JUnit Basic Testing Exercises

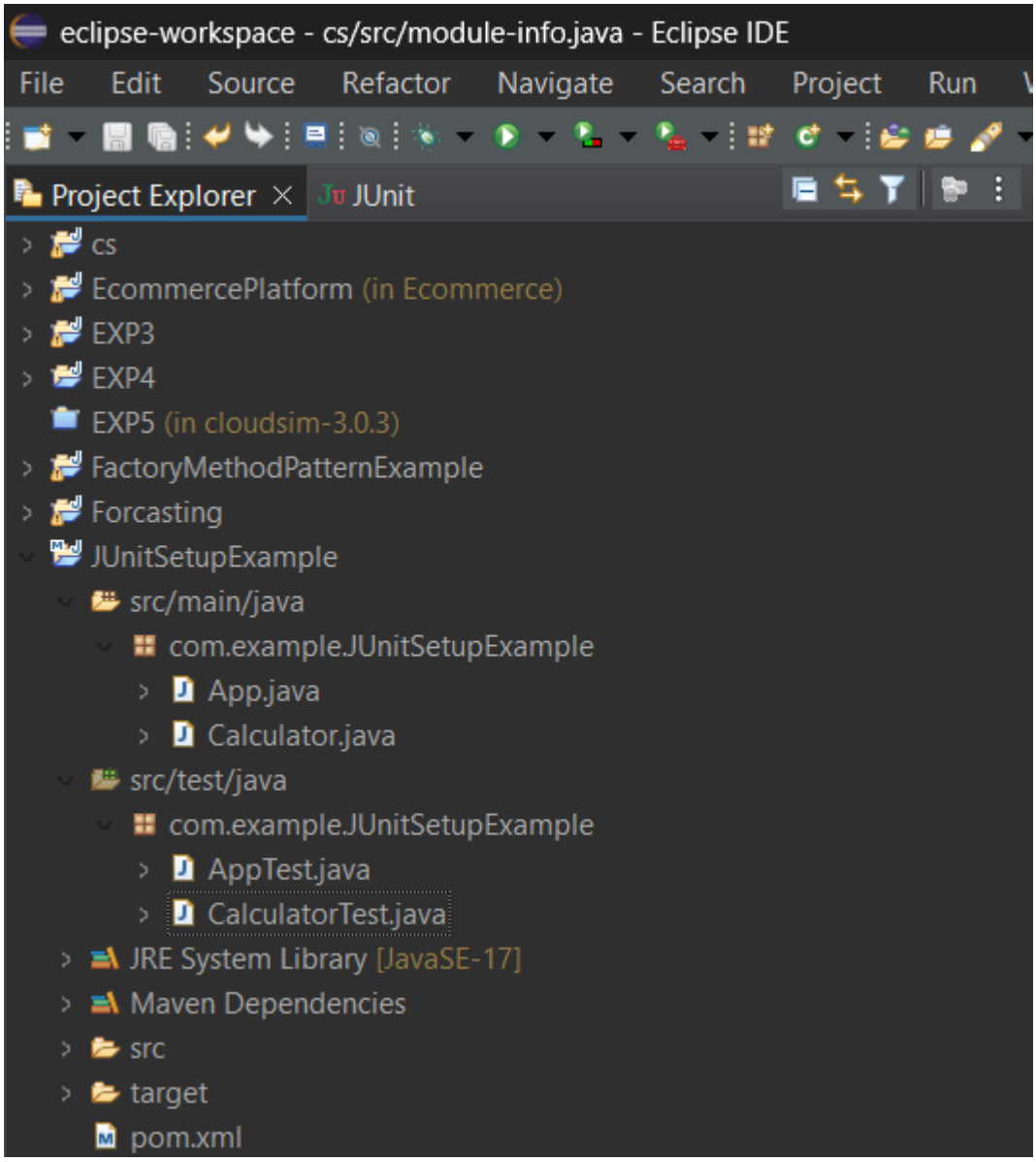
Exercise 1: Setting Up JUnit

Scenario:

You need to set up JUnit in your Java project to start writing unit tests.

Steps:

1. Create a new Java project in your IDE (e.g., IntelliJ IDEA, Eclipse).



2. Add JUnit dependency to your project. If you are using Maven, add the following to your pom.xml:

```
<dependency>
  <groupId>junit</groupId>
  <artifactId>junit</artifactId>
  <version>4.13.2</version>
  <scope>test</scope>
</dependency>
```

Pom.xml:

```
https://maven.apache.org/xsd/maven-4.0.0.xsd (xsi:schemaLocation)
<?xml version="1.0" encoding="UTF-8"?>
<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.
    <modelVersion>4.0.0</modelVersion>

    <groupId>com.example</groupId>
    <artifactId>JUnitSetupExample</artifactId>
    <version>0.0.1-SNAPSHOT</version>

    <properties>
        <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
        <maven.compiler.release>17</maven.compiler.release>
    </properties>

    <dependencies>
        <!-- ☑ JUnit 4 API -->
        <dependency>
            <groupId>junit</groupId>
            <artifactId>junit</artifactId>
            <version>4.13.2</version>
            <scope>test</scope>
        </dependency>
```

3. Create a new test class in your project.

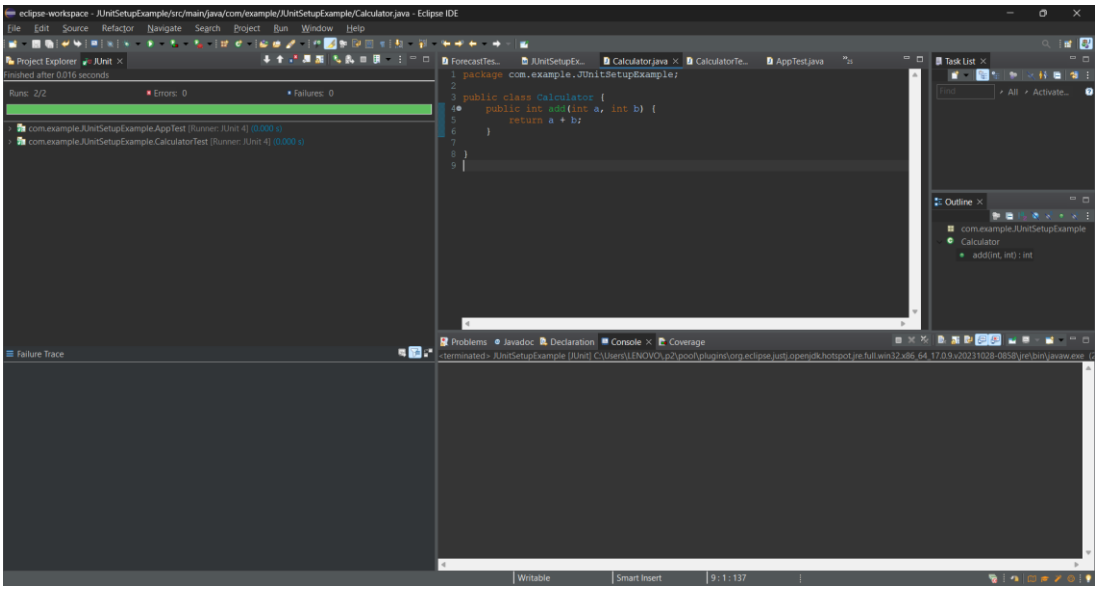
Class: Calculator

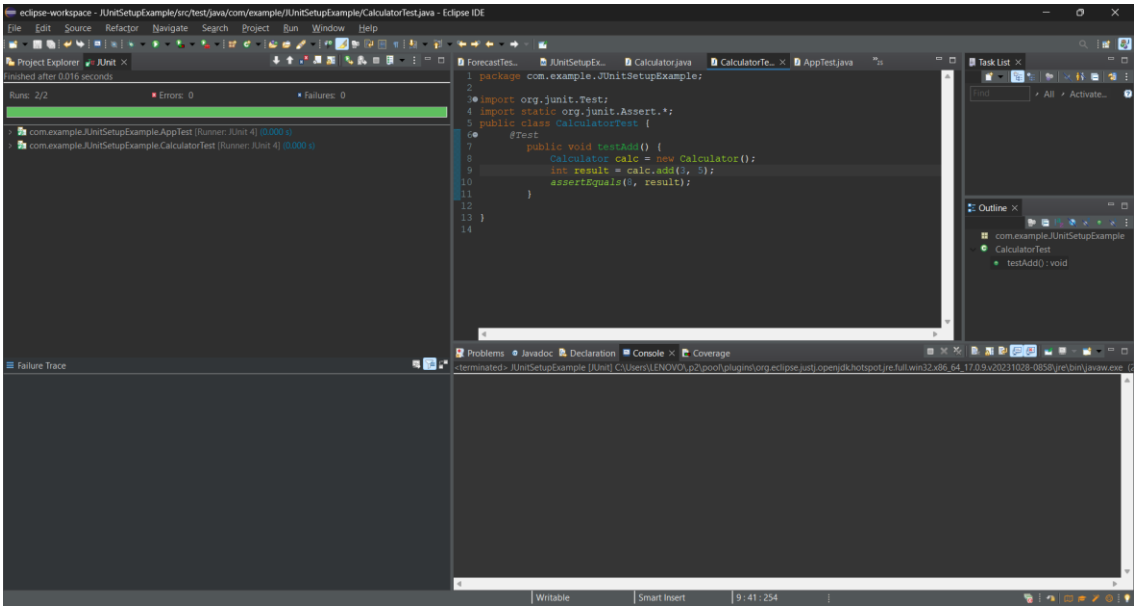
```java
package com.example.JUnitSetupExample;

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

}
```

Test Class: CalculatorTest

```java
package com.example.JUnitSetupExample;

import org.junit.Test;
import static org.junit.Assert.*;
public class CalculatorTest {
    @Test
        public void testAdd() {
            Calculator calc = new Calculator();
            int result = calc.add(3, 5);
            assertEquals(8, result);
        }

}
```

Output Screenshots:





Exercise 3: Assertions in JUnit

Scenario:

You need to use different assertions in JUnit to validate your test results.

Steps:

1.Write tests using various JUnit assertions.

Code:

```
public class AssertionsTest {
   @Test
   public void testAssertions() {
      // Assert equals
      assertEquals(5, 2 + 3);

      // Assert true
      assertTrue(5 > 3);

      // Assert false
      assertFalse(5 < 3);

      // Assert null
      assertNull(null);

      // Assert not null
      assertNotNull(new Object());
   }

}
```

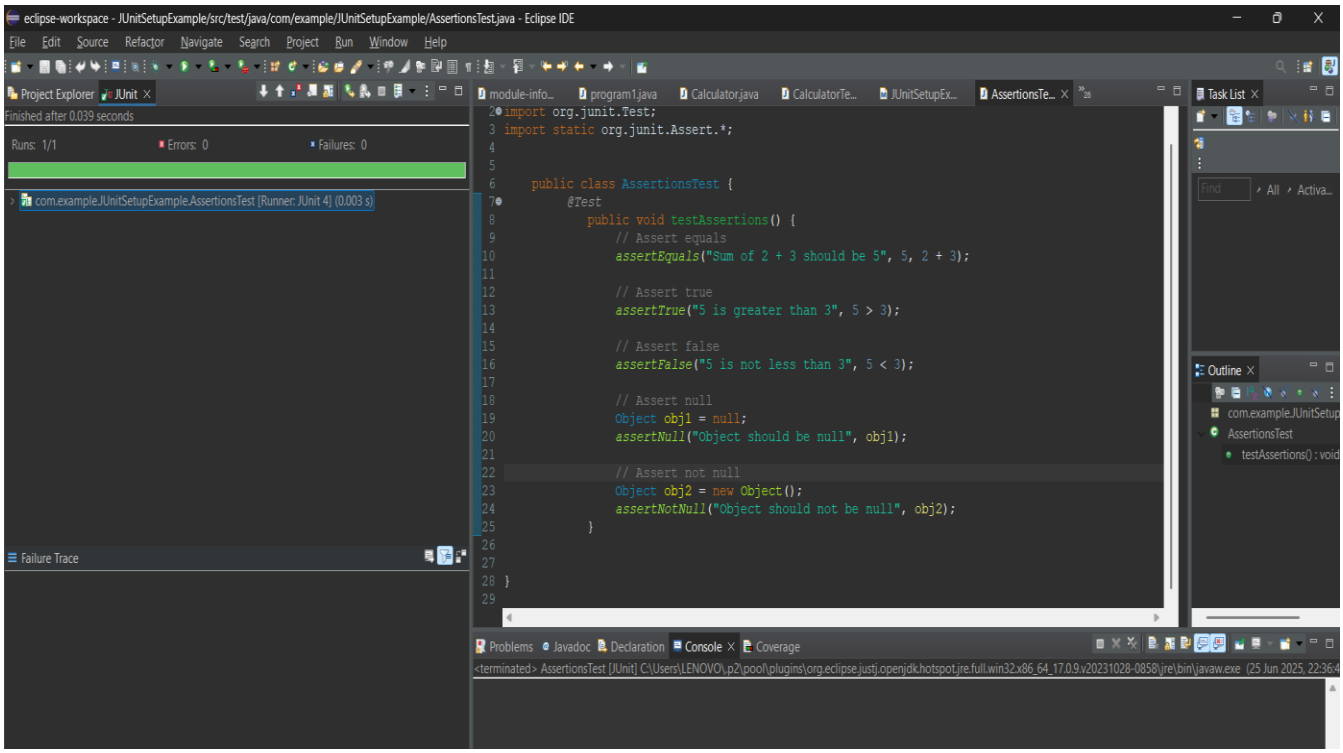
Test class: AssertionsTest

```java
package com.example.JUnitSetupExample;
import org.junit.Test;
import static org.junit.Assert.*;


    public class AssertionsTest {
         @Test
            public void testAssertions() {
                // Assert equals
                assertEquals("Sum of 2 + 3 should be 5", 5, 2 + 3);

                // Assert true
                assertTrue("5 is greater than 3", 5 > 3);

                // Assert false
                assertFalse("5 is not less than 3", 5 < 3);

                // Assert null
                Object obj1 = null;
                assertNull("Object should be null", obj1);

                // Assert not null
                Object obj2 = new Object();
                assertNotNull("Object should not be null", obj2);
            }


}
```

2.Test the code by run the class as :

Run as> JUnit Test

Output Screenshots:



Exercise 4: Arrange-Act-Assert (AAA) Pattern, Test Fixtures, Setup and Teardown Methods in JUnit

Scenario:

You need to organize your tests using the Arrange-Act-Assert (AAA) pattern and use setup and teardown methods.

Steps:

1. Write tests using the AAA pattern.

2. Use @Before and @After annotations for setup and teardown methods.

3. Run as JUnit Test to get the result for the class.

Calculator Class:

```
package com.example.JUnitSetupExample;

public class Calculator {
	public int add(int a, int b) {
		return a + b;
	    }
	    public int subtract(int a, int b) {
		return a - b;
	    }

}
```

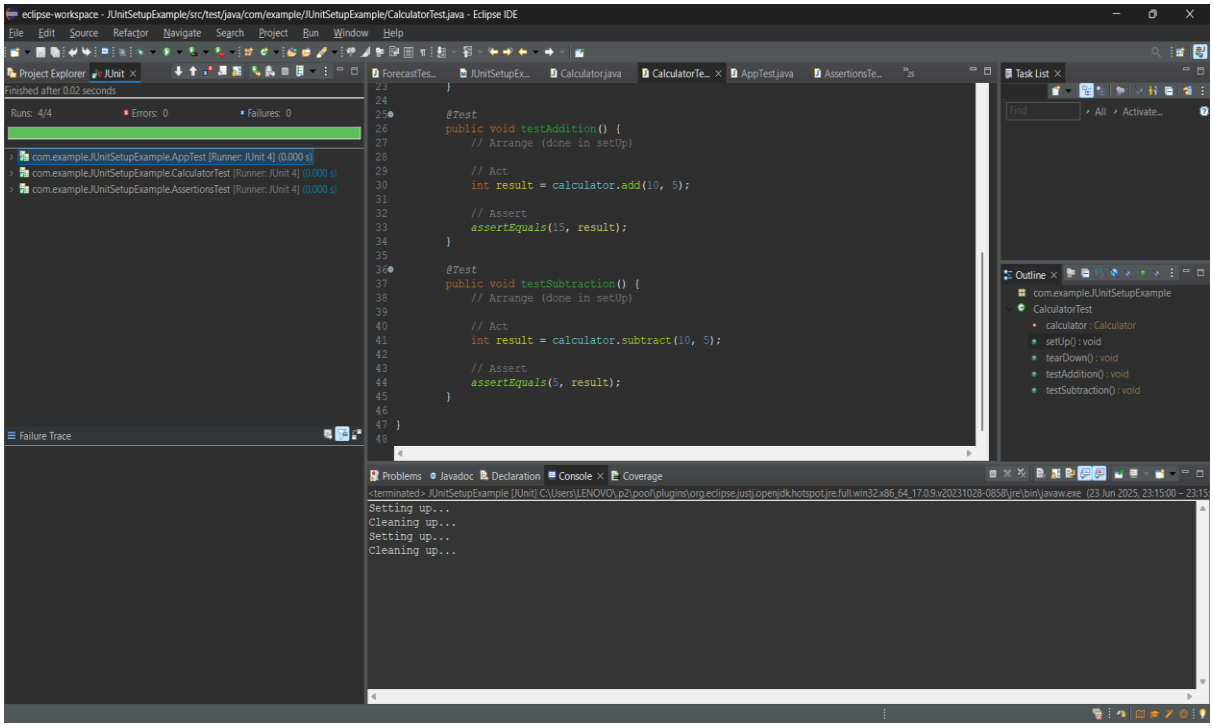
CalculatorTest class:

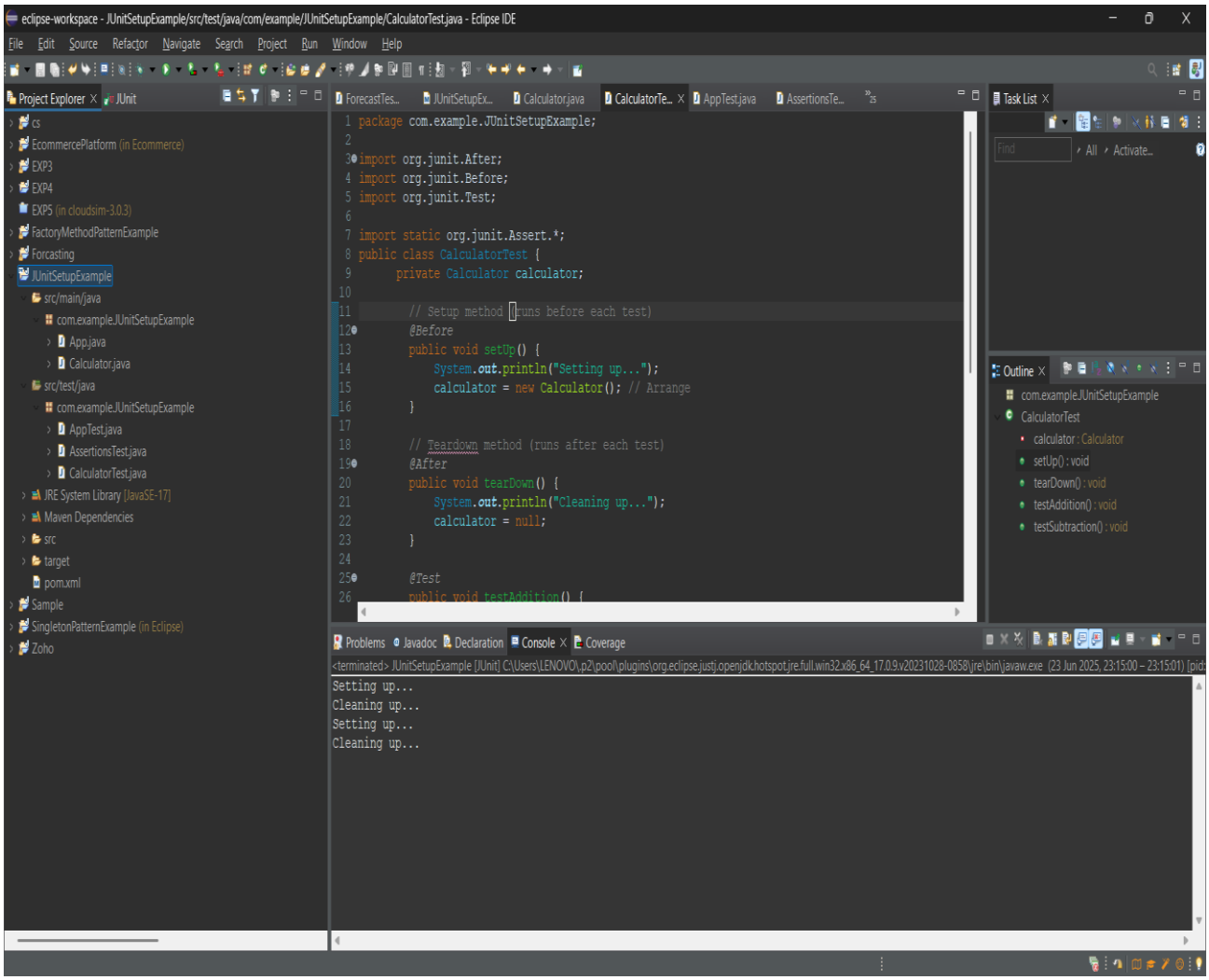
```
package com.example.JUnitSetupExample;


import org.junit.After;

import org.junit.Before;

import org.junit.Test;


import static org.junit.Assert.*;

public class CalculatorTest {
```

```
private Calculator calculator;

 // Setup method (runs before each test)
 @Before
 public void setUp() {
    System.out.println("Setting up...");
    calculator = new Calculator(); // Arrange
 }

 // Teardown method (runs after each test)
 @After
 public void tearDown() {
    System.out.println("Cleaning up...");
    calculator = null;
 }

 @Test
 public void testAddition() {
    // Arrange (done in setUp)

    // Act
    int result = calculator.add(10, 5);

    // Assert
    assertEquals(15, result);
 }
```

```
    @Test
    public void testSubtraction() {
        // Arrange (done in setUp)

        // Act
        int result = calculator.subtract(10, 5);

        // Assert
        assertEquals(5, result);
    }
}
```

Output Screenshot:




eclipse-workspace - JUnitSetupExample/src/test/java/com/example/JUnitSetupExample/CalculatorTest.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Project Explorer
JUnit
cs
EcommercePlatform (in Ecommerce)
EXP3
EXP4
EXP5 (in cloudsim-3.0.3)
FactoryMethodPatternExample
Forcasting
JUnitSetupExample
src/main/java
com.example.JUnitSetupExample
App.java
Calculator.java
src/test/java
com.example.JUnitSetupExample
AppTest.java
AssertionsTest.java
CalculatorTest.java
JRE System Library [JavaSE-17]
Maven Dependencies
src
target
pom.xml
Sample
SingletonPatternExample (in Eclipse)
Zoho
ForecastTes...
JUnitSetupEx...
Calculator.java
CalculatorTe...
AppTest.java
AssertionsTe...
1 package com.example.JUnitSetupExample;
2
3 import org.junit.After;
4 import org.junit.Before;
5 import org.junit.Test;
6
7 import static org.junit.Assert.*;
8 public class CalculatorTest {
9 private Calculator calculator;
10
11 // Setup method (runs before each test)
12 @Before
13 public void setUp() {
14 System.out.println("Setting up...");
15 calculator = new Calculator(); // Arrange
16 }
17
18 // Teardown method (runs after each test)
19 @After
20 public void tearDown() {
21 System.out.println("Cleaning up...");
22 calculator = null;
23 }
24
25 @Test
26 public void testAddition() {
Task List
Find
All
Activate...
Outline
com.example.JUnitSetupExample
CalculatorTest
calculator : Calculator
setUp() : void
tearDown() : void
testAddition() : void
testSubtraction() : void
Problems
Javadoc
Declaration
Console
Coverage
<terminated> JUnitSetupExample [JUnit] C:\Users\LENOVO\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.9.v20231028-0858\jre\bin\javaw.exe (23 Jun 2025, 23:15:00 – 23:15:01) [pid:
Setting up...
Cleaning up...
Setting up...
Cleaning up...