POLITECNICO
MILANO 1863

Software Engineering 2: "PowerEnJoy"

Code Inspection Document

Version 1.0

Piccirillo Luca - 790380
Zampogna Gian Luca - 863097
Zini Edoardo - 875275

February 5th, 2017

Contents

1 Classes assigned to the group

Source: `http://mirror.nohup.it/apache/ofbiz/apache-ofbiz-16.11.01.zip`

- ../apache-ofbiz-16.11.01/framework/common/src/main/java/org/apache/ofbiz/common/UrlServletHelper.java
- ../apache-ofbiz-16.11.01/framework/minilang/src/main/java/org/apache/ofbiz/minilang/method/envops/SetCalendar.java

2 Functional role of assigned set of classes

The inspection phase started with an attempt to understand the goal of the Apache OFBiz Framework. Given the lack of javadoc all our understanding has been based on official framework wiki pages and a careful reading of the code itself which contains a little amount of short comments.
After some researches, it turns out that the Open For Business Framework is intended to speed-up the development of FOD (*Forms-Over-Data*) web-based applications where most components (i.e. forms, views and data entities) are defined by XML files that matches the "minilang" schema.
Each element of the schema maps to a minilang statement or method which is actually implemented in Java extending different appropriate parent classes. To extend the built-in operations of minilang, Groovy scripts inclusion is supported.

2.1 UrlServletHelper class

File `UrlServletHelper.java`
Part of `org.apache.ofbiz.common` package.

`UrlServletHelper` is a container of static methods which handles URL to resource mapping implementing parsing and validation of the HTTP query string. It also handles resource mapping in multi-tenant environments where the same app works on different datasources.

2.1.1 UrlServletHelper constructor

```
private UrlServletHelper()
```

Default private constructor prevents this class from being instantiated outside of its context. There exist no calls to this constructor in the analyzed file version.

2.1.2 setRequestAttributes method

```
public static void setRequestAttributes(ServletRequest request, Delegator
delegator, ServletContext servletContext)
```

Populates the fields in the `HttpServletRequest` extension of the passed `request` object. It also updates fields in the `servletContext` in case multi-tenant mode is enabled to perform tenant selection based on URL domain name.

2.1.3 setViewQueryParameters method

`public static void setViewQueryParameters(ServletRequest request, StringBuilder urlBuilder)`

Parses the http querystring in `request` token by token and builds a new URL containing all parameters needed by the view to show results according to passed filters.

2.1.4 checkPathAlias method

`public static void checkPathAlias(ServletRequest request, ServletResponse response, Delegator delegator, String pathInfo)`

Checks whether exists a resource that is reachable by the alias specified in `pathInfo`. If it exists the request is forwarded to the right resource otherwise if the specified path belongs to a resource that needs to be reached by an alias, an HTTP `404 Not Found` error is returned to the client. If none of the previous conditions are met, nothing changes and the method returns gracefully.

2.1.5 invalidCharacter method

`public static String invalidCharacter(String str)`

Removes and trims invalid or dangerous characters from the input `str`.

2.2 SetCalendar class

File `SetCalendar.java`
Part of `org.apache.ofbiz.minilang.method.envops` package.

In minilang, the `SetCalendar` operation extends the Set operation providing the ability of adjusting the input timestamp by a specified timespan. It accepts as input data either a mix of expressions and constants or a script. This class actually implements those mechanics by parsing the corresponding `<set-calendar/>` minilang XML element and attributes, than adding some parameters validation.
Examples of `set-calendar` operation usage:

```
<set-calendar field="tomorrowStamp" from-field="nowTimestamp" day="1"/>
<set-calendar field="yesterdayStamp" from-field="nowTimestamp" day="-1"/>
```

2.2.1 autoCorrect method

```
private static boolean autoCorrect(Element element)
```

Provides compatibility with old minilang schema versions for the `element`. A comment states that this method should be deprecated after the transition to the new schema version.

2.2.2 parseInt method

```
private static int parseInt(String intStr)
```

Addresses a compatibility issue with older Java versions.

2.2.3 SetCalendar constructor

```
public SetCalendar(Element element, SimpleMethod simpleMethod) throws Mini-
LangException
```

Parses and validates the XML `element` which associated operation statement should be implemented by this instance. If validation succeeds all local properties are populated interpreting attributes value.

2.2.4 exec method

```
public boolean exec(MethodContext methodContext) throws MiniLangException
```

Actually executes the operation associated to `set-calendar` statement. Sets the value of the field either to the result of a script, the evaluation of an expression, the value of a constant or its default. In addition to the standard minilang `set` statement, it also manages to adjust the resulting timestamp by interpreting other `set-calendar` attributes.

2.2.5 toString method

`public String toString()`

Returns an XML string containing a `<set-calendar/>` element equivalent to the current instance of the `set-calendar` operation statement.

2.3 SetCalendarFactory class

File `SetCalendar.java`
Part of `org.apache.ofbiz.minilang.method.envops` package.

Implements a factory pattern that generates `SetCalendar` instances by implementing `Factory<T>` interface.

2.3.1 createMethodOperation method

`public SetCalendar createMethodOperation(Element element, SimpleMethod simpleMethod) throws MiniLangException`

Returns an instance of `SetCalendar` which implements the `simpleMethod` described in the minilang XML `element`.

2.3.2 getName method

`public String getName()`

Returns the name of minilang XML element that this factory provides. In this override the returned `String` constantly equals to `"set-calendar"`.

3 List of issues found by applying the checklist

3.1 Naming Conventions

1 Taking into account what is stated in the previous section of this document, both `UrlServletHelper.java` and `SetCalendar.java` contain classes, variables, methods and constants that have meaningful names and do what their names suggest.

2 There is no one-character variable used for non temporary purpose in both `UrlServletHelper.java` and `SetCalendar.java`.

3 `UrlServletHelper.java` class name is a properly formatted noun: mixed case with the first letter of each word in capitalized.
`SetCalendar.java` classes names are not nouns, even if they are properly formatted.

4 No interface is present in both files.

5 In `UrlServletHelper.java` the method `invalidCharacter` (line 205) is not a verb, even if it is properly formatted: mixed case with the first letter in lowercase and all the remaining words in the variable name have their first letter capitalized.
In `SetCalendar.java` methods names are properly formatted verbs.

6 Both `UrlServletHelper.java` and `SetCalendar.java` attributes names are properly formatted: mixed case with the first letter in lowercase and all the remaining words in the variable name have their first letter capitalized.

7 In `UrlServletHelper.java` the constant `module` (line 46) is not written using all uppercase.
In `SetCalendar.java` the constant `module` (line 50) does not match the convention.

3.2 Indention

8 Four spaces are consistently used for indentation in both files.

9 No tabs are used to indent in both files.

3.3 Braces

10 Both files follows the "Kernighan and Ritchie" style.

11 All `if`, `while`, `do-while`, `try-catch`, and `for` statements that have only one statement to execute are surrounded by curly braces.

3.4 File Organization

12 `UrlServletHelper.java` does not contain a blank line between lines 152 and 153. `SetCalendar.java` properly contains blank lines and comments to separate sections.

13 In `UrlServletHelper.java` sixteen lines contain more than 80 characters, but less than 120: line 18 (81 characters), 61 (105), 62 (106), 67 (90), 68 (99), 69 (96), 71 (98), 72 (83), 82 (93), 86 (86), 91 (97), 104 (109), 108 (105), 162 (88), 190 (82), 194 (90).
In `SetCalendar.java` a lot of lines contain more than 80 characters, but less than 120: lines 18 (81 characters), 82 (89), 105 (93), 108 (111), 109 (117), 113 (89), 114 (94), 115 (86), 116 (104), 123 (87), 126(99), 132 (90), 136 (94), 138 (90), 139 (92), 140 (88), 141 (90), 142 (94), 143 (94), 144 (92), 145 (111), 146 (107), 147 (83), 148 (97), 156 (83), 158 (109), 202 (87), 205 (90), 208 (86), 211 (88),214 (92), 217 (92), 220 (90), 223 (117), 235 (85), 237 (81), 239 (82), 241 (83), 243 (82), 245 (114), 248 (83), 258 (112), 303 (84), 306 (82), 324 (82), 326 (119).

14 In `UrlServletHelper.java` six lines contain more than 120 characters: line 50 (121 characters), 58 (133), 63 (147), 87 (135), 153 (127), 191 (127).
In `SetCalendar.java` seven lines contain more than 120 characters: lines 46 (167 characters), 110 (149), 111 (153), 134 (138), 183 (135), 192 (143), 200 (149).

3.5 Wrapping Lines

15 In `UrlServletHelper.java` line breaks of lines from 160 to 164 and from 189 to 192 does not occur after a comma or an operator.
In `SetCalendar.java` line breaks do occur after a comma or an operator.

16 In both files higher-level breaks are used.

17 In both files new statements are aligned with the beginning of the expression at the same level as the previous line.

3.6 Comments

18 In `UrlServletHelper.java` only `setRequestAttributes` is adequately commented, other parts of the class contain no comment or just a very brief comment.
In `SetCalendar.java` only a few pieces of code are commented: `autoCorrect` and `parseInt` methods.

19 No commented out code is present in both files.

3.7 Java Source Files

20 In `UrlServletHelper.java` there is only one public class.
In `SetCalendar.java` there are two public classes: `SetCalendar` and `SetCalendarFactory`.

21 In both files the first class is the public class, or one of the public classes.

22 In both files there are neither interfaces nor javadoc (see next point for further details on javadoc).

23 In `UrlServletHelper.java` there is no javadoc.
In `SetCalendar.java` javadoc covers only the two classes statements.

3.8 Package and Import Statements

24 In both files the first non-comment statements are packages, followed by import statements.

3.9 Class and Interface Declarations

25 In `UrlServletHelper.java` the order is respected, even if not all required points are satisfied due to the lack of some of the listed elements. In particular no variable can be found outside the methods.
In `SetCalendar.java` the order is not respected, this is the actual class order :
- Class documentation comment;
- Class statement;
- Static final attribute (`module`);
- Methods (`autoCorrect` and `parseInt`);
- Class private variables;
- Constructor;
- Methods (`exec` and `toString`);
- Factory class and methods.

26 In `UrlServletHelper.java` after the first method, which is a private constructor, there are three methods all requiring a `ServletRequest` and whose work imply managing that request (i.e. handling resource mapping parsing and validation of the HTTP query string). Differently from the previous ones, the last method just operates on a string.
In `SetCalendar.java` methods seem to be divided in `@override` methods and in non-`@override` methods: the only non-`@override` method is placed before attributes definition, while all the others are placed after constructor.

27 In `UrlServletHelper.java` there are the following long methods: `setViewQueryParameters` (61 lines), `checkPathAlias` (50 lines) and `invalidCharacter` (77 lines).
In `SetCalendar.java` there are the following long methods: `exec` (112 lines) and `toString` (54 lines).

3.10 Initialization and Declarations

28 In both files all variables and class members are of the correct type and have the right visibility.

29 In `UrlServletHelper.java` the variable `httpRequest` (line 154) is used only in the else branch of line 186, so it should have been declared within that scope. The variable `httpResponse` (line 155) is used only in the `if`, line 193, which is within the `else` branch of line 186, so it should have been declared within that scope.
In `SetCalendar.java` the variable `fromStamp` (line 173) is assigned at line 200, so it should be declared within the `try` scope.

30 In `UrlServletHelper.java` constructors are never required.
In `SetCalendar.java` constructors are called when a new object is desired.

31 In `UrlServletHelper.java` all the object are initialized before being used.
In `SetCalendar.java` all the object that are not immediately initialized are initialized before being used.

32 In `UrlServletHelper.java` all variables are immediately initialized.
In `SetCalendar.java` all variables are initialized when they are declared if their value does not depend from a computation.

33 In `UrlServletHelper.java` in nine cases declarations are made in the middle of the block: line 62, 63, 96, 97, 98, 99, 100, 102, 176.
In `SetCalendar.java` in sixteen cases declarations are made in the middle of the block: line 63, 70, 119, 124, 171, 172, 173, 174, 175, 176, 177, 178, 179, 180, 225, 233.

3.11 Method Calls

34 In both files all parameters are presented in correct order.

35 In both files the correct method is always called.

36 In both files all method returned values are used properly.

3.12 Arrays

37 No array is present in either `UrlServletHelper.java` or `SetCalendar.java`.

38 No array is present in either `UrlServletHelper.java` or `SetCalendar.java`.

39 No array is present in either `UrlServletHelper.java` or `SetCalendar.java`.

3.13 Object Comparisons

40 In `UrlServletHelper.java` the `==` operator is used only in line 171 to make a comparison with `null`.
In `SetCalendar.java` the `==` operator is used in lines 168, 185, 188, 194, 197 to make a comparison with `null`.

3.14 Output Format

41 No spelling or grammatical errors detected.

42 In `UrlServletHelper.java` all errors are logged in order to be used for bugfixing by developers, so they are not expected to provide guidance for the user on how to correct the problem.
In `SetCalendar.java` some errors have an error message, while others are only thrown.

43 Both files provide output which is not expected to be formatted.

3.15 Computation, Comparisons and Assignments

44 In `UrlServletHelper.java` examples of "brutish programming" can be found from line 105 to 119, from 122 to 151 and from 205 to 282.
In `SetCalendar.java` no example of "brutish programming" can be found.

45 In both files there is no complex computation.

46 In both files no operator precedence is needed.

47 In both files there is no division, thus there is no denominator either.

48 Both files contains no arithmetic operation concerning non-integer numbers.

49 In both files all comparisons and Boolean operators are correct.

50 In `UrlServletHelper.java` there is no `throw` expression since all exceptions are handled locally by `catch` expression whose error conditions are coherent with the previous operations within the `try` block.
In `SetCalendar.java` there are seven `throw` expressions: line 105, 134, 152, 223, 245, 258, 326.

51 In both files all type conversions are done via explicit casting.

3.16 Exceptions

52 In both files all relevant exceptions are caught.

53 In `UrlServletHelper.java` all the `catch` blocks always call `logError` or `logWarning`, ensuring that exceptions are properly handled.
In `SetCalendar.java` all the `catch` blocks always call a proper action.

3.17 Flow of Control

54 In both files there is no `switch` statement.

55 In both files there is no `switch` statement.

56 In `UrlServletHelper.java` there is only one `for` loop (line 105) which cycles on the element of a `List` the cycle never modifies, so it will always terminate after a finite number of steps. Concerning `while` statements, three of them can be found (line 272, 275, 278), and for all of them their Boolean condition values are affected by the instructions inside the `while` blocks: the instructions in the first two blocks reduce the length of the string the Boolean condition is based on; while the third block of instructions replace `--` with `-`, so as soon as there are no longer `--` the cycle ends. This means all cycles will always terminate after a finite number of iterations.
In `SetCalendar.java` there is no loop.

3.18 Files

57 In both files no file is used.

58 In both files no file is used.

59 In both files no file is used.

60 In both files no file is used.

4 Other problems

4.1 Design

In `UrlServletHelper.java` there is a potential performance issue since the `checkPath-Alias` method contains two queries: the first one is always performed on each method call, the other one it's performed each time the former turns out to be useless. If called on each web request, and most of the paths don't match to an alias, this code uselessly slows down the web page loading.

4.2 Potentially Deprecable Code

In `SetCalendar.java` there is a comment marking the `autoCorrect` method as temporally needed during software upgrade transitions between two versions of the framework, both precedent to the current one. May be reasonable to remove that if it's no longer needed and if there exists no upgrade paths from v1 to current version.

4.3 Spacing

In `UrlServletHelper.java` the three `while` statements (lines 272, 275, 278) are not followed by a blank space between `while` and `(condition)`.

5 Appendix

5.1 References

- https://cwiki.apache.org/confluence/display/OFBIZ/Multitenancy+support
- https://cwiki.apache.org/confluence/display/OFBADMIN/Mini+Language+-+minilang+-+simple-method+-+Reference

5.2 Effort Spent

Teamwork	∼4h
Piccirillo Luca	∼5h
Zampogna Gian Luca	∼8h
Zini Edoardo	∼5h

5.3 Revision History

Version	**Date**	**Changes**
1.0-RC1	05/02/2017	First deadline release.