Python Tooling

Guilhem Saurel

This presentation

Available at

`https://homepages.laas.fr/gsaurel/talks/python-tooling.pdf`

Under License

This presentation (continued)

Source

`https://gitlab.laas.fr/gsaurel/talks : python-tooling.md`

Discussions

https://im.laas.fr/#/room/#python-tooling:laas.fr

Outline

Motivation

`import this`

The Zen of Python, by Tim Peters (extracts)

- Beautiful is better than ugly.
- Readability counts.
- There should be one– and preferably only one –obvious way to do it.

Write readable code

Code is read much more often than it is written

— Guido

Write readable code

Code is read much more often than it is written

— Guido

- read → write
- teamwork
- future self

Style

PEP 8

Style Guide for Python Code

https://peps.python.org/pep-0008/

- indentation: 4 spaces
- maximum line length: 79
- whitespaces
- comments
- naming

pep8 → pycodestyle

https://github.com/PyCQA/pycodestyle

pep8 → pycodestyle

https://github.com/PyCQA/pycodestyle

```
$ pycodestyle --first optparse.py
optparse.py:69:11: E401 multiple imports on one line
optparse.py:77:1: E302 expected 2 blank lines, found 1
optparse.py:88:5: E301 expected 1 blank line, found 0
optparse.py:347:31: E211 whitespace before '('
optparse.py:357:17: E201 whitespace after '{'
optparse.py:472:29: E221 multiple spaces before operator
```

PEP 257 / pydocstyle

https://github.com/PyCQA/pydocstyle

Mostly helps as a reminder to write some doc

```
$ pydocstyle test.py
test.py:18 in private nested class `meta`:
        D101: Docstring missing
test.py:27 in public function `get_user`:
    D300: Use """triple double quotes""" (found '''-
quotes)
test:75 in public function `init_database`:
   D201: No blank lines allowed before function docstring (
```

flake8

https://github.com/PyCQA/flake8

pycodestyle + pyflakes + mccabe

yapf

https://github.com/google/yapf

clang-format / gofmt

```
x = {  'a':37,'b':42,

'c':927}
```

yapf

https://github.com/google/yapf

clang-format / gofmt

```
x = {  'a':37,'b':42,

'c':927}
```

```
x = {'a': 37, 'b': 42, 'c': 927}
```

black

https://github.com/psf/black

Figure 1: black

black main caveat

```
super_long_line.with_small_argument = [0]  # some commen
```

black main caveat

```
super_long_line.with_small_argument = [0]  # some commen

super_long_line.with_small_argument = [
    0
]  # some comment
```

black main caveat

```
super_long_line.with_small_argument = [0]  # some commen

super_long_line.with_small_argument = [
    0
]  # some comment

# some comment
super_long_line.with_small_argument = [0]
```

isort

https://github.com/PyCQA/isort

sorts imports.

Packaging

Poetry

https://github.com/python-poetry/poetry

- `poetry init`
- `pyproject.toml`
- minimal python version
- dependencies / dev-dependencies
- virtualenv
- sdist / wheel builder

Others

ref. “Managing Python Packages”

Tests

unittest

https://docs.python.org/3/library/unittest.html

```python
import unittest

class TestStringMethods(unittest.TestCase):

    def test_upper(self):
        self.assertEqual('foo'.upper(), 'FOO')

    def test_isupper(self):
        self.assertTrue('FOO'.isupper())
        self.assertFalse('Foo'.isupper())


if __name__ == '__main__':
    unittest.main()
```

unittest launch

```
$ python test_string.py
..
----------------------------------------------------------------------
Ran 2 tests in 0.000s

OK
```

unittest launch

```
$ python test_string.py
..
----------------------------------------------------------------------
Ran 2 tests in 0.000s

OK
```

ou

```
$ python -m unittest
..
----------------------------------------------------------------------
Ran 2 tests in 0.000s

OK
```

doctest

https://docs.python.org/3/library/doctest.html

```
def factorial(n):
    """Return the factorial of n, an exact integer >= 0.

    >>> [factorial(n) for n in range(6)]
    [1, 1, 2, 6, 24, 120]
    >>> factorial(30)
    265252859812191058636308480000000
    >>> factorial(-1)
    Traceback (most recent call last):
        ...
    ValueError: n must be >= 0
    """
```

doctest launch

```python
if __name__ == "__main__":
    import doctest
    doctest.testmod()
```

Others

- pytest
- tox

Coverage

Coverage.py

https://github.com/nedbat/coveragepy

```
$ coverage run -m unittest discover
$ coverage report -m
Name                      Stmts    Miss   Cover    Missing
-------------------------------------------------------------
my_program.py               20       4     80%   33-35, 39
my_other_module.py           56       6     89%    17-23
-------------------------------------------------------------
TOTAL                        76      10     87%
```

Coverage.py

https://github.com/nedbat/coveragepy

```
$ coverage run -m unittest discover
$ coverage report -m
Name                      Stmts   Miss  Cover   Missing
-------------------------------------------------------
my_program.py               20      4    80%   33-35, 39
my_other_module.py          56      6    89%   17-23
-------------------------------------------------------
TOTAL                       76     10    87%
```

```
$ coverage html
```

sample

online

- https://coveralls.io/
- https://about.codecov.io/

Gitlab

Settings → CI/CD → General pipelines → Test coverage parsing

Static analysis

Mypy: motivation

```python
def add(a: int, b: int) -> int:
    """Performs addition on integers.

    >>> add(3, 4)
    7
    """
    return a + b


if __name__ == "__main__":
    import sys

    print(add(sys.argv[1], sys.argv[2]))
```

Mypy

```
$ python add.py 3 4
34
```

Mypy

```
$ python add.py 3 4
34

$ mypy add.py
add.py:13: error: Argument 1 to "add" has incompatible
                  type "str"; expected "int"
add.py:13: error: Argument 2 to "add" has incompatible
                  type "str"; expected "int"
Found 2 errors in 1 file (checked 1 source file)
```

pyupgrade

https://github.com/asottile/pyupgrade

```
 class C(Base):
     def f(self):
-        super(C, self).f()
+        super().f()
```

Pylint

pass...

Meta

Editor / IDE integration

Bring your own ;)

pre-commit

https://github.com/pre-commit/pre-commit

.pre-commit-config.yaml

```yaml
repos:
-   repo: https://github.com/pre-commit/pre-commit-hooks
    rev: v4.2.0
    hooks:
    -   id: check-yaml
    -   id: end-of-file-fixer
    -   id: trailing-whitespace
-   repo: https://github.com/psf/black
    rev: 22.3.0
    hooks:
    -   id: black
```

pre-commit usage

```
pre-commit run -a
```

pre-commit usage

```
pre-commit run -a

pre-commit install
```

pre-commit usage

```
pre-commit run -a
```

```
pre-commit install
```

demo

pre-commit CI

https://pre-commit.ci/

Badges

- codecov 94%
- pre-commit.ci passed
- code style black
- coverage 96.00%

Questions ?

Thanks for your time :)

Proposition

- flake8
- black
- isort
- pyupgrade
- pre-commit