TP ICT-301 : Principes SOLID en Java

Diagrammes UML des exemples avant et après utilisation des principes solides

ZIEM DAVID DE JESUIS

Matricule : 23V2456

Pour chaque principe (SRP, OCP, LSP, ISP, DIP), nous présentons :

• Le diagramme de classes "Avant Refactoring" : qui illustre la violation du principe avec le code initial implémenté.

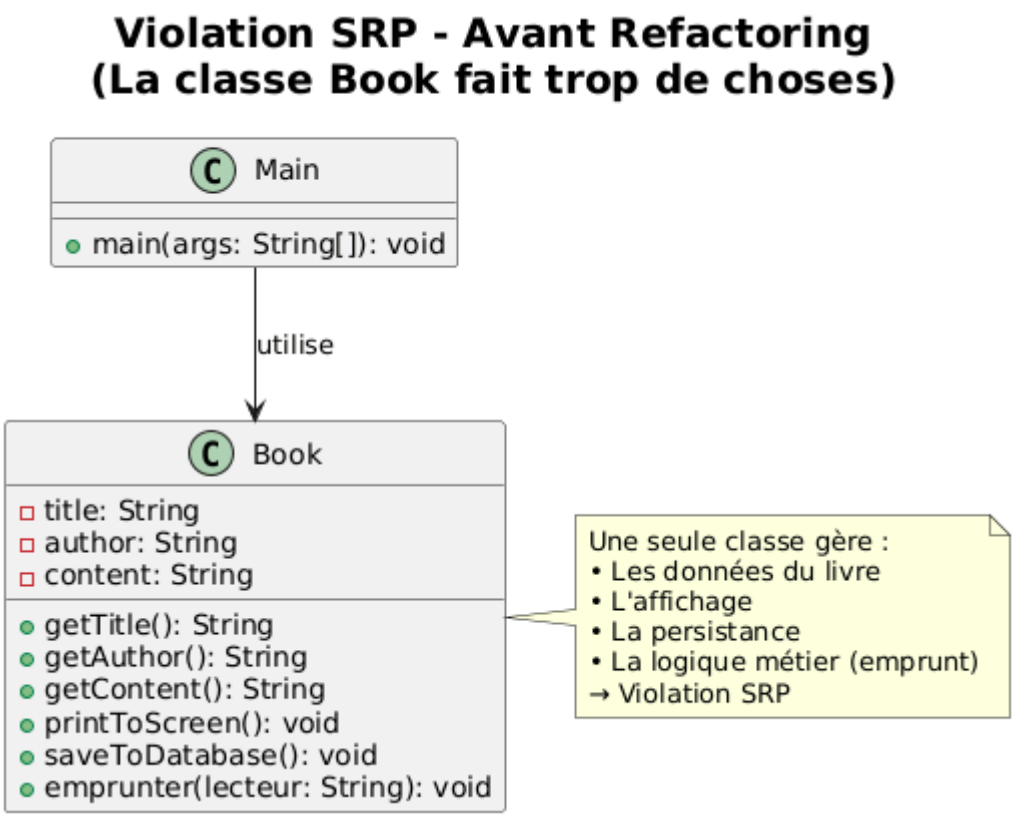
• Le diagramme de classes "Après Refactoring" : qui montre comment le respect du principe est obtenu grâce à une meilleure conception.

Certains principes incluent également un exemple supplémentaire issu directement des supports de cours pour compléter la démonstration.

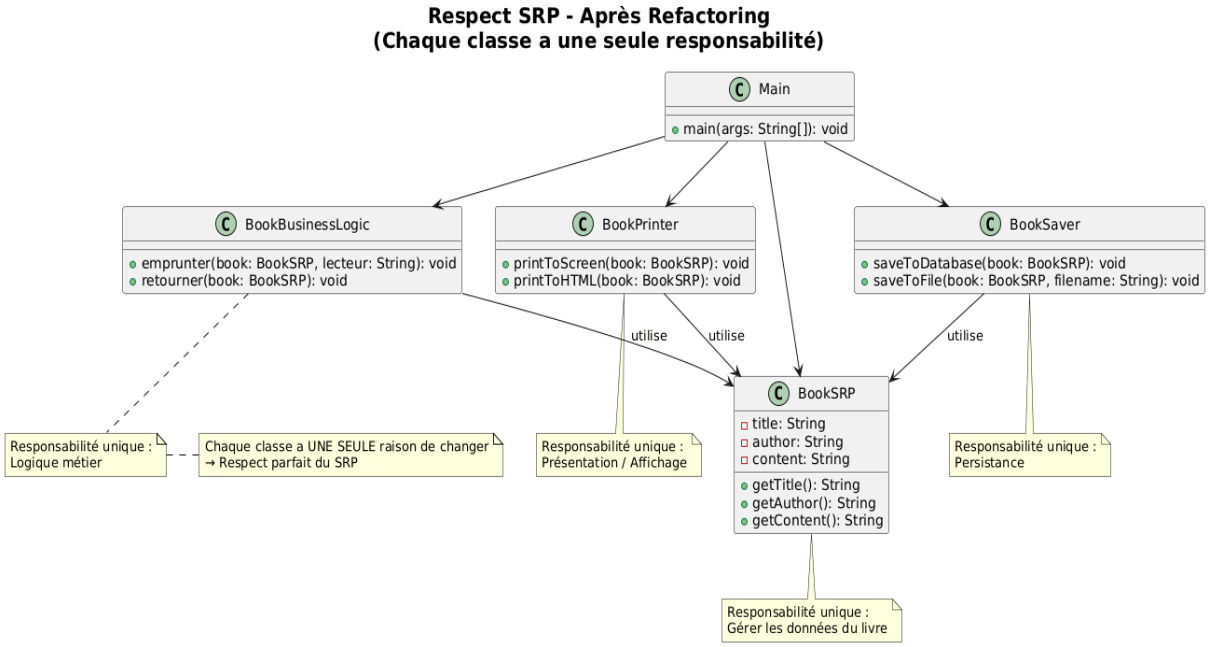
Ces diagrammes ont été réalisés à partir du code Java développé dans le cadre de ce TP, en respectant fidèlement les implémentations avant et après application de chaque principe SOLID.

1. PRINCIPE SRP(Single Responsibility Principle)

- Violation SRP - Avant Refactoring

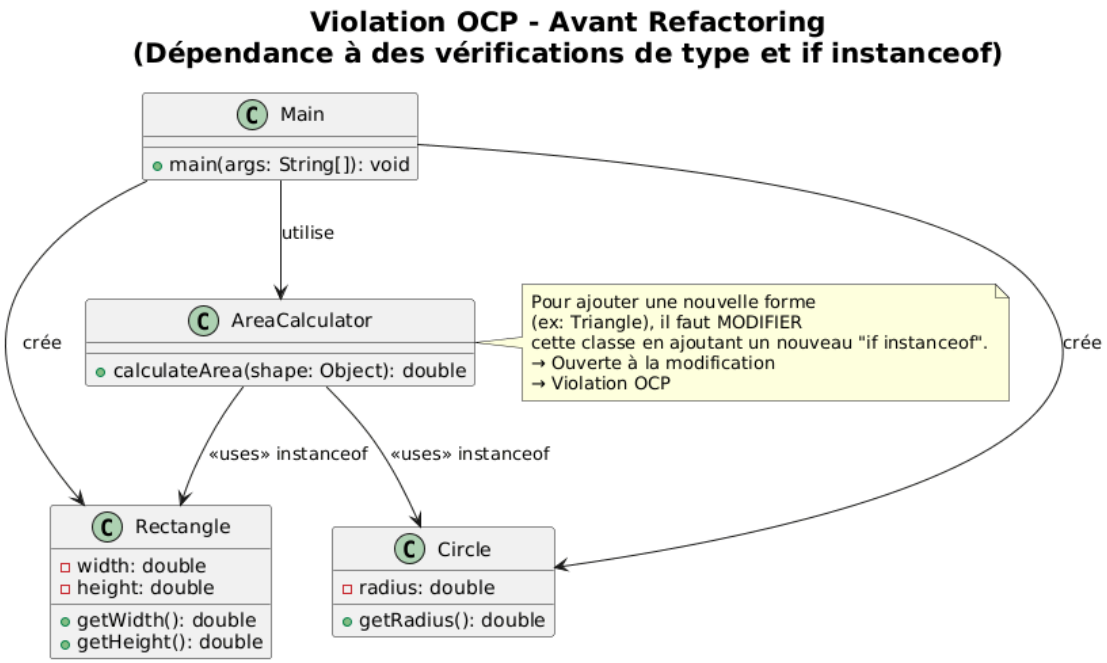


- Respect SRP - Après Refactoring

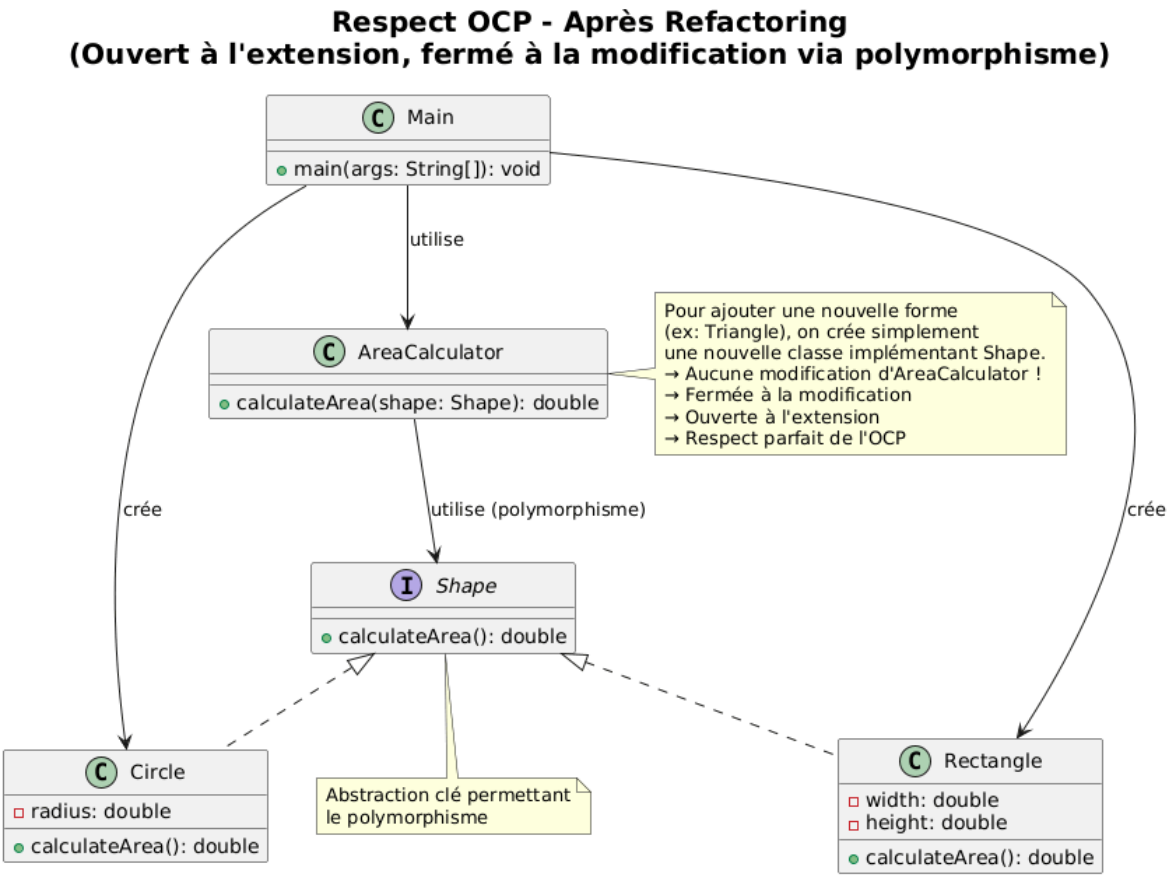


2 PRINCIPE OCP(Open/Closed Principle)

- Violation OCP - Avant Refactoring



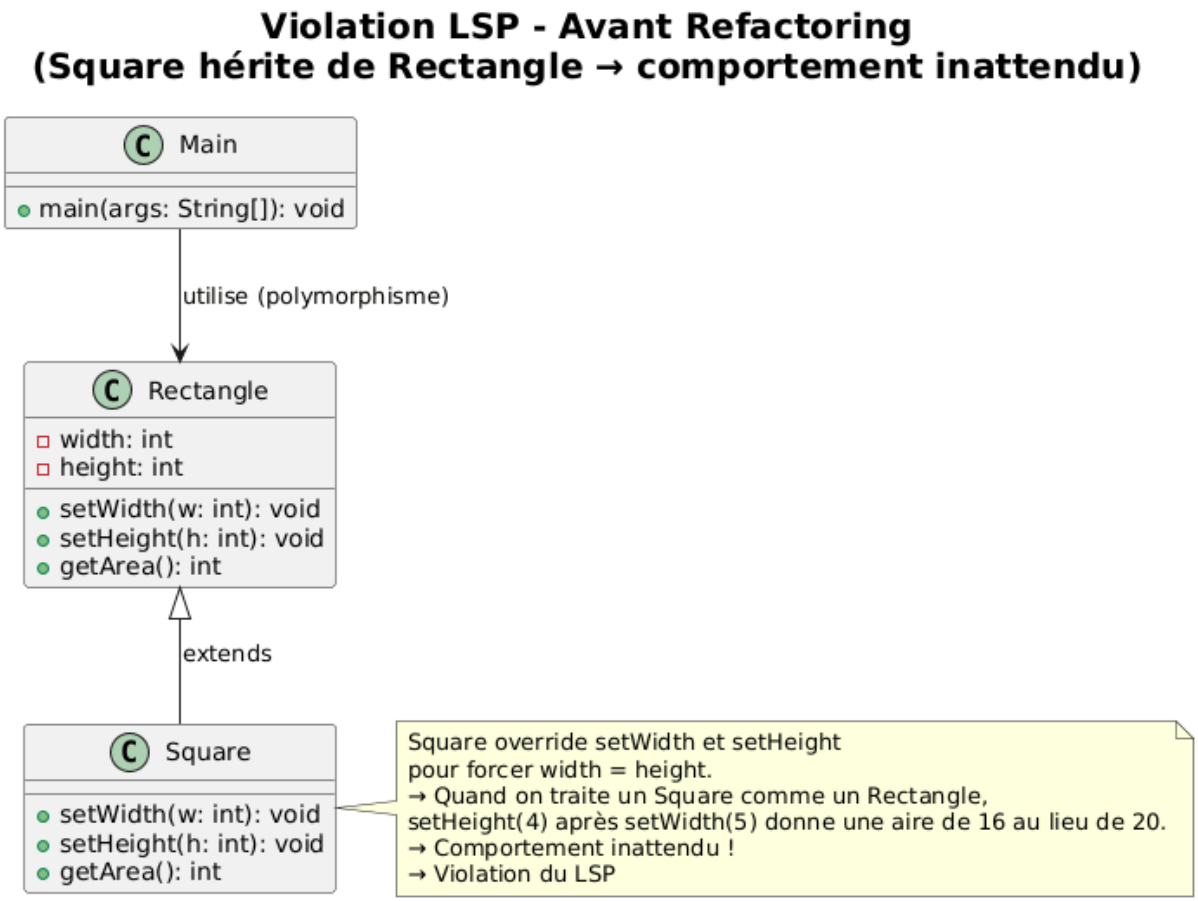
- Respect COP - Après Refactoring



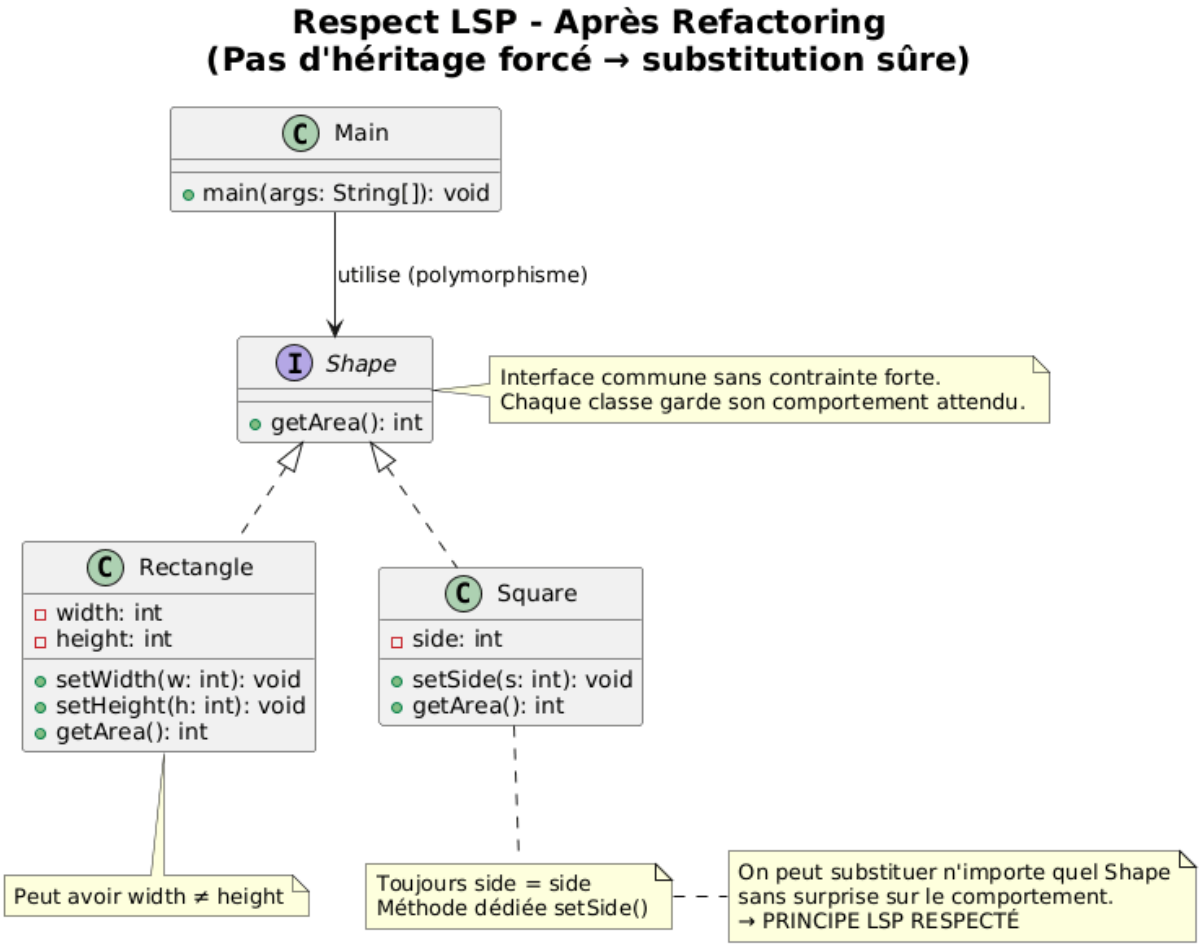
3 PRINCIPE LSP(Liskov Substitution Principle)

3.1- Exemple 1

- Violation LSP - Avant Refactoring

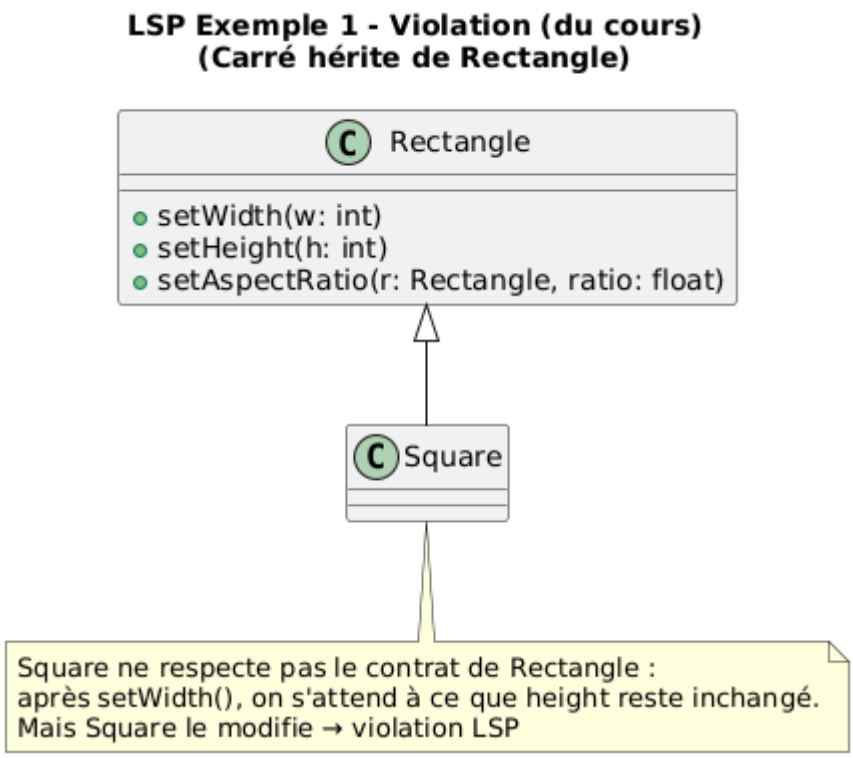


- Respect LSP - Après Refactoring

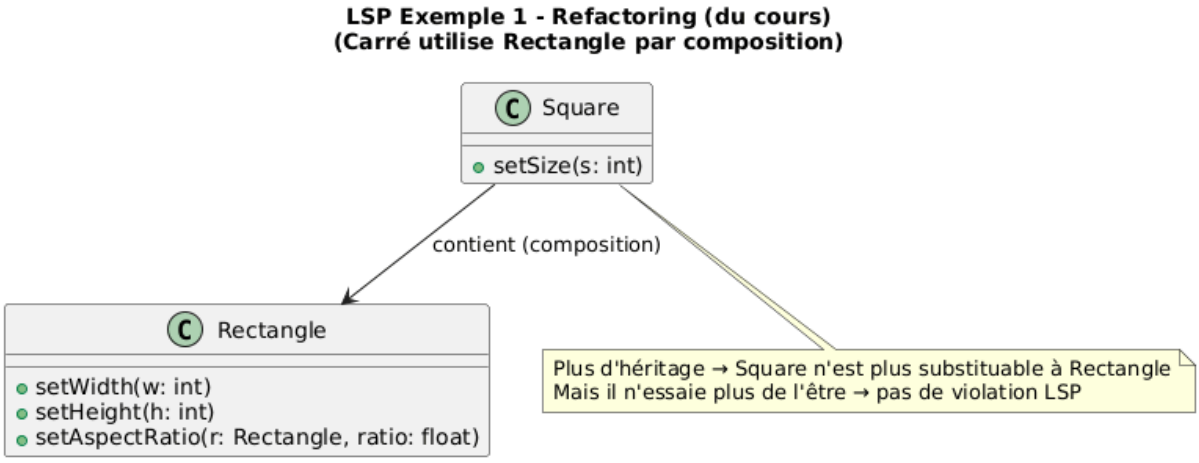


3.2 – Exemple 2

- Violation LSP - Avant Refactoring

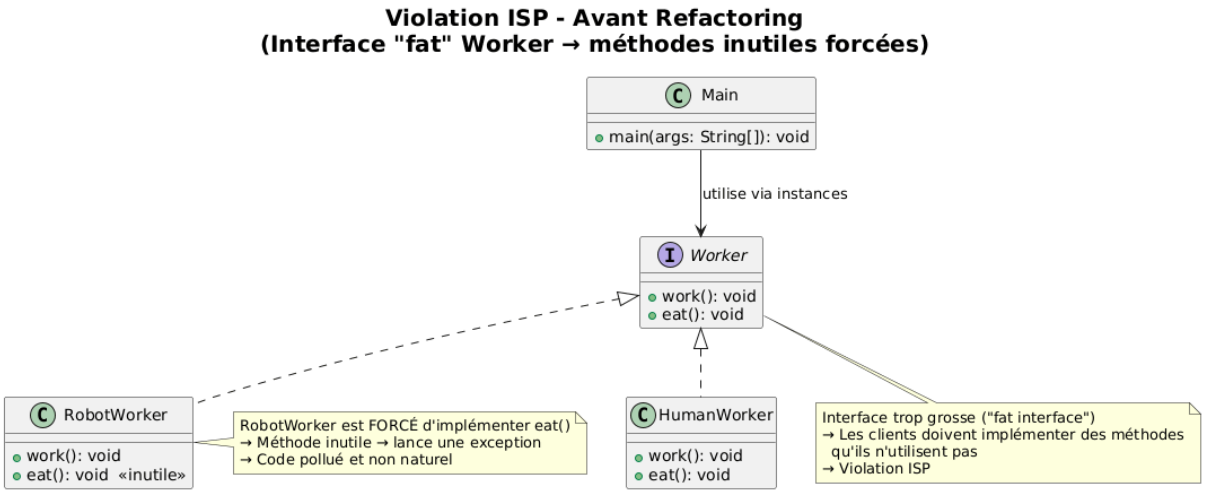


- Respect SRP - Après Refactoring

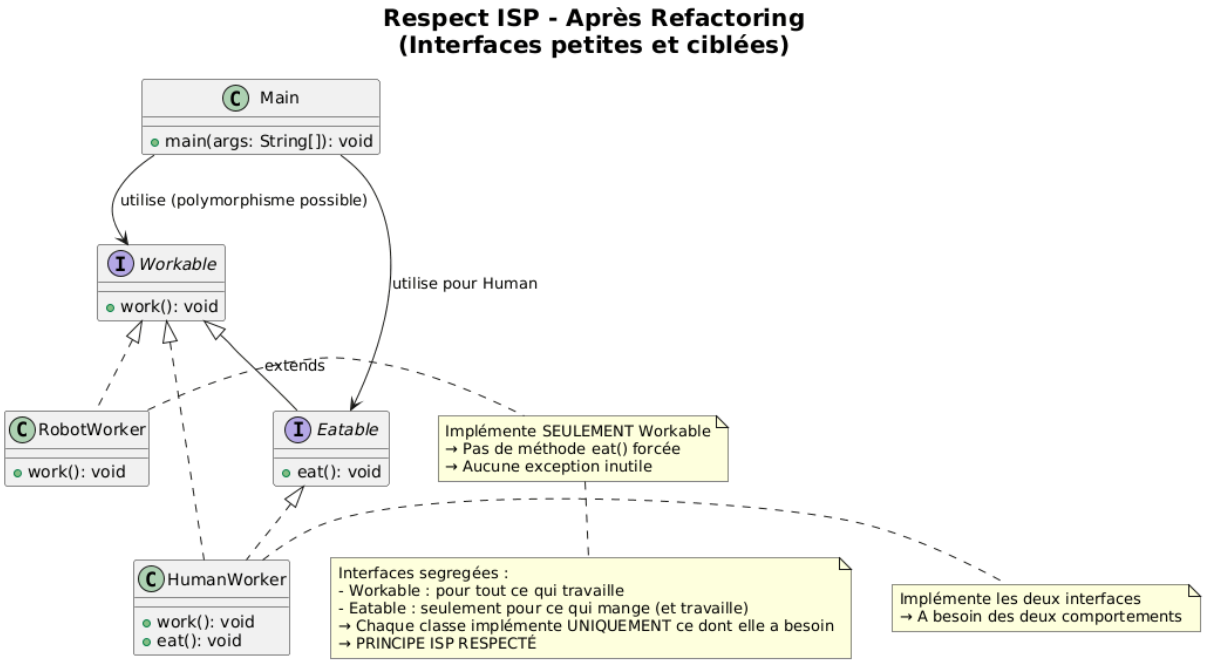


4 PRINCIPE ISP(Interface Segregation Principle)

- Violation ISP - Avant Refactoring



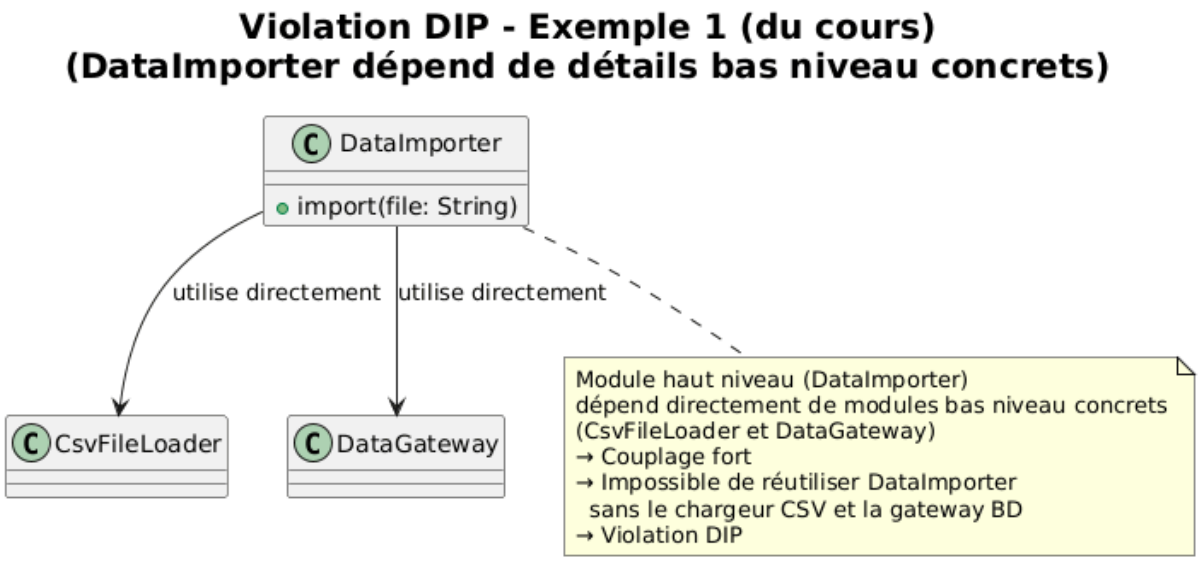
- Respect ISP - Après Refactoring



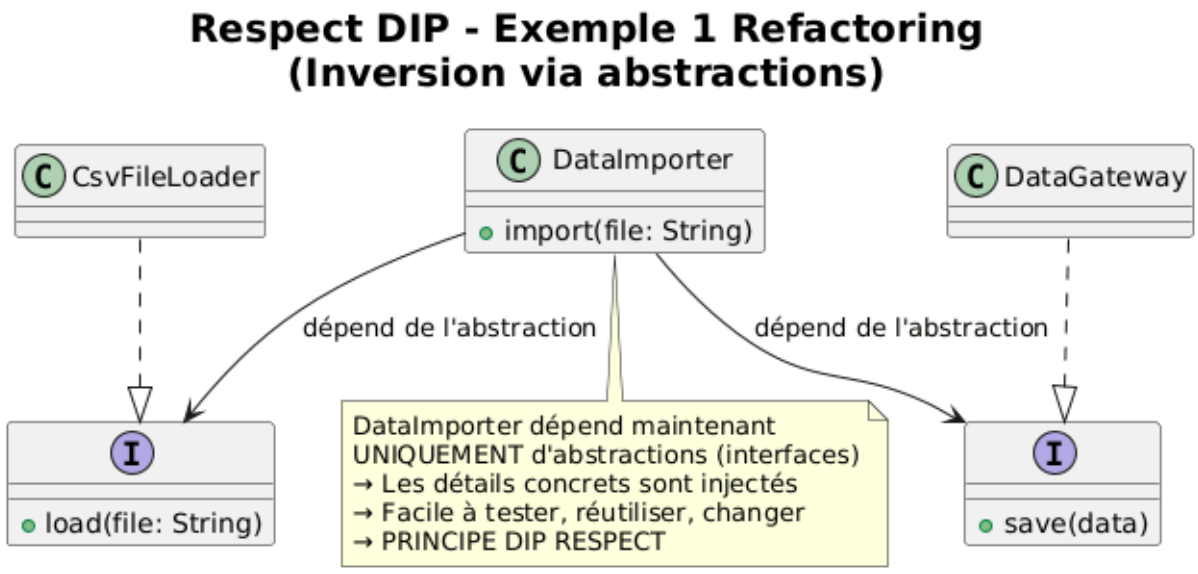
5. PRINCIPE DIP(Dependency Inversion Principle).

5.1 – exemple 1

- Violation DIP - Avant Refactoring

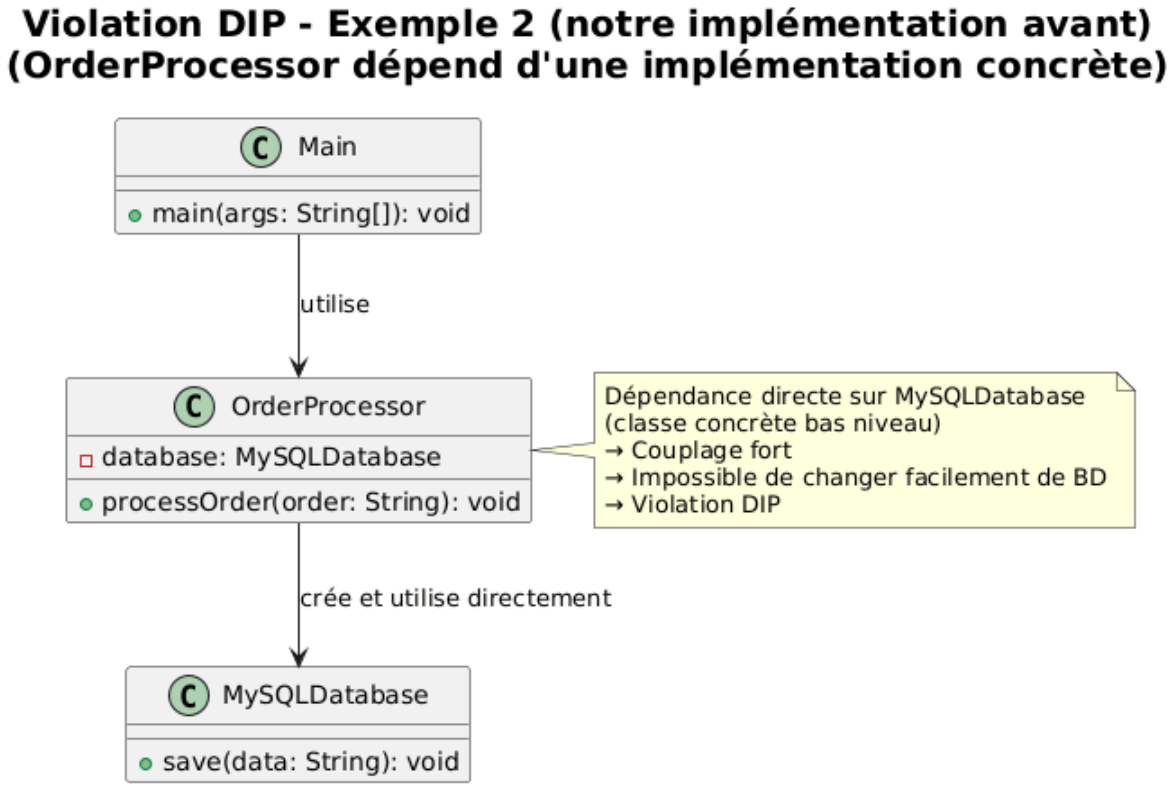


- Respect DIP - Après Refactoring

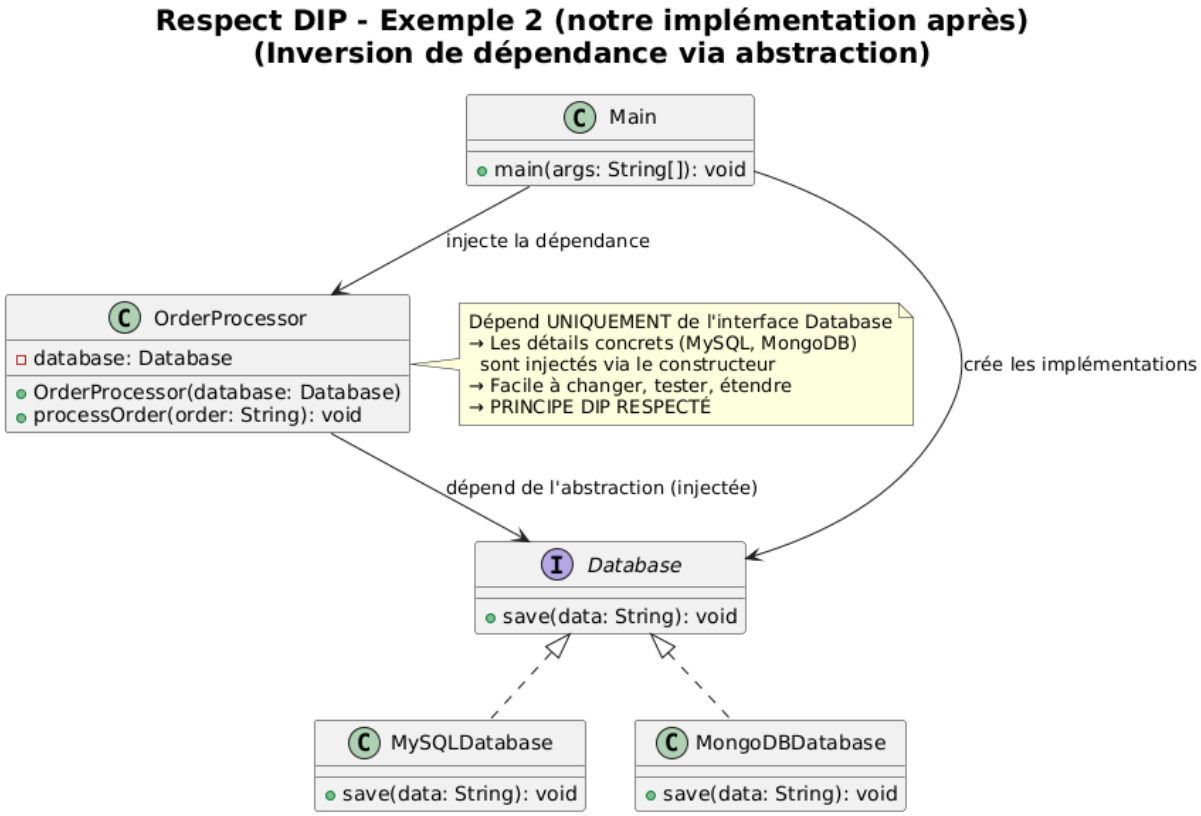


5.2- Exemple 2

- Violation DIP - Avant Refactoring



- Respect DIP - Après Refactoring



Les principes SOLID constituent les fondations d'une conception logicielle robuste, maintenable et extensible.