COMPLÉMENT AU CAHIER DES CHARGES

Nous expliquerons les différents choix que nous avons pris au cours de ce projet.

La classe Utilisateur

Elle représente l'utilisateur qui utilise l'application. Un utilisateur possède une liste de tâches et une liste de catégories. Elle s'occupe de sérialiser toutes les données.

La classe Categories

Elle représente un ensemble de catégories qui sont des chaînes de caractère. Chaque tâche possède alors un attribut qui est un élément de cette liste de catégories.

Placer cette liste de catégories dans la classe Tache n'aurait pas été judicieux ; non seulement on aurait eu une duplication des informations et l'ajout, la suppression et la modification des catégories aurait été moins efficace, (d'où l'importance d'une classe Utilisateur).

Organisation de la liste des choses à faire

ListeTache est une classe qui a en attribut une collection de tâches. Elle gère l'ajout, la suppression, la modification de tâches et le tri de ces dernières : les différentes méthodes de tris sont des méthodes de cette classe. Le tri par date d'échéances est réalisé grâce à un objet implémentant l'interface **Comparable** et le tri par date d'échéances intermédiaires grâce à un objet implémentant l'interface **Comparator**. Ces objets permettent d'imposer un ordre de tri.

Le modèle MVC

Ce modèle permet une conception propre en séparant la couche métier de la partie graphique.

1) Le modèle

Il s'agit de la classe Utilisateur composée des collections de tâches et de catégories.

2) La vue (classe TacheView)

C'est l'interface graphique. Elle comprend tous les composants swing (JPanel,JButton,etc)

3) Le contrôleur (classe TacheController)

Il fait le pont entre la vue et le contrôleur. Il possède donc en attribut le modèle et la vue. Le modèle et la vue communiquent au moyen d'écouteurs d'évènements qui sont placés sur les différents éléments de la vue par le biais d'accesseurs en lecture