Tests automatisés

# Définitions

## Quoi

* Etant donné (Given/Arrange) un état initial
* Lorsque (When/Act) on réalise quelque chose
* Alors (Then/Assert) on vérifie l'état final

## Qui

* Des **testeurs** logiciels (QA en anglais)
* Les **développeurs** eux-mêmes (de plus en plus)
* Certains disent le **client** (voir le format Gherkin et Cucumber)

# Type de tests

Une image contenant diagramme

Description générée automatiquement

### Tests automatisés

### Tests unitaires

vérifient le bon fonctionnement d'un morceau de code (unité)

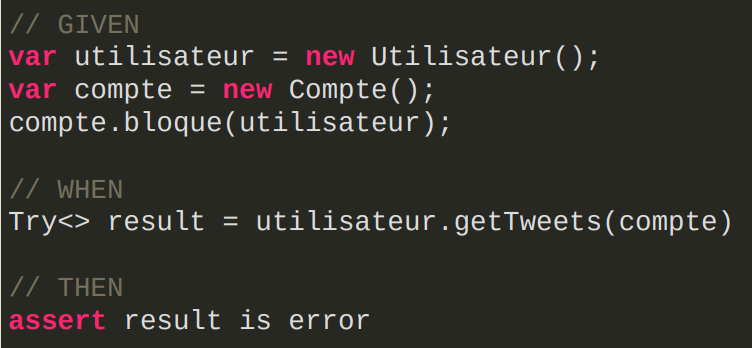
* + Ecrits avec du code
  + S'exécutent **très rapidement** (quelques ms par test -> plusieurs secondes au total)
  + Ecrits pour les programmeurs, par les programmeurs
  + En général testent de la technique mais peuvent tester du fonctionnel
  + Test un **petit montant de code** : en général méthode/fonction ou classe
  + Pas de consensus sur ce qui est une "unité"
  + Exemple : le resultat d’une fonction basique comme condenser(30000000) = « 30M »

Une image contenant texte

Description générée automatiquement

### Tests d'intégration / service

* + Vérifient le **bon** **assemblage** de plusieurs composants (plusieurs classes/modules)
  + Ecrits pour les programmeurs (souvent par les programmeurs)
  + Exemple : un utilisateur bloqué ne peut pas accéder au site



### Tests end-to-end

* + Vérifient le **bon fonctionnement du logiciel**
  + Ecrits pour le client et les programmeurs
  + Exercent l'UI **tel un utilisateur**
  + Font intervenir des composants tiers (web service)
  + S'exécutent **plus lentement** (plusieurs minutes au total)
  + Exemple : Etant donné un utilisateur abonné à un compte et étant sur la timeline sur l'application mobile, Lorsqu'un nouveau tweet est publié sur ce compte, Alors une notification indiquant le nouveau tweet devrait être affichée

Une image contenant texte

Description générée automatiquement

### Tests de fumée (Smoke test)

* + Vérifient le **logiciel** dans les **cas simples**
  + Le nom provient du hardware

### Tests de performance

* + Utilisation processeur
  + Utilisation mémoire
  + Utilisation disque
  + Montée en charge

# Implémentation

@Test void un\_cas\_de\_test() { // GIVEN ... // WHEN ... // THEN ... }

import org.junit.jupiter.api.Assertions;

Assertions.assertEquals(expectedValue, actualValue);

Assertions.assertArrayEquals(expectedArray, actualArray);

Méthodes executees avant/apres tous les tests :

Une image contenant texte

Description générée automatiquement

Méthodes exécutées avant/apres chaque test :

Une image contenant texte

Description générée automatiquement