Rapport de Projet
Shing Shang

Lucas Duval – Téo Meurdrac

TD3.2 – DUT INFORMATIQUE 1A

UNIVERSITE DE CAEN – IUT IFS

Introduction :

Le projet Shing Shang possède un objectif simple : reproduire au mieux un célèbre jeu asiatique du même nom. Le principe du jeu étant un savant mélange entre notre jeu de dame européen et une partie d'échec plus ou moins revisitée. L'enjeu majeur du jeu étant d'apporter l'un de ses dragons (un certain type de pion) sur le "portail" ennemi, le tout se jouant sur un plateau de jeu de type damier.

Les fonctionnalités du programme :

Le programme développé inclus les fonctionnalités suivante :

- Créer une partie
- Sauvegarder une partie (1 Slot)
- Ecraser une sauvegarde existante
- Charger une partie existante
- Sélection des bushis (nom des pions du Shing Shang) intuitive
- Gestion des erreurs de sauvegarde/d'accès

Logique et Organisation du programme :

1. Le plateau

Nous avons fait le choix de voir le plateau de jeu comme un tableau de pointeurs pointant vers un type de case. En effet, lors de l'initialisation du programme, nous créons 9 classe de case, chacun d'entre eux ayant 2 caractéristiques : une équipe et un type.

Dans le tableau de pointeurs "plateau", on trouvera 4 "grande classe de case"

1) Les pointeurs de valeur NULL indiquant que la case est vide et utilisable

2) Les pointeurs pointant vers les classes de cases ayant l'attribut équipe de valeur 1, correspondant aux cases occupées par des pions de l'équipe numéro 1

3) Les pointeurs pointant vers les classes de cases ayant l'attribut équipe de valeur 2, correspondant aux cases occupées par des pions de l'équipe numéro 2

4) Les pointeurs pointant vers les classes de cases ayant l'attribut équipe de valeur 0 qui correspondent à des cases innaccessibles

Note : On considère les portails comme des pions appartenant à leur équipe respective

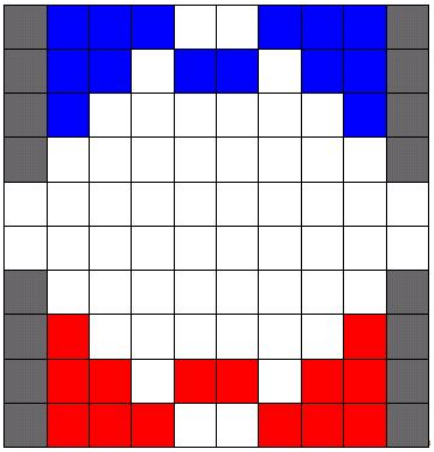
Exemple de Tableau Plateau

En gris les cases de valeurs équipe = 0

En bleu les cases de valeurs équipe = 1

En rouge les cases de valeurs équipe = 2

En blanc les pointeurs de valeur NULL



2. La sélection interactive

Nous qualifions la navigation et la manipulation de toutes les fonctionnalités du programme d'interactive. Par ce terme, nous souhaitons exprimer le fait qu'il n'est pas nécessaire d'appuyer sur entrée pour envoyer un message à la console. En effet notre souhait de départ était d'éviter que l'utilisateur est l'impression de se servir d'une invite de commande pour jouer à un jeu mais bien de jouer à un jeu, ayant comme support graphique l'invite de commande.

Pour faire en sorte que cette navigation et cette selection soit "interactive" nous modifions les paramètres du terminal au lancement du programme pour que celui ci s'execute en mode canonique. Cela signifie qu'il execute chaque caractères entré par l'utilisateur indépendament et immédiatement après la frappe de l'utilisateur.
Nous récupérons ce caractères à l'aide de la fonction getchar() dans notre programme, fonction getchar qui est appelée en permanence par notre boucle générale while, permettant de scanner en permanence les entrées de l'utilisateur.

Pour gérer la sélection sur le plateau nous utilisons deux variables de type structure coord représentant des coordonnées sur le plateau.

La variable focused qui indique la case mise en surbrillance par l'utilisateur (en jaune)

La variable selected qui indique la case selectionnée par l'utilisateur (si il en existe une, en rouge)

Afin de gérer la modification de la variable focused et donc le déplacement du curseur jaune, nous utilisons les flèches directionnelles du clavier.
En effet après quelques recherches , nous avons remarqué que, lorsque qu'un utilisateur presse une flèche, 3 caractères sont simultanément envoyés au terminal.
Ces caractères sont les suivants : "\033" "[" "A" ou "B" ou "C" ou "D"
Nous utilisons ceux ci de la facon suivante :

"\033"	"["	"A" ou "B" ou "C" ou "D"
Nous permets de savoir que la touche pressée est soit une flèche soit la touche echap	Nous passons ce caractères car il ne nous intéresse pas	Nous regardons si un troisième caractère a été envoyé. Si non, c'est que la touche echap a été pressée, si oui, c'est qu'une des flèches a été pressée

Chaque direction possédant une lettre unique (de A à D), nous modifions les coordonnées de la variable focused pour que le curseur aille dans la direction choisie.

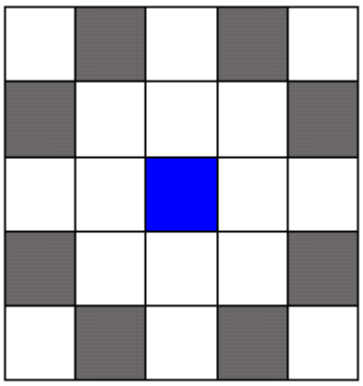
3. L'affichage (et la gestion de la victoire)

Le plateau est affiché à chaque entrée utilisateur. Une fonction parcours le tableau et affiche les caractères correspondant. Nous avons utilisé des caractères spéciaux utf8 pour représenter les bushis sur le terminal. Les coordonées focused et selected sont colorées (si elle existe). La spécificité de cette fonction est qu'elle gère également deux des trois conditions de victoires (on considère que si un joueur n'a plus de pion "simple" ou n'a plus de dragon, alors il a perdu). En effet, étant la seule fonction qui scanne totalement le plateau et plutot que de rajouter une boucle supplémentaire scannant tout le plateau, en affichant le plateau, celle ci scanne le plateau et compte les bushis et les dragons restant pour chaque équipe, retournant en cas de victoire une valeur spécifique en fonction de l'équipe.

4. Les déplacements

Lorsqu'une case est selectionnée est que l'utilisateur appuie a nouveau sur la touche entrée, la fonction requestmove() est appelée. Elle a pour rôle de choisir la fonction correspondante au type de pion effectuant le déplacement. La fonction appelée vérifie si le déplacement est un déplacement de 1 ou de 2, si il est de 2, si il est réalisable ou si c'est un saut. Pour mieux gérer les déplacements, on calcule le déplacement relatif que souhaite effectuer l'utilisateur (Exemple si l'utilisateur veut passer de la case (1,1) à la case (2,2), il effectue un déplacement de (+1,+1). On visualise en quelque sorte le déplacement comme un vecteur.

De plus, lors d'un déplacement de deux cases (saut ou non), il est interdit de se retrouver sur les cases grises suivante :



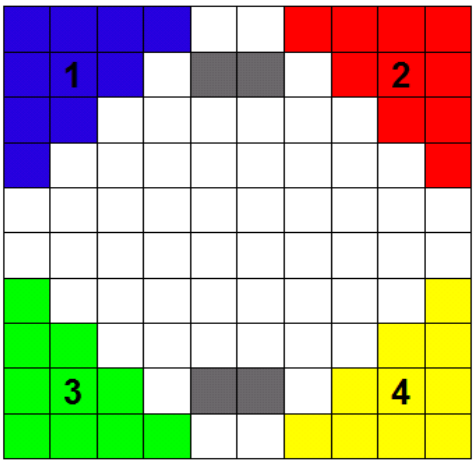
Le fait d'utiliser des vecteurs pour caractériser un déplacement permets de nous dire si (la valeur absolue du déplacement en x = 1 et la valeur absolue du déplacement en y = 2) ou (la valeur absolue du déplacement en x = 2 et la valeur absolue du déplacement en y = 1) alors le déplacement est interdit.

Aussi, le fait d'utiliser un vecteur permets, dans le cadre d'un saut et en le divisant par 2 d'obtenir la case sur laquelle se trouve le pion "sauté".

5. La génération du plateau

Pour générer le plateau nous utilison quatre boucle permettant de générer les 4 triangles dont est composé le plateau. Cette manière de faire particulière relève plus d'une satisfaction personnelle plutôt que d'une réelle optimisation du code.

On génére ainsi à l'aide de 4 boucles for 4 triangle en décomposant chacun de ces triangles en diagonales. Pour le premier triangle par exemple, les premières cases des diagonales sont forcéments des case inaccesibles. Puis en fonction de la diagonale on vient y poser tel ou tel type case.



6. Sauvegarde

Le principe de la sauvegarde est simple : l'application crée un dossier dans le répertoire courant et y insère un fichier de sauvegarde.

Ce fichier de sauvegarde contient 2 informations importante :
- Le tour du joueur courant
- L'état actuel du plateau

Le tour du joueur courant est stocké sous forme d'un chiffre au début du fichier. Ce chiffre est suivi par une chaine de caractères chacun d'entre eux représentant une case du plateau.

Répartition globale des taches :

Plateau : Lucas et Téo
Sélection interactive : Lucas et Téo
Affichage : Téo
Déplacement : Téo
Sauvegarde : Lucas
Génération : Lucas

Bilan qualitatif du binôme :

Nous sommes globalement très satisfait du rendu final de notre programme. Nos principales difficultés se sont trouvées sur la sélection interactive. C'est la chose qui nous a posé le plus de problèmes avec les déplacements où l'on a du résoudre certaines contraintes par des astuces mathématiques rendant justement c'est problématique d'autant plus intéressantes.

Mode d'emploi

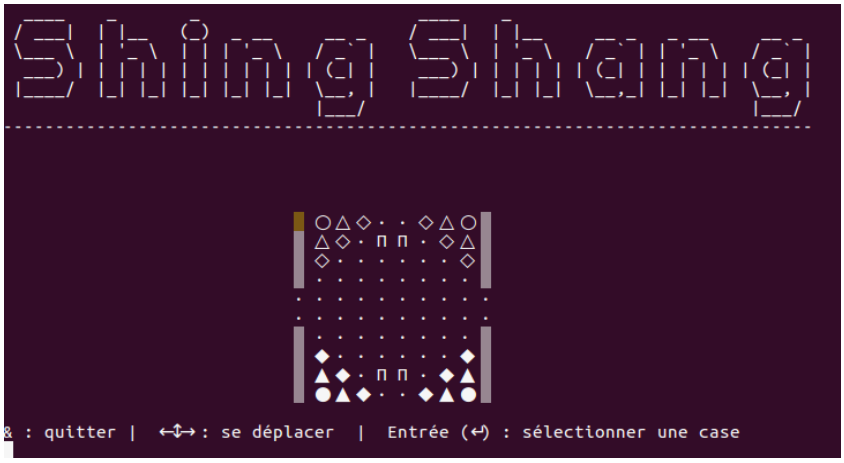
Pour compiler le programme (sous linux uniquement), ouvrez un terminal dans le dossier dans lequel se trouve les fichiers en ".c" et entrez la commande suivante :

```
gcc main.c display.c fileengine.c gameengine.c starter.c -o main
```

Une fois le programme compilé, il suffit de le lancer avec la commande `./main`

Lors du premier démarrage, on ne peut que lancer une partie "normale", aucune partie sauvegardée n'étant disponible.

En appuyant sur la touche entrée, le jeu démarre, vous pouvez alors jouer en utilisant les flèches directionnelles ainsi que la touche entrée. Appuyez une première fois sur entrée pour sélectionner un pion, puis une deuxième fois à l'endroit ou vous voulez le déplacer. Le jeu calculera si cette action est possible. Si oui, votre pion sera automatiquement déplacé. En fonction de votre coup, le jeu calculera si vous pourrez rejouer ou si cela esr au joueur suivant de jouer



Si vous souhaitez sauvegarder ou quitter votre partie, appuyez sur la touche &, un menu s'affichera et vous pourrez si vous le souhaitez sauvegarder votre partie, ou simplement l'abandonner et quitter.

Une fois la partie quittée et si vous avez sauvegardé votre partie, vous pourrez la retrouver en lancant le programme et en appuyant sur W pour charger la dernière partie sauvegardée.
Note : attention, on ne peut sauvegarder qu'une seule partie à la fois

Bilan personnel

Nous avons beaucoup apprécié travailler ensemble sur ce projet et sommes très satisfait du résultat. Ce travail nous a permis de mettre en commun et d'apprendre sur les connaissances, l'expérience et les caractères de chacun. Notre envie de toujours vouloir faire en sorte de faire comme on l'imagine nous a permis de nous pousser un peu plus loin notamment grâce à la sélection interactive, chose qui nous a fait entrer plus ou moins dans les "entrailles" du C. Ce projet nous a fait progressé, c'est sur tant dans une approche informatique que dans une approche humaine et plus encore. Je pense notamment aux mathématiques qui nous ont pas mal aidé sur les déplacements et également, et on aurait tendance à l'oublier, dans la gestion de projet.