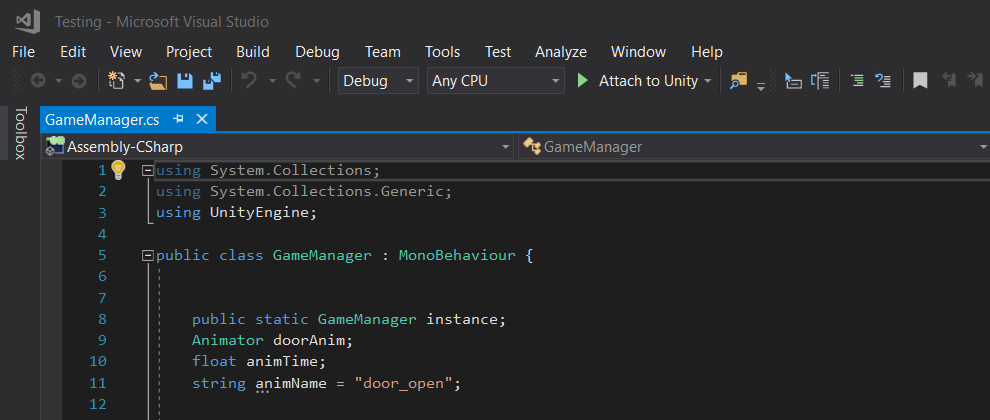
**Exit Door Package Doc**

*Game Manager Script*

The Game Manager script will serve as a central hub of information for the game. In more traditional way you might use the Game Manager script check player’s score, how long they have left to complete a level and so on. In this case we are going to use it to keep track of objects that we have to reference often such as the door. Let’s assume that every puzzle we create will have an end goal of opening the door. Since the door is a separate object, we’ll have to create a variable to hold this animator and we have to write functions to trigger that animation to run, the same things over and over again. It is much more convenient to store these methods and values in one place and then reference them as necessary. In order to be able to reference our Game Manager globally we going to make it a singleton. (**Singleton** is a design pattern in programming that allows us to ensure that only one instance of our class ever exits at one time and can be accessed from any other script) This is perfect for a Game Manager, which we definitely only want one of. Lets,

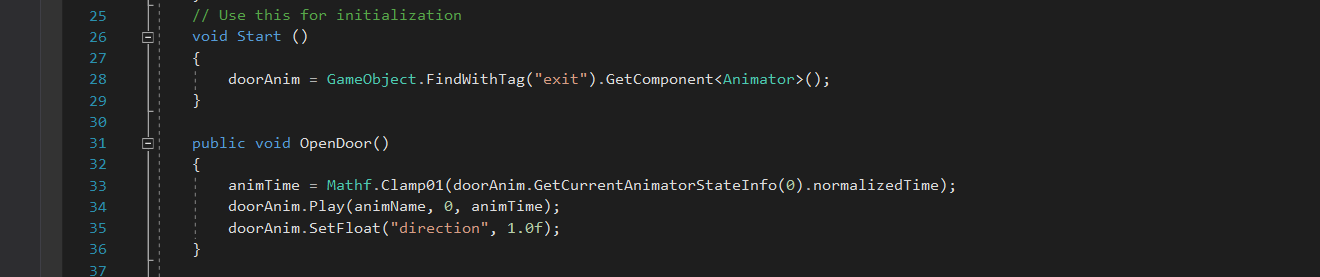
Create > c# Script > Name it Game Manager



First, we make a reference, as in other script we are going to call the Game Manager by GameManager.instance. Making a reference to the Animator, adding a float number to animTime, Referencing the animation as we have already named as “door\_open”.



Void Awake method is to check whether we have anything else in our game world that this game manager script is applied. From the “if” statement, it means if there is anything has this instance associated with it, we want to destroy that game object.



In “start” method, assigning the door animation created to the door object.

Door opening functionality,



Under the void “update” method, we are writing a “if” statement, if the key “A” pressed, calling the open door functionality.