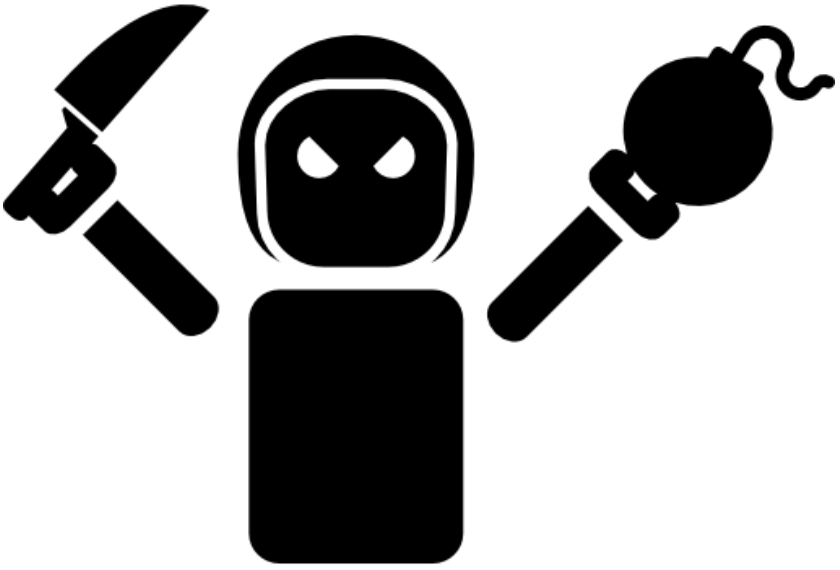
Rapport Programmation Avancée

Projet Robot Wars



Boutteville Nans - Vray Adrien
Sintes Jordan - Kuchta Daniel

https://github.com/Nans-Boutteville/PA_project

M1 MIAGE 2017-2018

Sommaire

Participation de chacun des membres

Nans Boutteville :

Nans a développé le moteur de jeu et la classe robot avec Adrien, ainsi que les annotations. Il a contribué aussi à modifier les plugins. De plus il donné les directives et attribuer les tâches aux autres membres de l'équipe.

Adrien Vray :

Adrien a développé le moteur de jeu et la classe robot avec nans. Il a été responsable du développement de l'IA et de la Vue. Adrien a aussi réalisé certains tests avec Daniel.

Jordan Sintes :

Jordan a développé le Classloader et lui ont été assigné certaines tâches comme les tests et la modification des plugins.

Daniel Kuchta :

Daniel a développé les plugins d'attaque, de déplacement et graphique. A rédigé le diaporama pour la soutenance ainsi qu'une partie du rapport.

Procédure à suivre pour tester le projet

Pour tester le projet les étapes à faire sont :

- Télécharger le projet et l'ouvrir avec un IDE supportant Maven.
- Lancer le Main du package moteur

Etant donné que le ClassLoader n'est pas fonctionnel il est nécessaire de remplacer ou ajouter le code du plugin souhaité dans la classe Robot_affichage_plugins pour un plugin graphique, dans la classe Robot_attaque_plugins pour un plugin d'attaque et dans la classe Robot_deplace_plugins pour un plugin de déplacement.

- Relancer le Main pour tester avec le ou les nouveaux plugins

- Pour lancer les tests, lancez les comme des tests classique

Calendrier des grandes étapes du projet

18 Décembre 2017 :

Début du projet - Travail sur le ClassLoader

28 Décembre 2017 :

Réunion pour réfléchir à la structure du projet

05 Janvier 2018 :

Réunion pour commencer le projet et se répartir les tâches

7 Janvier 2018 :

Classe Robot et Vue créés

8 Janvier 2018 :

Moteur, premiers plugins graphiques et annotations créés

9 Janvier 2018 :

IA simple implémenté, plugins déplacement et attaque créés

10 Janvier 2018 :

Plugins d'affichage graphique d'affichage de la vie et de l'énergie

Ajout des animations

11 Janvier 2018 :

Soutenance du projet

Tests et correctifs

12 Janvier 2018 :

Rendu du projet

Fonctionnalités

Notre logiciel est un jeu de combat de robots construit sur une architecture de plugins pouvant être chargés avec un ClassLoader. Une fenêtre JFrame affiche les robots (3 dans la configuration actuelle) qui se déplacent et qui attaquent. La partie se termine lorsqu'il ne reste plus qu'un seul robot en vie. Les robots sont gérés par une Intelligence Artificielle basique. D'après cette IA, les robots regardent les attaques possibles qu'ils peuvent réaliser selon leur énergie et leur portée, s'ils ne peuvent pas attaquer ils se déplaceront.

Les plugins que nous avons développés permettent d'afficher le robot sous forme d'un carré de couleur aléatoire ou avec une image de robot. Deux plugins distincts affichent les barres de vie et d'énergie de chaque robot. Les déplacements et les attaques consomment de l'énergie et ces dernières réduisent en plus la vie de leur cible. Pour le moment un seul type de déplacement a été implanté, il permet au robot de se mouvoir aléatoirement dans les quatres directions : haut, bas, gauche, droite ; ainsi qu'en diagonale. Quant aux attaque, nous avons soit une attaque au corps-à-corps qui inflige des dégâts à courte portée soit une attaque long distance. Ces deux attaques disposent toutes deux d'une animation propre à chacune.

Chargement dynamique

Le partie chargement dynamique faites par le ClassLoader a été développé mais une erreur nous empêche de charger dynamiquement nos plugins. Le problème était que nous arrivions à détecter le fichier jar, le ClassLoader arrivait à s'introduire dans le fichier jar, mais n'arrivait malheureusement pas à analyser le contenu de ce jar. En effet celui-ci nous confirmait qu'il n'existait aucun fichier dans le jar. Malgré plusieurs correction de la classe et d'une discussion dans le groupe, nous n'avons malheureusement pas pu résoudre le problème

Persistance

Nous n'avons pas développer la partie Persistance. Etant donné qu'il s'agit une fonctionnalité à développer en dernier et que nous avons accumulé trop de retard, nous avons passé cette partie. De plus nos parties sont très rapides, nous n'avons pas vu l'intérêt de proposer de la sauvegarde de partie.

Modularité

Concernant la modularité, nous avons conçu l'architecture en plusieurs parties avec le coeur du logiciel qui comprend le moteur de jeu qui gère toute la partie, les joueurs, les règles du jeu, etc. , le robot avec son IA et la Vue. Un package rassemble toutes les annotations. A cause de notre problème de ClassLoader nous avons mis les plugins dans un package et non dans un jar indépendant pour pouvoir lancer le programme et avoir un rendu visuel. Enfin dans un dernier package nous avons le ClassLoader.

Suivi

Concernant le suivi du projet, nous avons développé le projet sous IntelliJ pour profiter de Maven. Le repository étant sur GitHub, nous avons chacun utiliser le logiciel de versionning avec lequel nous étions le plus familier (SourceTree et Github Desktop) pour accéder au dépôt et pousser le code.
Les communications écrites entres les membres de l'équipe se sont faites principalement sur Slack. Nous avons utilisé Discord pour les communications vocales et pour le partage d'écran.

Conclusion

Pour conclure ce projet, nous pouvons donc dire que nous avons atteint certains objectifs du projet, notamment l'utilisation des plugins avec la technique de l'annotation. Le projet marche et est fonctionnel.

Cependant, des erreurs de choix d'architecture nous ont handicapés lors de la conception du projet, notamment nous avions eu un souci pour implémenter le ClassLoader dans notre projet qui ne fonctionne pas et n'arrive donc pas à charger les classes dans le jar.

Nous pouvons regretter un manque de clairvoyance dans la structuration de l'architecture et pouvons aussi regretter de ne pas avoir assez réfléchi sur la conception du projet. De plus nous pouvons aussi regretter de ne pas avoir eu le temps de développer plus de test mais aussi les sauvegardes de parties.