Mockito Hands-On Exercises

**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 :**

**ExternalAPI.java:**

package com.example;

public interface ExternalAPI {

String getData();

}

**MyServicee.java :**

package com.example;

public class MyServicee {

private ExternalApi api;

public MyService(ExternalApi api) {

this.api = api;

}

public String fetchData() {

return api.getData();

}

}

**TestService.java :**

package com.example;

import org.junit.Test;

import static org.junit.Assert.\*;

import static org.mockito.Mockito.\*;

public class TestService {

*@Test*

public void testVerifyInteractionEnhanced() {

// 1. Create mock

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

// 2. Stub the method

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

// 3. Inject mock into service

MyService service = new MyService(mockApi);

// 4. Call method and print result

System.***out***.println(" Calling fetchData()...");

String result = service.fetchData();

System.***out***.println(" Result: " + result);

// 5. Assert the result

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

// 6. Verify it was called once

*verify*(mockApi, *times*(1)).getData();

// 7. Ensure no other methods were called

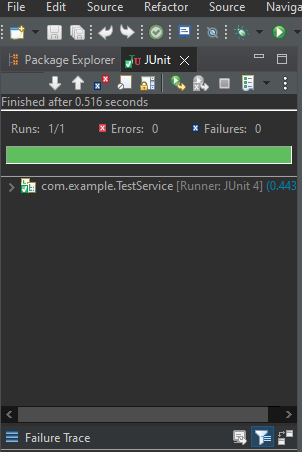
*verifyNoMoreInteractions*(mockApi);

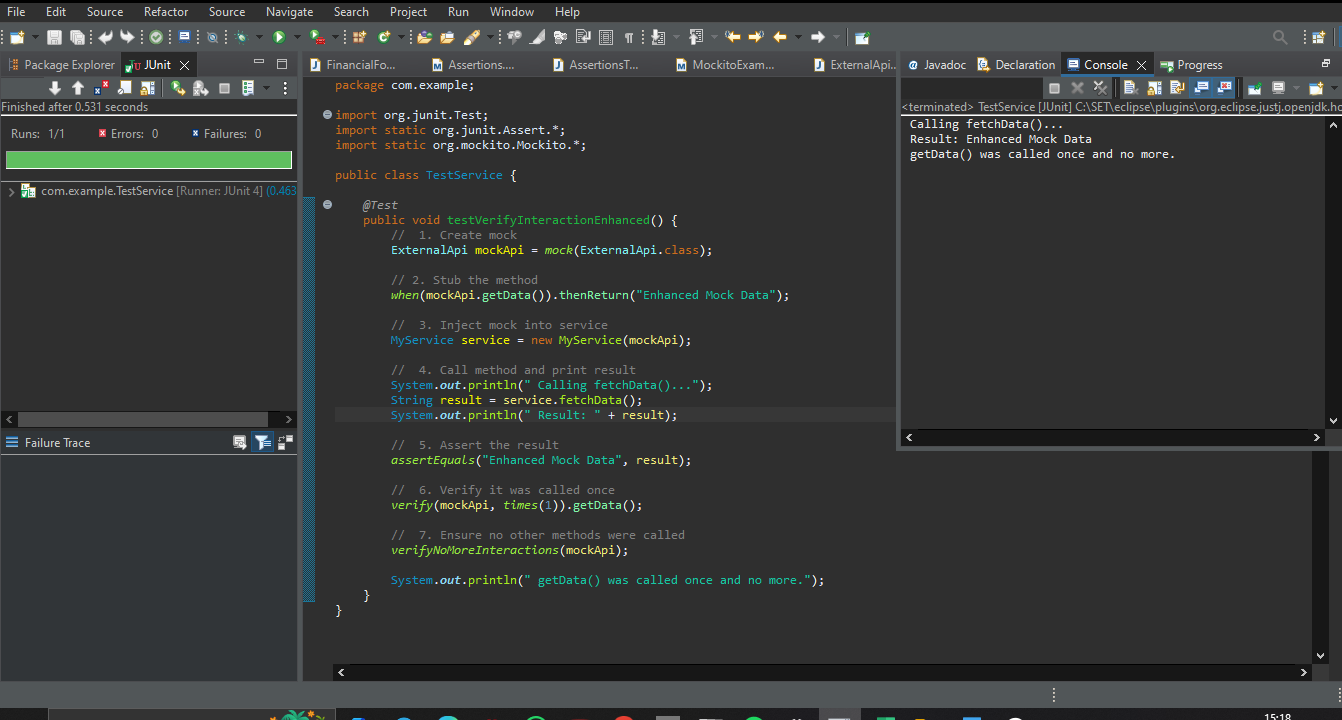
System.***out***.println(" getData() was called once and no more.");

}

}

**OUTPUT :**

****

****