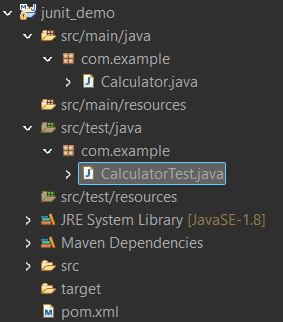
**Exercise 1: Setting Up JUnit**

**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.

**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

http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.example</groupId>

<artifactId>junit\_demo</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>jar</packaging>

<name>JUnit Demo Project</name>

<dependencies>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>4.13.2</version>

<scope>test</scope>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-surefire-plugin</artifactId>

<version>3.0.0-M5</version>

</plugin>

</plugins>

</build>

</project>

3. Create a new test class in your project.

**Calculator.java**

package com.example;

public class Calculator {

public int add(int a, int b) {

return a + b;

}

}

**CalculatorTest.java:**

package com.example;

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

import org.junit.Test;

public class CalculatorTest {

*@Test*

public void add\_TwoNumbers() {

Calculator calc = new Calculator();

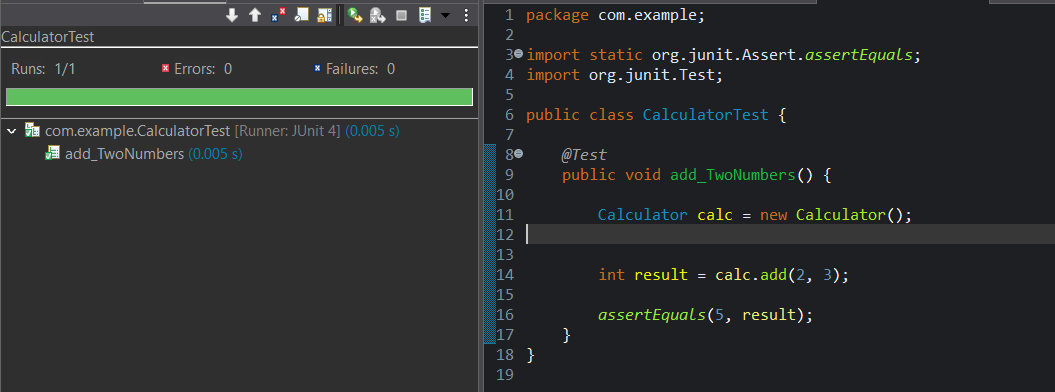
int result = calc.add(2, 3);

*assertEquals*(5, result);

}

}

**Output:**



**Exercise 2: 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.

**StringUtils.java:**

package com.example;

public class StringUtils {

public boolean isPalindrome(String input) {

if (input == null) return false;

String reversed = new StringBuilder(input).reverse().toString();

return input.equals(reversed);

}

public int[] getEvenNumbers() {

return new int[] {2, 4, 6, 8};

}

public String getGreeting(String name) {

return (name != null) ? "Hello, " + name : null;

}

}

**StringUtilsTest.java:**

package com.example;

import org.junit.Test;

import static org.junit.Assert.\*;

public class StringUtilsTest {

StringUtils utils = new StringUtils();

@Test

public void testIsPalindrome() {

assertTrue(utils.isPalindrome("madam"));

assertFalse(utils.isPalindrome("hello"));

}

@Test

public void testAssertEquals() {

assertEquals("Numbers should be equal", 5, 2 + 3);

assertEquals("Strings should match", "OpenAI", "OpenAI");

}

@Test

public void testAssertArrayEquals() {

int[] expected = {2, 4, 6, 8};

assertArrayEquals("Arrays should match", expected, utils.getEvenNumbers());

}

@Test

public void testAssertNotNull() {

assertNotNull("Greeting should not be null", utils.getGreeting("Vinay"));

}

@Test

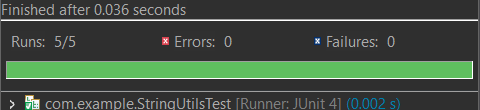
public void testAssertNull() {

assertNull("Greeting should be null for null input", utils.getGreeting(null));

}

}

**OUTPUT:**

****

**Exercise 3: 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. s.

**BankAccount.java:**

package com.example;

public class BankAccount {

private int balance;

public BankAccount() {

this.balance = 0;

}

public void deposit(int amount) {

if (amount > 0) {

this.balance += amount;

}

}

public void withdraw(int amount) {

if (amount > 0 && amount <= this.balance) {

this.balance -= amount;

}

}

public int getBalance() {

return this.balance;

}

}

**BankAccountTest.java:**

package com.example;

import org.junit.After;

import org.junit.Before;

import org.junit.Test;

import static org.junit.Assert.\*;

public class BankAccountTest {

private BankAccount account;

// Setup

*@Before*

public void setUp() {

System.***out***.println("Setup - Creating new bank account");

account = new BankAccount();

}

// Teardown

*@After*

public void tearDown() {

System.***out***.println("Teardown - Clearing bank account");

account = null;

}

*@Test*

public void testDeposit() {

account.deposit(1000);

*assertEquals*(1000, account.getBalance());

}

*@Test*

public void testWithdraw() {

account.deposit(1000);

account.withdraw(300);

*assertEquals*(700, account.getBalance());

}

*@Test*

public void testWithdrawExceedsBalance() {

account.deposit(500);

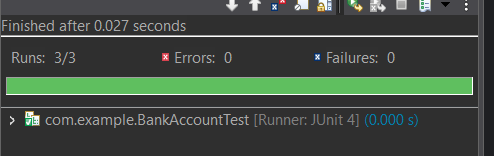
account.withdraw(1000);

*assertEquals*(500, account.getBalance());

}

}

**OUTPUT:**

****