# DARIAH Working Paper Workflow

## Work in Progress

Thorsten Vitt Mirjam Blümm

DARIAH-DE
Working Papers

**Thorsten Vitt, Mirjam Blümm: „DARIAH Working Paper Workflow". *DARIAH-DE Working Papers* Nr. 0. Göttingen: DARIAH-DE, 2016. URN: TODO: urn.**

## 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 und einige Spezifika der Working-Paper-Vorlage; Details zur Markdown-Syntax findet man z. B. auf der Pandoc-Homepage.

# Inhaltsverzeichnis

# 1 Artikel schreiben

## *1.1 Text*

Die Texte sollen mit Markdown ausgezeichnet werden. Zum Übersetzen wird Pandoc verwendet, es sind entsprechend also Konstrukte aus Pandoc's Markdown[1] möglich.

## *1.2 Titeldaten*

Titeldaten und einige Einstellungen gehören in einen Metadatenblock im YAML-Format. Der Block beginnt mit einer Zeile aus drei Bindestrichen `---` und endet mit einer Zeile aus drei Punkten `...`. Metadatenfelder beginnen mit dem Feldnamen am Anfang der Zeile, dann folgt ein Doppelpunkt und ein Leerzeichen und schließlich der Inhalt des Felds.

Einige Felder (z. B. die Autorenliste) kann mehrere Werte aufnehmen. Dazu schreibt man eine YAML-Liste: Die Zeile mit dem Feldnamen endet nach dem Doppelpunkt, darauf folgt ein Listeneintrag pro Zeile, beginnend mit einem Bindestrich. Das Feld `abstract` kann mehrere Absätze umfassen, dazu endet die Zeile mit dem Schlüsselwort mit einem `|` und es folgen die Textabsätze eingerückt. Der Metadatenblock kann also z. B. so aussehen:

```
---
title: DARIAH Working Paper Workflow
subtitle: Spaß mit Pandoc
author:
- Thorsten Vitt
- Mirjam Blümm
lang: de
date: 2016
abstract: |
    Für die Publikation der DARIAH-Working-Papers empfehlen wir einen Workflow
    auf der Basis von Markdown, das mit Pandoc und LuaLatex formatiert wird.

    Dieser Artikel beschreibt die Installation und einige Spezifika der
    Working-Paper-Vorlage; Details zur Markdown-Syntax findet man z.B. auf der
    Pandoc-Homepage.
...
```

Die folgenden Metadatenfelder stehen zur Verfügung:

| Feld | Bedeutung |
|---|---|
| title | Titel des Artikels. |

[1] http://pandoc.org/MANUAL.html#pandocs-markdown

| Feld | Bedeutung |
|---|---|
| subtitle (optional) | Untertitel. |
| lang | Sprache, in der der Artikel verfasst ist: `de` oder `en`. |
| author | Autor des Artikels. Bei mehreren Autoren Liste verwenden. |
| longauthor (optional) | Autoren mit Fußnotenzeichen für Institute |
| institute | Institut(e), ggf. mit Fußnotenzeichen (Liste möglich) |
| date | Veröffentlichungsjahr |
| abstract | Zusammenfassung |
| keywords-de | Schlagwörter auf Deutsch (als Liste) |
| keywords-en | Schlagwörter auf Englisch (als Liste) |
| wpno | DARIAH-Working-Papers Nr. (wird von der Redaktion eingesetzt) |
| urn | URN (wird von der Redaktion eingesetzt) |

Für Texte, die zuvor als Report veröffentlicht worden sind, sollen die folgenden Metadaten ergänzt werden:

| Feld | Bedeutung |
|---|---|
| report-number | Nummer des Reports, z. B. `1.2.3` |
| report-date | Veröffentlichungszeitraum, z. B. `Dezember 2015` |
| report-fkz (optional) | Förderkennzeichen |

### *1.3 Bibliographie*

Für die Bibliographie empfehlen wir, die Literaturverzeichnis-Einträge im BibLaTeX- oder BibTeX-Format in einer Datei mit gleichem Namen wie der Artikel und der Endung `.bib` zu verwalten und sich für die Zitationen an die entsprechenden Pandoc-Konventionen[2] zu halten – in diesem Fall wird das Literaturverzeichnis automatisch einheitlich und entsprechend der Stilvorlagen formatiert.

Wird ein solches automatisches Literaturverzeichnis verwendet, so muss der Artikel mit diesem Kommando enden:

```
\biblio
```

Das Kommando setzt automatisch die entsprechende Überschrift und passt die Formatierungsvorgaben an.

### *1.4 Bilder*

Bilder sollten als PDF, PNG oder JPEG-Datei mitgeliefert und in einer Bildreferenz im separaten Absatz referenziert werden:

[2] http://pandoc.org/MANUAL.html#citations

```
![Ein Beispielbild](img/Logo_Working-Papers.pdf)
```


*Abbildung 1: Ein Beispielbild*

Ohne weitere Angaben wird eine in den Bildmetadaten hinterlegte Druckgrößenangabe berücksichtigt, die Bildgröße jedoch auf die Größe des Textbereichs begrenzt. Da die entsprechenden Metadaten oft falsch sind, sollten sie bei Bildern in Seitengröße überprüft und ggf. korrigiert werden. Das geht z. B. mit ImageMagick[3], das folgende Kommando setzt z.B. die Auflösung aller JPEG-Bilder auf 300 dpi:

```
mogrify -density 300 -units PixelsPerInch *.jpg
```

Bei Gimp heißt die entsprechend Option *Print Size*. Alternativ sind Größenangaben beim Einbinden des Bilds möglich.

## 2 Redaktionsumgebung

### *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 Schrift WeblySleek UI[4] entsprechend dem Betriebssystem installieren. Für übliche Linux-Distributionen ist es ausreichend, die TTF-Dateien in den Ordner `~/.fonts` zu legen.

[3] http://www.imagemagick.org/script/command-line-options.php#density
[4] http://www.dafont.com/weblysleek-ui.font

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[5]. `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.

### *2.3 Benutzung*

Das beigefügte Skript `dwp.py` kann einfach mit `dwp artikeldatei.md` aufgerufen werden. Es ruft Pandoc mit den richtigen Parametern auf, um `artikeldatei.pdf` zu erzeugen. Das kann dann z. B. so aussehen:

```
pandoc -o article.pdf --latex-engine=lualatex --template=DWP \
       --filter=pandoc-citeproc --bibliography=article.bib \
       --csl=$HOME/.pandoc/templates/chicago-author-date.csl \
       --metadata=link-citations:true \
       article.md
```

`dwp` kann auch mit weiteren Pandoc-Parametern aufgerufen werden, es reicht alle Parameter an Pandoc weiter und lässt dafür ggf. die selbst generierten Versionen weg.

[5] https://www.zotero.org/styles?q=chicago&format=author-date

## *2.4 Troubleshooting*

### 2.4.1 irgendwas mit UTF-8

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

### 2.4.2 PDF-Datei kann nicht erzeugt werden, keine ordentliche Fehlermeldung

1. TeX-Datei erzeugen lassen – das geht entweder auf die entsprechende Rückfrage oder mit `dwp -o article.tex article.md`
2. TeX-Datei manuell mit LuaLaTeX übersetzen lassen: `lualatex article.tex`, Fehlermeldungen prüfen

## *2.5 Umgang mit Dateien in Office-Formaten*

Wenn Dateien in Office-Formaten angeliefert werden empfiehlt sich folgender Workflow:

1. OpenOffice-Dateien nach DOCX konvertieren, die Pandoc-Unterstützung von DOCX ist besser
2. `pandoc -o article.md --extract-media=article_assets article.docx`. Erzeugt aus der Worddatei eine Markdown-Datei namens `article.md`, extrahiert die Bilder und speichert sie im Verzeichnis `article_assets` (wird ggf. angelegt)
3. Markdown-Datei nachbearbeiten

Vorteil: Bilder werden gleich ausgepackt, Grundformatierungen bleiben erhalten