Plain Text Writing

This section deals with an aspect of digital sustainability. In this context, digital sustainability is understood to mean that digital products can be represented independently of specific manufacturers.

To explain where the problem lies, we must first distinguish between two file types: text files and binary files.

Text and Binary Files

Text files store their content as plain text. Formatting in such files occurs through control characters (see the ASCII table, specifically the examples of character 9 for a horizontal tab or character 10 for a line break) and through conventions (see the formatting in Markdown or LaTeX).
Accordingly, they can be read in any text editor.
In contrast, binary files are fundamentally bound to programs that can read their standard for creation, editing, and display. Their contents are stored as a sequence of bytes. An example of a binary file is a document created in MS Word.

This distinction shows that text files can fundamentally be displayed on any functioning computer with a text editor. By contrast, binary files usually require special programs that can interpret the respective file format, which can lead to a certain dependency on corresponding software.

Plain Text Formats for Text Editing

The simplest form of a text file is a `.txt` file. This usually involves just text without formatting.
The need for formatted text representations has led to various file formats. These include, for example, HTML files (`.html` or `.htm`: **H**yper **T**ext **M**arkup **L**anguage). This format was originally created, among other things, to display formatted text on a screen.

Another example is LaTeX. LaTeX is also a so-called markup language. However, the target medium of LaTeX is less the screen and more print. Today primarily the Portable Document Format (PDF), which has developed into a quasi-standard.

However, the scope of formatting possibilities in both HTML and LaTeX is so extensive that the barrier to using them is relatively high.

Against this background, a rather minimalistic markup language, Markdown (`.md`), was created. The Markdown language makes all essential formatting options, such as hierarchical headings or setting italicized text, directly available. Where formatting provided by Markdown options are insufficient for one's needs, HTML or LaTeX tags can be used as a fallback in many cases.

An overview of formatting commands in Markdown can be found on the Markdown Cheat Sheet website. Important formatting in LaTeX (mathematical formulas) is explained on the Wikibooks LaTeX/Mathematics website.

Converting Markdown to PDF

Markdown has the clear advantage of being readable in any text editor. Unfortunately, raw Markdown is not very aesthetic. Furthermore, most texts must be converted into a PDF for the recipient.

The following text provides a step-by-step guide for the conversion of a Markdown file to PDF.

Preparing for Conversion

The program Pandoc can be used for this conversion. Pandoc can convert Markdown into countless formats. The exact procedure is explained step-by-step below.

1. **Install Pandoc**
   The installer for Pandoc can be downloaded from the Pandoc website. Follow the link "Download the latest Installer". For Windows, download the file with the extension `.msi`.
   The downloaded installer can then be executed for installation with a double-click.
2. **Install LaTeX**
   For Pandoc to convert a Markdown document into a PDF, a LaTeX installation is required on the computer. LaTeX is available for Windows in various versions. The MiKTeX variant is the simplest to install. The corresponding installation package can be downloaded from the MiKTeX website.
   The downloaded installer can then be executed for installation with a double-click.
3. **Use Pandoc**
   Pandoc is a command-line program. This means there is no graphical user interface for using Pandoc. Pandoc is easiest to use if a terminal is opened in the folder containing the file to be converted.
   The conversion is performed with the following command:

```
pandoc -o output.pdf input.md
```

   `output.pdf` and `input.md` must be adapted to your own filenames. Alternatively, other file formats, such as `.docx` or `.html`, can be chosen as target formats. For a conclusive list of all possible target formats, please refer to the Pandoc website.

Specific Formatting of the Output

The version shown above leads to a cleanly structured PDF. However, essential document parts such as a title page or table of contents are missing.

For these advanced formattings, the necessary information is either prefixed to the document in a special header or passed to the conversion command in a separate document. The second variant will be presented here.

The document with the formatting information is a so-called YAML document (`.yaml`). YAML stands for **Y**et **A**nother **M**arkup **L**anguage. The document can have any name. Here in the example, it is called `format.yaml`. The following listing shows a formatting configuration with the author's name, the title, and other details. The meaning of the individual entries is largely self-explanatory.

```yaml
from: markdown                          # Source format
to: pdf                                 # Target format
standalone: true                        # Forces standalone document
pdf-engine: xelatex                     # Selects a specific pdf-engine
output-file: output.pdf                 # Name of the output file
metadata:                               # Document-specific details
  title: "Guide"                        # Title
  author: "Jacques Mock Schindler"      # Author
  lang: en                              # Language of the document (de-CH
for Swiss German)
  fontsize: 11pt                        # Font size
  geometry: left=2.5cm, right=3cm       # Margins
  date: "January 9, 2026"               # Date
  toc: true                             # Creates a table of contents
  number-sections: true                 # Numbers the headings
```

Indentation is important for the syntax in the YAML document. It consists of two spaces.

If the file `format.yaml` is intended to control the PDF format, the command for converting the Markdown document into a PDF is:

```
pandoc --defaults=format.yaml input.md
```

If the name of the newly created PDF document is to be defined individually with each command call, the line `output-file: output.pdf` can be be omitted from the YAML file. In that case, the conversion command is:

```
pandoc --defaults=format.yaml -o output.pdf input.md
```

If reliance on external LaTeX packages is necessary for complex formatting, these are loaded via a separate `.tex` document. This document can also fundamentally be named anything. For this example, the file is referred to as `header.tex`. The content of this file is to be formatted as follows:

```
\usepackage{caption}
\usepackage{booktabs}
```

The conversion command is now:

```
pandoc --defaults=format.yaml -H header.tex input.md
```

The CTAN Comprehensive TeX Archive Network website provides information on what LaTeX packages are available.