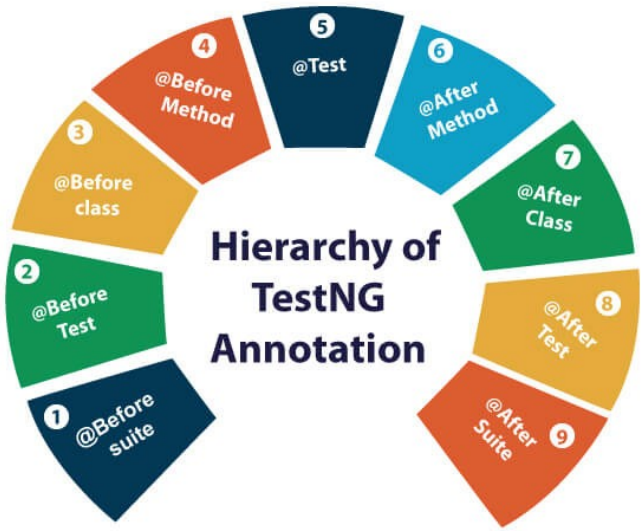
Test Java

Hiérarchie des annotations:



procédure de test typique :

Avant	Traitement	Après
Compte1 à 100€	Dépôt de 50	Compte1 à 150€
Compte1 à 100€	Dépôt de -50	Compte1 à 100€
Compte1 à 100€	Dépôt de 0	Compte1 à 100€
Compte1 à 100€	Dépôt de null	Compte1 à 100€ (on capture l'erreur)
Compte1 à 100€	Retrait de 50	Compte1 à 50€, true
Compte1 à 100€	Retrait de -50	Compte1 à 100€, false
Compte1 à 100€	Retrait de 150	Compte1 à 100€, false
Compte1 à 100€	Retrait de null	Compte1 à 100€, false
Compte1 à 100€	Compte1 à découvert	false
Compte2 à 0€	Compte2 à découvert	false
Compte3 à -100€	Compte3 à découvert	true
Compte4 à null	Compte4 à découvert	CompteNullExeception

Faire un test :

-faire un class de test qui import junit
-créé des variables
-faire un setUp après une annotation before pour instancier nos variable à une valeur avant le lancement d'un test
-créé les différente méthodes de test

exemple :

```
import static org.junit.Assert.*;
import org.junit.Before;
import org.junit.jupiter.api.RepeatedTest;

import junit.framework.Test ;


public class TestCompte {
	static Object compte1;
	static Object compte2;
	static Object compte3;
	static Object compte4;

	@Before
	public void setUp() {
		compte1 = 100;
		compte2 = 0;
		compte3 = -400;
		compte4 = null;
	}

	@Test
	public void test1() {
		assertEquals(100, compte1);
	}

}
```

pour lancer un test :

org.junit.runner.JunitCore.*main*("testPackage.NomDeLaCLasseATester);

exemple :
dans une nouvelle class :

```
package testPackage;
import junit.framework.TestSuite;

public class LancerTest extends TestCompte {
     public static void main(String[] args){

     org.junit.runner.JUnitCore.main("testPackage.TestCompte");
     }
```

```
}
```

les différents types d'assertion :
fail()
assertTrue(condition)/assertFalse(condition) ou aussi assertTrue(message, condition)/assertFalse(message, condition)

- assertEquals(attendu, effectif) pour type boolean/byte/char/int/long/short/Object => existe aussi avec message
- assertEquals(attendu, effectif, delta) pour type float/double => existe aussi avec message
- assertNotNull(objet)/assertNull(objet) => existe aussi avec message
- assertNotSame(attendu, effectif) => existe aussi avec message
- + des variantes pour indiquer un message d'erreur (1er paramètre).

Assert equals :
argument : assertEquals([une desciption],valeur attendue, valeur traiter,[type (0,0 pour int et int par exemple)]) ;

exemple de test complet :

classe de test :

```
package testPackage;

import static org.junit.Assert.*;

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

public class TestCompte {
	static CompteBancaireClient compte1;
	static CompteBancaireClient compte2;
	static CompteBancaireClient compte3;
	static CompteBancaireClient compte4;
	private double calcul;
	private int result;

	@Before
	public void setUp() {
		compte1 = new CompteBancaireClient();
		compte1.setSolde(100);
		compte2 =  new CompteBancaireClient();
		compte3 =  new CompteBancaireClient();
		compte4 =  new CompteBancaireClient();
	}

	@Test
	public void testDepot1() {
		this.result = 150;
		compte1.depot(50);
		this.calcul=this.compte1.getSolde();
```

```
            assertEquals(result, calcul, 0.0);
      }

      @Test
      public void testDepot2() {
            this.result = 100;
            compte1.depot(-50);
            this.calcul=this.compte1.getSolde();
            assertEquals(result, calcul, 0.0);
      }
}
```

classe de lancement :

```
package testPackage;
import junit.framework.TestSuite;

public class LancerTest extends TestCompte {
      public static void main(String[] args){

      org.junit.runner.JUnitCore.main("testPackage.TestCompte");
      }
}
```