詠ちゃんと学ぶ

同人誌組版の技術

Pandoc+LuaLaTeXで
Markdownから
技術同人誌を作ろう

神和電子

詠ちゃんと学ぶ同人誌組版の技術

Pandoc+LuaLaTeX で Markdown から技術同人誌を作ろう

神和電子

目次

はじめに

プロローグ

じりりりりりりりりん。じりりりりりりりりん。

ある日の夕方、午後 4 時半。父が遺した古い洋館、ひとりぼっちで暮らす大豪邸、その中の自室のベッドでごろごろしていた詠(よみ)は、廊下で鳴る黒電話のベルを聞いて、渋々立ち上がり部屋を出て、受話器を取りました。

「はい、西宮(にしのみや)です。.....あ、春(はる)ちゃんですか」

電話の相手は春子(はるこ)。詠の従姉妹で、同級生で、しかし詠と違ってちゃんと学校に行っていて成績もいい模範的少女です。

「同人誌をつくりたい.....ですか？」

春子が所属している情報科学部で、研究結果を同人誌にして文化祭で発表したいという案が出た。でも、部ではそういうのを作ったことがなく、定期試験の日程を考慮すると調査にあまり時間をかけている余裕がない。なので詠がもし詳しいのであれば良さそうなやり方を教えてほしい。と、そんなことを詠は春子から聞きました。

「うーん、一応ちょっとした本を作る心得はありますが.....」

と詠は答えます。実は、詠は屋敷に引きこもる日々で有り余る時間とお金を、夢想、或いは妄想を、本として実体化させるのに使ったことがあったのです。.....なんていう話をしたら、そのままとんとん拍子に、「明日は土曜日だから家に行くね！」と約束を取り付けられてしまいました。

「まったく、春ちゃんは強引なのです」

そんな風に呟く詠の表情は、言葉に反して輝いていました。

まえがき

この本では、Markdown で書いた原稿から同人誌をつくるための技術を解説しています。使う道具は、Pandoc と LuaLaTeX です。この 2 つのソフトウェアを使って、Markdown 形式→ LaTeX 形式→ PDF 形式という順番で変換していき、この PDF ファイルを印刷所に入稿することで同人誌をつくることができるのです。

Pandoc はとても優れたソフトであるため、Markdown から PDF ファイルをつくるだけであれば、相当簡単にできます。1 行のコマンドを叩くだけ、といっても過言ではないくらいです。

なのにこんな本を書いたのは、「PDF ファイルを作る」ことと「同人誌を作る」ことの間には結構な距離があるからです。「デフォルトの設定やデザインをそのままに、読むことは出来る文書をつくる」ことと、「デザインや機能性にこだわり、自分（たち）が素晴らしいと思う本をつくる」こととの差です。本書では、この前者と後者の間を埋めるための技術を解説できればと思っています。つまり、Pandoc 入門のその先、自分の望む冊子を作るにはどうすればいいのかという部分の解説が主です。

対象読者として想定するのは、いわゆる技術同人誌を作りたい IT エンジニアです。別に Pandoc や LuaLaTeX は用途を限るものではないですが、本書では縦書きの小説同人誌を作るための技術や、数式の美しい組み方等についての解説はしていません。また、コマンドラインである程度計算機を操作することに慣れているのを前提とした記述になっています。

本書が示す手順は、DevContainer という仮想化環境で実行することにしています。コンテナ技術を利用しているため可搬性は高いのですが、一部記述（フォント関係）は Windows を前提としたものとなっています。macOS や Linux の利用者は該当部分を読み替えてください。

サポートページについて

この本にはサポートページがあります。補訂内容や本文中のソースコードが掲載されていますので、適宜参照しながら読んでください。

- サポートページ：https://巫.jp/works/yomi-pandoc/
- ひみつのページ：（サンプルでは除去）
- サンプルコード：https://github.com/suzusime/yomi-pandoc-book

登場人物紹介

西宮（にしのみや）　詠（よみ）

中学二年生の少女。前作[1)]主人公。

少し前に両親を亡くしてしまい、実業家だった祖父が遺した立派な洋館に一人で暮らしている。自称魔法遣いなかんじの天才（？）少女なのだが、あまり人と関わるのが好きではないので、家にひきこもっている。春子（春ちゃん）は、数少ない心を許している存在。

IT はわりと好き。というのも、祖父が興した日本を代表する電工会社の社長を父がしているのを見ていて興味を持ち、昔から計算機（コンピュータ）には慣れ親しんできたからである。一方で、IT に対して家族を思い出させるという理由で微妙な感情を持っているものの、好きなものは好きなので、いつもパソコンに向かっている。

九條（くじょう）　春子（はるこ）

詠の従姉妹。中学二年生。

しっかりもので世話好きであり、詠のことが心配なのもあってよく詠の家に来る。そして、詠の家に来たときは何故かメイド服を着る。詠にとっては理由は謎。謎だが、衣裳に違わず手料理を作ってくれたり掃除をしてくれたりする。ありがたい。

私立のお嬢様学校に通っている（なお、詠も同じ学校に籍はあってクラスも同じなのだが、最近はほとんど行っていない）。お嬢様学校でありながらも中高一貫校でかなりの進学校なので、技術の授業でもちゃんとしたプログラミングが扱われていたり。

1）『詠の GStreamer 学習ノート』(https://巫.jp/works/yomi-gstreamer/)

第 1 章　全体計画

詠　というわけで、まずは全体的な流れを確認しましょうか。

春子　うん、お願い。

詠　わたしのほうで考えているのは Pandoc を使った方法なのですが、原稿の形式は Markdown で大丈夫ですか？

春子　大丈夫。部員はみんな Markdown に慣れてるから、むしろ Markdown でできるなら嬉しいかな。

詠　わかったのです。じゃあ Pandoc でいきましょう。

春子　ふんふん。で、Pandoc っていうのは？

詠　Pandoc っていうのは、いろいろな文書形式の間を変換できるソフトなのです。HTML 形式とか Word 形式とかほんとにいろいろ対応しているのですが、特に Markdown からの変換は得意ですね。これで Markdown の原稿から印刷用の PDF ファイルをつくるのです。

春子　ちょうどよさそうだね。

詠　ただ、Pandoc で一発で Markdown から PDF を作ることもできるのですが、一旦中間形式として LaTeX 形式を挟むのがいいかと思うのです。

春子　LaTeX っていうと、理系の論文を作るときによく使うって言うやつかな。

詠　そうなのです。LaTeX は、LaTeX 専用の形式で書かれた文書を綺麗な PDF に変換することができるソフトですね[1)]。実は、Pandoc から一発で PDF にするときにも内部では暗黙的に LaTeX を使っているのです[2)]。Markdown 形式→ LaTeX 形式→ PDF 形式という順番で変換するわけですね。今回は、Pandoc を使うのは Markdown 形式→ LaTeX 形式の変換だけにして、LaTeX 形式→ PDF 形式の変換は LaTeX を使う手順を教えようと思うのです。

---

1)　他にもできることはあります。

2)　厳密には LaTeX ではない TeX である ConTeXt を経由させることもできますが、99%の用途では LaTeX のほうがよいでしょう。

春子　それだと何が嬉しいの？

詠　一言でいうと、そのほうが細かい調整ができるのです。えぇと、今から話す手順は、絵にするとこんなかんじです。

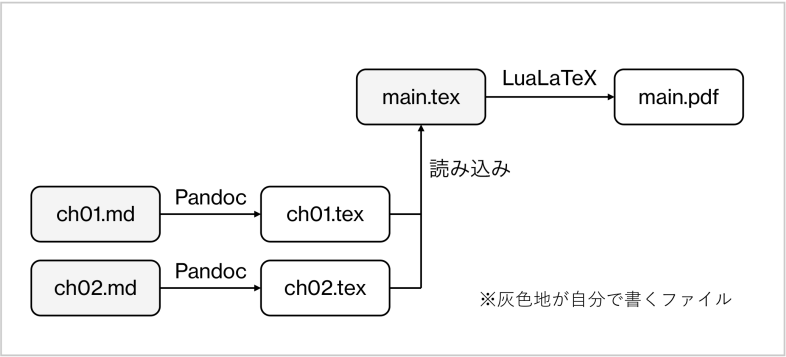

図 1.1　全体の流れ図

春子　うーんと、原稿は Markdown で書くけど、それとは別に `main.tex` という LaTeX 形式のファイルを用意するってこと？

詠　はい、各章の原稿を `ch01.md`、`ch02.md` みたいな Markdown のファイルで用意して、それを Pandoc で `ch01.tex`、`ch02.tex` といった LaTeX ファイルに変換します。で、`main.tex` にそれらの LaTeX ファイルを読み込むような指示を書いておいて、LaTeX で処理して PDF にするのです。そして、ここが大事なのですが、`main.tex` には「各章のファイルを読み込む」という命令だけじゃなくて、本全体のデザインの情報を書けるのです。

春子　あー、HTML と CSS みたいな関係ってことかな？ Markdown が HTML 相当で、`main.tex` に CSS みたいにスタイル情報を書くと。

詠　まさしくその通りです。Pandoc で一気に変換しようとするとデフォルトのスタイルしか使えない[3]けれど、`main.tex` を使うようにするとスタイルを自分で書ける。だから「細かい調整ができる」ってことです。

春子　なるほどね。じゃあ、その方法に決定！

3）Pandoc で一気に変換する場合もコマンドラインオプションでスタイルの調整ができなくはないのですが、この方法は LaTeX と Pandoc 両方に詳しくないと余計に難しいと思うので、本書では LaTeX 形式で一旦止める方法を採用します。

第 2 章　環境構築

2.1　DevContainer をつくる

詠　じゃあ、まずは環境構築ですが、普通にパソコンにインストールしていくと後で面倒になりがちなので、ここは DevContainer を使うのが良いと思うのです。

春子　うん、コンテナでででできるのならコンテナにするのはよさそうだね。

詠　確か春ちゃんは Windows でしたよね。WSL2 と Docker の環境と VSCode (Visual Studio Code) はもう入っていますか？[1)]

春子　うん。学校のプログラミングの授業でも使うし、入ってるよ。

詠　じゃあ、細かい説明はあとにして、WSL のコンソールに入って DevContainer 環境を作ってみましょうか。

```
# プロジェクト用のディレクトリを作成
$ mkdir mybook
$ cd mybook

# DevContainer用のファイルを作成
$ mkdir .devcontainer
$ touch .devcontainer/devcontainer.json
$ touch .devcontainer/Dockerfile

# VSCodeで開く
$ code .
```

詠　それぞれのファイルにはこんな風に書いてください。

```
// devcontainer.json
{
  "name": "Debian",
```

1)　ここの手順は本書では説明しませんので、適当にインターネットで検索してインストールしてください。

```
  "build": {
    "dockerfile": "Dockerfile"
  },
  "mounts": [
    "source=/mnt/c/Windows/Fonts,target=/usr/share/fonts/windows,type=bind」
    ↪  ,consistency=cached",
    "source=/mnt/c/Users/${localEnv:USER}/AppData/Local/Microsoft/Windows/」
    ↪  Fonts,target=/usr/share/fonts/windows-private,type=bind,consistenc」
    ↪  y=cached"
  ]
}
```

```
# Dockerfile

# TexLive インストール用コンテナ
# cf. https://github.com/Paperist/texlive-ja/blob/main/debian/Dockerfile
FROM ghcr.io/paperist/texlive-ja:debian AS texlive-installer
RUN tlmgr install lualatex-math selnolig # for pandoc lualatex template
RUN tlmgr install pdfjam # 塗り足し用


# メインのコンテナ
FROM mcr.microsoft.com/devcontainers/base:bookworm
WORKDIR /workdir
ENV PATH /usr/local/bin/texlive:$PATH
COPY --from=texlive-installer /usr/local/texlive /usr/local/texlive
RUN ln -sf /usr/local/texlive/*/bin/* /usr/local/bin/texlive

RUN apt-get update
RUN apt-get install -y pandoc fontconfig
```

春子　ん、ファイルを 2 つ作ったら「開発コンテナーで開きますか？」ってダイアログが出てきたね。

詠　はい、それを押して開発コンテナー (DevContainer) で開いてください。

春子　立ち上がった。.....けどちょっと時間がかかったね。

春子　これでも LaTeX を普通にインストールする手順よりは随分速いのですよ.....[2]。

2)　paperist/texlive-ja (https://github.com/Paperist/texlive-ja) という、日本語 LaTeX に必要なちょうど良いパッケージを含んだ Docker コンテナ（とてもありがたい）を元に Docker コンテナを作っているため、インストール時間が短くなっています。もし TeXLive のフルインストールをしようと思うと数時間かかります。

2.2 動作確認

詠　じゃあ、早速動作確認をしてみましょう。適当なフォルダをつくって、適当な Markdown ファイルを作ってみてください。

```
$ mkdir -p examples/ex01 && cd $_
$ touch main.md
```

春子　main.md の中身は……こんなかんじにしておこうかな。

```
# 大見出し
## 中見出し
### 小見出し

ここが本文です。
```

詠　よさそうです。じゃあ、こんな風にコマンドを打ってみてください。

```
# Markdown -> LaTeX
$ pandoc -f markdown -t latex --standalone --pdf-engine=lualatex -V
↪  documentclass=jlreq -o main.tex main.md

# LaTeX -> PDF
$ lualatex main.tex
```

春子　えーと……。あっ、`main.pdf` っていう PDF ファイルが出来た！うん、確かにそれっぽい感じに出来てるね！[3)]

図 2.1　生成された PDF

詠　無事動いていそうですね。とりあえずこれで Pandoc の Hello, world ができたのです。

3)　vscode-pdf という VSCode 拡張 (https://marketplace.visualstudio.com/items?itemName=tomoki1207.pdf) をインストールすると、VSCode のウインドウ内で PDF を確認することができ、とても便利です。

第 3 章　Pandoc 基礎

3.1　基本的な使い方

詠　次は、さっき打ってもらったコマンドの解説をしますね。

```
# Markdown -> LaTeX
$ pandoc -f markdown -t latex --standalone --pdf-engine=lualatex -V
↪  documentclass=jlreq -o main.tex main.md

# LaTeX -> PDF
$ lualatex main.tex
```

詠　まず、pandoc コマンドのほうですが、`-f markdown -t latex` は「Markdown 形式から LaTeX 形式にする」という意味です。from と to ですね。

春子　最後の`-o main.tex main.md` は入力と出力のファイル名だね。

詠　はい。だから、`--standalone --pdf-engine=lualatex -V documentclass=jlreq` のところがオプションらしいオプションなのです。まず、`--pdf-engine=lualatex` は「LaTeX のエンジンに LuaLaTeX を選ぶ」、`-V documentclass=jlreq` は「ドキュメントクラスとして `jlreq` を使う」という意味になります。

春子　LuaLaTeX？ドキュメントクラス？

詠　LaTeX と一口にいっても実はいろいろな処理系があって、LuaLaTeX はその一つです。名前の通り Lua 言語で拡張できるのが特徴なのですが、まあ Lua 部分を触ることはないと思うので、「日本語文書を作るなら LuaLaTeX を選ぶのがよい」というくらいの理解で大丈夫です。

春子　ふむふむ。

詠　で、ドキュメントクラスというのは……、まあ「デザインのテンプレート」みたいな感じのものだと思ってください。CSS だって、一から全部書くこと

はあまりなくて、Bootstrap とか TailwindCSS とかを使いますよね？

春子　最初から全部書くのは大変だもんね。基本的なデザインは既存のドキュメントクラスに任せて、必要なところだけカスタマイズするっていうことね。

詠　日本語文書のために LaTeX でよく使われるドキュメントクラスにはいくつかあるのですが、ここは `jlreq` を使うことにしましょう。細かいことは省きますが、日本語の本をつくるときの慣習に対応した、いいかんじのドキュメントクラスなのですよ[1)]。

春子　なるほど。じゃあ、`--standalone` のほうは？

詠　そうですね……。じゃあ、生成された `main.tex` を開いてみてください。

```
% Options for packages loaded elsewhere
\PassOptionsToPackage{unicode}{hyperref}
\PassOptionsToPackage{hyphens}{url}
%
\documentclass[
]{jlreq}

（中略。ここにたくさん呪文が書かれている）

\begin{document}

\hypertarget{ux5927ux898bux51faux3057}{%
\section{大見出し}\label{ux5927ux898bux51faux3057}}

\hypertarget{ux4e2dux898bux51faux3057}{%
\subsection{中見出し}\label{ux4e2dux898bux51faux3057}}

\hypertarget{ux5c0fux898bux51faux3057}{%
\subsubsection{小見出し}\label{ux5c0fux898bux51faux3057}}

ここが本文です。

\end{document}
```

1)　2024 年現在、LuaLaTeX + `jlreq` が「モダンな LaTeX 環境」のおそらく一番無難な選択です。5 年前くらいまでは LuaLaTeX はやや冒険的な選択肢で upLaTeX を使うほうが普通だったのですが、今では情報も増えてきたため LuaLaTeX を選んでもそう困ることはないと思います。最も有名な LaTeX の入門書である奥村晴彦／黒木祐介著『LaTeX 美文書作成入門』も、2020 年刊の第 8 版からは LuaLaTeX + `jlreq` を採用しています。なお、本書の解説内容は多分に LuaLaTeX に依存しており、pLaTeX 系では動作しません（特にフォント周り）。

春子 わ。なんかいっぱい書かれてる。

詠 初めて見ると面食らってしまうと思うのですが、まずは`\begin{document}`と`\end{document}`を探してみてください。ここに囲まれたところが本文なのです。

春子 ほんとだ、ここに Markdown の内容が変換されたっぽいのもが書かれてる。.....うん、さっきからのたとえだと、これは HTML の`<body>`タグみたいなものってことね。

詠 そうですそうです。ここで`--standalone` オプションについてですが、もしこのオプションを付けないと「`\begin{document}`と`\end{document}`の間にある部分だけ」が生成されるのです。HTML でいうと、`<body>`の中身だけが出てくるイメージです。

春子 なるほど？ でもそれだと LaTeX 形式のファイルとしてだめなんじゃない？

詠 はい、そのファイルをそのまま `lualatex` のコマンドに渡したらエラーになります。なので、`--standalone` を付けないで出てきたものは、そのままでは使わずに別のファイルから読み込んで使うのです。

春子 あ、最初に言っていた「`main.tex` から各章のファイルを読み込む」っていう話ね。

詠 ですね。だから、逆に`--standalone` を付けると、「それ一つで完結した（スタンドアロンな）LaTeX ファイル」が生成される、つまり`\begin{document}`より前の長々とした部分[2]も一緒に生成されるのです。`-V documentclass=jlreq` は、その部分の生成結果を調整するオプションですね。

春子 なるほどねー。

詠 `lualatex` コマンドのほうは、引数 1 つだけなので説明不要ですね。`main.tex` を入力として処理する、というだけです。

2） LaTeX 用語でプリアンブルと呼びます。

3.2　非スタンドアロンモードで使う

詠　じゃあ、Pandoc の基本的な動作が体験できたところで、最初に全体の設計として話した[3]「main.tex から各章のファイルを読み込む」方法、つまり非スタンドアロンモードでやるようにしましょうか。

```
# フォルダをつくる
$ mkdir -p examples/ex02 && cd $_
$ cp ../ex01/main.tex .
$ touch ch01.md
```

春子　さっき生成された main.tex をコピーしてくるんだね。

詠　はい。スタンドアロンモードで処理したときに生成されるのが要するに Pandoc のデフォルトのテーマなので、最初はここから始めるのがいいと思います。それで、\begin{document}と\end{document}の間をこんな風に書き換えてください。

```
（前略）
\begin{document}
\input{ch01}
\end{document}
```

春子　本文を直接書くかわりに、本文の内容は ch01 というファイルから読み込むって意味かな？

詠　はい、見ての通りですね。拡張子は省略していますが、これで「ch01.tex をここに読み込む」という意味になります[4]。この main.tex のビルドには ch01.tex が必要になるので、ビルドのコマンドはこうなります。

```
# Markdown -> LaTeX
$ pandoc -f markdown -t latex --pdf-engine=lualatex -o ch01.tex ch01.md

# LaTeX -> PDF
$ lualatex main.tex
```

3）第 1 章参照。

4）外部ファイルを読み込む LaTeX のコマンドには他に\include や\include*（後者は\usepackage{newclude}が必要）があります。\input はかなりプリミティブなコマンドなのであまり使うことを推奨されないのですが、本書の使い方だと恐らく問題がないため、ここではシンプルな\input を使うことにしました。