Rapport de Projet

1. Introduction

- Présentation rapide du projet et du contexte.
  Dans le cadre de notre projet pour l'ue programmation orientée objet ou on devait réaliser un jeux, pour cela on a opté pour une extension de notre projet du premier

  semestre qui était basé sur un jeu d'exploration spatial ou on pouvait naviguer entre différentes planètes

2. Choix de conception

- MVC : séparation claire modèle / vue / contrôleur.

  Modèle :

  Map :

  On a opté pour une grille classique contenant les informations telles que l'url de l'image de l'objet courant, sont état d'occupation …

  Un contrôleur central reçoit les messages de la map et met à jour la vue en conséquence.

  Monde :

  Le monde quant à lui est constitué d'un ensemble de map

  Autres:

  D'autres classes sont également précédentes pour la gestion des personnages et des items

  Vue:

  Il y'a 3 sections essentielles, une gérant, la sélection des items à placer sur la map,

  une seconde affichant le jeu (matrice) et la dernière affichant les statistiques du jeu

  Contrôleur:

C’est le pont entre le modèle et la vue, il est abonné aux messages provenant des modèles et met à jour la vue.

- Format JSON pour sauvegarde.
    - La sauvegarde du monde et des score ce fait au format json dans un dossier json
        - le jeu par défaut est dans **defaultLevels.json**
        - le dernier jeu exporté est dans **levels.json**
- Déplacements :
    - Les déplacements sont gérés par une file stockant les actions pour les différentes entités, ils sont synchronisés par une horloge qui traite les actions individuellement.

3. Difficultés et solutions

- La gestion et déplacement des items nous a posés de nombreux problèmes de synchronisation entre toutes ces entités ce qu’on a résolu en utilisant des mécanismes de files sûres et par le positionnement des verrous.

- La persistance des données :
    - L’usage du module jackson a été particulièrement contraignant car lors de l’exportation, certains objet n’étaient pas correctement exportés, du coup on a dû utiliser des annotations pour préciser comment les exportés correctement, et aussi des mécanisme de conversion automatique interne pour les autres:

4. Gestion de projet

- Outils : GitHub, Intellij, PlantUml, Whatsapp…

- Planning : phases (conception → dev → tests).

- Répartition :
    - Clara: principalement responsable de la persistance des données et la validation des tests

      Samy: en charge de l’interface utilisateur et des données

      Marcel : principalement en charge de la logique de déplacements des items mobiles

      Liban: principalement en charge de l’intégration global du code et des testes

Pour gagner un maximum d’expériences dans chaque aspect on effectuer des rotations régulières en suivant également une méthodologie scrum.