

Documenting Python Packages

Docs or it didn't happen!

Andrey Smelter

2019/04/11 (updated: 2019-04-25)

\$ whoami

Andrey (pronounced similar to André) Smelter

Postdoctoral Scholar at the [University of Kentucky Markey Cancer Center](#)

Links:

- : github.com/andreysmelter
- **in:** [linkedin.com/in/andreysmelter](https://www.linkedin.com/in/andreysmelter)
- : [instagram.com/andreysmelter](https://www.instagram.com/andreysmelter)
- : twitter.com/smelandr

Slides are available: bit.ly/DerbyPyWTD



A regular slide with text or code.

A transition slide or terminal output.



Why write docs?

You want **people to use** your code

You want your **code to be better**

You want **contributors**

You will be using **your code in [X] months**

To document or not to document?

"There is a magical feeling that happens when you release your code. It comes in a variety of ways, but it always hits you the same.
Someone is using my code?! A mix of terror and excitement."

Get Started

The Python Standard Library



"Batteries Included" Philosophy

Having a **rich** and **versatile** standard library

Which is **immediately available**

Without making the user **download separate** packages

pydoc

built-in

pydoc

The pydoc module **automatically** generates documentation from Python modules.

The documentation can be presented as:

- pages of **text on the console**
- served to a **web browser**
- or saved to **HTML files**

Let's use quadratic equation as an example

- The **quadratic equation** formula:

$$ax^2 + bx + c = 0$$

- The quadratic formula for the **roots**:

$$x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

- The **equations/quadratic.py** module:

```
import cmath

def quadratic(a, b, c):
    discriminant = cmath.sqrt(b**2.0 - 4.0 * a * c)
    x1 = (-b + discriminant) / (2.0 * a)
    x2 = (-b - discriminant) / (2.0 * a)
    return x1, x2
```

pydoc

The pydoc module "automatically" generates documentation from Python modules

```
$ python3 -m pydoc equations.quadratic
```

```
Help on module quadratic:
```

```
NAME
```

```
    quadratic
```

```
FUNCTIONS
```

```
    quadratic(a, b, c)
```

```
FILE
```

```
    equations/quadratic.py
```


pydoc

- Add **module-level** docstring.
- Add **function-level** docstring.

```
"""
This module contains routines for finding
roots of quadratic equations.
"""

import cmath

def quadratic(a, b, c):
    """Solve quadratic equation."""
    discriminant = cmath.sqrt(b**2.0 - 4.0 * a * c)
    x1 = (-b + discriminant) / (2.0 * a)
    x2 = (-b - discriminant) / (2.0 * a)
    return x1, x2
```

pydoc

The pydoc module **automatically** generates documentation from Python modules

```
$ python3 -m pydoc equations.quadratic
```

Help on module equations.quadratic in equations:

NAME

equations.quadratic

DESCRIPTION

This module contains routines for finding roots of quadratic equations.

FUNCTIONS

quadratic(a, b, c)
Solve quadratic equation.

FILE

equations/quadratic.py

pydoc: HTML docs

```
$ python3 -m pydoc -w equations.quadratic
```

pydoc

- **Module-level** HTML doc:

The screenshot shows a web page generated by pydoc for a Python module. The page has a blue header bar with the text 'examples.2 example.equations.quadratic' and two links: 'index' and 'equations/quadratic.py'. Below the header, a light blue box contains the text 'This module contains routines for finding roots of quadratic equations.' Below this is a purple bar with the heading 'Modules'. Underneath the purple bar, a link 'math' is displayed. Further down is an orange bar with the heading 'Functions'. Below the orange bar, the function 'quadratic(a, b, c)' is listed with the description 'Solve quadratic equation.'

[examples.2 example.equations.quadratic](#) [index](#) [equations/quadratic.py](#)

This module contains routines for finding roots of quadratic equations.

Modules

[math](#)

Functions

quadratic(a, b, c)
Solve quadratic equation.

pydoc

Pros:

- **Built-in** docs generator.
- Used by `>>> help()` function to pull and display docstrings from functions/methods, modules, and packages.

Cons:

- Static HTML docs are very **retro style**.



doctest

built-in

doctest

Searches for pieces of text that **look like interactive Python sessions**.

Then **executes those sessions** to verify that they work exactly as shown.

doctest

```
def quadratic(a, b, c):  
    """Solve quadratic equation.  
  
    For example:  
    >>> x1, x2 = quadratic(a=2, b=-8, c=-24)  
    >>> x1  
    (6+0j)  
    >>> x2  
    (-2+0j)  
    >>> 2*x1**2 - 8*x1 - 24  
    0j  
    >>> 2*x2**2 - 8*x2 - 24  
    0j  
    """  
    discriminant = cmath.sqrt(b**2.0 - 4.0 * a * c)  
    x1 = (-b + discriminant) / (2.0 * a)  
    x2 = (-b - discriminant) / (2.0 * a)  
    return x1, x2  
  
if __name__ == "__main__":  
    import doctest  
    print(doctest.testmod())
```

doctest

```
$ python3 -m equations.quadratic  
TestResults(failed=0, attempted=5)
```

doctest

```
$ python3 -m equations.quadratic -v
```

```
Trying:
```

```
    x1, x2 = quadratic(a=2, b=-8, c=-24)
```

```
Expecting nothing
```

```
ok
```

```
Trying:
```

```
    x1
```

```
Expecting:
```

```
    (6+0j)
```

```
ok
```

```
Trying:
```

```
    x2
```

```
Expecting:
```

```
    (-2+0j)
```

```
ok
```

```
Trying:
```

```
    2*x1**2 - 8*x1 - 24
```

```
Expecting:
```

```
    0j
```

```
ok
```

```
Trying:
```

```
    2*x2**2 - 8*x2 - 24
```

```
Expecting:
```

```
    0j
```

```
ok
```

```
1 items had no tests:
```

```
    __main__
```

```
1 items passed all tests:
```

```
    5 tests in __main__.quadratic
```

```
5 tests in 2 items.
```

```
5 passed and 0 failed.
```

```
Test passed.
```

```
TestResults(failed=0, attempted=5)
```


doctest

```
def quadratic(a, b, c):  
    """Solve quadratic equation.  
  
    For example:  
    >>> x1, x2 = quadratic(a=2, b=-8, c=-24)  
    >>> x1  
    (8+0j)  
    >>> x2  
    (-2+0j)  
    >>> 2*x1**2 - 8*x1 - 24  
    0j  
    >>> 2*x2**2 - 8*x2 - 24  
    0j  
    """  
    discriminant = cmath.sqrt(b**2.0 - 4.0 * a * c)  
    x1 = (-b + discriminant) / (2.0 * a)  
    x2 = (-b - discriminant) / (2.0 * a)  
    return x1, x2  
  
if __name__ == "__main__":  
    import doctest  
    print(doctest.testmod())
```

doctest

```
$ python3 -m equations.quadratic
```

```
=====
File "equations/quadratic.py", line 14, in __main__.quadratic
Failed example:
```

```
    x1
```

```
Expected:
```

```
    (8+0j)
```

```
Got:
```

```
    (6+0j)
=====
```

```
1 items had failures:
```

```
  1 of   5 in __main__.quadratic
```

```
Test Failed 1 failures.
```

```
TestResults(failed=1, attempted=5)
```

doctest

Pros:

- **Built-in** docs string test.
- Forces you to write **high-quality examples** in your doc strings.
- Makes sure that your doc string is **up-to-date**.

Cons:

- May not be useful for more **complex cases**.



CHEESE SHOP

a.k.a. Python Package Index (PyPI)

Sphinx

```
python3 -m pip install Sphinx
```


Sphinx

Makes it easy to create **beautiful documentation** for Python projects.

Uses **reStructuredText** as its markup language format.

Sphinx: sphinx-quickstart

```
$ mkdir docs && cd docs  
$ sphinx-quickstart
```

```
> Separate source and build directories (y/n) [n]:  
The project name will occur in several places in the built documentation.  
> Project name: equations  
> Author name(s): Andrey Smelter  
> Project release []: 0.1.0  
> Project language [en]:
```

```
Creating file ./conf.py.  
Creating file ./index.rst.  
Creating file ./Makefile.  
Creating file ./make.bat.
```

Finished: An initial directory structure has been created.
You should now populate your master file `./index.rst` and
create other documentation source files. Use the Makefile
to build the docs, like so:

```
make builder
```

where "builder" is one of the supported builders,
e.g. html, latex or linkcheck.

Sphinx: sphinx-quickstart

```
$ tree docs
```

```
.
├── docs
│   ├── _build
│   ├── conf.py
│   ├── index.rst
│   ├── make.bat
│   ├── Makefile
│   ├── _static
│   └── _templates
└── equations
    ├── __init__.py
    └── quadratic.py
```

Sphinx: conf.py

```
# import os
# import sys
# sys.path.insert(0, os.path.abspath('.'))

project = 'equations'
copyright = '2019, Andrey Smelter'
author = 'Andrey Smelter'

release = '0.1.0'

extensions = [

templates_path = ['_templates']

html_theme = 'alabaster'

html_static_path = ['_static']
```

Sphinx: index.rst

Welcome to equations's documentation!

=====

```
.. toctree::  
    :maxdepth: 2  
    :caption: Contents:
```

Sphinx: index.rst

Welcome to equations's documentation!

=====

The library for solving equations.

.. note::

The ``equations`` library currently can only solve quadratic equations.

.. toctree::

:maxdepth: 2

:caption: Contents:

Sphinx: Makefile

```
$ make html
```

```
Running Sphinx v2.0.1
```

```
making output directory... done
```

```
building [mo]: targets for 0 po files that are out of date
```

```
building [html]: targets for 1 source files that are out of date
```

```
updating environment: 1 added, 0 changed, 0 removed
```

```
reading sources... [100%] index
```

```
looking for now-outdated files... none found
```

```
pickling environment... done
```

```
checking consistency... done
```

```
preparing documents... done
```

```
writing output... [100%] index
```

```
generating indices... genindex
```

```
writing additional pages... search
```

```
copying static files... done
```

```
copying extra files... done
```

```
dumping search index in English (code: en) ... done
```

```
dumping object inventory... done
```

```
build succeeded.
```

The HTML pages are in `_build/html`.

Sphinx: html docs

equations

Navigation

Quick search

 Go

Welcome to equations's documentation!

The library for solving equations.

Note:

The `equations` library currently can only solve quadratic equations.

Indices and tables

- [Index](#)
- [Module Index](#)
- [Search Page](#)

©2019, Andrey Smelter. | Powered by [Sphinx 2.0.1](#) & [Alabaster 0.7.12](#) | [Page source](#)

Sphinx: write doc string in reStructuredText format

```
def quadratic(a, b, c):  
    """Solve quadratic equation.  
  
    :param float a: a coefficient of a quadratic equation.  
    :param float b: b coefficient of a quadratic equation.  
    :param float c: c coefficient of a quadratic equation.  
    :return: quadratic equation roots.  
    :rtype: :py:class:`tuple`.  
    """  
    discriminant = cmath.sqrt(b**2.0 - 4.0 * a * c)  
    x1 = (-b + discriminant) / (2.0 * a)  
    x2 = (-b - discriminant) / (2.0 * a)  
    return x1, x2
```

Sphinx: html docs

equations

Navigation

Contents:

The equations API
Reference

Quick search

Welcome to equations's documentation!

The library for solving equations.

Note:

The `equations` library currently can only solve quadratic equations.

Contents:

- [The equations API Reference](#)

Indices and tables

- [Index](#)
- [Module Index](#)
- [Search Page](#)

©2019, Andrey Smelter. | Powered by [Sphinx 2.0.1](#) & [Alabaster 0.7.12](#) | [Page source](#)

Sphinx: html docs

equations

Navigation

Contents:

The equations API
Reference

Quick search

The equations API Reference

This package provides routines for solving equations.

This module contains routines for finding roots of quadratic equations.

`equations.quadratic.quadratic(a, b, c)`

[\[source\]](#)

Solve quadratic equation.

Parameters:

- **a** ([float](#)) – a coefficient of a quadratic equation.
- **b** ([float](#)) – b coefficient of a quadratic equation.
- **c** ([float](#)) – c coefficient of a quadratic equation.

Returns:

quadratic equation roots.

Return type:

tuple

©2019, Andrey Smelter. | Powered by [Sphinx 2.0.1](#) & [Alabaster 0.7.12](#) | [Page source](#)

Sphinx: useful extensions

```
extensions = [  
    'sphinx.ext.autodoc',  
    'sphinx.ext.doctest',  
    'sphinx.ext.todo',  
    'sphinx.ext.coverage',  
    'sphinx.ext.mathjax',  
    'sphinx.ext.viewcode',  
    'sphinx.ext.intersphinx',  
    'nbsphinx'  
]
```

Sphinx: host on readthedocs.org

- **Register** an account on readthedocs.org.
- **Point** readthedocs to your **GitHub repo**.
- Add path to `requirements.txt` file under advanced settings.
- Push **commit to** trigger documentaion **build**.



Sphinx

Pros:

- Creates **beautiful** docs.
- Provides useful **extensions**.
- **Customizable** (templates and themes).
- Uses **reStructuredText**.

Cons:

- Uses **reStructuredText** (not really cons).

MkDocs

```
python3 -m pip install mkdocs
```

MkDocs

MkDocs is a **fast, simple** and **beautiful** static site generator for building project documentation.

Uses **Markdown** as its markup language format.

MkDocs: quickstart

```
$ mkdocs new equations
```

```
$ tree equations
```

```
├── docs
│   └── index.md
├── equations
│   ├── __init__.py
│   └── quadratic.py
└── mkdocs.yml
```

MkDocs: mkdocs.yml

```
site_name: equations documentation!
```

MkDocs: docs/index.md

```
# Welcome to `equations` documentation!
```

```
The library for solving equations.
```

```
### Note
```

```
The ``equations`` library currently can only solve  
quadratic equations.
```

MkDocs: html docs

```
$ mkdocs serve
```

```
INFO      - Building documentation...
```

```
INFO      - Cleaning site directory
```

```
[I 190425 02:27:22 server:298] Serving on http://127.0.0.1:8000
```

```
[I 190425 02:27:22 handlers:59] Start watching changes
```

```
[I 190425 02:27:22 handlers:61] Start detecting changes
```

MkDocs: html docs

equations documentation! 🔍 Search

Welcome to equations documentation!
Note

Welcome to equations documentation!

The library for solving equations.

Note

The `equations` library currently can only solve quadratic equations.

Documentation built with MkDocs.

MkDocs

Pros:

- **Lowers** the **barrier** for writing docs.
- Focuses on **high quality prose** docs.
- Creates **beatiful** docs.
- **Customizable** themes based on Bootstrap.
- Uses **Markdown**.

Cons:

- Cannot automatically pull **API docs**.

Conclusion: #writethedocs



Eric Holscher ✓

@ericholscher

Following



And Sphinx for documentation. Because nobody knows how to use the code without docs :) #writethedocs

Akash @sky0_1

Python libraries that were used to generate Black hole image.

astropy
ephem...

4:53 AM - 15 Apr 2019

18 Retweets 51 Likes



↻ 18

♥ 51

