Game Development

Welcome to the world of game design, creation, and development, where you will get a first-hand experience of working with 2-D and 3-D Games.

Table of Contents

Godot Background and Interface

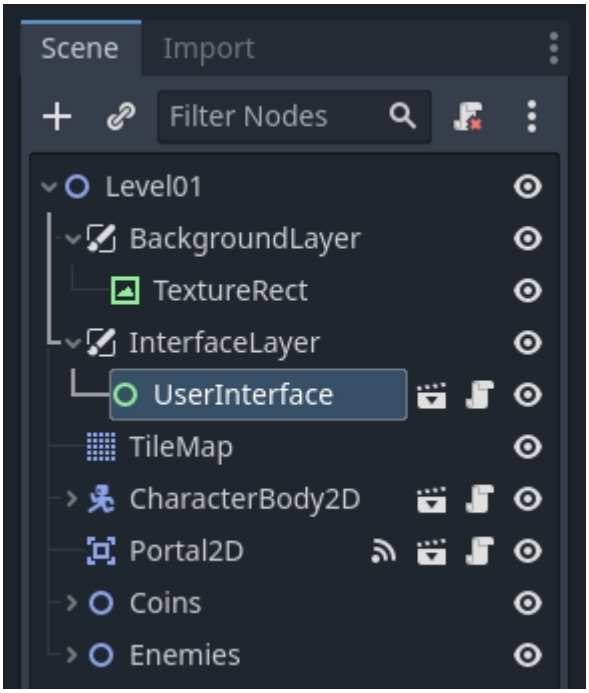
Keywords: nodes, scenes, scene trees, and signals
https://docs.godotengine.org/en/stable/getting_started/introduction/first_look_at_the_editor.html
https://docs.godotengine.org/en/stable/getting_started/introduction/key_concepts_overview.html

Godot is a *game engine*. Think of it as a special tool that helps you create video games. Just like how you use a pencil and paper to draw, game developers use Godot to make games.

Key Concepts in Godot

Scenes and Nodes

Scene: Imagine a scene like a page in a storybook. Each scene can be a different part of your game, like a menu, a level, or a character.
Node: Nodes are the building blocks of your scene. They can be characters, objects, or even things like sounds or lights.



Scene Tree

- **Scene trees:** a “tree” of scenes. And as scenes are trees of nodes, the scene tree also is a tree of nodes. But it's easier to think of your game in terms of scenes as they can represent characters, weapons, doors, or your user interface. [https://docs.godotengine.org/en/stable/getting_started/introduction/key_concepts_overview.html#the-scene-tree]
- **Signals:** Nodes emit signals when some *event* occurs. For example, buttons emit a signal when pressed. Other built-in signals can tell you when two objects collided, when a character or monster entered a given area, and much more. You can also define new signals tailored to your game. [https://docs.godotengine.org/en/stable/getting_started/introduction/key_concepts_overview.html#signals]

Scripting with GDScript

- **GDScript:** This is the special language we use to tell our game what to do. It's like giving instructions to your game.

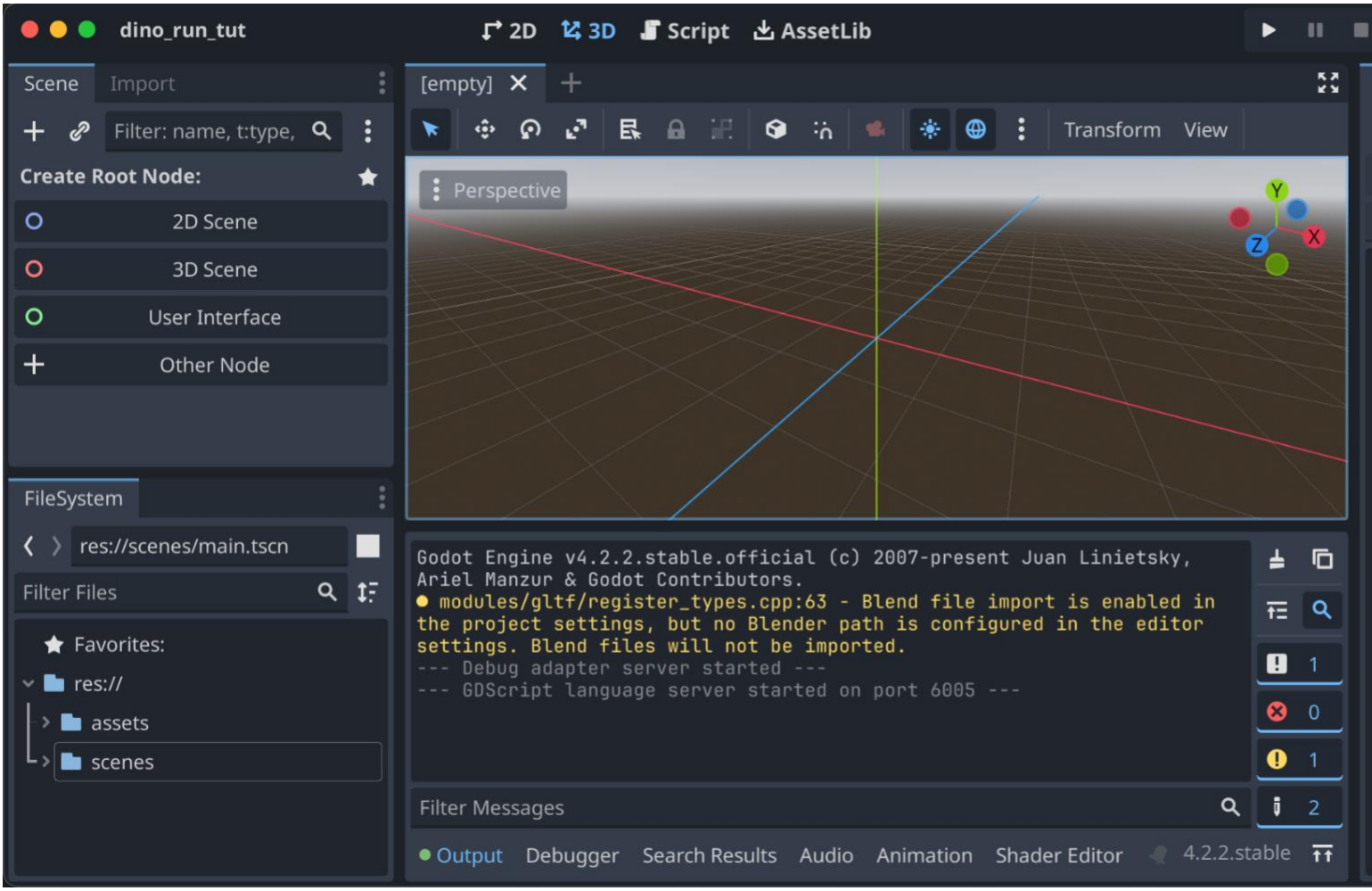
Sprites

- **Sprite:** A sprite is a 2D image or animation in your game. If you have a character or a coin, that's a sprite.

Physics

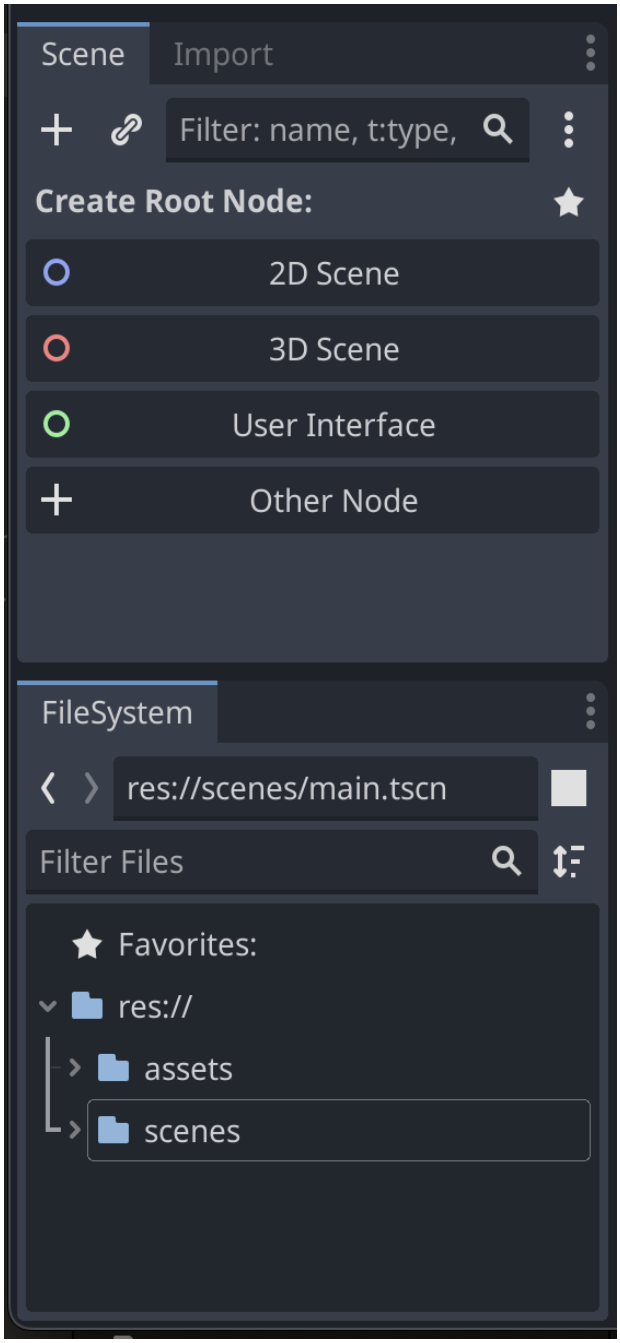
- **Physics:** This is what makes your game feel real. It's the rules that make things fall, bounce, or stop.

Godot Interface



Your time will be spent using all these sub panels in Godot, let’s break them down from left to right.

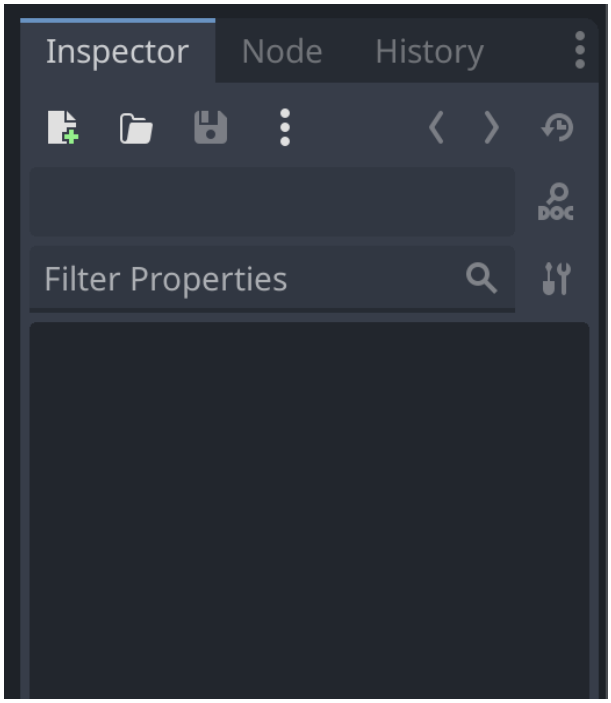
Left Most Panels



The left most panel has a two sub panels–the Scene (and an Import tab), and the FileSystem panel below. Let's start with what you might be familiar with, file systems. The File System panel is how you'll choose existing files and make new files. Using the right click button on existing folders or files you can pull up the context menu to manipulate the selection (delete, rename, move, etc.) or you can "Create New…". These new creations can be script files (.gd), a scene (.tscn), folders, resources (.tres, .rec, .occ), and text files (.txt, .log, and many more). This isn't the only way to add new scenes or files though.

The Scene panel is how you'll set up your game. You will use scenes to make each level, a start menu, rooms, and plenty more. You can also use scenes to group Nodes together, like enemies, obstacles, players, cameras, and any other bundled object. You'll often see a scene nested in another scene, like how a player (a scene) can be placed inside a level (another scene!). Right clicking in the scene panel will bring up a context menu that you can use to add more Nodes, or "children" (a nested node or scene).

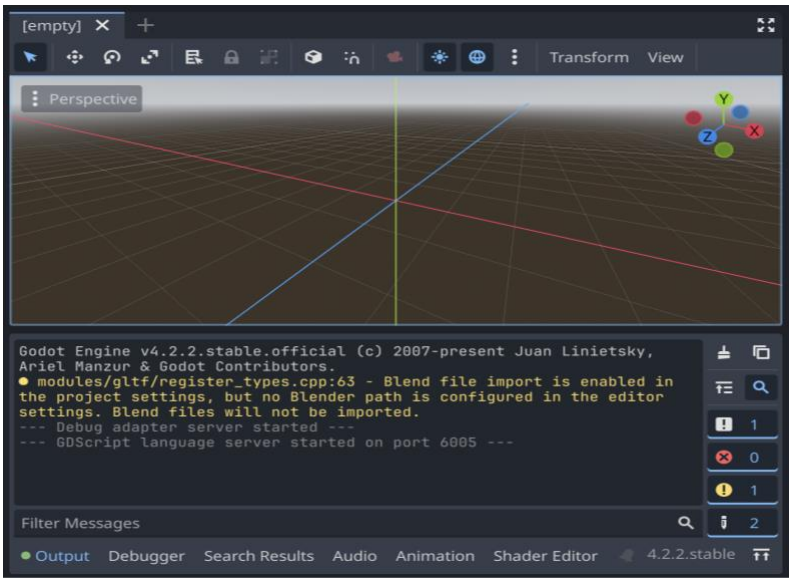
Right Most Panels



When you click on things in the Scene panel, you should also look to the right, at the Inspector panel and tab. This is how you'll make adjustments to the Node or Scene like adding a script, changing textures, directions it moves, frame references, and other properties. Each Node (and Scene) comes with different properties making it important for you to click the right node you want to work with. If you're unsure what you can do with the Node, click on the Node tab so you can see what functions (and functionality) are associated with each Node. It might help you pick what you need to edit.

For example, maybe you want to move the start position of the player, but not the camera. You'd need to select from the FileSystem or the Scene panel on the left. Let's say there's a level scene, a player scene, and camera scene. The level scene has the player and camera nested within it. Since you're moving the player, you'd want the level scene and then can click the player node within it. Or maybe, the player has a sound node, and you want to change how loud the sound is. You'd have to select the player scene from the file system, and then select the sound node in the scene panel and then look at the right inspector window to make changes to the sound.

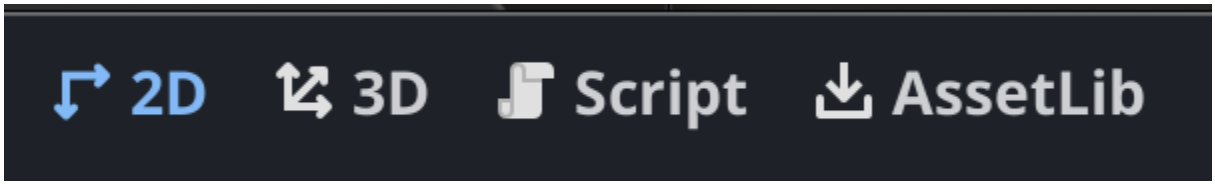
Middle Panels



The middle panels are also the largest panels and where you'll interact with what your users (human players) will see and play with. The top *viewport* can show the currently selected scene, code, and your asset library. You can interact with scenes by dragging nested scenes around, change the perspective from 2D to 3D. You can also edit text based files like .gd and .txt files if you open them from the FileSystem panel.

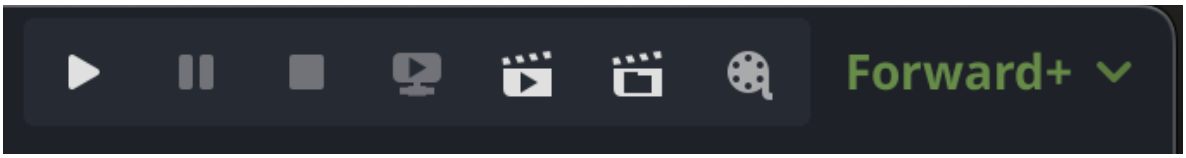
The bottom panel is both the most helpful and the most intense panel. It can tell you where to find bugs through the debugger, edit your audio, edit your shaders, and also print messages your game outputs to developers (not players!). This is a panel you'll often use when you're test playing your game.

Top Middle Bar



This toolbar is found above the scene viewport and is how you can control what you're focusing on such as the scene perspective, scripts, or loading assets. If your game is 2D, then you probably won't have any scenes or objects in the 3D perspective.

Top Right Bar



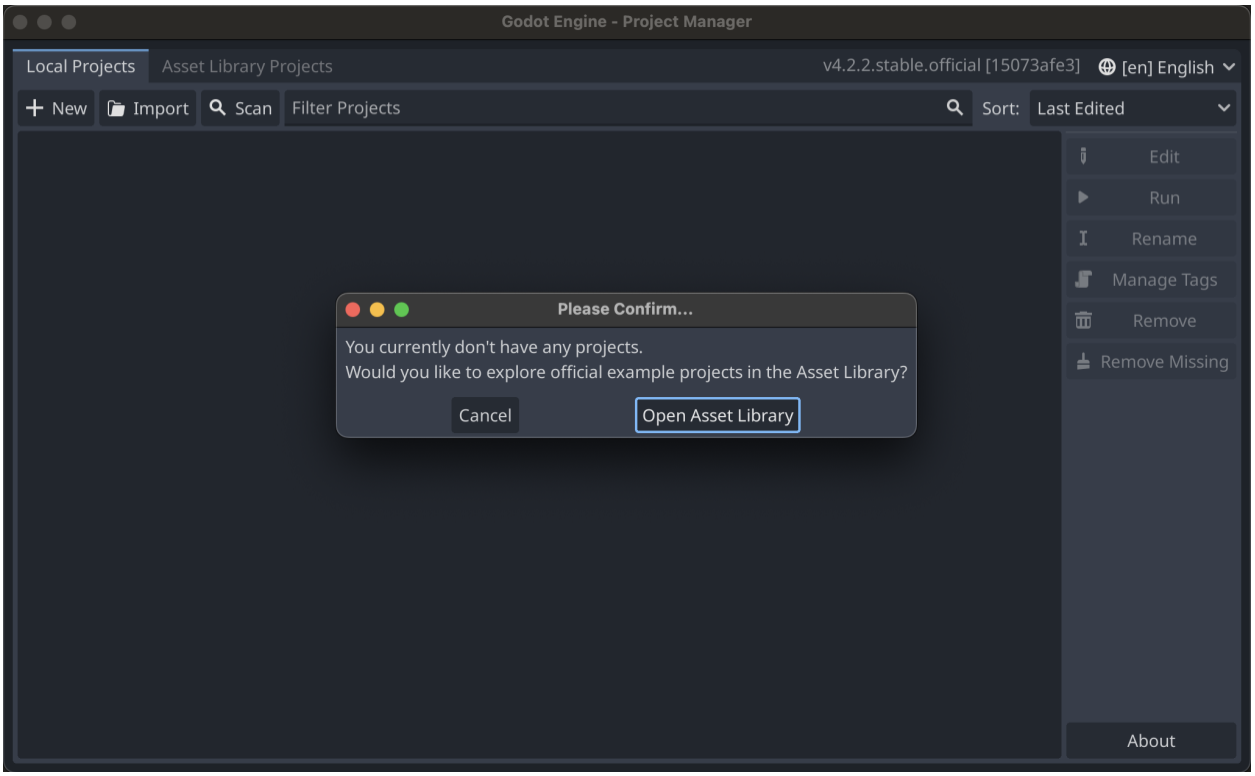
At the top right is another toolbar with some maybe familiar looking icons: play, pause, stop, and movie maker icons. Those first three buttons on the left will be used to play test your game and will open your game in a new window. Have fun!

You may have seen a dino running game while you are not connected to the internet. Let's work on a similar game.

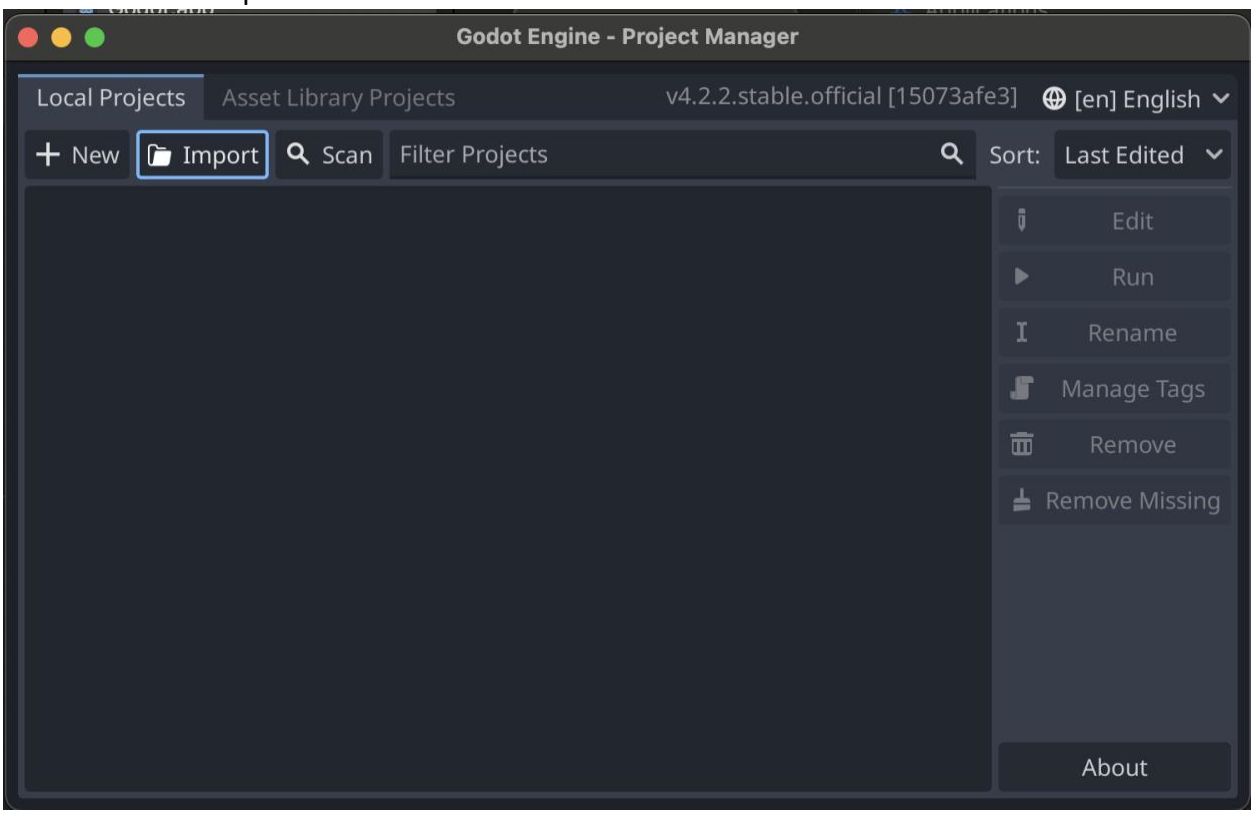
Dino Run

Opening the project

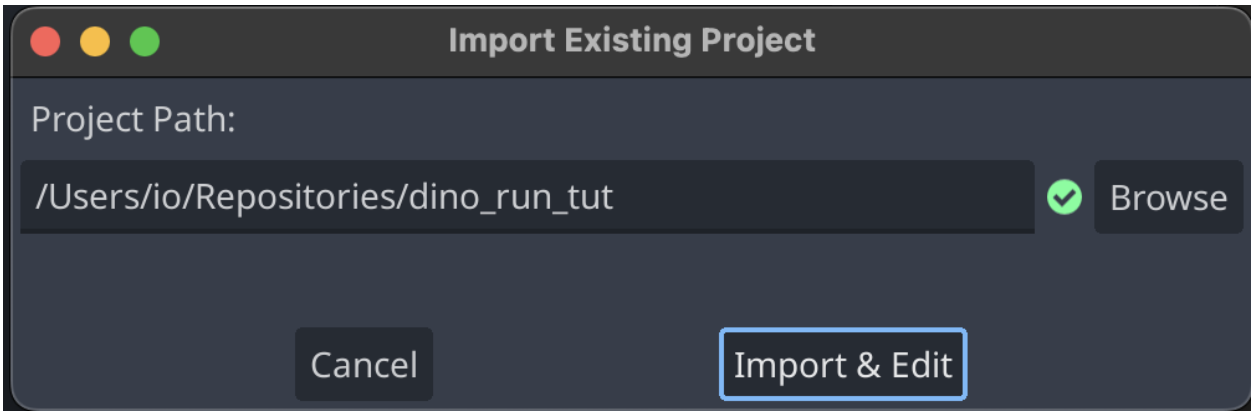
1. Open Godot_v4.2.2-stable_win64 from Godot_v4.2.2-stable_win64.exe folder
2. Click cancel on the Asset pop up



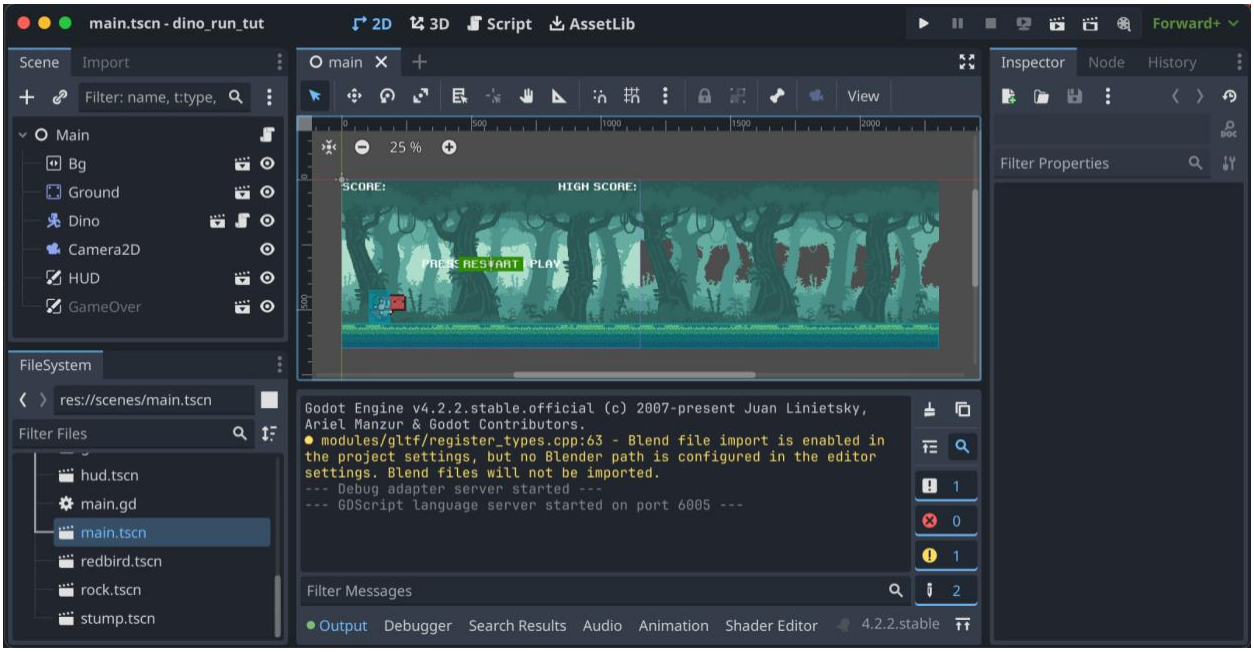
3. Click on import



4. Enter the following path in the dialog box: /Users/Downloads/ASU-SUCCESS/GameDev/GodotGame/dino_run
5. Click on Select Current Folder button on the bottom or the `project.godot` file in the window.
6. Click on Import and Edit and then open the project by click on Dino_run project.



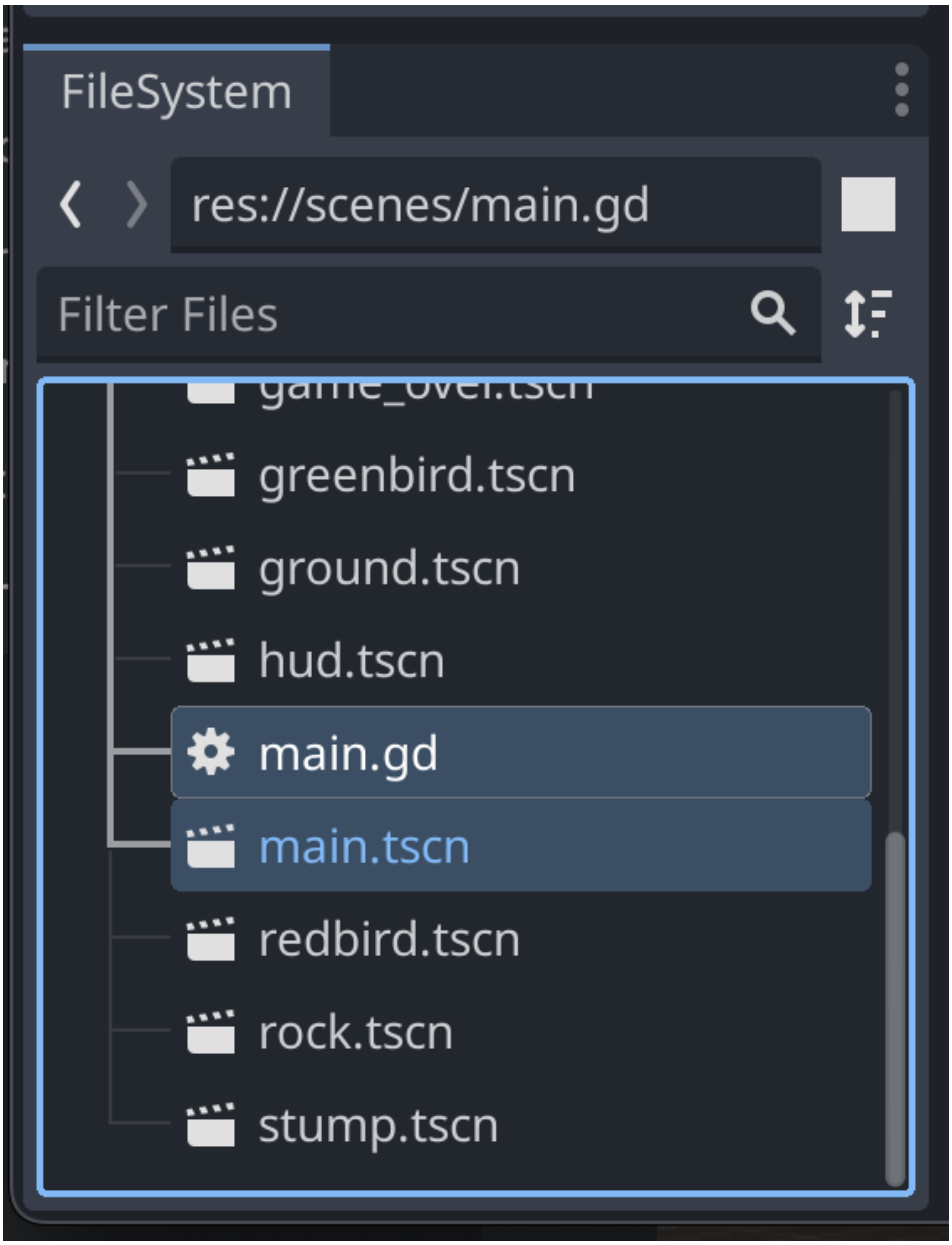
7. Welcome to Godot!



8. Now the next step is to test and run the game to make sure everything is working fine. Click on the play button at the top right toolbar. How high can you score?

Changing Parameters

1. Now the first thing you will do in this game is to change the speed
2. Open `main.gd` file, to open that click on scenes on the bottom left corner.
3. In the scenes folder click on `main.gd` file. Remember there's two types of files for a scene: the GDScript (.gd) and the scene file (.tscn).
    a. We'll often need to work with both as we construct a scene, but for this instance we only need the .gd file. As an exercise, open the .gd file but not the .tscn files. If you click the 2D and Script buttons at the top, what do you notice?
    b. What if you close the .gd file (right click on the file in the viewport, click close all) and then open the .tscn and click 2D and Script buttons at the top, what do you notice?



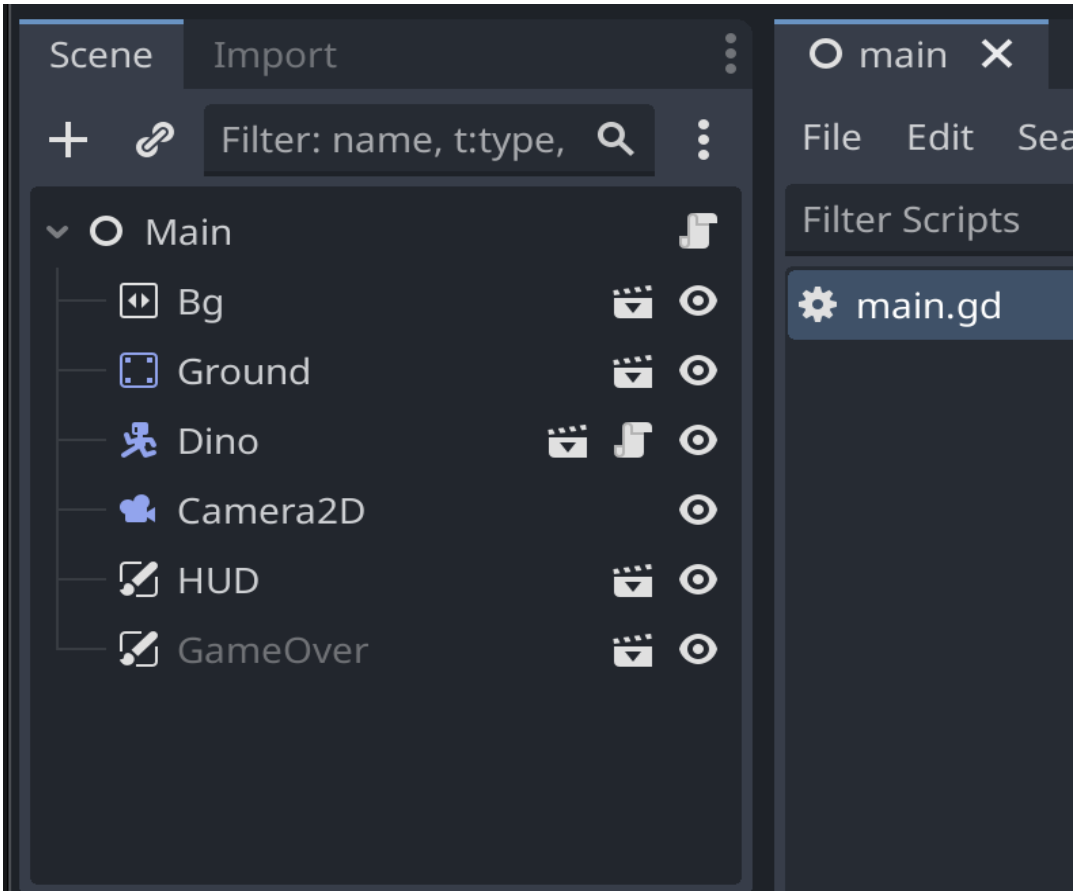
4. In this file there are 3 variables that you can tinker with
    a. `START_SPEED`
    b. `MAX_SPEED`
    c. `SPEED_MODIFIER`
5. Try to play around with these to see how it affects the game. `START_SPEED` is the speed at which the game starts, `MAX_SPEED` is the maximum speed the game can reach and `SPEED_MODIFIER` is the speed at which the game speeds up.
6. Save the file by pressing CTRL + S
7. Now you will be adding a new obstacle in the game, to do so, go to line 7 and uncomment it by pressing CTRL + / or by deleting the `#` symbol
8. After doing that add the variable `barrel_scene` to `obstacle_types` variable on line 10.
9. After adding your obstacle_types should look like this

```
var obstacle_types := [rock_scene,stump_scene,barrel_scene]
```

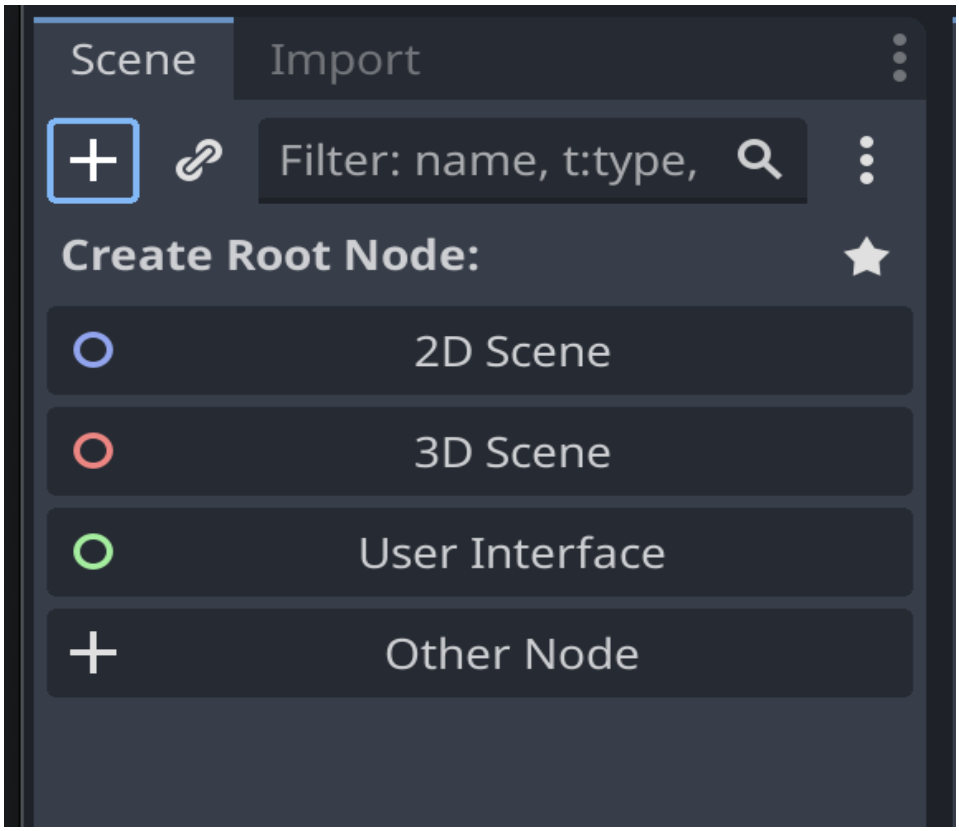
10. How has your game changed?

Creating Custom Obstacles

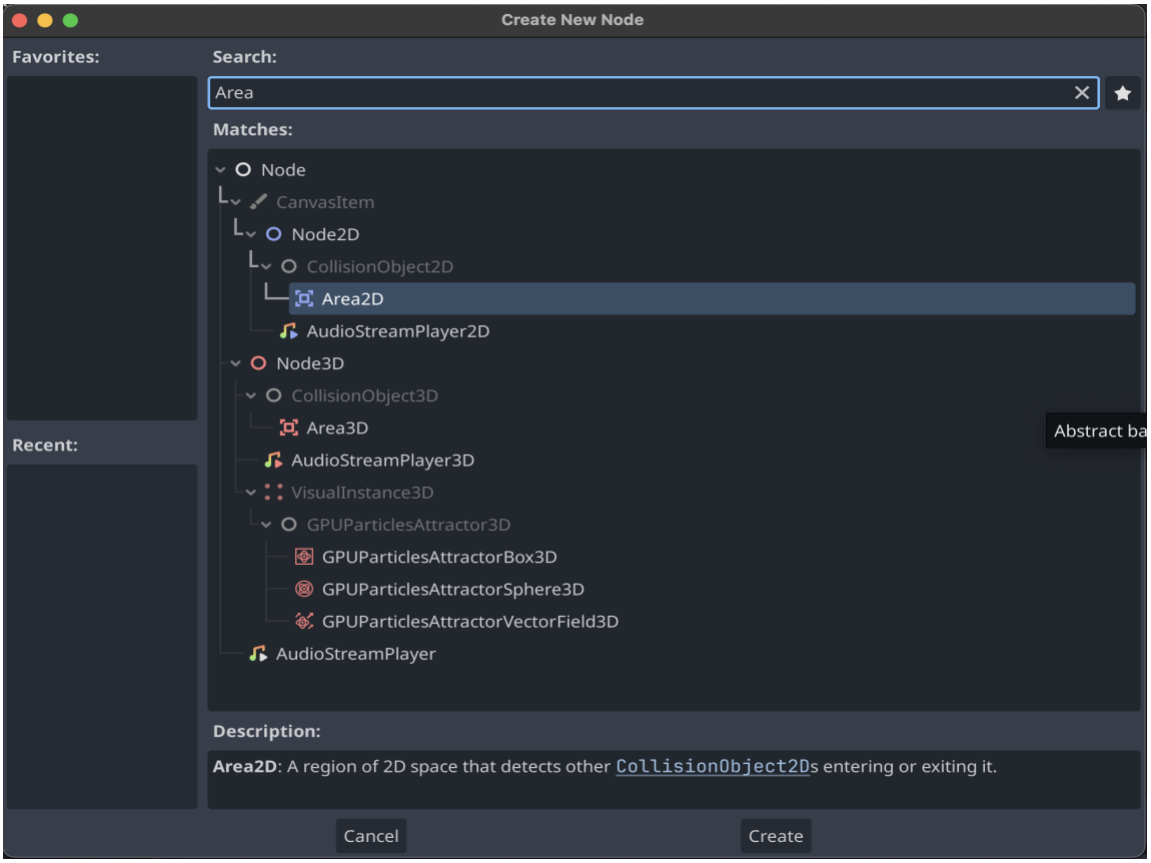
1. If there is a scene loaded in the scene panel, exit out of it by clicking the x in the viewport panel.



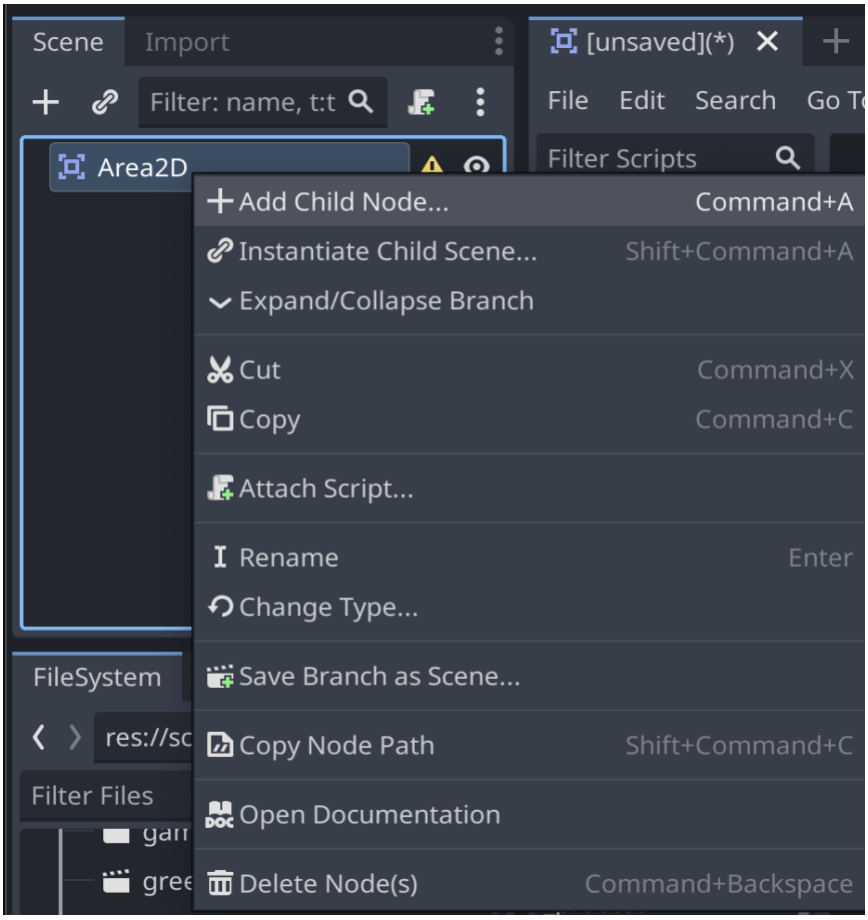
2. Now click on Other Node



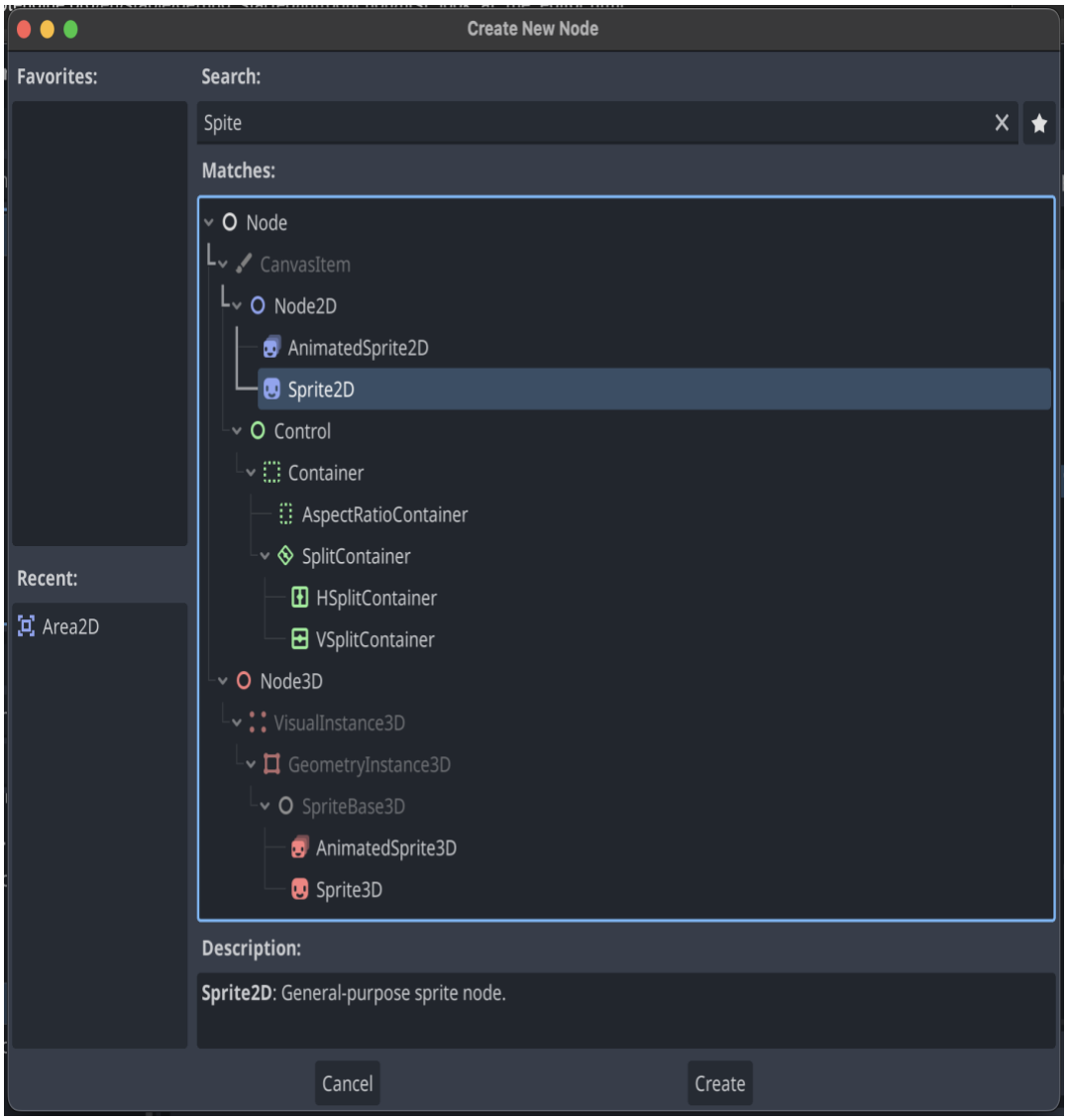
3. Search for Area in the top dialog box and then double click on `Area2D` and or click `Area2D` once and then click Create on the bottom



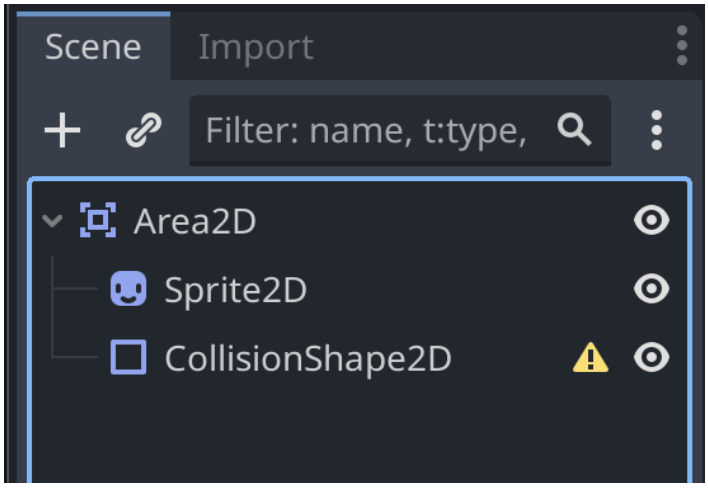
4. Now on the left pane, right click on Area2D and then click on Add Child Node.



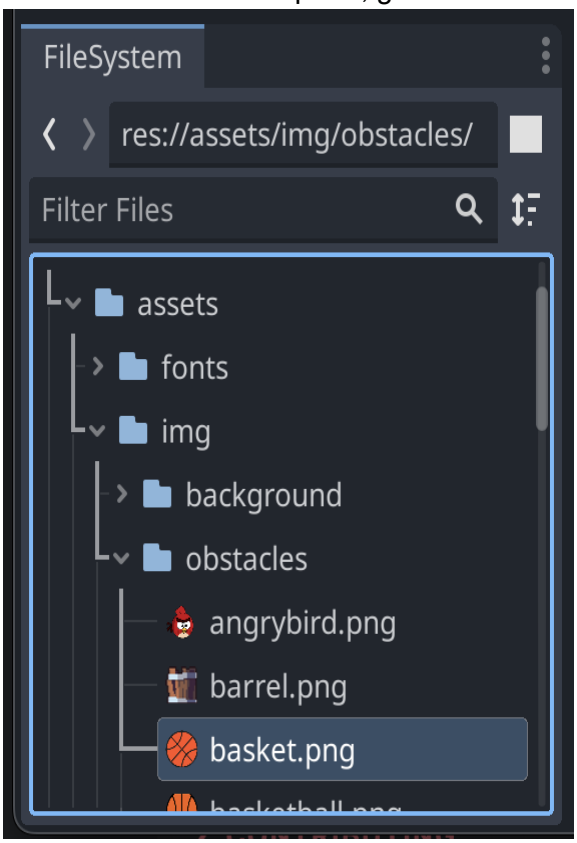
5. Type in Sprite in the search box and click on `Sprite2D`



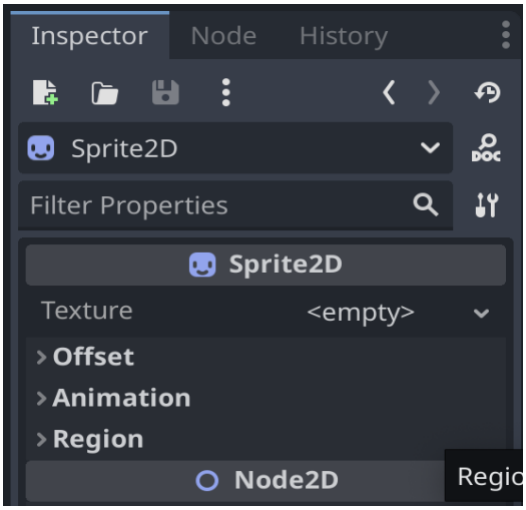
6. Now on the left pane, right click on Area2D and then click on Add Child Node.
7. Type in Collision in the search box and click on `CollisionShape2D`
8. If all the steps were done correctly then the Scene Panel should look like the following

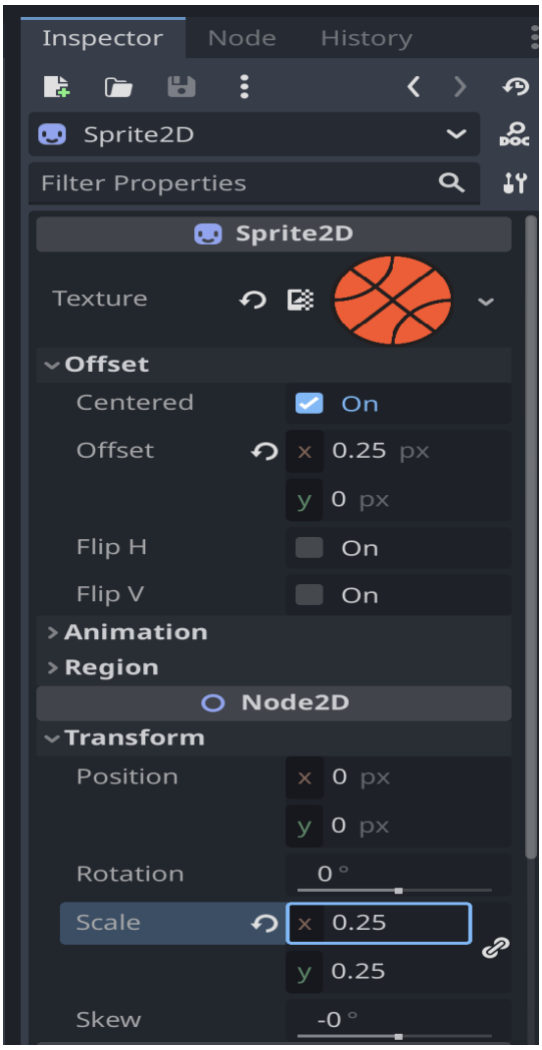


9. Now in the bottom left pane, go to assets -> obstacles -> basket.png

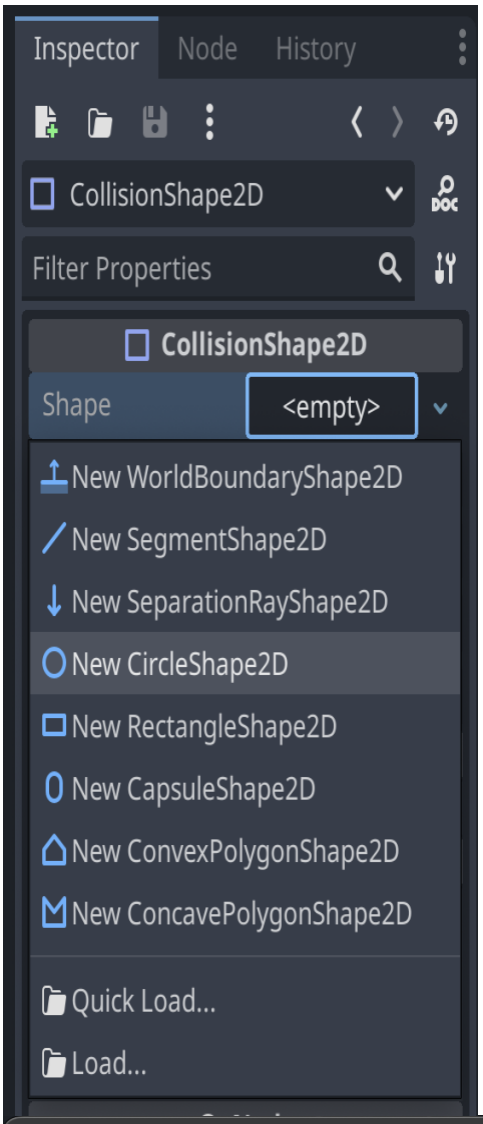


10. Click on Sprite2D in the Scene panel first.
11. Next, drag basket.png from the FileSystem Panel to the far right Inspector Panel and drop it onto the Texture box that says <empty>.

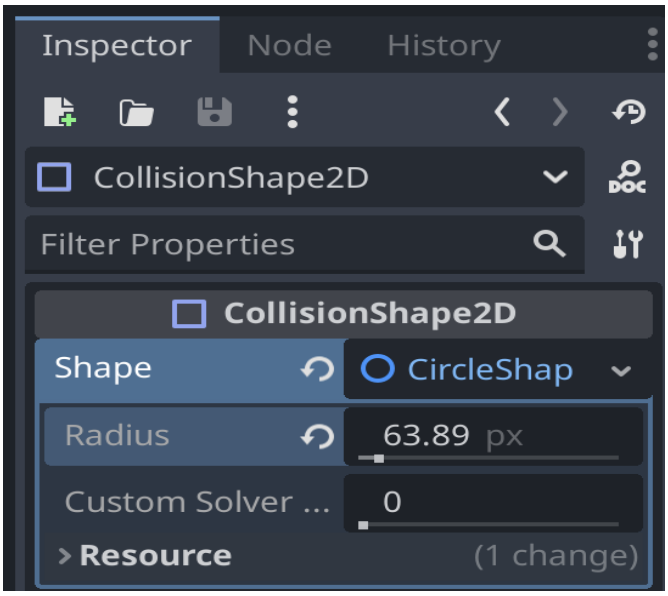




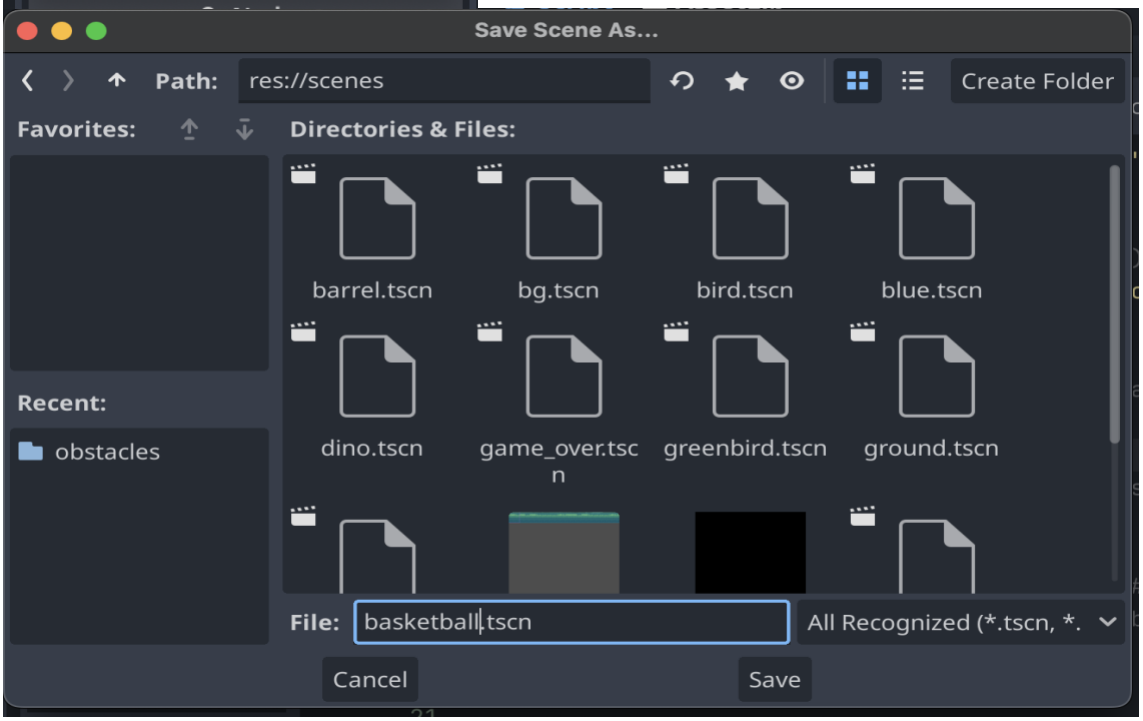
12. Now change the properties Offset to centered and Scale to 0.25 as shown in the image above. You may need to click on the right pointing arrow to open the property!
13. Now click on the `CollisionShape2D` and then click on the Shape property and then click on the New CircleShape2D option.



14. Now click on the `CircleShape2D` to change the shape. Next, click the Shape field again to open a drop down so you can change the Radius to 63.89



15. Now press CTRL + S to save the scene and place the scene in scene folder and give your scene a name as `basketball.tscn`



CHALLENGE 1

Adding the created Obstacle to Scene.

Instructions

Your challenge is to add the obstacle in a game scene:

1. Take a newly created basketball scene to the main scene using GDScript.
2. Make the basketball an obstacle in the dino_run game!

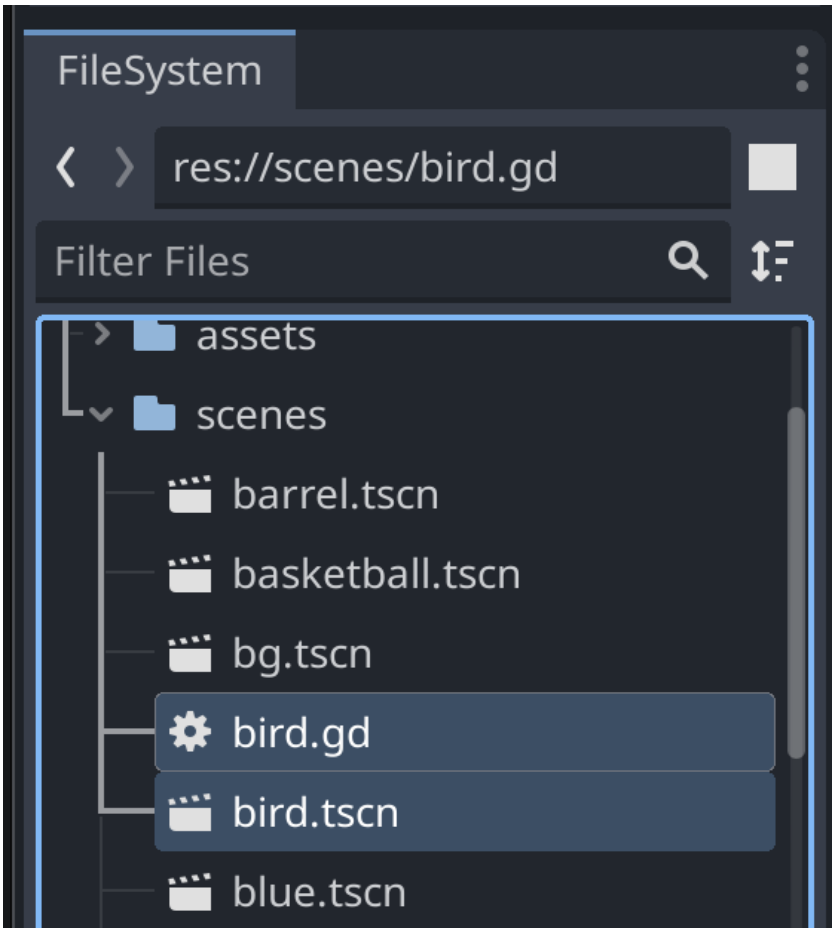
Hint 1: You can copy code that looks similar to what you might need!
Hint 2: If your obstacle isn't appearing but you loaded the scene, did you remember to add it to the obstacle list?

Once you've completed the challenge, try other ways to put the basketball into your game!

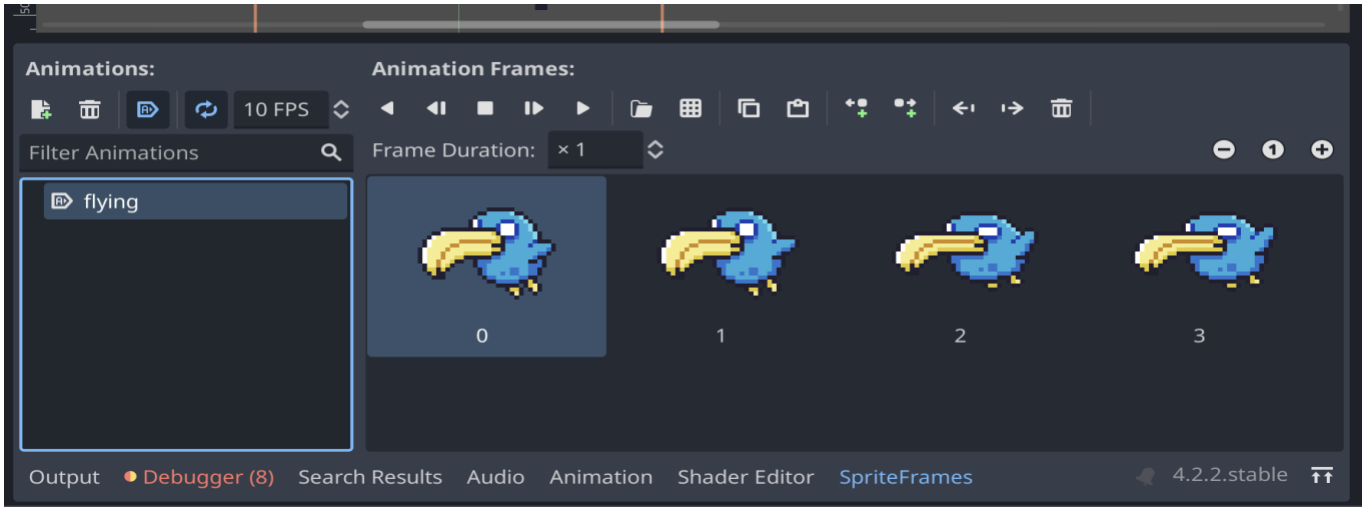
Animating the Bird

Now lets begin to Animate the Bird!

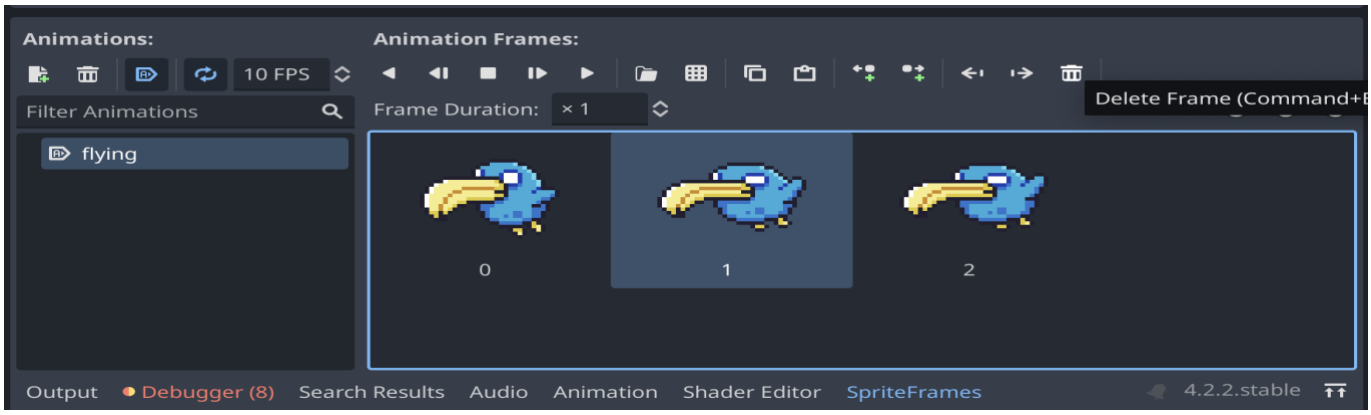
1. Go to the `bird.tscn` from the scenes folder on the bottom in the FileSystem Panel



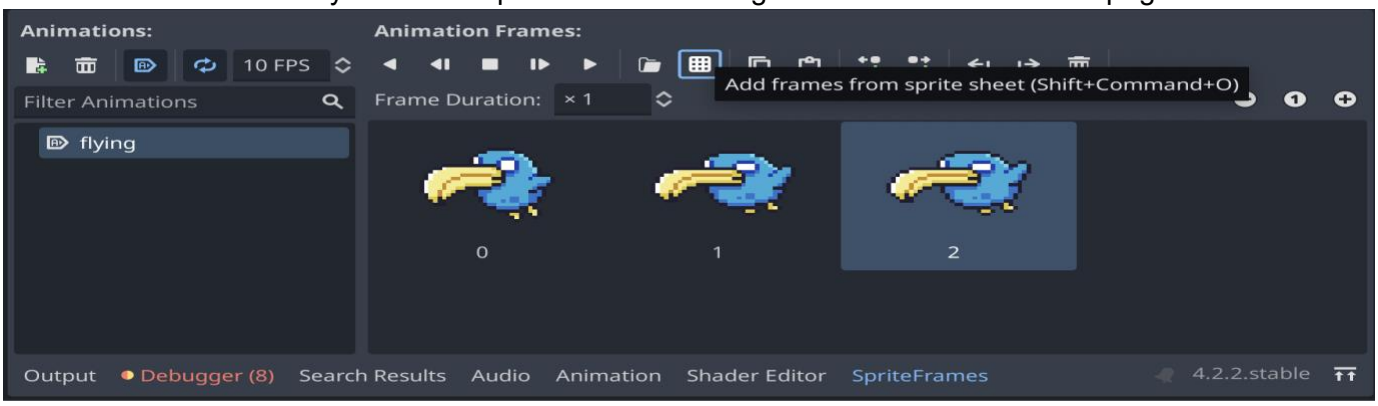
2. Click on the `AnimatedSprite2D` Node in the Scene Panel to open up the SpriteFrames Panel on the Debugger window on the bottom.
3. You can run the animation forward & backward at 2 speeds. These individual images are called frames.
    a. Does the dino character have frames too?



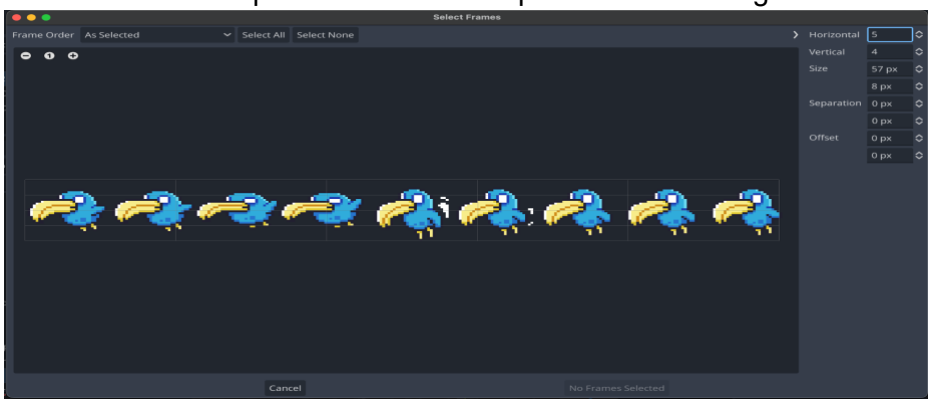
4. Let's edit the frames. Delete any selected frame(s) by clicking the trashcan to the right of the animation frame controls.



5. Now add frames from sprite sheet by click on the square next to the folder by the animation frame controls. Make sure you're in the path res://assets/img/obstacles and select bird.png.



6. If you need to zoom in click the plus button on the top left of the viewing window



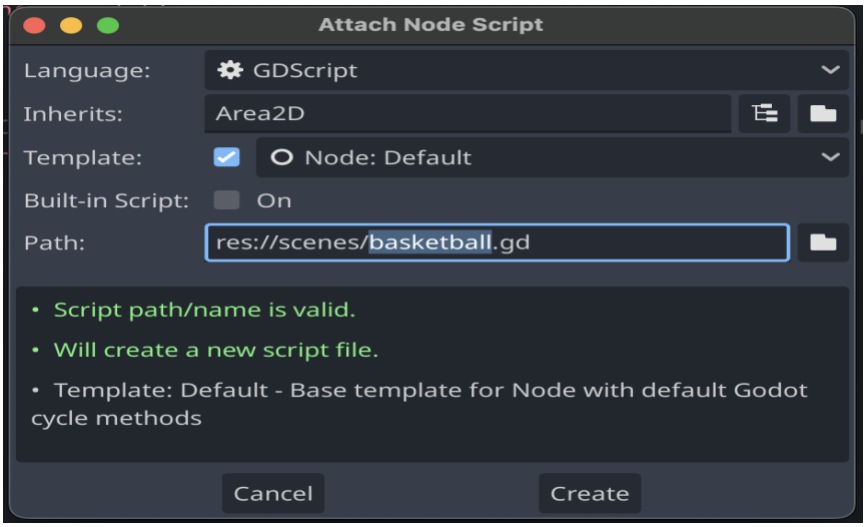
7. These frames are cut rather strangely so let’s fix the grid lines.
8. On the right text panel, change horizontal to 9 and vertical to 1 and then select the first four birds.
    a. What happens if you select them out of order?

9. If you don’t like the animation, try again! Delete a few frames, add new ones, or leave it as is.
10. Now Save the scene using Ctrl + S.
11. Using the FileSystem Panel open main.gd again.
12. Uncomment line 10 by CTRL + / or by deleting the #
13. Now uncomment line 11. This variable helps control bird's height
14. Now uncomment lines starting from line 118 to line 124. This will add bird to the scene at random.
15. Now press CTRL + S to save

CHALLENGE 2

Let's up the difficulty. The basketball is pretty boring not moving so let's make it rotate in place.

1. Place the basketball as an obstacle in your game scene (you might have done this in the last challenge!)
2. Add the frames to make the ball rotate.
    a. Add a new Node to your `basketball.tscn` in the Scene Panel by adding a new child node. Check with `bird.tscn` to see Node you're missing.
    b. To add sprite frames to a new `AnimatedSprite2D` node, click on the Animation dropdown in the Inspector Panel on the far right and select New Sprite Frames. Don't forget to click it again to pull up the SpriteFrames editor
3. Make a new script for the basketball scene by clicking the `basketball.tscn` to make sure it's open, then go to the Scene Panel and click Area2D. On the far right in the Inspector Panel scroll to the bottom to find the Node section and the sub-property Script <empty>. Click on it to create a new `.gd` file connected to the basketball scene!



4. For the rotating the basketball use the following script to add to basketball scene code:

```
extends Area2D
# Called when the node enters the scene tree for the first time.
var rotation_speed : float = 2 * PI

func _ready():
	# Start the rotation process
	set_process(true)

func _process(delta):
	# Rotate the basketball
	rotation += rotation_speed * delta
```

5. Feel free to adjust the rotation_speed variable to change how fast the basketball rotates.
6. Once you've completed the challenge you might need to adjust the size of the basketball. Play with the scale in the Transformation section in the Inspector Panel to get it just right.

Hint 1: Check out how we adjusted the frames in the bird scene to select the frames for the basketball!
*Hint 2: If your code is giving you errors, remember to indent or tab the function body in, like set_process(true) and rotation += rotation_speed * delta as shown above.*
Hint 3: Once you've added the rotation code to the basketball scene, don't forget to make sure the basketball scene is in the obstacle list.

Object Runner

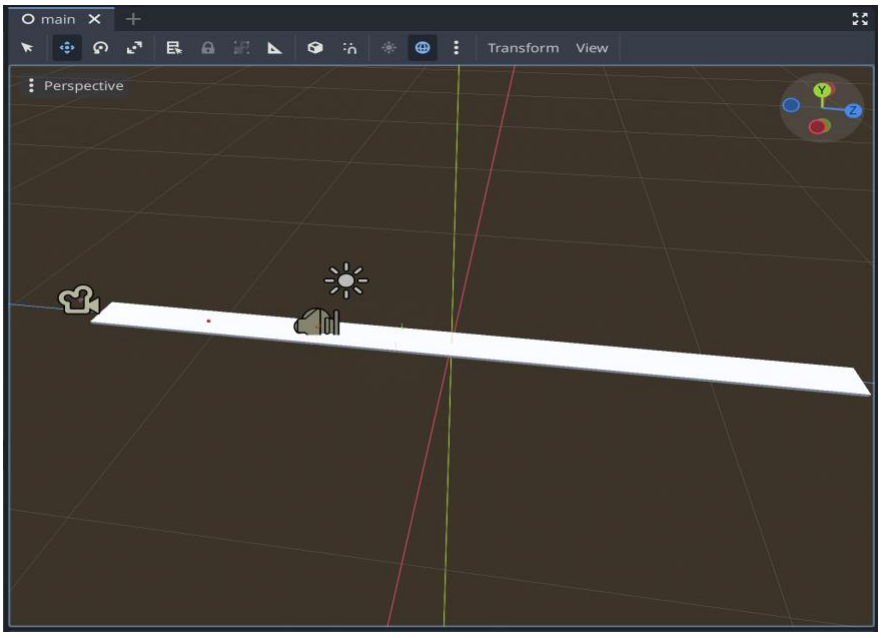
Opening the project

1. Open Godot_v4.2.2-stable_win64 from Godot_v4.2.2-stable_win64.exe folder
2. Click on Import at the top of the window and navigate to the 3D game or enter the following path in the dialog box: /Users/io/Repositories/ASU-SUCCESS/GameDev/GodotGame/3d/3D game and click on Select Current Folder button on the bottom or double click the `project.godot` file.
3. Click on Import.
4. Open the main scene (`main.tscn`) and click 3D at the top to make sure the level loads.
5. Let's test and run the game to make sure everything is working fine. Click on the play button. Did you know instead of the arrow keys you can use WASD keys?

Movements in the Scene

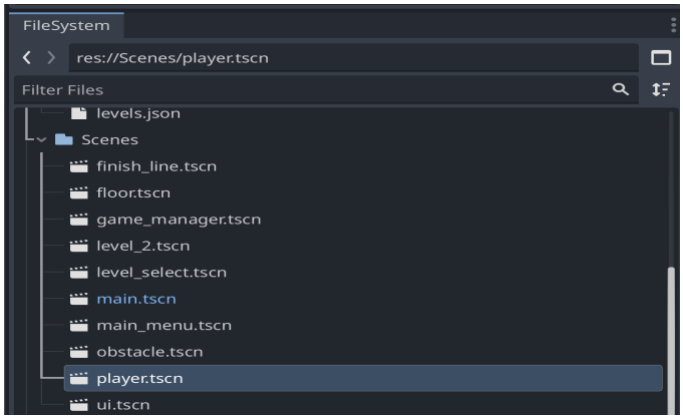
As you build a scene in a 3D perspective you'll need to learn how to move your viewport around the scene so you can place more components like obstacles, players, and other objects.

1. Open `main.tcsn` and click on 3D in the top bar if you haven't already so you can see how things change as you do the following actions:
2. Right Click and hold and while pressing any one of the following key on the keywords for movement.
    a. To move closer to the scene press W.
    b. To move further away to the scene press S.
    c. To move Left press A.
    d. To move Right press D.
    e. To move Up press E.
    f. To move Down press Q.
3. This approach is great for small movements, but if you're finding it moving too slow try using the scroll wheel or a double finger scrolling or zooming motion on a mouse pad to zoom in or out.
4. You can also click and drag the globe with coordinate axes (X, Y, Z) around to move the world around too!



Changing Parameters

1. Let's change the values of the game again.
2. To increase speed of the player, go to the `player.gd` script and click on Script button at the top.



3. Change the value of speed line 9 to speed the player up or slow it down.
4. Now if you want to make the player jump at place, go to line 46 through 48 and uncomment it using Ctrl+/ or remove the #

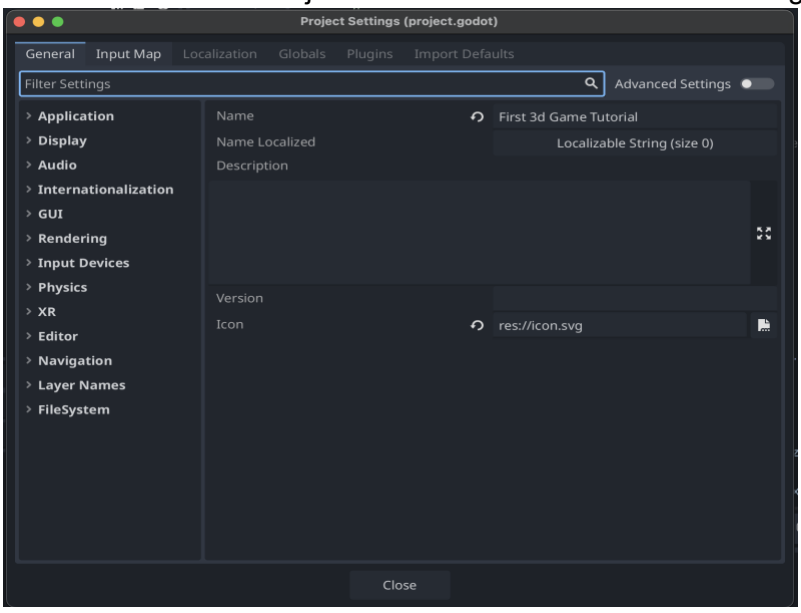
```
41  func _input(event):
42      # Change my key bindings
43      input = Vector3(Input.get_action_raw_strength("left") - I
44
45      # Now add jump effect to body
46      #if Input.is_action_just_pressed("jump") and can_jump:
47          #apply_central_impulse(Vector3(0, JUMP_FORCE, 0))
48          #can_jump = false
```

5. Now save the script using Ctrl+S and click on play button to test out your changes!

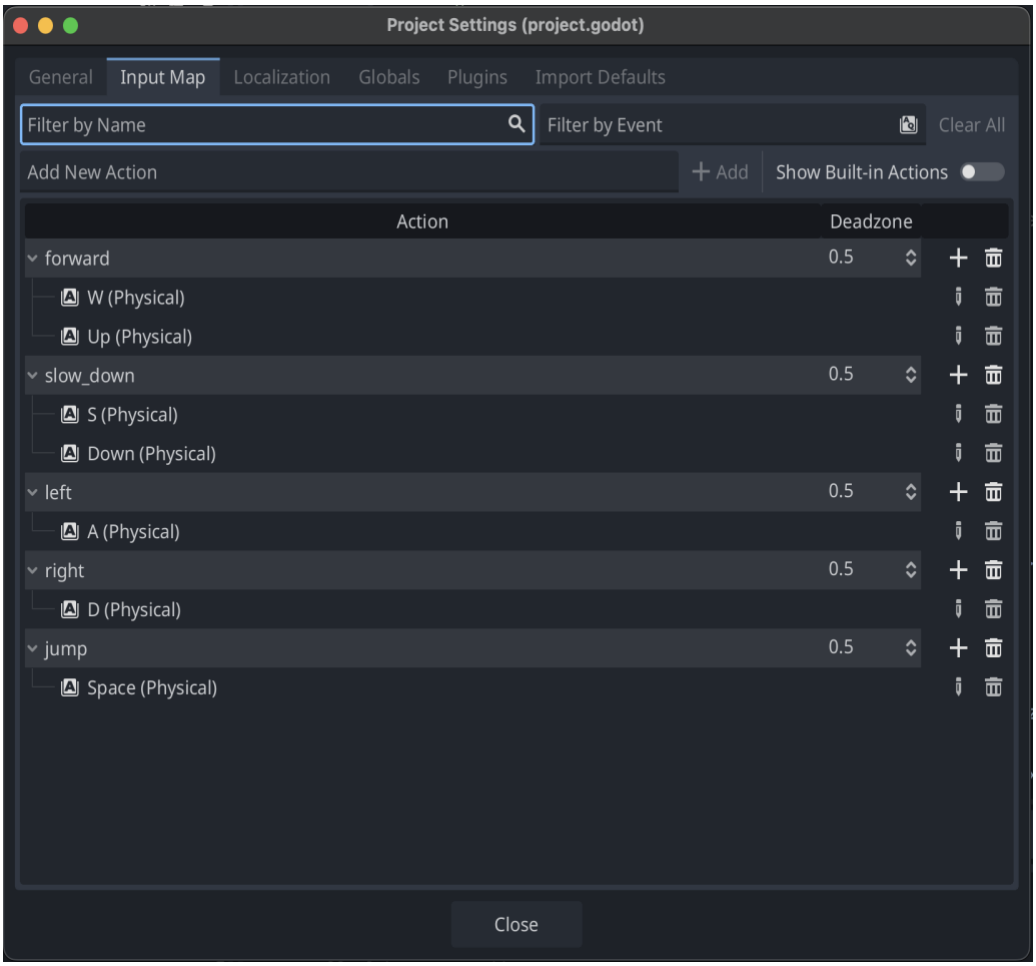
Changing Key Bindings

Did you like using WASD? You probably noticed the left and right arrows didn't work. Let's get them working!

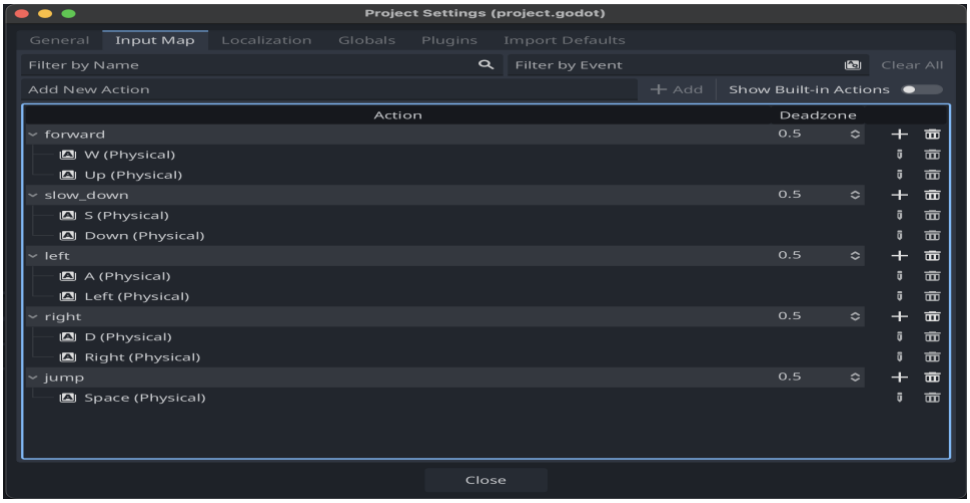
1. Click on the Project from the Navbar as shown in image and then click on Project Settings



2. Click on Input Map



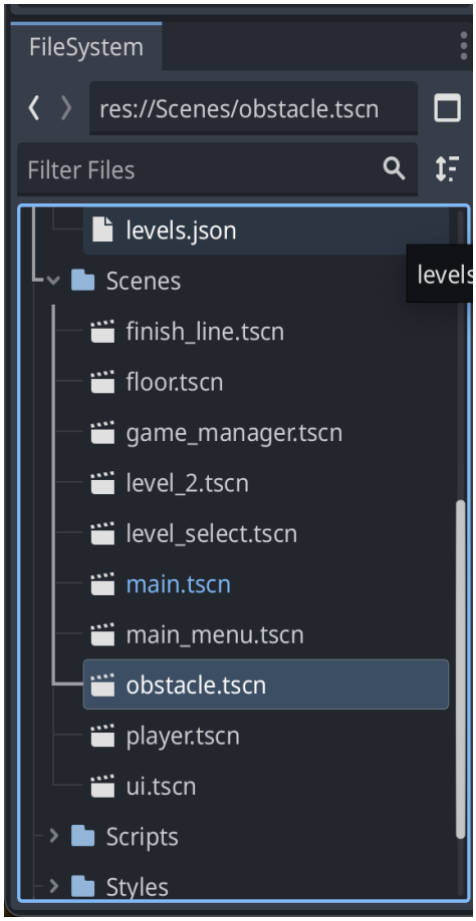
3. Under Action you will be able to see for what actions are selected for what keys.
4. Now click on the Pencil icon and press the key for the action you'd like to change.
    a. For example, left should use the left key! Press the left key and then click ok.
5. You can also add extra keys for an action by clicking on the + icon for the type of action
    a. For example, you can add a new left action back in by pressing A and then clicking ok.



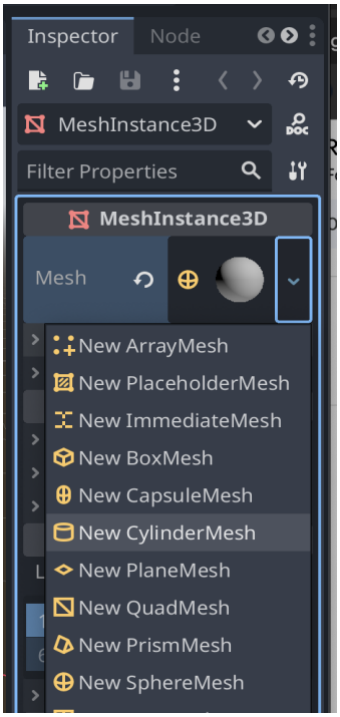
6. Here's how the actions should look like if you want to fix the arrow buttons!
7. Don't make an action share a key with another action. Or do, and see what happens to your game
    a. For example, making forward and left actions both use the W key. What happens to your game?
8. Click on OK and click on Play Button. Play the game according to your new key bindings.

Customizing the Obstacle

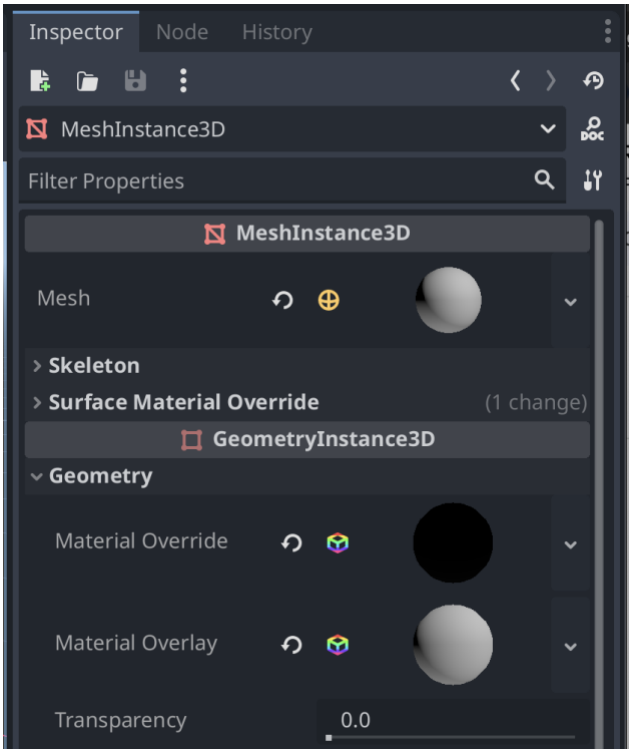
1. In the Scenes folder, double click on `obstacle.tscn` file and then click on 3D at the top.

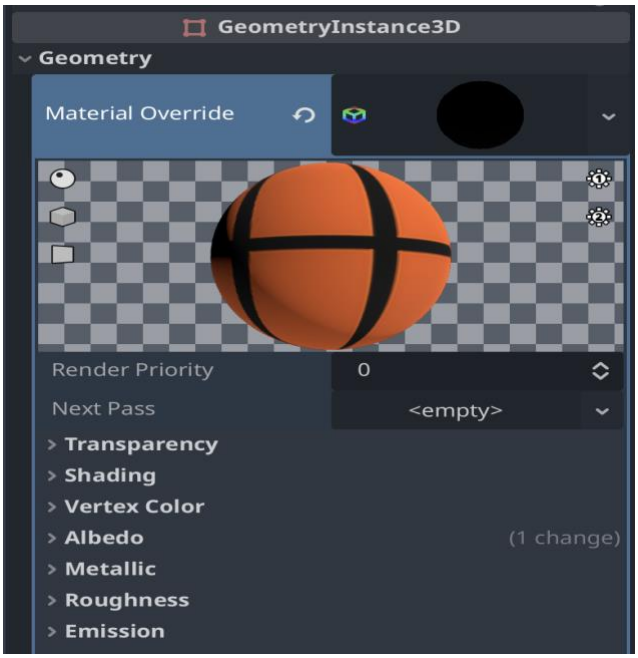


2. Click on `MeshInstance3D` on Scene Panel and look at the Inspector Panel on the right.
3. In the Inspector Panel select Mesh and choose any of the solid shape meshes like New `BoxMesh`, New `CapsuleMesh`, New `CylinderMesh`, New `SphereMesh`. Remember which solid shape you chose, you'll need it later.

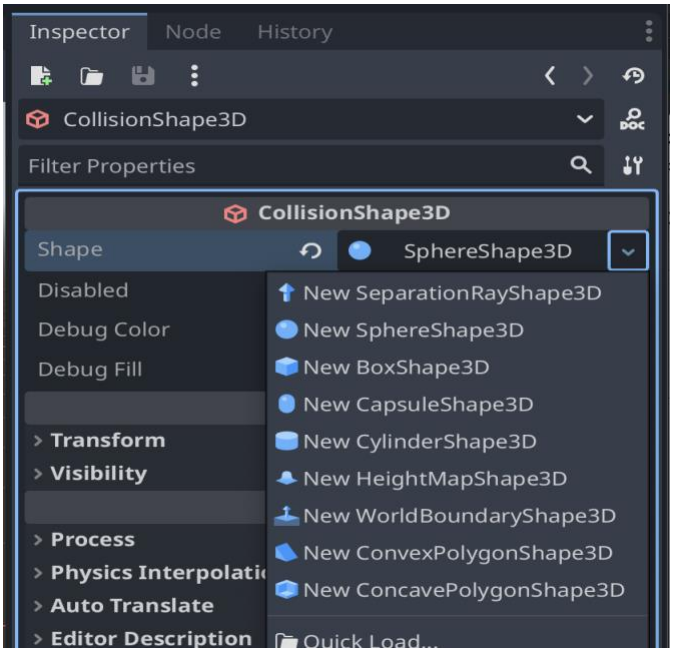


4. Now expand the Asset folder from the FileSystem Panel on the left and drag the basketball image to the Material Override and Material Overlay under GeometryInstance3D. Make sure the image is there by clicking the down arrow on the right for each. You can move the image around for each by clicking and dragging on the image in each section.





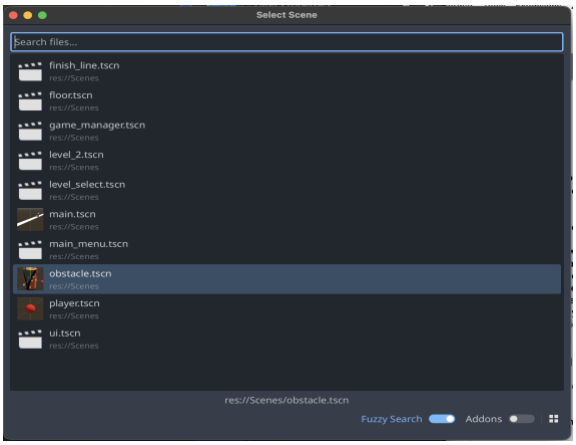
5. Go back to the left to the Scene Panel and click on `CollisionShape3D`. In the Inspector Panel on the right click the drop down of the Shape section to click the mesh shape that you chose in `MeshInstance3D`. For example, if you chose `CapsuleMesh`, make sure you choose `CapsuleShape3D` here.



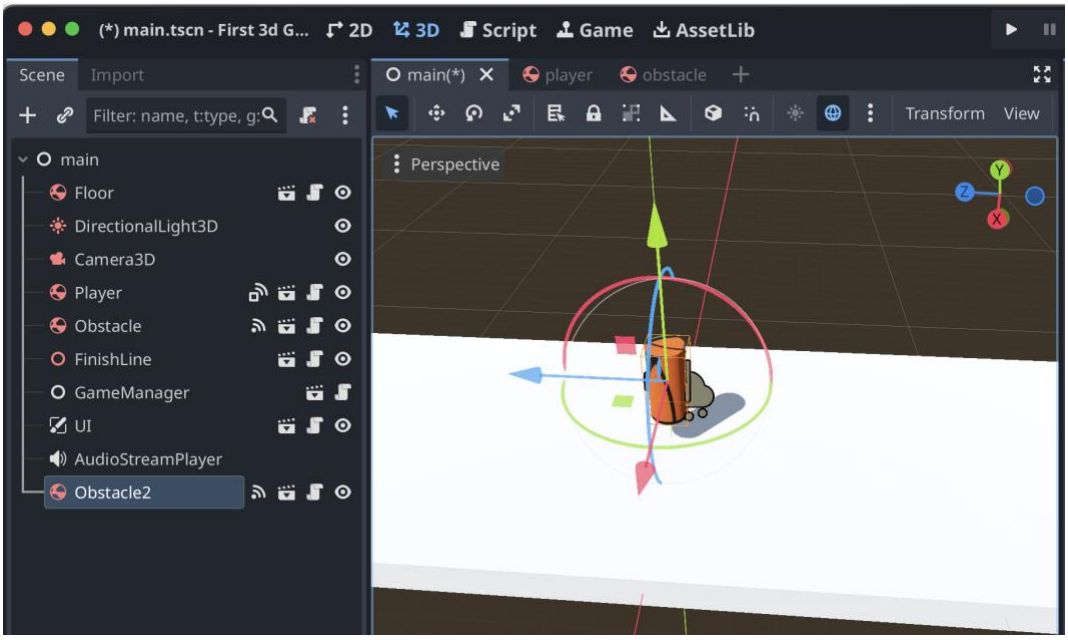
6. Save the Obstacle using Ctrl+S.
7. Click on Play button to the see the changes in the game. This is a cylinder example!

Adding an Object to the Scene

1. Double click on `main.tscn` and make sure that main appears in the Scene Panel
2. Click main, the top level item in the panel.
3. Click the chain link next to the plus sign at the top to Instantiate Child Scene and select an Obstacle from the scene.



4. You will see your obstacle appears on the viewport scene in the 3D window and also is nested in the Scene Panel under main



5. Now click on the obstacle under main and you will see three axis visible.Green will be Y-axis(position.y), Red will be x-axis(position.x),Blue will be Z-axis(position.z).
6. Drag any of the axis and place the obstacle at the desired position make sure it is within the floor.
    a. Don't forget you can move your view around by right clicking and holding while you press a key, or use the globe at the top right
7. Save your scene with CTRL + S.

CHALLENGE !!!

Respawn the Player

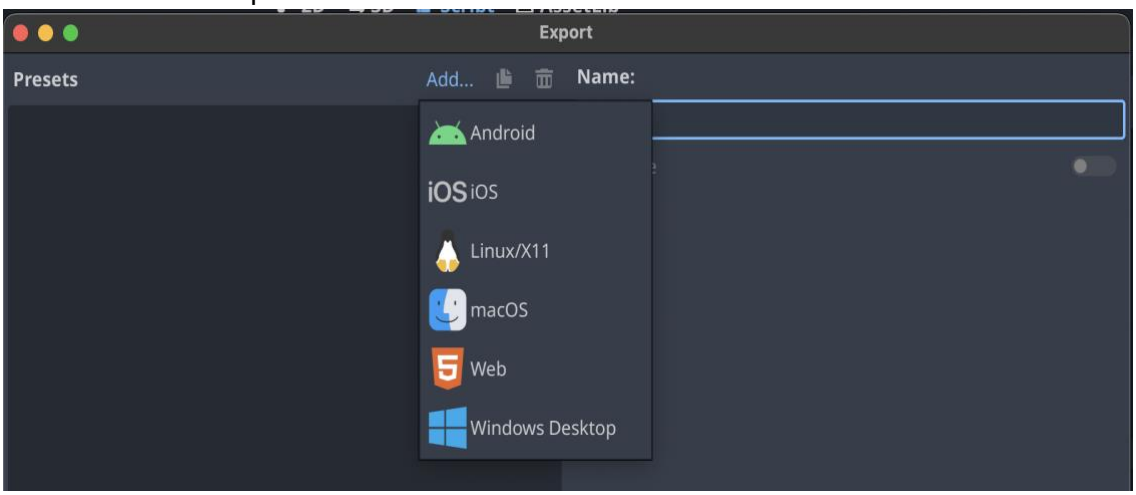
Instructions

Your challenge is to make the player respawn when they fall off the floor:

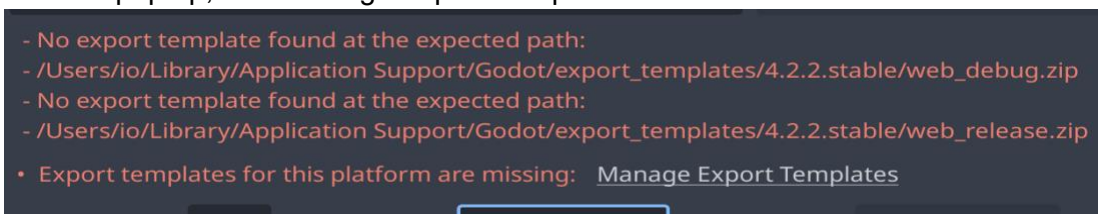
1. Open the `player.gd` script.
2. Find the line of code related to respawning the player.
3. Uncomment that line to enable the respawn functionality.
4. Don't forget to test your changes out!

Exporting to the Web

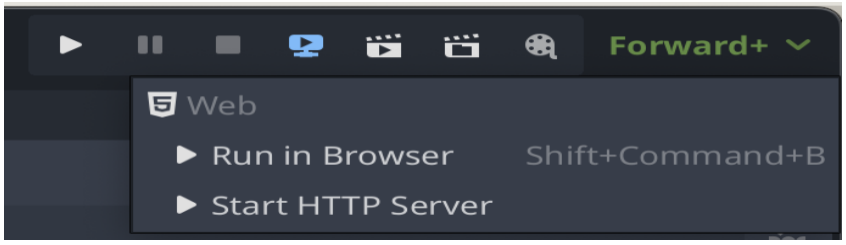
1. Click on the Project from the very top and then click Export...
2. Click on Add... besides presets and select Web



3. If you have errors pop up, click Manage Export Templates



4. Click on Go online and then Download and Install Preset. It may take a few minutes.
5. You may need to close this window and click Project, then Export… again
6. Select web and then click export project.
7. Close the window and go to the top bar on the right to click the fourth button that has a monitor with a play icon.



8. Click Start HTTP Server
9. Click the top button with the monitor and play button again and then click Run in Browser
10. Now you can play your game on the web!
11. When you're done playing, remember to go back to that button to click Stop HTTP Server