

**Software Engineering 2: “PowerEnJoy”**

**Code Inspection (V. 1.0)**

Authors:

Sergio CAPRARA

Soheil GHANBARI

Erica TINTI

Milan, Italy

05/02/2017

**Table of contents**

[1. Introduction 4](#_Toc474064141)

[1.1 Revision History 4](#_Toc474064142)

[1.2 Purpose and Scope 4](#_Toc474064143)

[1.3 Definitions and Abbreviations 4](#_Toc474064144)

[1.4 Reference Documents 4](#_Toc474064145)

[1.5 Document Structure 5](#_Toc474064146)

[2. Classes 6](#_Toc474064147)

[2.1 BillingAccountWorker 6](#_Toc474064148)

[2.2 PaymentWorker 6](#_Toc474064149)

[3. Functional role 8](#_Toc474064150)

[4. List of Issues 10](#_Toc474064151)

[4.1 Naming Conventions 10](#_Toc474064152)

[4.2 Indention 10](#_Toc474064153)

[4.3 Braces 10](#_Toc474064154)

[4.4 File Organization 11](#_Toc474064155)

[4.5 Wrapping Lines 11](#_Toc474064156)

[4.6 Comments 12](#_Toc474064157)

[4.7 Java Source Files 12](#_Toc474064158)

[4.8 Package and Import Statements 12](#_Toc474064159)

[4.9 Class and Interface Declaration 12](#_Toc474064160)

[4.10 Initialization and Declarations 12](#_Toc474064161)

[4.11 Method Calls 12](#_Toc474064162)

[4.12 Arrays 12](#_Toc474064163)

[4.13 Object Comparison 13](#_Toc474064164)

[4.14 Output Format 13](#_Toc474064165)

[4.15 Computation, Comparisons, and Assignments 13](#_Toc474064166)

[4.16 Exceptions 13](#_Toc474064167)

[4.17 Flow of Control 13](#_Toc474064168)

[4.18 Files 13](#_Toc474064169)

[5. Other problems 14](#_Toc474064170)

[6. Hours of work 15](#_Toc474064171)

# Introduction

## Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Authors | Description |
| 1.0 | 05/02/2017 | S. Caprara, S. Ghanbari, E. Tinti | First release |

## Purpose and Scope

The objective of this document is to provide details about the inspection of the line of codes of some specific class contained in the Apache OFBiz (Open For Business) release.

The code inspection aims at finding possible issues inside the code and tries to provide solutions. All the aspects of the code are reviewed, such as the name of the methods, attributes and their definition, their usage, the presence of comments and descriptions, the length of LOC and so on. A check-list helps in doing the code analysis.

The Apache OFBiz is an open source product for the automation of enterprise processes that includes framework components and business applications, as stated on the official website.

## Definitions and Abbreviations

* **LOC:** lines of code.
* **DB:** Database.

## Reference Documents

The documents used as a reference to provide the design document are:

* Code Inspection Assignment Task Description.pdf

## Document Structure

The sections of the document are:

* **Introduction:** this chapter contains the purpose and scope of this document and introduces the code inspection.
* **Classes:** here, we describe the classes of this study and their respective methods.
* **Functional role:** this section contains an overview of the role that the classes we are inspecting play in the entire framework.
* **List of Issues:** this chapter contains all the issues that have been found in the inspected classes and provides the line number to make it easier to find them.
* **Other problems:** in this section, we describe other issues not included in the check-list and possible solutions.

# Classes

Two classes were assigned to our group for the code inspection.

## BillingAccountWorker

**Namespace**: org.apache.ofbiz.accounting.payment

**Methods**:

* makePartyBillingAccountList(GenericValue, String, String, Delegator, LocalDispatcher)
* getBillingAccountOpenOrders(Delegator, String)
* getBillingAccountAvailableBalance(GenericValue)
* getBillingAccountAvailableBalance(Delegator, String)
* getBillingAccountNetBalance(Delegator, String)
* availableToCapture(GenericValue)
* calcBillingAccountBalance(DispatchContext, Map<String, ? extends Object>)

## PaymentWorker

**Namespace**: org.apache.ofbiz.accounting.payment

**Methods**:

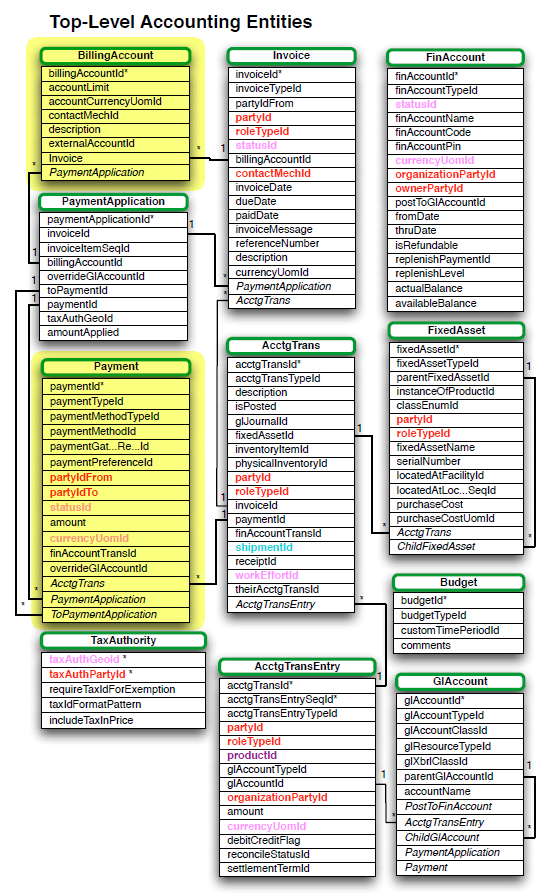
* getPartyPaymentMethodValueMaps(Delegator, String)
* getPartyPaymentMethodValueMaps(Delegator, String, Boolean)
* getPaymentMethodAndRelated(ServletRequest, String)
* getPaymentAddress(Delegator, String)
* getPaymentsTotal(List<GenericValue>)
* getPaymentApplied(Delegator, String)
* getPaymentApplied(Delegator, String, Boolean)
* getPaymentAppliedAmount(Delegator, String)
* getPaymentApplied(GenericValue)
* getPaymentApplied(GenericValue, Boolean)
* getPaymentNotApplied(GenericValue)
* getPaymentNotApplied(GenericValue, Boolean)
* getPaymentNotApplied(Delegator, String)
* getPaymentNotApplied(Delegator, String, Boolean)

# Functional role

The two classes ***PaymentWorker*** and ***BillingAccountWorker*** are part of the Accounting section of the Apache OFBiz project. The JavaDoc details are missing in their description, as the only annotations are respectively “Worker methods for Payments” and “Worker methods for BillingAccounts”.

It may be helpful to remark that the ***PaymentWorker*** class has the “final” keyword in its signature, so it can’t be extended by other classes.

The diagram shown below is part of the online documentation concerning the Data Model. It may clarify the main structure of the Accounting package.



The two classes discussed in this document refer to the highlighted elements and make queries to control the presence of payments or to retrieve the status of a billing account (balance and open orders) on the DB.

The classes used for building the queries are contained in the package *org.apache.ofbiz.entity*.

# List of Issues

## Naming Conventions

**BillingAccountWorker**

Line 67: String currencyUomId may be not clear enough.

Line 85: billingAccountVO variable has a name not clear.

Line 88: thruDate not clear the meaning of "thru".

Line 156: pAi stands for paymentApplicationIterator but not clear.

Line 180: availableToCapture() method name could be more meaningful.

Line 230: variable delegator may be not clear enough.

Line 262: actualCurrencyUomId variable has a name not clear.

Line 180: method availableToCapture() has not a verb as a name.

Lines 54-55: declared constants are not all uppercase.

**PaymentWorker**

Lines 50-52: declared constants are lowercase, instead of being uppercase.

## Indention

**BillingAccountWorker**

No issues found.

## Braces

**BillingAccountWorker**

Line 64: the if condition has only one statement without braces.

**PaymentWorker**

Line 66: the if condition has only one statement without braces.

## File Organization

The only problems reported here concern line length. A possible solution is breaking the lines after commas or when a method call takes place.

**BillingAccountWorker**

Lines 67-68, 70-71, 73, 75, 77-79, 85, 88-89, 91-92, 94-95, 98, 103, 111, 113-116, 121-122, 127-130, 133, 138-139, 144-145, 151, 154-156, 158, 161-162, 175, 180-181, 187, 194, 196, 198, 202-205, 210, 212, 216-218: limit of 80 characters is exceeded.

Lines 67, 70, 71, 73, 111, 121-122, 127, 130, 138-139, 144, 151, 154-155, 162, 175, 181, 187, 194, 218: limit of 120 characters is exceeded.

**PaymentWorker**

Lines 51-52, 57, 61-62, 64, 66, 69, 73-90, 99, 128-132, 159, 161, 163, 165, 176, 180-181, 184, 191, 193, 214, 225, 229, 231, 236, 242, 252, 256, 259-263, 274, 283, 285, 288, 293-294, 296-297, 300-305, 308, 319, 324-326, 328, 331, 335, 337, 342, 348, 350: limit of 80 characters is exceeded.

Lines 61, 64, 128-132, 159, 161, 163, 165, 181, 191, 193, 256, 260, 263, 297, 302, 305, 326, 350: limit of 120 characters is exceeded.

## Wrapping Lines

**BillingAccountWorker**

No issues found.

## Comments

No issues found.

## Java Source Files

No issues found.

## Package and Import Statements

No issues found.

## Class and Interface Declaration

**BillingAccountWorker**

Line 59: package variables are declared after private class variables (lines 57-58)

## Initialization and Declarations

**BillingAccountWorker**

Line 127: method visibility could be private

Line 151: method visibility could be private

Line 180: method visibility could be private

## Method Calls

No issues found.

## Arrays

No issues found.

## Object Comparison

No issues found.

## Output Format

**PaymentWorker**

Lines 238, 268: the message "Problem getting Payment" is not clear enough for the user.

Line 337: the message "Null delegator is not allowed in this method" is not very clear.

## Computation, Comparisons, and Assignments

No issues found.

## Exceptions

No issues found.

## Flow of Control

No issues found.

## Files

No issues found.

# Other problems

Methods of the class ***BillingAccountWorker*** are only called internally (or never called) and no methods are called outside the class. According to this, we can tell that this class is currently not useful and may be removed. Moreover, the methods listed below are never used:

* makePartyBillingAccountList(GenericValue, String, String, Delegator, LocalDispatcher), line 67
* getBillingAccountOpenOrders(Delegator, String), line 111
* getBillingAccountAvailableBalance(Delegator, String), line 138

The following methods of the class ***PaymentWorker*** are never used and may be removed:

* getPartyPaymentMethodValueMaps(Delegator, String), line 57
* getPaymentMethodAndRelated(ServletRequest, String), line 99
* getPaymentApplied(Delegator, String), line 225
* getPaymentAppliedAmount(Delegator, String), line 252
* getPaymentNotApplied(GenericValue, Boolean), line 324
* getPaymentNotApplied(Delegator, String), line 331

# Hours of work

To make this document we have spent:

* Sergio Caprara, 18 hours
* Soheil Ghanbari, 1 hours
* Erica Tinti, 18 hours