AsciiDoc Versus Markdown

Dan Allen

The most compelling reason to choose a lightweight markup language for writing is to minimize the number of technical concepts an author must grasp in order to be immediately productive. In other words, the goal is to be able to *write without friction*.

A popular choice is Markdown. (At least, that's what you call it in the beginning). The main advantage of Markdown lies in its primitive syntax: its manual and cheatsheet are one and the same. But this advantage is also its greatest shortcoming.

As soon as authors need something slightly more complex (tables, cross references, footnotes, embedded YouTube videos, etc.), they find themselves diving directly into HTML or seeking out alternative implementations. Markdown has become a maze of different implementations, termed "flavors", which make a universal definition evasive.

NOTE	The IETF has declared "there is no such thing as "invalid" Markdown." See This Is Markdown! Or: Markup and Its Discontents.

Here's how the story inevitably goes. You start out with Markdown. Then it's Markdown + X. Then Markdown + X + Y. And down the rabbit hole you go. What's worse, X and Y often take the form of HTML, unnecessarily coupling content with presentation and wrecking portability. Your instinct to choose Markdown is good. There are just better options.

AsciiDoc presents a more sound alternative. The AsciiDoc syntax is more concise than (or at least as concise as) Markdown. At the same time, AsciiDoc offers power and flexibility without requiring the use of HTML or "flavors" for essential syntax such as tables, definition lists, admonitions (tips, notes, warnings, etc.) and table of contents.

It's important to understand that AsciiDoc was initially designed as a plain-text alternative to the DocBook XML schema. AsciiDoc isn't stuck in a game of whack-a-mole trying to satisfy publishing needs like Markdown. Rather, the AsciiDoc syntax was explicitly designed with the needs of publishing in mind, both print and web. If the need arises, you can make full use of the huge choice of tools available for a DocBook workflow using Asciidoctor's DocBook converter. That's why mapping to an enterprise documentation format like DocBook remains a key use case for AsciiDoc.

And yet, AsciiDoc is simple enough to stand in as a better flavor of Markdown. But what truly makes AsciiDoc the right investment is that its syntax was designed to be extended as a core feature. This extensibility not only means that AsciiDoc has a more to offer, with room to grow, it also fulfills the objective of ensuring your content is maximally reusable.

Comparison by example

The following table shows the AsciiDoc syntax as it compares to Markdown. Since AsciiDoc supports a broader range of syntax than Markdown, this side-by-side comparison focuses mainly on areas where the syntax overlaps.

Table 1. AsciiDoc language features as they compare to Markdown

Language	Feature	Markdown	AsciiDoc

Bold (constrained)	`**bold**`	`*bold*`
Bold (unconstrained)	`**b**old`	`**b**old`
Italic (constrained)	`*italic*`	`_italic_`
Italic (unconstrained)	*n/a*	`__i__talic`
Monospace (constrained)	`` `monospace` ``	`` `monospace` ``
Monospace (unconstrained)	`` `m`onospace ``	`` ``m``onospace ``
Link with label	`[Asciidoctor](https://asciidoctor.org)`	`https://asciidoctor.org[Asciidoctor]`
Relative link	`[directory structure](../structure)`	`link:../structure[directory structure]`
File link	`[get the PDF]({% raw %}{{ site.url }}{% endraw %}/assets/mydoc.pdf)`	`link:{site-url}/assets/mydoc.pdf[get the PDF]`
Cross reference	*n/a*	`<<installation>>`
Inline Image w/ Alt Text	`![Logo](/images/logo.png)`	`image:logo.png[Logo]`
Block Image w/ Alt Text	*n/a*	`image::logo.png[Logo]`
Section		

heading*	`## Heading 2`	`== Heading 2`

Blockquote*	`> Quoted text.`
	`>`
	`> Another paragraph in quote.`
		`____`
		`Quoted text.`
		
		`Another paragraph in quote.`
		`____`

Code block*	```` ```java ````
	`public class Person {`
	`}`
	```` ``` ````
		`[source,java]`
		`----`
		`public class Person {`
		`}`
		`----`

Unordered list	`* apples`
	`* orange`
	`  * temple`
	`  * navel`
	`* bananas`
		`* apples`
		`* oranges`
		`  ** temple`
		`  ** navel`
		`* bananas`

Ordered list	`1. first`
	`2. second`
	`3. third`
		`. first`
		`. second`
		`. third`

Thematic break (aka horizontal rule)*	`***`
	
	`* * *`
	
	`---`
	
	`- - -`
	
	`___`
	
	`_ _ _`
		`'''`

Typographic (aka “Smart”) Quotes	Enabled through an extension switch, but offer little control in how they are applied.	`One of Jekyll's best aspects is that it is "`blog aware`".`
Document header	*Slapped on as front matter*	

`---`
`layout: docs`
`title: Writing posts`
`prev_section: defining-frontmatter`
`next_section: creating-pages`
`permalink: /docs/writing-posts/`
`---`
	Native support!
	`= Writing posts`
	`:awestruct-layout: base`
	`:showtitle:`
	`:prev_section: defining-frontmatter`
	`:next_section: creating-pages`	

Admonitions	*n/a*	`TIP: You can add line numbers to source listings by adding the word `numbered` in the attribute list after the language name.`
Sidebars	*n/a*	**Lightweight Markup** Writing languages that let you type less and express more.
Block titles	*n/a*	*Grocery list* • Milk • Eggs • Bread
Includes	*n/a*	`include::intro.adoc[]`
Custom CSS classes	*n/a*	`[.path]_Gemfile_`

* Asciidoctor also supports the Markdown syntax for this language feature.

You can see that AsciiDoc has the following advantages over Markdown:

- AsciiDoc uses the same number of markup characters or less when compared to Markdown in nearly all cases.
- AsciiDoc uses a consistent formatting scheme (i.e., it has consistent patterns).

- AsciiDoc can handle all permutations of nested inline (and block) formatting, whereas Markdown often falls down.
- AsciiDoc handles cases that Markdown doesn't, such as a proper approach to inner-word markup, source code blocks and block-level images.

NOTE Certain Markdown flavors support additional features, such as tables and definition lists. However, since these features don't appear in plain Markdown, they are not included in the comparison table.

Asciidoctor, which is used for converting AsciiDoc on GitHub and GitLab, emulates "the good parts" of the Markdown syntax, like headings, blockquotes and fenced code blocks, making migration from Markdown to AsciiDoc fairly simple.