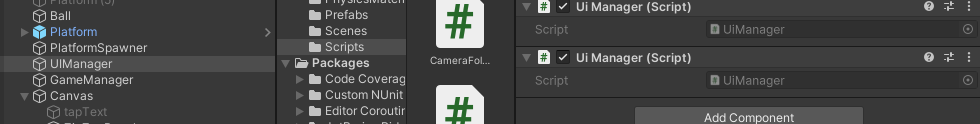
So in this lesson we will start creating our UI Manager

So let’s get started

Now we can see UI manager in our Heirarchy



We have also attached UI Manager script to out UI manager

Now we double click on the script to see the script opened in the editor

So in the script we want all the reference of all UI IOA which you want to activate and deactivate

So we will create few reference in UI Elements

So we add a reference for ZigZag panel (and will drag and drop obviously)

And we will set its animation and will play it

Now we will add a reference to gameOver Panel

Since we need to enable that animation should play

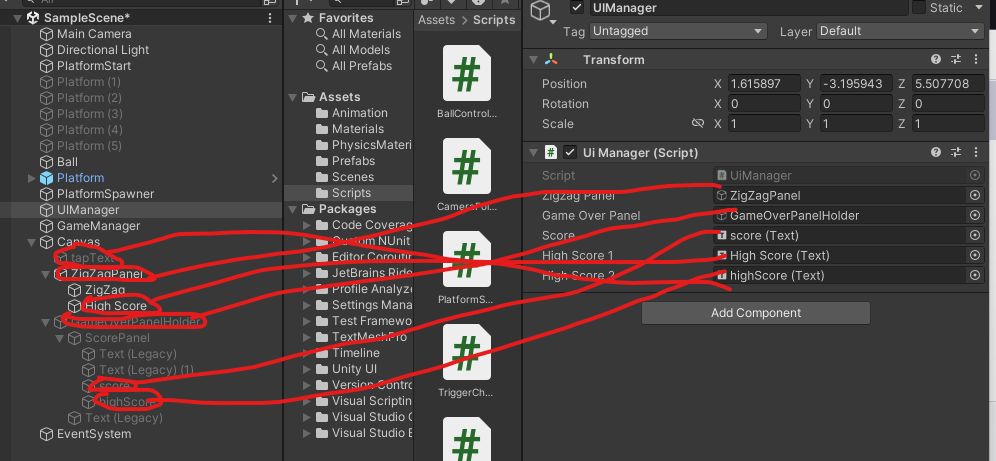
Before referencing any script add a dependencies other wise we will not be able to refer any UI element (related class like Text(maybe))

So now we add reference to the score that is of Text type

Through this we can change the score any time

Then we need a text reference to the highScore

And text reference to the another high score(who knows)



Now reference will be like above

Now in UI manager script

We are going to create few public function and we are making it public so that other script can also use it

We name it as GameStart

So it will just do everything when a game start

And it will just do everything when it needs to do when the game start

We create another function called gameOver

So this will be called whenever the gamover happens

And if the gameStart what you gonna do

One thing more we need

is tapText we will disable it when the gameStart

So go back in Unity and drag and drop it

So now when the gameStart then in GameStart function what we need to do is make the tap Text

Not visible by using method setActive and pass it false to it Now our code will look like below

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

public class UiManager : MonoBehaviour

{

public GameObject zigzagPanel;

public GameObject gameOverPanel;

public Text score;

public Text highScore1;

public Text highScore2;

public GameObject taptext;

// Start is called before the first frame update

void Start()

{

}

public void GameStart()

{

taptext.SetActive(false);

}

public void GameOver()

{

}

// Update is called once per frame

void Update()

{

}

}

4:05

Now we need to start the animation of the zigzag of moving up when the game plays but as we know zigzag panel is attached to new state that is static in nature so we in game Start function with the help of getComponent we get Animator component and use play method and pass the animation name that you want to use that is attached to our zigzag panel so now the code will look like below

public void GameStart()

{

taptext.SetActive(false);

zigzagPanel.GetComponent<Animator>().Play("panelUp");

}

So whenever the gameStart function is called then the code associated with it will be called

Now when the gameover happened we make the gameOver Panel enabled inside the GameOver function

public void GameOver()

{

gameOverPanel.SetActive(true);

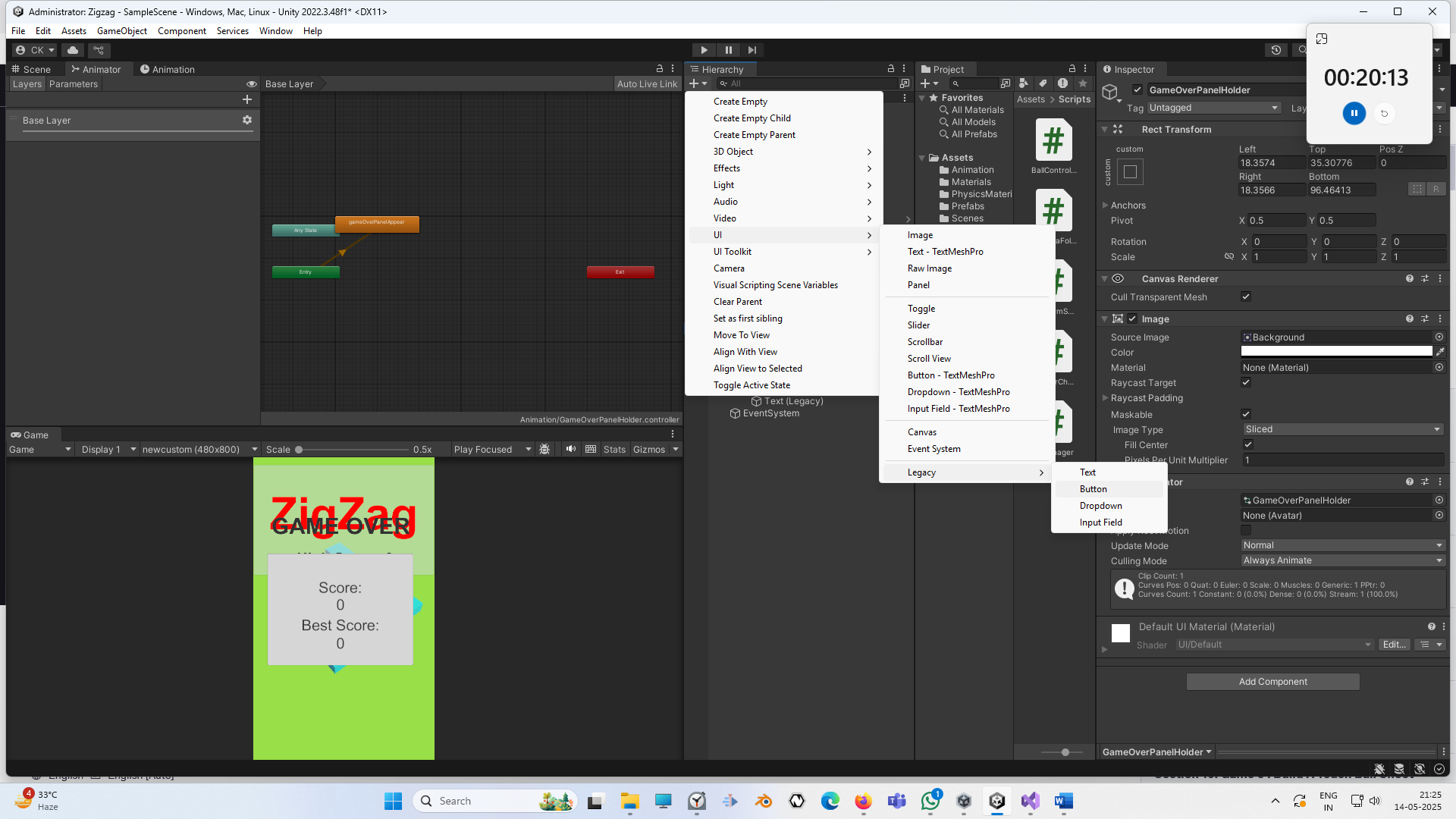
}

And the animation will be automatically as it become enabled become animation is not set to any new empty sate in animator panel

One more thing we need to do is to create a reset button

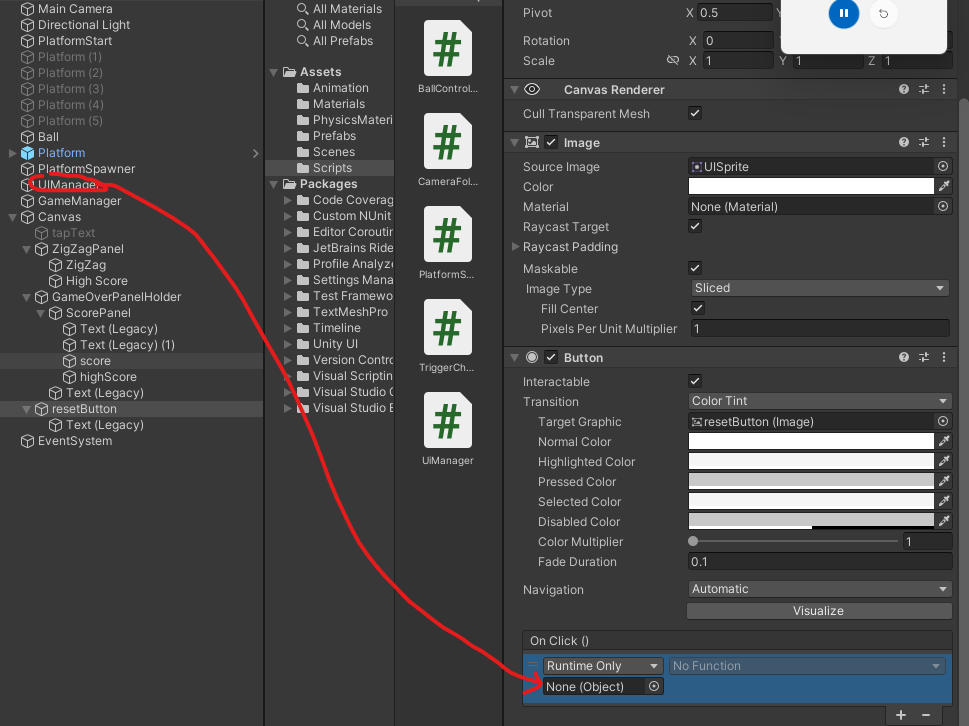
We enable Game over panel and then inside it we create a reset button inside it

By selecting it and then go to + and then to UI and then to Legacy and then to Button as shown below



And rename it resetButton and then make the font text of text inside it 40

Now select the reset button and then go to onClick property in inspector and then add evetn handler so that we can handle click on the button otherwise it cannot handle the click on it



So click on the + button and then drag and drop the UIManager IOA on the above marked field

As the UIManager IOA contains the necessary code that we want to run whwnever this button is clicked

Now we go to UIManager file and then add a public function reset

What this reset function will do

It will start the game from the beginning

So we need to reload the scene from the start

Whenever we reload the scene we use dependancy UnityEngine.Scenemangement

8:27

After unity 5.3 you need to always use this dependency

we use this to do any type of screen management

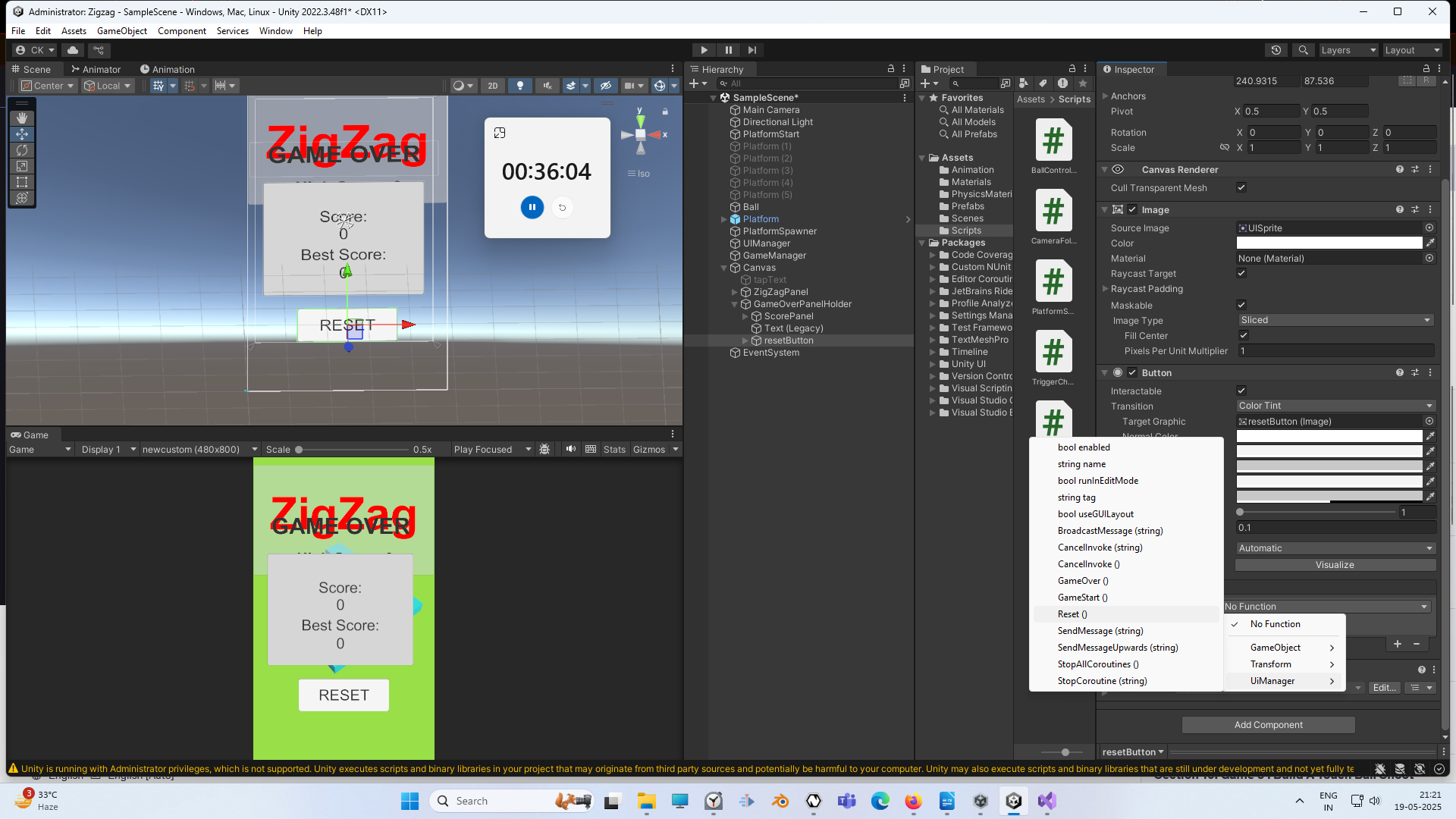
Now we reload our game scene by

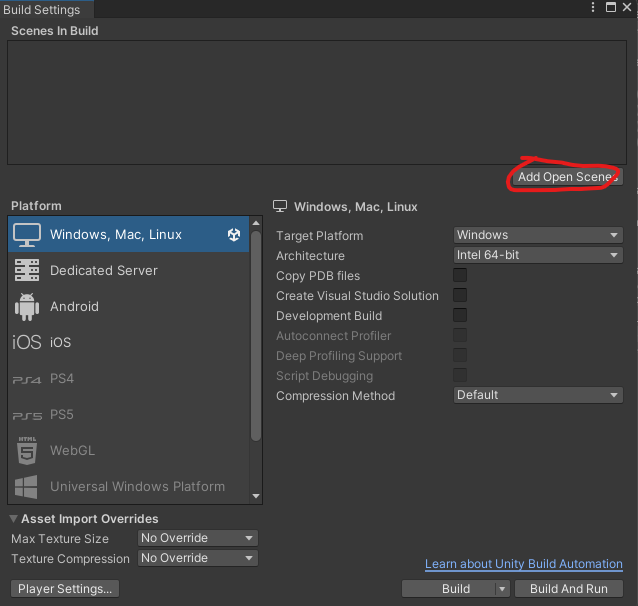
SceneManager.LoadScene()

In parameter tell the index of the scene you want to load

Since we have only one scene here we pass it zero

we can also pass the name of the scene but for our ease set it to index

Now inside the onclick field of the reset button set the function by clicking on the no function and you will see a submenu as shown above and then go to UIManager and then to Reset() option



Now we open build setting and then click on Add open scene as marked above

So this will add current scene to the build only then we can load this scene when the game will be built

You can see this scene indexed as 0

We again disable our gameOverHolder

So now when the gameStart function is called then

TapText is disabled and then play the ZigZag’s panelUp animation

Now what we need to create static instance of the UIManager

and we need to make it a Singleton Pattern

What is Singleton pattern?

It means there should be only one instance of this UI Manager

While you may ask why we need to do that?

If we do make this an UIManager Singleton then we can access any function or any public variable of this UIManager from any other script and that would be very helpful

Now in our UIManager script we add a public variable that is static and class UIManager we name this variblae as instance

public static UiManager instance;

If from any other script if we want to access this UIManager we just need to write UIManager.instance and then we can access naything from here

Now we write another function that is called Awake function

This is called Awake function and it is called before start and before starting the game So we need to set up everything here

Now inside we set it by

Adding an if and we check whether the instance variable is null if it is then we set iut to the current instance that is created by initialising it with this

So next time awake is called(who knows) then since instance will not be null then it will not be intialised again with this (or any more instances who knows)

So there will be only one single instance of this UI Manager

That why it is called Singleton pattern.

12:30

Now if we want to access this UIManager in any other script we just need to write UIManager.instance

Now we can access it anywhere we want

So this is how we created our UI Manager

Next we are going to create our game manager and our score manager and we are going to access it anywhere and make our game run

So see you in the next lesson