In this lesson we will learn how to make a simple Android game using Unity Game Engine

Now, if you have been following Teacher for quite a while, you know that I have made a video on this topic

in 2017, 18, 19, and even before that.

So this time you're going to make a new and updated video.

So before starting lets look at what you gonna be building at the end of this video.

AS Teacher is showing the intro screen of the game I cannot show as udemy does not allow screenshot for this course

The intro is showing press any key to start the game

Now a game is visible where we hit the ball with pedal after it get returned by the wall and whenever we hit the ball then score is incremented by 1

Now as you can see as the ball goes down the game start again so this is the core function of the game and we are going to built it from scratch

So lets get started

OK so now lets start downloading and installing the tools that we need to build our Android game.

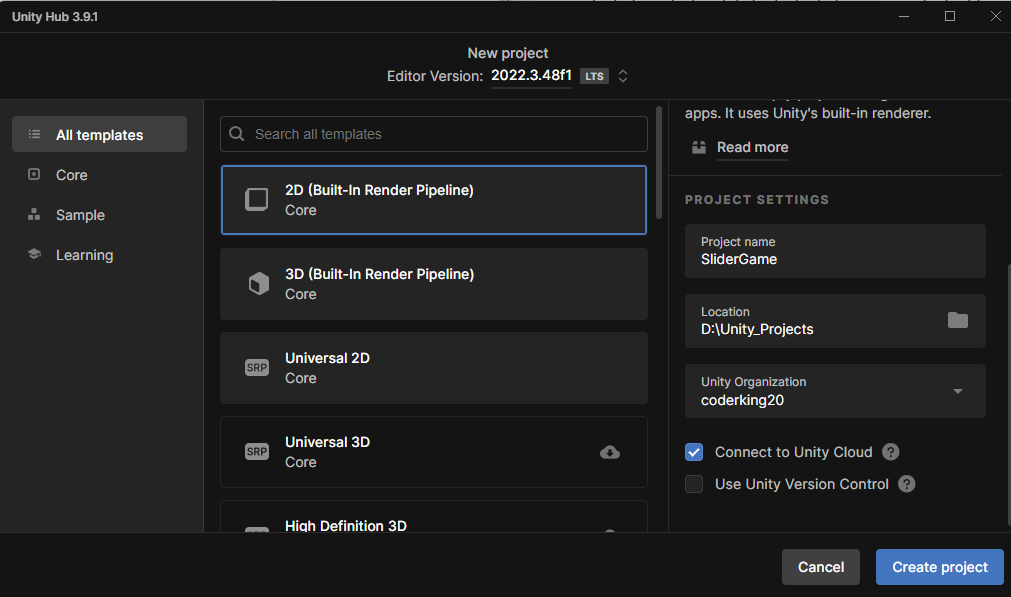
So now as always we are going to create our game using the Unity Game Engine so we can go to Google and search Download unity. I think the teacher is going to Download and install the tools I think the teacher is again going to show how to download and install Unity so we will skip some part of the lesson

**Note:-**

Before a few years ago, it was really hard to set up all these things( components that we install with our editor ). Now all these things come internally so that so that you don't have to do anything from the outside ( download and install component manually ).

Now this time we are going to make 2D project

The configuration for our project



Click on create project button to create our 2d project

Teacher prefer 2 by 3 layout

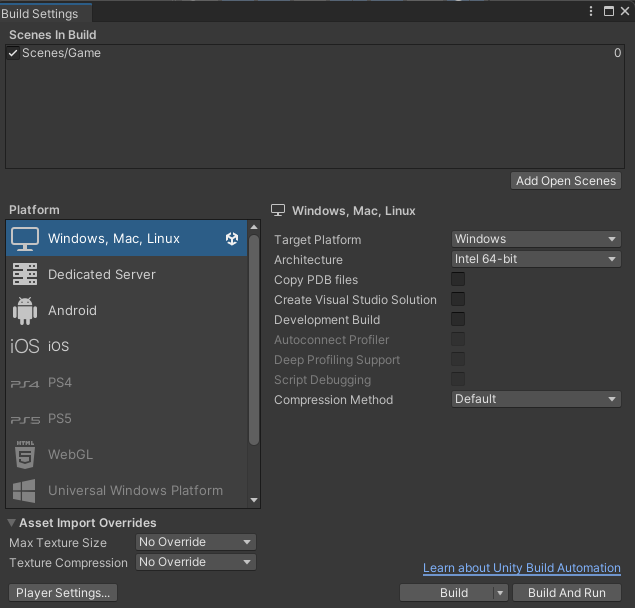
So first thing you need to do is save the scene So file -> save

The time the teacher is in here we don’t need to create scenes manually we get one by default (it mean he is somewhat near to the current modern features)

Now we rename the scene Sample Scene to Game in the scene folder inside asset folder in the project panel

6:30

Now we open our build settings



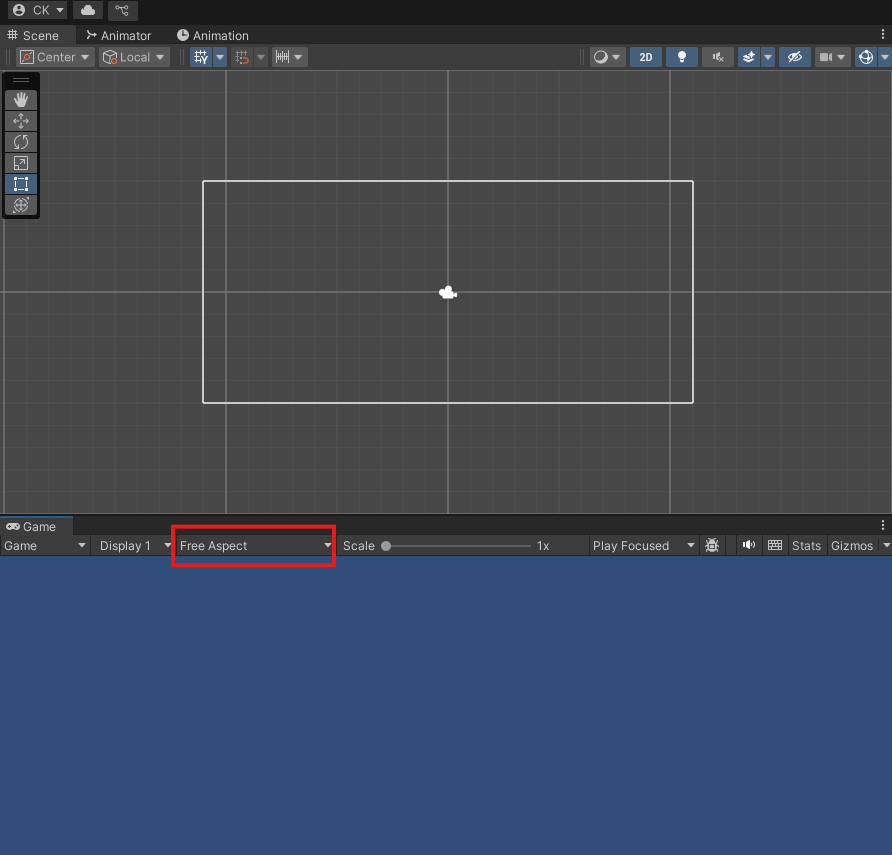
Our current platform is Window, Mac, Linux standalone but we were to build for android devices

If you have the Android version installed or the Android component installed while installing Unity then you will not be able to do this.

Now you know what you need to do if you have reached this part of the course then you know what to do.

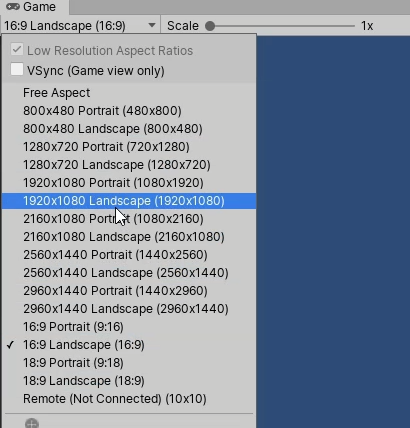
So we switch to the Android

Now we need to select the aspect ratio for our game

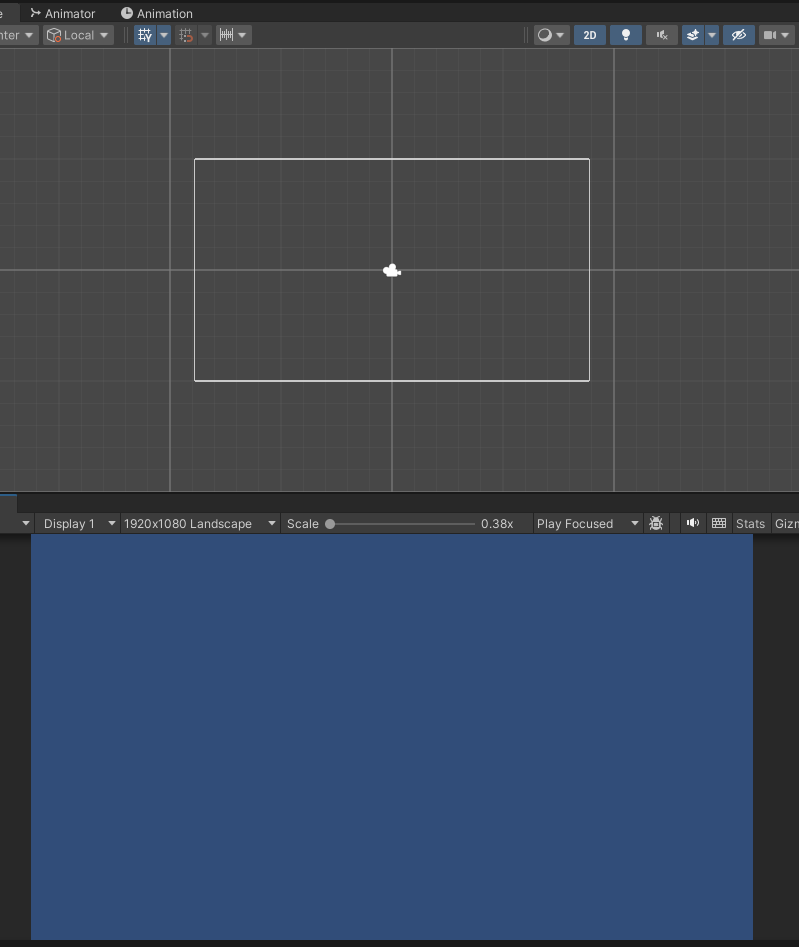


AS you can see in the marked above that the aspect ratio is Free aspect

Now click on free aspect to 1920 X 1080 Landscape(1920 X 1080) as highlighted below



Now the game and scene panel will look like below



7:40

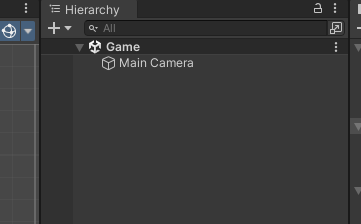
Real knowledge for making game starts from here

Now we need to start working on our game So inside the assets folder, we're going to keep all our assets are all our project elements like sprites, audios and scripts and all these things here.

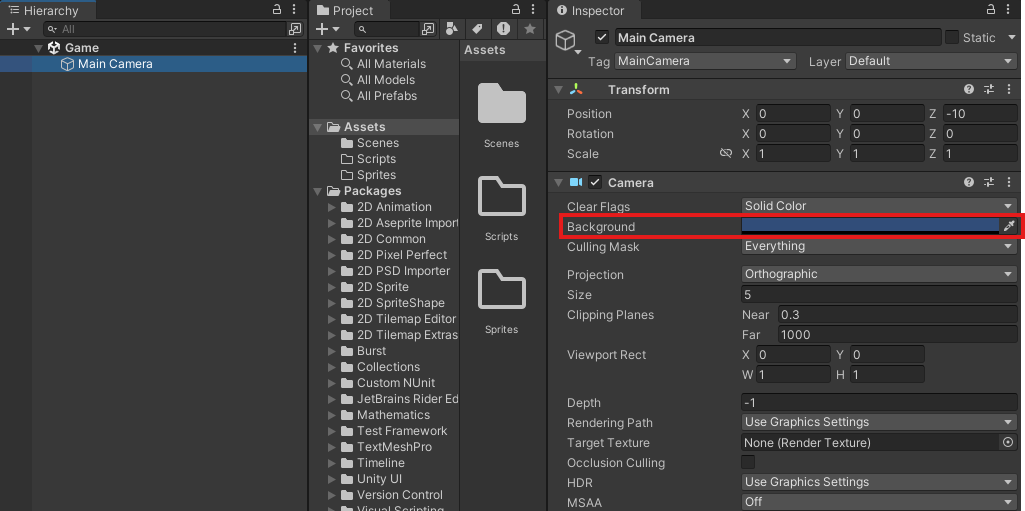
So we need different folders to organize these things So here I'm going to create a new folder and I'm going to name it Sprite. We are going to keep sprites for our images.

We can also create a new folder and will name it scripts. Where are we going to keep all our scripts

Now in our hierarchy panel we have our Game scene inside which all the element of the game are present



Currently we have this main camera so we select the main camera so we can select and in the inspector you can see its properties



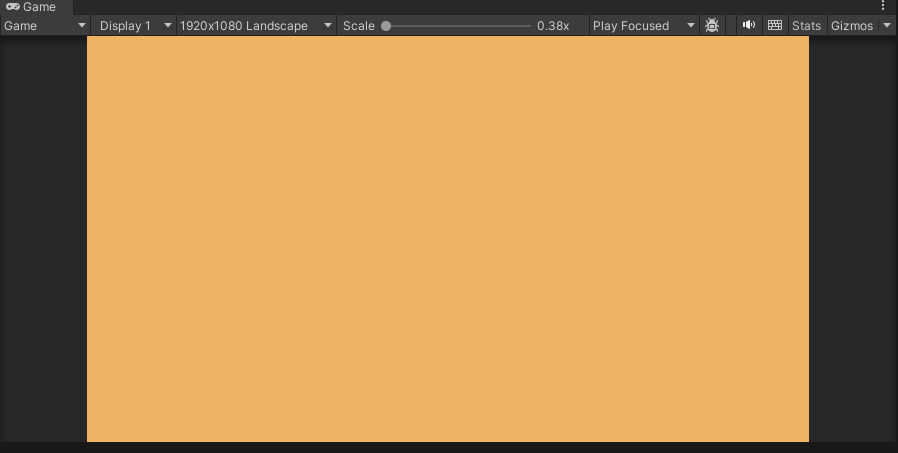
In the background as marked above here you can change the color of the background means the background you see in the screen of the game.

You can see that the clear flags is set to Solid Color ( I think it is telling the type of background )

Now set the color picker as shown below



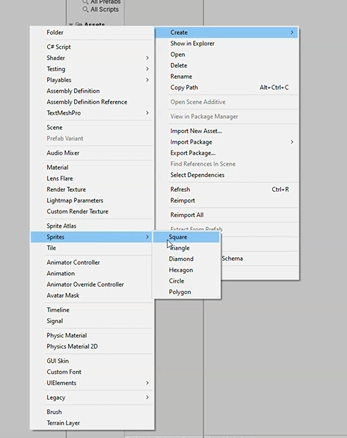
Now our game bg color look like below



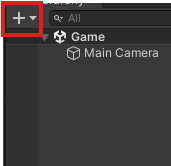
Ctrl + s to save you scene and file as well as (who knows)

So now we are going to draw boundaries for our game

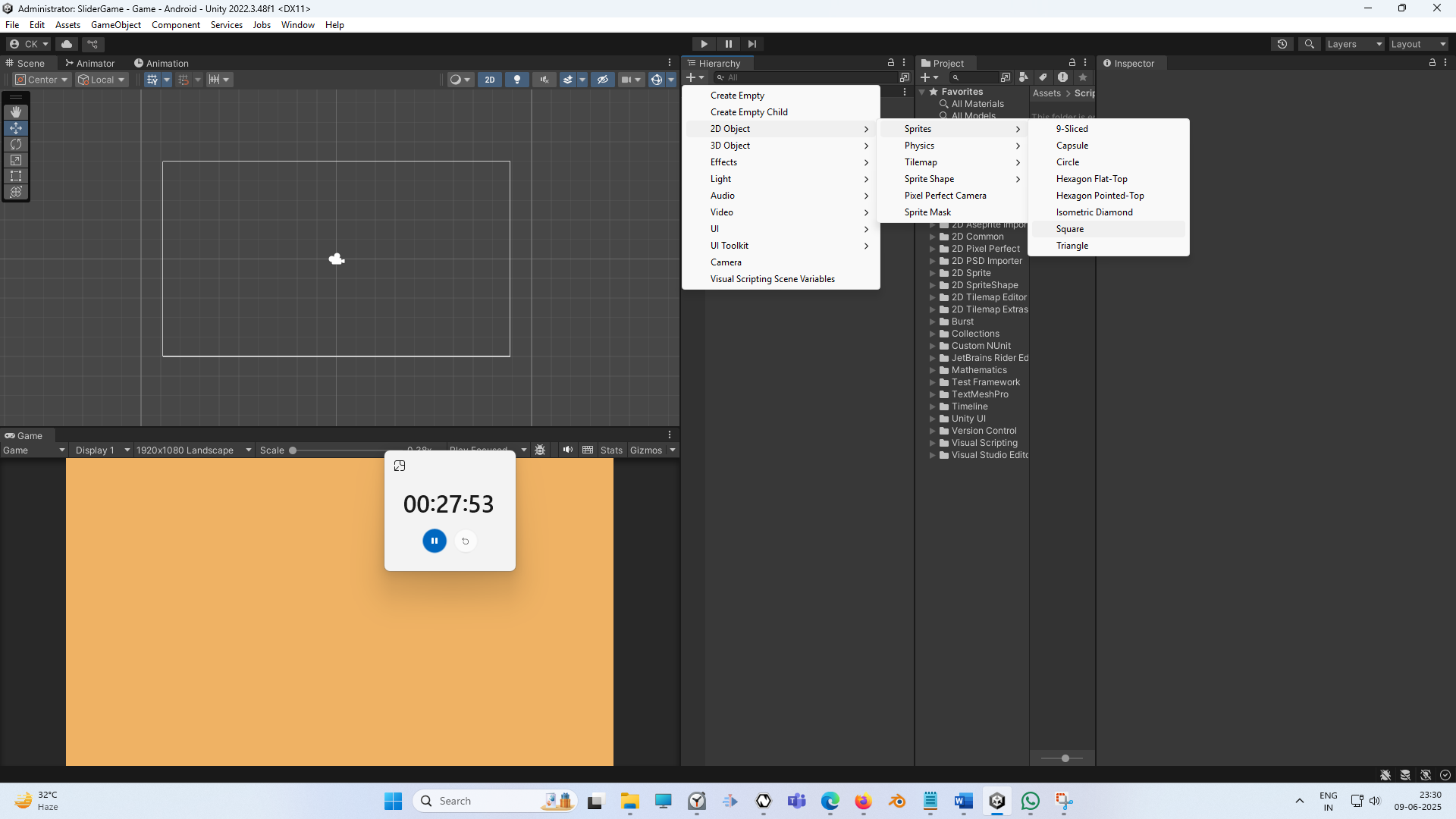
Now go to sprites folder and then right click on it to open a submenu



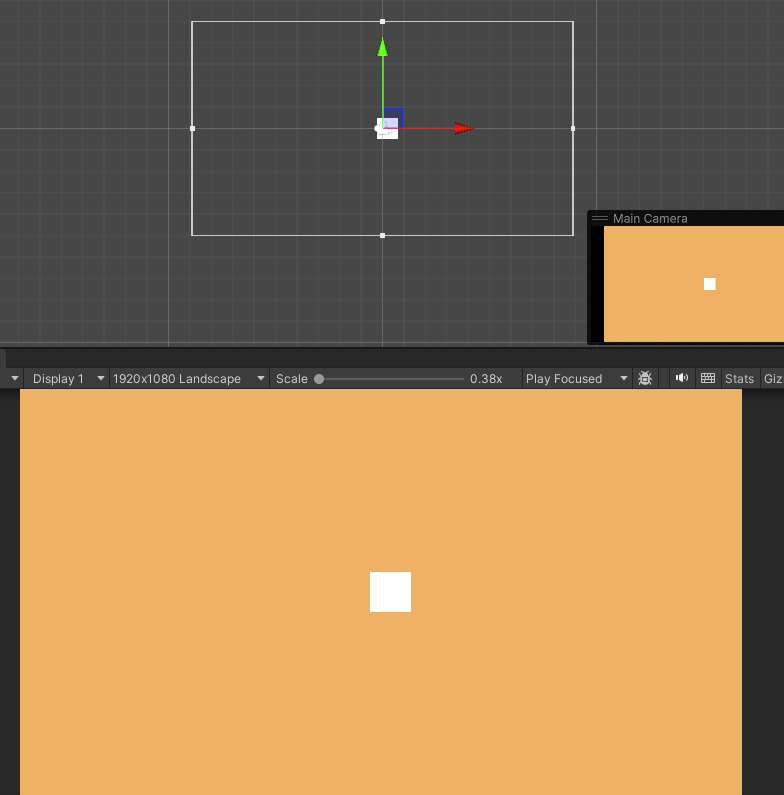
This make an image of square which we will drag on the Heirarchy to make an object but it does not work anymore we now directly place it in hierarchy



Now in submenu choose 2D object -> Sprite -> Square



Now you can see the screen a square will be shown to us



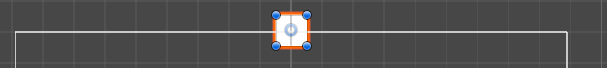
**Note:-**

Some feature are specific to platform so first switch the platform you want to work on then use the Unity

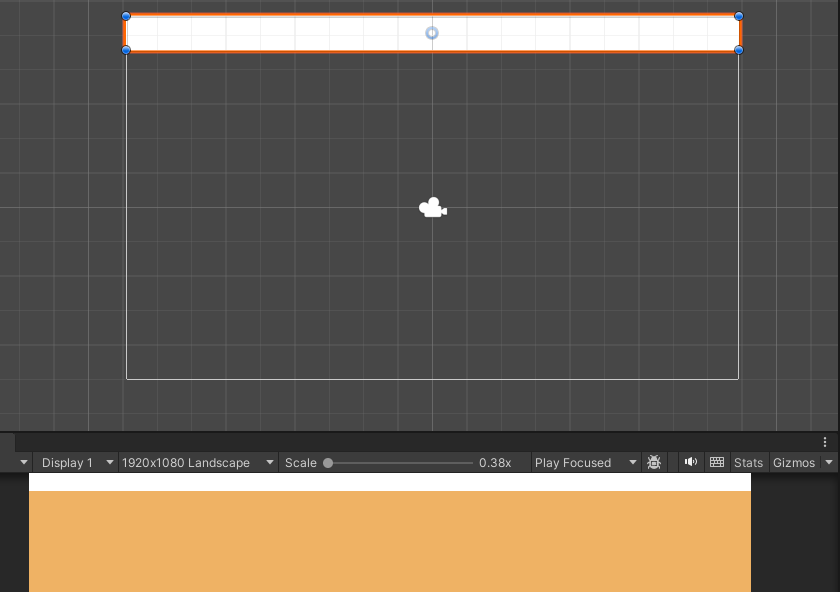
9:45

Now we have a squsare in the center

Now move the square to the top of the screen as shown below. Make sure rect tool is selected

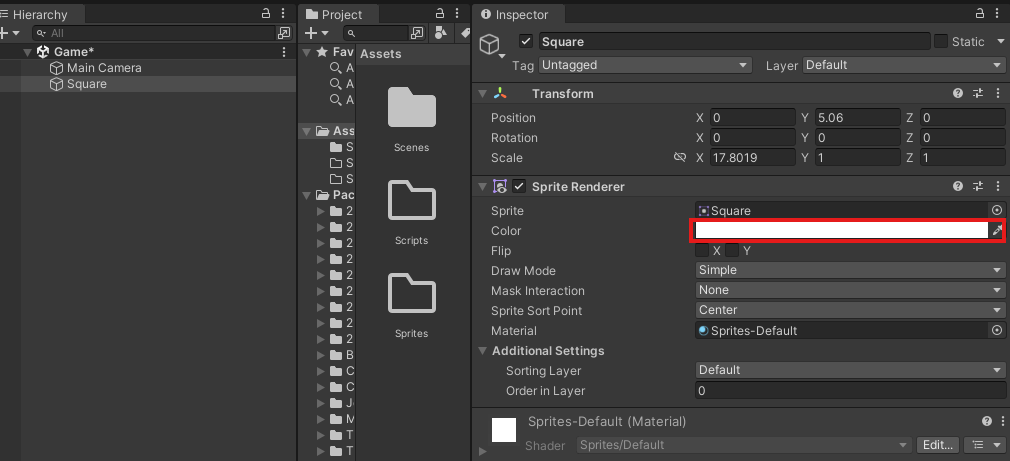


Now we can simple press the Alt button and hold it and drag it to make it bigger by dragging such that it increases in width. When we drag and then Alt then the size is increased such that the opposite is also increases in opposite direction so they grow equally in opposite direction due to which an abject size is increases uniformly as shown below



You can also do it manually

Now change the object color by clicking on the marked below

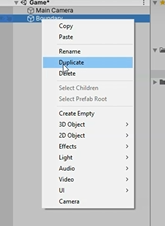


Now in color picker choose the color



The color picker set the value as above

Now we rename the square to Boundary

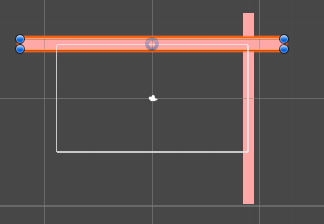


Now right click on the boundary IOA in the Heirarchy panel

**Note:-**

There is z axis in 2D unity but there are shown from 3D perspective means just seeing the same console( who knows )

Now by rotating the duplicated boundary in 90 degree in z axis and push it to the right direction until it just align with the right edge of the camera as shown below



Teacher have missed one step of adding RogidBody2D by selecting the boundary IOA and then go to inspector panel and then click on Add Component and then in submenu go to physics and then to BoxCollidor2D



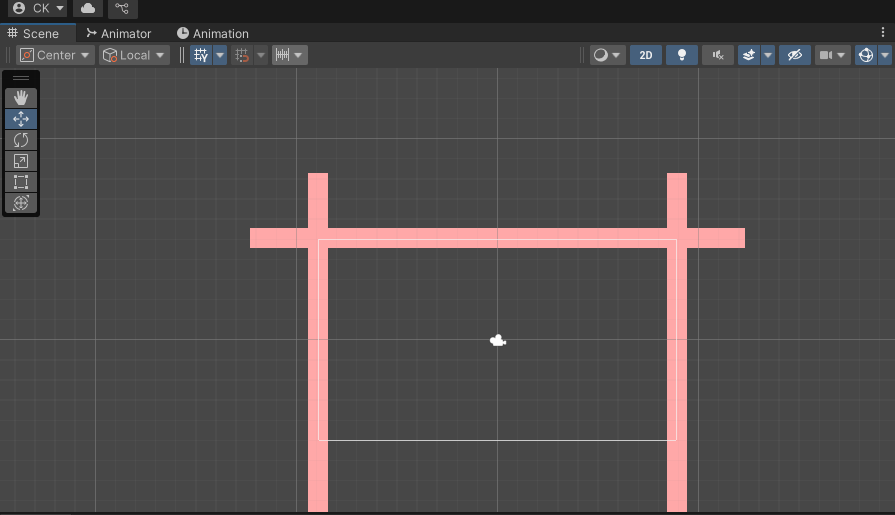
Now you can see above it that boundary is surrounded by green outline which covers the whole boundary IOA

Now our object will collide with other game object. So if we don't give it a boundary or if we don't give it a collider, then our balls will pass through it and they will not collide or bounce from the walls

But if we want the ball to bounce from the walls that’s why we give it a Box collider

Now we apply the box collidor to the right boundary in the similar way

Now duplicate the right boundary and set its x valuer to -ve so that it comes on the left side of the camera Now it look like below



So now these things are looking pretty good and all of them have a box collidor attached to them.

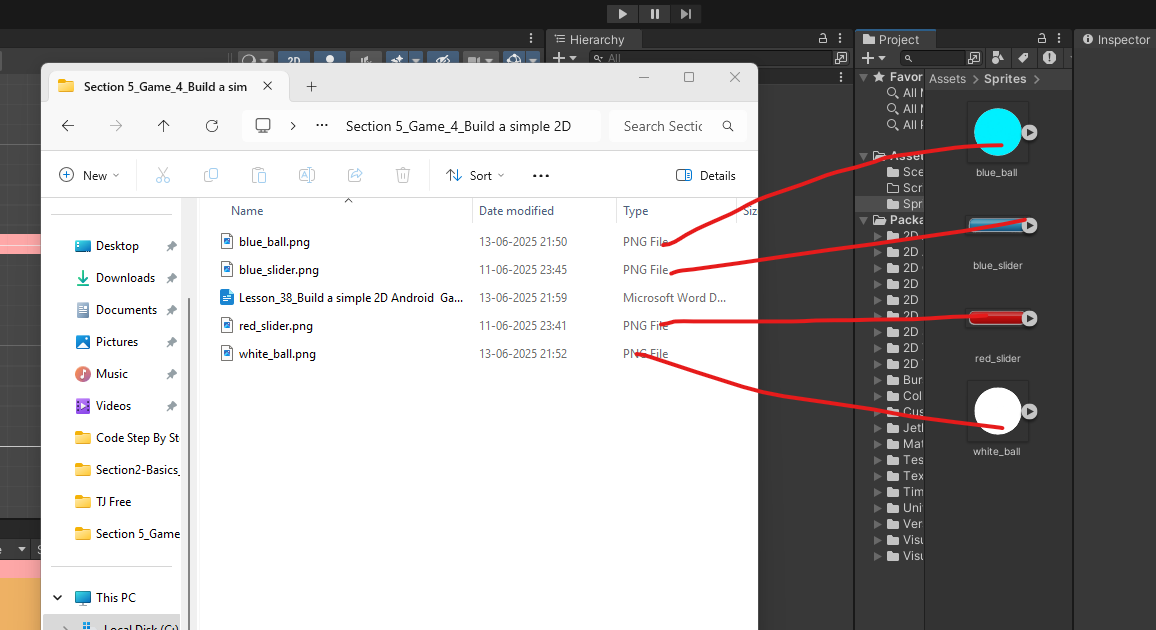
So now we have created boundaries, now we need to bring the pedal and the ball in our scene.

So now we can add pedals and ball to the game

12:55

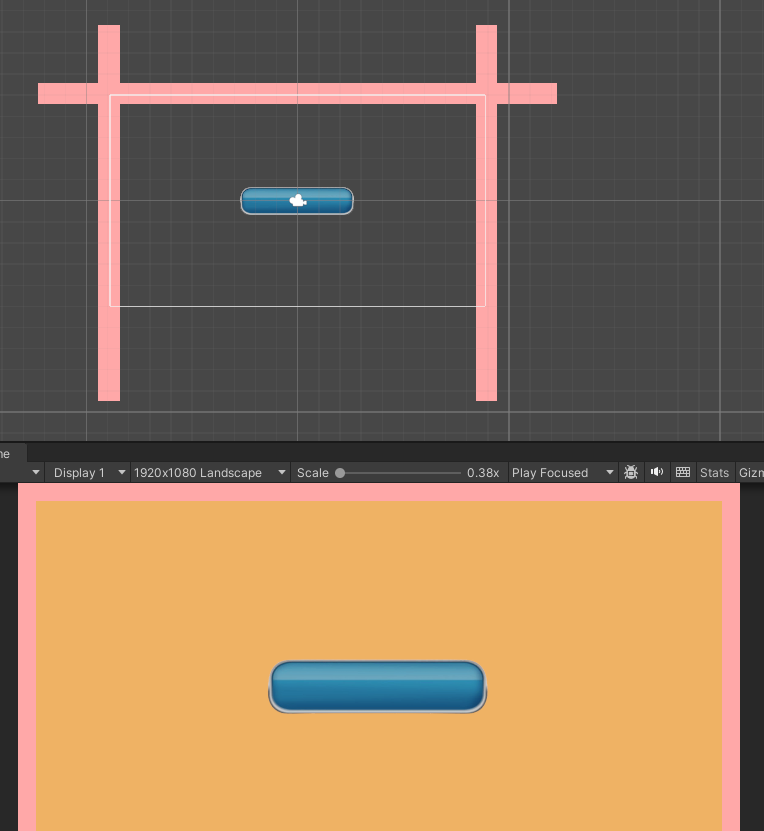
So in sprite folder we added the images of Paddles and balls you can download from any website

Add it by dragging the image over the sprites folder in project panel sass shown below



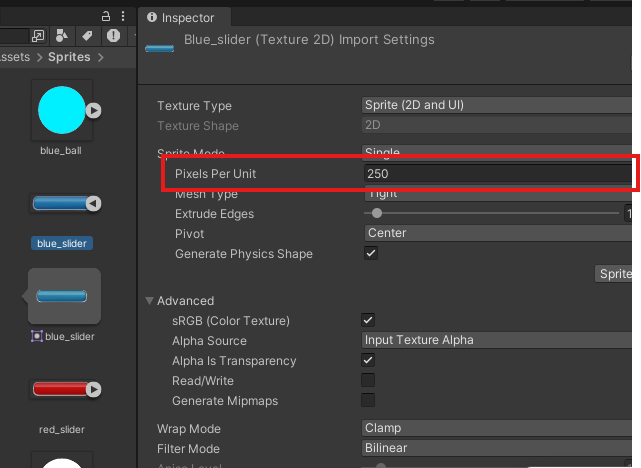
Now drag from the blue\_ball image from the sprites to the hierarchy pane so that it can be included in our game or as IOA in Heirarchy(who knows)

Now we can see it in our scene and game panel that it is bigger in size

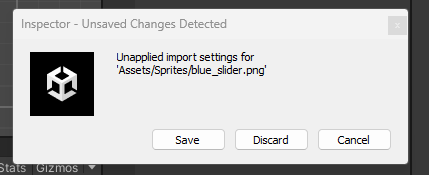


You can reduce its size by scaling tool or going to the IOA source image that is blue\_pedal in sprite folder of project panel select it and then in inspector panel go to Pixels per unit and then to make it small size obviously we will choose large number so that more pixel from total pixels from an image will constitute in a unit.

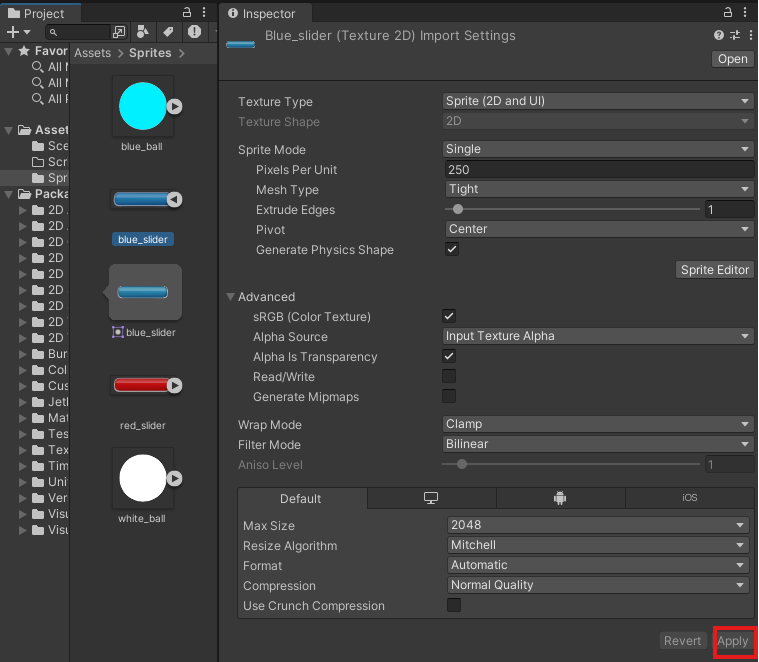
Now I will turn the pixel per unit from 100 to 250 in pixels per unit as marked below



Now when you activate panel then you see popup as shown below

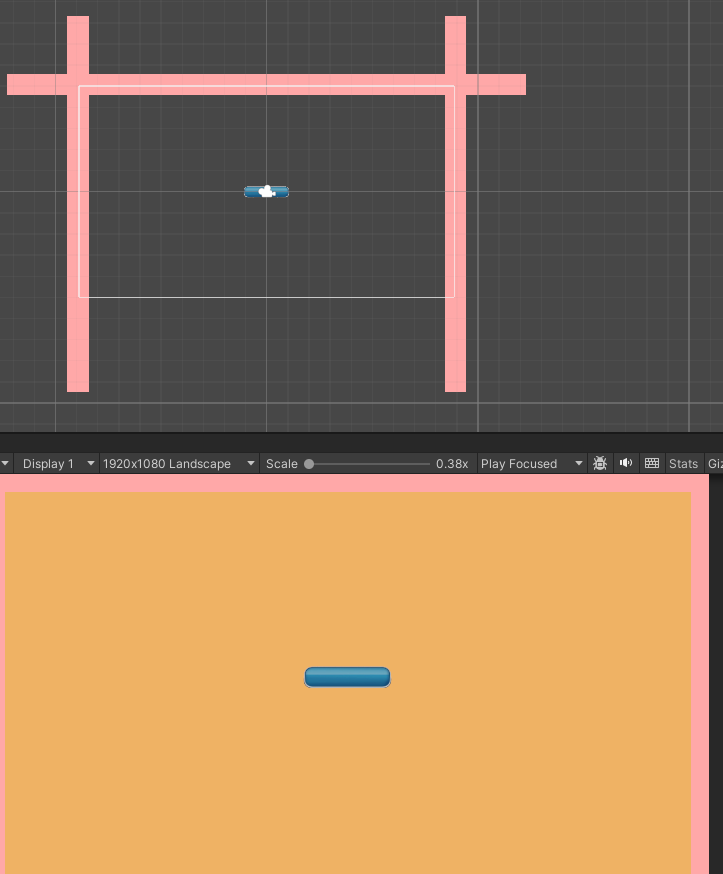


Click on save button to save the changes we have done in the inspector of that sprite so that its effect can be seen in IOA generated from it (who knows)



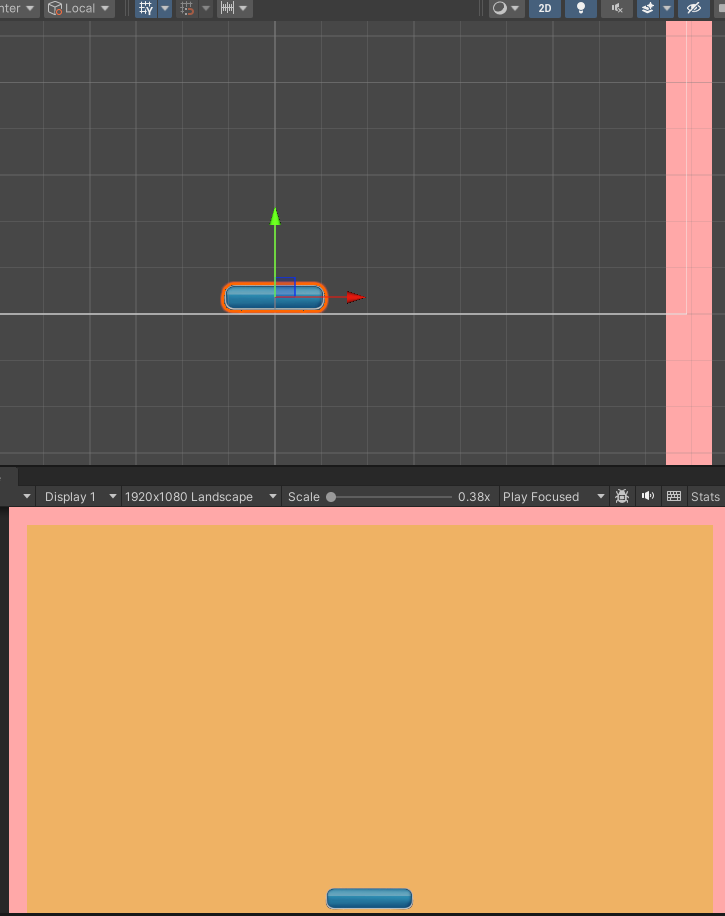
Or you can after changing the setting in Inspector you click on the apply button as marked above to being changes to the IOA generated from it.

Now our blue\_pedal looks like below

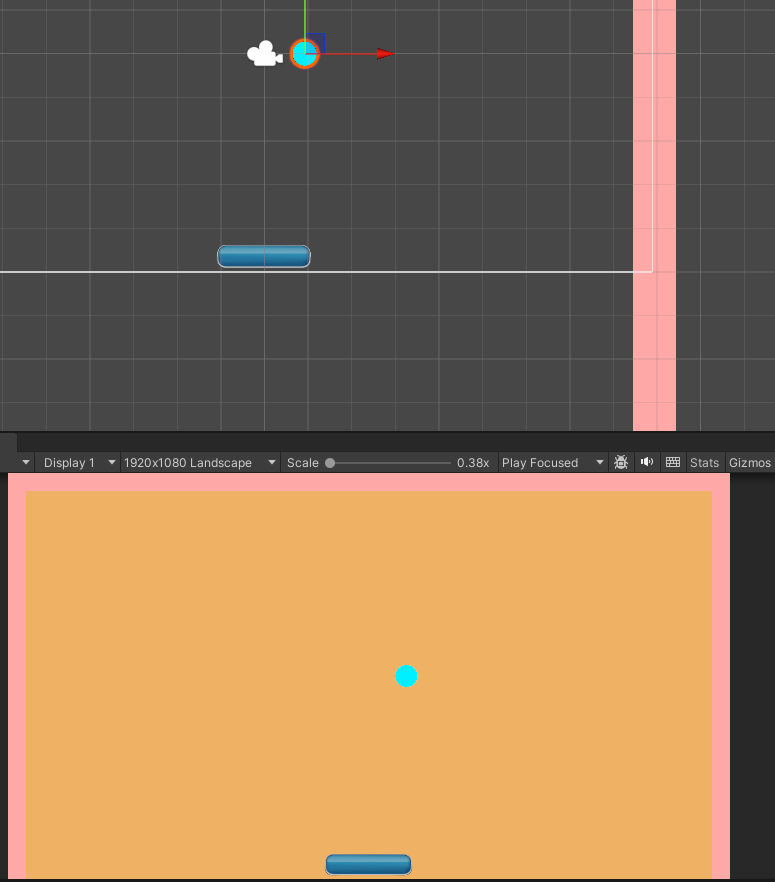


Now it is smaller in size as shown above

Now we select it move it downward until it touch the bottom edge of the screen



The same way we can select our ball and drag it like this. And then again make it smaller by pixel per unit property Now you can see it in your screen below



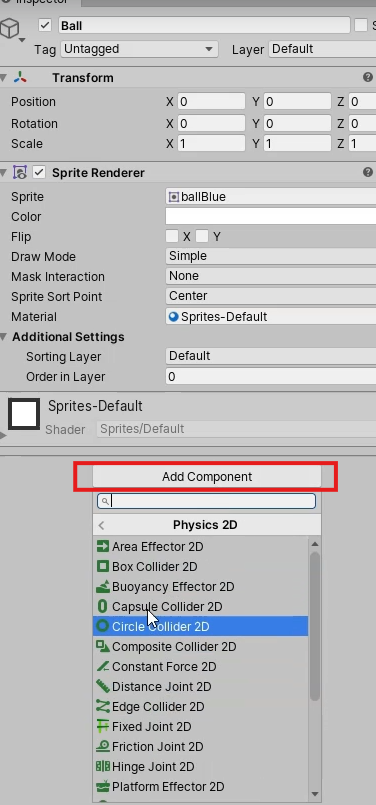
So now we have our panel as well as our ball right here on the screen Now we need to find ways so that we can move our ball, and move our medal, bounce a ball and do all these thing you want And whenever you make some changes make sure you save the scene

In hierarchy rename the blue\_ball IOA to ball

Now we can add a physics component to our ball. Now we want our ball in the game to bounce, to bounce off the walls, to bounce from the pedal. And then we want to we want it to collide with the pedal and the boundaries. So that's why we need to add some physics mechanisms to the ball.

To do that in unity, we need to simply add a component called rigid body 2D

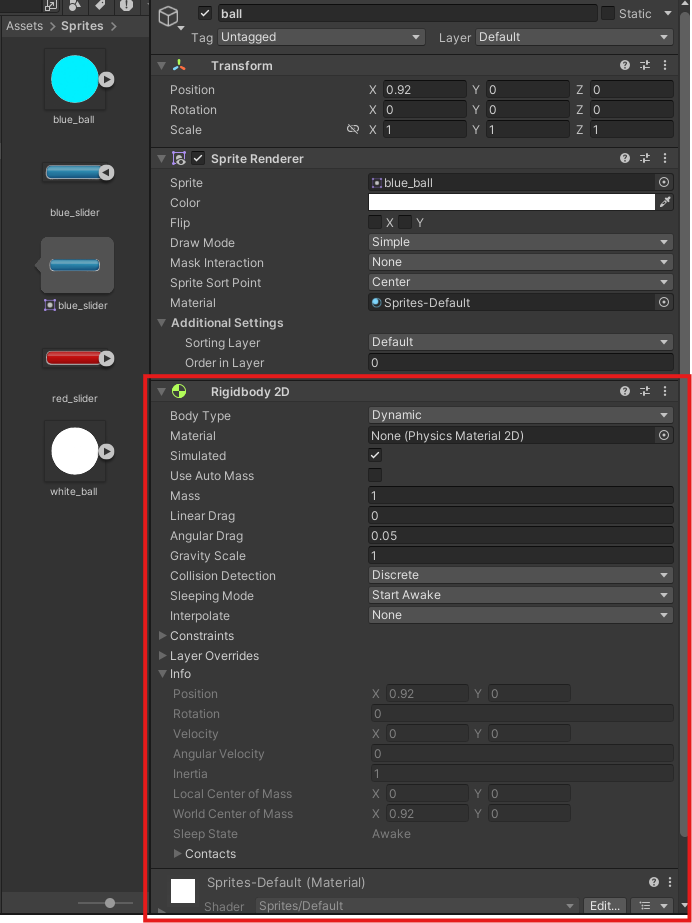
Now select the ball IOA and then in inspector panel we click on add component as marked below



And then in submenu go to physics2D and then to RigidBody2D

when we added this body to the component, it automatically has these properties like Mass.

Drag, gravity, scale and all these things as marked below

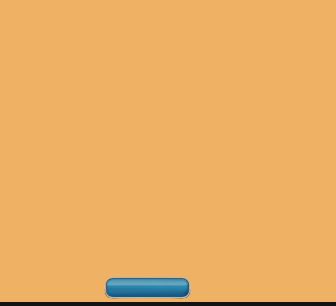


so unity’s physics engine gives this thing to us already

15:20

Or By default when we add this rigidbody to the component to any game object

Now if we press the play button then ball will drop down as shown below



the ball is not affected by physics and that's why it is falling down.

Now, we need to give it 2D collider to the ball so that it can collide with other game objects.

So we are going to go to add component and then in submenu go to physics 2D and then to circle collider because our ball has a circular shape.

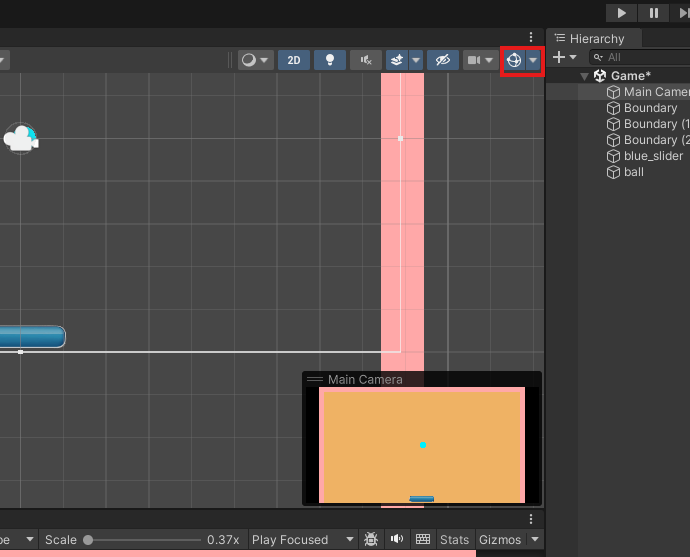
**Gizmo:-**

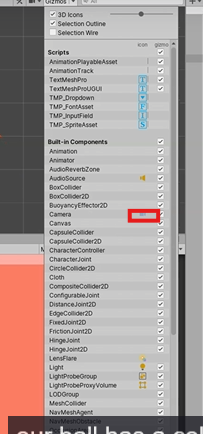
**A gizmo is a visual tool or widget in 2D/3D software that helps you interact with objects — usually for:**

* **Moving (Translate)**
* **Rotating**
* **Scaling**

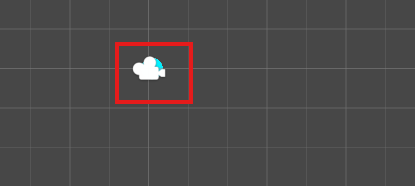
And I think in this case camera is somewhat responsible for changing in position of the objects so that’s why teacher is calling it the gizmos

The camera was interfering with the objects so teacher make it invisible as it was a gizmos we can make it invisible by going to the marked option in the marked below menu in the scene panel





and then click on the camera icon beside the camera option in the submenu as marked above so that its internal camera icon as marked below become invisible (not disable) in the scene panel.



And if you uncheck the checkbox to just make its outline invisible

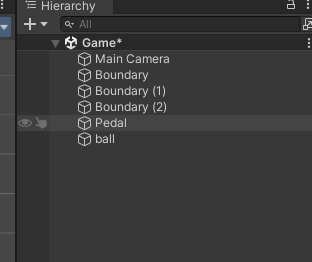


Now you can see above that the ball boundary is in green color as collider is attached to the ball

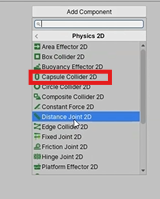
So now we have a collidor attached to the board, but still the ball will fall down because our paddle

doesn't have a collider.

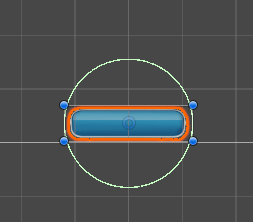
All right, so now we're going to select our blue\_slider and let's rename it to Pedal.



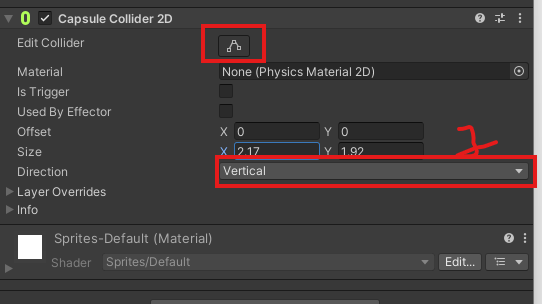
And now we go to add component and then in submenu choose Physics and then to Capsule Collidor as marked below



Now you can see a collider is attached to the paddle as shown below

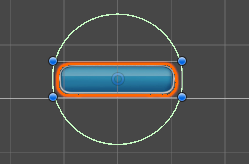


Now you can see the collider as shown above but the collide is circular in shape and its is not aligned according to the paddle boundary So we edit the shape by clicking on the edit collider option in the Capsule Collidor panel as marked below

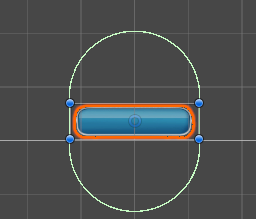


Now you can see above marked 1 that is direction there are two option that are vertical and horizontal if the vertical is chosen then if you increase the value of y field of size option above the marked 1 then only upper and bottom part will be elongated and but the left and right side will be flatten out for example if we increase the y then it will look like below

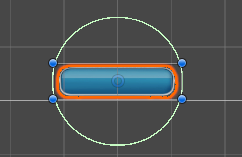
**Before**

****

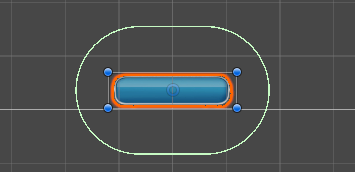
After increasing value of y



the horizontal is chosen then if you increase the value of x field of size option above the marked 1 then only left and right part will be elongated and but the top and bottom side will be flatten out for example if we increase the y then it will look like below



After increasing value of x



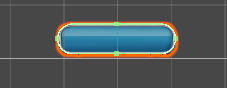
So for our paddle shape we set the direction from vertical to horizontal  
you can also use edit collider button to edit it directly (experience)

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Note:-**

The edit collider also get manipulated by the direction property(maybe see in future)

So now we have a nice capsule collider all around our pedal as shown below



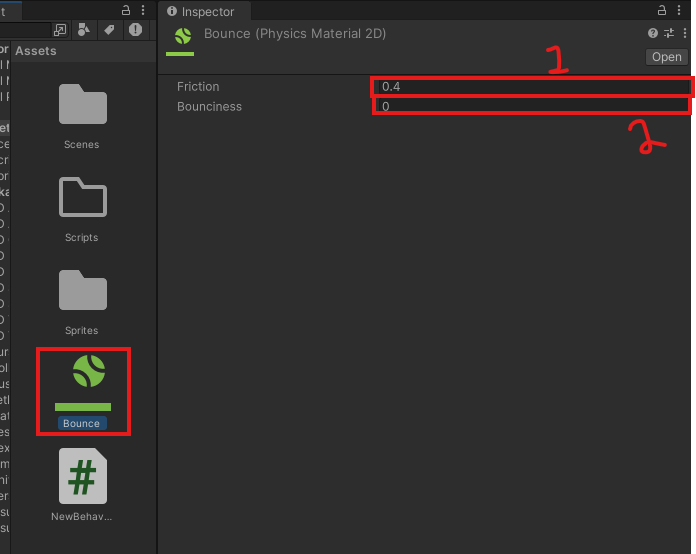
So now our panel is ready to take all the collisions. So now go ahead and click on Play

18:00

Now you will see the fall down and the ball will fall down but it will not go through the pedal that’s because our pedal has a collider attached to it now So now let's go ahead and move on to the next step. To make the ball bounce, we need to add a physics material to the ball.

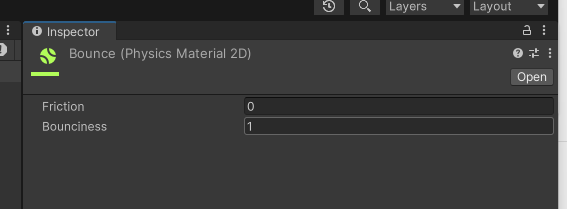
I think physics material tells use that how a object behave when it interact with the other object whether there will be a friction or not or another things o think by this the object can behave as it is a metal or plastic etc

Now go to project panel and then right click on the assets folder and then Go to create and then to submenu 2D (according to new unity)and then to Physics Material 2D and Now a Physics Material 2D is formed in assets folder we are going to name it Bounce as marked below



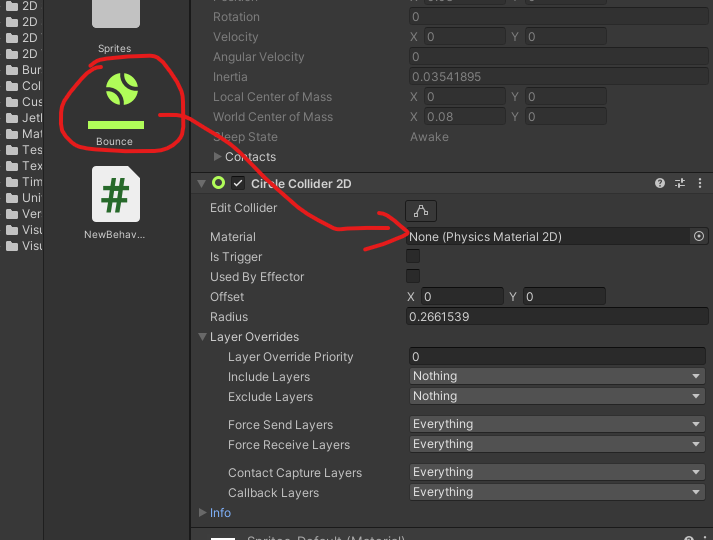
Now if you select it and then go to inspector as shown above

And now we're going to change its friction ( as marked 1 above )from this 0.4 to zero. That means we don't want any friction and it change the bounciness(as marked 2) to one. So now our bounciness is set to one that means the ball has the maximum bounce property as shown below

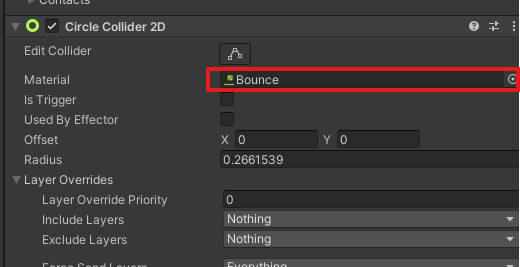


OK, so now, go ahead and we can play, you will see the ball fall down and it will not bnounce because we have created the property, but we have not added it to the ball.

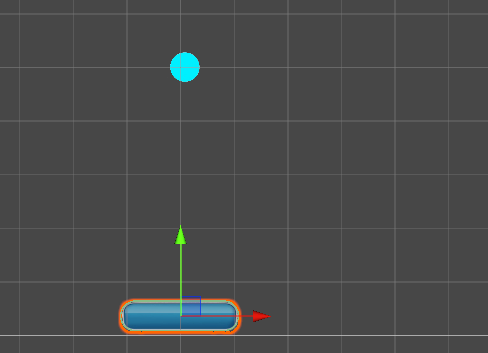
Now in Circle collider 2D panel you can see the material property as marked below



Now we will drag our Bounce physics material from inside the assets folder and then drop it to Material property as shown above so our ball no has this bounce property attached to it as marked below



Now we can click on play. And you will see the ball keeps bouncing as shown below and this is the first movable thing you have created and probably you will love this

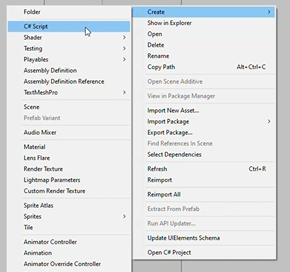


**Note:-**

The special thing about unity engine is that when the ball touch the paddle edge then it go sideway instead

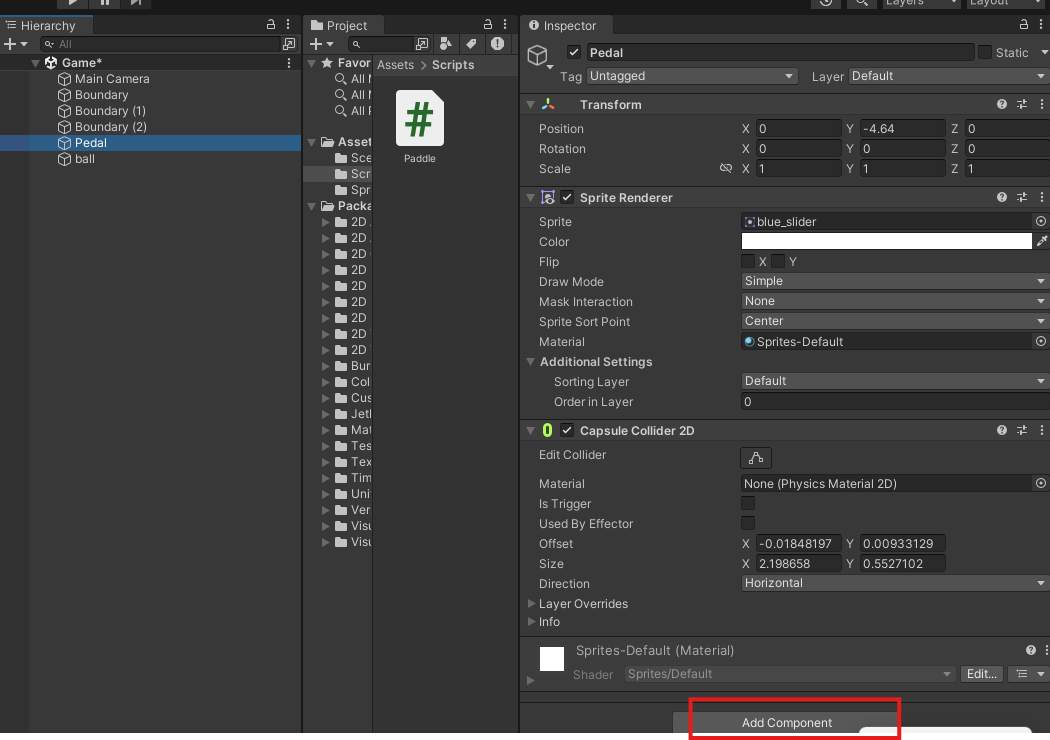
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

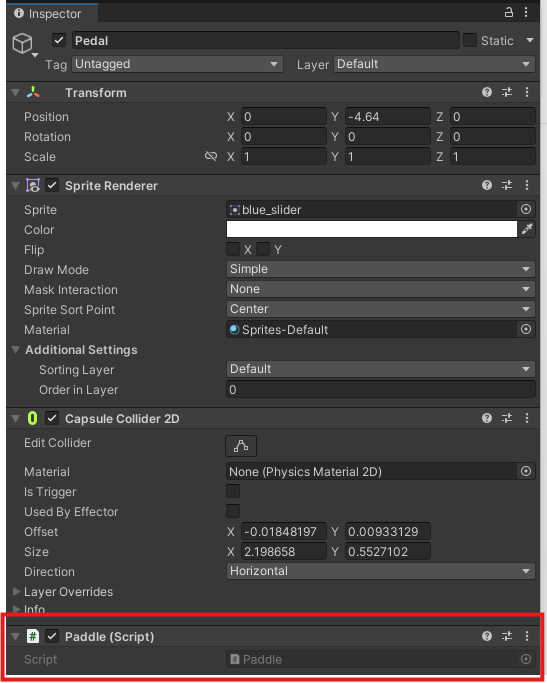
Now, what we need to do is we need to move our pedal. And also we need to add some force to the ball, to do all the things we need to create our very first scripts The scripts give behaviour to our game objects. In this case, we want to keep some behaviour to the pedal and also to the ball. So first of all, let's try to move our pedal left and right. So let's go ahead and create a new script. So I'm going to go inside the scripts folder. Right Click and then in menu go to Create and then to submenu C# Script as shown below



and make it rename it directly here, because if you try to rename it later, you will not be able to do it.(maybe possible now)

Now we simply name it pedal and drag and drop the script to the add component in the inspector panel of padel IOA in hierarchy panel as marked below



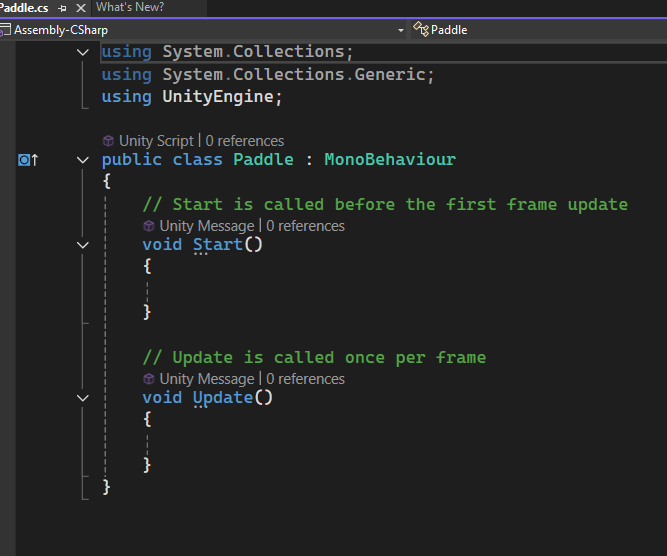


Now you can see that paddle script is attached to the paddle IOA as marked above

So now the script is attached to a pedal that means anything that we write inside the pedal will control or affect the pedal.

20:50

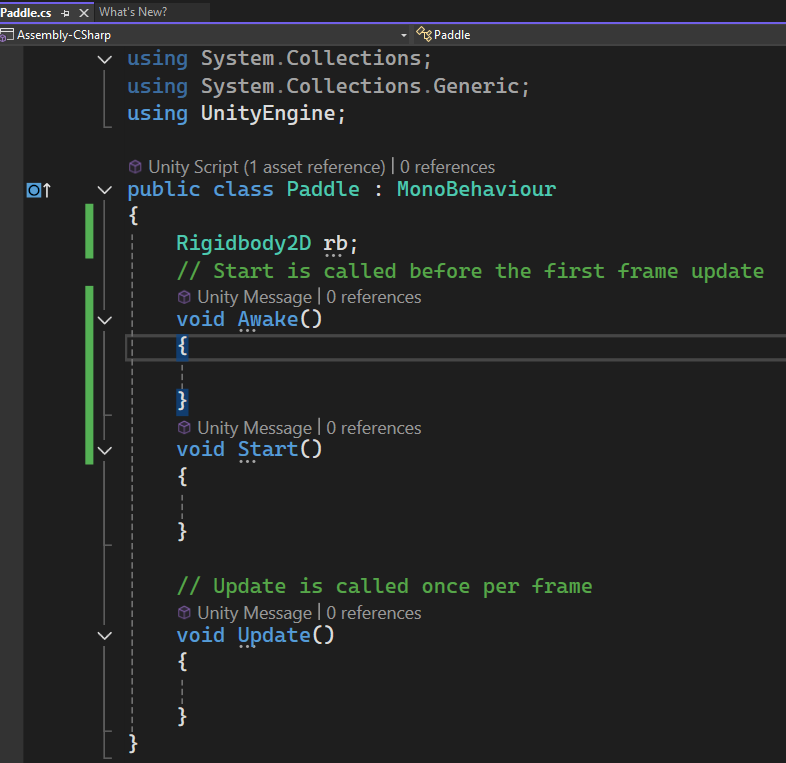
No, I can simply double click on it to open it in Visual Studio now, in your case, if you have selected any other editor, it may open there.



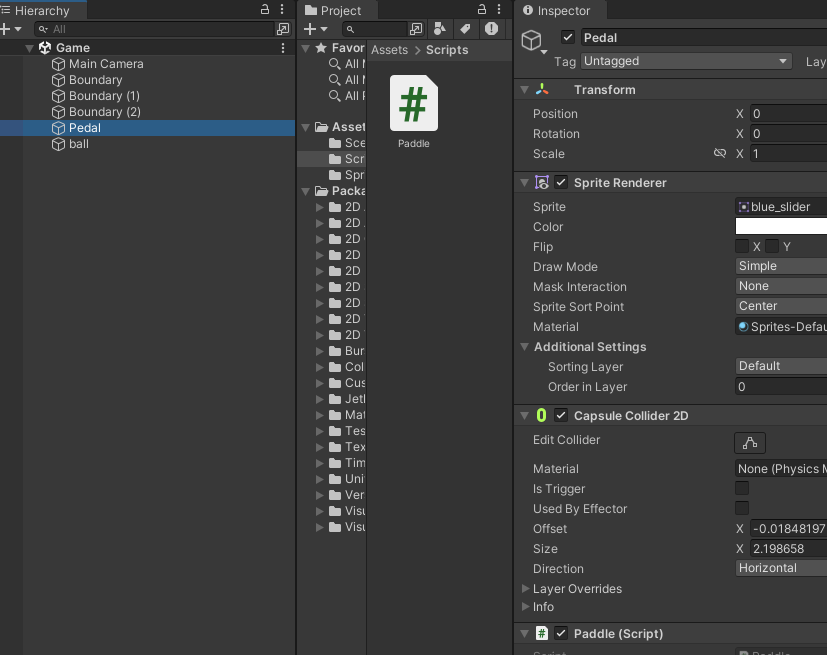
Now our script will be opened like above now ctrl and scroll to make zoom in the editor. Now, here we need to write our very first code.

Now we're going to create a new variable of type RigidBody2D So this variable will store a reference to the rigid body that we have attached to our pedal and then we can use that reference to move our Pedal.

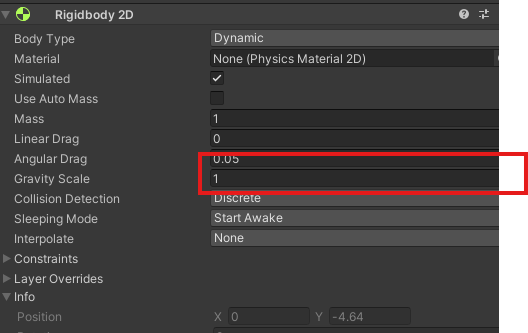
So now here we are gonna create a new function called void Awake. So this function gets called even before this start function automatically by unity. So whenever we need to get a reference to any objects, we need to use this awake function. Now our code will look like below



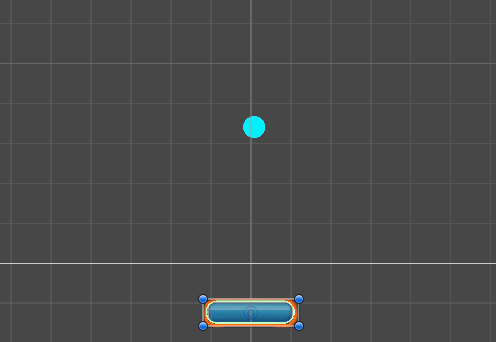
Now let's go back to Unity and check few things. So if I go back to Unity and select out panel, you can see we have not attached a rigid body to our pedal as shown below in its inspector panel



So first of all, to go to add component and then to Physics 2D and add a rigid body to the component to our paddle. All right, now, as you can see as marked below this one has this gravity as well

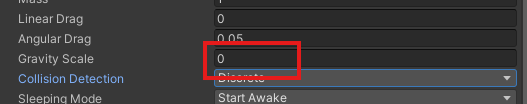


but we don't want our pedal to fall down when It is played as shown below



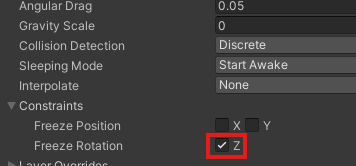
**Note:-**

I have observed that when unity is not active window and another thing is active window then the game get paused and only play when the unity is active but we don't want that.So that's why we're going to make the gravity scale zero. As marked below



Now if I play the game then the paddle remains stationary but when the ball collide with it then it start moving because maybe of second law of motion (and maybe unity physics follows the second low of motion)

And then from the constraints, we're going to freeze the rotation of paddle along z axis by checking the marked below



That means we don't want our panel to rotate around the axis like this. So we want it to stay still and not move. So now what we can do is from our script, we need to get a reference to this rigid body today that is attached to this component.

And then by using the repository and by using the physics functions, we need to move our pedal using some physics forces.

And now in awake function we get the RigidBody component attached to the Paddle and give it to the rb variable as shown below in awake method

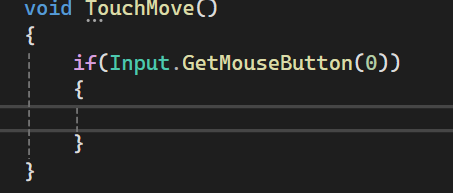


So this is a Syntax when we want to get access to any component that is attached to our game object in this case, we are getting access to the rigid body, to the component that is attached to our pedal and we're storing it inside this rb variable. Then we can do anything with this rigid body attached to anything with the rigid body attached to our panel using this RB reference. Now, here we are going to create another name it public float moveSpeed. So this is the speed by which we want to move our pedal to the left and right direction.

So here we can create a new function, we can call it void touchMove, so inside this function, we're

24:15

gonna give all the functionality that we have for our paddle to move over the touch input. So first of all, to detect, if we have touched on the screen or put on the screen, we're going to say Input.mouseButton(0).So we want to detect if we have pressed the little mouse button on our mouse and this left mouse button will work ad the touch input on our android device as well .So we want to detect whether we are pressing on the screen or whether we have left clicked on the screen and currently on hold. So this one will check if we are clicking on the screen and quickly on the screen now as shown below

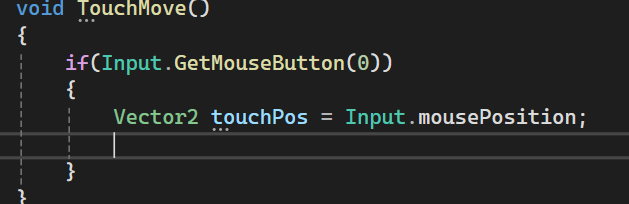


we had to detect whether we are clicking on the left side of the screen or whether we are clicking on the right side of the screen. So to do that. First of all, we need to get the position where we are touching. So to do that and when Teacher gonna create a new variable “touchPos” to do that of type Vector2 (as it is a 2D game). So this variable will store the position where we have touched and how we really how will we get the touch position we will get at from the mouse position. So we are going to say Input.mousePosition

If we click on the screen, it will call the input Dockett mouse button function ( basically Input.mousePosition will checks every frame that is there is any touch on the screen. If yes then it returns true for that frame and also upcoming frame if the touch remains but will return false if no touch is detected in a frame )

Lets suppose the center of the screen is zero zero that is zero x and zero y. and on the left and right we have negative x values and x positive values. So like this, we have to detect whether we are pressing at a position where the X value is negative. Then we want to move the pedal to the left. And if we are clicking on the position whether its value is positive, then we have to move the pedal to the right.

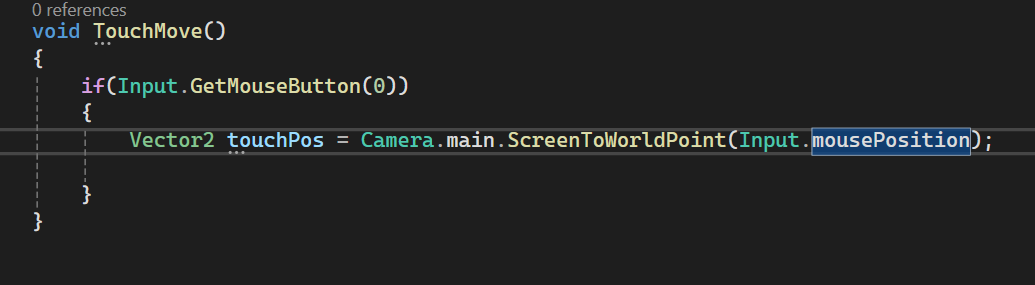
As you can see, we are getting the mouse position where we are touching an storing it in touchPos of type Vector2 as shown below



but this will not work because this is in screen coordinates. But whenever we are working on the games, most of the time we need the world coordinates(in this case the center is 0,0 otherwise I thik that in screen coordinate top left is 0,0(who knows)) not to convert

now to convert it from screen to world coordinates.

we write Camera dot main dot to ScreenToWorldPoint and tehn pass the mouse position of left click or touch as shown below



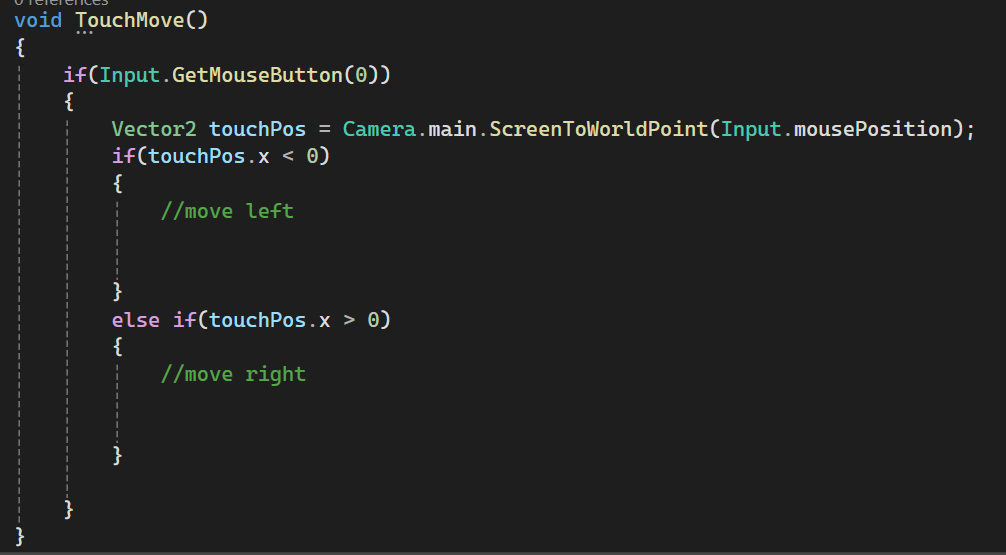
So now it will take this position converted from screen point to world point and store it inside the

touchPos variable as shown above

27:33

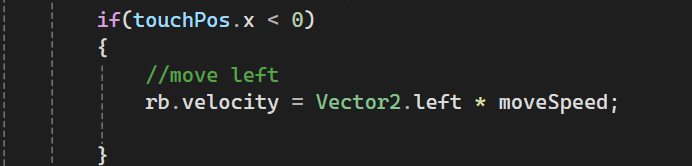
All right, so now we need to check if touhPos dot X is less than zero.That means we are pressing on the left side of the screen and else if touch was dot X is greater than zero.

Now the code will look like below

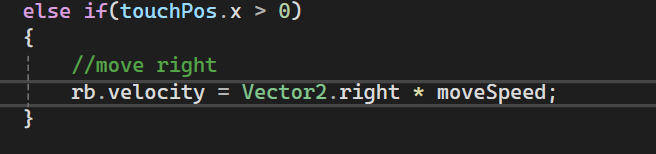


So if we are pressing in a position where the x value is negative, then we want to move to the left, and if we are pressing at a position where the x value is positive, then we going to move to the right.so how can we move to the left and right

We going to say, rb, that is a rigid body dot velocity. So we give a velocity to the rigid body And how much is the velocity? We can say vector2 dot left multiply movespeed variable. So here we are, adding a more speed on the left direction on our velocity value as shown below



So now whatever value we'll give, it will move to the left side of our screen the same way we're going to move it to the right side so we can simply go ahead and copy and paste it right here in the else if and instead of writing Vector2.Left we write Vector2.right as shown below

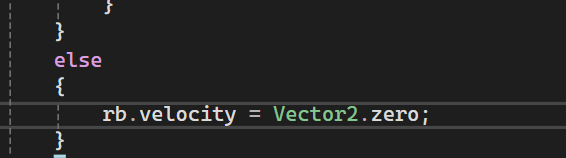


So now we can add this force or multiply the force on the right direction of our screen and then we're going to store the velocity in our Rigidbody. This is the amount of velocity that we are adding to our rigid body. So now whenever we click on the left it moves to the left, whenever we are clicking on the right, it will move to the right.

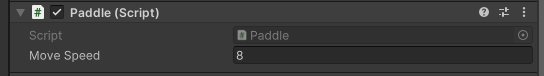
But when we are not clicking anywhere, we want our paddle to stop. Otherwise, it will keep moving.

Now in the outer if statement which is checking that the screen is getting is pressed or not

we can add an else statement here. we say else That means if we are not pressing on the screen, then we are going to say rb dot velocity equals vector2 dot zero. So when we are not clicking anywhere, we want the velocity of the pedal to be zero as shown below

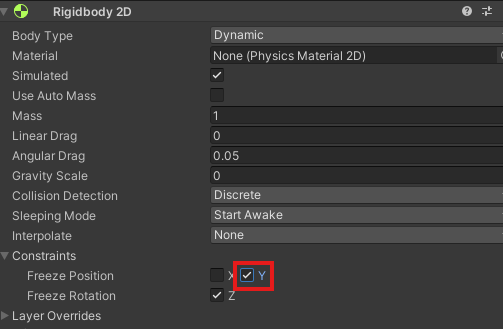


Now go to paddle(script) panel and set the move speed to 8 as shown below



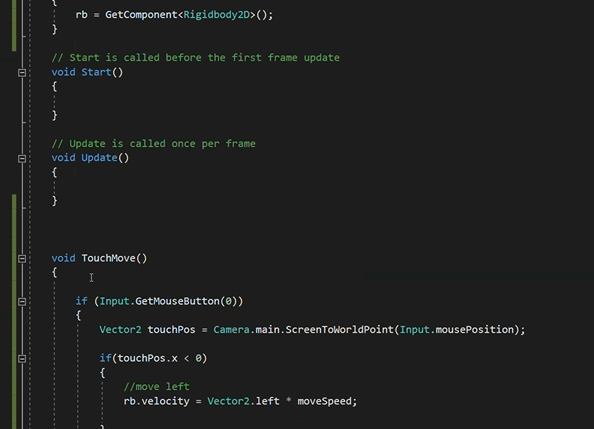
Now click on Play button But it will not work. And when we click on the left and right, nothing works and the paddle goes down so we can see nothing works and the paddle goes down because of the force of the ball

So what we can do is we can simply check the first position Y as marked below in the rigidBody section of paddle in Rigidbody2D so that the pedal will not move down



You only move to the left and right.

And now if we go back to our script, you can see that we have created this function TouchMove, but we have not used it anywhere.as shown below



So you have to call it from somewhere so that we can use it.

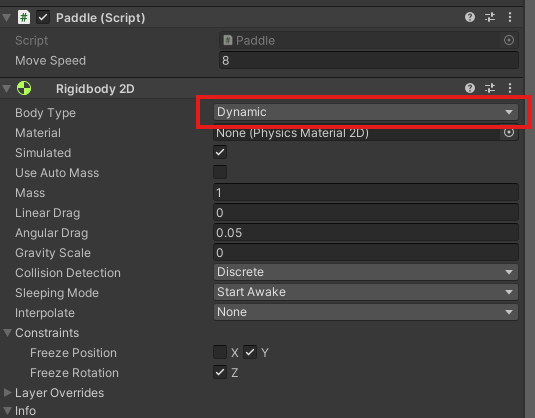
To do that, we can create another function void, fixed update.

30:50

So in Touch move function as you can see, we are moving our paddle, using the physics functions on the physics properties, we need to call that inside this fixed update function.

**Discovery:-**

When in rigidbody if you choose body type as Dynamic and by code you set its velocity to 0 then it will affect the object that is colliding with it but if you freeze y then it will not happen but if you chose kinematic then you don’t have to freeze y as this does not exist in that case You choose body type as marked below

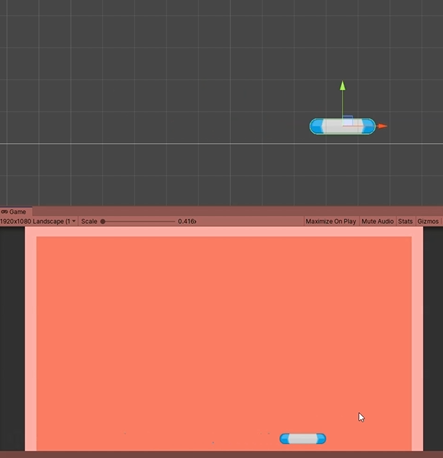


\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Inside the fixedUpdate function we call Touchmove function now this fixed object function is automatically get called by unity again and again after 0.02 second(according to internet independent of frame).

So our Touchmove function will be called after every 0.2 second and all the task in the TouchPos will happen again and again So now lets see the script and move back to the unity

Now in unity click on Play button And I click on left the paddle goes left and I click on right paddle goes right as shown below



Now you can see that we cannot move beyond the boundary as boundary has a collider as same as paddle. You can increase the speed of the paddle by updating the speed field in the script panel of paddle Inspector panel. All right, so you can adjust it according to your own wish and your pedal is ready. So now we to add some code or write a script for a ball so that we can make it bounce whenever we want and do whatever we want So let's start coding the ball.

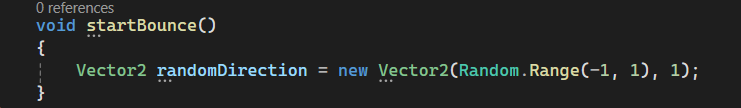
So first of all inside the script folder create a new C# script file and name it ball

Attach the ball script to the ball(YKI(You know it)) Now open the ball script by double clicking on the script field in script panel of Ball IOA. So the same way, first of all, and to get a reference to the rigid body that is attached to the ball.

Ao we declare a variable rb of type RigidBody2D and inside the awake function we are going to get access it in the same we get it by getComponent method

Now we are going to create another variable we call it float bounceforce and we are going to make it public because if we make it public, then we can change the value from our unity editor.

Now we create a method here that is StartBounce, this function will make our bold move to a random direction or and force the ball in a random direction. So first we are going to create a new variable vector to random direction and how can we get the random direction to create a random direction by creating an object of type Vector2 and in argument for x pass a random value between 1 and -1 which signifies that the ball will move from left and right and y takes only 1` means the ball will only go up Now our code will look like blow



So this randomDirection is a vector2 variable and vector2 has two valuers that is x value and y value

So our first parameter in the constructor of Vector2 is saying x value and we are making the x value as positive and negative one and making a value as 1. Now we need to get them to use this random value and add force to ball in this random direction to do that

**Discovery:-**

When we apply any physical activity with code then unity follows otherswise it uses normal physics I think so(who knows maybe I am wrong)

To do that

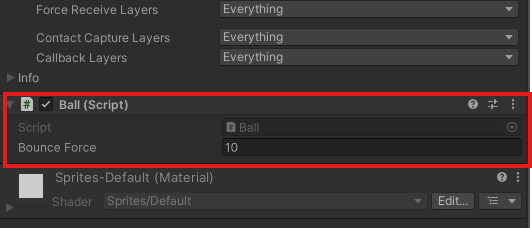
35:37

we are going to say rb.addForce and then we need to give a direction in which to add force for the right direction and by how much amount we want to add the force we are going to multiply it by bounceForce variable So that means we are adding a force in the random direction by the amount of this bounce force and then we pass ForceMode2D.Impulse It will give more force to our ball and we are going to make it easier to move faster in lower bounds force value. So now all we need to do is we need to call this startBounce function sometime.

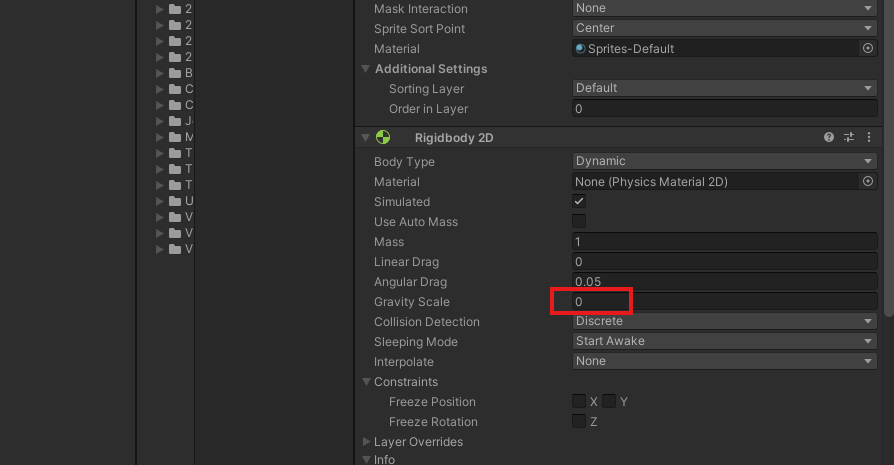
So in this case, whenever we are clicking or pressing on our screen and whenever we are pressing any key, we want to start this StartBounce bounce function.

So inside the update function, we got to see if input dot any key down. So whenever we are pressing any key or touching anywhere on the screen(maybe), then we want to call this startBounce function. So whenever we are clicking anywhere, we want to make our ball start bouncing and from there we can start working on the game. whenever we are going to touch anywhere, our balls to start bouncing. So let's go back to our unity editor here.

We have our ball so we can give a value to the bounce force in our scripts section as marked below

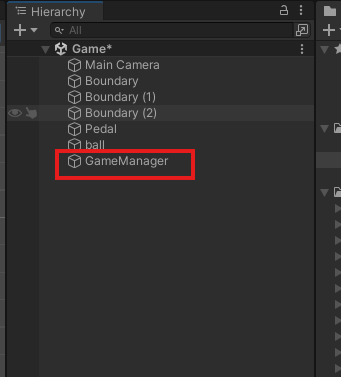


Give this value 10. And also, I'm going to select the ball and change its gravity scale from one to zero in RigidBody component in the inspector as marked below because we don't want it to move and fall down.



Now, if I click on play. Anybody any key or does touch on the screen, as you can see in random force, has been added to the ball. So , I can click on anywhere and as you can see, the force is added to the ball in a random direction.

Now if I click on the screen then force is added on the ball and pedal is moved according to the area we have touched .Make a bounceForce 8 or 5 then it would be accurate. So now we have added the script to the ball as well as with the paddle. So now we can move the paddle and also we can move the ball or make the ball bounce now will to do other like when the ball falls down we need to restart the game. We also need to add score and we also need to create the menu. We also need to add score and we also need to create the menu. So that's what we going to do next. OK, so now let's see how we can restart the game when the ball falls down. So to handle all these things, all these different mechanisms of the game, we're going to create a new script called Game Manager, and we're going to create a new game manager object that will handle all these things.   
So create an empty object by right clicking on the hierarchy panel and in the menu opened go to create empty and we are going to name it gameManager now It will look like below in the Heirarchy panel as marked below



Now create a script called gameManager and connect it to GameManager object. Now open the GameManager script

Now we want all our scripts to access this key managers script easily, because here we will have all the controls of the key. So that's why we're going to make it a static instance. And when we create a static instance, all other scripts will have easy access to this game manager script.

So we are going to write public static GameManager instance and we are going to create awake function and then inside we are going to say instance equals this. All right, so here we are creating a static instance of the game manager and we're setting the instance to this (that is the object of the file we are coding in ).

40:05