pytest

Тестирование приложений

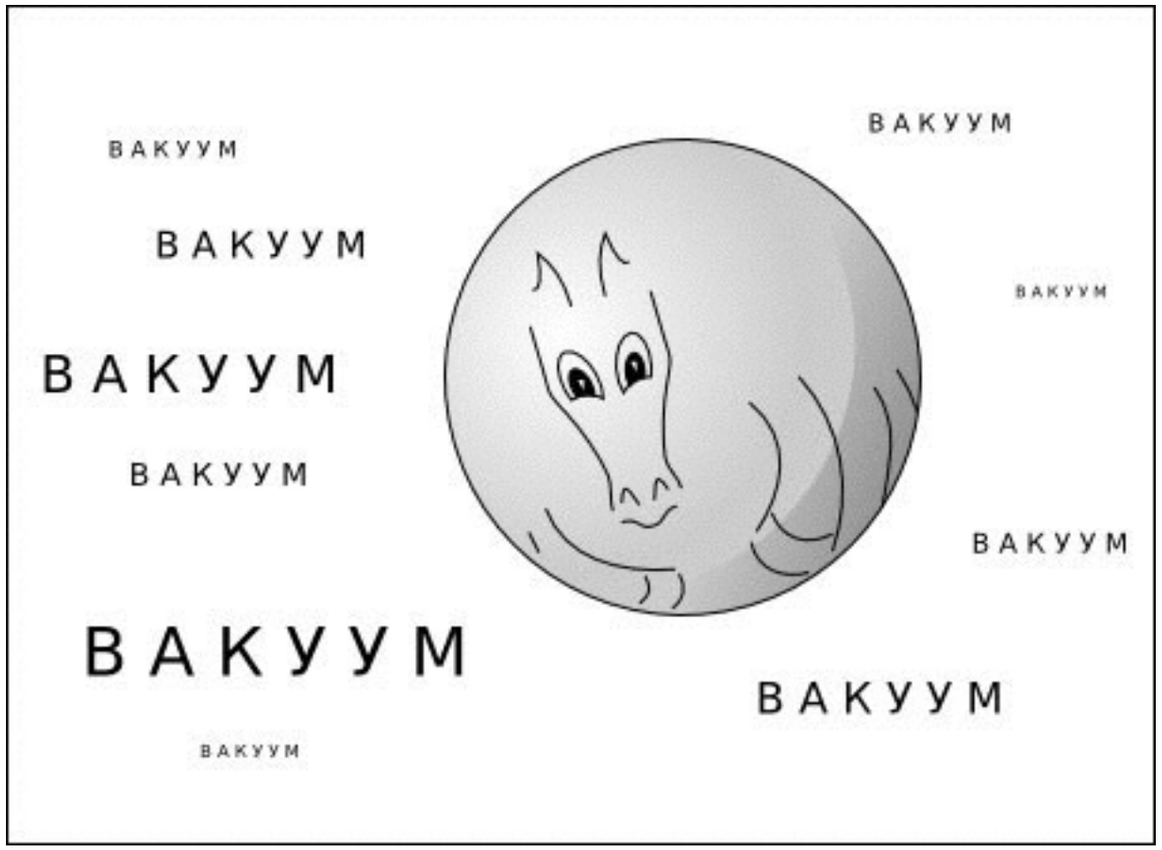
- Является банальной необходимостью при больших проектах, или для проектов, где несколько исполнителей
- Тесты бывают разные
- Приложение без тестов - не считается рабочим

Типы тестов

- Unit
- Интеграционные
- Нагрузочные

Общие положения

- Все тесты работают в изоляции
- Все тесты построены на принципе воспроизводимости
- В одном тесте должно быть одно утверждение


ВАКУУМ
ВАКУУМ
ВАКУУМ
ВАКУУМ
ВАКУУМ
ВАКУУМ
ВАКУУМ
ВАКУУМ
ВАКУУМ
ВАКУУМ
ВАКУУМ

Практика

Файлы "basic_unit_test/*.py"

Fixtures

- Из-за того, что тесты выполняются в изоляции, следует: что мы должны управлять состоянием, в котором они работают из-вне
- Для такого управления используются fixtures
- Они представляют собой просто набор данных, который должен быть в системе для корректной работы
- Еще они могут заключать в себе ожидаемый результат

Unit-тесты

- Тестируют работу одного unit'а кода: чаще всего функции (метода)
- На один unit кода приходится несколько тестов
- Просты в написании и выполнении

Интеграционные тесты

- Интеграционные тесты используются в больших проектах, где необходимо проверить большое количество взаимодействий разных подсистем

- Интеграционные тесты - медленные, сложны в написании

Mock

- Из того, что интеграционные тесты запускаются редко следует, что необходим некий механизм, который бы симулировал работу других частей системы (внутренних и внешних) в тестах

- Mock - инструмент, позволяющий подменять такие части динамически

- Особенно полезен при работе с внешними сервисами

- Создает дополнительную изоляцию

Что есть?

- https://habrahabr.ru/post/121162/
- http://docs.python-guide.org/en/latest/writing/tests/
- https://docs.djangoproject.com/en/1.9/topics/testing/
- https://pypi.python.org/pypi/mock

Насколько хорошо написаны тесты?

- Тестов мало не бывает

- Тесты должны покрывать код - https://coverage.readthedocs.org/en/coverage-4.0.3/

Практика

Тесты к pizza_project