Adding functionality to classes while adhering to their anatomy

When extending the functionality of C# scripts, our added code needs to comply with the basic anatomy, syntax, and naming conventions.

When we create Instances of Prefabs at a spawn location, we want to destroy them at some point trough mechanisms like a 'lifetime' property.

One way to do this is adding some functionality to our Random Walker class.

adding an expiry date to our random walkers

Open the **Random Walker** Script and after the Class Declaration add:

```
public float lifetime = 5f; //the lifetime of the GameObject
private float age = 0f; //the age of the GameObject
```

In the **Update ( )** function add:

```
age += Time.deltaTime;
        if (age >= lifetime)
       {
           Destroy(gameObject);
       }
```

> Now, every lifetime seconds, the spawned game objects get destroyed.

Making random walkers fade away instead of disappearing instantly

When our 'random walkers' expire into the ether, it might be more appropriate to just let them fade out instead of seeing them instantly disappear, whenever age >= lifetime is fulfilled and dictates them to be destroyed.

first, we need an additional variable after the class declaration:

```
public float fadeTime = 3f;
```

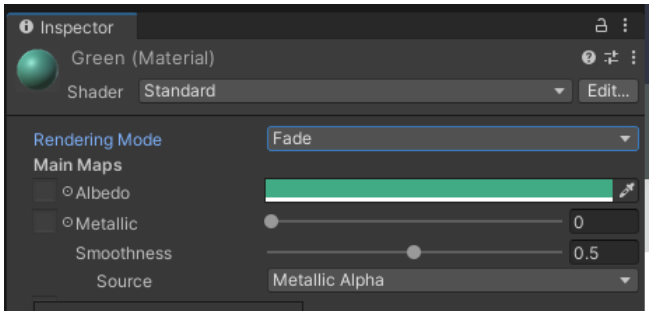
Then we'll add a coroutine with IEnumerator, add the following block after the last closing bracket } of the `Update()` function:

```
IEnumerator FadeOutAndDestroy() {
        Renderer renderer = GetComponent<Renderer>(); // get the renderer component
        Color currentColor = renderer.material.color; // get the current color of the material
        currentColor.a = 1f; // set the alpha to 1 (opaque)
        while (currentColor.a > 0) {  // loop until the alpha is 0 (invisible)
            currentColor.a -= Time.deltaTime / fadeTime; // fade out the alpha over time
            renderer.material.color = currentColor; // set the new color of the material
            yield return null; // wait for the next frame
        }
        Destroy(gameObject); // destroy the game object
    }
```

Now we have to make a modification to the `Update()` function. Instead of `Destroy(gameObject)` when the condition of the if statement `(age >= lifetime)` is met, we have to start the coroutine that will fade out the instance and then destroy it! Replace the if statement as follows:

```
if (age >= lifetime && destroyable)
        {
            StartCoroutine(FadeOutAndDestroy());
        }
```

> this will only work if Material is set to Fade in the inspector:



Revision #7
Created 10 April 2023 23:34:59 by Laura Wagner
Updated 22 January 2024 12:40:47 by Laura Wagner