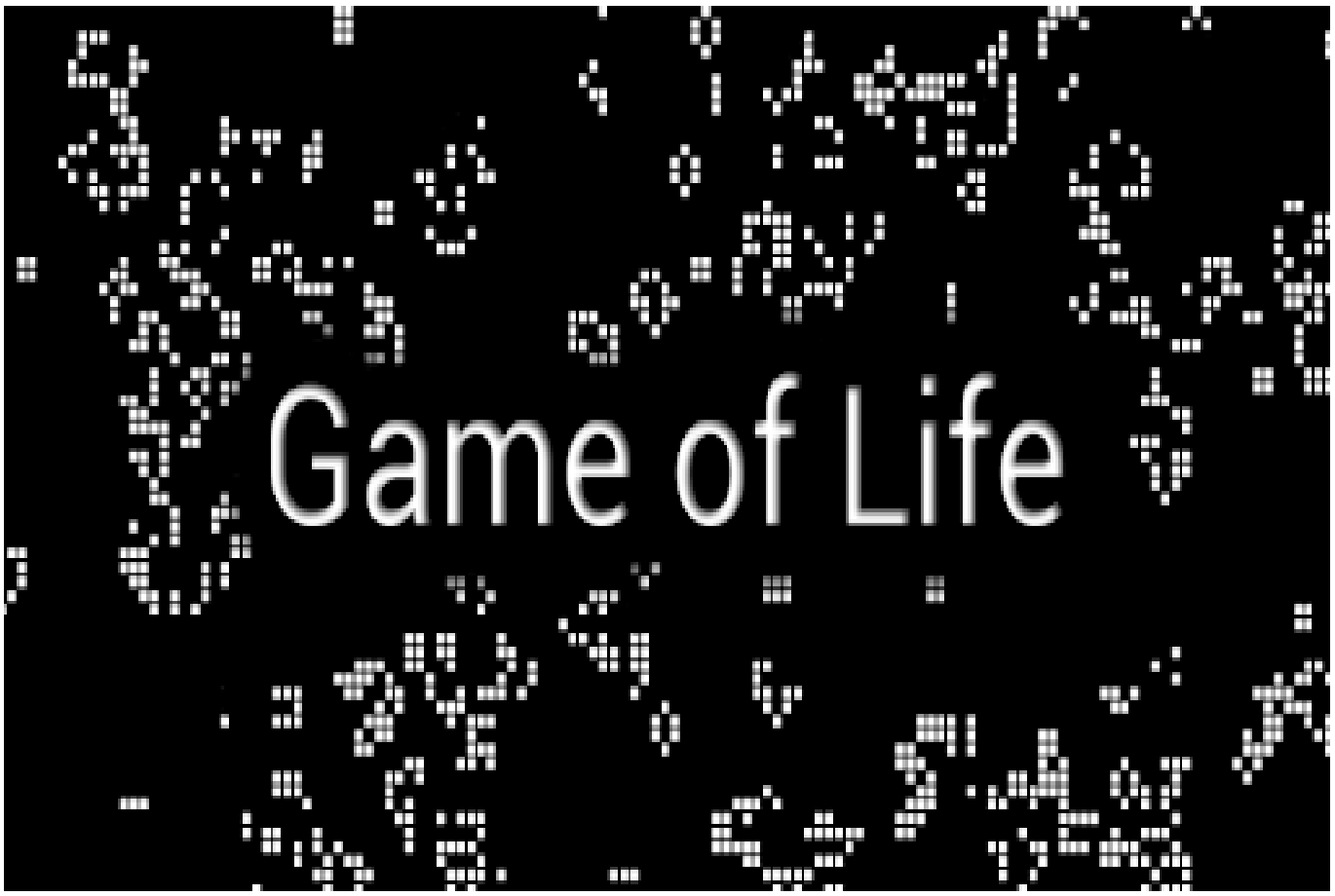
Rapport de Projet



Auteurs :
Baptiste Saint-Léger, Valdrin Salihi

Année :
2021 - 2022

Table des matières

I - Structure du programme :

Pour cette application, nous nous sommes appuyés sur le modèle MVC que nous avons vu dans le cours durant le semestre. Nous avons donc séparé en plusieurs sous-packages les fichiers de notre application. Tout d'abord, nous avons créé un package initial du nom de "jeudelavie" comportant à sa racine un fichier "Main.java" permettant de lancer notre application. Ainsi que quatre autres répertoires respectivement, "jeudelavie.controleur", "jeudelavie.image", "jeudelavie.model" et "jeudelavie.vue".

À travers ces différents répertoires, nous y avons mis chacun nos fichiers afin de respecter le modèle MVC exigé par le sujet. Pour réaliser le développement de l'application nous nous sommes mis d'accord pour utiliser la version 8.2 de NetBeans et le jdk 1.8, ainsi qu'un répertoire Git pour pouvoir centraliser tout le code. En plus de cela, nous avons pris la décision d'utiliser du fxml et donc SceneBuilder pour concevoir la partie graphique de l'application.

- Controleur :

Dans le package controleur, nous avons mis les deux controleurs qui géraient l'application dans son ensemble ; Le premier étant le "ControleurBasique.java" qui sert de squelette à l'application. Il gérera les différentes propriétés dédiées à une barre de menu. Si nous avons dissocié ce premier controleur du suivant, c'est de pouvoir faire en sorte que l'on puisse conserver un menu peu importe où l'on se trouverait dans l'application si de différents controleurs il devait y avoir. Le menu que va gérer ce controleur contient différentes fonctionnalités permettant un confort de navigation pour l'utilisateur. L'on peut à partir de la barre de menu, quitter l'application, agrandir/réduire l'écran avec un redimensionnement qui s'adaptera à l'option choisie. Il aura la possibilité d'avoir des informations sur l'application ainsi qu'un tutoriel d'utilisation du jeu.

- Controleur :

Quant au second controleur du nom de “ControleurApplication.java” il est le cœur principal du fonctionnement de notre application. Il va s’occuper d’initialiser et de gérer les composants nécessaires des modèles et des vues au jeu de la vie.

Au sein de ce controleur, nous créons le plateau de jeu ainsi que la zone tampon qui sont tous deux des GridPanes composés de label.

On appellera aussi dans cette classe le service nous permettant d’effectuer chacun des tours du jeu. Comme dit précédemment, il va initialiser tous les composants gravitant autour de l’application, ainsi qu'à attribuer des méthodes nécessaires pour les nœuds qui doivent en avoir.

Comme par exemple le comportement que devra avoir le plateau de jeu lorsque l’on fait un clique gauche dessus ou bien scroller dessus pour zoomer / dézoomer etc..

- Image :

Le répertoire “jeudelavie.image” va simplement servir à stocker les différentes images du jeu sous leur différent format afin de centraliser la localisation de ces dernières pour une accession de ces ressources plus fluides.

- Model :

Le package “jeudelavie.model” lui va essentiellement contenir la classe Game qui va établir le fonctionnement du jeu de la vie avec les différentes dont il aura besoin. Le modèle de la grille va contenir deux tableaux de booléens à double entrée pour simuler une le plateau. Le premier tableau de booléens contient les positions de chaque cellule à un instant T et le second tableau permet de contenir temporairement ces positions et ainsi d’appliquer les modifications nécessaires du modèle et de coller ce nouveau tableau dans le premier pour enfin passer au tour suivant.

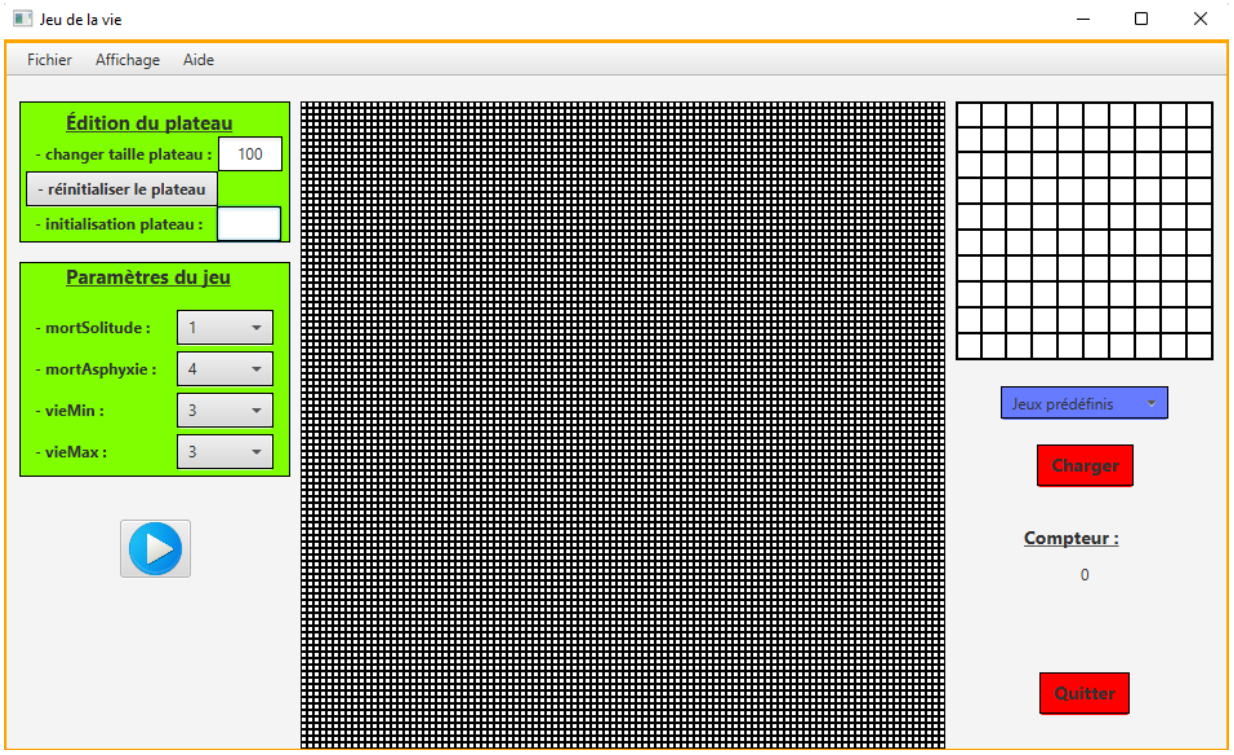
- Model :

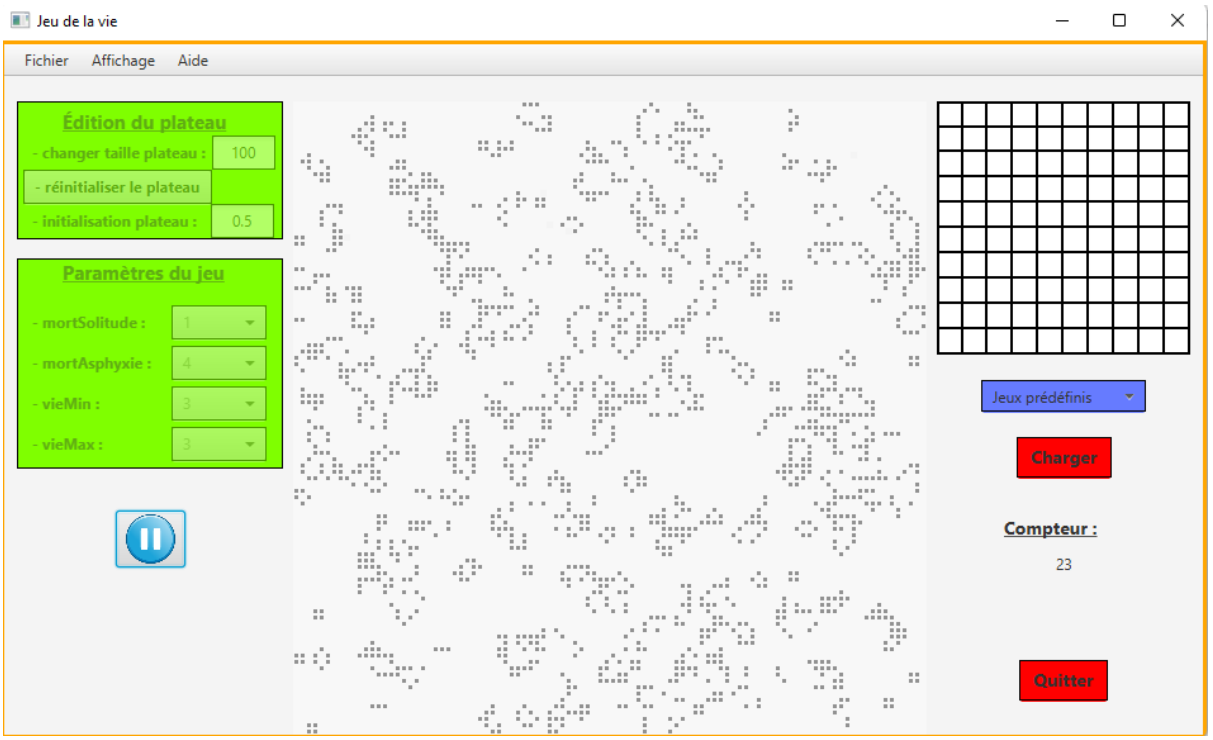
Le répertoire modèle va aussi contenir deux autres classes intéressantes : “DefaultTampon.java” et “Tutoriel.java”. La première classe va permettre de contenir en dur tous les modèles de tampons que l’on aura créés pour que l’utilisateur puisse les utiliser directement dans la zone tampon.
La classe Tutoriel, comme son nom l'indique, va servir de modèle héritant d’un stage pour une interface de tutoriel que l’utilisateur pourra voir en cliquant sur l’item dans le menu.

- Vue :

Au niveau du répertoire de la vue, on va contenir trois vues, une faite en java “View.java” servant à dessiner la grille du plateau du jeu. Et deux autres vues faites en fxml gérant le visuel de l’application dans son ensemble, la première “VueBasique.fxml” s’occupe de la barre menu et la seconde “VueApplication.fxml du plateau de jeu.

Visuel de l’application





II - Résolution de problèmes :

Durant ce projet, nous avons rencontré diverses formes de problèmes concernant la conception du Jeu de la Vie. En effet, dans un premier temps, nous avons dû concevoir la partie fonctionnelle du jeu. Le premier problème rencontré était celui de la taille du tableau à deux dimensions. Dans la fonction pour mettre à jour le jeu à un temps T + 1, nous faisons appel à une fonction qui va compter le nombre de cellules vivantes autour d'une position.

Cependant, toutes les cases se trouvant à l'extrémité du tableau provoquaient une erreur d'indexation. Par exemple : la case se trouvant en position (0;0) ne peut pas récupérer la valeur de la case (-1;-1). Pour corriger cette erreur, nous avons décidé que chaque dimension serait égale à la dimension voulue + 2.

Nous nous sommes ensuite occupés de la partie visuelle du projet. Nous avons décidé d'utiliser un GridPane de Label pour représenter le plateau de jeu. Le problème rencontré ici était la taille des labels. Nous n'arrivions pas à définir la taille des Labels à 5 pixels. Pour résoudre ce problème nous avons redéfini, pour chaque Label, la taille du Min, Max et Pref Size à 5 pixels car sinon la hauteur des Labels ne descendait pas en dessous d'environ 10 pixels.

```
Exception in thread "JavaFX Application Thread" java.util.ConcurrentModificationException
        at java.util.HashMap$HashIterator.nextNode(HashMap.java:1437)
        at java.util.HashMap$EntryIterator.next(HashMap.java:1471)
        at java.util.HashMap$EntryIterator.next(HashMap.java:1469)
        at java.util.AbstractCollection.addAll(AbstractCollection.java:343)
        at java.util.HashSet.<init>(HashSet.java:119)
        at javafx.scene.CssStyleHelper.resetToInitialValues(CssStyleHelper.java:441)
        at javafx.scene.CssStyleHelper.createStyleHelper(CssStyleHelper.java:180)
        at javafx.scene.Node.reapplyCss(Node.java:8985)
        at javafx.scene.Node.reapplyCss(Node.java:9014)
        at javafx.scene.Node.impl_processCSS(Node.java:9182)
        at javafx.scene.Parent.impl_processCSS(Parent.java:1249)
        at javafx.scene.control.Control.impl_processCSS(Control.java:868)
        at javafx.scene.Node.processCSS(Node.java:9058)
        at javafx.scene.Node.processCSS(Node.java:9051)
        at javafx.scene.Node.processCSS(Node.java:9051)
        at javafx.scene.Node.processCSS(Node.java:9051)
        at javafx.scene.Node.processCSS(Node.java:9051)
        at javafx.scene.Node.processCSS(Node.java:9051)
        at javafx.scene.Scene.doCSSPass(Scene.java:545)
        at javafx.scene.Scene.access$3600(Scene.java:159)
        at javafx.scene.Scene$ScenePulseListener.pulse(Scene.java:2392)
        at com.sun.javafx.tk.Toolkit.lambda$runPulse$30(Toolkit.java:355)
        at java.security.AccessController.doPrivileged(Native Method)
        at com.sun.javafx.tk.Toolkit.runPulse(Toolkit.java:354)
        at com.sun.javafx.tk.Toolkit.firePulse(Toolkit.java:381)
        at com.sun.javafx.tk.quantum.QuantumToolkit.pulse(QuantumToolkit.java:510)
        at com.sun.javafx.tk.quantum.QuantumToolkit.pulse(QuantumToolkit.java:490)
        at com.sun.javafx.tk.quantum.QuantumToolkit.lambda$runToolkit$404(QuantumToolkit.java:319)
        at com.sun.glass.ui.InvokeLaterDispatcher$Future.run(InvokeLaterDispatcher.java:95)
        at com.sun.glass.ui.win.WinApplication._runLoop(Native Method)
        at com.sun.glass.ui.win.WinApplication.lambda$null$148(WinApplication.java:191)
        at java.lang.Thread.run(Thread.java:745)
```

Enfin, notre plus gros problème est un problème de Thread. Nous n'avons pas compris exactement l'exception qui se levait, nous pensions dans un premier temps à la taille du tableau qui demandait trop de ressources de calcul, car en effet lorsque que l'on diminue la taille du plateau l'exception se manifeste moins souvent.

Mais dans un deuxième temps nous nous sommes dit que ce n'était pas la solution, car elle apparaissait toujours, mais que le fait que le plateau soit plus petit, le phénomène qui levait l'exception, avait moins de chance de se manifester. Nous pensons donc que c'est une disposition de jeu bien précise qui lève l'exception car elle ne se produit que certaines fois pendant un tour. Malheureusement, nous n'avons pas trouvé de solution.

III - L’avenir du projet

Comme vous avez pu le constater, nous avons fait en sorte que le jeu ressemble graphiquement au schéma du cahier des charges, mais nous pourrions améliorer la partie visuelle pour l'embellir et rendre l’application plus attractive.

Dans un premier temps, nous avions imaginé l’application avec un tableau qui modifierait sa taille en fonction de la taille de la fenêtre. Nous avons réussi dans une autre version de l'application à le faire à l’aide de binding.

Pour améliorer la zone tampon, nous pourrions rajouter un bouton “save” afin que l’utilisateur puisse sauvegarder ses propres modèles plutôt que de les redessiner à chaque fois.

Nous pourrions aussi ajuster la dimension de la zone tampon afin d’éviter de copier trop de cases sans cellules.

Pour le plateau, imaginons que l’utilisateur passe du temps à modéliser tout un tas de choses bien précises, mais qu’il ne veut pas perdre ce qu'il a fait, alors la solution serait de proposer une sauvegarde du plateau et de pouvoir changer ces sauvegardes.

Conclusion :

Pour conclure ce projet, nous vous délivrons une application fonctionnelle avec un visuel conforme au cahier des charges. La partie Model correspond aux règles du jeu et peut être modifiée lorsque le jeu est en mode pause.

Nous avons implémenté une zone tampon permettant de dessiner ou de charger des modèles en même temps que le jeu est en cours pour ensuite les copier/coller sur le plateau de jeu.

Le jeu est améliorable, nous avions pensé à la mise en forme ou une forme de sauvegarde pour la zone tampon ou le plateau. Comme dis précédemment, il reste des problèmes à gérer, notamment avec les exceptions du Thread.

Ce projet nous a permis de nous rendre compte de la rigueur et du travail à apporter dans ce type d'application, nous avons pris malgré les erreurs de code beaucoup de plaisir à le produire. Pour finir, nous avons apprécié travailler ensemble. Ce fut un vrai travail d'équipe !

REMERCIEMENT