Programming 5

Testing - introduction

Karel de Grote Hogeschool

- **Introduction**
- **JUnit**

Introduction

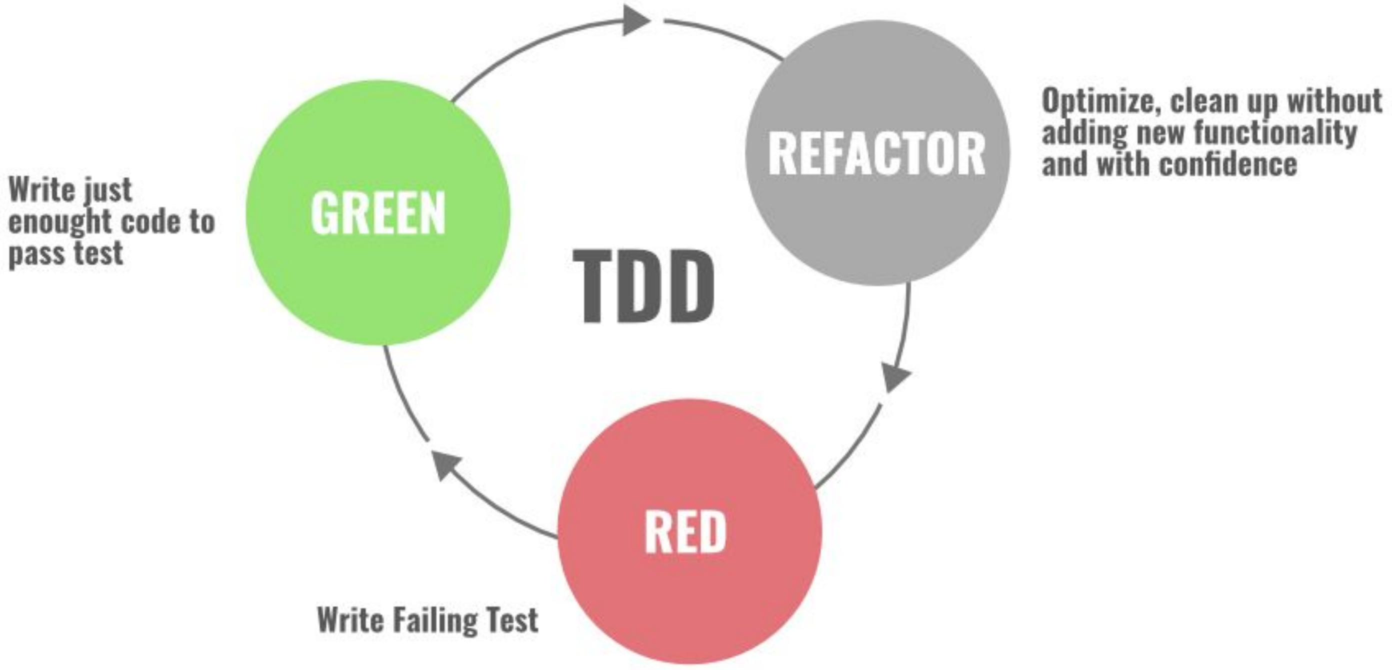
- Manual testing versus **automated testing**
  - Efficiency
  - Proof for correctness
  - Refactor-safe

⇒ Maintaining codebase with confidence!

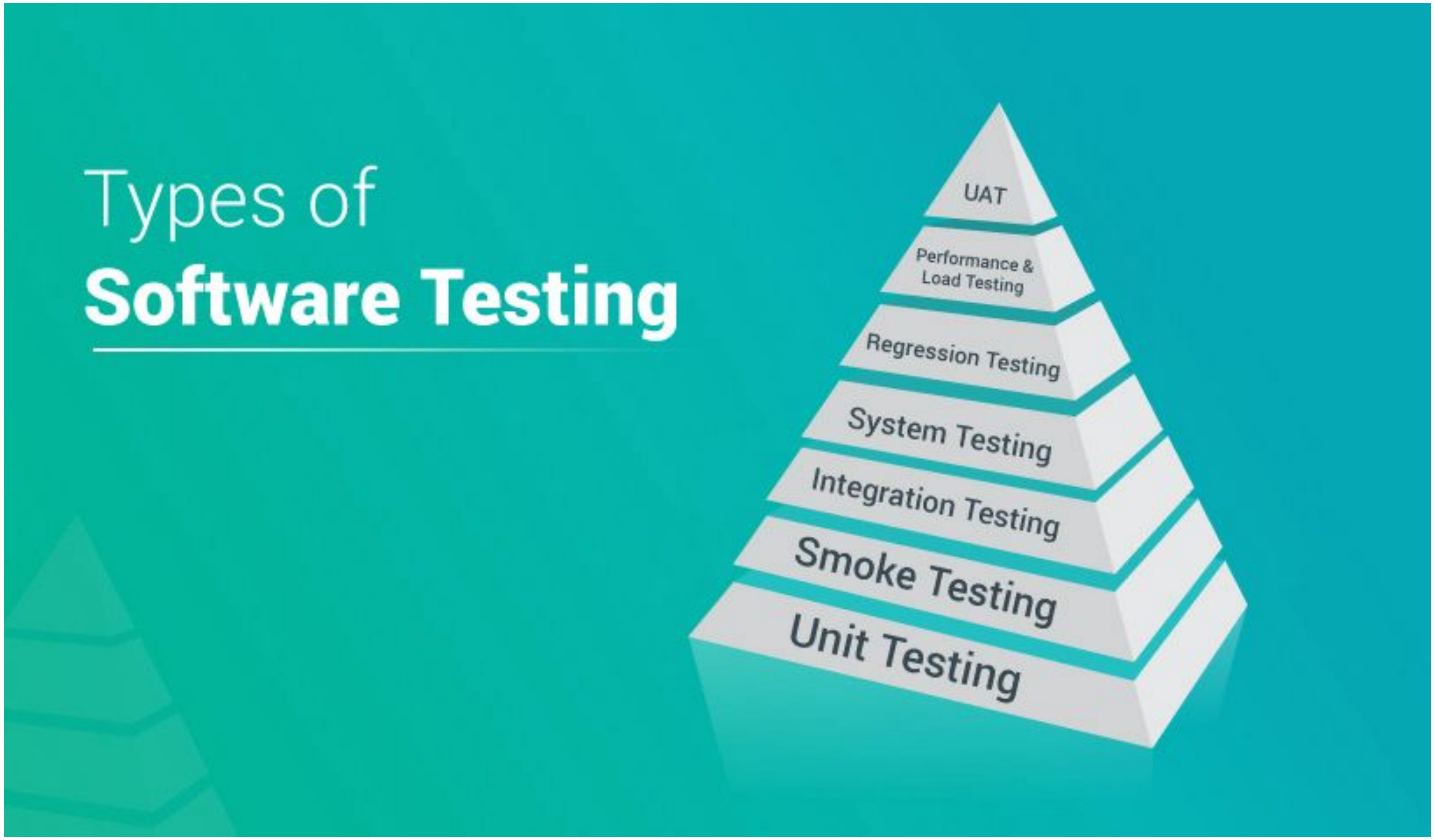
- Certain input leads to a certain output
- Write a separate test-class for each class
  - A certain method can have multiple corresponding test methods

Introduction

- Test Driven Development: write **tests first**, then the actual implementation


Optimize, clean up without adding new functionality and with confidence
REFACTOR
Write just enought code to pass test
GREEN
TDD
RED
Write Failing Test

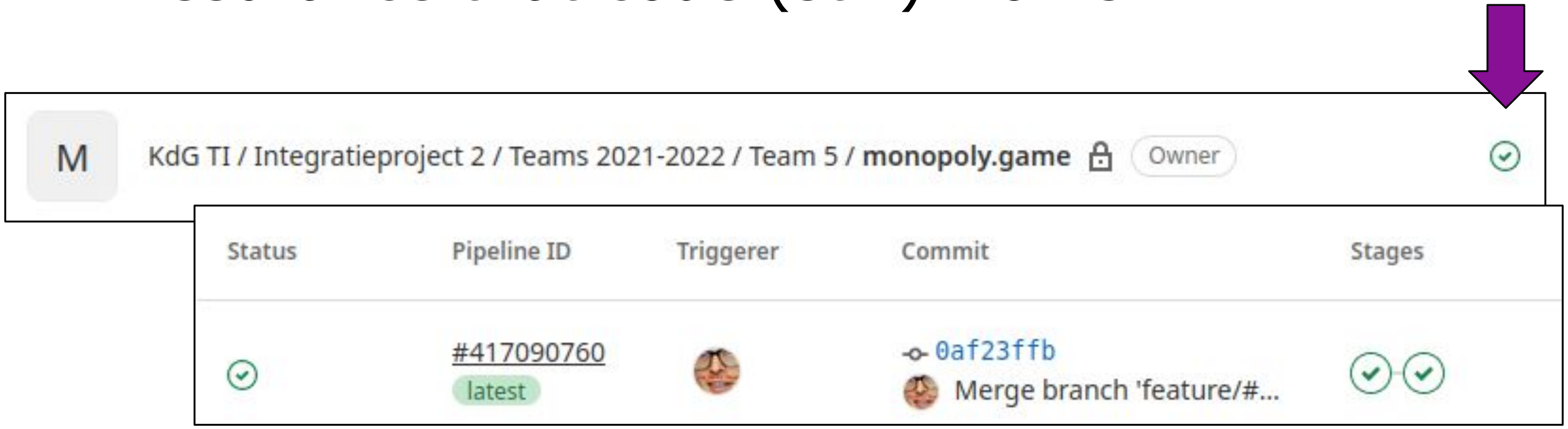
Testing Pyramid



https://testsigma.com/blog/the-different-software-testing-types-explained/

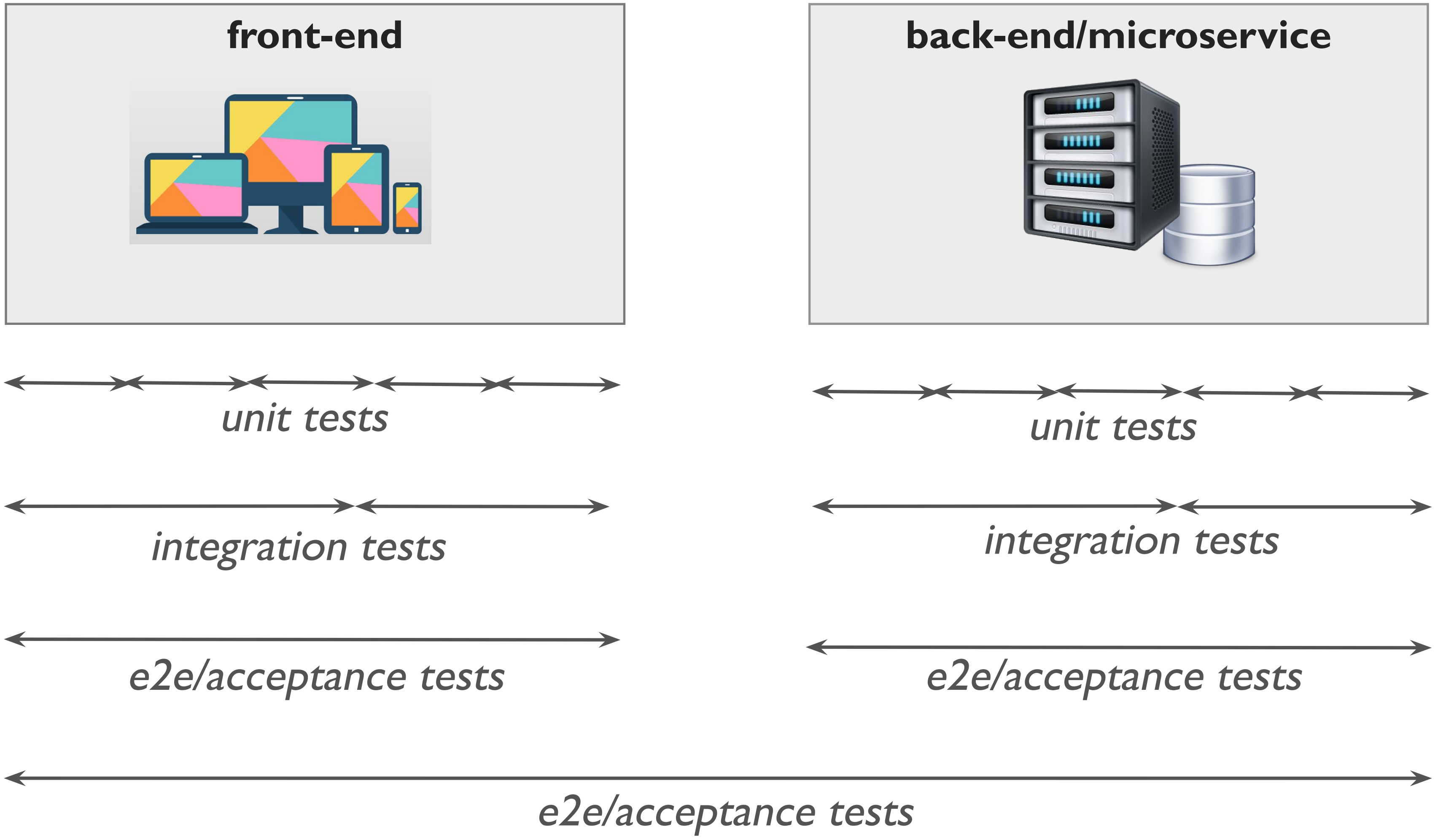
Introduction

- Regression testing
  Regression testing is re-running functional and non-functional tests to ensure that previously developed and tested software still performs after a change.
- Tests as part of a CI/CD pipeline
  - Fully automated tests
  - Assurance that code (still) works


M KdG TI / Integratieproject 2 / Teams 2021-2022 / Team 5 / **monopoly.game** Owner ✓

Status	Pipeline ID	Triggerer	Commit	Stages
✓	#417090760 latest		0af23ffb Merge branch 'feature/#...	✓ ✓

Types of software tests



*e2e = end-to-end = 'system' => test that passes the entire architecture

What makes a good test?

- Arrange / Act / Assert (AAA)
- Cover different kinds of inputs
  - Think of edge cases
  - Don’t just test the happy path!
  - Cover different fail scenarios
  - Cover all code paths (‘if’ branches, …)
  - Cover the entire contract / responsibilities of a method or component

What makes a good test?

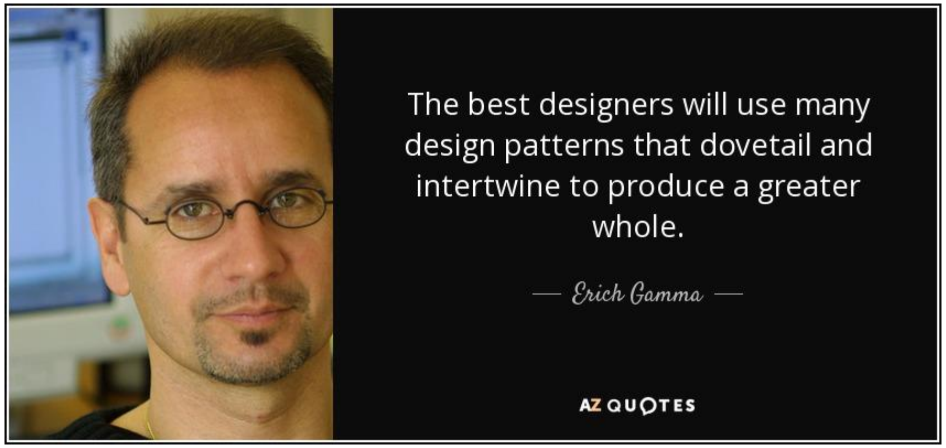
- Test your own code, don’t test a framework’s code
  - Frameworks have their own set of tests
- Try not to test the same code twice
  - Sometimes unavoidable (i.e., unit test + e2e test)
- Use meaningful method names
  - `acceptHeaderShouldBeRespected`
  - `nonExistentBookShouldReturnNotFound`
- Each test method should test a *single* case
  - Some success scenario, a specific failure, …
  - A certain method should be tested using *multiple* testing methods

- Introduction
- JUnit

JUnit 5

- JUnit 5: Unit Testing Framework

- **Erich Gamma**
  The "Gang of Four",
  Eclipse IDE
  Visual Studio Code IDE

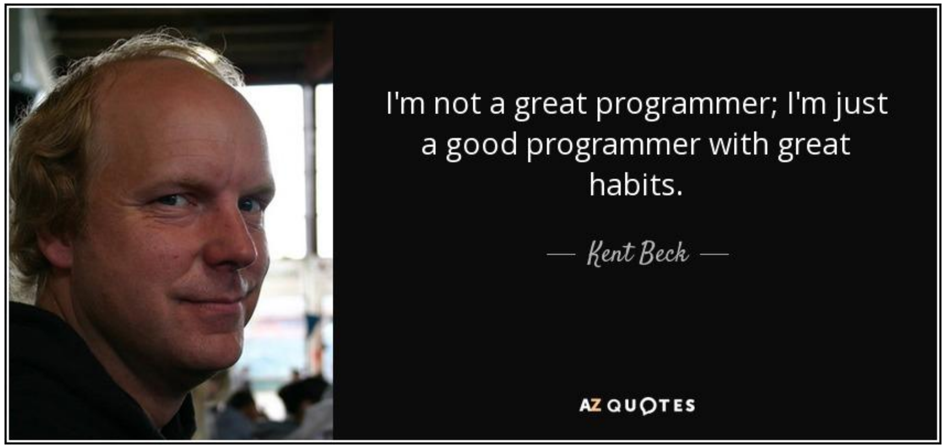


The best designers will use many design patterns that dovetail and intertwine to produce a greater whole.

— Erich Gamma —

AZ QUOTES

- **Kent Beck**
  Extreme Programming,
  CRC Cards, Agile



I'm not a great programmer; I'm just a good programmer with great habits.

— Kent Beck —

AZ QUOTES

JUnit 5

- Package: **`org.junit.jupiter`**
  - Packages under **`org.junit`** (*without* jupiter) are from JUnit 4 and earlier
  - Only use JUnit 5!
- **`build.gradle`:**

```
dependencies {
    ...
    testImplementation 'org.junit.jupiter:junit-jupiter-api:5.9.2'
    testRuntimeOnly 'org.junit.jupiter:junit-jupiter-engine:5.9.2'
}

test {
    useJUnitPlatform()
}
```

We won't be needing these dependencies (directly) when using Spring Boot.

JUnit 5

```java
import org.junit.jupiter.api.*;

public class FoobarTest {
   @BeforeAll
   public static void setUpClass() throws Exception {
       // Code executed before the first test method
   }

   @BeforeEach
   public void setUp() throws Exception {
       // Code executed before each test
   }

   @Test
   public void oneThing() {
       // Code that tests one thing
   }

   @AfterEach
   public void tearDown() throws Exception {
       // Code executed after each test
   }

   @AfterAll
   public static void tearDownClass() throws Exception {
       // Code executed after the last test method
   }
}
```

JUnit 5

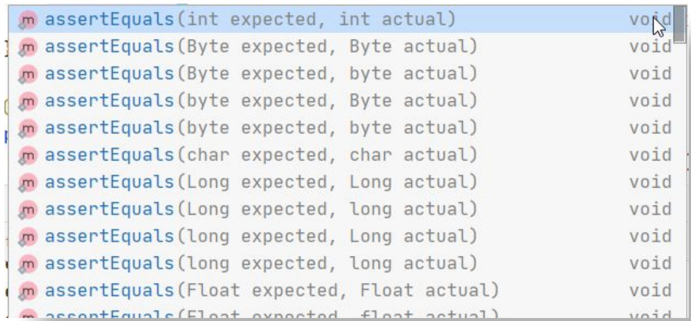
- `@Test`
  - All public void methods annotated with @Test will be executed. There are no guarantees about the order in which they are executed.
- `@BeforeEach`
  - Executed before each test
- `@AfterEach`
  - Executed after each test
- `@BeforeAll`
  - Executed once before all tests of this class
- `@AfterAll`
  - Executed once after all tests of this class
- `@Disable`
  - Temporarily disable a test

Assertions

- **`org.junit.jupiter.api.Assertions`**
- The assert methods are static

  ⇒ **`import static`**
- We’ll highlight a few asserts methods. For each method, there are the following overloaded variants:
  - Without error message
    - Standard Junit error message
  - With error message
    - final parameter: type String
    - final parameter: type Supplier<String>

Assertions


```
assertEquals(int expected, int actual)       void
assertEquals(Byte expected, Byte actual)     void
assertEquals(Byte expected, byte actual)     void
assertEquals(byte expected, Byte actual)     void
assertEquals(byte expected, byte actual)     void
assertEquals(char expected, char actual)     void
assertEquals(Long expected, Long actual)     void
assertEquals(Long expected, long actual)     void
assertEquals(long expected, Long actual)     void
assertEquals(long expected, long actual)     void
assertEquals(Float expected, Float actual)   void
```

- **`assertEquals`**
  - Method overloading: exists for different data types
  - For **`float`** and **`double`**, you can optionally provide an additional argument “delta” to specify a tolerance:

```
assertEquals(double expected, double actual, double delta);
```

- **`assertNotEquals`**
  - Tests the opposite of **`assertEquals`**

Assertions

- **`assertSame(Object expected, Object actual)`**
- **`assertNotSame(Object expected, Object actual)`**
  - Both arguments must refer to the *same object*. This is *not* the same as **`assertEquals`**!
- **`assertTrue(boolean condition)`**
  - The expression or lambda must yield true
  - There’s also an assertFalse for both variants
- **`assertNotNull(Object object)`**
- **`assertNull(Object object)`**
  - There must be an object. In other words, not null.

Assertions

- **`assertArrayEquals(Object[] expected, Object[] actual)`**
  - Both arrays contain the same elements (according to **`equals`**)
  - There are also overloaded methods for arrays of primitives
- **`assertIterableEquals(` `Iterable expected,`**
  `Iterable actual)`
  - Both iterables (Collections, …) contain the same elements (according to **`equals`**)

Assertions

- **`assertThrows(Class<T> expectedException, Executable lambda)`**
    - Assert whether the lambda throws the specified exception

```java
@Test
void authorNameIsUnique() {
    var author1 =  new Author();
    author1.setName("Unique author");
    authorRepository.save(author1);

    assertThrows(DataIntegrityViolationException.class, () -> {
        var author2 =  new Author();
        author2.setName("Unique author");
        authorRepository.save(author2);
    });
}
```

Assertions

- `fail()`
- `fail(String message)`
- `fail(Throwable exception)`
  - Causes a test to fail

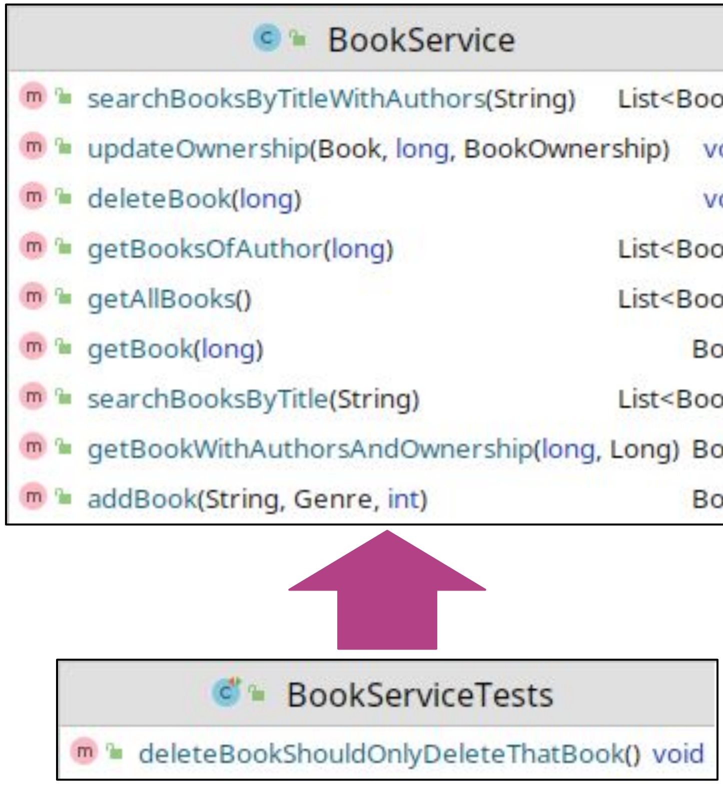
- Every test method should end with
  - Either one or more asserts
  - Or a fail

JUnit 5

- A Guide to JUnit 5
  - https://www.baeldung.com/junit-5
- Using JUnit 5 with Gradle
  - https://www.baeldung.com/junit-5-gradle

Organising tests

- Usually, we create one test class to correspond with an actual tested class



BookService

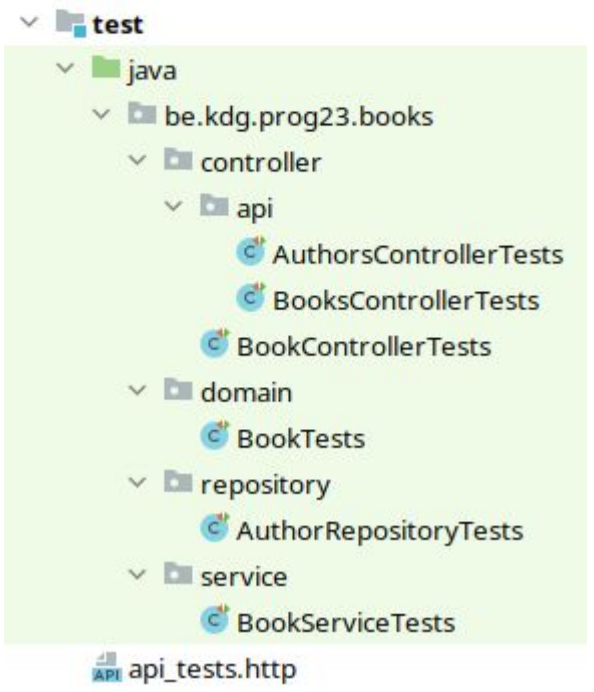
searchBooksByTitleWithAuthors(String)	List<Book>
updateOwnership(Book, long, BookOwnership)	void
deleteBook(long)	void
getBooksOfAuthor(long)	List<Book>
getAllBooks()	List<Book>
getBook(long)	Book
searchBooksByTitle(String)	List<Book>
getBookWithAuthorsAndOwnership(long, Long)	Book
addBook(String, Genre, int)	Book

BookServiceTests

deleteBookShouldOnlyDeleteThatBook()	void

Organising tests

- Test code is separated from application code (and is not distributed as part of the application)
- In Gradle/Maven:
  - **`src/test`**
  - (As opposed to **`src/main`**)
- Place test classes in the same package as the tested class



```
test
  java
    be.kdg.prog23.books
      controller
        api
          AuthorsControllerTests
          BooksControllerTests
        BookControllerTests
      domain
        BookTests
      repository
        AuthorRepositoryTests
      service
        BookServiceTests
  api_tests.http
```