Melthon

Release 2.1.0

Apr 21, 2021

Contents

CHAPTER 1

Overview

docs	
tests	
package	

Minimalistic static site generator

- Free software: GNU GPLv3 license

1.1 Installation

Use Docker (preferred way):

```
# --rm           : Remove container after execution
# -u ${UID}      : Run container as current user
# -v"$(pwd):/src": Make source and output accessible inside container
docker run --rm -u ${UID} -v"$(pwd):/src" jenswbe/melthon
```

Use pip:

```
pip install melthon
```

1.2 Documentation

https://python-melthon.readthedocs.io/

CHAPTER 2

Installation

Use Docker (preferred way):

```
# --rm           : Remove container after execution
# -u ${UID}      : Run container as current user
# -v"$(pwd):/src": Make source and output accessible inside container
docker run --rm -u ${UID} -v"$(pwd):/src" jenswbe/melthon
```

Use pip:

```
pip install melthon
```

CHAPTER 3

Usage

3.1 Folder structure

A Melthon project has following folder structure. Folder names can be changed as option to the melthon command.

templates This is the only mandatory folder. It contains your Mako tempates (*.mako) which will be rendered into HTML pages. This folder supports subfolders. To prevent a file from rendering, like base template or reusable parts, name your template `*.template.mako` or `*.part.mako`.

If you want to render the same template multiple times, use suffix `*.repeat.mako`. See *Repeated templates* for more info.

static This folder contents will be copied to the root of the output folder. You can use this folder for static assets like CSS, JavaScript, images, ...

data If you want to have certain information available in a template, e.g. a telephone number, you can provide this information in YAML files. Each YAML file inside the `data` folder will be available as `data['<FILENAME>']` in your templates. E.g. `general.yml` will become `data['general']`.

middleware In this folder, you can provide custom middleware. The middleware will run before and after the rendering of your site. It has access to the context. Please use following template:

```
from melthon.middleware import Middleware


class Middleware1(Middleware):
  def before(self, context):
    # <YOUR CUSTOM CODE>
    return context

  def after(self, context):
    # <YOUR CUSTOM CODE>
    return context
```

You can define multiple middlewares (classes) in the same file. Method `before` or `after` can be omitted in case it's not required.

output This folder will contain the rendered result

3.2 Command options

Melthon currently supports 2 commands: `melthon build` and `melthon clean`. Please use `melthon --help` and `melthon <command> --help` to list the available options.

3.3 Repeated templates

To render the same template multiple times, use suffix `*.repeat.mako` for your template. Next to this, you'll have to supply a `repeat.yml` in the root of your site.

This file contains the mapping between your template and the collection in the data files. E.g. to use the list `events` in file `data/general.yml` for template `events.repeat.mako`, you have to provide following `repeat.yml`:

> events: "/general/events"

Melthon expects an attribute `slug` for each item in the repeat collection. The slug defines the output name of the repeated page. The whole item will be passed in variable `page` inside the template.

Tip: To have an index page for your repeated pages, you can specify a template with the same name. E.g. `events.mako` and `events.repeat.mako`

You can check https://github.com/JenswBE/python-melthon/tree/master/tests/test_site if you want an example of repeated pages.

CHAPTER 4

Reference

4.1 melthon

CHAPTER 5

Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

5.1 Bug reports

When reporting a bug please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

5.2 Documentation improvements

Melthon could always use more documentation, whether as part of the official Melthon docs, in docstrings, or even on the web in blog posts, articles, and such.

5.3 Feature requests and feedback

The best way to send feedback is to file an issue at https://github.com/jenswbe/python-melthon/issues.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that code contributions are welcome :)

5.4 Development

To set up *python-melthon* for local development:

1. Fork python-melthon (look for the "Fork" button).
2. Clone your fork locally:

```
git clone git@github.com:jenswbe/python-melthon.git
```

3. Create a branch for local development:

```
git checkout -b name-of-your-bugfix-or-feature
```

   Now you can make your changes locally.

4. When you're done making changes, run all the checks, doc builder and spell checker with tox one command:

```
tox
```

5. Commit your changes and push your branch to GitHub:

```
git add .
git commit -m "Your detailed description of your changes."
git push origin name-of-your-bugfix-or-feature
```

6. Submit a pull request through the GitHub website.

5.4.1 Pull Request Guidelines

If you need some code review or feedback while you're developing the code just make the pull request.

For merging, you should:

1. Include passing tests (run `tox`)[1].
2. Update documentation when there's new API, functionality etc.
3. Add a note to `CHANGELOG.rst` about the changes.
4. Add yourself to `AUTHORS.rst`.

5.4.2 Tips

To run a subset of tests:

```
tox -e envname -- pytest -k test_myfeature
```

To run all the test environments in *parallel* (you need to `pip install detox`):

```
detox
```

[1] If you don't have all the necessary python versions available locally you can rely on Travis - it will run the tests for each change you add in the pull request.
It will be slower though ...

CHAPTER 6

Authors

- Jens Willemsens - https://jensw.be

CHAPTER 7

Changelog

7.1 2.1.0 (2020-04-21)

- Refresh project by adding support for Docker, improving unit tests and freezing dependencies

7.2 2.0.0 (2020-05-22)

- Simplify project by dropping support for all Python versions except for the current version (3.8)
- Add support for repeated pages (e.g. events -> event details)

7.3 1.4.0 (2020-01-19)

- Add Python 3.8 support
- Drop Python 3.4 support
- Don't remove output folder, but remove it's contents instead. Removing the folder would cause permission issues, when serving with a local web server for development.

7.4 1.3.0 (2019-11-27)

- Add option to render Mako exceptions

7.5 1.2.0 (2019-10-18)

- Set default input encoding to UTF-8

7.6 1.1.0 (2019-10-08)

- Added subdirectory support
- Added support for pretty urls (default)
- Both <PAGE NAME>.template.mako and <PAGE NAME>.part.mako files won't be rendered now

7.7 1.0.0 (2019-09-25)

- Template rendering is working
- Added YAML data file support
- Improved program verbosity

7.8 0.1.0 (2019-09-21)

- Drop support for Python 2
- Implement middlewares

7.9 0.0.0 (2019-09-21)

- First release on PyPI

CHAPTER 8

Indices and tables

- genindex
- modindex
- search

Python Module Index

m

Index

M