User Manual for Apiros3's Website Template

Apiros3

2025-09-08

Abstract

This guide attempts to cover the main struture of the website template, how the layers are separated and automatically made via the makefile. I also detail the documentation on how to add new blog posts, publications, notes, and reading list items.

1 Introduction

I originally wanted to make a website where I can directly place TeX files and have them automatically converted to HTML and PDF, so that I can write blog posts in TeX. To make the frontpage, I decided to reference some other websites that had blog posts and take their format, splitting the website respectively.

2 System Overview

2.1 Architecture

The website follows a three-tier architecture:

1. **Content Layer**: TeX source files, JSON metadata files, and static assets
2. **Processing Layer**: Python scripts, build tools, and conversion pipelines
3. **Presentation Layer**: Generated HTML

I tried to separate the functionality as far away from HTML as possible so that the contents can be updated via JSON metadata files, blog posts by TeX source files, and website color scheme by a single CSS file. Blog posts

follow the template html, and then everything is created automatically by running the Makefile.

The system consists of several key components:

- TeX Processing Pipeline: converts TeX files into md and pdf format. Each md file is then converted to html format, which is exactly the blog post shown on the top.
- JSON Processing Pipeline: takes the metadata and uses it to generate the rest of the page. Almost every text shown on the website can be manipulated through the JSON files placed throughout the directory.

2.2 Directory Structure / Metafiles

```
Apiros3.github.io/
|-- index.html                 # Main homepage (GitHub Pages entry)
|-- site.meta.json             # Site configuration metafile
|-- posts/                     # Blog posts and TeX sources
|   |-- index.html             # Blog listing page
|   |-- [yyyy]-[mm]-[dd]-[title].tex  # TeX source files
|   `-- [title]/               # Generated blog post directories
|-- publications/              # Publications and talks
|   |-- index.html
|   |-- data/                  # Publication metadata
|   `-- scripts/               # Generation scripts
|-- Notes/                     # Notes
|-- templates/                 # HTML templates
|-- script/                    # Python generation system
|-- css/                       # Stylesheets
|-- images/                    # Static assets
|-- build_html.sh              # TeX conversion script
|-- Makefile                   # Build system
`-- README.md                  # Documentation
```

The system uses several metafiles:

- `site.meta.json`: Main site configuration (essentially most information that does not fall into the rest of the items)
- `notes.meta.json`: Notes metadata, choosing which notes to show from the Notes submodule.

- `reading-list.meta.json`: Reading list metadata
- `publications/data/*.meta.json`: Publication metadata
- `posts/*.meta.json`: Blog post metadata

The Python system consists of several modules:

- `config.py`: Configuration loading and management
- `page_generators.py`: HTML page generation logic
- `template_engine.py`: Template rendering functions
- `data_loader.py`: Data loading and processing
- `generate_site_new.py`: Main orchestration script

The page generation then follows this workflow:

1. Load configuration from metafiles
2. Process TeX files and extract metadata
3. Load publication and talk data
4. Generate HTML pages using templates
5. Apply styling and JavaScript
6. Output final HTML files

2.2.1 Site Configuration (`site.meta.json`)

This metadata conssits of information regarding the general site. It also lets me decide what to show on the navigation panel, and what each description of those sections looks like.

```
{
  "site": {
    "title": "Apiros3",
    "description": "Academic Portfolio",
    "author": "Apiros3"
  },
  "about": {
```

```
    "title": "About Me",
    "content": "Multi-paragraph content...",
    "profile_picture": "images/profile.jpg",
    "profile_alt": "Some profile pic alt"
  },
  "contact": {
    "email": "email@institution.edu",
    "institution": "University Name",
    "location": "City, Country"
  },
  "navigation": {
    "brand": "Apiros3",
    "items": [
      {
        "name" : "Section Title"
        "url": "corresponding-page/index.html"
        "current" : false
      }
    ]
  },
  "notes": {
    "title": "Notes / Paper Summaries",
    "description": "Paragraph content description..."
  },
  "reading_list": {
    "title": "Reading List",
    "description": "Paragraph content description..."
  }
}
```

2.2.2 Notes (`notes.meta.json`)

Notes are organized by subject with metadata. The actual notes to be referenced is a separate public repository that I have incorporated into the site as a submodule (this is updated periodically via GitHub Actions CI).

```
{
  "title": "Title of Notes",
  "slug": "folder-name-under-Notes",
  "description": "Short description of notes",
  "pdf_file": "notes.pdf" or ["notes1.pdf", "notes2.pdf"] if there are multiple,
```

```
  "category": "!-- deprecated --!"
}
```

2.2.3 Reading Lists (`reading-list.meta.json`)

The reading list tracks resources I use so that I have access / have a page to revisit when I forget about material I found interesting in the past:

```
{
  "title": "Book/Paper Title",
  "author": "Author Name",
  "year": "2025",
  "type": "book / paper / article",
  "status": "reading / completed / planned",
  "description": "Brief description..."
}
```

2.2.4 Publication / Talks (`publications/data/*.meta.json`)

Publications are managed through the JSON metadata files in the following format under `title.meta.json` (we require a spearate metadata file per paper). When any fields are removed, they will not appear in the page, which lets me place links only if they are available. There will always be a default pdf link based on the title of the metadata, which it will send the user to the pdf placed inside Notes/publication/[title].

```
{
  "title": "Paper Title",
  "authors": ["Author One", "Author Two"],
  "conference": "Conference Name",
  "year": "2025",
  "abstract": "Abstract text...",
  "arxiv": "https://arxiv.org/abs/...",
  "doi": "https://doi.org/...",
  "code": "https://github.com/...",
  "venue": "CONF 2025",
  "pages": "1-10"
}
```

Talks follow a similar structure with additional fields. The difference is that talks are placed inside a single `talks.meta.json` file. This will appear as an independent section on the website as can be seen on this one.

```
{
  "title": "Talk Title",
  "type": "workshop / seminar / conference / invited",
  "venue": "Venue Name",
  "location": "City, Country",
  "date": "2025-09-08",
  "year": "2025",
  "slides": "https://slides.com/...",
  "video": "https://youtube.com/...",
  "abstract": "Talk abstract...",
  "coauthors": ["Author One", "Author Two"]
}
```

2.2.5 Blog Posts (`posts/*.meta.json`)

Blogs are automatically incorporated into the structure of the website by creating a suitable `[yyyy]-[mm]-[dd]-[title].tex` file under the posts directory. We also need to manually add a `[title].meta.json` which consists of the following:

```
{
  {
  "title": "Title of Blog Page",
  "tags": ["tag1", "tag2"],
  "abstract": "Short abstract of blog..."
  }
}
```

2.3 Build System and Automation

The Makefile provides several build targets:

- `all`: Full build
- `blog`: Blog posts only
- `main`: Main pages only
- `pub`: Publications only
- `clean`: Clean generated files
- `help`: Show help

2.4 CSS Architecture

The styling system uses a modular CSS approach:

- `base.css`: Core variables, typography, and base styles
- `layout.css`: Layout components and responsive design
- `components.css`: Reusable UI components
- `pages.css`: Page-specific styles
- `markdown.css`: Content styling for blog posts
- `main.css`: Main stylesheet that imports all others

3 Other Notes / Changes

This is my first attempt at creating a website with this many automated features, and there are some parts of the current management that feels suboptimal. Whenever there is substantial change to the format, this guide will hopefully be updated accordingly to incorporate these features. I will keep this section to record any previous features of the page if there ever is. I probably won't write much if there is change to the HTML/CSS format, but rather when there is change to the main pipeline which transforms the metadata files.