Project Documentation For 3D Application - Inheritance

Author: Mason Fraser
Date: March 1, 2019

Subject

Table of Contents

List of Tables

List of Figures

1. Project Overview

Take your previous 3D application and add sub-types of GameItems.

2. Project Requirements

Convert provided code and previous assignment code into more OOP oritented code. EG. Take a flopping ball and make it two different types of flopping balls, one that doesn't leave the ground and one that does.

2.1. Requirements (stated by Customer or Teacher)

- Car:
  - moves on the ground only, bouncing off the borders, and other GameItems
- Plane:
  - moves through the air (x-y-z) and bounces off the borders (top and Bottom included).
- Car and Plane start per GUI controls.
  - Car and Plane start slow, and stay slow...easier to control this way,
- You may or may not include:
  - "flubber" bounce.
  - gravity

3. Design Plans

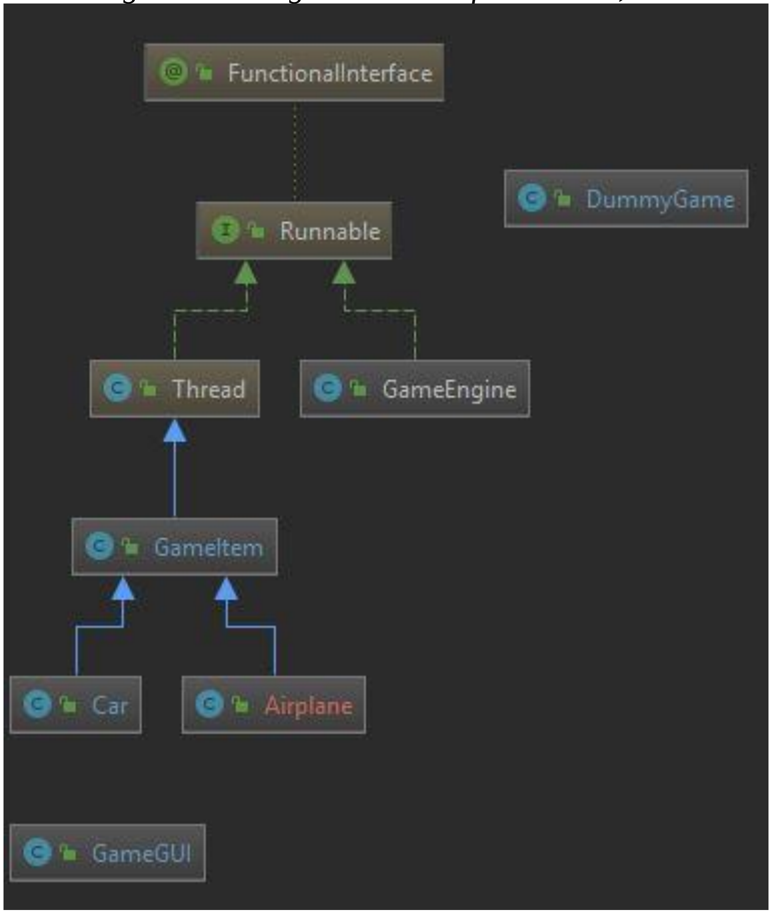
Well I was hoping to cheese the assignment, but I figured out that's harder than doing it half-right and will make life harder down the road.

3.1. Diagrams

Diagrams now included, with extra ZEST!

3.1.1. Class Diagrams

Class diagram showing what comes form where, and what's super-duper important.



3.1.2. Collision

I didn’t make any changes to collision since it worked flawlessly the way you had it before, aside from removing gravity it’s the same.

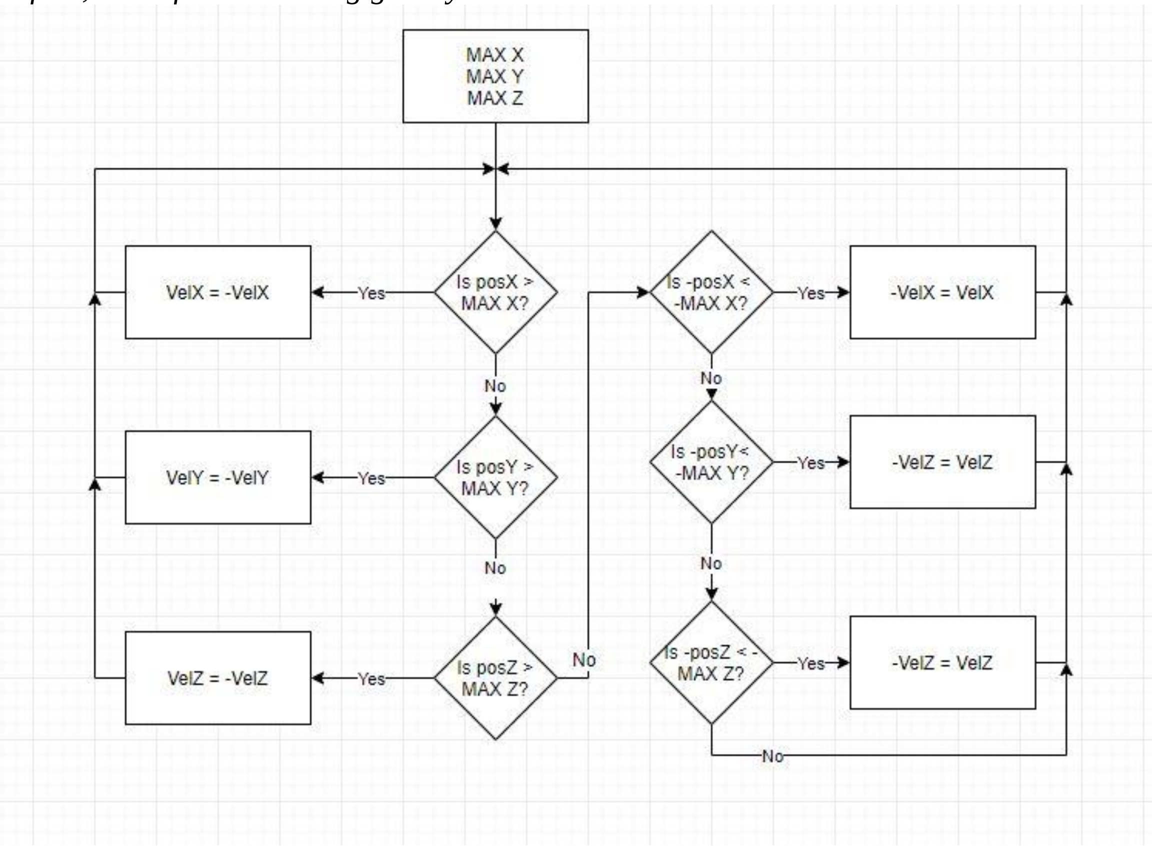
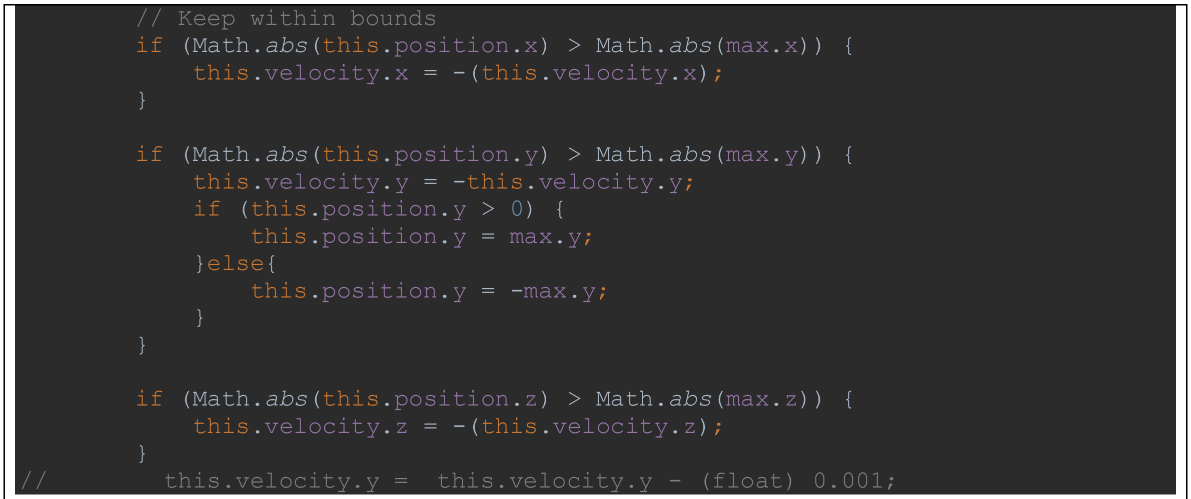


Table 1: Collision Code



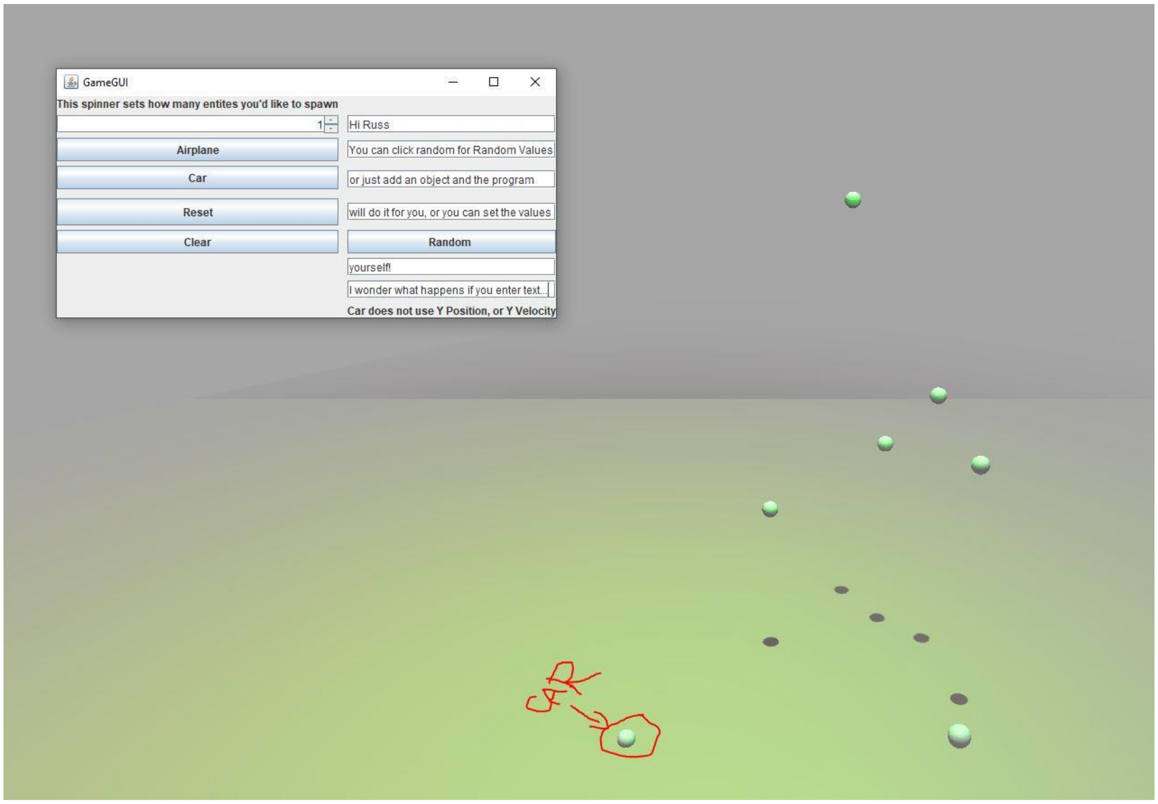
```
        // Keep within bounds
        if (Math.abs(this.position.x) > Math.abs(max.x)) {
            this.velocity.x = -(this.velocity.x);
        }

        if (Math.abs(this.position.y) > Math.abs(max.y)) {
            this.velocity.y = -this.velocity.y;
            if (this.position.y > 0) {
                this.position.y = max.y;
            }else{
                this.position.y = -max.y;
            }
        }

        if (Math.abs(this.position.z) > Math.abs(max.z)) {
            this.velocity.z = -(this.velocity.z);
        }
//        this.velocity.y =  this.velocity.y - (float) 0.001;
```

3.2. User GUI Controls

You can add an airplane which behaves as expected, or a car, which also behaves as expected. Values can be entered manually, left blank, or randomly. Leaving the values blank will cause every new generation to be random, if you click random the values get set to random values and stay those values until you hit random again, thus the objects created will spawn the same way every time.



GameGUI

This spinner sets how many entites you'd like to spawn

1

Hi Russ

You can click random for Random Values

or just add an object and the program

will do it for you, or you can set the values

yourself!

I wonder what happens if you enter text...

Car does not use Y Position, or Y Velocity