


entornos\_desarrollo\_2024-25 / UD6\_refactorizacion\_documentacion / apuntes  
/ ud6\_3\_markdown.md 



danielmartinan Minor bugs

298b1ae · 2 months ago



619 lines (437 loc) · 19.8 KB

# UD6 - Optimización y Documentación

- [1. Introducción a Markdown](#)
  - [1.1. Ventajas de Markdown](#)
  - [1.2. Usos de Markdown](#)
- [2. Sintaxis básica](#)
  - [2.1. Encabezados](#)
    - [2.1.1. Encabezado de nivel 4](#)
  - [2.2. Párrafos y saltos de línea](#)
  - [2.3. Negrita y cursiva](#)
  - [2.4. Listas](#)
    - [2.4.1. Listas no ordenadas](#)
    - [2.4.2. Listas ordenadas](#)
    - [2.4.3. Listas de tareas](#)
  - [2.5. Citas y bloque de citas](#)
  - [2.6. Código en línea y bloques de código](#)
    - [2.6.1. Código en línea](#)
    - [2.6.2. Bloques de código](#)
  - [2.7. Imágenes y enlaces](#)
  - [2.8. Emojis y otros caracteres especiales](#)
  - [2.9. Escapado de caracteres](#)
    - [2.9.1. Caracteres que requieren escapado](#)
  - [2.10. Tablas](#)
    - [2.10.1. Sintaxis básica de tablas](#)
    - [2.10.2. Ejemplo de tabla](#)

- [2.11. HTML en Markdown](#)
  - [2.11.1. Sintaxis de HTML en Markdown](#)
  - [2.11.2. Ejemplo de HTML en Markdown](#)
- [2.12. Markdown extendido](#)
  - [2.12.1. Diagramas con Mermaid](#)
  - [2.12.2. Ecuaciones matemáticas con MathJax](#)
- [3. Referencias](#)

# 1. Introducción a Markdown

Markdown es un lenguaje de marcado ligero diseñado para facilitar la escritura de texto formateado de manera sencilla y eficiente. Fue creado por John Gruber en 2004 con el objetivo de permitir la escritura de documentos estructurados de forma intuitiva y sin necesidad de etiquetas complicadas como en HTML.

Es ampliamente utilizado en documentación técnica, blogs, foros y plataformas de desarrollo como GitHub y GitLab. Su simplicidad permite que cualquier persona pueda escribir texto enriquecido sin necesidad de aprender un sistema complejo de formateo.

## 1.1. Ventajas de Markdown

Markdown ofrece diversas ventajas en comparación con otros formatos de documentación, entre las que destacan:

- **Simplicidad:** Su sintaxis es fácil de aprender y utilizar, incluso para quienes no tienen experiencia en lenguajes de marcado.
- **Portabilidad:** Los archivos Markdown son simples archivos de texto plano, lo que los hace ligeros y fáciles de compartir.

[entornos\\_desarrollo\\_2024-25](#) / [UD6\\_refactorizacion\\_documentacion](#) / [apuntes](#)  
/ [ud6\\_3\\_markdown.md](#)

↑ Top

Preview

Code

Blame

Raw



- **Legibilidad:** Los documentos en Markdown son comprensibles incluso en su forma sin procesar, lo que facilita la revisión y edición.

## 1.2. Usos de Markdown

Markdown se emplea en una gran variedad de contextos, entre los cuales destacan:

- **Documentación técnica:** Es utilizado para redactar manuales de usuario, guías de instalación y especificaciones de software.
- **Repositorios de código:** En plataformas como GitHub y GitLab, se usa para escribir archivos README.md, documentación de proyectos y wikis.

- **Blogs y artículos:** Sistemas de gestión de contenido como Jekyll, Hugo y Ghost permiten escribir artículos en Markdown y convertirlos a HTML.
- **Foros y mensajería:** Se usa en plataformas como Discord, Reddit y Stack Overflow para formatear mensajes y comentarios.
- **Presentaciones y reportes:** Puede combinarse con herramientas como Pandoc o Remark.js para generar diapositivas o informes en PDF.

Markdown es una herramienta versátil que permite estructurar documentos de forma clara y sencilla sin necesidad de depender de editores de texto avanzados. En las siguientes secciones se abordará su sintaxis y las mejores prácticas para su uso en documentación técnica.

## 2. Sintaxis básica

---

Markdown ofrece una sintaxis sencilla para aplicar formato al texto sin necesidad de utilizar etiquetas complejas. A continuación, se explican los elementos fundamentales de su sintaxis.

### 2.1. Encabezados

Los encabezados permiten estructurar el contenido y se crean utilizando el símbolo `#`. La cantidad de `#` determina el nivel del encabezado.

Ejemplo:

```
#### Encabezado de nivel 4
##### Encabezado de nivel 5
##### Encabezado de nivel 6
```



Esto se renderiza como:

#### 2.1.1. Encabezado de nivel 4

Encabezado de nivel 5

Encabezado de nivel 6

Fíjate que las almohadillas `#` van seguidas de un espacio antes del texto del encabezado.

### 2.2. Párrafos y saltos de línea

Para crear un párrafo, basta con escribir el texto seguido de una línea en blanco antes de empezar otro párrafo.

Si se desea un **salto de línea dentro de un mismo párrafo**, se debe finalizar la línea con dos espacios en blanco antes de presionar Enter.

Ejemplo:

Este es un párrafo.



Este es otro párrafo separado por una línea en blanco.

Esta línea está en el mismo párrafo que la anterior.

## 2.3. Negrita y cursiva

Markdown permite resaltar texto en **negrita** o *cursiva* mediante el uso de asteriscos ( \* ) o guiones bajos ( \_ ).

Ejemplo:

**\*\*Este texto está en negrita\*\***

*\*Este texto está en cursiva\**

**\_\_Este también está en negrita\_\_**

*\_Este también está en cursiva\_*

**\*\*\*Este texto está en negrita y cursiva\*\*\***



Renderizado:

**Este texto está en negrita**

*Este texto está en cursiva*

**Este también está en negrita**

*Este también está en cursiva*

***Este texto está en negrita y cursiva***

## 2.4. Listas

Las listas permiten estructurar elementos en un formato organizado.

### 2.4.1. Listas no ordenadas

Se crean con - , + o \* seguidos de un espacio.

- Elemento 1

- Elemento 2

- Sub-elemento 2.1

- Sub-elemento 2.2

+ Elemento 3

\* Elemento 4



Renderizado:

- Elemento 1
- Elemento 2
  - Sub-elemento 2.1
  - Sub-elemento 2.2
- Elemento 3
- Elemento 4

### 2.4.2. Listas ordenadas

Se numeran con 1. , 2. , 3. , etc.

1. Primer elemento
2. Segundo elemento
  1. Sub-elemento 2.1
  2. Sub-elemento 2.2
3. Tercer elemento



Renderizado:

1. Primer elemento
2. Segundo elemento
  - i. Sub-elemento 2.1
  - ii. Sub-elemento 2.2
3. Tercer elemento

### 2.4.3. Listas de tareas

Las listas de tareas son un tipo especial de listas en Markdown, utilizadas en plataformas como **GitHub**, **GitLab** y **JIRA** para gestionar pendientes. Se crean usando corchetes [ ] para tareas sin completar y [x] para las completadas.

Ejemplo:

- [ ] Revisar la documentación
- [x] Implementar la funcionalidad principal
- [ ] Escribir pruebas unitarias
- [x] Actualizar el README



Renderizado:

- ☐ Revisar la documentación

- ☒ Implementar la funcionalidad principal
- ☐ Escribir pruebas unitarias
- ☒ Actualizar el README

Las listas de tareas son útiles para organizar proyectos y hacer seguimiento de avances en desarrollo de software.

## 2.5. Citas y bloque de citas

Las citas se crean usando el símbolo `>` al inicio de la línea.

```
> Esto es una cita.  
> Se puede extender en varias líneas.
```



Renderizado:

```
Esto es una cita.  
Se puede extender en varias líneas.
```

Puedes crear citas anidadas, añadiendo un segundo `>` en cada nivel.

```
> Esto es una cita.  
>> Esta es una cita anidada.
```



Renderizado:

```
Esto es una cita.  
    Esta es una cita anidada. Fin de la cita anidada.  
Fin de la cita principal.
```

Dentro de una cita, puedes usar otros elementos de Markdown, como listas, negritas, cursivas, etc.

```
> #### Encabezado dentro de la cita  
>  
> - Elemento 1  
> - **Elemento 2**  
> - *Elemento 3*
```



Renderizado:

```
Encabezado dentro de la cita  
• Elemento 1
```

- Elemento 2
- *Elemento 3*

## 2.6. Código en línea y bloques de código

### 2.6.1. Código en línea

Se utiliza comillas invertidas ``` para resaltar fragmentos de código dentro de un texto.

Este es un ejemplo de código en línea: ``System.out.println("Hola Mundo");``



Renderizado:

Este es un ejemplo de código en línea: `System.out.println("Hola Mundo");`

### 2.6.2. Bloques de código

Para bloques de código con resaltado de sintaxis, se utilizan tres comillas invertidas ````` antes y después del código, indicando el lenguaje.

Ejemplo en Java:

```
```java public class HolaMundo { public static void main(String[] args) {  
System.out.println("Hola Mundo"); } } ```
```

Renderizado:

```
public class HolaMundo {  
    public static void main(String[] args) {  
        System.out.println("Hola Mundo");  
    }  
}
```



Ejemplo en Markdown:

```
# Esto es un comentario en Markdown  
**Texto en negrita**  
- Elemento de lista
```



Estos bloques permiten **resaltar sintaxis** específica según el lenguaje.

Otros lenguajes soportados:

- plaintext o text
- bash o shell

- python
- javascript o js
- html
- css
- sql
- xml
- ...

## 2.7. Imágenes y enlaces

Para insertar imágenes en un documento Markdown, se utiliza la siguiente sintaxis:

```
![Texto alternativo](URL de la imagen)
```



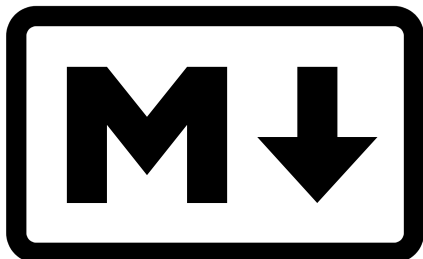
El texto alternativo se muestra si la imagen no puede cargarse o si se visualiza en un lector de pantalla. La URL de la imagen puede ser una dirección web o una ruta relativa al archivo de Markdown.

Ejemplo:

```
![Logo de Markdown]  
(https://upload.wikimedia.org/wikipedia/commons/4/48/Markdown-mark.svg)
```



Renderizado:



Para crear enlaces a otras páginas web o a secciones dentro del mismo documento, se utiliza la siguiente sintaxis:

```
[Texto del enlace](URL del enlace)
```



Ejemplo:

```
[Markdown Guide](https://www.markdownguide.org)
```



Renderizado:



## [Markdown Guide](#)

Si se desea que el enlace se abra en una nueva pestaña del navegador, se puede añadir el atributo `target="_blank"` al enlace:

```
[Markdown Guide](https://www.markdownguide.org){:target="_blank"}
```



Renderizado:

[Markdown Guide](#){:target="\_blank"}

Puedes añadir un título, en forma de *tooltip*, al enlace, de la siguiente forma:

```
[Markdown Guide](https://www.markdownguide.org "Guía de Markdown")
```



Renderizado:

[Markdown Guide](#)

También podemos crear enlaces internos a secciones del mismo documento utilizando el siguiente formato:

```
[Texto del enlace](#nombre-de-la-seccion)
```



Por ejemplo, para enlazar a la sección "Imágenes y enlaces" de este documento, se utilizaría el siguiente enlace:

```
[Ir a Imágenes y enlaces](#imágenes-y-enlaces)
```



Renderizado:

[Ir a Imágenes y enlaces](#)

Por último, podemos crear enlaces de referencia. Estos se definen al final del documento y se utilizan para referenciar enlaces largos o complejos en el texto. La sintaxis es la siguiente:

```
[Texto del enlace][nombre de la referencia]  
...  
[nombre de la referencia]: URL del enlace
```



Ejemplo:



...

[1]: <https://www.markdownguide.org>

Esta sección ha cubierto la sintaxis esencial de Markdown para escribir documentos estructurados y con formato. En la siguiente sección se abordarán las tablas y los enlaces, herramientas clave para mejorar la legibilidad de la documentación.

## 2.8. Emojis y otros caracteres especiales

En Markdown, se pueden utilizar emojis y otros caracteres especiales para añadir expresividad al texto. Los emojis se escriben entre dos puntos `:` y pueden representar emociones, objetos, animales, entre otros.

Ejemplo:

```
:smile: :heart: :rocket:
```



Renderizado:



Puedes encontrar una lista completa de emojis [aquí](#)

## 2.9. Escapado de caracteres

En Markdown, algunos caracteres tienen un significado especial y se utilizan para aplicar formato al texto. Si se desea mostrar estos caracteres de forma literal, es necesario **escaparlos** con una barra invertida `\`.

### 2.9.1. Caracteres que requieren escapado

- `\` (barra invertida)
- ``` (comillas invertidas)
- `*` (asterisco)
- `_` (guion bajo)
- `{` y `}` (llaves)
- `[` y `]` (corchetes)
- `(` y `)` (paréntesis)
- `#` (almohadilla)
- `+` (signo más)
- `-` (guión)
- `.` (punto)

- ! (signo de exclamación)
- > (signo mayor que)
- < (signo menor que)
- ~ (virgulilla)
- | (barra vertical)
- & (ampersand)

Ejemplo:

```
\*Texto en negrita\*. Añado más caracteres especiales: \ ` \_ \{ \} \[ \]
\ ( \) \# \+ \- \. \! \> \< \~ \|
```



Renderizado:

**\*Texto en negrita\***.

Añado más caracteres especiales: ` \_ { } [ ] ( ) # + - . ! > < ~ |

Fíjate como no se interpreta el significado del asterisco sino que se muestra literalmente como un carácter más del texto.

## 2.10. Tablas

Las tablas son una forma efectiva de organizar y presentar datos de manera estructurada. En Markdown, se pueden crear tablas utilizando barras verticales | y guiones - para separar las celdas y definir los encabezados.

### 2.10.1. Sintaxis básica de tablas

La sintaxis básica de una tabla en Markdown es la siguiente:

```
| Encabezado 1 | Encabezado 2 | Encabezado 3 |
| ----- | :-----: | -----: |
| Celda 1 | Celda 2 | Celda 3 |
| Celda 4 | Celda 5 | Celda 6 |
```



- Las barras verticales | se utilizan para separar las celdas.
- La segunda fila define la alineación de cada columna:
  - :--- indica alineación a la izquierda.
  - :---: indica alineación centrada.
  - ---: indica alineación a la derecha.
  - --- indica alineación predeterminada a la izquierda.
- Las celdas pueden contener texto, enlaces, imágenes, listas y otros elementos de Markdown.

- Las tablas deben tener al menos tres guiones - en la segunda fila para separar los encabezados del contenido.

### 2.10.2. Ejemplo de tabla

A continuación, se muestra un ejemplo de una tabla sencilla en Markdown:

```
| Nombre | Edad | Ciudad |
| :-----: | :----: | :-----: |
| Juan | 25 | Madrid |
| María | 30 | Barcelona |
| Carlos | 28 | Valencia |
```



Renderizado:

Nombre	Edad	Ciudad
Juan	25	Madrid
María	30	Barcelona
Carlos	28	Valencia

La creación de tablas en markdown tiene una serie de limitaciones, como la falta de soporte para celdas combinadas o la limitación en la personalización del estilo. Sin embargo, para tablas sencillas y rápidas, es una herramienta útil y efectiva.

## 2.11. HTML en Markdown

Una de las ventajas de Markdown es su capacidad para integrar código HTML en un documento. Esto permite añadir elementos de HTML, como estilos, scripts o elementos multimedia, para personalizar la apariencia y funcionalidad del contenido.

### 2.11.1. Sintaxis de HTML en Markdown

Para insertar código HTML en un documento Markdown, se utiliza la siguiente sintaxis:

```
<etiqueta atributo="valor">Contenido</etiqueta>
```



Donde:

- `<etiqueta>` : Es el nombre de la etiqueta HTML, como `div` , `p` , `a` , `img` , etc.
- `atributo="valor"` : Son los atributos y valores de la etiqueta, como `href` , `src` , `class` , `id` , etc.
- `Contenido` : Es el contenido dentro de la etiqueta, como texto, imágenes, enlaces, listas, etc.

- Se pueden añadir estilos CSS en línea o en un archivo externo para personalizar la apariencia.

### 2.11.2. Ejemplo de HTML en Markdown

A continuación, se muestra un ejemplo de cómo integrar código HTML en un documento Markdown:

```
<div style="background-color: lightblue; padding: 10px;">  
  <h2 style="color: navy;">Título de la sección</h2>  
  <p>Este es un párrafo con <a href="https://www.example.com">un enlace</a>  
    
</div>
```

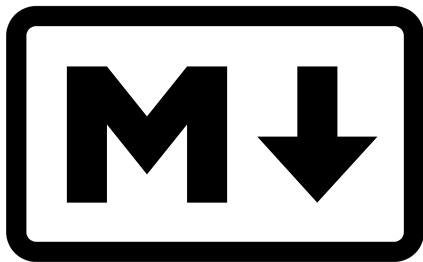


Renderizado:

## Título de la sección

---

Este es un párrafo con [un enlace](#).



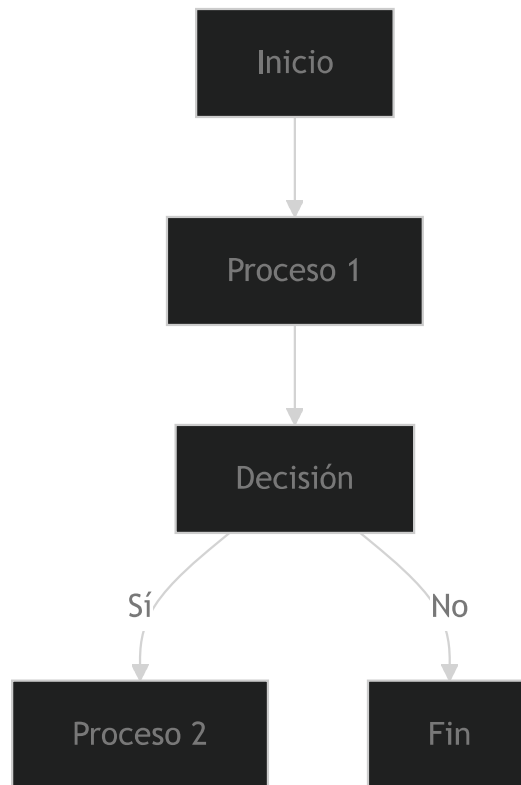
### 2.12. Markdown extendido

Además de su sintaxis básica, Markdown admite extensiones que permiten añadir funcionalidades avanzadas como **diagramas**, **código resaltado** y **fórmulas matemáticas**. Estas herramientas son especialmente útiles en documentación técnica y científica.

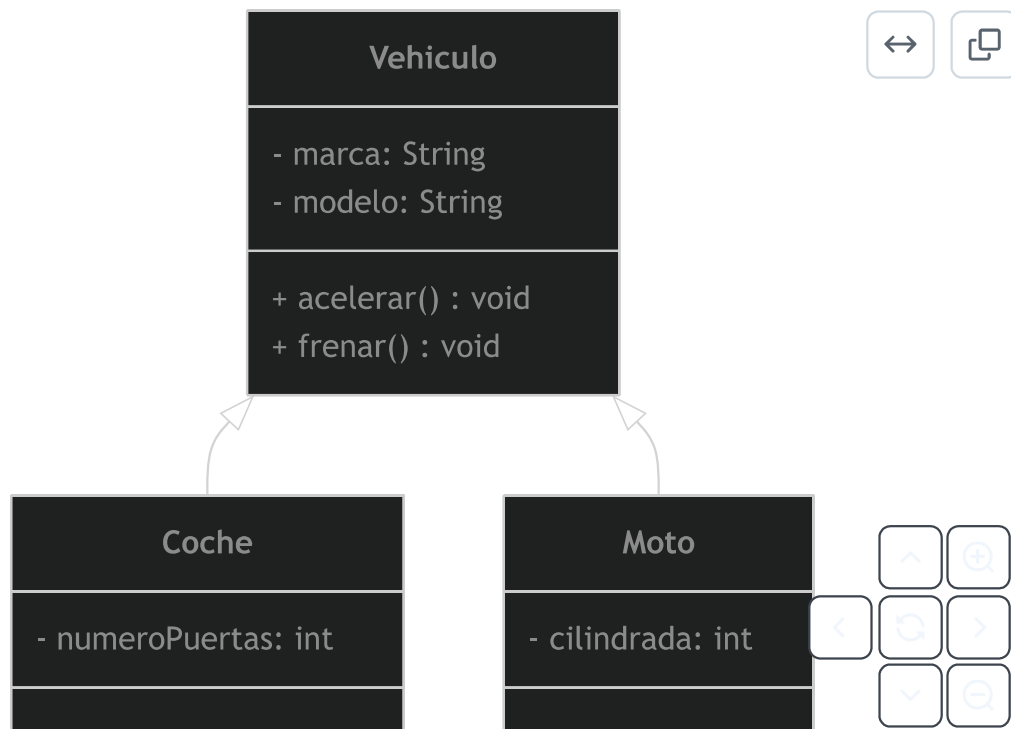
#### 2.12.1. Diagramas con Mermaid

[Mermaid](#) es una librería que permite crear diagramas dentro de Markdown mediante una sintaxis sencilla. Se utiliza en plataformas como GitHub, GitLab y Notion para generar diagramas de flujo, UML, gráficos de Gantt, entre otros.

Ejemplo de diagrama de flujo en Mermaid:



Ejemplo de diagrama de clases en Mermaid:



Los diagramas de Mermaid permiten visualizar estructuras de datos, flujos de procesos y relaciones entre entidades dentro de la documentación. Además, al ser creados mediante texto, pueden modificarse fácilmente sin necesidad de herramientas gráficas complejas.

### 2.12.2. Ecuaciones matemáticas con MathJax

Markdown permite escribir expresiones matemáticas utilizando MathJax y la sintaxis de [LaTeX](#). Esto es útil para representar fórmulas, ecuaciones y símbolos matemáticos de forma clara y precisa.

Ejemplo de ecuación en línea:

La ecuación de la relatividad es  $E=mc^2$ .



Renderizado:

La ecuación de la relatividad es  $E = mc^2$ .

Ejemplo de ecuación en bloque:

$$F = G \frac{m_1 m_2}{r^2}$$



Renderizado:

$$F = G \frac{m_1 m_2}{r^2}$$

También podemos utilizar un **bloque de código** para escribir ecuaciones matemáticas en Markdown:

```
```\math
\int_a^b x^2 dx
```
```

Renderizado:

$$\int_a^b x^2 dx$$

Estas funciones avanzadas amplían las posibilidades de Markdown en entornos científicos y de programación.

## 3. Referencias

- [Markdown Guide](#)
- [GitHub Markdown Basics](#)
- [Pandoc - Conversión de documentos](#)
- [Remark.js - Creación de presentaciones](#)
- [Markdown Cheatsheet](#)
- [Markdown Preview Enhanced - Visual Studio Code](#)

- [Markdown All in One - Visual Studio Code](#)
- [Markdown Lint - Visual Studio Code](#)
- [Mermaid - Diagramas en Markdown](#)
- [MathJax - Ecuaciones matemáticas en Markdown](#)