docconvert Documentation

Release 2.1.0

Cameron Billingham

Jan 31, 2023

Contents

docconvert will parse one or several python files and retrieve existing docstrings. Then, for all found modules/functions/methods/classes, it will convert docstring formatting (including parameters, returns, and other fields) to the newly specified style.

Currently, the managed styles in input/output are:

Input

- epytext
- reST (re-Structured Text, as used by Sphinx)

Output

- google
- numpy
- epytext
- reST (re-Structured Text, as used by Sphinx)

CHAPTER 1

Contents

1.1 Introduction

1.1.1 Installation

Docconvert can be installed through pip:

```
pip install docconvert
```

1.1.2 Usage

```
usage: docconvert [-h] [-i {guess,rest,epytext}] [-o {google,numpy,rest,epytext}]
                  [--in-place] [-c CONFIG] [-t THREADS] [-v]
                  source

positional arguments:
source                The directory or file to convert.

optional arguments:
-h, --help            show this help message and exit
-i {guess,rest,epytext}, --input {guess,rest,epytext}
                      Input docstring style. (default: guess)
-o {google,numpy,rest,epytext}, --output {google,numpy,rest,epytext}
                      Output docstring style to convert to. (default: google)
--in-place            Write the changes to the input file instead of printing diffs.
-c CONFIG, --config CONFIG
                      Location of configuration file to use.
-t THREADS, --threads THREADS
                      Number of threads to use. (default: cpu count)
-v, --verbose         Log more information.
```

Examples:

Convert files in `src/mypackage` to google using 4 threads.

```
docconvert --output google --threads 4 src/mypackage/
```

Convert file `src/mypackage/myfile.py` from rest to google.

```
docconvert --input rest --output google src/mypackage/myfile.py
```

1.1.3 Custom Configuration

You can configure optional conversion arguments in a json config file. Just specify a config filepath to the commandline tool.

```
docconvert --config path/to/config.json src/mypackage/
```

Example configuration file:

```
{
    "input_style": "rest",
    "output_style": "google",
    "accepted_shebangs": [
        "python"
    ],
    "output": {
        "first_line": true,
        "replace_quotes": "\"\"\"",
        "standard_indent": "    ",
        "tab_length": 4,
        "realign": true,
        "max_line_length": 72,
        "use_optional": false,
        "remove_type_back_ticks": "true",
        "use_types": true,
        "separate_keywords": false
    }
}
```

Let's walk through the configuration options currently available.

input_style

```
"input_style": "rest"
```

This option lets you specify the input style directly if you know it. By default, input style is guessed by looping through docstring lines, and matching on the first parser that recognizes a line.

output_style

```
"output_style": "google"
```

Specify the output docstring style. Defaults to **google**.

accepted_shebangs

```
"accepted_shebangs": [
    "python"
]
```

Provide a list of file shebangs that are considered python scripts. If a file does not have an extension, docconvert will check if the file's first line starts with a shebang (`#!`) and contains an item in the accepted shebang list. By default only shebangs that contain "python" will match.

For example, with the default settings, if a file starts with `#!python2.7` it will match, but a file starting with `#!jython` would not.

output

All configuration under the **output** key is output specific options.

first_line

```
"first_line": true
```

If first_line is true, the output docstring's first line will be adjacent to the docstring starting quotes. Default is True.

quotes

```
"quotes": "\"\"\""
```

Specify the docstring quotes as single or double. By default uses source quotations.

standard_indent

```
"standard_indent": "    "
```

Specify the standard indentation for the project. Defaults to 4-spaces.

tab_length

```
"tab_length": 4
```

Define the length of a tab in spaces. If **standard_indent** is defined with tabs, this value will be used to calculate line lengths for realigning. Defaults to 4.

realign

```
"realign": true
```

Realign continuous descriptions, wrapping to max line length. Defaults to True.

For example, a rest docstring like

```
"""
:param input: This is the input dict. Make sure it is a good dict
    with lots of happy items.
:type input: collections.OrderedDict
"""
```

would be realigned to a google docstring with **max_line_length: 72** as

```
"""
Args:
    input (collections.OrderedDict): This is the input dict. Make
        sure it is a good dict with lots of happy items.
"""
```

max_line_length

```
"max_line_length": 72
```

Specify max line length used in realignment. Defaults to the PEP8 docstring length of 72 characters.

use_optional

```
"use_optional": false
```

If True, append `optional` to parameter types that are keywords. Defaults to False.

> *Note:*
> *If separate_keywords is True, optional is dropped from all type definitions.*

remove_type_back_ticks

```
"remove_type_back_ticks": "true"
```

Remove back ticks from types. Defaults to "true". If this is on, isolated back ticks around type definitions are removed. This option has 3 modes:

- `"false"`: No back ticks will be removed.
- `"true"`: Back ticks will be removed, except from sphinx directives. For example:
  - `` `list` of `str` `` becomes `list of str`
  - `:py:class:`Test`` stays as `:py:class:`Test``
  - `` lot`s of `bool`s `` becomes `lot`s of bools`
- `"directives"`: All back ticks, including directives, will be removed. For example:
  - `` `list` of `str` `` becomes `list of str`
  - `:py:class:`Test`` becomes `Test`
  - `` lot`s of `bool`s `` becomes `lot`s of bools`

use_types

```
"use_types": true
```

Use types in argument output. Defaults to True. If False, argument, keyword-argument, and attribute type definitions will be skipped. This could be turned False for Python 3, where Sphinx recognizes annotations.

separate_keywords

```
"separate_keywords": false
```

Separate keyword-arguments into their own docstring section. Defaults to False. If set to False, all keyword-arguments are documented with the other arguments.

1.2 Contributing

1.2.1 Running the Tests

Tests are executed through tox.

```
tox
```

Tests are written with pytest. Please add unit tests under the *tests/* directory to cover any new functionality you have added.

1.2.2 Code Style

Code is formatted using black.

You can check your formatting using black's check mode:

```
tox -e formatting
```

You can also use pre-commit to format every commit with black:

```
pip install pre-commit
pre-commit install
```

1.2.3 Building Documentation

You can build the documentation through tox.

```
tox -e docs
```

The built documentation will be output to doc/build

1.2.4 Releasing

Before releasing please remember to:

1. Run the tests and check that they pass
2. Build the new documentation and check it
3. Update version in `doc/source/conf.py`, `src/docconvert/__init__.py`, and `setup.cfg`
4. Add new release notes to `RELEASE_NOTES.rst`
5. Commit, tag the version number, and push the changes

To release

```
git clean -idx
python setup.py sdist bdist_wheel
twine upload dist/*
```

CHAPTER 2

Release Notes

2.1 Release Notes

See previous releases here

Versions follow Semantic Versioning (`<major>.<minor>.<patch>`).

2.1.1 v2.1.0 (2023-01-30)

Features

- Support async function definitions in docstring parsing.

2.1.2 v2.0.0 (2020-07-11)

Breaking Changes

- Remove google specific output from config and moved `use_types` and `separate_keywords` into global output config.

Features

- Support numpy docstring output.
- Support reST docstring output.
- Support epytext docstring output.

2.1.3 v1.2.0 (2020-06-08)

Features

- #1: Support converting docstrings nested inside functions.
- Add support for Python 3.8.

2.1.4 v1.1.0 (2018-06-17)

Features

- Output diffs by default, add in-place flag for overwriting files in place.
- Can convert attribute docstrings.

2.1.5 v1.0.0 (2018-03-12)

- Initial release.

CHAPTER 3

Indices and tables

- genindex
- modindex
- search