**WEEK 2 – MOCKITO - 6428652**

**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 Code:**

**import static org.mockito.Mockito.\*;**

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

**import org.mockito.Mockito;**

**public class MyServiceTest {**

**@Test**

**public void testExternalApi() {**

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

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

**MyService service = new MyService(mockApi);**

**String result = service.fetchData();**

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

**}**

**}**

**Solution:**

package sample2;

public class ExternalApi {

String getData();

}

package sample2;

public class MyService {

private ExternalApi api;

public MyService(ExternalApi api) {

this.api = api;

}

public String fetchData() {

return api.getData();

}

}

package sample2;

import static org.mockito.Mockito.\*;

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

import org.junit.jupiter.api.Test;

public class MyServiceTest {

@Test

public void testExternalApi() {

// Arrange

ExternalApi mockApi = mock(ExternalApi.class);

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

MyService service = new MyService(mockApi);

// Act

String result = service.fetchData();

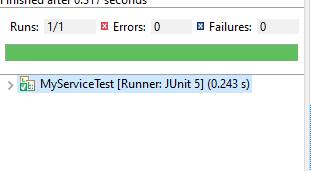
// Assert

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 Code:**

**import static org.mockito.Mockito.\*;**

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

**import org.mockito.Mockito;**

**public class MyServiceTest {**

**@Test**

**public void testVerifyInteraction() {**

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

**MyService service = new MyService(mockApi);**

**service.fetchData();**

**verify(mockApi).getData();**

**}**

**}**

**Solution:**

package mockitotest;

public class ExternalApi {

void getData();

}

package mockitotest;

public class MyService {

private ExternalApi externalApi;

public MyService(ExternalApi externalApi) {

this.externalApi = externalApi;

}

public void fetchData() {

// Some logic, maybe logging, validation, etc.

externalApi.getData(); // This should be called during the test

}

}

package mockitotest;

import org.junit.jupiter.api.Test;

import static org.mockito.Mockito.\*;

public class MyServiceTest {

@Test

public void testVerifyInteraction() {

// Step 1: Create a mock

ExternalApi mockApi = mock(ExternalApi.class);

// Step 2: Inject into service

MyService service = new MyService(mockApi);

// Step 3: Call method

service.fetchData();

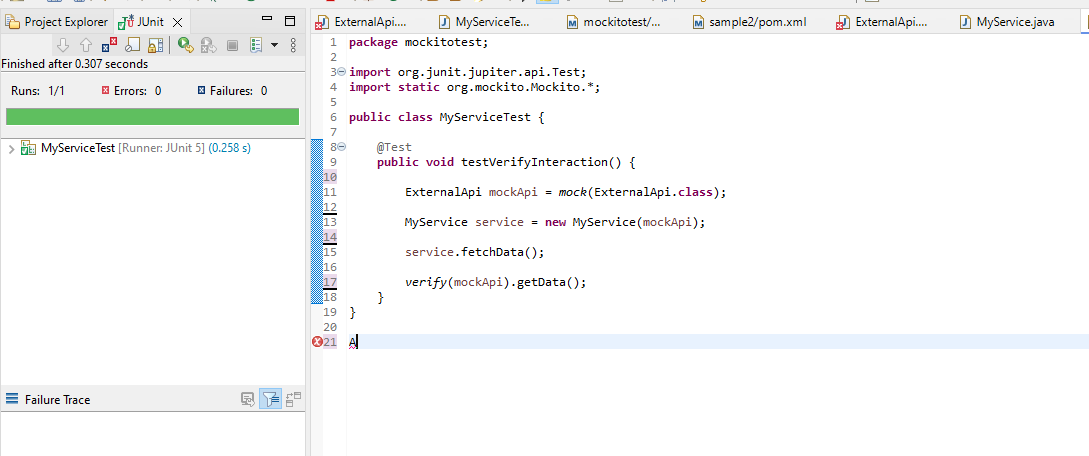
// Step 4: Verify interaction

verify(mockApi).getData(); // ✅ Test will fail if this method is not called

}

}

**Output:**

****