# Anatomy of Unity C# Scripts

In Unity game engine, C# scripts are used to add logic and behavior to game objects. Here is a basic anatomy of a C# script in Unity:

## 1. Namespace Declaration:

A C# script usually starts with a namespace declaration, which is a container that groups related code elements, such as classes and functions, into a named scope. For example:

```
namespace MyGame {
    // C# code goes here
}
```

## 2. Class Declaration:

Within the namespace, the script should contain at least one class declaration, which defines the behavior and properties of a game object. For example:

```
5   public class MyScript : MonoBehaviour {
6       // C# code goes here
7   }
```

## 3. Variables:

The class may contain variables, which are used to store data that can be accessed and modified by the class's methods. Variables can be declared as public, private, or protected, depending on their accessibility. For example:

```
11      public float speed = 2f; // speed of movement
12      public float range = 3f; // range of movement
13      private Vector3 targetPosition;
```

## 4. Methods:

The class may also contain methods, which are functions that define the behavior of the game object. Methods can be declared as public, private, or protected, depending on their accessibility. For example:

```
33      public void Move() {
34     // C# code goes here
35      }
36      private void Attack() {
37     // C# code goes here
```

## 5. Start and Update Methods:

Two special methods that are commonly used in Unity scripts are the `Start()` and `Update()` methods. The `Start()` method is called once when the game object is created, and the `Update()` method is called once per frame. These methods can be used to initialize variables, update the game object's position, or interact with other game objects. For example:

```
15      void Start()
16      {
17          targetPosition = transform.position + Random.insideUnitSp
18      }
```

```
    void Update()
{
        // move towards target position
        transform.position = Vector3.MoveTowards(transform.positi
```

## 6. Event Functions

The class may contain event functions, which are special methods that are automatically called by Unity in response to certain events, such as collisions, triggers, or input. For example:

```
24      void OnTriggerEnter(Collider other) {
25       if (other.gameObject.tag == "Enemy") {
26        health -= 10;
27      }
```

This code will decrease the game object's health by 10 if it collides with an object tagged as "Enemy".

---

Revision #1
Created 10 April 2023 23:45:23 by Laura Wagner
Updated 11 April 2023 00:00:21 by Laura Wagner