

Sign in here!



DevLUp FSU

GBM #2

Intro to Godot

January 30, 2025

Welcome to DevLUp at FSU!



Meeting Schedule

Date	GBM #	GBM Title	Secondary Event	Presenter
9 Jan		(No Meeting)	(oops, no involvement fair!)	
16 Jan	1	Introductions and Design Activity		All
23 Jan		(No Meeting)	SNOW!!!	
30 Jan	2	Intro to Godot (Orbital Odyssey Minigame)		Dion
6 Feb	3	Intro to Game Design	ASLC Showcase	Jake
13 Feb	4	Art Fundamentals (for artists and non-artists)		Parker
20 Feb	5	Accessibility in Games		Ares
27 Feb	6	Godot Tidbits		Dion
6 Mar	8	Intro to Unity		Jake
13 Mar		(No Meeting)	Spring Break	
20 Mar	7	Intro to Stencyl (Point & Click)		Whalen
27 Mar	9	1 Hour Game Jam (or Design Sprint)		Whalen
3 Apr	10	Game Jam Fundamentals	Game Jam?	Dion
10 Apr	11			
17 Apr	12		Innovators Showcase	
24 Apr	13			
1 May		(No Meeting)	Finals	

ASLC Indie Games Fest!

- We're tabling for DevUp and showing off some of our member's games
- **Sunday, Feb. 9th, 2-6PM**
- Submit your game to us (here or in the announcement)



Description

Who's in the mood for something a little bit more lowkey? Join game committee and explore some fantastic Indie titles that have come out in recent years including Balatro, Hollow knight, Bug Fables, and a true Indy game: Indiana Jones and the Great Circle. You don't want to miss out!

Indie Games Fest



Date and Time

Sunday, February 9 2025 at 2:00 PM EST to Sunday, February 9 2025 at 6:00 PM EST
Add To [Google Calendar](#) | [iCal/Outlook](#)



Location

Askew Student Lifer Center
942 Learning Way , Tallahassee, Florida
[View Map](#)

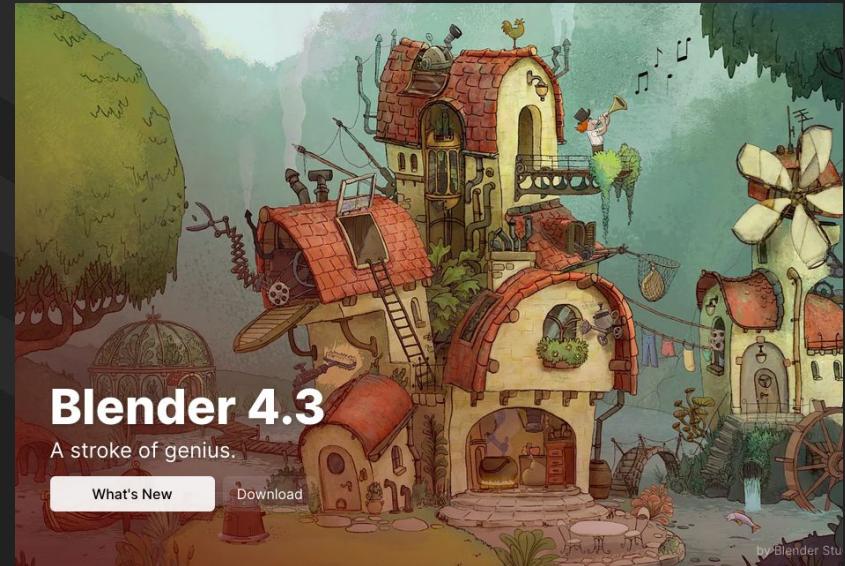
RSVP to Event

Figure χ: They haven't made one of those cool promotional graphics yet :(

Download Blender

<https://www.blender.org/>

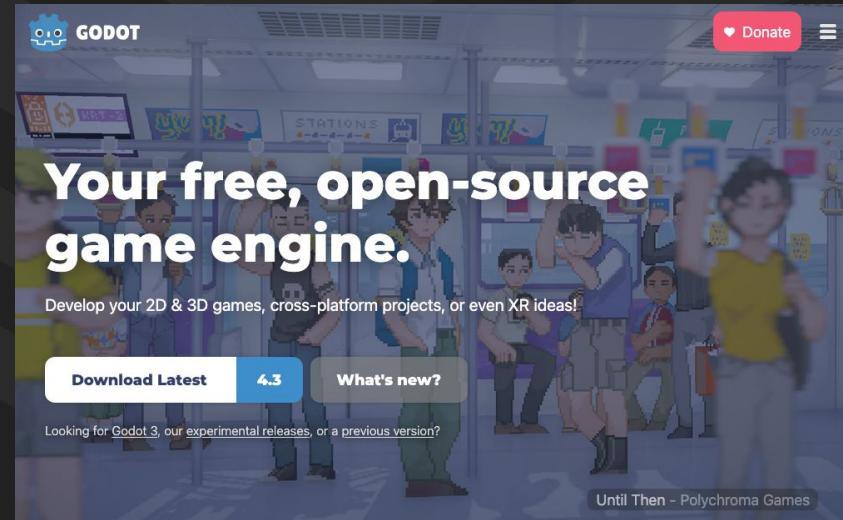
Latest version (v4.3)



Download Godot

<https://godotengine.org/>

Latest version (v4.3)



Download Orbital Odyssey Starter Files

<https://github.com/devlup-fsu/party-game/archive/831e1a11a1533a874551dd58469e2fea1349b7e1.zip>



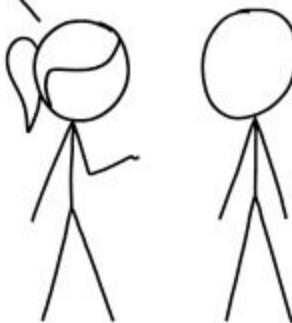
# showoff

Recommended Further Learning



SILICATE CHEMISTRY IS SECOND NATURE TO US GEOCHEMISTS, SO IT'S EASY TO FORGET THAT THE AVERAGE PERSON PROBABLY ONLY KNOWS THE FORMULAS FOR OLIVINE AND ONE OR TWO FELDSPARS.

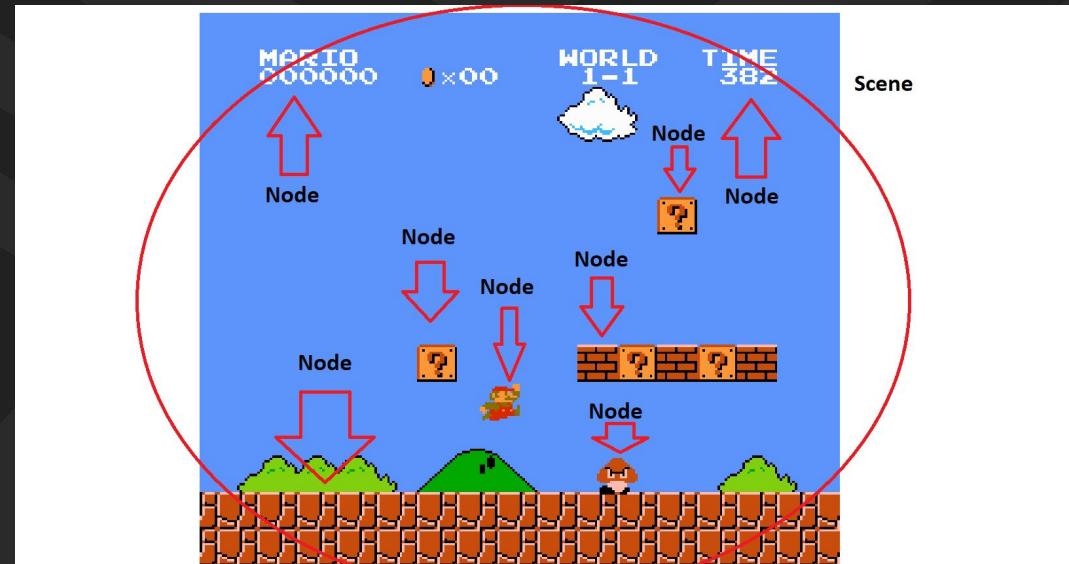
AND QUARTZ, OF COURSE.
OF COURSE.



EVEN WHEN THEY'RE TRYING TO COMPENSATE FOR IT, EXPERTS IN ANYTHING WILDLY OVERESTIMATE THE AVERAGE PERSON'S FAMILIARITY WITH THEIR FIELD.

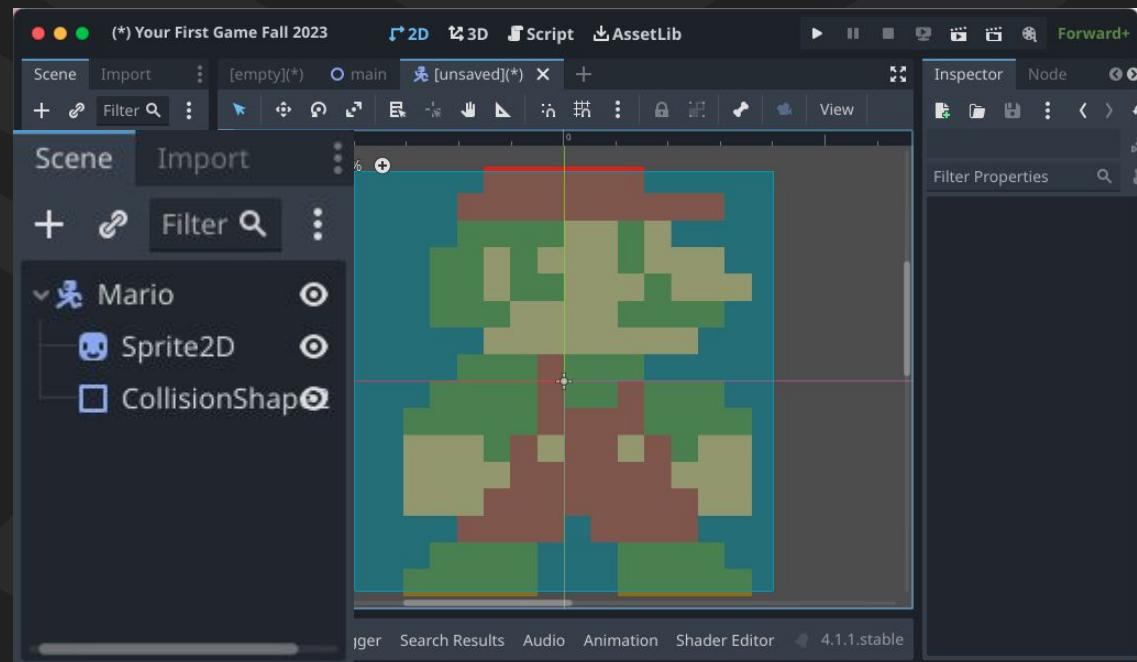
Scenes and Nodes

- The building blocks of a game
- The nodes are the individual parts of the game from Mario to the score to the enemies on the stage
- Can think of the scene as the level of a game or a collection of nodes



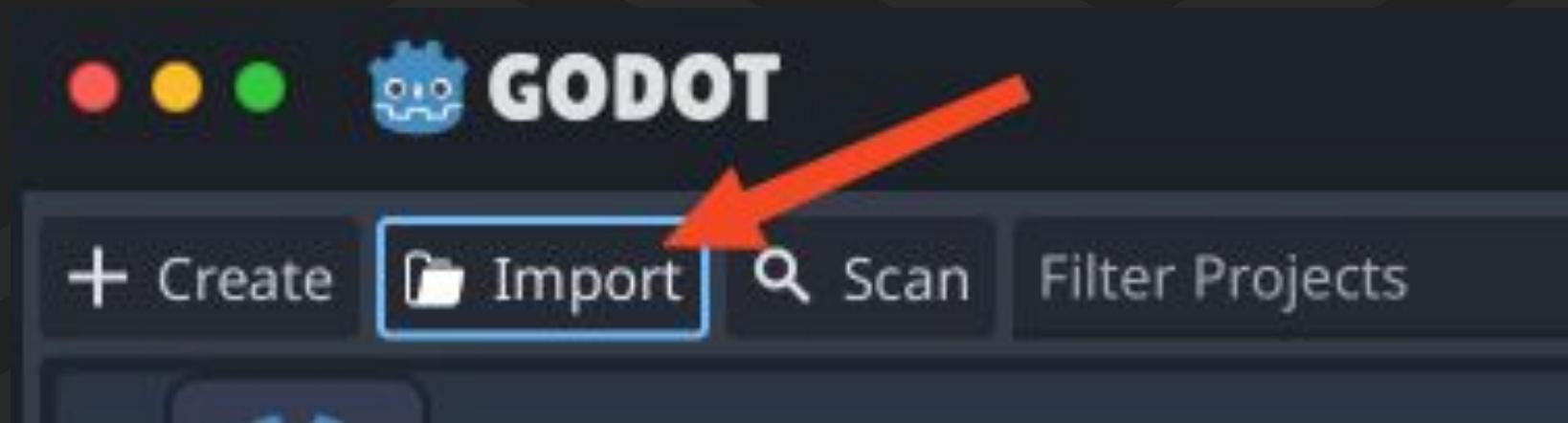
Nodes are Scenes

- Nodes can also have children too
- Children nodes can contain functionality like the collision of the parent node or the texture

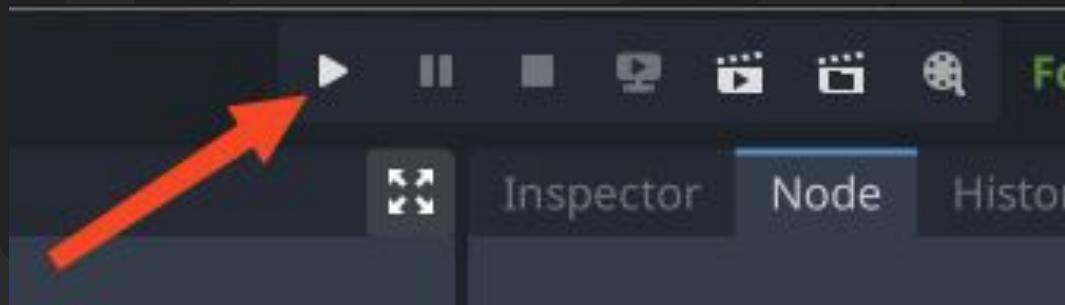


Setup the Project

Import Orbital Odyssey



Run the Game

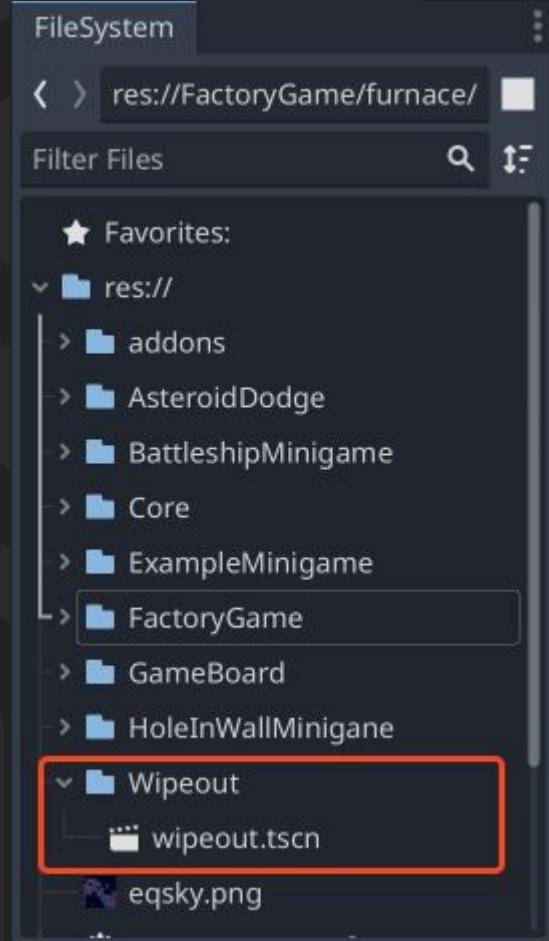


Making a Minigame

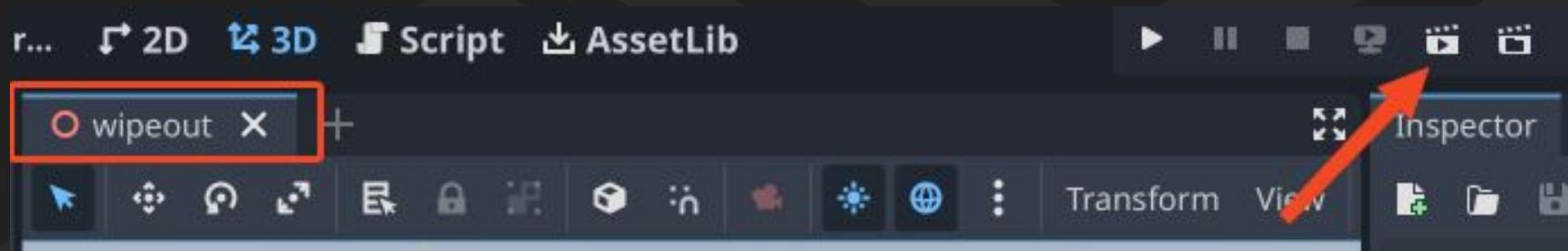
What We're Making Today



Each Minigame Has Its Own Folder

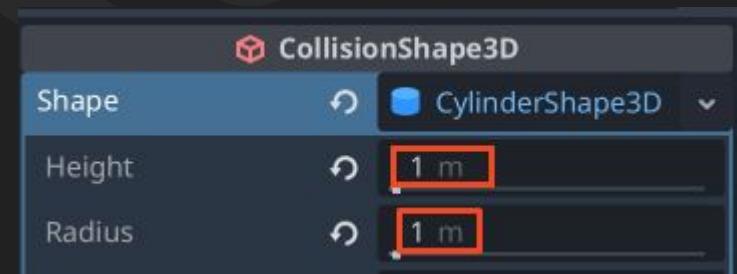
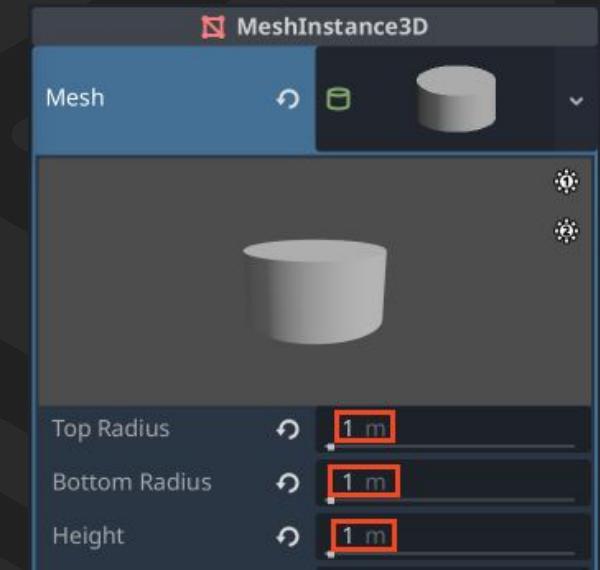
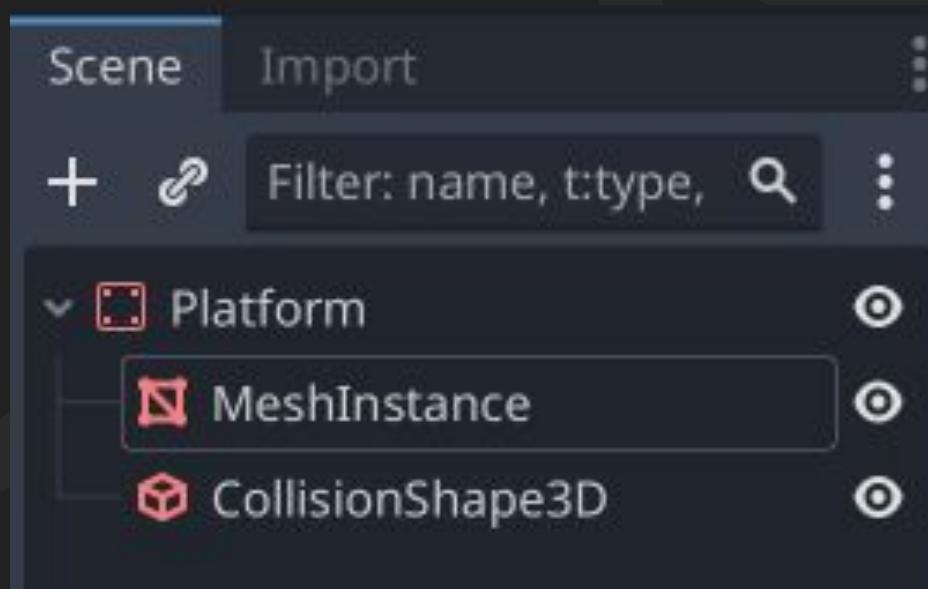


Run Your Minigame Directly

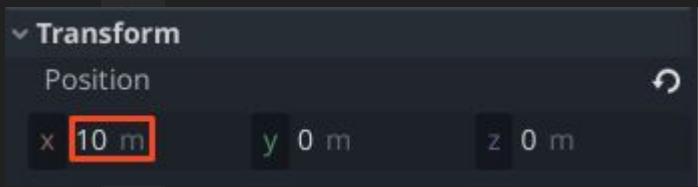
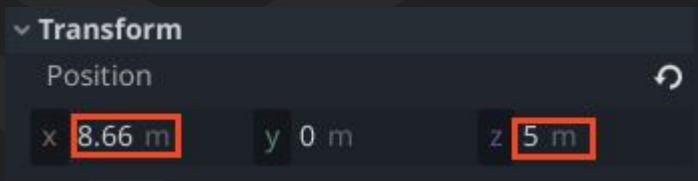
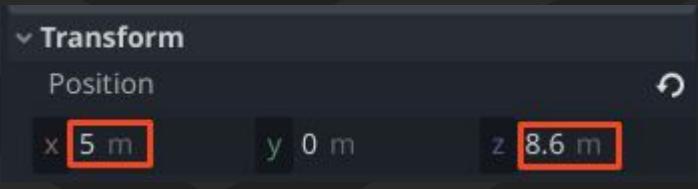
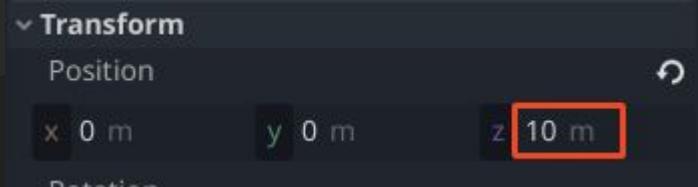
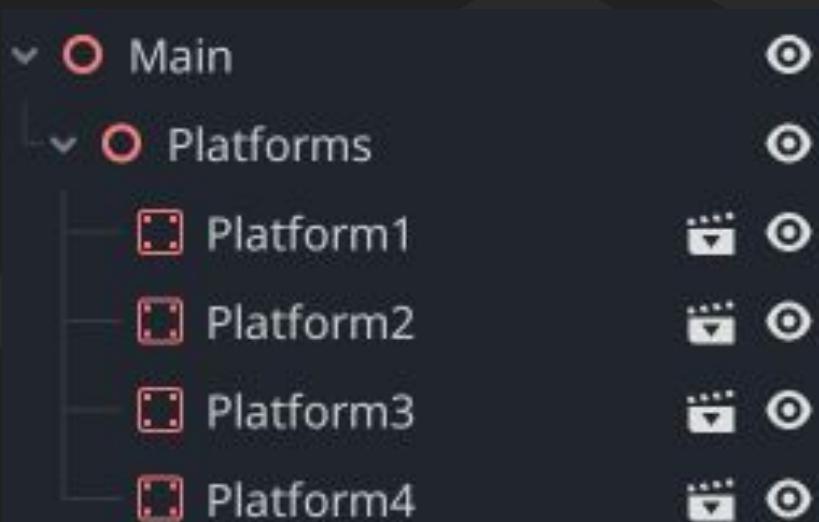


Creating the Platforms

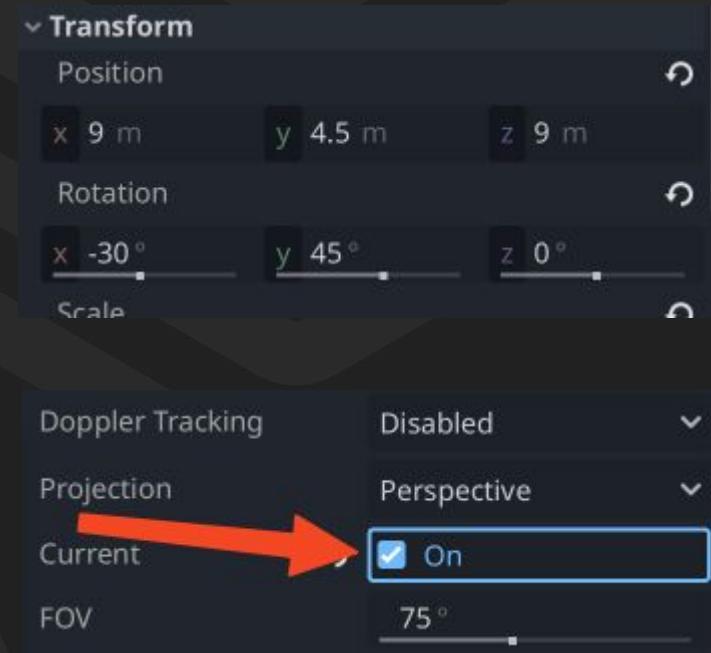
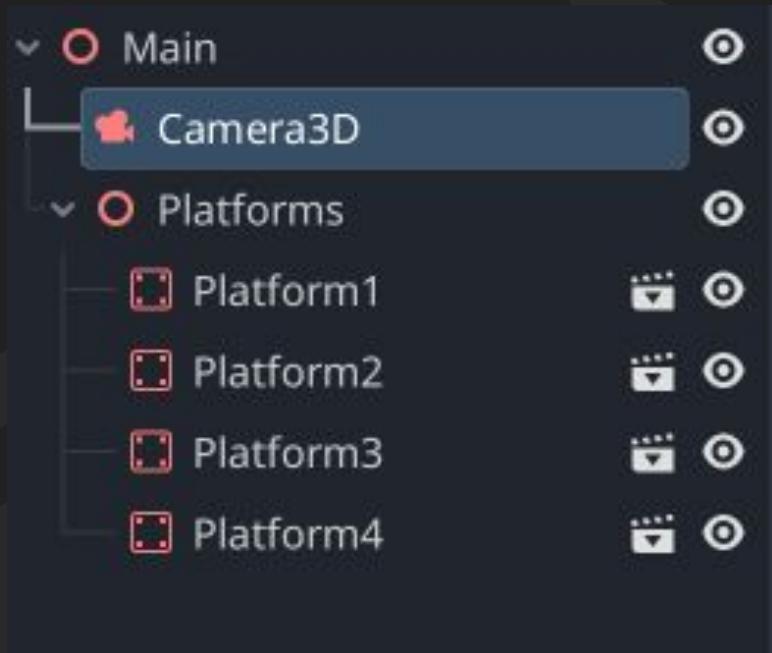
Create the Platform Scene



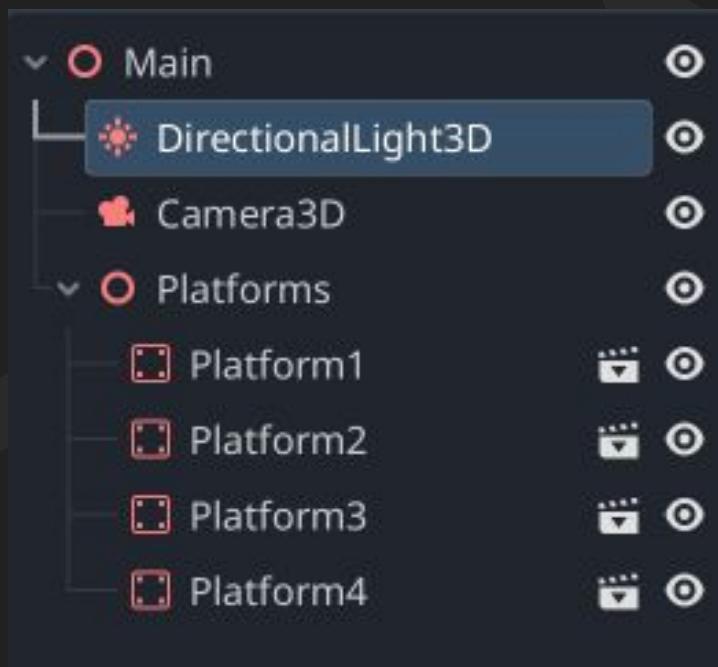
Place the Platforms



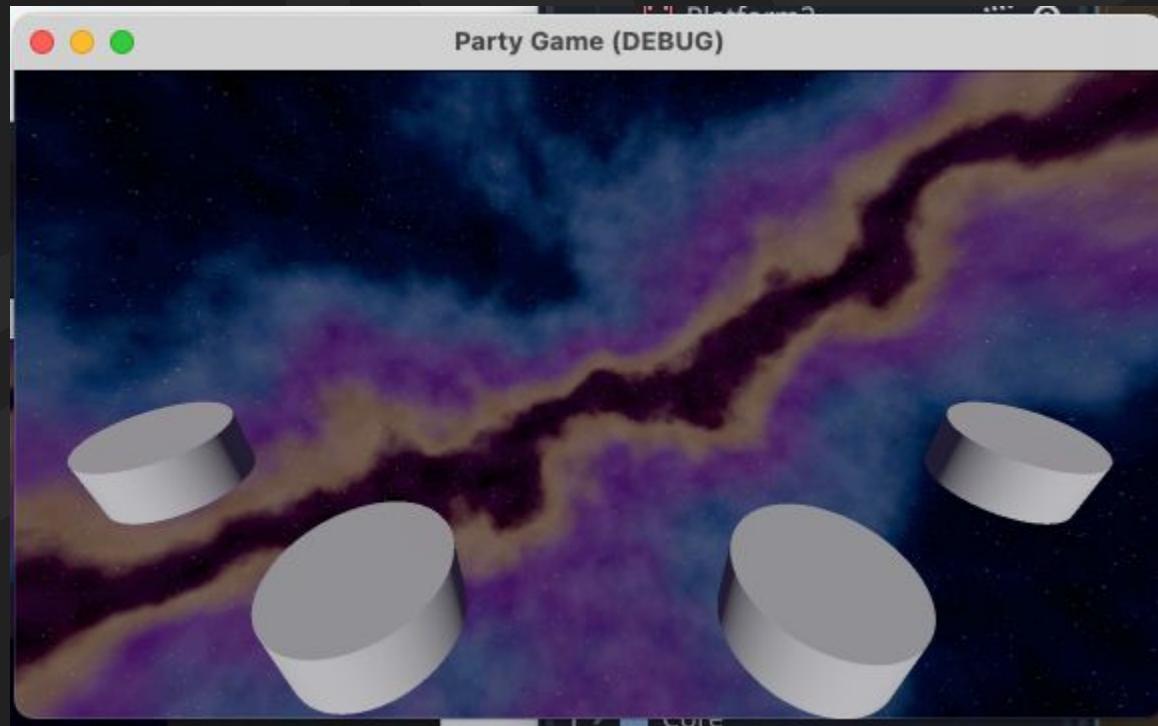
Add a Camera



Add Light

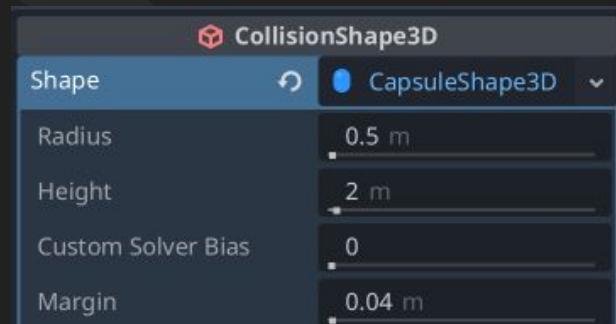
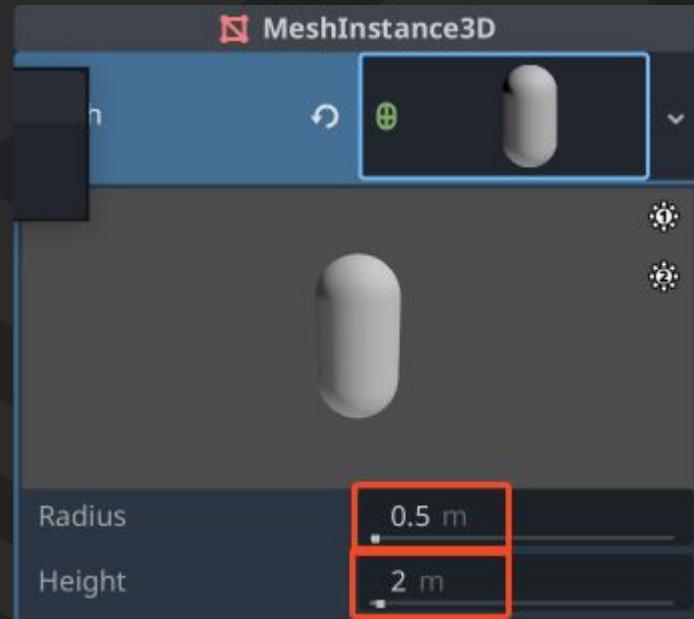
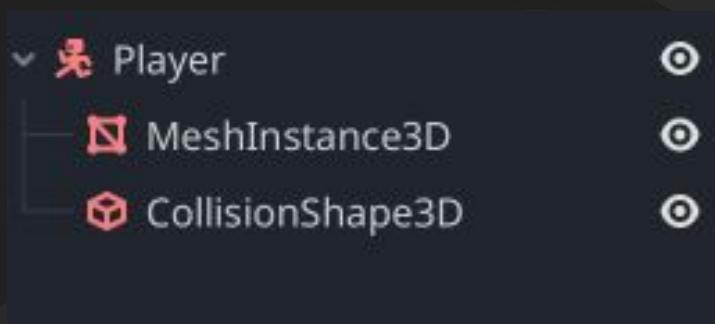


Run the Game

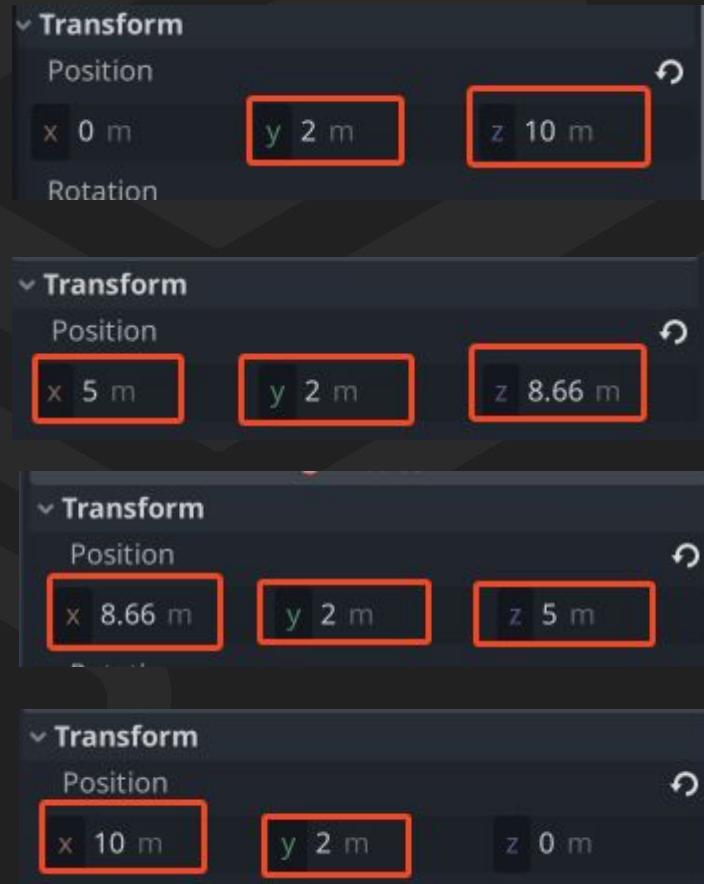
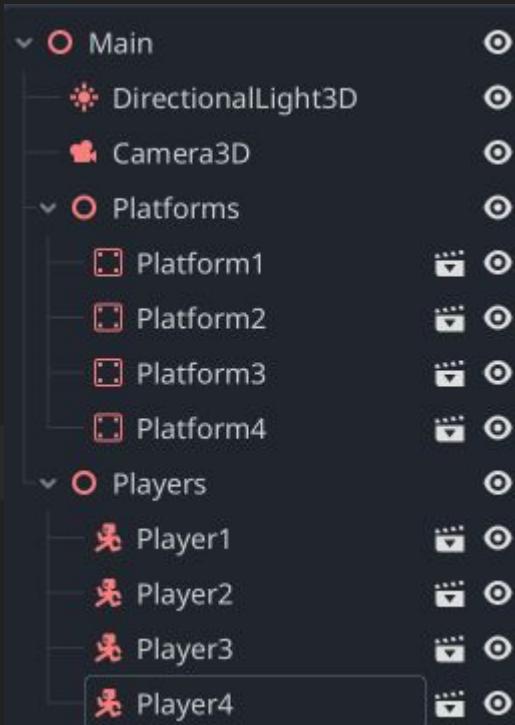


The Player

Create the Player Scene



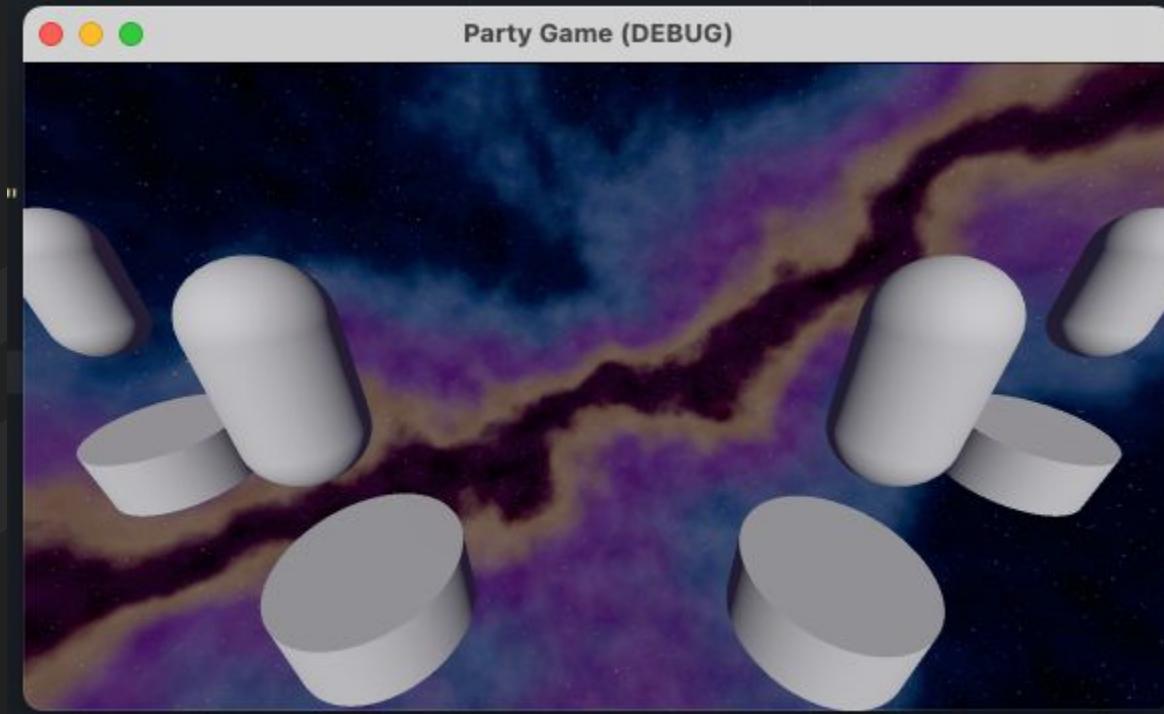
Place the Players



Add Jumping Code Snippet #1 - player.gd

```
1  extends CharacterBody3D
2
3
4  const JUMP_VEL = 4
5  const GRAVITY_ACC = -4
6
7
8  func _physics_process(delta: float) -> void:
9    # Add the gravity.
10   if not is_on_floor():
11     velocity.y += GRAVITY_ACC * delta
12
13   # Handle jump.
14   if Input.is_action_just_pressed("core_player_jump") and is_on_floor():
15     velocity.y = JUMP_VEL
16
17   move_and_slide()
18 |
```

Run the Game

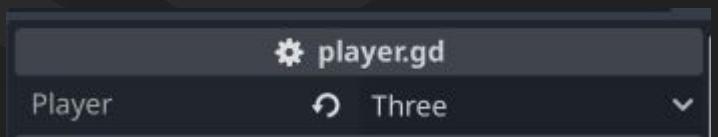
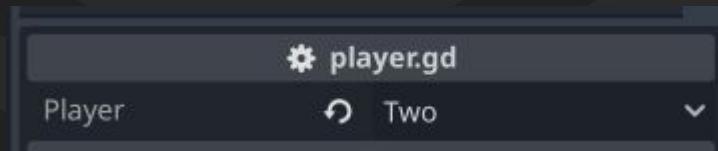
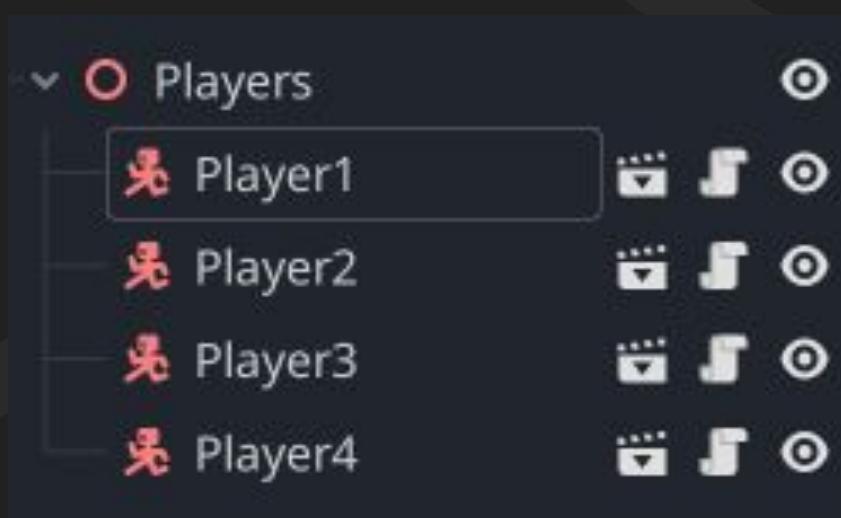


Support Multiple Players

Code Snippet #2 - player.gd

```
1  extends CharacterBody3D
2
3
4  const JUMP_VEL = 4
5  const GRAVITY_ACC = -4
6
7  @export var player: Controls.Player
8
9
10 func _physics_process(delta: float) -> void:
11     # Add the gravity.
12     if not is_on_floor():
13         velocity.y += GRAVITY_ACC * delta
14
15     # Handle jump.
16     if Controls.is_action_just_pressed(player, "core_player_jump") and is_on_floor():
17         velocity.y = JUMP_VEL
18
19     move_and_slide()
20 |
```

Set Each Player

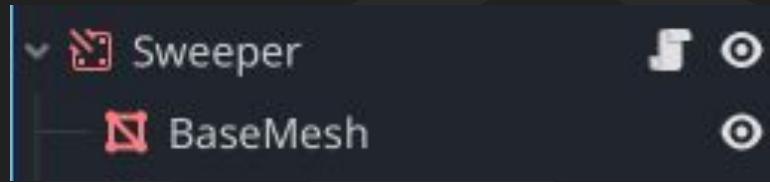


Run the Game



The Sweeper

Create the Sweeper Scene



The Sweeper Arm

- ✓ Sweeper
- ✗ BaseMesh
- ✗ ArmMesh
- ✗ ArmCollisionShape

The screenshot shows the Godot Editor's properties tab for a node named "Sweeper". The node tree on the left lists "Sweeper", "BaseMesh", "ArmMesh", and "ArmCollisionShape". The "Sweeper" node is expanded, revealing its children: "MeshInstance3D" and "Node3D".

MeshInstance3D Properties:

- Mesh:** A preview window showing a vertical cylinder.
- Top Radius:** 0.5 m
- Bottom Radius:** 0.5 m
- Height:** 12 m

Node3D Properties:

- Transform**
 - Position:** x: 0 m, y: 1.5 m, z: 6 m
 - Rotation:** x: 90°, y: 0°, z: 0°

CollisionShape3D Properties:

- Shape:** CylinderShape3D
- Height:** 12 m
- Radius:** 0.5 m
- Custom Solver Bias:** 0
- Margin:** 0.04 m

Transform Properties:

- Position:** x: 0 m, y: 1.5 m, z: 6 m
- Rotation:** x: 90°, y: 0°, z: 0°
- Scale:** (not explicitly shown in the screenshot)

The Sweeper Arm

- ✓ Sweeper
- ✗ BaseMesh
- ✗ ArmMesh
- ✗ ArmCollisionShape

The screenshot shows the Godot Editor's properties tab for a node named "Sweeper". The node tree on the left lists "Sweeper", "BaseMesh", "ArmMesh", and "ArmCollisionShape". The "Sweeper" node is expanded, revealing its children: "MeshInstance3D" and "Node3D".

MeshInstance3D Properties:

- Mesh:** A preview window showing a vertical cylinder.
- Top Radius:** 0.5 m
- Bottom Radius:** 0.5 m
- Height:** 12 m

Node3D Properties:

- Transform**
 - Position:** x: 0 m, y: 1.5 m, z: 6 m
 - Rotation:** x: 90°, y: 0°, z: 0°

CollisionShape3D Properties:

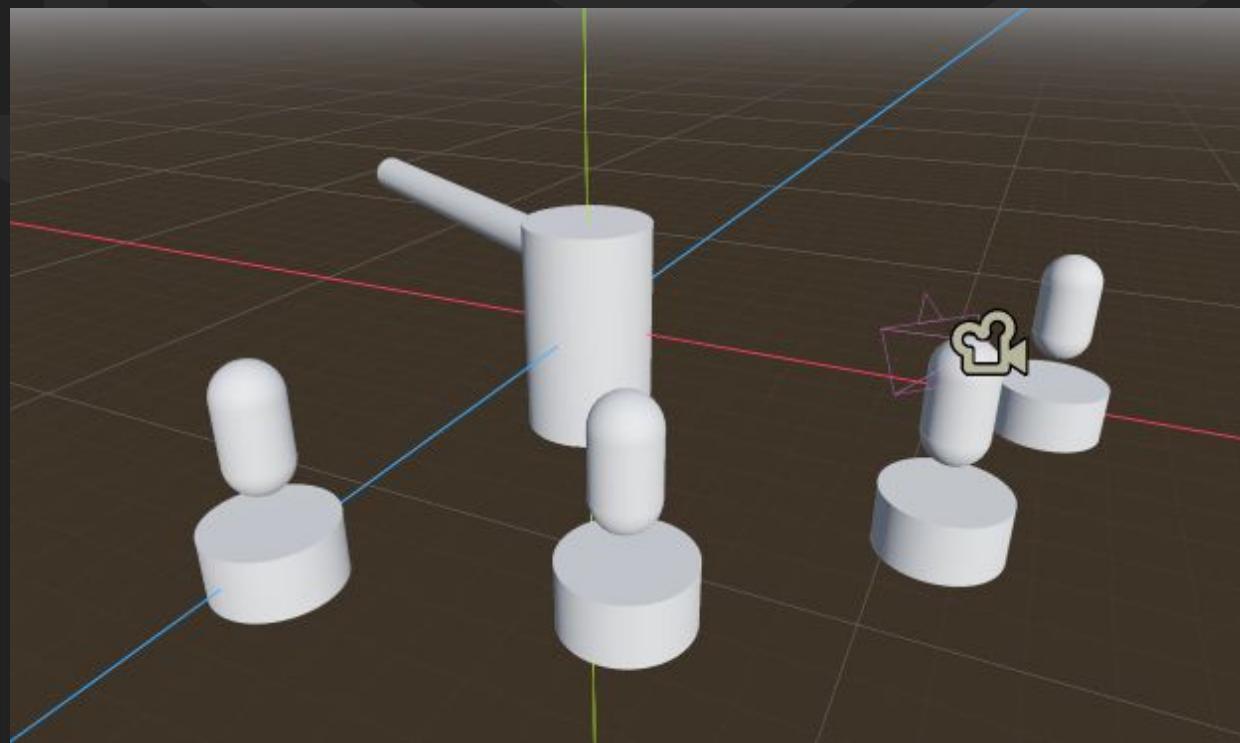
- Shape:** CylinderShape3D
- Height:** 12 m
- Radius:** 0.5 m
- Custom Solver Bias:** 0
- Margin:** 0.04 m

Transform Properties:

- Position:** x: 0 m, y: 1.5 m, z: 6 m
- Rotation:** x: 90°, y: 0°, z: 0°
- Scale:** (not explicitly shown in the screenshot)

Add the Sweeper the Wipeout Scene

▼ ○ Main	○
● DirectionalLight3D	○
● Camera3D	○
▼ ○ Platforms	○
□ Platform1	☒ ○
□ Platform2	☒ ○
□ Platform3	☒ ○
□ Platform4	☒ ○
▼ ○ Players	○
● Player1	☒ ⚡ ○
● Player2	☒ ⚡ ○
● Player3	☒ ⚡ ○
● Player4	☒ ⚡ ○
☒ Sweeper	☒ ⚡ ○

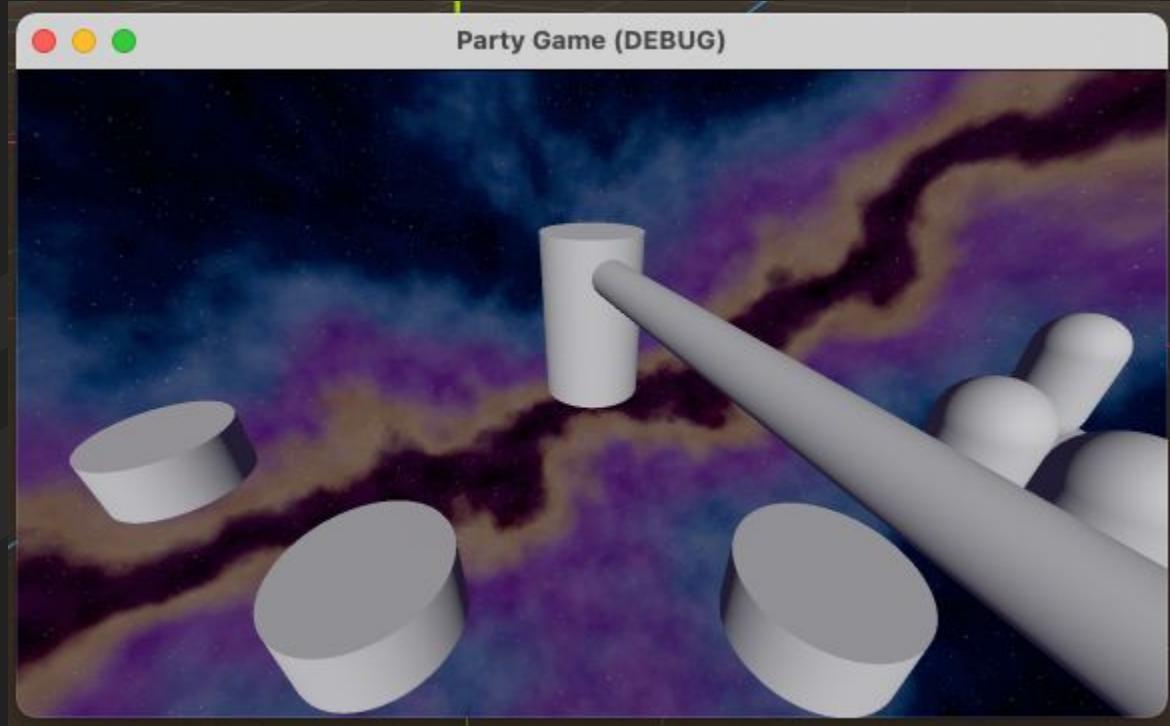


Sweeper Rotation

Code Snippet #3 - sweeper.gd

```
1  class_name WipeoutSweeper
2  extends AnimatableBody3D
3
4  var rotational_vel_deg = 45
5
6
7  ↵ func _physics_process(delta: float) -> void:
8    >   global_rotation_degrees.y += rotational_vel_deg * delta
9
```

Run the Game



Ending the Game

Detect Collisions

Code Snippet #4 - player.gd

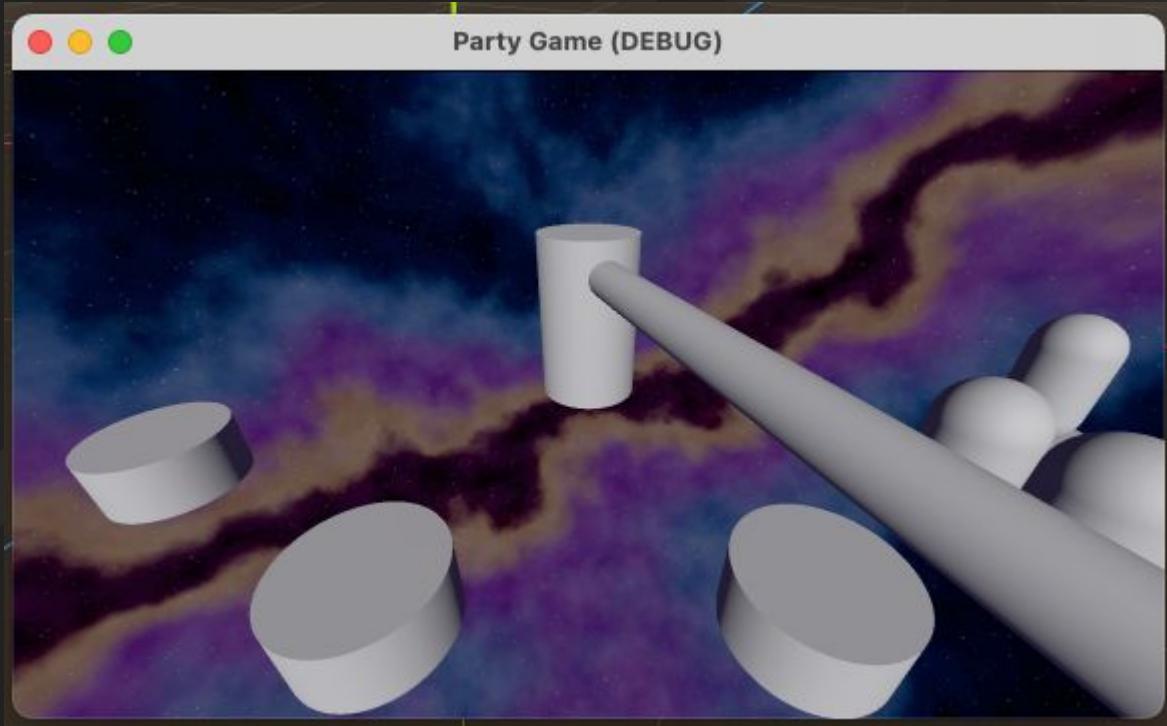
```
1  class_name WipeoutPlayer
2  extends CharacterBody3D
3
4  const JUMP_VEL = 4
5  const GRAVITY_ACC = -4
6
7  signal hit_by_sweeper(player: WipeoutPlayer)
8
9  @export var player: Controls.Player
10
11
12 func _physics_process(delta: float) -> void:
13     # Add the gravity.
14     if not is_on_floor():
15         velocity.y += GRAVITY_ACC * delta
16
17     # Handle jump.
18     if Controls.is_action_just_pressed(player, "core_player_jump") and is_on_floor():
19         velocity.y = JUMP_VEL
20
21     move_and_slide()
22
23     for collision_idx in range(get_slide_collision_count()):
24         var collision = get_slide_collision(collision_idx)
25         if collision.get_collider() is WipeoutSweeper:
26             hit_by_sweeper.emit(self)
27             $CollisionShape3D.disabled = true
28
```

Track Hits

Code Snippet #5 - wipeout.gd

```
1  extends Node3D
2
3  var players: Array[WipeoutPlayer] = []
4  var player_hit_order: Array[Controls.Player] = []
5
6  func _ready() -> void:
7    for player in $Players.get_children():
8      if player is WipeoutPlayer:
9        players.push_back(player)
10     player.hit_by_sweeper.connect(_on_player_hit_by_sweeper)
11
12
13 func _process(_delta: float) -> void:
14  if players.is_empty():
15    SceneManager.exit_minigame(
16      player_hit_order.find(Controls.Player.ONE),
17      player_hit_order.find(Controls.Player.TWO),
18      player_hit_order.find(Controls.Player.THREE),
19      player_hit_order.find(Controls.Player.FOUR)
20    )
21
22
23 func _on_player_hit_by_sweeper(player: WipeoutPlayer) -> void:
24  players.erase(player)
25  player_hit_order.push_front(player)
26 |
```

Run the Game



1st: Player 4
2nd: Player 3
3rd: Player 2
4th: Player 1

If we have time...

- Better collision detection
- Sweeper rotates faster over time
- More sweepers
- Sweeper moves up and down
- UI?

Sign in here!



DevLUp FSU

GBM #2



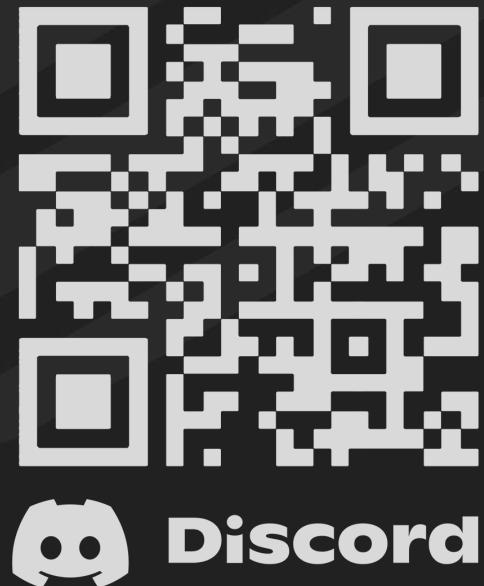
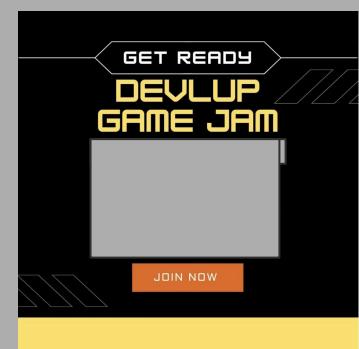
Intro to Godot

January 30, 2025

Join our Discord and visit our website!

Join our Discord server using the QR code to stay in touch with the club!

Meet other DevLUp FSU members, “showoff” your work, share resources, take part in game jams, and stay up to date on all our future events!



 fsu.devlup.org