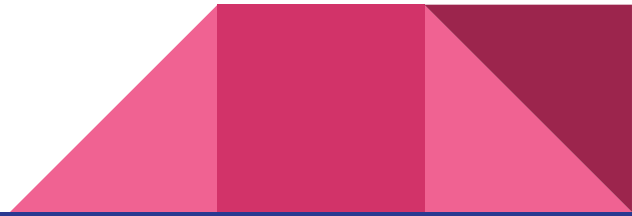
tCRIL presents:

A Standard Python Toolbox

March 2022

Quick Q&A

- **Who**: The Whole Community!
- **What**: python packages
- **Where**: EVERYWHERE!
- **When**: ASAP!
- **Why**: Standardization



Formatting Tools

isort: Sorts your imports

- Sorts imports alphabetically and automatically
- Command line utility
- Plugins for a variety of editors

isort: Sorts your imports

Before:

```
1 from my_lib import Object
2 import os
3 from my_lib import Object2
4 import sys
5 import matplotlib.pyplot as plt
6 import pandas as pd
7 import numpy as np
8 import sys
9 from __future__ import absolute_import
```

After:

```
 1 from __future__ import absolute_import
 2
 3 import os
 4 import sys
 5
 6 import matplotlib.pyplot as plt
 7 import numpy as np
 8 import pandas as pd
 9
10 from my_lib import Object, Object2
```

Pylint: a Python static code analysis tool

- Looks for programming errors
- Helps enforce a coding standard
- Offers simple refactoring suggestions
- Highly configurable

Pylint: a Python static code analysis tool

Code:

```
1 A = 23
2 B = 45
3 C = A + B
4
5 print(C)
6
```

Pylint:

```
pylint_example.py:1:0: C0114: Missing
module docstring
(missing-module-docstring)

-----------------------------------
Your code has been rated at 7.50/10
```

Pycodestyle: checks your code against PEP8

- Adding new checks is easy.
- Jump to error location in your editor.
- Just one Python file, requires only stdlib.
- Comes with a comprehensive test suite.

Pycodestyle: checks your code against PEP8

Code:

```
1 import json, random
2
3 # This line opens a new json file. That
  new json file is clearly named
  'new_file.json'.
4 f = open('new_file.json')
5
6 x = {i: random.randint(0, 100) for i in
  range(100)}
7
8 json.dump(x, f)
```

Pycodestyle:

```
pycodestyle_example.py:1:12: E401
multiple imports on one line
pycodestyle_example.py:3:80: E501 line
too long (87 > 79 characters)
pycodestyle_example.py:8:16: W292 no
newline at end of file
```

Black: uncompromising Python code formatter

- Reformats entire files in place
- Produces the smallest diffs possible
- Has a comprehensive test suite

Black: uncompromising Python code formatter

Before:

```
1 def add(a,       b):
2   answer       = a      +    b
3
4    return answer
5
6
7 def sub(c    ,
8   d):
9
10    answer = c   -    d
11
12   return answer
```

After:

```
1 def add(a, b):
2    answer = a + b
3
4    return answer
5
6
7 def sub(c, d):
8
9    answer = c - d
10
11   return answer
12
```

Code Quality Tools

Mypy: Adds type annotations

- Catches many programming errors without having to run it
- Features type inference, gradual typing, generics and union types
- Makes code easier to understand, debug, and maintain

Mypy: Adds type annotations

Dynamic Typing:

```
1 def fib(n):
2     a, b = 0, 1
3     while a < n:
4         yield a
5         a, b = b, a + b
6
```

Mypy Static Typing:

```
1 from typing import Iterator
2
3
4 def fib(n: int) -> Iterator[int]:
5     a, b = 0, 1
6     while a < n:
7         yield a
8         a, b = b, a + b
9
```

Pytest: a test framework

- Easy to write small tests
- Scales to support complex functional testing
- Detailed info on failing assert statements

Pytest: a test framework

Code:

```
1 def func(x):
2     return x + 1
3
4
5 def test_answer():
6     assert func(3) == 5
7
```

Pytest:

```
pytest_example.py F                                  [100%]

=============== FAILURES ===============
_____________ test_answer _____________

    def test_answer():
>       assert func(3) == 5
E       assert 4 == 5
E        +  where 4 = func(3)

pytest_example.py:6: AssertionError
======= short test summary info ========
FAILED pytest_example.py::test_answer
========== 1 failed in 0.04s ===========
```

Coverage: measures code coverage

- Determines which lines are executable
- And which have been executed
- Command Line Interface with many options

Coverage: measures code coverage

Code (example.py):

```python
def traffic_lights(color):
    if color == "green":
        return "go"
    elif color == "yellow":
        return "slow down"
    elif color == "red":
        return "stop"
    else:
        raise Exception("Invalid color")
```

Test Code (test_example.py):

```python
from example import traffic_lights

def test_traffic_lights_green():
    assert traffic_lights('green') == "go"

def test_traffic_lights_yellow():
    assert (
        traffic_lights('yellow') == "slow")
```

Coverage:

```
Name                                 Stmts   Miss  Cover
Missing
------------------------------------------------------------------
-
coverage_example/example.py              8      3    62%   6-9
coverage_example/test_example.py         5      0   100%
```

Documentation Tools

Sphinx: creates intelligent & beautiful documentation

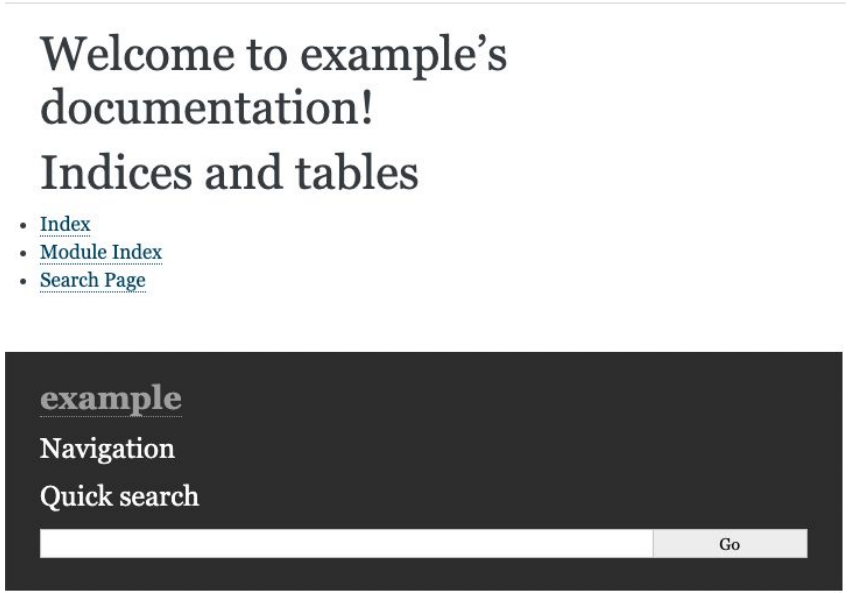
- uses reStructuredText (RST) as its markup language
- Outputs to HTML, plain text, LaTeX, and PDF
- Extensive cross-references
- Hierarchical Structures with automatic indices

Sphinx: creates intelligent & beautiful documentation

RST:

```
1  Welcome to example's documentation!
2  ===================================
3
4  .. toctree::
5     :maxdepth: 2
6     :caption: Contents:
7
8  Indices and tables
9  ==================
10
11 * :ref:`genindex`
12 * :ref:`modindex`
13 * :ref:`search`
14
```

Sphinx rendered html:



Standard Python Toolbox

- Formatting
  - isort
  - Pylint
  - Pycodestyle
  - Black
- Code Quality
  - Mypy
  - Pytest
  - Coverage.py
- Documentation
  - Sphinx

Did I miss anything?