Mockito Hands-On Exercises

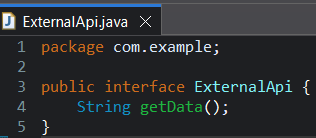
Exercise 1: Mocking and Stubbing

Scenario:

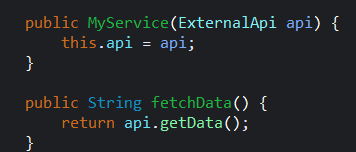
You need to test a service that depends on an external API. Use Mockito to mock the external API and stub its methods.

Steps:

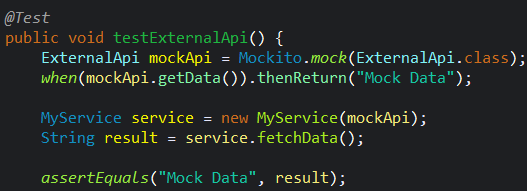
1. Create a mock object for the external API.



1. Stub the methods to return predefined values.



1. 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");

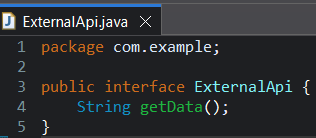
MyService service = new MyService(mockApi); String result = service.fetchData(); assertEquals("Mock Data", result);

}

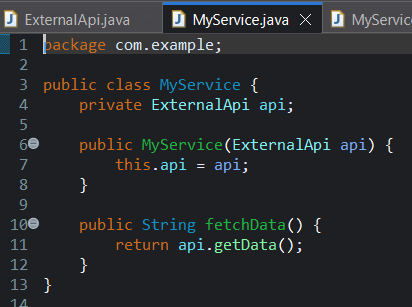
}

**COMPLETE CODE:**

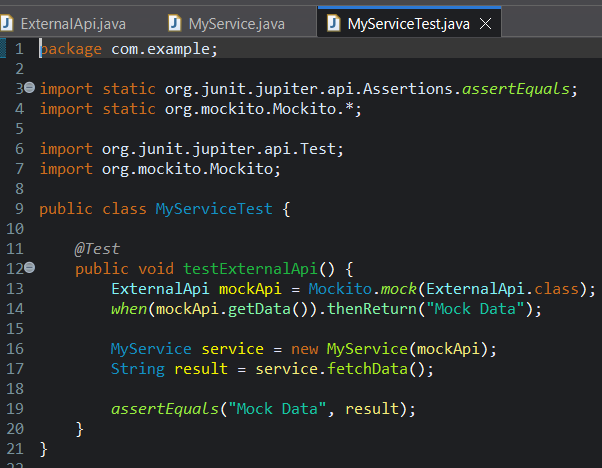
**ExternalApi.java:**



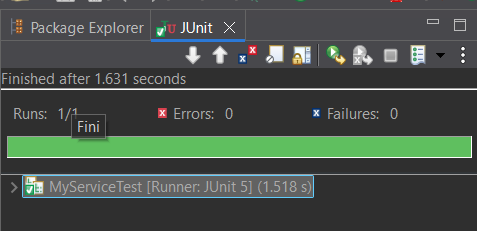
**Myservice.java:**



**MyserviceTest.java:**



**Output:**



Exercise 2: Verifying Interactions

Scenario:

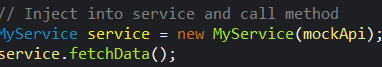
You need to ensure that a method is called with specific arguments.

Steps:

1. Create a mock object.



1. Call the method with specific arguments.



1. 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() {

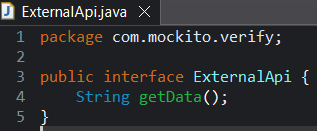
ExternalApi mockApi = Mockito.mock(ExternalApi.class); MyService service = new MyService(mockApi); service.fetchData(); verify(mockApi).getData();

}

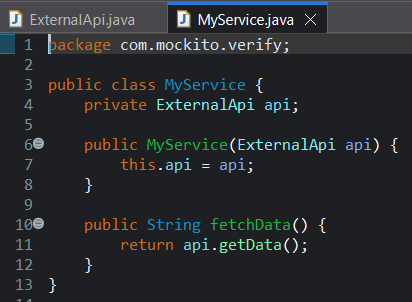
}

**COMPLETE CODE:**

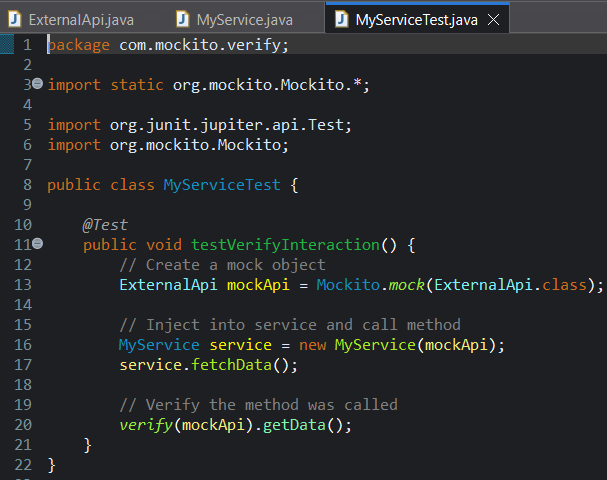
**ExternalApi.java:**



**Myservice.java:**



**MyserviceTest.java:**



**Output:**

