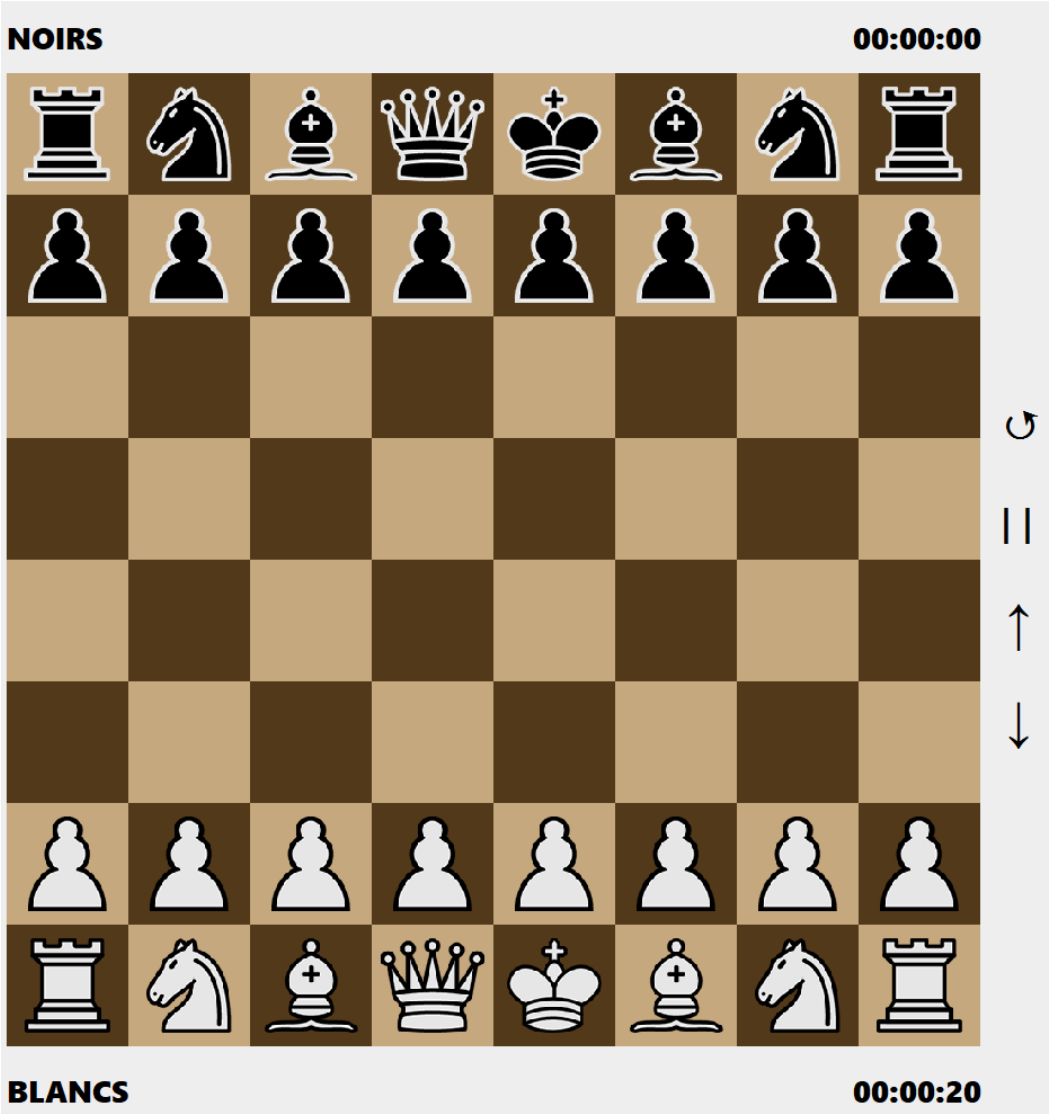
ECHEC ET MAT



Groupe 50 – Promo 63

REIS Alexis

REKIK Yasmine

TEYSSIER Lisa

UNGRIA Arthur

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES LYON

Nous avons tous joué aux échecs au moins une fois dans notre vie. Ce jeu oppose deux joueurs avec chacun 16 pièces et demande une grande réflexion et une connaissance des règles bien précise si l'on souhaite piéger son adversaire et remporter la partie. Vieux de plusieurs siècles, c'est un jeu aujourd'hui très populaire en ligne où les joueurs s'affrontent chacun devant leur écran. Ayant désormais des connaissances algorithmiques bien développées, nous avons donc décidé de développer ce jeu de plateau.

Nous avions pour but de réaliser un échiquier jouable par deux joueurs en local. Chacun des joueurs avec ses pions et avec la possibilité de jouer autant de parties qu'ils le souhaitent. Les coups jouables devront être légaux (vérification si le coup met en échec et si oui impossibilité de jouer ce coup, déplacement des pièces selon les règles…) et seront affichés sur le plateau afin d'aider le joueur à mieux visualiser ces possibilités de jeu.

Les parties pourront être classiques ou on pourra démarrer la partie à des stades prédéfinis pour effectuer des parties plus rapides ou tout simplement s'entrainer dans certaines situations. Ces stades définis seront complétements personnalisables selon la notation FEN déjà normée.

Notre programme s'est divisé en trois packages (Graphique, Pieces et Echecs) que nous allons détailler un par un. Les diagrammes UML présentés dans ce rapport ne sont que des extraits car il était trop grand pour l'insérer dans ce type de document. Il est disponible en intégralité dans l'archive du projet et dans la bibliographie à la fin de ce rapport.

Tableau 1 : Pourcentage d'investissement dans le projet

Alexis	35 %
Arthur	35 %
Lisa	15 %
Yasmine	15 %

1er package : Graphique

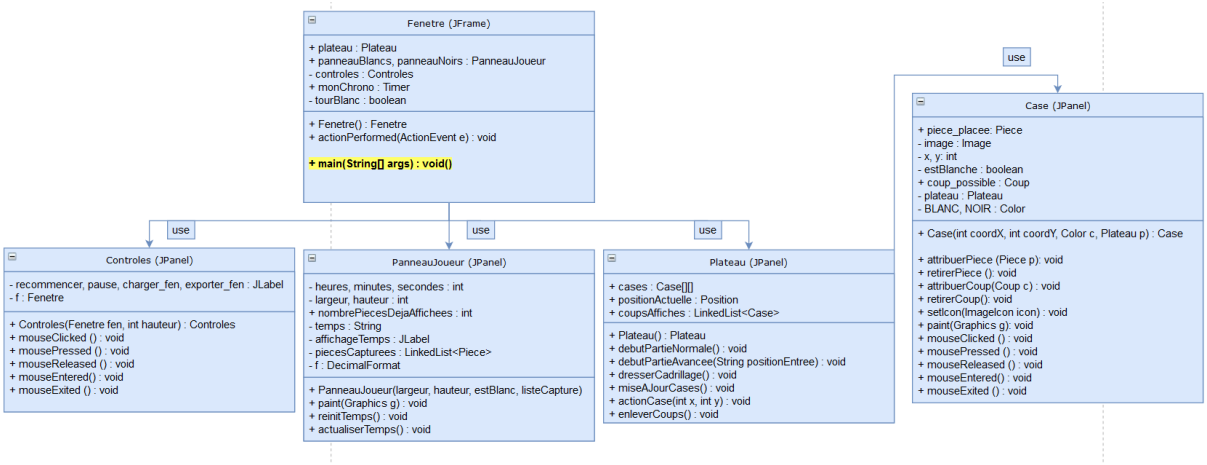


Figure 1 : Diagramme UML du package Graphique

La classe contenant le main est la classe Fenêtre, héritée de JFrame dont le constructeur forme une fenêtre contenant l'échiquier pour jouer, des panneaux propres à chaque joueur, un panneau de contrôles et un chrono permettant de chronométrer le temps de réflexion des joueurs.

Plateau est une classe héritée de JPanel, disposée en grille 8 par 8 via un GridLayout. Chaque cellule contient un Case, un JPanel contenant ou non une Piece en fonction de la position de l'échiquier. Nous affichons la pièce via la méthode paint et drawImage qui permet de redimensionner l'image en fonction de la taille de la Case. Si on clique sur une Case contenant une Piece, les coups légaux possibles par cette pièce sont générés et affichés sur les Cases concernées via un disque de couleur verte. Chacune de ces cases se retrouve alors dans une liste dont nous pouvons rapidement supprimer le contenu via une méthode afin de supprimer les points verts à l'affichage et de repaint seulement les cases concernées. A chaque déplacement de pièce, seulement les Cases concernées par le déplacement sont actualisées. La méthode miseAJourCases met à jour toutes les cases selon la position actuelle.

PanneauJoueur, est un JPanel propre à chaque joueur, il affiche trois éléments : la couleur du joueur concerné et le temps qu'il a écoulé depuis le début de la partie (via deux JLabel) et les pièces qu'il a capturées (via paint). Cet affichage des pièces capturées est mis à jour à chaque changement de tour et si le nombre de pièces capturées est différents du nombre de pièces déjà affichées. Nous avons veillé à ce que la taille de ces pièces et de leur espacement puisse assurer l'affichage de toutes les pièces si toutes ont été capturées. L'affichage du timer est mis à jour chaque seconde, pour le joueur qui joue uniquement. Pour afficher le temps avec deux chiffres, nous avons utilisé des DecimalFormat.

Figure 2 : Le PanneauJoueur dans le pire des cas possible pour le joueur adverse

Controles est un JPanel contenant des JLabel permettant de recommencer une partie, mettre la partie en pause, charger une position FEN ou exporter l'actuelle pour pouvoir reprendre la partie plus tard ou s'entrainer sur une position de l'échiquier bien définie. Lorsque que l'on clique dessus, un JOptionPane est affiché et permet d'effectuer l'action voulue. Ce sont des JLabel car nous avons eu des problèmes de redimensionnement en utilisant des JButton.

D'ailleurs tous les emplacements et tailles de ces éléments (taille de police incluse) sont fonction de la résolution de l'écran maximale et exploitable sur votre moniteur. L'affichage s'adapte parfaitement à chaque ordinateur et même chaque système d'exploitation que nous avons pu tester.

2ème package : Pièces

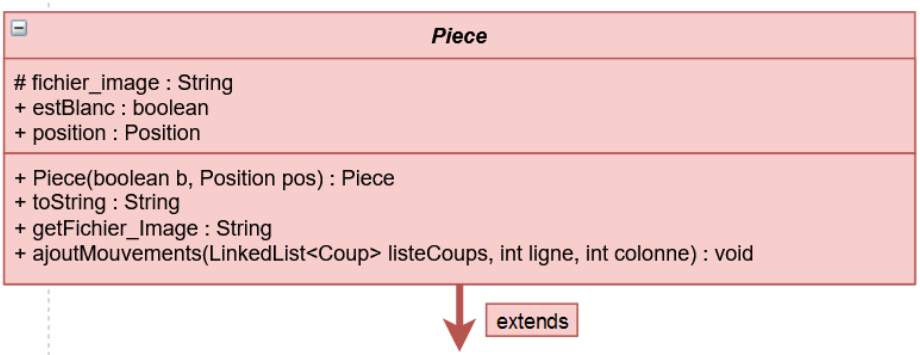
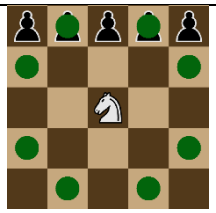
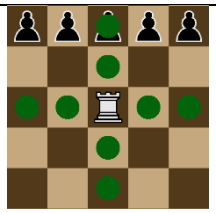


Figure 3 : La classe abstraite Piece sans ses classes héritées

Chaque pièce est une classe héritée de la classe abstraite Piece que nous avons créée. Ce sont les classes Pion, Tour, Cavalier, Fou, Roi et Dame. Elles possèdent chacune un constructeur et ont en attribut :

- un lien vers le fichier image du pion correspondant, deux liens par pièce, un pour chaque couleur de pièce
- un boolean indiquant si la pièce est blanche ou noire, afin de choisir le bon fichier et donc afficher la bonne pièce
- une Position, soit le plateau actuel représentée par des pièces qui nous permet de tester les positions autour de la pièce pour savoir si elle peut s'y déplacer ou non

Les pièces ont chacune des déplacements possibles différents. La méthode ajoutMouvements est définie pour chacune de celles-ci et renvoie une liste de coups jouables par la pièce pour la position actuelle du plateau. Cette méthode est appelée pour chacune des pièces présentes sur l'échiquier pour faire la liste de tous les Coups possiblement jouables. C'est aussi utilisé pour afficher les coups possibles lorsque que l'on clique sur une Piece pour afficher ses coups jouables sur le Plateau. Voici ce que cela donne pour chacune des pièces du jeu :

Le pion	Le cavalier	La tour	Le fou	La dame	Le roi
					

Lorsque le programme détecte que l'un des rois est en échec, le fichier image de cette pièce est modifié par la même pièce avec un contour rouge pour indiquer que l'on est en échec et que le joueur devra jouer pour ne plus l'être. Ci-contre le roi des noirs est en échec.

3ème package : Echecs

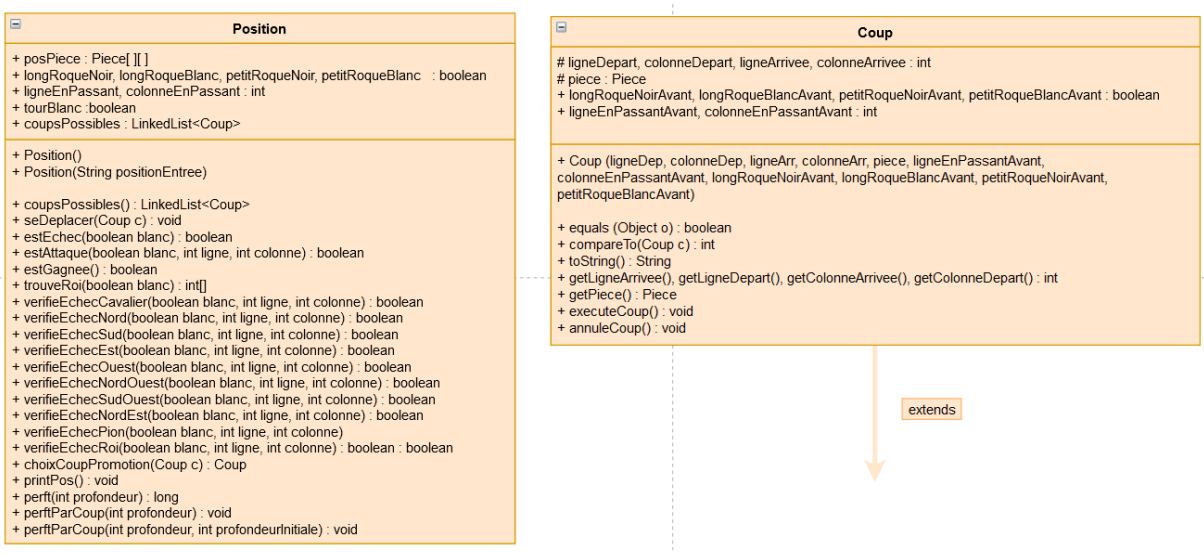


Figure 4 : La classe Position et la classe Coup sans ses classes héritées

- La génération des coups

Le jeu d'échec détient un grand nombre de coups. Dans notre programme, dans l'optique de la POO, chacun de ces coups est une classe héritière de la classe Coup. Parmi ces classes héritières on compte : le Coup Deux Cases Pion, le Coup Roque, le Coup Capture, le Coup Capture Promotion, le Coup En Passant et le Coup Promotion. Les coups promotions sont effectuables lorsque qu'un pion traverse l'entièreté du plateau sur sa hauteur, il est alors promu. Le joueur à le choix entre quatre pièces : la dame, le fou, la tour ou le cavalier. Ce choix est obligatoire et est effectué en appelant un JOptionPane.

Tous les déplacements sont repérés par les lignes et colonnes de départ et d'arrivée. On vient tester les cases où le déplacement d'une des pièces est possible. Si la case d'arrivée est vide, ou si une pièce adverse y est placée et que le coup joué ne nous met pas en échec, le coup est légal, il peut être joué. Tous ces coups légaux sont stockés dans la liste coupsPossibles. Lorsque que l'on clique sur une pièce sur le plateau, l'ensemble des coups légaux de cette pièce sont affichés. On les place alors dans la liste coupsAffiches de Plateau en prenant chaque coup de coupsPossibles pour lequel la pièce déplacée est celle sur laquelle on vient de cliquer. Les déplacements des pièces en interne sont effectués par la classe Position.

Ces coups possibles sont générés procéduralement en parcourant l'ensemble des cases du plateau une à une et en appelant une fonction de l'objet Piece qui ajoute ces coups à la liste. La liste ainsi générée contient cependant des coups illégaux car ils peuvent mener le joueur en échec. Un filtrage est alors effectué en jouant chaque coup et vérifiant que la position suivante ne met pas le joueur qui joue en échec.

- Position

C'est la classe la plus longue du programme. C'est elle qui gère l'échiquier et met en place en permanence la position et le déplacement des pièces. C'est ici que l'on gère les droits de coups spéciaux et où on vérifie après chaque coup si un des deux joueurs est en échec, s'il y a échec et mat, si un pion peut être promu, etc…

L'attribut posPieces est une matrice de 8 lignes et 8 colonnes contenant les pièces du plateau sur leur emplacement. Ainsi sont déduites les coordonnées des pièces.

La vérification de l'échec du roi s'effectue sur les lignes, les diagonales et les cases correspondant au déplacement du cavalier par rapport à celle du roi par les fonctions verifieEchecXXX qui renvoient une valeur booléenne. Si aucune de nos pièces n'est entre une pièce adverse et notre roi et que cette pièce peut se déplacer de manière à éliminer le roi la fonction renvoie true. Le roi est alors en échec et les seuls coups qui seront affichables et jouables par l'autre joueur seront les coups qui éliminent la pièce adverse mettant le roi en échec ou déplacent le roi de manière à enlever l'échec. Si plus aucun coup n'est jouable, c'est-à-dire que la liste coupsPossibles générée est vide, il y a échec et mat. La fin de partie est indiquée par un JOptionPane qui permet, une fois la défaite digérée, de recommencer une partie si on le souhaite.

A noter dans cette classe, un constructeur qui permet d'entrer une position FEN.

- La notation FEN (Forsyth-Edwards Notation)

Cette notation permet de représenter la position de toutes les pièces sur l'échiquier.

Exemple pour un échiquier en début de partie :

$$rnbqkbnr/pppppppp/8/8/8/8/PPPPPPPP/RNBQKBNR\ w\ KQkq\ -\ 0\ 1$$

Lien pour plus d'informations : https://www.dcode.fr/notation-echecs-fen

Pourquoi utiliser cette notation :

Tout d'abord, c'est plus rapide pour tester les coups possibles (roque, en passant…) et les problèmes du programme, plutôt que de devoir avancer toutes les pièces manuellement pour se trouver dans une configuration voulue.

De plus, la notation FEN rend plus simple l'enregistrement d'une position et la possibilité de la réutiliser, pour reprendre une partie ultérieurement par exemple. Pour cela :

- On enregistre d'abord la position actuelle, et on invite le joueur à la copier en quittant la partie
- A chaque lancement du jeu, on propose au joueur de coller une FEN afin de reprendre la partie à laquelle il jouait

Choix algorithmiques :

Pour récupérer la FEN sous forme d'une chaine String et la lire caractère par caractère :

- Attribution d'une pièce blanche ou noire (selon si le caractère est une majuscule ou minuscule) à chaque case du tableau posPieces qui comprend les pièces présentes sur le plateau de jeu ligne par ligne
- Détermination du joueur dont c'est le tour, des possibilités de roque et d'en passant

Pour récupérer la position du plateau de jeu pour la convertir en FEN (chaine de caractères) en parcourant le tableau posPieces qui comprend les pièces présentes sur le plateau :

- Remplir la chaine de caractères correspondant à la FEN en fonction de la position du plateau, du joueur dont c'est le tour, des possibilités de roque et d'en passant

- Perf

Les règles des échecs étant très complexes (surtout pour les en passant et les droits de roques), il a été nécessaire de vérifier l'exactitude de la génération des coups. C'est le rôle de la fonction perft. Cette fonction énumère de manière exhaustive l'ensemble des positions atteignables à une certaine profondeur de jeu (par exemple 8 coups). Le temps nécessaire au calcul de cette fonction croit de manière exponentielle. Il n'est pas recommandé de dépasser une profondeur de 7 ou 8 grand maximum

au vu de la vitesse de notre programme. Il est alors possible de vérifier, à l'aide d'un autre programme d'échec réputé fiable implémentant la même fonctionnalité que l'on obtient les mêmes résultats. Ceci sur de nombreuses positions différentes à différents stages de la partie. Cela nous a permis de corriger de nombreux bogues rarement visibles en partie normale humaine surtout sur la gestion des droits d'en passant. Nous avons utilisé le programme d'échecs Stockfish pour effectuer ces vérifications (https://github.com/official-stockfish/Stockfish).

Conclusion :

Afin de conclure, ce travail de groupe, aussi enrichissant soit-il, a été fastidieux. En effet pour travailler en groupe nous avons utilisé le logiciel Github, qui nous permettait de voir ce que chacun faisait et de pouvoir actualiser le programme. Mais sa prise en main a été longue et nous a fait perdre beaucoup de temps. L'écart de niveau entre les membres du groupe s'est aussi fait ressentir bien que nous n'hésitions pas à nous poser des questions entre nous pour toujours pouvoir avancer de son côté de manière hebdomadaire. Les cours à distance ont aussi été un frein, néanmoins nous avons su nous adapter afin de terminer ce projet.

Cependant, notre programme aussi fonctionnel soit-il, reste encore très imparfait. Si du côté positif nous pouvons noter notre affichage complètement adaptatif, et la génération parfaite des coups légaux nous pouvons aussi nous reprocher de ne pas avoir fait d'animations ou d'effets sonores. Aussi avec plus de temps et d'investissement nous aurions pu améliorer l'affichage de certains éléments graphiques comme celle de choix de promotion ou encore les interfaces permettant de rentrer une position FEN qui ne sont que des JOptionPane actuellement. Nous aurions pu également développer une petite intelligence artificielle permettant de jouer contre la machine.

Bibliographie :

- DCode [en ligne]. [consulté le 30 Mars 2021]. Notation Forsyth-Edwards. Disponible sur https://www.dcode.fr/notation-echecs-fen
- De Neef, Daniel. Tuto de rien [en ligne]. Derien Daniel, [consulté le 30 Mars 2021]. Les Switch sur Java. Disponible sur https://www.tutoderien.com/les-switch-sur-java/
- Meyer-Kahlen Stefan. Le fou numérique [en ligne]. Juin 2001, Mis à jour le 12 Novembre 2012, [consulté le 8 mars 2021]. Programmation aux échecs. Disponible sur https://lefouduroi.pagesperso-orange.fr/smk/programmation.htm#3
- Ecole Apprendre-les-echecs [en ligne]. Juin 2001, Mis à jour le 12 Novembre 2012, [consulté le 16 mars 2021]. Les règles du jeu d'échecs pour les débutants. Disponible sur https://ecole.apprendre-les-echecs.com/regles-echecs/
- Sebatian Lague [en ligne] [consulté le 30 Mars 2021]. Coding Adventure : Chess AI. Disponible sur https://www.youtube.com/watch?v=U4ogK0MIzqk&t=1243s
- Voici le lien à partir duquel, vous pouvez retrouver notre diagramme UML complet : https://app.diagrams.net/#G1LoW39Vyq7FxLlUy5sHlLmp8aE-eqmPqx