ALAIMO Julien, FEYDEL Hugo, HUMBERT Thibaud, KEKA Enver,

Compte rendu TP Vidéoclub

Architecture du projet:

- Github : https://github.com/A-Julien/tp_coo_M1_KAL2000
- CI (Travis) : https://travis-ci.com/A-Julien/tp_coo_M1_KAL2000
- Compte-Rendu : `CR.pdf`
- JavaDoc : `doc`
- Diagramme de classe : `VideoClubUMLFinal.png`
- SRS : `README.md`
- Scenari de tests : `Scenari`

Gestion de projet

Nous nous sommes organisé avec un git et une CI sur Travis. Nous n'avons pas eu le temps de réaliser beaucoup de tests. Les seules tests existant sont les scnénari. Pour avoir accés aux logs des tests il suffit de cliquer sur le bouton `build` sur la page du Github.

Organisation du code :

Nous avons choisi d'organiser notre code selon un modèle MVC. Pour simuler la base de données, nous utilisons des fichiers dans lesquels nous enregistrons les modifications aux films, clients et dvds. Ces fichiers sont produit grâce à la serialisation et des trois tableau formant la base de donnée (grâce aux classes `ObjectInputStream(), ObjectOutputStream()`)

- Le Contrôleur:
  - Il simule le fonctionnement de la machine permettant de louer les dvds dans la classe `Kal2000` et son système de comptes pour les clients grâce aux classes `UserInterface`.
  - `Kal2000` contient la base de données et gére la lecture et l'enregistrement de celle-ci grâce aux methodes `powerOff(), boot()`.
  - `UserInterface` gére la connection et l'identification du client. Elle gére aussi la detection de la connection d'un adminisatrateur. Pour simplifier la gestion des droits, si on se connect avec l'identifiant `12345` et le mot de passe `12345` on passe en mode administateur.

- Le Modèle :
  - Les paquetages constituant le modèle sont les suivants :
  - `Card` gère et défini les différents types de cartes
  - `Client` défini le modèle de données des clients
  - `Exception` défini les différents types d'exceptions pouvant être générées par l'utilisation du programme
  - Movies défini la gestion des films et des dvds
  - Rent défini la gestion des locations de dvds

- La Vue:
  - Elle simule les différentes interfaces de la machine de location en version textuelle :
  - `ConnectionView` est l'écran de base de la machine , elle simule l'interface de connexion et la création de comptes clients.
  - `AdminView` communique les options disponibles à l'administrateur du système s'il s'est connecté en tant que tel. Il peut par exemple ajouter ou retirer des films et/ou dvds.
  - `UserView` affiche les options disponibles aux utilisateurs selon s'ils sont abonnés ou non. Elle permet par exemple aux clients de gérer leurs locations de films.

Conclusion sur l'évolution du projet :

Nous avons remarqué au cours du projet que notre spécification UML de départ n'était pas assez développée et ne nous permettait pas d'intégrer toutes les fonctionnalités demandées par le client. C'est pour palier à ça que nous avons ajouté des classes et des dépendances par la suite. (Voir l'UML ci-joint pour les détails)