HOW TO WRITE IN LAZYTEX INSTEAD OF STRICT LATEX

HUGO ROY

TL;DR. Use Pandoc, write your document in a combination of YAML, Markdown and, when you need it, inline LaTeX. Read Pandoc's README.

Keywords. LaTeX – Pandoc – YAML – Markdown

TeX is awesome. LaTeX was made to make it easier to use TeX and produce high-quality documents.

Still, there are two downsides with using LaTeX:

1. the source of your document is a bit cryptic for people who aren't used to source code
2. TeX was designed for paper as the output and thus comes with its limitations.

Today, most LaTeX documents end up as PDF and/or printed on paper (which is *kind of* the same). This is nice, but PDF and paper are not mediums that enable others to co-edit the text (unless they can work with the LaTeX source, but most people won't learn that, see point 1 above).

This is especially sad because LaTeX is not only able to produce awesome typesets, it's also able to produce part of the content of the document, thanks to a myriad of packages that you can use in LaTeX (for instance, varioref).

> For example, the varioref package is the best program I've seen to make automated references to another point in a document. Using varioref, LaTeX is able to print something like: "see section 3, on the facing page" automatically (or "... on the next page" if the document isn't supposed to be printed twosides aka *recto verso*).
> That's great, but that only makes sense for documents in pages, like paper or PDF. It does not make sense for a document in HTML that will be a single "web page" (of course, we could also emulate pages in HTML but, seriously, why are people doing that?) although it still makes sense to be able to refer to another part of the doc (like I did above.)

So what I'm doing these days is mostly **LazyTeX**.

What's LazyTeX?

LazyTeX is a way to use TeX that is lazy, but has the potential to overcome the two donwsides of using strict LaTeX that I just described.

Date: November 12, 2015.

Mainly, LazyTeX is just a funny name I have given to the combination of Markdown, YALM and inline LaTeX, that can be used through Pandoc in order to produce beautiful LaTeX PDF.

The upside to doing this, is that the source is way more legible for people like LibreOffice or Microsoft Office users, and the output will not necessarily be PDF but, in some cases, could as well be HTML or plain text, or something else.

Why's this lazy? There are two reasons to this:

First, the markdown syntax is lazier than the latex syntax. For instance, a list in markdown is as simple as writing:

```
This is some text.

- This is the first item of a list
- This is the second item of a list

This is some text.
```

whereas a list in LaTeX cannot really be more simple than:

```
This is some text.

\begin{itemize}
\item This is the first item of a list
\item This is the second item of a list
\end{itemize}

This is some text.
```

You get the idea. Sometimes, even documents that I only need as PDF, for which I could use plain, strict LaTeX — I today prefer to write them in a combination of YAML, Markdown and inline LaTeX — that means, what I call from now on LazyTeX.

However, lazy also has a downside. Mainly, if I mike a mistake in the source file, there are more risks of producing a PDF with the mistake showing in plain sight, rather than having a compilation error.

When I do a mistake in a LaTeX file, usually the compilation will give me an error and not produce the result. Thus, the error flags me that I need to fix something.

However, when I do a mistake in a LazyTeX file (for instance, misplacing a list inside a list because of wrong indention, or misplacing an asterisk that's supposed to make something bold) — in such cases, the LazyTeX file might compile correctly and will just print the mistake. So I may need to review the PDF more thoroughly, which can be cumbersome for long documents. So, in some cases, maybe LazyTeX should be avoided and strict LaTeX prefered.

How does it work?

Pandoc is what makes this possible. Pandoc has its own Markdown variant, which enables Markdown to be a bigger subset of HTML than the "vanilla" Markdown is.

But Pandoc also has some neat tricks that makes Markdown an interesting source for LaTeX. For instance, the `pandoc-citeproc` program that's shipped with Pandoc enables you to use the bibliography engines of LaTeX.

Pandoc also parses YAML data, which you can then use to generate parts of your LaTeX document, especially the preamble.

Pandoc also allows you to have inline LaTeX, meaning you can write some LaTeX inside your markdown and Pandoc will work it out. (Although this has some limitations).

Obviously, one of the biggest upside of Pandoc, is the ability to convert documents from one format to another.

What's not working?

The problem is that Pandoc's LaTeX "reader" isn't a full LaTeX parser (yet). So the markdown+inlineLaTeX combination may cause issues for non-LaTeX outputs.

So be careful with some commands.

See the sample document and sample PDF to see some of the limitations.

My solution right now, is to add another layer of complexity, to make things worse: I use custom directives in Emacs' Pandoc-mode.