Университет ИТМО, факультет программной инженерии и компьютерной техники
Двухнедельная отчётная работа по «Информатике»: аннотация к статье
Дата прошедшей лекции: 06.12.23 Номер прошедшей лекции: №7 Дата сдачи: 20.12.23

Выполнил(а) Васильев Н. А., № группы P3108, оценка _____
Фамилия И.О. студента не заполнять

Разработка через тестирование. Совместное использование JUnit 5 и Mockito		
ФИО автора статьи (или e-mail) **NewTechAudit**	**Дата публикации (не старше 2020 года)** "10" ноября 2023 г.	**Размер статьи (от 400 слов)** 2137 слов

Прямая полная ссылка на источник или сокращённая ссылка (bit.ly, tr.im и т.п.)
https://habr.com/ru/companies/sberbank/articles/773142/

Теги, ключевые слова или словосочетания
Тестирование, TDD, JUnit, Mockito

Перечень фактов, упомянутых в статье (минимум три пункта)
1. TDD (test-driven development) — разработка через тестирование.
2. Основная схема TDD заключается в написании теста, на основании которого пишется код.
3. При написании тестов существует несколько принципов: разбивка задачи, структуризация, написание чистого и рабочего кода.
4. Для большинства языков существует множество фреймворков для тестирования.
5. Механизм тестирования — это компонент, отвечающий за выполнение тестов и представление результатов.
6. Самый оптимальный для TDD фреймворк на Java – JUnit позволяет создавать собственные механизмы тестирования для удовлетворения конкретных требований и использовать их с различными инструментами сборки, средами разработки и системами непрерывной интеграции.
7. Фреймворк для тестирования содержат классы и методы, которые можно использовать непосредственно в коде.

Позитивные следствия и/или достоинства описанной в статье технологии (минимум три пункта)
1. Тесты выявляют ошибки на ранних этапах разработки, что способствует созданию более стабильного и качественного кода.
2. Разработка через тестирование позволяет оптимизировать процессы обработки данных.
3. Разработчик получает быстрый обратный отклик о работоспособности кода после каждого изменения, что упрощает выявление и исправление проблем.
4. Тесты служат своего рода документацией для кода, описывая ожидаемое поведение компонентов системы.

Негативные следствия и/или недостатки описанной в статье технологии (минимум три пункта)
1. Поддержка большого количества тестов может потребовать дополнительных усилий, и в некоторых случаях тесты сами могут стать источником ошибок.
2. Написание тестов перед написанием кода может потребовать дополнительного времени, особенно на начальных этапах разработки.
3. TDD не всегда подходит для всех типов проектов и не может быть эффективным в некоторых сценариях, таких как экспериментальная разработка или прототипирование.

Ваши замечания, пожелания преподавателю *или* анекдот о программистах