Krambook

Write Books like an `UB3R 1337` *(Hacker)*

Yann Esposito

November 8, 2010

Chapter 1

Write Books like a Hacker

Quality and scalability of LaTeX *&* readable as `markdown`.

Idea provide macros for `markdown` then transform the text in LaTeX and generate a `pdf` file.

Why not using LaTeX directly? Simply because LaTeX is verbose and full of backslashes. To prove my point, simply compare a LaTeX and a `markdown` file.

```
%--- LaTeX source file ----
\documenttype{article}
\usepackage[utf-8]{inputenc}
\usepackage{fontenc}
\usepackage{amsmath}

... % This is the ritual header

\begin{document}
This is a test file.
I begin by making a list of bullet:
\begin{itemize}
\item the first point is
    \LaTeX is a bit verbose
\item the second point is
    \Latex has \textem{more} \textbackslash{} than Markdown
\item I believe you understood now.
\end{itemize}
\end{document}
```

```
I begin by making a list of bullet:

- the first point is LaTeX is a bit verbose
- the second point is LaTeX has _more_ \ than Markdown
- I believe you understood now
```

The generated result will be almost the same:

This is a test file I begin by making a list of bullet:

- the first point is LaTeX is a bit verbose
- the second point is LaTeX has *more* \ than Markdown
- I believe you understood now

Why macros? Because without them, `markdown` simply does not scale. For example imagine you can't declare `\su` to be generated as $\sum_{n=0}^{\infty} u_n$ in a thesis where this expression is repeated 1000 times.

What makes LaTeX so excellent?

- Advanced typography features,
- no bug: TeX has no more known bug since many years,
- scalable:
  - you can include many LaTeXfiles into one
  - you can create *macros* that minimize mistake done by repeating pattern.
- equation done with latex are easy to create and render just impressively,
- versionnable:
  - you use a text format that can be easily handled by most versionning system,
  - easy to work on the same document with many different people.

And many other reasons I can't write them all here. Yes, LaTeX *rocks*.

Chapter 2

Installation

First, install LaTeX, ruby and the kramdown[1] gem.

- Download this source.
- Change the title of your document and the author name in the template.tex file.
- Create and write in kramdown format
- run rake (or rake compile) to create and show a .pdf file.

This proposition is already really good. You can version your book and separate each part of the book in different files organized in folders[2].

The inclusion is done *automagically* using file name (you can change this make a bit of ruby inside the Rakefile). But to have a really scalable solution, you need to have the ability to make macros in kramdown.

This is not a problem, I've done this. Here is how you can declare macros inside a kramdown file:

```
%%% simple %%% A Simple Macro %%%
%%% amacro %%% a
                    macro
                    on many lines %%%

%%% code %%% ruby: "a"*3 %%%

%%% complex %%% ruby: (1..5).map do |x|
x*x
end.join(" : ") %%%
```

[1] kramdown is an amelioration of the original markdown format.

[2] For now, files are ordered from their name. I then suggest you to name your files and folder with number prefixes. For example like 00_intro.md, 01_section/01_subsection.md, etc... Of course it is easy to ameliorate this make a bit of ruby (search sort in the Rakefile file).

These transformations will occur on the markdown file before it is transformed in LaTeX.

You can also declare macro that will be processed after the file was transformed in LaTeX.

```
LLL latex LLL \LaTeX LLL
```

In markdown, you simply write `%macroname` or `%code` and it will be transformed correctly in your pdf.

Chapter 3

Install

You'll need to install ruby, rake (installed by default on most computer).

```
[Ubuntu]> sudo apt-get install ruby rake
```

You'll also need a XeLaTeX installation (may I suggest TexLive full install?). You'll also need the kramdown gem.

```
gem install kramdown
```

And you should be ok to work.

Chapter 4

How do I write a book using it?

Write some file into content. Their format is the kramdown one (very close to Markdown)

Just run

```
rake
```

Of course there is also a

```
rake clean
```

and

```
rake clobber
```

The inclusion of files is done naturally by `Dir[content/**/*.md]`. If you want a more versatile way of doing it, simply look at the Rakefile and do a bit of ruby to sort file as you wish.

Chapter 5

Introduction

It is a simple demonstration of how macros are working. They were declared inside the markdown like this:

```
%%% multiline %%% a
multiline
macro %%%
%%% ruby %%% ruby: "a"*3 %%%
%%% complex %%% ruby: (1..5).map do |x|
x*x
end.join(" : ") %%%
LLL latex LLL \LaTeX LLL
LLL tldr LLL {\em Too long don't read: } LLL
```

Now if I write:

> `%tldr` A simple demonstration of how macros are working.

It renders as:

> *Too long don't read:* A simple demonstration of how macros are working.

The `%multine` macro render as:

> a
> multiline
> macro

The output should be in LaTeX and was compiled from a markdown-like format.

- Simple list ;
- Example ;
- Another one item.

```
Hello there
```

```
this is some code block
```

LaTeX Some LaTeX definition

A simple math mode x_i and a protected one $$x_i$$. A long formula now:

$$\sum_{i=0}^{n} \sqrt{x_i + y_i}$$

Even with some ruby code inside:
Here is the result of the `%ruby` macro:
aaa
and a more complex one:
1 : 4 : 9 : 16 : 25

Chapter 6

This is the first section or paragraph...

6.1 and a sub-section or section may-be

Here the text begins.