Danielle Gray  
10/6/2014

**Grape Eater**

The concept of my world is a mouse that moves around and ‘eats’ grapes that have fallen on the ground. If the mouse doesn’t eat the grapes in time, the grapes disappear. The game keeps track of how many grapes have been eaten and how many have been missed. If the number of missed grapes gets up to three, the game ends, and you lose. There are random ‘spills’ on the floor that slow the mouse down. In order to regain normal speed, the mouse has to ‘wash’ itself off in the blue ‘spill’, which is supposed to be water. The mouse navigates itself through addition using Vector3s. If the player is in a panic and knows they cannot reach the grape before it destroys itself, then they can use the ‘Help’ option up to three times.

The grapes spawn randomly and do not overlap with a set time interval. Over time, the time interval slowly decreases until it reaches its minimum limit. With the idea of using rectangles that I got from you, I created a list of rectangles because I found that rectangles have an overlap function. In my spawn function, I have a random x and random y value, and I then create a proposed rectangle using those values. I then go through the list and check for each other rectangle already in the list (existing grapes) whether this new proposed rectangle overlaps any of those. If it does not, a new grape object is then created, and the proposed rectangle gets added to the list. I had a lot of difficulty with this at first, as I had tried using bounds.min.x/bounds.min.y/ etc. of the renderer, and using math to add and subtract things to get widths, and it just wasn’t working out. Using rect made things a lot easier.

I learned through this project how to use Coroutines. These helped out a lot with delaying the spawn time and self-destruction time. InvokeRepeating didn’t work well for me simply because I needed the delay to change over time, and not be set in the start. They’re super simple and easy to use. Although, I did have issues because Unity itself wasn’t using my updating code (even when I hit save) and I spent a long time trying to figure out what I did wrong, but it ended up being nothing. You basically make the function an ‘IEnumerator’ and then call it using StartRoutine(Name of your function). To have a delay you simply write yield return new WaitForSeconds(Seconds) in the function before whatever code you want to delay.

The help option destroys all the grapes on the screen. I had trouble with this because no matter what I had, it would only destroy the latest grape. It was ironic, because in the beginning of this project, I could only get it to delete all the grapes. I ended up creating another script that found all the objects of type grape on the screen and then destroys them all.