

AssertJ & tests unitaires

Joël Costigliola
Développeur java freelance

@JoCosti

AssertJ : historique

- Créé en mars 2013
- Pourquoi avoir forké Fest Assert ?
 - Projet gelé
 - Peu ouvert à la communauté
 - Volonté de réduire le nombre d'assertions
- Philosophie d'AssertJ
 - Offrir un grand nombre d'assertions
 - Community driven !

AssertJ assertions

- Spécifique au type testé
- Facile à utiliser :
 - IDE friendly / Auto discovery / little details
- Proche du langage naturel
- Documentées (javadoc et assertj.org)
- Point d'entrée unique : classe Assertions
- Message d'erreurs multilignes à la mockito
- Soft assertions : lister toutes les erreurs d'un test
- Lenient equals assertions

AssertJ modules

- AssertJ modules:
 - assertj-core : assertions pour les types du JDK
 - assertj-guava : assertions pour Guava
 - assertj-joda-time : assertions pour Joda Time
 - assertj-neo4j : assertions pour Neo4J (Florent Biville)
 - assertj-examples : executable doc
 - en prévision : assertj-swing (Christian Rösch)

Exemple

```
// assertions simple
assertThat(gandalf.getName()).isEqualTo("Gandalf");

// il y a 3 anneaux elfiques dans le Seigneur des Anneaux
List<Ring> elvesRings = newArrayList(vilya, nenia, narya);

// assertions sur les collections
assertThat(elvesRings).isEmpty()
                    .hasSize(3)
                    .contains(nenia, vilya, narya)
                    .doesNotContain(oneRing);
```



Démonstration !

**Code source sur Github dans le projet
assertj-examples - branche octo - package octo**

<https://github.com/joel-costigliola/assertj-examples>



AssertJ : extensions

- Utilisation de Condition
- Écrire des assertions pour ses propres classes
- Générateur d'assertions :
 - Se base sur les propriétés de vos classes métiers
 - Se lance avec maven
 - Gain de temps important lorsque l'on veut écrire des assertions métiers pour un grand nombre de classes
 - Plugin eclipse/idea prévu

AssertJ : limites

- Etendre les assertions des types existants :
 - Faisable mais difficulté sur le point d'entrée unique
- Fournir trop d'assertions peut perdre l'utilisateur
 -
- Antipattern : `assertThat(1 == 2)`
- Messages d'erreur : à mieux intégrer dans les IDE
- Assertions sur les exceptions perfectibles
 - Utiliser : <http://code.google.com/p/catch-exception/>

Tests

- Utiliser des librairies :
 - assertj:)
 - mockito
 - ...
- Code coverage : pour voir ce que l'on n'a pas testé
- Si on manque de temps :
 - Tester le code commun
 - Tests de plus haut niveau (fonctionnel / intégration)Code coverage : pour voir ce que l'on n'a pas testé