DP2 2023-2024
Lint Report

Acme Software Factory

Repository: https://github.com/rafcasceb/Acme-SF-D04

Student #5:

Vento Conesa, Adriana	adrvencon@alum.us.es

Other members:

Castillo Cebolla, Rafael	rafcasceb@alum.us.es
Flores de Francisco, Daniel	danflode@alum.us.es
Heras Pérez, Raúl	rauherper@alum.us.es
Mellado Díaz, Luis	luimeldia@alum.us.es

GROUP C1.049

Version 1.0

25-05-24

Content Table

Abstract

This report offers a detailed examination of the lint analysis conducted for project deliverable D03 and D04. It presents a thorough evaluation of "bad smells" reported by Sonar Lint, justifying their innocuous nature if appropriate.

Revision Table

Date	Version	Description of the changes	Deliverable
20/04/2024	V1	• Abstract. • Introduction. • Contents: task detailing. • Conclusions.	3
22/04/2024	V1	• Added missing analysis.	3
25/05/2024	V1	• Updated lint report for D04.	4

Introduction

In these two phases of delivery, the focus was directed towards the implementation of different features, creating multiple bad smells in the process of doing so. The subsequent analysis in this document pertains solely to the lint evaluation of student five's section, while the lint report for the collective group work will encompass assessments of other sections.

The document structure is outlined as follows: an introductory section that contextualizes the lint analysis, followed by a comprehensive enumeration of bad smells detected by Sonar Lint within student five's section. Finally, a succinct conclusion will encapsulate the report, summarizing key findings.

Contents

Bad Smells

The following bad smells appeared several times in the individual files analyzed for this report. For the sake of simplicity, they have been consolidated into a singular enumeration.

1. Override the "equals" method in this class.
   As we follow the Acme Framework recommendations given in the theory lectures, this bad smell can be considered innocuous.

2. Use concise character class syntax '\\d' instead of '[0-9]'.
   This bad smell is innocuous, as it performs the same functionalities as the alternative option proposed and does not hinder the maintainability of the code. The change is only more concise.

3. Rename this package name to match the regular expression '^[a-z_]+(\.[a-z_][a-z0-9_]*)*$'.
   As we follow the structure recommendations given by the Acme Jobs example project, this bad smell can be considered innocuous.

4. Replace this assert with a proper check.
   As we follow the Acme Framework recommendations given in the theory lectures, this bad smell can be considered innocuous.

5. Define a constant instead of duplicating this literal “X” times.
   This bad smell is innocuous by nature, as the duplication of literals does not directly affect the implementation of the features. It could potentially pose a refactoring problem, but as the recommendations and examples given by the subject also duplicate literals, we can safely ignore this.

6. Add explicit curly braces to avoid dangling else.
   This bad smell cannot be corrected, as the workspace automatically removes redundant curly braces in if-else structures. Nevertheless, it can be considered innocuous.

Conclusions

In conclusion, this lint report document allows us to examine the obstacles faced in analyzing and implementing diverse requirements, as well as a revision of our code to avoid future maintainability problems.

Bibliography

Intentionally blank.