Introducción XML.

1. El lenguaje XML.

XML es un metalenguaje extensible de etiquetas que permite **definir la gramática de lenguajes específicos**. Por lo tanto XML no es realmente un lenguaje en particular, sino una manera de definir lenguajes para diferentes necesidades.

XML no se creó sólo para su aplicación en Internet, sino que se propone como un estándar para el intercambio de información estructurada entre diferentes plataformas. Se puede usar en bases de datos, editores de texto, hojas de cálculo y en casi cualquier cosa imaginable.

Es decir, la tecnología XML busca dar solución al problema de expresar información estructurada de la manera más abstracta y reutilizable posible. Que la información sea estructurada quiere decir que se compone de partes bien definidas, y que esas partes se componen, a su vez, de otras partes. De esta forma, se tiene un árbol de pedazos de información. Estas partes se llaman *elementos*, y se las señala mediante *etiquetas*.

Una **etiqueta** consiste en una marca hecha en el documento, que señala una porción de éste como un elemento o pedazo de información con un sentido claro y definido. Las etiquetas tienen la forma `<nombre>`, donde *nombre* es el nombre del elemento que se está señalando.

1.1. Objetivos de diseño de XML.

Antes de continuar, es recomendable comentar el conjunto de objetivos que el equipo de expertos que desarrolló XML se puso como meta.

1. XML debe ser directamente utilizable sobre Internet de forma que los documentos XML puedan ser transferidos entre diferentes sistemas a través de este medio. Al utilizar formato texto, esta transmisión por red es sencilla.
2. XML debe soportar una amplia variedad de aplicaciones de forma que la norma no esté vinculada a ninguna tecnología en concreto.
3. XML debe ser compatible con SGML. Esto se logró haciendo que XML fuese un subconjunto de SGML.
4. Debe ser fácil la escritura de programas que procesen documentos XML. Si se quería que XML fuera compatible con otras aplicaciones, era necesario que el análisis de los documentos en formato XML fuera sencillo.
5. El número de características opcionales en XML debe ser absolutamente mínimo, idealmente cero. De esta forma se consiguen analizadores más simples y estándares.
6. Los documentos XML deben ser legibles por los usuarios de este lenguaje y razonablemente claros. De esta forma, se combina la eficacia de los procesos automáticos con la posibilidad de que las personas podamos obtener información de un documento XML.
7. El diseño de XML debe ser formal, conciso y preparado rápidamente. Dicho de otra forma, la representación de los datos se debe poder hacer en poco tiempo.
8. XML debe ser simple pero perfectamente formalizado.
9. Los documentos XML deben ser fáciles de crear. Con un simple editor plano se pueden elaborar documentos XML, aunque existen aplicaciones muy útiles y eficaces.
10. La brevedad en las marcas XML es de mínima importancia. Se prefiere que cada etiqueta usada tenga su correspondiente etiqueta de cierre, aunque haya que escribir más. Por ejemplo, HTML no es tan estricto en este sentido, pero ello suele generar problemas en los exploradores ya que analizan de forma distinta lo que no está perfectamente cualificado.

1.2. Estructura básica de un documento XML.

Empecemos con un ejemplo para indicar la sintaxis y la estructura de un documento XML. Supongamos que queremos registrar en un documento XML los datos de una película, por ejemplo: "Mar adentro" dirigida por "Alejandro Amenábar" y protagonizada por "Javier Bardem". Simplemente abrimos un nuevo documento con el Bloc de notas (u otro editor "plano") y escribimos:

```
<?xml version="1.0"?>
<pelicula>
    Mar adentro
    <director>Alejandro Amenabar</director>
    <actores>
        <actor>Javier Bardem</actor>
    </actores>
</pelicula>
```

El archivo lo guardaremos con el nombre *mar.xml*. En realidad, la extensión del archivo puede ser cualquiera, pero es recomendable usar *xml* para identificar más fácilmente su tipo, tanto por nosotros como por las aplicaciones.

El documento se compone de dos partes. La primera es el prólogo en donde, entre otras cosas, se indica que es un documento XML que se ajusta a una versión específica de la norma. La segunda parte es el cuerpo del documento. El inicio del cuerpo lo determina una etiqueta, en este caso <pelicula>, y el final del cuerpo la etiqueta de cierre correspondiente, es decir, </pelicula>.

Dentro del cuerpo se pueden colocar tantas etiquetas de apertura y cierre como se necesiten, siempre que estén correctamente anidadas. Por ejemplo, el siguiente fragmento sería incorrecto:

```
<actores>
  <actor>Javier Bardem</actores>
</actor>
```

Sin esta simple regla un programa no podría leer el documento y extraer la información que contiene.

Podríamos incluir elementos vacíos o, dicho de otra forma, podría ser que entre una etiqueta de apertura y su correspondiente de cierre no hubiera nada. Imaginemos que no conocemos ningún actor de la película, en este caso podríamos colocar la etiqueta actores de la forma:

```
<actores></actores>
```

Los navegadores, exploradores o clientes web, además de leer documentos de hipertexto, pueden leer documentos XML. La visualización de una página XML en un cliente web tiene muchas ventajas frente a la que proporciona un editor de texto:

1. Se resalta la sintaxis. El prólogo se ve de un color, la etiquetas de otro y el texto de otro. Esto permite diferenciar unas partes del documento de otras facilmente.
2. Expansión y contracción de los elementos. Si se hace clic con el ratón sobre un elemento que presente un guión a su izquierda, el elemento se contraerá ocultando los elementos que aparezcan a continuación de él y el guión cambia al símbolo (+). Si se hace clic sobre él se volverá a expandir.
3. Comprobación del documento. Si el documento está mal construido, el navegador nos avisará presentando un mensaje de error, además de cómo solucionar el problema. Errores típicos son los que se producen cuando las etiquetas no están correctamente anidadas, o el identificador de la etiqueta de apertura es dintinto a la correspondiente de cierre.

2. Documentos bien formados.

En este apartado trataremos las reglas sintácticas por las que se rige XML. Estas reglas son importantes y deben ser seguidas por los que quieran usar este formato. La finalidad es conseguir documentos *bien formados*.

Veremos las partes de que se compone un documento XML: prólogo, cuerpo y, opcionalmente, epílogo. Cada una de estas partes está constituida por otras más pequeñas. Dentro del cuerpo, las partes

constituyentes se denominan *elementos*.

2.1. Partes de un documento XML.

Es importante recordar que XML es un lenguaje de marcas, y que estas están delimitadas por los símbolos "<" y ">". Lo que se encuentre entre estos dos símbolos es una marca o etiqueta. Utilizaremos el código del principio del tema para que nos sirva de ejemplo.

```
<?xml version="1.0"?>
<pelicula>
   Mar adentro
   <director>Alejandro Amenabar</director>
   <actores>
       <actor>Javier Bardem</actor>
   </actores>
</pelicula>
```

2.1.1. Prólogo.

Todo documento XML debe comenzar con un prólogo en donde se indique el tipo de documento, la norma que va a cumplir y otras informaciones. En el ejemplo, el prólogo es:

```
<?xml version="1.0"?>
```

La etiqueta es diferente a las del resto del documento pues tiene la forma `<?...?>`. Este formato indica que la etiqueta es una **instrucción de proceso**. Dicha instrucción no forma parte del contenido del documento y se utiliza para dar información a las aplicaciones que procesan el documento.

En este caso, se indica que el documento se ciñe totalmente a la norma establecida en la versión 1.0. El atributo *version* es el que permite indicarlo. La última versión es la 1.1. Básicamente, es como la 1.0 con mejoras a la hora de interpretar los caracteres Unicode usados para nombrar los elementos.

Si no se especifica un juego de caracteres para el documento, se supone que se usará UTF-8 ó UTF 16. Para indicar una codificación concreta se escribe el atributo *encoding,* seguido del signo (=) y, entre comillas, el nombre del código a usar: (UTF-8, UTF-16, ISO-8859-1, etc.). Por ejemplo:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
```

```
<?xml-stylesheet type="text/css" href="estilillo.css"?>
```

Esta instrucción indica que existe una hoja de estilo que se puede aplicar al documento. Con el atributo *type* se especifica el tipo de hoja de estilo, en este caso, una hoja de estilo en cascada CSS. También se puede usar hojas de estilo tipo XSLT para realizar transformación de los datos y presentarlos de diferentes formas. El atributo *href* señala la ubicación y archivo en que se encuentra la hoja de estilo.

Estas dos instrucciones de proceso aparecen siempre en el prólogo, pero existen otras instrucciones de proceso que pueden aparecer en el cuerpo del documento, aunque de forma muy poco habitual. Se usan normalmente para incluir instrucciones orientadas a una aplicación específica indicando que se realice algún proceso. El prólogo puede contener otras informaciones, pero las veremos cuando sea oportuno.

2.1.2. Cuerpo.

Los datos que almacena un documento XML se organizan mediante elementos y están ubicados en lo que se conoce como cuerpo. En el caso del ejemplo, se trata de todo lo que se encuentra entre `<pelicula>` y `</pelicula>`, incluidas las dos etiquetas. Este elemento se conoce como el elemento raíz del documento y debe ser único.

2.1.3. Epílogo.

El epílogo es opcional y se sitúa a continuación del cuerpo del documento. Puede contener instrucciones de procesamiento, pero no como las de tipo de documento o declaraciones xml. Realmente, su uso es muy poco habitual.

2.2. Elementos.

Un elemento se identifica por medio de una etiqueta y puede ir acompañado de uno o varios atributos. Además, cada elemento normalmente tiene un contenido.

La estructura de un elemento es:

```
<etiqueta atr1="valor_atr1" atr2="valor_atr2" ...>contenido</etiqueta>
```

Para comprender bien XML es necesario conocer las reglas para los elementos:

- Cada etiqueta de inicio debe tener su etiqueta de cierre correspondiente.
- Las etiquetas no se deben superponer.
- Los documentos XML sólo pueden tener un único elemento raíz.
- Los nombres de los elementos deben cumplir ciertas convenciones de nomenclatura.
- XML establece diferencias entre mayúsculas y minúsculas.
- XML mantiene los espacios en blanco del texto.

El orden de los elementos es muy importante, en el sentido de que deben estar estructurados jerárquicamente en forma de árbol, correctamente anidados y sin superponerse o solaparse entre ellos. Hay que tener en cuenta, también, que sólo puede haber un elemento raíz en el que estén contenidos todos los demás; por lo tanto, habrá una etiqueta contenedora que englobe a todo el contenido y que sea única.

Puede haber etiquetas vacías, es decir sin contenido. En este caso, como no engloban nada en vez de la forma:

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

se puede utilizar la notación:

```
<etiqueta/>
```

A lo mejor nos preguntamos para qué sirve un elemento vacío. Algunas veces, la sola presencia del elemento ya aporta información al documento. Otras veces, puede que la inclusión de un elemento con ese nombre sea obligatoria, pero no sea obligatorio el que tenga contenido dicha etiqueta.

En XML, no es lo que mismo <actor> que <Actor>. Es decir, se distingue entre mayúsculas y minúsculas. Por ello, se dice que XML es *sensible a mayúsculas*. Para evitar errores lo mejor es seguir una norma; por ejemplo, escribir la etiquetas siempre en minúsculas o mayúsculas.

Los nombres de los elementos, atributos, etc. deben empezar siempre por una letra, pudiéndose continuar con letras, dígitos, guiones, rayas, puntos o dos puntos.

En XML se pueden usar los espacios en blanco. Se considera espacio en blanco no sólo el carácter generado con la barra espaciadora sino, también, el tabulador, el retorno de carro y el salto de línea. Los espacios en blanco nos permiten a las personas leer y estructurar de forma más fácil el documento. Para este fin, es conveniente seguir estas reglas:

- Colocar cada elemento en una línea del documento.
- Usar indentación mediante tabuladores para situar unos elementos dentro de otros.
- Usar espacios para conseguir una buena lectura.

2.2.1. Atributos.

Las etiquetas pueden aprovecharse para incluir otros datos usando atributos. Por así decirlo, los atributos permiten añadir propiedades a un elemento, algo así como adjetivarlo.

Un atributo está formado por el nombre del mismo y el valor que toma entrecomillado (comillas simples o dobles), separados por el signo de igualdad (=). Si un elemento tiene un atributo, es obligatorio asignarle un valor. Los atributos sólo se pueden escribir en las etiquetas de apertura.

Normalmente, el uso de atributos permite diferenciar elementos del mismo tipo. Supongamos que hemos encontrado los nombres de otros actores que aparecen en la película *Mar adentro* y queremos añadirlos al documento XML. Con los atributos se puede diferenciar entre los actores protagonistas y secundarios:

```
<?xml version="1.0" encoding="utf-8"?>
<?xml-stylesheet type="text/css" href="estilillo.css"?>
<pelicula>
    Mar adentro
    <director>Alejandro Amenábar</director>
    <actores>
        <actor papel="protagonista">Javier Bardem</actor>
```

```
        <actor papel="secundario">Celso Bugallo</actor>
    </actores>
</pelicula>
```

Como hemos hecho en el ejemplo anterior, es recomendable que un atributo de una etiqueta no tenga la misma categoría que las etiquetas contenidas, pues si fuera de la misma categoría sería conveniente crear otra etiqueta contenida. Así, por ejemplo, tendría más sentido añadir el atributo *idioma* a la etiqueta película, pero no tanto un atributo *vestuario*. Este atributo sería mejor sustituirlo por otra etiqueta que quedara contenida en `<pelicula>`.

```
<?xml version="1.0" encoding="utf-8"?>
<?xml-stylesheet type="text/css" href="estilillo.css"?>
<pelicula idioma="español">
    Mar adentro
    <director>Alejandro Amenábar</director>
    <actores>
        <actor papel="protagonista">Javier Bardem</actor>
        <actor papel="secundario">Celso Bugallo</actor>
    </actores>
    <vestuario>Sonia Grande</vestuario>
</pelicula>
```

Si una etiqueta dispone de varios atributos, el orden en que aparecen es irrelevante ya que los atributos no se organizan de forma jerárquica.

La experiencia dice que no hay que abusar del uso de atributos, ya que a la hora de procesar y obtener los datos es más cómodo si éstos se presentan como contenidos de elementos que como valores de atributos. Es habitual usar sólo atributos cuando se necesita identificar un elemento mediante un valor clave. Por ejemplo, si disponemos de un documento XML con información de varias películas, podríamos añadir un atributo *código* al elemento `<pelicula>` de forma de que cada película tuviera un código distinto:

```
<videoteca>
 <pelicula codigo="1">
. . .
 </pelicula>
 <pelicula codigo="2">
. . .
 </pelicula>
</videoteca>
```

2.2.2. Entidades predefinidas.

Ya sabemos que, como contenido de un elemento, podemos incluir otros elementos o texto independiente. Dicho texto no tiene que limitarse a una línea. Por ejemplo:

```
<critica>
    Un lento, plácido y doloroso viaje a la muerte
    a través de un hermoso, cálido y poético canto a la vida,
    que plasma el final del periplo de este marinero gallego
    que amaba tanto la vida que deseaba la muerte por no
    prolongarla indignamente.
</critica>
```

Existen algunas limitaciones sobre los caracteres que pueden ser escritos como contenido de un elemento, ya que se usan en la sintaxis de XML. Los caracteres &, < , >, ', y " tienen un significado especial, por lo que, para poder escribirlos directamente se deben utilizar las llamadas *entidades* o *referencias de entidad*. Las entidades comienzan con "&" y acaban con ";". El listado siguiente muestra la entidad que sustituye al símbolo que le precede.

```
& &amp;
< &lt;
> &gt;
' &apos;
" &quot;
```

Por ejemplo, si una película se titulara *Black & White* deberíamos escribirlo como:

```
<pelicula idioma="inglés">
    Black &amp; White
    . . .
</pelicula>
```

Cualquier carácter se puede codificar usando *referencia de caracteres*. Se expresan como cadenas de la forma `&#nnn;` ó `&#xnnn`, donde *nnn* es el valor Unicode expresado en decimal o en hexadecimal, según se use un formato u otro. Así, el símbolo de *copyright* © su referencias serían *©* ó *©*.

2.2.3. Secciones CDATA.

Una vez escrito un documento XML, lo normal es que se procese con alguna herramienta software para extraer los datos. Antes de la extracción de datos suele usarse un analizador para comprobar que el documento está bien construido. Además, el analizador permite acceder a los diferentes elementos que componen el documento XML.

Puede ocurrir que sea necesario incluir en el documento partes que no deban ser procesadas por el analizador. Con este fin, se utilizan las secciones CDATA (*Character DATA*) que permiten especificar datos utilizando cualquier carácter, especial o no, sin que se interprete como marcado.

Por ejemplo, a veces, hay que incluir trozos de código en un lenguaje de programación o partes que tienen muchos caracteres especiales y que hay que sustituir por entidades quedando el documento muy engorroso o poco claro. En estos casos, lo mejor es usar una sección CDATA e incluir dichos trozos.

El elemento CDATA comienza con `<![CDATA[` y termina con `]]>`. Todo lo incluido en su interior no se interpretará por el analizador. (Todo no; lógicamente la cadena de cierre `]]>` no puede ser incluida).

Por ejemplo, supongamos un documento XML con una etiqueta `<funcion>` que debe tener como contenido la definición de una función de un lenguaje de programación:

```
function saludo(int a, int b)
{
        if(0 < a && a < b)
         alert("Hola");
}
```

Si usamos CDATA nos quedaría de la forma:

```
<funcion>
    <![CDATA[
        function saludo(int a, int b)
        {
            if(0 < a && a < b)
                alert("Hola");
        }
    ]]>
</funcion>
```

Si no, tendríamos que usar entidades, quedando mucho menos legible:

```
<funcion>
    function saludo(int a, int b)
    {
        if(0 &lt; a &amp;&amp; a &lt; b)
        alert("Hola");
    }
</funcion>
```

2.2.4. Comentarios.

A veces es necesario, incluso conveniente, incluir datos dentro de un documento XML que nos sirvan como referencia, pero que no formen parte de los datos del documento XML. Por ejemplo, el autor del documento, la fecha de creación, o lo que nos venga en gana.

Estas informaciones que sirven de documentación para nosotros o para otras personas, llamadas comentarios, se pueden incluir usando un tipo especial de etiquetas. Para abrir un comentario se usa la secuencia de caracteres `<!--`, y para cerrarlo `-->`. Lo que quede dentro estará oculto para el analizador.

Se pueden usar tantos comentarios como se desee, pero no se pueden anidar unos dentro de otros.

Otro uso de los comentarios es ocultar parte del documento al analizador. Por ejemplo:

```
<!-- Autor: Pepe Cohete
     Fecha: Sextilis MMXII -->
<pelicula idioma="español">
   Mar adentro
   <director>Alejandro Amenábar</director>
   <actores>
      <actor papel="protagonista">Javier Bardem</actor>
```

```
        <!-- No listar más de cinco secundarios -->
        <actor papel="secundario">Celso Bugallo</actor>
    </actores>
</pelicula>
```

2.3. Nodos.

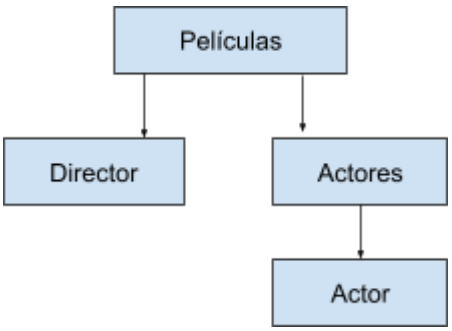
Técnicamente, un documento XML se estructura en forma de árbol. Un árbol es una estructura de datos compuesta de nodos conectados entre sí, donde en su nivel superior existe un nodo llamado raíz que conecta con sus nodos hijos o descendientes. Estos, a su vez, se pueden conectar con sus propios nodos hijos y, así, sucesivamente.

Una propiedad importante de un árbol es cada nodo y sus hijos forman también un árbol. Así, un árbol es una estructura jerárquica de árboles en la cual cada árbol está constituido por pequeños árboles.

En un documento XML, cada nodo está compuesto por una etiqueta, sus atributos y su contenido. El contenido es todo lo que está entre la etiqueta de apertura y la de cierre, lo que nos permite incluir otros nodos.

Si en el contenido de un nodo hay otros nodos, haremos referencia a ellos como nodos descendientes y así sucesivamente. En esta estructura de nodos, siempre debe haber un único nodo raíz del cual descenderán los posibles nodos hijos.

A partir de nuestro documento XML de ejemplo, podemos hacer un grafo con la jerarquía de nodos:



El primer nodo, y raíz del árbol, es `pelicula`. Este nodo tiene dos descendientes: `director` y `actores`. A su vez, `actores` tienen un descendiente, `actor`.

Para que la jerarquía esté bien descrita debemos seguir las reglas que ya vimos sobre el uso de las etiquetas: deben estar perfectamente anidadas y sin que se produzcan solapamientos.