

Python. Тестирование.

@pvavilin

5 июня 2022 г.

Outline

Какое бывает тестирование

- manual testing (ручное тестирование)
- performance testing (тестирование производительности)
- unit-testing (юнит-тестирование)
- functional testing (функциональное тестирование)
- integration testing (интеграционное тестирование)
- regression testing (регрессионное тестирование)

Ручное тестирование

Обычно у тестировщика есть некоторый план *стандартных задач и действий пользователя*. Тестировщик по этому плану проходит, может пробовать вводить нестандартные значения в формы, пытаться "перехитрить" нашу программу

Performance testing

Время работы

```
import time
start = time.time()
COUNT = int(1e7)
result = sum(x for x in range(COUNT))
finish = time.time()
print(
    f"Time to summarize {COUNT:.0E}: "
    f"{finish-start:.5f}"
)
Time to summarize 1E+07: 0.41765
```

Performance testing

Время работы

```
import timeit
from pprint import pp
pp(timeit.repeat(
    "sum(x for x in range(int(1e7)))",
    number=1,
    globals=globals()
))
[0.40609834200222394,
0.40540882700224756,
0.4056113639999239,
0.40464486299970304,
0.4057523429983121]
```

Performance testing

Профилирование по времени

```
import cProfile  
  
cProfile.run(  
    "sum(x for x in range(int(1e7)))"  
)
```

Performance testing

Профилирование по памяти

```
@profile
def my_func():
    a = (x for x in range(int(1e7)))
    b = [x for x in range(int(1e7))]
    del b
    return a

if __name__ == '__main__':
    my_func()

python -m memory_profiler mem_prof.py
```


Performance testing

Apache Benchmark

```
| sudo apt install apache2-utils  
| ab -V  
| ab -c 10 -n 8000 http://localhost:8000/
```

unit-tests

- Обычно пишутся самими программистами.
- В идеале, сначала программисты пишут тесты, а потом покрывают эти тесты кодом. TDD: Test Driven Development
- Каждый тест должен быть независим от других, многие фреймворки позволяют запускать тесты параллельно.

Unittest vs Pytest

- Unittest встроен в Python, pytest надо устанавливать.
- pytest богаче в плане вывода отчётов, параметризации тестов, поэтому используется чаще.
- pytest поддерживает запуск тестов для unittest (имеет обратную совместимость с unittest)

fixtures

фикстуры

- Описывают данные, которые должны быть созданы в тестовой базе данных.
- Можно выгрузить данные из реальной базы и затем добавить их в тестовые данные.

Faker

pip install faker

```
from faker import Faker
```

```
fake = Faker("ru_RU.UTF-8")  
print(fake.address())  
print(fake.phone_number())  
print(fake.ipv4())
```

с. Ессентуки, ул. Комсомольская, д. 84, 612596
+7 (983) 284-20-62
61.77.16.12

FactoryBoy

документация

- Описывает фикстуры как Python-код
- Возможность использовать Faker для создания случайных данных

functional testing

Главное отличие от Unit-tests в том, что пишущий такие тесты не знает об устройстве программы. Такой подход называется **black-box**.

Функциональные тесты проверяют, что вызов некоторого API с конкретными параметрами вернёт конкретный результат.

functional testing

Doctest

```
import doctest
def square(x):
    """Return the square of x.

    >>> square(2)
    4
    >>> square(-2)
    0
    """
    return x * x
if __name__ == "__main__":
    doctest.testmod()
```


integration testing

В отличие от unit tests, мы тестируем модули на совместную работу. Например

Создание комментариев к записям и удаление записей в отдельных unit-тестах уже протестировано. В интеграционном тесте нам необходимо протестировать удаление записей после создания комментария.

BDD

Behave Driven Development

Feature: Rocking with behave and django

Scenario: тестовый клиент Django

When django-клиент обращается к адресу "/"

Then это должно вернуть страницу удачно

And я увижу заголовок вкладки \

"Последние обновления | Yatube"

Тестирование GUI

```
pip install selenium

from selenium import webdriver
from selenium.webdriver.common.keys import Keys

driver = webdriver.Chrome()
driver.get("http://www.python.org")
assert "Python" in driver.title
elem = driver.find_element_by_name("q")
elem.clear()
elem.send_keys("pycon")
elem.send_keys(Keys.RETURN)
assert "No results found." not in \
    driver.page_source
driver.close()
```

Автоматизация тестирования

```
| vim .git/hooks/pre-commit  
| #!/bin/bash  
| pylint .  
| [ $? -ne 0 ] && exit 1  
| mypy *.py  
| [ $? -ne 0 ] && exit 1  
| pytest .  
| [ $? -ne 0 ] && exit 1  
| chmod +x .git/hooks/pre-commit
```

Полезные ссылки

- TDD
- MDN. Тестирование Django
- Pytest + Django
- Selenium & Python
- Selenium & BDD

Вопросы-ответы

