Une image contenant oiseau, oiseau aquatique, canard, Sauvagine

Description générée automatiquement

**FOX AND GEESE**

Documentation Technique

Auteurs : CUREAU Melvin

Date : 08/08/2023

Ce document comporte 5 pages :

* Page 1 : Page de garde ; Sommaire
* Page 2 : Introduction ; Architecture ;
* Page 3 : Choix techniques ; Structure de données
* Page 4 : Structure de données
* Page 5 : Références ; Conclusion

# Introduction/

***Fox and Geese*** est un jeu de stratégie pour 2 joueurs édité par ***Puzzle Games***. Le but du jeu Fox and Geese est d'être le premier joueur à atteindre la rangée opposée à celle de départ de son pion. Chaque joueur a un pion placé sur le plateau de jeu au début de la partie, et peut soit déplacer son pion d'une case à la fois, soit poser une barrière pour bloquer le passage de l'adversaire.

Le plateau de jeu est composé de case, lesquels forment un labyrinthe que les joueurs doivent naviguer pour atteindre la ligne opposée. Les joueurs peuvent placer des murs pour bloquer les chemins de leurs adversaires, mais ils ne peuvent pas bloquer complètement l'accès à la ligne d'arrivée.

Le jeu peut être joué seulement à deux joueurs. Le jeu est considéré comme un jeu de stratégie et de tactique en fonction du camp de chaque joueur, qui nécessite de planifier plusieurs coups à l'avance pour remporter la partie.

Ce document explique les détails techniques de l'implémentation et de la réalisation de notre projet en ***Python*** et utilisant la librairie graphique ***Tkinter***.

# Architecture/

Voici la structuration du dossier du projet :

* Un dossier ‘*Docs’* / contenant toutes les documentations nécessaires
* Un dossier ‘*Src’* / contenant tous nos fichiers en python.
* Le fichier ‘LICENSE’ / contient la licence MIT qui régule l’accès à notre projet
* Le fichier ‘README’ / explique brièvement notre projet
* Le dossier ‘song’ / contenant les sons joués lors de l’exécution du jeu

Structuration du dossier source de notre projet :

• Un fichier ‘grid.py’/ responsable de l’initialisation de la grille de jeu, de l’affichage du du plateau de jeu et de la boucle du jeu.

• Un fichier ‘main.py’ / responsable de l’initialisation du jeu, et de la gestions des menus au démarrage de celui-ci.

• Un fichier ‘rules.py’ / contenant toutes les règles du jeu pour l'utilisateur.

• Le dossier ‘song’ / contenant les sons joués lors de l’exécution du jeu

• Le dossier ‘img’ / contenant les images utilisées lors de l’exécution du jeu

# Choix techniques/

* Langage de programmation : Python avec la librairie Tkinter
* Editeur de code : PyCharm
* Système de contrôle de version : Git et GitHub
* Système d’exploitation : Windows

Le code est structuré en plusieurs fichiers pour faciliter la lisibilité et la maintenance. Les fonctions sont découpées de manière modulaire et indépendante pour faciliter la réutilisation du code. L’interface utilisateur est gérée via la librairie graphique Tkinter.

Pour la réalisation du projet, le langage Python avec la librairie Tkinter a été utilisé. L’utilisation du langage de programmation Python et de sa librairie Tkinter se justifie par des bases acquises lors de cours de python au préalable et notamment avec la librairie Tkinter.

Afin de réaliser le projet de façon la plus collaborative possible nous avons utilisés deux outils principaux. D’un autre côté, *GitHub* est une plateforme de gestion de code source qui facilite la collaboration entre les développeurs. Les membres peuvent partager leur code, effectuer des modifications, créer des branches pour expérimenter et fusionner les modifications pour les incorporer dans le code principal. Cela facilite grandement la gestion de version, la collaboration et le suivi des modifications apportées au code. Ces deux outils nous ont permis de travailler ensemble de manière plus efficace.

Un diagramme des différentes classes a aussi été réalisé, qui compose le code pour une meilleur compréhension de celui-ci.

# Structure de données/

Le jeu utilise plusieurs structures de données pour stocker les informations nécessaires.

**Algorithmes de gestion de la grille :**

La grille est stockée dans une matrice à deux dimensions. Chaque case contient un entier représentant son état actuel :

* 0 -> représente une case vide
* 1 -> représente l’affichage d’une case disponible (pions noir)
* 2 -> représente l’affichage des pions rouge (renards)
* 3 -> représente affichage des pions vert (oies)
* 4 -> représente l’affichage des futurs déplacements sans capture possible
* 5 -> représente l’affichage des futurs déplacements avec une capture possible

Par le biais de la fonction ***displayWindows***(), chaque entier est représenté graphiquement à une couleur et une forme distincte grâce à la librairie graphique Tkinter.

**Algorithmes implémentant les déplacements et les captures :**

Les déplacements sont effectués par l’intermédiaire de la fonction ***onClick***().

Cette fonction récupère les coordonnées du clic pour afficher ensuite les déplacements possibles pour le pion sélectionné.

Une image contenant texte, Police, capture d’écran, nombre

Description générée automatiquement

Pour effectuer un déplacement, l’algorithme mis en place fait en sorte de remplacer la case choisie parmi celles proposée en modifiant directement dans la matrice l’entier correspondant, comme vu précédemment. Bien sûr une vérification stricte à été mise en place afin de ne pas faire déplacer un pion dans une case vide ou sur un autre pion déjà existant.

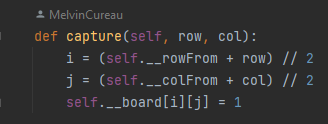
Une seconde fonction **clearBoard()**, permet de nettoyer la grille de tous les pions qui indique au joueurs les éventuels déplacements possibles (voir ci-dessous).

Une image contenant texte, capture d’écran, Police

Description générée automatiquement

Pour cela elle parcourt la grille, et remplace tous les entiers égaux à 4 ; 5 ou 6 (représentant les déplacements et captures possibles) par 1 (représentant une case disponible).

Enfin une fonction de capture a été mis en place, celle-ci récupère les coordonnées du déplacement avant (rowFrom et colFrom) et après (row et col) pour remplacer l’entier qui se trouve entre les deux positions dans la matrice.



# Références/

Pour la réalisation de ce projet l’IDE PyCharm a été utilisé. Le code de ce projet sera publié sur GitHub, contenant nos documentations utilisateurs et techniques ainsi que nos fichiers de licence et le readme.

# Conclusion/

Pour résumer, la réalisation de ce projet a été une expérience enrichissante et stimulante. Ce m'a permis de mettre en pratique les compétences en programmation que j'ai acquises, tout en découvrant de nouveaux outils de gestion de fichiers en Python.

En ce qui concerne l'architecture du projet, j'ai opté pour une structure de code modulaire qui favorise une meilleure organisation et facilite la maintenance, contribuant ainsi à rendre le code plus lisible et plus simple.

En ce qui concerne les perspectives d'amélioration, il serait intéressant d'explorer la possibilité d'introduire une fonctionnalité de mise en pause et de reprise des parties, en sauvegardant l'état du plateau de jeu dans un fichier au format .txt, permettant ainsi aux joueurs de reprendre là où ils s'étaient arrêtés.

En fin de compte, je suis satisfait du travail accompli, et je suis convaincu que le jeu *Fox and Geese* offre une expérience de jeu à la fois divertissante et stimulante pour les joueurs de jeux de société. J'espère que cette documentation technique vous sera utile pour comprendre le fonctionnement interne du jeu et pour d'éventuelles améliorations futures.

Je reste disponible pour toute question ou commentaire à propos de ce projet.