

# From .txt to .md

---

## Table of contents

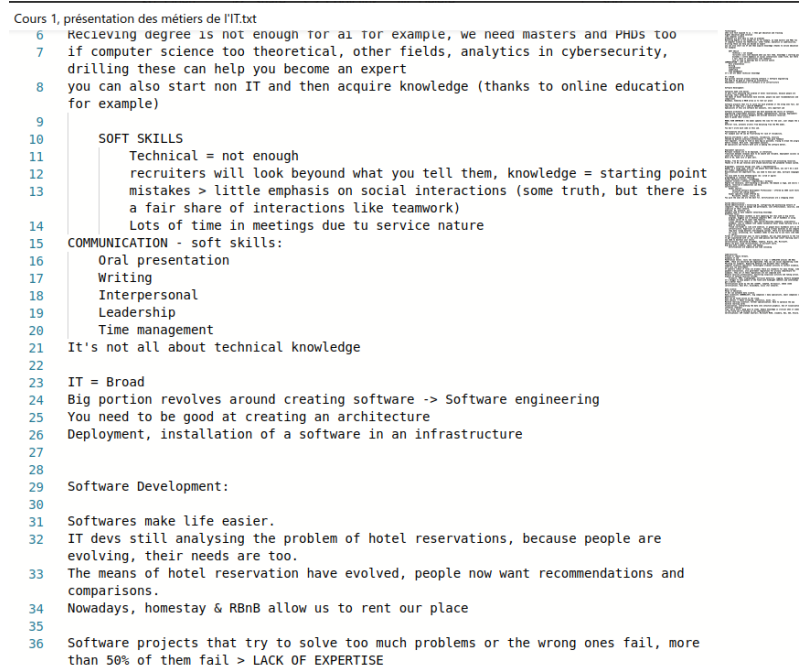
- [From .txt to .md](#)
  - [Table of contents](#)
  - [Markdown for note-taking](#)
    - [The pros of markdown](#)
    - [Markdown here and there](#)
  - [Markdown for documentation](#)
  - [Markdown for blogging](#)
    - [HUGO](#)
  - [Conclusion](#)

## Markdown for note-taking

As a student following a remote formation, note taking plays a crucial part in keeping up with classes. Over my four years so far, my note-taking system has slowly evolved.

I immediately started with plain text, the main issue was keeping a consistent and readable formatting.

I initially started with big blocks of texts on notepad++, using tabulations to indent nested information. And an extra line break whenever a new section came in.



```
Cours 1, présentation des métiers de l'IT.txt
6  recieving degree is not enough for a1 for example, we need masters and PHDs too
7  if computer science too theoretical, other fields, analytics in cybersecurity,
  drilling these can help you become an expert
8  you can also start non IT and then acquire knowledge (thanks to online education
  for example)
9
10  SOFT SKILLS
11  Technical = not enough
12  recruiters will look beyond what you tell them, knowledge = starting point
13  mistakes > little emphasis on social interactions (some truth, but there is
  a fair share of interactions like teamwork)
14  Lots of time in meetings due to service nature
15  COMMUNICATION - soft skills:
16  Oral presentation
17  Writing
18  Interpersonal
19  Leadership
20  Time management
21  It's not all about technical knowledge
22
23  IT = Broad
24  Big portion revolves around creating software -> Software engineering
25  You need to be good at creating an architecture
26  Deployment, installation of a software in an infrastructure
27
28
29  Software Development:
30
31  Softwares make life easier.
32  IT devs still analysing the problem of hotel reservations, because people are
  evolving, their needs are too.
33  The means of hotel reservation have evolved, people now want recommendations and
  comparisons.
34  Nowadays, homestay & RBNB allow us to rent our place
35
36  Software projects that try to solve too much problems or the wrong ones fail, more
  than 50% of them fail > LACK OF EXPERTISE
```

This was quick, but could hardly be parsed, too much of an effort was needed to even differentiate between title and contents.

Sustainable, but hardly readable.

Eventually, I decided to try and change my formatting style to one that's easier to parse. With a clearer separation between sections, titles and contents.

```

Day 6, Introduction to SAN and NAS Storage.txt
62 -----NAS protocols-----
63
64 SMB / CIFS.
65   Server message block, Common Internet File System.
66   Designed for windows, can be accessed by unix based clients through SAMBA.
67
68 NFS.
69   Network File System.
70   Originally for UNIX.
71   Commonly used for VMDS datastores in vmware.
72
73
74 -----SAN protocols-----
75
76 Fibre channel.
77   Terminology.
78     LUN (logical unit number), disk presented to a host.
79     Initiator, client
80     Target, storage system.
81
82   Original SAN protocol, uses dedicated adapters, cables and switches, different
83   than ethernet.
84   Uses FCP to send SCSI commands over the Fibre Channel Network.
85   Very stable and reliable (lossless).
86   Supports 1 to 128 Gbps bandwidths.
87
88   Fibre channel is a different infrastructure from the ethernet one. Server is
89   linked to both network and storage infrastructure.
90
91   FCP uses WWN (world wide names). WWNN (World wide node name) is assigned to a
92   node in the storage network, same WWNN can identify multiple network interfaces
93   of a single network node.
94
95   WWPN (World wide port name), assigned to every individual port on a node.
96   Configured by constructor, globally unique, similar to MAC address. Don't need
97   to manually configure.

```

This formatting made more use of tabulations and was easy to read through, but then, a different set of problems presented themselves.

- The toll of manually formatting each title was too great, making it hard to take notes in real time, I'd be lagging behind every time a new lesson came around, if it was a LinkedIn-learning course, then I would have to pause at the start of every new video, or write every title in advance ;
- I can only indent so many times before I start to fill the available horizontal space. On a single-screen setup, where I had to share space between notepad++ and microsoft teams/LinkedIn learning, I'd often find myself in a situation where the line wrap was only a few words away for each line.

Readable, but hardly sustainable.

Still, I kept using this formatting, it was only at the middle of my second year, that I finally decided to seek a better note-taking method. That's when I eventually tried using markdown.

It was at the start of January 2022 that I started taking notes in markdown, and it's been my format of choice ever since.

## The pros of markdown

Markdown was able to solve all problems I had with note taking, it is both very easy to parse, and also minimalist in its syntax.

The formatting options are simple and intuitive, you do not initially have to worry about font, alignment, or other microsoft word features, but you also get a very robust rule set that allows you to set up a proper document hierarchy and formatting. Such rules include :

- A blank line to separate elements/paragraphs ;
- Title levels using a variable number of # ;
- Bullet point lists and numbered lists ;
- Code blocks ;
- etc...

Markdown is extensively integrated in many parts of the internet and can even be rendered and printed as pdf files.

## Markdown here and there

Once you become aware of markdown's existence, you start to realize that many people before you already found out how convenient it is, and notice it's used in many places.

Here is a [non exhaustive list of the places where markdown is used](#) one way or another.

I haven't tried using obsidian for nota-taking yet, but have heard [lots of praise about it](#).

## Markdown for documentation

If you're a developer, you're probably most familiar with markdown thanks to github, and its trusty README.md file that automatically embeds as a nicely-formatted document in your repository's front page. The more you think about it, the more you realize "who **wouldn't** pick markdown for suh a fitting task?", after all :

- The formatting of .md files plays very nicely with git and the diff ;
- Markdown has very performant web renderers ;
- Markdown has great support for code blocks, and even has syntax highlighting.

Even for local documentation, you can easily find [extensions to render your markdown files as PDF](#) (spoiler alert, all this does is render your markdown as html code before "printing" it to pdf via a headless browser). This whole article was actually written and rendered this way.

Because markdown renders as html, it means you can also use html in markdown (depending on how you're planning to display your file). This allows you to add extra styling to your document. This allows you, for example, to use css to modify the flow of your document, make images floating, restyle titles/paragraphs, rescale images, add page breaks for PDF and printing, etc...

VSCode has great built-in tooling for markdown, allowing you to live-preview your file using `ctrl+shift+v`, and facilitating images embedding (you're able to easily drag and drop, or even paste an image from your clipboard).

So if you're a developer and want to document your repository, go out there and start messing around with your README.md file, if you want to share knowledge while not thinking much about formatting, [make a gist](#), make many gists, make github your personal blog!

...speaking of blogs...

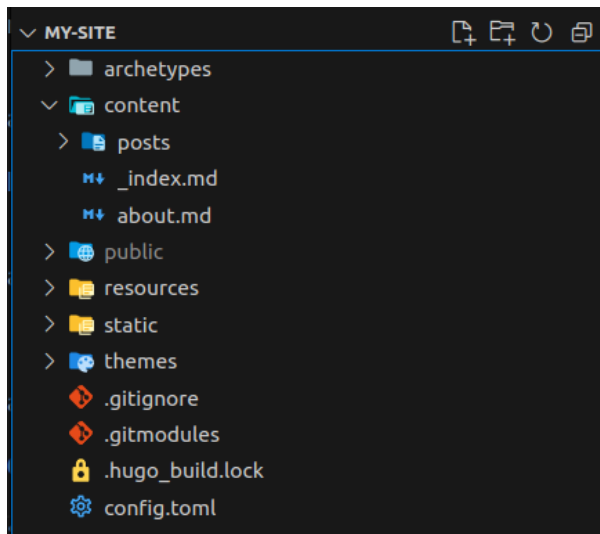
## Markdown for blogging

I will not talk about my views on the centralization of self expression in social media platforms, which are progressively turning into walled garden, as that would be beyond the point of this article, but basically : I, for a long time, had wanted to make myself a personal blog, but as much as I tried, I always got stuck in technicalities. I wasted so much time thinking of the best way to build a website and manage content that I was already worn out and unmotivated to make the content itself.

That was until I learnt that a friend of mine made his blog until HUGO.

## HUGO

Hugo is a framework for rendering static websites, it's built using GO and a content-centric approach.



All the content is managed in the `content` root folder while binaries are in the `public` folder, everything else is Theming and templating.

Basic markdown support is provided out of the box, but new elements can be created using shortcodes :

`themes/your-theme/layouts/shortcodes/audio.html`

```
<div class="container">
  <audio controls style="width: 80%; margin: 0 auto ; margin-bottom:
20px">
    {{- $audio_src := .Get "src" -}}
    {{ with .Get "src" }}<source src="{{ . | relURL }}"
type="audio/mpeg">{{ end }}
    Your browser does not support the audio element.
  </audio>
</div>
```

`content/posts/your-article.md`

```
{{<audio src="/audios/your-audio-file.mp3">}}
```

final result



Because HUGO renders as a static web page, you can easily upload it to any hosting service of your choice, or even github, all with minimal troubleshooting.

## Conclusion

Markdown is overall a very powerful, yet succinct format. The intuitive rule set is easy to remember and allows you to think content-first, while still giving you fine-grained control over the styles with html and CSS.

It integrates and renders very well on the web and on PDF, making it a prime candidate for both documentation and note taking.

Finally many tools and frameworks such as HUGO or obsidian are built on top of Markdown, integrating it into their more comprehensive solution.

All of this is the reason why, whenever I have to jolt down information, I end my new file with `.md`.