Лабораторна робота №5

# Модульне тестування (Unit-тести) та рефакторинг.

**Мета роботи**: оволодіти навичками створення програмного забезпечення за методологією TDD та ознайомитися з процедурами рефакторинга.

**Задача:**

1. Використовувати методологію Test Driven Development для створення класів архітектурної програмної моделі.

2. Скласти тестові сценарії, які продемонструють функціювання всіх методів проектованої моделі.

3. Виконати юніт-тестування складових частин (внутрішніх класів), що реалізують об'єкт моделювання.

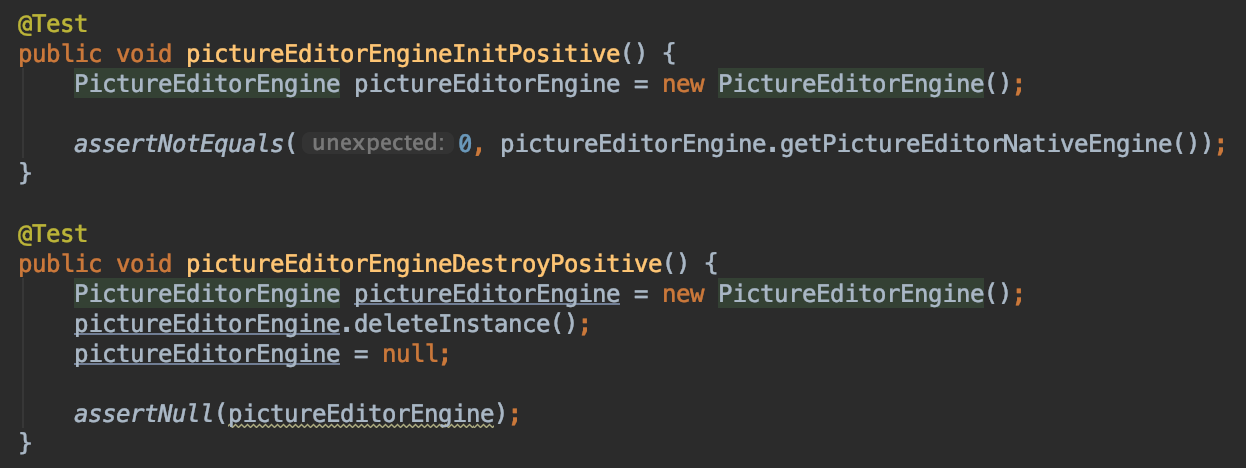
4. Виконати "зовнішнє" юніт-тестування для API.

5. Провести рефакторинг коду програми, для поліпшення реалізації.

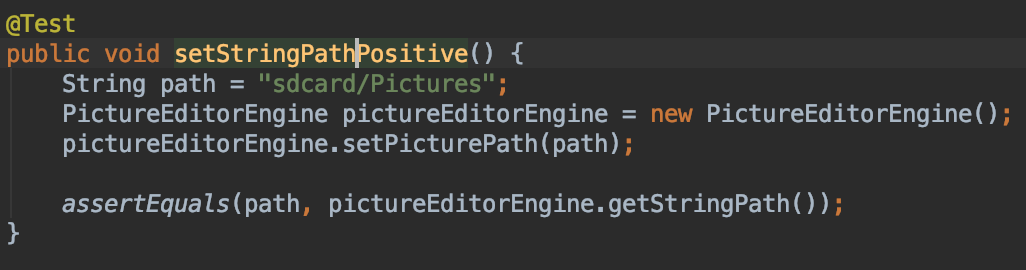
# Хід роботи

У цій лабораторній роботі буде покриватися публічне API бібліотеки яка редагує фотографії “pictureeditorlib”

1. Тестові сценарії:
2. Перевірити створення та видалення нативної С++ бібліотеки.
3. Перевірити правильність запису шляху фотографій у бібліотеку
4. Тестування логіки бібліотеки:
   1. Негативний та позитивний тест фільтрації фотографії
   2. Тест зберігання фільтрованої фотографії
   3. Негативний та позитивний тест блюру фотографії
   4. Тест зберігання заблюрованої фотографії
5. Скріншоти тестів:
6. Перевірити створення та видалення нативної С++ бібліотеки.



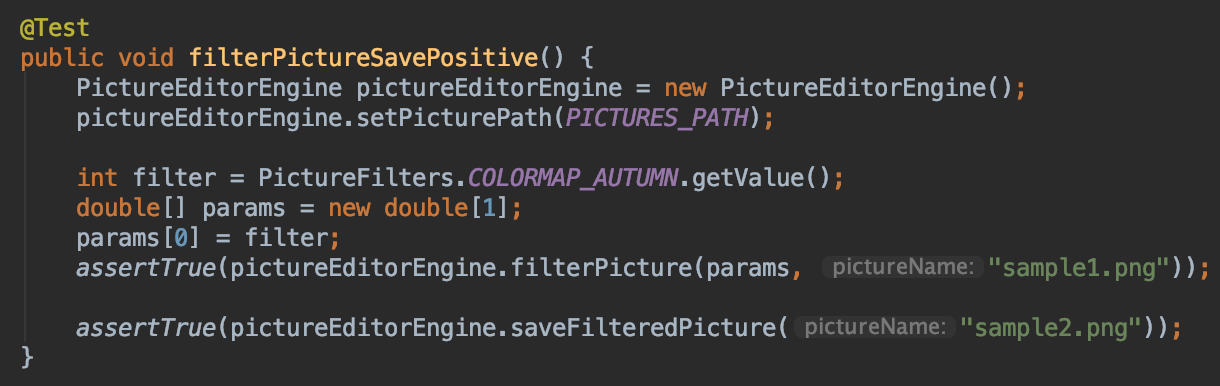
1. Перевірити правильність запису шляху фотографій у бібліотеку



1. Тестування логіки бібліотеки:
   1. Негативний та позитивний тест фільтрації фотографії



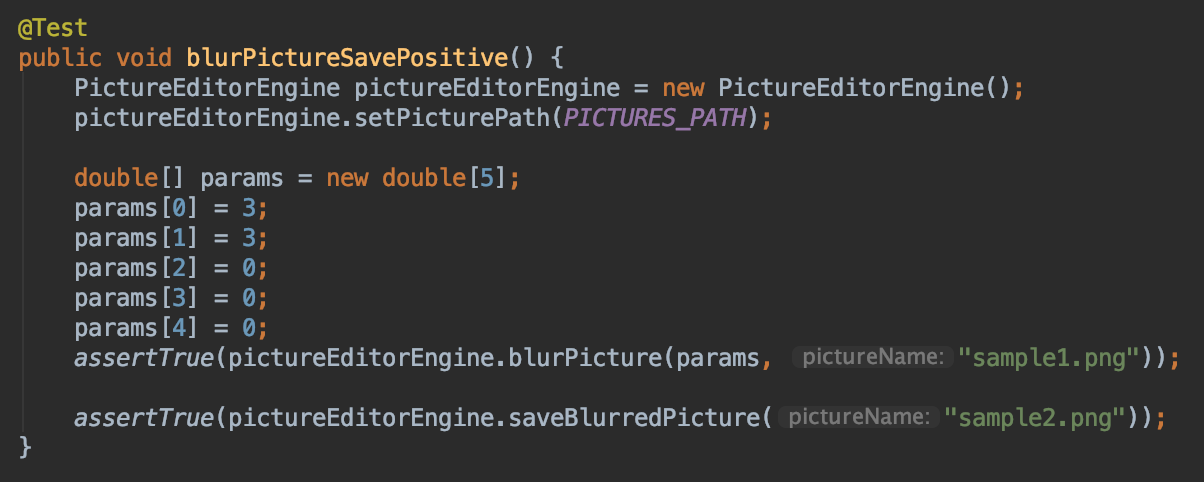
* 1. Тест зберігання фільтрованої фотографії



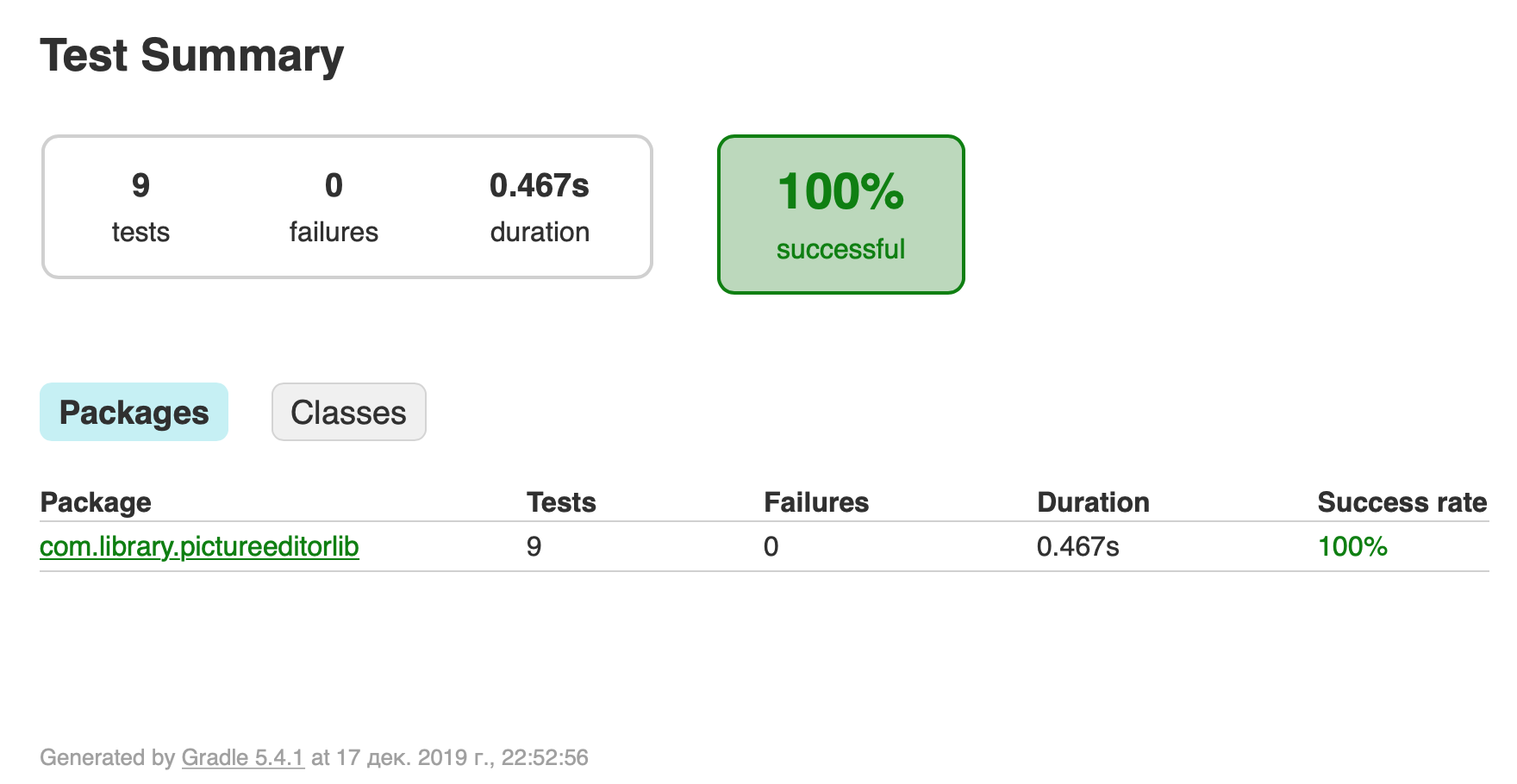
* 1. Негативний та позитивний тест блюру фотографії



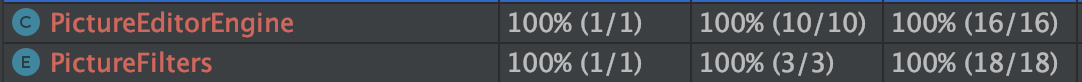
* 1. Тест зберігання заблюрованої фотографії



1. Згенеруємо тест репорт за допомогою команди *./gradlew createDebugCoverageReport*:
   1. Результат проходження тестів



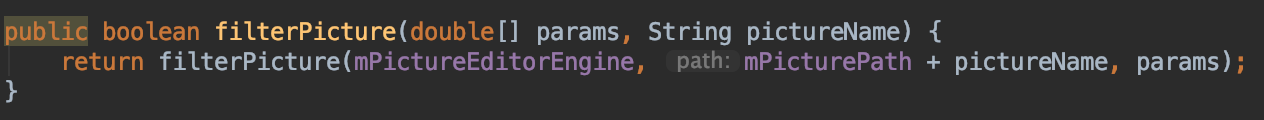
* 1. Результат покриття тестів



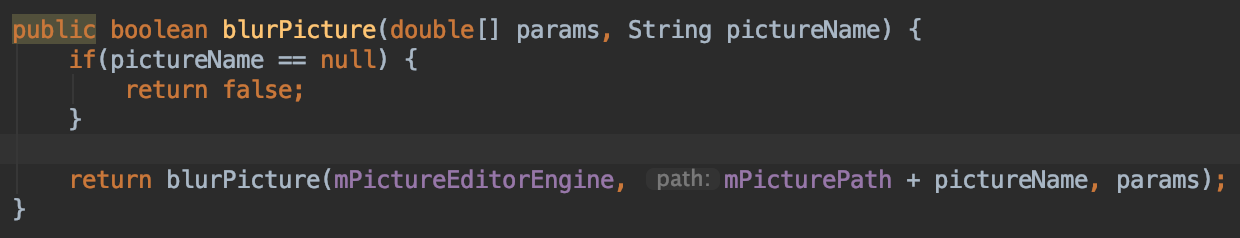
1. Проведемо рефакторинг коду:

Додамо перевірку на null для шляху до картинки, для функції де визивається функція фільтрація фотографії.

Було:



Стало:



# Висновок

В ході даної лабораторної роботи було розроблено тестові сценарії для перевірки роботи інформаційної системи. Розроблені юніт-тести успішно пройшли та покрили 100% коду. Також було проведено рефакторинг коду програми для підвищення читабельності програми та для того, щоб уникнути подальших помилок та непорозумінь.