

M1if01

Poneymon

LEMEUNIER Clement 11103680, ZAFIFOMENDRAHA Gabriello 11311399| Gestion de Projet et Génie Logiciel | 2018-2019

# INTRODUCTION

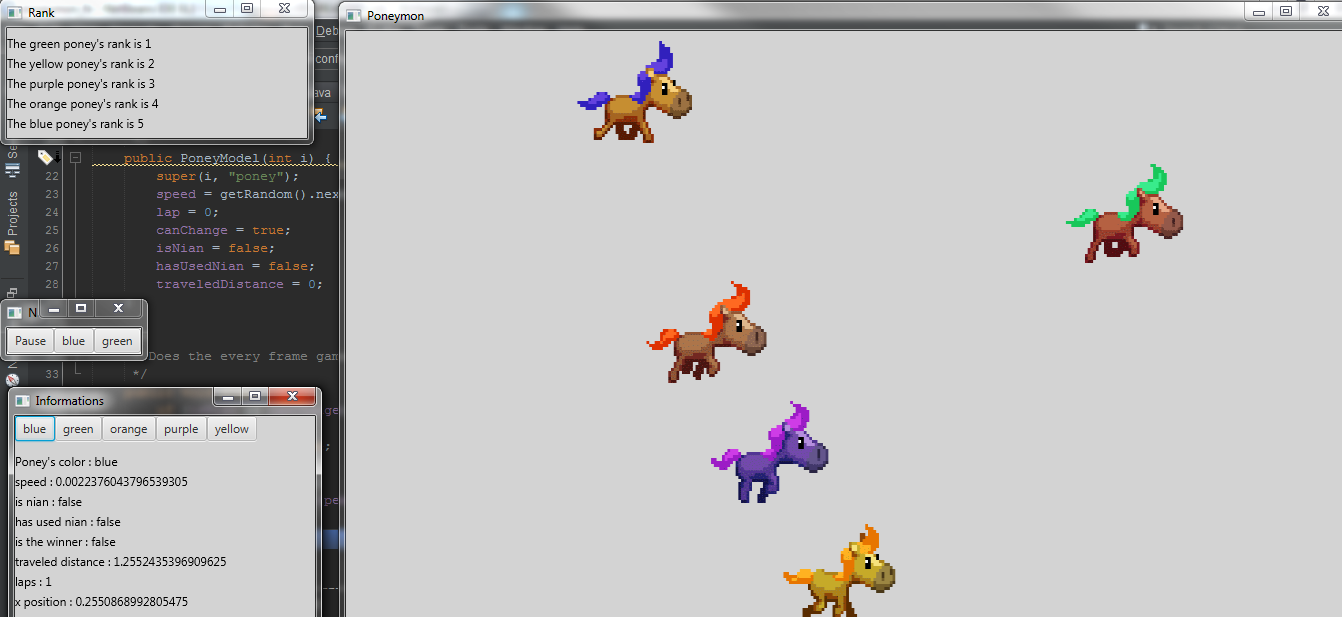
PoneyMon est un projet java faite en binôme. C’est un jeu de course de poney (5 poneys en tout) sur un certain nombre de tour (choisi par nous-même). Pendant la course, sur 5 poneys, deux sont contrôlé par l’utilisateur et les trois autres par une IA.

Les poneys peuvent accéder à un pouvoir, le mode « Nian », qui leurs permet de booster leur vitesse durant la tour où elle est activée. De plus, des pièces bonus sont placés sur le terrain, ceux-ci procurent aux poneys qui les récupèrent le mode Nian.

## Fonctionalités

Plusieurs fonctionnalités ont été implémentées dans le jeu :

* La possibilité de mettre le jeu en pause et redémarrer grâce à un bouton
* La possibilité de passer les poneys en mode Nian grâce à un bouton portant le nom de la couleur du poney
* Affichage de l’évolution du rang de chaque poney
* Affichage des informations sur chaque poney (couleur, vitesse, distance parcourue, Nian utilisé ou pas, rang, …)
* Une IA qui contrôle les poneys qui ne sont pas contrôlés par l’utilisateur





## architecture et design patterns

Le projet a été faite sur la base du modèle MVC (Modèle Vue Contrôleur).

Le modèle va contenir nos données et l’état de l’application, la vue va traiter l’interface graphique (la vue va présenter les données issues du modèle sans les modifier) et le contrôleur va faire le lien entre la vue et le modèle (il va utiliser les données du modèle, les traités en fonction de l’action de l’utilisateur, et les envoyer à la vue afin qu’elle les affiche).

Design Patterns :

Les designs patterns sont les bonnes pratiques de combinaison d’un ensemble de modules, d’objets ou de classes (réutilisabilité, maintenabilité et vocabulaire commun)

Plusieurs choix de « design pattern » ont été appliqués au projet, les voici :

* Singleton : qui permet de s’assurer d’avoir une instance unique d’une classe
* Nous l’utilisons dans notre modèle et contrôleur du projet, dans le sens ou tous les constructeurs de la classe sont privés (seules les méthodes de la classe peuvent y accéder).
* Factory : un objet fabrique des instances conformes à une interface ou une classe abstraite
* Etat : qui permet de changer le comportement apparent d’un objet en fonction de son état
* Nous l’utilisons notamment pour implémenter l’IA

