TP 4 - Héritage

Objectif pédagogique

Utilisation de la notion d'héritage, remise en cause de la conception d'une classe pour la rendre extensible et ainsi plus réutilisable.

Prérequis

TP 1, 2, 3 et notes du cours d'initiation à la POO.

Sujet

Au lieu de reprendre les principes de la classe `Ensemble` par duplication (copier/coller) dans de nouvelles classes, on se propose de mettre plutôt en place une hiérarchie de classes réutilisables.

Ces classes permettront de mettre en œuvre l'annuaire de manière aisée et rapide par dérivation de cette base.

Q0) Mise en œuvre de la classe EnsembleP

On réalisera (et on testera) d'abord la mise en œuvre d'une classe `EnsembleP` modélisant les ensembles de personnes. On pourra bien sûr reprendre les mêmes algorithmes que ceux de `Ensemble`.

Q1) Mise en œuvre de la classe Annuaire

Adapter (et tester) alors la classe `Annuaire`, en ramenant sa programmation au strict nécessaire par dérivation de la classe `EnsembleP`.

Pour permettre cette réutilisation par une classe dérivée, on pourra remettre en cause adéquatement le statut des attributs de la classe de base qui sont pour l'instant privés et inaccessibles.

Q2) Une variante d'ensemble de personnes : la classe EnsembleTriéP

On constate que, concernant l'efficacité, le choix de baser l'annuaire sur un simple ensemble n'est pas le plus judicieux.

En effet une recherche dans un tel annuaire (une tâche fréquente) fera un parcours exhaustif, ce qui est coûteux et rédhibitoire pour un véritable annuaire contenant un nombre d'éléments important. Au contraire, si les éléments étaient triés, la gestion de l'ensemble pourrait être améliorée, car dans un ensemble trié :

- l'ajout serait un peu plus lent qu'auparavant ;

- la recherche pourrait en revanche être programmée de manière très accélérée.

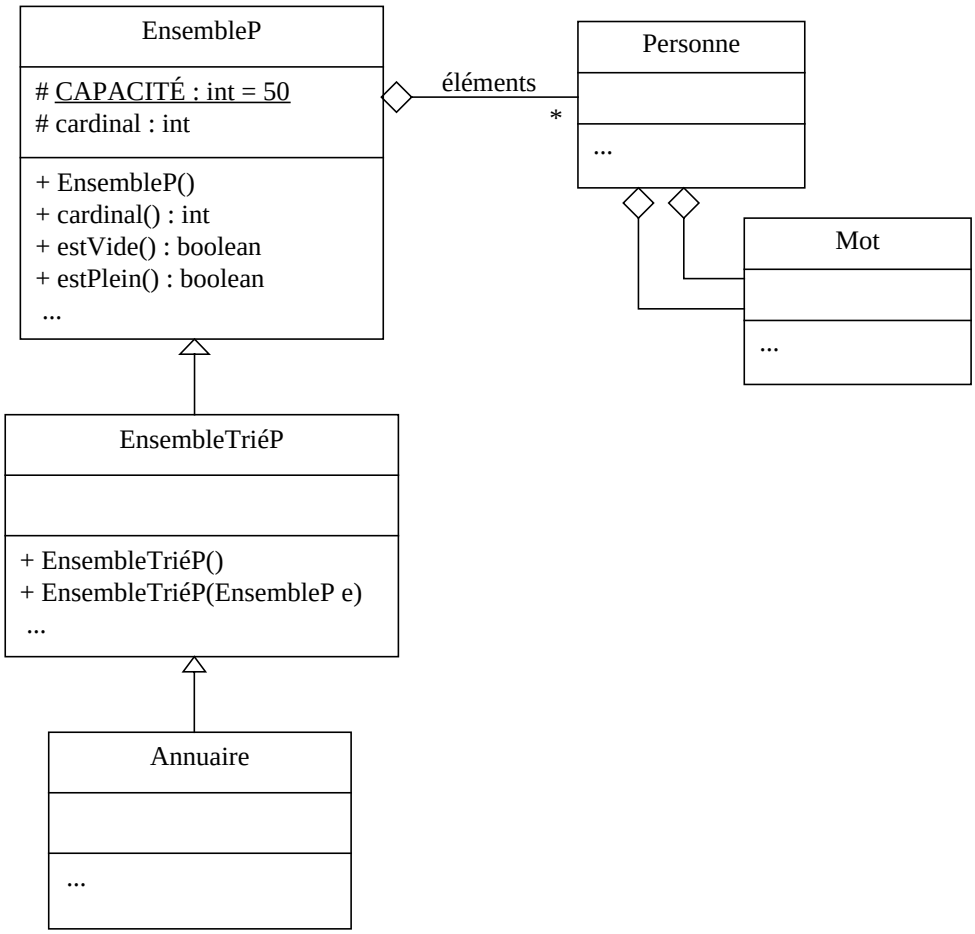
Cette situation est évidemment favorable pour un annuaire, où les ajouts sont peu fréquents par rapport aux recherches.

1) Pour pouvoir classer les éléments il faut pouvoir les comparer : ajouter une méthode `inférieurA(...)` dans la classe `Personne` afin de savoir si une personne se classe avant une autre ou pas.

2) Définir (et tester) alors une nouvelle variante de mise en œuvre d'ensemble de personnes, la classe `EnsembleTriéP`, par dérivation de la classe de base `EnsembleP` et transformant uniquement ce qui doit l'être pour tirer parti d'un tri des éléments dans la représentation interne.

Q3) Révision finale de la classe Annuaire

Renoncer alors à dériver `Annuaire` directement de `EnsembleP`, ce qui était un choix peu efficace, mais plutôt faire dériver cet annuaire de `EnsembleTriéP` pour tirer parti sans effort du gain en efficacité apporté par l'ensemble trié.



Remarques

Ce document se trouve sous `/home/clyde/d1/insa3/commun/java-sdd/tp4`