A.A.
2016/
2017

Code Inspection

VERSION 1.0

PIETRO CROVARI – ENRICO GNECCO

Table of Contents

1. Introduction

In this section of the project we want to face the problem of the Code Inspection; in other terms, given a checklist, we are going to follow each point to standardize the code improving readability and our skills. This will impact on the latter phase of the project; indeed, software maintenance is one of the greatest costs once that the program is running. The aim of this part is to minimize as possible those costs, working on the possible weak points of the code. Better code really improves the whole.

1.1 Classes assigned to the Group

JavaMailContainer.java

Path: ../apache-ofbiz-16.11.01/framework/service/src/main/java/org/apache/ofbiz/service/mail/JavaMailContainer.java

1.2 Functional role of assigned set of classes

The main role JavaMailContainer is to connect to the server and download all the mail of the account that is running the application. It is conceptually divided in three parts: the class itself and two inner classes, namely LoggingStoreListener and PollerTask.

As we can see by the implementation JavaMailContainer performs the following tasks:

- Through the function *start*, as the name suggests, is in charge of booting the process, logging the user and setting all the necessary parameters for the correct functioning of the class. Moreover, it sets a poller on each store the user is related to. On the other hand, the function *stop* stops the processes, destroying all the poller.
- *MakeSession* is necessary to translate the information held in the *clientProps* data structure in an istance of a session, used in the method described above.
- *GetStore,* instead, tries to get the store from the session and connect to it. If the chance goes well it is successful, otherwise an exception is thrown.
- The last function worth of being noticed is *UpdateUrlName*. It takes as input an urlName and a porperties data structures, and return a new urlName, after aving parsed and checked all the information contained in the Properiy object.
- The first inner class is *LoggingStoreListener,* it creates a notification if a StoreEvent has been catched.
- The second is *PollerTask*, it is made by sereval functions:
    - *Run* method calls *checkMessages* function and close the connection with the store;
    - *CheckMesages* open the connection with the store and, if it sees there are some messages not read, it download them;
    - *ProcessMessage* wraps wails in a wrapper and makes them being processed by ServiceMcaUtil class.

2. List of issues found by applying the checklist

2.1 Naming Conventions

1

No problems found

2

No problems found

3

No problems found

4

No problems found

5

- Line 272: *public void notification(StoreEvent event) {*

6

- Line 61 to 74: no class variable begins with "_";
- Line 289, 290: *dispatcher, userLogin* don't begin with "_";

7

- Line 61: *module* violates constant rule;

2.2 Indention

8

No problems found

9

No problems found

2.3 Braces

10

No problems found

11

- Line 185: if rule violated;
- Line 265: if rule violated;
- Line 283: if rule violated;
- Line 363: if rule violated;
- Line 365: if rule violated;
- Line 369: if rule violated;

2.4 File Organization

12

- Line 58: missing comment;
- Line 268: missing comment;
- Line 286: missing comment;

13-14

Lines exceeding 80 charaters:

- 113
- 123
- 135
- 202
- 247
- 256
- 265
- 307
- 357
- 359
- 363
- 365
- 369

2.5 Wrapping Lines

15

No problem found, in the switch case starting in line 274 the line break occurs after the "*:*" symbol, but we do not consider it as an error

16

No problems found

17

No problems found

2.5 Comments

18

- Line 156: not commented method;
- Line 161: useless comment;
- Line 207: not commented method;
- Line 268: not commented new inner class;
- Line 272: not commented new method;
- Lines 286 to 392: no comments are used to explain methods or the class;

In general, it's quite well commented and comprehensible and we wouldn't cut off any comment

19

No problems found

2.7 Java Source Files

20

No problems found, the file contains a class with other two inner classes

21

No problems found

22

Unfortunately, we didn't find any related Javadoc.

23

Unfortunately, we didn't find any related Javadoc.

2.8 Package and Import Statements

24

No problems found

2.9 Class and Interface Declarations

25

No problems found

26

Methods not grouped by functionality in the latter part of the main class

27

No problems found

2.10 Initialization and Declaration

28

No problems found

29

No problems found

30

No problems found

31

No problems found

32

No problems found

33

- Line 111: Sting *runAsUser* declared after some assignments;
- Line 123: List *config* declared in the middle of the block;
- Line 241: *portProps* declared in the middle of the block;
- Line 242: *portStr* declared in the middle of the block;
- Line 337: *totalMessages* declared in the middle of the block;
- Line 345: *messages* declared in the middle of the block;
- Line 344: *profile* declared in the middle of the block;

2.11 Method Calls

34

No problems found

35

No problems found

36

- *Start()* in JavaMailContainer Class: It is Boolean, but, as we can see in lines 134/138, the error of missing Store object is not reported, returning *true* as if the function had worked correctly. An int value could have helped in differentiating cases;
- *ProcessMessage()* method in line 380 should return a Boolean value, according whether the process has been successful or it has thrown an exception;

2.12 Arrays

37

All arrays are accessed only through foreach construct, therefore they are well-treated

38

No problems found

40

No problems found

2.13 Object Comparison

41

No problems found

2.14 Output Format

42

- Line 115: *cannot* is a colloquial term instead of *can't;*
- Line 247: *please check* is missing of the object (e.g. "*it*");
- Line 256: *please check* is missing of the object (e.g. "*it*");
- Line 357: *cannot* is a colloquial term instead of *can't;*
- Line 365: "…is marked *as* seen" (*as* missing);

43

No problems found, fix guidance is missing since all the messages are only debug ones;

44

No problems found

45

No problems found

46

No problems found

47

No division used

48

No problems found

49

No problems found

50

No problems found

51
No problems found

2.16 Exceptions

52
No problems found

53
- Line 310: exceptions thrown not processed at all;

2.17 Flow of Control

54
No problems found

55
- Line 274: *default* condition missing;

56
Only *foreach* construct, therefore no problems in increments and exit conditions;

2.18
Files not used in the code.

3. Any other problem
Analysing this class, we didn't find any criticity that can generate some error, all variables are checked before use, all exceptions are more or less treated as well. Anyway, as we said in the previous checklist, we noticed a general lack of attention in details: even if the code is well written and seems correct, we believe that the total absence of comments and documentation, joined to the presence of all those if clause without the correct brackets will brings a lot of difficulties in what will be the maintenance of the code. Moreover, since it is an open source project, it is even higher the possibility that who is going to modify this file is another person from the author, and this will lead to a huge waste of time in understanding what has already done.