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 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:**

**MyserviceTest.java**

package com.example;

import static org.mockito.Mockito.\*;

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

import org.junit.jupiter.api.Test;

import org.mockito.Mockito;

public class MyServiceTest {

*@Test*

public void testExternalApi() {

// Create mock of ExternalApi

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

// Stub method

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

// Use mocked API in service

MyService service = new MyService(mockApi);

// Test and assert

String result = service.fetchData();

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

}

}

**MyService.java**

package com.example;

public class MyService {

private ExternalApi api;

public MyService(ExternalApi api) {

this.api = api;

}

public String fetchData() {

return api.getData();

}

}

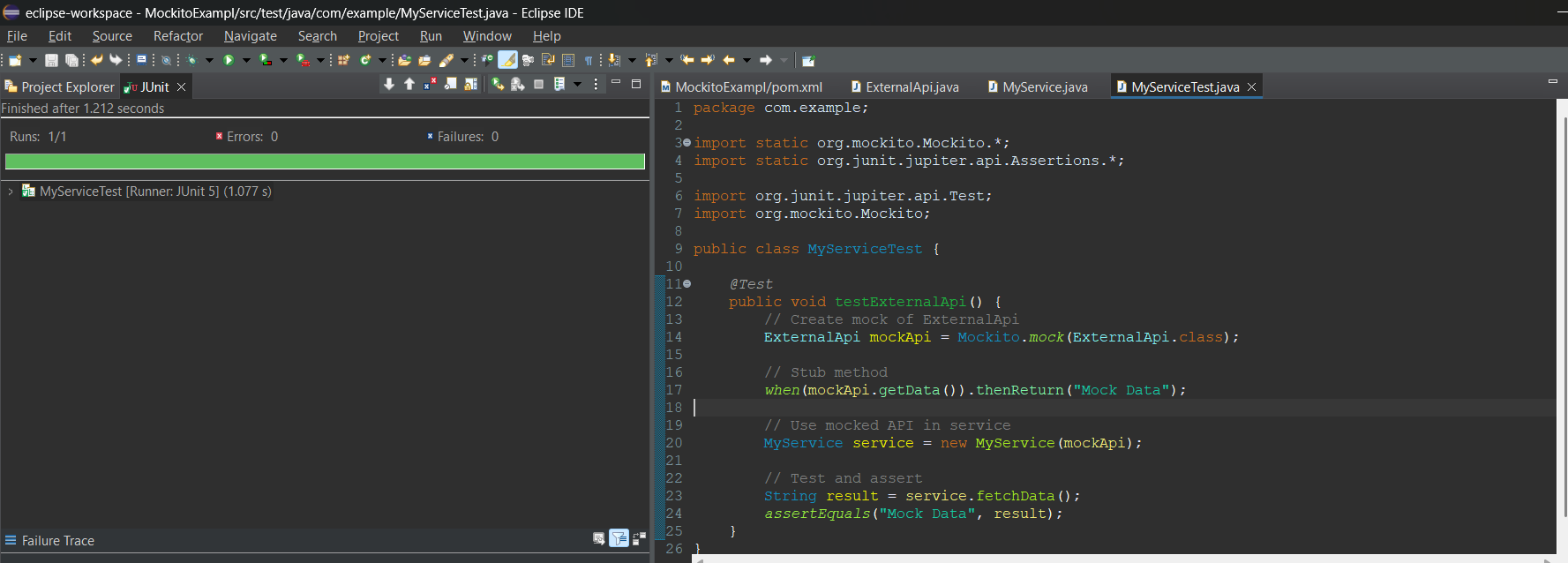
**ExternalAPI.java**

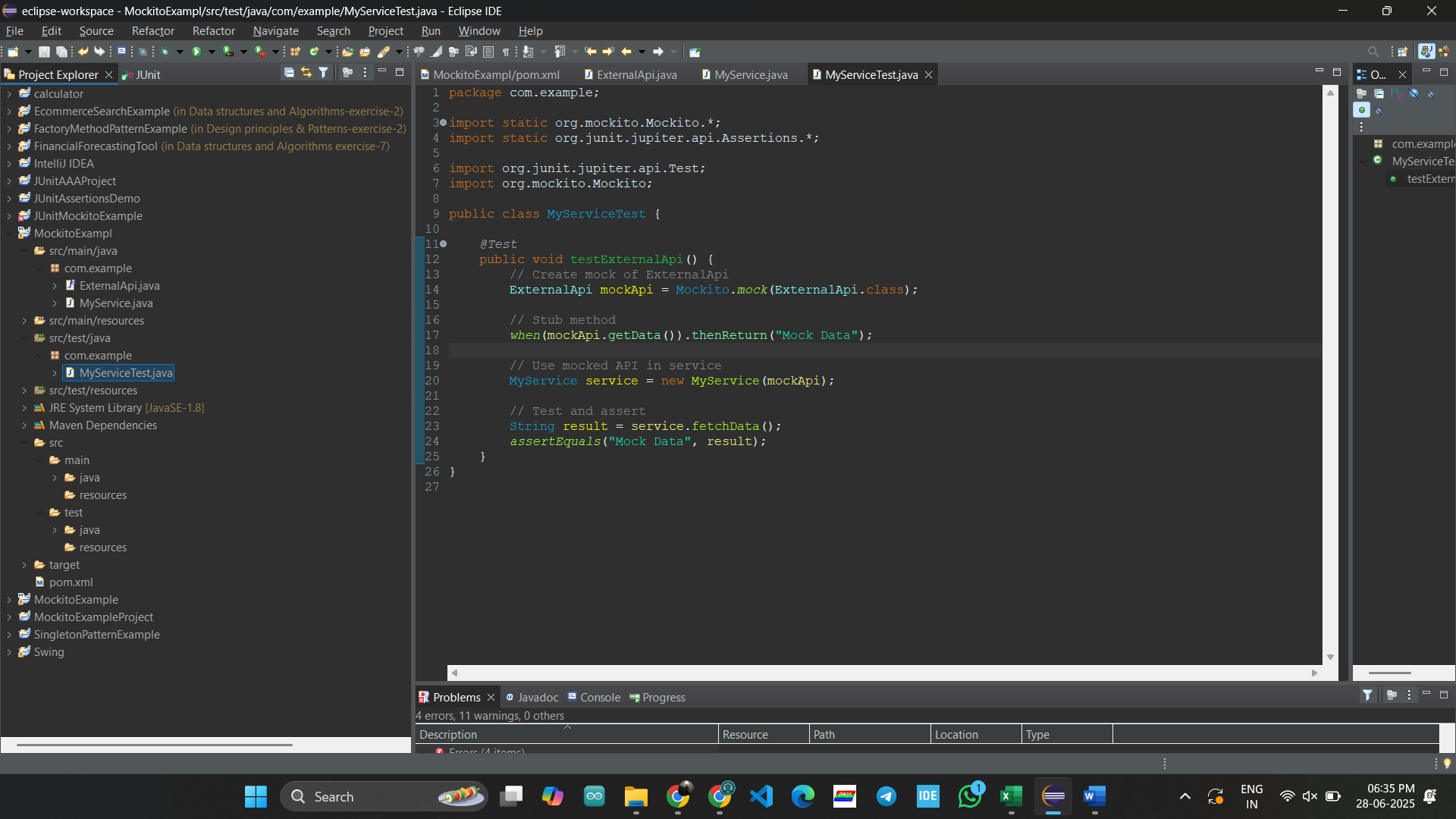
package com.example;

public interface ExternalApi {

String getData();

}





# 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:**

**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(); // This will be verified

}

}

**MyServiceTest.java**

package com.example;

import static org.mockito.Mockito.\*;

import org.junit.jupiter.api.Test;

import org.mockito.Mockito;

public class MyServiceTest {

@Test

public void testVerifyInteraction() {

// Step 1: Create a mock

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

// Step 2: Use the mock in service

MyService service = new MyService(mockApi);

service.fetchData();

// Step 3: Verify that getData() was called

verify(mockApi).getData();

}

}

