TiefDownConverter Documentation

Tiefseetauchner et al.

March 11, 2025

Contents

1 The what?

If you want to skip the funny written explanations, skip to the Usage section.

Well, that's a good question. Ask it later.

Jk, of course you may ask it now. TiefDown is a project format I made up to make it easier to convert my markdown files into something pretty. As a matter of fact, this documentation is managed by a TiefDown project!

The important thing is that this isn't a markdown parser, replacement or anything like that. It's a project format, and it's not even a format, it's pretty much just a manifest file and an executable.

1.1 Why?

I wonder myself every day. But alas, I should know, I wrote this cluster**** so let me explain. The initial concept was born from pain (as many are). I was pretty tired of exporting my markdown files, then converting them, overwriting my old files, then converting them again, overwriting all history in the process, it was just. A mess.

So I did what any sane person would do. I learned python.

Well, I'm being facetious. I didn't "learn python", I just used expanded my capabilities to calling programs from the command line.

So my script, at first, just called pandoc, then pdflatex, and then pdflatex again for good measure. It created a pdf, overwriting my old one. It was basically just converting a single markdown file into a pdf with a basic TeX template (in my case, LiX Novel).

Then I realised that writing a 40 chapter story in a single markdown file was even dumber than whatever I made in python. So I added a little combination logic. In the process, I had to write lua filters as well, and then I added versioning, and then I added conversion to multiple different pdfs and then I added epub support and you know what. That was a dumb idea. The python script soon reached 200 lines of code, which was untanable.

So yeah, I decided to make a new book. And of course. ***Everything*** broke. Instantly. I had to copy and paste things, adjust my python script, rewrote it bit and boom, suddenly I had two different projects with different processes, different outputs, different versions, different everything.

And then. I started a third book. Aaaand the python script didn't really fulfill my needs, so I rewrote it in bash. But worse.

I thought I had it all figured out. With python. Then bash. Then I started a short story and lost my ******* mind.

1.2 How, oh wise programmer, did you solve this problem?

I'm glad you asked! I'm glad. I. I hope you asked? Well, regardless of whether or not you did, I'll tell you.

I learned rust

For real this time, I learned a completely new programming language just for this. But there was a reason, or a few rather:

1. I wanted cross platform support

2. I wanted a single executable
3. I needed a language with good CLI support because, believe it or not, I'm *awful* at GUIs
4. I'm crazy

These reasons led me to two options: python, a language I was somewhat familiar with, but didn't particularly enjoy writing in, and rust, a language I had never written in before, but was very interested in.

Evidently, I chose rust.

So I started. A CLI interface, command line calls, so on. So here's the rundown of how it works internally:

- You initialise a project with `tiefdown init`. This creates a few bits and bobs, but importantly the `manifest.toml` file. This contains all the information needed to actually manage and convert the project.
- You can then manipulate the project, so on so forth.
- When you added your markdown files to the Markdown directory, running `tiefdown convert` will do a few things:
  - Create a new folder for the current compilation. That way, you have a history.
  - Combine all the markdown files into one megafile called `combined.md`.
  - Run Pandoc conversion to TeX, Epub, or Typst. This uses lua filters that are defined in the `manifest.toml` file.
  - Run xelatex on all TeX templates, typst on all Typst templates, so on. It even supports Epub conversion.
  - Copies the files around so you end up with your output files in the right places.

Isn't that simple?

It isn't. But oh well. We've got a lot of work to do on this, and if you're interested, don't shy away from the Contributing section!

1.3 So, what's the point?

Really? Making my life easier. If I can make yours easier as well, then I'm the happiest woman alive.

2 Usage

The basic usage of `tiefdownconverter` is relatively simple. The difficult part is understanding the templating system and how to customise it for your usecases. Presets can only do so much.

2.1 Installation

Currently the only way to install `tiefdownconverter` is to either build it yourself or download a precompiled binary from the releases page. Then just add it to the path and you're good to go. You can of course also just call it relatively by placing the binary in your project folder or something like that.

There are a few dependencies that you need to install.

- Pandoc: Conversion from Markdown to TeX, Typst and Epub.
- A TeX distribution: For converting TeX files to PDF. It has to include xelatex.
  - If using TeX Live you may need to additionally install `texlive-xetex` depending on your system.
  - If using MikTeX, no need to do anything.
- Typst: For converting Typst files to PDF.

Now you should be able to run `tiefdownconverter` from the command line. You can test it by initialising a test project using `tiefdown init testproject` and running `tiefdown convert` in the project directory or `tiefdown convert -p testproject`.

2.2 Getting started

3 Usage Details

Below are the usage details for the various commands. **Note:** These are autogenerated! For clearer documentation, please see the Usage section.

3.1 `tiefdownconverter`

Version: `tiefdownconverter 0.1.0`

3.1.1 Usage:

```
A CLI tool for managing TiefDown Projects

Usage: tiefdownconverter <COMMAND>

Commands:
  convert  Convert a TiefDown project. By default, it will convert the
    current directory.
  init     Initialize a new TiefDown project.
  project  Update the TiefDown project.
  help     Print this message or the help of the given subcommand(s)

Options:
  -h, --help     Print help
  -V, --version  Print version
```

3.1.2 Subcommands:

- convert
- init
- project

3.2 `tiefdownconverter convert`

Version: `tiefdownconverter 0.1.0`

3.2.1 Usage:

```
Convert a TiefDown project. By default, it will convert the current
    directory.

Usage: tiefdownconverter convert [OPTIONS]

Options:
  -p, --project <PROJECT>         The project to convert. If not provided,
    the current directory will be used.
  -t, --templates <TEMPLATES>...  The templates to use. If not provided,
    the default templates from the manifest file will be used.
```

```
  -h, --help                        Print help
```

3.3 tiefdownconverter init

Version: `tiefdownconverter 0.1.0`

3.3.1 Usage:

```
Initialize a new TiefDown project.

Usage: tiefdownconverter init [OPTIONS] [PROJECT]

Arguments:
  [PROJECT]  The project to initialize. If not provided, the current
    directory will be used.

Options:
  -t, --templates <TEMPLATES>...     The preset templates to use. If not
    provided, the default template.tex will be used.
                                     For custom templates, use the update
    command after initializing the project.
                                     If using a LiX template, make sure to
    install the corresponding .sty and .cls files from
    https://github.com/NicklasVraa/LiX. Adjust the metadata in
    template/meta.tex accordingly.
                                      [possible values: template.tex,
    booklet.tex, lix_novel_a4.tex, lix_novel_book.tex, template_typ.typ,
    default_epub]
  -n, --no-templates                 Do not include the default templates.
    You will need to add templates manually with Update
  -f, --force                        Delete the project if it already
    exists.
  -m, --markdown-dir <MARKDOWN_DIR>  The directory where the Markdown files
    are located. If not provided, Markdown/ will be used.
  -h, --help                         Print help
```

3.4 tiefdownconverter project

Version: `tiefdownconverter 0.1.0`

3.4.1 Usage:

```
Update the TiefDown project.

Usage: tiefdownconverter project [PROJECT] <COMMAND>

Commands:
  add-template     Add a new template to the project.
  remove-template  Remove a template from the project.
```

```
  update-template  Update a template in the project.
  update-manifest  Update the project manifest.
  list-templates   List the templates in the project.
  validate         Validate the TiefDown project structure and metadata.
  clean            Clean temporary files from the TiefDown project.
  help             Print this message or the help of the given
    subcommand(s)

Arguments:
  [PROJECT]  The project to edit. If not provided, the current directory
    will be used.

Options:
  -h, --help  Print help
```

3.4.2 Subcommands:

- add-template
- remove-template
- update-template
- update-manifest
- list-templates
- validate
- clean

3.5 `tiefdownconverter project add-template`

Version: `tiefdownconverter 0.1.0`

3.5.1 Usage:

```
Add a new template to the project.

Usage: tiefdownconverter project add-template <TEMPLATE> [TEMPLATE_FILE]
    [TEMPLATE_TYPE] [OUTPUT] [FILTERS]...

Arguments:
  <TEMPLATE>       The name of the template to create. If using a LiX
    template, make sure to install the corresponding .sty and .cls files
    from https://github.com/NicklasVraa/LiX. Adjust the metadata in
    template/meta.tex accordingly.
  [TEMPLATE_FILE]  The file to use as the template. If not provided, the
    template name will be used.
  [TEMPLATE_TYPE]  The type of the template. If not provided, the type will
    be inferred from the template file. [possible values: tex, typst, epub]
```

```
  [OUTPUT]         The output file. If not provided, the template name will
    be used.
  [FILTERS]...     The luafilters to use for pandoc conversion of this
    templates markdown.

Options:
  -h, --help  Print help
```

3.6 tiefdownconverter project remove-template

Version: `tiefdownconverter 0.1.0`

3.6.1 Usage:

```
Remove a template from the project.

Usage: tiefdownconverter project remove-template --template <TEMPLATE>

Options:
  -t, --template <TEMPLATE>  The template to remove.
  -h, --help                 Print help
```

3.7 tiefdownconverter project update-template

Version: `tiefdownconverter 0.1.0`

3.7.1 Usage:

```
Update a template in the project.

Usage: tiefdownconverter project update-template [OPTIONS] <TEMPLATE>

Arguments:
  <TEMPLATE>  The template to update.

Options:
      --template-file <TEMPLATE_FILE>
          The file to use as the template. If not provided, the template
    name will be used.
      --template-type <TEMPLATE_TYPE>
          The type of the template. If not provided, the type will be
    inferred from the template file.
          Changing this is not recommended, as it is highly unlikely the
    type and only the type has changed. It is recommended to create a new
    template instead. [possible values: tex, typst, epub]
      --output <OUTPUT>
          The output file. If not provided, the template name will be used.
      --filters <FILTERS>...
          The luafilters to use for pandoc conversion of this templates
```

```
    markdown.
      --add-filters <ADD_FILTERS>...
          The luafilters add to the template.
      --remove-filters <REMOVE_FILTERS>...
          The luafilters to remove from the template.
  -h, --help
          Print help
```

3.8 tiefdownconverter project update-manifest

Version: tiefdownconverter 0.1.0

3.8.1 Usage:

```
Update the project manifest.

Usage: tiefdownconverter project update-manifest [OPTIONS] [PROJECT]

Arguments:
  [PROJECT]  The project to manipulate. If not provided, the current
    directory will be used.

Options:
  -m, --markdown-dir <MARKDOWN_DIR>  The directory where the Markdown files
    are located.
  -h, --help                         Print help
```

3.9 tiefdownconverter project list-templates

Version: tiefdownconverter 0.1.0

3.9.1 Usage:

```
List the templates in the project.

Usage: tiefdownconverter project list-templates [PROJECT]

Arguments:
  [PROJECT]  The project to manipulate. If not provided, the current
    directory will be used.

Options:
  -h, --help  Print help
```

3.10 tiefdownconverter project validate

Version: tiefdownconverter 0.1.0

3.10.1 Usage:

```
Validate the TiefDown project structure and metadata.

Usage: tiefdownconverter project validate [PROJECT]

Arguments:
  [PROJECT]  The project to validate. If not provided, the current
    directory will be used.

Options:
  -h, --help  Print help
```

3.11 `tiefdownconverter project clean`

Version: `tiefdownconverter 0.1.0`

3.11.1 Usage:

```
Clean temporary files from the TiefDown project.

Usage: tiefdownconverter project clean [PROJECT]

Arguments:
  [PROJECT]  The project to clean. If not provided, the current directory
    will be used.

Options:
  -h, --help  Print help
```

4 Contributing