Введение в модульное тестирование на языке Python

Введение

- Что такое тестирование программного обеспечения в целом?
- Тестирование представляет собой отдельную сложную дисциплину. Специалисты по тестированию – это отдельные специалисты, которые не занимаются разработкой ПО.
- В настоящее время тестирование рассматривается как элемент DevOps.
- Тем не менее, в большинстве проектов разработчик должен осуществлять модульное тестирование, чтобы дать гарантии того, что его код работает.

Разработка через тестирование

- Предполагается что тесты не вводятся пользователем, а пишутся в виде программ, так называемое «автоматизированное тестирование».
- В методологии «экстремального программирования» одним из основных элементов является «разработка через тестирование» или test-driven development, TDD.
- Не существует прямых доказательство того, что написание большого количества тестов делает код более качественным.
- Но большое количество тестов позволяет разработчику более уверенно вносить в ПО изменения.

Разработка через тестирование - 2

- Одним из наиболее парадоксальных утверждений TDD является утверждение о том, что нужно писать минимальный код, который успешно проходит тестирование.
- Конечно, это утверждение не работает при проектировании сложных алгоритмов.
- Но это утверждение соответствует YAGNI-принципу.
- Также это утверждение соответствует подходу «бережливой разработки программного обеспечения».

Разработка через тестирование - 3

- Для того, чтобы разорвать сложные цепочки зависимостей между объектами и тестировать элементы системы отдельно, применяется подход на основе Mock-объектов.
- Использование подходов IOC и DI не связано напрямую с тестированием. Но применение этих подходов очень упрощает тестирование, так как позволяет создавать тестовые «точки сборки» с применением Mock-объектов.
- Необходимо отметить, что IOC/DI является одним из принципов SOLID.

Разработка через поведение

- Behavior-driven development, BDD.
- В отличие от TDD позволяет писать тесты непрограммирующим пользователям.
- Описание подхода в википедии.
- Описание языка Gherkin.
- Как правило, BDD-фреймворки являются надстройками над TDD-фреймворками.

Инструменты на основе Python (встроенные)

- Список фреймворков для тестирования.
- Встроенный фреймворк unittest. Классический подход к unit-тестированию.
  - Тесты реализуются как методы класса.
  - Методы assert для проверки условий тестирования.
  - Служебные методы для инициализации и финализации теста.
  - Создание Mock-объектов.
- Встроенный фреймворк doctest. Позволяет упрощать написание тестов с использованием doc-строк или текстовых файлов.

Инструменты на основе Python (внешние)

- Top 6 BEST Python Testing Frameworks [Updated 2022 List].
- Одним из наиболее развитых фреймворков является pytest.
  - Поддержка Mock-объектов.

Инструменты BDD на основе Python

- Статья с обзором фреймворков.
- Фреймворк radish:
  - Схема работы.
  - Github проекта.
- Фреймворк pytest-bdd.
- Фреймворк behave.
- Фреймворк Robot. Не только реализует BDD, но и позволяет создавать тесты на основе произвольных текстовых сценариев.