Custom Physics Simulation

Documentation for My Custom Physics Simulation

# Demonstration Brief

The capabilities of my custom physics simulator is demonstrated by my application in the following ways:

* Creation of dynamic and static physics objects
  + By clicking the left mouse button, a dynamic physics object with a random velocity is created, represented by a blue circle outline which moves along the screen.
  + By clicking the right mouse button, a static physics object is created, represented by an orange circle with stays in place, regardless of forces put upon it.
* Addition of forces to physics objects
  + Gravity can be turned on or off by clicking the center mouse button, which adjusts the forces put on the physics objects in the scene. The gravity applied can also be changed with the arrow buttons. This is represented by the dynamic physics objects moving down or up on the screen in accordance with the gravity applied.
  + There are also forces applied when collisions occur, as detailed below.
* Collision detection and resolution between circles
  + When collisions occur, forces are applied to both objects involved in the collision to push them away from each other (unless one or both objects are static, in which case only dynamic objects are shifted). This is demonstrated by the dynamic circles on the screen “bouncing” off of each other, or by the dynamic objects sitting on top of static objects if their velocity isn’t strong enough to pull them one way or the other.

# Class Diagram

Graphical user interface

Description automatically generated with medium confidence

# Public API

## game

### game() - constructor

Initialize all variables to suitable defaults. Note that the fixedTimeStep is not initialized – this is up to the user to do in their own game loop.

### float fixedTimeStep

The target amount of time between time steps as expressed in seconds. Evaluated by the shouldTickPhysics() function to determine if a physics update is due.

### void init()

Create our window and initializes our rendering context.

# Potential Future Improvements

I hope to come back in the future to implement functioning AABB-AABB collisions and Circle-AABB collisions, as well as more elements to make the application function as more of a game than a simulation.

# Credits

Josie DeRosier © 2021