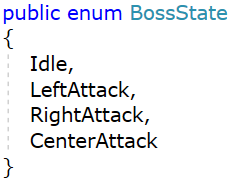
# Boss Fight Overview

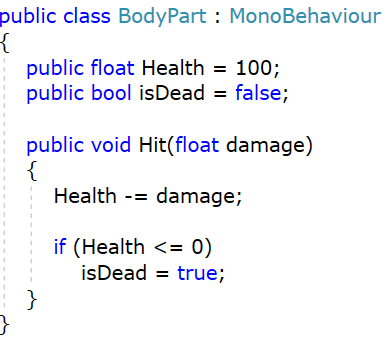
## Boss STate

* The BossState enum contains all the possible states that the boss can be in
* The boss should have different behaviours depending on their state
* We need to a way to
  + Update the state to be a new value
  + Check what the current state is and have the behaviours execute



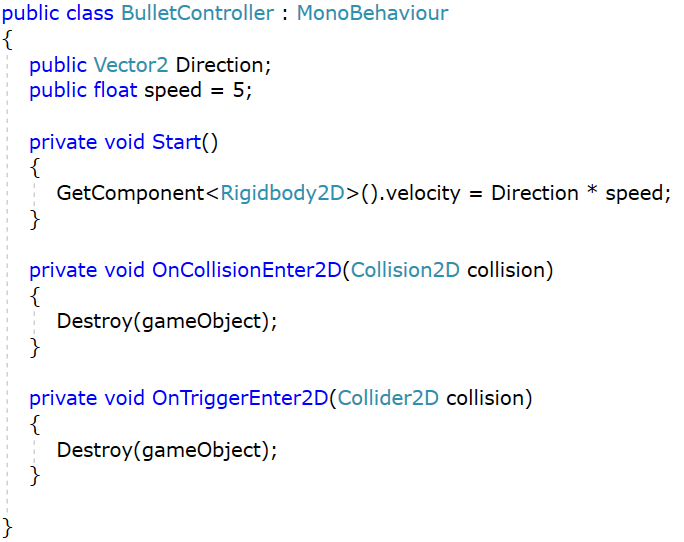
## Body Part

* A boss can be made of multiple body parts
* Each body part will have its own health
* In order to defeat a boss the player must destroy each body part
* The Hit method provides a way for the player to apply damage to body part
  + When the damage is applied we check if the body is out of health
  + If it is then we set the isDead variable to be true
* Our boss controller should handle the death of body part
  + Left and Right arms should stop moving
  + Left and Right arms should stop firing bullets

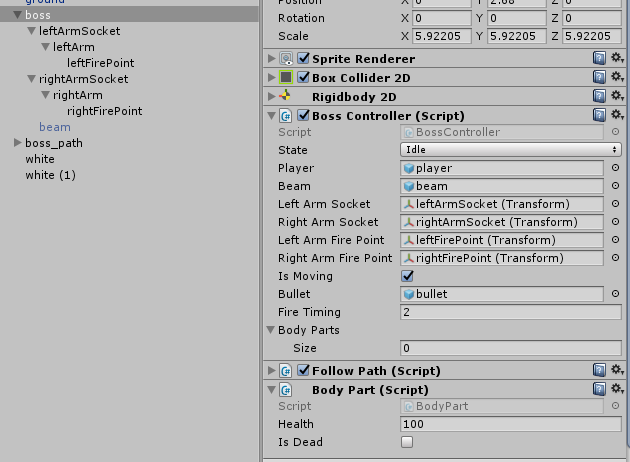
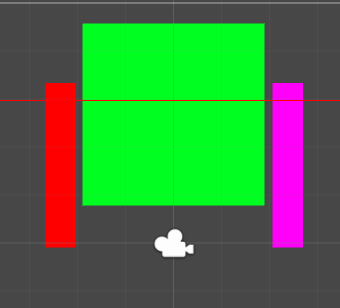


## Bullet Controller

* The bullet controller moves an object in a direction at a specific speed
* When the bullet is created it’s direction is calculated in the boss controller
  + The Direction variable is the set on the new bullet



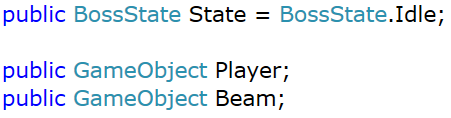
## Boss Controller



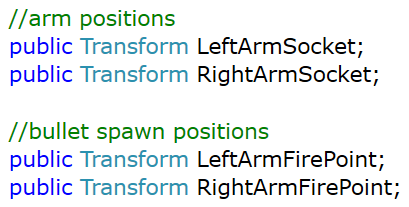
* The boss controller is where the majority of the game code is placed
* The boss needs to be able to
  + Check if all its body parts are dead
  + Update the state of the boss after a timer has completed
  + Fire bullets from a specific position
  + Handle when a new state is set (what needs to be turned on/off)
  + Control whether the boss is able to move
  + Update the arms to point at the player
* The boss has three objects attached to it
  + LeftArm
  + RightArm
  + Beam (turned off by default)

### Variables

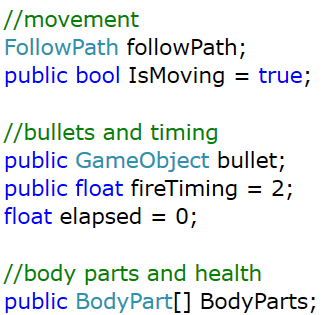
* State holds the current BossState of the controller. This will be used later to decide what the boss should be doing
* The Player holds a reference to the player. This is used later for determine where the arms should be pointing when tracking the player
* Thea Beam hols a reference to the beam object which is attached to player



* These four transforms hold references that are used to
  + Move the arms
  + Locations of where to spawn bullet from

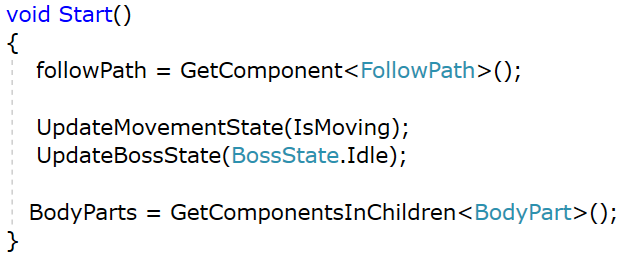


* FollowPath is where you want to the boss to move around while the game running
* IsMoving controls whether the boss should be moving or not
* Bullet spawning is based on Semester 1 code
  + After a number of seconds (fireTiming) spawn a bullet (bullet)
* Each part if the boss has a BodyPart script. This script tracks the health of each individual body part attached to the boss
  + We don’t know how many body parts the boss might have
  + We have an array to hold all the BodyPart scripts
  + When the game starts we need to fill this array with all BodyParts attached to the boss



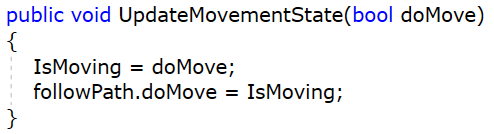
### Start

* When start is called
  + Get the Path that the boss is going to move along
  + Set the boss to start moving (UpdateMovementState)
  + Set the starting state of the boss (UpdateBossState)
  + Find all body parts scripts that may be contained in objects attached to the boss



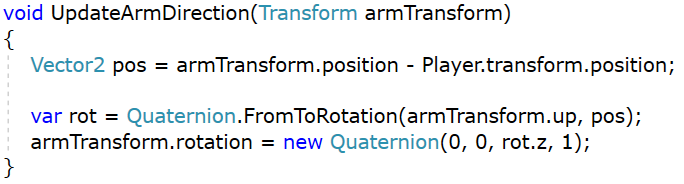
### Update Movement State

* Hold the current movement state of the boss in IsMoving
* Toggle whether the boss should be moving along its path
* **We could remove IsMoving from the boss and just use the CustomPath.doMove**



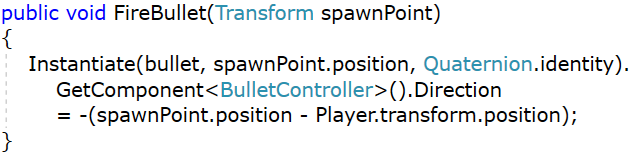
### Update Arm Direction

* Given a transform (position, rotation, scale of an object)
  + Get the direction between the arm and the player
  + Rotate the arm to point in that direction



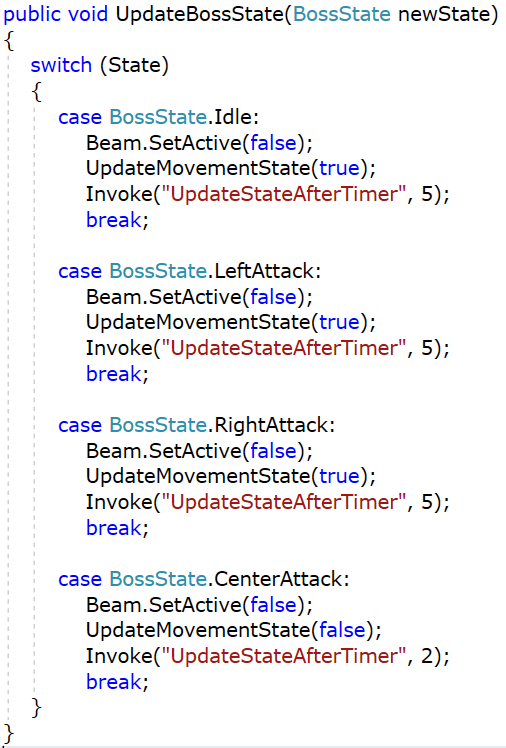
### Fire Bullet

* Given the transform of where we want a bullet to spawn from
  + Create a new bullet
  + Sets its direction to be the direction between the spawn point and the player
    - This could be modified to reuse the arm transform from above



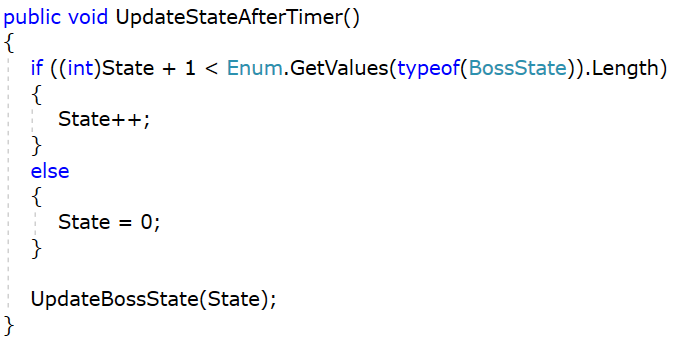
### Update Boss STate

* When a boss enters a state it needs to enable/disable parts it does not want (shooting/movement/beam)
* Each state last for a number of seconds
* After the timer for a state is up we move on to the next state the boss can be in



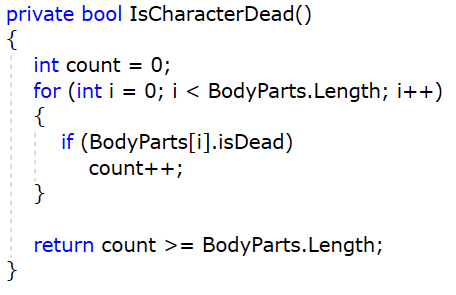
### Update State after Timer

* BossState is a enum
  + Enums are a named integer list
* If the next state is a valid one
  + The BossState enum has 4 values
  + If the current state + 1 is less than the total number of BossState
    - Increment the state by one (moves it to the next state)
  + Else we are at the end of the values in BossState
    - Restart the State back at zero
* Update the State with the new one



### Is Character Dead

* The boss has multiple body parts
* Each body part has a health variable
* The boss is defeated if all the body parts have a health value of zero
  + Loop over the BodyParts array
  + Check the isDead of each body part
* If the number of dead body parts is equal to the number stored in the body part array then we have beaten the boss



### Update

* Check the state of the boss (State) and od what is required for that stage
* LeftAttack
  + Make the left arm point at the player
  + Fire a bullet on a timer
* RightAttack
  + Make the right arm point at the player
  + Fire a bullet on a timer
* CenterAttack
  + Enable the beam object

