Design Report: eco-system of a rabbit eating carrot in a forest

1. Project description

There is a rabbit in a forest decorated with grass and trees. There are some carrots appeared in the forest indefinitely. The rabbit seeks and eats the nearest carrot. A new carrot appears randomly once a carrot has been eaten.

2. Proposed Structure (Classes)

Rabbit

Rational: hold a rabbit object's properties with fields and capabilities with methods

Fields and responsibilities:

- PVector pos, dim; // private fields for holding its position and dimensions (width and height)
- Double velocity; // private field for holding its moving velocity scale
- double scale; // private field for holding the scale factor for the rabbit

+ Rabbit(PVector pos, PVector dim, double vel, double scale) // constructor to instantiate a rabbit object with parameters of pos, dim, vel, and scale
+ move() // public method to move the rabbit
+ draw(Graphics g) // public method to draw the rabbit
+ eat(ArrayList<Carrot> carrots) // public method to eat the carrot and spawn a new carrot

Tree

Rational: hold a tree object's properties with fields and capabilities with methods

Fields and responsibilities:

- ArrayList<Tree> trees; // private static field for holding its instances
- PVector pos, dim; // private fields for holding its position and dimensions (width and height)
- double scale; // private fields for holding the scale factor for the tree

- Tree(PVector pos, PVector dim, double scale) // constructor to instantiate a tree object with parameters of pos, dim, and scale
+ init(int n) // public static method to create n instances randomly
+ draw(Graphics g) // public method to draw the tree

Forest

Rational: hold a forest object's properties with fields and capabilities with methods

Fields and responsibilities:

- PVector pos, dim; //public fields for holding its position and dimensions (width and height)

+ Forest(PVector pos, PVector dim) // constructor to instantiate a forest object with parameters of pos and dim
+ draw(Graphics g) // public method to draw the forest

Carrot

Rational: hold a carrot object's properties with fields and capabilities with methods

Fields and responsibilities:

+ ArrayList<Carrot> carrots; // public static field for holding its instances
- PVector pos, dim; // public fields for holding its position and dimensions (width and height)
- double scale;// public field for holding its scale factor

- Carrot(PVector pos, PVector dim, double scale) // constructor to instantiate a carrot object with parameters of pos, dim, and scale
+ draw(Graphics g) // public method to draw the carrots

3. Execution flow in the panel class: RabbitPanel <- a subclass of JPanel

Fields and responsibility:

- Rabbit rabbit; // field for referencing the Rabbit object as the main character
- Forest forest; // field for referencing the Forest object as the environment

+ RabbitPanel() // constructor where the objects of Rabbit, Tree, Forest, and Carrot will be created and assign to the fields declare above
+ paintComponent(Graphics g) // public method inherited from the JPanel class, where the draw methods will be called upon to render them
+ actionPerformed(ActionEvent e) // public method inherited from ActionListener interface to call Rabbit's move and eat method

4. PPP for Rabbit's main methods

draw method (a Graphics 2D* parameter needed here as a placeholder for an argument for translation and drawing):

Use the Graphics2D object passed in:

1. Translate to the current location as held in pos
2. Scale the drawing with the scale factor as stored in scale field
3. Set color for its body
4. Draw its body using an oval
5. Draw its two hands using ovals
6. Draw its two legs using ovals
7. Draw its head using ovals
8. Draw its tail using an oval
9. Draw its ears using ovals
10. Set color for its inner-ears
11. Draw its inner-ears using ovals
12. Set color for its eyes
13. Draw its eyes using ovals

move method (no parameters):

1. Loop through all carrots and find the closest carrot
2. Find the unit vector toward that carrot
3. Multiply that unit vector by the vel factor
4. Add that vector to pos

eat method (an array list of carrot objects from the static field of Carrot class):

1. Loop through all carrots and see if any carrot is touching the rabbit
2. Remove that carrot(s) from the array list
3. Spawn a new carrot and add to the array list