

PowerEnJoy

Code Inspection

A.Y 2016/2017

Francesco Tinarelli (matr:806146)

Marco Wenzel (matr:878021)

Versione 1.0

**Index**

1) Introduction------------------------------------------------------------------3

1.1) Purpose---------------------------------------------------------------3

1.2) Classes Assigned ---------------------------------------------------3

2) EntityDataReader inspection -------------------------------------------4

2.1) Functional role ----------------------------------------------------4

2.2) Analysis method ---------------------------------------------------4

2.3) Code ------------------------------------------------------------------5

2.4) Issues ----------------------------------------------------------------6

3) SetOperation inspection --------------------------------------------------6

3.1) Functional role ----------------------------------------------------6

3.2) Analysis method --------------------------------------------------7

3.3) Code ------------------------------------------------------------------7

3.4) Issues ----------------------------------------------------------------10

4) Appendix --------------------------------------------------------------------11

4.1) Hours of work -----------------------------------------------------11

1. Introduction

1.1 Purpose

Code inspection is the systematic examination (often known as peer re-

view) of computer source code. It is intended to find mistakes overlooked

during the initial development phase, with the aim of improving both the

overall quality of software and the developers' skills.

1.2 Classes Assigned

* **Name**: EntityDataReader
  + **Location:** /apache-ofbiz-16.11.01/framework/entity/src /main/java/org/apache/ofbiz/entity/config/model
  + **Package:** org.apache.ofbiz.entity.config.model
  + **Modifier:** public-final
  + **Method:** EntityDataReader(String), EntityDataReader(Element), getName, getResourceList
* **Name**: SetOperation
  + **Location**: /apache-ofbiz-16.11.01/framework/minilang/

src/main/java/org/apache/ofbiz/minilang/method/envops

* + **Package:** org.apache.ofbiz.minilang.method.envops
  + **Modifier:** public final
  + **Method:** autoCorrect, SetOperation, exec, toString, <inner class> SetOperationFactory

2. EntityDataReader inspection

2.1 Functional role

For understanding the functional role of this class, first we need to understand what are nodes and elements.

The node is the smallest datatype in the DOM, is an interface implemented by almost all object.

The element is a type of node, is an interface that extends the node interface, this type of node corresponding to structure in XML pages.

The EntityDataReader class have two constructors:

The first one permits through a string variable to insert a value on the variable: name; and provide an empty list of resource.

The second one permits through an Element to extract all children elements node that they have the same type of node (element node) and with the node name equals to “resource”. After the extraction insert for each element the respective resource object into the resource list.

For control the unexpected results this class has an exception if the EntityDataReader have already a name value or if the content of the parameter: element is empty.

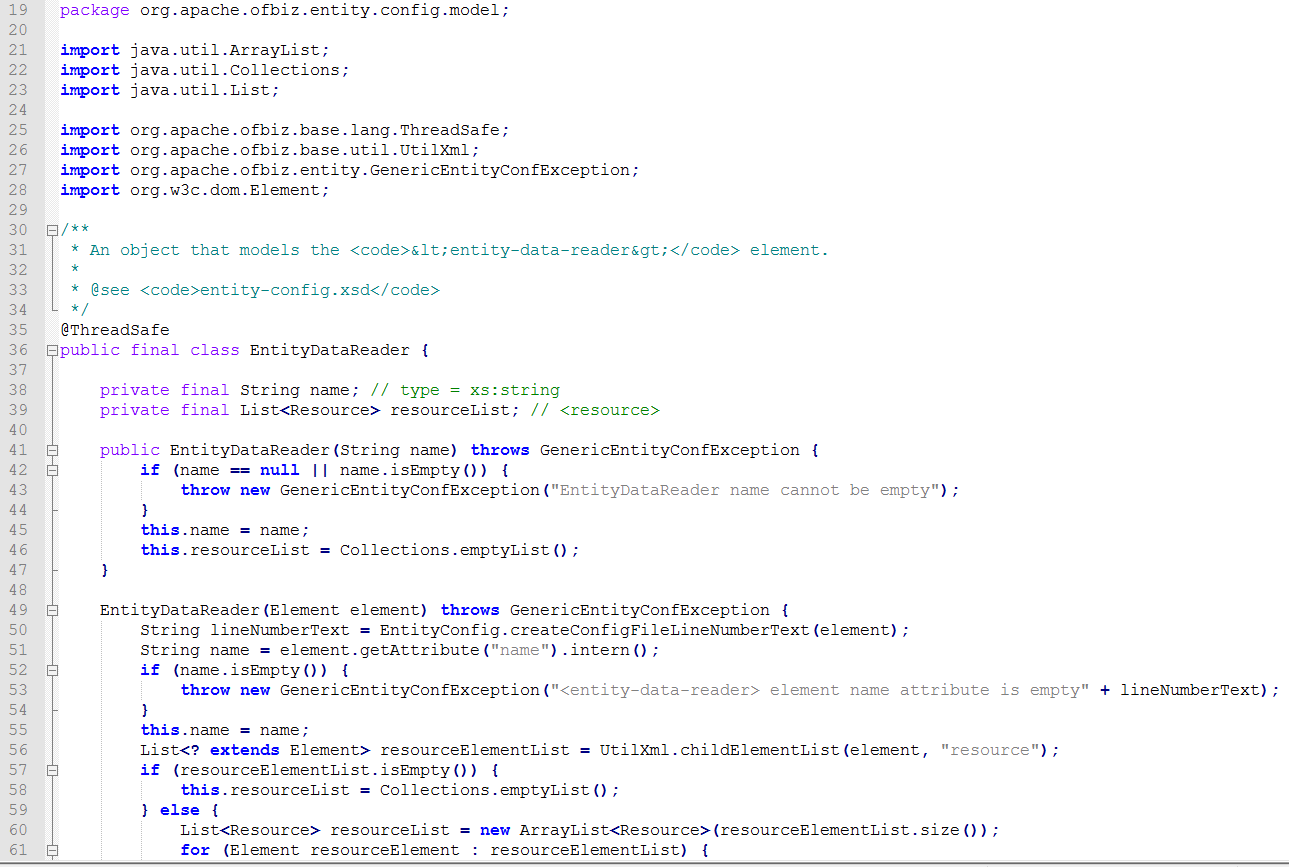
So, this class should be use for capture the content and the data structure of a specific XML file.

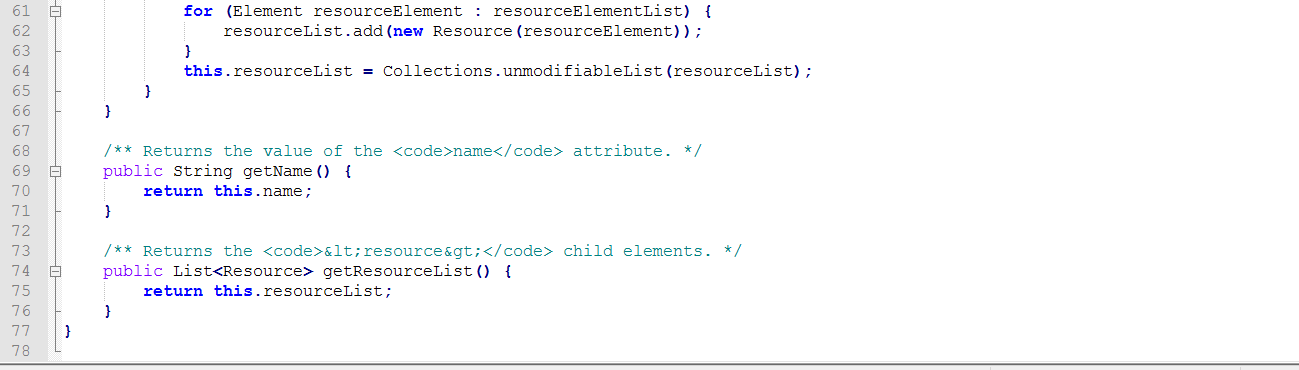
2.2 Analysis method

For analyze this class we start from analyze the modifier of its attributes and methods than the double constructor and the external method calls.

To understand the intrinsic meaning, we use the OfBiz documentation but this work was hard because in this class miss the Javadoc.

2.3 Code





* 1. Issues

The numbers of the issues refer to the corresponding numbers in the ispection checklist.

1) createConfigFileLineNumberText(element) this method doesn’t have a meaningful name because returns a string that indicates the number of the starting line a more meaningful name can be startingLineNumber(element).

13) Line 53 (110c) Line 56 (92c) Line 60(83c).

18) There are no comments about code are doing.

19) No comment for explain the reasoning behind the code.

23) There are no Javadoc about this class.

27) There is double constructor but they are control well and the variable: name is defined two times but the two variables are not the same and is controlled well(this.name=name).

36) lineNumberText variable is not used if the variable: name is empty.

56) That FOR type is correct but we don’t know the initial value.

3. SetOperation inspection

3.1 Functional role

This class takes an element and first correct the deprecated value with the autoCorrect() method, then this class “opens” the attribute searching script or operation and puts them into the variables. This searching of the operation is done by using Scriptlet, flexibleMapAccessor and flexibleStringExpander.

First the class searches the Script form the “from-attribute”, second uses the flexibleMapAccessor on the same attribute, third searches from the “value” attribute and finally uses the default attribute.

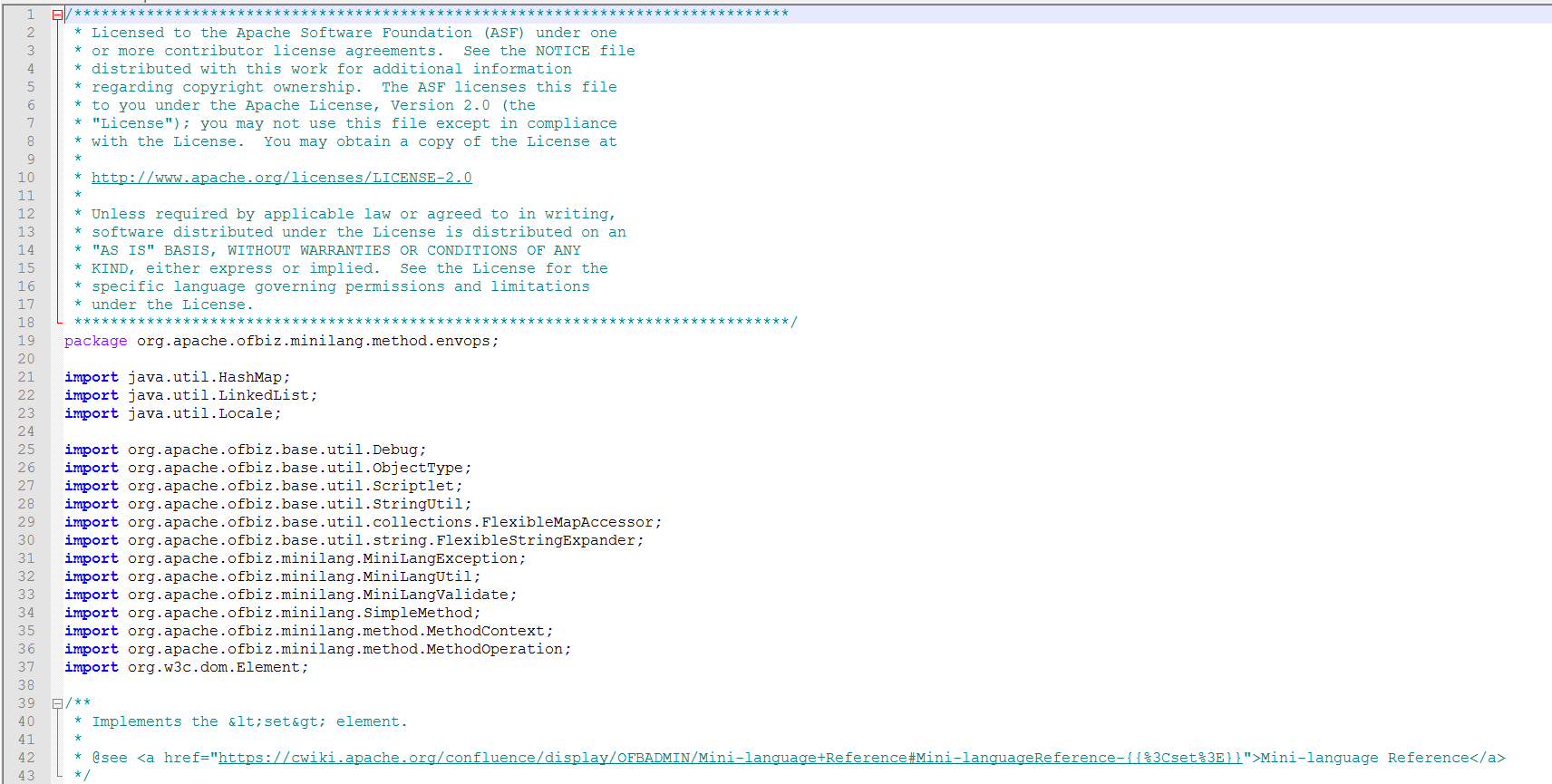
The method exec executes the operation that are in the variables. This method return, true if script execution should continue, or false if script execution should stop. For the control of the execution this class use the exception.

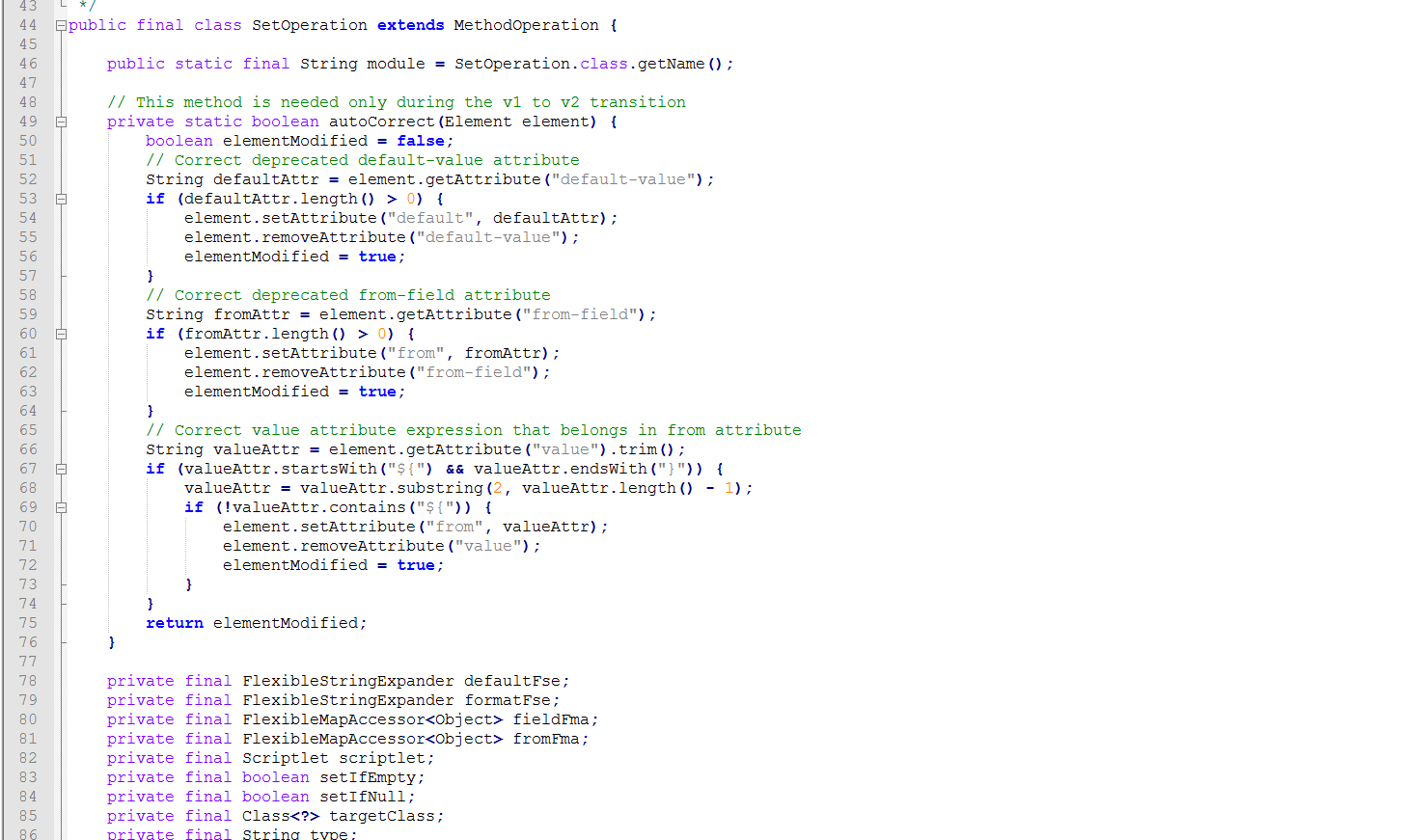
An object SetOperation can be created by the object factory that is an override of the method that is in the MethodContext superclass.

3.2 Analysis method

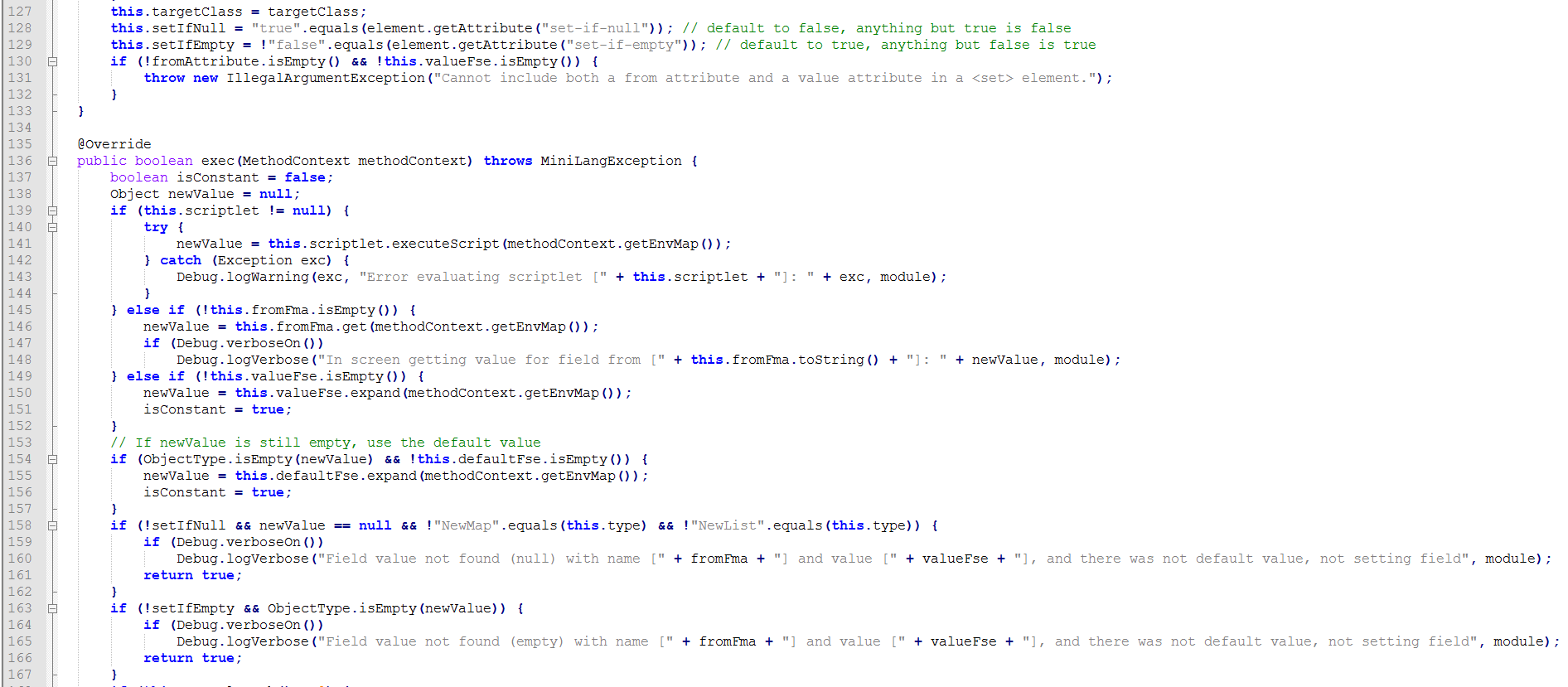
For this class we do a top-down analysis. We start from understand how is the role of this class in high terms, second we analyze the role of every attributes of the class, entering in detail. Than we analyze the methods and the return of this method. At last we control all the variables and if that are useless, the exception lauched, and the right use of spaces and braces control.

3.3 Code

****

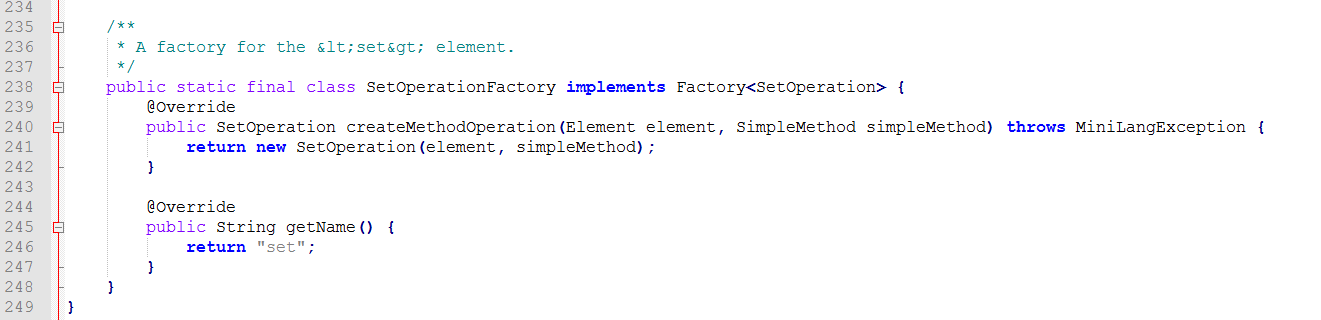












3.4 Issues

1) SetOperation it’s a mining full name, don’t explain what the class do.

11) Line 147-159-164-197 one statement without graph braces.

13) Line 94, line 128, line 129, line 143, line 148, line 160, line 165, line 187, line 190.

14) Line 94(172c) Line 160(166c) Line 165(168c) Line 187(123c) Line 190(182c).

18) The comment at the beginning of the class is useless, the internet page that the URL of this document see is moved to another page.

19) All the comment out of code are without the date.

23) There are no Javadoc about this class.

25) AutoCorrect() method came first than private final variables.

33) Line 107: variable stringAttribute.

40) Line 42, line 158, line 180.

4. Appendix

4.1 Hours of work

**Marco Wenzel**: 10 hours.

**Francesco Tinarelli**: 8 hours.