

Continuous Documentation for Your Python code

by **Anastasiia Tymoshchuk**

Associate Director of Engineering @ Soundwide



anastasiatymo



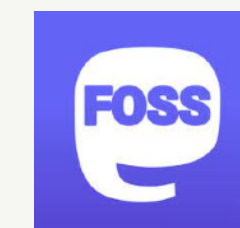
anastasiatymo

Few words about myself

- Associate Director of Engineering at Soundwise in Berlin
- PyBerlin organiser
<https://www.meetup.com/PyBerlin/>
- 11 years in software development
- 7 years in Python
- Happy Pythonista 🐍 😊



anastasiatymo



anastasiatymo

Do you document your code?

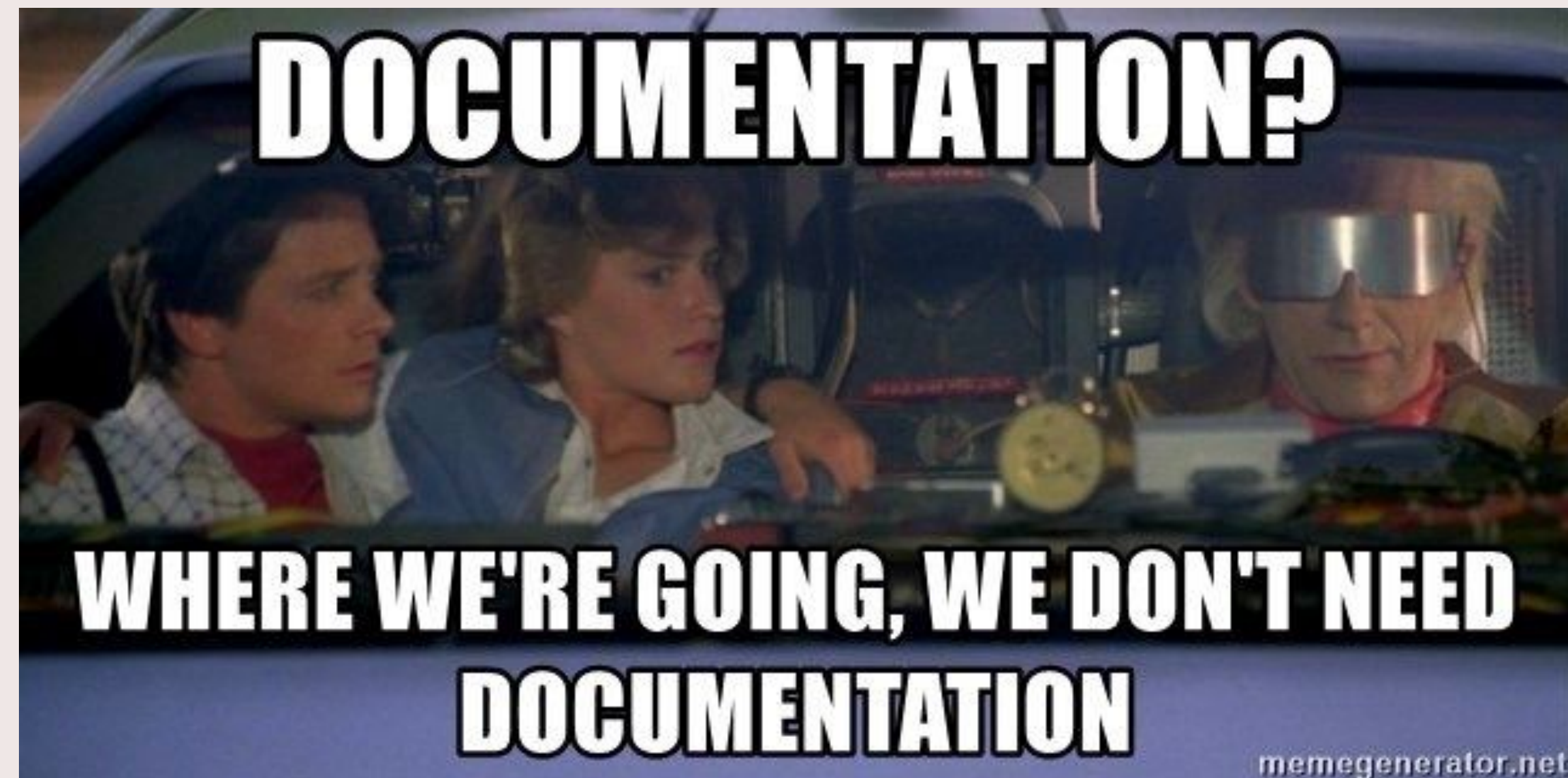


anastasiatymo



anastasiatymo

Me 10 years ago..



Let the future begin...



anastasiatymo



anastasiatymo

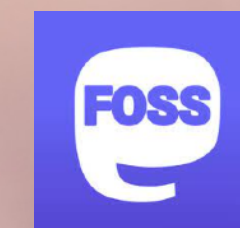


main.py



```
1 from fastapi import FastAPI
2
3 app = FastAPI()
4
5
6 @app.get("/")
7 - async def root():
8     return {"message": "Hello World!"}
9
10 @app.get("/why/{param_1}")
11 - async def why_function(param_1: int, some: Optional[str] = None):
12     return {"message": "What am I doing here?", "param_1": param_1, "some": some}
13
14
```

Sad Code





main.py

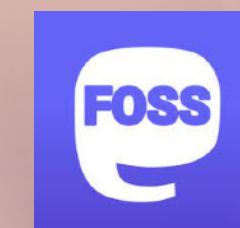


```
1 from fastapi import FastAPI
2
3 app = FastAPI()
4
5
6 @app.get("/")
7 - async def root():
8     return {"message": "Hello World!"}
9
10 @app.get("/why/{param_1}")
11 - async def why_function(param_1: int, some: Optional[str] = None):
12     return {"message": "What am I doing here?", "param_1": param_1, "some": some}
13
14
```

Sad Code



anastasiatymo



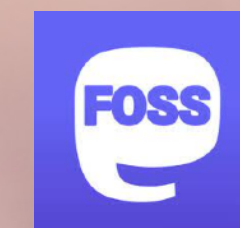
anastasiatymo



main.py



```
1 from fastapi import FastAPI
2
3 app = FastAPI()
4
5
6 @app.get("/")
7 async def root():
8     return {"message": "Hello World!"}
9
10 @app.get("/why/{param_1}")
11 async def why_function(param_1: int, some: Optional[str] = None):
12     return {"message": "What am I doing here?", "param_1": param_1, "some": some}
13
14
```

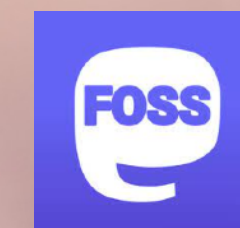




main.py

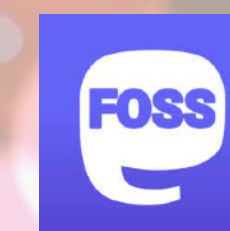


```
1 from fastapi import FastAPI
2
3 app = FastAPI()
4
5
6 @app.get("/")
7 async def root():
8     return {"message": "Hello World!"}
9
10 @app.get("/why/{param_1}")
11 async def why_function(param_1: int, some: Optional[str] = None):
12     return {"message": "What am I doing here?", "param_1": param_1, "some": some}
13
14
```



Sad Code

```
main.py +
1 from fastapi import FastAPI
2
3 app = FastAPI()
4
5
6 @app.get("/")
7- async def root():
8     return {"message": "Hello World!"}
9
10 @app.get("/why/{param_1}")
11- async def why_function(param_1: int, some: Optional[str] = None):
12     return {"message": "What am I doing here?", "param_1": param_1, "some": some}
13
14
```



Sad Code

```
main.py +
1 from fastapi import FastAPI
2
3 app = FastAPI()
4
5
6 @app.get("/")
7- async def root():
8     return {"message": "Hello World!"}
9
10 @app.get("/why/{param_1}")
11- async def why_function(param_1: int, some: Optional[str] = None):
12     return {"message": "What am I doing here?", "param_1": param_1, "some": some}
13
14
```



Sad Code

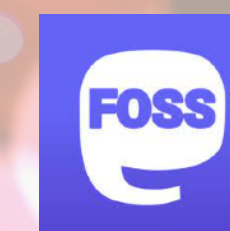
```
main.py +
1 from fastapi import FastAPI
2
3 app = FastAPI()
4
5
6 @app.get("/")
7- async def root():
8     return {"message": "Hello World!"}
9
10 @app.get("/why/{param_1}")
11- async def why_function(param_1: int, some: Optional[str] = None):
12     return {"message": "What am I doing here?", "param_1": param_1, "some": some}
13
14
```



4 pieces of advice



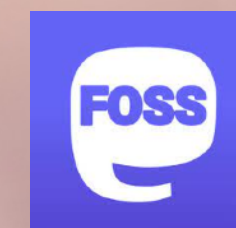
Picture was taken from "Doctor Who"



Listen carefully...



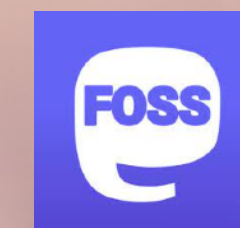
anastasiatymo



anastasiatymo

Sad Code

```
main.py +
1 from fastapi import FastAPI
2
3 app = FastAPI()
4
5
6 @app.get("/")
7- async def root():
8     return {"message": "Hello World!"}
9
10 @app.get("/why/{param_1}")
11- async def why_function(param_1: int, some: Optional[str] = None):
12     return {"message": "What am I doing here?", "param_1": param_1, "some": some}
13
14
```

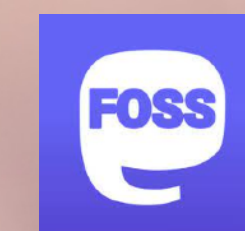


Goal-oriented approach - How-To Guides

"How-to guides are directions that take the reader through the steps required to solve a real-world problem. How-to guides are goal-oriented."

How-to guides can be thought of as recipes, directions that guide the reader through the steps to achieve a specific end."

Source: <https://diataxis.fr/how-to-guides/>



Sad Code

```
main.py +
1 from fastapi import FastAPI
2
3 app = FastAPI()
4
5
6 @app.get("/")
7 async def root():
8     return {"message": "Hello World!"}
9
10 @app.get("/why/{param_1}")
11 async def why_function(param_1: int, some: Optional[str] = None):
12     return {"message": "What am I doing here?", "param_1": param_1, "some": some}
13
14
```

1st advice -
How to guide

☰ README.md ✎

Simple Docs Setup

Code snippets on how to setup documentation for a Python code.

Requirements: Python 3.x

Installation

Please create your Python virtual environment and install the requirements, for example with venv:

```
python3 -m venv venv
source venv/bin/activate
pip install -r requirements/app.txt
```



```
main.py +
1 from fastapi import FastAPI
2
3 app = FastAPI()
4
5
6 @app.get("/")
7 async def root():
8     return {"message": "Hello World!"}
9
10 @app.get("/why/{param_1}")
11 async def why_function(param_1: int, some: Optional[str] = None):
12     return {"message": "What am I doing here?", "param_1": param_1, "some": some}
13
14
```

Sad Code

Unwatch 1 Star 1 Fork 0

I got a friend!

README.md

Simple Docs Setup

Code snippets on how to setup documentation for a Python code.

Requirements: Python 3.x

Installation

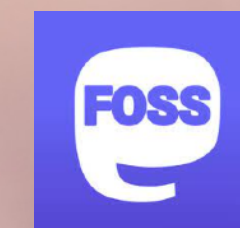
Please create your Python virtual environment and install the requirements, for example with venv:

```
python3 -m venv venv
source venv/bin/activate
pip install -r requirements/app.txt
```

Learning-oriented approach - Tutorials

"Tutorials are lessons that take the reader by the hand through a series of steps to complete a project of some kind."

Source: <https://diataxis.fr/tutorials/>



2nd advice - Tutorial

Sad Code

```
main.py +
1 from fastapi import FastAPI
2
3 app = FastAPI()
4
5
6 @app.get("/")
7- async def root():
8     return {"message": "Hello World!"}
9
10 @app.get("/why/{param_1}")
11- async def why_function(param_1: int, some: Optional[str] = None):
12     return {"message": "What am I doing here?", "param_1": param_1, "some": some}
13
14
```

Tutorial - Basic Sphinx setup

Follow the next steps:

- add Sphinx and recommonmark (Markdown support into your requirements file)
- install Sphinx package

```
pip install sphinx
```

- create a new docs directory in your project

```
mkdir docs
```

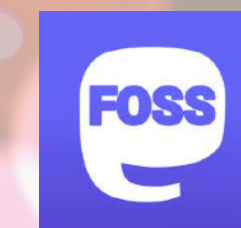
- go to docs directory

```
cd docs
```

- follow basic Sphinx setup by running

```
sphinx-quickstart
```

It's still too early to run build, we need to setup our configuration to make sure that we are able to generate code_reference.



```
main.py +
1 from fastapi import FastAPI
2
3 app = FastAPI()
4
5
6 @app.get("/")
7- async def root():
8     return {"message": "Hello World!"}
9
10 @app.get("/why/{param_1}")
11- async def why_function(param_1: int, some: Optional[str] = None):
12     return {"message": "What am I doing here?", "param_1": param_1, "some": some}
13
14
```

Sad Code

Unwatch 1 Star 2 Fork 0

I got another friend!

Tutorial - Basic Sphinx setup

Follow the next steps:

- add Sphinx and recommonmark (Markdown support into your requirements file)
- install Sphinx package

```
pip install sphinx
```

- create a new docs directory in your project

```
mkdir docs
```

- go to docs directory

```
cd docs
```

- follow basic Sphinx setup by running

```
sphinx-quickstart
```

It's still too early to run build, we need to setup our configuration to make sure that we are able to generate code_reference.

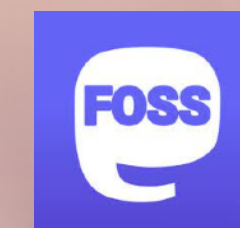
Understanding-oriented approach - Explanation

"Explanation, or discussions, clarify and illuminate a particular topic.

...

Explanation clarifies, deepens and broadens the reader's understanding of a subject."

Source: <https://documentation.divio.com/>



```
main.py +
1 from fastapi import FastAPI
2
3 app = FastAPI()
4
5
6 @app.get("/")
7 async def root():
8     return {"message": "Hello World!"}
9
10 @app.get("/why/{param_1}")
11 async def why_function(param_1: int, some: Optional[str] = None):
12     return {"message": "What am I doing here?", "param_1": param_1, "some": some}
13
14
```

Sad Code

3rd advice - Explanation

Why do we need documentation?

Explanation and motivation

Do you document your code? Do you think it is important?

Benefits of documenting your code:

- if you will get back to your code in 6 month, you will not become a detective to find out how this code works
- easy to onboard new team members
- easy to share knowledge
- if your code is open source - easy to start contributing
- easy to see purpose and motivation of each piece of code

I love this quote from [The Documentation System](#):

There is a secret that needs to be understood in order to write good software documentation: there isn't one thing called documentation, there are four.

They are: tutorials, how-to guides, technical reference and explanation. They represent four different purposes or functions, and require four different approaches to their creation. Understanding the implications of this will help improve most documentation - often immensely.



```
main.py +
1 from fastapi import FastAPI
2
3 app = FastAPI()
4
5
6 @app.get("/")
7 async def root():
8     return {"message": "Hello World!"}
9
10 @app.get("/why/{param_1}")
11 async def why_function(param_1: int, some: Optional[str] = None):
12     return {"message": "What am I doing here?", "param_1": param_1, "some": some}
13
14
```

Sad Code

Unwatch ▾

1

★ Star

3

🔗 Fork

0

And more friends!

Why do we need documentation?

Explanation and motivation

Do you document your code? Do you think it is important?

Benefits of documenting your code:

- if you will get back to your code in 6 month, you will not become a detective to find out how this code works
- easy to onboard new team members
- easy to share knowledge
- if your code is open source - easy to start contributing
- easy to see purpose and motivation of each piece of code

I love this quote from [The Documentation System](#):

There is a secret that needs to be understood in order to write good software documentation: there isn't one thing called documentation, there are four.

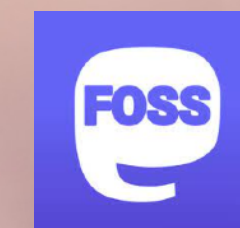
They are: tutorials, how-to guides, technical reference and explanation. They represent four different purposes or functions, and require four different approaches to their creation. Understanding the implications of this will help improve most documentation - often immensely.



Information-oriented approach - Reference

"Reference guides are technical descriptions of the machinery and how to operate it"

Source: <https://diataxis.fr/reference/>



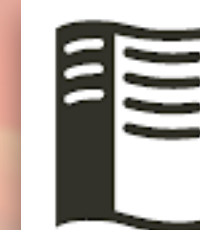
Sad Code

```
main.py +
1 from fastapi import FastAPI
2
3 app = FastAPI()
4
5
6 @app.get("/")
7- async def root():
8     return {"message": "Hello World!"}
9
10 @app.get("/why/{param_1}")
11- async def why_function(param_1: int, some: Optional[str] = None):
12     return {"message": "What am I doing here?", "param_1": param_1, "some": some}
13
14
```

Code Reference

- [app package](#)
 - [Submodules](#)
 - [app.main module](#)

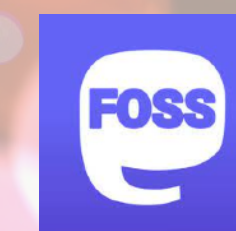
4th advice -
Reference guide



Read *the* Docs



anastasiatymo



anastasiatymo

"Someone, who can teach you a lesson, but not a teacher.

Someone, who can guide you to a goal, but not a tour guide.

Someone, who can tell you everything about technical specs of your functions, but not an encyclopaedia.

Someone, who can explain you a particular topic, to help you to understand, but not Google."

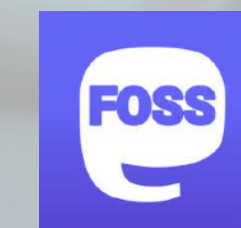
**Was it all about you?
Are you Documentation?**

"In other words, what we call *documentation* is fundamentally not one thing, but four. Understanding the implications of this, and how those four different things work, can help improve most documentation."

Source: <https://diataxis.fr/>

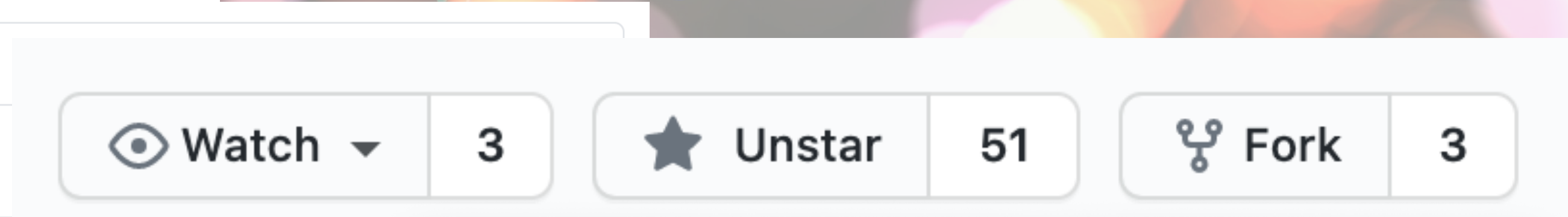
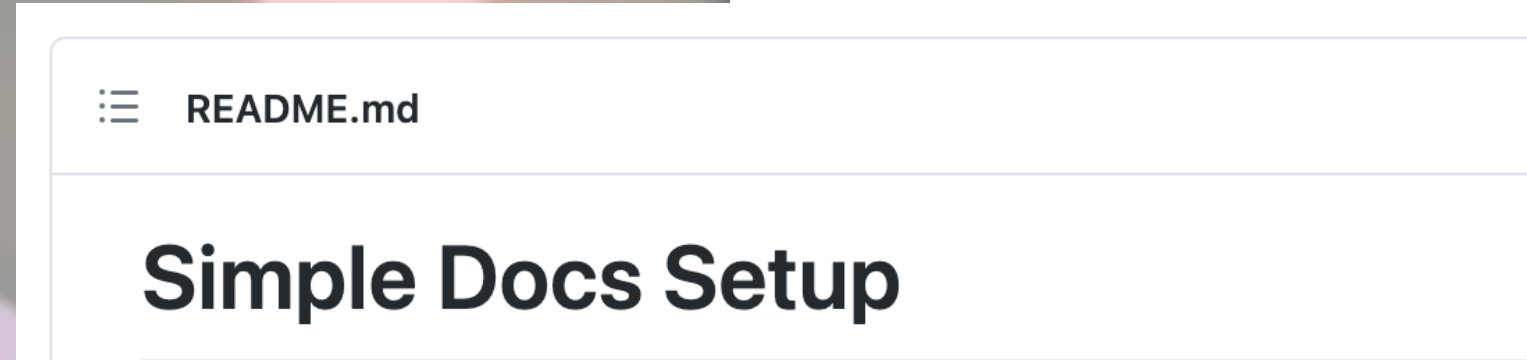


anastasiatymo



anastasiatymo

Code Reference



I got many friends!!!

Tutorial - Basic Sphinx setup

Follow the next steps:

Explanation and motivation

Do you document your code? Do you think it is important?

Benefits of documenting your code:

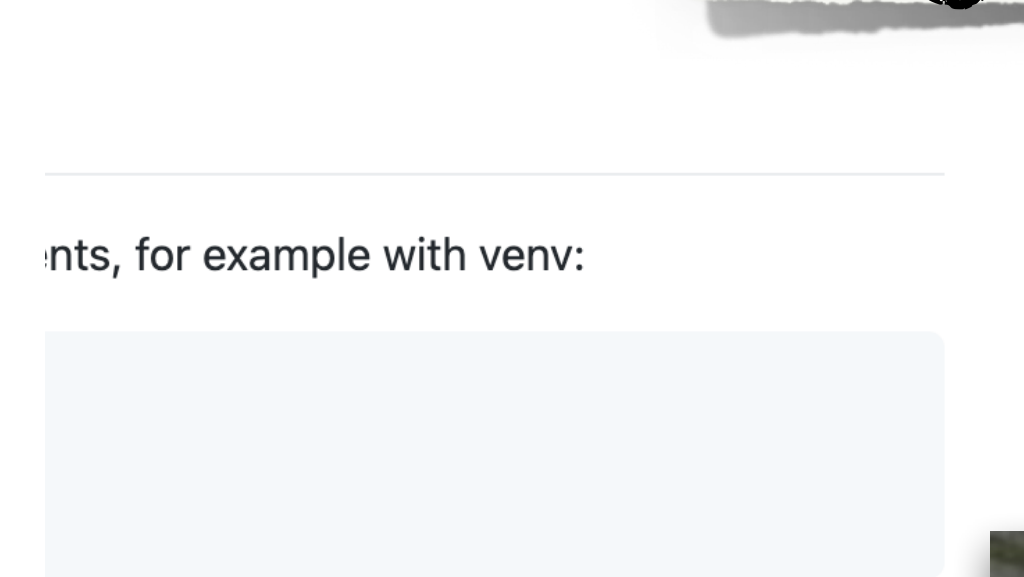
- if you will get back to your code in 6 month, you will not become a detective to find out how this code works

```

main.py
1 from fastapi import FastAPI
2
3 app = FastAPI()
4
5
6 @app.get("/")
7 async def root():
8     return {"message": "Hello World!"}
9
10 @app.get("/why/{param_1}")
11 async def why_function(param_1: int, some: Optional[str] = None):
12     return {"message": "What am I doing here?", "param_1": param_1, "some": some}
13

```

Happy Code!

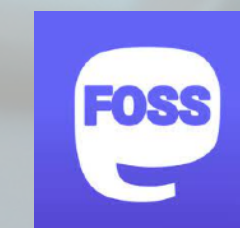


Picture was taken from "Back to the future"

Would you document your code?



anastasiatymo



anastasiatymo

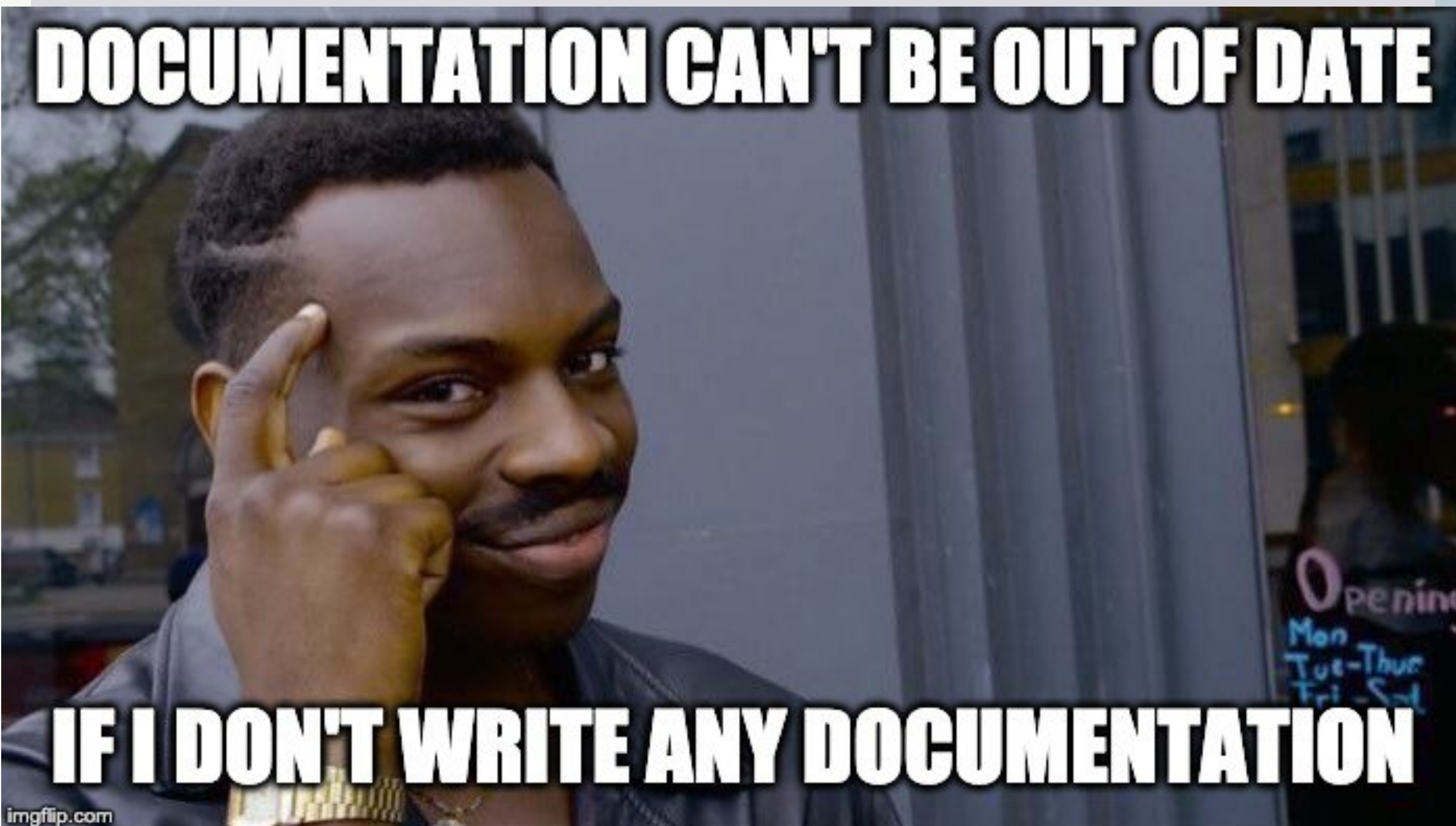
- **people forget things**
- **people leave the code alone**
- **new people come to contribute**

Optimistic developer: "This is a common sense feature that doesn't need documentation"

End user:



Documentation is very important.



- choose main source/tool for documentation
- make sure that it's up-to-date

- <https://diataxis.fr/>
- <https://understandlegacycode.com/blog/where-to-put-documentation/>
- <https://realpython.com/python-project-documentation-with-mkdocs/>

How to start?

- **start as simple as possible**
- **go to version controlled docs**


Documentation for your code?

- **add docstrings to your code**
- **use Sphinx, MkDocs to prepare documentation**
- **try Read The Docs, GitHub Pages**
- **add more documentation**



Demo? 🧑💻

https://github.com/atymoshchuk/simple_docs_setup

Read the Docs  atymoshchuk ▾

Projects > **simple_docs_setup** [View Docs](#)

[Overview](#) [Downloads](#) [Search](#) [Builds](#) [Versions](#) [Admin](#)

Versions

Public [Edit](#)

Build a version

▾

[Build version](#)

Repository

https://github.com/atymoshchuk/simple_docs_setup.git


Project Slug

simple-docs-setup

Last Built

1 month, 2 weeks ago passed

Maintainers



Badge

docs passing [i](#)

SimpleDocTutor

Navigation

[Intro into Simple Docs Setup](#)

[Why do we need documentation?](#)

[Code Reference](#)

Quick search


Welcome to SimpleDocTutor's documentation! 🍷

- [Intro into Simple Docs Setup](#)
 - [Main idea](#)
 - [Tutorial - Basic Sphinx setup](#)
- [Why do we need documentation?](#)
 - [Explanation and motivation](#)
 - [Used sources and books to read](#)
- [Code Reference](#)
 - [app package](#)

Indices and tables

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

©2020, Anastasiia Tymoshchuk. | Powered by [Sphinx 3.0.3](#) & [Alabaster 0.7.12](#) | [Page source](#)

 v: latest ▾



**You can not force people
to **read** documentation**

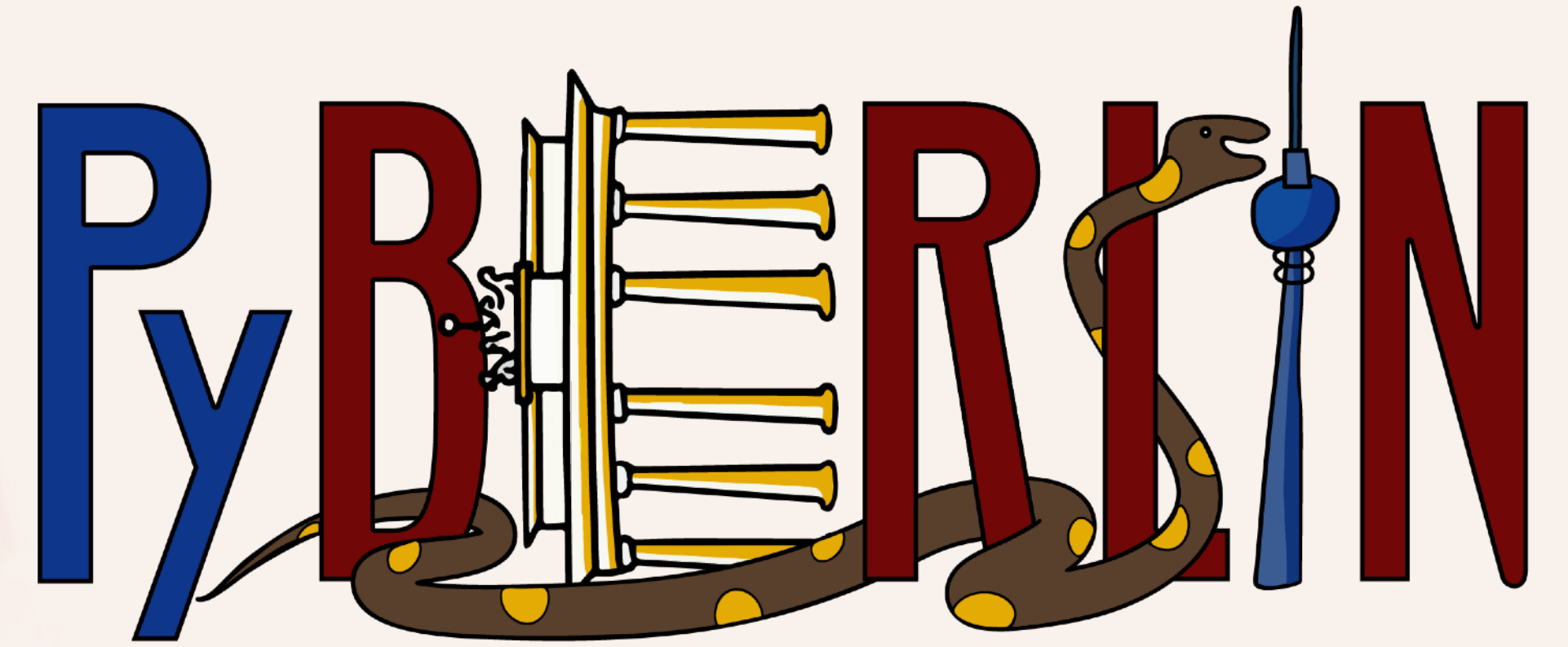
Trying random stuff for
hours instead of reading
the documentation



anastasiatymo



anastasiatymo



<https://www.meetup.com/PyBerlin/>

**I would love to hear
back from you!**

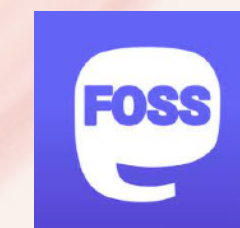
<https://atymo.me/>

https://github.com/atymoshchuk/simple_docs_setup

Thank you!



anastasiatymo



anastasiatymo