Rapport jeu de dame, Android, IOS

Billy Ronico, Said Ismael, L3 informatique

1er mai 2021

1 Introduction

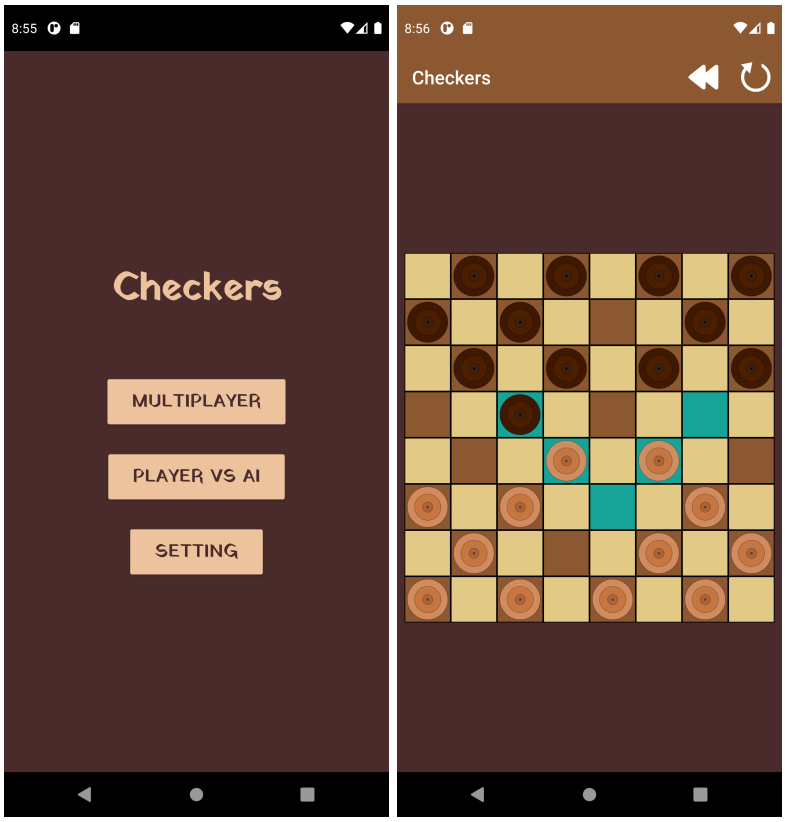
Dans le cadre de la licence informatique, plus précisement dans le cours de developpement mobile, nous avons choisi de réaliser **un jeu de dame** sur Android et IOS.

2 *Regle du jeu de dame* [2]

- Le jeu se joue à 2 joueurs sur un plateau de taille n * n.
- Les joueurs jouent chacun à leur tour. Les blancs commencent toujours.
- Le but du jeu est de capturer tous les pions adverses.
- Si un joueur ne peut plus bouger, même s'il lui reste des pions, il perd la partie.
- Chaque pion peut se déplacer d'une case vers l'avant en diagonale.
- Un pion arrivant sur la dernière rangée et s'y arrêtant est promu en « dame ».
- La dame se déplace sur une même diagonale d'autant de cases qu'elle le désire, en avant et en arrière.
- Un pion peut en prendre un autre en sautant par dessus le pion adverse pour se rendre sur la case vide située derrière celui-ci. Le pion sauté est retiré du jeu.
- La prise est obligatoire.
- Lorsque plusieurs prises sont possibles, il faut toujours prendre du côté du plus grand nombre de pièces.
- La dame doit prendre tout pion situé sur sa diagonale (s'il y a une case libre derrière) et doit changer de direction à chaque fois qu'une nouvelle prise est possible.

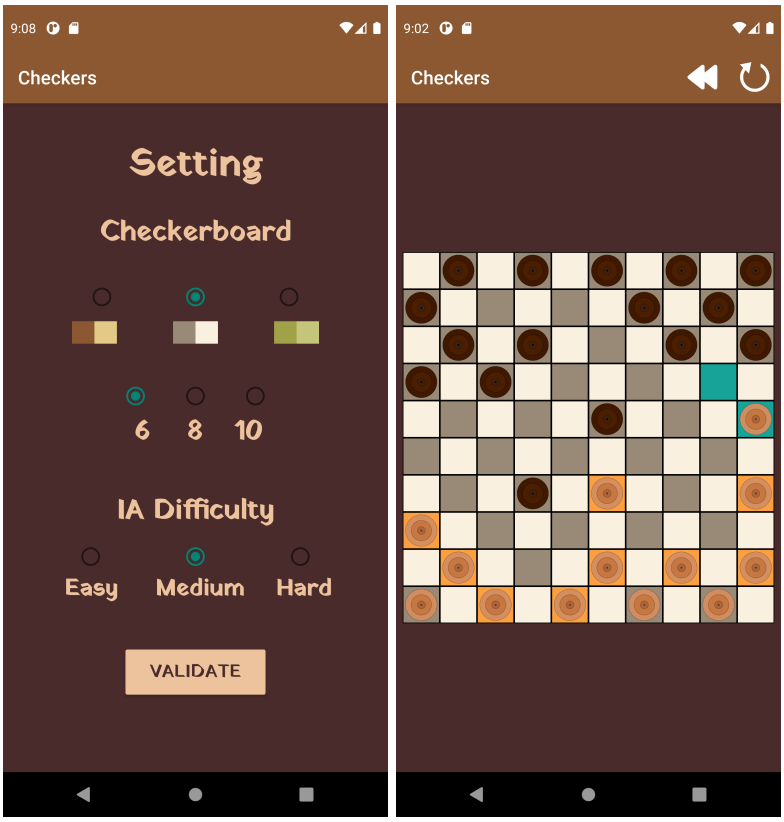
3 Description générale de l'application

Voici une capture du menu principal et d'une partie du jeu

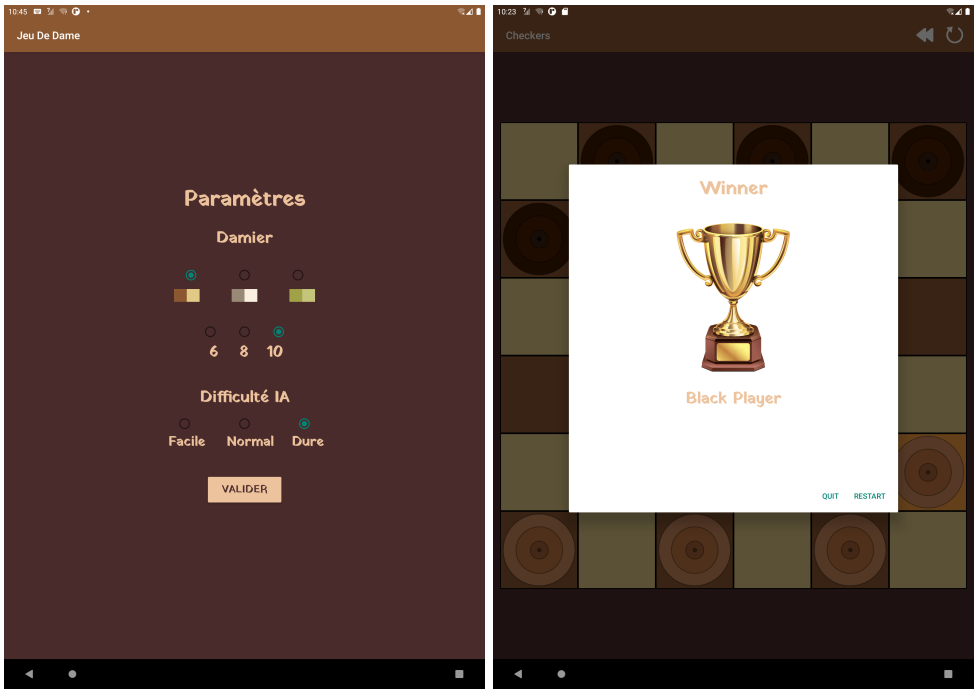


Fonctionnalités proposé par le jeu :

1. Un mode multijoueur :
   Ce mode consiste à faire affronté 2 joueurs sur un même plateau de jeu.
   Les joueurs joue tour par tour sur les 2 cotés du téléphone. De ce fait, pour des raisons d'IHM, on a décidé d'éxclure le mode *paysage* du jeu
2. Un mode joueur contre une **Inteligence Artificielle**
   Ce mode consiste à faire affronté un joueur contre un IA. L'IA a été implémenté en utilisant l'algorithme *minimax* [1]
3. Paramètres Permet de personnaliser la couleur des cases, le taille du damier (6*6, 8*8, 10*10) et la difficulté de l'IA (Facile, Moyen, Difficile)



4. Option retour en arrière («) Le bouton gauche de l'OptionMenu qui permet de revenir en arrière sur la partie en cours (disponible sur les 2 modes de jeu cité ci-dessus)
5. Option restart Game Le bouton droit de l'OptionMen qui permet de restart la partie en cours
6. Application bilingue et responsive
   L'application est disponible en français (par défaut) et en anglais.
   De plus, l'applcation est résponsive, et s'adapte sur tout taille d'écran.



Et enfin, un petit pop up sympa lorsqu'un joueur gagne la partie, ça n'a pas de prix

4 Architecture du code

L'implémentation du jeu est différente sur les deux plateformes.

En éffet, au début du projet du projet, nous étions parti sur une même base.

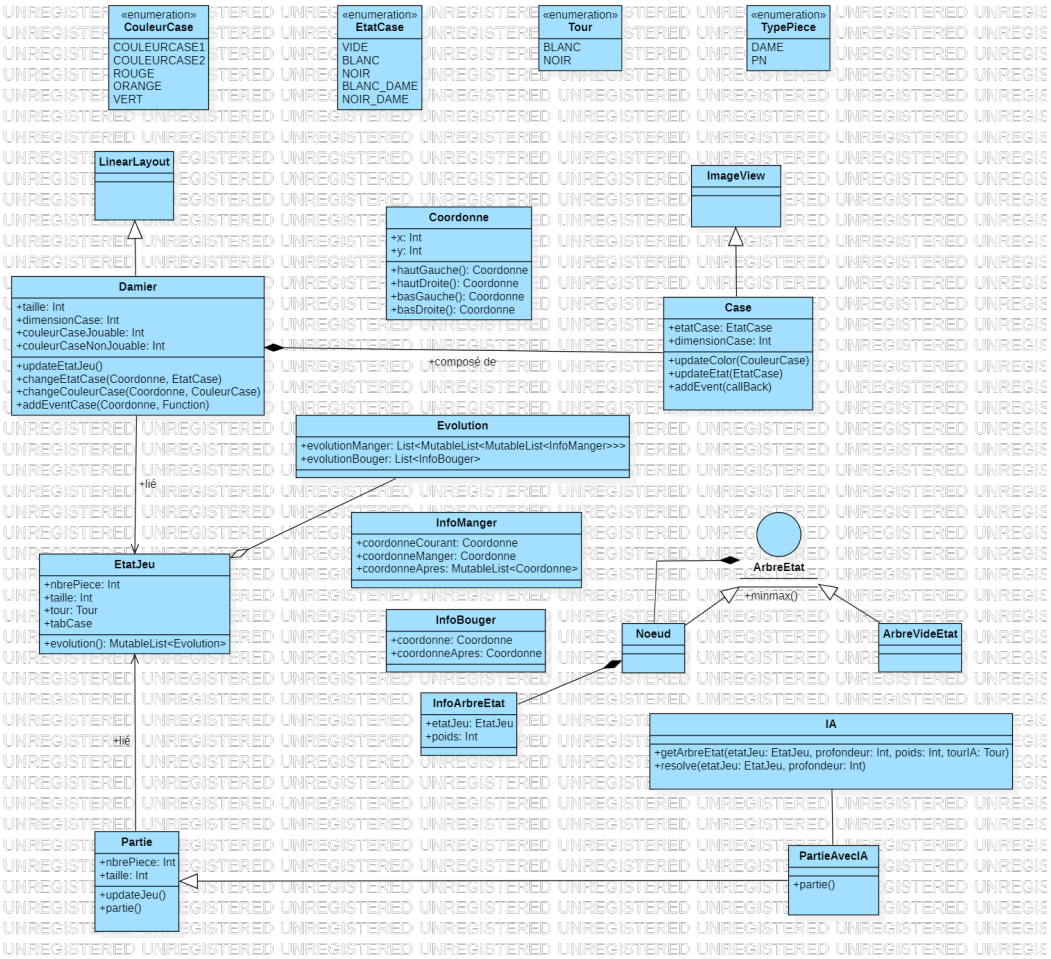
Mais pour implementer l'IA sur Android, on a du retravaillé tout la struture du code (ce qui fut une bonne idée d'ailleurs)

4.1 Android

La structure du code adopté pour l'application sur Android se veut plus précis et éfficace mais moins académique.

En effet, dans l'implémentation, on a éviter de mettre beacoup de classes utilitaires unitiles comme Joueur, Plateau, Piece, etc...

Voici le diagramme de classe de la version Android :



En résumé :

4.1.1 EtatCase (Model)

Cette classe représente l'état du Jeu à un instant donné. Cela se représente très facilement par un **List<List<EtatCase»**

Avec cette classe, on a la possibilité de retourner tout les évolutions possible du jeu. C'est à dire, les cases que l'on pourra manger et les endroits où on pourra se déplacer.

Les méthodes manger et bouger permettent à partir des classes **InfoManger** et **InfoBouger** de retourner une nouvelle **EtatJeu** avec les changements adéquats.

Cela nous permet de faire évoluer notre partie et de mettre en place un IA

4.1.2 Damier (Vue)

Cette classe est un **LinearLayout** qui va contenir des **Case** qui sont des **ImageView**.

C'est elle qui va afficher le damier et qui va mettre en place tout la partie Vue de notre jeu. C'est elle aussi qui gère les evenements sur les cases.

4.1.3 Partie (Controlleur)

Cette classe permet la liaison entre la vue **Damier** et le model **EtatCase** et permet de mettre en place les evenement, colorier les cases, etc...

4.1.4 IA

Extrait de code de la fonction minMax

```
fun minMax(tourIA: Tour): Int {

      return when(this) {

          is ArbreVideEtat -> 0
          is NoeudArbreEtat ->

              if ( fils.all { it is ArbreVideEtat } ) infoArbreEtat.poids
              else {

                  if (infoArbreEtat.etatJeu.tour === tourIA)
                      max(fils.map { it.minMax(tourIA) } as MutableList<Int>)
                  else min(fils.map { it.minMax(tourIA) } as MutableList<Int>)

              }
          else -> 0
      }

  }
```

4.2 iOS

5 Quelques points délicats/intéressants

6 Conclusion

Références

[1] Algorithme minimax. https://fr.wikipedia.org/wiki/Algorithme_minimax.

[2] Regle du jeu de dame. http://www.lecomptoirdesjeux.com/regle-jeu-dames.htm.