markdific

THE COMPLETE MARKDOWN REFERENCE

Markdown Cheat Sheet

Every Syntax with Examples • 2026

Every CommonMark and GitHub Flavored Markdown feature, with a copy-paste example, from headings and tables to code, math, and diagrams.

Download Markdific

The most complete markdown cheat sheet on the internet. Every CommonMark and GitHub Flavored Markdown feature, with a working example for each.

Bookmark this page. Or grab the free markdown cheat sheet PDF version to keep it within reach offline.

Markdown syntax at a glance

Element	Syntax
Heading	`# H1` through `###### H6`
Bold	`**bold**`
Italic	`*italic*`
Strikethrough	`~~struck~~`
Link	`[text](https://example.com)`
Image	`![alt](image.jpg)`
Inline code	`` `code` ``
Code block	three backticks, your code, then three backticks
Blockquote	`> quoted text`
Unordered list	`- item`
Ordered list	`1. item`
Task list	`- [ ] to do` and `- [x] done`
Table	`| A | B |` with `| --- | --- |` under it
Horizontal rule	`---`

What is Markdown?

Markdown is a plain-text formatting syntax created by John Gruber and Aaron Swartz in 2004. It uses simple symbols to mark up text so it can be converted into HTML and other formats. The goal was readability. A markdown file should look clean as plain text and only get prettier when rendered.

Today markdown is everywhere. GitHub READMEs, Reddit posts, Discord messages, Notion pages, Obsidian notes, AI chatbot responses, and most documentation sites use it. If you write anything on a computer in 2026, you are almost certainly producing markdown whether you realize it or not.

For more on the format itself, read what is a .md file, how to open and read a .md file, or Markdown vs HTML to understand how the two formats differ. For a complete, element-by-element reference, see the Markdown documentation.

Headings

Use `#` symbols at the start of a line to create headings. The number of hashes sets the heading level, from H1 (largest) to H6 (smallest).

```
# Heading 1
## Heading 2
### Heading 3
#### Heading 4
##### Heading 5
###### Heading 6
```

Alternative Heading Syntax (Setext)

For H1 and H2 only, you can use underline-style headings:

```
Heading 1
=========

Heading 2
---------
```

Most writers stick with the `#` syntax because it works at every level and is easier to scan.

Tips:

- Always put a space between the `#` and the heading text
- Use only one H1 per document (it is the title)
- Leave a blank line before and after headings for safest rendering

Full guide: Headings in Markdown →

Paragraphs and Line Breaks

Paragraphs are separated by a blank line. Just hit Enter twice.

```
This is the first paragraph.

This is the second paragraph.
```

Line Breaks Within a Paragraph

To break a line without starting a new paragraph, end the line with **two spaces** followed by Enter, or use a backslash `\` :

```
First line.
Second line on a new line, same paragraph.

First line.\
Second line using backslash.
```

Both produce a `<br>` tag in HTML. The trailing-spaces method is more traditional but invisible in your editor, which causes confusion. The backslash method is clearer and supported by GitHub Flavored Markdown.

Full guide: Paragraphs → · Line breaks →

Bold, Italic, and Other Text Formatting

These are the formatting basics you will use constantly.

```
**bold text** or __bold text__
*italic text* or _italic text_
***bold and italic*** or ___bold and italic___
~~strikethrough text~~
```

Bold and italic are part of core CommonMark and work everywhere. Strikethrough is a GFM extension covered in more detail below.

When to Use Asterisks vs Underscores

Both `**` and `__` produce bold. Both `*` and `_` produce italic. They are interchangeable, so pick one and stick to it for consistency.

The practical difference shows up inside a word. CommonMark and GFM ignore underscores between letters, so underscores stay literal, while asterisks always trigger emphasis:

```
Stays literal:  this_word_has_underscores
Becomes italic: this*word*has*asterisks
```

Use underscores in prose that contains underscores (like `file_name`) to avoid accidental italics, and use asterisks when you actually want emphasis inside a word:

```
un**believ**ably
```

Full guide: Emphasis: bold and italic →

Lists

Lists come in two flavors: unordered (bullets) and ordered (numbered).

Unordered Lists

Use `-`, `*`, or `+` followed by a space. All three work identically. Most style guides prefer `-`.

```
- First item
- Second item
- Third item
  - Nested item (indent 2 or 4 spaces)
  - Another nested item
    - Deep nested item
- Fourth item
```

Ordered Lists

Use numbers followed by a period. The actual numbers do not matter, markdown renumbers automatically:

```
1. First item
2. Second item
3. Third item
   1. Nested item
   2. Another nested item
4. Fourth item
```

You can also write this and get the same output:

```
1. First item
1. Second item
1. Third item
```

This is a feature, not a bug. It means you can reorder list items without renumbering by hand.

Mixed Lists

You can nest unordered lists inside ordered ones and vice versa:

```
1. Step one
2. Step two
   - Sub-point A
   - Sub-point B
3. Step three
```

Full guide: Lists in Markdown →

Links

Links use square brackets for the visible text and parentheses for the URL.

Basic Inline Links

```
[Markdific](https://markdific.com)
[Visit our blog](https://markdific.com/blog "Optional hover title")
```

Automatic Links

Wrap a URL in angle brackets to auto-link it:

```
<https://markdific.com>
<hello@example.com>
```

Reference-Style Links

Useful when the same link appears multiple times or when you want to keep the prose clean:

```
Read our [cheat sheet][1] or the [blog][2].

Later in the document or at the bottom:

[1]: https://markdific.com/resources/markdown-cheat-sheet
[2]: https://markdific.com/blog/
```

You can also use named references:

```
Check out [Markdific][md-home].

[md-home]: https://markdific.com
```

Linking to Sections (Anchor Links)

GitHub and most renderers auto-generate anchor IDs from heading text (lowercased, spaces replaced with hyphens):

```
[Jump to Tables](#tables)
[Jump to GitHub Alerts](#github-alerts)
```

Full guide: Links → · Automatic links → · Heading IDs →

Wikilinks and Embeds

Obsidian, and some other note tools, add double-bracket wikilinks and embeds on top of standard Markdown:

```
[[Note title]]            link to another note
[[Note title|Alias]]      link with custom text
![[Note title]]           embed another note's contents
![[image.png]]            embed an image
```

These are an Obsidian extension, not standard Markdown, so they do not resolve on GitHub or most static sites. See Markdown in Obsidian for the full set.

Images

Image syntax is almost identical to links, with an exclamation mark in front.

```
![Alt text describing the image](path/to/image.jpg)
![Alt text](path/to/image.jpg "Optional hover title")
```

Image Sizing

Pure markdown does not support image sizing. You have two options:

Option 1: Use HTML directly

```
<img src="path/to/image.jpg" alt="Description" width="500">
```

Option 2: Use platform-specific extensions

Obsidian, for example, supports:

```
![[image.jpg|500]]
```

Reference-Style Images

```
![Markdific logo][logo]

[logo]: /images/logo.png "Markdific"
```

Linking an Image

To make an image clickable, wrap the image syntax in a link:

```
[![Alt text](image.jpg)](https://destination-url.com)
```

For cases pure markdown does not cover, local files, precise sizing, or Base64-embedded images, use the raw HTML `<img>` tag shown above.

Full guide: Images in Markdown →

Code and Code Blocks

Code formatting is one of markdown's most useful features.

Inline Code

Wrap code in single backticks:

```
Use the `print()` function in Python.
```

If your code contains a backtick, wrap it in double backticks:

```
The character `` ` `` is called a backtick.
```

Code Blocks (Fenced)

Use triple backticks for multi-line code blocks. Add a language identifier after the opening backticks for syntax highlighting:

````
```python
def hello_world():
    print("Hello, world!")
```
````

````
```javascript
function helloWorld() {
  console.log("Hello, world!");
}
```
````

````
```bash
npm install markdown-it
```
````

Common Language Identifiers

- `python`, `py`
- `javascript`, `js`
- `typescript`, `ts`
- `bash`, `sh`, `shell`
- `html`, `xml`
- `css`, `scss`
- `json`, `yaml`
- `sql`, `graphql`
- `markdown`, `md`
- `diff` (for git diffs)
- `plaintext`, `text` (no highlighting)

Indented Code Blocks (Legacy)

Indent any line by four spaces or one tab to make it a code block. This works but is less readable than fenced blocks. Avoid it.

Full guide: Code → · Fenced code blocks → · Syntax highlighting →

Blockquotes

Use `>` at the start of a line to create a blockquote.

```
> This is a blockquote.
> It can span multiple lines.

> You can also have a single-line blockquote.
```

Nested Blockquotes

Stack `>` characters:

```
> Outer quote.
>
> > Nested quote.
> >
> > > Deeply nested quote.
```

Blockquotes with Other Markdown

Blockquotes can contain headings, lists, code, and other formatting:

```
> ### A heading inside a quote
>
> - A list item
> - Another item
>
> Some `inline code` and **bold text**.
```

Full guide: Blockquotes →

Horizontal Rules

Three or more hyphens, asterisks, or underscores on their own line:

```
---
***
___
```

All three render as `<hr>`. Use them sparingly. Most modern documents use heading hierarchy instead.

Full guide: Horizontal rules →

Tables

Tables are a GitHub Flavored Markdown extension, not standard CommonMark, but they are supported almost everywhere now.

Basic Table

```
| Header 1 | Header 2 | Header 3 |
|----------|----------|----------|
| Cell 1   | Cell 2   | Cell 3   |
| Cell 4   | Cell 5   | Cell 6   |
```

Column Alignment

Add colons to the separator line:

```
| Left-aligned | Center-aligned | Right-aligned |
|:-------------|:--------------:|--------------:|
| Text         | Text           | Text          |
| Longer text  | Longer text    | Longer text   |
```

- `:---` left-align (default)
- `:---:` center-align
- `---:` right-align

Tips for Cleaner Tables

- Outer pipes are optional but improve readability
- Column widths in the source do not affect output (renderers ignore extra spaces)
- Pipe characters inside cells must be escaped as `\|`
- Standard GFM tables cannot have multiline cells or merged cells; use HTML for those

For a full guide including how to handle complex tables, plus our free markdown table generator, read our complete guide to markdown tables.

Full guide: Tables in Markdown →

Task Lists

Task lists (also called to-do lists) are a GFM extension supported by GitHub, GitLab, and most modern editors.

```
- [x] Completed task
- [ ] Open task
- [ ] Another open task
  - [x] Nested completed sub-task
  - [ ] Nested open sub-task
```

The `[x]` is case-sensitive in some renderers. Use lowercase `x` to be safe.

Full guide: Task lists →

Footnotes

Footnotes are a GitHub extension, and are also supported by MultiMarkdown and Pandoc. They are perfect for citations and asides.

```
This is a sentence with a footnote.[^1]

You can have multiple footnotes.[^note]

[^1]: This is the first footnote content.
[^note]: This is a named footnote with longer content
    that can span multiple lines if indented.
```

Footnotes auto-number themselves and appear at the bottom of the rendered document with backlinks.

Full guide: Footnotes →

Definition Lists

Definition lists are supported by Pandoc, MultiMarkdown, and some other flavors but not core GFM.

```
Term
: Definition of the term.

Markdown
: A lightweight markup language for creating formatted text.
: Created by John Gruber in 2004.
```

Check your target renderer before relying on these. They are inconsistent across platforms.

Full guide: Definition lists →

Abbreviations

MultiMarkdown and PHP Markdown Extra let you define an abbreviation once and have it expand wherever the term appears:

```
The HTML spec is maintained by the W3C.

*[HTML]: HyperText Markup Language
*[W3C]: World Wide Web Consortium
```

This is not part of CommonMark or GFM, so most renderers (including GitHub) show the definition lines as literal text. Use it only where you know the target supports it.

Strikethrough

Wrap text in double tildes:

```
~~This text is struck through~~
```

Rendered: This text is struck through

GFM extension. Universally supported in modern renderers.

Full guide: Strikethrough →

Subscript, Superscript, and Highlight

These three inline marks are extended features. None are part of core CommonMark or GitHub Flavored Markdown, so support varies, but each has a reliable HTML fallback that works anywhere.

Subscript and Superscript

Pandoc, MultiMarkdown, and a handful of other flavors support tilde and caret syntax:

```
Water is H~2~O.
The area is 10 m^2^.
```

Where that syntax is not supported (GitHub and most CommonMark renderers), use the HTML tags instead, they render everywhere:

```
Water is H<sub>2</sub>O.
The area is 10 m<sup>2</sup>.
```

Highlighting

Obsidian and some other tools highlight text wrapped in double equals signs:

```
Markdown makes ==this part== stand out.
```

For renderers that do not support `==`, use the HTML `<mark>` tag: `<mark>highlighted</mark>`.

Full guide: Subscript & superscript → · Highlight →

Emoji

Three ways to use emoji in markdown:

1. Paste Directly

Just type or paste the actual emoji character: 🎉 ✅ 🚀

This works everywhere because emoji are just Unicode characters.

2. Shortcodes (GFM)

GitHub and GitLab support shortcode syntax that converts text codes into emoji:

```
:tada: :white_check_mark: :rocket:
```

These render as 🎉 ✅ 🚀 on supported platforms. Discord and Slack use similar `:code:` syntax but with their own emoji libraries, so the available codes differ. See Markdown in Discord and Markdown in Slack for how each chat app handles formatting.

3. HTML Entities

For maximum compatibility you can also use HTML entity codes, though shortcodes are far more readable.

Full guide: Emojis in Markdown →

HTML in Markdown

You can embed raw HTML inside markdown for cases the syntax does not cover:

```
<details>
<summary>Click to expand</summary>

This content is hidden by default and shown when the user clicks.

</details>
```

Common uses for raw HTML:

- `<details>` and `<summary>` for collapsible sections
- `<sub>` and `<sup>` for subscript and superscript
- `<kbd>` for keyboard keys (renders as Ctrl + C styling)
- `<mark>` for highlighting

- `<img>` with width attributes for sized images
- Tables with merged cells, multiline cells, or complex layouts

Keyboard Keys Example

```
Press <kbd>Ctrl</kbd> + <kbd>C</kbd> to copy.
```

Caveats

Some platforms strip or sanitize HTML for security (Reddit, for example). GitHub allows most HTML but blocks `<script>` and a few other tags. Always test on your target platform.

Full guide: Inline HTML in Markdown →

Comments

Markdown has no official comment syntax, but there are two reliable ways to leave notes in a file that never appear in the rendered output.

HTML Comments

Any renderer that allows HTML also honours HTML comments:

```
<!-- This note is invisible in the rendered page. -->

Visible content continues here.
```

The comment is skipped when the file is rendered. It still exists in the HTML source, though, so treat it as hidden-from-readers rather than truly private.

The Reference-Link Trick

For renderers that strip HTML (some chat apps and sandboxed viewers), use an empty reference-link definition. It produces no output because nothing ever links to it:

```
[//]: # (This is a comment that renders nothing.)

[comment]: # (Another way to write the same thing.)
```

This works in almost every parser because a link definition that is never referenced is simply dropped. It is the most portable way to hide a note.

Escape Characters

To display a markdown character literally, prefix it with a backslash:

```
\* This is not italic \*
\# This is not a heading
\[This is not a link\](not-a-url)
```

Characters you can escape:

\ `` * _ {} [] () # + - . ! |`

Full guide: Escaping characters →

Math Equations (LaTeX)

Math is supported by GitHub, GitLab, Obsidian, Notion, and most modern documentation tools as a renderer feature. Use LaTeX syntax inside dollar signs.

Inline Math

Single dollar signs for inline math:

```
The Pythagorean theorem is $a^2 + b^2 = c^2$.
```

GitHub also supports an alternate inline delimiter that wraps the math in a dollar sign and a backtick on each side, for cases where your math contains characters that conflict with Markdown:

```
Here $`a + b = c`$ uses the alternate delimiters.
```

Block Math

Double dollar signs for block-level equations:

```
$$
\int_{-\infty}^{\infty} e^{-x^2} dx = \sqrt{\pi}
$$
```

Common LaTeX Patterns

```
Fractions:        $\frac{a}{b}$
Square root:      $\sqrt{x}$
Powers:           $x^{2}$
Subscripts:       $x_{i}$
Greek letters:    $\alpha, \beta, \gamma, \pi, \theta$
Sum:              $\sum_{i=1}^{n} i$
Integral:         $\int_{0}^{1} x \, dx$
Matrix:           $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$
```

Renderers vary on syntax. GitHub uses MathJax. Some platforms require KaTeX which has slightly different conventions.

Full guide: LaTeX math →

Mermaid Diagrams

Mermaid is a JavaScript library that turns text into diagrams. GitHub, GitLab, Notion, Obsidian, and most modern markdown renderers support it natively. It is one of the most useful features in modern markdown.

Flowchart

````
```mermaid
flowchart TD
    A[Start] --> B{Is it working?}
    B -->|Yes| C[Ship it]
    B -->|No| D[Debug]
    D --> B
```
````

Sequence Diagram

````
```mermaid
sequenceDiagram
    User->>Browser: Open .md file
    Browser->>Markdific: Render markdown
    Markdific-->>Browser: Return HTML
    Browser-->>User: Show formatted page
```
````

Other Mermaid Diagram Types

- `flowchart` (process flows)
- `sequenceDiagram` (interactions over time)
- `classDiagram` (UML classes)
- `stateDiagram-v2` (state machines)
- `erDiagram` (database schemas)
- `gantt` (project timelines)
- `pie` (pie charts)
- `mindmap` (mind maps)
- `journey` (user journeys)
- `gitGraph` (git branch visualizations)

Mermaid supports far more diagram types than shown here. Which ones render depends on your tool's Mermaid version, GitHub and GitLab keep theirs current, so most diagram types work out of the box.

Full guide: Mermaid diagrams →

GitHub Alerts

GitHub Alerts (also called "callouts" or "admonitions") are a GitHub feature rolled out in late 2023. They render special blockquotes with icons and colors, and are now widely supported across other markdown renderers.

```
> [!NOTE]
> Useful information that users should know.

> [!TIP]
> Helpful advice for doing things better.

> [!IMPORTANT]
> Key information users need to know.

> [!WARNING]
> Urgent info that needs immediate attention.

> [!CAUTION]
> Advises about risks or negative outcomes.
```

The five supported types are `NOTE`, `TIP`, `IMPORTANT`, `WARNING`, and `CAUTION`. Other markdown tools (Obsidian, MkDocs, Docusaurus) use similar but slightly different syntax for callouts. See Markdown on GitHub for how alerts and other GitHub features render.

YAML Frontmatter

Frontmatter is a block of YAML metadata at the top of a markdown file, used by static site generators (Hugo, Jekyll, Astro, Next.js), CMS platforms, and note-taking apps.

```
---
title: "My Article Title"
date: 2026-01-15
author: Jane Doe
tags:
  - markdown
  - tutorial
  - reference
draft: false
description: A short summary for SEO.
---

# Article content starts here
```

Three hyphens open and close the block. Inside, use standard YAML syntax. Common fields:

- `title`
- `date`
- `author` or `authors`
- `tags` or `categories`
- `description` (for meta tags)
- `draft` (boolean)
- `slug` or `permalink`
- `image` (featured image path)

Frontmatter is invisible in the rendered output. It is purely metadata for the system processing the file.

If your frontmatter shows up as literal text in the output, the parser likely does not support it, check that the file starts on line 1 with `---` and has no blank line above it.

Full guide: Front matter →

Markdown Flavors Compared

Markdown has several "flavors" with different features. Knowing which one your target platform uses prevents broken rendering.

Flavor	Where Used	Key Features
CommonMark	Reddit, Stack Overflow	The standardized core. Predictable but limited.
GitHub Flavored Markdown (GFM)	GitHub, GitLab, most modern tools	CommonMark + tables, task lists, strikethrough, autolinks, footnotes, alerts
MultiMarkdown	Academic and technical writing	Adds tables, footnotes, math, citations, metadata
Pandoc Markdown	Pandoc converter	Most powerful. Adds definition lists, math, tables, citations, raw HTML
Obsidian Markdown	Obsidian note-taking app	GFM + wikilinks `[[note]]`, callouts, embeds, tags
R Markdown	R, RStudio, Quarto	Adds executable code chunks for data analysis
Discord / Slack	Chat apps	Custom subsets with platform-specific syntax (spoilers, mentions, emoji codes)

When in doubt, write GFM. It is the closest thing to a universal markdown standard in 2026. For platform-specific behavior, see Markdown in Obsidian and Markdown in Notion.

Full guide: Markdown flavors compared →

Common Mistakes

These trip up writers constantly. Avoid them.

1. Forgetting Blank Lines

Markdown often needs a blank line between elements:

```
This is a paragraph.
- This list might not render
- depending on the renderer.
```

Fix with a blank line:

```
This is a paragraph.

- This list renders correctly.
- Every renderer agrees.
```

2. Mixing Indentation

When nesting lists or adding paragraphs to list items, use consistent indentation (2 or 4 spaces). Mixing tabs and spaces breaks rendering.

3. Trailing Whitespace for Line Breaks

The "two spaces at end of line" trick for soft line breaks is invisible and breaks easily. Use a backslash instead, or just accept that paragraphs are usually what you want.

4. Not Escaping Special Characters

Writing `func_name_with_underscores` in prose is safe in CommonMark and GFM, which ignore underscores between letters, though some older parsers still read them as emphasis. Asterisks are riskier here, since `a*b*c` italicizes. To be safe everywhere, wrap technical terms in backticks for inline code. Other common collisions: `<` and `>` being read as HTML tags, and `|` inside table cells breaking the table layout.

5. Relying on Non-Standard Features

If you use Obsidian wikilinks `[[note]]` and then publish to a static site that does not support them, your links break. Know your target renderer.

6. Tables That Are Too Wide

GFM tables cannot scroll horizontally. If a table is too wide for the page, it overflows. Break wide tables into multiple narrower ones or use HTML.

7. Empty Headings or Headings With Only Formatting

```
## **Bold heading**
```

This works but some renderers strip the formatting from headings or generate weird anchor IDs. Keep headings plain.

Download the PDF cheat sheet

Want this cheat sheet offline? Download the printable PDF version to keep on your desk or share with your team.

Frequently asked questions

What is a markdown cheat sheet?

A markdown cheat sheet is a quick reference that lists every markdown syntax element alongside an example of how to write it, headings, bold and italic text, lists, links, images, code blocks, tables, and more. This page covers all of CommonMark plus the GitHub Flavored Markdown extensions.

How do I make text bold in markdown?

Wrap the text in two asterisks: `**bold text**`. Two underscores (`__bold text__`) also work. For italic, use a single asterisk or underscore; for bold and italic together, use three: `***text***`.

How do I create a table in markdown?

Separate columns with pipe characters and add a row of dashes under the header row to mark it as a table:

```
| Name | Role |
|------|------|
| Ada  | Lead |
```

Or skip the syntax entirely and build one visually with our free markdown table generator.

What is the difference between Markdown and GitHub Flavored Markdown?

CommonMark is the standardized core syntax. GitHub Flavored Markdown (GFM) extends it with tables, task lists, strikethrough, autolinks, footnotes, and alerts. GFM is the most widely supported flavor, so it is the safest default in 2026.

Is markdown the same as HTML?

No. Markdown is a lightweight plain-text syntax that converts into HTML. Markdown is for writing and editing; HTML is for display. Our guide to Markdown vs HTML explains when to use each.

How do I add a link in markdown?

Put the visible text in square brackets followed by the URL in parentheses: `[Markdific](https://markdific.com)`. To turn an image into a link, wrap the image syntax in the same brackets.

How do I add a comment in markdown?

Markdown has no official comment syntax. The most portable method is an empty reference link, `[//]: # (your note)`, which renders nothing. In any renderer that allows HTML, an HTML comment `<!-- your note -->` also works.

How do I create a checklist in markdown?

Use a task list: start each item with `- [ ]` for an open task or `- [x]` for a completed one. Task lists are a GitHub Flavored Markdown feature supported by GitHub, GitLab, and most modern editors.

How do I add an image in markdown?

Use an exclamation mark, alt text in square brackets, and the path in parentheses: `![Alt text](path/to/image.jpg)`. To size an image, use an HTML `<img>` tag with a `width` attribute instead.

How do I write a code block in markdown?

Wrap the code in three backticks on their own lines. Add a language name after the opening backticks for syntax highlighting. For a short snippet inside a sentence, use single backticks around the `code`.

How do I make a table of contents in markdown?

Markdown has no built-in table of contents, but you can build one from anchor links like `[Section title](#section-title)`, using the lowercased heading with spaces replaced by hyphens. GitHub, GitLab, and many editors also auto-generate one.

How do I convert Markdown to Word or PDF?

Use a converter. Markdific has free online tools to convert Markdown to Word, HTML, and more, all running in your browser with no signup.

Related guides and tools

Markdown by platform. See exactly how Markdown behaves in the tools you use every day:

- Markdown on GitHub
- Markdown in Obsidian
- Markdown in Notion
- Markdown in VS Code
- Markdown in Discord
- Markdown in Slack
- Markdown in Jupyter
- Markdown in Confluence
- How to write a README
- Markdown files for AI agents

Convert your Markdown. Free online converters that run in your browser, no signup:

- Markdown to Word
- Markdown to HTML
- Markdown to Text
- Markdown to Image

Key takeaways

That is every markdown syntax you will realistically encounter in 2026. The core takeaways:

- **CommonMark + GFM** covers 95% of real-world markdown usage
- **Tables, task lists, footnotes, math, mermaid, and alerts** are the extensions worth memorizing

- **HTML fallbacks** handle the edge cases markdown does not
- **Frontmatter** is essential for any static site or content system
- **Mermaid diagrams** are one of the most underused features in markdown

The format will keep evolving as AI-generated content floods the web with .md files, but the core syntax has been stable since 2004 and will not change. Learn it once and it pays off for decades.

Want to see your markdown rendered cleanly? Markdific is a fast, dedicated markdown viewer and editor for Mac and Windows. For more, read Markdown vs HTML, browse our markdown resources, or read the full Markdown documentation.

Markdific

Markdific turns raw markdown into clean, readable documents on any device. Open any `.md` file and instantly see proper headings, tables, code, math, and images, then edit with autosave, switch themes, and export to PDF, Word, or HTML.

---

markdific

Markdific is a full-featured Markdown viewer and editor that opens any .md file and renders it instantly, turning raw Markdown into clean, formatted headings, tables, code, math, and images. Edit with autosave, switch themes, and export to PDF, Word, or HTML.

Download Markdific