Projet Java III - 2022-2023
HELBAquarium

POPADIUC CLAUDIU

Monsieur Riggio, Jonathan

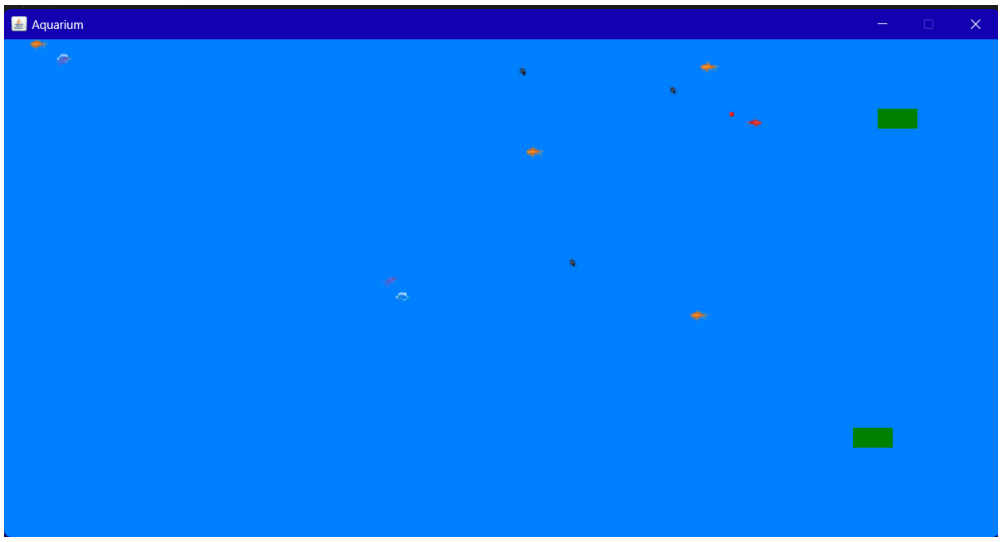
Table des matières

Introduction

Dans le cadre du cours de JAVA Q III, il nous a été demandé de réaliser un aquarium. Pour avoir un certain début, monsieur Riggio nous a proposés de partir du « Snake » comme base fonctionnelle, c'est ce que j'ai fait personnellement. Le projet a été réalisé en oriente objet, c'est une condition obligatoire.

Lors de ce rapport je vais commencer par expliquer les fonctionnalités présente dans mon Aquarium ainsi que leur fonctionnement, je continuerai par expliquer l'ajour de la fonctionnalité supplémentaire et sont, fonctionnements, ensuite à l'aide d'un diagramme, j'expliquerais la structure du projet, pour donner suite à cela je parlerais des limitations de mon projet, et je finirai par mon avis lors d'une conclusion.

Fonctionnalités de base



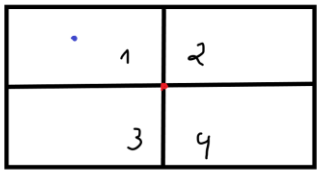
Les fonctionnalités suivantes sont fonctionnelles dans le projet :

Mouvements des 4 types de poissons :

Pour les déplacements de tous les poissons, j'ai créé une classe parent « Fish ». Tous les déplacements viennent de cette classe. Dedans Il y a une fonction « moveFishToTarget », cette fonction permet de donner une Target à tous les poissons.
Ainsi ils sauront tous ou aller tout le temps car chaque poisson aura une Target à chaque instant.
Dans « Fish », je crée aussi une méthode « positionFish » qui fera en sorte d'ajouter tous les poissons dans des position aleatoire au lancement de la partie.
Chaque poisson peut se déplacer dans l'aquarium, soit verticalement, soit horizontalement, soit en diagonale grâce au système que je viens d'expliquer, leurs Target dépendra de leur comportement, et pour cela il y aura une classe pour chaque type de comportement, ils hériteront tous de la class « Fish ». Ils auront tous leurs positions aléatoires et leur déplacement vers une Target. Chaque poisson a aussi un ID car c'est nécessaire lors du fonctionnement des insectes, pellets et des poissons qui doivent mourir, le poisson rouge doit savoir quel poisson il devra tuer exactement, ceci sera mieux expliquée lors de son comportement.
Toutes les variables qui sont utilisées plusieurs fois dans les comportements de poisson ont été ajouter une seule fois dans Fish, il y aura un getter et parfois un set en fonction du besoin de la variable dans les autres classes, leur utilisation sera expliquée.

1. Les poissons orange se déplacent constamment aléatoirement jusqu'à rencontrer un bord de l'aquarium, à ce moment précis, ils prendront une nouvelle position aléatoire.
   Pour cela j'ai créé une class « FishOrange ».
   A l'intérieur j'ai ajouté une méthode « randomEdgePosition » qui donnera un bord aléatoire au poisson orange comme Target position.
   Dans cette méthode je crée un random de 1 à 4 car il y a 4 bords, et chaque random donne les coordonner d'un des bords mais tout à fait aléatoirement que ce soit le bord gauche au milieu, plus bas ou plus haut, etc…
   Une fois cette Target définie, le poisson orange ira jusqu'à cette Target grâce à « super.update », cette ligne de code est place dans d'autre méthodes mais pour comprendre pourquoi, il va falloir comprendre le fonctionnement des Pellets et des insectes, cela viendra plus tard.
   Il faut aussi savoir que la Target a une hitbox, car le poisson avance par plus d'un pixel en x et y (il avance en +6).

2. Les poissons mauves se déplacent toujours dans la direction la plus éloignée du poisson rouge le plus proche. Les poissons mauves sont également stimulés par les décorations de l'aquarium, ce qui a un effet sur leur vitesse de déplacement.
   Pour cela j'ai créé une class « FishPurple », dans cette class il y a une méthode « oppositeDirectionOfTheRedFish », cette méthode prendra en compte toute la liste des poissons, une condition vérifiera que si le poisson en cours, c'est un poisson rouge, et si c'est le cas, il calculera le poisson rouge le plus proche en utilisant ce calcul :
   (Distance = Math.sqrt(x*x + y*y)).
   Une fois que le poisson mauve aura calculé la distance de tous les poissons rouges, il prendra le plus proche, en comparant les distance, ensuite le poisson mauve ira dans la direction la plus éloignée du poisson rouge.
   Pour mieux expliquer ce qui va suivre je vous fais ce dessin :



   Admettons que le poisson rouge est au milieu, si le poisson mauve est dans la case n°1, il ira dans le coin supérieur gauche, si le mauve est dans la case n°3, il ira dans le coin inférieur gauche, etc…
   Ainsi le poisson mauve ira toujours dans la direction la plus éloigné du poisson rouge. C'est ce qui se passe dans la fin de la méthode à l'aide de plusieurs vérifications, la Target du poisson mauve sera toujours un des 4 coins de l'application.
   J'indique aussi que le poisson mauve a une vitesse de basse similaire aux autres poissons mais j'ajoute à sa vitesse, le nombre d'obstacles dans l'aquarium (6+n°Obstacle).

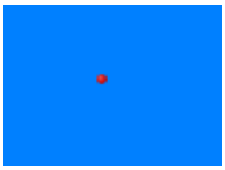
3. Les poissons bleus se déplacent toujours dans la direction du poisson mauve ou bleu le plus proche. Pour ce fait comme pour le poisson mauve, j'ai calculé le poisson le plus proche en utilisant le calcul de la distance, cette fois-ci, il ira chercher le poisson mauve ou bleu le plus proche et le donnera comme Target. Mais une des conditions en plus dans la méthode de direction du poisson bleu c'est qu'il ne peut pas se définir lui-même comme Target, cela arrivait et le code ne fonctionnait plus alors j'ai dû vérifier que dans la liste de poissons, le poisson en cours n'avait pas la même position que le poisson bleu pour lequel tout le calcul était fait.
   Ce poisson est aussi plus rapide que la vitesse de base, sa vitesse passe à 7 au lieu de 6.

4. Les poissons rouges se déplacent toujours dans la direction du poisson, non rouge, le plus proche. Quand un poisson rouge rencontre un poisson d'une autre couleur, celui-ci le mange, ce qui a pour effet de le faire disparaitre de l'aquarium.
   Les poissons rouges sont stimulés par la température de l'aquarium. Ce qui veut dire que si j'appuie sur la touche 1 ils iront moins vite, la touche 2 ce sera la vitesse de base et la touche 3 ils seront plus rapides que tous les poissons, cela sera mieux expliquer lors des touches du clavier.
   Comme pour le poisson mauve le poisson rouge fera le même calcul de distance pour trouver un poisson d'une autre couleur, il prendra le plus proche, une fois qu'il rentrera dans sa hitbox il le supprimera de la liste des poissons, cela est fait via la méthode « fishKilling », qui prendra en compte dans le Board, une méthode qui supprimera le poisson qui a été touché de la liste de poissons, cela en utilisant l'ID du poisson qui a été ajouté préalablement dans la classe « Fish ».

Déplacements et vitesse :

Sauf exception, les poissons n'arrêtent jamais leur déplacement dans l'aquarium.
Les poissons se déplacent avec une certaine vitesse de base. Cette vitesse de base je l'ai mise pour chaque poisson à 6.
Pour la vitesse de base, fait de mettre moins de 6 aurait causé un souci pour la contrainte que aucun bonus ne peut dépasser le double de la vitesse de base. Le mauve est stimulé par le nombre d'obstacles j'en ai alors mi qu'entre 1 et 3 afin d'ajouter 1,2,3 au poisson mauve en vitesse, il ira donc jusqu'à 9.
Avec les bonus d'insecte ça monte à 10 pour tous les poissons.
C'est pour cela que j'ai choisi 6 en vitesse de base, il fallait calculer le fait que le poisson mauve peut être stimulé et que le rouge doit être dans tous les cas le plus rapide des poissons dans le cas où la touche 3 est pressé, alors même si le mauve a 6+3 en vitesse, le rouge avec la touche 3 il passerait à 11 et il serait dans tous les cas, le plus rapide même si un poisson venait à prendre un insecte.

Pastilles :



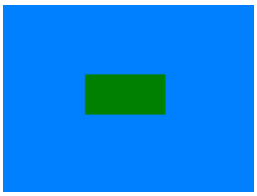
Au lancement de la simulation, l'aquarium peut contenir un certain nombre de pastilles. Ces éléments sont placés de façon aléatoire dans l'aquarium. Ceci est réalisé grâce à la class « EdiblePellet », comme pour les poissons les pellets prendront une position aléatoire, mais cet fois ci, cela viendra de la classe « GameObjectElement », la classe des pellets héritera de celle-ci, il y aura dans la classe d'élément, une méthode qui mettra tout les pastilles dans une position aleatoire, une condition sera appliqué pour vérifier que la pastilles se voit bel et bien dans l'aquarium, pour qu'on ne la voit pas qu'a moitié, au cas ou la position du pellet serait exactement la limite de l'aquarium.
Revenons à la classe des pastilles, à l'intérieur de celle-ci, une méthode vérifiera si un poisson entre en collision avec la hitbox d'une pastille, une fois que ce sera fait, on rappel la méthode de position aleatoire d'éléments afin de supprimer la pastille et d'en ajouter une autre ailleurs dans l'aquarium, ainsi l'aquarium ne manquera jamais de pastilles.
Une fois que cela sera fait, une variable compteur prendra 10 secondes, en fonction du nombre de pellet dans l'aquarium, le nombre dans la variable sera différent, mais vu qu'il y aura plus de pellet, la méthode sera appelée plus de fois, alors il est important d'ajouter au compteur le nombre de poissons pour bien arriver à 10 secondes. Ensuite une variable ID, prendra en compte l'ID du poisson qui a touché la pastille et celui-ci sera vérifié dans la classe « Fish », ci la variable ID est différente de -1 (qui est l'ID par défaut quand aucun poisson ne prend de pellet), alors le poisson avec cet ID pourra avancer normalement et tout les autres ne pourront plus avancer. Il faut savoir aussi que le compteur décrémentera et une fois qu'elle arrivera à 0, l'ID du poisson qui avait touché la pastille prendra -1 il n'y aura donc plus de

poissons qui pourrai avoir un ID de la pastille touché, alors tous les déplacements se feront normalement.

Insectes :

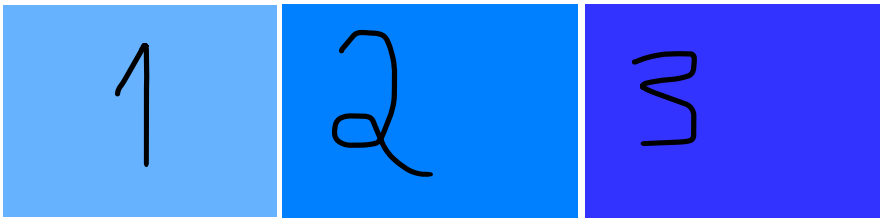
Au lancement de la simulation, l'aquarium peut contenir un certain nombre d'insectes. Ces éléments sont placés de façon aléatoire dans l'aquarium. Ceci est réalisé grâce à la class « Insect », comme pour les pastilles, les insectes prendront une position aléatoire grâce à la classe des éléments.
Dans « Insect » j'ai ajouté une méthode avec 3 noms diffèrent, chaque insecte aura un nom diffèrent et cela est fait pour pouvoir savoir quel insecte des 3 a été touché, lorsqu'un poisson rentrera dans la hitbox d'un insecte, le nom du poisson sera enregistré dans une variable, la condition vérifiera aussi quel nom l'insecte porte, grâce à la liste de Fish qu'on traversera avec un « for » et en fonction du nom, le temps de chaque insecte sera diffèrent mais donnera à tous le même bonus de vitesse qui sera 11. Comme pour les pastilles, le poisson qui aura touché l'insecte, une variable enregistrera son ID.
Dans la classe « Fish », on vérifiera comme pour les pastilles, si l'ID est diffèrent de -1 si oui, on remplacera simplement la vitesse du poisson en question à 11, le compteur qui prend en compte le temps en fonction du nom d'insecte, débute dans la classe « Insecte » quand l'ID est différent de -1, il décrémente jusque 0, une fois à 0, l'ID du poisson qui touche l'insecte se remet a -1 et la vitesse du poisson revient a sa vitesse de base.

Obstacle :



Pour les obstacles, j'ai créé une class « Obstacle » dans lequel chaque obstacle aura une position random dans l'aquarium, ceci grâce à l'héritage de « GamElement »
A l'intérieur de cette classe, il y a une méthode qui vérifiera si un poisson rentre en collision avec la hitbox de l'obstacle. Quand cela arrivera une variable prendra l'ID du poisson qui a touché la hitbox. Comme pour les insectes, une vérification dans « Fish » sera faite pour voir si un poisson est en contact avec un obstacle et si oui, il le contourne, si la Target du poisson est au-dessus du poisson, donc le poisson est en bas de l'obstacle, le poisson ira d'abord par la droite. Phénomène similaire s'il vient d'en haut, et si il vient de la droite ou de la gauche il montera d'abord.

Température de l'aquarium :



L'aquarium possède une température de base, qui par défaut est tiède. Pour ma part elle sera mise en un bleu normal (image 2), Il est possible de changer la température de l'aquarium sur trois configurations différentes : Froid, tiède et chaud. Pour les 3 changements il y aura du bleu mais diffèrent, bleu clair (image 1) pour le cold et bleu foncé (image 3) pour le hot.
Dans la configuration froide, les poissons rouges ont un malus de vitesse par rapport à leur vitesse de base, ce qui les rend plus lents que tous les autres types de poissons. Pour cela dans mes touches de clavier, je ferais un changement de vitesse du poisson rouge et du background. Il y a un setter qui changera la vitesse du poisson rouge de 6 à 10 pour la température chaude, et de 6 à 3 pour la température froide, pour la température tiède, cela reste à 6. D'autres explication seront donnée dans les touches 1,2 et 3.

Reproduction :

Quand deux poissons de la même couleur entrent en contact, ils peuvent s'accoupler. Les deux poissons qui sont entrés en contact disparaissent, et trois nouveaux poissons de la même couleur que les deux précédents sont placés, chacun, à une nouvelle position aléatoire dans l'aquarium.
Dans ma classe « Fish » j'ai ajouté une méthode « couplingFish » qui vérifiera si le poisson du for en question n'a pas la même position du poisson de la classe même, si ce n'est pas le cas il vérifiera si le poisson en cours entre dans en contact via une hitbox avec un autre poisson. Pour donner suite à cela il fera une dernière vérification si les 2 poissons sont de la même couleur. Si c'est le cas, il supprimera les 2 poissons et ajoutera 3 autres de la même couleur dans des positions aléatoires via les méthodes implémenter dans Board.
Vu que monsieur Riggio nous a demandés de faire en sorte de limiter la reproduction en fonction du nombre de poissons présents dans l'aquarium alors j'ai ajouté une variable qui prendra en compte la taille de la liste de poissons à chaque instant, avec cette variable j'en ajoute une autre qui fera un random du nombre de poissons, pour finir je vérifie que si ce random est égal à 0, ils se reproduise, sinon il ne font rien ainsi au plus il y aura de poissons au plus il y aura un grand nombre random, au moins il y aura de chance que ce soit égal à 0, donc moins de chance de se reproduire.

Commandes de l'aquarium :

Afin de permettre des interactions entre l'utilisateur et l'aquarium, il existe un système de commandes permettant d'influencer la simulation. Quand l'utilisateur appuie sur une touche, cela déclenche un évènement précis :

Ces touches ont été ajoutées dans la classe Board, avec une méthode « keyPressed », pour chaque touche une condition vérifiera si la touche a été pressée et il y aura ensuite à l'intérieur soit une fonction soit directement le code du comportement de la touche :

- 0 : Reset l'aquarium.

Une méthode fera en sorte de supprimer tous les éléments de l'aquarium, de mettre le compteur des pellets à 0 au cas où, avant que la touche soit pressée, un des poissons aurait pris un pellet et avait arrêté les autres poissons pour 10 secondes.
Le background reprendra sa couleur par défaut (tiède) et la méthode « initGame » sera appelée pour recommencer le jeu.
La vitesse du poisson rouge ira de nouveau à 6 au cas où elle aurait été modifier par les touches 1,2 ou 3.
Je mets aussi les Boolean des touches 6,7,8 à « false », au cas où elle était pressé, car ces touches change le comportement des poissons et pour bien tout réinitialisé, il faut aussi par cela.
Je redonne aussi des valeurs aléatoires pour les pellets, obstacles et insectes afin de ne pas avoir les mêmes valeurs qu'au démarrage de l'aquarium.

Je remets aussi le nom du poisson à « Default », qui avait été enregistrer si l'une des touches R, M, B, O avait été pressé, afin de reprendre leurs déplacements si la touche 0 est pressé.

- 1 : Définir la température de l'aquarium sur froid.

Diminue la vitesse du poisson rouge avec un set de vitesse qui est initialisé dans le poisson rouge, qui changera donc la vitesse du poisson rouge de 6 à 3, alors le poisson avancera par pas de 3 en x et en y au lieu de 6.
Il y aura aussi un changement du background en un bleu plus claire.

- 2 : Définir la température de l'aquarium sur tiède.

Similaire que pour la touche précédente mais le poisson rouge prend sa vitesse de base (6), avec la couleur par défaut du background

- 3 : Définir la température de l'aquarium sur chaud.

Similaire, sauf avec un background en un bleu plus foncé, et une vitesse supérieure que tous les poissons, de 6 à 11.
Chose importante à savoir, il y avait certains bugs s'il prenait un insecte alors il aurait changé sa vitesse de 11 à 10 (10 la vitesse bonus d'un insecte) et ne serait plus, plus rapide que tous les autres poissons, alors dans ma classe « FishRed », j'ai indiqué soigneusement que si le poisson avait une vitesse = à 11 alors il ne prendrait pas en compte la vitesse bonus d'insecte qui est seulement de 10 car dans tous les cas le poisson rouge serait le plus rapide en 11.

- 4 : Ajouter à une position aléatoire un insecte.

Cela est fait via la commande add, pour la liste d'insectes, dans le Board, comme au démarrage de l'application quand il y a un certain nombre par défaut de poisson et d'éléments, rien de bien compliqué.

- 5 : Ajouter à une position aléatoire une pastille comestible.

Similaire que pour la touche 4 mais pour des pastilles.

- 6 : Passer en mode insectivore. Dans ce mode, tous les poissons qui ne sont pas

Stoppés, se dirigent vers l'insecte le plus proche, s'il y en a un dans l'aquarium.
Un Boolean « insectivorMod » par défaut sera à « false », si la touche 6 est pressé, le Boolean passe à « true ». Dans la classe « Fish », on vérifiera lors du update si le Boolean est à « true », si oui alors une méthode sera appelle, cette méthode fera en sorte que tout les poissons qui ne sont pas à l'arrêt vont vers l'insecte le plus proche, cela est fait comme pour les autres poissons, en calculant la distance de tout les insectes et en prenant la plus proche. Une fois que ceci est fait, la Target des poissons ne sera plus leur comportement mais l'insecte le plus proche de chacun.

- 7 : Passer en mode pastille. Dans ce mode, tous les poissons qui ne sont pas stoppés, se dirigent vers la pastille la plus proche, s'il y en a une dans l'aquarium.

Similaire que pour l'insecte sauf que ce sera la pastille la plus proche.

- 8 : Passer en mode reproduction. Dans ce mode, tous les poissons qui ne sont pas stoppés, cherchent à rencontrer, le poisson de la même couleur, le plus proche.

Similaire que pour l'insecte, mais ce sera pour le poisson de la même couleur le plus proche.

- 9 : Ajoute un poisson, de type aléatoire, à une position aléatoire dans l'aquarium.

Une méthode qui fera un random jusque 5 (5 : voir fonctionnalité supplémentaire), pour choisir une couleur de poisson aléatoire, une fois que c'est fait il l'ajoutera simplement dans la liste des poissons, comme pour la touche 4 et 5.

- R : Cela permet de stopper tous les poissons sauf les rouges.

Un string prendra le nom du poisson rouge, et dans la classe du poisson rouge, lorsqu'il est censé aller vers sa Target position, il vérifiera d'abord la variable du nom de la touche R, si c'est « FishRed » ou « Default » il continuera d'avancer.
En ce qui concerne « Default », c'est la valeur par défaut du nom de poisson dont la touche est appuyée, elle ne changera pas tant qu'une des 4 touches n'ont pas été enclencher, il est important donc de faire avancer tous les poissons tant que la variable reste à « Default », sinon ils ne bougeront plus.
Dans la classe « Fish » il y aura aussi une vérification, si la couleur de la touche pressée est différente de « Default », il fera avancer seulement les poissons avec la couleur de la variable du nom de la touche, ici ce sera rouge, pour le reste des poissons il n'avanceront plus jamais.

- B : Cela permet de stopper tous les poissons sauf les bleus.

Similaire que pour la touche R.

- M : Cela permet de stopper tous les poissons sauf les mauves.

Similaire que pour la touche R.

- O : Cela permet de stopper tous les poissons sauf les oranges.

Similaire que pour la touche R.

Fonctionnalités supplémentaires

Additionnement, il nous a été demandé d'implémenter au minimum une fonctionnalité Supplémentaire originale, j'ai ajoutée alors le « FishBlack ».

Le poisson noir : l'ennemie jurée des poissons rouges, son seul et unique but sera d'exterminer tous les poissons rouges, avant que celui-ci n'ait tué tous les autres poissons. Ainsi il pourra protéger tous les autres poissons de leur prédateur, pour y arriver il est le seul poisson sur qui il n'y aura pas d'effets si un des autres poissons prend un pellet, il pourra continuer de traquer les poissons rouges, plus de cela le poisson noir pourra toujours prendre les insectes, donc il pourra avoir les bonus de vitesse des insectes afin de rattraper tous les poissons rouges. Malheureusement, il serait aussi affecté par le comportement des pastilles, si un poisson venait à en prendre.

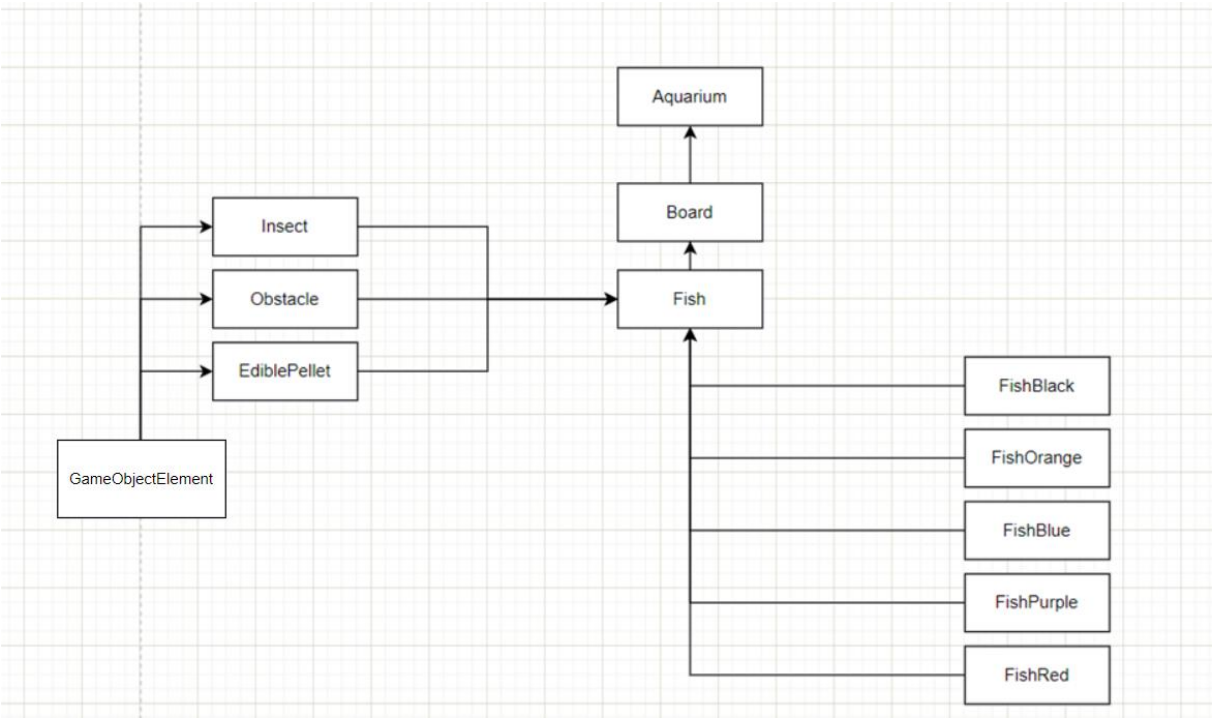
Si le poisson Noir venait à s'accoupler, il n'aurait pas 3 mais 1 seul enfant, les parents disparaitront et 1 seul poisson noir arrivera dans l'aquarium. Les façons pour le poisson noir d'exister c'est en pressant la touche « 9 », et c'est seulement s'il a de la chance qu'il apparaitra car c'est un random jusque 5, afin de faire apparaitra n'importe quel poisson, et pas seulement le noir.

Le poisson noir pourra aussi exister en pressant la touche « A » sûr le clavier, mais cela est à double tranchant si l'utilisateur appuie sur cette touche alors un autre poisson rouge apparaitra aussi, il y aura donc 1 poisson noir mais un 2ème poisson rouge dans l'aquarium ! Ainsi de suite si la touche est encore pressée (J'ai fait cela pour toute fois compliquée un peu la tâche du poisson noir, sinon il aurait trop souvent gagné).

Pour faire tout cela, j'ai donné comme Target au poisson noir, le poisson rouge le plus proche, une fois qu'il atteindra sa hitbox, il le supprimera de la liste des poissons, comme déjà expliquer dans le poisson rouge.

En ce qui concerne l'accouplement, lorsque 2 poisson noir se touche, il n'y aura juste pas 3 mais 1 seul qui apparaitra, via la méthode d'ajout de poisson pour l'accouplement dans Board, en fonction du nom des poissons qui se sont touché, si c'est noir ce ne sera pas 3 mais 1, une simple condition est réalisée.

Analyse



Aquarium est la classe qui démarrera l'aquarium, c'est là qu'il y aura le main. Le Board quant à lui, c'est là qu'énormément de chose ce passera comme les ajouts des poissons dans leurs listes, leur image etc... Et Board sera utilisé lors du démarrage de l'aquarium.

Les class Insect, Obstacle ainsi que EdiblePellet hériteront de GameObjectElement, ainsi ils auront chacun une poisitiona aléatoire dans l'aquarium, chacune des classes aura son propre comportement, j'ai fait cela pour evité la duplication de code. Et chacune des 3 classes aura une certaine verification dans la class « Fish » afin de réalise leur comportement.

Il y a aussi la class Fish, ou chaque poisson y trouvera sa position aléatoire avec leurs déplacements et leurs fonctions réalisées lors de ce projet.

Enfin chaque couleur de poisson héritera des positions aléatoires de la classe Fish, ainsi que les mouvements avec Target, et chaque type de poisson aura son propre comportement dans sa classe respective, la plupart des variables similaires dans les comportements de poisson ont été ajouté une fois dans « Fish » afin de les réutiliser dans les autres classes.

Limitations

Limite du nombre d'éléments dans l'aquarium :

J'ai choisi de mettre un nombre assez faible d'obstacle, de pellet ainsi que d'insecte.

Ce choix est stratégique car il y aura des problèmes à mettre 100 pellets, cela donnera trop vite la possibilité au poisson de prendre un pellet ce qui arrêtera toujours les déplacements des autres poissons. Similaires que pour les insectes, leur vitesse sera toujours augmentée, plus il y aura de pellets et d'insectes plus l'aquarium ne ressemblera plus à rien, déjà à partir de 20, 30 ça commence, même si en termes de fait ce n'est pas problématique si les poissons sont plus rapides mais ce sera trop fréquent, le plus gros souci sera pour les pellets, plus aucun poisson ne bougera appart celui qui là pris et on ne verra aucun comportement appart d'un seul poisson.

Pour ce qui est d'obstacle si on en met déjà 50 on ne verra presque plus l'aquarium car il sera submergé par les obstacles. Plus de cela on ne verra plus les poissons leurs déplacements seront étranges ils ne sauront plus où aller, car ils sont censés les contourner ou pour l'orange prendre un autre bord. Ce problème sera amplement plus grand avec un plus grand nombre d'obstacles.

On peut parler aussi du nombre de poisson, si on en mettait 1000, de couleurs au hasard, les poissons rouges tueraient très vite tous les autres poissons (testé et approuvés : moins de 1 seconde). Cependant vu le nombre de poissons même s'il y a peu de pellet, il y en aura toujours 1 qui touchera un pellet et si ce n'est pas un rouge alors les poissons tueront peut-être 800 poissons en 1 milliseconde (sans exagérer vu qu'il y a au moins 200 poissons rouges), du coup il y aura le même problème que s'il y avait 1000 pellets, in ne verrait que le comportement d'un seul poisson.

Limite de la vitesse du poisson Mauve :

La vitesse du poisson mauve augmente en fonction du nombre d'obstacles présents dans l'aquarium.Pour cela j'ai fait en sorte que ce soit un random entre 1 et 3 obstacles mais si on en met déjà un de plus, la vitesse du poisson mauve de base qui est de 6, montrera à 10, et c'est la vitesse qui est augmentée lorsqu'un poisson prend un insecte alors celui-ci n'aura plus d'effet sur le poisson mauve, s'il y a 10 obstacles, la vitesse du poisson mauve augmentera considérablement et dépassera le double de la vitesse de base à partir du 6^{e} obstacle et compromettra les restrictions du projet, qui est qu'aucun poisson ne peut dépasser le double de la vitesse de base. Si on donne la possibilité d'ajouter 1000 obstacles, le poisson mauve se déplacera tellement vite qu'il sera impossible de le voir bouger.

Limite de reproduction :

La reproduction fonctionne mais à partir du moment où ils sont beaucoup plus nombreux dans l'aquarium, suite aux reproductions, ils ont évidemment beaucoup plus de chance de se reproduire malgré le fait que j'ai diminuer leur chance de reproduction au maximum, et à partir d'un certain moment l'aquarium crash tellement il y a de reproduction. C'est un fait que j'ai pu remarquer (qui est quand même rare).

Exception des déplacements :

Une fois que le poisson rouge a mangé tous les poissons de l'aquarium, il n'aura plus de Target, il fera donc partie des exceptions des poissons qui ne bouge plus. Il s'arrêtera et ne bougera plus tant que aucun poisson n'a pas été ajouté.
Pour le poisson mauve il s'avère qu'il s'arrête au coin de l'aquarium une fois qu'il y arrive, car c'est la position la plus éloigné du poisson rouge le plus proche, similaire que pour le mauve le poisson bleu suit le poisson mauve et s'arrêtera avec lui dans le coin. Mais de temps en temps si un autre coin est plus loin que le poisson rouge, le mauve changera de coin.

Plus de temps :

Si j'avais eu plus de temps j'aurais fini le projet à 100%.
Les fonctionnalités que je n'ai pas réussi à régler sont : les obstacles.
En ce qui concerne les obstacles normalement tout est là pour que ça fonctionne, le code a l'air bon, je ne vois pas mon erreur, mais malheureusement de temps en temps un poisson s'arrête devant l'obstacle et ne bouge plus, de temps en temps il le traverse.
J'ai aussi remarque que les poissons fessaient des vas et viens dans l'aquarium sans raison, de temps en temps.

Conclusion

Je ne mentirais pas si je disais que j'ai passé plus ou moins 150 heures sur ce projet, et je peux dire haut et fort que je ne regrette rien, j'ai tellement appris en faisant ce projet. J'en suis absolument fière, je pense que c'est la chose la plus incroyable intellectuellement que j'ai faite de ma vie ! Alors un grand MERCI monsieur Riggio pour m'avoir donné la possibilité de me surpasser lors de ce projet. J'ai pu m'améliorer énormément en tout ce qui est programmation, java, orienter objet, etc…

Au début c'était très difficile, rien que d'y penser j'en avais peur, j'ai beaucoup galéré, mais à l'aide de votre cours de ce que j'ai appris l'année passée, j'ai pu réussir à surmonter les moments difficiles et passer plusieurs journées entières sur ce projet pour finalement réussir (Même si à mon avis c'est réussi à 90%).

Il y avait des moments où je me disais : « bon c'est que 3 crédits », « laisse tomber », je pensais très sincèrement ne jamais y arriver, j'étais assez pessimiste, je ne voyais pas de progrès je n'arrivais pas à avancer, je me souviens lors d'un cours en JAVA, vous aviez posé comme question dans votre questionnaire sur l'avancement du projet, j'ai commencé à dire que c'était en d'autres mots que c'était impossible pour moi car je n'avais pas assez de compétence pour y arriver que ce n'était pas le temps le problème. Et devinez quoi, j'en suis là maintenant, j'écris ces mots en sachant que j'ai pratiquement fait fonctionner tout ce qui a été demandé lors de ce projet.

Finalement ceci a pu être réalisé, car j'étais seul, et même si je n'étais pas du tout content de cette manière de fonctionner, je me dis que je n'aurais jamais fait tout ça si jetais avec quelqu'un. Je ne me serais jamais surpassé à ce point-là, alors je voudrais vous dire que vous prenez une excellente décision en nous laissant tout seul pour vos projets !

Cependant une chose est sûre même si je ne regrette rien, j'ai dû faire certains sacrifices en termes de choix de travail que ce soit pour l'école ou la vie réelle, et ce dû au nombre de choses que vous demandiez à faire pour le projet.

Mon seul conseil serait peut-être de diminuer la charge du projet, peut-être mettre moins de chose à faire, moins de condition, restrictions etc… car ces petits détails prennent énormément de temps je sais que pour certains, cela ne pose pas de problème et ils peuvent faire ce projet en y travaillant quelques heures par jour en 1 semaine mais pour d'autres comme moi, cela m'a fait faire certains sacrifices. Peut-être que d'autres ne feront pas ces sacrifices et choisiront de rater le projet juste parce que c'est un peu trop de travail, j'espère que vous avez pu comprendre mon avis sur la longueur du projet.

Je pense très certainement garder ce projet tout ma vie, un jour je relierais ce rapport et je me souviendrais où tout a commencé.