

**POLITECNICO DI MILANO**  
**Computer Science and Engineering**  
**Project of Software Engineering 2**

# **Code Inspection**

Authors: Falci Angelo 875123  
Lanzuise Valentina 807364  
Lazzaretti Simone 875326

Reference Professor: Di Nitto Elisabetta

## **SUMMARY**

<b>1. Table of content.....</b>	<b>3</b>
<b>2. Classes assigned.....</b>	<b>3</b>
<b>3. Functional Role of Assigned Set of Classes.....</b>	<b>3</b>
<b>4. List of Issue Found By Applying the Checklist.....</b>	<b>4</b>
<b>5. Effort spent.....</b>	<b>8</b>

## **1 Table of Content**

In this document we are going to analyse the part of code that was commissioned to us. We divided the content of the document in the following parts:

- First of all, we wrote the links of the assigned classes
- Then, we gave a brief description of the role and function of the classes
- In conclusion, we wrote a list of the errors found in the two classes according to the “Code inspection checklist” in the assignment document

## **2 Classes assigned**

These are the classes that were assigned to us:

../apache-ofbiz-

16.11.01/applications/party/src/main/java/org/apache/ofbiz/party/content/PartyContentWrapper.java

../apache-ofbiz-

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

## **3 Functional Role of Assigned Set of Classes**

*EntityDataLoader.java*

This class has two main functionalities:

- First, thanks to the methods “getUrlList” and “getUrlByComponentList”, this class provides and returns a list of URLs based on different properties like helperName or componentName.
- Second, thanks to the method “loadData”, we can change the data included in an XML file (we send the URL of the XML as a parameter) and return how many rows the method modified.

*PartyContentWrapper.java*

This class is used to take information about an entity called party.

## 4 List of Issue Found By Applying the Checklist

### *EntityDataLoader.java*

For this class we made the following analysis:

- All classes, interfaces, methods and variables have meaningful names, that respect the signature.
- The static variable `module` at line 56 is written in lowercase but, like all constants, it is better writing it in uppercase.
- Indentation and braces are correct.
- Some rows are longer than 120 characters and it is a bad thing because the code becomes unreadable. Here the list of lines which exceed the limit:
  - line 119: 125 char
  - line 127: 176 char
  - line 137: 131 char
  - line 222: 139 char
  - line 226: 154 char
  - line 230: 212 char
  - line 259: 125 char
  - line 287: 124 char
  - line 294-297: 159 char
- There is a wrong indentation at line 281: the code must be on the same level of the previous line.
- There are two blocks of comments containing code lines that aren't explained: one at the line 243, and the second at the end of the class, line 293. In this way comments are useless.
- The class might have a Javadoc description, but it is absent.
- We must import the two classes "EntityUtil" and "EntitySanReader" used respectively in the methods "public static <E> List<URL> getUrlList(String helperName, String componentName, List<E> readerNames)" at line 195 and "public static int loadData(URL dataUrl, String helperName, Delegator delegator, List<Object> errorMessages, int txTimeout, boolean dummyFks, boolean maintainTx, boolean tryInsert) throws GenericEntityException." at line 230.
- Class and interface declaration is correct.
- The method `public static <E> List<URL> getUrlList(String helperName, String componentName, List<E> readerNames)` at line 89 is too long, and should have been divided into more short methods.
- The assignment "`EntitySaxReader reader = null;`" at line 144 is useless because in the following rows the variable is set.
- The assignment "`String readerName = null;`" at line 96 is useless because in the following rows the variable is set.
- In the method "public static <E> List<URL> getUrlList(String

helperName, String componentName, List<E> readerNames)“ it is useless the initialization “URL url = null;” at line 175 because the value “null” is never used.

- In the initialization in the form “List <Type> nameVariable=new List<Type>();” there is redundancy because it is useless writing “Type” again after the equal sign (examples at line 196, 198).
- The declaration “EntityDataReader entityDataReaderInfo = null;” at line 114 must be declared at the beginning of the block of the method.
- Parameters of the methods are written in the right order.
- In the method “public static <E> List<URL> getUrlList(String helperName, String componentName, List<E> readerNames) “ at line 97 there are three 'if' nested each other. In this case it is better to use 'switch'.
- Methods return the right values.
- The error messages are written in the right form, except for line 104 where “ReadData(s)” is missing; in fact it is written as a comment, but it is better to write it in the output.
- There aren't arithmetic expressions in our code, so the precedences of parenthesis and operators are respectful.
- Exceptions are always managed in the best way, but they are “Generic”.
- Switch statement aren't used and loops are correctly formed.
- There aren't file used in this classes.

### *PartyContentWrapper.java*

For this class we found the following problems:

- All classes, interfaces, methods and variables have meaningful names, that respect the signature.
- The static variable “module” at line 57 is written in lowercase but, like all constants, it is better writing it in uppercase.
- The static variable “UtilCache” at line 60 is written in lowercase but, like all constants, it is better writing it in uppercase.
- Indentation and braces are correct.
- There is a wrong indentation at the following lines: the code must be on the same level of the previous line  
line 133  
line 139  
line 148  
line 152  
line 261-266  
line 297-301
- Some rows are longer than 120 characters and it is a bad thing because the code becomes unreadable:

line 46: 130 char  
 line 65: 169 char  
 line 70: 137 char  
 line 102: 138 char  
 line 110: 139 char  
 line 112: 144 char  
 line 112: 157 char  
 line 116:152 char  
 line 121: 134 char  
 line 122: 136 char  
 line 126: 134 char  
 line 155: 121 char  
 line169: 128 char  
 line 174: 266 char  
 line 175: 138 char  
 line 178: 281 char  
 line 188: 138 char  
 line 198: 150 char  
 line 207: 168 char  
 line 218: 124 char  
 line 233: 123 char  
 line 246: 224 char  
 line 261: 172 char  
 line 269: 144 char

- Comments are written in order to separate the code and explain what it is doing.
- The class might have a Javadoc description, but is absent.
- There aren't import missing.
- Class and interface declaration is correct.
- The internal variable `List<String> contentList = new LinkedList<String>()` makes an useless usage of the type “String” at the right of the expression.
- In the internal variable `“Map<String, Object> inContext = new HashMap<String, Object>()”` there is redundancy because it is useless writing “String, Object” again after the equal sign.
- The method `“public static String getPartyContentAsText(GenericValue party, String contentId, String partyContentTypeId, Locale locale, String mimeTypeId, Delegator delegator, LocalDispatcher dispatcher, boolean useCache, String encoderType)”` is too long (it consists of 50 lines, more or less), it could be divided in more sub-methods to lower its complexity.
- The method `“public static void getPartyContentAsText(String contentId, String partyId, GenericValue party, String partyContentTypeId, Locale locale, String mimeTypeId, Delegator`

delegator, LocalDispatcher dispatcher, Writer outWriter, boolean cache) throws GeneralException, IOException” is too long (it consists of 60 lines). It could be divided into three sub-methods, in order to lower its complexity.

- The method `“public static List<String> getPartyContentTextList(GenericValue party, String partyContentTypeId, Locale locale, String mimeTypeId, Delegator delegator, LocalDispatcher dispatcher) throws GeneralException, IOException”` it's too long. It could be divided in two sub-methods to make it better readable and to lower its complexity.
- At line 146 `“cacheKey”` isn't initialized, is declared in the wrong position: it must be declared at the beginning of the method.
- At line 163-166 `Writer outWriter = new StringWriter();` and `String outString = outWriter.toString();` aren't declared at the beginning of the block.
- In the method `“public List<String> getList(String contentTypeId)”` at line 99 there are two useless 'catch' with `“IOException”` and `“Exception”`. These exceptions are useless because they are already caught from the `“GeneralException”` put in a previous 'catch'.
- In the method at line 138 `“public static String getPartyContentAsText(GenericValue party, String contentId, String partyContentTypeId, Locale locale, String mimeTypeId, Delegator delegator, LocalDispatcher dispatcher, boolean useCache, String encoderType)”` there is an useless 'catch' with `“IOException”`. It is useless because, as in the point above, it is already caught from `GeneralException”` put in a previous 'catch'.
- In the method at line 191 `“public static void getPartyContentAsText(String contentId, String partyId, GenericValue party, String partyContentTypeId, Locale locale, String mimeTypeId, Delegator delegator, LocalDispatcher dispatcher, Writer outWriter, boolean cache) throws GeneralException, IOException”` the last 'return' is useless because there is nothing after it.
- On line 73, the method `“public StringUtil.StringWrapper get(String contentTypeId, String encoderType)”` is an inherited one by the ContentWrapper interface. It's necessarily to add an `“@Override”` before the method.
- Methods on line 116 and 120 can be avoided just calling with correct null parameters the method `“getPartyContentAsText”`. In this way can be avoided to repeat useless code.
- The code block line 215-227 and 230-242 could be merged in a method with a different call, in order to avoid repetition of some lines of code
- The usage of `“==”` operator is adequate.
- The error messages are written in the right form.
- There aren't arithmetic expressions in our code, so the precedences of

- parenthesis and operators are respectful.
- Switch statement aren't used and loops are correctly formed.
- There aren't file used in this classes.

**Effort spent**

- Falci Angelo 8 hours
- Lazzaretti Simone 10 hours
- Lanzuise Valentina 10 hours