Public-facing documentation

Introduction to Python Software Development on GitHub 2023

Lyda Hill Department of Bioinformatics

UT Southwestern Medical Center

Why do we care about documentation?

- It helps us remember how and why we built our software.
- It helps other people use our software.
- It helps other people contribute to our software.
- It helps us write better software.

https://www.writethedocs.org/guide/writing/beginners-guide-to-docs/

Types of documentation

- **Requirements** – What does the software do? High level.
- **Architecture** – How do the pieces of the software fit together?
- **Technical** – How does each function work? Application programming interface (API) docs.
- **End user** – Getting started guides. How-tos.

Software licenses are an important part of documentation, and should be chosen based on what you want to do with your software

- https://choosealicense.com/

Which of the following best describes your situation?

I need to work in a community.

Use the **license preferred by the community** you're contributing to or depending on. Your project will fit right in.

If you have a dependency that doesn't have a license, ask its maintainers to **add a license**.

I want it simple and permissive.

The **MIT License** is short and to the point. It lets people do almost anything they want with your project, like making and distributing closed source versions.

Babel, **.NET**, and **Rails** use the MIT License.

I care about sharing improvements.

The **GNU GPLv3** also lets people do almost anything they want with your project, *except* distributing closed source versions.

Ansible, **Bash**, and **GIMP** use the GNU GPLv3.

GPLv3 is the premier copyleft license, but if you want to license your code to a company, you are better off with an MIT license

GNU GPLv3

Permissions of this strong copyleft license are conditioned on making available complete source code of licensed works and modifications, which include larger works using a licensed work, under the same license. Copyright and license notices must be preserved. Contributors provide an express grant of patent rights.

Permissions
- Commercial use
- Distribution
- Modification
- Patent use
- Private use

Conditions
- Disclose source
- License and copyright notice
- Same license
- State changes

Limitations
- Liability
- Warranty

View full GNU General Public License v3.0 »

MIT License

A short and simple permissive license with conditions only requiring preservation of copyright and license notices. Licensed works, modifications, and larger works may be distributed under different terms and without source code.

Permissions
- Commercial use
- Distribution
- Modification
- Private use

Conditions
- License and copyright notice

Limitations
- Liability
- Warranty

View full MIT License »

Companies may enforce use of their own preferred licenses

- UT Southwestern's open-source software license is here: https://www.utsouthwestern.edu/about-us/administrative-offices/technology-development/agreements/open-source-release-of-software.html

- You MUST use this license if you develop software on any UTSW machine or for any UTSW purpose.

- UNLESS you are using code from a GPLv3-licensed software, in which case I think you should use a GPLv3 license, but you should confirm this with a UTSW lawyer.

Documentation languages

- Hypertext markup language (HTML) (.html)
- Markdown (.md)
- reStructuredText (.rst)
- Many more: https://en.wikipedia.org/wiki/List_of_document_markup_languages

Documentation languages are simple languages, close to standard word processing

Element	Markdown Syntax

Heading	`# H1`
	`## H2`
	`### H3`

Bold	`**bold text**`
Italic	`*italicized text*`
Blockquote	`> blockquote`

Ordered List	`1. First item`
	`2. Second item`
	`3. Third item`

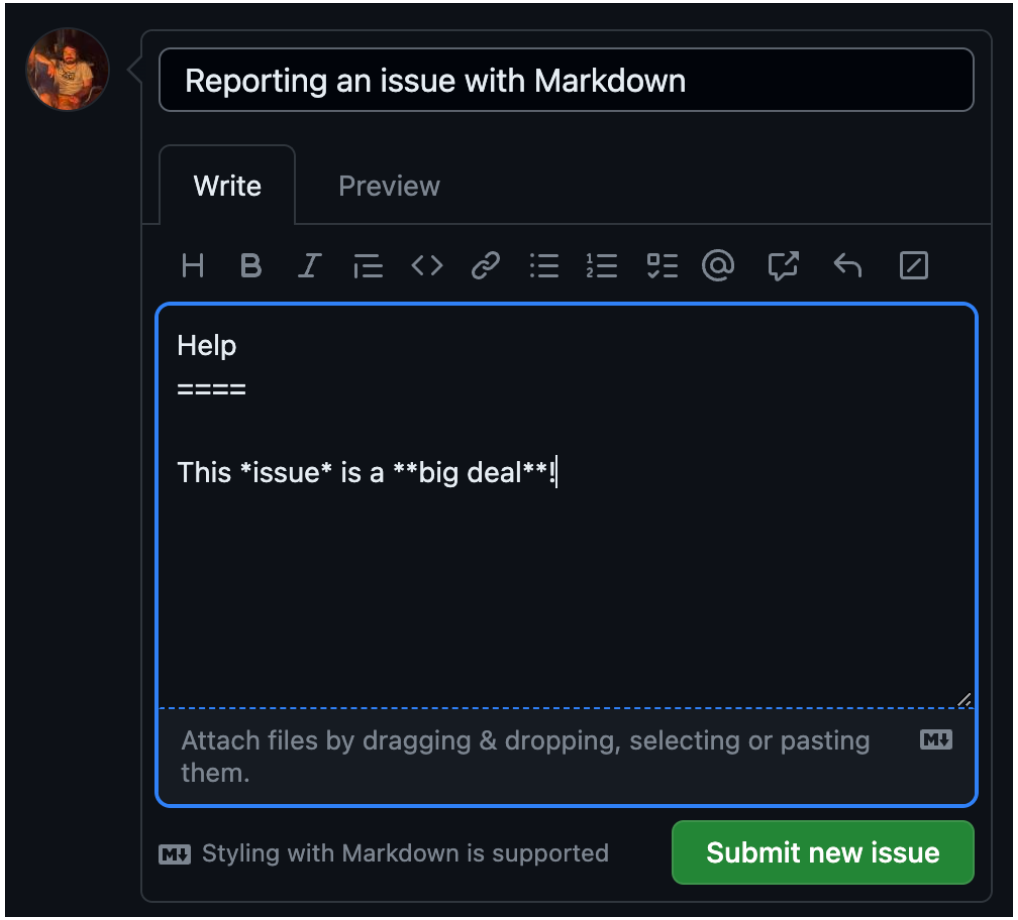
Unordered List	`- First item`
	`- Second item`
	`- Third item`

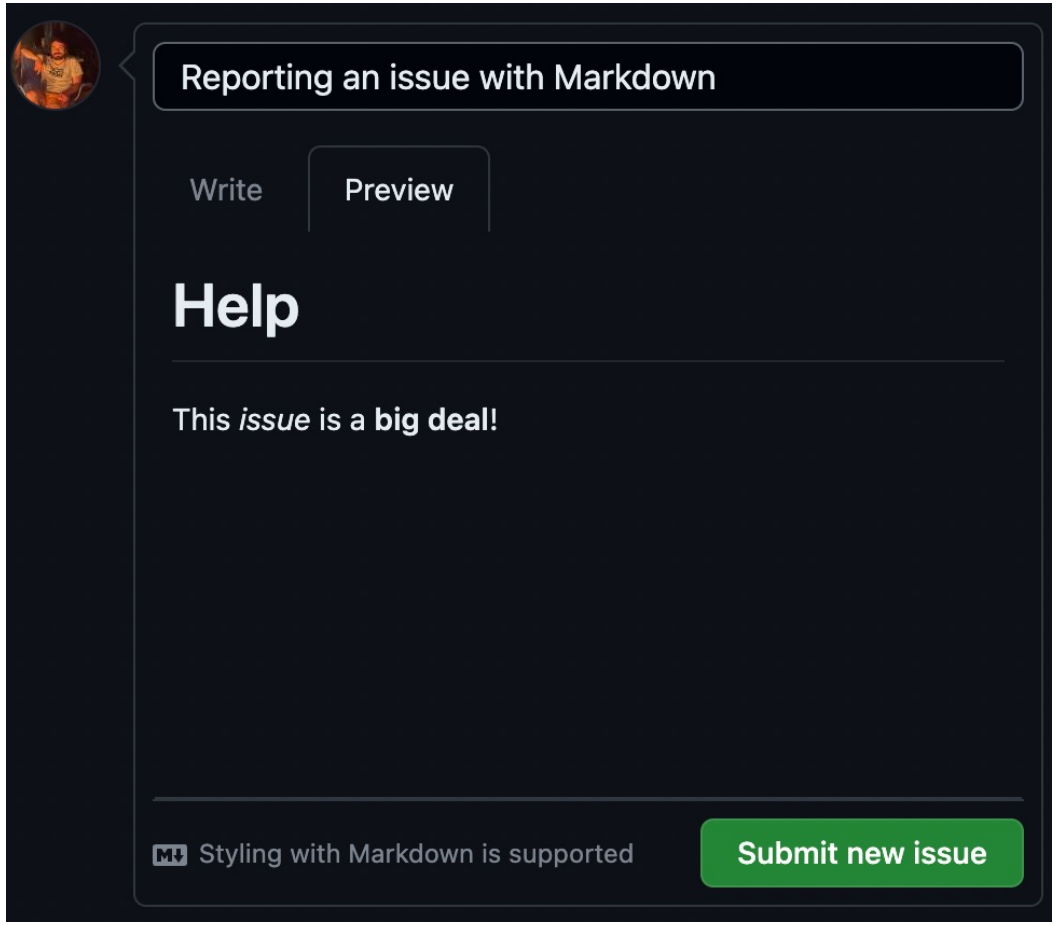
Code	`` `code` ``
Horizontal Rule	`---`
Link	`[title](https://www.example.com)`
Image	`![alt text](image.jpg)`

https://www.markdownguide.org/cheat-sheet/

https://sphinx-tutorial.readthedocs.io/cheatsheet/

GitHub uses Markdown





Exercise: Write a README.md

- README.md files serve as an entry point to your entire code base. They should include
  - A brief description of the project
  - Installation instructions
  - Links to further documentation, including how to contribute
  - How to cite the repository, if applicable
  - Badges, if applicable
  - See https://github.com/mwaskom/seaborn/blob/master/README.md for a good example

- GitHub will display README.md by default on the home page of your repo

Python docstrings can be written in a variety of documentation languages, including reStructuredText

reStructuredText

```python
def repeat_string(string, n_repeats):
    """Returns a string repeated n_repeats times.

    :param string: The string to repeat.
    :type string: str
    :param n_repeats: The number of copies of string.
    :type n_repeats: int
    :returns: A string repeated n_repeats times.
    :rtype: str
    """
    return string*n_repeats
```

Google style

```python
def repeat_string(string, n_repeats):
    """Returns a string repeated n_repeats times.

    Args:
        string (str): The string to repeat.
        n_repeats (int): The number of copies of string.

    Returns:
        str : A string repeated n_repeats times.
    """
    return string*n_repeats
```

Python docstrings can be written in a variety of documentation languages, including reStructuredText

NumPy style

```python
def repeat_string(string, n_repeats):
    """Returns a string repeated n_repeats times.

    Parameters
    ----------
    string : str
        The string to repeat.
    n_repeats : int
        The number of copies of string.

    Returns
    -------
    str
        A string repeated n_repeats times.
    """
    return string*n_repeats
```

Documentation frameworks

- Automate documentation of code
- Import the code to generate documentation based on runtime inspection
- Parse and analyze the code statically (without running it)
- Compile documentation languages to PDFs, HTML, etc.

https://wiki.python.org/moin/DocumentationTools

SPHINX is the primary documentation framework for Python

- https://www.sphinx-doc.org/

- Compiles Python docstrings and reStructuredText files to PDFs, HTML

- Can be modified to use Markdown: https://www.sphinx-doc.org/en/master/usage/markdown.html

An example of a docstring compiled to HTML by Sphinx

```python
def evaluateExpression(expression):
    """
    Evaluate the given mathematical expression and return the result.

    This function takes a mathematical expression as input, evaluates it using the `eval()` function,
    and returns the result as a string. If an error occurs during evaluation, it returns an error message.

    Args:
        expression (str): The mathematical expression to evaluate.

    Returns:
        str: The result of the evaluation as a string, or an error message if evaluation fails.
    """
```

An example of a docstring compiled to HTML by Sphinx

pycalc

Navigation

pycalc
- pycalc.constants
- pycalc.controller
- pycalc.model
  - pycalc.model.evaluateExpression
    - **evaluateExpression()**
- pycalc.pycalc
- pycalc.view

Quick search

Go

pycalc.model.evaluateExpression ¶

`pycalc.model.`**`evaluateExpression`**`(`*`expression`*`)`

Evaluate the given mathematical expression and return the result.

This function takes a mathematical expression as input, evaluates it using the *eval()* function, and returns the result as a string. If an error occurs during evaluation, it returns an error message.

Args:

expression (str): The mathematical expression to evaluate.

Returns:

str: The result of the evaluation as a string, or an error message if evaluation fails.

Let’s build some documentation for our program

- Follow along at https://www.sphinx-doc.org/en/master/usage/quickstart.html

First, make sure we have the right dependencies

- Does anyone know how to install the right sphinx dependencies for documentation?

- `pip install -e .[docs]` from within your repository folder (`pip install -e '.[docs]'` on a Mac)

- This installs the optional dependencies listed under `docs` in `pyproject.toml`

Let’s generate a source folder

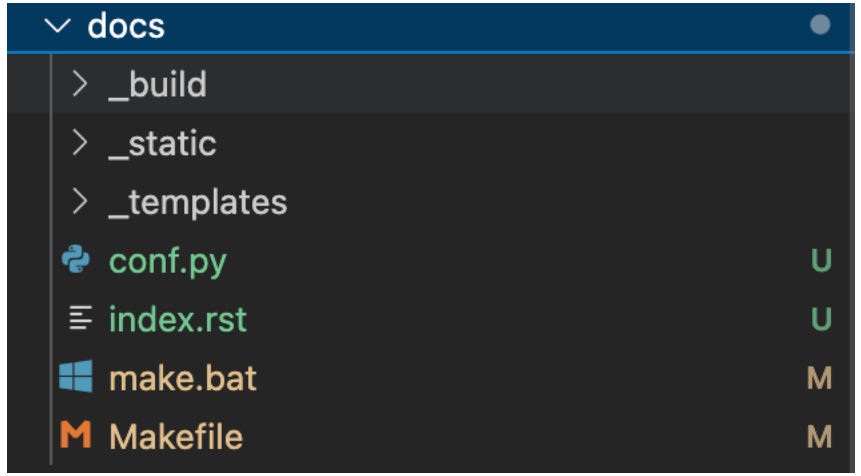
- Create a `docs/` folder in your repo.

- Run `sphinx-quickstart` in this folder.

```
(calc) zachmarin@SW562094 docs % sphinx-quickstart
Welcome to the Sphinx 5.3.0 quickstart utility.

Please enter values for the following settings (just press Enter to
accept a default value, if one is given in brackets).

Selected root path: .

You have two options for placing the build directory for Sphinx output.
Either, you use a directory "_build" within the root path, or you separate
"source" and "build" directories within the root path.
> Separate source and build directories (y/n) [n]:
```

```
(calc) zachmarin@SW562094 docs % sphinx-quickstart
Welcome to the Sphinx 5.3.0 quickstart utility.

Please enter values for the following settings (just press Enter to
accept a default value, if one is given in brackets).

Selected root path: .

You have two options for placing the build directory for Sphinx output.
Either, you use a directory "_build" within the root path, or you separate
"source" and "build" directories within the root path.
> Separate source and build directories (y/n) [n]:

The project name will occur in several places in the built documentation.
> Project name: pycalc
> Author name(s): Tim Jones
> Project release []:

If the documents are to be written in a language other than English,
you can select a language here by its language code. Sphinx will then
translate text that it generates into that language.

For a list of supported codes, see
https://www.sphinx-doc.org/en/master/usage/configuration.html#confval-language.
> Project language [en]:

Creating file /Users/zachmarin/CI2023-MVC-calculator-answerkey/docs/conf.py.
Creating file /Users/zachmarin/CI2023-MVC-calculator-answerkey/docs/index.rst.
Creating file /Users/zachmarin/CI2023-MVC-calculator-answerkey/docs/Makefile.
Creating file /Users/zachmarin/CI2023-MVC-calculator-answerkey/docs/make.bat.

Finished: An initial directory structure has been created.

You should now populate your master file /Users/zachmarin/CI2023-MVC-calculator-answerkey/docs/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.
```

Directory structure of documentation folder



Now try `make html` in the docs folder

- There's a web page! But nothing on it.

pycalc

Navigation

Quick search

Go

Welcome to pycalc's documentation! ¶

Indices and tables

- Index
- Module Index
- Search Page

©2023, Tim Jones. | Powered by Sphinx 5.3.0 & Alabaster 0.7.13 | Page source

The toctree defines what we see

- toctree stands for Table of Contents Tree
- It is empty by default
- You can add documents by listing them in three

docs > index.rst > Indices and tables

```
.. pycalc documentation master file, created by
   sphinx-quickstart on Sat Oct 21 18:07:19 2023.
   You can adapt this file completely to your liking, but it should at least
   contain the root `toctree` directive.

Welcome to pycalc's documentation!
==================================

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

   usage/installation.rst
   usage/quickstart.rst
   api.rst



Indices and tables
==================

.
```

Autosummary helps us with the API documentation

- https://www.sphinx-doc.org/en/master/usage/extensions/autosummary.html

docs > api.rst > API Reference

```
API Reference
=============

.. autosummary::
    :toctree: _autosummary
    :recursive:

    pycalc
```

docs > conf.py > ...

```
# -- General configuration ------------------------------------------------------
# https://www.sphinx-doc.org/en/master/usage/configuration.html#general-configuration

extensions = ["sphinx.ext.autosummary"]
```

Autosummary helps us with the API documentation

pycalc

Navigation

Contents:

API Reference

- pycalc

Quick search

Go

pycalc

Modules

`pycalc.constants`	
`pycalc.controller`	
`pycalc.model`	
`pycalc.pycalc`	Simple calculator built with Python and PyQt5 library
`pycalc.view`	

pycalc

Navigation

Contents:

API Reference

- pycalc

Quick search

pycalc.controller

Classes

`PyCalcCtrl`(model, view)	Main calculator controller class.

Templating

- Sphinx uses Jinja templates (https://jinja.palletsprojects.com/)
- These allow us to change the way data is presented on different pages
- For example, the autosummary module template is located at

Users > zachmarin > opt > miniconda3 > envs > calc > lib > python3.8 > site-packages > sphinx > ext > autosummary > templates > autosummary > module.rst

```
{{ fullname | escape | underline}}

.. automodule:: {{ fullname }}

   {% block attributes %}
   {% if attributes %}
   .. rubric:: {{ _('Module Attributes') }}
```

Templates enable us to produce a more comprehensive autosummary

- custom_module.rst: https://github.com/sphinx-doc/sphinx/issues/7912#issue-650871700

docs > api.rst > API Reference

```
1  API Reference
2  =============
3
4  .. autosummary::
5      :toctree: _autosummary
6      :template: custom_module.rst
7      :recursive:
8
9      pycalc
10
```

```
docs
  _autosummary
  _build
  _static
  _templates
    custom_module.rst   U
  api.rst               U
```

Templates enable us to produce a more comprehensive autosummary

Navigation

Contents:

API Reference

- pycalc

Quick search

Go

class pycalc.controller.**PyCalcCtrl**(*model*, *view*)

Main calculator controller class.

This class serves as the controller for a simple calculator application. It connects the calculator's model and view components and handles user interactions.

Attributes:

_evaluate (callable): The function or object responsible for evaluating expressions. _view (PyCalcView): The view component responsible for displaying the calculator's interface.

Methods:

__init__(self, model, view):

Initializes a PyCalcCtrl instance.

_calculateResult(self):

Evaluates the expression in the calculator display and updates the view with the result.

_buildExpression(self, sub_exp):

Builds an expression for calculation based on user input and updates the

Exercises

- Write `installation.rst` and/or `quickstart.rst`. Compile the new docs to HTML. Verify the compilation worked by opening the docs in your web browser.

- What happens if you change `api.rst` to the following? Why?

docs > api.rst > API Reference

```
API Reference
=============

.. autosummary::
    :toctree: _autosummary
    :template: custom_module.rst
    :recursive:

    pycalc.constants
    pycalc.controller
    pycalc.model
    pycalc.pycalc
    pycalc.view
```

Public-facing documentation

- We want to put our compiled HTML files on the internet.
- Ideally, we do this automatically, updating whenever new documentation is written.
- GitHub actions lets you do this easily with GitHub Pages.

GitHub Pages

- https://pages.github.com/

- Hosts websites directly out of a GitHub repository

TheDeanLab / ASLM

Type / to search

ode | Issues 35 | Pull requests | Discussions | Actions | Projects 1 | Security 49 | Insights | Settings

General

Access

Moderation options

Code and automation

Pages

GitHub Pages

GitHub Pages is designed to host your personal, organization, or project pages from a GitHub repository.

Your site is live at https://thedeanlab.github.io/ASLM/

Last deployed by github-pages[bot] 3 weeks ago

Visit site

Build and deployment

Source

Deploy from a branch

Branch

Your GitHub Pages site is currently being built from the `gh-pages` branch. Learn more about configuring the publishing source for your site.

gh-pages | / (root) | Save

Learn how to add a Jekyll theme to your site.

thedeanlab.github.io/ASLM/

ASLM

Search docs

/ Autonomous Software for Light Microscopy

View page source

Autonomous Software for Light Microscopy

ASLM is a Python library to control microscopes in the UT Southwestern ecosystem. It aims to abstract hardware and allow for maximum customization. The library contains a feature list that acts as a recipe maker for automated microscope routines.

Note

This project is under active development. See our GitHub repository for updates.

Exercise: Create a GitHub workflow that builds your docs and deploys it to a GitHub page

- Hint: https://docs.github.com/en/pages/getting-started-with-github-pages/using-custom-workflows-with-github-pages