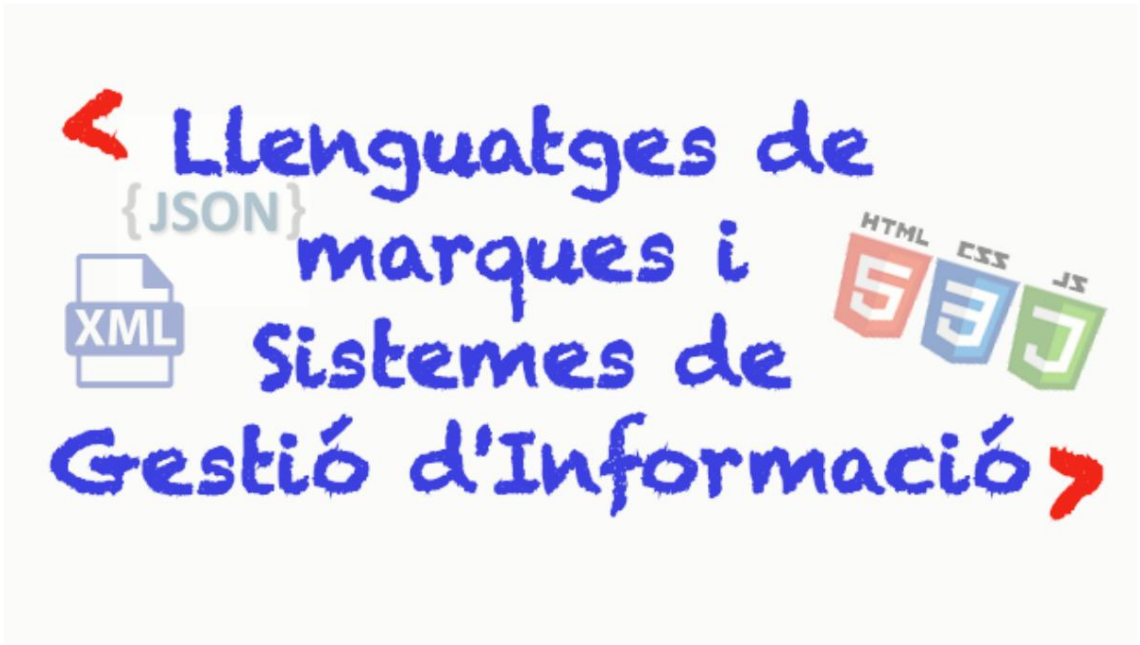

UNIDAD 1: INTRODUCCIÓN A LOS LENGUAJES DE MARCAS

PARTE 2 – DOCUMENTOS XML

Tabla de contenidos

1. DOCUMENTOS XML

XML son las siglas de extensible markup language (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, así que cualquier documento XML es uno a la vez un documento SGML correcto.

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

- Sería fácil crear documentos XML y que los humanos pudieran leerlos y entender fácilmente.
- No sería complicado realizar programas de ordenador que trabajaran con XML.
- El XML se pudiera utilizar en lo máximo posible de campos de aplicación y mantuviera la compatibilidad con SGML.

El resultado ha sido que el XML:

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

2. METALENGUADO

En realidad el XML no es un lenguaje de marcas, sino que es un lenguaje que nos permitirá definir nuestros propios lenguajes de marcas. Esto significa que podremos definir lenguajes de marcas específicos para cada uno de los campos de interés.

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

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

3. ELEMENTOS

La base de XML son los elementos. Un elemento normalmente estará formado por la apertura de una etiqueta –con o sin atributos–, 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, después el contenido Pere Martí y posteriormente cerramos la etiqueta:

```
<nom>Pere Martí</nom>
```

Éste 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 abertura de la etiqueta.

```
<nom càrrec="director">Pere Marti</nom>
```

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

3.1. ETIQUETAS

Las etiquetas se definen dentro del documento XML y se define un formato por separarlos 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 ambos tipos de etiquetas. Así se logra una forma 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.

Y las etiquetas de cierre se definen al igual que las de apertura pero a la vez de empezar la etiqueta se especifican dos símbolos: un símbolo de menor y una barra </. De esta forma se pueden distinguir fácilmente los dos tipos de etiquetas y queda claro en qué sitio debe añadirse el contenido.

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

La especificación XML define claramente cómo crear las etiquetas en los documentos XML, pero en cambio no define ningún tipo ni significado asociado a ningún de las etiquetas. Si, por ejemplo, utilizamos en HTML la etiqueta <b>, ésta nos está indicando que el contenido que contendrá deberá representarse en negrita. Esto obliga a que para hacer documentos XML se deban conocer las etiquetas.

Para XML las etiquetas sólo son una forma de separar el contenido y definir la estructura de los datos que contiene. La interpretación de qué significan los datos y cómo deben representarse se deja a quién leerá el documento.

A pesar de la libertad a la hora de dejar que los usuarios elijan sus propias etiquetas, XML es mucho más riguroso que otros lenguajes a la hora de definir cómo deben escribirse. 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 tanto, una de las cosas que hace el XML es evitar las excepciones tanto como sea posible.

Una de las reglas básicas de XML es que cualquier etiqueta que se abra siempre debe cerrarse.

A pesar de que no existen etiquetas definidas, ya que el XML permite crear las etiquetas que nos hagan falta, por un motivo de legibilidad y de interpretación de los datos recomienda que las etiquetas sean autoexplicativas y pronunciables.

3.2. CONTENIDO

El contenido de un elemento será todo lo que haya entre las etiquetas de apertura y de cierre. En el contenido podemos tener simplemente texto como el siguiente:

```
<persona>Pere Martí</persona>
```

O el contenido pueden ser otros elementos. En el siguiente ejemplo el elemento <persona> contiene otros dos elementos: <nombre> y <apellido>, que además tienen contenido en su interior:

```
<persona>
    <nom>Pere</nom>
    <cognom>Martí</cognom>
</persona>
```

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

Podemos utilizar cualquier carácter representable con el código de caracteres.

- El contenido puede ser tan largo como necesitamos.
- Puede escribirse en cualquier idioma del mundo.
- No importa que haya espacios en blanco o saltos de línea dentro del contenido.

Secciones CDATA

Puede darse el caso de que el contenido de un elemento sea HTML o bien código en alguno lenguaje de programación. Esto obligaría a sustituir por entidades una gran cantidad de símbolos y además restaría legibilidad en el documento. Para evitarlo el 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 <![CDATA[antes de especificar el contenido y se terminan con la combinación de caracteres]]>. Por tanto, podemos definir contenido con símbolos prohibidos dentro de las secciones CDATA sin problemas.

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

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 lo siguiente:

- Definir grandes fragmentos de texto que requieran muchas sustituciones de entidades.
- Si se debe incluir 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.

3.3. ATRIBUTOS

Una forma 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 asociados, mientras que el segundo nos definirá el contenido del atributo.

```
<nom carrec="professor">Frederic Garcia</nom>
```

Puede verse cómo el atributo cargo da un nivel más de información a nombre. Ahora se sabe que hace referencia al nombre de un profesor.

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

```
<persona nom="Xavier" cognom="Sala" />
```

```
<persona cognom="Sala" nom="Xavier" />
```

Otra característica importante de los atributos es que no se pueden repetir los nombres de los atributos dentro de un mismo elemento.

Atributo xml:lang

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

El atributo "xml:space"

El atributo xml:space hace referencia a cómo deben tratarse los espacios que hay en el contenido de un elemento determinado.

Cuando alguien edita un documento XML es corriente utilizar 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.

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

3.4. NOMBRES VÁLIDOS XML

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

Las reglas para definir nombres correctos son:

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

Siguiendo estas reglas tendremos que el siguiente ejemplo:

```
<Ciutat comarca="Alt Empordà">Figueres</Ciutat>
```

La etiqueta <Ciudad> es correcta porque cumple todas las reglas. La primera letra del nombre comienza por un carácter del alfabeto, C, no hay ningún espacio en medio del nombre, y no comienza por las letras xml.

Asimismo, el atributo comarca también es correcto porque comienza con un carácter de el alfabeto, c, no tiene espacios y no comienza por xml.

En cambio no es correcta la etiqueta <Alt Empordà> porque tiene espacios en medio del nombre:

```
<Alt Empordà></Alt Empordà>
```

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

```
<carrera>
  <1aPosicio>Manel Garcia</1aPosicio>
  <2aPosicio>Pere Pi</2aPosicio>
</carrera>
```

Una de las cosas que no debe olvidarse 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.

3.5. COMENTARIOS

A menudo en los documentos XML se especifican datos extra que realmente no forman parte del documento. Estos datos se llaman comentarios y se hacen servir para muchas cosas diversas, tales como:
Generar documentación sobre el documento.

Indicar a la gente que pueda recibir el documento lo que se quería hacer al crearlo.
Otros.

Los analizadores XML son programas que, a partir de la estructura de los documentos XML, nos permite el acceso al contenido a partir de sus etiquetas.
Lo habitual suele ser procesar el documento para generar una salida que pueda ser visualizada más fácilmente, etc.
Generalmente los analizadores XML simplemente 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 —> y se pueden poner comentarios en cualquier lugar del documento XML excepto dentro de las etiquetas. Como ejemplo, en el siguiente documento se ha incluido el comentario "Lista de alumnos" para indicar en qué lugar comienza la lista de alumnos:

```
<professors>
   <nom>Marcel Alaba</nom>
</professors>
<!-- Llista d'alumnes -->
<alumnes>
   <nom>Frederic Garcia</nom>
   <nom>Federicu Pi</nom>
</alumnes>
```

3.6. INSTRUCCIONES DE PROCESO

Las instrucciones de proceso son una forma de dar instrucciones a los programas que leerán el documento. Se definen dentro de los símbolos ”<?” y posteriormente siempre debe especificarse a qué programa van dirigidas con la única restricción que no pueden ser las letras “XML” en cualquier combinación de mayúsculas o minúsculas.

Por tanto, para añadir información que esté destinada a un programa llamado php deben definirse las instrucciones de proceso con una instrucción de proceso:

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

4. DECLARACIÓN XML

La especificación es una línea que define una forma de identificar que uno documento es XML por medio de una etiqueta especial llamada declaración XML, que sirve para indicar que un documento es XML.

La declaración XML es opcional, aunque se considera recomendable, ya que tiene algunos atributos que pueden ayudar a los programas a entender características del documento como el código de caracteres que se utiliza, la versión de XML que se ha usado, etc.

Si la declaración está presente debe tenerse en cuenta que:

- La declaración debe estar en la primera línea del documento y debe empezar en el primer carácter del documento.
- Debe tener obligatoriamente el atributo version, que actualmente sólo puede ser 1.0 o 1.1.

Por tanto, la declaración siguiente es errónea porque deja una línea en blanco, deja espacios frente a la declaración y no define el atributo version:

```

  <?xml ?>
```

Debido a que la declaración es opcional podemos encontrar documentos XML sin esta declaración. Si esto ocurre los programas que lean el documento deben suponer que se trata de un documento XML versión 1.0.

Atributos de la declaración XML

Atributo	Objetivo	Obligatorio?
version	Define la versión de XML que se ha utilizado para crear el documento.	Sí
encoding	Define el código de caracteres que utiliza el documento.	No
standalone	Sirve para indicar si el documento no depende de otros documentos.	No

Ejemplo:

```
<?xml version="1.0" encoding="ISO-8859-15"?>
```

5. CORRECTEZA

Los programas de ordenador no tienen la flexibilidad que tienen los documentos XML y necesitan poder trabajar con datos muy pautados, y precisamente esto es lo que intentan hacer las reglas, evitar que haya partes del documento que puedan ser malinterpretadas. Debido a que la mayoría de las veces los documentos XML deberán de ser leídos por programas, es muy importante tener alguna forma de comprobar que este documento XML esté bien formado, que cumpla las reglas de definición del XML.

Se considera que un documento es correcto o bien formado si cumple con las reglas básicas de creación de un documento XML. Por lo general podemos definir estas reglas como:

- Sólo puede haber un elemento raíz.
- Todas las etiquetas que se abren deben cerrarse.
- Las etiquetas deben estar imbricadas correctamente.
- Los nombres de las etiquetas deben ser correctos.
- Los valores de los atributos deben estar entre comillas.

Sólo puede haber un elemento raíz

```
<persona>
   <nom>Pere</nom>
   <cognom>Garcia</cognom>
</persona>
```

La etiqueta que sale en primer lugar, <persona>, y que termina al final, </persona>, define lo que se llama elemento raíz. Que un elemento sea raíz significa que no tiene ningún elemento que lo contenga.

En este otro ejemplo, en cambio, existen dos elementos raíz, <persona> y <perro>:

```
<persona>
   <nom>Pere Garcia</nom>
</persona>
<gos>
   <nom>Rufy</nom>
</gos>
```

Puede verse que tanto <persona> como <perro> no tienen ningún elemento que los contenga y, por tanto, ambos son elementos raíz. Esto hace que no sea un documento XML bien formado.

También existen dos elementos raíz en el caso en que las etiquetas sean las mismas. Por ejemplo, en el documento siguiente tenemos dos elementos raíz <persona>. Por este motivo, este documento tampoco está bien formado.

También existen dos elementos raíz en el caso en que las etiquetas sean las mismas. Por ejemplo, en el documento siguiente tenemos dos elementos raíz <persona>. Por este motivo, este documento tampoco está bien formado.

```
<persona>
   <nom>Pere</nom>
   <cognom> Pérez </cognom>
</persona>
<persona>
   <nom>Marià</nom>
   <cognom>Garcia</cognom>
</persona>
```

Si se quisiera convertir el documento anterior en un documento bien formado se podría poner un elemento nuevo que englobara a los dos elementos raíz. Por ejemplo, se añade un elemento <personas> y así se crea un documento XML bien formado:

```
<persones>
    <persona>
       <nom>Pere</nom>
       <cognom> Pérez </cognom>
    </persona>
    <persona>
        <nom>Marià</nom>
        <cognom>Garcia</cognom>
    </persona>
</persones>
```

Todas las etiquetas que se abren deben cerrarse

A pesar de lo que ocurre en otros lenguajes de marcas, por ejemplo HTML, en el XML todas las etiquetas que se abran deben ser cerradas obligatoriamente. Por tanto, éste sería un ejemplo correcto:

```
<nom>Pere</nom>
```

Mientras que éste no sería correcto:

```
<nom>Pere
```

En algunos momentos puede ser necesario reflejar algún dato que no requiera ningún valor. Por ejemplo si tenemos una lista de alumnos podría interesarnos marcar cuál de ellos es el delegado. Para definirlo no necesitaríamos ningún dato, ya que simplemente el alumno que tenga la etiqueta <delegado> es el delegado. Por tanto, para definir que Federico Pi es el delegado podríamos estar tentados de hacer esto:

```
<alumnes>
    <alumne>
        <nom>Pere Garcia</nom>
    </alumne>
    <alumne>
        <nom>Frederic Pi</nom>
        <delegat>
    </alumne>
    <alumne>
        <nom>Maria Puigdevall</nom>
    </alumne>
</alumnes>
```

Esto es incorrecto. Si abrimos una etiqueta debemos cerrarla; por tanto, deberíamos hacer:

```
...
<alumne>
   <nom>Frederic Pi</nom>
   <delegat></delegat>
</alumne>
...
```

O bien utilizar la versión de definición de etiquetas vacías terminando la etiqueta con ”/>”:

```
...
<alumne>
    <nom>Frederic Pi</nom>
    <delegat/>
</alumne>
...
```

Las etiquetas deben estar imbricadas correctamente

Para mantener la jerarquía, XML define que las etiquetas no se pueden cerrar si todavía hay alguna etiqueta que forma parte del contenido que no ha sido cerrada. Esto se debe a que no se permite que un elemento tenga una parte del suyo

contenido en un elemento y otra parte en otro, ya que esto rompería la jerarquía de datos.

Por tanto, siempre que se abra una etiqueta dentro de un elemento ésta debe ser cerrada antes de terminar el elemento. Este ejemplo sería correcto porque el elemento <persona> en su contenido abre la etiqueta <nombre> pero antes de terminar cierra </nom>:

```
<persona><nom>Pere</nom></persona>
```

En el XML la sangría se hace en función del nivel en el que se encuentran los datos. En general cada uno de los hijos hace que se incremente la sangría. Por ejemplo:

```
<alumne>
    <persona>
        <nom>
        </nom>
    </persona>
</alumne>
```

Usando la sangría es más fácil determinar en qué lugar se ha producido un error de cierre de etiquetas y arreglarlo. Como podemos ver en el siguiente ejemplo, si las etiquetas de cierre no están en la misma columna que las de apertura es que hay algo mal.

```
<institut>
    <classe>
        <alumne>Pere</alumne>
        <alumne>Joan</alumne>
    </institut>
</classe>
```

Los nombres de las etiquetas y de los atributos deben ser correctos

XML da mucha libertad a la hora de definir los nombres de los elementos y atributos pero no todos los nombres son aceptados.

Por lo general tendremos que:

- No se pueden poner nombres que no empiecen por un carácter ni que contengan espacios en su interior.
- Las mayúsculas y minúsculas son caracteres diferentes para XML.
- Los nombres que comienzan por xml están reservados.

Los valores de los atributos deben estar entre comillas

En el apartado de definición de los atributos ya se comenta que los valores de los atributos siempre deben ir entre comillas pero podemos resumir lo más importante.

No importa cuáles sean las comillas que utilizamos, dobles o simples, siempre que se utilicen las mismas al abrir que al cerrar.

```
<classe nom="Llenguatges de marques">
    <alumne delegat='si'>Pere Garcia</alumne>
    <alumne>Frederic Pi</alumne>
</classe>
```

6. ESTRUCTURA DE LOS DOCUMENTOS XML

Una de las cosas que se consiguen al forzar a que los documentos XML sigan sus normas es hacer que la información que contiene un documento se organice de forma jerárquica.

Por ejemplo, tenemos el siguiente documento XML:

```
<?xml versión="1.0" encoding="UTF-8" ?>
<classe>
   <professor>
       <nom>Marcel</nom>
       <cognom>Puig</cognom>
   </professor>
   <alumnes>
       <alumne>
           <nom>Filomeno</nom>
           <cognom>Garcia</cognom>
       </alumne>
       <alumne>
           <nom>Frederic</nom>
           <cognom>Pi</cognom>
       </alumne>
       <alumne>
           <nom>Manel</nom>
           <cognom>Puigdevall</cognom>
           <delegat/>
       </alumne>
   </alumnes>
</classe>
```

Gracias a que las etiquetas tienen nombres claros se puede deducir que en éste documento XML se están definiendo estructuradamente una clase con un profesor y tres alumnos. Hay un elemento raíz llamado <clase> que engloba todo el documento, y como contenido tiene otros dos elementos, <profesor> y <alumnos>.

Los elementos que forman parte del contenido de un nodo se llaman hijos.

Por tanto, <clase> tiene dos elementos hijos: <profesor> y <alumnos>. Al mismo tiempo el elemento <profesor>, que es uno de los hijos de <clase>, también tiene dos hijos: <nombre> y <apellido>.

Los elementos <nombre> y <apellido> no tienen nodos hijos sino que contienen las datos del documento. Los nodos finales se llaman hojas y generalmente serán siempre nodos de datos.

Todos los elementos que cuelgan de un elemento se llaman genéricamente descendientes. Por tanto, <alumnos>, <alumno> y <nombre> son descendientes de <clase>.

7. CREACIÓN DE DOCUMENTOS XML

Generalmente los documentos XML serán creados y leídos desde programas de ordenador, pero algunas veces también se puede dar el caso de que deban crear manualmente.

Crear un documento XML manualmente es mucho más sencillo que crear un documento binario, ya que no difiere en nada de crear un documento de texto. Simplemente necesitamos un editor de texto plano, como el editor recomendado en la unidad anterior (Visual Studio Code).

Otros editores que permiten trabajar de forma avanzada con XML son:

- Comerciales:
  - oXygen XML Editor (Windows, Linux, Mac OS X)
  - Edite XML Editor (Windows, Linux, Mac OS X)
  - Altova XMLSpy XML editor (Windows)
  - Stylus Studio (Windows)
  - XMLmind (Windows, Linux, Mac OS X)
  - XMLwriter (Windows)
  - Liquido XML Studio (Windows)
  - Serna Enterprise XML Document Editor (Windows, Linux, Mac OS X, Solaris)
- Código abierto:
  - Serna Free Open Source XML Editor (Windows, Linux, Mac OS X, Solaris)
  - XML Copy Editor (Linux)

7.1. CREACIÓN DE UN DOCUMENTO XML

A la hora de definir un grupo de datos dentro de un documento XML habrá que realizar toda una serie de pasos previos que permiten determinar cuáles son los datos necesarios almacenar y posteriormente definir cuál es la estructura que debe darse a estos datos; así pues:

1. Determinación de los datos
2. Determinación de la estructura

Determinación de los datos

Es básico antes de crear un documento XML saber claramente cuáles son los datos que deben ponerse. A menudo esto estará determinado por el programa que las debe

procesar posteriormente, pero también puede que el programa todavía no se haya creado y por tanto la creación esté determinada por las preferencias personales, etc. Si no se está restringido por un programa que ya determine la estructura del documento XML, lo que hace falta es determinar qué datos deben almacenarse. Hay muchos sistemas para hacerlo pero lo sencillo es hacer una lista con todos los datos relevantes.

Es básico antes de crear un documento XML saber claramente cuáles son los datos que deben ponerse. A menudo esto estará determinado por el programa que las debe procesar posteriormente, pero también puede que el programa todavía no exista y por tanto la creación esté determinada por las preferencias personales, etc. Si no se está restringido por un programa que ya determine la estructura del documento XML, lo que hace falta es determinar qué datos deben almacenarse. Hay muchos sistemas para hacerlo pero lo sencillo es hacer una lista con todos los datos relevantes.

Ejemplo:

Document XML per desar les dades d'una biblioteca

Volem crear una biblioteca en què es puguin emmagatzemar les dades dels llibres que hi ha. Per tant, fem una llista amb les dades que considerem que cal que hi hagi en el document:

- Títol
- Subtítol
- Autor
- Any de publicació
- Editorial
- Nom de la col·lecció
- Idioma

Determinación de la estructura

Otra de las cosas básicas a la hora de crear un documento XML es definir cuál es la estructura que tendrán que tener los datos. Esta estructura estará determinada por las necesidades del programa o de la persona que utilizará el

documento XML. De modo que las posibilidades a la hora de crear una estructura para los datos que queremos utilizar son muchos: agrupar por autor, agrupar por libro, agrupar por editorial, agrupar por tema…

Continuamos el ejemplo anterior:

Tria de l'estructura

En una biblioteca podem fer l'estructura des de molts punts de vista. Per exemple, la podem fer a partir dels autors, com aquesta:

- autor 1
  - llibre 1
  - llibre 2
- autor 2
  - llibre 1
- etc.

O bé podem la fer a partir dels llibres d'aquesta manera:

- llibre 1
  - autor 1
- llibre 2
  - autor 1
- etc.

La primera opció és la que s'ha triat per desenvolupar en aquest exemple.

Creación del documento

Como se ha elegido una forma de organización por medio de los autores lo que queda claro es que el primer nivel será una lista de elementos autor más o menos de esta forma:

```
<biblioteca>
    <autor></autor>
    <autor></autor>
    <autor></autor>
    ...
</biblioteca>
```

Dentro de cada uno de los autores los datos para almacenar serán los datos personales del autor (el nombre, en nuestro ejemplo) y la lista de los libros del autor que haya en la biblioteca:

```
<autor>
   <nom>Nom de l'autor</nom>
   <llibres>
       Llista de llibres
   </llibres>
</autor>
```

En el último nivel se pueden poner todos los datos del libro obviando los que ya están implícitas porque están en etiquetas superiores. En el ejemplo, el autor ya queda implícito y, por tanto, no hace falta volver a ponerlo:

```
<llibre>
    <titol>Titol del llibre</titol>
    <subtitol>Subtítol</subtitol>
    <any>Any en què s'ha publicat el llibre</any>
    <idioma>Idioma en què està escrit</idioma>
    <editorial>
        <nom>Nom de l'editorial</nom>
        <col·lecció>Adreça de l'editorial</col·lecció>
    </editorial>
</llibre>
```

En cualquier momento se pueden crear nuevos niveles para agrupar los datos según algún sentido que interese marcar. Por ejemplo, esto es lo que se ha hecho con el editorial.

Hay que tener en cuenta que no es necesario preocuparse si en algunos casos los datos no tienen sentido o no existen, ya que al crear el documento simplemente se pueden eliminar las etiquetas que no tengan sentido.

Por tanto, el documento final podría quedar de esta manera:

```
<biblioteca>
    <autor>
        <nom>John Ronald Reuel Tolkien</nom>
        <llibres>
          <llibre>
              <titol>El Hòbbit</titol>
              <any>2010</any>
              <idioma>català</idioma>
              <editorial>
                  <nom>Edicions de la magrana</nom>
                  <col·lecció>L'Esparver</col·lecció>
              </editorial>
          </llibre>
          <llibre>
              <titol>El senyor dels anells</titol>
              <subtitol>La germandat de l'anell</subtitol>
              <any>2002</any>
              <idioma>català</idioma>
              <editorial>
                  <nom>Editorial Vicens Vives</nom>
              </editorial>
          </llibre>
        </llibres>
    </autor>
    <autor>
        <nom>Isaac Asimov</nom>
        <llibre>
              <titol>Jo, robot</titol>
              <any>2001</any>
              <idioma>català</idioma>
              <editorial>
                  <nom>Edicions Proa</nom>
                  <col·lecció>Proa Butxaca</col·lecció>
              </editorial>
        </llibre>
    </autor>
</biblioteca>
```

8. ESPACIOS DE NOMBRES

Con el XML se pueden crear las etiquetas con el nombre que se desee en el momento en que sea necesario. Por tanto, si tenemos un documento XML en el que se especifican habitaciones de alquiler podríamos tener un documento como este:

```
<lloguer>
    <adreca>
        <carrer>Escudillers, 23</carrer>
        <ciutat>Figueres</ciutat>
        <codi_postal>17600</codi_postal>
    </adreca>
    <habitacio>
        <rect>
            <finestres>2</finestres>
            <portes>1</portes>
        </rect>
    </habitacio>
</lloguer>
```

En el documento XML anterior estamos especificando que hay una habitación de alquiler en la calle Escudillers, 23, de Figueres, que tiene forma rectangular con dos ventanas y una puerta.

Aunque teóricamente el sistema parece perfecto, en la práctica tiene el problema que el lenguaje humano es limitado y además muchas veces hay palabras con varios sentidos. Esto hace que mucha gente pueda elegir las mismas etiquetas para hacer cosas que sean totalmente distintas. Esto de entrada no es un problema hasta que es necesario mezclar documentos que vienen de fuentes diferentes.

Los espacios de nombres permiten mezclar lenguajes XML en el mismo documento y además definir claramente a qué vocabulario pertenece cada etiqueta.

Lo que hacen los espacios de nombres es cambiar los nombres de las etiquetas para que sean únicos. En teoría se podría utilizar cualquier combinación de caracteres, pero cómo que habría el mismo problema que con las etiquetas (alguien podría usarlas) generalmente se hace por medio de una URL (uniform resource locator), que en principio son únicas. Así se podría definir una URL única en nuestro espacio de nombres (http://www.ioc.cat/lloguer) y la URL de SVG (http://www.w3.org/2000/svg) delante

de los elementos y el programa ya no tendrá problemas para determinar cuál vocabulario pertenece cada etiqueta:

```
<http://www.ioc.cat/lloguer:lloguer>
    <http://www.ioc.cat/lloguer:adreça>
        <http://www.ioc.cat/lloguer:carrer>Escudillers, 23</http://www.ioc.cat/lloguer:carrer>
        <http://www.ioc.cat/lloguer:ciutat>Figueres</http://www.ioc.cat/lloguer:ciutat>
        <http://www.ioc.cat/lloguer:codi_postal>17600</http://www.ioc.cat/lloguer:codi_postal>
    </http://www.ioc.cat/lloguer:adreca>
    <http://www.ioc.cat/lloguer:habitacio>
        <http://www.ioc.cat/lloguer:rect>
            <http://www.ioc.cat/lloguer:finestres>2</http://www.ioc.cat/lloguer:finestres>
            <http://www.ioc.cat/lloguer:portes>1</http://www.ioc.cat/lloguer:portes>
        </http://www.ioc.cat/lloguer:rect>
    </http://www.ioc.cat/lloguer:habitacio>
    <http://www.ioc.cat/lloguer:imatge>
        <http://www.w3.org/2000/svg:svg>
            <http://www.w3.org/2000/svg:rect x="0" y="0" width="30" height="60" />
        </http://www.w3.org/2000/svg:svg>
    </http://www.ioc.cat/lloguer:imatge>
</http://www.ioc.cat/lloguer:lloguer>
```

Debido a que escribir todas las direcciones cada vez hace que se pierda la legibilidad del documento, XML permite definir alias para cada una de las URL. Los alias se definen por medio del atributo xmlns de los elementos y se heredan en todo el contenido del elemento:

```
<element xmlns:alies="http://adreca"/>
```

Por tanto, si se define el atributo en la raíz del documento se heredarán los alias a todos los elementos del documento.

```
<lloguer xmlns:lloguer="http://www.ioc.cat/lloguer"
         xmlns:svg="http://www.w3.org/2000/svg" >
    <lloguer:adreca>
        <lloguer:carrer>Escudillers, 23</lloguer:carrer>
        <lloguer:ciutat>Figueres</lloguer:ciutat>
        <lloguer:codi_postal>17600</lloguer:codi_postal>
    </lloguer:adreca>
    <lloguer:habitacio>
        <lloguer:rect>
            <lloguer:finestres>2</lloguer:finestres>
            <lloguer:portes>1</lloguer:portes>
        </lloguer:rect>
    </lloguer:habitacio>
    <lloguer:imatge>
        <svg:svg>
            <svg:rect x="0" y="0" width="30" height="60" />
        </svg>
    </lloguer:imatge>
</lloguer:lloguer>
```

xmlns también permite un espacio de nombres por defecto y que, por tanto, podrá utilizar sus etiquetas sin sobrenombre. Si algún atributo xmlns no define alias se convierte en el espacio de nombres por defecto.

```
<lloguer xmlns="http://www.ioc.cat/lloguer">
    <adreca>
        <carrer>Escudillers, 23</carrer>
        <ciutat>Figueres</ciutat>
        <codi_postal>17600</codi_postal>
    </adreca>
    <habitacio>
        <rect>
            <finestres>2</finestres>
            <portes>1</portes>
        </rect>
    </habitacio>
    <imatge>
        <svg xmlns="http://www.w3c.org/2000/svg">
            <rect x="0" y="0" width="30" height="60" />
        </svg>
    </imatge>
</lloguer>
```

De esta forma todos los descendientes del elemento <alquiler> siguen el suyo espacio de nombres excepto cuando se llega al elemento <svg>, el cual cambia el espacio de nombres por defecto para él y para sus descendientes.

9. BIBLIOGRAFÍA Y REFERENCIAS

Sala, Javier. (2023) Lenguajes de marcas y sistemas de gestión de información. Instituto Abierto de Cataluña