Programmation avancée et application • TD 5

JUnit 5

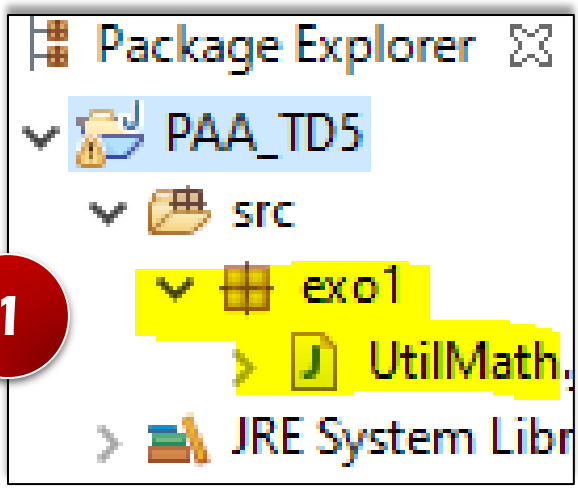
Tests unitaires

Exercice I
Faire des tests unitaires pour les méthodes fact et comb du TD précédent (Exercice I, Exceptions et UtilMath).

TD 4

```
package exo1;
public class UtilMath {

	// calcule la factorielle n! d'un entier positif n
	public static long fact(int n) throws IllegalArgumentException
	{
		// Le mot-cle throws dans la signature indique que la methode est susceptible
de lever une exception

		if(n < 0)
			throw new IllegalArgumentException("Le paramètre n est négatif.");
		// Le mot-cle throw suivi d'un objet de type Exception permet de lever
l'exception.

		/* java.lang.IllegalArgumentException.IllegalArgumentException(String s)
		 * Constructs an IllegalArgumentException with thespecified detail message.
		 * Parameters:s the detail message.
		 */

		long result = 1;
		for(int i = 1 ; i <= n ; i++)
			result *= i;

		return result;
	}

	// calcule la combinaison de deux entiers positifs p et n
	public static long comb(int n, int p) throws IllegalArgumentException
	{
		if(p > n)
			throw new IllegalArgumentException("Le paramètre p est plus grand que
le paramètre n.");

		return ( ( fact(n) ) / ( fact(p) * fact(n - p) ) );
	}
}
```

Rappel : $\binom{n}{p} = \frac{n!}{p! \times (n-p)!}$, avec $p \leqslant n$.

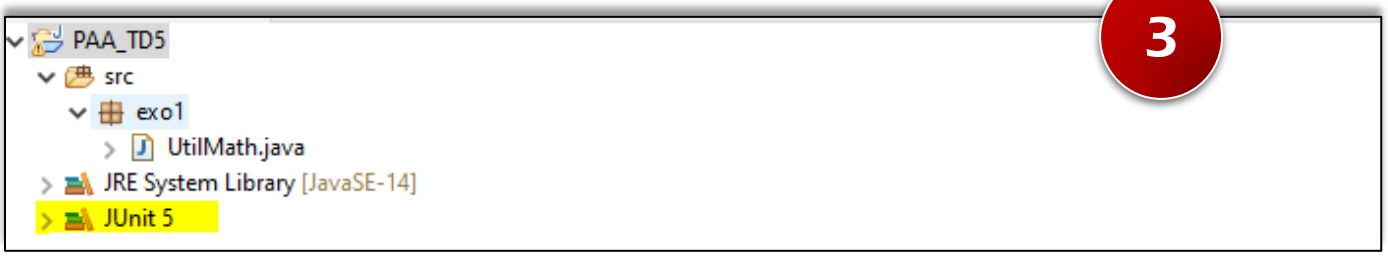


Configure JUnit5 in Eclipse

Step By Step

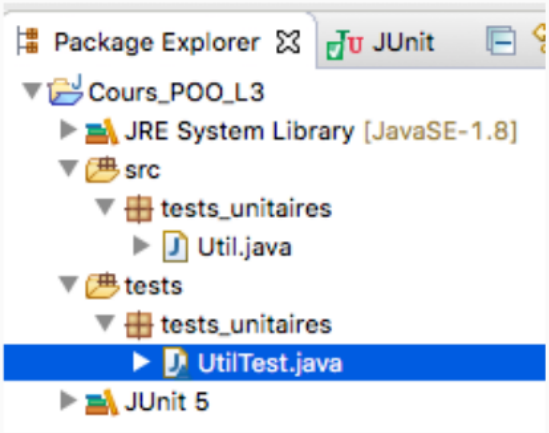
https://www.youtube.com/watch?v=K7Z8dVgA9rY

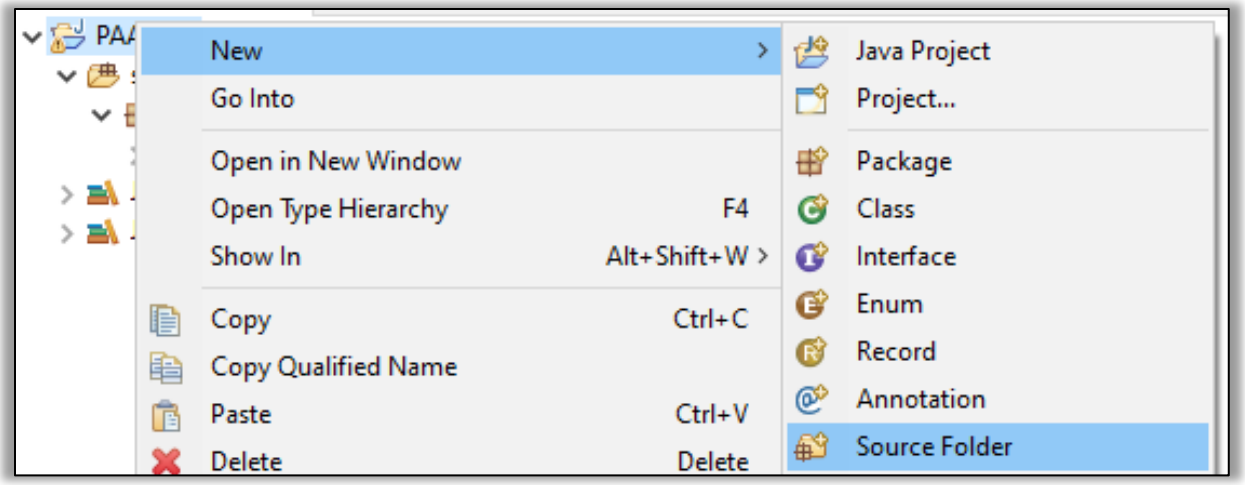
2

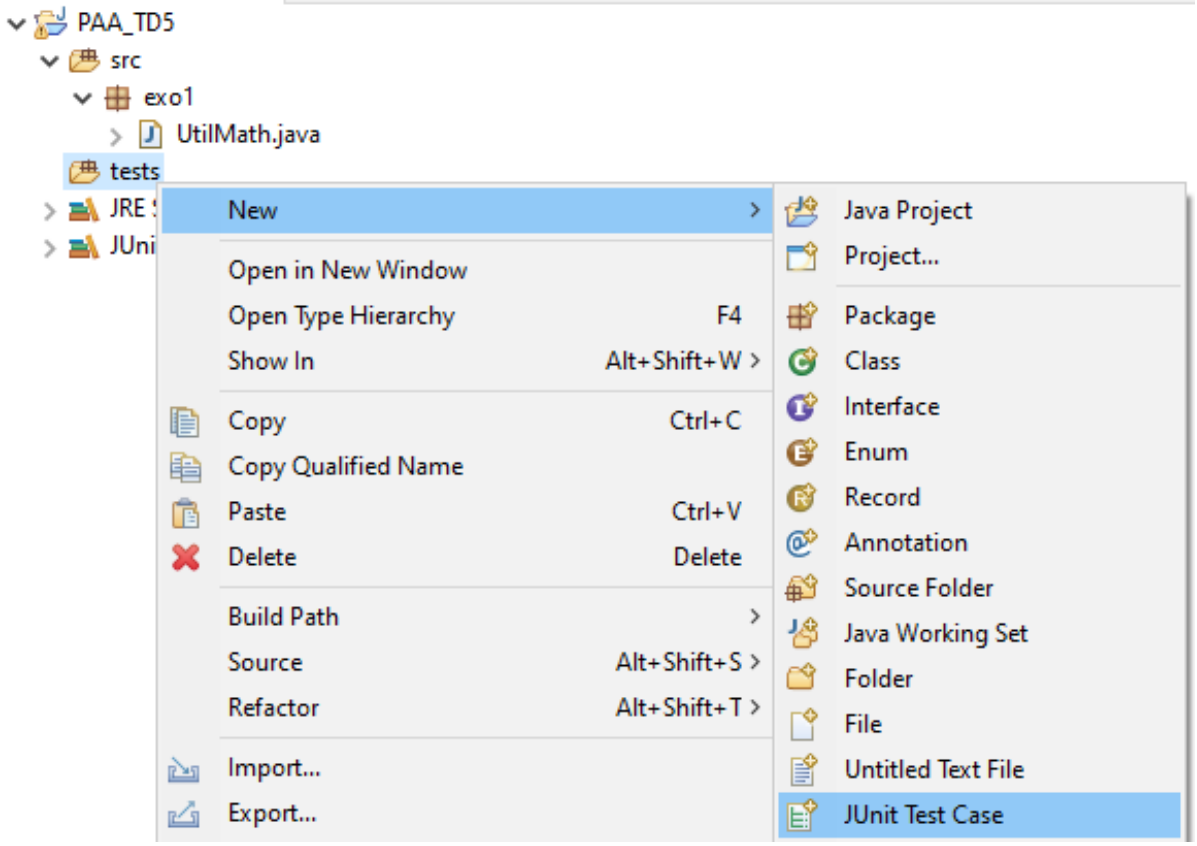


Description du fonctionnement de JUnit 5

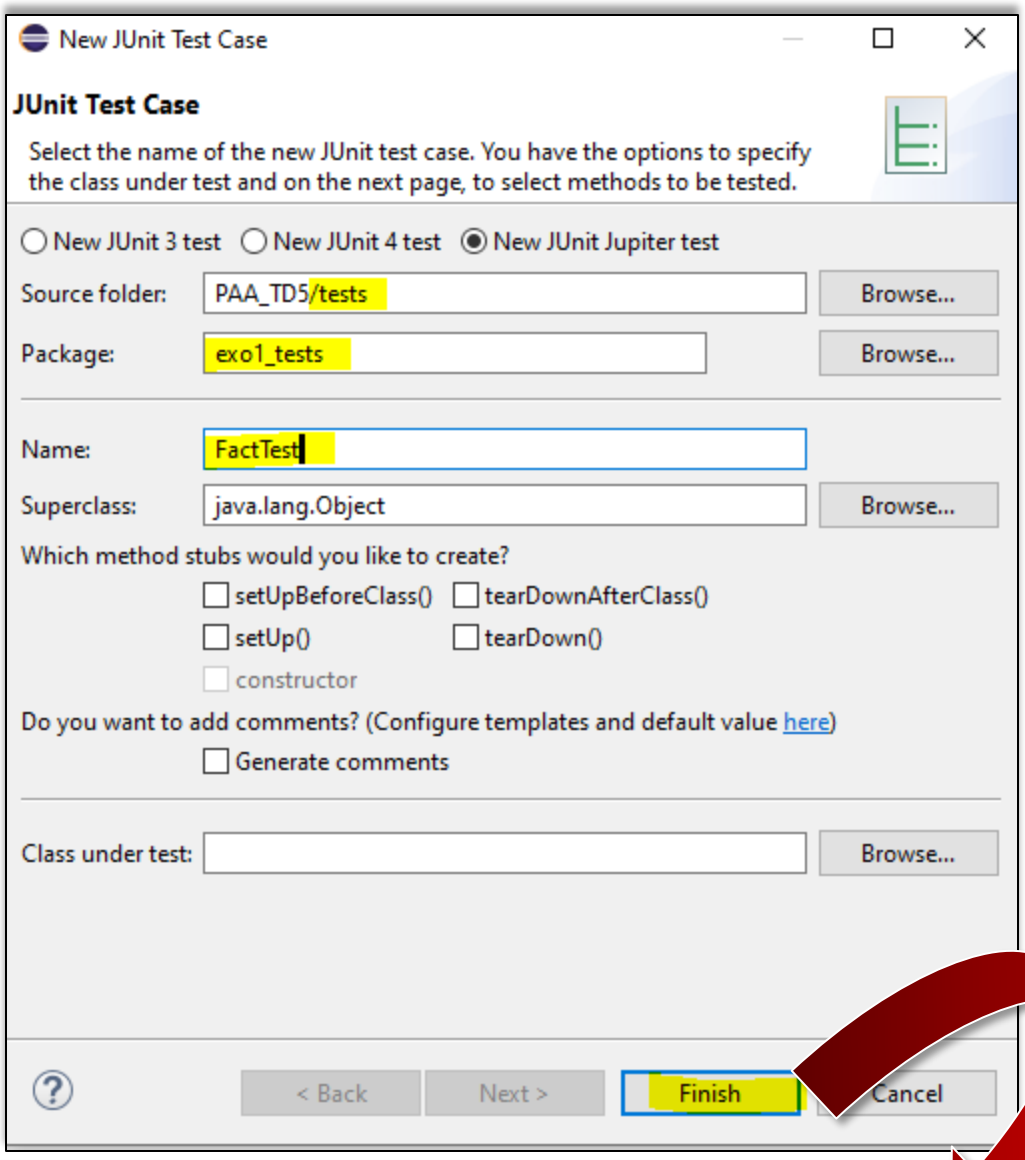
- Un test unitaire : une méthode « spéciale » qui utilise l'API JUnit 5
- Les tests unitaires qui sont liés sont regroupés dans une même classe
- Une classe de tests correspond à une classe ou une méthode de `src`
- Les classes de tests sont regroupées dans répertoire `tests` situé au même niveau que le répertoire `src`
- Le répertoire `tests` est composé des mêmes packages que `src`
- Une méthode dans `src` $\longrightarrow$ au moins un test

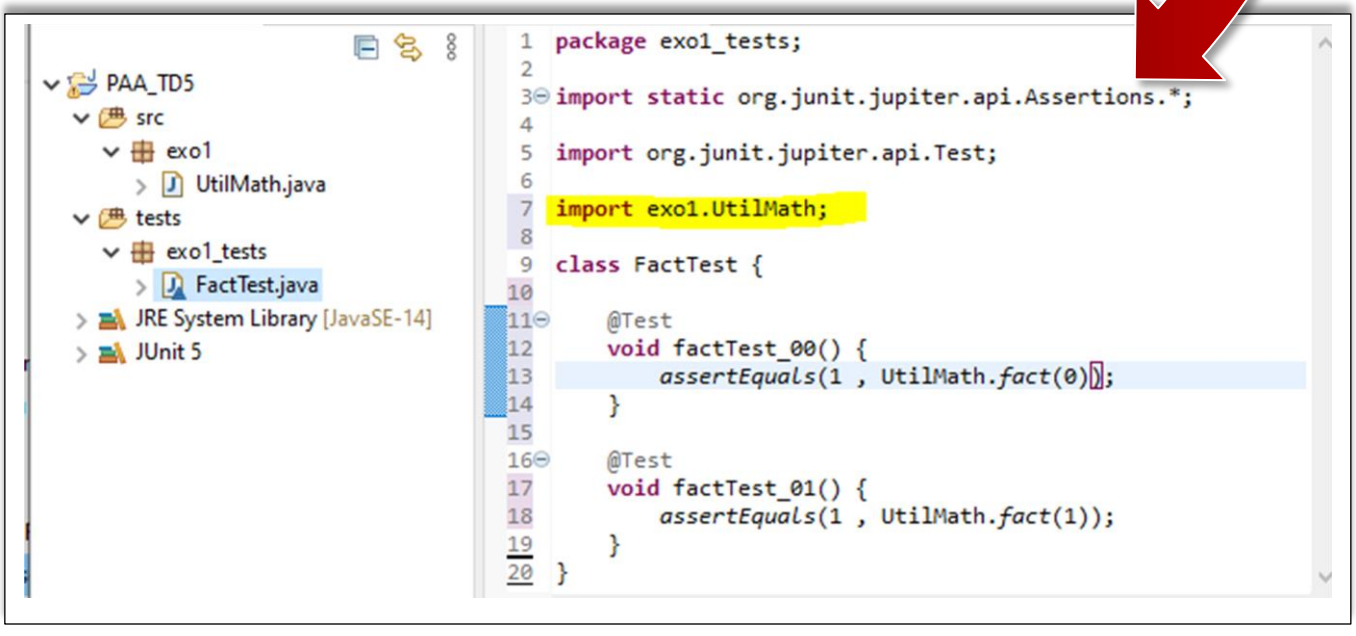






- Une classe de tests correspond à une classe ou une méthode de `src`
- Le répertoire `tests` est composé des mêmes packages que `src`

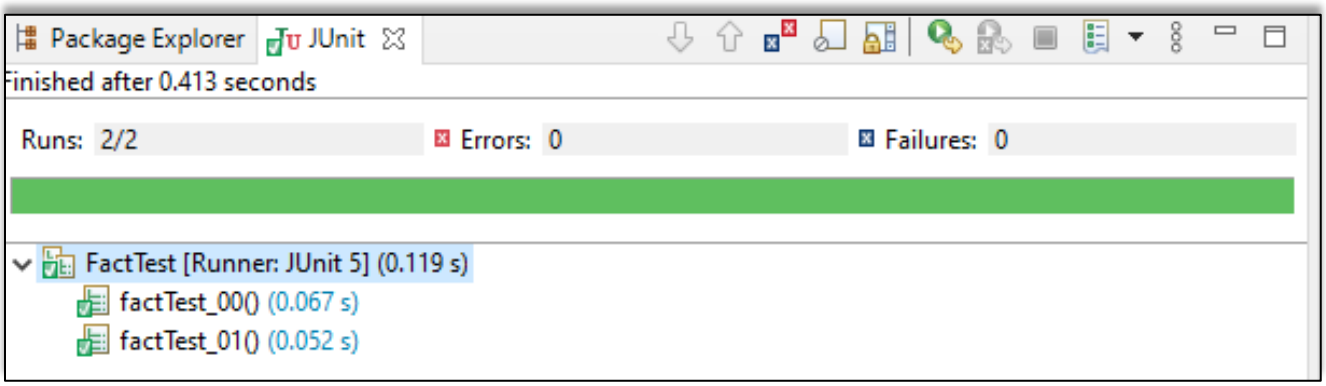

New JUnit Test Case
JUnit Test Case
Select the name of the new JUnit test case. You have the options to specify the class under test and on the next page, to select methods to be tested.
New JUnit 3 test
New JUnit 4 test
New JUnit Jupiter test
Source folder:
PAA_TD5/tests
Browse...
Package:
exo1_tests
Browse...
Name:
FactTest
Superclass:
java.lang.Object
Browse...
Which method stubs would you like to create?
setUpBeforeClass()
tearDownAfterClass()
setUp()
tearDown()
constructor
Do you want to add comments? (Configure templates and default value here)
Generate comments
Class under test:
Browse...
< Back
Next >
Finish
Cancel


PAA_TD5
src
exo1
UtilMath.java
tests
exo1_tests
FactTest.java
JRE System Library [JavaSE-14]
JUnit 5
package exo1_tests;
import static org.junit.jupiter.api.Assertions.*;
import org.junit.jupiter.api.Test;
import exo1.UtilMath;
class FactTest {
@Test
void factTest_00() {
assertEquals(1 , UtilMath.fact(0));
}
@Test
void factTest_01() {
assertEquals(1 , UtilMath.fact(1));
}
}

FactTest.java > Run As > Junit Test

Solution pour le message d'erreur
https://youtu.be/zK6bVOhrxrc

Résultat



FactTest.java

```java
package exo1_tests;

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

import exo1.UtilMath;

class FactTest {

	@Test
	void factTest_00() {
		assertEquals(1 , UtilMath.fact(0));
	}

	@Test
	void factTest_01() {
		assertEquals(1 , UtilMath.fact(1));
	}

	@Test
	void factTest_02() {
		assertEquals(2 , UtilMath.fact(2));
	}

	@Test
	void factTest_03() {
		assertEquals(6 , UtilMath.fact(3));
	}

	@Test
	void factTest_04() {
		assertEquals(24 , UtilMath.fact(4));
	}

	@Test
	void factTest_05() {
		assertEquals(120 , UtilMath.fact(5));
	}

	@Test
	void factTest_06() {
		assertEquals(479001600 , UtilMath.fact(12));
	}

	/* ****************************************** */

	@Test
	void testNegativeValue_00() {
		assertThrows(IllegalArgumentException.class, () -> {UtilMath.fact(-1);});
	}

	@Test
	void testNegativeValue_01() {
		assertThrows(IllegalArgumentException.class, () -> {UtilMath.fact(-2);});
	}

	@Test
	void testNegativeValue_02() {
		assertThrows(IllegalArgumentException.class, () -> {UtilMath.fact(-3);});
	}

	@Test
	void testNegativeValue_03() {
		assertThrows(IllegalArgumentException.class, () -> {UtilMath.fact(-1000);});
	}
}
```