How to test and document your Python code

ZWE Software Workshop

Max-Planck-Institut für Intelligente Systeme

Python Introductory Workshop, Stuttgart, December 18, 2020

MAX PLANCK INSTITUTE
FOR INTELLIGENT SYSTEMS

Outline

Outline

Why?

- Write better code
- Save time
- Not look like an idiot
- It is fun! :)

Different types of testing

- Unit testing: testing an individual component/functionality
- Integration testing: testing components grouped together
- Functional testing: testing the generated output (black-box)
- Acceptance/validation testing: testing outputs against requirements
- Alpha testing: Testing by developers before release
- Beta testing: Testing by customers before release
- ...

How?

The standard framework for testing Python code is unittest.

- Create `tests` packages
- Create modules that contains the tests with appropriate names
- Run for example:

```
$ python -m unittest
$ python -m unittest test_module1 test_module2
$ python -m unittest test_module.TestClass
$ python -m unittest test_module.TestClass.test_method
```

Alternative: pytest

Outline

Why?

- Code usability
- Knowledge transfer
- Manage expectations
- It is ~~fun~~ rewarding! :)

How?

The standard framework for writing documentation for Python code is Sphinx.

- Create `doc` folder
- Install sphinx with pip:
  ```
  $ pip install sphinx
  $ pip install sphinx_bootstrap_theme
  ```
- Run:
  ```
  $ sphinx-quickstart
  $ make html
  ```
  or
  ```
  $ sphinx-build -b html source_dir build_dir
  ```

Alternative: Doxygen