Code Structure and Readability

Naming Conventions

Follow c# documentation standards: [C# Coding Conventions]

Public fields/variables/methods are PascalCase:

```
public int ResourceCount

public void GetResourceCount()
```

private methods/functions also use PascalCase:

```
private void CalculatePosition()
```

Private fields/variables use underscore then camelCase:

```
private bool _hasResource;
```

Constants and Public static variables are in all caps with underscore:

```
const float GRAVITY

public static float DISTANCE_TO_PLAYER
```

Local variables use camelCase

```
public void GetResourceCount()
{
        int currentResourceCount = 0;
}
```

Make sure to name variables and methods in a way that conveys what it's for. Do not name variables stuff like "c" or "variable1", be specific even if the variable name is longer/more cumbersome: "c" vs "currentResourceCount"

Accessibility

Avoid using public variables if you can. Instead use getters and setters or if you need to access a variable through the inspector use serialized fields:

```
public float Speed;

[SerializeField]
private float speed;

public float Speed { get; private set; }

private float speed;
public float GetSpeed()
public void SetSpeed()
```

Comments

Leave comments explaining what a method or confusing block of code does.

Prefer events over Update function

We should avoid running code on every frame if it does not need to be updated on every frame. Certain things like movement make sense to update every frame but other things like player health or ammo amount can be updated only when the player takes damage or when the player fires their weapon/reloads.

Use events instead.

Multi-use Classes/Components

A class should only have one function. For example the player controller should not also handle player health or the weapon logic. Player health should be its own component as should the weapon.