Design Rationale – Lim Shir Yin, Tan Jie Yen

Player

Players are able to choose their own names and they will get an empty Inventory and an amount of EcoPoints at the beginning of the game.

Ground

An ArrayList of type Object will be used to hold all objects that currently existing in its coordinates eg. (1, 1) on the ground. Implementing it by using an ArrayList instead of Array is to make it mutable in size and easier for adding and subtracting. There will be two attributes in Ground objects, which is x and y coordinates. For example, a Map with width 3 and height 3 will instantiate the Ground objects with coordinates (1, 1), (1, 2), (1, 3), …(3, 3). (1, 1) means that the first element in each tuple is x coordinate and the second element is y coordinate.

Bush

It is one of the items contained within the Ground ArrayList. Implementing an ArrayList called fruits for fruits that are produced from the bush. When there is a 10% that bush will produce a fruit, the fruit will be added into the fruits ArrayList. An allowableActions is implemented to allow player to perform PickFruitAction(). Fruits in fruits ArrayList will be removed once the player has picked up the fruit on the particular bush.

Tree

It is one of the items contained within the Ground ArrayList. Implementing an ArrayList called fruits for fruits that are produced from the tree. A method called incrementEcoPoints has created to increase the ecoPoints by 1 when a ripe fruit is produced by the tree. In the tick() method, another fruitsToDrop ArrayList has created to collect fruits that are drop from the tree. Fruits in fruitsToDrop ArrayList will be removedin fruits ArrayList. The dinosaur will not be able to step on the coordinates of the ground that contains Tree. An allowableActions is implemented to allow player to perform PickFruitAction().

World

At the beginning of the game, the player will choose the dimension, a World and GameMap will then be created based on the dimensions chosen. The Ground instance will then be created. For example, dimension such as 3x3 will create 9 ground instances, Bush Instance will then add into the Ground.

Eatable

An interface class is implemented. Food that can be eaten by dinosaur. Every Eatable must have a food level.

Dinosaur

This class will be the parent class of Stegosaur, Brachiosaur and Allosaur. It will implement attributes which are type, gender, growth stage. The children class that inherited from Dinosaur class which are Stegosaur, Brachiosaur and Allosaur have almost the same methods and attributes.

- pregnantThreshold is created to get the maximum turns of pregnant before laying eggs
- eggHatchThreshold is created to get the maximum turns of egg before hatching into baby dinosaur
- babyThreshold is created to get the maximum turns of baby before turning into adult
- unconsciousThreshold is created to get the maximum turns of unconscious before die
- deadThreshold is created to get the maximum turns of death before disappearing from location
- hungryThreshold is created for the hit point to become hungry
- breedThreshold is created for the enough hit point to breed
- corpseFoodLevel is created for the food level of the corpse when this dinosaur dies
- eggEcoPoints is created for the eco points of dinosaur's egg

An enum is created for Stage of Dinosaur (eg. baby, adult). A hashmap is created for behaviours of each dinosaur.

A getAllowableActions() is created to returns a collection of the Actions that the otherActor can do to the current Actor in the Dinosaur class.

Stegosaur

There will be methods inherited from Dinosaur class.

There will be attributes such as, pregnantThreshold, eggHatchThreshold, babyThreshold, unconsciousThreshold, etc.

A getAllowableActions() method is created to returns a collection of the Actions that the otherActor can do to the current Actor in the Stegosaur class.

A getEatAction() method is created for a new eat action when the food in the location is eatable for stegosaur.

A canEat() method is to check whether the food is eatable for stegosaur. In this case, only fruit and mealkit are eatable.

Once a Stegosaur becomes unconscious, the status will change to unconscious and it will be dead after 20 turns, the corpse will remain in the location for 40 turns, then a Stegosaur instance will be removed.

The corpse can be eaten by Allosaurs, Stegosaur instance will then be removed.

Brachiosaur

There will be methods inherited from Dinosaur class.

There will be attributes such as, pregnantThreshold, eggHatchThreshold, babyThreshold, unconsciousThreshold, etc.

A getAllowableActions() method is created to returns a collection of the Actions that the otherActor can do to the current Actor in the Brachiosaur class.

A getEatAction() method is created for a new eat action when the food in the location is eatable for brachiosaur.

A canEat() method is to check whether the food is eatable for brachiosaur. In this case, only fruit and mealkit are eatable.

Once a Brachiosaur becomes unconscious, the status will change to unconscious and it will be dead after 15 turns, the corpse will remain in the location for 20 turns, then the Brachiosaur instance will be removed.

The corpse can be eaten by Allosaurs. Brachiosaur instance will then be removed.

Allosaur

There will be methods inherited from Dinosaur class.

There will be attributes such as, pregnantThreshold, eggHatchThreshold, babyThreshold, unconsciousThreshold, etc.

A getEatAction() method is to create a new eat action when the food in the location is eatable for allosaur.

A canEat() method is to check whether the food is eatable for allosaur. In this case, only egg, corpse and mealkit are eatable.

A getAttackAction() method is implemented for a dinosaur attack action.

An ArrayList will be created to add the dinosaurs that is attacked by the allosaur.

Once the food level is 0, the corpse will remain in the location for 20 turns, then the Allosaur instance will be removed.

The dinosaur being attacked by allosaur will be marked, the mark will then be removed after 20 turns. Allosaur will not be able to attack the marked dinosaur.

Food

This class will be the parent class of Egg , Fruit and MealKit. It will implement cost method and amount of food level increased method that will be inherited by Egg, Fruit and MealKit.

Egg

A class that inherits methods from Food class. There will be "cost" method that is inherited from the Food class.

Once an egg is hatched based of their number of turns, the Egg instance will be removed, and a Dinosaur instance will be created.

A method called incrementEcoPoints has created to increase the ecoPoints when an egg is being hatched.

An egg instance will be created once an egg is bought from the Vending Machine.

If the egg is breed from female adult dinosaur, an Egg instance will be created after number of breed turns.

If an egg is eaten by Allosaurs, the Egg instance will be removed.

Fruit

A class that inherits methods from Food class. There will be "cost" method and "amountFoodLevelIncreased" method that is inherited from the Food class.

- rotTurns is created to check the number of turns fruit lays on the ground.

- A Fruit instance will be deleted after being left on the ground for 15 turns.

- A Fruit instance will be deleted after being eaten by dinosaur.

- A Fruit instance will be added in Inventory if it is picked up by player.

MealKit

A class that inherits methods from Food class. A "type" attribute is created to decide the type of MealKit (vegetarian or carnivore). An instance of MealKit will be removed once it is eaten by Dinosaur, provided that the Dinosaur hasn't reached the maximum of food level.

Inventory

Inventory contains an array of items that are collected by the player. Java Array is being used as the size of inventory needs to be at a fixed size to prevent unfairness. Items are not allowed to be added once the inventory is full. Each item that is being collected consumes one space of the inventory Java array.

LaserGun

It contains a “cost” attribute of 500 eco points. Once the amount of stegosaurs has reached at a point, laser gun will be used to attack stegosaurs, the stegosaurs instance will be removed.

VendingMachine

A Java Array that contains the items to be sold to the players. There might be more than one vending machines on the map which it will then extend from the Ground class to fix the position.

EcoPoint

It acts as the game currency. It doesn’t require to create a class as it can be created as an attribute in the Item class (such as Egg, MealKit, etc.). It can be added to or subtracted from after a Player makes a purchase on vending machine.