J UNIT TESTING

Calculater.java

package com.ecommerce.junit;

public class Calculator {

public int add(int a,int b) {

return a+b;

}

public int divide(int a,int b) {

return a/b;

}

}

Test Cases

CalculatorTest

package com.ecommerce.junit;

import static org.junit.jupiter.api.Assertions.\*;

import org.junit.jupiter.api.Test;

class CalculatorTest {

@Test

void testAddPositiveValues() {

Calculator cal = new Calculator();

int a=2;

int b=5;

assertEquals(7, cal.add(a, b));

}

@Test

void testAddWhenAddingNegativeValues() {

Calculator cal = new Calculator();

int a=-2;

int b=-3;

assertEquals(-5, cal.add(a, b));

}

@Test

void testAddWhenUsingLargeValues() {

Calculator cal = new Calculator();

int a=2500;

int b=1000;

assertEquals(3500, cal.add(a, b));

}

}

assertionTest

package com.ecommerce.junit;

import static org.junit.jupiter.api.Assertions.\*;

import org.junit.jupiter.api.Test;

class AssertionsTestDemo {

@Test

void test() {

String str = null;

String str2 = "some value";

String[] a1 = { "A", "B" };

String[] a2 = { "A", "B" };

int a=4;

int b=0;

assertTrue(a > b);

assertFalse(5 < 1);

assertNull(str);

assertNotNull(str2);

assertSame(str, str);

assertNotSame(str, str2);

assertEquals(5, 5);

assertNotEquals(5, 6);

assertArrayEquals(a1, a2);

assertThrows(RuntimeException.class, () -> {

throw new RuntimeException();

});

}

}

ConditionalTest

package com.ecommerce.junit;

import static org.junit.jupiter.api.Assertions.assertEquals;

import org.junit.jupiter.api.Test;

import org.junit.jupiter.api.condition.EnabledOnOs;

import org.junit.jupiter.api.condition.OS;

class ConditionalTest {

@Test

@EnabledOnOs({OS.WINDOWS})

public void testAddOnWindows() {

Calculator cal =new Calculator();

int x=2;

int y=5;

assertEquals(7, cal.add(x, y));

}

// @Test

// @EnabledOnOs({OS.LINUX})

// public void testAddOnLinux() {

// Calculator cal =new Calculator();

//

// int x=2;

// int y=5;

//

// assertEquals(7, cal.add(x, y));

// }

}

DITest

package com.ecommerce.junit;

import static org.junit.jupiter.api.Assertions.*assertTrue*;

import org.junit.jupiter.api.DisplayName;

import org.junit.jupiter.api.Tag;

import org.junit.jupiter.api.Test;

import org.junit.jupiter.api.TestInfo;

class DITestDemo {

*@Test*

*@DisplayName*("TEST 1")

*@Tag*("addition")

*@Tag*("calculator")

void test(TestInfo testinfo) {

// ...as usual test our calculator add functionality

System.***out***.println("I am a test case. My display name is "+ testinfo.getDisplayName());

System.***out***.println("I am a test case. I have been tagged as "+ testinfo.getTags());

*assertTrue*(testinfo.getTags().contains("addition"));

}

}

DynamicTest

package com.ecommerce.junit;

import static org.junit.jupiter.api.Assertions.*assertThrows*;

import static org.junit.jupiter.api.Assertions.*assertTrue*;

import static org.junit.jupiter.api.DynamicTest.*dynamicTest*;

import java.util.Arrays;

import java.util.Collection;

import org.junit.jupiter.api.DynamicTest;

import org.junit.jupiter.api.TestFactory;

class DynamicTestsDemo {

*@TestFactory*

Collection<DynamicTest> dynamicTests1() {

return Arrays.*asList*(

*dynamicTest*("Dynamic test 1", () -> *assertTrue*(7 ==new Calculator().add(2, 5))),

*dynamicTest*("Dynamic test 2 for cal div", () -> *assertTrue*(2 ==new Calculator().divide(5, 2))),

*dynamicTest*("Dynamic test 3 for cal div by 0", () -> *assertThrows*(ArithmeticException.class, () -> new Calculator().divide(5,0)))

);

}

/\*

\* @TestFactory Collection<DynamicTest> dynamicTests2() {

\*

\* int[] a= {2,3,4} ; int[] b= {5,6,7}; int[] result= {7,9,11}; int i=0;

\*

\* return Arrays.asList(

\*

\* dynamicTest("Dynamic test ", () -> assertTrue(result[i++] ==new

\* Calculator().add(a[i++],b[i++]))), dynamicTest("Dynamic test ", () ->

\* assertTrue(result[i++] ==new Calculator().add(a[i++],b[i++]))),

\* dynamicTest("Dynamic test ", () -> assertTrue(result[i++] ==new

\* Calculator().add(a[i++],b[i++])))

\*

\* );

\*

\* }

\*/

}

NestedTest

package com.ecommerce.junit;

import org.junit.jupiter.api.Nested;

import org.junit.jupiter.api.Test;

class NestedTestDemo {

@Test

void test() {

System.out.println("Inside test()");

}

@Nested

class GroupA {

@Test

void testA1() {

System.out.println("Inside testA1()");

}

@Test

void testA2() {

System.out.println("Inside testA2()");

}

}

@Nested

class GroupB {

@Test

void testB1() {

System.out.println("Inside testB1()");

}

@Test

void testB2() {

System.out.println("Inside testB2()");

}

}

}

RepeatedTest

package com.ecommerce.junit;

import static org.junit.jupiter.api.Assertions.assertTrue;

import org.junit.jupiter.api.RepeatedTest;

class RepeatedTestDemo {

@RepeatedTest(5)

void testAddPositiveValues() {

Calculator cal = new Calculator();

int a=2;

int b=5;

assertTrue(7 ==cal.add(a, b));

}

}

Test1

package com.ecommerce.junit;

import org.junit.jupiter.api.AfterAll;

import org.junit.jupiter.api.AfterEach;

import org.junit.jupiter.api.BeforeAll;

import org.junit.jupiter.api.BeforeEach;

import org.junit.jupiter.api.DisplayName;

import org.junit.jupiter.api.Test;

@DisplayName("This is my first Test class")

class Test1 {

@BeforeAll

static void myBeforeAll() {

// Create a Connectn Obj here that will used in the Test cases Test1, Test2,....

System.out.println("Inside myBeforeAll()");

}

@AfterAll

static void myAfterAll() {

// Close the Connectn Obj here.

System.out.println("Inside myAfterAll()");

}

@BeforeEach

void myBeforeEach() {

// Create a Statement Obj here that will used in the Test cases Test1, Test2,....

System.out.println("Inside myBeforeEach()");

}

@AfterEach

void myAfterEach() {

// close Statement Obj here so that it will release system resources.

System.out.println("Inside myAfterEach()");

}

@Test

@DisplayName("This is my first Test case Test 1")

void test1() {

// Test a JDBC SQL Query select \* for eproduct

System.out.println("Test1 ");

}

@Test

@DisplayName("This is my second Test case Test 2")

void test2() {

// Test a JDBC SQL Query for eproduct where price>1000

System.out.println("Test2 ");

}

}