

COMPTE RENDU DE PROJET LICENCE 2

SEMESTRE 4 EN PROGRAMMATION

ORIENTE OBJET 2 (Java)

New Super Chess Deluxe Definition Edition++

Par Yann TOUSSAINT et Gabriel MOURAD

<Sommaire>

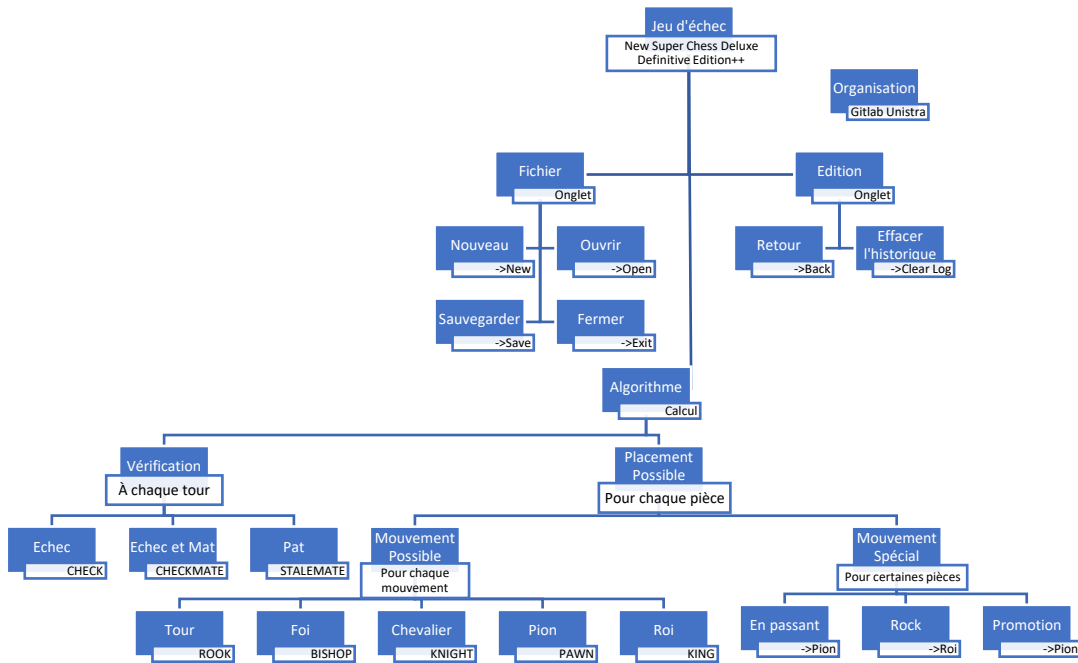
1	Organigramme	2
1.1	Organigramme Graphique.....	2
1.2	Graphique UML	2
2	Technologie utilisée	2
2.1	Java	3
2.2	Organisation	3
2.2.1	Equipe	3
2.2.2	Package.....	3
3	Implémentations.....	3
3.1	Choix de modélisation et d'implémentation.....	3
3.2	Algorithmes principaux	3
3.3	Sauvegarde en JSON.....	3
3.4	Pièces.....	5
3.5	Règles	5
4	Conclusion	5
5	Annexes	6

Code source : <https://git.unistra.fr/gmourad/new-super-chess-deluxe-definitive-edition>

1 Organigramme

Cette partie sera consacrée au cahier des charges synthétique que nous avons fait avant d'attaquer la programmation. Il nous permettra de se reposer dessus lorsque je parlerais de l'implémentation des différentes fonctions présentes.

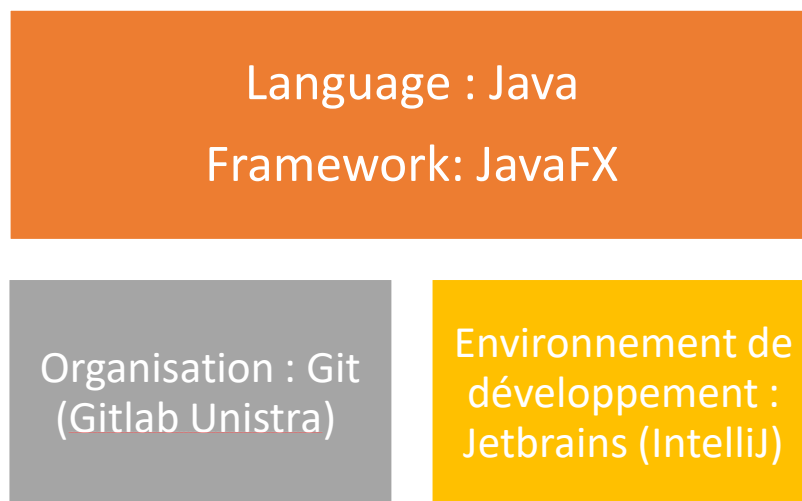
1.1 Organigramme Graphique



1.2 Graphique UML

Disponible dans l'Annexes (en portrait pour une meilleur lisibilité)

2 Technologie utilisée



Vue d'ensemble

2.1 Java

Swing n'étant plus à jour et très vieux dans sa conception, nous avons préférés opter pour la Solution JavaFX qui a certes, moins de documentation mais fait + de sens dans sa conception (à peu près comme de l'HTML) et utilisable en Java. Pour les dépendances, nous avons utilisé Maven et inclut JavaFX, et Gson, une librairie de Google qui simplifie l'utilisation des Json (nous en reparlerons + tard)

2.2 Organisation

2.2.1 Equipe

On a généralement travaillé physiquement ensemble sur les problèmes majeurs pour pouvoir réfléchir sur comment résoudre différents problèmes ensemble avant de coder ça ensemble directement ou séparément quand c'était trop fastidieux. C'était plutôt fun et je pense que c'était la meilleure chose à faire, nous avons utilisé Git pour faire des push, au début c'est surtout un seul compte car nous le faisons ensemble mais ensuite c'est devenu + diversifié !

2.2.2 Package

Code : se trouve dans le dossier « src/main/java ».

Textures : se trouve dans le dossier « src/main/resources ».

Documentation : se trouve dans le dossier « doc/ »

Programme : se trouve dans le fichier « out/artifacts/chess_jar.chess.jar » (pas présent dans le git)

3 Implémentations

3.1 Choix de modélisation et d'implémentation

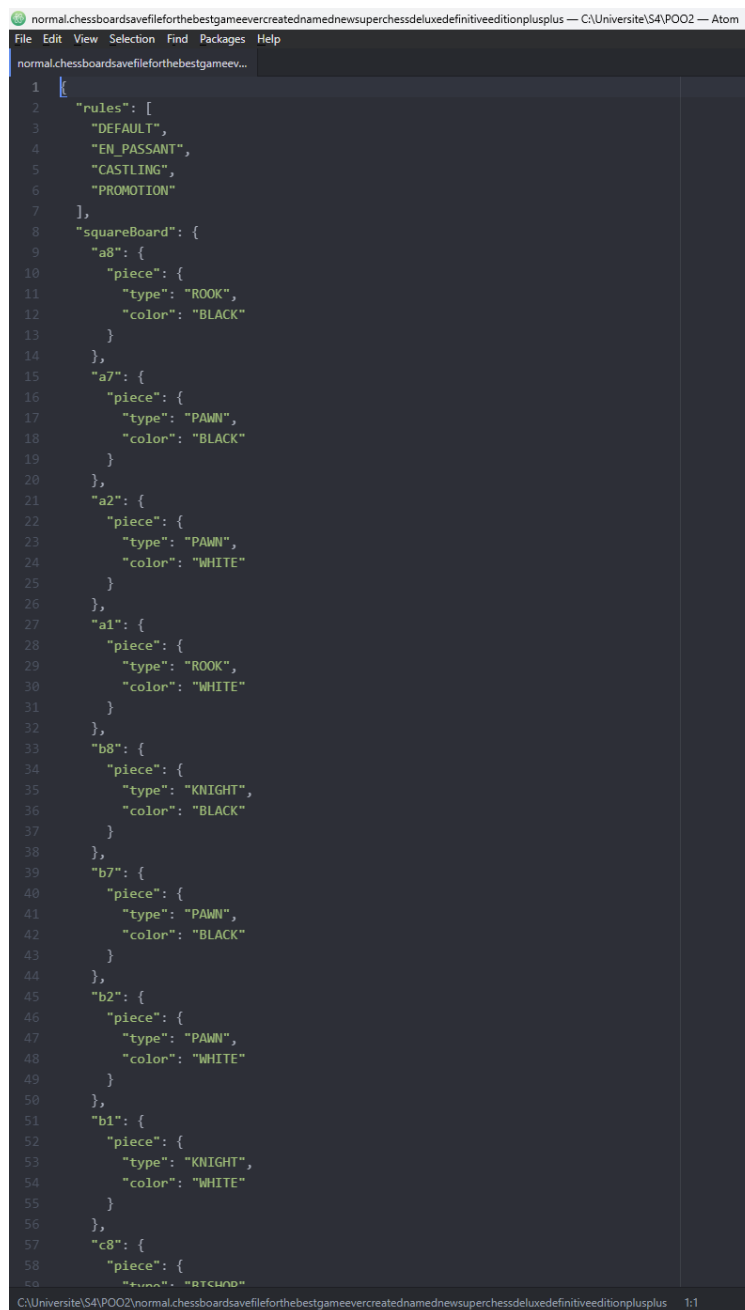
Le modèle et la vue sont séparés, dans notre main on crée le modèle avec **ChessBoard** et notre vue avec **ChessRender**, le contrôleur est implémenté dans **ChessRender**, car il est suffisamment simple pour ne pas à avoir à être séparé du rendu. **ChessBoard** contient principalement à qui c'est le tour, la case sélectionnée pour savoir quels coups possibles tester et afficher, et un tableau en 2 dimensions de cases d'échecs, chaque case contient ses coordonnées sur le plateau ainsi que la pièce contenue sur la case (qui vaudra null s'il n'y en a pas). **ChessRender** est composé principalement d'un tableau en 2 dimensions (avec les mêmes dimensions que le plateau d'échecs) de boutons qui correspondront chacun à une case du plateau.

3.2 Algorithmes principaux

Quand on clique sur une case, s'il y a une case sélectionnée, on va tester si la pièce sur la case sélectionnée peut aller sur la case cliquée alors on déplacera la pièce en question sur la case cliquée et on change de joueur actuel, sinon s'il n'y a pas de case sélectionnée, on sélectionne la case cliquée s'il y a une pièce de la couleur du joueur dont c'est le tour sur la case cliquée.

3.3 Sauvegarde en JSON

Nous avons mis en place un bouton qui permet de sauvegarder et de charger des parties d'échec, cela a été extrêmement utile pour debug et permet de faire des parties personnalisé (en modifiant le fichier, il est facilement compréhensible par un humain et cela permet de paramétrer le plateau).

A screenshot of a code editor window titled 'normal.chessboardsavefileforthebestgameevercreatednamednewsuperchessdeluxedefinitiveeditionplusplus'. The editor displays a JSON object representing a chess game state. The JSON is formatted with syntax highlighting and line numbers from 1 to 58. It includes a 'rules' array with 'DEFAULT', 'EN_PASSANT', 'CASTLING', and 'PROMOTION'. The 'squareBoard' object contains piece information for various squares, such as 'a8' (Black Rook), 'a7' (Black Pawn), 'a2' (White Pawn), 'a1' (White Rook), 'b8' (Black Knight), 'b7' (Black Pawn), 'b2' (White Pawn), 'b1' (White Knight), and 'c8' (Black Pawn). The editor's status bar at the bottom shows the file path and a 1:1 zoom level.

```
1 {
2   "rules": [
3     "DEFAULT",
4     "EN_PASSANT",
5     "CASTLING",
6     "PROMOTION"
7   ],
8   "squareBoard": {
9     "a8": {
10      "piece": {
11        "type": "ROOK",
12        "color": "BLACK"
13      }
14    },
15    "a7": {
16      "piece": {
17        "type": "PAWN",
18        "color": "BLACK"
19      }
20    },
21    "a2": {
22      "piece": {
23        "type": "PAWN",
24        "color": "WHITE"
25      }
26    },
27    "a1": {
28      "piece": {
29        "type": "ROOK",
30        "color": "WHITE"
31      }
32    },
33    "b8": {
34      "piece": {
35        "type": "KNIGHT",
36        "color": "BLACK"
37      }
38    },
39    "b7": {
40      "piece": {
41        "type": "PAWN",
42        "color": "BLACK"
43      }
44    },
45    "b2": {
46      "piece": {
47        "type": "PAWN",
48        "color": "WHITE"
49      }
50    },
51    "b1": {
52      "piece": {
53        "type": "KNIGHT",
54        "color": "WHITE"
55      }
56    },
57    "c8": {
58      "piece": {
59        "type": "PAWN",
60        "color": "BLACK"
61      }
62    }
63  }
64 }
```

À quoi ressemble le fichier, c'est en format JSON

Nous avons utilisé l'API Gson, et mis un type adapter pour que ce soit dans le format {coord : valeur} qui est + lisible et donc facilement éditable par un humain

Setup manuel d'un état du plateau :

Il est possible de mettre la configuration exacte que l'on souhaite en faisant un fichier JSON de sauvegarde avec tout d'abord les règles que l'on souhaite avoir, les règles :

- Par défaut (règles usuelles qui ne sont pas incluses dans les règles ci-dessous)
- Prise en passant
- Le roque
- La promotion

Après il y aura pour chaque case où il y a une pièce dans sa notation usuelle (b8, a4, e6...), le type de la pièce et sa couleur.

Le plus simple reste de lancer une partie, faire File -> Save et modifier par la suite manuellement le fichier JSON généré pour pouvoir enfin faire File -> Open

3.4 Pièces

Toutes les pièces d'échec classiques sont présentes, et vu qu'une pièce est définie par ses mouvements, on peut définir une pièce comme étant la combinaison de mouvements d'autres pièces, il est donc assez simple d'implémenter les pièces féériques qui sont la princesse, l'impératrice et le noctambule présentés sur la page https://fr.wikipedia.org/wiki/Pi%C3%A8ce_f%C3%A9erique :

- La princesse combine les mouvements d'un cavalier et d'un fou.
- L'impératrice combine les mouvements d'un cavalier et d'une tour.
- Le noctambule qui prolonge le mouvement du cavalier en ligne droite, dans notre code il a suffi de mettre le même code que pour le cavalier, à l'exception que ses mouvements étaient "récurifs", c'est-à-dire qu'il pouvait prolonger ses mouvements.

On a également implémenté le Mastondon, qui est pareil qu'un pion mais qui peut en plus également capturer devant lui.

On peut choisir les pièces qui seront présentes sur le plateau avec le fichier JSON de sauvegarde, comme indiqué dans la partie 3.3.

Le design des pièces classiques sont celles prises à l'adresse <https://github.com/lichess-org/lila/tree/master/public/piece/pixel>

3.5 Règles

Les règles de base sont celles définies sur Lichess et chess.com pour la version classique, tandis que pour la version féérique, les pièces indiquées ci-dessus ont été ajoutées.

On peut restreindre les règles avec le fichier JSON de sauvegarde, comme indiqué dans la partie 3.3.

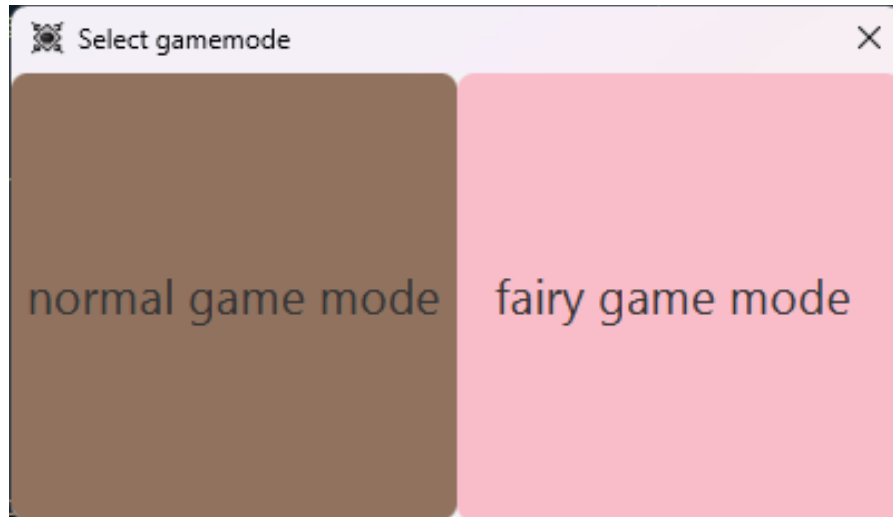
4 Conclusion

Pour faire ce projet, nous avons eu à faire une utilisation poussée de Java, s'intéresser au design également des pièces, du plateau des menus... On a dû faire nos propres designs pour les pièces féériques vu qu'on a pris les designs pixel de pièces sur le GitHub de lichess, on a modifié ces designs pour faire les pièces féériques ainsi que le logo de l'application.

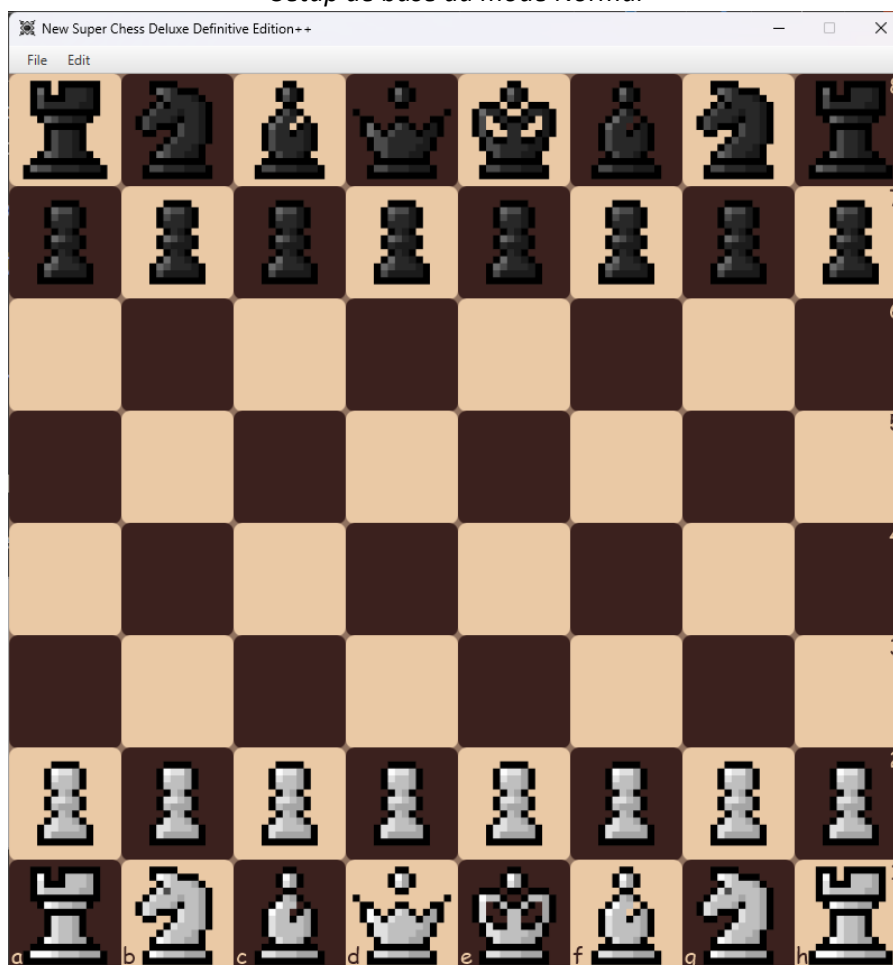
5 Annexes

Images de **New Super Chess Deluxe Definitive Edition++**

Menu de sélection du mode de jeu



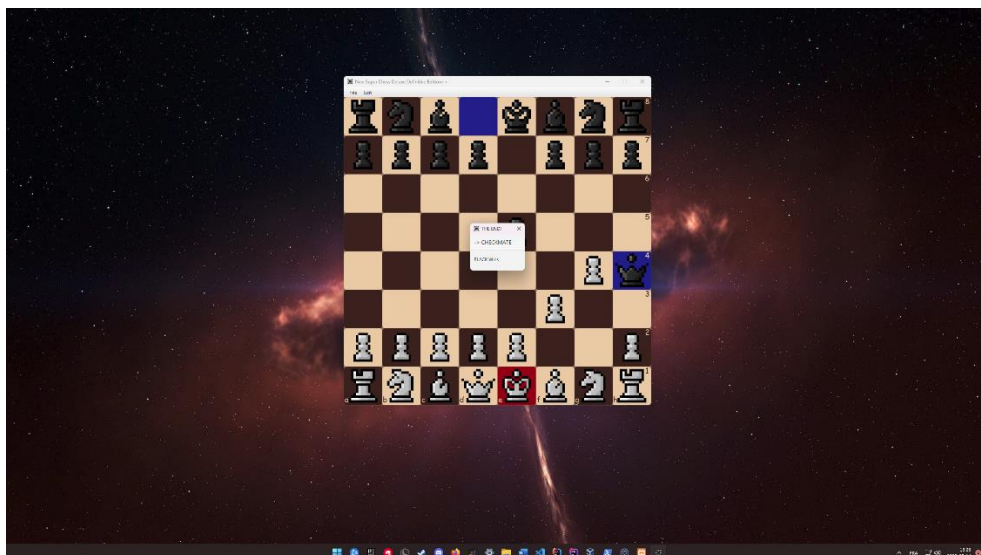
Setup de base du mode Normal



Setup de base du mode Fairy



Échec et mat



Graphique UML

