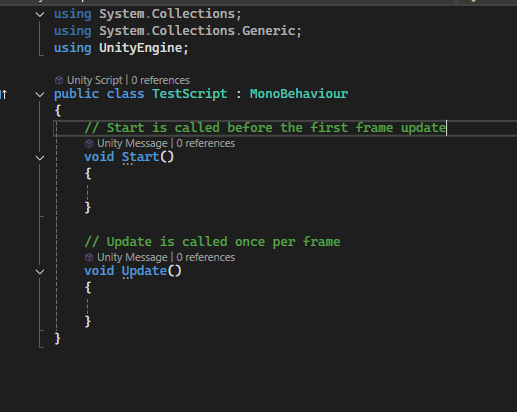
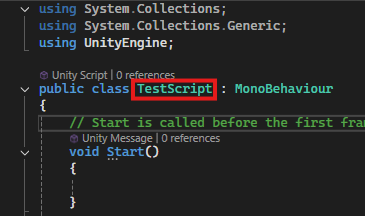
So in the last lesson we have learned about what is scripting in Unity and we have also learned How we can actually create new script And we have also learned various ways to attach those scripts, to various game objects. So the last lesson we have already created a script and we have named it best script and we have named it TestScript and we had opened it in VSStudio code. So in this lesson what we are going to do be doing is. We are going to be learning the code that unity enter in our script by default as shown below



So we're going to be learning what these things mean and why this unity includes it by default and what will happen if we remove those things. So one thing you have to look at is whenever you are creating a new script, what happens is you give your script a name as an example Whenever we created our first rate, we give it a name of Testscript. OK, so creating a script in unity basically means creating a new class. So whenever you create a new script by your name, Unity create a class by that name. as ma



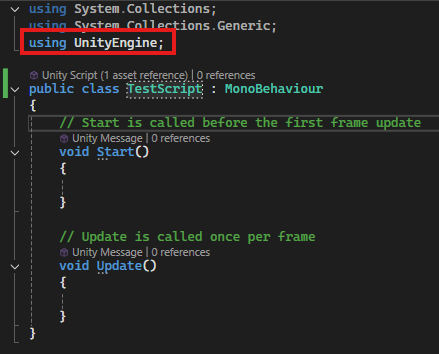
We'll learn more about classes later on. But as of now, you can understand that in since the shop is an object oriented programming language. But as of now, you can understand that C# is an object oriented programming language. So classes are the very basic things of object oriented programming. So whenever you create a new script, it creates a class for you with the same name. And if you try to rename this class to something else, you will see some error but not now in modern unity there is no errors are shown now and now it is acceptable.

**Discovery:-**

I think in older version Unity used to find the script that is attached to the object by using the object name but it does not need it maybe because coding logic slightly changed(who knows)

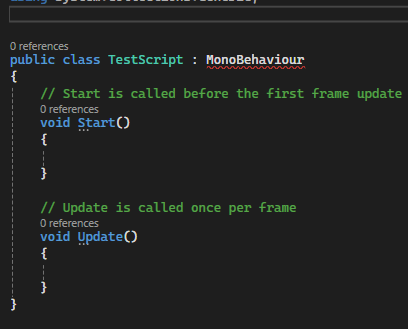
3:00

Whenever you create the script the very first line is that using UnityEngine as marked above



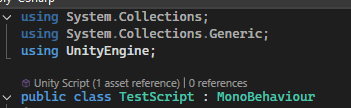
And also using System.Collection These things are what we call namespaces So in C# there is a concept of namespace So if you if you have to use codes from certain namespace, you have to include that namespace or you have to use that namespace at the very beginning of your code.

So in our code we are using UnityEngine . So that basically means we can use all the things that are defined inside this unity engine namespace. And most of the things you're going to use are actually part of this unity engine namespace. In object oriented in object oriented programming there's a feature called inheritance. And as you can see we have public lass TestScript and then colon and hten MonoBehaviour it mean this class inheriting from this MonoBehaviour This model behavior is the base class for everything in unity. OK, so if you want to use start and update functions and anything. Most of the things, not just the things, all of the things are actually inside this MonoBehavior class. So we are inheriting from the MonoBehaviour class . That basically means that we can use anything that is inside this MonoBehaviour class. And this mono Behaviour is also part of this unityEngine namespace. So if we remove this use unityEngine then MonoBehviour class will show error as shown below as there is read line below MonoBehaviour

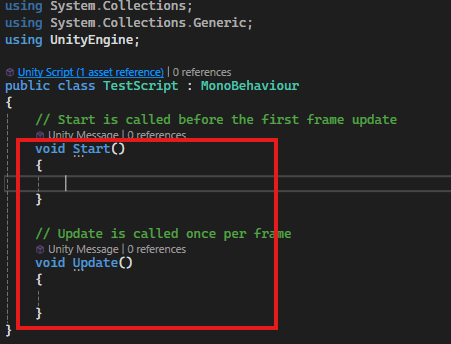


5:37

Because unity is not able to find this mono behaviour which is inside this namespace which is inside the UnityEngine namespace. So if we want to use anything from this unityEngine namespace, we have to use that at the beginning of our code. So if we written it in top then the error will be removed that is red line is removed from MonoBehaviour as shown below

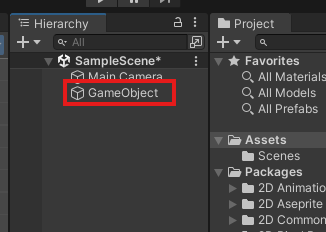


So these things are called namespaces OK, and this thing beside Monobehaviour is called the class that we are created. And this thing is called Mono behavior from which everything is defined, we've got everything from here. And this is something called inheritance. We'll learn more about that in later lessons. Teacher just want to let us know why what you are doing and what unity has included the very first time you got a new script So then what you see is after the name of the class and the name of the monobehavior, you can see two curly braces and our class ranges between these curly braces. So inside this we have the body of the class OK, so everything is going to do is we're going to include them inside this body of the class. And we have two more things. These things are called functions. Such as start method and update method as marked below



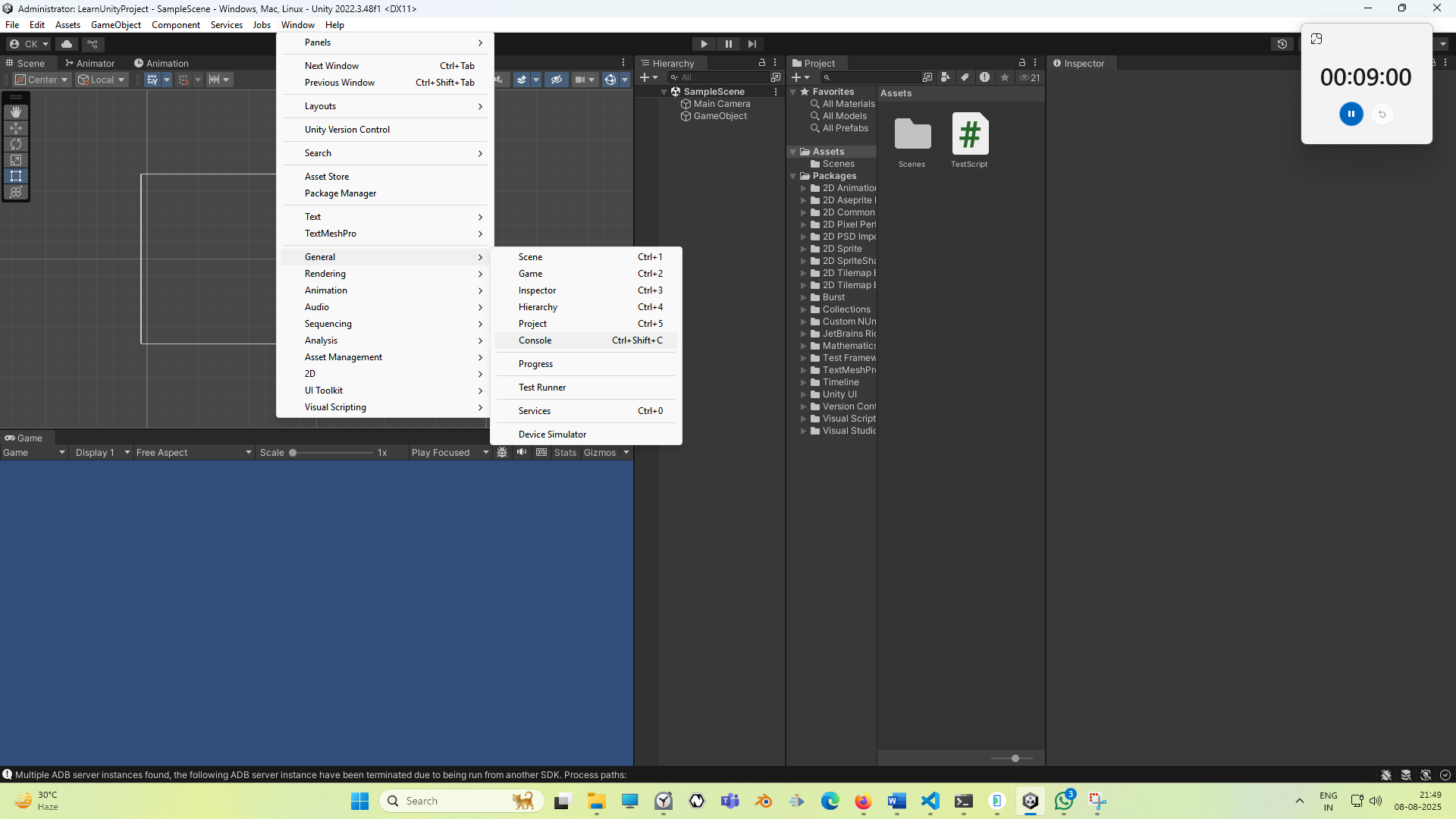
So when you create in the script, unity by default includes these two things in your script. If you want, you can delete these things. But these things have some specific rules in your code and if we don't inherit from mono behaviour, Unity will not be able to do proper things with these method. OK, since these things are also part of this mono behaviour class. As Teacher have said before, most of the things you can do are actually part of this mono behaviour class. Okay so this start function is what gets called at the start of the script and this update function is going to be called in each and every frame. Don’t worry we will learn about this start and update methods in later lessons. OK, so. We have pretty much learned the basics. We have learned what this namespace means, you have learned about this class. We have learned about this inheritance. You have learned the body of the class, and you have learned about these two methods that are included by default. So we have created this scripts but this script is not working right now. So in order to make this script to work, you have to actually attach the script to some game object. You can attach it to anything And if you only attach this to something, only after that it will work. So lets create a new empty gameObject

Now it is added as marked below in the hierarchy panel as marked below

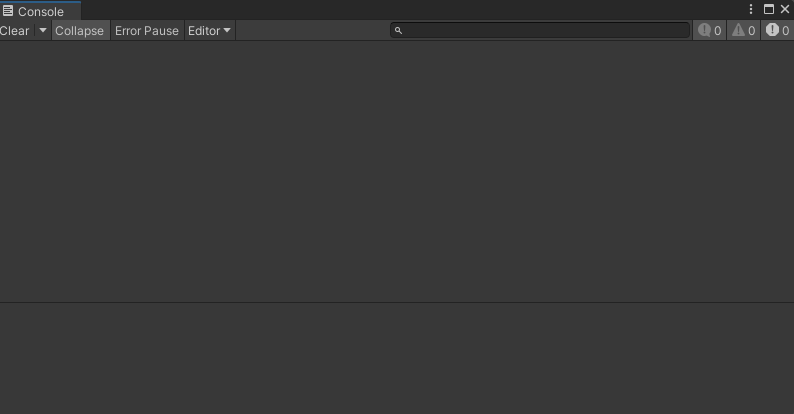


8:56

And if I go and attach this script with it now this script is in a position to work. So now we can actually you can actually control the behaviour of the empty game object using this script. And we can also we can also change the behaviour of the complete game by using this script in this game object. let's go write our own very our first simple code SO in the start method we write Debug.log and print the message Start. So what these things mean is this thing means that we are actually trying to log something and we are printing that inside the console. So this helps us in debugging our code later on. So if you do this, what will happen is, as Teacher have said before, the start function gets called at the very first time you create a script and run your game. So when you run your game, the start function gets called and this Debug.log happens. And that is why the start string gets printed on our console.

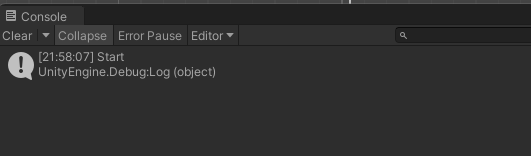


So open the console by going to window and then to general and then to console and its shortcut key is **ctrl + shift + c.** The console window look like below



you will see this thing that start string gets printed if we play the game by click in on play button

Now we click on play we will see Start in console as shown below



since the start function getting call at the first time, so this Start string is getting printed here.

10:46

Now you might have understanf these basic thing mean the unity includes very first time you include a script it's actually very, very important to understand what unity includes in your script before starting to learning writing your own code, because this gives you a basic understanding of what you are doing and why you are doing that, which is very, very important in computer programming. Teacher think this is it for this lesson . In the next lesson we will start writing our own code So see you in the next lesson.