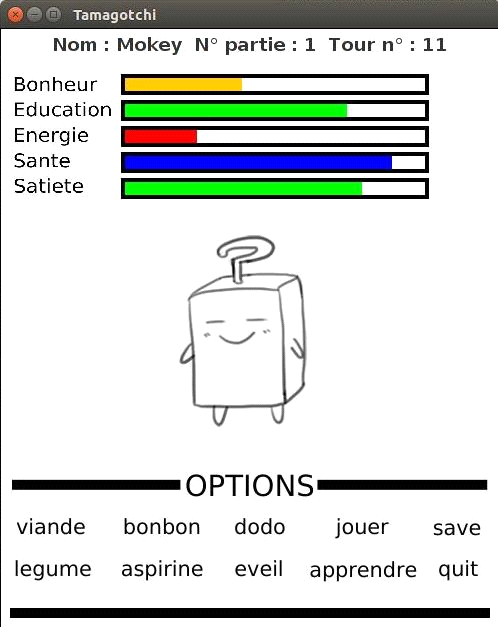
Document de synthèse

Projet informatique - Tamagotchi

Ridley CAMPBELL

Yanzhuo PENG

IT1- 2017/2018



SOMMAIRE

I. Introduction

II. Conception

III. Réalisation et apprentissage

IV. Développement et Utilisation

V. Journal de Bord

VI. Conclusion

I. Introduction

En choisissant le sujet du projet, nous n’avions pas en premier lieu fait attention au nombre d’étoiles à côté de ceux-ci, nous avions donc choisi celui du Tamagotchi simplement parce que nous l’avions trouvé amusant et nous avions pensé qu’il avait du potentiel. Après s’être rendu compte qu’il était plus simple que les autres sujets ( 1 étoile de difficulté indicative seulement ) nous avons décidé de nous compliquer la tâche en rajoutant une interface graphique, une base de données et la gestion du jeu en temps réel. On a donc abordé ce projet avec beaucoup de motivation et d’ambition, en planifiant dès le début de rajouter des fonctionnalités en plus.

II. Conception

En rédigeant le rapport d’analyse, nous avions tout d’abord pensé que les diagrammes de base allaient être simples et basiques. Cependant, après avoir commencé à coder le programme, nous nous sommes rendus compte que nos 3 classes de base (Partie, Tamagotchi et Jeux) ne couvraient pas l’ensemble du programme. Nous avons donc changé cela en rajoutant plusieurs classes supplémentaires : Launcher pour lancer le programme, les sous-classes de Jeux (Morpion et Blackjack,qui a lui même deux sous-classes Carte et Deck), ainsi que MyApplication et MyPanel pour l’interface graphique.

Nous avons approximativement suivi le planning de notre diagramme GANTT, ce qui fait que nous avons plus ou moins fini tout ce que nous avions prévu de faire. Nous avons cependant abandonné l’idée d’intégrer une base de donnée car nous avons trouvé qu’il était plus simple d’utiliser une ArrayList de String. Une simple liste suffit amplement pour analyser les quelques mots appris par le tamagotchi, et analyser les mots appris petit à petit pour vérifier s’ils ne sont pas des pluriels/singuliers, ou le même mot avec des lettres en majuscule/minuscule par exemple. On a donc choisi de ne pas inclure de base de données pour cette fonctionnalité par souci de simplification. Le jeu étant censé à nos yeux rester simple, il n’était pas nécessaire d’accéder efficacement aux mots appris, ni de stocker un volume considérable de mots.

A part cela, nous avons bien réalisé ce que nous avions prévu au départ, à savoir le système de Tamagotchi présenté dans les consignes de départ, la gestion en temps réel et l’interface graphique. Avec plus de temps, on aurait continué de développer des fonctionnalités, principalement pour rendre le jeu plus amusant. Parmi les autres fonctionnalités que nous avons jugé pas nécessaires sont : une gestion du temps réel même lorsque le programme ne tourne pas, un état de maladie qui pourrait être guéri, des affichages graphiques pour les jeux Blackjack et Morpion, plus de customisation du Tamagotchi (plusieurs Tamagotchis disponibles, choix de chapeaux), une croissance du Tamagotchi depuis un œuf vers l’âge adulte, voire même des difficultés croissantes des minijeux en fonction de l’âge ou de l’intelligence du Tamagotchi. Toutes ces choses ont été discutées mais on a préféré se concentrer sur le projet de base qui devait rester simple et aussi fonctionnel que possible.

Le passage du temps « en-dehors » de la session pourrait se faire assez facilement par le biais d’un fichier de sauvegarde. Quand l’utilisateur enregistre sa partie, le programme note l’heure locale de cette sauvegarde. Au chargement de cette partie, on peut comparer cette heure avec l’heure actuelle, en déduire le temps qui s’est écoulé, et appliquer les changements nécessaires. On pourrait par exemple considérer que 1h passée = 1 tour dans le jeu, et diminuer toutes les caractéristiques du Tamagotchi de 1 par heure de décalage. On a préféré se concentrer sur des autres fonctionnalités comme l’affichage graphique.

III. Réalisation et Apprentissage

Le travail a été réparti équitablement entre les deux membres du groupe. Les classes initiales comme Tamagotchi et Partie ont été faites en commun, vu qu’elles commencent assez simplement avec une liste d’attributs pour le Tamagotchi par exemple. La seule vraie séparation du travail était pour la classe Jeux : Ridley s’est occupé de la classe Blackjack, et Yanzhuo du Morpion. Nous avons commencé par mettre en commun nos modifications sur GitHub et le logiciel SmartGit, pour fusionner efficacement nos codes. De plus, lorsqu’on a décidé vers la fin d’inclure l’affichage graphique, c’est plutôt Ridley qui a écrit le code, et Yanzhuo qui a apporté sa « touche artistique ».

Nous avons appris à faire un rapport d’analyse qui consiste à analyser le sujet (c’est-à-dire les besoins de l’utilisateur, les applications du programme, etc). Nous avons donc appris de ce fait à réaliser des diagrammes d’utilisateur, de cas, de classes et autres avec le logiciel Modelio à partir d’un sujet plutôt abstrait. Les diagrammes eux-mêmes ne montrent pas vraiment la complexité du code, vu qu’ils ne sont pas très exhaustifs. Nous avons préféré donner une vue d’ensemble du fonctionnement et ne pas détailler l’encapsulation. Par exemple le diagramme de cas d’utilisation a une case « Interpréter un message » qui regroupe l’ensemble des fonctionnalités des jeux. Toutes les entrées de message, les ouvertures de jeux, et les actions effectuées par le programme pour interpréter et effectuer ces messages passent par deux cases du diagramme.

Nous avons appris à utiliser Java d’une manière plus autonome, c’est-à-dire en décidant nous-mêmes les classes que nous considérons nécessaires, plutôt que de suivre un sujet qui nous les définit déjà. Nous avons aussi renforcé nos connaissances sur Java en général concernant ce que nous avons appris en cours. Nous avons également appris à utiliser Java2D, et plus précisément javax.swing. Nous avons redéfini ses fonctions comme le cours sous forme powerpoint l’a indiqué. En redéfinissant des classes de base, nous avons pu importer des images (comme des dessins du Tamagotchi) pour les afficher dans une fenêtre. On a ainsi découvert et appliqué l’outil de l’affichage graphique, que nous n’avions jusque là jamais utilisé !

IV. Développement et Utilisation

D’un point de vue plus technique, le programme est centré autour de la classe Partie. **Pour** **lancer le programme**, il suffit d’**exécuter la classe Launcher**, la seule qui ait une méthode main(). Elle va créer une instance de la classe la plus importante, la classe Partie. De manière générale, c’est dans cette classe (Partie) que se trouvent toutes les méthodes fondamentales des mécaniques de jeu : le déroulement d’un tour, la vérification de la fin, et surtout les appels aux autres classes. Les classes Blackjack et Morpion n’ont presque aucun lien avec le fonctionnement du programme, si ce n’est un argument ‘partie’ qui permet de transmettre entre autre le surnom du Tamagotchi pour l’utiliser pendant les mini-jeux. Les sous-classes de Blackjack servent à modéliser un vrai jeu de cartes avec 52 instances différentes de carte pour simuler une version simplifiée du Blackjack. Et la classe Tamagotchi ne sert qu’à regrouper les attributs du Tamagotchi, qui seront tous utiles à la sauvegarde et à l’affichage.

L’interface graphique vient ‘se superposer’ au fonctionnement du programme. Au début de chaque tour, on transmet les informations de la partie à une classe héritant de JFrame, MyApplication, qui utilise Jlabel qui permet d’afficher du texte. Cette classe appelle une autre fonction héritant de Jpanel qui elle utilise des images (ImageIcon) et des dessins (FillRectangle) pour afficher les informations de la partie ! Des barres de couleurs représentent les attributs, ainsi que le nom et le numéro de partie sont affichés en haut.

V. Journal de Bord

03/04 : Commencement du projet, analyse du sujet, rédaction du rapport

Nous avons réfléchi à ce que nous voulions et ce que nous pouvions réaliser pour ce projet.

Etant donné que le sujet est à une étoile, nous avions décidé de le rendre un peu plus compliqué et d’y intégrer des fonctionnalités supplémentaires comme la gestion du temps réel et une interface graphique.

Nous avons également commencé la rédaction du rapport d’analyse ainsi que les diagrammes de cas d’utilisation et de classes.

09/04 : Finalisation du rapport

Nous avons fini théoriquement la rédaction du rapport d’analyse qui est susceptible d’être modifié après avoir commencé la partie programmation.

Nous avons fini de programmer les classes basiques (Tamagotchi, Partie et Launcher).

Nous avons commencé à réfléchir aux classes Jeux, Blackjack et Morpion.

04/05 : Rentrée, finalisation des jeux

Nous avons fini de coder les jeux, et nous avions commencé à apprendre Java2D et à coder des programmes basiques avec. Cette partie reste toutefois difficile car nous ne l’avons pas appris en cours.

14/05 : Interface graphique et peaufinement du code

Nous avons rendu le programme plus esthétique et nous avons travaillé sur l’interface graphique en grande partie. Nous avons également dessiné des profils pour le tamagotchi.

16/05 et 17/05: Finalisation du code, rédaction du document de synthèse

Nous avons fini de peaufiné nos codes et nous avons rédigé le document de synthèse ainsi que les documentations.

VI . Conclusion

Globalement, ce projet informatique est une expérience enrichissante qui nous a aidés non seulement pour nous familiariser avec le langage informatique Java, mais aussi pour l’analyse et la gestion d’un projet, ce qui va être nécessaire dans notre future vie étudiante et professionnelle.

Tout le code provient du cours de Java de Clément BOUCHE de cette année ou des cours d’informatique de nos études précédentes. Les images utilisées dans l’interface graphique ont été créées par nous-mêmes, excepté la dernière image « explosion.jpg » qui appartient à pngtree.