

Programmation Orientée Objets avec Java

Projet POO

Stéphane Malandain / Yassin Rekik
Semestre d'automne 2024

Introduction

Dans le cadre du cours de Poo / Java, vous allez réaliser un projet entier. Ce travail se divise en 4 étapes. Chaque étape fait l'objet d'un rendu à une date fixe, avec une note. La moyenne des 4 notes vous donne une note de projet.

Le projet est individuel. Plagiat ou copie seront sévèrement puni ! Les modalités de dépôt vous seront communiquées sur le Git ultérieurement.

1 Première étape : les bases d'un gestionnaire de tâches -> Terminé !

Ce projet consiste en la création d'un gestionnaire de tâches. Par définition, une tâche est un travail qui doit être effectué dans un temps donné. Les tâches peuvent être donc très différentes et très variées, réalisées par une, 2 ou plusieurs personnes.

2 Seconde étape : héritage, interface et collections.

2.1 Différents types de tâches :

- `Task` : Classe abstraite de base définissant les attributs et le comportement de base d'une tâche. Elle définit deux méthodes abstraites : `display()` (affichage de la tâche) et `toString()`.
- `T_Base` : Tâche de base qui hérite de `Task`
- `T_Prio` : Tâche avec un critère de priorité (implémente l'interface `Priority` et hérite de `Task`).
- `T_Cal` : Tâche avec une date limite (pas besoin de l'heure). Elle implémente l'interface `Agenda` et hérite de `Task`.
- `T_Cal_Prio` : Tâche ayant une priorité et une date limite

Vous devez définir les deux interfaces :

- `Priority` : impose les méthode `getPriorite(...)` / `setPriorite(..)`
- `Agenda` : impose méthode `getDate(...)` / `setDate(...)`

Implémenter les classes `Task`, `T_Base`, `T_Prio`, `T_Cal`, `T_Cal_Prio`.

2.2 Tri des tâches

On souhaite avoir la possibilité de comparer et de trier les différentes tâches. Pour cela, il faut implémenter l'interface `Comparable` décrite ci-après :

L'interface `Comparable` modélise les objets qui possède un ordre total : Elle déclare une seule méthode `compareTo` :

```
1 interface Comparable {  
2     int compareTo(Object o);  
3 }
```

Cette méthode retourne un entier selon que l'objet auquel elle est appliquée est plus grand, est égal ou est plus petit que o.

Définir ainsi une méthode `sort` pour chacune des classes tâches.

Les éléments de la classe `T_Base` seront triés selon leur `Id`;

Les éléments de la classe `T_Prio` seront triés selon leur priorité puis leur `Id` si besoin;

Les éléments de la classe `T_Cal` seront triés selon leur date limite, puis leur `Id` si besoin;

Les éléments de la classe `T_Cal_Prio` seront triés selon la priorité, la date limite puis l'`id`;

2.3 Changez la structure de donnée pour le stockage des tâches :

Vous devez remplacer le tableau des tâches par une collection type `ArrayList`. Crée la classe `TasksList` permettant de stocker vos tâches. Adaptez les fonctionnalités définies dans l'étape 1 avec cette nouvelle structure de données.

2.4 Adaptez le menu ; nous devons pouvoir désormais créer et stocker conjointement les différentes tâches à notre guise dans la même liste. Ajouter le menu pour afficher les tâches triés.

3 Troisième étape : A Préciser ultérieurement