pytest-ordering Documentation

Release 0.5

Frank Tobia

Nov 04, 2017

Contents

pytest-ordering is a pytest plugin to run your tests in any order that you specify. It provides custom markers that say when your tests should run in relation to each other. They can be absolute (i.e. first, or second-to-last) or relative (i.e. run this test before this other test).

Note: pytest-ordering is currently alpha-quality software. Notably, some of this documentation may be aspirational in nature. If something you read here isn't currently implemented, rest assured that I am working on it (or feel free to ping me: https://github.com/ftobia)

CHAPTER 1

Quickstart

Ordinarily pytest will run tests in the order that they appear in a module. For example, for the following tests:

```
def test_foo():
    assert True

def test_bar():
    assert True
```

Here is the output:

```
$ py.test test_foo.py -vv
============================= test session starts ==============================
platform darwin -- Python 2.7.5 -- py-1.4.20 -- pytest-2.5.2 -- env/bin/python
collected 2 items

test_foo.py:2: test_foo PASSED
test_foo.py:6: test_bar PASSED

========================== 2 passed in 0.01 seconds ==========================
```

With pytest-ordering, you can change the default ordering as follows:

```
import pytest

@pytest.mark.order2
def test_foo():
    assert True

@pytest.mark.order1
def test_bar():
    assert True
```

```
$ py.test test_foo.py -vv
============================= test session starts ==============================
```

```
platform darwin -- Python 2.7.5 -- py-1.4.20 -- pytest-2.5.2 -- env/bin/python
plugins: ordering
collected 2 items

test_foo.py:7: test_bar PASSED
test_foo.py:3: test_foo PASSED

========================== 2 passed in 0.01 seconds ==========================
```

This is a trivial example, but ordering is respected across test files.

CHAPTER 2

Other markers

You can also use markers such as “first”, “second”, “last”, and “second_to_last”:

```
import pytest

@pytest.mark.second_to_last
def test_three():
    assert True

@pytest.mark.last
def test_four():
    assert True

@pytest.mark.second
def test_two():
    assert True

@pytest.mark.first
def test_one():
    assert True
```

```
$ py.test test_foo.py -vv
============================= test session starts ==============================
platform darwin -- Python 2.7.5 -- py-1.4.20 -- pytest-2.5.2 -- env/bin/python
plugins: ordering
collected 4 items

test_foo.py:17: test_one PASSED
test_foo.py:12: test_two PASSED
test_foo.py:3: test_three PASSED
test_foo.py:7: test_four PASSED

========================== 4 passed in 0.02 seconds ===========================
```

CHAPTER 3

Aspirational

This section is for functionality I'd like to implement. Documentation-driven design :)

3.1 Ordinals

```
import pytest

@pytest.mark.run('second-to-last')
def test_three():
    assert True

@pytest.mark.run('last')
def test_four():
    assert True

@pytest.mark.run('second')
def test_two():
    assert True

@pytest.mark.run('first')
def test_one():
    assert True
```

```
$ py.test test_foo.py -vv
============================= test session starts ==============================
platform darwin -- Python 2.7.5 -- py-1.4.20 -- pytest-2.5.2 -- env/bin/python
plugins: ordering
collected 4 items

test_foo.py:17: test_one PASSED
test_foo.py:12: test_two PASSED
test_foo.py:3: test_three PASSED
test_foo.py:7: test_four PASSED
```

```
========================== 4 passed in 0.02 seconds ==========================
```

3.2 By number

```
import pytest

@pytest.mark.run(order=-2)
def test_three():
    assert True

@pytest.mark.run(order=-1)
def test_four():
    assert True

@pytest.mark.run(order=2)
def test_two():
    assert True

@pytest.mark.run(order=1)
def test_one():
    assert True
```

```
$ py.test test_foo.py -vv
============================= test session starts ==============================
platform darwin -- Python 2.7.5 -- py-1.4.20 -- pytest-2.5.2 -- env/bin/python
plugins: ordering
collected 4 items

test_foo.py:17: test_one PASSED
test_foo.py:12: test_two PASSED
test_foo.py:3: test_three PASSED
test_foo.py:7: test_four PASSED

========================== 4 passed in 0.02 seconds ==========================
```

3.3 Relative to other tests

```
import pytest

@pytest.mark.run(after='test_second')
def test_third():
    assert True

def test_second():
    assert True

@pytest.mark.run(before='test_second')
def test_first():
    assert True
```

```
$ py.test test_foo.py -vv
============================= test session starts ==============================
platform darwin -- Python 2.7.5 -- py-1.4.20 -- pytest-2.5.2 -- env/bin/python
plugins: ordering
collected 3 items

test_foo.py:11: test_first PASSED
test_foo.py:7: test_second PASSED
test_foo.py:4: test_third PASSED

========================== 4 passed in 0.02 seconds ==========================
```