# Writing at the Command Line

Jake Zimmerman

November 20, 2016

# We have lots of things to write.

We…

- write essays.
- typeset math & computer science homework.
- make presentations.
- document our code.
- write blog posts.
- …

Let’s take a look at how we make documents right now.

# Microsoft Word

Pros:

- …

Cons:

- bloated
- expensive
- finicky
- proprietary
- can’t use Vim!

# LaTeX

Pros:

- plain text format
- open source compiler
- open source PDF viewers
- Crazy powerful
- “Least bad” solution for complex math
- Easy to generate PDFs

Cons:

- Doesn’t optimize for the “common case”
- Lots of boilerplate
- Somewhat verbose

# HTML

Pros:

- plain text format
- open source web browsers
- Website are super portable

Cons:

- Same drawbacks as LaTeX:
  - Doesn’t optimize for the “common case”
  - Lots of boilerplate
  - Somewhat verbose

What would an “ideal” solution be?

# We want…

- an open format
- to use open source software
- to optimize for the “common case”
- something concise (not verbose)
- to make PDFs
- to write content for the web

# Markdown provides a “happy medium”

Pros:

- For simple documents, the syntax is simple
- Completely open
- Converters for generating HTML, PDFs, and more

Cons:

- Doesn’t handle complex documents well
- No native way to handle math
- Syntax extensions differ with each implementation

# Getting Started with Markdown

# There are lots of different implementations

- Markdown.pl
- Github-Flavored Markdown
- Pandoc Markdown
- CommonMark
- … and many more

Each supports the core syntax to some degree, and usually has some number of syntax extensions, which each implementation supports to varying degrees.

# Learning Markdown Syntax

If you only look at one guide:

- CommonMark

If you want to start comparing implementation differences:

- GitHub Flavored Markdown
- Markdown.pl
- Pandoc Markdown

# Good Markdown Style

For readability:

- Put two newlines before a top-level heading
- Put one newline after any heading
- Hard wrap your lines (72 or 80 characters per line)
  - See Vim's `gq` operator and `textwidth` setting
- Use reference-style links ( `[link][href]` or `[link]` )

To avoid confusion:

- Use hyphens for lists
- Use underscores for emphasis

For consistency:

- Prefer spaces to tabs
- Use ATX-style headings ( `#` , `##` , …)
- Use headings over bolded text on its own line

# Using Markdown Online

# GitHub uses Markdown for basically everything:

- Your README
- Pull request and issue descriptions
- Comments
- Markdown files committed to your codebase
- Your repo's wiki
- …

*(Demo)*

# StackEdit.io is a great Markdown editor

- `https://stackedit.io`
- Open source
- Web based
- Good for personal note taking

# Writing at the Command Line

# Previewing GitHub Markdown with `grip`[1]

To use:

- Install (`pip install grip`)
- Write a `README.md` file
  - … at the command line!
- Run `grip` from the same directory

[1] GitHub Readme Instant Preview

# Making PDFs with Markdown

- Install `pandoc` (i.e., `brew install pandoc`)
- Install LaTeX locally (i.e., `brew cask install mactex`)
  - … installing LaTeX takes a long time!
- Write a file like `written-sols.md`
- Compile with `pandoc`:
  - `pandoc written-sols.md -o written-sols.pdf`

# Pandoc is a veritable Swiss Army Knife

Pandoc supports formatting your Markdown in lots of ways.

I've compiled all the starter files I have lying around into one place:

- `https://github.com/jez/pandoc-starter`

Each comes with a README and a Makefile to compile the sample document.

# Jekyll lets you create blogs with Markdown

```
$ gem install bundler
$ jekyll new my-awesome-site
$ cd my-awesome-site
$ bundle exec jekyll serve
```

You can write your page templates as HTML once, then write each post in Markdown.

See more at https://jekyllrb.com.

# Some extra workflow hacks

- You can run `:make` in Vim to access your Makefile
- You can run `:make view` to compile the target named `view`
- You can add this line to your vimrc to abbreviate this:
  - `command! WV w | make view`
  - (i.e., save the file, then run `:make view`)
  - to use: `:WV`
- You can have Vim and your PDF previewer open in halves of your screen

(More detailed instructions)