**Superset ID: 6394725**

**Mockito Hands-On Exercises**

**Exercise 1: Mocking and Stubbing**

Scenario:

You need to test a service that depends on an external API. Use Mockito to mock the external API and stub its methods.

Steps:

1. Create a mock object for the external API.

2. Stub the methods to return predefined values.

3. Write a test case that uses the mock object.

**Solution:**

First edit the pom.xml file by adding the below dependencies:

<dependencies>

<!-- JUnit 5 -->

<dependency>

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

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

<version>5.10.2</version>

<scope>test</scope>

</dependency>

<dependency>

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

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

<version>5.10.2</version>

<scope>test</scope>

</dependency>

<!-- Mockito -->

<dependency>

<groupId>org.mockito</groupId>

<artifactId>mockito-core</artifactId>

<version>5.12.0</version>

<scope>test</scope>

</dependency>

</dependencies>

Steps:

1. Create a mock object for the external API.

package com;

public interface ExternalApi {

String getData();

}

2. Stub the methods to return predefined values.

package com;

public class MyService {

private final ExternalApi api;

public MyService(ExternalApi api) {

this.api = api;

}

public String fetchData() {

return api.getData(); // Calls the mocked method in test

}

}

3. Write a test case that uses the mock object.

package com;

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

import static org.mockito.Mockito.\*;

import org.junit.jupiter.api.Test;

import org.mockito.Mockito;

public class MyServiceTest {

*@Test*

public void testExternalApi() {

// Step 1: Create a mock of ExternalApi

ExternalApi mockApi = Mockito.*mock*(ExternalApi.class);

// Step 2: Stub getData() to return "Mock Data"

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

// Step 3: Use the mock in MyService

MyService service = new MyService(mockApi);

// Act

String result = service.fetchData();

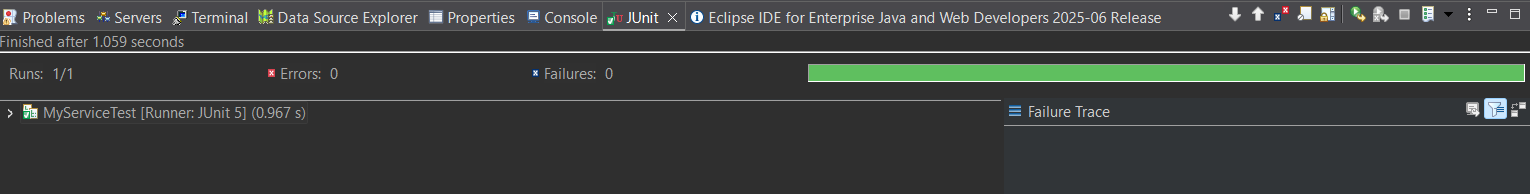
// Assert

*assertEquals*("Mock Data", result);

}

}

Output:



**Exercise 2: Verifying Interactions**

Scenario:

You need to ensure that a method is called with specific arguments.

Steps:

1. Create a mock object.

2. Call the method with specific arguments.

3. Verify the interaction.

**Solution:**

package com;

import static org.mockito.Mockito.\*; // for mock(), verify(), etc.

import org.junit.jupiter.api.Test;

public class MyServiceTest1 {

*@Test*

public void testVerifyInteraction() {

// Step 1: Create a mock object

ExternalApi mockApi = *mock*(ExternalApi.class);

// Step 2: Call the method that uses the mock

MyService service = new MyService(mockApi);

service.fetchData(); // internally calls mockApi.getData()

// Step 3: Verify interaction with the mock

*verify*(mockApi).getData(); // confirms that getData() was called once

}

}

Output:

