Coding Standards

Unity - Editor Version 2021.3.18f1

C# - [C# docs - get started, tutorials, reference. | Microsoft Learn]

- **Program Design**
  - **File use**
    - USE FULL ENGLISH – unless not possible or is commonly refereed to as its abbreviation over its full name.

      Ex. AI – Artificial Intelligence
    - No duplicate names of Files
    - Have Scripts tailored to their specified feature while keeping in mind Cohesion. A file meant for movement should not house functions for health effects and attack features.
    - If unsure if a function should be housed together, separate them or ask a team member for reassurance.

- **Naming Conventions**
  - **General names**

    Make names descriptive and tailored to their use. Please do not use variables named Temp, X, or any other unrelated nondescriptive name as it will make code scripts hard to understand

  - **File names/Classes and Structs - PascalCase**

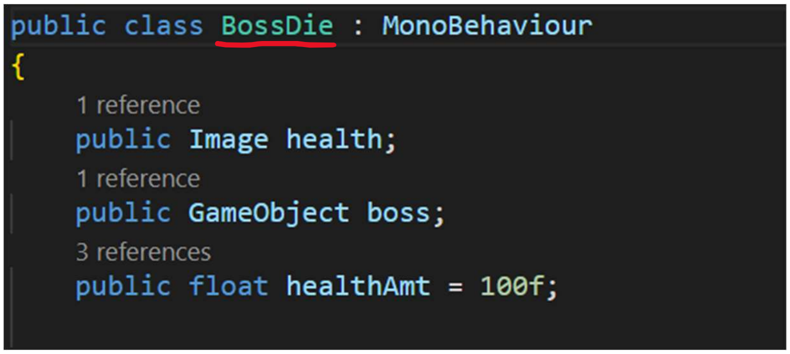
    When naming a file/Class the use of Pascal case will allow all files between Member's to be uniform and allow easier readability such as.

    Pascal case is a Capital at the beginning and first letter of each word that may follow.

    Ex. (BossDie.cs)

```
BossDie.cs
```

- File name



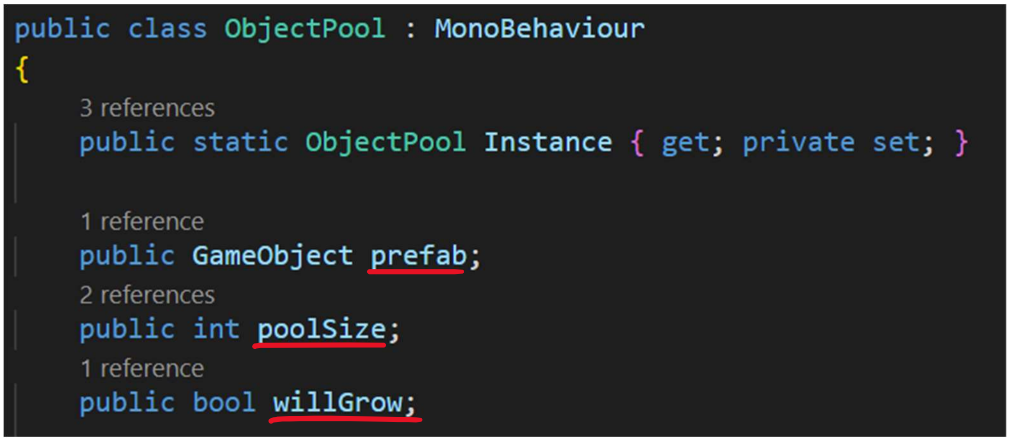
```
public class BossDie : MonoBehaviour
{
        1 reference
        public Image health;
        1 reference
        public GameObject boss;
        3 references
        public float healthAmt = 100f;
```

- Class

  - **Public, Protected and Private *Variables* - camelCase**

    This is also applied to function parameters.

    Ex. (ObjectPool.cs) - Lucien

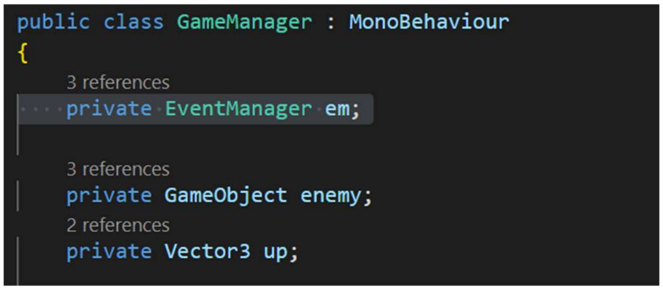


```
public class ObjectPool : MonoBehaviour
{
    3 references
    public static ObjectPool Instance { get; private set; }

    1 reference
    public GameObject prefab;
    2 references
    public int poolSize;
    1 reference
    public bool willGrow;
```

- Class and Variables

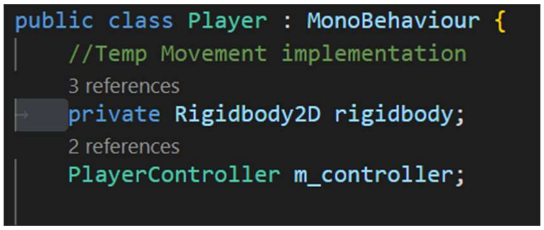
- **Formatting Conventions**
  - Ensure indentation are 4 spaces or Tab of size 4.
    - Set as base as it's the default within VSCode
      Ex. (GameManager.cs) – Noah



```
public class GameManager : MonoBehaviour
{
    3 references
    private EventManager em;

    3 references
    private GameObject enemy;
    2 references
    private Vector3 up;
```

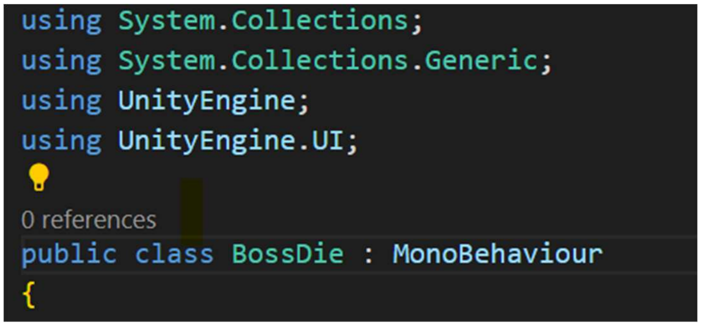
- example of spaces



```
public class Player : MonoBehaviour {
    //Temp Movement implementation
    3 references
    private Rigidbody2D rigidbody;
    2 references
    PlayerController m_controller;
```

- example of TAB of size 4

  - Place "using(s)" before class.

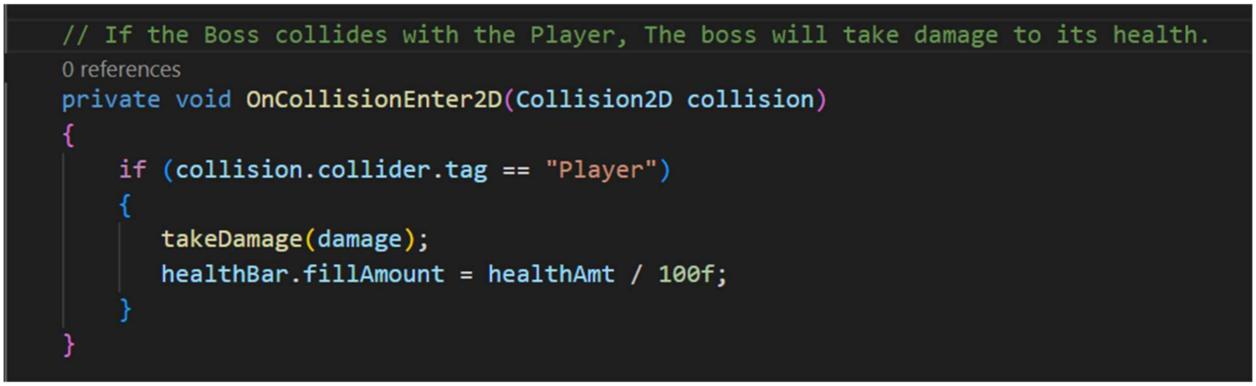


```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

0 references
public class BossDie : MonoBehaviour
{
```

- **Documentation**

- Comments
  - Place before a function and classes
  - Have a space between the start of a comment and the beginning of the sentence.
  - If a comment is long split, it onto another line.
  - Make comments brief while describing its purpose.



```
// If the Boss collides with the Player, The boss will take damage to its health.
0 references
private void OnCollisionEnter2D(Collision2D collision)
{
    if (collision.collider.tag == "Player")
    {
        takeDamage(damage);
        healthBar.fillAmount = healthAmt / 100f;
    }
}
```

- File Location
  - Place files in there appropriate location i.e: In your personal folder where items are grouped by type