

Tests unitaires (JUnit)



JUnit est un framework open source de développement et d'exécution de tests unitaires automatisables, pour le langage de programmation Java. Le principal intérêt est de s'assurer que le code répond toujours aux besoins même après d'éventuelles modifications. Plus généralement, ce type de tests est appelé tests unitaires de non-régression.

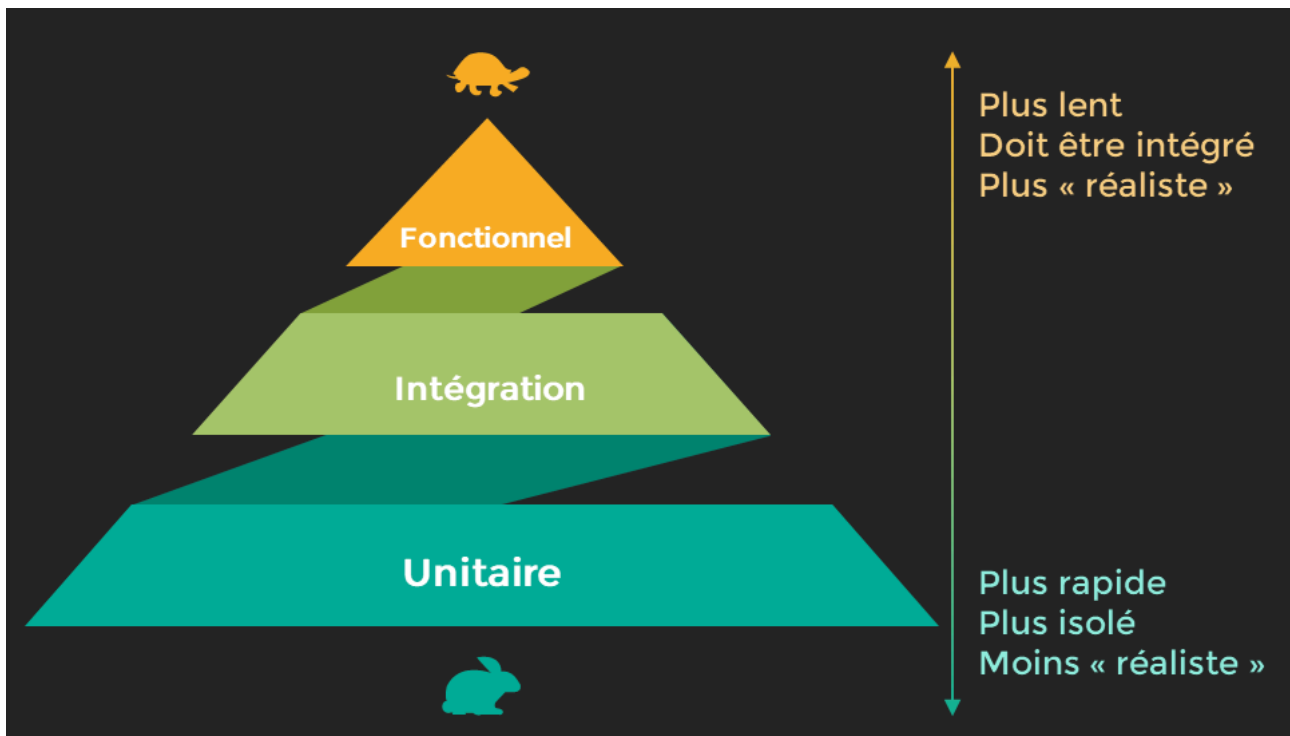
JUnit a été initialement développé par Erich Gamma et Kent Beck.

Comme Java est un langage orienté Objet, les tests JUnit sont regroupés dans des classes de test. Généralement, on groupe dans une classe les tests ayant la même classe comme point d'entrée et on nomme la classe de test à partir du nom de la classe testée préfixé ou suffixé par Test en la plaçant dans le même package.

Pourquoi faire des tests unitaires ? A quoi ça sert ?

Un test unitaire permet de s'assurer du fonctionnement correct d'une partie déterminée d'une application ou d'une partie d'un programme. Il a pour objectif d'isoler le comportement de la partie de code à tester de tout facteur extérieur et de vérifier qu'il est conforme à ce qui est attendu.

Le test unitaire va donc être écrit pour tester une toute petite partie du code source, indépendamment de l'environnement qui l'entoure. Il doit être déterministe, c'est-à-dire qu'exécuté plusieurs fois, il devra toujours retourner le même résultat.



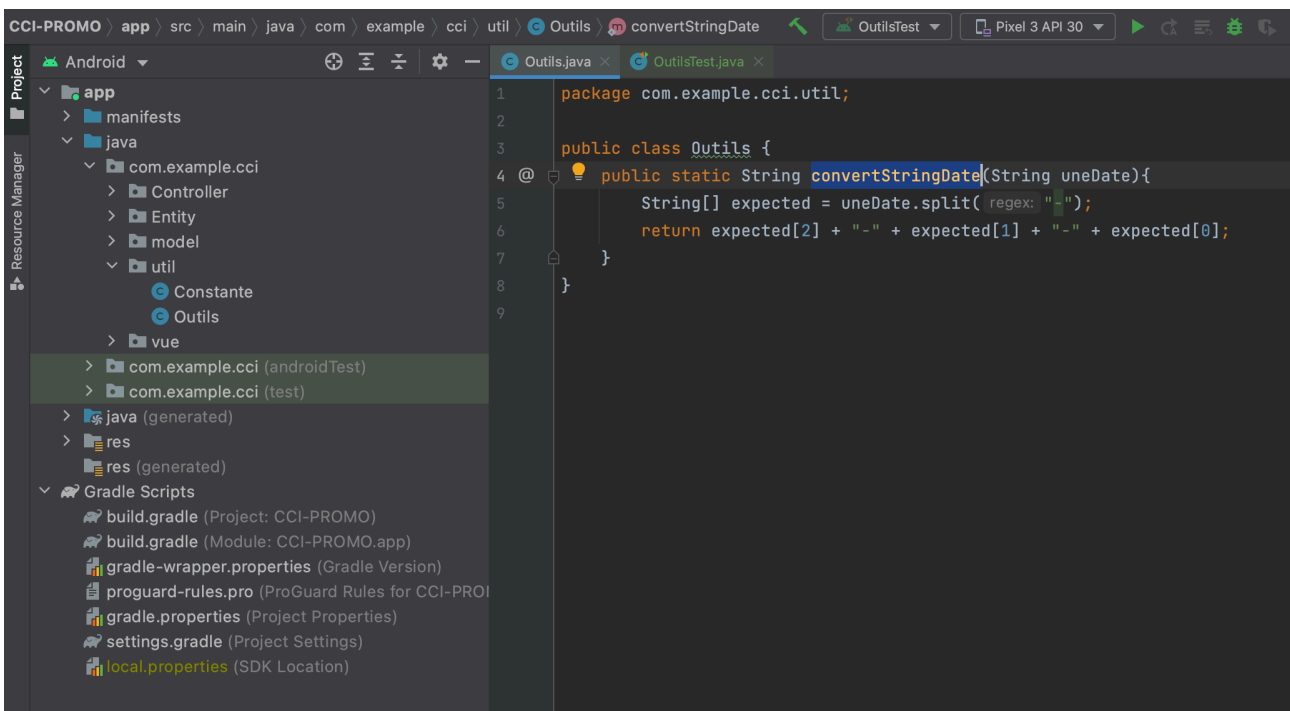
Exemple 1 présent au sein de l'application mobile CCI-PROMO

Au sein de mon application mobile CCI-PROMO j'ai une fonction dans la classe Outils permettant de convertir les dates.

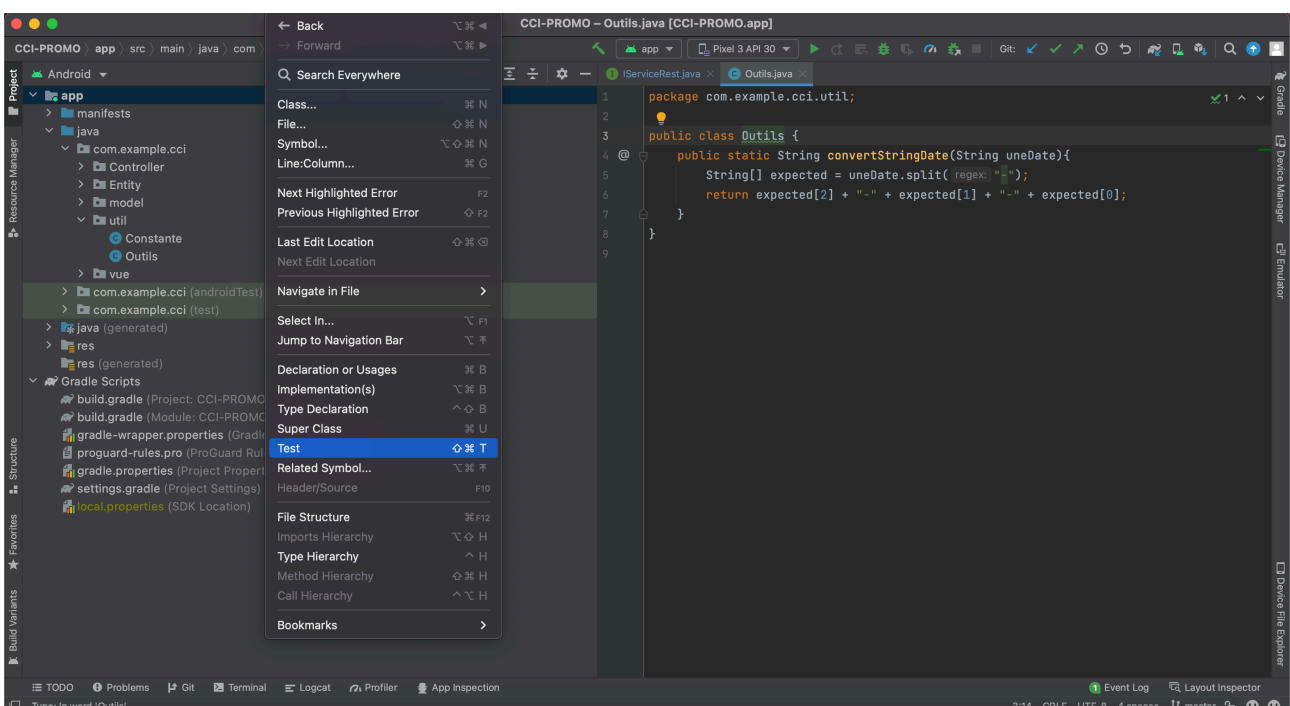
Plus précisément elle va découper une date qui utilise « - » comme séparateur et réunir tous les morceaux dans un tableau. Enfin elle va réécrire la date en positionnant les morceaux dans l'ordre souhaité.

Je décide donc de tester ma fonction. Pour ce faire :

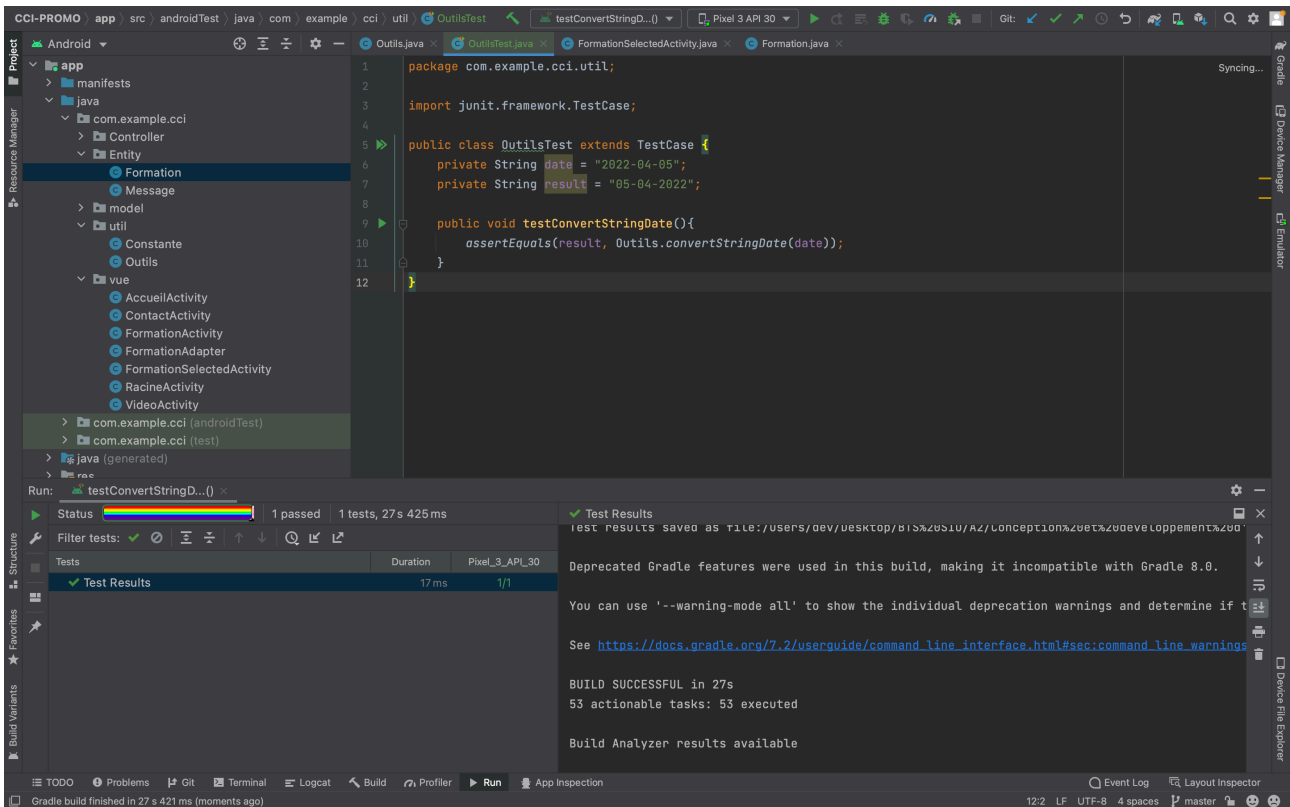
Dans le menu « Navigate », il faut sélectionner « Test », une classe « OutilsTest » va alors se générer. Il suffit alors de coder la fonction souhaitée.



```
1 package com.example.cci.util;
2
3 public class Outils {
4     @ public static String convertStringDate(String uneDate){
5         String[] expected = uneDate.split( regex: "-");
6         return expected[2] + "-" + expected[1] + "-" + expected[0];
7     }
8 }
9
```

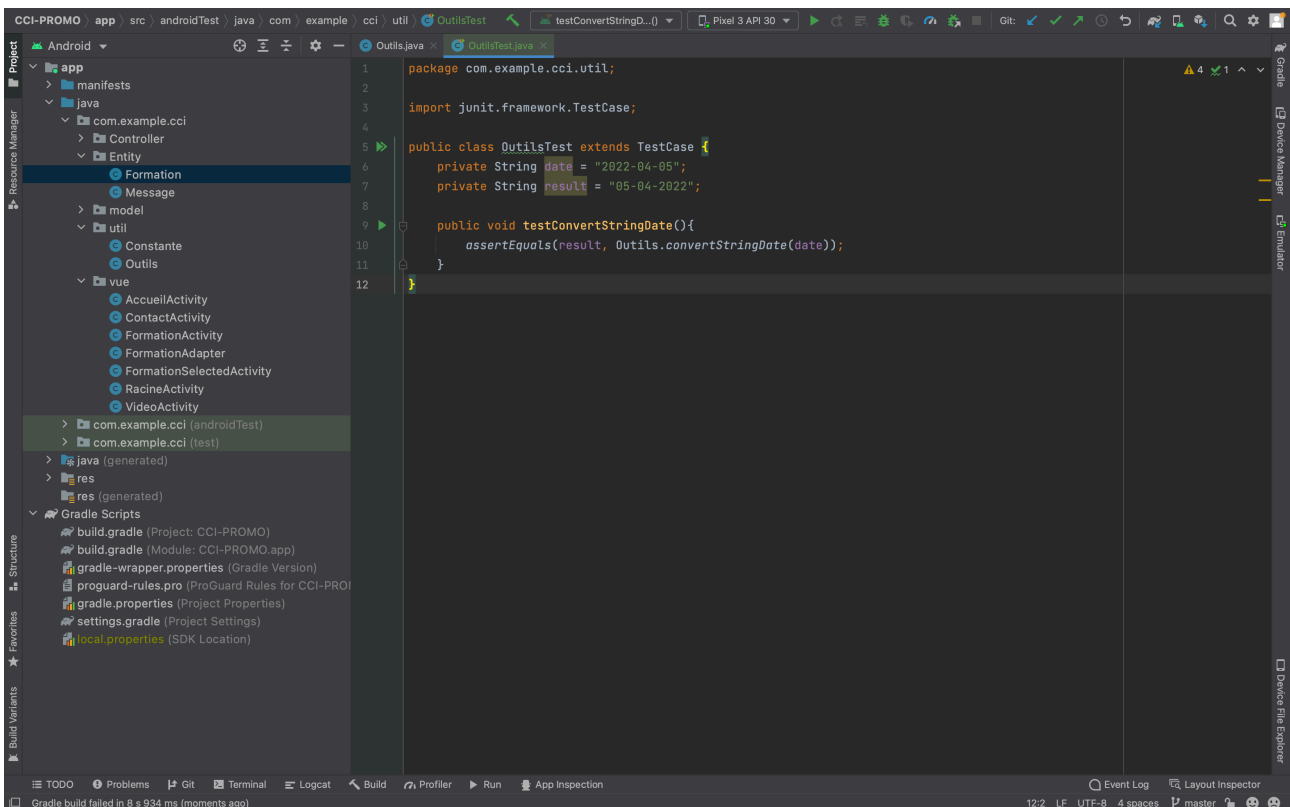


```
1 package com.example.cci.util;
2
3 public class Outils {
4     @ public static String convertStringDate(String uneDate){
5         String[] expected = uneDate.split( regex: "-");
6         return expected[2] + "-" + expected[1] + "-" + expected[0];
7     }
8 }
9
```

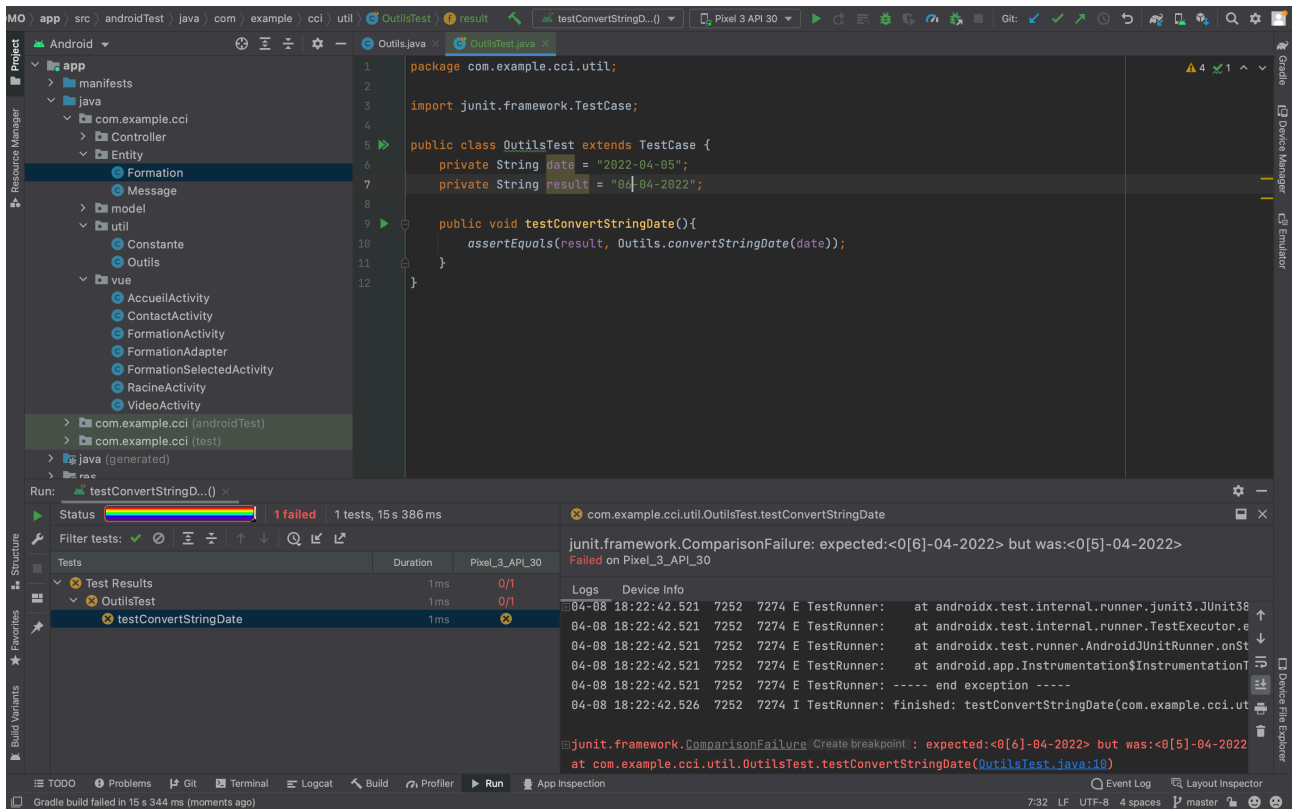


Enfin il faut run la fonction afin de la tester, voici 2 types de réponses possibles :

- **BUILD SUCCESSFUL**, le test est opérant et réussi

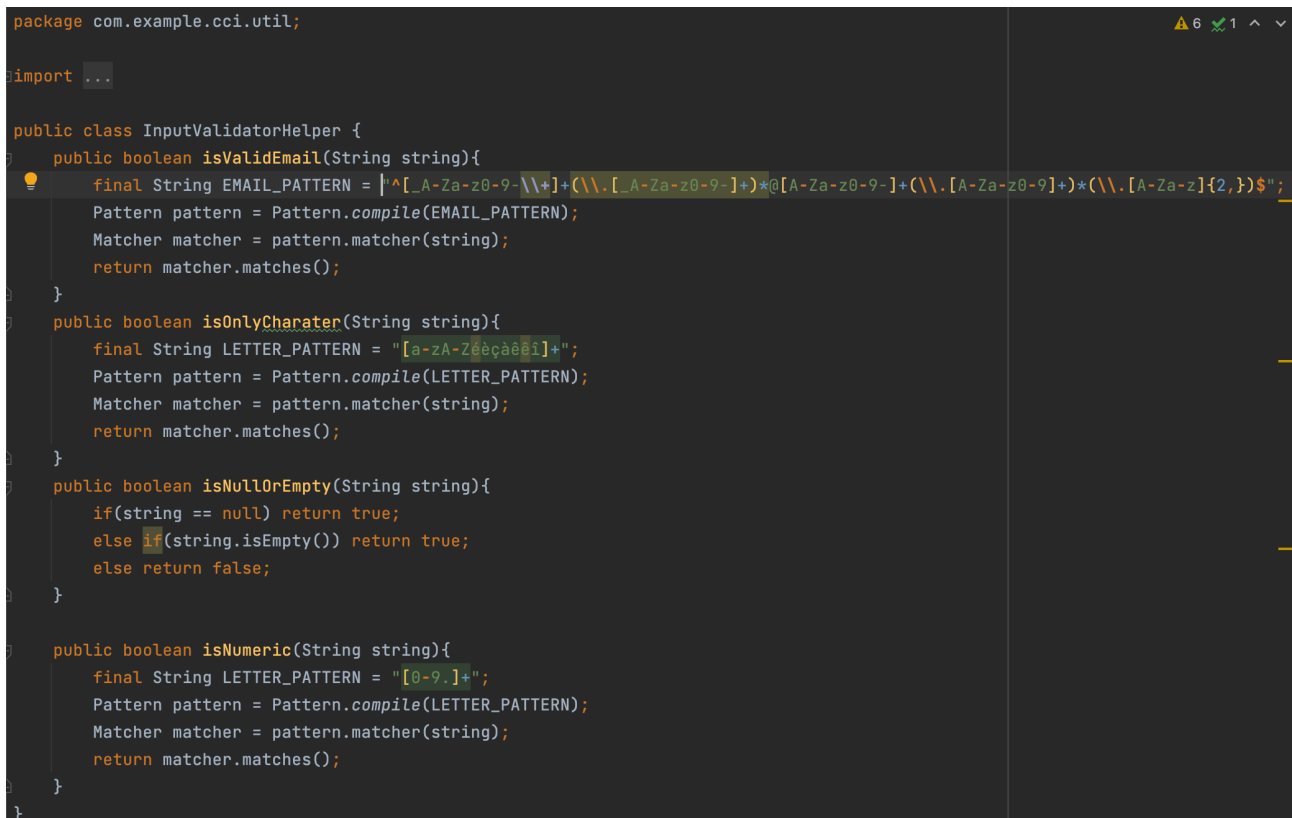


- **FAILED**, le test a échoué, il y a alors une explication



Exemple 2 présent au sein de l'application mobile CCI-PROMO

Grâce à cette classe je vérifie les données qui sont soumis dans le formulaire de contact.



Ainsi je crée une autre classe afin d'effectuer les tests de mes différentes fonctions.

```
package com.example.cci.util;

import junit.framework.TestCase;

public class InputValidatorHelperTest extends TestCase {
    private static String TAG = "InputValidatorHelperTest";
    private InputValidatorHelper inputValidatorHelper;

    public InputValidatorHelperTest() { inputValidatorHelper = new InputValidatorHelper(); }

    public void testIsValidEmail1() {
        String mail = "3azerty@test.a";
        assertEquals("expected: false", inputValidatorHelper.isValidEmail(mail));
    }

    public void testIsValidEmail2() {
        String mail = "2azertArotest.com";
        assertFalse(inputValidatorHelper.isValidEmail(mail));
    }

    public void testIsValidEmail3() {
        String mail = "nom@prenom.fr";
        assertTrue(inputValidatorHelper.isValidEmail(mail));
    }

    public void testIsValidEmail4() {
        String mail = "nom@pre!nom.fr";
        assertFalse(inputValidatorHelper.isValidEmail(mail));
    }

    public void testIsOnlyCharacter6() {
        String chaine = "àçèéêî";
        assertTrue(inputValidatorHelper.isOnlyCharacter(chaine));
    }

    public void testIsNullOrEmpty1() {
        String chaine = "23456";
        assertFalse(inputValidatorHelper.isNullOrEmpty(chaine));
    }

    public void testIsNullOrEmpty2() {
        String chaine = null;
        assertTrue(inputValidatorHelper.isNullOrEmpty(chaine));
    }

    public void testIsNullOrEmpty3() {
        String chaine = "";
        assertTrue(inputValidatorHelper.isNullOrEmpty(chaine));
    }

    public void testIsNullOrEmpty4() {
        String chaine = " ";
        assertFalse(inputValidatorHelper.isNullOrEmpty(chaine));
    }

    public void testIsNumeric1() {
        String chaine = "224S";
        assertFalse(inputValidatorHelper.isNumeric(chaine));
    }
}
```

Puis je réalise tous mes tests.

