Rapport mini projet

I- Déroulement d'une partie du jeu Sushi Go Round (description des algorithmes).

a) Trouver la fenêtre de jeu :

Le programme qui joue à Sushi Go Round démarre après l'exécution du main de la classe JoueSushiGoRound. La première fonction appelée dans le main, ouvrefenetre() de la classe SushiGoRound ouvre le jeu Sushi Go Round dans le navigateur par défaut, puis la fonction trouveFenetre() de la même classe cherche le coin supérieur droit de la fenêtre de jeu.

Nous avons choisi le coin supérieur droit de la fenêtre de jeu, car le coin supérieur gauche porte une inscription qui indique la version du jeu, et cela aurait pu poser un problème dans le cas où la version du jeu serait modifiée. On cherche la fenêtre de jeu pendant 120 secondes, après lesquelles, si la fenêtre de jeu n'a pas été trouvée, une exception qui arrête l'exécution du projet est lancée. Si la fenêtre de jeu est trouvée, on appelle la méthode jeu() de la classe Sushi Go Round qui démarre le jeu.

Coin supérieur gauche de la fenêtre de jeu

Coin supérieur droit de la fenêtre de jeu, utilisé pour la trouver la fenêtre de jeu

b) Initialiser les variables en fonction de la position de la fenêtre de jeu dans l'écran :

La méthode jeu() de la classe Sushi Go Round commence tout d'abord par initialiser différentes variables nécessaires au jeu, telles que le niveau qui est initialisé au niveau1, les boutons play, skip, continue qui permettent d'accéder au jeu, ou encore la position des clients et de leur commandes. Ces variables sont initialisées après l'obtention des coordonnées de la fenêtre de jeu dans l'écran car les cordonnées des différents boutons et clients, dans l'écran de l'ordinateur sont relatives aux coordonnées de la fenêtre du jeu (le coin supérieur gauche du jeu est l'origine du repère).

c) Débuter le jeu :

Ensuite, la méthode commenceJeu() de la même classe clique sur les boutons play, skip et continue pour accéder au premier niveau du jeu. A partir de cet instant et jusqu'à la fin du jeu, la méthode tourJeu() de la classe SushiGoRound réalise toutes les actions nécessaires pour jouer au jeu. Cette méthode sert les clients en appelant la méthode chacunSonTour() de la classe EnsClients tourne en boucle afin de :

- Servir les clients
- Vérifier si une image à l'écran indique la fin du niveau

- Vérifier si on est proche de la fin du jeu pour modifier le comportement du programme (voir partie IA/Optimisation)

d) Servir les clients :

Cf. Diagramme page 6.

e) Gérer les niveaux :

Comme indiqué plus haut, la méthode joue() de SushiGoRound détecte si le niveau du jeu courant est terminé. Si le niveau est perdu, on considère que le jeu est terminé (voir partie suivante). Si le niveau est gagné, la méthode fait appel à la fonction nextLevel() qui modifier l'attribut niveau, qui devient un attribut qui a le type du niveau suivant. Par exemple, si le programme vient de gagner le niveau 1, niveau qui était un attribut de type niveau 1 devient également un attribut de type niveau 2.

f) Détecter la fin du jeu :

Si un niveau est perdu, on considère que le jeu est fini et la méthode verifPerdu() de la classe SushiGoRound lance une exception de type JeuFini qui arrête l'exécution du programme.

Si le programme gagne le niveau 7, qui est le dernier niveau du jeu, on fait appel à la méthode nextLevel() de la classe Niveau7 qui lance une exception de type JeuFini qui arrête l'exécution du programme.

II- Préparation :

La première étape de préparation a été de réfléchir à une première architecture du programme, et de nous mettre d'accord sur la manière de représenter les clients, les commandes, les ingrédients et les sushis ainsi que sur la manière de gérer les stocks d'ingrédients

La seconde étape a consisté à récupérer plusieurs informations contenues dans le jeu, et nécessaires pour jouer au jeu.

Il a tout d'abord fallu récupérer les différentes images nécessaires à la reconnaissance d'images et notamment, le coin supérieur droit de la fenêtre de jeu pour détecter la position du jeu dans l'écran. Pour pouvoir écrire la partie du programme qui sert les clients, nous avons également du récupérer les images des sushis nécessaires pour reconnaître les commandes.

Images utilisées pour la reconnaissance des commandes

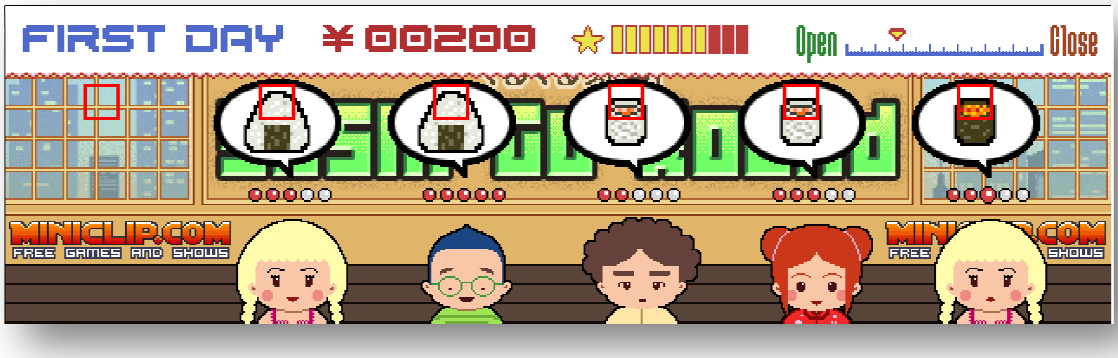
Nous avons également récupérer deux images différentes pour détecter si la partie qui vient de se terminer a été gagnée ou perdue.

Image utilisée pour reconnaître lorsque le niveau est perdu. Il s'agit d'une partie du message « sorry » affiché

Image utilisée pour reconnaître lorsque le niveau est perdu. Il s'agit d'une partie du bouton continue

Pour récupérer les images, nous avons réalisé des captures d'écran à l'aide de la classe Robot, puis nous avons enregistré ces images car à la méthode write de la classe ImageIO. Nous avons fait ce choix, plutôt que de réaliser les captures d'écran manuellement à l'aide d'un impr. écran, pour la raison suivante : pour deux mêmes images, les images obtenues grâce à une capture d'écran manuelle et celles obtenues grâce à une capture d'écran réalisées par la classe Robot n'ont pas la même taille en octet, et ne sont donc pas comparables.

Il a également été nécessaire de récupérer les coordonnées des différents boutons (ingrédients, téléphone, tapis de préparation, assiettes des clients), et des endroits où réaliser les captures d'écran pour la reconnaissance d'images.



Les carrés rouges représentent les endroits où sont réalisées les captures d'écran pour reconnaître les commandes

Pour récupérer ces coordonnées nous avons utilisé le logiciel Gimp qui indique les cordonnées d'un pixel dans une image.

Pour obtenir les informations relatives à un niveau, il est nécessaire d'arriver à ce niveau, donc pour obtenir les images ou les recettes des sushis pour chaque niveau, nous avons intégrer une méthode qui prend régulièrement des captures d'écran des commandes. Cela n'a pas posé de problèmes au niveau du programme, car, si on prend par exemple le programme muni des informations pour jouer au niveau 1, ce même programme peut jouer au niveau 2, mais il considérera le nouveau sushi du niveau 2 comme une client absent, et prendra des captures d'écran du sushi.

Diagramme de l'algorithme qui gère les commandes et le stock