# Krambook

Yann Esposito

November 5, 2010

# 1 Write books like a hacker

Imagine the power and quality of LaTeX but with the clarity and simplicity of markdown. This is what this project is all about. Here is my idea:

What make 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*.

Then why simply don't use LaTeX?

Simply because LaTeXis verbose and full of backslashes. Just make a comparison between LaTeXand markdown

```
%--- 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}
%--- LaTeX source file ----
```

```
{:comment}
--- Kramdown source file ----
{/:comment}
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
```

The result will be similar:

> 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

How to have the clarity of Markdown without losing all advantages of LaTeX?

Here is my proposition:

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].

`,` 01_section/01_subsection.md‘, etc. . .

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:

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

[2] As the sort of file is done via the `Dir[content/**/*.md]` I suggest you to name your files and folder with prefixes for their position, like ‘00_intro

```
%%% 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(" : ") %%%
```

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.

## 2 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.

## 3 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.

# 4 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

# 5 This is the first section or paragraph...

## 5.1 and a sub-section or section may-be

Here the text begins.