por un programa, y los programas se hacen más complejos a la hora de tratar con excepciones. Por lo tanto, una de las cosas que hace XML es **evitar las excepciones** en la medida de lo posible.

> Las etiquetas que se abran siempre se tendrán que cerrar.

Para evitar las excepciones, una de las reglas básicas de XML es que cualquier etiqueta que se abra siempre debe cerrarse. El siguiente ejemplo no sería un documento XML válido:

```
<nombre > Pedro García
```

Y en cambio este sí:

```
<nombre > Pedro García </ nombre >
```

A pesar de que no hay etiquetas definidas, por un motivo de legibilidad y de interpretación de los datos se recomienda que las etiquetas sean auto explicativas y pronunciables. Esto es para evitar cosas como la del ejemplo siguiente:

```
<PCCC > Figueres </ PCCC >
```

Según este segundo ejemplo, ¿cómo determinamos qué es “Figueres”? Para un programa de ordenador seguramente no sería un problema demasiado grande;

simplemente cogería el dato y lo interpretaría como se haya programado. Para un humano un documento con etiquetas como ésta sería imposible de interpretar.

> Se recomienda elegir los nombres de las etiquetas de modo que puedan ser interpretadas por quien las leerá independientemente de si las procesa una persona o un programa.

El ejemplo anterior será más fácil de interpretar si cambiamos la etiqueta:

```
<ciudad > Figueres </ ciudad >
```

2-Contenido

1. El contenido de un elemento será **todo lo que haya entre las etiquetas de apertura y de cierre**. En el contenido puede ser simplemente texto:

```
<persona> Pedro Martín </persona>
```

2. El contenido pueden ser también otros elementos. En el ejemplo siguiente el elemento <persona>contiene dos elementos más: <nom> y <cognom>, que además tienen contenido en su interior:

```
<persona >
   <nombre> Pedro </nombre>
   <apellido> Martín </apellido>
</persona>
```

3. Una tercera posibilidad sería combinar los dos casos anteriores y que el contenido sea una mezcla de texto y elementos.

```
<persona>
   Pedro Martín
   <cargo> director </cargo>
```

```
</persona>
```

También es posible definir etiquetas sin contenido de modo que su significado está determinado por el nombre de la etiqueta.

```
<persona >
</persona>
```

Los elementos huecos también se pueden definir usando el símbolo /> en cerrar la etiqueta. De modo que no hay ninguna diferencia entre definir <director> de esta manera:

```
<director/>
```

o de ésta:

```
<director> </ director>
```

Los elementos vacíos, aunque se definan con su formato especial, deben seguir las mismas normas que los elementos normales. Por lo tanto, serían correctas las siguientes definiciones:

```
<director  />
<director
/>
```

Y no lo serían estas:

```
<Director />
<director / >
```

XML no define ninguna restricción a la hora de definir el contenido de los elementos. Por lo tanto:

- Podemos utilizar cualquier carácter representable usando el código de caracteres.
- El contenido puede ser tan largo como nos haga falta.
- Se puede escribir en cualquier idioma del mundo.
- No importa que haya espacios en blanco o saltos de línea dentro del contenido.

- **Entidades**

Lo único que se debe tener en cuenta a la hora de definir contenidos, es si hay una serie

de caracteres que pueden provocar confusiones a la hora de interpretar el documento XML, y por tanto se declaran como *ilegales*. No hay que olvidar que uno de los objetivos de la especificación es "que sea fácil escribir programas que procesen los documentos", y por lo tanto se deben evitar las interpretaciones.

Si pasamos el código siguiente en un programa de ordenador tendremos un problema:

```
<algebra >
x <ye y>z
</algebra >
```

Para un humano esto no sería un problema porque puede interpretar los datos por medio de su conocimiento, pero en cambio un programa al leer el contenido del documento interpretaría que <yey> es el comienzo de una nueva etiqueta.

Para evitar este problema se han definido una serie de valores, llamados **entidades**, que se utilizan para poder incluir los caracteres problemáticos dentro del contenido de las etiquetas (Tabla 3.1).

Referencias de caracteres
En realidad, cualquier carácter se puede representar como una entidad utilizando su valor numérico en Unicode, utilizando los símbolos &# y los tres dígitos de su valor numérico.
Por ejemplo: &#169

Tabla 3.1. Entidades definidas en XML

símbolo	sustitución
<	& Lt;
>	& Gt;
"	& Quot;
'	& Apos;
&	& Amp;

- **Secciones CDATA**

Puede darse el caso de que el contenido de un elemento sea HTML o código en algún lenguaje de programación. Esto obligaría a sustituir por entidades una gran cantidad de símbolos y además quedaría la legibilidad al documento. Para evitarlo XML permite utilizar las secciones CDATA dentro del contenido de un elemento. Todo lo que esté dentro de una sección CDATA no será interpretado por ningún programa.

Las secciones CDATA funcionan como las etiquetas normales. Se definen empezando por los caracteres <! [CDATA [antes de especificar el contenido, y terminan con la combinación de caracteres]] >. Por lo tanto, podemos definir contenido con símbolos prohibidos dentro de las secciones CDATA sin problemas.

```
<valor>
   <! [CDATA [
      if (x <y and y> z)
      {
         printf ( "Correcto!");
      }
   ]]>
</ valor >
```

Además, se puede ver fácilmente que para un humano es más difícil interpretar esto:

```
<titulo > Los documentos HTML empiezan por & lt; html & gt; </titulo >
```

Que su equivalente con CDATA:

```
<titulo >
 <! [CDATA [
   Los documentos HTML empiezan por <HTML>
 ]]>
</titulo >
```

Otro problema que solucionan las secciones CDATA es que permiten añadir contenido en un lenguaje de marcas que no es necesario que esté bien formado. Por ejemplo, si el contenido tiene etiquetas que no se cierran sólo se pueden definir en una sección CDATA o bien el programa las interpretará como etiquetas del documento y dará un error.

Las secciones CDATA normalmente se utilizan para:

- Definir grandes fragmentos de texto que requieran muchas sustituciones de entidades.
- Si se incluye contenido en HTML, Javascript o algún lenguaje similar. La sustitución con entidades le resta legibilidad.
- Si el contenido está en un lenguaje de marcas y no está bien formado.

3-Atributos

Una manera alternativa de añadir contenido a los documentos XML es por medio de los atributos. Los atributos son un par de valores separados por un =que sólo se especifican en las etiquetas de apertura.

El primer valor del par nos indica cuál es el nombre del contenido y se utilizará para interpretar qué indican los datos que tiene asociadas, mientras que el segundo nos definirá el contenido del atributo.

```
<nombre  cargo = "profesor" > Federico García </nombre >
```

Se puede ver cómo el atributo cargo da un nivel más de información a nombre. Ahora sabemos que hace referencia al nombre de un profesor.

Para especificar los atributos se pueden utilizar tanto las comillas dobles como las comillas simples. Eso sí, siempre hay que cerrar las comillas tal como se han abierto.

<corredor posicion=“1”>

o <corredor posicion=‘1’>

pero nunca <corredor posicion=“1’>

A diferencia de lo que ocurre en otros lenguajes de marcas como el HTML, XML es mucho más estricto y obliga a que todos los valores de los atributos estén rodeados por comillas. En otros lenguajes de marcas se permite que los valores numéricos no vayan entre comillas, pero en XML no se puede hacer. Por lo tanto, el ejemplo siguiente sería incorrecto porque el atributo no tiene comillas.

```
<corredor  posicion = 1 > Manuel Roure </corredor >
```

Si se quiere especificar debe ponerse entre comillas simples:

```
<corredor  posicion = '1' > Manuel Ruiz </corredor >
```

o dobles:

```
<corredor  posicion = "1" > Manuel Ruiz </corredor >
```

Cuando el atributo no necesite especificar un valor porque se considera que se puede interpretar su significado por el nombre del atributo, es obligatorio incluir las comillas. Por lo tanto, el ejemplo siguiente no sería correcto porque el atributo no tiene valor:

```
<alumno delegado > Jaime Ramírez </alumno >
```

Lo tendríamos que especificar con la cadena vacía:

```
<alumno  delegado = "" > Jaime Ramírez </alumno >
```

O poniendo algún valor:

```
<alumno  delegado = "si" > Jaime Ramírez </alumno >
```

XML permite definir tantos atributos como sea necesario sin ningún tipo de restricción. Las únicas condiciones que debemos seguir es que cada uno de los atributos debe estar separado de los otros al menos por un espacio.

```
<persona  nombre = "Javier"  apellido = "Sala" />
```

El orden en que aparecen los atributos no tiene ninguna importancia, por lo que los dos códigos siguientes serían equivalentes:

```
<persona  nombre = "Javier"  apellido = "Sala"  />
<persona  apellido = "Sala"  nombre = "Javier"  />
```

¿Cuándo se utilizarán atributos y cuándo elementos?
Ha habido muchos debates sobre cuándo deben utilizarse atributos y cuando se deben utilizar elementos sin que se haya encontrado ningún argumento totalmente aceptado por todos.

Al final el resultado práctico es que es virtualmente el mismo hacer esto:

```
<persona  nombre = "Pedro"  />
```

Que hacer:

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

Y por lo tanto siempre se puede usar el método que nos guste más.

Otra característica importante de los atributos es que no se pueden repetir los nombres de los atributos dentro de un mismo elemento. De modo que no es correcto especificar dos atributos cargo:

```
<político  cargo = "alcalde"  cargo = "diputado" > Juan Ruiz </político>
```

Por lo tanto, se debe buscar otra manera de especificarlo. Por ejemplo, con una lista de valores:

```
<político  cargo = "alcalde diputado" > Juan Ruiz </ político>
```

o bien poniendo nombres de atributos diferentes:

```
<político  alcalde = "si"  diputado = "si" > Juan Ruiz </ político >
```

- **Atributo xml:lang**

El atributo xml:lang es un atributo predefinido y sirve para especificar el idioma que se ha utilizado para escribir el contenido tanto del elemento como de los otros atributos.

El valor del atributo xml:lang puede ser vacío o bien debe estar en una de las formas definidas por el IETF en el BCP 47 (http://www.rfc-editor.org/rfc/bcp/bcp47.txt), que utiliza códigos ISO que definen las regiones y los idiomas. Sin entrar en profundidad esto implica

que podemos definir el código ISO de un idioma o bien una combinación de códigos ISO que definan la región y el idioma.

El atributo xml:lang nos permite definir en qué idioma está el contenido de un elemento o de un documento XML.

```
<mensaje  xml:lang = "es" > Hola </mensaje>
```

También sería válido utilizar cualquiera de las combinaciones de región e idioma:

```
<mensaje  xml: lang = "es-ES" > Hola </mensaje >
<mensaje  xml: lang = "es-AD" > Salud </mensaje >
```

El atributo sólo hace referencia al elemento en que se ha especificado y a su contenido, por lo que aquí:

```
<saludar  xml:lang = "es" >
  <llegar > Hola </ llegar >
  <marchar > Adiós </ marchar >
</ saludar >
```

se puede considerar que los elementos llegar y marchar tienen también el atributo xml:lang porque son contenido <saludar>, y por tanto su contenido y el valor de sus atributos está en español.

Siempre se puede evitar esta forma de herencia del atributo redefiniendo el atributo xml:lang como en este ejemplo:

```
<saludar  xml:lang = "es" >
  <llegar > Hola </ llegar >
  <llegar  xml:lang = "en" > Hello </ llegar >
</ saludar >
```

Se definen dos elementos arriba, uno que tiene el contenido en español, ya que la ha heredado del elemento saludar (que lo contiene), y el otro que tiene el contenido en inglés para lo que se le ha especificado el idioma explícitamente.

El objetivo del atributo xml:lang es permitir que los programas puedan interpretar el idioma en el que está el contenido de los documentos o de los elementos XML. Hay muchos programas que utilizan esta característica para poder hacer que los programas adapten el contenido que deben mostrar en el idioma del que los usa.

Si se tiene un programa que quiere mostrar mensajes por pantalla en el idioma que utilice el usuario se le podría preparar un fichero de datos como el siguiente, que le permitiría hacerlo una vez haya determinado el idioma del sistema operativo:

```
<programa >
  <mensaje  xml:lang = "es" > Bienvenido </ mensaje >
  <mensaje  xml:lang = "fr" > Soyez le Bienvenu </ mensaje >
  <mensaje  xml:lang = "en" > Welcome </ mensaje >
</programa >
```

- **El atributo " xml: space "**

El atributo xml:space hace referencia a cómo se deben tratar los espacios que hay en el contenido de un elemento determinado.
Cuando alguien edita un documento XML es corriente que use espacios y saltos de línea para que el documento sea más "bonito" pero realmente estos espacios y saltos de línea no forman parte del contenido. Con el uso del atributo xml:space se define si estos espacios son parte del contenido o no.

Sólo hay dos valores aceptados para el atributo xml:space: default y preserve (Tabla 3.2).

Tabla 3.2. Valores aceptados para el atributo`xml:space`

valor	significado
default	Implica que el documento debe mostrar sólo un único espacio de separación entre las palabras. Cualquier otra cosa debe ser suprimida.
preserve	Hace que los espacios se dejen tal como están en el documento. Por lo tanto, si hay 5 espacios después de una palabra en el contenido deben preservar estos espacios.

Cualquier otro valor se considerará un error leve y no se tendrá en cuenta.

Siguiendo lo que hemos definido, en el siguiente documento:

```
<mensaje  xml:space = "default" >
Hola:
   ¿Como va todo?
</mensaje>
```

Como el elemento <mensaje> tiene de parámetro xml:space="default", le estamos especificando que se supriman los espacios, o sea, que sería como tener:

```
<mensaje > Hola: ¿Cómo va todo? </mensaje>
```

En cambio, si el atributo hubiera sido preserve el valor del contenido de <mensaje> sería interpretado respetando los espacios y los saltos de línea.

```
<mensaje >
Hola:
   ¿Como va todo?
</mensaje >
```

Los procesadores XML deben pasar todos los espacios que no sean etiquetas al programa que lea el documento. Por tanto, a pesar del nombre *por defecto* se utiliza *preserve*.

4- Nombres válidos XML

El XML permite definir nuestras etiquetas y atributos libremente pero no todas las combinaciones posibles son aceptables. Los nombres de las etiquetas y los atributos deben cumplir unas reglas para ser considerados "correctas".

Aunque los dos puntos están aceptados no se recomienda que se utilicen en la creación de etiquetas para que se reservan para su uso en espacios de nombres.

Las reglas para definir nombres correctos son sencillas:

1. Los nombres deben empezar por una letra del alfabeto, el carácter de subrayado (_) o un guion (-). También se acepta el carácter de dos puntos (:), pero está reservado.
2. Los caracteres en mayúsculas son diferentes de los caracteres en minúsculas.
3. No puede haber espacios en medio del nombre.
4. No pueden empezar por la palabra XML tanto si cualquiera de las letras está en mayúsculas o en minúsculas. Estas palabras se reservan para estandarizaciones futuras.

Siguiendo estas reglas tendremos que el ejemplo siguiente:

```
<pueblo  provincia = "Sevilla" >  Dos hermanas  </Ciudad >
```

La etiqueta <pueblo> es correcta porque cumple todas las reglas. La primera letra del nombre empieza por un carácter del alfabeto y no hay ningún espacio en medio del nombre, y no comienza por las letras xml.
Asimismo, el atributo provincia también es correcto porque comienza con un carácter del alfabeto, tiene espacios y no empieza por xml.

En cambio, no es correcta la etiqueta <La provincia> porque tiene espacios en medio del nombre:

```
<La provincia > </ La provincia >
```

Ni lo serían las etiquetas <1aPosicion> y <2aPosicion>, ya que empiezan por un dígito:

```
<carrera >
  <1aPosicion> Manuel García </1aPosicion >
  <2aPosicion > Pedro  Gil </2aPosicion >
</carrera >
```

En la Tabla 3.3 hay ejemplos de etiquetas correctas e incorrectas.

Tabla 3.3. Ejemplos de etiquetas correctas e incorrectas

etiquetas correctas	etiquetas incorrectas
<Correu1 />	<1Cor />
<Correo electrónico />	<Correo electrónico />
<Elemento />	<Elemento />
<_Carai />	<% Descuento />
<Svg: rectángulo />	<Xml-rectángulo />

A pesar de estas restricciones la libertad que permite el estándar es bastante importante.

> La versión 1.1 de XML permite que sean válidos toda una serie de caracteres extra para adaptarse a las nuevas versiones de Unicode, pero en general no aportan nada si no utilizamos algún idioma asiático.

Una de las cosas que no hay que olvidar nunca es que uno de los objetivos de XML es hacer que los documentos con XML puedan ser leídos y entendidos fácilmente por las personas y, por tanto, se recomienda que no se utilicen nombres que no tengan sentido o que sean difícilmente entendidos por una persona.

5-Comentarios

A menudo en los documentos XML especifican datos extra que realmente no forman parte del documento. Estos datos se denominan comentarios y se usan para muchas cosas diversas, tales como:

- Generar documentación sobre el documento.
- Indicar a la gente que pueda recibir el documento que se quería hacer en crearlo.
- Otros.

Los *analizadores* XML son programas que, a partir de la estructura de los documentos XML, nos permiten acceder al contenido de sus etiquetas.
Lo más habitual suele ser procesar el documento para generar una salida que pueda ser visualizada más fácilmente, etc.

Generalmente los analizadores XML suelen descartar los comentarios, ya que la especificación XML dice que no es necesario que sean procesados.
Los comentarios se especifican incluyéndolos entre los símbolos: **<! -** y **->**; se pueden poner comentarios en cualquier lugar del documento XML excepto dentro de las etiquetas. Como ejemplo, en el documento siguiente se ha incluido el comentario "Lista de alumnos" para indicar en qué lugar comienza la lista de alumnos:

```
<profesores >
  <nombre > Marcelino Álava </nombre >
</profesores>
<!- Lista de alumnos ->
<alumnos >
  <nombre > Fernando García </nombre >
  <nombre > Pedro Gil </nombre >
</alumnos >
```

6-Instrucciones de proceso

Las instrucciones de proceso son una manera de dar instrucciones a los programas que van a leer el documento. Se definen dentro de los símbolos: **<?**; posteriormente siempre se especificará a qué programa van dirigidas, con la única restricción de que no pueden ser las letras "XML" en cualquier combinación de mayúsculas o minúsculas.

Por lo tanto, para añadir información que vaya destinada a un programa llamado php:

```
<? php ... ?>
```

Este sistema permite que en un documento XML se puedan añadir instrucciones de proceso diferentes para programas diferentes:

```
<? Programa1 ... ?>
<? programa2 ... ?>
```

El contenido de los datos puede ser cualquier cosa que tenga sentido para el programa que las leerá, y por tanto no necesariamente tiene que tener sentido para los procesadores XML. Como no forman parte del documento XML en sí pueden aparecer en cualquier lugar del documento excepto en medio de una etiqueta.