# Rapport TP1

Julien Herbaux

Octobre 2023

## Contents

# 1 Introduction

Dans ce TP, on se propose de mettre en œuvre les techniques de refactoring, aliées à des outils de mesure de la qualité logicielle (analyse statique). Dans ce but, nous utilisons un kata. Les kata sont des exercices de programmation, souvent disponibles sur le web, dont l'objectif est de s'entrainer en travaillant certaines techniques de programmation ou de conception. Nous utilisons ici le kata "Guilded Rose" en Kotlin.

Les outils d'analyse utilisés sont les suivants:

- Jacoco
- Pitest

De plus, indispensable lors d'un travail de refactoring, on s'appuie ici sur un gestionnaire de version (Git).

# 2 GuildedRose

Le kata "Gilded Rose" présente un scénario où différents types d'articles nécessitent différentes règles de mise à jour pour leurs qualités et jours restants avant péremption. Le code d'origine utilise souvent des conditionnelles pour distinguer ces comportements, ce qui rend le code complexe et difficile à suivre.

# 3 Nouvelle architecture

Une fois les tests établis, validés à 100% et un coverage de 100%, on s'est décidé dans le cadre du refactoring de ce code d'utiliser du **polymorphisme**.

Le polymorphisme nous permet de traiter des objets de différentes classes comme s'ils appartenaient à une même classe parente. Dans le contexte de "Gilded Rose", cela signifie que nous pouvons définir une méthode de mise à jour générique pour tous les articles et ensuite l'adapter pour chaque type d'article.

Voilà les avantages qu'apportent le polymorphisme:

1. Lisibilité: Grâce au polymorphisme, chaque type d'article a sa propre classe, avec ses propres règles de mise à jour. Cela réduit la nécessité de longues chaînes de conditions et rend le comportement de chaque article immédiatement clair et explicite.

2. Maintenabilité: Si un comportement d'article doit être modifié ou corrigé, il suffit d'aller directement à la classe de cet article. Il n'est plus nécessaire de naviguer à travers un enchevêtrement de conditionnelles.

3. Évolutivité: Si un nouvel article est ajouté au système, plutôt que d'ajouter une nouvelle conditionnelle au code existant, nous pouvons simplement

créer une nouvelle classe pour cet article. Cela permet une extension facile et propre du système.

La seule fonctionnalité restante dans GuildedRose.class est maintenant donc le passage des jours et l'appelle de la mise à jour sur l'item en question.

# 4 Conclusion

En utilisant le polymorphisme, le refactoring du kata "Gilded Rose" a transformé un code dense et enchevêtré en un système organisé et orienté objet. Chaque type d'article a sa propre logique clairement définie, rendant le code plus lisible, maintenable et évolutif.