**Involved classes:**

* Weapon - This class uses most of the weapon settings contained in this class
* Projectile - Manages the projectiles used by the player's weaponry

**Player Weapon class**

// The following are set in Inspector:  
create int health  
create float hitTimer  
create float baseFiringCD  
create float weaponSwapCD   
create GameObject list weapons  
create GameObject projectile  
create GameObject gameOver

create int weaponChoice and set it to 0  
create float fireRateCounter and set it to 0  
create bool hasDied and set it to false  
create float hitCounter and set it to 0  
create Weapon list weaponList  
create Vector3 mousePos

Start procedure:

for each object in weapons, get their Weapon component and add it to weaponList  
Deactivate all GameObjects in weapons  
Execute SwitchWeapon(0)

Update procedure:  
set mousePos to the mouse cursor's screen position  
Rotate this gameObject to face mousePos

If left mouse button is pressed AND fireRateCounter >= weaponList[weaponChoice]'s firingCooldown:  
 set fireRateCounter to 0  
 Execute FireWeapon()

If right mouse button is pressed AND fireRateCounter >= weaponSwapCD:  
 set fireRateCounter to 0  
 Execute SwitchWeapon(weaponChoice+1)

add Time.deltaTime to fireRateCounter  
add Time.deltaTime to hitCounter

SwitchWeapon procedure:  
Parameter: int newWeapon

hide the GameObject weapons[weaponChoice]  
set weaponChoice to the mod between newWeapon and the length of weapons  
show the GameObject weapons[weaponChoice]

FireWeapon procedure:

if (weaponList[weaponChoice]'s isMultishot = true):  
 create float projDivision and set it to weaponList[weaponChoice]'s weaponArc/(projectileCount-1)  
 create float currAngle and set it to (this gameObject's rotation - weaponList[weaponChoice]’s weaponArc/2)  
 for (int i = 0; i < weaponList[weaponChoice]’s projectileCount; i++):  
 Execute CreateProjectile(currAngle)  
 add projDivision to currAngle  
else:  
 Execute CreateProjectile(this gameObject's rotation)

CreateProjectile procedure:  
Parameter: Vector3 spawnAngle

create GameObject newProjectile with prefab projectile at this gameObject's origin, rotated to spawnAngle  
in newProjectile's Projectile component, execute ProjectileSettings(weaponList[weaponChoice])

ReduceHealth public procedure:

if hitCounter >= hitTimer:  
 reduce health by 1  
 if health <= 0 AND hasDied = false:  
 set hasDied to true  
 activate gameOver object