



**POLITECNICO**  
MILANO 1863

# Software engineering 2

## Code inspection document

Luca Scannapieco - 877145  
Andrea Pasquali - 808733  
Emanuele Torelli - 876210

05/02/2017

## Table of contents

1. Introduction.....	2
2. Assigned classes.....	2
3. Functional role of assigned set of classes.....	2
4. List of issues found by applying the checklist.....	3
4.1. EntityDataAssert.java .....	3
4.2. EntityCrypto.java .....	4
5. Other highlighted problems .....	4

## 1. Introduction

The aim of this document is to report on the quality status of some code extracts from the Apache OFBiz project, an open source product for the automation of enterprise processes. Such inspection is performed with the support of a review checklist that contains all possible issues that can be found in Java source code.

## 2. Assigned classes

The paths of the two assigned classes are, respectively:

../apache-ofbiz-

16.11.01/framework/entity/src/main/java/org/apache/ofbiz/entity/util/EntityDataAssert.java

../apache-ofbiz-16.11.01/framework/entity/src/main/java/org/apache/ofbiz/entity/util/EntityCrypto.java

For the sake of simplicity, following in the document, we will call the two classes with only their names, that are respectively EntityDataAssert and EntityCrypto.

## 3. Functional role of assigned set of classes

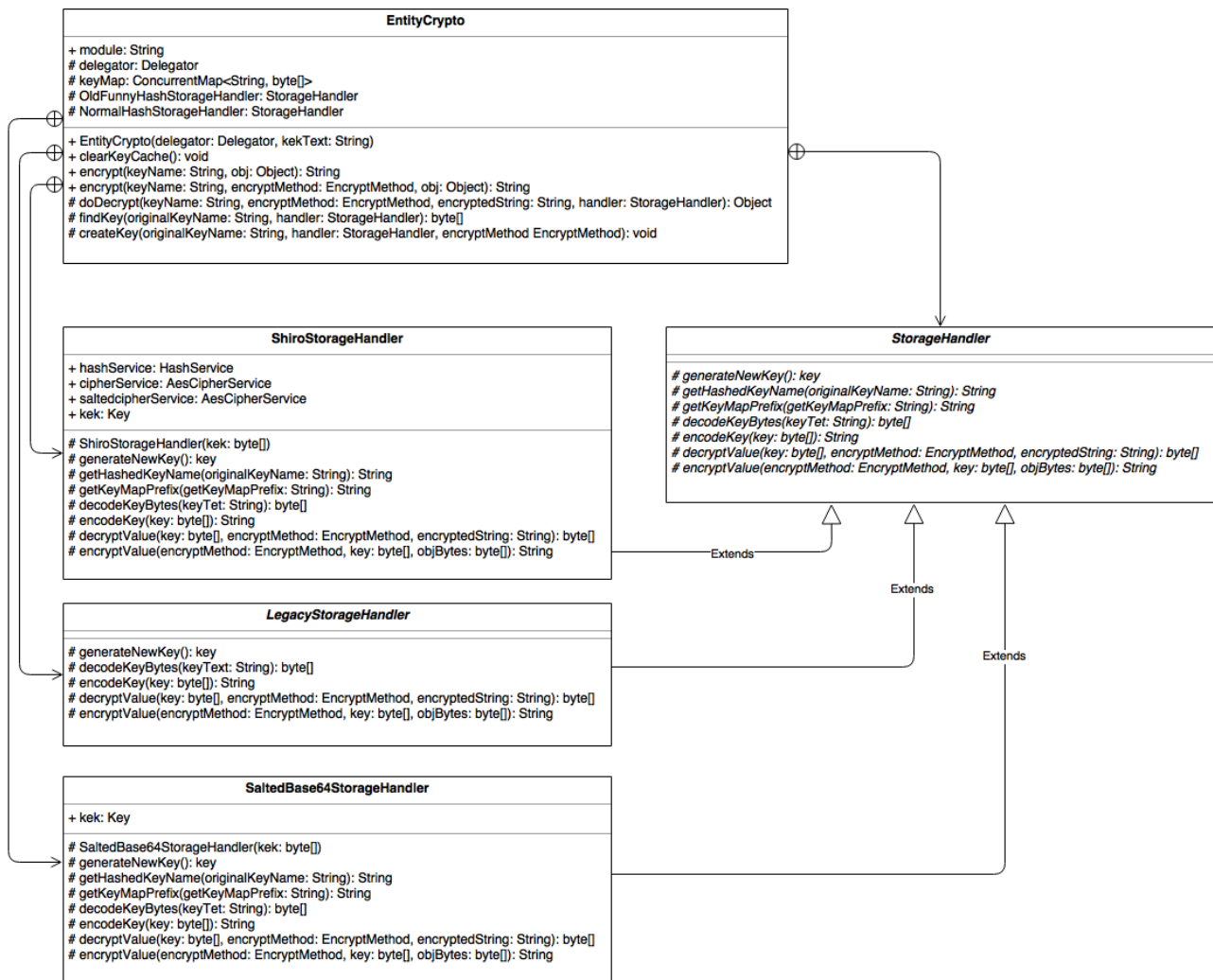
In this section we are going to define the roles of the two source files assigned. Due to a consistent lack of Javadoc we can only rely on our knowledge of Java language in order to identify the core functionalities of the two files.

The aim of the class EntityDataAssert is to provide some utility routines to load data, as specified in the very concise related Javadoc. It contains the methods assertData, checkValueList and checkSingleValue, which most likely have the goal of retrieving data from a XML document and checking its validity.

EntityDataAssert
+ module: String
+ assertData(dataUrl: URL, delegator: Delegator, errorMessages: List<Object>): int + checkValueList(valueList: List<GenericValue>, delegator: Delegator, errorMessages: List<Object>): void + checkSingleValue(checkValue: GenericValue, delegator: Delegator, errorMessages: List<Object>): void

The aim of EntityCrypto is to provide some security functionalities, as suggested by the methods named encrypt, decrypt, doDecrypt and so on. The source file contains, in addition of the homonymous public class, some other protected classes called “storage handlers”. The one called generically StorageHandler is abstract and contains abstract methods, the other classes ShiroStorageHandler, LegacyStorageHandler and SaltedBase64StorageHandler extend the StorageHandler one, overriding the methods and containing some peculiar attributes, except from the LegacyStorageHandler one which is abstract too. The various kinds of handlers suggest that each of them provides different techniques of encryption/decryption of entities, as suggested by the signature of the methods, e.g. generateNewKey, getHashedKeyName, encryptValue, decryptValue and so on. EntityCrypto class contains both instance and static variables, in particular the static ones are instances of LegacyStorageHandler with some overridden methods. Below we provide a

class diagram to further clarify the structure of the source file and the relations between the internal classes.



## 4. List of issues found by applying the checklist

In this section we are going to list all the issues found according to the checklist provided.

### 4.1. EntityDataAssert.java

1. The class hasn't got a meaningful name, furthermore a verb shouldn't be used for a class name.
2. The name of the method declared in line 42 is ambiguous, the verb "assert" isn't clear in the context of the class.
3. The heading comment figures as Javadoc due to the multiple stars.
4. Declarations of methods in lines 42, 70 and 78 exceed the 120-character limit per line.
5. Javadoc is missing in every method.
6. The variable `checkValue` of type `GenericValue` has an ambiguous name and the use of a verb at the imperative is more likely to suggest a method name and not that the variable is actually a value to

be checked. This possible misunderstanding occurs every time the variable `checkValue` is involved (e.g. line 56, 73, 74, 78, 79, 84, 94, 102). A better name could be `valueToCheck`.

7. Same story for the `checkField` object (e.g. at lines 102, 105, 107).
8. Same story for the `checkPK` object (e.g. at lines 102, 105, 107).
9. The “PK” acronym should be cased consistently, maybe it’s better to avoid lowercases for example as for the variable `nonPkFieldName` which might be difficult to read (e.g. lines 95, 97, 98, 102, 103, 106), as well as mixed cases like for the method name `getNoPkFieldNames` (e.g. line 95, 106).
10. The “PK” acronym, which most likely means primary key, should be explained, for example in the existing comment in line 83.
11. The comment in lines 62-63 includes some unused code (line 63) and the motivation has been cut off (line 62), but the reason why the code is left there isn’t explained.
12. In method `checkSingleValue`, the variables `checkPK` (line 84), `modelEntity` (line 94), `checkField` (line 102) and `currentField` (line 103) aren’t declared at the beginning of a block.

#### 4.2. EntityCrypto.java

1. All the classes of the source file haven’t got the related Javadoc (lines 50, 214, 227, 296, 351).
2. Javadoc is missing in every method starting from the one at line 124.
3. Lines 97-98, line 102: the comments smell of something going wrong.
4. The variable `kek` of type `byte[]` used in the `EntityCrypto` constructor (lines 60, 61, 63, 64) has an unclear name (we suppose that `kek` stands for key encryption key).
5. Same story for the parameter `kekText` (lines 58, 61).
6. At line 130 there is a comment that suggest that the following code is incomplete.
7. The comment in lines 62-63 includes an exception (line 63), but the reason why it has been cut out is unclear.
8. The methods `decrypt`, declared in line 124, and `doDecrypt`, declared in line 142, have very similar names. Furthermore the code of method `decrypt` suggests that it only delegates the method `doDecrypt` passing some parameters, so the real operation of decrypting is carried out by the `doDecrypt` method.
9. At line 169 the variable `keyValue` isn’t declared at the beginning of a block.
10. The static final attributes `OldFunnyHashStorageHandler` and `NormalHashStorageHandler`, respectively in lines 328 and 340, are declared almost at the end of the source file instead of before the instance variables.
11. The static final attributes `OldFunnyHashStorageHandler` and `NormalHashStorageHandler`, respectively in lines 328 and 340 begin with a capital letter, which is a naming convention preferred for classes or interfaces. Since they are static variables, maybe they’re likely to be intended as constants, therefore the suggestion is to rename them with all uppercase characters, separated with an underscore.
12. Lines 103, 128, 142, 191 and 360 exceed the 120-character limit per line.

#### 5. Other highlighted problems

During the inspection we haven’t found any problem or potential bug.