

POLITECNICO DI MILANO

2016-2017

Software Engineering 2

Code Inspection

Version 1.0

Peverelli Francesco

Reppucci Federico

Released on:

**TABLE OF CONTENT**

|  |  |
| --- | --- |
| 1. ASSIGNED CLASSES  2. FUNCTIONAL ROLE OF ASSIGNED SET OF CLASSES  2.1 INTRODUCTION  2.2 CONSTRUCTOR  2.3 THE EXEC METHOD  2.4 OTHERS  3. LIST OF ISSUES FOUND BY APPLYING THE CHECKLIST  3.1. NAMING CONVENTIONS  3.2. INDENTION  3.3. BRACES  3.4. FILE ORGANIZATION  3.5. WRAPPING LINES  3.6. COMMENTS  3.7. JAVA SOURCE FILES  3.8. PACKAGE AND IMPORT STATEMENTS  3.9. CLASS AND INTERFACE DECLARATIONS  3.10. INITIALIZATION AND DECLARATIONS  3.11. METHOD CALLS  3.12. ARRAYS  3.13. OBJECT COMPARISON  3.14. OUTPUT FORMAT  3.15. COMPUTATION, COMPARISONS AND ASSIGNMENTS  3.16. EXCEPTIONS  3.17. FLOW OF CONTROL  3.18. FILES |  |

**1. ASSIGNED CLASSES**

*Class path:* apache-ofbiz-16.11.01/framework/minilang/src/main/java/org/apache/ofbiz/

minilang/method/envops/SetCalendar.java

*Class name:* SetCalendar

**2. FUNCTIONAL ROLE OF ASSIGNED SET OF CLASSES**

**2.1. INTRODUCTION**

The SetCalendar class is part of the minilang section of the ofbiz framework. Minilang is a language defined as an XML schema, used specifically to code events and services to use within the framework. It aims to ease the development process by allowing whoever is writing the code to operate at a higher level, without needing to worry about declaring variables and managing data structures. The SetCalendar class defines an operation which allows to create a new timestamp by adding the desired amount of time to another timestamp (“from” attribute) or to a specified constant (“value” attribute), and storing it in a specified destination (“field” attribute). For more detail on the syntax see:

<https://cwiki.apache.org/confluence/display/OFBADMIN/Mini+Language+-+minilang+-+simple-method+-+Reference>

**2.2 CONSTRUCTOR**

The class constructor starts with the validation of the syntax of the element, if the validation option is on, and notifies if any problem arises. Then, the autoCorrect method of the class is called, to modify any deprecated syntax in the name of the attributes and correct if any expression belonging to the “from” attribute has been placed in the “value” attribute. In the following section it is established whether the “from” attribute will be specified as an object returned by a script or as an object present in a map structure, and auxiliary objects are instanced accordingly. If no “from” value is specified, the value of the “value” attribute is retrieved. If both are present, an exception is thrown. Afterwards, the value of the attributes representing the time to add to the timestamp is retrieved, as well as the settings to align the resulting value either to the start or the end of the date’s year, month, week or day, the local calendar and the local time zone.

**2.3 THE EXEC METHOD**

The exec method is the method which executes the operation described in the minilang command. It starts by retrieving the value of the “from” attribute, either by executing a script or by retrieving the object from a map structure, and assigning it to a “newValue” attribute. If the “from” attribute is not present, the value of the “value” attribute is assinged newValue. If newValue is still empty but a default value is available, the default value is assigned. If no default value is available and the attribute “setIfNull” is false, the method returns. After that, the variables representing time constants are declared and initialized to ‘0’, and the local calendar and time zone to null. Moreover, a timestamp variable “fromStamp” is also initialized to null. After the initialization, the values specified in the set calendar element are assigned to all the variables declared above, and the value of fromStamp is used to create a calendar object, add the specified time constants to it, taking into account the local calendar and time zone, and the returned value is assigned to a “toStamp” variable. Finally, the desired period alignments are performed, and the value of toStamp is put in a map structure, and the method returns.

**2.4 OTHERS**

The toString method and the SetCalendarFactory are both well-known java constructs and behave as expected.

**3. LIST OF ISSUES FOUND BY APPLYING THE CHECKLIST**

**3.1. NAMING CONVENTIONS**

*All class names, interface names, method names, class variables, method variables, and constants used should have meaningful names and do what the name suggests.*

• classes: no problems detected.

• interfaces: no problems detected.

• methods: no problem detected.

• class variables: no problem detected.

• method variables: no problem detected.

• constants:

*If one-character variables are used, they are used only for temporary “throwaway” variables, such as those used in for loops.*

*Class names are nouns, in mixed case, with the first letter of each word in capitalized. Examples: class Raster; class ImageSprite;*

*Interface names should be capitalized like classes.*

*Method names should be verbs, with the first letter of each addition  word capitalized. Examples: getBackground(); computeTemperature().*

*Class variables, also called attributes, are mixed case, but might begin with an underscore (‘ ’) followed by a lowercase first letter. All the remaining words in the variable name have their first letter capitalized. Examples: windowHeight, timeSeriesData.*

*Constants are declared using all uppercase with words separated by an underscore. Examples: MIN WIDTH; MAX HEIGHT.*