Document Typesetting Template Readme – Draft

Genrep Software

June 16, 2020

CONTENTS

1 Preamble

The text above is a "document preamble." It looks pretty normal within the `README.md` Markdown file, and it is invisible in the `README.pdf` generated by Pandoc. However it is sometimes visible as a table in the GitHub Markdown view for the repository. In Markdown, the YAML-formatted preamble looks like:

```
---
title: Document Typesetting Template Readme -- Draft
author: Genrep Software, LLC.
date: June 16, 2020
---
```

If reading the PDF, notice that the above has fancy syntax highlighting. Pandoc does this for us! Also note that three variables are set: `title`, `author`, and `date`. All of these must be set for the document to compile properly. If you are converting a non-Markdown document and cannot set the metadata values, run Pandoc (or the compilation script) with the `-M` or `--metadata`[1] option as follows, for example:

```
pandoc -M title:"Fancy title" --metadata author:"Genrep" README.md -o README.tex
```

Note that this technique of manually including metadata is particularly useful for compiling `.docx` Microsoft Word documents, which may have been exported and downloaded from Google Docs. Look in the `compile.sh` script for an exmample of how this is used. Read the Usage section to learn about how to properly compile `.docx` documents using the compilation script.

Unfortunately, even though Markdown syntax is similar across platforms that use it, there is no official standard. When writing Markdown documents to version control and eventually typeset using this template and script, refer to the Pandoc Markdown Reference.[2]

2 Introduction

Genrep Software specializes in, among other things, report generation. As a result, it is absolutely critical that the documents put out by the company look clean and professional. Using an advanced typesetting system is a good way to rapidly create professional-looking documents with highly consistent formatting. Furthermore, scripting the process of typesetting documents aligns with our mission of automating report generation.

For typesetting, we primarily use LaTeX (LaTeX). LaTeX is an advanced document typesetting system, originally created decades ago by famed computer scientist Donald Knuth. Pandoc compiles

[1] These two options are completely interchangeable. `-M` makes more sense for use in the terminal, whereas `--metadata` makes much more sense for scripts where readability is a concern.

[2] `https://pandoc.org/MANUAL.html#pandocs-markdown`

Markdown to LaTeX for us, so that we don't have to learn and remember the fancy LaTeX commands. That being said, it is possible to inline LaTeX commands. For example, if reading in the PDF, it will not be obvious that the weird-looking word `LaTeX` is created by typing `\LaTeX{}` into the Markdown. GitHub renders the Markdown without converting it to LaTeX, so it will look like raw commands, rather than nice formatting there.

One benefit of using LaTeX is that we can also inline math, which will be beautifully typeset for us. The nice math below will not render correctly on GitHub unfortunately. To see how nice the math looks, view the `README.pdf` file in the repository.

$$\mathrm{Borwein}(n) := \int_0^\infty \prod_{i=0}^n \frac{\sin(x)}{x} dx$$

The code to render the math above is:

```
$$ \mathrm{Borwein} (n) := \int_0^\infty \prod_{i = 0}^n \frac{\sin (x)}{x} dx $$
```

3 Quick Start

3.1 Setup

Make sure that you have Pandoc and LaTeX installed locally. There are other ways to compile `.tex` files, such as Overleaf,[3] but for now, it is easiest to have a local distribution running. Installation instructions can be found at the following links (respectively):

- `https://pandoc.org/installing.html`
- `http://www.tug.org/texlive/acquire-netinstall.html`

3.2 Usage

Using the template is just a matter of cloning the repository and copying `template.tex`, `compile.sh`, and `logo.png` into the folder of a document you want to compile and running the compile script at the command line. One day, the process of installing the required files in a destination directory will be automated with a small script.

3.2.1 Markdown Files

Pandoc-flavored Markdown files can contain metadata (title, author, and date) in YAML format.[4] This data is used while compiling the document, and having a `title` field, an `author` field, and a `date`

[3] `https://www.overleaf.com/`
[4] `https://pandoc.org/MANUAL.html#metadata-blocks`

field is *absolutely required*. If any of these fields is not included, the document may be formatted strangely, and LaTeX may give cryptic errors. When debugging this script and the template, the first thing to check is whether the test case has the proper metadata. This script assumes that if a Markdown file is being typeset, it will have data inside.

To compile a Markdown file with YAML metadata inside the file, use:

```
./compile.sh <filename to convert (ending in .md)>
```

3.2.2 Google Docs Links

For comiling Google Docs links, it is necessary to include the URL and a title. There are also optional `author` and `date` arguments, but “Genrep Software, LLC.” and the date of compilation will be used if they are not provided. Make sure to surround the command-line arguments with quotation marks.

Also ensure that any Google Docs link used has the correct sharing settings. In particular, it must be set so that “Anyone with the link can view,” that way this script can access the link to export the document. If the sharing permissions are set incorrectly, there may be cryptic errors due to LaTeX trying to compile a mostly empty document.

The general usage of the command for Google Docs links is:

```
./compile.sh <Google Docs link> "<title>" "[author]" "[date]"
```

In particular, since the `author` and `date` arguments are optional, we can do:

```
./compile.sh \
  "https://docs.google.com/document/d/1w-AzMzVo05u5aeyMHdYXJei0wdfGFMCQxZ2Ks7_QWw8/edit" \
  "Old To-do List"
```

This will generate a nicely-formatted, typeset file called `Old To-do List.pdf`.

3.2.3 Miscellaneous Document Files

To compile any other type of file, it is necessary to manually include the title, author, and date metadata. This can be done by specifying them as command-line arguments for the compilation script in that order. It is very important that the arguments be surrounded by quotation marks if they have spaces in them. For example:

```
./compile.sh proposal.docx "This is the title" "Genrep Software, LLC" "4/20/69"
```

The general usage of the command for non-Markdown files is:

```
./compile.sh <input file> "<title>" "<author>" "<date>"
```

The compilation script will throw an error and fail if it is given a non-Markdown file with too few arguments. If, in the generated PDF, the displayed title is only a single word and/or the author or date is only a single word, make sure to surround the command-line arguments with quotation marks.

4 Modifying the Template

Modifications to the template may be required for aesthetic reasons, or if there are compilation errors due to necessary packages not being imported. If there are errors because of packages, consult the `pandoc-default-template.tex` file included in this repository. The file was generated by running

```
pandoc -D latex > pandoc-default-template.tex
```

The file will include various packages based on internal Pandoc variables that are set during document compilation. Do some digging in there based on the compilation error and see if it is possible to determine what the missing package is. Google is your friend here.

5 Appendix

5.1 Compilation script

Included below is a syntax-highlighted dump of the compilation script.

```
#!/bin/bash

# Compile to LaTeX based on a company-wide template
# Created by Jacob Strieb
# June 2020

set -e

# XXX: Adjust below if not running in Windows Subsystem for Linux with native
# Windows Pandoc and pdfLaTeX
PANDOC=pandoc.exe
PDFLATEX=pdflatex.exe

# Echo usage if the user asks for help
if echo "$1" \
  | grep --ignore-case --quiet "^\-h$\|^\-\-help$"; then
  echo "Usage: $0 <infile.md>"
  echo "   or  $0 <Google Docs URL> \"<title>\" \"[author]\" \"[date]\""
  echo "   or  $0 <infile.*> \"<title>\" \"<author>\" \"<date>\""
  exit
fi

# If the input is a Google docs/drive URL, handle and exit
if echo "$1" \
  | grep --ignore-case --quiet \
    "^https\?:\/\/[a-z]*\.google\.com\/document\/d\/"; then
  if [ -z "$2" ]; then
    echo "Usage: $0 <infile.md>"
    echo "   or  $0 <Google Docs URL> \"<title>\" \"[author]\" \"[date]\""
    echo "   or  $0 <infile.*> \"<title>\" \"<author>\" \"<date>\""
    exit
  fi
  DOC_TITLE="$2"
  DOC_AUTHOR="Genrep Software, LLC."
  if [ -n "$3" ]; then
    DOC_AUTHOR="$3"
  fi
  DOC_DATE="$(date +'%A, %B %d, %Y')"
  if [ -n "$4" ]; then
    DOC_DATE="$4"
  fi

  EXPORT_FORMAT="docx"

  # Generate an export URL
  echo "Generating export URL..."
  EXPORT_URL="$(echo $1 | grep -o --ignore-case \
    '^https\?:\/\/[a-z]*\.google\.com\/document\/d\/[a-z0-9\_-]*\/')""export?format=$EXPORT_FORMAT"
  echo "Exporting from $EXPORT_URL..."

  OUTFILE="out.tex"
  TEMPFILE="out-temp.tex"
  INFILE="in.$EXPORT_FORMAT"

  # Download the document
  curl --output "$INFILE" --location "$EXPORT_URL"

  # Convert to LaTeX
  $PANDOC \
    --extract-media "." \
    --template "template.tex" \
    --metadata title:"$DOC_TITLE" \
    --metadata author:"$DOC_AUTHOR" \
    --metadata date:"$DOC_DATE" \
    "$INFILE" \
    --output "$TEMPFILE"

  # FIXME: this needs to be improved
  # Strip unnecessary quote environments
  cat "$TEMPFILE" \
    | sed "s/\\\\\(begin\|end\){quote}//g" > "$OUTFILE"

  # Compile LaTeX to PDF -- do it twice for TOC update purposes
  $PDFLATEX "$OUTFILE"
  $PDFLATEX "$OUTFILE"

  # Clean up
  cp "out.pdf" "$DOC_TITLE.pdf"
  rm in* out*
  # find . -name "in*" | xargs rm
  # find . -name "out*" | xargs rm

  exit
fi

# Set the input and output file based on command-line arguments
INFILE="README.md"
if [ -n "$1" ]; then
  INFILE="$1"
else
  echo "Please specify an input file to convert!"
  echo
  echo "Usage: $0 <infile.md>"
  echo "   or  $0 <Google Docs URL> \"<title>\" \"[author]\" \"[date]\""
  echo "   or  $0 <infile.*> \"<title>\" \"<author>\" \"<date>\""
  echo
  echo "Defaulting to \"README.md\"..."
fi
# Strip everything from the last period to the end of the string (inclusive)
OUTFILE="$(echo $INFILE | sed 's/\(.*\)\.[^\.]*$/\1/')"".tex"

# Get additional arguments for metadata if not Markdown based on extracting the
# file extension
if echo $INFILE \
  | sed 's/.*\.\([^\.]*\)$/\1/' \
  | grep --ignore-case --quiet md; then
  # Assume there is metadata in the file if it is Markdown
  # Output TeX file
  $PANDOC \
    --template=template.tex \
    "$INFILE" \
    --output "$OUTFILE"
else
  # If not Markdown, look for title, author, and date arguments (respectively)
  if [ "$#" -eq 4 ]; then
    # Output TeX file
    $PANDOC \
      --template=template.tex \
      --metadata title:"$2" \
      --metadata author:"$3" \
      --metadata date:"$4" \
      "$INFILE" \
      --output "$OUTFILE"
  else
    echo "Please specify the title, author, and date in quotes!"
    echo
    echo "Usage: $0 <infile.md>"
    echo "   or  $0 <Google Docs URL> \"<title>\" \"[author]\" \"[date]\""
    echo "   or  $0 <infile.*> \"<title>\" \"<author>\" \"<date>\""
    exit
  fi
fi

# Compile LaTeX to PDF -- do it twice for TOC update purposes
$PDFLATEX "$OUTFILE"
$PDFLATEX "$OUTFILE"
```