PROJET TUTEURE – ANGRY BIRDS

Groupe L5

Hachemi Sonia
Spinnewyn Quentin
Rocchia Tom
Dupriez Alexandre

Sommaire

1/ Liste des courbes paramétrées

Nos courbes sont calculées grâce à ces différents paramètres, que l'utilisateur génére grâce au système "Drag and drop" qu'il effectue à l'aide de sa souris.

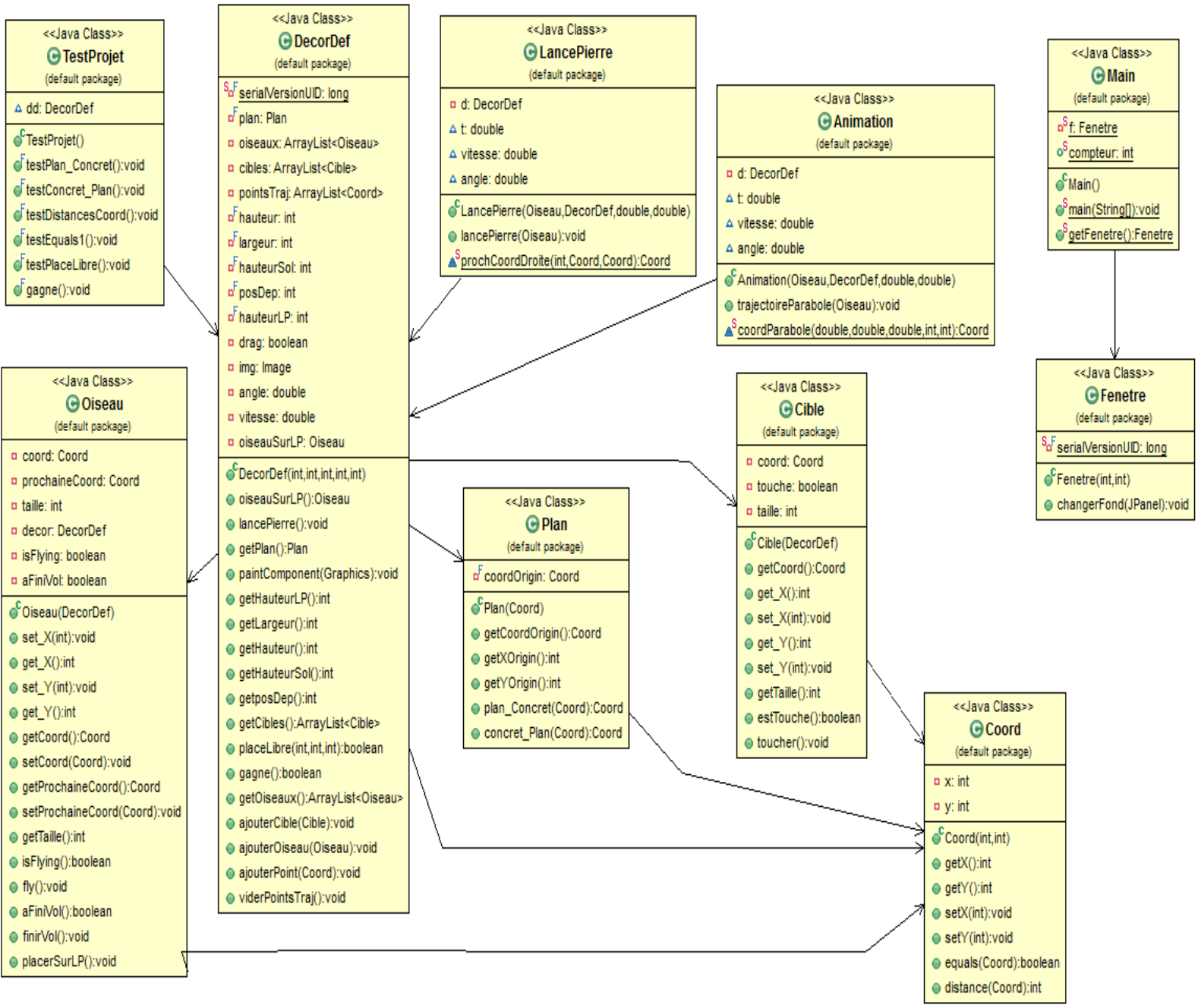
Au plus l'oiseau est tiré vers l'arrière, au plus la vitesse augmente, et au plus il est levé l'angle baisse.

Ainsi, il est plus simple de toucher les obstacles puisque le mécanisme permet de visualiser où notre oiseau va atterir.

$$x = v*\cos(angle)*t$$
$$y=-1/2gt^2 + v*\sin(angle)*t + h$$

avec : v → la vitesse de départ
angle → l'angle de départ
t → le temps
g → la constante G = 9,81
h → la hauteur de départ

2/ Schéma UML du projet



3/ Rôles de chacun dans le groupe

Sonia Hachemi : chef de projet, travail effectué dans le développement essentiellement, et a effectué les répartitions des rôles au sein du groupe. Recherche sur les raisonnements mathématiques au sujet du Drag & Drop. Modifications et réparations de quelques bugs.

Quentin Spinnewyn : travail effectué dans le développement essentiellement, et dans la réalisation de l'UML. Recherche sur les raisonnements mathématiques au sujet du Drag & Drop. Modifications et réparations de quelques bugs.

Tom Rocchia : travail effectué dans le développement essentiellement ainsi que la réalisation de la Javadoc. Recherche sur les raisonnements mathématiques au sujet du Drag & Drop. Modification et optimisation des tests.

Alexandre Dupriez : travail effectué dans le développement essentiellement. Recherche sur les raisonnements mathématiques au sujet du Drag & Drop. Modification et optimisation des tests

4/ Tests

Les trois classes de tests ont été rassemblés pour en former une seule et unique.

La classe est donc TestProjet réunissant les tests du plan, de certaines méthodes de décor, et des coordonnées.

4/ MVC

Nous nous sommes servi du Design pattern Observable/Observer.
Les oiseaux et les cibles sont des Observable dont les couleurs et les coordonnées sont modifiées au cours du jeu soit par le biais d'une interaction de l'utilisateur, soit dû à sa trajectoire.
La classe DecorDef est observeur et observe chaque oiseau et cibles qu'il contient et est redessinée dès lors que l'un de ces élements sont modifiées.