Парадигмы и Конструкции Языков Программирования

ИУ-5, бакалавриат, 3 семестр

Введение в модульное тестирование и паттерны проектирования

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».
- В автоматизированном тестировании часто используется термин fixture.
- Для того, чтобы разорвать сложные цепочки зависимостей между объектами и тестировать элементы системы отдельно, применяется подход на основе Mock-объектов.

Инструменты TDD на основе Python

- 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 setUp и tearDown для инициализации и финализации теста.
  - Создание Mock-объектов.
- Встроенный фреймворк doctest. Позволяет упрощать написание тестов с использованием doc-строк или текстовых файлов.

Инструменты TDD на основе Python (внешние)

- Top 6 BEST Python Testing Frameworks.
- Одним из наиболее развитых фреймворков является pytest.
  - Поддержка Mock-объектов.

Принципы IOC/DI, SOLID, GRASP

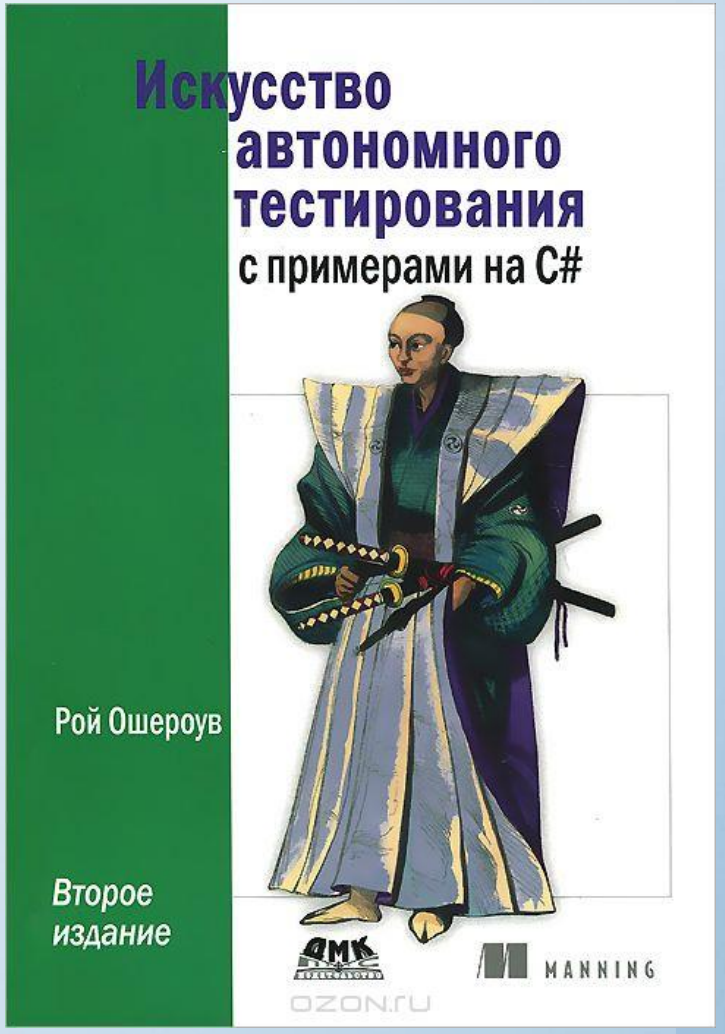
- При проектировании программных систем (прежде всего ООП-систем) выделяют как **принципы или «общие шаблоны» проектирования**, так и конкретные **шаблоны проектирования**.
- Использование подходов IOC и DI не связано напрямую с тестированием. Но применение этих подходов очень упрощает тестирование, так как позволяет создавать тестовые «точки сборки» с применением Mock-объектов.
- Необходимо отметить, что IOC/DI является одним из принципов SOLID (статья с описанием SOLID).
- Также IOC/DI позволяет реализовать «общие шаблоны» GRASP.

IOC/DI – реализация

- Общие принципы (Java). Понятие **точки сборки**.
- Для языка программирования C#:
  - **Описание на сайте Metanit**.
  - Пример из документации.
  - Список фреймворков.
  - Производительность фреймворков.
- Для языка программирования Python:
  - Статья.
  - **Комплексный пример с внедрением зависимостей и тестами**.

IOC/DI и тестирование

- Использование DI-фреймворков в тестах.
- Облегчение тестирования с использованием принципов DI.
- Использование Mock-объектов в DI.
- Создание тестовых точек сборки.

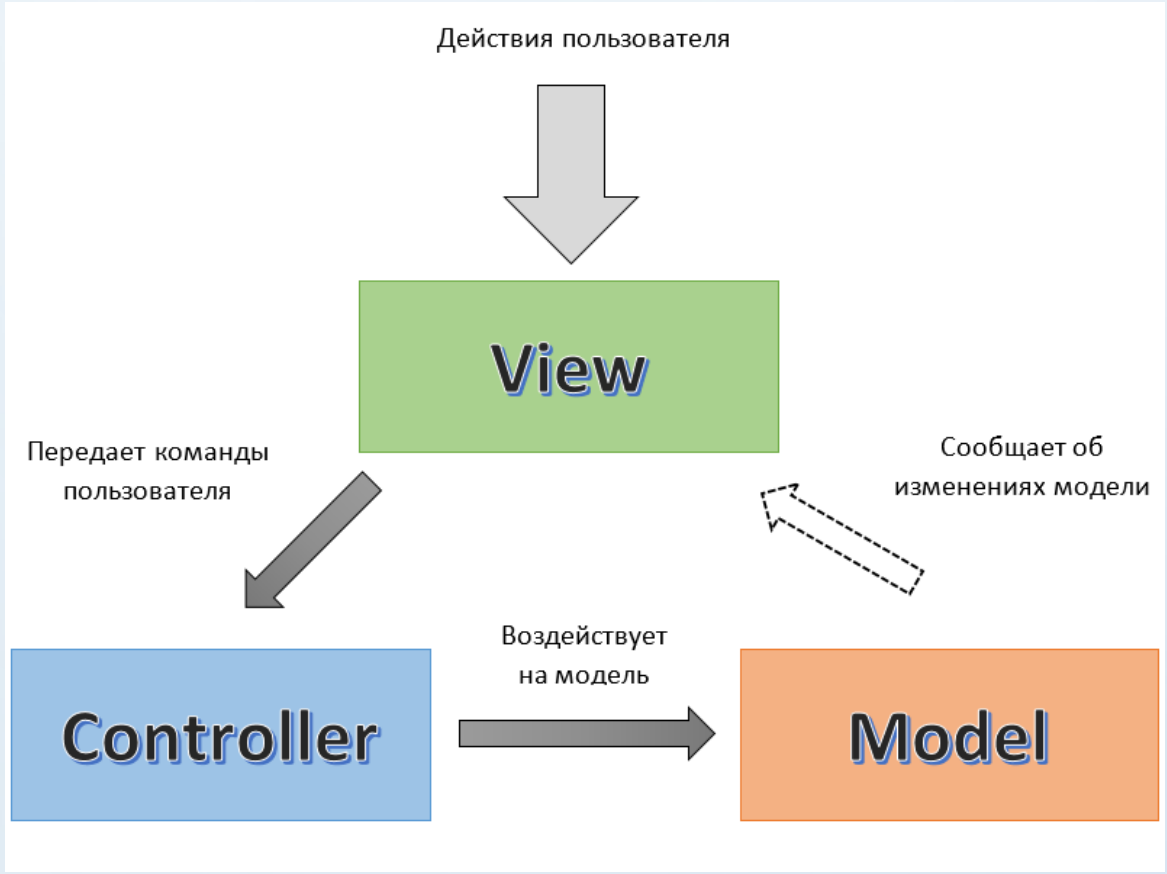


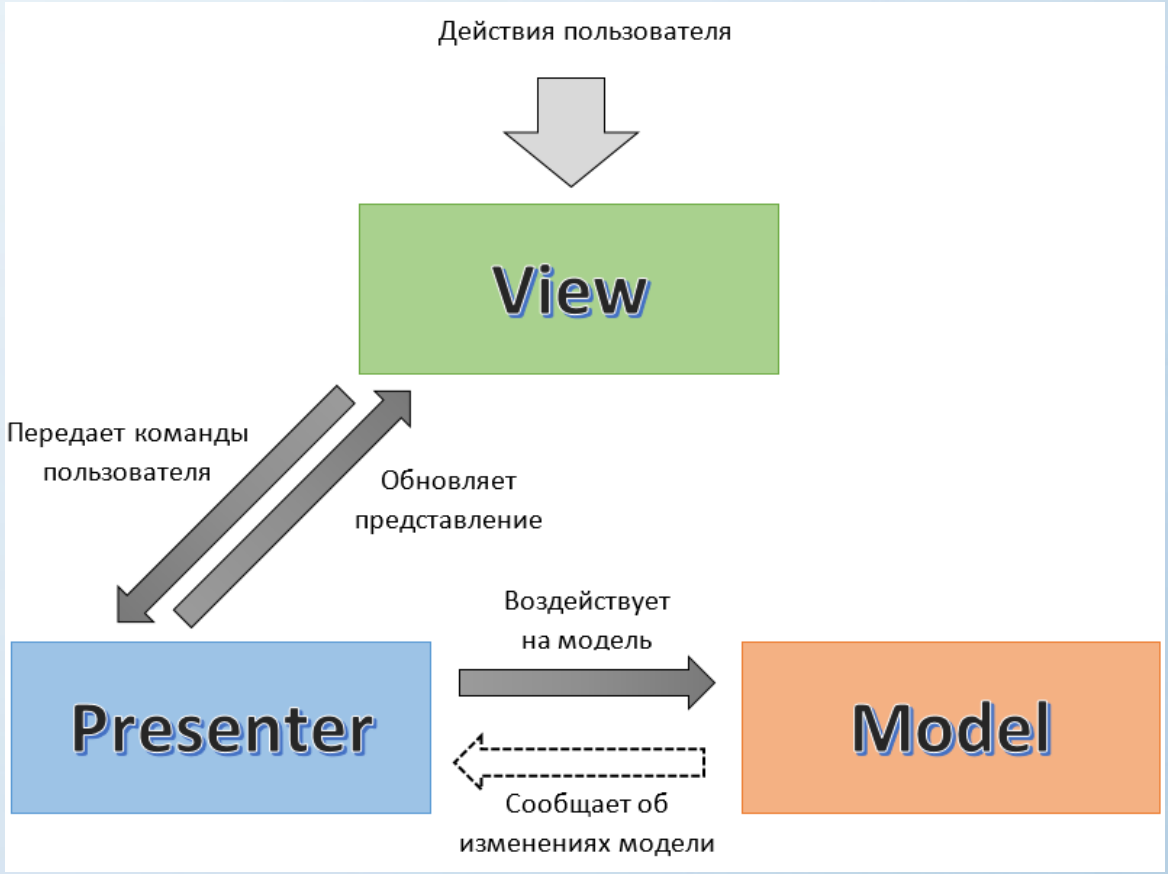
Шаблоны проектирования

- Перечень шаблонов (*Википедия*).
- Awesome Software and Architectural Design Patterns.
- Язык UML:
  - Описание в википедии.
  - Список инструментов.
  - PlantUML.

Шаблоны проектирования - MVC, MVP

- Архитектурные шаблоны (википедия)
- MVC, MVP, MVVM
- Статья с примерами.





Разработка через поведение (BDD)

- 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 BDD на основе Python

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