Python

базовый тренинг

Рузин Алексей
ruzin@me.com

Тестирование

- doctests
- unittests
- py.test
- mock, patch

doctests

```
def square(x):
    """
    >>> square(2)
    4
    """
    return x*x
```

unittest

- import unittest
- class XYZTestCase(unittest.TestCase):
    def test_xyz(self):
        self.assertEqual(3, 3)

- тесты класть в файлы вида: **test_*.py**

unittest (2)

- assertEqual(a, b)
- assertTrue(x)
- assertIs(a, b)
- assertIsNone(x)
- assertIn(a, b)
- assertIsInstance(a, b)
- assertNotEqual(a, b)
- assertFalse(x)
- assertIsNot(a, b)
- assertIsNotNone(x)
- assertNotIn(a, b)
- assertNotIsInstance(a, b)

unittest (3)

- assertRaises(exception, callable, *args, **kwargs)

unittest (4)

- def setUp(self):
  …
- def tearDown(self):
  …
- @classmethod
  def setUpClass(cls):
  …
- @classmethod
  def tearDownClass(cls):
  …

pytest

- def func(x):
    return x + 1

- def test_func():
    assert func(3) == 4
    assert func(4) == 6

pytest (2)

- файлы:
  test_*.py или *_test.py

- функции:
  def test_*()

- методы в классах:
  class TestXYZ():
    def test_*()

pytest (3)

- Fixtures
- pytest --fixtures
- def test_func(tmpdir):
  print(tmpdir)
- @pytest.fixture
  def **myfixture**():
  return …

  def test_func(**myfixture**):
  …

pytest (4)

- @pytest.fixture(scope=«module»)
  def fixturename():
    print(‘setup code’)
    yield ‘somevalue’
    print(‘teardown code’)

pytest (5)

- def test_func():
    with pytest.raise(ValueError(‘msg’)) as excinfo:
        call_method_which_raises_value_error()
    assert excinfo.value.contains(‘msg’)

unittest.mock

- from unittest.mock import Mock, MagicMock

- mock = Mock()
  mock = MagicMock()

unittest.mock (2)

- MagicMock(return_value=3)
- MagicMock(side_effect=Exception('msg'))
- MagicMock(side_effect=iterable)
- mock = MagicMock()
  mock.return_value = …
  mock.side_effect = …

  mock.my_method.return_value = …

unittest.mock (3)

- mock = MagicMock()
  mock.my_method.return_value = 3
  assert mock.my_method(4) == 3
  mock.my_method.assert_called_with(4)

- obj = SomeObject()
  obj.method = MagicMock(return_value=3)
  obj.another_method() # calls self.method()
  obj.method.assert_called_with()

- mock.my_method.mock_calls # список вызовов

unittest.mock (4)

- import collections

```
with patch('collections.defaultdict') as mock:
    mock.return_value.__getitem__.return_value = 3
    d = collections.defaultdict()
    assert d[5] == 3
```

unittest.mock (5)

- mock = MagicMock(spec=SomeClass)

- mock.non_existing_method() # raises Exception

unittest.mock (6)

```
@patch('lesson_05.answers.Employee.get_salary', lambda self: 100)
def test():
    from lesson_05.answers import Employee
    e = Employee('Olga', 333)
    assert e.get_salary() == 100
```

ССЫЛКИ

- http://docs.python-guide.org/en/latest/writing/tests/

Задача

- Придумать и написать тесты к задаче 2 из занятия 5 (спроектировать систему классов Сотрудник-Менедежер-Отдел).