Making 2D Games with Unity

Tutorial 3

Objectives:

* Creating a simple enemy controlled by game

**1. Creating an Enemy**

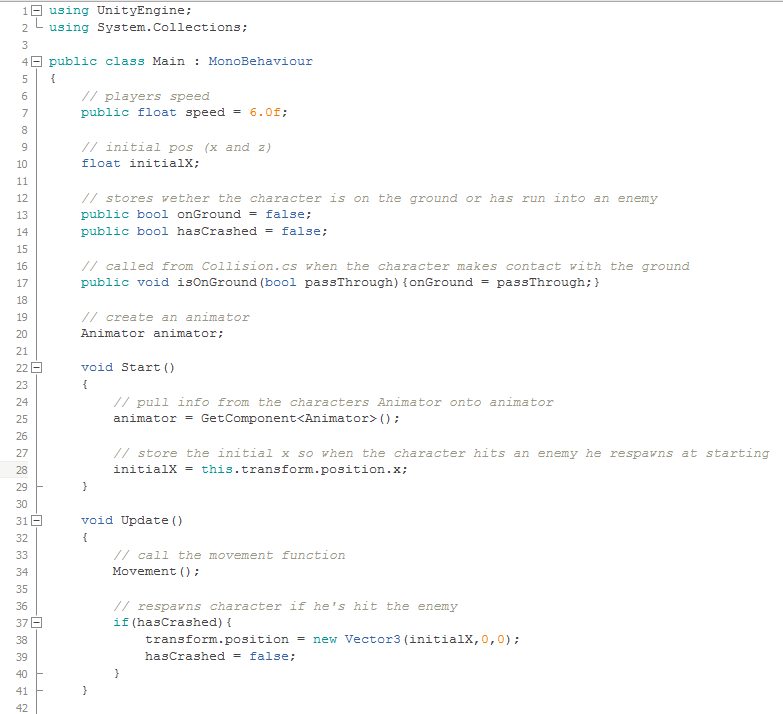
1. Starting where the last tutorial left off, we’re going to create and animate an enemy.
2. To create the enemy let’s first import the sprite map into the Sprite folder
   1. Project tab>Sprites>Right click empty space>Import new asset>Navigate to the sprite map.
3. With the sprite map imported change the import settings
   1. click on the enemy sprite map>import settings tab
      1. Sprite mode>Multiple
      2. Filter mode>Point
      3. Format>Truecolor
4. Click Sprite Editor>Slice>Type>Automatic>Slice>Apply
5. Now let’s add the sprite to the scene and add necessary components
   1. drag the enemy sprite map into the Hierarchy tab
   2. click on the enemy sprite from the hierarchy>Add component
      1. Add an “Animator”
      2. Add a “Rigid Body 2D”
      3. Add a “Box Collider 2D”
6. With the new components added change the following values
   1. under Animator>Uncheck Root Motion
   2. under Rigidbody 2D>Check Is Kinematic
   3. under Box Collider 2D check is Trigger
7. Head back to the Project tab>Sprites>enemy>expand by clicking the small arrow>Shift Click all of the animation states and drag them into the animation tab timeline. (Make sure the enemy is also clicked from the hierarchy tab or you’ll most likely get the main characters animations).
8. Once dragged name the animation “EnemyPatrol” and save it inside the animations folder.
9. Similarly to how we set up animations for the main character change the sample size to the size of the slices, so in this case there is 6.

**2. Moving an Enemy**

1. With the enemy implemented and animations created we must add collision detection and movement.
2. First let’s create a new script
   1. Project tab>Scripts>Right click empty space>Create>C# Script>Name it EnemyBehavior
3. Once created drag it onto the enemy gameObject in the Hierarchy tab
4. To get the enemy to move without being affected by the physics engine (by checking is Kinematic earlier) we can move it by simply changing its transform position. To do so this simply script oscillates the enemy back and forth via the mathf PingPong function. The reason we checked off “is trigger” earlier is to allow detection of collisions within the enemy. Once there’s a collision it changes the boolean hasCrashed in the main character script to true and the player respawns back at home.



1. Within the main character script let’s create that boolean hasCrashed and state what happens when the character has crashed with an enemy.





1. With the scripts updated and created let’s link the public variables
   1. Click enemy from Hierarchy tab>under inspector change speed to 5
   2. Click enemy from Hierarchy tab>under inspector click the small circle next to Main>Scene>Character
2. With the enemy being kinematic there’s no gravity applied to it. Therefore place the enemy where you want it to go, it won't fall onto the ground like the main character does.