
Recommonmark Documentation

Release 0.4.0

Lu Zero, Eric Holscher, and contributors

Apr 30, 2017

Contents

1	Contents	3
2	Getting Started	5
2.1	Links	5
2.2	AutoStructify	5
3	Development	7
4	Why a bridge?	9
5	Why another bridge to docutils?	11
6	Acknowledgement	13

A `docutils`-compatibility bridge to `CommonMark`.

This allows you to write `CommonMark` inside of `Docutils` & `Sphinx` projects.

Documentation is available on Read the Docs: <http://recommonmark.readthedocs.org>

CHAPTER 1

Contents

- API Reference
- AutoStructify Component

CHAPTER 2

Getting Started

To use `recommonmark` inside of Sphinx only takes 2 steps. First you install it:

```
pip install recommonmark
```

Then add this to your Sphinx `conf.py`:

```
from recommonmark.parser import CommonMarkParser

source_parsers = {
    '.md': CommonMarkParser,
}

source_suffix = ['.rst', '.md']
```

This allows you to write both `.md` and `.rst` files inside of the same project.

Links

For all links in commonmark that aren't explicit URLs, they are treated as cross references with the `:any:` role. This allows referencing a lot of things including files, labels, and even objects in the loaded domain.

AutoStructify

To use the advanced markdown to rst transformations you must add `AutoStructify` to your Sphinx `conf.py`.

```
# At top on conf.py (with other import statements)
import recommonmark
from recommonmark.transform import AutoStructify

# At the bottom of conf.py
```

```
def setup(app):
    app.add_config_value('recommonmark_config', {
        'url_resolver': lambda url: github_doc_root + url,
        'auto_toc_tree_section': 'Contents',
    }, True)
    app.add_transform(AutoStructify)
```

See <https://github.com/rtfd/recommonmark/blob/master/docs/conf.py> for a full example.

AutoStructify comes with the following options. See http://recommonmark.readthedocs.org/en/latest/auto_structify.html for more information about the specific features.

- **enable_auto_toc_tree**: enable the Auto Toc Tree feature.
- **auto_toc_tree_section**: when True, Auto Toc Tree will only be enabled on section that matches the title.
- **enable_auto_doc_ref**: enable the Auto Doc Ref feature.
- **enable_math**: enable the Math Formula feature.
- **enable_inline_math**: enable the Inline Math feature.
- **enable_eval_rst**: enable the evaluate embedded reStructuredText feature.
- **url_resolver**: a function that maps a existing relative position in the document to a http link

CHAPTER 3

Development

You can run the tests by running `tox` in the top-level of the project.

We are working to expand test coverage, but this will at least test basic Python 2 and 3 compatability.

CHAPTER 4

Why a bridge?

Many python tools (mostly for documentation creation) rely on `docutils`. But `docutils` only supports a `ReStructuredText` syntax.

For instance [this issue](#) and [this StackOverflow question](#) show that there is an interest in allowing `docutils` to use markdown as an alternative syntax.

Why another bridge to docutils?

`recommonmark` uses the `python` implementation of `CommonMark` while `remarkdown` implements a stand-alone parser leveraging `parsley`.

Both output a `docutils` document tree and provide scripts that leverage `docutils` for generation of different types of documents.

CHAPTER 6

Acknowledgement

recommonmark is mainly derived from [remarkdown](#) by Steve Genoud and leverages the python CommonMark implementation.

It was originally created by [Luca Barbato](#), and is now maintained in the Read the Docs (rtd) GitHub organization.