Politecnico di Milano
MSc Computer Science and Engineering

Software Engineering 2
Academic Year 2016-2017

Code Inspection Document

PowerEnJoy

Authors:

Melloni Giulio 876279
Renzi Marco 878269
Testa Filippo 875456

Reference Professor:
Mottola Luca

Release Date: February 5th, 2017
Version 1.0

Table of Contents

1 | Code description

1.1 Assigned class

1.2 Functional role

2 Code issues

2.1 Notation

2.2 Checklist issues

2.2.1 Naming Conventions

Line	*Issue*
602 610 611	Variables without a meaningful names
67 68	Constants are not declared using all uppercase with words separated by an underscore

Other variables names, methods names and class name are used properly and have a meaningful name.
Only *throwaway* are sometimes composed by a one-character word.
Class name is written with the first letter in capitalized and method names are verbs with the first letter of each addition word capitalized.
Variables, methods and class are written with the camel notation.
The other constant is written using all uppercase with words separated by an underscore.

2.2.2 Indention

Line	*Issue*

2.2.3 File Organization

Line	*Issue*

2.2.4 Wrapping Lines

Line	*Issue*

2.2.5 Comments

Line	*Issue*
204 240 287 313 314	Comments used doesn't explain anything more than what the code say

The remaining part of the code is not commented sufficiently: there could be more lines which explain the function of a code block, especially in the critical parts.
The other lines of code are meaningful and explain what the code is doing correctly.

2.2.6 Java Source File

Line	*Issue*

2.2.7 Package and Import Statements

Line	*Issue*

2.2.8 Class and Interface Declarations

Line	*Issue*
393 621 684	Methods aren't grouped by functionality, scope or accessibility to the other close to it
95 201 443	Methods are long. They should be divided into more sub-methods that are maybe useful to reuse i

Other class, variables and methods positions are respected and are grouped by functionality rather than by scope or accessibility.
Moreover, the code doesn't contain any duplicates.

2.2.9 Initialization and Declarations

Line	*Issue*

2.2.10 Method Calls

There is no error to underline.
Every method has parameters presented in the correct order.
When a method is called, it is called the right one, although there are similar names.
Return value of the method is used properly in each case.

2.2.11 Arrays

Line	*Issue*

2.2.12 Object Comparison

Every object comparison is done with the *java* method **equals()** and not with the == or the !=.
The object is correctly compared to something with the == when it is important to see if the variable is null.

2.2.13 Output Format

Line	*Issue*

2.2.14 Computation, Comparisons and Assignments

Line	*Issue*

2.2.15 Exceptions

Line	*Issue*
142 328	Exception used isn't meaningful

Other exceptions are used correctly and have a correct meaning that helps to find the problem of different program routine.

2.2.16 Flow of Control

Line	*Issue*

2.2.17 Files

Line	*Issue*

2.3 Other issues

This section underline problems that are not specified in the *Checklist issues* because of their importance.

Line	*Issue*
285 367	TODO must be commented differently with the notation *TODO* so that every IDE can find it ea
360	It is better to have near the same type of parameters. In this example, *String* ones are divided
694	Eliminate wrapping lines that aren't used to visualize the code better

3 | Effort Spent

4 | Revision History