Rapport de Projet :

Gestion des Images et de leurs Métadonnées

Sujet :

Développement d'une application permettant de gérer des images et leurs métadonnées avec une interface CLI et GUI.

Présenté par :

- Ouardia
- Fariza

Groupe : B8

Responsable de Formation : LEMAIRE Marc

Date : 16 janvier 2025

Sommaire

1-Introduction

Ce projet a pour objectif de développer une application permettant de gérer des images et leurs métadonnées. Nous avons choisi d'intégrer deux modes d'utilisation : une interface en ligne de commande (CLI) et une interface graphique (GUI). L'application doit être compatible avec les formats JPEG, PNG et WEBP [1]et proposer des fonctions d'exploration, d'extraction et d'affichage des métadonnées.

L'objectif principal est de mettre en pratique les concepts de Programmation Orientée Objet (POO) en Java tout en livrant une solution fonctionnelle et bien structurée.

1.1-Contexte

Les images numériques sont omniprésentes dans notre vie quotidienne, et leur gestion efficace est essentielle pour des domaines variés tels que le Web, la photographie et les applications mobiles. Les formats pris en charge par ce projet offrent une diversité de solutions pour la compression et l'ajout de métadonnées :

- **JPEG** : Format largement utilisé pour sa compression avec perte, réduisant efficacement la taille des fichiers tout en conservant une qualité acceptable.
- **PNG** : Idéal pour les images de haute qualité sans perte, souvent utilisé pour les graphismes et illustrations.
- **WEBP** : Format moderne combinant compression avec ou sans perte, optimisé pour le Web.

Les métadonnées telles que EXIF et XMP enrichissent ces fichiers avec des informations supplémentaires comme la date, la résolution ou la localisation GPS. Ces informations sont essentielles pour organiser et exploiter les collections d'images.

---

1 Les formats JPEG, PNG et WEBP ont été choisis en raison de leur adoption généralisée et de leur compatibilité avec une variété d'applications et de plateformes.

1.2-Objectifs du Projet

Ce projet vise plusieurs objectifs clés :

1. **Simplifier la gestion des images** : Fournir un outil convivial pour explorer, lister et afficher des images et leurs métadonnées.
2. **Améliorer la qualité des données** : Assurer la précision des informations extraites, notamment les dimensions, la résolution et les données GPS.
3. **Mettre en œuvre les principes de POO** : Structurer le code de manière modulaire et maintenable, facilitant les éventuelles extensions futures.
4. **Offrir une compatibilité multiplate-forme** : Garantir le bon fonctionnement de l'application sur différents systèmes, grâce à Java SE 21.

2-Diagrammes UML

2.1-Diagramme de cas d'utilisation[2]

Ce diagramme illustre les interactions entre l'utilisateur et les principales fonctionnalités de l'application, notamment :

- Lister les images dans un répertoire.
- Extraire des statistiques globales.
- Afficher les métadonnées d'une image spécifique.
- Prendre un snapshot.
- Consulter un snapshot.
- Naviguer dans les dossiers via l'interface graphique.

Ces interactions permettent une gestion intuitive et efficace des fichiers image, quel que soit le mode d'utilisation choisi.

2.2-Diagramme de Classes

Le diagramme de classes du projet comprend les principales entités nécessaires pour gérer efficacement les images et leurs métadonnées. Voici les classes incluses :

- **ApplicationCLI** : Représente l'interface en ligne de commande, permettant à l'utilisateur d'interagir avec les fonctionnalités via des commandes.

2Le diagramme de cas d'utilisation met en lumière les principales fonctionnalités de l'application, telles que définies dans les exigences fonctionnelles.

- **ApplicationGUI** : Implémente l'interface graphique pour une navigation plus intuitive.
- **Répertoire** : Gère les opérations sur les répertoires, notamment la liste des fichiers et la navigation dans l'arborescence.
- **Fichier** : Modélise un fichier générique avec des propriétés comme le nom, la taille et la date de modification.
- **FichierImage** : Hérite de la classe `Fichier` et ajoute des propriétés spécifiques aux images, comme les dimensions et le type MIME.
- **Statistique** : Fournit des méthodes pour calculer les statistiques globales sur un répertoire (nombre total d'images, répartition par format, etc.).
- **Métadonnées** : Gère l'extraction et la représentation des métadonnées (EXIF, XMP), y compris les dimensions, la résolution et les balises GPS.

Ces classes interagissent pour offrir une solution robuste et modulaire, facilitant la gestion des fichiers image dans les deux modes (CLI et GUI).

Tableau 1: Exemple de données sur les images et leurs métadonnées.

Nom de l'image	Format	Résolution	Date
photo1.jpg	JPEG	1920x1080	2025-01-01
image2.png	PNG	2560x1440	2025-01-02
graphique.webp	WEBP	1280x720	2025-01-03

3-Organisation et Planning

3.1-Répartition des tâches

La répartition des tâches a été organisée comme suit, en fonction des responsabilités et des phases du projet :

Ouardia	Fariza	Collaboration
Lister les images avec vérification MIME et extension	Lister les images avec vérification MIME et extension	Élaboration des diagrammes UML.
Extraction de métadonnées	Exploration visuelle des images et leur métadonnées	Recherche et documentations sur les formats d'images et métadonnées
Visualisation et filtres avancés	Comparaison des snapshot en GUI	Test et rédaction du rapport final
Comparaison des snapshots en CLI	Recherche et filtre avancés d'images par critères	Vidéo de démonstration

Pour voir les résultats techniques des tâches assignées, référez-vous à la section 4-Résultats Techniques

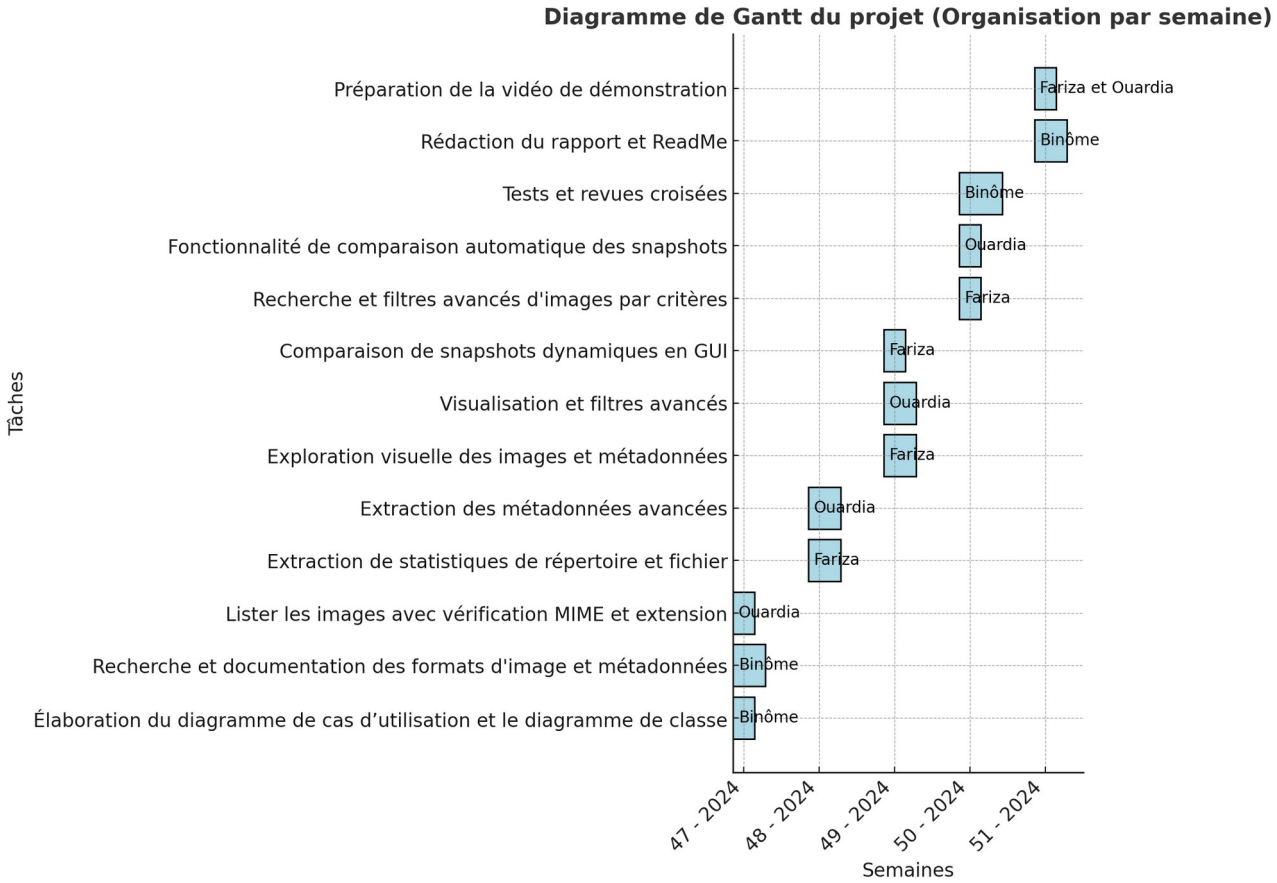

Illustration 1: Un diagramme de Gantt détaillé, présenté ci-dessus, illustre les étapes du projet et leurs durées respectives :

Le diagramme de Gantt qui a était réalisé par GanttProject , permet de visualiser clairement l'organisation et la progression des tâches, tout en mettant en avant les contributions spécifiques de chaque membre du binôme.

4-Résultats Techniques

4.1-Fonctionnalités Implémentées

4.1.1-Mode Console (CLI) :

- Lister les images dans un répertoire spécifique, avec une vérification des extensions et types MIME.
- Extraire et afficher les métadonnées d’une image donnée, incluant la résolution, la taille et les balises GPS.
- Afficher des statistiques globales sur un répertoire, comme le nombre total d'images par format.

Pour voir des exemples visuels de l'interface en ligne de commande, voir la section 5.1-Interface CLI :

4.1.2-Mode Graphique (GUI) :

- Navigation intuitive dans les répertoires avec affichage d'une arborescence des fichiers.
- Affichage des images et de leurs métadonnées dans une interface détaillée.
- Visualisation des miniatures directement dans l'interface.

Pour voir des captures d'écran de l'interface graphique, voir la section 5.2-Interface GUI :

Ces fonctionnalités offrent une expérience utilisateur riche, adaptée aux besoins de gestion d'images.

4.2-Difficultés Rencontrées

- La gestion des métadonnées XMP a requis l’intégration de bibliothèques externes, notamment Metadata et XMPCore, pour garantir une extraction fiable des informations.
- Une difficulté majeure a été de décider comment implémenter la fonctionnalité `snapshotCompare`. Une question clé était de savoir s'il fallait ajouter une nouvelle classe `Snapshot` pour gérer cette fonctionnalité ou simplement intégrer une méthode `snapshotCompare` directement dans la classe `Répertoire`. Cette décision a nécessité une analyse approfondie des implications sur la structure du code et la modularité.

5-Captures d'Écran

5.1-Interface CLI :

```
C:\Users\cylia\eclipse-workspace>java -jar cli.jar -d ..\OneDrive\pictures --stat
Statistiques des fichiers du répertoire :
Total de fichiers : 13
Total de fichiers image : 10
Nombre d'images PNG : 3
Nombre d'images JPEG : 5
Nombre d'images WEPB : 2

C:\Users\cylia\eclipse-workspace>java -jar cli.jar -d ..\OneDrive\pictures --list
Liste des fichiers images :
angers.jpg
boutique.png
Chaton.jpeg
dior.png
```

Illustration 2: Une capture montrant les commandes pour afficher les statistiques globales et lister les fichiers.

5.2-Interface GUI :

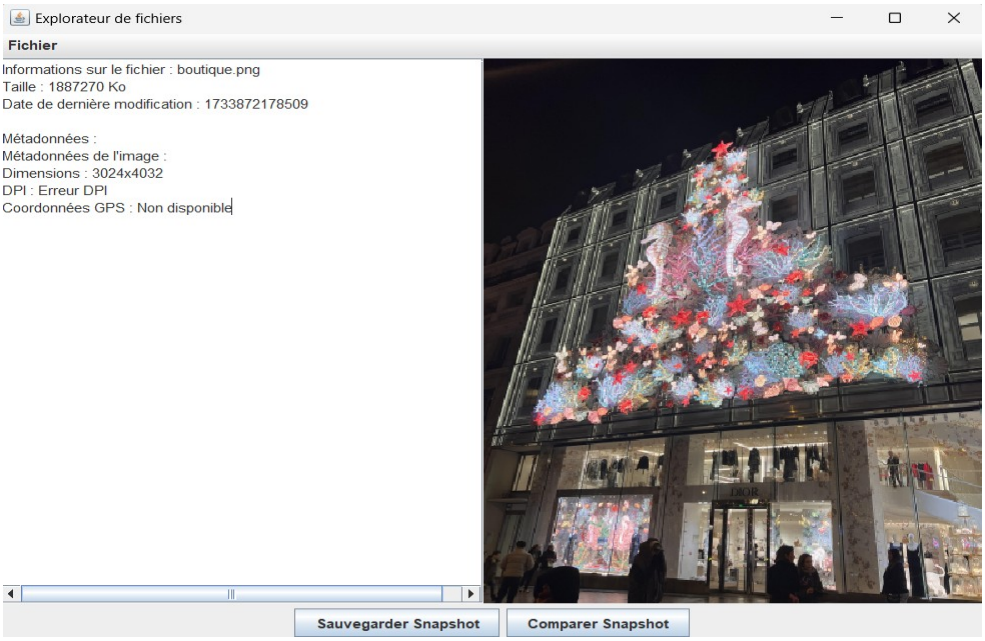

Illustration 3: Une capture illustrant l'affichage d'une image avec ses métadonnées.

6-Conclusion

Le projet a atteint ses objectifs principaux. Nous avons implémenté une application fonctionnelle en CLI et GUI permettant de gérer les images et leurs métadonnées. Les concepts de POO ont été appliqués de manière rigoureuse, et la documentation Javadoc fournit une référence claire pour les développeurs.

6.1-Points d'Amélioration

Pour aller plus loin, nous pourrions :

- Ajouter la compatibilité avec d'autres formats d'image (GIF, BMP).
- Implémenter des fonctions d'édition des métadonnées, comme l'ajout de titres ou la modification des descriptions.
- Optimiser l'interface graphique pour inclure des options avancées de recherche et de tri des images.

6.2-Perspectives[3]

L'application pourrait être enrichie par des fonctionnalités comme l'intégration avec des services de cloud pour la sauvegarde et le partage d'images. De plus, une version mobile pourrait être développée pour une utilisation sur smartphones et tablettes.

3L'intégration avec des services cloud pourrait inclure des fonctionnalités comme la synchronisation automatique des images et des sauvegardes programmées.

6.3-Contributions à la Communauté

Ce projet offre une base solide pour développer des outils similaires. La documentation claire et l’architecture bien pensée permettent à d’autres développeurs d’adapter et d’étendre l’application selon leurs besoins.

7-Annexes

- Documentation Javadoc.
- Fichier `readme.md` incluant les instructions d’installation et d’utilisation.
- Codes source organisés par module.
- Fichiers `.jar` pour les modes CLI et GUI.

Date de livraison : 29 décembre 2024