My AsciiDoc setup

Tobie Nortje

Table of Contents

Contents

This post contains some notes on my AsciiDoc setup. Whilst it is realtively easy to set everything up I found it quite frustrating to find the info needed that would supply a good toolchain from markup to html, pdf, epub etc. I decided to start documenting my progress. Maybe this will be usefull to someone out there. Dont get me wrong all the tools exist but figuring out how they all work, play together and how to style the different output types proved hard for me to figure out.

Others have done this

There are a couple of posts that I found were writers have tried to document their respective experiences with setting up the toolchain. Here are some of the more notable ones:

1. This guy
2. This oak

AsciiDoc processor

I am using the AsciiDoctor engine to render my documents. Just found it easier to install. It is also better documented and much faster.

Text editor

I am using Submlime text with the AsciiDoc plugin

Create a new file, save it as `<filename>.asc`

Syntax highlighting

Because I am using the AsciiDoctor engine, which is written in Ruby, I installed the pygments.rb package.

Use the `:source-highlighter: pygments` snippet at the top of the document to enable this. Here is a quick example:

```
import pandas as pd
from multiprocessing.dummy import Pool as ThreadPool  # <1>


pool = ThreadPool(16)

files = !ls /Users/tobie/Documents/johan_pdf/*.pdf
files = files[2:-1]

def pt(filename):
    a = !pdf2txt.py $filename      # <2>
    for j,i in enumerate(a):      # <3>
        if "ACCOUNT MONTH" in i:    # <4>
                print filename, a[j+2]  # <5>
                return a[j+2]      # <6>

results = pool.map(pt, files)      # <7>

df = pd.DataFrame( {'filename':files,'month':results})
df.to_excel('/Users/tobie/Documents/johan_pdf/dates_1st_pass.xls')
```

Live rendering in your browser

I am using the Chrome plugin for live rendering. In the settings set `safe mode` to `server` to ensure external files are linked. There are also plugins for other browsers.

NOTE The live rendering now became part of my writing workflow. If you use a second screen you can open up a browser. Point it to the asciidoc file you are currently editing. You will see it redered in the browser. Each time you save the document it should refresh itself. Now you do not need to compile the document each time. Now you can focus on the writing and have a look in real time how it renders.

AsciiDoctor cheatsheet

Realy all you realy need to know about the markup can be found at the AsciiDoctor reference guide

PDF generation

The asciidoctor-pdf package makes it possible to generate PDF documents from your source. After installation you can run the following command to get basic PDF generation.

```
asciidoctor-pdf example.adoc
```

EPUB generation

The asciidoctor-epub3 packages makes it possible to generate EPUB3 ready outputs from your source. Run the following command to generate a basic EPUB ouput.

```
asciidoctor-epub3 example.adoc
```

It is important to remember that you need to wrap an ebook in an outside chapter. This means that you have to add a file like to one below to wrap the book in. run the `asciidoctor-epup3` command agains this file name.

epub.asc

```
= Asciidoctor EPUB3: My Setup
Tobie Nortje
:doctype: book
 include::yourbook.asc[]
 include::yourbook.asc[]
```

Makefile

For now I am using the following command to create al of the outputs

```
asciidoctor readme.asc & asciidoctor-epub3 ebook.asc & asciidoctor-pdf readme.asc
```

Custom stylesheet

The current setup works ok. For now I will keep it like it is.

Conditional compilation

I am sure later on I would need to set some directives to change values based on the output for example changing image sizes between HTML, pdf and epub.