

Rapport SAÉ Sudoku

SOMMAIRE :

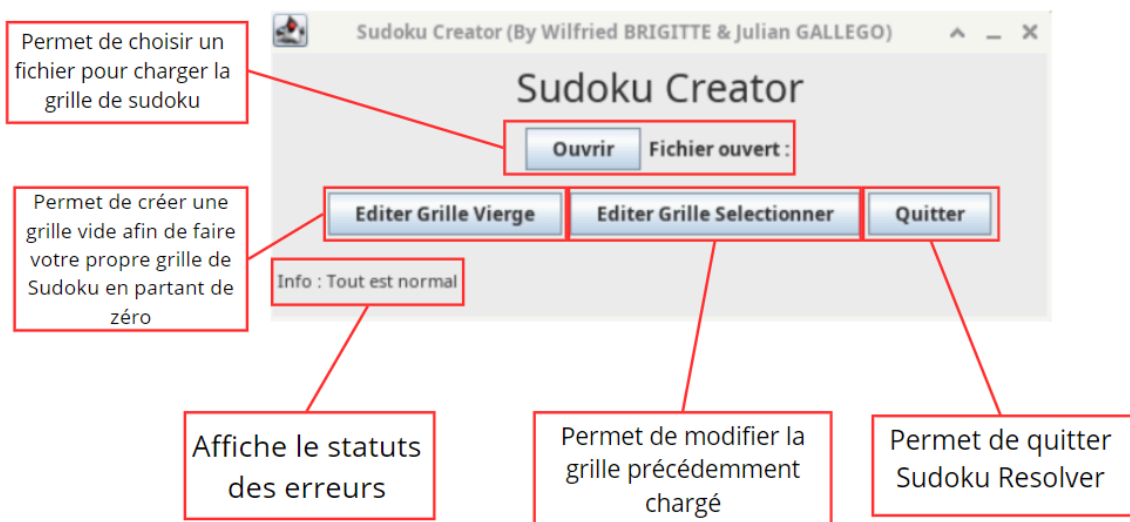
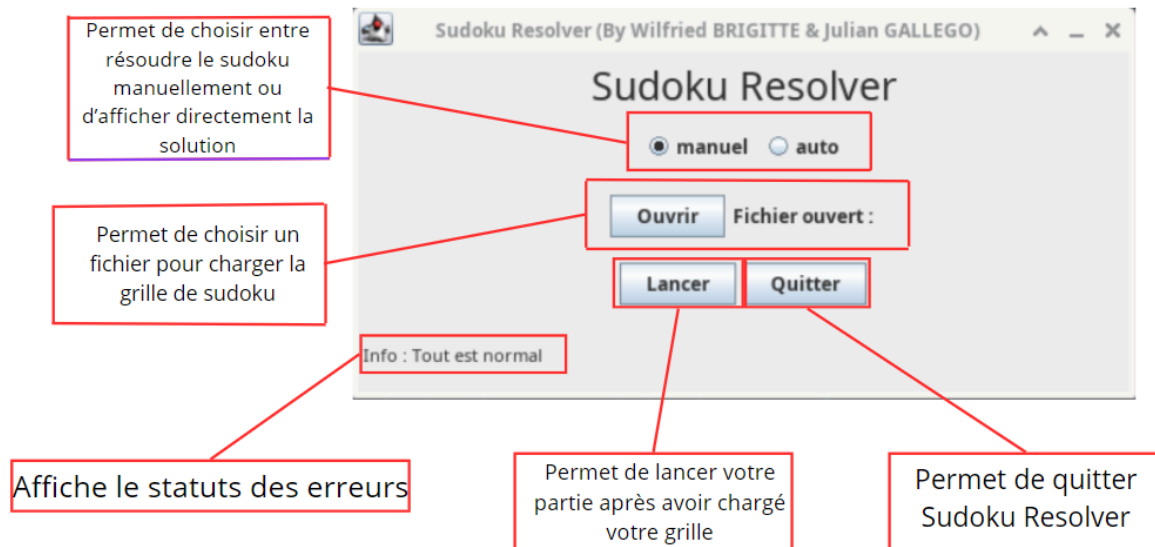
- 1) Introduction
- 2) Description des fonctionnalités du programme
- 3) Présentation de la structure de chaque programme
- 4) Explication de l'algorithme qui résout les grilles
- 5) Conclusions personnelles

Introduction:

Pour ce projet il nous a été demandé de développer une interface conviviale pour la création et la résolution de grilles de Sudoku (tout cela en Java). Deux programmes distincts seront réalisés, l'un pour la création de grilles de départ et l'autre pour leur résolution. Les fonctionnalités incluent la création et la modification de grilles, ainsi que leur sauvegarde dans des fichiers.

				9	5			4
5	3		4		8	7		2
			7			6		3
9				3	4		8	
	4			1			7	
	2		5	7				6
4		9			2			
6		7	9		3		2	1
2			6	5				

Description des fonctionnalités du programme :



Structure de chaque programme :

Pour ce projet, nous avons choisi de structurer le code en plusieurs fichiers source. Il y a les fichiers classiques tels que "mainProgramme1" et "mainProgramme2", servant de points de départ pour les deux programmes respectifs. Ensuite, nous avons les fichiers qui gèrent les menus de chaque programme, à savoir "menuProgramme1" et "menuProgramme2". Nous avons également la classe "JeuFini", qui est activée lorsque la grille de Sudoku est résolue et permet de féliciter le joueur. De plus, la classe "JTextFieldCharLimit" a été mise en place pour limiter les caractères pouvant être placés dans la grille. Une classe dédiée a été créée spécialement pour l'algorithme résolvant les grilles, mais celle-ci est également utilisée pour valider les chiffres dans la grille de Sudoku. Enfin, la classe la plus importante du programme est "Grille", qui gère tous les aspects des grilles de jeu, tels que l'affichage, l'exportation ou le chargement des grilles, par exemple.

Diagramme Programme 1 (Editeur de Grilles) :

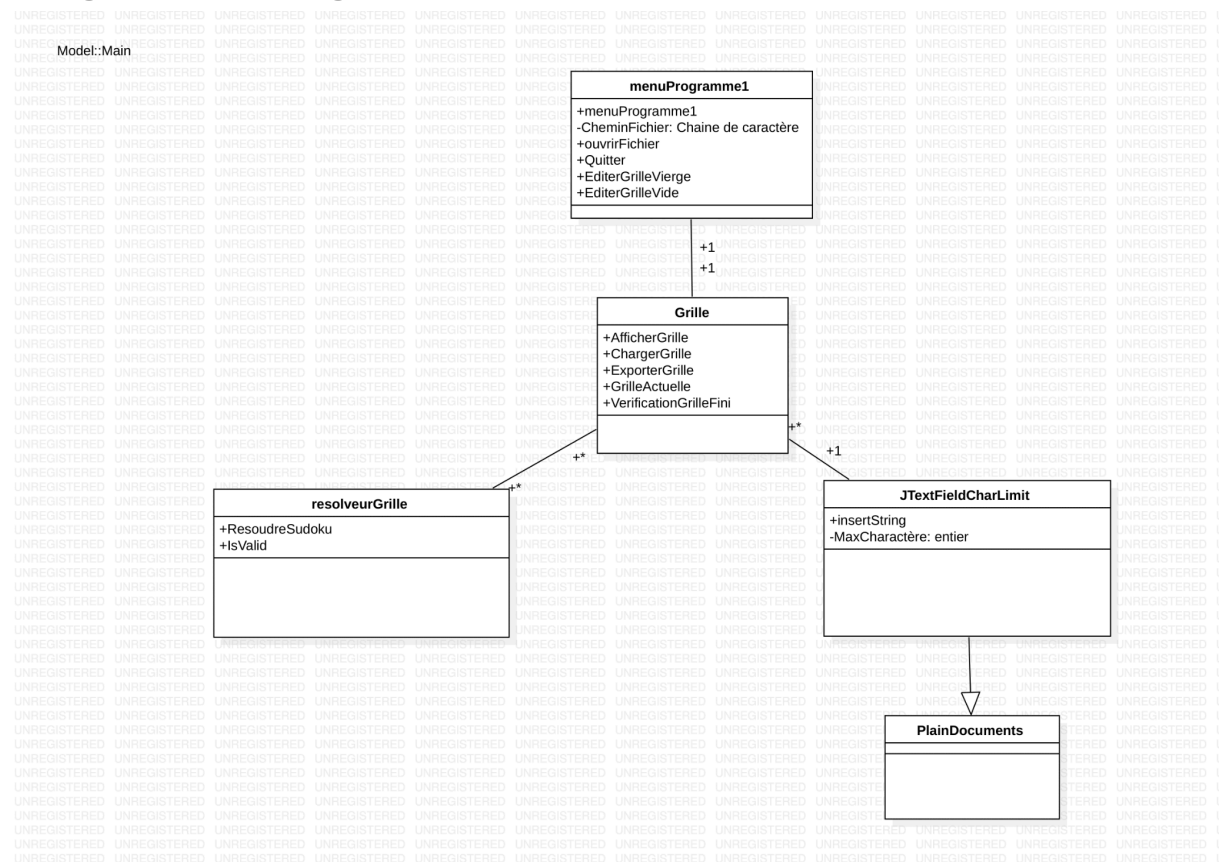
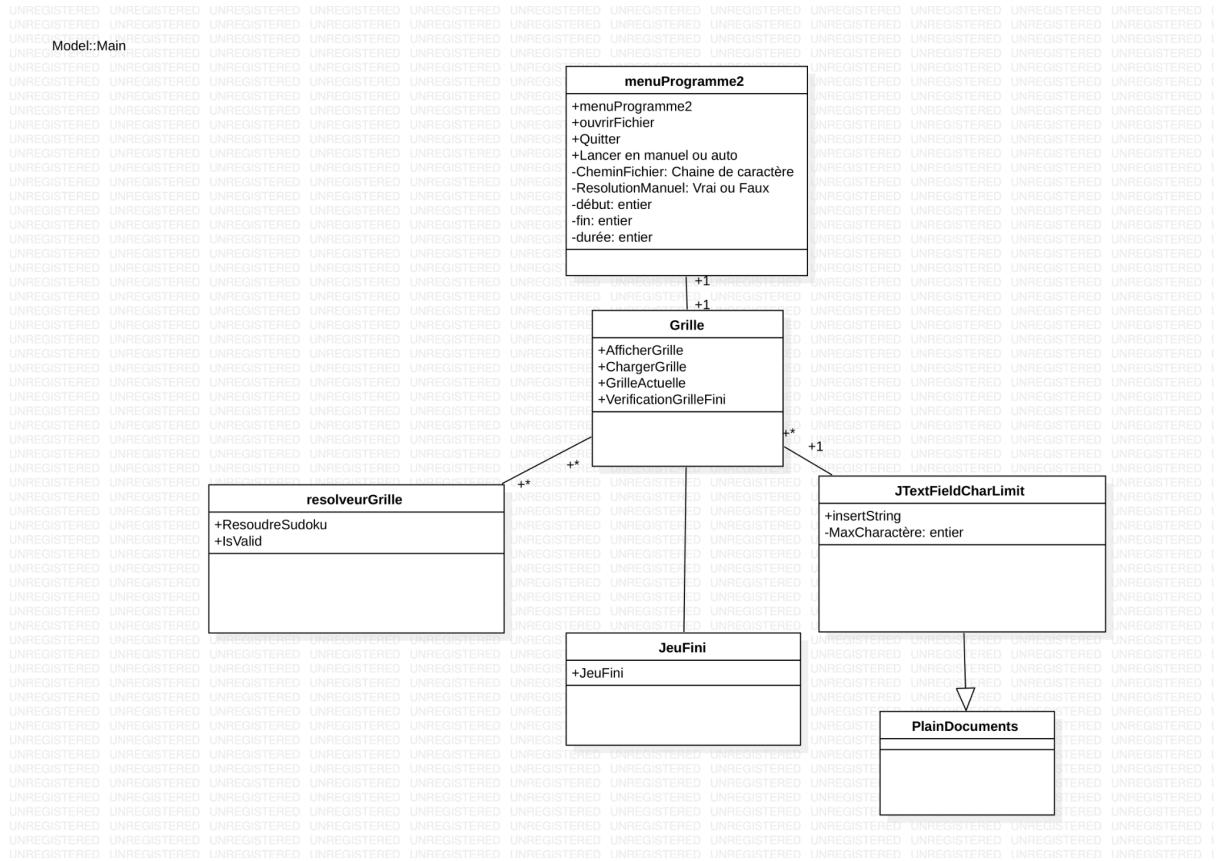


Diagramme Programme 2 (Résolveur de Grilles) :



Explication Algorithme :

L'algorithme commence par initialiser une grille de solution vide, représentée par une matrice de 9x9 (un tableau d'entiers de 9 par 9 en java). Ensuite, il explore la grille de Sudoku, case par case, de manière récursive. Lorsqu'il rencontre une case vide, il essaie les chiffres de 1 à 9 pour cette case.

Pour chaque chiffre essayé, l'algorithme vérifie s'il est valide dans la case en cours selon les règles du Sudoku. Il vérifie que le chiffre n'apparaît pas déjà dans la même ligne, colonne ou région.

Si un chiffre est valide pour une case donnée, l'algorithme met à jour la grille et passe à la case suivante, en appelant récursivement la fonction sur la nouvelle grille.

Si aucun chiffre valide n'est trouvé pour une case donnée, l'algorithme fait un retour en arrière pour essayer une autre valeur pour la case précédente. Il continue ce processus jusqu'à ce qu'une solution valide soit trouvée pour la grille entière.

L'algorithme utilise une approche qui teste toutes les combinaisons possibles de chiffres jusqu'à trouver une solution valide. Lorsqu'une solution est trouvée, elle est copiée dans une autre grille qui est retournée en tant que solution finale.

Conclusions Personnelles :

Wilfried :

Personnellement, j'ai beaucoup apprécié travailler sur ce projet. L'une des principales raisons est la possibilité de créer quelque chose de concret et plus approfondi que les exercices pratiques habituels. De plus cela m'a permis de consolider et approfondir certains points dans le développement en Java.

Julian :

Ce projet de deuxième semestre à été très enrichissant pour moi car j'ai pu mettre en place mon savoir a un but concret et aussi à pouvoir comprendre certains concepts que je n'avais pas totalement compris comme les événements. D'autre part, j'ai préféré faire ce projet plutôt que celui du premier semestre car bien qu'ils soient différents je trouve que le java permet de faire les choses plus simplement.