**LENGUAJES DE MARCAS**

Un Lenguaje de Marcas es un modo de codificar un documento donde se incorporan etiquetas, marcas o anotaciones con información adicional relativa a la estructura del texto.

Todo el lenguaje de marcas está definido en un documento llamado *DTD* (Document Type Definition).

En un documento pueden combinarse varios tipos diferentes de lenguajes de marcas, aunque éstos se pueden clasificar en:

* Lenguaje de presentación: Define el formato del texto.
* Lenguaje de procedimientos: Orientado a la presentación, pero el programa que presenta el documento debe interpretar el código en el mismo orden en que aparece.
* Lenguaje descriptivo o semántico: Describen las diferentes partes en las que se estructura el documento, pero sin especificar cómo deben representarse.

Algunos ejemplos de lenguajes de marcado:

* Documentación electrónica:
  + RTF (Rich Text Format): Formato de texto enriquecido desarrollado por Microsoft.
  + TeX: Su objetivo es crear ecuaciones matemáticas complejas.
  + Wikitexto: Permite la creación de páginas WIKI en servidores preparados para este lenguaje.
  + DocBook: Permite generar documentos separando la estructura lógica del documento de su formato.
* Tecnologías de internet:
  + HTML y XHTML: Su objetivo es la creación de páginas web.
  + RSS: Permite la difusión de contenidos web.
* Otros lenguajes especializados:
  + MathML: Expresa el formalismo matemático para que sea entendido por otros sistemas y aplicaciones.
  + VoiceXML: Intercambio de información entre un usuario y una aplicación con capacidad de reconocimiento del habla.
  + MusicXML: Intercambio de partituras entre distintos editores de partituras.

**EVOLUCIÓN DE LOS LLMM**

Surgen en los años 70 como lenguajes formados por el conjunto de códigos de formato que los procesadores de texto introducen en los documentos para dirigir el proceso de presentación mediante una impresora.

Posteriormente, se añadieron como un medio de presentación a la pantalla. Los códigos de estilo de visualización anteriores ya no aparecen, y se emplean otros medios para marcados.

**GML**

Por la falta de estandarización, IBM encarga la construcción de un sistema de edición, almacenamiento y búsqueda de documentos legales. El objetivo de GML es describir los documentos de tal forma que el resultado fuese independiente de la aplicación utilizada.

**SGML**

GML evolucionó hasta que en 1986 dio lugar a SGML. Era un lenguaje muy complejo y requería software caro. Esas son las razones principales de la no utilización actual de este lenguaje.

**HTML**

Es una evolución del GML (Generalized Markup Languaje) que se creó para organizar, enlazar y compatibilizar información. Es una combinación de dos estándares ya existentes:

* ASCII: Es el formato que cualquier procesador de textos sencillo puede reconocer y almacenar.
* SGML: Lenguaje que permite dar estructura al texto, resaltando los títulos o aplicando diversos formatos al texto.

En definitiva, es una versión simplificada de SGML (Standard Generalized Markup Languaje).

**XML**

Resuelve los problemas que presentaba HTML, aunque no se incluye nada relacionado con el diseño, es puramente estructural.

* Hay que definir etiquetas propias
* Permite asignar atributos a las etiquetas
* Usa un esquema para definir las etiquetas y atributos
* La estructura y el diseño son independientes.

XML es un conjunto de otros estándares relacionados entre sí:

* XSL (eXtensible Style Languaje): Permite definir hojas de estilo para XML.
* XML Linking Languaje: Incliye Xpath, Xlink, Xpointer. Determinan aspectos sobre los enlaces entre documentos XML.
* XML Namespaces: Diferencia etiquetas mediante “descripciones”.
* XML Schemas: Define restricciones que se aplicarán a un XML. Los más usados son los DTD.

**XML vs HTML**

|  |  |
| --- | --- |
| **XML** | **HTML** |
| Es un perfil de SGML | Es una aplicación de SGML |
| Especifica cómo se deben definir conjuntos de etiquetas | Aplica un conjunto limitado de etiquetas sobre un documento |
| Modelo de hiperenlaces complejo | Modelo de hiperenlaces simple |
| El navegador es una plataforma para el desarrollo de aplicaciones | El navegador es simplemente un visor de páginas |
| Fin de la guerra de los navegadores y etiquetas propias | El problema de la “no compatibilidad” entre navegadores es difícilmente solucionable |

**XML vs SGML**

|  |  |
| --- | --- |
| **XML** | **SGML** |
| Su uso es sencillo | Su uso es complejo |
| Trabaja con documentos bien formados, no exige que estén validados | Sólo trabaja con documentos válidos |
| Facilita el desarrollo de aplicaciones de bajo coste | Su complejidad hace que las apps sean muy costosas |
| Es muy usado en informática y en más áreas | Sólo se usa en sectores específicos |
| Compatibilidad e integración con HTML | No hay compatibilidad con HTML |
| Formateo con estilos fáciles de aplicar | Formateo y estilos complejos |
| No usa etiquetas opcionales |  |

**ETIQUETAS**

Las etiquetas se señalizan con los símbolos < y >. Siempre hay una de inicio y una de fin. En la señalización de fin se añade una / delante del >.

Ejemplo: **<p>Hola mundo.</p>**

**HERRAMIENTAS DE XML**

EDITORES XML

Se pueden usar editores de texto específicos de XML, ayudan al etiquetado, al formateo, a añadir scripts, etc.

PROCESADORES XML

Un navegador se considera procesador XML, pero existen procesadores específicos para documentos más pesados y complejos.

**XML: ESTRUCTURA Y SINTAXIS**

XML es un lenguaje de etiquetas, organiza la estructura del documento, pero no lo representa como tal. Está compuesto por parejas de etiquetas, y estructurado en árbol.

Las características básicas de XML son:

* Trabaja con HTTP y URLs.
* Todo documento que verifique las reglas de XML está conforme con SGML.
* No se requieren conocimientos de programación para hacer tareas sencillas.
* Los XML son fáciles de crear.
* La difusión de los documentos XML está asegurada por la fácil legibilidad.
* El marcado de XML es legible para los humanos.
* El diseño XML es formal y conciso.
* XML es extensible, adaptable y aplicable a una gran variedad de situaciones.
* XML es orientado a objetos.
* Todo documento XML se compone solo de datos de marcado.

En XML se pueden introducir comentarios al igual que en HTML, con **<!Comentario -->**

Una etiqueta vacía se puede resumir en: **<etiqueta/>**

**DECLARACIÓN XML**

La declaración XML también es llamada prólogo. Se escribe de la siguiente manera: **<?xml version=”1.1” encoding=”UTF-8” standalone=”no/yes” ?>**.

Se forma con la versión de XML a utilizar, el conjunto de caracteres que se vaya a usar, y si depende de otros archivos o no (*standalone*).

La declaración del tipo de documento es igual que en HTML: **<!DOCTYPE Nombre\_tipo>**. No es necesario, pero si recomendable incluirlo.

**EJEMPLAR DE UN XML**

El ejemplar de un XML es el elemento raíz del documento. Todos los datos de un XML han de pertenecer a un elemento del ejemplar.

Los nombres de las etiquetas han de ser autodescriptivos.

Hay varias reglas a seguir en un documento XML:

* Solo debe existir un documento raíz.
* Todos los elementos tienen una etiqueta de inicio y una de cierre.
* Al anidar elementos hay que tener en cuenta que hay que cerrarlos en orden de apertura.
* Los nombres de las etiquetas de inicio son los mismos que los de las de cierre.
* Los nombres de las etiquetas no pueden contener la cadena “]]>” por compatibilidad con SGML. Tampoco se pueden usar los signos >, <, &, “ y ‘.
* Para usar caracteres especiales hay que usar las expresiones &#D; o $#H; donde D y H representan un número decimal o hex perteneciente a un carácter. Ejemplo: € = &#8364; ó &#x20AC;

**ATRIBUTOS**

Permiten añadir propiedades a los elementos de un documento. No tienen jerarquía. Pueden estar entre comillas dobles “ o simples ‘.

**DOCUMENTOS BIEN FORMADOS**

Hay que seguir ciertas normas para que un documento se considere bien formado:

* El documento ha de tener un prólogo con la declaración XML completa.
* Existe un único elemento raíz en el cual se encuentran todos los demás documentos.
* Hay que cumplir las reglas sintácticas del XML.

**UTILIZACIÓN DE ESPACIOS DE NOMBRES**

Permite definir la pertenencia de los elementos y los atributos de un XML al contexto de un vocabulario XML. En resumen, permiten dar un nombre único a cada elemento.

Esto nos permite usar las mismas etiquetas para contenidos distintos, por ejemplo *nombres de alumnos* y *nombres de profesores*.

Ejemplo:

**<?xml version=”1.1” encoding=”UTF-8” standalone=”yes”?>**

**<alumnos xmlns:alumnos=”**[**http://ASIR/alumnos**](http://asir/alumnos)**>**

**<profesores xmlns:profesores=”**[**http://ASIR/profesores**](http://asir/profesores)**>**

**<asistentes>**

**<alumnos:nombre>Fernando González</alumnos:nombre>**

**<alumnos:nombre>Isabel Ramírez</alumnos:nombre>**

**<alumnos:nombre>Ana González</alumnos:nombre>**

**<alumnos:nombre>Francisco López</alumnos:nombre>**

**<profesores:nombre>Miguel Carmona</profesores:nombre>**

**<profesores:nombre>Isabel Vaz</profesores:nombre>**

**<profesores:nombre>Javier González</profesores:nombre>**

**</asistentes>**