Chris Drazek-Loosier

Benjamin Mueller

**Domain Class Documentation**

* **Entity (abstract)**
  + The abstract “Entity” class is responsible for outlining the base functionality and parameters of all entities, including bees and flowers. Functions include:
    - Track collisions between the entity and a given position based on the Euclidian distance formula
    - Have parameters for xPosition, yPosition, and energy, and implement functionality to set/get those parameters as necessary.
  + This class will be implemented by Benjamin Mueller
* **Bee (abstract)**
  + The abstract “Bee” class is responsible for outlining the base functionality and parameters that are shared among all types of bees. Bee will also contain all the functionality inherited from Entity. Functions include:
    - Handle a base case for movement
    - Handle a general collision event for a bee that adds energy or takes it away depending on what type of flower the bee collides with
  + This class will be implemented by Benjamin Mueller
* **BeeDirected**
  + The “BeeDirected” class is responsible for handling the behavior and functionality of the type of bee that chooses a flower at random and moves toward it, and once it reaches it selects another flower and does the same. Functions include:
    - Functionality inherited from “Bee” (and by default “Entity”)
    - A modified movement function to reflect the necessary behavior of this type of bee.
    - The ability to choose a flower at random
  + This class will be implemented by Benjamin Mueller
* **BeeRandom**
  + The “BeeRandom” class is responsible for handling the behavior and functionality of the type of bee that moves through the garden bed in a preset fashion such as in lines or circles. Functions include:
    - Functionality inherited from “Bee” (and by default “Entity”)
    - A modified movement function to reflect the necessary behavior of this type of bee.
  + This class will be implemented by Benjamin Mueller
* **Flower (abstract)**
  + The abstract “Flower” class is responsible for outlining the base functionality and parameters that are shared among all types of flowers. Functions include:
    - Functionality inherited from “Entity”
    - Change the energy of the flower when there is a collision
  + This class will be implemented by Chris Drazek-Loosier
* **GoodFlower**
  + The “GoodFlower” class is responsible for handling the behavior and functionality of the type of flower that gives bees energy. Functions include:
    - Functionality inherited from “Flower” (and by default “Entity)
    - Track when there is a collision to deplete the energy level of the flower so that it will not continue giving out energy indefinitely
  + This class will be implemented by Chris Drazek-Loosier
* **BadFlower**
  + The “BadFlower” class is responsible for handling the behavior and functionality of the type of flower that takes energy from bees. Functions include:
    - Functionality inherited from “Flower” (and by default “Entity)
    - It should be noted that the “energy” parameter for the BadFlower will be negative, so that when there is a collision the operation will remain the same.
  + This class will be implemented by Chris Drazek-Loosier
* **GardenBed**
  + The GardenBed class is responsible for containing Bee and Flower objects. Functionality includes what is listed below.
    - Creating instances of each kind of bee and flower with random x and y coordinates
    - Store locations of each type of object
    - Stepping forward in time so that the movement of bees may be observed, and functions carried out
  + This class will be implemented by Chris Drazek-Loosier

Build Confirmation & Proof of Classes

A screenshot of a computer screen

Description automatically generated

Class Diagram

A close up of a map

Description automatically generated