Documenting Software

Basics, Sphinx, and Jupyter Book

Jason DeBacker

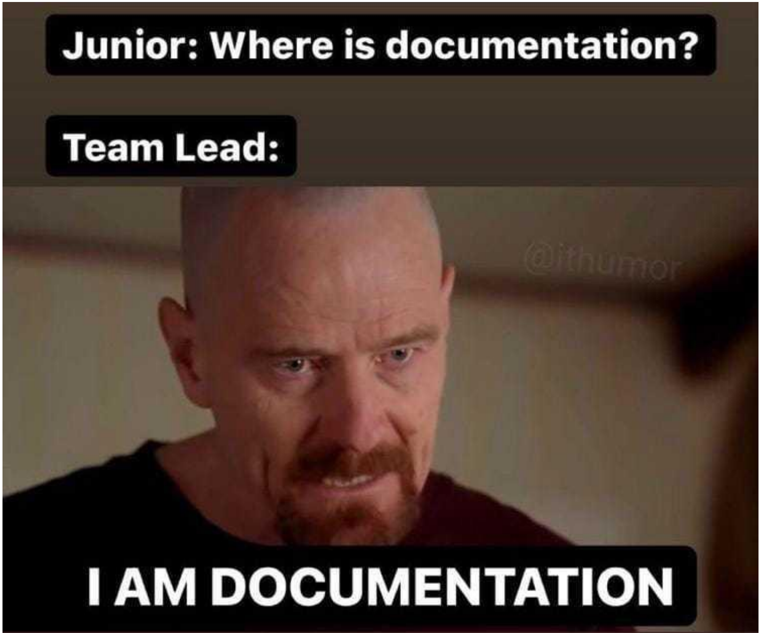
January 17, 2025

Roadmap

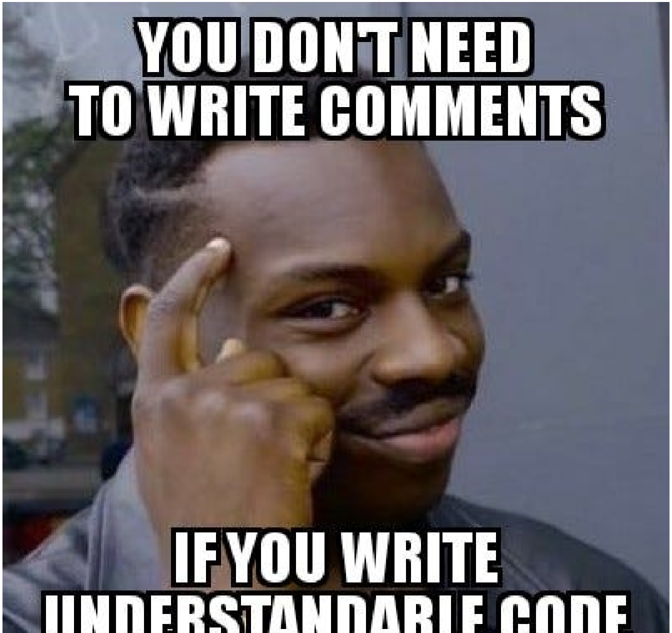
Documenting Code

I don't need to document, Good code documents itself
6 Months later...
WHAT THE HELL IS GOING ON HERE? WHY DIDN'T I DOCUMENT?!
quickmeme.com

Documenting Code



Documenting Code



Why document code?

- Selfish:
  - → Helps you understand your code – be considerate to your future self
  - → Reduces errors
  - → Aids coding copilot
- Selfless:
  - → Helps others understand your code
  - → Encourages collaboration
  - → Helps with reproducibility

Levels of documentation for software

1. In-line comments
2. Docstrings
3. Sphinx documentation for Python code
4. Jupyter Book documentation for a software/research project

Roadmap

The most basic form of documentation: in-line comments

- These are comments that are written directly in the code
- They are useful for explaining what a block of code does
- Typically very short
- In Python, they are denoted by a `#` symbol

In-line comment example

```python
# define a function that adds two numbers
def add_numbers(a, b):
    # make computation
    result = a + b
    return result  # return the result
```

Roadmap

Intro

In-line comments

Docstrings

Sphinx

Jupyter Book

One step up from in-line comments: docstrings

- Docstrings are multi-line strings that are written at the beginning of a function or class
- The are enclosed in triple quotes (either single or double)
- They are used to describe what the function or class does
- They are useful for providing information about the inputs and outputs of a function
- There are some agreed-upon conventions for writing docstrings in Python
  - → Numpydoc
  - → Google

Docstring example (Google style)

```python

def add_numbers(a, b):
    """
    A function to add two numbers

    Args:
        a (float): first number
        b (float): second number

    Returns:
        result (float): sum of a and b
    """
    result = a + b
    return result
```

Roadmap

Sphinx

- Sphinx is a tool that makes it easy to create nicely formatted documentation for code
- It was originally created for the Python documentation
- It can be used to document software written in any language
- It can be used to create documentation in many formats, including HTML, PDF, and ePub

Sphinx syntax

- Sphinx uses reStructuredText (reST) as its markup language
- reST is a lightweight markup language
- It is similar to Markdown, but more flexible

Sphinx docs

- Sphinx can be used on it's own to create documentation from docstrings and other documentation files in a project
- This is one reason it's helpful to write docstrings and do so in a conventional way (e.g., Google or Numpydoc)
- Sphinx can also be used to create documentation for a project that is not written in Python
- But Sphinx is also integrated with Jupyter Book

Roadmap

Intro

In-line comments

Docstrings

Sphinx

Jupyter Book

Jupyter Book

- Jupyter Book is a tool for creating publication-quality books and documents from computational material
- It is built on top of Sphinx, allowing you to use MyST, Markdown, and reStructuredText to write documentation
- It can be used to create documentation for a software project, a research project, or a course
- It can be used to create documentation from Jupyter notebooks

Jupyter Book examples

- Jupyter Book documentation
- QuantEcon
- Numpy Tutorials
- And, this course!