**TDD using JUnit5 and Mockito**

**Exercise 1: Setting Up Junit**

**Code:**

<?xml version="1.0" encoding="UTF-8"?>

<project xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.example</groupId>

<artifactId>JUnitMockitoDemo</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>JUnitMockitoDemo</name>

<url>http://www.example.com</url>

<properties>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

<maven.compiler.source>17</maven.compiler.source>

<maven.compiler.target>17</maven.compiler.target>

</properties>

<dependencies>

<dependency>

<groupId>org.junit.jupiter</groupId>

<artifactId>junit-jupiter-api</artifactId>

<version>5.11.0</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>org.junit.jupiter</groupId>

<artifactId>junit-jupiter-engine</artifactId>

<version>5.11.0</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>org.mockito</groupId>

<artifactId>mockito-core</artifactId>

<version>5.12.0</version>

<scope>test</scope>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-surefire-plugin</artifactId>

<version>3.0.0-M9</version>

</plugin>

</plugins>

</build>

</project>

package com.example;

public class Calculator {

public int add(int a, int b) {

return a + b;

}

public int subtract(int a, int b) {

return a - b;

}

}

package com.example;

import org.junit.jupiter.api.Test;

import static org.junit.jupiter.api.Assertions.\*;

public class CalculatorTest {

*@Test*

void testAdd() {

Calculator calc = new Calculator();

*assertEquals*(10, calc.add(7, 3));

}

*@Test*

void testSubtract() {

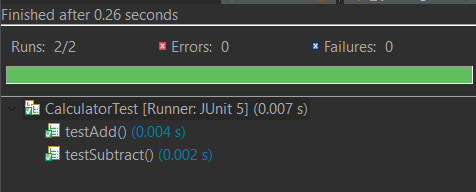
Calculator calc = new Calculator();

*assertEquals*(4, calc.subtract(9, 5));

}

}

**Output:**

****

**Exercise 3: Assertions in Junit  
  
Code:**

package com.example;

import org.junit.jupiter.api.Test;

import static org.junit.jupiter.api.Assertions.\*;

public class AssertionsDemoTest {

*@Test*

void testEquality() {

int expected = 42;

int actual = 40 + 2;

*assertEquals*(expected, actual);

}

*@Test*

void testNotEqual() {

*assertNotEquals*(5, 10);

}

*@Test*

void testTrueCondition() {

*assertTrue*(100 > 10);

}

*@Test*

void testFalseCondition() {

*assertFalse*(3 > 10);

}

*@Test*

void testNullCheck() {

String name = null;

*assertNull*(name);

}

*@Test*

void testNotNullCheck() {

String name = "JUnit";

*assertNotNull*(name);

}

*@Test*

void testException() {

*assertThrows*(ArithmeticException.class, () -> {

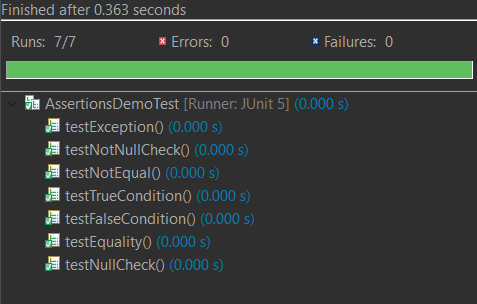
int result = 10 / 0;

});

}

}

**Output:**

****

**Exercise 4: Arrange-Act-Assert (AAA) Pattern, Test Fixtures, Setup and Teardown Methods in Junit**

**Code:**

package com.example;

import org.junit.jupiter.api.\*;

import static org.junit.jupiter.api.Assertions.\*;

*@TestInstance*(*TestInstance*.*Lifecycle*.***PER\_CLASS***)

public class CalculatorTest {

private Calculator calculator;

*@BeforeAll*

void beforeAllTests() {

System.***out***.println("🔧 Before all tests");

}

*@AfterAll*

void afterAllTests() {

System.***out***.println("✅ After all tests");

}

*@BeforeEach*

void setUp() {

calculator = new Calculator(); // Arrange

System.***out***.println("⏳ Setup before test");

}

*@AfterEach*

void tearDown() {

System.***out***.println("🧹 Cleanup after test");

}

*@Test*

void testAdd() {

// Act

int result = calculator.add(5, 3);

// Assert

*assertEquals*(8, result);

}

*@Test*

void testSubtract() {

// Act

int result = calculator.subtract(10, 6);

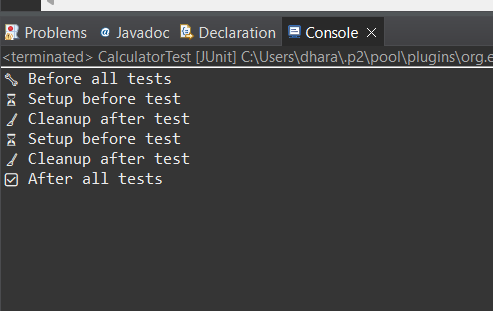
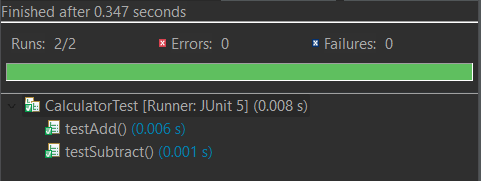
// Assert

*assertEquals*(4, result);

}

}

**Output:**

**  
  
  
Exercise 1: Mocking and Stubbing  
Code:**

package com.example;

public class User {

private int id;

private String name;

public User(int id, String name) {

this.id = id;

this.name = name;

}

public int getId() { return id; }

public String getName() { return name; }

}

package com.example;

public interface UserRepository {

User findUserById(int id);

}

package com.example;

public class UserService {

private final UserRepository userRepository;

public UserService(UserRepository userRepository) {

this.userRepository = userRepository;

}

public String getUserName(int userId) {

User user = userRepository.findUserById(userId);

return user != null ? user.getName() : "Unknown";

}

}

package com.example;

import org.junit.jupiter.api.Test;

import static org.junit.jupiter.api.Assertions.\*;

import static org.mockito.Mockito.\*;

public class UserServiceTest {

*@Test*

void testGetUserName\_returnsCorrectName() {

// Arrange: Mock & stub

UserRepository mockRepo = *mock*(UserRepository.class);

*when*(mockRepo.findUserById(1)).thenReturn(new User(1, "Alice"));

UserService service = new UserService(mockRepo);

// Act

String name = service.getUserName(1);

// Assert

*assertEquals*("Alice", name);

}

*@Test*

void testGetUserName\_returnsUnknownForNull() {

UserRepository mockRepo = *mock*(UserRepository.class);

*when*(mockRepo.findUserById(99)).thenReturn(null);

UserService service = new UserService(mockRepo);

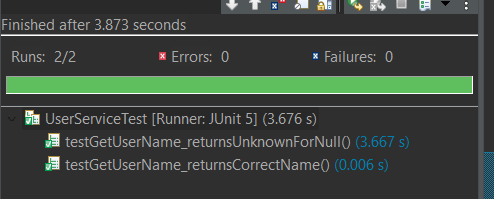
String name = service.getUserName(99);

*assertEquals*("Unknown", name);

}

}

**Output:**

****

**Exercise 2: Verifying Interactions**

**Code:**

package com.example;

public class User {

private int id;

private String name;

public User(int id, String name) {

this.id = id;

this.name = name;

}

public int getId() { return id; }

public String getName() { return name; }

}

package com.example;

public interface UserRepository {

User findUserById(int id);

}

package com.example;

public class UserService {

private final UserRepository userRepository;

public UserService(UserRepository userRepository) {

this.userRepository = userRepository;

}

public String getUserName(int id) {

User user = userRepository.findUserById(id);

return user != null ? user.getName() : "Unknown";

}

}

package com.example;

import org.junit.jupiter.api.Test;

import static org.junit.jupiter.api.Assertions.\*;

import static org.mockito.Mockito.\*;

public class UserServiceTest {

*@Test*

void testInteractionWithRepository() {

// Arrange

UserRepository mockRepo = *mock*(UserRepository.class);

*when*(mockRepo.findUserById(1)).thenReturn(new User(1, "Alice"));

UserService service = new UserService(mockRepo);

// Act

service.getUserName(1);

*verify*(mockRepo).findUserById(1);

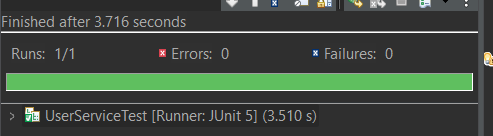
*verify*(mockRepo, *times*(1)).findUserById(1);

*verify*(mockRepo, *never*()).findUserById(2);

}

}

**Output:**

****