JUnit 5 @RepeatedTest

The **@RepeatedTest** annotation is used to write **repeatable test templates** that could run multiple times in JUnit. The *repetition frequency* can be configured as a parameter to @RepeatedTest annotation.

Table Of Contents

**1. @RepeatedTest Annotation**

The *@RepeatedTest* annotation is used to mark a test method that should repeat a specified number of times with a configurable display name.

To repeat a test with different arguments, consider using [@ParameterizedTest](https://howtodoinjava.com/junit5/parameterized-tests/).

**1.1. Syntax**

To create a repeatable test, annotate the test method with @RepeatedTest.

In the given example, the test method uses *@RepeatedTest(5)* annotation. It means that the test will be executed five times.

@DisplayName("Add operation test")

@RepeatedTest(5)

**void** addNumber(TestInfo testInfo) {

Calculator calculator = **new** Calculator();

Assertions.assertEquals(2, calculator.add(1, 1), "1 + 1 should equal 2");

}

**1.2. Lifecycle Methods**

Please note that each invocation of **a repeated test behaves like the execution of a regular test**with full support for the same lifecycle callbacks and extensions.

It means that [@BeforeEach](https://howtodoinjava.com/junit5/before-each-annotation-example/) and [@AfterEach](https://howtodoinjava.com/junit5/after-each-annotation-example/) annotated **lifecycle methods will be invoked for each invocation** **of the test**.

**package** com.howtodoinjava.junit5.examples;

**import** org.junit.jupiter.api.AfterAll;

**import** org.junit.jupiter.api.AfterEach;

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

**import** org.junit.jupiter.api.BeforeAll;

**import** org.junit.jupiter.api.BeforeEach;

**import** org.junit.jupiter.api.DisplayName;

**import** org.junit.jupiter.api.RepeatedTest;

**import** org.junit.jupiter.api.RepetitionInfo;

**import** org.junit.jupiter.api.Test;

**import** org.junit.jupiter.api.TestInfo;

**import** org.junit.platform.runner.JUnitPlatform;

**import** org.junit.runner.RunWith;

@RunWith(JUnitPlatform.**class**)

**public** **class** RepeatedTestExample {

@BeforeAll

**public** **static** **void** init(){

System.out.println("Before All init() method called");

}

@BeforeEach

**public** **void** initEach(){

System.out.println("Before Each initEach() method called");

}

@DisplayName("Add operation test")

@RepeatedTest(5)

**void** addNumber(TestInfo testInfo, RepetitionInfo repetitionInfo)

{

System.out.println("Running addNumber test -> " + repetitionInfo.getCurrentRepetition());

Assertions.assertEquals(2, Calculator.add(1, 1), "1 + 1 should equal 2");

}

@AfterEach

**public** **void** cleanUpEach(){

System.out.println("After Each cleanUpEach() method called");

}

@AfterAll

**public** **static** **void** cleanUp(){

System.out.println("After All cleanUp() method called");

}

}

The output of the above test:

Before All init() method called

Before Each initEach() method called

After Each cleanUpEach() method called

Before Each initEach() method called

Running addNumber test -> 1

After Each cleanUpEach() method called

Before Each initEach() method called

Running addNumber test -> 2

After Each cleanUpEach() method called

Before Each initEach() method called

Running addNumber test -> 3

After Each cleanUpEach() method called

Before Each initEach() method called

Running addNumber test -> 4

After Each cleanUpEach() method called

Before Each initEach() method called

Running addNumber test -> 5

After Each cleanUpEach() method called

After All cleanUp() method called