

BERNARD Geoffrey
LAGANE Alexandre
Binôme : B24

Projet de Programmation

JAVA

I. Introduction

Dans le cadre de nos études en L3 Informatique, nous avons eu l'occasion de développer en Java lors d'un projet de programmation.

Le principe de l'application étant de confronter les personnages de chaque équipe (étudiants de la licence) dans une arène. Nous devons donc réfléchir à des stratégies afin de remporter ce tournoi ! Une occasion d'avoir un aperçu sur le monde du travail et d'améliorer notre expérience dans le langage Java.

Ce rapport sera constitué de trois parties : une première partie expliquant le sujet, une seconde partie détaillant la version locale et une dernière partie expliquant la version collective.

II. Projet

1. Sujet

A partir d'un code existant, nous devons ajouter des Personnages et Potions ayant certaines caractéristiques et stratégies.

Nous avons à disposition un Personnage d'exemple, une arène, un serveur, une IHM, de quoi avoir un programme fonctionnel dès le départ.

L'accent était donc mit sur la réutilisation de code et l'application de nos connaissances Java acquise lors du premier semestre de notre 3ème année de Licence.

Le développement devait de préférence s'effectuer en binôme, nous pouvions aussi choisir de le faire seul ou à trois.

Nous avons deux versions à faire : une version locale et une version collective. Chaque version sera détaillée plus loin.

2. Choix

- Développement en équipe

Avec mon binôme, nous avons décidé d'utiliser le gestionnaire de version Git afin de pouvoir développer notre application ensemble de manière simple et efficace. En effet, Git permet de "partager" et de gérer les différentes versions du code.

- Stratégie du personnage de tournois

La stratégie utilisée pour le tournois est l'utilisation du charisme : 99 charisme et 1 de vie.

Notre personnage fuit le plus fort et tente d'aller en duel contre les personnages avec lesquels il gagne. Il tente des coups d'états afin de ne pas avoir de leader (Le gagnant étant celui ayant la plus grande équipe après 10min de jeu).

3. Personnages

Chaque personne a des caractéristique bien précises.

Dans la version **locale**, un personnage a (en plus d'un nom) une certaine quantité de :

- force : permet de tuer une autre personne
- charisme : permet d'ajouter un personnage à son équipe
- vie : 0 ou 1 qui dit si un personnage est en vie ou non
- leader : la référence de notre leader (-1 si on en a pas)

Dans la version **collective**, un personnage a (en plus d'un nom) une certaine quantité de :

- force
- charisme
- hp : nombre de points de vie du personnage
- vitesse : distance de vision d'un personnage (10 à 20)
- défense : permet de réduire les dégâts subits (valeur de 0 à 60)
- leader

De plus, dans la version **collective**, afin qu'un personnage puisse arriver dans l'arène, il faut que ses caractéristiques respectent l'équation suivante :

$$\text{Force} + \text{Charisme} + \text{Vie} + (10/60) * \text{Défense} \leq 100$$

4. Potions

Chaque potion permet d'augmenter (ou abaisser) les caractéristiques d'un personnage qui la ramasse.

Exemple : une potion ayant 50 de force va rajouter 50 de force à la personne la ramassant.

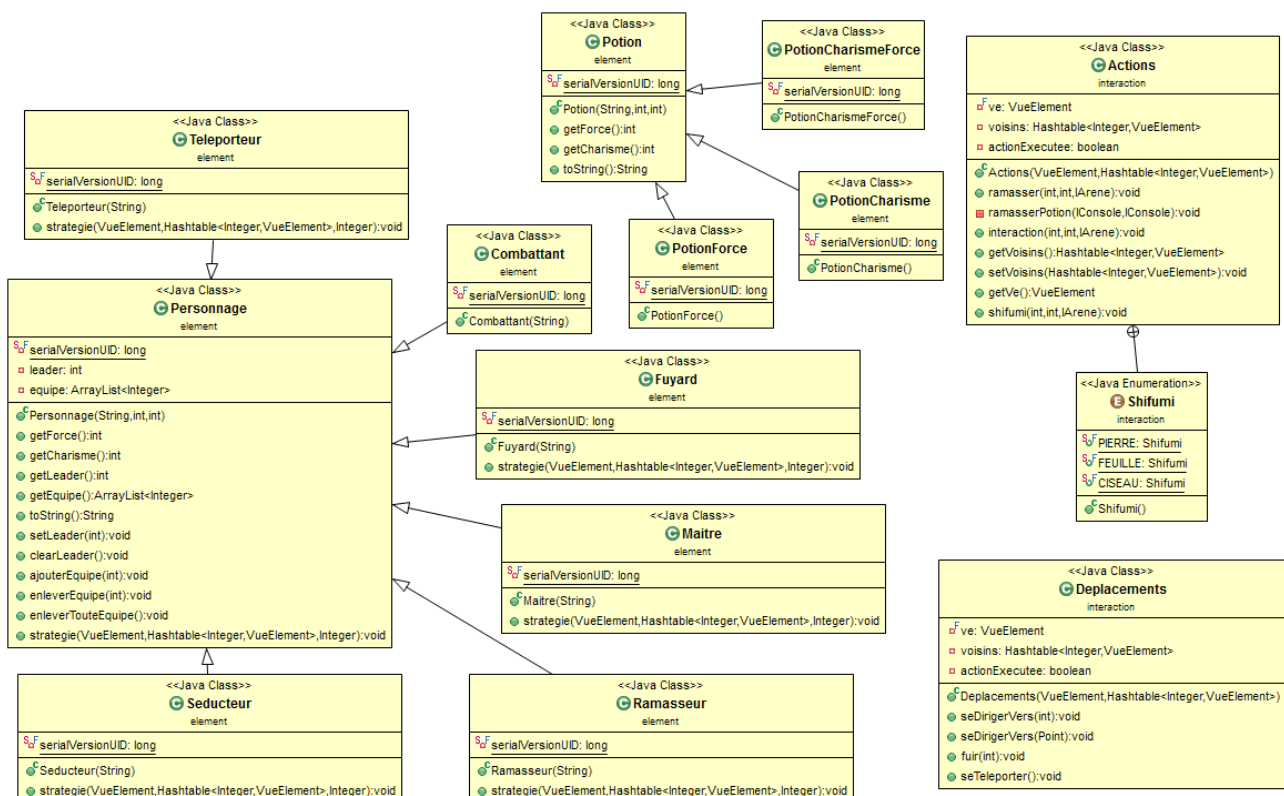
III. Version Locale

La version locale est une version dans laquelle nous avons le droit d'effectuer toutes les modifications de code souhaitées (dans toutes les classes du projet, nous n'étions pas limité).

L'objectif étant de créer au moins 5 personnages, 3 potions, d'ajouter quelques méthodes dans le package "interaction" (package gérant principalement les duels et les déplacements) et d'utiliser les méthodes afin de développer des stratégies de personnages.

1. Diagramme de classes

Nous avons utilisé le plugin *ObjectAid* afin de générer le diagramme de classes pour la version locale, la version réseau n'ajoutant aucune classe :



2. Personnages & Stratégies

- Combattant

Le combattant est un personnage qui effectue la stratégie de base (mise à disposition dans le projet fourni : il erre, ramasse les potions ou effectue des duels s'il voit quelqu'un) mais possède plus de force qu'un personnage de base.

- Fuyard

Un fuyard a des caractéristiques relativement faibles. Il tente donc de fuir (sans sortir de l'arène) dès qu'il voit un ennemi autour de lui. Si il n'y a personne autour, il ne bouge pas.

- Maître

Ce personnage a des caractéristiques très élevées et effectue la stratégie d'un personnage de base sans ramasser les potions (il n'en a pas besoin au vu de ses caractéristiques très élevées).

- Séducteur

Le séducteur, s'il voit un ennemi avec un charisme inférieur au sien, va tenter de le prendre en duel, sinon il fuit la personne. S'il n'a pas de voisins, il erre.

- Téléporteur

Il reste immobile tant qu'il n'aperçoit pas d'ennemi. Lorsqu'il voit un ennemi, il se téléporte aléatoirement sur une case vide de l'arène. Lorsqu'il voit une potion, il la ramasse.

- Ramasseur

Le ramasseur erre à la recherche de potions et les ramasse dès qu'il est à portée. Il ignore (erre) les autres personnages.

3. Potions

- Charisme

Elle donne 50 de charisme lorsqu'on la ramasse.

- Charisme et force

Elle donne 50 de charisme et 50 de force lorsqu'on la ramasse.

- Force

Elle donne 50 de force lorsqu'on la ramasse.

4. Ajout de méthodes dans le package Interaction

Ajout d'une méthode **fuir** afin de fuir une référence donnée.

Ajout d'une méthode **seTeleporter** afin de se téléporter aléatoirement sur l'arène en allant sur une case vide.

Ajout d'une méthode **shifumi** afin d'avoir une autre façon de combattre.

5. Modification du package IHM

Nous avons remplacé l'affichage de base de l'IHM.

A l'origine les éléments de l'arène (Personnages & Potions) étaient représentés par des cercles (pleins) de couleur. Afin d'avoir une représentation plus jolie et de pouvoir distinguer facilement chaque élément, nous avons appliqué une image à chacun de ces éléments.

IV. Version Collective

La version collective est une version en réseau. Les modifications que l'on peut faire au code sont limitées.

Le but est de créer un personnage et de développer sa stratégie afin qu'il gagne la partie.

C'est cette version qui est utilisée pour le tournoi. Chaque groupe a le droit de faire apparaître un et un seul personnage dans l'arène de tournoi.

- Personnages & Stratégies

- Fuyard

Le fuyard fuit continuellement vers l'extérieur de l'arène (fuit un ennemi en priorité s'il en voit un).

Ce dernier gagnait assez simplement des points en fin de partie (il sort de l'arène et continu d'avancer vers l'extérieur, donc personne ou presque n'arrive à l'attraper) car la règle de base donnait des points aux survivant (après 10 minutes de jeu).

Or, la stratégie a été modifiée en cours de développement (l'équipe la plus grande remporte les points), c'est pourquoi nous avons ensuite développé un autre personnage : le Charismatique.

- Charismatique

Le personnage Charismatique fuit les personnages contre lesquels il perd le duel.

Il essaye de convertir les personnages qu'il peut convertir (si le rapport de charisme le lui permet).

S'il se fait capturer (il a donc un leader), il essaye de faire des coups d'état (prendre en duel son leader afin de prendre sa place).

V. Conclusion

Ce projet nous a permis de mettre en pratique nos connaissances Java.

En effet, en plus d'avoir l'occasion de coder, nous avons dû lire, comprendre et utiliser un code déjà écrit par d'autres personnes.

Nous avons donc vu l'importance qu'ont les commentaires, sans lesquels il serait très long et complexe de pouvoir comprendre, maintenir et utiliser du code existant.

Ce projet était motivant car il aboutissait sur un tournoi final entre étudiants. Cela nous a donc poussé à trouver une stratégie permettant de gagner la partie, et donc à développer un code performant.

Le fait d'avoir codé en binôme nous a permis de voir les avantages et inconvénients de travailler en groupe.

Par exemple, trouver comment se répartir les tâches et comment réunir le code de chacun (solution retenue : utilisation du gestionnaire de version Git).

De plus, si l'un de nous était bloqué, le fait de pouvoir s'appuyer sur son binôme permettait de trouver une solution plus rapidement et donc de développer plus rapidement.

Cette expérience a été bénéfique pour nous deux, aussi bien sur le plan personnel que professionnel.