# AsciiDoc Sample Document

## Table of Contents

# 1. About AsciiDoc

AsciiDoc is a superset of Markdown with some modifications. Like markdown, documents are written in plain text (extension .adoc) and rendered to an output format (html, pdf, docx, etc.), if needed. Having the source in plain text allows easy version control (git) of AsciiDoc documents. Plain text also allows choosing your favorite text editor.

AsciiDoc is intentionally as simple as possible to enable quick learning and to allow focus on writing. You can try it live in your browser. It has been developed for publishing all kinds of content to the web, to books, to manuals, and more. See an example document in plain and rendered format, respectively.

There is a wide variety of AsciiDoc projects and widespread support for the AsciiDoc format. Prominent examples include

- O'Reilly books
- The git book (HTML and printed)
- Khronos documentation
- GitHub and GitLab

AsciiDoc and git are open source, therefore they make MPEG independent of a company's rule.

AsciiDoc supports automatic conversion to html, pdf, docx (Word), and other formats. Conversion can be done with one command. Html and pdf are the main target formats, they support perfect conversion. Conversion to Word is not perfect, but good: general style, sections, images, math, references are all correct.

# 2. AsciiDoc Demo

## 2.1. Text highlighting

**bold** and *italic* work as expected and ***can be combined***.

```
# Comment
for i in range(10):
    print(f"helo, {i}")
```

inline source code `print("stuff")` is like this.

Testing console highlighting

```
# Comment
cp -r file.txt
find -name "*.txt" | xargs echo
```

c++ code highlighting test

```
// Comment
int i{0};
if(element == 0){
    return 0;
}
```

testing note taking:

**NOTE** this is a note

Testing super$^{script}$, works as well as sub$_{script}$.

## 2.2. References

References to Section 2.4.3, Figure 1, and Table 1.

## 2.3. Image

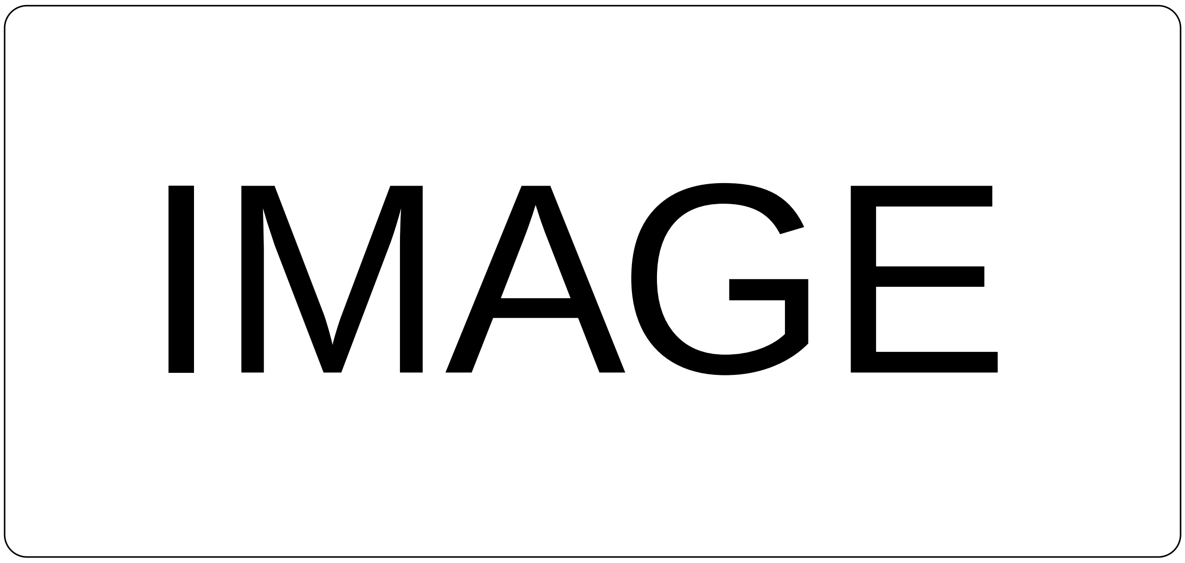

*Figure 1. My image caption*

## 2.4. Lists

And here comes an unnumbered list:

- one item
- another item
- yet another one
    - nested item with numbered items
        1. asdf
        2. aser
            a. bla

Enumerated lists can of course stand alone:

1. asr
2. awer

### 2.4.1. Links

https://asciidoctor.org - automatic!

Asciidoctor

Embedding a YouTube-Video:

► https://www.youtube.com/watch?v=4LOB3WeOUJk *(YouTube video)*

### 2.4.2. Table Test

Note that you can easily do multi-row and multi-column cells

*Table 1. CTC Sequences*

| Usage | ID | Name |
|---|---|---|
| Mandatory | CG-A | ClassroomVideo |
| | CG-B | Museum |
| | CG-O | Fan |

### 2.4.3. Text replacements

© Creators

Do first thing → then do second thing

### 2.4.4. Math

See https://asciidoctor.org/docs/user-manual/#mathematical-expressions. You need to set `:stem:` in the document's header to activate this. This is rendered correctly everywhere, except by GitHub's HTML preview. $\sqrt{4} = 2$, another trye is $H_2O$. A matrix can be written as $\begin{bmatrix} 1 & 2 & 3 \\ a & b & c \end{bmatrix}$.

Block stem:

$$\sqrt{2} = 5$$

# 3. Alternatives to AsciiDoc

AsciiDoc has been deemed the best option out of a pool of candidates. The most promising alternatives are presented in this section.

## 3.1. Microsoft Word Online

Switching from shared .zip files to shared documents on OneDrive or Sharepoint would already significantly improve collaboration and solve some inefficiency. The group has worked with word, so this wouldn't require learning new tools. However, collaborators from some of Microsoft's competitors, such as Apple, are not allowed to use Word Online. This alone prevents usage of Word Online. In addition, we found significant drawbacks when evaluating word online for collaborating on standards, described in this section.

Editing a document in Word online and simultaneously in Word desktop immediately causes the versions to get out of sync and merge conflicts arise. The fundamental problem is the discrepancy between Word online and the Word desktop application: Word online offers better collaboration support, with simultaneous edits and live synchronization from many people. Word desktop lacks this interactivity but offers many more editing features and a better version comparison view.

When editing a document in Word online, the program decides when to save the document. This does not give meaningful versions. Word online offers document versioning, but no version control. This has severe consequences: the document version history does not enable the user to search for meaningful commit messages, meetings, or discussions relating to a proposal. These searches are currently enabled by manual file versioning but will disappear when using one online file. There is also no straightforward way for determining who authored a section or paragraph, or to get the change history of a section.

From a technical perspective, setting up Word Online is not trivial: the documents may not be publicly accessible, so every collaborator needs a Microsoft account. In general, this would add yet another platform to the already fragmented MPEG infrastructure. Reusing GitLab would avoid this.

On a related subject, Word Online lacks integration into MPEG's existing GitLab infrastructure Thus, no straightforward linking to issues, comments, or meetings is possible.

In contrast to plain text alternatives, Word Online doesn't solve formatting issues: content is not separated from style, so pasting something in the wrong format into the document is possible. This is not an issue for the plain text AsciiDoc, where styling is applied when converting to the output format.

In general, given that Word is proprietary software and Word files are written in a proprietary format, it lacks openness and interactivity. There is no straightforward or automated way of publishing Word documents to HTML or to the web. Providing easy access to output documents is extremely important today and will gain more importance as the world gets ever more connected through the web. Additionally, AsciiDoc output documents allow to link directly to sections within a document, which simplifies sharing and collaborating. This is not possible with Word. Looking into more advanced usage, plain text tools such as AsciiDoc enable us to apply tools such as automatically verifying implementations against documents, which is once more not possible with Word. We will continue discovering new usages of plain text documents once we switch to them. Openness and interactivity with other tools are key for a future proof and professional way of working. However, these are not given or impeded by Word.

Finally, having multiple simultaneous proposals in a shared Word online document lacks separation and causes discussions to interfere. When searching the history of a document, this also makes it difficult isolate a discussion relating to a specific proposal.

In general, transitioning to Word Online would be a conservative choice considering the past. It doesn't tackle many of the current issues and introduces additional ones. AsciiDoc is a forward-looking choice that resolves many problems and prepares MPEG for the future.

## 3.2. reStructuredText

The syntax concept of reStructuredText (reST) is similar to AsciiDoc's. However, some of AsciiDoc's syntax design decisions are more reasonable than reST's. AsciiDoc is clearly influenced by modern syntax such as Markdown. Two examples for saner syntax are headings and tables. Heading level n is set by prefixing the equals sign in AsciiDoc n times. reST requires to underline headings with characters from a special set, and heading level is determined by the succession of underlinings. Tables in AsciiDoc are simpler to write as well as more powerful compared to reST.

AsciiDoc has also better toolchain support. With AsciiDoc, the conversion and preview tools come from one project and are well synchronized and working. To get the reST toolchain to work, lots of manual configuration was required. reST documents are rendered by GitLab, but not their math equations.

## 3.3. Markdown

Markdown is extremely widespread and simple to use. However, its features are lacking essential layout requirements from our group. For instance, Markdown does not support math, image captions, image references, merged cells in tables. These things can only be done by injecting HTML into Markdown. HTML, however, is much too complex for daily use by editors.

## 3.4. Others

LaTeX, HTML, and Org-mode require too much boilerplate code to write and are difficult to learn. Wacko wiki markup has insufficient functionality.