Mockito_Hands_On_Exercises

Exercise 1: Mocking and Stubbing with Mockito

Scenario:

You want to test a service (MyService) that depends on an external API (ExternalApi) by using **Mockito** to:

- Mock the API,
- Stub the method getData(),
- Verify the result in the service.

Step 1: Add Mockito + JUnit 5 to pom.xml

```
<!-- JUnit 5 -->
<dependency>
<groupId>org.junit.jupiter</groupId>
<artifactId>junit-jupiter</artifactId>
<version>5.9.3</version>
<scope>test</scope>
</dependency>

<!-- Mockito -->
<dependency>
<groupId>org.mockito</groupId>
<artifactId>mockito-core</artifactId>
<version>5.7.0</version>
<scope>test</scope>
</dependency>
</dependencies>
```

Step 2: Create the Interface (External API)

src/main/java/com/example/ExternalApi.java

```java
package com.example;

public interface ExternalApi {
    String getData();
}

```

Step 3: Create the Service That Uses the API

src/main/java/com/example/MyService.java

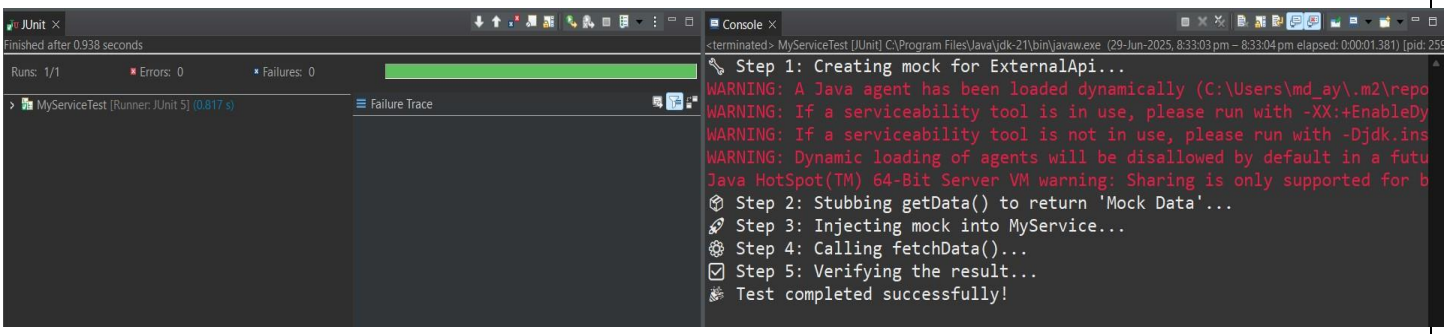
```java
package com.example;

public class MyService {
    private ExternalApi externalApi;

    public MyService(ExternalApi externalApi) {
        this.externalApi = externalApi;
    }

    public String fetchData() {
        return externalApi.getData();
    }
}

```

Step 4: Create the Test Using Mockito

src/test/java/com/example/MyServiceTest.java

```java
package com.example;

import org.junit.jupiter.api.Test;
import static org.junit.jupiter.api.Assertions.*;
import static org.mockito.Mockito.*;

public class MyServiceTest {

    @Test
    public void testExternalApi() {
        System.out.println("🔧 Step 1: Creating mock for ExternalApi...");
        ExternalApi mockApi = mock(ExternalApi.class);

        System.out.println("📦 Step 2: Stubbing getData() to return 'Mock Data'...");
        when(mockApi.getData()).thenReturn("Mock Data");

        System.out.println("🚀 Step 3: Injecting mock into MyService...");
        MyService service = new MyService(mockApi);

        System.out.println("⚙️ Step 4: Calling fetchData()...");
        String result = service.fetchData();

        System.out.println("☑️ Step 5: Verifying the result...");
        assertEquals("Mock Data", result);

        System.out.println("🎉 Test completed successfully!");
    }
}
```

Expected Output



Exercise 2: Verifying Interactions

Scenario:

ExternalApi1.java

```java
package com.example;

public interface ExternalApi1 {
    String getData();
}

```

MyService.java

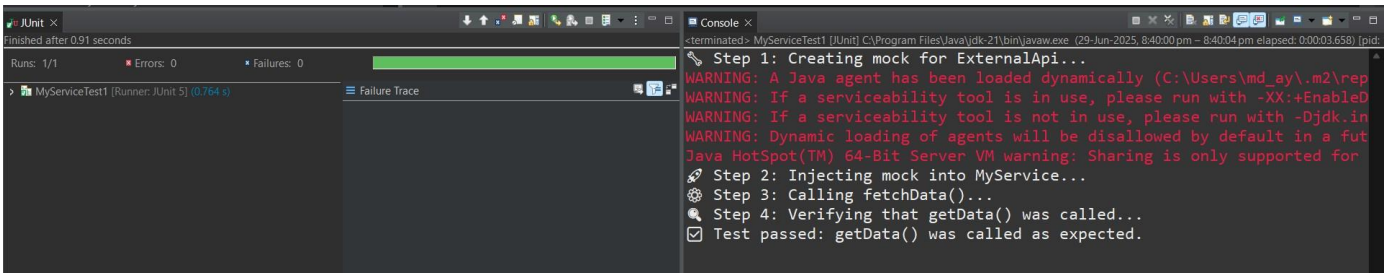
```java
package com.example;

public class MyService1 {
    private ExternalApi externalApi;

    public MyService1(ExternalApi externalApi) {
        this.externalApi = externalApi;
    }
    public String fetchData() {
        return externalApi.getData();
    }

}

```

Updated Test Code with Verification + SOPs

MyServiceTest.java

```java
package com.example;

import org.junit.jupiter.api.Test;
import static org.mockito.Mockito.*;
import static org.junit.jupiter.api.Assertions.*;

public class MyServiceTest1 {

    @Test
    public void testVerifyInteraction() {
        System.out.println("🔧 Step 1: Creating mock for ExternalApi...");
        ExternalApi mockApi = mock(ExternalApi.class);

        System.out.println("🚀 Step 2: Injecting mock into MyService...");
        MyService service = new MyService(mockApi);

        System.out.println("⚙️ Step 3: Calling fetchData()...");
        service.fetchData();

        System.out.println("🔍 Step 4: Verifying that getData() was called...");
        verify(mockApi).getData();

        System.out.println("☑️ Test passed: getData() was called as expected.");
    }
}
```

Expected Output



KANMANI MURUGHAIYAN
Superset id: 6407636