Microgame Jam Tutorial (temp version)

Thank you for participating in the Microgame Jam! Something something

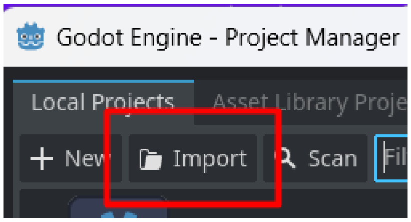
Things I think should be written:

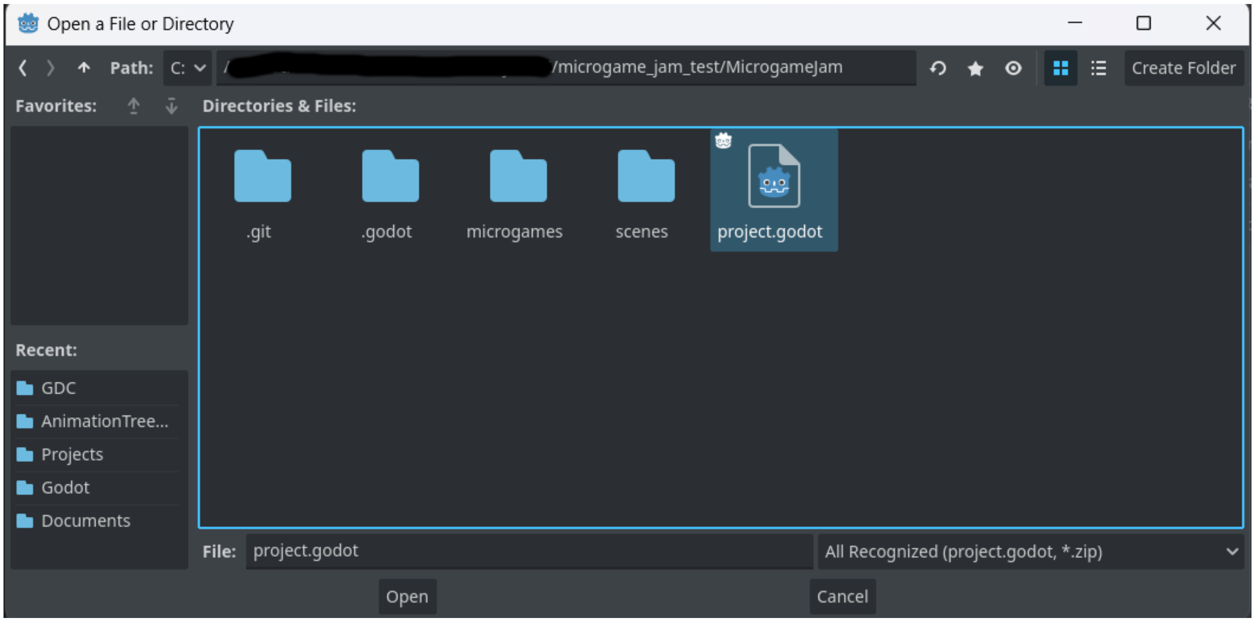
- This is a beginner jam intended for ppl who haven’t made a full game yet
- Godot only, use the template
- 4-8 second microgame
- You will have from date 1 to date 2 to make the gane, submit by date 2
- Come to the tutorial on tutorial date for step-by-step guide on properly making the game!
- Prizes will be offered

This doc looks kinda ugly and hopefully we can update it later lol

Step 1

Import and edit the project by finding the project.godot file.

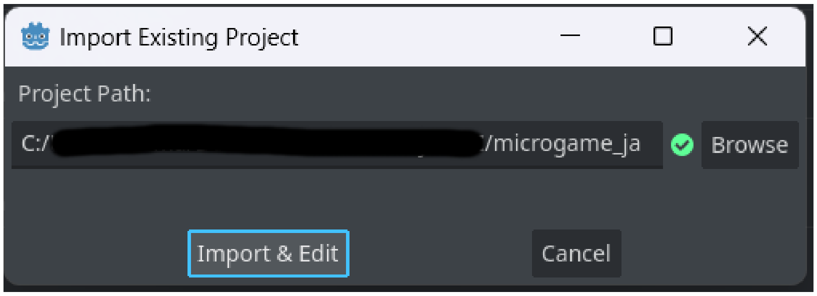




Path: C: / [redacted] /microgame_jam_test/MicrogameJam

File: project.godot

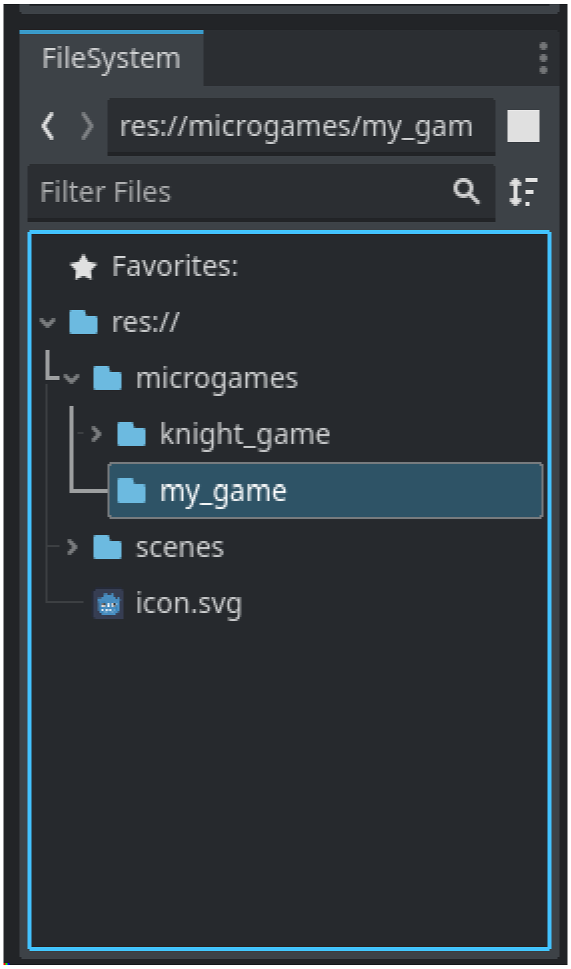
All Recognized (project.godot, *.zip)

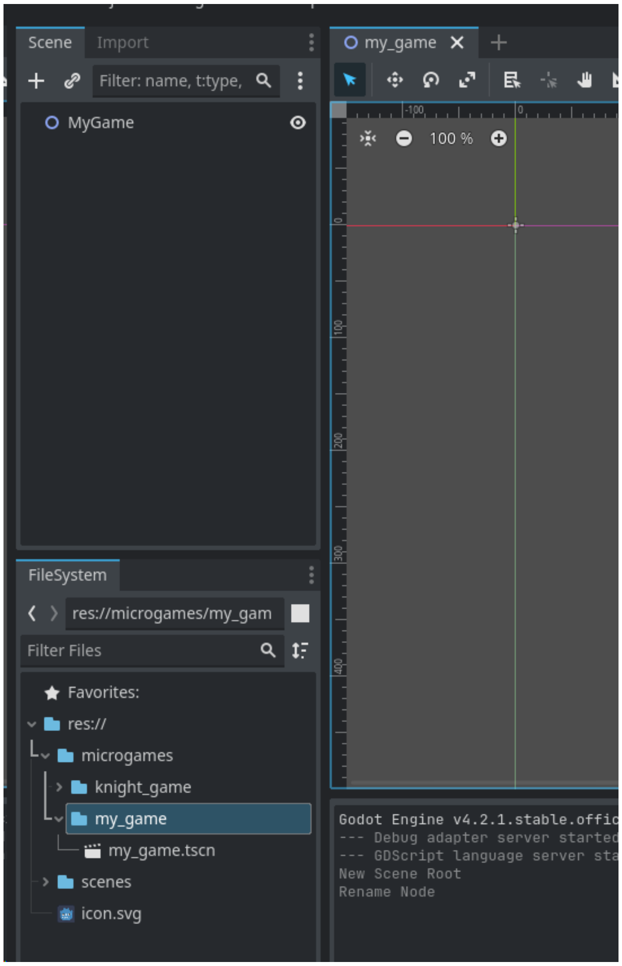


Project Path: C:/[redacted]/microgame_ja

Step 2

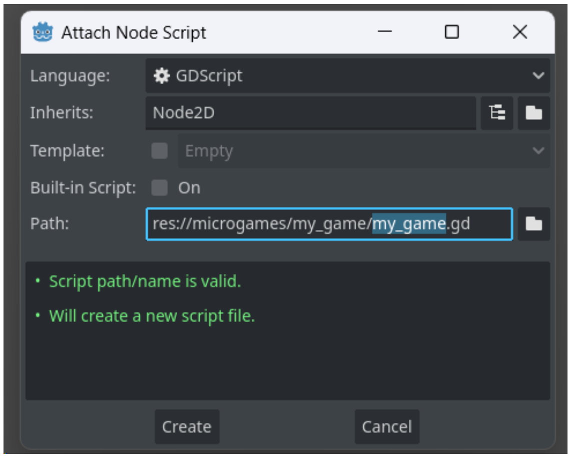
Make a new folder for your microgame and make a new Node2D that will be your microgame.





```
Godot Engine v4.2.1.stable.offic
--- Debug adapter server started
--- GDScript language server sta
New Scene Root
Rename Node
```

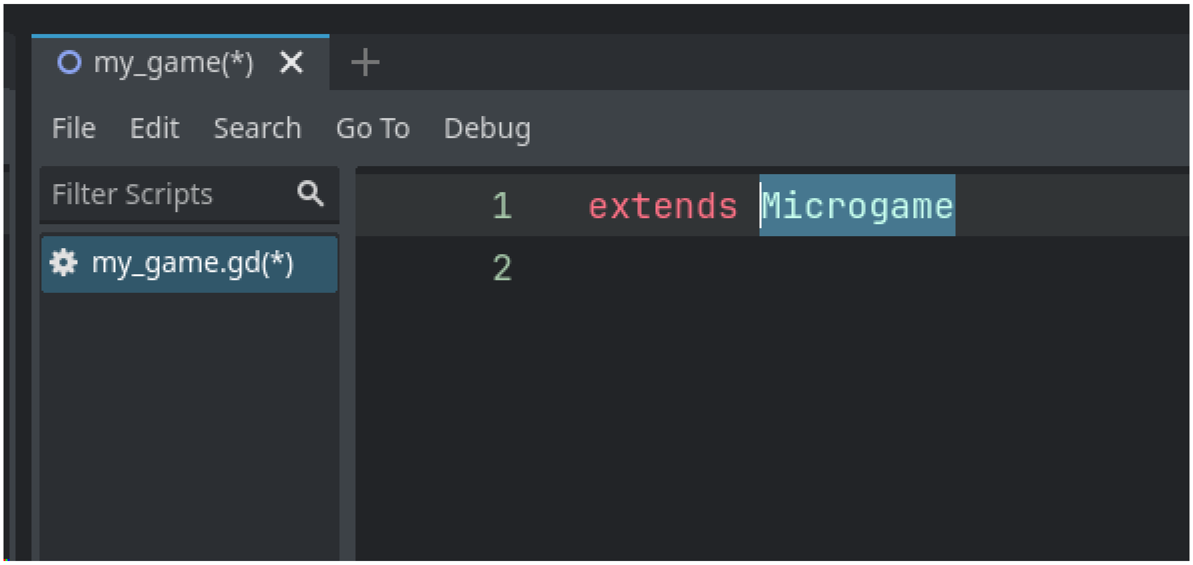
Make a new script for your microgame – the main script



Language: GDScript
Inherits: Node2D
Template: □ Empty
Built-in Script: □ On
Path: res://microgames/my_game/my_game.gd

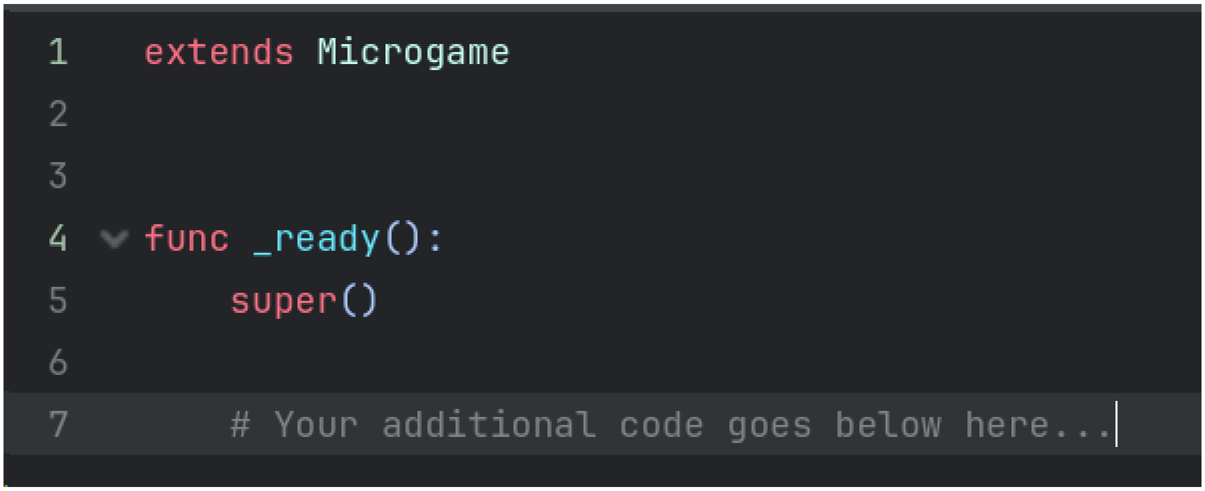
- Script path/name is valid.
- Will create a new script file.

MOST IMPORTANT PART: make sure your microgame extends the Microgame class. This makes it so that it has all the settings for a microgame and can interface with the main game.



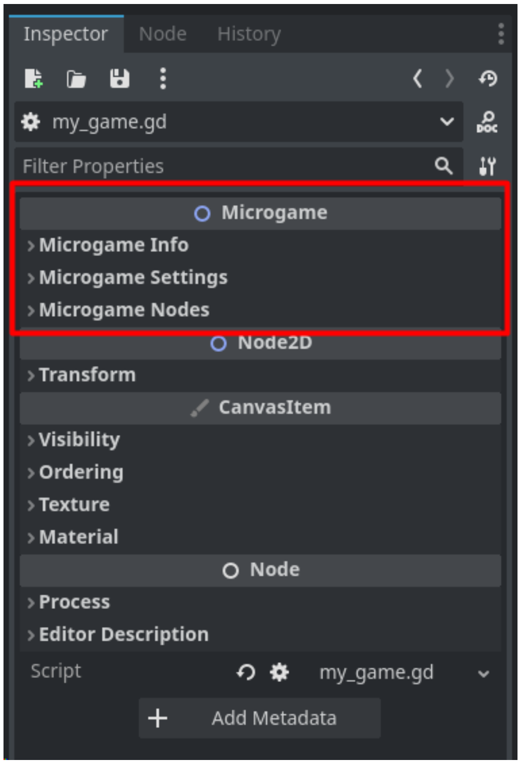
```
extends Microgame

```

Second most important part: if your main microgame script uses the _ready() function make sure to call super() so that Microgame’s _ready() is called

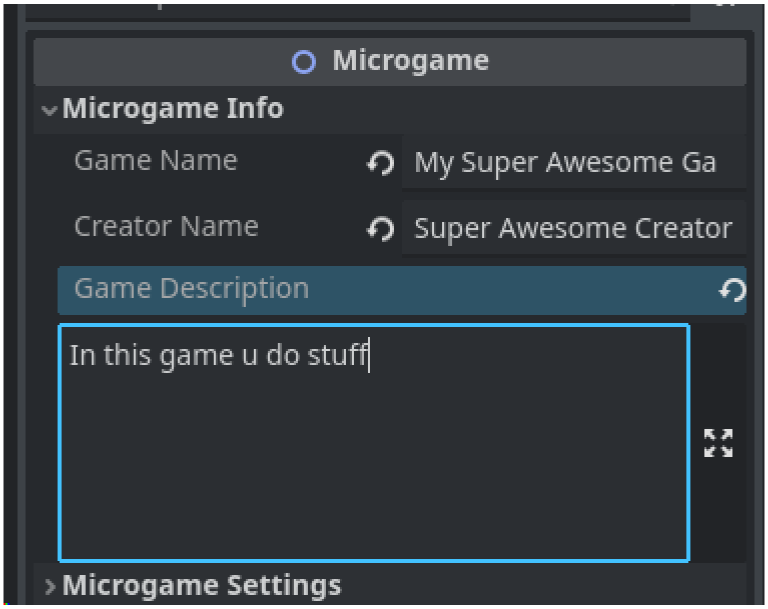


```
extends Microgame


func _ready():
	super()

	# Your additional code goes below here...
```

Extending Microgame exposes a lot of new settings in the inspector for your game.



Microgame info is just general stuff you can fill out. Make sure to fill out the creator name so that we can know who u are in case u win



Microgame

Microgame Info

Game Name: My Super Awesome Ga

Creator Name: Super Awesome Creator

Game Description: In this game u do stuff

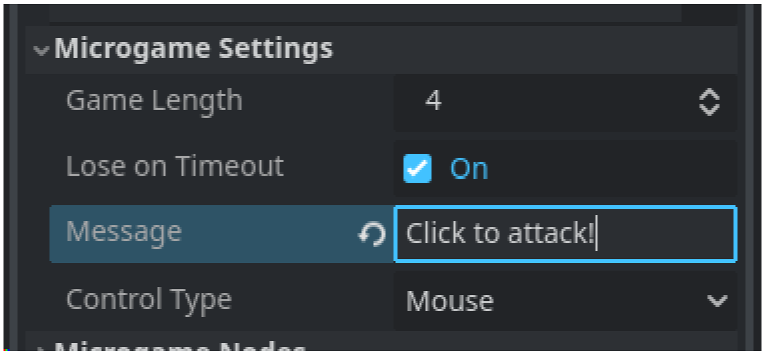
Important microgame settings

Game length: how long your game will be before timing out (in seconds)

Lose on timeout: specifies whether your game will lose/win when the time runs out

Message: A small message that will appear before your game starts, make it short and memorable and maybe with a small instruction

Control Type: specify which controls your game uses. Three options: mouse, WASD, or both. Will show up in the main game below the message



Microgame Settings

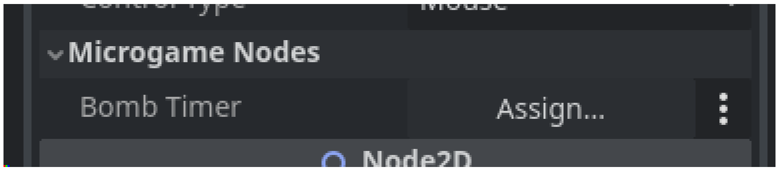
Game Length: 4

Lose on Timeout: ☑ On

Message: Click to attack!

Control Type: Mouse

Last part, one that's essential. Gonna need a bomb timer for your microgame

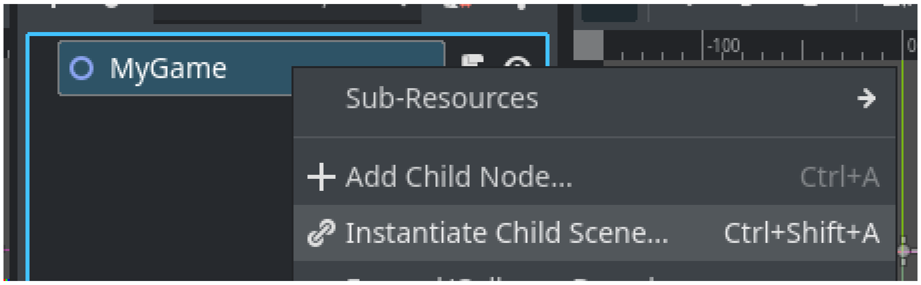


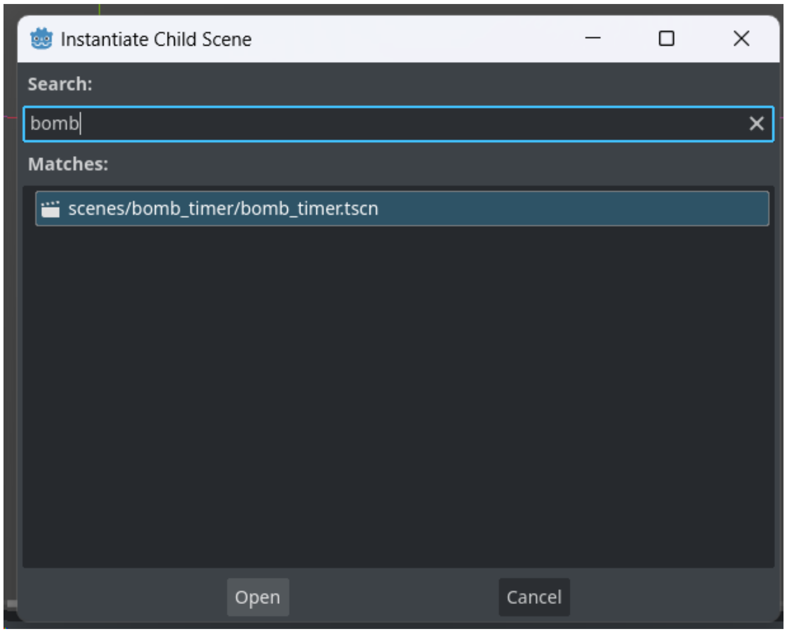
Control Type: Mouse

Microgame Nodes

Bomb Timer: Assign...

Add a new bomb timer to ur game





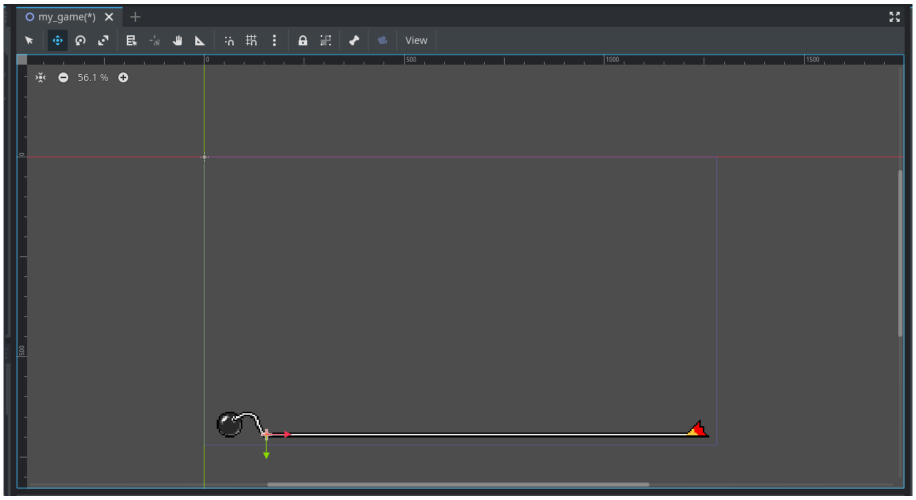
Search:

bomb

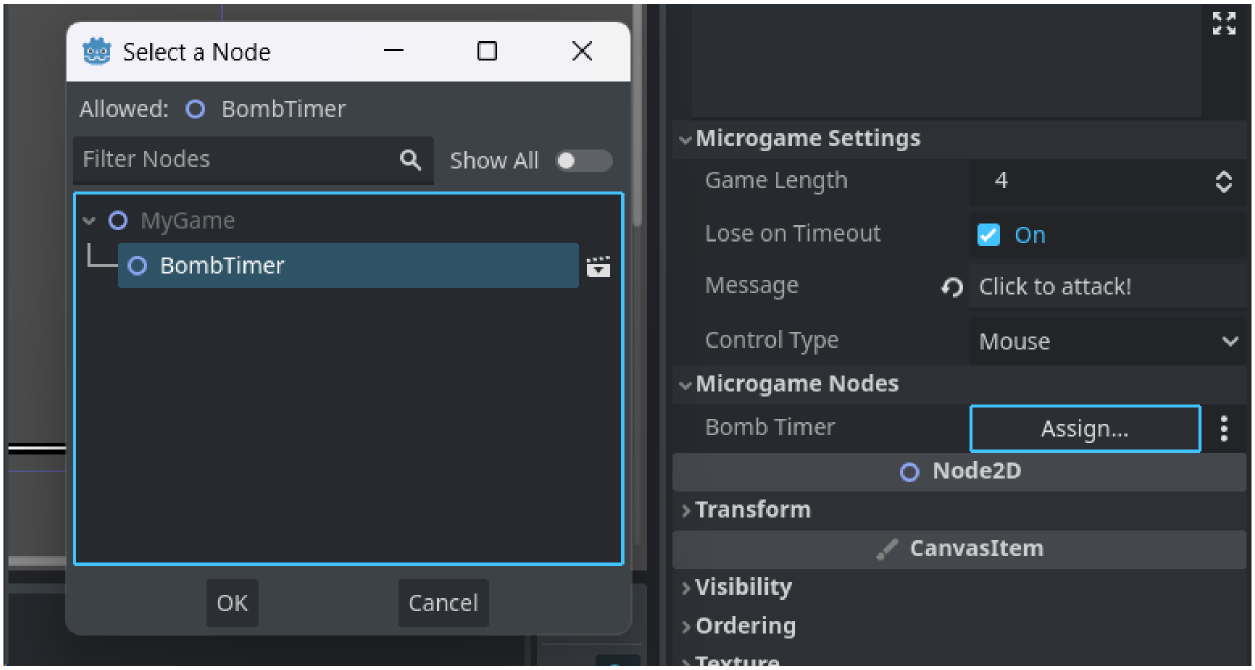
Matches:

scenes/bomb_timer/bomb_timer.tscn

Position it in a reasonable spot



And make sure to set it as the script’s bomb timer



Microgame Settings

Game Length 4

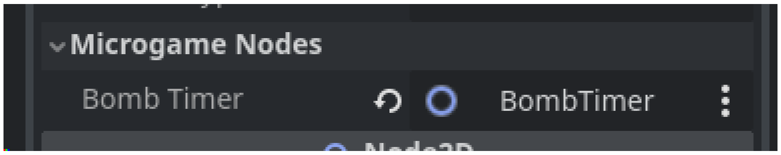
Lose on Timeout ☑ On

Message Click to attack!

Control Type Mouse

Microgame Nodes

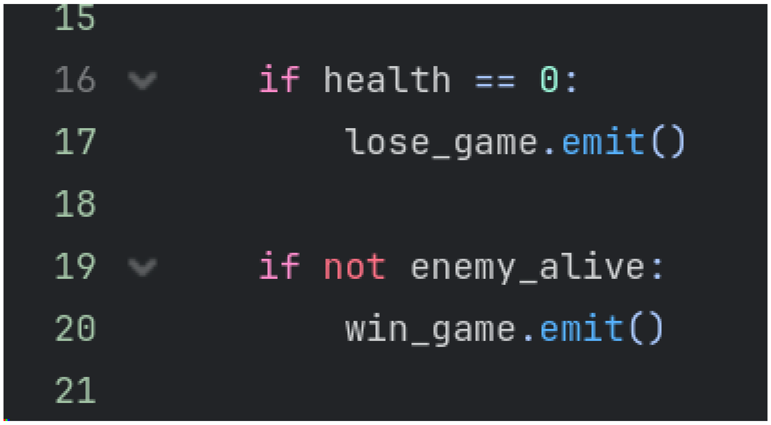
Bomb Timer Assign...



Microgame Nodes

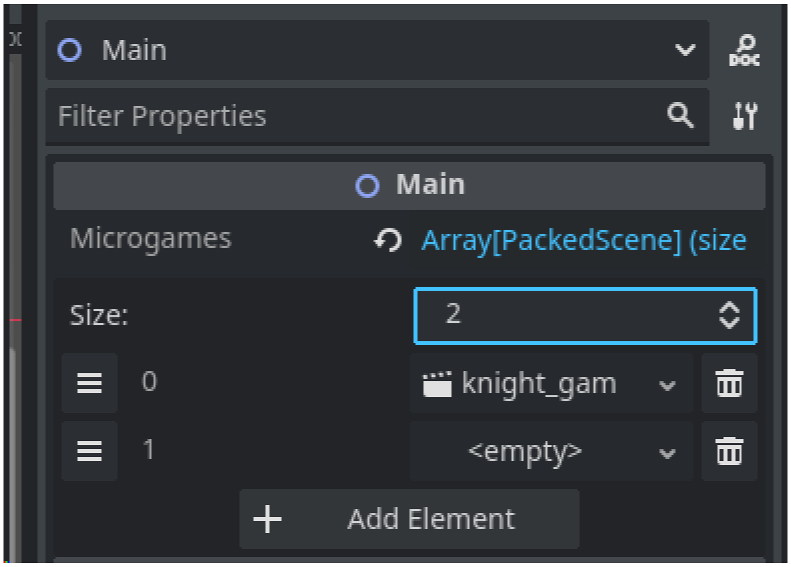
Bomb Timer ○ BombTimer

There is also another way to win/lose before the timer is up. Just emit the win_game and lose_game signals, this is an example:



```
    if health == 0:
        lose_game.emit()

    if not enemy_alive:
        win_game.emit()

```

Once you are done your microgame, you can add it to the main game by increasing the microgame list size and adding ur game to the list



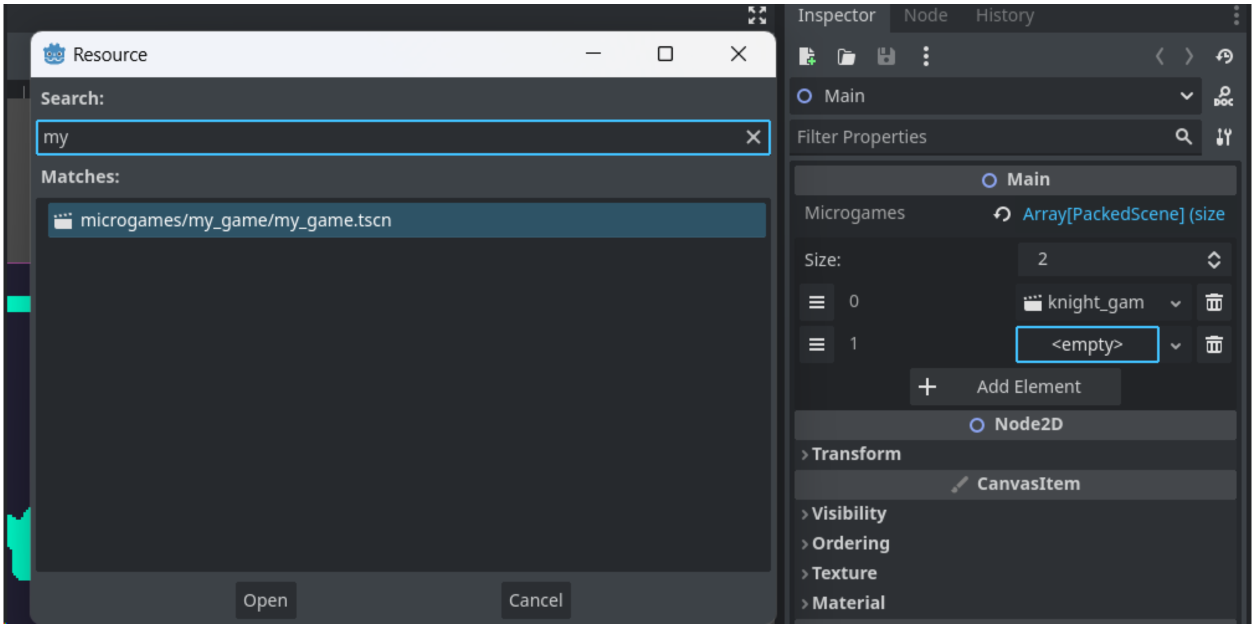
Main

Microgames Array[PackedScene] (size

Size: 2

0 knight_gam

1 <empty>



Resource

Search: my

Matches: microgames/my_game/my_game.tscn

Main

Microgames Array[PackedScene] (size

Size: 2

0 knight_gam

1 <empty>