Tema 4. Lenguajes de marcado. Información de sus etiquetas

XHTML vs HTML 5

XHTML distingue entre mayúsculas y minúsculas en sus etiquetas, mientras que HTML5, así como HTML, no son sensibles a mayúsculas y minúsculas.

La siguiente diferencia es la compatibilidad del navegador: XHTML es compatible con todos los navegadores, mientras que HTML 5 tiene ciertas etiquetas semánticas que no son compatibles en versiones antiguas de algunos navegadores.

HTML5 es más adecuado para dispositivos móviles, mientras que XHTML es más adecuado para pantallas de escritorio. HTML5 es más permisible que XHTML a la hora de escribir las etiquetas y elementos.

Tema 4. Lenguajes de marcado. Información de sus etiquetas

4.7. LaTeX

Ilustración 16: Logo lenguaje de marcado LATEX. Fuente: dominio público.

Donald E. Knuth desarrolló **TEX** en el año 1977 como un sistema de composición tipográfica para preparar libros, especialmente aquellos que contienen expresiones matemáticas.

Basado en él, Leslie Lamport desarrolló **LATEX** (llamado LATEX 2.09) en 1985 para preparar documentos permitiendo enfocarse en la estructura de un documento en lugar de sus detalles de formato.

LATEX consiste en un paquete de macros, un **sistema de preparación de documentos** para una composición tipográfica de alta calidad. Se utiliza con mayor frecuencia para **documentos técnicos o científicos** de tamaño medio a grande, pero se puede utilizar para casi cualquier forma de publicación. Permite compilar un archivo en formato .text a otros formatos como pdf.

LATEX se puede utilizar para preparar cartas, aplicaciones, artículos, informes, publicaciones, tesis, libros, documentos de química, física, computación, biología, leyes, literatura, música y cualquier otra temática.

LATEX tiene la posibilidad de **generar automáticamente** varias listas de contenidos, índices, glosario, numerar capítulos y figuras, incluir y organizar la bibliografía

adecuada, mantener índices y referencias cruzadas, etc. Por tanto, la posibilidad de cometer algún error en la numeración y referencia de elementos (apartados, tablas, figuras o ecuaciones), al elegir el tamaño y el tipo de fuente para las diferentes secciones y subsecciones, o en la preparación de referencias bibliográficas, puede evitarse.

Las **operaciones principales** para crear documentos en LATEX son: editar, compilar, y visualizar:

- **Editar**: utilizar un editor o IDE para generar un archivo, con terminación .tex, que contiene el código en LATEX describiendo la estructura y el contenido del documento.
- **Compilar.** Compilar es el proceso, realizado por el motor de LATEX, que convierte los archivos .tex en documentos con formato .pdf que se pueden imprimir y ver en pantalla.
- **Visualizar.** Una vez compilado el documento, si no hubo errores, se puede visualizar el documento generado por LATEX. En TeXnicCenter, por ejemplo, hay un botón que permite iniciar el visualizador de documentos.

Entre las **ventajas** de LATEX se encuentran:

- LATEX es ampliamente utilizado en entornos científicos. Muchas revistas aceptan documentos escritos en LATEX.
- Excelente calidad del documento final con salida en distintos formatos: dvi, pdf, ps, etc.

- Los ficheros fuente .tex son ficheros ASCII y pueden ser compilados en cualquier sistema operativo.
- Es gratuito.
- Muy potente.

Como **desventajas** destaca:

No es un procesador del tipo **wysiwyg** (***W**hat **Y**ou **S**ee **I**s **W**hat **Y**ou **G**et*) «lo que ves es lo que consigues», por lo que es necesario un proceso de compilación (con posibles errores en caso de que algo esté mal escrito).

LATEX no es un procesador de textos, permite a los autores evitar preocuparse por la apariencia de los documentos, sino concentrarse en obtener el contenido correcto.

Ejemplo de código LATEXT:

```
\documentclass{article}

\title{Cartesian closed categories and the price of eggs}

\author{Jane Doe}

\date{September 1994}

\begin{document}

\maketitle

Hello world!

\end{document}
```

Tema 4. Lenguajes de marcado. Información de sus etiquetas

4.8. MarkDown

Ilustración 17: Logo lenguaje de marcado ligero Markdown. Fuente: dominio público.

Markdown es un lenguaje de marcado ligero que puede usar para agregar elementos de formato a documentos de texto sin formato. Markdown, creado por John Gruber en 2004, es ahora uno de los lenguajes de marcado más populares del mundo.

Cuando se crea un archivo con formato de Markdown, se agrega la sintaxis de Markdown al texto para indicar qué palabras y frases deben verse diferentes:

- **Encabezado**: se agrega un signo de número antes de él (por ejemplo, # Encabezado uno).
- Una frase en **negrita** agregue dos asteriscos antes y después (por ejemplo, ** este texto está en negrita **).

Tema 4. Lenguajes de marcado. Información de sus etiquetas

Markdown no es un lenguaje **WYSIWYG**. Según su creador, la sintaxis de Markdown está diseñada para ser legible y simple, por lo que el texto en los archivos de Markdown se puede leer incluso si no se procesa, es por eso por lo que es considerado un lenguaje de marcado ligero.

Uso de Markdown

Entre los principales usos de Markdown se encuentran:

- Contenido para sitios web estáticos autogenerados (Jekyll, GatsbyJS, Hugo)
- Documentos
- Notas
- Libros
- Presentaciones
- Mensajes de correo electrónico
- Documentación técnica

Como ventajas de Markdown destaca que es **portable**, a diferencia de las aplicaciones de procesamiento de texto tradicionales como Microsoft Word que bloquean su contenido en un formato de archivo propietario.

Markdown es **independiente de la plataforma**. Puede crear texto con formato Markdown en cualquier dispositivo que ejecute cualquier sistema operativo.

Markdown se utiliza en las principales plataformas colaborativas de internet. Los sitios web como Reddit y GitHub admiten Markdown, y muchas aplicaciones de escritorio y basadas en la web lo admiten.

Tema 4. Lenguajes de marcado. Información de sus etiquetas

¿Cómo funciona markdown?

Cuando se escribe un documento en Markdown, el texto se almacena en un archivo de texto sin formato que tiene una **extensión .md** o markdown. Para visualizar el archivo Markdown es necesaria una aplicación capaz de renderizar el archivo.

Las aplicaciones de Markdown utilizan algo llamado **procesador Markdown** (también conocido como "analizador" o "implementación") para tomar el texto con formato Markdown y enviarlo a formato HTML de manera que pueda verse en un navegador web o combinarse con una hoja de estilo e imprimirse.

4.9. Otras tecnologías

JSON

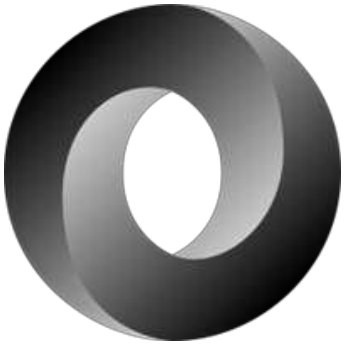

Ilustración 18: Logo lenguaje de intercambio de información JSON. Fuente: dominio público.

JSON es un **formato de intercambio de datos.** Los formatos para el intercambio de datos son formatos de texto que se utilizan para intercambiar datos entre plataformas software.

Las aplicaciones informáticas necesitan formatos de intercambio de datos, como XML y JSON, para intercambiar datos entre diferentes sistemas informáticos.

La capacidad de estos sistemas únicos para comunicarse es parte integral de muchas empresas y organizaciones. Si cada uno de estos sistemas necesitara un traductor para todas las formas en que otros sistemas estructuran sus datos, entonces las comunicaciones consumirían una cantidad adecuada de tiempo y recursos.

Para evitar esa problemática, los sistemas acuerdan un formato único para obtener datos y emplear un solo traductor. JSON es un formato de intercambio de datos que muchos sistemas han acordado utilizar para la comunicar datos entre sí.

Muchos, pero no todos los sistemas, han acordado JSON para comunicar datos. Existen formatos de intercambio de datos, como Extensible Markup Language (XML), que fueron antes de que siquiera se popularizara JSON.

Muchos sistemas tienen y siguen utilizando otros formatos, como XML o formatos más tabulares y delimitados, como valores separados por comas (CSV).

JSON son las siglas de ***J**ava**S**cript **O**bject **N**otation*. El nombre de esta tecnología para el intercambio de datos puede inducir a error a las personas a pensar que necesitarán aprender JavaScript para poder utilizar JSON.

No es necesario aprender JavaScript para poder utilizar JSON, pero cabe destacar que su gran popularidad viene conferida gracias a que el lenguaje JavaScript lo interpreta nativamente. Eso significa que JSON es un formato idóneo para intercambiar datos entre el backend y el frontend de una aplicación web.

E l **frontend** de una aplicación web puede ser una aplicación desarrollada en JavaScript vanilla o con algún framework (**Angular, React, Vue**) y recibir datos en formato JSON desde una **API REST** en el **backend**. Al ser JSON un formato nativo para JavaScript, no se pierde tiempo en conversión de formatos para poder utilizar los datos desde JavaScript.

Tema 4. Lenguajes de marcado. Información de sus etiquetas

YAML

Ilustración 19: Logo lenguaje de marcado XHTML. Fuente: dominio público.

YAML es un lenguaje de serialización de datos que se usa a menudo para escribir archivos de configuración. YAML puede considerarse otro lenguaje de marcado pero en general YAML es para datos, no para documentos. YAML se utiliza en conjunto a otros lenguajes de programación.

Sintaxis

YAML tiene características que provienen de Perl, C, XML, HTML y otros lenguajes de programación. YAML también es un superconjunto de JSON, por lo que los archivos JSON son válidos en YAML.

YAML usa sangría estilo Python para indicar anidamiento. Los caracteres de tabulación no están permitidos, por lo que se utilizan espacios en blanco en su lugar. No hay símbolos de formato habituales, como llaves, corchetes, etiquetas de cierre o comillas. Los archivos YAML usan una extensión **.yml** o **.yaml**.

La estructura de un archivo YAML es un mapa o una lista:

- Los mapas le permiten asociar pares clave-valor. Cada clave debe ser única y el orden no importa. Piense en un diccionario de Python o una asignación de variable en un script Bash.
- Un mapa en YAML debe resolverse antes de que se pueda cerrar y se crea un nuevo mapa. Se puede crear un nuevo mapa aumentando el nivel de sangría o resolviendo el mapa anterior e iniciando un mapa adyacente.

- Una lista incluye valores enumerados en un orden específico y puede contener cualquier número de elementos necesarios. Una secuencia de lista comienza con un guión (-) y un espacio, mientras que la sangría la separa del padre. Puede pensar en una secuencia como una lista de Python o una matriz en Bash o Perl. Se puede incrustar una lista en un mapa.

YAML también contiene escalares, que son datos arbitrarios (codificados en Unicode) que se pueden usar como valores como cadenas, enteros, fechas, números o booleanos.

Al crear un archivo YAML, se deben seguir unas reglas de sintaxis de manera que los archivos sean válidos. Un linter es una aplicación que verifica la sintaxis de un archivo. El comando yamllint puede ayudar a garantizar que haya creado un archivo YAML válido antes de entregarlo a una aplicación.

Usos de YAML

Uno de los usos más comunes de YAML es crear archivos de configuración. Se recomienda que los archivos de configuración se escriban en YAML en lugar de JSON, aunque se pueden usar indistintamente en la mayoría de los casos, porque YAML tiene una mejor legibilidad y es más fácil de usar.

YAML también es utilizado por la herramienta de automatización para sistemas, como es el caso de las configuraciones de las herramientas:

- Ansible para crear procesos de automatización
- Recursos e implementaciones de Kubernetes.

Una ventaja de usar YAML es que los archivos YAML se pueden agregar al control de código fuente, como GitHub, para que los cambios se puedan rastrear y auditar.

Tema 4. Lenguajes de marcado. Información de sus etiquetas

Ansible

Los Playbooks de **Ansible** se utilizan para orquestar los procesos de TI. Un Playbook en Ansible es un archivo YAML que contiene 1 o más reproducciones y se usa para definir el estado deseado de un sistema.

Cada Playbook puede ejecutar una o más tareas, y cada tarea invoca un módulo de Ansible. Los módulos se utilizan para realizar tareas de automatización en Ansible. Los módulos Ansible se pueden escribir en cualquier lenguaje que pueda devolver JSON, como Ruby, Python o bash.

Un Playbook de Ansible consta de mapas y listas. Para crear un Playbook, se inicia una lista YAML que nombre la obra y luego enumere las tareas en una secuencia. La sangría no es una indicación de herencia lógica. Es necesario pensar cada línea como un tipo de datos YAML (una lista o un mapa).

Mediante el uso de plantillas YAML, en Ansible se pueden programar tareas repetitivas para que sucedan automáticamente sin tener que aprender un lenguaje de programación avanzado.

Kubernetes

Kubernetes funciona según el estado definido y el estado real. Los objetos de Kubernetes representan el estado de un clúster y le dicen a Kubernetes cómo desea que se vea la carga de trabajo. Los recursos de Kubernetes, como pods, objetos e implementaciones, se pueden crear con archivos YAML.

Al crear un objeto de Kubernetes, deberá incluir especificaciones para definir el estado deseado del objeto. La API de Kubernetes se puede utilizar para crear el objeto. La solicitud a la API incluirá las especificaciones del objeto en JSON, pero la mayoría de las veces proporcionará la información requerida a kubectl como un archivo YAML.

Kubectl convertirá el archivo a YAML cuando realice la solicitud de API. Una vez que se ha creado y definido un objeto, Kubernetes trabaja para asegurarse de que el objeto siempre exista.

Los desarrolladores o administradores de sistemas especifican el estado definido mediante los archivos YAML o JSON que envían a la API de Kubernetes. Kubernetes usa un controlador para analizar la diferencia entre el nuevo estado definido y el estado real en el clúster.