Python toolbox

LINUS ÖSTBERG

linus.ostberg@scilifelab.uu.se

SciLifeLab
Data Centre

Sphinx

- Deployment

Code style checkers (linters)

- Flake8
- Pylint
- Pydocstyle
- wemake-python-styleguide
- black

Continous Integration

- Travis
- Github Actions

SPHINX

- Automated generation of e.g. html from rst/md files
- Used to generate e.g. the documentation of python.org
- The autodoc extension allows automated documentation of code using docstring
- Can use markdown for documentation pages, but (afaik) rst is needed for autodoc (code documentation) pages

Quick setup

- Required modules:
    - sphinx
    - recommonmark (markdown support)
    - The modules used by the code to be documented

QUICK SETUP

1. Create a config, fill in e.g. project name

   ```
   sphinx-quickstart --ext-autodoc
   ```

2. Generate a list of code files to generate documentation for

   ```
   sphinx-apidoc -o source <path-to-code>
   ```

3. Make html documentation

   ```
   make html
   ```

4. Fix the conf so you can actually do anything

Configuration

- Fix the paths by uncommenting and updating:

```python
import os
import sys
sys.path.insert(0, os.path.abspath('../../backend'))
```

- The above is in relation to `conf.py`
- Try to build again and open the file in a browser

CONFIGURATION

- Add modules to index.rst to include the autodoc files

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

   modules
```

- Should now list the documentation, but without e.g. parameter parsing

SPHINX DOCSTRING

```python
def getServices(is_owner(entity=None, user=None)):
    """
    Check if the current user owns the given entity

    Defaults to current user if none is provided.

    :param entity: The dentity to check.
    :type dataset: str
    :param user: User to check. Defaults to None. More text
    about the user parameter.
    :type dataset: str, optional

    :return: Whether the user owns the dataset/project.
    :rtype: bool

    :raises ValueError: Raised for some reason.
    """
```

“Google Style”

https://sphinxcontrib-napoleon.readthedocs.io/en/latest/example_google.html

```
def is_owner(entity=None, user=None):
    """
    Check if the current user owns the given entity

    Defaults to current user if none is provided.

    Args:
        entity (str): The entity to check.
        user (str, optional): User to check. Defaults to None.
            More text about the user parameter.

    Returns:
        bool: Whether the user owns the dataset/project.

    Raises:
        ValueError: Raised for some reason.

    """
```

`utils.`**`is_owner`**(*entity=None*, *user=None*)

Check if the current user owns the given entity

Defaults to current user if none is provided.

Parameters:

- **dataset** (*str*) – the dataset to check.
- **user** (*str, optional*) – User to check. Defaults to None. More text about the user parameter.

Returns: whether the user owns the dataset/project.

Return type: bool

Raises: **ValueError** – raised for some reason.

CONFIGURATION

- Add the extension `sphinx.ext.napoleon` to support "Google style" docstrings

```
extensions = [
    'sphinx.ext.autodoc',
    'sphinx.ext.napoleon',
]
```

Markdown

- Add the extension `recommonmark` the same way as `sphinx.ext.napoleon` was added
- Create a markdown file and link it from one of the current files.

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

   modules
   test.md
```

- Sphinx
    - Deployment

- Code style checkers (linters)
    - Flake8
    - Pylint
    - Pydocstyle
    - wemake-python-styleguide
    - black

- Continous Integration
    - Travis
    - Github Actions

ReadTheDocs

- Log in with GitHub to `https://readthedocs.org/`
- Import the wanted project
- Setup
  - Branch
  - Location of sphinx files
  - `requirements.txt`
- Webhook created automatically
- Updated whenever the branch gets a push
- `https://<repos>.readthedocs.io/`
  - `https://scilifelab-data-tracker.readthedocs.io/`
  - `https://dp_api.readthedocs.io/`
  - Can be referenced by other URL

ReadTheDocs

- Pros:
  - Updated automatically (Webhook)
  - Only source files need to be in git
  - Supports docs for multiple versions
- Cons:
  - No Python 3.8 support (yet)
  - Must be able to run sphinx
    - If your code needs a config file, you must have a special branch with one provided

GITHUB PAGES

- Integrated in github
- To use with sphinx:
  - Generate sphinx documentation locally
  - Make sure there is a `.nojekyll` file
  - Rename `_static` to `static`
  - Change all references in the html files (`sed`)
  - Use `git subtree push` to push to the branch `gh-pages`
- Updated whenever `gh-pages` gets a push
- `https://<org/user>.github.io/<repos>/`
  - `https://scilifelabdatacentre.github.io/SciLifeLab-Data-Tracker/`
  - `https://scilifelabdatacentre.github.io/dp_api/`
  - Can be referenced by other URL

Modifying the HTML Build

```
sed -i "s/_sources/sources/g" build/html/*html
sed -i "s/_modules/modules/g" build/html/*html
sed -i "s/_static/static/g" build/html/*html

mv build/html/_static build/html/static
mv build/html/_modules build/html/modules
mv build/html/_sources build/html/source

git subtree push --force --prefix docs/build/html/
```

GITHUB PAGES

- Pros:
  - Updated automatically when pushing to `gh-pages`
  - Does not run sphinx
  - Default URL shows that software is made by us
- Cons:
  - Html build must be stored in git
  - The html build must be modified to work

Sphinx

- Deployment

Code style checkers (linters)

- Flake8
- Pylint
- Pydocstyle
- wemake-python-styleguide
- black

Continous Integration

- Travis
- Github Actions

LINTERS

- Code style checkers
- May inform about better ways of coding

Sphinx

- Deployment

Code style checkers (linters)

- Flake8
- Pylint
- Pydocstyle
- wemake-python-styleguide
- black

Continous Integration

- Travis
- Github Actions

Flake8

- Checks if the code fulfills PEP8
- Checks for some errors
- Does not support Python 3.8 (yet) :=
- Unwanted checks can be turned off

Config file (.flake8)

```
[flake8]
max-line-length = 100
```

Flake8

Example

```
import uuid

def functionName(a = 5) :
    c=a**2 # comment
    print(f'Data: {c}')
```

Flake8

Example

```
lostb@traveller ˜ $ flake8 bad_code.py
bad_code.py:1:1: F401 ’uuid’ imported but unused
bad_code.py:3:1: E302 expected 2 blank lines, found 1
bad_code.py:3:19: E251 unexpected spaces around keyword /
    parameter equals
bad_code.py:3:21: E251 unexpected spaces around keyword /
    parameter equals
bad_code.py:3:24: E203 whitespace before ’:’
bad_code.py:4:6: E225 missing whitespace around operator
bad_code.py:4:11: E261 at least two spaces before inline comment
```

Sphinx

- Deployment

Code style checkers (linters)

- Flake8
- Pylint
- Pydocstyle
- wemake-python-styleguide
- black

Continous Integration

- Travis
- Github Actions

PYLINT

- Another style checker
- Error checker

Pylint

Example

```python
import uuid

def functionName(a = 5) :
    c=a**2 # comment
    print(f'Data: {c}')
```

Pylint

Example

```
lostb@traveller ~ $ pylint bad_code.py
************* Module bad_code
bad_code.py:3:19: C0326: No space allowed around keyword argument assignment
def functionName(a = 5) :
                   ^ (bad-whitespace)
bad_code.py:3:24: C0326: No space allowed before :
def functionName(a = 5) :
                        ^ (bad-whitespace)
bad_code.py:4:5: C0326: Exactly one space required around assignment
    c=a**2 # comment
     ^ (bad-whitespace)
bad_code.py:1:0: C0114: Missing module docstring (missing-module-docstring)
bad_code.py:3:0: C0103: Function name "functionName" doesn't conform to snake_case
    naming style (invalid-name)
bad_code.py:3:0: C0103: Argument name "a" doesn't conform to snake_case
    naming style (invalid-name)
bad_code.py:3:0: C0116: Missing function or method docstring (missing-function-docstring)
bad_code.py:4:4: C0103: Variable name "c" doesn't conform to snake_case
    naming style (invalid-name)
bad_code.py:1:0: W0611: Unused import uuid (unused-import)

-------------------------------------
Your code has been rated at -12.50/10
```

Sphinx

- Deployment

Code style checkers (linters)

- Flake8
- Pylint
- Pydocstyle
- wemake-python-styleguide
- black

Continous Integration

- Travis
- Github Actions

PYDOCSTYLE

- Evaluate docstrings

Pydocstyle

Example

```python
def missing():
    pass


def bad():
    """
    This function does nothing useful
    """


def bad_fixed():
    """Do nothing useful."""


def good(asd):
    """
    Return the given value.

    Args:
        asd (str): something

    Returns:
        str: the value of asd
    """
    return asd
```

Pydocstyle

Example

```
lostb@traveller ˜ $ pydocstyle bad_code2.py
bad_code2.py:1 at module level:
        D100: Missing docstring in public module
bad_code2.py:2 in public function ‘missing‘:
        D103: Missing docstring in public function
bad_code2.py:7 in public function ‘bad‘:
        D200: One-line docstring should fit on one line with quotes (found 3)
bad_code2.py:7 in public function ‘bad‘:
        D400: First line should end with a period (not ’l’)
bad_code2.py:7 in public function ‘bad‘:
        D401: First line should be in imperative mood; try rephrasing (found ’This’)

lostb@traveller ˜ $ pylint bad_code2.py
************* Module bad_code2
bad_code2.py:1:0: C0114: Missing module docstring (missing-module-docstring)
bad_code2.py:2:0: C0116: Missing function or method docstring (missing-function-docstring)

------------------------------------------------------------------
Your code has been rated at 6.67/10 (previous run: 8.33/10, -1.67)
```

Sphinx

- Deployment

Code style checkers (linters)

- Flake8
- Pylint
- Pydocstyle
- wemake-python-styleguide
- black

Continous Integration

- Travis
- Github Actions

WEMAKE-PYTHON-STYLEGUIDE

- “the strictest and most opinionated python linter ever”
- A lot of flake8 plugins

WEMAKE-PYTHON-STYLEGUIDE

```
lostb@traveller ~/project_portal/backend (linus-dev) $ flake8 app.py
lostb@traveller ~/project_portal/backend (linus-dev) $ pylint app.py

------------------------------------
Your code has been rated at 10.00/10

lostb@traveller ~/project_portal/backend (linus-dev) $ flake8 app.py
app.py:1:1: C101 Coding magic comment not found
app.py:29:32: WPS510 Found ‘in‘ used with a non-set container
app.py:40:1: DAR101 Missing parameter(s) in Docstring: - response
app.py:40:1: DAR201 Missing "Returns" in Docstring: - return
app.py:40:8: WPS421 Found wrong function call: hasattr
app.py:48:19: WPS432 Found magic number: 404
app.py:51:1: DAR101 Missing parameter(s) in Docstring: - _
app.py:51:1: DAR201 Missing "Returns" in Docstring: - return
app.py:51:34: WPS432 Found magic number: 404
app.py:57:1: DAR201 Missing "Returns" in Docstring: - return
app.py:62:34: WPS432 Found magic number: 5000
```

Sphinx

- Deployment

Code style checkers (linters)

- Flake8
- Pylint
- Pydocstyle
- wemake-python-styleguide
- black

Continous Integration

- Travis
- Github Actions

BLACK

- Code formatter
- Formats code according to what it considers the correct way
- Ignores earlier formatting unless `# fmt: off` is used
- “do whatever makes `pycodestyle` happy”
- Formatting only

BLACK

```
lostb@traveller ~ $ pylint bad_code
************* Module bad_code
bad_code.py:3:19: C0326: No space allowed around keyword argument assignment
def functionName(a = 5) :
                   ^ (bad-whitespace)
bad_code.py:3:24: C0326: No space allowed before :
def functionName(a = 5) :
                        ^ (bad-whitespace)
bad_code.py:4:5: C0326: Exactly one space required around assignment
    c=a**2 # comment
     ^ (bad-whitespace)
bad_code.py:1:0: C0114: Missing module docstring (missing-module-docstring)
bad_code.py:3:0: C0103: Function name "functionName" doesn’t conform to snake_case naming styl
bad_code.py:3:0: C0103: Argument name "a" doesn’t conform to snake_case naming style (invalid-
bad_code.py:3:0: C0116: Missing function or method docstring (missing-function-docstring)
bad_code.py:4:4: C0103: Variable name "c" doesn’t conform to snake_case naming style (invalid-
bad_code.py:1:0: W0611: Unused import uuid (unused-import)

----------------------------------------------------------------------
Your code has been rated at -12.50/10 (previous run: -12.50/10, +0.00)
```

BLACK

```
lostb@traveller ~ $ black bad_code.py
reformatted bad_code.py
All done!
1 file reformatted.

lostb@traveller ~ $ pylint bad_code
************* Module bad_code
tmp.py:1:0: C0114: Missing module docstring (missing-module-docstring)
tmp.py:4:0: C0103: Function name "functionName" doesn't conform to snake_case naming style (in
tmp.py:4:0: C0103: Argument name "a" doesn't conform to snake_case naming style (invalid-name)
tmp.py:4:0: C0116: Missing function or method docstring (missing-function-docstring)
tmp.py:5:4: C0103: Variable name "c" doesn't conform to snake_case naming style (invalid-name)
tmp.py:1:0: W0611: Unused import uuid (unused-import)

--------------------------------------------------------------------
Your code has been rated at -5.00/10 (previous run: 6.67/10, -11.67)
```

Sphinx

- Deployment

Code style checkers (linters)

- Flake8
- Pylint
- Pydocstyle
- wemake-python-styleguide
- black

Continous Integration

- Travis
- Github Actions

Sphinx

- Deployment

Code style checkers (linters)

- Flake8
- Pylint
- Pydocstyle
- wemake-python-styleguide
- black

Continous Integration

- Travis
- Github Actions

Travis

- Environment for CI
- Runs tests in a vm/container
- Can be automated to run for any push to a github repository
- `https://travis-ci.com/`

.TRAVIS.YML

```
language: python
python:
 - "3.6"
 - "3.7"
 - "3.8"
services:
  - mongodb
install:
    - pip install -r backend/requirements.txt
    - pip install -r test/requirements.txt
before_script:
- test/travis_setup.sh
script:
    - test/travis_script.sh
```

TRAVIS_SCRIPT.SH

```
#!/bin/sh
pytest backend
pylint backend/*py
pydocstyle backend/*py
```

Sphinx

- Deployment

Code style checkers (linters)

- Flake8
- Pylint
- Pydocstyle
- wemake-python-styleguide
- black

Continous Integration

- Travis
- Github Actions

Github Actions

- CI integrated in Github
- Many people seems to like it