**3. Mockito exercises**

**Exercise 1: Mocking and Stubbing using Mockito**

Step 1: Create a mock object for the external API

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

Step 2: Stub the methods to return predefined values

We tell the mock what to return when getData() is called. This is called **stubbing**.

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

Step 3: Write a test case that uses the mock object

package com.example.mock;

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() {

ExternalApi mockApi = mock(ExternalApi.class);

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

MyService service = new MyService(mockApi);

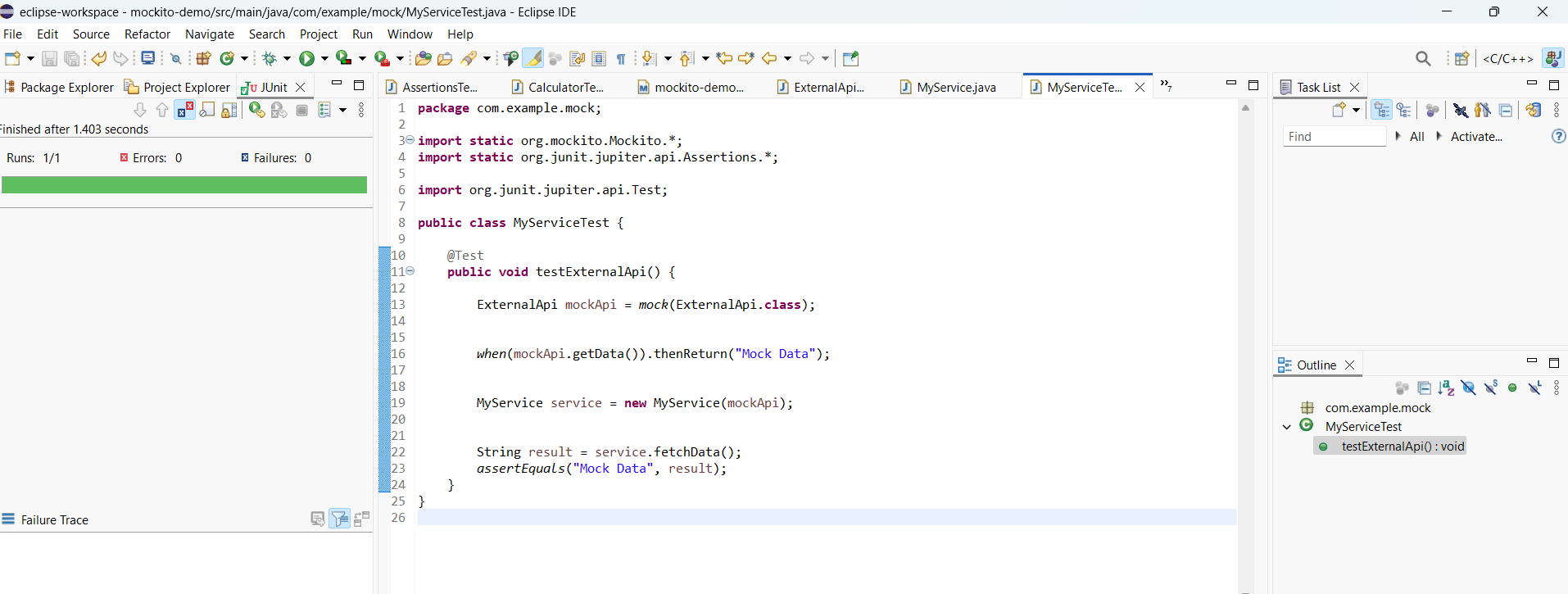
String result = service.fetchData();

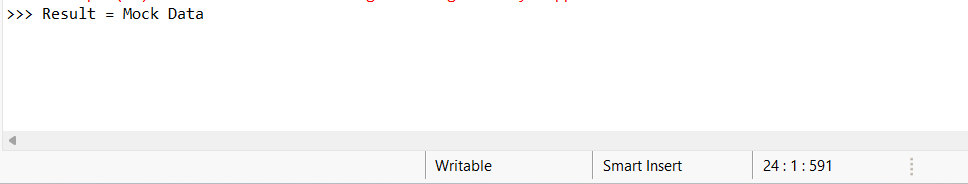
assertEquals("Mock Data", result);

}

}

Output





**Exercise 2: Verifying Interactions Using Mockito**

**Step 1: Create a mock object**

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

**Step 2: Call the method with specific arguments**

MyService service = new MyService(mockApi);

service.fetchData();

**Step 3: Verify the interaction**

package com.example.mock;

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();

}}

Output

