Projet Tutoré 2021

13 JANVIER

IUT Sénart-Fontainebleau
Lucas Grandjean, Adil Hammerschmidt

Projet Tutoré : Blocus

Table des matières

Présentation du rapport

Dans le cadre de notre cursus scolaire, pour notre première année de DUT Informatique à l'IUT de Sénart-Fontainebleau, il nous a été demandé de réaliser un jeu de Blocus en C89.

Vous pourrez alors, à travers ce rapport, trouver tous les détails de la création du jeu, les fonctionnalités, des explications et bien d'autres encore.
Pour une navigation plus fluide, je vous invite à vous référer à la table des matières ci-dessus.

Présentation du jeu

Qu'est-ce que le Blocus ?
Pour y répondre, laissez-moi vous citer les règles du jeu d'après les consignes de notre projet tutoré :

Le terrain de jeu prend la forme d'une grille carrée dont la taille sera choisie à chaque partie entre 3 et 9. L'un après l'autre les deux joueurs choisissent une case où placer leur pion (les deux pions ne peuvent pas partager la même case).
Une fois le terrain mis en place, les joueurs alternent les tours. Durant le tour d'un joueur, il doit déplacer son pion vers une case adjacente (y compris en diagonale), puis choisir une case libre qui sera condamnée. Le premier joueur qui ne peut pas déplacer son pion (car toutes les cases adjacentes sont condamnées ou occupées) a perdu.

Le joueur va avoir la possibilité de jouer contre un ami ou contre une intelligence artificielle (I.A). Une partie est de courte durée mais elle demande de faire preuve de réflexions et de bien comprendre l'ensemble des stratégies qu'un jeu si libre possède.

Le fonctionnement du jeu

Le jeu a été codé en C89 (C ANSI).
Grâce à un fichier « Makefile », l'utilisateur peut directement accéder au jeu en tapant la commande « make run » dans un terminal.

Dès lors, le menu se lance. Il vous sera proposé 3 options menant chacune à leur sous-menu.

- **Joueur contre Joueur** permet de lancer une partie contre un autre utilisateur, à condition qu'il joue sur le même ordinateur.
- **Joueur contre IA** permet de lancer une partie contre une intelligence artificielle.
- **Options** permet de changer des aspects du jeu (tel que la possibilité d'activer un thème sombre)

Si vous avez sélectionnés l'une des deux premières options, il vous sera ensuite demandé d'entrer la taille du terrain : celui-ci est divisés, comme stipule les règles, en plusieurs carrés formant une grille. Vous pourrez donc vous affronter une grille de format 3x3 jusqu'à une grille de format 9x9.

Une fois la taille sélectionnée, *board.c* va se charger de faire dérouler la partie de jeu. Elle créera la fenêtre, selon la taille de la grille, et attribuera des valeurs spécifiques à chaque case de la grille dans une structure.

Grâce à ça, chaque case possédera leurs coordonnées sur la grille ainsi que leur « état » (state).
Le joueur va alors pouvoir se déplacer sur la case de son choix, tant qu'elle est libre (définie par un state d'une valeur de 0).
Une fois le joueur s'étant déplacé (et ,s'il en a l'occasion, ayant bloqué une case), le tour passe, et c'est donc à son adversaire de jouer.
La partie va se dérouler jusqu'à ce qu'une personne ne puisse plus se déplacer le moment où son tour vient.

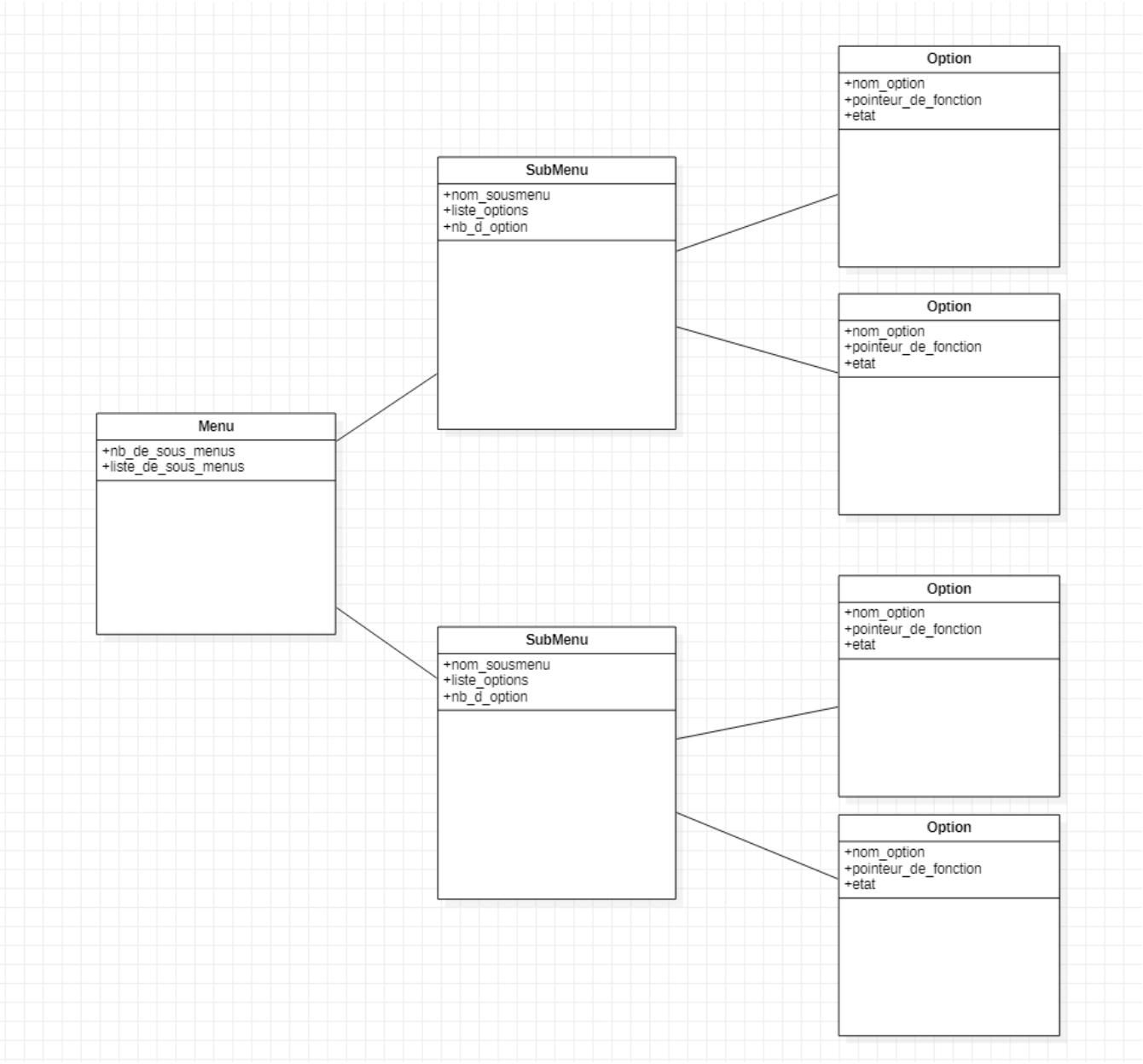
Fonctionnement du menu

Le menu est décomposé en plusieurs parties :

- **Menu :** Une structure contenant une liste de sous menus. Elle sert à différencier le menu principal et l'écran de fin dans le code et à accéder aux sous menus et n'est donc jamais affichée à l'écran.
- **Sous menu :** Une structure contenant une liste d'options au choix. Dans le cas du menu principal, « Joueur contre Joueur », « Joueur contre IA » et « Options » sont les sous menus.
- **Option :** Une structure associée à une fonction du programme au choix qui va nous permettre de déclencher des actions depuis le menu (tel que l'activation du mode sombre ou le lancement d'une partie). Elle est définie par un état qui permet au programme de savoir si elle activable (façon ON/OFF) ou non.

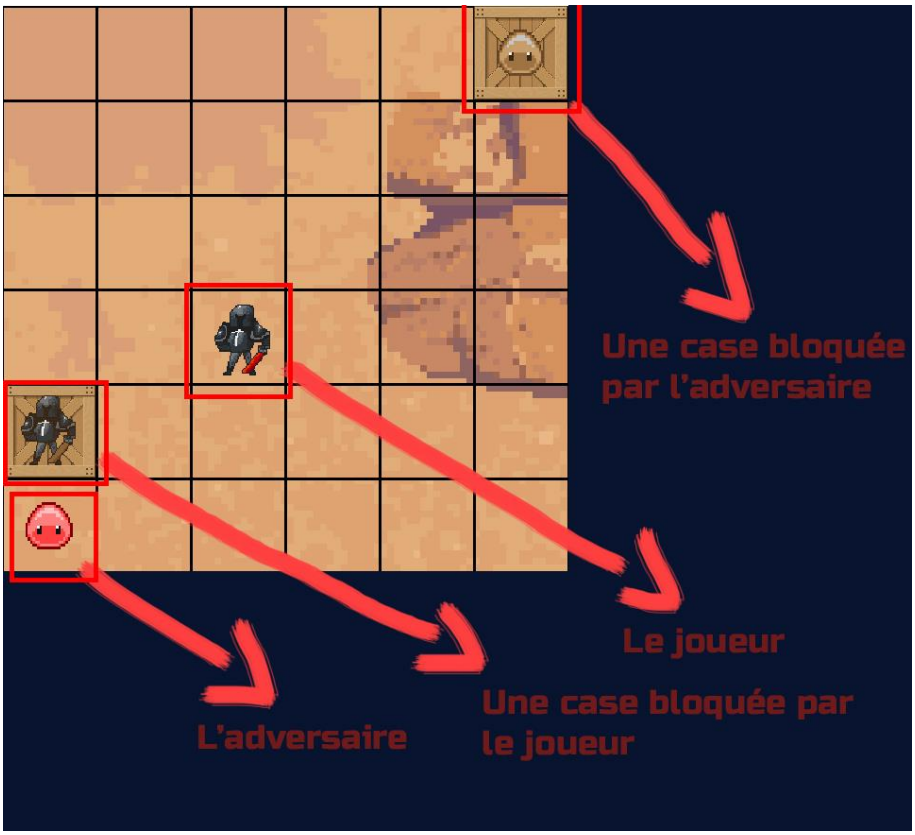
Schéma du menu (simplifié)

Ce schéma uniquement juste de comprendre le fonctionnement du programme : il n'est pas complet et ne respecte pas certaines normes : son utilité seule est de fournir un plan visuel pour avoir une idée globale de comment le programme agit.



Fonctionnement d'une partie

Une partie est caractérisée par son nombre de tours et la déclaration d'un gagnant.
Chaque fois qu'un joueur change de case et qu'il en bloque une autre, son tour passe.
Les tours impairs sont réservés au joueur 1, et les tours pairs sont réservés au joueur 2.
A chaque fois qu'un tour se termine, le programme analyse si un des joueurs est bloqué : est-ce qu'il existe une case libre (de state 0) dans les cases aux alentours des joueurs ?
Si le programme remarque qu'un des joueurs est bloqué, c'est la fin de partie, et son adversaire est déclaré gagnant.

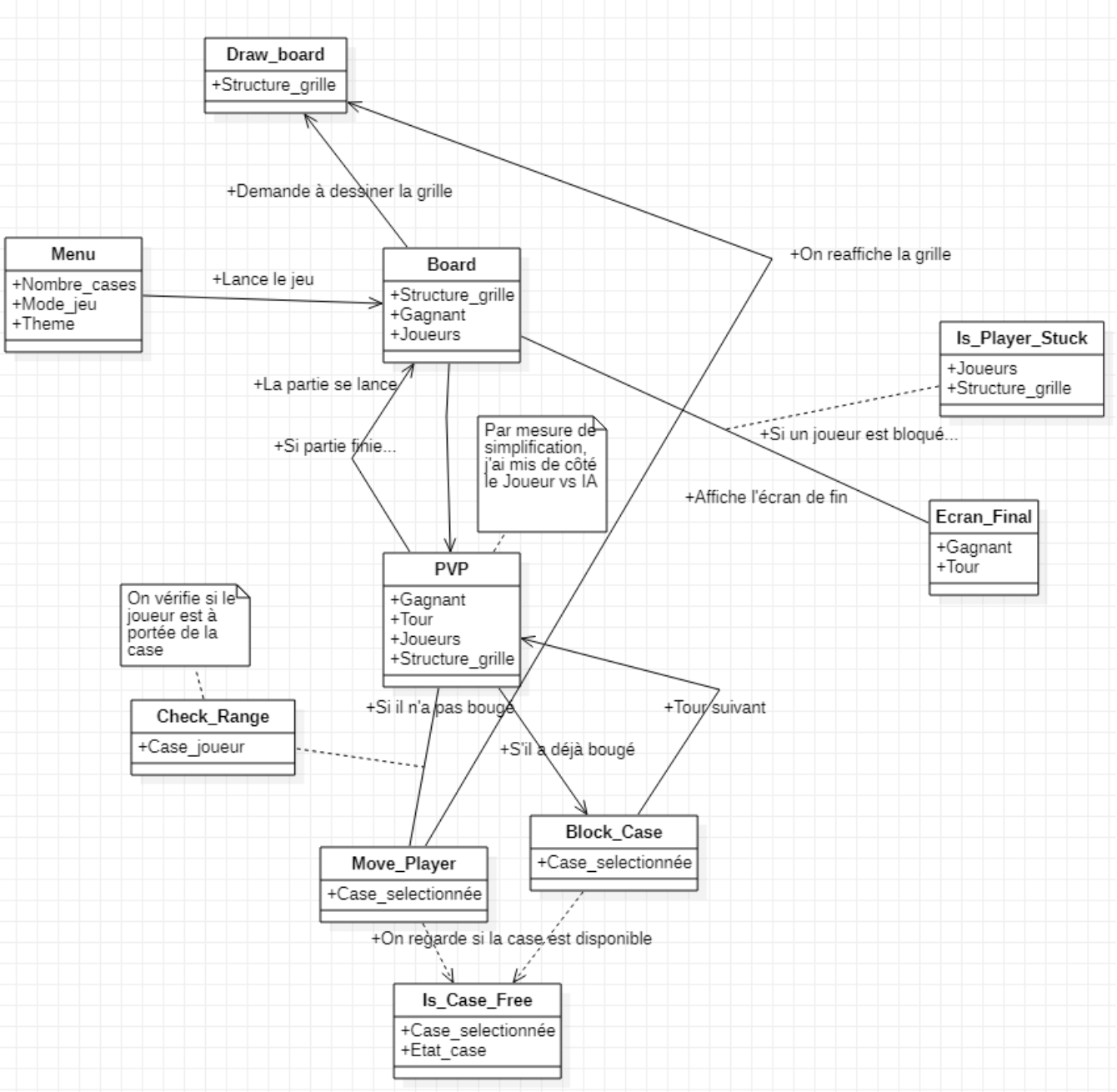


Fonctionnement du bot

Le bot (ou intelligence artificielle) est un peu plus compliqué à faire fonctionner qu'un joueur, puisque son train de pensée doit être alimenté par un programme.
Pour ce programme, le bot ne sera pas forcément plus malin que vous : en effet, il se contentera de chercher dans la grille si une des cases à ses côtés est libre. Dès qu'il en détecte une, il s'y déplacera et bloquera une case au hasard, pour autant qu'elle soit libre.

Schéma du programme (simplifié)

Ce schéma permet uniquement de comprendre le fonctionnement du programme : il n'est pas complet et ne respecte pas certaines normes : son utilité seule est de fournir un plan visuel pour avoir une idée globale de comment le programme agit.



Conclusion

Point de vue de Lucas

Pour ce qui est de mon avis, je pense que ce projet tutoré était un bon exercice pour nous faire appliquer les bases de ce que l'on a appris.
Néanmoins, si je peux critiquer un point de cet exercice, c'est qu'il ne reflète pas suffisamment le travail d'un développeur.
L'exercice était de nous mettre dans le boulot d'un développeur indépendant qui mettait en place son propre jeu : or, une version « simplifiée » et « rapide » d'un jeu ne montre pas assez la créativité et le travail nécessaire, contrairement à un jeu plus « complet ». C'est pourquoi il est possible pour certaines personnes de créer des programmes mal organisés, peu malléables, peu optimisés et non « future-proofing » : des projets de faible taille ne demandent pas une organisation exemplaire, ce qui pour moi devrait être un des points essentiels de cet exercice. Forcer les développeurs à faire des projets plus conséquents en échange d'un délai plus large serait de mon point de vue plus bénéfique.

Point de vue de Adil

Je pense que ce projet m'a été bénéfique et m'a permis d'être plus à l'aise avec les notions vues en cours ainsi qu'avec git qui a dévoilé sa vraie force et sa présence indispensable dans les travaux de groupe. J'ai également appris à mieux organiser mon code et à respecter les normes imposées en C89. Malgré le fait que je trouve la bibliothèque graphique peu pratique, je pense que ce projet reste un bon exercice d'application permettant d'assimiler tout ce qui a été appris depuis le début de l'année.