Ricochet Robots

...

SMP

Titouan André-Louis Laurène Aurélien Oumaima

Sommaire :

- Présentation de l'UML
- Présentation des classes
  - Robot
  - plateauRicochet
  - Case
  - Master
  - Objectifs
- Etat d'avancement
- Difficultés rencontrées
- Conclusion

UML

Description des Acteurs :

Joueur

- **Sélectionner un robot** : Choisir parmi les robots rouge, vert, bleu ou jaune (R/G/B/Y)
- **Annoncer nombre de coups** : Déclarer combien de déplacements seront nécessaires
- **Choisir une direction** : Indiquer où déplacer le robot (Haut/Bas/Gauche/Droite)
- **Consulter les scores** : Voir les points de chaque joueur

Master

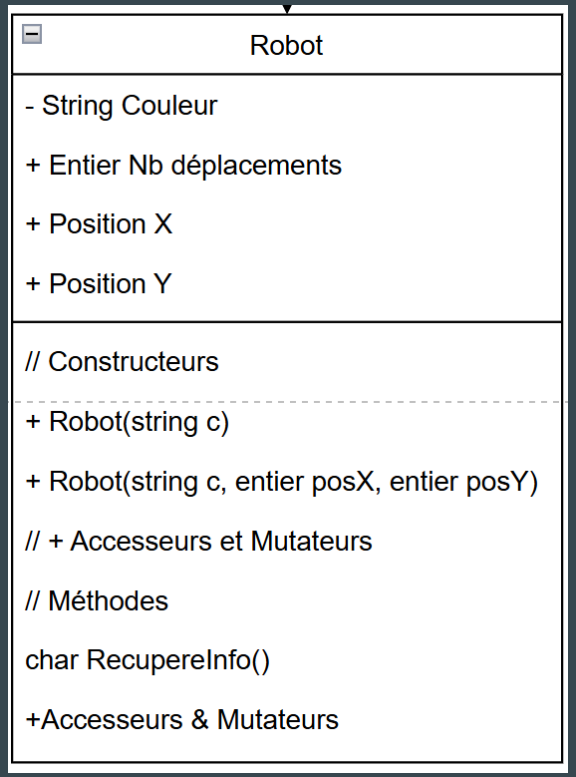
- **Configurer le jeu** : Paramétrage initial du plateau et des règles
- **Ajouter des joueurs** : Enregistrer les participants et leurs informations
- **Visualiser les statistiques** : Suivi des performances

Fonctionnalités du Système

- **Initialiser le plateau** : Création du plateau de jeu 16x16 avec les murs
- **Initialiser robots/objectifs** : Placement aléatoire des robots et configuration des objectifs
- **Gérer un tour de jeu** : Orchestration des actions des joueurs et validation des coups
- **Déplacer un robot** : Exécution du mouvement jusqu'à rencontrer un obstacle
- **Valider un objectif** : Vérification si un robot atteint sa cible
- **Attribuer points** : Mise à jour des scores après réussite
- **Gérer le temps** : Suivi du temps de réflexion avec le sablier
- **Afficher plateau de jeu** : Rendu graphique du plateau avec les robots et objectifs

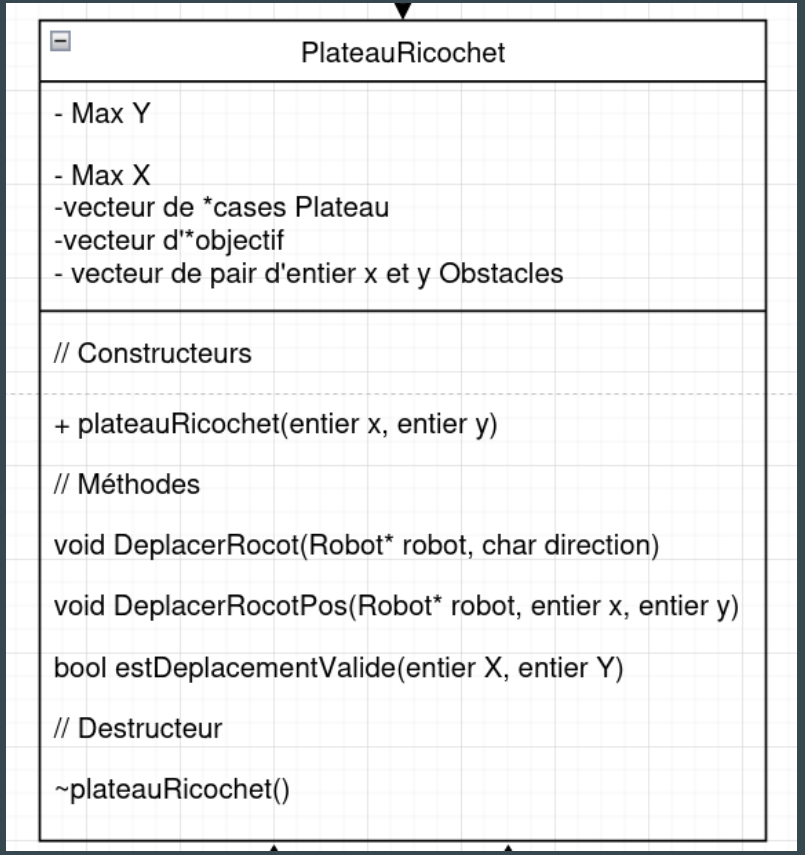
Les Robots et leurs attributs

- Récupération des commandes de clavier pour la direction
- Initialisation des attributs des Robots



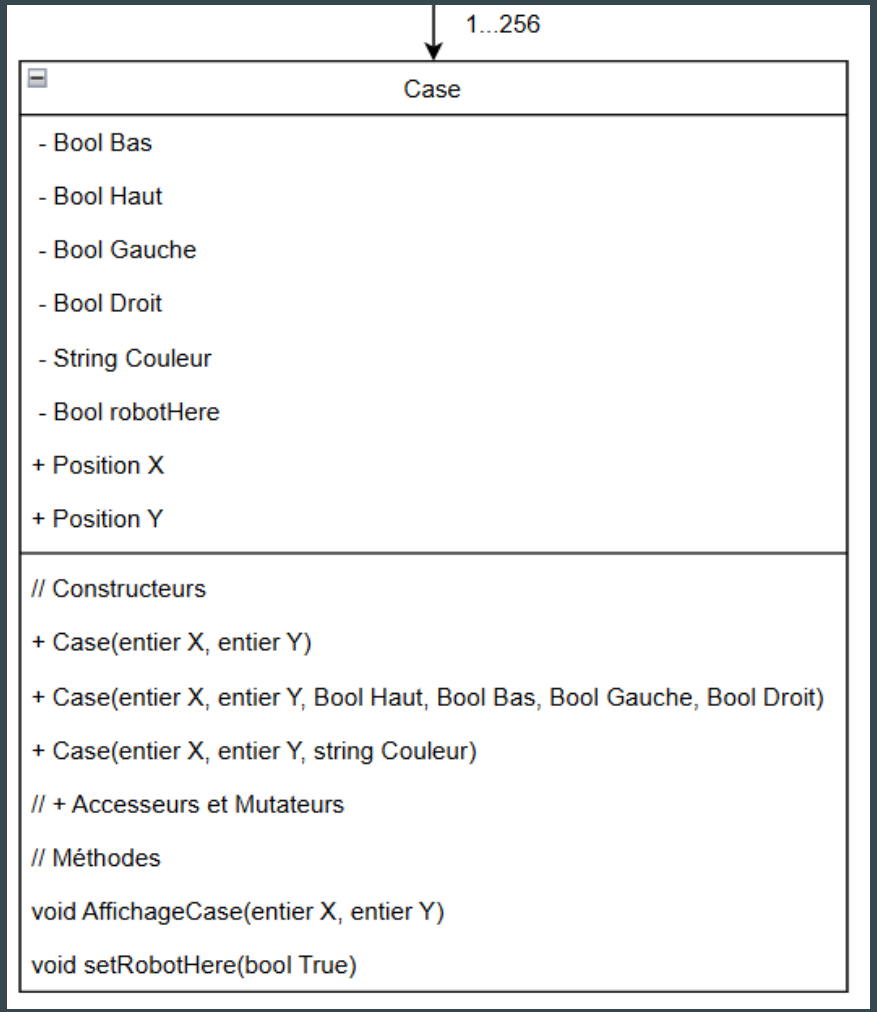
Plateau ricochet

- Génération et gestion des cases
- Génération des bords
- Déplacement des robots
  - en fonction des bords et des robots présent sur le plateau
  - renvoie la nouvelle position



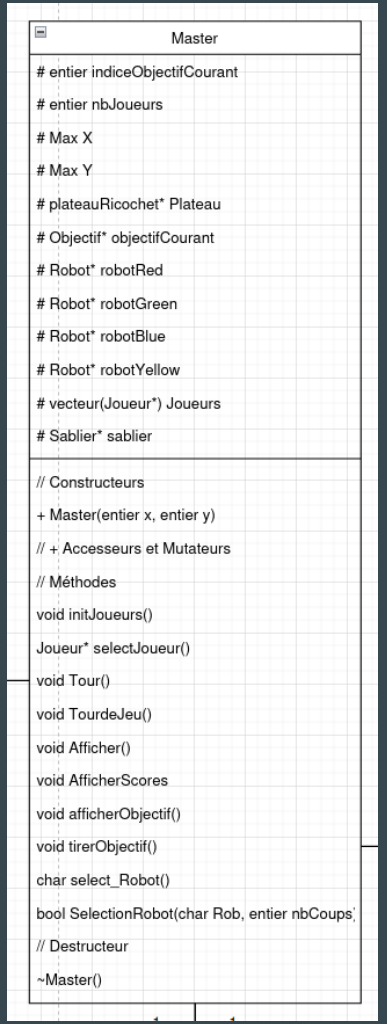
Case plateau :

- Génération des bords d'une case
- Attribution de la couleur (objectif ou non)
- Vérification de la présence d'un robot



Architecture du programme : rôle centrale de Master

- Génération et gestion des objets (robot, plateau, objectif)
- Gestion du jeux et des tours

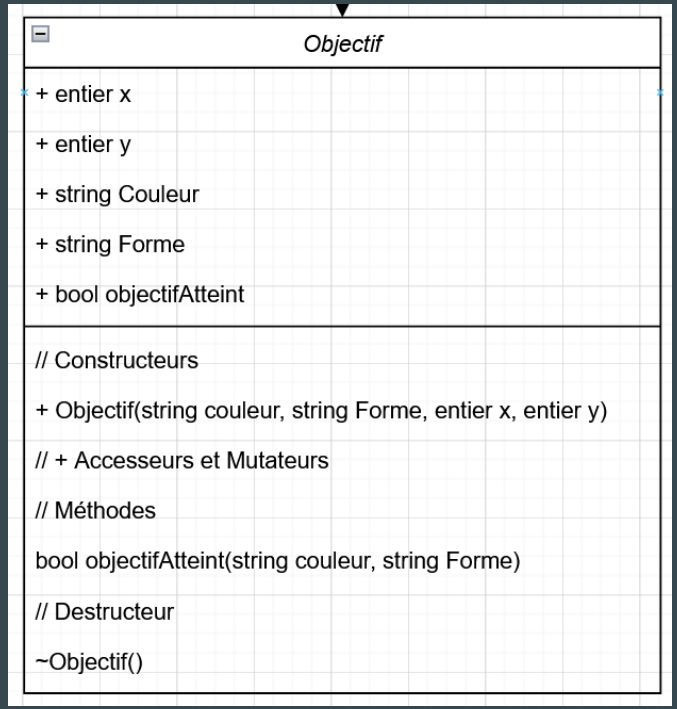


Système de jeux tour par tour:

- Réflexion
- Lorsqu'un joueur trouve une solution, le sablier se lance
- Les joueurs annoncent ensuite leurs coups
- Déroulement du jeu
- Si le joueur sélectionné atteint son objectif, alors la manche se termine. Sinon, on passe au joueur suivant. Les scores des joueurs sont mis à jour

Gestion des objectifs

- Classe Objectif
- Création d'un vecteur d'objectifs
- Tirage d'un objectif à atteindre
- Affichage
  - Objectifs sur le plateau
  - Objectif à atteindre
- Fin de jeu



Etat d'avancement du projet

- Création d'une partie
  - Plateau
  - Robot
  - Joueurs
  - Objectifs
- Jeu fonctionnel
  - Joueurs
  - Tours
  - Objectifs atteints
  - Fin de jeu

Difficultés rencontrées

- Utilisation de Git/GitHUB
- Conception du diagramme UML
- Développement à plusieurs

Conclusion

Axes d'amélioration :

- Rajouts de tests unitaires
- Implémentation du sablier
- Amélioration de l'affichage
- Amélioration système de direction
- Affichage de l'objectif central (plutôt que sous forme d'indication)
- Mise en commun UML

```
nombre de joueurs ?
2
Nom du joueur 1 ?
Al
Nom du joueur 2 ?
Laurène
Objectif : carre
Couleur : rouge
Position : (6, 5)
```