Projet de Fin de Module

Programmation Orientée Objet

A. El Hibaoui

2023

TABLE DES MATIÈRES

1 Chapitre 1
Projet de Fin de Module – Gestionnaire de Tâches

1.1 Objectif *1*
1.2 Concepts Java à mettre en œuvre *1*
1.3 Caractéristiques possibles *1*
1.4 Remise du Projet *2*
1.4.1 Étapes à suivre *2*
1.4.2 Dates Importantes *2*

2 Chapitre 2
Annexe - Gestionnaire de Tâches

2.1 Gestionnaire de Tâches en Console avec Java *3*
2.1.1 Création de la classe Task : *3*
2.1.2 Création de la classe TaskManager : *4*
2.1.3 Classe principale (Main) : *4*
2.1.4 Gestion de la persistance des données : *5*
2.1.5 Notifications : *6*
2.1.6 Améliorations possibles : *6*
2.2 Quelques liens utiles *7*

CHAPITRE

PROJET DE FIN DE MODULE – GESTIONNAIRE DE TÂCHES

1

1.1 Objectif

Le but de ce projet est de se familiariser avec la Programmation Orientée Objet. Pour ce faire, vous êtes amenés à créer une application de gestion de tâches qui permet aux utilisateurs de créer, éditer, supprimer et suivre leurs tâches quotidiennes.
L'application doit offrir une interface utilisateur conviviale en ligne de commande ou, si vous vous sentez ambitieux, une interface graphique.

1.2 Concepts Java à mettre en œuvre

— Classes et objets
— Collections (Listes, en particulier)
— Gestion d'exceptions
— Entrées/Sorties ou Gestion de Bases de données (pour la sauvegarde des données)
— Interfaces utilisateur (en cas d'usage d'interface graphique)

1.3 Caractéristiques possibles

— **Ajout de tâches :** Les utilisateurs peuvent ajouter des tâches avec des détails tels que le nom, la description, la date d'échéance, la priorité, etc.

— **Liste de tâches :** Affiche la liste des tâches actuelles avec des options de tri basées sur la date d'échéance, la priorité, etc.

— **Modification et suppression :** Permet aux utilisateurs de modifier ou supprimer des tâches existantes.

— **Marquage des tâches terminées :** Possibilité de marquer une tâche comme terminée.

— **Enregistrement des données :** Utilise la sérialisation/désérialisation ou une base de données simple pour enregistrer les tâches persistantes entre les sessions.

— **Notifications :** Ajoute une fonctionnalité de notification pour rappeler aux utilisateurs des tâches à venir.

1.4 Remise du Projet

1.4.1 Étapes à suivre

1. Créez un répertoire sur votre Drive dont le nom est de la forme `'NomCompletÉtudiant'`.
2. Partagez ce répertoire avec quiconque possède le lien.
3. Glissez le code source de votre application dans ce répertoire.
4. Préparez une présentation vidéo dans laquelle vous expliquez le travail réalisé. Il est fortement recommandé que votre présentation inclue une modélisation UML de l'application, une démonstration de l'application.
   La durée de cette vidéo sera entre **7 à 10 minutes** et n'oubliez pas de la glisser également dans votre répertoire.
5. Remplissez le formulaire ci-dessous. Formulaire de Soumission du Projet Java - IWA 2023

https://docs.google.com/forms/d/e/1FAIpQLSdsGNMnA9i2nTtYOLKWU1zWsH_BZ6mW7n-apXz4ge2VzplDdg/viewform

1.4.2 Dates Importantes

— Date limite de soumission du projet : **20 Décembre 2023.**

— Date de présentation du projet : **à partir du 21 Décembre 2023.**

CHAPITRE 2

ANNEXE - GESTIONNAIRE DE TÂCHES

2.1 Gestionnaire de Tâches en Console avec Java

2.1.1 Création de la classe Task :

1. **Attributs :** nom, description, date d'échéance, priorité, statut (terminé ou non).
2. **Méthodes :** constructeur, getters/setters.

```java
import java.time.LocalDate;

public class Task {
    private String name;
    private String description;
    private LocalDate dueDate;
    private int priority;
    private boolean completed;

    public Task(String name, String description, LocalDate dueDate, int priority)
    {
        ....
    }

    // Getters and setters...

    public void markAsCompleted() {
        ...
    }

    @Override
    public String toString() {
       ...
    }
}
```

2.1.2 Création de la classe TaskManager :

1. **Attribut :** Liste de tâches (utilise ArrayList<Task> par exemple).
2. **Méthodes :**
   - ‘`addTask(Task task)`’ : ajoute une tâche à la liste.
   - ‘`removeTask(Task task)`’ : supprime une tâche de la liste.
   - ‘`displayTasks()`’ : affiche la liste des tâches.
   - ‘`displayTaskDetails(Task task)`’ : affiche les détails d’une tâche spécifique.
   - ‘`markTaskAsDone(Task task)`’ : marque une tâche comme terminée.

```java
import java.util.ArrayList;
import java.util.List;

public class TaskManager {
    private List<Task> tasks;

    public TaskManager() {
        ...
    }

    public void addTask(Task task) {
       ...
    }

    public void removeTask(Task task) {
        ...
    }

    public void displayTasks() {
       ....
    }

    public void displayTaskDetails(Task task) {
       ...
    }

    public void markTaskAsDone(Task task) {
      ....
    }
}
```

2.1.3 Classe principale (Main) :

Créez un menu interactif en console avec des options numérotées pour que l’utilisateur puisse choisir.

- Initialise une instance de TaskManager.
- Boucle principale pour l’interaction avec l’utilisateur.
- Affiche un menu avec des options telles que "Ajouter une tâche", "Afficher les tâches", etc.
- Utilise un scanner pour capturer l’entrée de l’utilisateur.
- Implémente des méthodes pour gérer chaque option du menu.

```java
import java.time.LocalDate;
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        TaskManager taskManager = new TaskManager();
        Scanner scanner = new Scanner(System.in);

        while (true) {
            System.out.println("1. Ajouter une tache");
            System.out.println("2. Afficher les taches");
            System.out.println("3. Afficher les details d'une tache");
            System.out.println("4. Marquer une tache comme terminee");
            System.out.println("0. Quitter");

            int choice = scanner.nextInt();
            ...

            switch (choice) {
                case 1:
                    ....
                    break;

                case 2:
                    ....
                    break;

                case 3:
                    ....
                    break;

                case 4:
                    ....
                    break;

                case 0:
                     ....
                    break;

                default:
                    System.out.println("Choix non valide.");
            }
        }
    }
}
```

2.1.4 Gestion de la persistance des données :

Utilise la sérialisation/désérialisation ou une base de donnée pour sauvegarder et charger la liste des tâches depuis un fichier ou depuis une base de données.

1. **En utilisant les fichiers**

```java
...
public class TaskManager implements Serializable {

    // ... Autres methodes de TaskManager ...

    // Methode pour sauvegarder la liste des taches dans un fichier
```

```
    public void saveTasksToFile(String fileName) {
        ....
    }

    // Methode pour charger la liste des taches depuis un fichier
    public void loadTasksFromFile(String fileName) {
        ....
    }
}
```

2. **En utilisant une Base de données**

```
....
import java.sql.*;

public class TaskManager {
    private List<Task> tasks;
    private Connection connection;

    // ... Autres methodes de TaskManager ...

    // Methode pour etablir la connexion a la base de donnees
    public void establishConnection() throws SQLException {
        String url = "jdbc:mysql://localhost:3306/votre_base_de_donnees";
        String username = "votre_nom_utilisateur";
        String password = "votre_mot_de_passe";

        connection = DriverManager.getConnection(url, username, password);
    }

    // Methode pour sauvegarder la liste des taches dans la base de donnees
    public void saveTasksToDatabase() throws SQLException {
        ....
        }
    }

    // Methode pour charger la liste des taches depuis la base de donnees
    public void loadTasksFromDatabase() throws SQLException {
        tasks = new ArrayList<>();
        ....
    }
}
```

2.1.5 Notifications :

Implémentez une fonctionnalité simple de rappel pour les tâches à venir. Par exemple, affiche un message lorsque l'utilisateur lance l'application.

2.1.6 Améliorations possibles :

Ajoutez à l'utilisateur la possibilité de

— définir la priorité de chaque tâche.

— rechercher des tâches par nom, date, priorité, etc ;

— modifier les détails d'une tâche existante.

2.2 Quelques liens utiles

1. **Video liée 01 :** Java JDBC tutorial | Java Database Connectivity | Java Tutorial For Beginners | Simplilearn https://www.youtube.com/watch?v=3OrEsC-QjUA
2. **Video liée 02 :** WindowBuilder Java Créer Une Interface Graphique Sans Code
   https://www.youtube.com/watch?v=QAAZlJQcQDs
3. **Vidéo liée 03 :** Java Eclipse GUI Tutorial 9 # Populate JTable data from database in java Eclipse and Sqlite
   https://www.youtube.com/watch?v=6cNYUc2PIag
4. **Vidéo liée 04 :** JAVA - How to Create a Table with JTable in Eclipse
   https://www.youtube.com/watch?v=pybU3E-eKfw
5. **Vidéo liée 05 :** Install4J Full Advanced Tutorials by GOXR3PLUS Studio
   https://www.youtube.com/watch?v=CsACTkyQygc