Using Pandoc on iOS (Sorta)

W. Caleb McDaniel

May 3, 2013

As I've explained before, I now do almost all of my writing---including my academic writing---in plain-text, Markdown files. I then use the incomparable document-conversion tool, Pandoc, to turn these files into HTML, Microsoft Word documents, PDFs, or EPUBs. Even this website is produced with Pandoc.

One virtue of this preference for plain-text files is that it enables me to write on my iOS devices. Plain text is mobile. But Pandoc is not, or so I have long assumed. Because Pandoc is a command-line utility, my ability to convert Markdown into Pandoc's other file formats has so far been tethered to my laptop.[1]

Enter Docverter, an open-source tool that makes it possible to use Pandoc without having it installed on your machine. To use it, you post `multipart/form-data` requests to a URL and get Pandoc output as a response, using the Docverter API to select which Pandoc options you want to use. So, for example, you can send Markdown text to Docverter and get back formatted PDF output, which you can then write to a PDF file---all without having a computer with Pandoc installed.

But can you use Docverter without having a desktop or laptop computer at all? At first glance, no: the documentation suggests using another command-line Unix utility, `curl`, to make requests to Docverter. But you can't use `curl` from an iPhone or an iPad, or at least not easily. If you want to use Pandoc from an iOS device, you need a different way to access Docverter.

Pythonista + Docverter = iPandoc (Sorta)

The solution is Pythonista, an inexpensive iOS app that allows you to write and execute Python scripts, including ones that interact with web services like---you guessed it---Docverter. Last weekend I spent a day writing a script that uses `httplib`, one of the standard Python libraries included with

Pythonista, to post requests to Docverter from my iPad and get back Pandoc output.

You can grab the script, with explanatory comments, as a Gist.

Briefly put, the script works like this. I open a text editing app like Nebulous Notes and put some Markdown text on my iOS clipboard. I open Pythonista and run my iMDtoPDF script. A few moments later, a PDF output file appears in my Dropbox.[2]

This method will help me quite a bit in the following sort of situation: I am away from my laptop, but I need to email a long, footnote-heavy document to someone. Or a student requests a recommendation letter that is due before I can get to my computer, and I need to create a PDF version with letterhead from the plain text version of the letter that I keep under version control. Now I can run this script, go to my iOS Dropbox app, and email the PDF file from there, all without leaving the iPad.

In other words, I can use Pandoc on the iPad ... sorta. I haven't yet figured out how to utilize all of Docverter's options in my script. And unlike Pandoc, Docverter does not use LaTeX to produce PDFs. Instead, it converts Markdown to HTML and then uses a service called Flying Saucer to print that HTML to a PDF. Nonetheless, this method still allows you to control some of the styling of the PDF by using a CSS stylesheet and embedded fonts. For example, here's the PDF output of this post using the CSS declarations included in my script.[3]

Extending the Script

In the Gist version of this script, I've deliberately kept things simple. But the version I'm using has several extended features.

For example, the public version automatically gives the output PDF file a generic, time-stamped name. But you can use Pythonista's console module to prompt yourself for a specific title.

```
## Name output file with user input
import console
title = console.input_alert(&#39;Output filename&#39;,
&#39;Enter title below&#39;)
outfile = title + &#39;.pdf&#39;
```

By adding these two lines to the end, I can also have the script automatically open the output file in GoodReader, where I can then preview, annotate or email it:

```
## Get temporary URL for file in Dropbox
share_url = dropbox_client.media(outfile)
url = share_url.get(&#39;url&#39;, &#39;not found&#39;)

## Prefix the URL with &#39;g&#39; to open in Goodreader
import webbrowser
webbrowser.open(&#39;g&#39; + url)
```

Python makes numerous other features possible: my modified version, for example, prints a timestamp in the footer of the PDF's first page. And Docverter itself can be used to extend the script: with a slight tweak of the script's `fields` variable, for instance, you can output Docx files or EPUB files instead. Finally, by stringing together Pythonista with other apps like Drafts, it's possible to automate this script without having to manually launch Pythonista every time a conversion takes place.[4]

These solutions, to be sure, do not bring Pandoc in all its glory to the iOS platform. But they do *sorta* make Pandoc mobile. And I think that's sorta cool.

---

1. I should note that many iOS text editors now feature the ability to create Markdown previews or convert Markdown into HTML. But they lack some of the features of Pandoc that I most rely on, like footnotes and PDF output.

2. The ability to upload the file to Dropbox requires taking two other steps in addition to this script. First, you have to create a free Dropbox development app. Then, you have to create this separate script in Pythonista; it includes a function for interacting with the Dropbox API that is called by my script. For more information, see Using the Dropbox module in the Pythonista forums.

3. Flying Saucer's method of producing PDFs does have disadvantages over Pandoc's native use of LaTeX; for example, you'll notice in the PDF version of this post that quotation marks within code blocks are converted into HTML character codes, which wouldn't happen if I had made the PDF using Pandoc on my laptop. For most of the mobile use cases that I can imagine, however, Flying Saucer should fine.

4. For tips on automation and a ton of other Pythonista ideas, see Frederico Vitti's monster post on Macstories.