ROBOTWAR

Florian Muller – Antoine Huguet – Manon Ingrassia

Rappel du sujet

Le but de ce projet est de construire un jeu de combat virtuel de type RobotWar dont les fonctionnalités pourront être étendues par plugin. De base, le jeu se déroule dans une arène de combat en 2D vue de dessus où des robots s'affrontent, gérés par une IA basique. Chaque robot dispose d'une vie et d'une énergie qu'il utilise pour attaquer et qu'il récupère régulièrement. Il devra être possible d'ajouter des plugins au logiciel pour changer l'apparence, le déplacement ou l'attaque des robots.

PLUGINS DE GRAPHISME

Ces plugins servent à la représentation graphique des robots et de leurs armes. Le jeu devra contenir de base un plugin qui représente les robots par un carré de couleur aléatoire.

PLUGINS DE DEPLACEMENT

Ces plugins servent à décider du déplacement du robot. Le plugin le plus simple consiste en un déplacement aléatoire mais d'autres plus compliqués peuvent être implémentés.

PLUGINS D'ATTAQUE

Ces plugins servent à décider attaquer les autres robots et à diminuer leur barre de vie. Chaque attaque sera définie par une portée et un graphisme. De base un plugin implémentant une attaque de courte portée sera proposé.

Les 5 critères

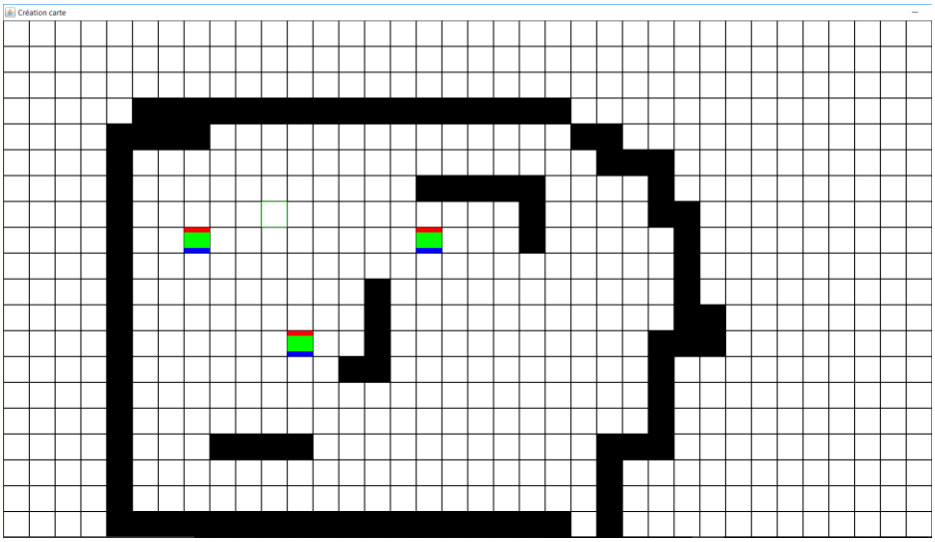
LES FONCTIONNALITES

Notre programme permet de créer un nombre de 100 robots et peuvent tous être configurés individuellement, avec en tout 9 plugins développés donc nous parlerons plus tard.

Egalement la possibilité de placer les robots ou on le souhaite au départ sur la carte ce qui permet de pouvoir mettre en place des scénarios de jeu ainsi que permettre de faire des tests dans des situations concrètes sans attendre que les robots se mettent tout seuls dans la position souhaité.

Pour accentuer cette idée de scénarios, nous avons mis en place la possibilité de créer des murs au début de partie. Cette fonctionnalité supplémentaire permet de mettre dans des situations plus complexes les robots que si la carte était vide et qu'il n'y avait rien pour les séparer. Les parties sont donc plus complexes et plus stimulantes.

Une des points de ce projets qui est a souligné est son originalité dans la création de la carte et de la multitude de situations qui peuvent être imposés de part ce choix.



Ici nous pouvons voir la phase de création de la carte avec 3 robots (ici les carrés rouges, verts et bleus) en noir. Les barres rouges et bleues sont respectivement des indicateurs de vie et d'énergie. Les mures créent ici pour rétrécir la carte avec 3 murs à l'intérieur.

LES 5 CRITERES

LA GESTION DES PLUGINS ET CHARGEMENT DYNAMIQUE

Elle a été gérer de la manière suivants : Nous avons mis tous les plugins dans un package commun. Un « ClassLoader » ainsi qu'un « PluginLoader » permettent de charger les « .class » en les affichant sur une interface graphique qui permet à l'utilisateur de les choisir..

Le chargement dynamique est partiel dans notre programme. C'est-à-dire, qu'une fois de la partie est lancé, on ne peut pas changer de graphisme du robot par exemple. Cependant, lors de la création de la carte, les robots peuvent chacun être configurés indépendamment. Tous les plugins sont donc chargés à ce moment-là et nous pouvons choisir de les utiliser ou pas.

LA PERSISTANCE

La persistance a été une des dernières choses à être développer et elle est à présent fonctionnelle. Nous utilisons une hashmap dans laquelle, pour les plugins on met le nom et le path. On recharge ensuite en utilisant le classLoader. Sinon, c'est une sérialisation classique et le projet est sauvegarder dans un fichier « robot.ser » qui est créé à côté du « .jar ».

DEPENDANCE

Le projet n'est pas dépendant de l'environnement, les plugins peuvent être déplacé dans une autre partie du projet ou même à l'extérieur. Cela est dût au fait que nous utilisons un « JFileChooser » au début qui permet à l'utilisateur de cibler le répertoire dans lequel les plugins sont chargés. On pourrait très bien imaginer qu'une personne code ses plugins de manière totalement indépendante dans un autre dossier et charge ses plugins là à la place de ceux que nous avons créés.

GESTION DU PROJET

Pour un bon suivit de l'avancé du projet, nous avons utilisés GitHub qui nous a permis de créer des tickets en fonction des besoins et d'avancer dans le projet tout en connaissant l'état d'avancement des tâches des autres membres.

PROCEDURE DE CREATION DE NOUVEAUX PLUGINS

Procédure de création de nouveaux plugins

Dans notre projet, une des forces est la faciliter pour créer de nouveaux plugins.

Pour créer un plugin, il y a cependant quelques règles à respecter. Il faut qu'ils implémentent l'interface à laquelle il correspond. C'est-à-dire, si c'est un plugin qui a pour objet la configuration de la stratégie d'attaque, il va implémenter l'interface IAttack. Si c'est un plugin de déplacement il va implémenter « IMovement » et si c'est un plugin graphique il va implémenter « IDrawing ».

Une fois que cela est fait, il faut coder les classes implémentées selon ce que vous vouler faire avec ce plugin.

Pour plus de simplicité, actuellement, les plugins sont situés à l'endroit suivant : PA2016 → plugins→ src → main → java → plugins

Il y a actuellement 9 plugins fonctionnels. 3 d'attaque, 3 de mouvement et 3 graphiques.

Pour ce qui est des plugins d'attaques nous avons les plugins suivants :

-« AttackCac » qui attaque uniquement les ennemies si ils sont proches

-« AttackDef » qui attaque de plus loin que « AttackCac »

-« AttackpeaceFul » qui n'attaque pas. C'était important dans nos scénarios de mettre un robot comme ça pour tester le comportement des autres robots.

Les plugins de mouvement :

-« MovementDefault » fonce vers l'ennemie le plus proche

-« MovementEscape » essaie de s'éloigner des ennemies

-« MovementRandom » se déplace aléatoirement

Les plugins graphique :

-« DefaultDrawing » est un carré vert

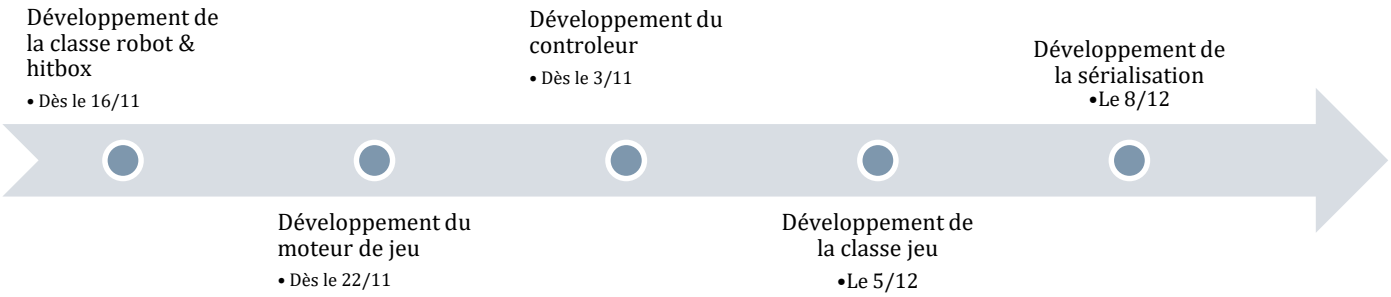
-« PacManDrawing » représente un petit « pacman »

-« TriangleDrawing » représente un triangle vert

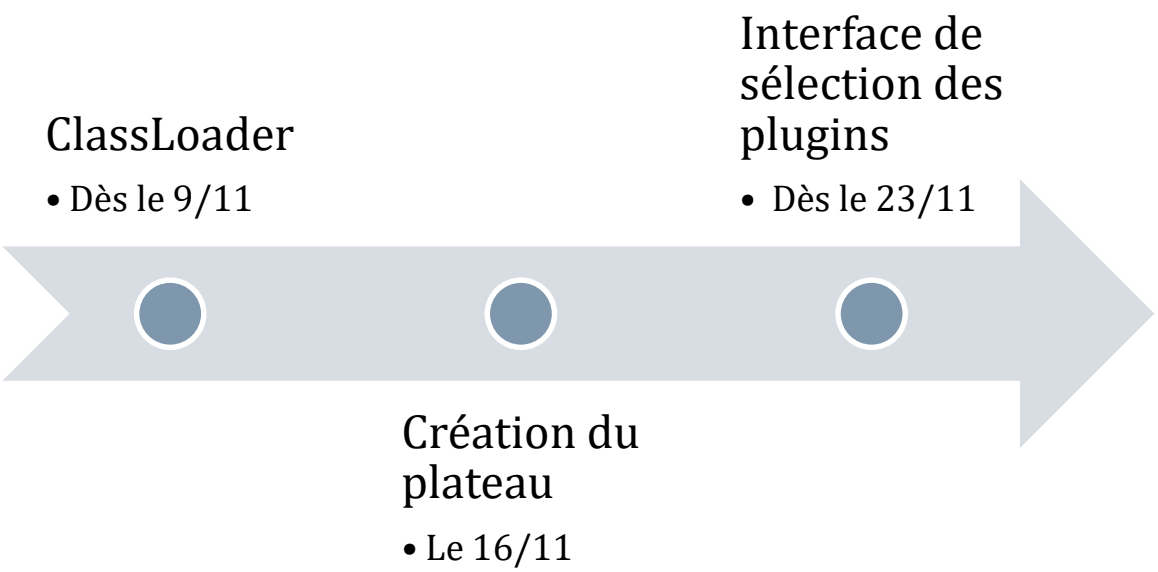
PARTICIPATION

Participation

FLORIAN



ANTOINE



PARTICIPATION

MANON