

Cognizant - DN 4.0 Deep Skilling Java FSE

Week 02 – TDD using JUnit5 and Mockito

Superset ID: 6386074

Name: A Sri Pranav

Exercise 1: Setting Up JUnit

```
//CalculatorTest
package testcases;

import org.junit.Test;
import static org.junit.Assert.*;

public class CalculatorTest {

    @Test

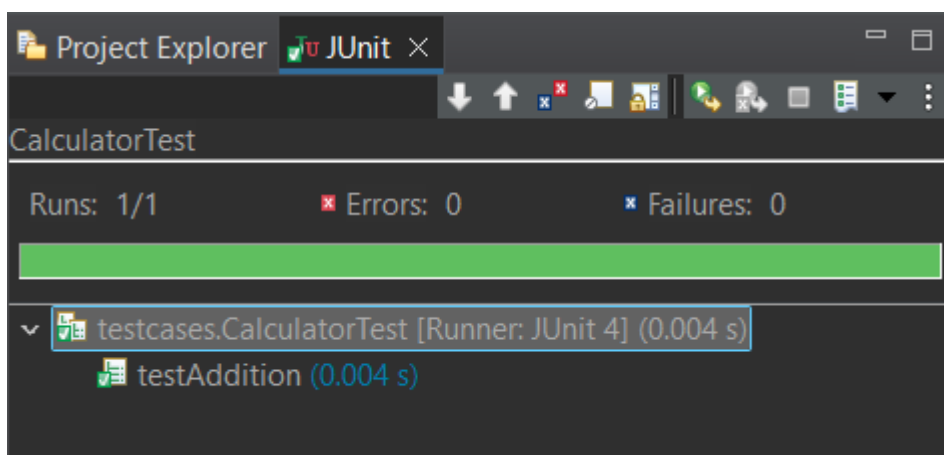
    public void testAddition() {

        assertEquals(5, 2 + 3);

    }

}
```

Output:



Exercise 2: Assertions in JUnit

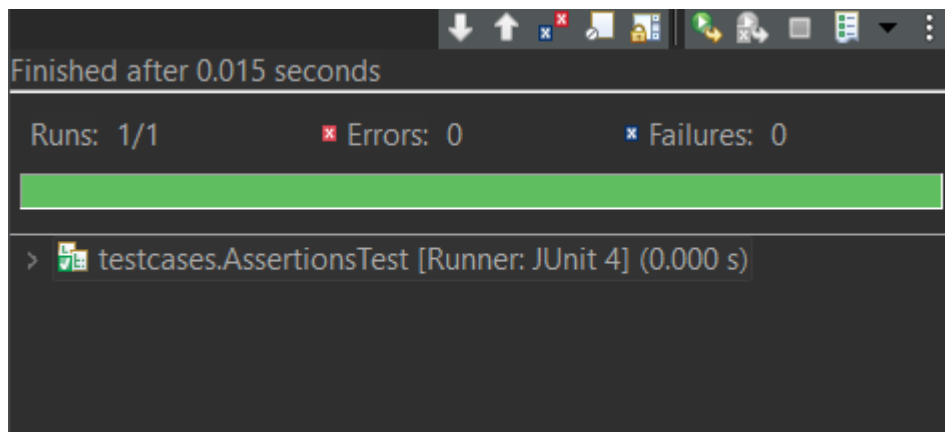
```
//AssertionTest.java

package testcases;

import static org.junit.Assert.*;
import org.junit.Test;

public class AssertionsTest {
    @Test
    public void testAssertions() {
        assertEquals(5, 2 + 3);
        assertTrue(5 > 3);
        assertFalse(5 < 3);
        assertNull(null);
        assertNotNull(new Object());
    }
}
```

Output:



Exercise 3: Arrange-Act-Assert (AAA) Pattern, Test Fixtures, Setup and Teardown Methods in JUnit

```
//Calculator.java

package testcases;

public class Calculator {

    public int add(int a, int b) {
        return a + b;
    }
}
```

```
    public int subtract(int a, int b) {  
        return a - b;  
    }  
}
```

```
//CalculatorTest.java  
package testcases;  
import static org.junit.Assert.*;  
import org.junit.Before;  
import org.junit.After;  
import org.junit.Test;  
public class CalculatorTest {  
    private Calculator calculator;  
    @Before  
    public void setUp() {  
        calculator = new Calculator();  
        System.out.println("Setup complete");  
    }  
    @After  
    public void tearDown() {  
        calculator = null;  
        System.out.println("Teardown complete");  
    }  
    @Test  
    public void testAddition() {  
        // Arrange  
        int a = 5;  
        int b = 3;  
        // Act  
        int result = calculator.add(a, b);
```

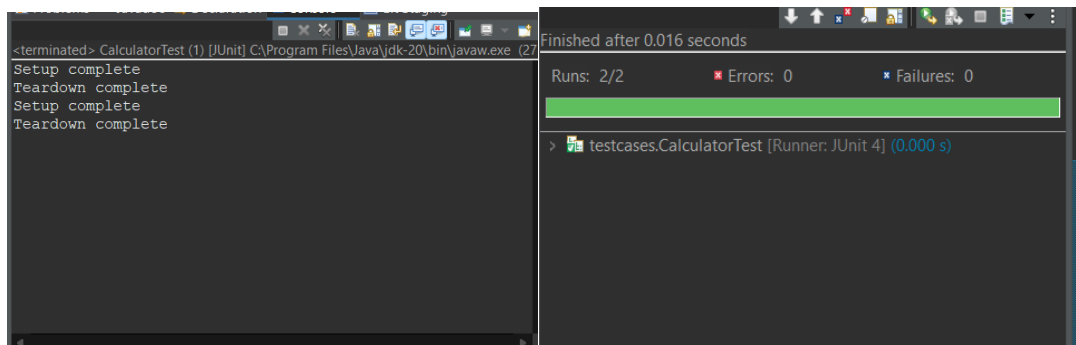
```

    // Assert
    assertEquals(8, result);
}

@Test
public void testSubtraction() {
    // Arrange
    int a = 10;
    int b = 4;
    // Act
    int result = calculator.subtract(a, b);
    // Assert
    assertEquals(6, result);
}
}

```

Output:



Exercise 4: Mocking and Stubbing

```

//pom.xml
<dependencies>
    <!-- JUnit 4 -->
    <dependency>
        <groupId>junit</groupId>
        <artifactId>junit</artifactId>
    </dependency>
</dependencies>

```

```
<version>4.13.2</version>
<scope>test</scope>
</dependency>
<!-- Mockito -->
<dependency>
  <groupId>org.mockito</groupId>
  <artifactId>mockito-core</artifactId>
  <version>4.11.0</version>
  <scope>test</scope>
</dependency>
</dependencies>
```

```
//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();
    }
}
```

```
//MyServiceTest.java
package com.example;

import org.junit.Test;
import static org.junit.Assert.*;
import static org.mockito.Mockito.*;
import org.mockito.Mockito;

public class MyServiceTest {

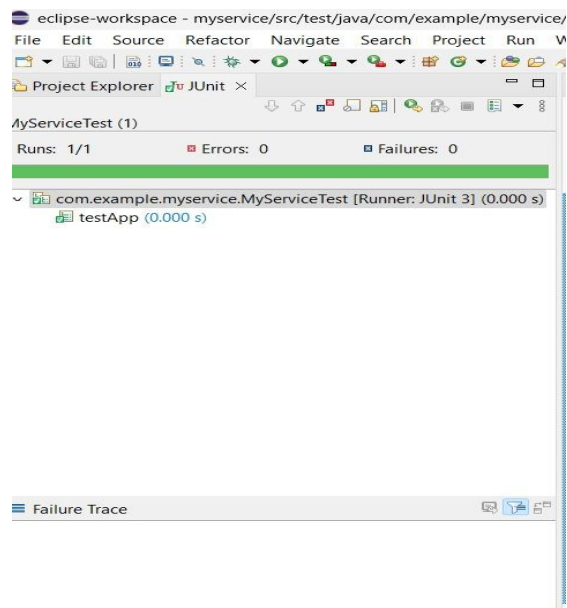
    @Test
    public void testExternalApi() {

        // Arrange
        ExternalApi mockApi = Mockito.mock(ExternalApi.class);
        when(mockApi.getData()).thenReturn("Mock Data");
        MyService service = new MyService(mockApi);

        // Act
        String result = service.fetchData();

        // Assert
        assertEquals("Mock Data", result);
    }
}
```

Output:



Exercise 5: Verifying Interactions

//MyService.java

```
package com.example;

public class MyService {
    private ExternalApi api;

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

    public String fetchData() {
        return api.getData();
    }
}
```

//MyServiceTest.java

```
package com.example;

import org.junit.Test;
import static org.mockito.Mockito.*;
import org.mockito.Mockito;

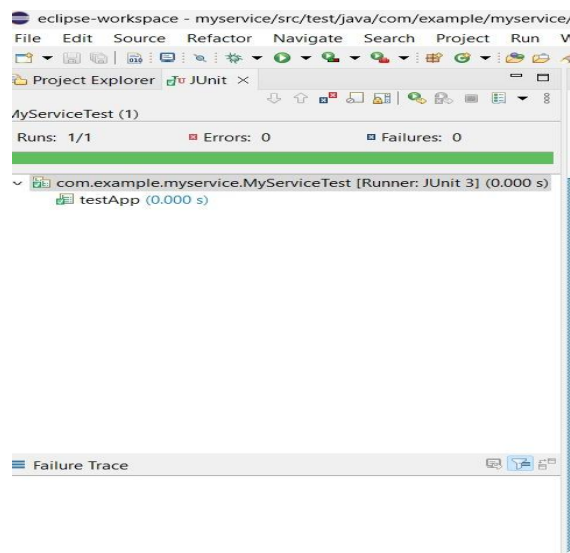
public class MyServiceTest {
```

```

@Test
public void testVerifyInteraction() {
    // Arrange
    ExternalApi mockApi = Mockito.mock(ExternalApi.class);
    MyService service = new MyService(mockApi);
    // Act
    service.fetchData();
    // Assert
    verify(mockApi).getData();
}

```

Output:



Exercise 6: Logging Error Messages and Warning Levels

```

//pom.xml
<project xmlns="http://maven.apache.org/POM/4.0.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
    https://maven.apache.org/xsd/maven-4.0.0.xsd>

  <modelVersion>4.0.0</modelVersion>

  <groupId>com.example</groupId>

  <artifactId>LoggingDemo</artifactId>

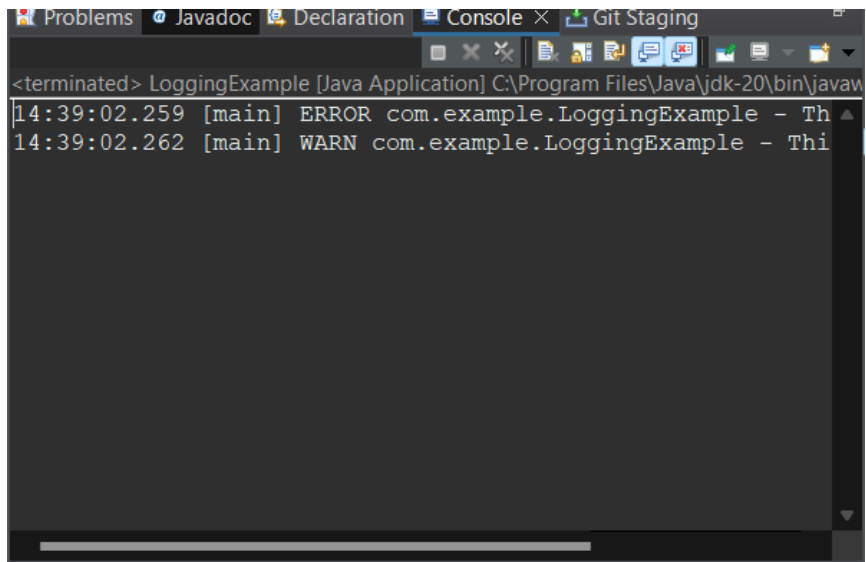
```



```
<version>0.0.1-SNAPSHOT</version>
<dependencies>
  <dependency>
    <groupId>org.slf4j</groupId>
    <artifactId>slf4j-api</artifactId>
    <version>1.7.30</version>
  </dependency>
  <dependency>
    <groupId>ch.qos.logback</groupId>
    <artifactId>logback-classic</artifactId>
    <version>1.2.3</version>
  </dependency>
</dependencies>
</project>
```

```
//LoggingExample.java
package com.example;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
public class LoggingExample {
    private static final Logger logger = LoggerFactory.getLogger(LoggingExample.class);
    public static void main(String[] args) {
        logger.error("This is an error message");
        logger.warn("This is a warning message");
    }
}
```

Output:



The screenshot shows an IDE's console window with the following tabs: Problems, Javadoc, Declaration, Console (active), and Git Staging. The console output is as follows:

```
<terminated> LoggingExample [Java Application] C:\Program Files\Java\jdk-20\bin\javaw
14:39:02.259 [main] ERROR com.example.LoggingExample - Th
14:39:02.262 [main] WARN com.example.LoggingExample - Thi
```