PT 2.1 APL 2019/2020 :

Jeu du Sudoku

Lucas Bruton
Gabriel Chavanon

SOMMAIRE

Lucas Bruton
Gabriel Chavanon

I/ Introduction au sujet :

Le but de notre projet est de proposer une interface conviviale pour concevoir ou résoudre des grilles de Sudoku.

Ce jeu de réflexion demande de remplir une grille en respectant une configuration de départ différente à chaque partie et quelques règles simples.

Notre projet se décomposera en deux programmes ayant les buts suivants:

• Le premier programme servira à l'élaboration des grilles de départ:

Pour construire une grille, il sera possible de partir d'une grille vide ou d'en charger une existante depuis un fichier. Il est alors possible d'ajouter ou d'enlever des numéros dans la grille, bien sûr, le programme doit empêcher les placements de valeurs fausses. Quand la grille est terminée, elle sera sauvegardée dans un fichier en .gri .

• Le second programme servira à résoudre une grille:

On commencera par charger une grille depuis un fichier grâce à un JFileChooser. Puis on devra choisir si on veut résoudre la grille manuellement ou automatiquement. En mode automatique, le programme affichera la grille résolue et le temps que le programme à mit pour résoudre la grille.

En mode manuel, le joueur pourra ajouter des chiffres et en enlever à sa guise, sauf si ces chiffres sont contradictoires. En cas de doute, il pourra faire rentrer jusqu'à quatre chiffres dans une même case. Le joueur sera félicité par le programme quand toutes les cases contiendront un seul chiffre.

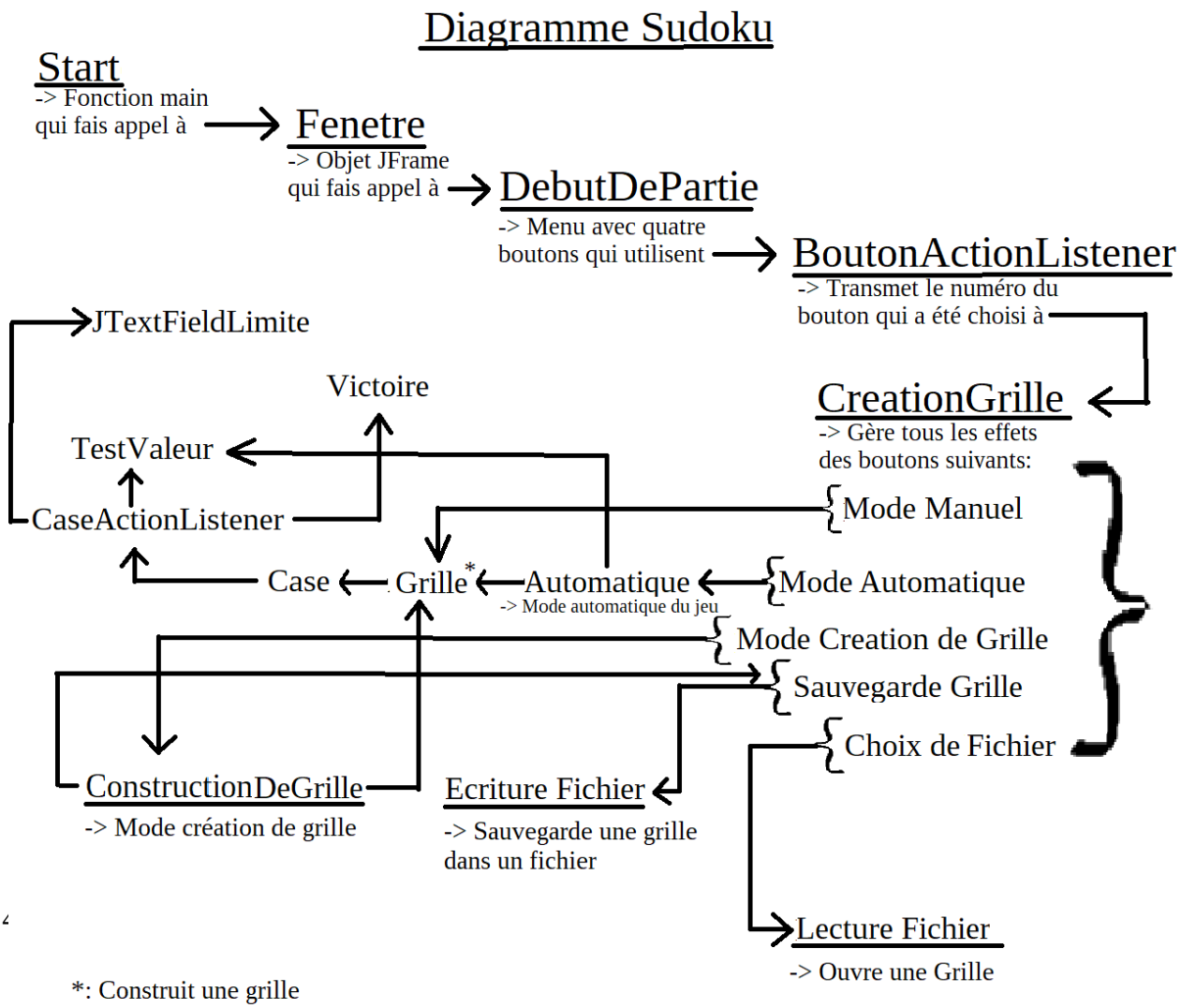
Lucas Bruton
Gabriel Chavanon

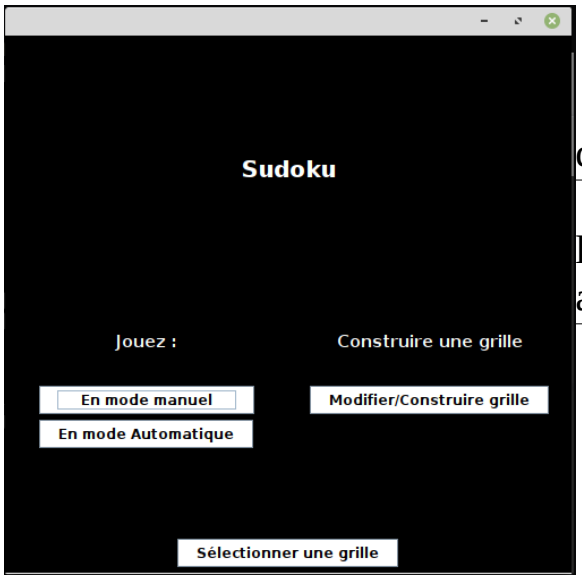
II/ La structure du programme

Le programme est composé de plusieurs fonctions qui vont s'appeler entre elles.

Tout d'abord, quand le programme est lancé, la page du menu s'affiche, en cliquant avec la souris, la page se ferme et la grill est construite, la partie commence immédiatement, si on appuie sur le bouton en mode manuel ou automatique, si on appuie sur le bouton «Mode Création de Grille», alors on peut construire une grille que l'on pourra sauvegarder dans un fichier pour ensuite l'utiliser lors d'une partie. On peut aussi modifier les grilles créées via ce même bouton. Le bouton «Choix Gril» Le programme peut être donc être quitté de nouveau par un clic de la souris.

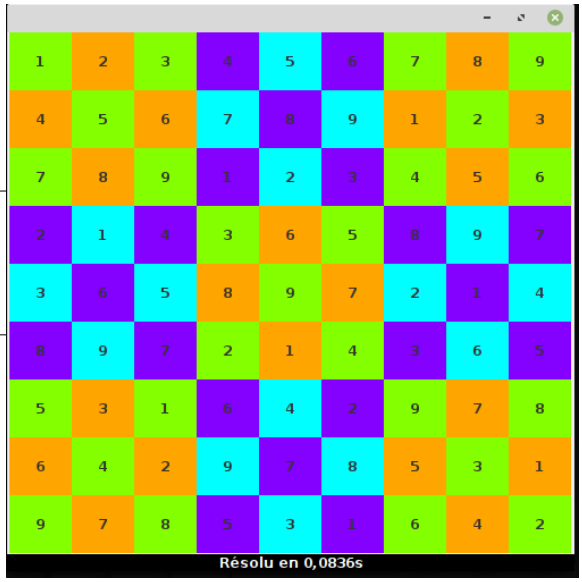
Comme montré dans le diagramme ci-dessous, le main va d'abord appeler la fenêtre de menu, l'utilisateur a ensuite le choix entre plusieurs boutons.
Chaque bouton a un usage propre à lui seul.
À la fin de l'utilisation de la fonction d'un bouton ou à la fin d'une partie, on peut fermer le programme et le relancer pour rejouer.

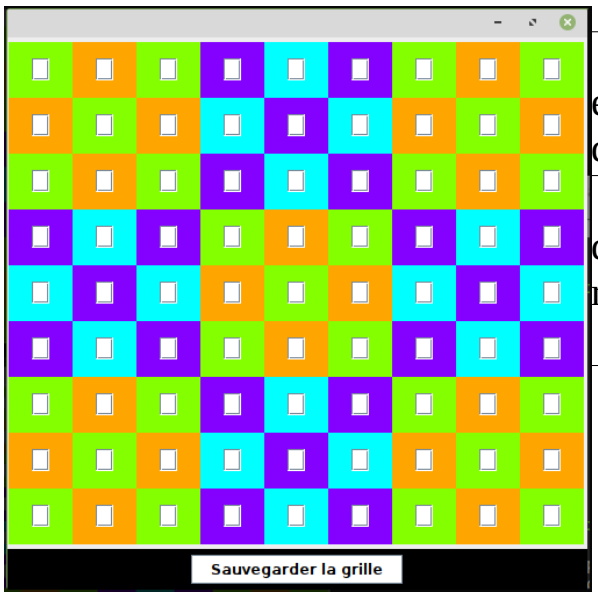




← Fenêtre de menu du programme

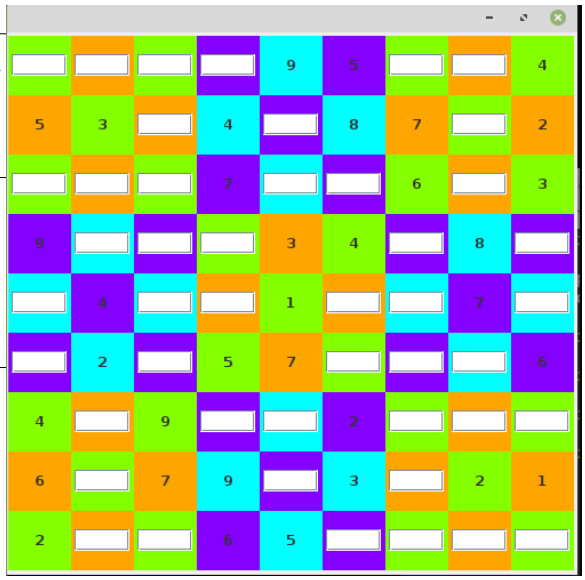
Grille résolue par l'algo en mode automatique →

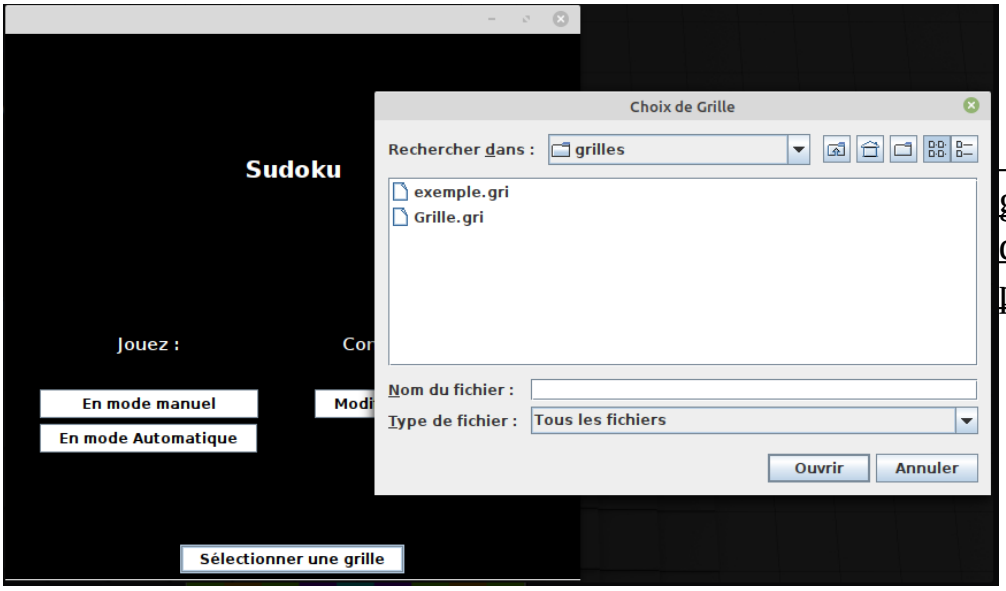




← Mode de création et de modification des grilles

Grille en résolution dans le mode de résolution manuel →





← Sélection d'une grille pour les différents boutons du programme

Lucas Bruton
Gabriel Chavanon

III/ Explication du fontionnement de l’algorithme du mode automatique de résolution du soduku:

Le fonctionnement de la résolution automatique et de l’algorithme permettant de résoudre les grilles est le suivant:

Le programme va tout d’abord récupérer la grille créée au préalable par l’utilisateur via le bouton de sélection de grille, ensuite, il récupère les valeurs dans les cases de bases.
Avec la méthode de création grille, elle modifie les cases pour compléter les grilles. Elle appelle une méthode qui utilise l’algorithme appelé «resoudreSudoku» pour le faire. L’algorithme quant à lui fonctionne comme ceci:

Il utilise un technique, le BackTracking, grossièrement il va retourner le tableau, affecter ce nouveau tableau à la grille et remplacer les anciennes valeurs par les nouvelles et va compter que c’est la méthode algo qui a réussi le sudoku.

Pour être plus précis au sujet du fonctionnement de l’algorithme:

Une méthode récupère la grille fournie par la classe automatique, cette méthode va affecter une variable pour le temps de départ de l’algorithme avant de commencer. Ensuite, l’algo va au tout début de la grille, il va l’analyser jusqu’à ce qu’il trouve une case vide, il va commencer un compteur à 1, et il va regarder si cette valeur est possible avec un objet de la classe TestValeur, tant qu’il ne trouve pas de valeur qui est contradictoire, il va incrémenter son compteur jusqu’à trouver une valeur qui est concordante, si le premier compteur de la première case va au-delà de 9, le sudoku ne peut pas être résolu. Après avoir complété la grille , il va trouver une autre case pour recommencer le processus et il va continuer à parcourir chaque case jusqu’à ce que toutes les cases vides soient pleines, quand l’algorithme s’occupe d’une autre case que celle de départ, si la valeur de celle-ci va au-delà de 9, l’algo va revenir à la case précédente au lieu de comprendre que le sudoku est impossible. Il ne va pas commencer son compteur à 1 mais à partir du chiffre qu’il y avais avant dans cette même case au part avant, si la valeur de cette case va au delà de 9, il retourne à la case précédente et ainsi de suite. Quand l’algorithme touche à sa fin, il va affecter une valeur au temps de fin de celui-ci et va par une soustraction à la valeur du début, calculer le temps de résolution de la grille.

Lucas Bruton
Gabriel Chavanon

IV/Conclusions personnelles

Conclusion personnelle (Gabriel Chavanon)

Conclusion :

J’ai trouvé que ce projet était assez compliqué de part les différents aspects qui nous ont été donnés de réaliser. Je pense en particulier à la création des grilles qui a été un vrai calvaire au début du projet. Je m’explique, j’ai eu beaucoup de mal à imaginer comment faire en sorte que la grille et ses différentes régions soient connectées et affichées à l’écran. J’ai essayé pendant une bonne partie de la première semaine de trouver un solution à ce problème, en vain ce qui m’a poussé à travailler en équipe et à demander de l’aide à mon binôme qui a su m’expliquer son raisonnement au sujet des différentes étapes du projet et se méthodes. J’ai donc fait tout mon possible pour ne pas me décourager à cause des différents problèmes rencontrés tout au long de ces trois semaines de travail. Cependant, ces difficultés m’ont apporté une nouvelle compréhension de la programmation en java et des possibilités que celles-ci peuvent permettre au programmeur. Je me suis donc efforcé de comprendre la logique derrière les programmes de Lucas pour ne pas avoir deux visions différentes de notre projet. J’ai d’ailleurs essayé d’aider mon binôme comme j’ai pu quand bien même mes compétences en programmation ne soient pas forcément optimales. J’en conclus donc que ce projet a été une bonne expérience puisque le sudoku m’a apporté un regard nouveau sur le java. Je me suis d’ailleurs donné pour objectif de refaire ce projet par moi même dans le futur afin de réussir là où j’ai échoué lors de ce projet.

Je remercie encore une fois mon binôme pour ses explications en java.

Conclusion personnelle (Lucas Bruton)

Conclusion :
Coder la quasi-totalité du sudoku m'a beaucoup appris dues aux nombreuses difficultés rencontrées. La première difficulté que j'ai croisée était de réfléchir sur comment coder les différentes parties du sujet. En plus de lier le côté graphique au reste du projet, il fallait chercher comment lier les différentes classes afin d'avoir le maximum de code qui soit commun entre les deux programmes demandés. La principale difficulté que j'ai rencontrée dans la partie graphique du sujet était de positionner les différents éléments graphiques aux positions que j'ai souhaité. Cette difficulté a été la plus pénible à surmonter dans le fichier DebutDePartie.java puisque c'est ce fichier qui comporte le plus d'éléments graphiques différents. Dans le reste du projet, j'ai croisé deux difficultés majeures. La première était de modifier la grille qui serait charger après qu'un fichier soit sélectionné par le joueur. Après avoir pensé à plusieurs solutions, la meilleure solution que j'ai trouvée était d'avoir tous les boutons que j'ai codé lier à un même objet. Cet objet est de la classe CreationGrille, comme expliqué dans le compte-rendu écrit par Gabriel, s'occupe de tous les effets de boutons. La deuxième difficulté que j'ai rencontrée était de trouver un algorithme qui résout les sudokus. Après quelques recherches sur Internet, je suis tombé sur un algorithme qui utilise la méthode du "Backtracking". Ce sera au final cette technique que j'utiliserai dans le fichier Automatique.java. Dues à ces difficultés rencontrées, j'ai trouvé que ce projet était instructif et amusant à coder.

Je vais finir sur ce que j'ai apprécié le plus et sur ce que j'ai le moins aimé faire dans le projet :
• La partie que j'ai préféré réaliser était les limitations qui fallait créer pour cases. C'était un challenge que j'ai pris plaisir à surmonter.

• La partie que j’ai le moins apprécié faire était ajouter les commentaires javadoc à chaque fichier que j’ai codé. J’ai trouvé que c’était très long et répétitif à faire.