MarkDown Thesis Framework

Jan SOBĚSLAV

2018-2019

Proclamation

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla aliquet erat justo, sed finibus nisi interdum tempor. Sed sit amet ullamcorper mi. Suspendisse purus est, vehicula et dapibus nec, gravida non odio. Aenean tincidunt, dui at ultrices vulputate, turpis felis elementum nulla, et mollis enim nisl ut odio.

Abstract

Lorem Ipsum dolor sit amet

Keywords

markdown

Table of contents

List of figures

List of tables

Overview and philoshophy

As stated in the readme, this framework is a tool that utilizes Pandoc[1] to create beautiful standalone ebooks and documents from multiple MarkDown[2] markup files. The text contents are written in plain text, and can therefore be tracked by version control system, like Git, and before exporting to pdf or epub, the MD code is translated to, and interpreted as LaTeX[3].

The main reasons, why I decided to try this approach, instead of continuing to use any office document processing tool, were following:

- Modular nature: I wanted to divide a lengthy document into more managable set of files - chapters. That way, I can see two chapters side by side and modify them parallelly without scrolling all the time (I figured out, that such feature would be invaluable, only after writing a 13k words bachelor thesis)
- Version control: I thought it would be useful to see different versions of the document, and changes made between them. Moreover, remote VCS makes perfect backups, and even makes possible collaboration way easier
- Clearer style: Since MD markup is quite limited, I like to think of it as of a benefit: I hope it will give the document more unified, more readable look

Features checklist

There are certain features this framework in general, and other tools I use, must provide for me. Some features must be satisfied by recommended text processing program, other by the MD-to-PDF compilation tool. It's important to note, that theese points are my personal requirements, and I present them only to give an idea on the philosophy behind the project, and what could it give you.

As noted further in the docs, the framework relies on Pandoc[4] tool, and I personally chosed Haroopad[5] as text processor, which suits me the most.

My text processor requirements:

[1] https://pandoc.org
[2] https://daringfireball.net/projects/markdown/
[3] https://www.latex-project.org
[4] https://pandoc.org
[5] http://pad.haroopress.com/user.html

- [x] Live compilation result preview
  - I need to see the result of MD processing as I write. I prefer two-pane layout, instead of WYSIWIG, and this requirement is satisfied by Haroopad
- [x] Ability to view multiple files simoutanously
  - Haroopad works in windowed mode, and therefore it is possible to arrange the files for convenience
- [x] Multiple monitor support
  - Again, windowed mode enables spreading the work on multiple monitors. This is feature, which might not be able to accomplish with a single-window IDE (which could, however, at least support viewing multiple files at the same time)
- [ ] Tabbed interface of opened documents
  - Haroopad does not have this feature. Instead, it works in windowed mode, which is not perfect to me personally, but it is satisfactory enough
- [x] Diacritics support (UTF8)
  - Haroopad works well with UTF-8

Framework requirements and goals:

- [x] Git support
  - MD files are basically plain-text and therefore perfectly compatible with any VCS
- [x] Export: Export to PDF (EPUB), merge of multiple files
  - Pandoc does this, and much more
- [x] Content: images, code preview
  - Works like a charm, as seen in example chapter
- [ ] Content: tables
  - It's difficult to maintain the tables in plain text, but it's possible, as seen in example chapter. I intend to add CSV or XLS import function.
- [x] Organization: Table of contents, list of tables, list of figures, bibliography (References)
  - Another few of nice features PandocLaTeXprovides
- [ ] Organization: pagebreaks, heading numbering
  - Hasn't been tested yet
- [x] Style: Template support
  - There are templates for both document and chapter
- [ ] Custom formatting / style
  - Templates are working, but they yet need some tweaking and cleaning. Once that had been done, it should be easier to add more formating options.

- [ ] Formalized FW
  - In the end, I want to present a publishable version of the project, with accurate docs and all features working and tested
- [ ] Generalize the script
  - I want to make it possible to use the script with multiple documents. Perheaps, some GUI, or file watchers may be usefull, too.

News

The script has been generalized so it can be used with multiple documents.

I have successfully used the solution for compiling my 101 pages long thesis, and I was very satisfied with the process and the results. I am happy to see that the effort has actually paid off, as I am certain this approach saved me much time in revisions and visual formatting.

I will now try to finish the features I designed while working on it (namely shorthands for certain useful LaTeX commands, which do not have MarkDown counterpart). Furthermore, there are still other features pending (namely direct spreadsheet import and better looking LaTeX templates).

Lastly, I am uncertain, how would I make the solution user friendly, and available to others. GUI app is way beyond scope of this project, but I was thinking about the posibilities of packing the code with PHP interpreter into single executable file. Nevertheless, this does not have any priority in the development.

- v0.0.6.0 April 2019

As I returned to the project after a while, I realized that I need to rewrite it, in order to be able to continue with more complex features.

Since I don't have strong knowledge of scripting languages like bash or Windows CMD, I decided to go on with the one language I am most familiar with, PHP.

- v0.0.5.0, October 2018

I made it possible to declare a variable within chapter file itself, which overwrites global declarations, and can be used in that very file (as can global variables be used, naturally). I will update the docs with next commit.

I need that functionality because of the new `template_chapter` file, a template that wrapps every chapter to get more control over the result formating. I want to use variables in this template as I do in `template_document`.

Once this feature has been done, I may move some variables (notably the ones related to table of contents, and bibliography) to separate chapters, so that `metadata_content` and `metadata_style` would only contain variables, that affects the whole document, regardless of its content. I also want to remove the `header-includes` variables, since they contain TeX declarations that may be incorporated into the document template, having only its control variables present in the YAML (I would be user friendly, having variables like `break-lines-after-chapter: true` instead `after-includes: \pagebreak`).

Finally, another feature, that I have in mind, is using CSV or XLS tables as resource included in the file much like an image. That's important, since noone wants to write table in plain text, does someone?

Another minor change that had been done is rewritting the `templates` folder structure in a way, that makes it possible to store multiple custom templates for different document types. Another step in that direction will be generalizing the script for use with multiple documents (it is possible to simply copy the framework for each document, but that does not seem right).

- v0.0.4.0, March 2018

I realized, that compiling the MD template to LaTeX may not be optimal, and I will probably try to tweak the LaTeX tamplate instead. My goal is to make all the settings done by YAML variable declarations, so that I wouldn't have to touch LaTeX every time.

Moreover, when writing the example section, I realized, that writing tables in MD isn't user friendly at all, and I would like to seek another solution.

- v0.0.3.0, February 2018

I have just finished the first working verion of the framework, and will test it on a real example soon. When that happens, I will probably update the tool with some fixes, or additional options.

I am quite happy with the results so far, with the only exception being the template format. I wanted the template to be written in MD as well and to be translated to LaTeX during the compillation, but there were errors, and that's why I left the template in LaTeX. I will try to look into it later.

- v0.0.1.0, February 2018