FOURNIER Bastien
ABDOULAYE Hamat

ColorSwitch

Compte Rendu

Nous avons choisi d'implémenter le jeu ColorSwitch en python avec la librairie Pygame pour s'essayer à python, d'en apprendre un peu plus sur ce langage que nous n'avons peu pratiqué.

Pygame nous semblait être une librairie parfaite pour le projet avec une documentation très bien expliquée et détaillée mais en avançant dans le projet, on s'est vite rendu compte que la librairie était limitée avec parfois des fonctions obsolètes comme la fonction rotate() qui ne permet pas une réelle rotation mais plutôt un redimensionnement de la surface dans laquelle la figure va effectué sa rotation or une surface est un quadrilatère.
En rencontrant, cette difficulté de faire tourner nos figures, nous avons songé à changer de librairie début février après avoir seulement implémenté l'interface et le cercle, sachant qu'à ce moment-là, nous avons perdu un membre de notre groupe.
Finalement, après avoir testé d'autres librairies en Java et en C, nous avons conclu qu'il nous aurait fallu trop de temps pour apprendre une nouvelle librairie et implémenter le jeu de base à 2 avant la date du rendu.

Nous nous sommes donc mis à cherché un moyen de faire tourner nos figures avec des formules mathématiques, Hamat a donc pensé à la rotation plane, nous avons alors testé plusieurs implémentations de cette méthode mais sans succès.
En effet, à chaque rotation la figure rétrécissait, la figure devait donc être réinitialisée à chaque tour, ce qui n'était pas du tout fluide.
Voyant que la rotation plane ne marchait pas, nous nous sommes penchés sur la rotation d'un point sur un cercle par rapport au centre qui nous a permis de faire la rotation que l'on a actuellement.

Certaines figures de Pygame (les arcs par exemple) laissent paraître le fond, pour palier à ce problème, nous avons du afficher deux fois la même figure mais décalé d'un pixel, ce qui marche plutôt bien.
Pareil pour l'antialiasing, nous avons dessiné des courbes ou des lignes aux bordures des figures pour affiner l'affichage.

Pour les figures, nous avons décidé d'en faire des Sprites ce qui nous facilite la tâche pour les regrouper en groupe de Sprites et de les dessiner sur une Surface en une ligne avec la fonction draw() de Group, sans devoir recopier le même code d'affichage à chaque fois.
Pour les collisions, nous avons principalement utilisé les masks des Sprites.
Les masks permettent de prendre en compte la forme de la figure dessinée dans la Surface du Sprite et non pas le Rect de la Surface, ce qui est utile pour les formes arrondies.

Après avoir créé les principales figures c'est-à-dire la Ligne et le cercle nous avons pu réaliser par la suite plus facilement le carré et la Croix par exemple, ensuite pour les autres figures ce sont simplement des empilements de figures qui ont été effectué.

Bastien s'est occupé d'implémenter le cercle, la Ligne et la Croix tandis que Hamat a implémenté le carré, le Triangle et le parallélogramme qui a remplacé la classe Carrée.
Nous avons ensuite pu implémenter les différentes extensions facilement.

Le MVC a été intégré dès le départ dans le projet, le contrôleur contient les événements c'est-à-dire les actions du joueur (contrôle clavier et souris).
Le modèle contient toutes les classes du projet et les fonctions permettant de gérer les figures au cours du jeu (génération aléatoire des figures).
La vue permet de gérer tout ce qui est visuel c'est-à-dire les menus, l'interface en jeu, le background, et l'affichage des Sprites.

Pour les extensions, nous avons essayé d'implémenter le projet sous Android sans grand succès malheureusement, pygame utilise la librairie SDL 1.0 et pour implémenter Pygame sous Android il faut utiliser une version expérimentale de Pygame sous la SDL 2.0.
Cette dernière version n'est que très peu documenter et toutes les fonctionnalités ne sont pas implémentées, ils nous ont donc fallu parcourir le code source et suivre les indications pour porter le projet sur smartphone (https://github.com/renpytom/rapt-pygame-example).
Malheureusement certaines figures comme les arcs ne sont pas disponibles dans cette version et les Mask ne sont pas implémentés donc les collisions ne sont possibles qu'avec des Rect.
Cependant nous avons réussi à faire fonctionner le projet sur Android mais de manière très simpliste.

En conclusion, le projet nous a permis de nous améliorer en python, de mieux concevoir le modèle MVC. Ce qui a été le plus compliqué dans projet a été l'implémentation de la rotation mais une fois passé celle-ci le projet s'est passé sans encombre malgré que l'on ne soit que deux, on regrette juste ne pas avoir totalement réussi le portage sur Android.

GitHub : https://github.com/FournierBastien/Projet_S6

PS : Pour le son, nous avons du mettre des chemins en dur, c'est pourquoi nous avons les sons en commentaire.