POLITECNICO
MILANO 1863

Politecnico di Milano

A.A. 2016 – 2017

Software Engineering 2: "PowerEnJoy"

Code Inspection Document

Marco Festa

February 01, 2017

Contents

1 Introduction

1.1 Purpose

In this document we are asked to report on the quality status of the assigned code extract using the specified Java code inspection list.

1.2 Assigned Project and Class

The Java class assigned for code inspection belongs to the **Apache OFBiz** open source project.

(http://mirror.nohup.it/apache/ofbiz/apache-ofbiz16.11.01.zip)

The specific class/package assigned is:

/framework/entityext/src/main/java/org/apache/ofbiz/entityext/eca/**EntityEcaRule.java**

1.3 Reference Documents

- CodeInspectionAssignmentTaskDescription.pdf
- ApacheOFBizCookbook.pdf

2 Functional Role

OFBiz is an open source ERP system which integrates a suite of enterprise applications to automate a variety of business processes.

The class assigned is part of the OFBiz **Entity Engine**. The Entity Engine is a set of tools and patterns used to model and manage specific data. The core idea is to build relations between different entities and their assigned fields.

Ignoring the specifics on how an Entity is represented and used inside the OFBiz framework it's interesting to understand how the system allows us to trigger a specific action according to a said change in our entity (e.g. a field value is updated). This is achieved through **Entity Event Condition Actions** (**EECA**). Our class is in fact the object representing EECAs.

Some of the attributes of our class are:

- entityName: the name of the entity the EECA is assigned to.
- operationName: the name of the operation performed by the EECA.
- eventName: the name of the event triggering the EECA.
- conditions: a list of the conditions triggering the EECA.
- actionsAndSets: a list of actions to perform upon trigger.

EECA rules are checked before and/or after a call to access the database. Services defined by the action element are invoked when the configured entity is accessed by way of the operation attribute setting.

3 List of Issues

To better understand the meaning of each code requirement listed below refer to the Code Inspection Assignment Document.

3.1 Naming Conventions

1. **Meaningful names:** ✓
2. **One-character variables:** ✓
3. **Class names:** ✓
4. **Interface names:** ✓
5. **Method names:** ✓
6. **Class variables:** ✓
7. **Constants:** ✓

3.2 Indentation

8. **Number of spaces:** ✓
9. **Tabs for indentation:** ✓

3.3 Braces

10. **Consistent bracing style:** ✓
11. **One-line statement bracing:** ✓

3.4 File Organization

12. **Blank lines separation:** ✓
13. **Line length under 80 chars:** some lines exceed the 80 characters limit, this are long method declarations or debug messages.
14. **Line length under 120 chars:** only debug messages exceed the 120 characters limit.

3.5 Wrapping Lines

15. **Line break after a comma or operator:** ✓
16. **Higher-level breaks:** ✓
17. **Statements aligned to previous one:** ✓

3.6 Comments

18. **Comments correct use:** some comments are unclear and badly aligned (lines **126**, **160**, **161**).
19. **Commented code:** line **118** is commented without a clear and explained reason. No date is assigned to suggest a possible comment removal date.

3.7 Java Source Files

20. **Single public class or interface:** ✓
21. **First class in file:** ✓
22. **Consistent javadoc interfaces**: there is almost no javadoc at all in the whole document.
23. **Javadoc is complete:** javadoc is completely missing except for one single method marked as deprecated.

3.8 Package and Import Statements

24. **Package and import statement order:** ✓

3.9 Class and Interface Declarations

25. **Class declaration order:** the order is correct but there are no comments describing the different sections or attributes.
26. **Methods valid grouping:** ✓
27. **No duplicates or long elements:** ✓

3.10 Initialization and Declaration

28. **Variables/methods of correct type and visibility:** ✓
29. **Proper variables declaration:** ✓
30. **Constructor calling:** ✓
31. **Object references initialization:** ✓
32. **Variable initialization:** ✓

33. **Declarations position:** declaration position doesn't always appear at the beginning of a code block (lines **127**, **145**). This is still acceptable since the method could terminate without using these variables at all due to previous checks.

3.11 Method Calls

34. **Methods parameters order:** ✓
35. **Correct methods are called:** ✓
36. **Return values use:** ✓

3.12 Arrays

37. **Off by one errors:** since only iterable objects are used in loops there is no risk to face off-by-one errors.
38. **Out of bound indexing:** ✓
39. **Constructors calls:** ✓

3.13 Object Comparison

40. **Object comparison with .**equals**:** the only use of == instead of .equals expression (lines **123**, **129**) is to compare integers and null values.

3.14 Output Format

41. **Displayed output correctness:** the only debug messages (lines **83**, **84**, **75**, **114**, **164**) are correctly written and spelled. They are clear and explain the outcome of the operation.
42. **Error messages correctness:** ✓
43. **Alignment and spacing correctness:** ✓

3.15 Computation, Comparisons Assigment

44. **Avoided "brutish" programming:** ✓
45. **Computation/evaluation order:** ✓
46. **Operators parenthesis problems:** ✓
47. **Possible zero divisions:** ✓

48. **Truncation or rounding avoidance:** ✓
49. **Boolean operators are correct:** ✓
50. **Throw-catch expressions:** ✓
51. **Implicit type conversions:** ✓

3.16 Exceptions

52. **Relevant exceptions are caught:** ✓
53. **Proper action for each catch block:** ✓

3.17 Flow of Control

54. **Break or return in switch statements:** ✓
55. **Default branch in switch statements:** ✓
56. **Correctly formed loops:** ✓

3.18 Files

57. **Files declaration:** ✓
58. **Files closing procedure:** ✓
59. **EOF conditions detected:** ✓
60. **File exceptions:** ✓

4 Other Problems

No other relevant problem was found. The lack of Javadoc and general available documentation (official technical guide) make the process of code inspection more difficult and less accurate.

5 Appendix

4.1 Tools

- MS Word for the whole document.

4.2 Effort spent

Marco Festa: 25 hours

4.3 Revisions

- CID v1.0 published February 01, 2017.