

INSTITUT NATIONAL DES SCIENCES APPLIQUÉES
DE RENNES

Diaballik

Rapport de conception

Morgane CAM
Isabelle GUILLOU

Septembre 2018 - Janvier 2019
INSA de Rennes

Table des matières

1	Diagramme de classes	2
2	Diagrammes de séquence	2
2.1	Création d'une nouvelle partie	2
2.2	Chargement d'une partie sauvegardée	3
2.3	Comportement d'un tour	3
2.4	Quitter une partie	3
3	API REST	3
4	Mock-up	4
4.1	Initialisation partie	4
4.2	"Nouvelle partie"	4
4.2.1	Création de la partie	4
4.2.2	Tours du premier joueur	4
4.2.3	Tours de l'ordinateur	4
4.2.4	Tours du second joueur	4
4.2.5	Quitter	5
4.2.6	Fin du jeu	5
4.3	"Charger une partie sauvegardée"	5
4.4	"Rejouer une partie"	5
5	Annexes	6

Tous les diagrammes présentés dans ce rapport sont joints sous forme vectorielle.

1 Diagramme de classes

Le diagramme de classe décrit l'organisation des données qui modélisent le jeu. Nous avons utilisé différents patrons de conception qui nous seront utiles lors du développement :

- Builder : Builder est premièrement utilisé pour la construction du plateau de jeu dans la classe BuildBoard. Il permet différentes implémentations de création, ce qui s'avère utile dans notre cas puisqu'il existe plusieurs scénarios de jeu, c'est-à-dire différentes positions initiales des pièces et de la balle sur le plateau. Nous utilisons le patron Builder une seconde fois, dans la classe BuildGame, qui permet de créer une partie. Encore une fois, on pourra donc avoir plusieurs implémentations différentes. En effet, Game va varier en fonction du nombre de joueurs, mais aussi en fonction du fait que le joueur souhaite charger une partie sauvegardée, en créer une nouvelle, etc. La classe BuildGame appellera la classe BuildBoard.
- Strategy : Ce patron est utilisé lors du choix du niveau de jeu. Strategy va permettre de définir une famille d'algorithmes interchangeables dynamiquement. Il s'agit de la stratégie que l'ordinateur qui joue face à l'utilisateur va adopter. Ainsi, il sera possible de changer d'implémentation pendant l'exécution en fonction du contexte, et de s'adapter à une nouvelle situation.
- Command et Memento : Ces patrons sont utilisés lors des déplacements des pièces et balles. On pourra ainsi considérer un appel de méthode comme un objet pour pouvoir le stocker et le manipuler comme un objet. Grâce à eux, nous pourrons faire du undo/redo sur des actions lorsque l'on revisionnera une partie car nous stockerons toutes ces commandes dans une liste. Il permettront également de sauvegarder l'état de la pièce/balle à un instant donné sans faire exploser l'occupation de la mémoire. L'état sauvegardé pourra ainsi être ré-appliqué à l'objet.

D'autres patrons de conception seront utilisés même s'ils ne sont pas visibles dans le diagramme de classes, tels que Observer, Factory, etc.

2 Diagrammes de séquence

Nous avons également réalisés 3 diagrammes de séquences :

- création d'une nouvelle partie,
- chargement d'une partie sauvegardée,
- comportement d'un tour,
- quitter une partie.

2.1 Création d'une nouvelle partie

Lorsqu'un joueur souhaite commencer une partie, plusieurs choix s'offrent à lui. Il peut soit commencer une nouvelle partie, soit charger une partie sauvegardée qui a déjà été commencée. Dans le premier cas, il crée alors une nouvelle partie. Lors de la création d'une nouvelle partie, le joueur indique alors s'ils sont une ou deux personnes à jouer. Si le joueur est seul, il jouera donc contre une IA. Il va alors entrer son nom, la couleur des pièces qu'il veut, le scénario de la partie et le niveau de l'IA qu'il va affronter. Par contre, si 2 joueurs s'affrontent il faudra dans ce cas entrer les 2 noms des joueurs, attribuer chacune des 2 couleurs et entrer le scénario de la partie.

Toutes les informations que l'utilisateur entrent font partie du FrontEnd, et elles seront transmises au Backend qui va ainsi créer la partie.

2.2 Chargement d'une partie sauvegardée

Comme évoqué précédemment, lorsqu'un joueur souhaite commencer une partie, il peut faire le choix de charger une partie sauvegardée. Il va alors choisir quelle partie il va charger dans la liste des sauvegardes.

2.3 Comportement d'un tour

Un tour comprend six déplacements, trois de chaque joueur. Une action d'un joueur est divisée en deux étapes. La fonction `canDo()` permet de vérifier que la case que l'on vise est libre ou contient une pièce de la bonne couleur, et si le déplacement que l'on souhaite faire respecte les règles du jeu. La fonction `do()`, quant à elle, déplace l'objet sur la case visée. La commande est ensuite sauvegardée dans la pile des actions effectuées. Tant que le premier joueur n'a pas effectué ses trois actions, le second joueur attend son tour. Le jeu donne ensuite automatiquement la main au second joueur. Une fois que les deux joueurs ont joué, le tour est terminé.

2.4 Quitter une partie

Lorsque le joueur souhaite quitter une partie en cours de jeu, deux possibilités s'offrent à lui. S'il souhaite sauvegarder sa partie (première opérande du `alt`), le plateau, les déplacements, les joueurs, ... toutes les caractéristiques de la partie sont enregistrées dans un fichier `.json` à l'aide de la fonction `save()`. Le joueur quitte ensuite le jeu à l'exécution de la fonction `exit()`. S'il ne souhaite pas sauvegarder sa partie (seconde opérande du `alt`), le joueur quitte immédiatement le jeu à l'exécution de la fonction `exit()`.

3 API REST

L'API REST est fournie par le client au serveur. Les différentes actions possibles sont décrites ci dessous :

- création d'une partie PvP,
POST `/game/newGamePvP/{name_p1}/{col_p1}/{name_p2}/{col_p2}/{scenario}`
- création d'une partie PvIA,
POST `/game/newGamePvIA/{name_p1}/{col_p1}/{scenario}/{level}`
- bouger une pièce,
PUT `/player/{name_p}/movePiece/{tx}/{x2}/{ty}/{y2}`
- bouger la balle,
PUT `/player/{name_p}/moveBall/newMoveBall/{tx}/{x2}/{ty}/{y2}`
- sauvegarder la partie,
POST `game/save/{nomFichier.json}`
- charger une partie sauvegardée,
GET `game/load/{nomFichier.json}`
- replay une partie sauvegardée,
GET `game/replay/{nomFichier.json}/{board}/{undoCommands}`
- quitter / Sauvegarder et quitter.
PUT `game/exit`
PUT `game/save/{newNomFichier.json}/exit`

4 Mock-up

Les maquettes interfaces utilisateur permettent de mieux visualiser ce que nous allons développer par la suite. Cependant, il s’agit seulement d’un design initial de l’interface graphique de l’application. Les maquettes ont été réalisées à l’aide de IntelliJ, et donc de JavaFX. Nous avons créé un fichier .fxml par maquette.

4.1 Initialisation partie

La première page permet d’initialiser une partie : cliquer sur “Nouvelle partie”, en charger une s’il y a une sauvegarde de disponible : cliquer sur “Charger la partie sauvegardée” et rejouer la dernière partie sauvegardée : cliquer sur “Rejouer la partie”. Si le joueur ne veut finalement pas jouer, il peut cliquer sur “Quitter” et le jeu se fermera (cf. Annexe n°1).

4.2 “Nouvelle partie”

4.2.1 Création de la partie

Lors de la création d’une nouvelle partie, le premier joueur choisit le placement initial des pièces et balles (scénario Standard, Ball random ou Enemy among us), son nom et sa couleur (Rouge ou Bleu). Il indique également s’il souhaite jouer contre un ordinateur ou une seconde personne. Il valide ensuite ses choix en cliquant sur “Valider” (cf. Annexe n°2).

S’il a choisi de jouer contre une IA, il doit ensuite choisir son niveau : noob, starting ou progressive (cf. Annexe n°3).

Cependant, s’il a choisi d’affronter une autre personne, le second joueur aura à son tour à choisir un nom et une couleur, puis valider ses choix. N’ayant que deux couleurs de joueur possibles, il ne pourra en réalité sélectionner que la couleur non-choisie par le premier joueur (cf. Annexe n°4).

4.2.2 Tours du premier joueur

Une fois la partie créée, ce sera toujours au premier joueur de commencer à jouer. Pour chaque action, il aura le choix entre déplacer une de ses sept pièces (représentée par un disque) ou sa balle (représentée par une sphère). Une fois un déplacement effectué, le compteur représentant les mouvements restants se décrémente. Une fois trois déplacements faits, le tour du premier joueur est automatiquement terminé et celui du second joueur commence.

Dans notre exemple, la partie commence. Le premier joueur a les pièces bleues. Il a déplacé deux fois une de ses pièces. Il ne lui reste donc qu’une action à effectuer comme indiqué par le compteur (cf. Annexe n°5).

4.2.3 Tours de l’ordinateur

Lorsque le second joueur est un ordinateur, pendant son tour, le premier joueur peut voir ses déplacements réalisés. Une fois ses trois mouvements passés, le jeu redonne automatiquement la main au premier joueur qui peut à nouveau jouer (cf. Annexe n°6).

4.2.4 Tours du second joueur

Lorsque le second joueur est un humain, le premier joueur ne peut pas voir ses déplacements car ils partagent le même écran d’ordinateur. L’écran lui permet alors d’effectuer ses trois déplacements (cf. Annexe n°7).

4.2.5 Quitter

Pendant son tour, un joueur humain peut choisir de quitter la partie. Pour cela, il lui suffit de cliquer sur le bouton “Quitter” et une fenêtre apparaît. On a alors le choix entre “Sauvegarder et quitter” : la partie en cours est sauvegardée puis le jeu se ferme, ou “Quitter sans sauvegarder” : le jeu se ferme sans sauvegarder la partie en cours (cf. Annexe n°8).

4.2.6 Fin du jeu

Dès qu’un joueur réussit à placer sa balle sur la ligne à l’extrême opposé du plateau, il gagne. La partie est alors terminée et le nom du vainqueur s’affiche. Le premier joueur à alors le choix de “Quitter” le jeu, ou retourner au “Menu” de démarrage (cf. Annexe n°9).

4.3 “Charger une partie sauvegardée”

Lorsque le joueur a sauvegardé une partie, il peut la reprendre là où il s’était arrêté en sélectionnant “Charger une partie sauvegardée” au démarrage du jeu (cf. Annexe n°1). Il a alors le choix de reprendre une partie parmi toutes celles suspendues (cf. Annexe n°10). Le joueur retrouve alors le tour qu’il avait suspendu.

4.4 “Rejouer une partie”

Au démarrage, le joueur peut demander à revoir la dernière partie terminée (si elle existe) en sélectionnant "Rejouer une partie" (cf. Annexe n°1). Il choisit la partie qu’il souhaite revisualiser (cf. Annexe n°10) puis retrouve un plateau sur lequel sont placées pièces et balles en fonction du scénario qui avait été choisi. Il peut ensuite revoir les déplacements un à un en cliquant sur "Redo". S’il souhaite revenir en arrière, il peut cliquer sur "Undo" et le dernier déplacement s’annule (cf. Annexe n°11).

5 Annexes



FIGURE 1 – Page d'accueil du jeu Diaballik

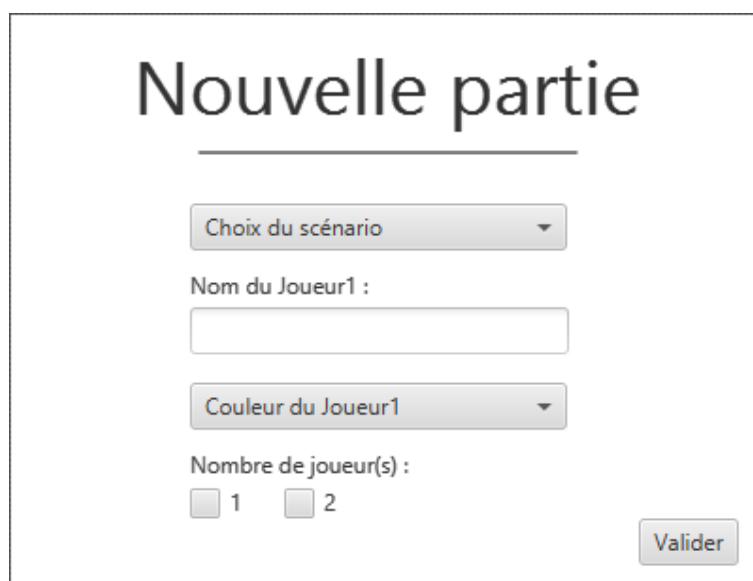
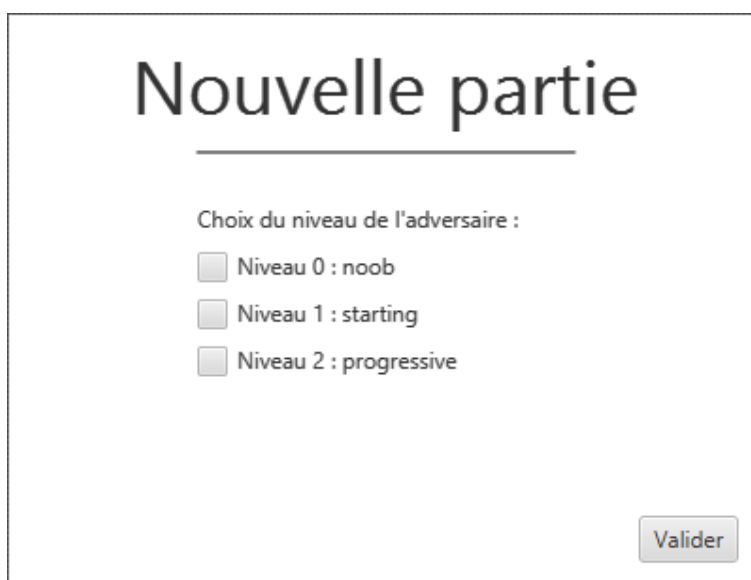


FIGURE 2 – Création d'une nouvelle partie : configuration du premier joueur



Nouvelle partie

Choix du niveau de l'adversaire :

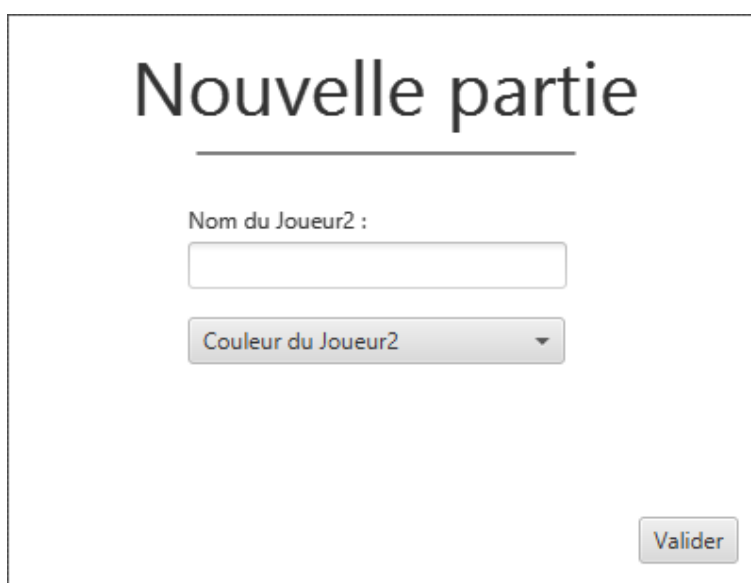
☐ Niveau 0 : noob

☐ Niveau 1 : starting

☐ Niveau 2 : progressive

Valider

FIGURE 3 – Création d’une nouvelle partie : configuration du niveau de l’ordinateur



Nouvelle partie

Nom du Joueur2 :

Couleur du Joueur2 ▼

Valider

FIGURE 4 – Création d’une nouvelle partie : configuration du second joueur

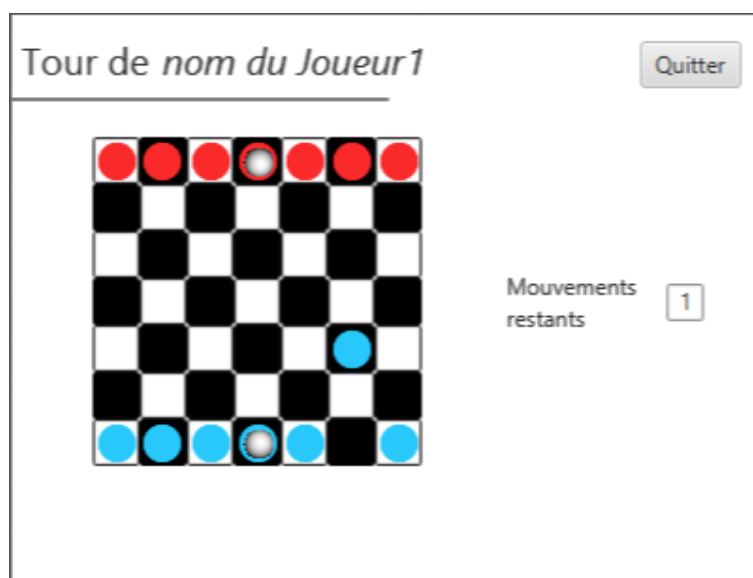


FIGURE 5 – Représentation d'un tour du premier joueur

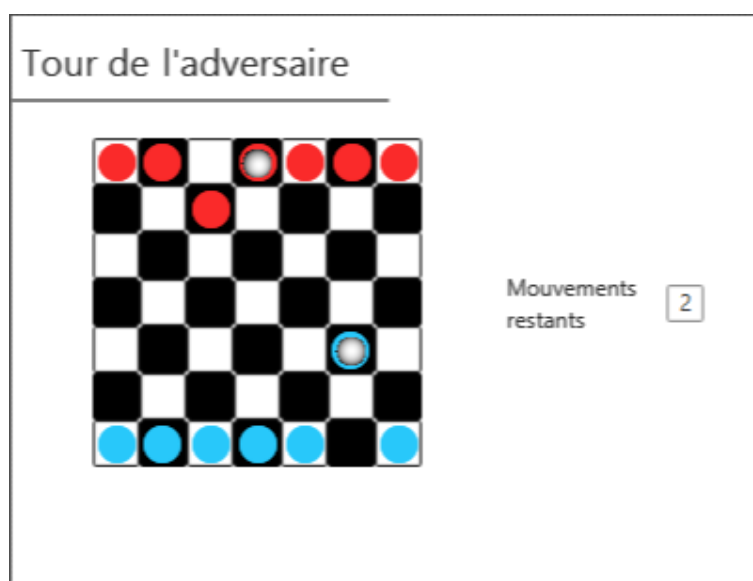


FIGURE 6 – Représentation d'un tour lorsque le second joueur est une IA

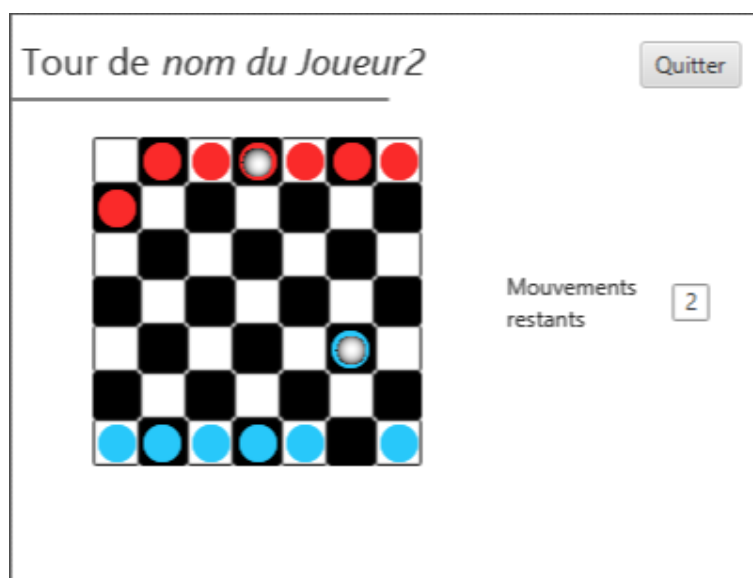


FIGURE 7 – Représentation d'un tour lorsque le second joueur est un humain

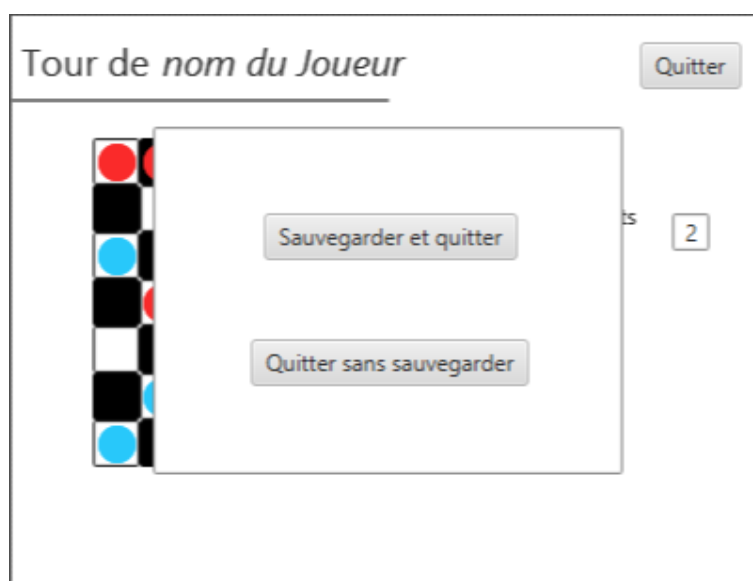


FIGURE 8 – Quitter la partie en cours



FIGURE 9 – Fin d'une partie



FIGURE 10 – Choix d'une partie à reprendre ou revisualiser

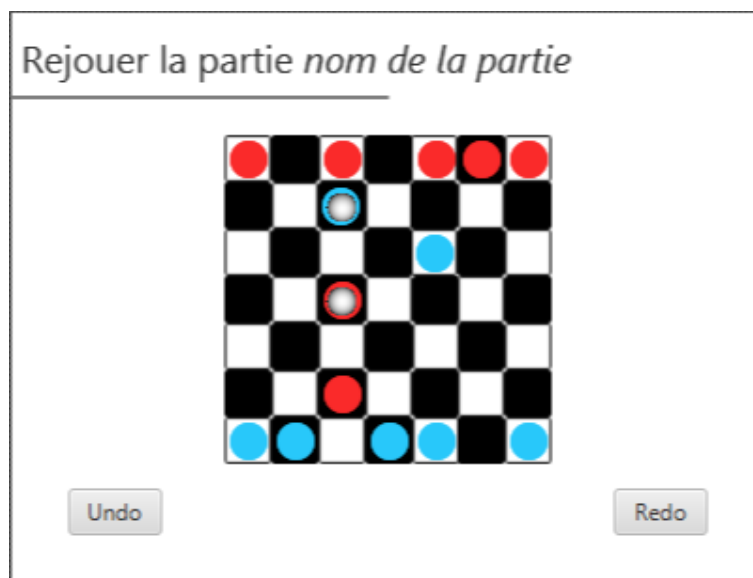


FIGURE 11 – Revisualisation d'une partie sauvegardée