

# Pandoc, Weasyprint, and PDF generation

## From plain text to PDF

2020-01-25T07:19:09-06:00

I love writing in plain text. Given the choice between writing a document in Word, Google Docs, or my text editor (I primarily use [atom](#), don't @ me), I will choose my text editor every time. I prefer my text editor because I don't want to pay attention to layout / formatting when I'm writing, I just want to write.

But no one wants to *read* a document in a text editor; they want to read on a tablet, or in the browser, or to print a document. Fortunately, there's a surprisingly powerful tool, [pandoc](#) that I have been using a lot lately and that makes conversion of plain text into readable formats an easy task.

Without spending too much time on the details, pandoc is a command-line tool that converts documents from one format to another. Have a markdown file and want a .docx? Pandoc can do that! Have an ePub and want an HTML? Pandoc to the rescue.

For example, this blog post is, itself, written in markdown. But suppose I wanted a PDF copy of it. Using pandoc, all I need to do is to run:

```
pandoc content/posts/2020-01-25-pandoc-weasyprint-and-pdfs.md -o
static/files/pandoc-test.pdf
```

If you're interested, you can [see what it looks like](#).

But let's say you wanted to get *fancy* with your PDF, and add some better styles. For example, suppose I wanted to add colors to my links. To do *that*, I need to pass pandoc a "variable," specifically: `colorlinks: true`. At the command line, I run:

```
pandoc content/posts/2020-01-25-pandoc-weasyprint-and-pdfs.md -V
colorlinks:true -o static/files/pandoc-basic-style-test.pdf
```

Check it out, you can [see there are colors now](#).

Now, let's say you want to go ahead and really gussy it up. It turns out you can use Cascading Style Sheets when creating a PDF, too. Don't like the default font for the body text? Just create a [styles.css](#) file, add a `-t html` flag, and run:

```
pandoc content/posts/2020-01-25-pandoc-weasyprint-and-pdfs.md
--css=styles.css -t html -o static/files/pandoc-styled-test.pdf
```

Another **improvement!** Though, frustratingly, there seem to be some CSS limitations / elements that just don't work. For example, I really want my links to have dotted underlines, but it just won't do it.

So now, for the final touch, if I *really really* want a beautiful PDF, I can use [Weasyprint](#), which is another tool that specializes in converting HTML to PDF. Here, I use the exact same stylesheet I used before, but pass a `-s` (standalone) flag and pipe to weasyprint:

```
pandoc content/posts/2020-01-25-pandoc-weasyprint-and-pdfs.md -s
-t html | weasyprint - static/files/pandoc-weasyprint-test.pdf -s
styles.css
```

Now, it's **just like I want it!**

There are obviously many more things that are possible options available here, but it's exciting to realize that plain text can be fun to write, and beautiful to read.