33 things you want to do better

Speaker : **Tom Bujok**, Suisse – personal blog sur reficio.org

Devoxx France 2014

Vendredi 17/04/2014 14h35 – 15h25

1. Lombok
    a. Génération de hashCode, equals .. à partir de quelques annotations (ex : @ToString).
    b. Plugin pour Eclipse et IntelliJ
    c. Permet d'éviter le boilerplate code
    d. L'annotation @Builder permet de générer Person.builder().name("James")
    e. L'annotation @Log4j permet d' « injecte » un logger : log.info
    f. L'annotation @Cleanup permet de simuler le try with resources de Java 7
        i. @Cleanup InputStream in = new FileInputStream(input)
2. Guava
    a. Intérêt de la classe Optional
        i. Pratique avec les Map pour différencier une valeur vide ou un élément non présent
    b. Pour la validation, utiliser la classe Preconditions
        i. Preconditions.checkState("initialized") ;
        ii. Preconditions.checkNotNull
    c. Multimap
        i. Map<String, List<Long>> => Multimap<String, Long>
        ii. Map<String, Map<String, Long> => HashedBasedTable
3. LambdaJ
    a. Permet d'utiliser des lambdas avant Java 8
    b. filter(having(on(Person.cass).getAge(), greaterThan(30), list) ;
    c. Closurer println = closure() ; { of(System .out).println(var(String.class)) ; }
        i. Le ; est nécessaire
4. Log4j
    a. Attention au log.error(ex) qui effectue un toString() mais n'affiche pas la pile d'appel
    b. Utiliser SLF4J qui ne compile pas cette méthode
5. Bad defaults
    a. Dans un catch() : e.printStrackTrace() ; System.exit(1) ; new RuntimeException()
    b. Public partout : mauvais usage. Commencer par mettre du private. Surtout dans les API
    c. Références : Effective Java, Clean Code
6. Spock
    a. Framework de tests en Groovy
    b. Annotation @Unroll avec un bloc expect : et un autre where : décrivant les entrées et les résultats attendus
    c. setup: when: then:
7. Unitils
    a. Annotation @FileContent(value="/input.txt", encoding="UTF-8")
    b. Annotation @TempFile("output.txt")
    c. assertReflectionEquals(user1, user2)

    d. assertReflectionEquals(asList(1, 2), list, LENIENT)
    e. Annotation @InjectInto
    f. Annotation @InjectIntoStatic
8. JUnitParams
    a. Petite librairie permettant de simplifier l'utilisation de JUnit
    b. Annotation @Paramerters
    c. Paramètre $
9. Awaitility
    a. Permet de tester du code asynchrone
    b. await().atMost(10, SECONDS).untilCall(to(userRepository).size(), equalTo(3)) ;
10. Byteman
    a. A utiliser dans les tests pour changer le bytecode du code à tester, par exemple pour lever une exception.
    b. Utilisation de l'annotation@BMRule(name, targetClass, targetMethod, action)
11. Groovy
    a. XML Parsing
        i. en Java afficher une liste de nœuds prend une 12 aine de lignes de code
        ii. une seule ligne en Groovy avec XmlSlurper() : permet de naviguer dans l'arbre XML (ex : it.symbol.text())
    b. Grape
        i. Les annotations @Grapes et @Grab permettent de récupérer les dépendances maven utilisées par le script Groovy. Le script est donc autonome
    c. Gradle
        i. Moins verbeux que maven
12. Tools
    a. Git : 20 commandes suffisent pour pouvoir utiliser Git
        i. Tom utilise git-svn tous les jours
    b. Bash : regex, awk, sed, grep, sort
    c. Babun project : framework par-dessus cygwin
        i. 1ère release le 23 avril 2014 sur babun.github.io
        ii. Installe tout seul cygwin
        iii. Ajoute des packages manager : apt-get, yum …
        iv. Très stable selon Tom
    d. IDE : apprendre les raccourcis
        i. Evoque IntelliJ et son plugin permettant d'apprendre le shortcuts
13. Code reviews
14. Wrap-up : rester dans la course, veille sur les outils …