While Loops

While loops are loops that will continue to run as long as the test condition is true.

I have used one in my game in order to space out the spawn time of my chickens.

while (timer != 0)

{

while (chickenCount > 0 && timer <= 0)

{

spawn = UnityEngine.Random.Range(0, spawnPoints.Length);

chickenClone = Instantiate(chicken, spawnPoints[spawn].transform.position, Quaternion.identity) as GameObject;

chickenClone.AddComponent<ChickenAI>();

chickenClone.AddComponent<ChickenHealth>();

chickenCount--;

timer = 5f;

}

timer -= Time.deltaTime;

yield return new WaitForSeconds(timer);

}

this is actually a while loop with a nested while loop. The outer loop condition is set so that as long as the timer doesn’t = 0 the inner while loop will be considered. The inner while loop has two conditions that need to be true for the code to run. The first condition is that the variable chickenCount has to be greater than 0, the second condition that needs to be true in order to run the code is that the timer has to be less than or equal to 0. As long as both of those conditions are true the code in the inner while loop will be executed. As soon as one of those conditions fails to return true the loop will stop running. Basically what this loop does is wait till the timer is 0 then it will spawn a chicken at a random spawn point and then subtract 1 from the chickenCount. It will then reset the timer so which will make the inner loop stop running and cause a delay before the outer while loop condition will evaluate as true again. This process will continue until the chicken count is 0. That will cause the inner while loop to no longer run which will cause the timer to no longer be reset and that will cause the outer loop to stop running. While loops are a lot like for loops but these don’t have a specific number of times they will run. While loops will run as long as their conditions are true.