Rapport TPFinal POO

Application de gestion d'événements

Ngapgou Farelle
3ème année Génie Informatique
ENSPY

Mai 2025

1. Introduction

Ce projet a pour objectif de développer une application de gestion d'événements (concerts, conférences, etc.), permettant à un organisateur de créer, modifier, rechercher et supprimer des événements, avec un système de persistance des données.

2. Choix de conception

2.1 Paradigme orienté objet

Le projet est conçu selon le paradigme de la programmation orientée objet. Les entités principales (Événement, Organisateur, etc.) sont modélisées sous forme de classes avec des relations d'héritage et de composition.

2.2 Utilisation de patrons de conception

Nous avons utilisé le patron Singleton pour la classe `GestionEvenements`, garantissant une seule instance partagée dans toute l'application pour gérer les événements.

2.3 Sérialisation JSON

Les événements sont sauvegardés et rechargés à partir d'un fichier JSON à l'aide de la bibliothèque `Jackson`, afin de garantir une persistance simple et portable.

2.4 Interface utilisateur

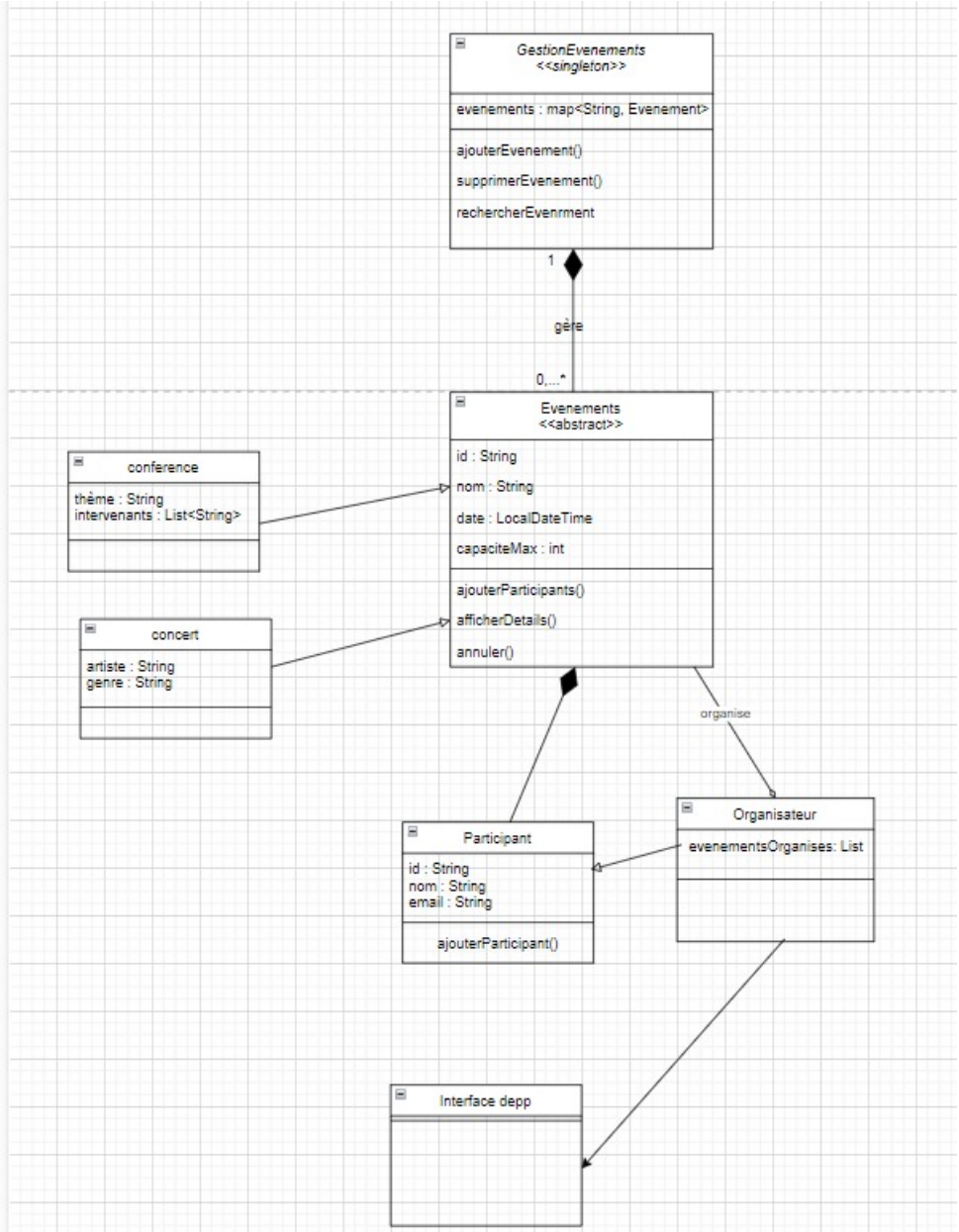
L'interface a été développée avec Java Swing (ou JavaFX si applicable), permettant la saisie des événements et organisateurs via des formulaires simples.

3. Architecture logicielle

L'application est structurée autour des entités suivantes :

- **Evenement** (classe abstraite) : classe de base pour tous les types d'événements.
- **Concert** et **Conference** : classes dérivées avec des champs spécifiques.
- **Organisateur** : personne créant l'événement.
- **GestionEvenements** : Classe Singleton to centralize the operations.

Les classes communiquent entre elles via des objets bien encapsulés. diagramme Uml :



4. Cas d'utilisation

4.1 Création d'un événement

L'utilisateur saisit les données de l'organisateur et les informations de l'événement. Les champs sont validés et une instance est créée, puis ajoutée au gestionnaire.

4.2 Ajout d'un participant

Le participant est ajouté à la liste si la capacité n'est pas dépassée. En cas de dépassement, une exception personnalisée est levée.

4.3 Sauvegarde et chargement

La sauvegarde s'effectue automatiquement après création, et le chargement se fait au lancement à partir du fichier `evenements.json`.

5. Avantages de cette conception

- Séparation claire des responsabilités
- Réutilisabilité du code (ex : les types d'événements peuvent être étendus)
- Utilisation de standards modernes (JSON, POO)
- Bonne maintenabilité

6. Limites et pistes d'amélioration

- Ajout d'une base de données pour gérer de grands volumes
- Interface graphique plus avancée avec JavaFX ou framework web
- Authentification de l'utilisateur

7. Conclusion

Ce projet a permis de mettre en pratique les principes de la programmation orientée objet en Java, en construisant une application modulaire, maintenable et fonctionnelle avec persistance. Il peut être enrichi facilement par de nouvelles fonctionnalités.