Rapport du projet de Bataille Navale 2020

Dans ce rapport je vais tout d'abord vous citer les fonctionnalités particulières qui n'étaient pas demandé et même les améliorations que j'ai pu effectuer sur les fonctionnalités demandées. Ensuite je décrirai point par point chaque classe et leurs fonctionnalités dans ce jeu tout en expliquant mes choix de conception.

Description des ajouts non demandés, de l'IA et des ajouts qui ont été voulu :

Pour commencer je ne sais pas si cela me pénalisera ou non mais lorsqu'une partie commence l'ordinateur place ses bateaux aléatoirement comme ce qui était demandé mais le jouer lui place ses bateaux sur sa grille. Pour faire ceci j'ai ajouté un MouseWheelEvent qui fait en sorte que lorsque le jour scroll la molette de sa souris vers le haut le prochain bateau sera placé verticalement et à l'inverse si le joueur scroll la molette vers le bas le prochain bateau sera placé verticalement. Lors de cette phase de placement de bateau le joueur ne peut pas agir sur la grille adverse puisqu'elle est désactivée. Pour le placement des bateaux lorsque vous appuyez sur une case pour le placer il faut imaginer que cette case sera l'avant du bateau et que par conséquent la suite du bateau sera en dessous ou sur la droite de cette case en fonction du sens choisi. Un fois tous les bateaux placés la grille du joueur n'est plus accessible pour lui-même et la grille de l'ordinateur elle est activé. Pour ce qui est de l'IA je n'ai pas pu le développé comme je pourrais et voudrais le faire par manque de temps. Elle est donc très basique et vise simplement une case du joueur aléatoire (qui n'a pas déjà été toucher). J'aurais aimé faire en sorte que l'IA conserve les cases adjacentes d'un bateau qu'elle touche afin de viser aléatoirement une de ces cases et ainsi de suite jusqu'à ce que le bateau soit couler comme le ferait un vrai joueur. Je le ferai sans doute de mon côté même lorsque le projet sera rendu. Enfin j'ai réalisé quelques petites modifications esthétiques tel que des couleurs de fond ou de texte, une icône de frame représentant un bateau ou encore des titres de frame modifier. Mais encore une fois avec plus de temps j'aurai aimé mettre des images d'océan, de bateau (avant et après avoir été touché) etc. Mais en essayant cela n'avait pas l'air simple et rapide à faire alors j'ai préféré rester sur les couleurs.

Fonctionnement du jeu :

Maintenant parlons de l'interface du jeu. Lorsque vous débuterez vous pourrez choisir entre charger une partie ou alors débuter une nouvelle partie. Pour charger une partie il faut évidemment qu'une sauvegarde existe, dans le cas contraire vous serez redirigée vers le menu principal. Comme expliqué précédemment la partie débute avec le placement de vos bateaux sur votre grille (ils sont indiqués en vert une fois placé). Sur la gauche de votre grille le sens de votre prochain bateau vous est indiqué. Une fois vos bateaux placés vous pourrez alors attaquer l'ordinateur. L'ordinateur jouera automatiquement après vous. Pour cela il vous suffit de cliquer sur une case adverse de votre choix avec la bombe de votre choix. Vous avez à disposition 4 bombes différentes sur votre gauche. Lorsqu'une bombe à usage unique est utilisée vous ne pouvez plus y accéder et la bombe sélectionnée passera par défaut sur la bombe classique. Lorsqu'une case est touchée vous ne pourrez plus l'attaquer. Lorsque vous ou l'ordinateur attaquera il sera indiqué « Raté » ou « Touché » en dessous de chaque grille. Lorsqu'un bateau sera coulé cela sera indiqué en dessous de chaque grille encore une fois

mais dans le second cadre. Lorsqu'un bateau allié ou ennemi est touché la case correspondante devient rouge. À tout moment de la partie vous pouvez réinitialiser la partie, sauvegarder la partie ou quitter sans sauvegarder la partie via les boutons à votre disposition. Lorsque l'un des joueurs gagne la partie, un message apparait. Et il vous est alors proposé soit de quitter le jeu soit de commencer une nouvelle partie.

Description et fonctionnement de chaque classe :

Jeu.java :

La classe Jeu est sans doute la classe la plus importe de ce projet alors je vais commencer par elle. Elle contient un JFrame Fin qui est le Frame qui s'ouvre en fin de partie. Le JLabel Winner lui contient la phrase qui va indiquer si le joueur à perdu ou gagner. Les JLabel infoJoueur et infoIA sont ceux qui vont afficher « raté » ou « coulé » en dessous de chaque grille. Les JLabel infoCoulerJoueur et infoCoulerIA sont ceux qui vont afficher le bateau coulé en dessous de chaque grille. Le String Sens lui contient l'information en temps réel de la position du bateau en fonction du mouvement de la molette. L'entier BombeJoueur lui contient l'information de quel bombe le joueur à selectionner (1.classique ,2.verticale, 3.horizontale, 4.croix). Les tableaux plateauJ et plateauIA représentent les grilles de chaque joueur avec des chiffres (0-case vide, 1-case adjacente d'un bateau, entre 2 et 6 ce sont les id de chaque bateau le 2 représentant le plus grand et le 6 le plus petit, 8-bateau touché, 9-case touché mais pas de bateau touché). Les tableaux CaseJoueur et CaseIA repésentent les grilles de chaque joueur mais avec les vraies cases. Les entiers nbBateauJoueur et nbBateauIA representes les nombres de bateau restant dans chaque camp afin de déterminer la victoire d'un joueur plus tard lorsqu'un des deux sera égal à 0. L'entier Joueur permet d'identifier le joueur qui joue (0 pour l'ordinateur et 1 pour le joueur). L'entier bat permet de déterminer quel bateau le joueur place. Le String victoire contient la phrase afficher en fin de partie. Le Booléen placementOK lui permettra de déterminer si le bateau peut se placer à l'endroit voulu par le Joueur. Les derniers éléments sont les bombes et les bateaux de chaque joueur.

La méthode createBateauJoueur est la méthode qui va gérer la création des bateaux tout en vérifiant si l'emplacement choisi est correct.

La méthode createBateauIA crée les bateaux de l'IA aléatoirement. Tant que le placement d'un bateau n'est pas correct la boucle while va générer un emplacement aléatoire jusqu'à ce que ce soit bon.

Les méthodes d'emplacement vide vérifient que le bateau ne sorte pas de la grille et que son placement ne soit pas adjacent ou superposer à un autre bateau grâce aux numéros de la grille. Tant que cette méthode ne retourne pas true le bateau ne pourra pas être placer.

Les méthodes de placement comme elles l'indiquent placent les bateaux sur la grille lorsque l'emplacement est validé et place des 1 tout autour du bateau pour délimiter sa zone adjacente.

La méthode jouer elle va permettre d'indicer la grille avec des 8 et des 9. Lorsque l'IA joue, elle choisira une bombe aléatoirement et un emplacement à attaquer aléatoirement. Les valeurs de la grille du jouer seront alors modifier en conséquence tout en prenant compte les

cases déjà touché ou encore les effets de chaque bombe sur une bordure de grille par exemple ou encore sur des cases déjà touché. Lorsque le joueur joue le principe est identique excepté pour la zone touchée et la bombe choisi sont les valeurs sont dans les paramètres et dans la variable BombeJoueur. Cette méthode gère donc en même temps les textes et les boutons de l'interface en fonctions des actions effectués par le joueur.

Les méthodes toucher vont gérer les textes et les nombre de bateaux quand elles seront appelées dans la méthode jouer lorsqu'un bateau est touché (lorsqu'un indice 8 de la grille est visée).

Enfin la méthode CheckVictory elle renvoie un Booléen qui permettra de savoir quand la partie est terminé grâce aux nombres de bateaux.

Bateau.java :

La classe Bateau génère un bateau avec un id (entre 2 et 6 comme expliqué précédemment), une taille, l'emplacement de sa première case avec ligne et colonne et enfin un Booléen qui permet de savoir sa direction (true=verticale et false=horizontale).

BombesIA.java :

La classe BombesIA représente les bombes spéciales de l'ordinateur, elles sont initialisées à true pour signifier qu'elles ne sont pas encore utilisées et passé à false dans la méthode jouer de la classe Jeu lorsqu'elles sont utilisées.

BombesJ.java :

La classe BombesJ à tout d'abord des caractéristiques identiques à la classe BombesIA, elle possède en plus les boutons représentants les bombes du joueur sur le plateau.

CaseDeIA.java :

La classe CaseDeIA représente une case de la grille de l'ordinateur comme son nom l'indique. Durant la partie lorsque j'attaque une case de l'IA une action se réalise et déclenche la méthode jouer de la classe jeu et permet ensuite à l'ordinateur de jouer aussi en appelant encore une fois la méthode jouer. Cette action sur le bouton permet aussi de gérer la victoire du joueur ou de l'ordinateur (CheckVictory), elle gère aussi la couleur et l'accessibilité de la case lorsqu'un bateau est touché ou encore le fait de placé une petite croix lorsque c'est raté et ceux sur les grilles de chaque joueur.

CaseDuJoueur.java :

La classe CaseDuJoueur représente une case de la grille du joueur comme son nom l'indique. Sa principale fonction est de placé les bateaux du joueur (grâce à bat qui est incrémenté à chaque fois qu'un bateau est placé) sur la grille lorsque celui-ci clique sur une case afin de placer l'un de ces bateaux. Cette classe gère aussi l'affichage qui indique au joueur quel bateau placé grâce à la méthode changePlacement.

BatailleNavale.java :

Enfin la classe BatailleNavale elle contient le main du programme et c'est cette classe qui va gérer l'initialisation des grilles, les couleurs, les actions des boutons, les action de la molette, l'apparition et la disparition de certains Frame, bref c'est cette classe qui gère l'interface de la Bataille Navale. Il ne me semble pas nécessaire de la détaillé car c'est beaucoup d'incrémentation et de création Composant. Le plus important sera détaillé dans la partie conception du programme.

Choix de conception du programme :

Ce programme a été conçu en s'appuyant beaucoup sur le codage c++ avec lequel je suis à l'aise. C'est pourquoi mon programme contient beaucoup de beaucoup de boucles for, while, if else etc. C'est donc pour cela que précédemment j'ai essayé d'expliquer le plus de chose possible afin de vous faciliter la compréhension de ce programme car je sais que pour quelqu'un d'extérieur à la conception de ce programme la lecture peut vite être compliqué et on peut rapidement s'y perdre.

Dans ce programme tous les composants qui ne sont pas utilisés dans les autres classes sont en privés afin de les protégés un maximum. Pour les autres, ils sont tous en protégé afin d'évité une confusion avec les autres projets.

Toutes les classes implémentes la sérialisation afin que le programme se déroule sans soucis.

Pour l'interface j'ai utilisé beaucoup de GridBagLayout qui permettent de placé chaque élément ou l'on souhaite grâce aux coordonnés x et y. De plus j'ai utilisé plusieurs panels à l'intérieur de ce Layout pour séparés chaque élément qui sont tous des GridLayout. Les GridLayout sont les plus simple à manipuler que ce soit pour la mise en forme comme pour les couleurs de fonds.

La classe BatailleNavale est souvent répétitive car elle regroupe toutes les informations nécessaires à l'affichage de la partie. C'est beaucoup de composants et d'actions similaire.

Pour les boutons des bombes j'ai choisi de crée un groupe de bouton pour faire en sorte qu'un seul bouton soit sélectionnable.

J'au aussi appris à utiliser les box qui permettes de regrouper plusieurs composants dans une même case, notamment pour les boutons de bombes et de sauvegardes/réinitialisation.

Les choix des classes est totalement personnel puisque je les ai créés en fonction de l'image que je me faisais du jeu ou des besoins que j'avais dans mon programme lorsque je rencontrais des difficultés.

L'utilisation des classes et des méthodes n'est clairement pas parfaite mais j'ai essayé d'arranger au mieux les éléments afin que ce soit le plus compréhensible et optimisé possible pour le temps que j'avais pour la réalisation de ce programme.

Conclusion :

Pour conclure je dirai que la réalisation de ce projet m'a fait découvrir beaucoup de fonctionnalité que je ne connaissais pas en JAVA. Il m'a donné envie d'améliorer ce jeu même après la remise de ce projet. Je trouve que la réalisation de projet dès la L2 peut être une bonne chose pour les prochaines années. Nous apprenons mieux de nous-mêmes en résolvant nos problèmes et en cherchant de nouvelles fonctionnalités. De manière générale la pratique sur le long terme est plus enrichissante plutôt que de réaliser un devoir en 2h avec beaucoup de contraintes et de stress.