**Harshitha M**

**Superset id : 6390499**

**WEEK 2**

**TDD USING JUNIT5 AND MOCKITO**

**1)JUNIT\_BASIC TESTING EXERCISE**

**Exercise 1: Setting Up Junit**

junit-setup/  
├── src/  
│ ├── main/java/  
│ └── test/java/  
└── pom.xml

STEP 2: Add JUnit Dependency to pom.xml

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>4.13.2</version>

<scope>test</scope>

</dependency>

STEP 3: Create Your Main Class

📁 src/main/java/com/example/Calculator.java

package com.example;

public class Calculator {  
public int add(int a, int b) {  
return a + b;  
}  
}

STEP 4: Create Your Test Class

📁 src/test/java/com/example/CalculatorTest.java

package com.example;

import org.junit.Test;

import static org.junit.Assert.assertEquals;

public class CalculatorTest {

@Test

public void testAdd() {

Calculator calculator = new Calculator();

int result = calculator.add(2, 3);

assertEquals(5, result);

}

}

**OUTPUT:**

**A screenshot of a computer

AI-generated content may be incorrect.**

**Exercise 3: Assertions in Junit**

Step 1: Create a Test Class

📂 Location: src/test/java  
📁 Package: com.example  
📄 File: AssertionsTest.java

package com.example;  
import org.junit.Test;  
import static org.junit.Assert.\*;  
public class AssertionsTest {  
 @Test  
 public void testEquals() {  
 *assertEquals*(5, 2 + 3);  
 }  
 @Test  
 public void testTrue() {  
 *assertTrue*(5 > 3);  
 }  
 @Test  
 public void testFalse() { *assertFalse*(5 < 3);  
 }  
 @Test  
 public void testNull() { *assertNull*(null);  
 }  
 @Test  
 public void testNotNull() {  
 *assertNotNull*(new Object());  
 }  
}

**OUTPUT:**

**A screenshot of a computer

AI-generated content may be incorrect.  
Exercise 4: Arrange-Act-Assert (AAA) Pattern, Test Fixtures, Setup and Teardown Methods in Junit**

Step 1: Create Calculator.java in src/main/java

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;

}

}

Step 2: Create CalculatorTest.java in src/test/java

package com.example;

import org.junit.After;

import org.junit.Before;

import org.junit.Test;

import static org.junit.Assert.\*;

public class CalculatorTest {

private Calculator calculator;

@Before

public void setUp() {

System.out.println("🔁 Before - setting up calculator");

calculator = new Calculator(); // Arrange

}

@After

public void tearDown() {

System.out.println("♻️ After - cleaning up");

calculator = null;

}

@Test

public void testAdd() {

int result = calculator.add(10, 5)

assertEquals(15, result);

}

@Test

public void testSubtract() {

int result = calculator.subtract(10, 4);

assertEquals(6, result);

}

}

**OUTPUT:**

**A screenshot of a computer

AI-generated content may be incorrect.**

**2)MOCKITO EXERCISES :**

Exercise 1: Mocking and Stubbing

MockitoDemo

└── src

├── main

│ └── java

└── test

└── java

Open pom.xml (at root of project)

<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>MockitoDemo</artifactId>

<version>1.0-SNAPSHOT</version>

<dependencies>

<!-- JUnit 5 -->

<dependency>

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

<artifactId>junit-jupiter</artifactId>

<version>5.10.2</version>

<scope>test</scope>

</dependency>

<!-- Mockito -->

<dependency>

<groupId>org.mockito</groupId>

<artifactId>mockito-core</artifactId>

<version>5.11.0</version>

<scope>test</scope>

</dependency>

</dependencies>

<build>

<plugins>

<!-- Enable JUnit 5 -->

<plugin>

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

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

<version>3.1.2</version>

</plugin>

</plugins>

</build>

</project>

Right-click src/main/java → New → Package → Name it: com.example

Create two classes:

**ExternalApi.java**

package com.example;

public interface ExternalApi {

String getData();

}

MyService.java

package com.example;

public class MyService {

private ExternalApi api;

public MyService(ExternalApi api) {

this.api = api;

}

public String fetchData() {

return api.getData();

}

}

Right-click src/test/java → New → Package → Name it: com.example

Create: MyServiceTest.java

package com.example;

import org.junit.jupiter.api.Test;

import static org.junit.jupiter.api.Assertions.assertEquals;

import static org.mockito.Mockito.\*;

public class MyServiceTest {

@Test

public void testExternalApi() {

ExternalApi mockApi = mock(ExternalApi.class);

when(mockApi.getData()).thenReturn("Mock Data");

MyService service = new MyService(mockApi);

String result = service.fetchData();

assertEquals("Mock Data", result);

}

@Test

public void testVerifyInteraction() {

ExternalApi mockApi = mock(ExternalApi.class);

MyService service = new MyService(mockApi);

service.fetchData();

verify(mockApi).getData();

}

}

Output:

A screenshot of a computer

AI-generated content may be incorrect.

**Exercise 2: Verifying Interactions**

**Create the Interface** — ExternalApi.java

package com.example;

public interface ExternalApi {

String getData();

}

Step 2: **Create the Service Class** — MyService.java

package com.example;

public class MyService {

private final ExternalApi api;

public MyService(ExternalApi api) {

this.api = api;

}

public String fetchData() {

return api.getData(); //

}

}

Step 3: **Write the Test Case** — MyServiceInteractionTest.java

package com.example;

import org.junit.jupiter.api.Test;

import static org.mockito.Mockito.\*;

public class MyServiceInteractionTest {

@Test

public void testVerifyInteraction() {

ExternalApi mockApi = mock(ExternalApi.class);

MyService service = new MyService(mockApi);

service.fetchData();

verify(mockApi).getData();

System.out.println("✅ Verified: getData() was called exactly once.");

}

}

Step 4: **Ensure Dependencies in pom.xml**

<dependencies>

<dependency>

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

<artifactId>junit-jupiter</artifactId>

<version>5.10.2</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>org.mockito</groupId>

<artifactId>mockito-core</artifactId>

<version>5.11.0</version>

<scope>test</scope>

</dependency>

</dependencies>

**Output:**

****