DARIAH Working Paper Workflow

Redaktionsumgebung – Work in Progress

Thorsten Vitt Mirjam Blümm

DARIAH-DE
Working Papers

Thorsten Vitt, Mirjam Blümm: „DARIAH Working Paper Workflow“. *DARIAH-DE Working Papers* No. 0. Göttingen: DARIAH-DE, 2016. TODO: doi.

Zusammenfassung

Für die Publikation der DARIAH-Working-Papers gibt es einen Workflow auf der Basis von Markdown, das mit Pandoc und LuaLatex formatiert wird.

Dieser Artikel beschreibt die Installation der Working-Paper-Vorlage und einige weitere Themen für Redakteure. Autoren finden Hinweise zur Formatierung des Texts in den Autorenhinweisen.

Inhaltsverzeichnis

1 Docker-basierte Redaktionsumgebung aufsetzen

Es gibt eine experimentelle Redaktionsumgebung in einem Dockercontainer, in dem bereits alles verpackt wurde, um aus einer DWP-Markdowndatei ein fertiges Working-Paper-PDF zu erzeugen (Pandoc, LaTeX, Vorlage, Schriften etc.). Docker kann auf dem Mac z.B. mit `brew cask install docker` installiert werden.

Um damit eine Markdown-Datei `Artikel.md` im aktuellen Verzeichnis zu übersetzen, kann man folgendes Kommando verwenden:

```
docker run -it -u "$(id -u):$(id -g)" -v "$(pwd)":/data thvitt/dwp-template Artikel.md
```

Zur Erläuterung: `docker run` führt einen Dockercontainer aus. Beim ersten mal wird das Image dabei heruntergeladen. `-it` führt zum interaktiven Betrieb, sodass Meldungen sofort angezeigt werden. `-u "$(id -u):$(id -g)"` bewirkt, dass der Container mit dem aktiven Benutzer ausgeführt wird – ansonsten gehören die Dateien hinterher möglicherweise root. `-v "$(pwd)":/data` stellt dem Docker-Container das aktuelle Verzeichnis (`$(pwd)`) zur Verfügung, sodass es darin schreiben kann. `thvitt/dwp-template` ist der Name des Images, und alle darauffolgenden Parameter werden dem dwp-Skript (s.u.) bzw. Pandoc übergeben.

Ich empfehle, das kurze Shellskript ddwp herunterzuladen, ausführbar zu machen und in den $PATH zu legen, dann muss man nurnoch `ddwp Artikel.md` tippen.

2 manuelle Redaktionsumgebung aufsetzen

Wer nicht mit der Docker-Umgebung arbeiten mag, muss ein wenig manuell installieren:

2.1 Voraussetzungen

- Pandoc
- LaTeX-Installation mit LuaLaTeX, z.B. aktuelles TeX-Live
- pandoc-citeproc (für das Literaturverzeichnis-Processing)
- Python 3 (für das Convenience-Skript)

2.2 Installation

(Es gibt ein einfaches, experimentelles Installationsscript `install.sh`, das auf Linux und MacOS X funktionieren sollte.)

Die Schriften WeblySleek UI[1] und Iosevka in der Default-Variante (Download `01-*`)[2] entsprechend dem Betriebssystem installieren. Für übliche Linux-Distributionen ist es ausreichend, die TTF-Dateien in den Ordner `~/.fonts` zu legen.

Einige Dateien aus diesem Verzeichnis müssen über das Dateisystem verteilt werden. Ich empfehle, die entsprechenden Dateien per symbolischem Link zu verlinken statt sie zu kopieren, um für Aktualisierungen gerüstet zu sein:

- **DWP.latex** ist das Pandoc-Template, es muss in das Verzeichnis `~/.pandoc/templates`.
- Die **`*.csl`-Dateien** sind die Styles für die Literaturverwaltung, sie stammen aus dem Zotero Style Repository[3]. `dwp.py` sucht diese Styles ebenfalls im Verzeichnis `~/.pandoc/templates`
- Die Bilder aus dem `img`-Ordner werden für die Titelseite benötigt. Sie werden in einem LaTeX-Baum gesucht.
- `dwp.py` ist ein Script zum einfachen Aufrufen von Pandoc mit entsprechenden Parametern. Es sollte irgendwo in den `$PATH`.

Unter der Annahme, dass dieses Verzeichnis unter `~/projects/dwp-template` liegt:

```
mkdir -p ~/.pandoc/templates
cd ~/.pandoc/templates
ln -s ~/projects/dwp-template .
mkdir -p `kpsexpand '$TEXMFHOME/tex/latex'`
cd `kpsexpand '$TEXMFHOME/tex/latex'`
ln -s ~/projects/dwp-template .
cd /usr/local/bin
sudo ln -s ~/projects/dwp-template .
```

Nach erfolgreicher Installation sollte es in jedem Verzeichnis möglich sein, mit `dwp datei.md` die entsprechende Datei in PDF zu übersetzen.

[1] http://www.dafont.com/weblysleek-ui.font
[2] https://be5invis.github.io/Iosevka/
[3] https://www.zotero.org/styles?q=chicago&format=author-date

3 Benutzung beider Umgebungen

Das beigefügte Pythonskript `dwp.py` ist der Einstiegspunkt des Dockercontainers und kann auch in der manuellen Installation aufgerufen werden. Es ist ein kleiner Wrapper um Pandoc, der versucht, pandoc mit den richtigen Parametern aufzurufen. Benutzung, hier in der oben empfohlenen Dockervariante als ddwp:

Der Standard-Aufruf ist `ddwp Artikeldatei.md`, wenn alles gut geht wird dann `Artikeldatei.pdf` erzeugt. Wichtig ist, dass die Eingabedatei immer das *letzte Argument* ist, Optionen aller Art müssen *vor* der Artikeldatei eingefügt werden. Folgende Optionen stehen zur Verfügung:

- `--debug` oder `-D` hat nur Auswirkungen, wenn beim Übersetzen etwas schief geht; in diesem Falle wird mit pandoc eine `.tex`-Datei erzeugt und diese einmal durch lualatex laufen gelassen, sodass .tex-, .log- und sonstige Hilfsdateien entstehen und analysiert werden können.
- `-o Artikeldatei.tex` erzeugt einfach die .tex-Datei, ohne LaTeX-Lauf.
- `-v` oder `--verbose` lässt pandoc ausführlichere Meldungen ausgeben.
- Beliebige Pandoc-Optionen können eingefügt werden.

3.1 Wenn keine Markdown-Datei vorhanden ist (Word-Workflow)

… dann rufe man zunächst `ddwp Artikeldatei.docx` auf. Dies konvertiert (mit Pandoc) die Artikeldatei nach Markdown (und versucht dann gleich, sie zu übersetzen, sodass man schonmal eine Vorschau bekommt). Danach arbeitet man dann mit der Markdown-Datei `Artikeldatei.md` weiter und gibt auch die beim ddwp-Aufruf an.

Achtung: Eingebettete Dateien etc. werden in ein Unterverzeichnis `media` des aktuellen Verzeichnisses gepackt.

Für LibreOffice-Dateien kann es sinnvoll sein, die zunächst in LibreOffice nach DOCX zu konvertieren, da da ggf. die Pandoc-Unterstützung besser ist – am besten beides ausprobieren.

4 Troubleshooting

4.1 irgendwas mit UTF-8

Eingabedatei ist nicht UTF-8-codiert. Im Texteditor öffnen und als UTF-8 speichern.

4.2 PDF-Datei kann nicht erzeugt werden, keine ordentliche Fehlermeldung

`dwp --debug Artikeldatei.md` erzeugt tex und lässt es mit LuaLaTeX laufen, ggf. sieht man dann mehr Fehlerursachen.

4.3 Fehlermeldung wegen `\bibname`

Metadatenfeld `lang: de` oder `lang: en` angeben!

4.4 `.bib` vorhanden und kein Literaturverzeichnis

Heißt die `.bib` so wie die `.md`?

Wird auch wirklich zitiert, dh mit `@mueller2012` (entsprechend den Keys im `.bib`)? Siehe `DWP-Autorenhinweise.md` und `DWP-Autorenhinweise.bib`: Im `.md` muss `@Eijkhout1991` oder `[@Eijkhout1991]` stehen, um den u.a. Eintrag zu zitieren:

```
@BOOK{Eijkhout1991,
  title = {\TeX\ by Topic. A \TeX nician's Reference},
  publisher = {Addison-Wesley},
  year = {1991},
  author = {Victor Eijkhout},
  address = {London},
  keywords = {general},
}
```

Literatur, die nicht in Pandoc-Syntax zitiert wird, landet auch nicht im Literaturverzeichnis.

Wenn Autoren `.bib` anliefern, aber *nicht* korrekt zitieren, kann man sich behelfen, indem man exakt folgenden Metadateneintrag zum YAML-Header hinzufügt:

```
nocite: '@*'
```

In diesem Fall werden alle Einträge der `.bib`-Datei im Literaturverzeichnis gesetzt, auch solche, die nicht zitiert werden. Convenience-Features wie einheitliche Zitationskeys und Links von der Jahreszahl ins Literaturverzeichnis gehen damit natürlich nicht.

4.5 Schwieriges LaTeX

Im Prinzip kann man einfache LaTeX-Kommandos und -Umgebungen direkt ins Markdown schreiben. Wenn das nicht klappt, muss man Pandoc sagen, dass ein entsprechender Abschnitt rohes TeX ist. Dazu verwendet man die Raw-Attribute[4]-Funktion: Man schreibt einen Codeblock und unmittelbar dahinter `{=latex}`, also für größere Blöcke

[4] https://pandoc.org/MANUAL.html#extension-raw_attribute

```{=latex}
\begin{tabular}{ll}
   \begin{tabular}{l} foo \\ bar \end{tabular} & x \\
\end{tabular}
```