

**Name:- Vidya CS**

**Track: JAVA FSE**

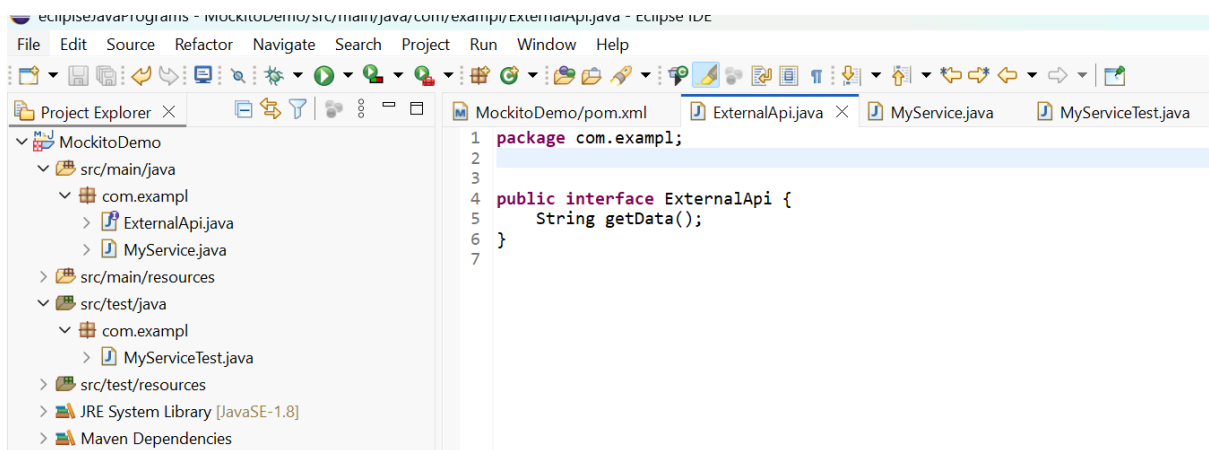
## Mockito Hands-On Exercises

### Exercise 1: Mocking and Stubbing

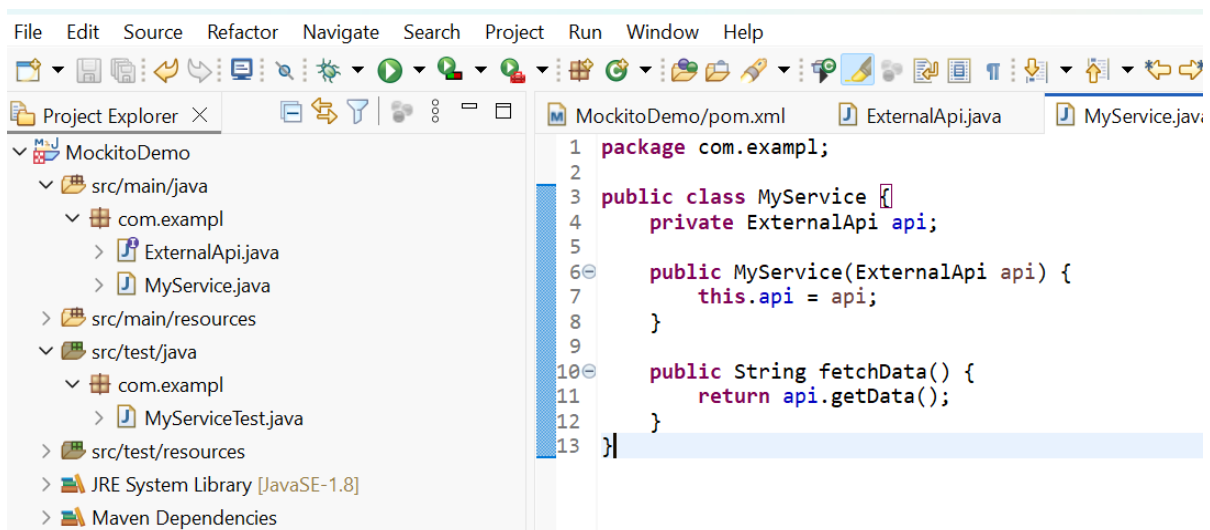
**Scenario:** You need to test a service that depends on an external API. Use Mockito to mock the external API and stub its methods.

**Outcome:**

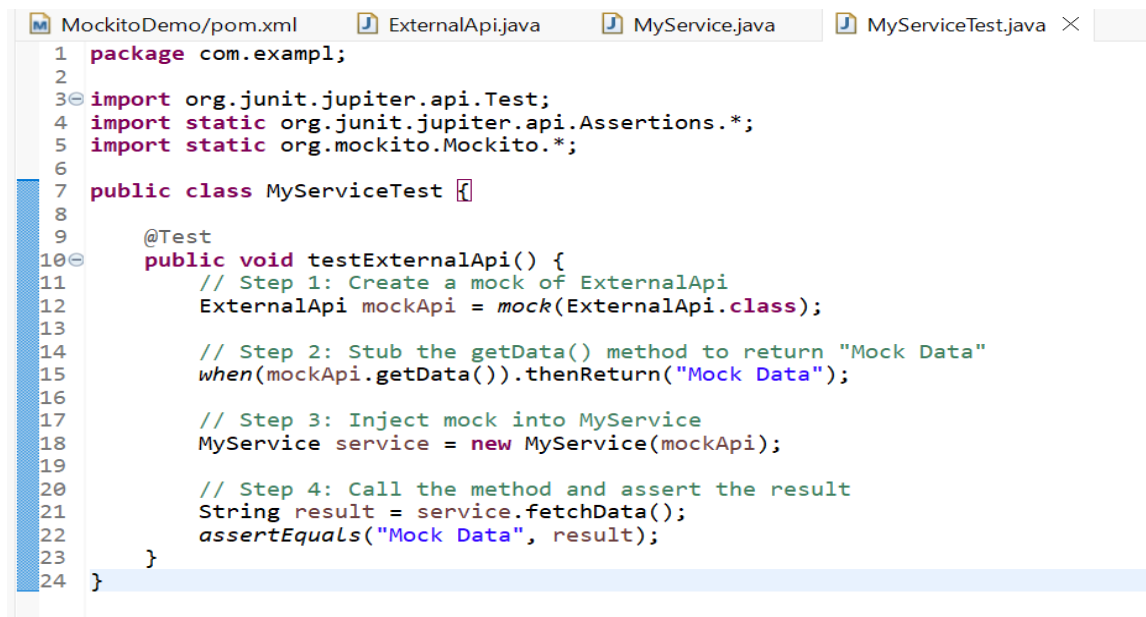
#### Step1: Create a mock object for the external API



#### Step2: Stub the methods to return predefined values

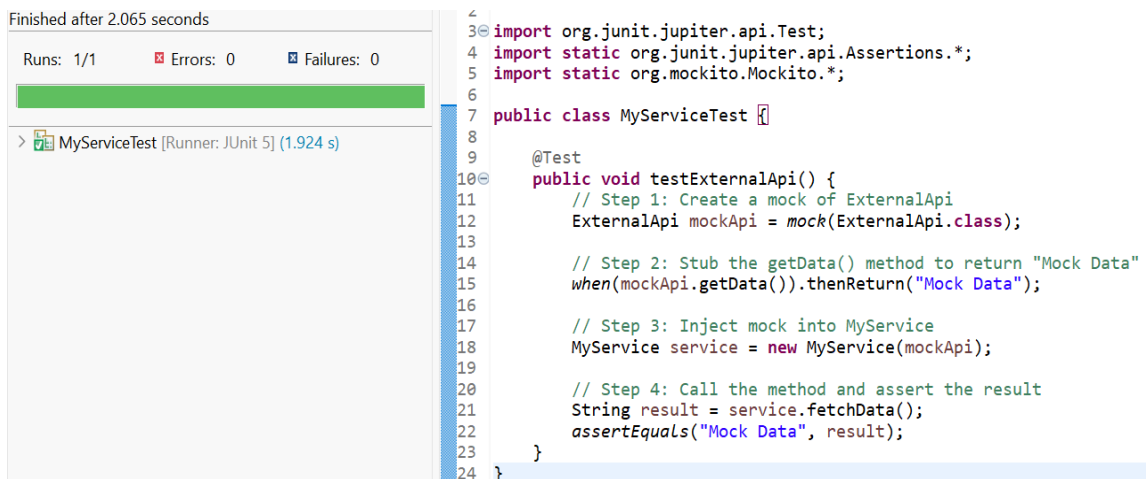


### Step3 :Write a test case that uses the mock object



```
1 package com.exempl;
2
3 import org.junit.jupiter.api.Test;
4 import static org.junit.jupiter.api.Assertions.*;
5 import static org.mockito.Mockito.*;
6
7 public class MyServiceTest {
8
9     @Test
10     public void testExternalApi() {
11         // Step 1: Create a mock of ExternalApi
12         ExternalApi mockApi = mock(ExternalApi.class);
13
14         // Step 2: Stub the getData() method to return "Mock Data"
15         when(mockApi.getData()).thenReturn("Mock Data");
16
17         // Step 3: Inject mock into MyService
18         MyService service = new MyService(mockApi);
19
20         // Step 4: Call the method and assert the result
21         String result = service.fetchData();
22         assertEquals("Mock Data", result);
23     }
24 }
```

### Output :- Test pass



```
4
3 import org.junit.jupiter.api.Test;
4 import static org.junit.jupiter.api.Assertions.*;
5 import static org.mockito.Mockito.*;
6
7 public class MyServiceTest {
8
9     @Test
10     public void testExternalApi() {
11         // Step 1: Create a mock of ExternalApi
12         ExternalApi mockApi = mock(ExternalApi.class);
13
14         // Step 2: Stub the getData() method to return "Mock Data"
15         when(mockApi.getData()).thenReturn("Mock Data");
16
17         // Step 3: Inject mock into MyService
18         MyService service = new MyService(mockApi);
19
20         // Step 4: Call the method and assert the result
21         String result = service.fetchData();
22         assertEquals("Mock Data", result);
23     }
24 }
```

## Exercise 2: Verifying Interactions

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

1. Create a mock object.

**Code :-** `ExternalApi mockApi = Mockito.mock(ExternalApi.class);`

2. Call the method with specific arguments.

**Code:-** `MyService service = new MyService(mockApi);`  
`service.fetchData();`

3. Verify the interaction.

**Code:-** `verify(mockApi).getData();`

**Output:- Test successful**

