Assignment 2 REQ1: Let it grow!

Overview:

Due to the reason that mature tree can now spawn flying koopa (Assignment 3 REQ2) and sprout and sapling have 50% chance to spawn fire flowers (Assignment 3 REQ4) when it reached the growth age, I had updated the class diagram, sequence diagram and the design rationale. However, since only minor changes are made, therefore the most part of the design rationale for this REQ will be no change. In other words, only Mature, Sapling and Sprout class will be modified due to the changes made. Please note that although Tree class extends HighGround (i.e., a new class created in Assignment 2), however, the purpose of creating HighGround is to achieve the feature in Assignment 2 REQ2, therefore HighGround class will not be considered in the design rationale and UML diagrams for REQ1.

1) Tree class

No changes are made between Assignment 2 and Assignment 3.

2) Sprout class

What changed in the design between Assignment 2 and Assignment 3 and Why:

The only change that I made between Assignment 2 and Assignment 3 is when the Sprout instance' age reaches 10, before we change the type of ground at the location to be Sapling, we will have a 50% chance to spawn a fire flower on that location. This could be done by using an if statement. However, the rest of the part remains the same as previous assignments. Thus, please refer to the design rationale for this class in previous assignments for more details.

3) Sapling class

What changed in the design between Assignment 2 and Assignment 3 and Why:

The only change that I made between Assignment 2 and Assignment 3 is when the Sapling instance's age reaches 10, before we change the type of ground at the location to be Mature, we will have a 50% chance to spawn a fire flower on that location. This could be done by using an if statement. However, the rest of the part remains the same as previous assignments. Thus, please refer to the design rationale for this class in previous assignments for more details.

4) Mature class

What changed in the design between Assignment 2 and Assignment 3 and Why:

The only change that I made between Assignment 2 and Assignment 3 is that for each turn, the mature tree will have 15% chance to spawn either Koopa or FlyingKoopa on that location. However, the rest of the part remains the same as previous assignments. Thus, please refer to the design rationale for this class in previous assignments for more details.

5) Enemy class

No changes are made between Assignment 2 and Assignment 3.

6) GeneralKoopa class

It is an abstract class that functions as a base class for subclasses (e.g., FlyingKoopa, Koopa). However, since GeneralKoopa class will not be implemented in this REQ (i.e., it's implementation will only be considered in Assignement 2 REQ3), therefore the design rationale for GeneralKoopa class will not be available in this section. **Hence, please go to the Assignement 2 REQ3 section in the following pages to see the design rationale for GeneralKoopa class.**

7) Koopa class

No changes are made between Assignment 2 and Assignment 3.

8) FlyingKoopa class

FlyingKoopa class is a class that represents one of the enemies in this game, which is Flying Koopa. However, since FlyingKoopa class will not be implemented in this REQ (i.e., it's implementation will only be considered in Assignment 3 REQ2) and in this REQ, it only creates the FlyingKoopa instance without considering any features of a FlyingKoopa instance, therefore the design rationale for FlyingKoopa class will not be available in this section. **Hence, please go to the Assignment 3 REQ2 section in the following pages to see the design rationale for FlyingKoopa class.**

9) Coin class

No changes are made between Assignment 2 and Assignment 3.

10) Dirt class

No changes are made between Assignment 2 and Assignment 3.

11) Status class

No changes are made between Assignment 2 and Assignment 3.

12) FireFlower class

FireFlower class is a class that represents the fire flower in the game map. After it is consumed by the actor, it will allow actor to perform the fire attack action. However, since FireFlower class will not be implemented in this REQ (i.e., it's implementation will only be considered in Assignment 3 REQ4) and in this REQ, it only creates the FireFlower instance without considering any features of a FireFlower instance, therefore the design rationale for FireFlower class will not be available in this section. **Hence, please go to the Assignment 3 REQ4 section in the following pages to see the design rationale for FireFlower class.**