The Journal of Open Source Software

Article Writing with Markdown and the Open Journals publishing pipeline

Albert Krewinkel [1,2¶], **Juanjo Bazán** [1*], **and Arfon M. Smith** [1,3*]

1 Open Journals **2** Pandoc Development Team **3** GitHub ¶ Corresponding author * These authors contributed equally.

DOI: 10.21105/joss.00000

Software

- Review
- Repository
- Archive

Editor: Open Journals

Reviewers:

- @SingleSourcePub
- @IlonaSilverwood

Submitted: 01 January 2020
Published: 23 May 2020

Summary

This article describes the features of the Journal of Open Source Software (Smith et al., 2018) publishing pipeline. The publishing method is similar to the model described by Krewinkel & Winkler (2017), in that Markdown is used as the input format. The author-provided files serves as the source for all generated publishing artifacts.

Apart from the main text, articles should also provide a metadata section at the beginning of this article is formatted using YAML, a human-friendly data serialization language (*The Official YAML Web Site*, 2022). This information is included in the title and sidebar of the generated PDF.

Authors who face difficulties while writing are referred to the paper by Upper (1974).

Statement of Need

The journal publisher, in most cases where you'd be reading this, Open Journals, maintains a detailed and helpful article on the requirements that articles must satisfy in order to be considered for publication in that journal. However, submission requirements do not help with the technical aspects of paper writing. The process for JOSS and similar journals is different, in that the paper should be written in the lightweight markup language *Markdown*.

This article explains the technical details and describes the publishing system's capabilities. It can also be used as a test document, or serve as a template that can be used as a starting point.

Markdown primer

Markdown is based on email conventions. It was developed by John Gruber and Aaron Swartz. This section provides a brief introduction to Markdown syntax. Certain details or alternatives will be omitted,

If you are already familiar with Markdown, then you may want to skip this section and continue with the description of article metadata.

Inline markup

The markup in Markdown should be semantic, not presentations. The table below gives a small example.

Table 1: Basic inline markup and examples.

Markup	Markdown example	Rendered output
emphasis	`*this*`	*this*
strong emphasis	`**that**`	**that**
strikeout	`~~not this~~`	~~not this~~
subscript	`H~2~0`	H_2O
superscript	`Ca^2+^`	Ca^{2+}
underline	`[underline]{.ul}`	underline
inline code	`` `return 23` ``	`return 23`

Links

Link syntax is `[link description](targetURL)`. E.g., this link to the Journal of Open Source Software is written as
`[Journal of Open Source Software](https://joss.theoj.org/)`.

Open Journal publications are not limited by the constraints of print publications. We encourage authors to use hyperlinks for websites and other external resources. However, the standard scientific practice of citing the relevant publications should be followed regardless.

Images

Markdown syntax for an image is that of a link, preceded by an exclamation mark `!`.

The main use of images in papers is within figures. An image is treated as a figure if

1. it has a non-empty description, which will be used as the figure label and
2. it is the only element in a paragraph, i.e., it must be surrounded by blank lines.

Example:

```
![Figure caption](path/to/image.png)
```

Images that are larger than the text area are scaled to fit the page. It can sometimes be useful to give images an explicit height and/or width, e.g. when adding an image as part of a paragraph. The Markdown `![Nyan cat](nyan-cat.png){height="9pt"}` includes the image "nyan-cat.png"  while scaling it to a height of 9 pt.

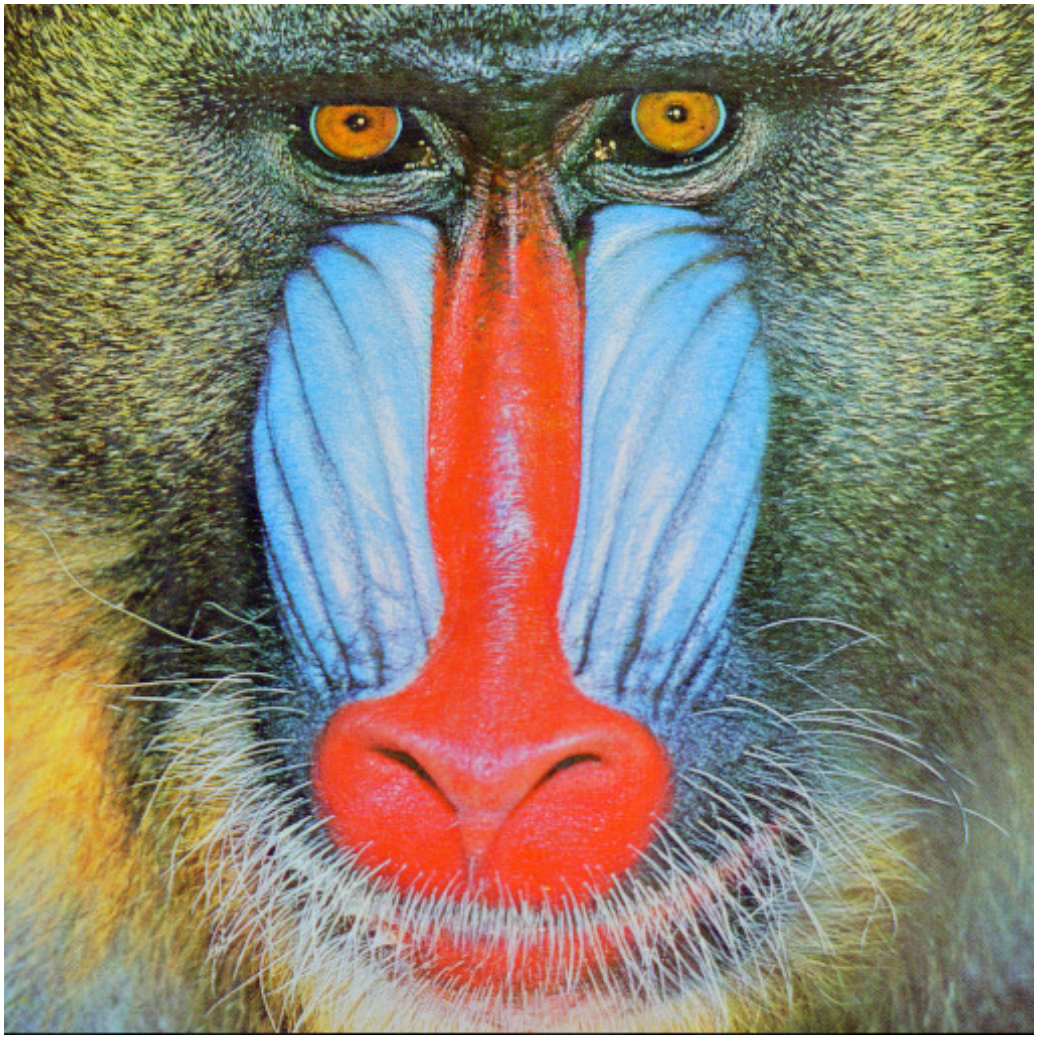

Figure 1: The "Mandrill" standard test image, sometimes erroneously called "Baboon", is a popular sample photo and used in image processing research.

Citations

Bibliographic data should be collected in a file `paper.bib`; it should be formatted in the BibLaTeX format, although plain BibTeX is acceptable as well. All major citation managers offer to export these formats.

Cite a bibliography entry by referencing its identifier: `[@upper1974]` will create the reference "(Upper, 1974)". Omit the brackets when referring to the author as part of a sentence: "For a case study on writers block, see Upper (1974)." Please refer to the pandoc manual for additional features, including page locators, prefixes, suffixes, and suppression of author names in citations.

Mathematical Formulæ

Equations and other math content has is marked by dollar signs (`$`). A single dollar sign should be used for math that will appear directly within the text, and `$$` should be used when the formula is to be presented in "display" style, i.e., centered and on a separate line. The formula itself must be given using TeX syntax.

To give some examples: When discussing a variable x or a short formula like $\sin \frac{\pi}{2}$, we would

write `$x$` and `$\sin \frac{\pi}{2}$`, respectively. However, for more complex formulæ, display style is more appropriate. Writing `$$\int_{-\infty}^{+\infty} e^{-x^2} \, dx = \sqrt{\pi}$$` will give us

$$\int_{-\infty}^{+\infty} e^{-x^2} \, dx = \sqrt{\pi}$$

Numbered equations and internal cross-references are discussed futher below.

Footnotes

Syntax for footnotes centers around the "caret" character ^. The symbol is also used as a delimiter for superscript text and thereby mirrors the superscript numbers used to mark a footnote in the final text.[1]

```
Articles are published under a Creative Commons license[^1].
Software should use an OSI-approved license.

[^1]: An open license that allows reuse.
```

Note numbers do not have to be sequential, they will be reordered automatically in the publishing step. In fact, the identifier of a note can be any sequence of characters, like `[^marker]`, but may not contain whitespace characters.

The above example results in the following output:

> Articles are published under a Creative Commons license[2]. Software should use an OSI-approved license.

Blocks

The larger components of a document are called "blocks".

Headings

Headings are added with # followed by a space, where each additional # demotes the heading to a level lower in the hierarchy:

```
# Section

## Subsection

### Subsubsection
```

Please start headings on the first level. The maximum supported level is 5, but paper authors should usually try to limit themselves to headings of the first two or three levels.

Deeper nesting

Forth- and fifth-level subsections – like this one and the following heading – are supported by the system; however, their use is discouraged.

Avoiding excessive nesting

Usually lists, as described in the next section, should be preferred over forth- and fifth-level headings.

[1] Although it should be noted that some publishers prefer symbols or letters as footnote markers.
[2] An open license that allows reuse.

Lists

Bullet lists and numbered lists, a.k.a. enumerations, offer an additional method to present sequential and hierarchical information.

```
- apples
- citrus fruits
  - lemons
  - oranges
```

- apples
- citrus fruits
  - lemons
  - oranges

Enumerations start with the number of the first item. Using the the first two laws of thermodynamics as example.

```
0. If two systems are each in thermal equilibrium with a third, they are
   also in thermal equilibrium with each other.
1. In a process without transfer of matter, the change in internal
   energy, $\Delta U$, of a thermodynamic system is equal to the energy
   gained as heat, $Q$, less the thermodynamic work, $W$, done by the
   system on its surroundings. $$\Delta U = Q - W$$
```

Rendered:

0. If two systems are each in thermal equilibrium with a third, they are also in thermal equilibrium with each other.
1. In a process without transfer of matter, the change in internal energy, ΔU, of a thermodynamic system is equal to the energy gained as heat, Q, less the thermodynamic work, W, done by the system on its surroundings.

$$\Delta U = Q - W$$

Article metadata

Names

Providing an author name is straight-forward: just set the `name` attribute. However, sometimes fine-grained control over the name is required.

Name parts

There are many ways to describe the parts of names; we support the following:

- given names,
- surname,
- dropping particle,
- non-dropping particle,
- and suffix.

We use a heuristic to parse names into these components. This parsing may produce the wrong result, in which case it is necessary to provide the relevant parts explicitly.

The respective field names are

- `given-names` (aliases: `given`, `first`, `firstname`)
- `surname` (aliases: `family`)
- `suffix`

The full display name will be constructed from these parts, unless the `name` attribute is given as well.

Particles

It's usually enough to place particles like "van", "von", "della", etc. at the end of the given name or at the beginning of the surname, depending on the details of how the name is used.

- `dropping-particle`
- `non-dropping-particle`

Literal names

The automatic construction of the full name from parts is geared towards common Western names. It may therefore be necessary sometimes to provide the display name explicitly. This is possible by setting the `literal` field, e.g., `literal: Tachibana Taki`. This feature should only be used as a last resort.

Example

```yaml
authors:
  - name: John Doe
    affiliation: '1'

  - given-names: Ludwig
    dropping-particle: van
    surname: Beethoven
    affiliation: '3'

  # not recommended, but common aliases can be used for name parts.
  - given: Louis
    non-dropping-particle: de
    family: Broglie
    affiliation: '4'
```

The name parts can also be collected under the author's `name`:

```yaml
authors:
  - name:
      given-names: Kari
      surname: Nordmann
```

Internal references

Markdown has no default mechanism to handle document internal references, often called "cross-references". This conflicts with goal of Open Journals is to provide authors with a seamless and pleasant writing experience. This includes convenient cross-reference generation, which is why a limited set of LaTeX commands are supported. In a nutshell, elements that were marked with `\label` and can be referenced with `\ref` and `\autoref`.

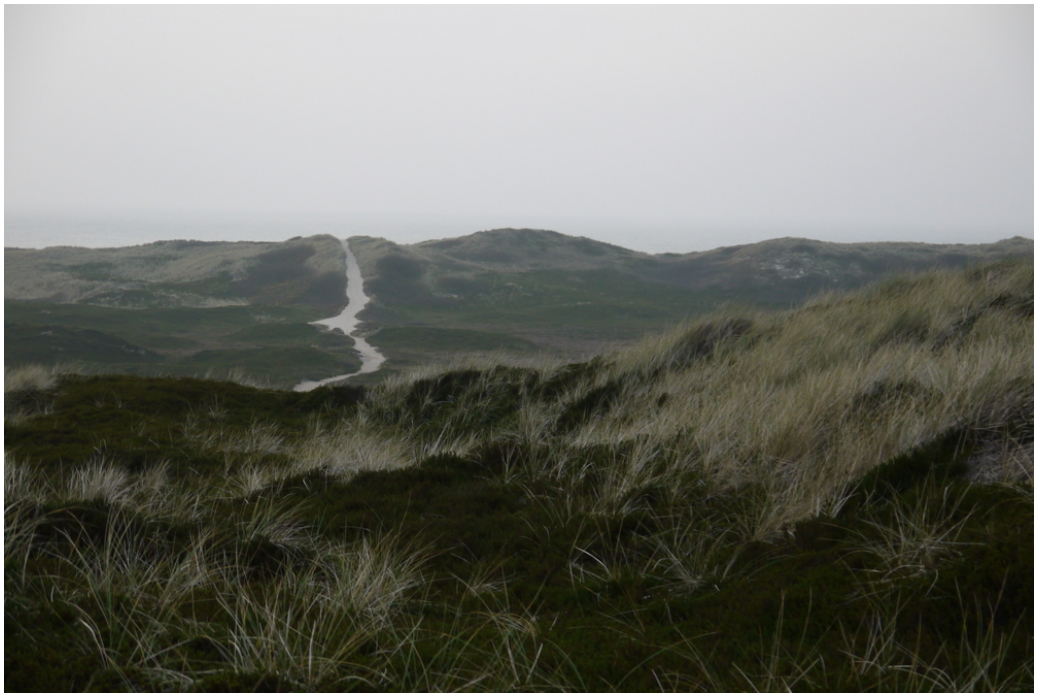

Figure 2: View of coastal dunes in a nature reserve on Sylt, an island in the North Sea. Sylt (Danish: *Slid*) is Germany's northernmost island.

Tables and figures

Tables and figures can be referenced if they are given a *label* in the caption. In pure Markdown, this can be done by adding an empty span `[]{label="floatlabel"}` to the caption. LaTeX syntax is supported as well: `\label{floatlabel}`.

Link to a float element, i.e., a table or figure, with `\ref{identifier}` or `\autoref{identifier}`, where `identifier` must be defined in the float's caption. The former command results in just the float's number, while the latter inserts the type and number of the referenced float. E.g., in this document `\autoref{proglangs}` yields "**??**", while `\ref{proglangs}` gives "**??**".

Table 2: Comparison of programming languages used in the publishing tool.

Language	Typing	Garbage Collected	Evaluation	Created
Haskell	static, strong	yes	non-strict	1990
Lua	dynamic, strong	yes	strict	1993
C	static, weak	no	strict	1972

Equations

Cross-references to equations work similar to those for floating elements. The difference is that, since captions are not supported for equations, the label must be included in the equation:

```
$$a^n + b^n = c^n \label{fermat}$$
```

Referencing, however, is identical, with `\autoref{eq:fermat}` resulting in "subsection ".

$$a^n + b^n = c^n$$

Authors who do not wish to include the label directly in the formula can use a Markdown span to add the label:

```
[$$a^n + b^n = c^n$$]{label="eq:fermat"}
```

Behind the scenes

Readers may wonder about the reasons behind some of the choices made for paper writing. Most often, the decisions were driven by radical pragmatism. For example, Markdown is not only nearly ubiquitous in the realms of software, but it can also be converted into many different output formats. The archiving standard for scientific articles is JATS, and the most popular publishing format is PDF. Open Journals has built its pipeline based on pandoc, a universal document converter that can produce both of these publishing formats – and many more.

A common method for PDF generation is to go via LaTeX. However, support for tagging – a requirement for accessible PDFs – is not readily available for LaTeX. The current method used ConTeXt, to produce tagged PDF/A-3, a format suited for archiving (*Document Management – Electronic Document File Format for Long-Term Preservation – Part 3*, 2012).

Document management – electronic document file format for long-term preservation – part 3: Use of ISO 32000-1 with support for embedded files (PDF/A-3). (2012). [Standard]. International Organization for Standardization.

Krewinkel, A., & Winkler, R. (2017). Formatting open science: Agilely creating multiple document formats for academic manuscripts with pandoc scholar. *PeerJ Computer Science*, *3*, e112. https://doi.org/10.7717/peerj-cs.112

Smith, A. M., Niemeyer, K. E., Katz, D. S., Barba, L. A., Githinji, G., Gymrek, M., Huff, K. D., Madan, C. R., Cabunoc Mayes, A., Moerman, K. M., Prins, P., Ram, K., Rokem, A., Teal, T. K., Valls Guimera, R., & Vanderplas, J. T. (2018). Journal of open source software (JOSS): Design and first-year review. *PeerJ Computer Science*, *4*, e147. https://doi.org/10.7717/peerj-cs.147

The Official YAML Web Site. (2022, April 19). https://yaml.org/

Upper, D. (1974). The unsuccessful self-treatment of a case of "writer's block". *Journal of Applied Behavior Analysis*, *7*(3), 497. https://doi.org/10.1901/jaba.1974.7-497a