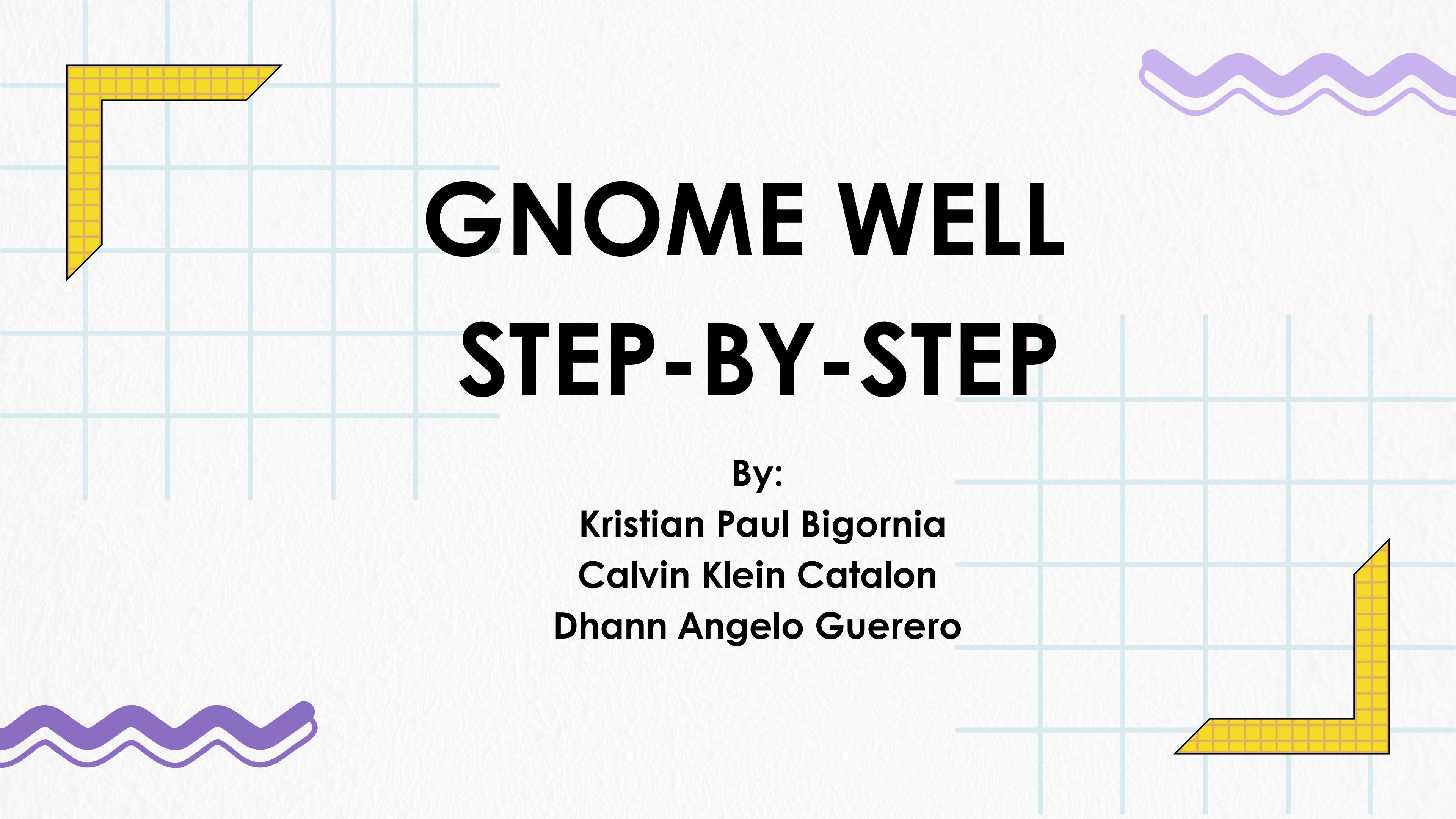


GNOME WELL

STEP-BY-STEP

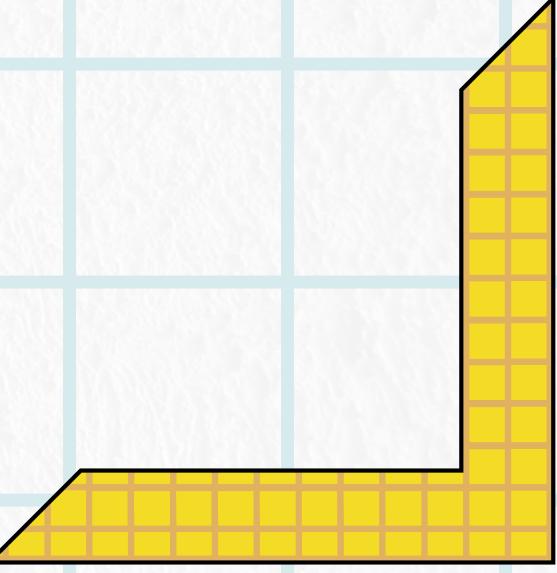


By:

Kristian Paul Bigornia

Calvin Klein Catalon

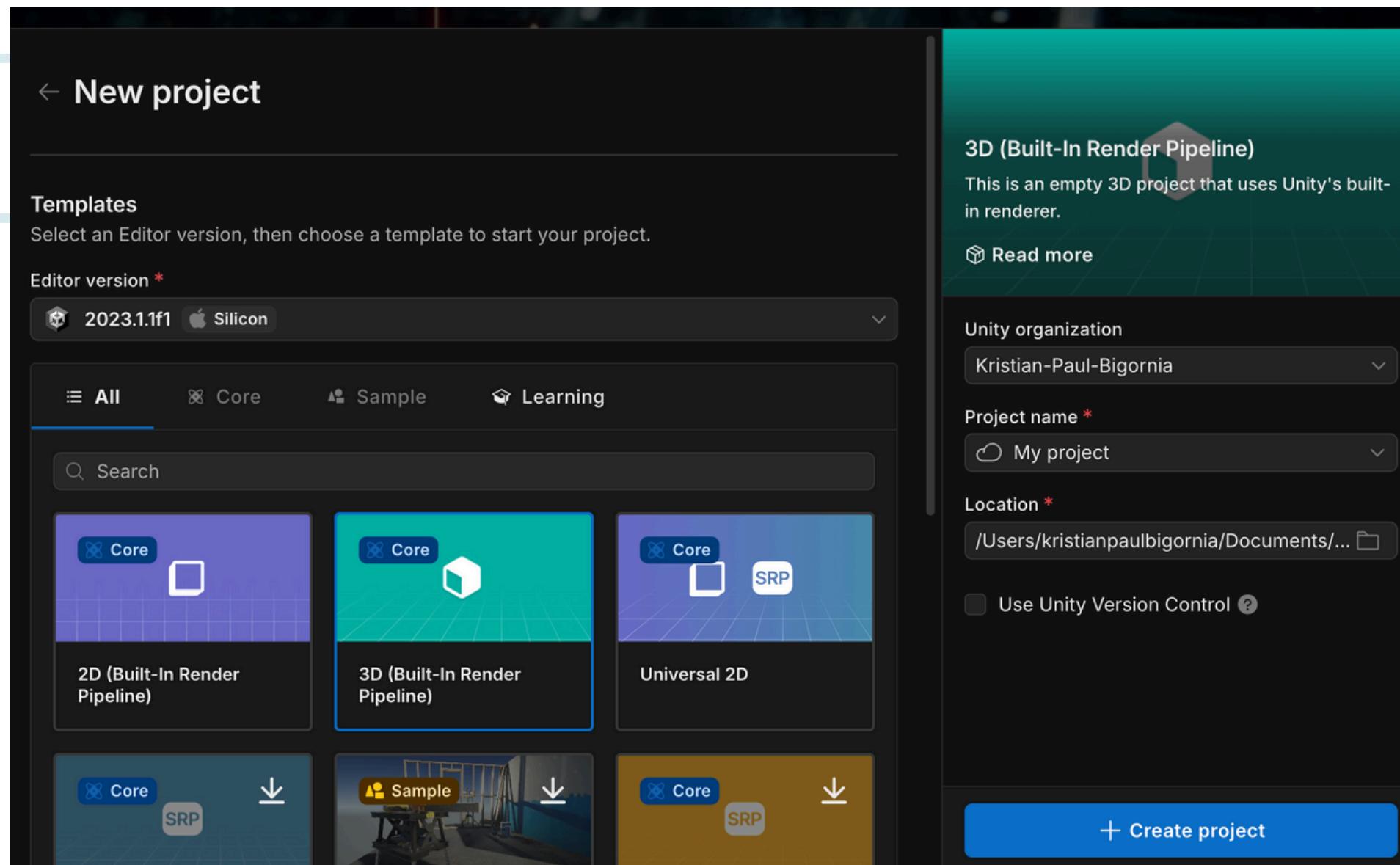
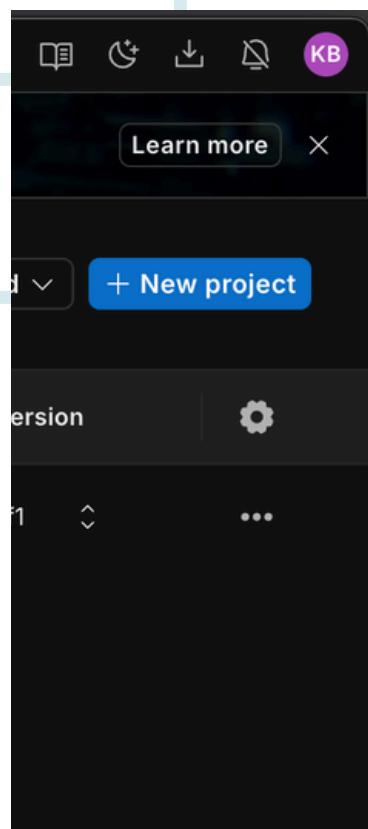
Dhann Angelo Guerrero



STEP 1

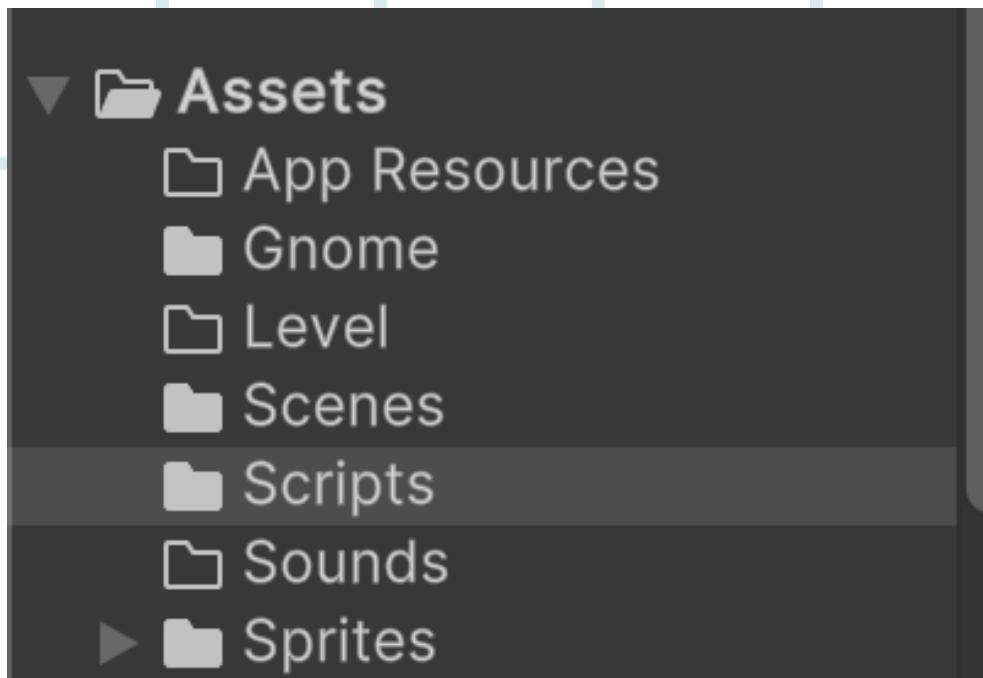
First create a new project in unity

Then Select the Editor Version and The Core Render, We used the 2D.



STEP 2

After Creating the Project, in the assets tab, we created this folders



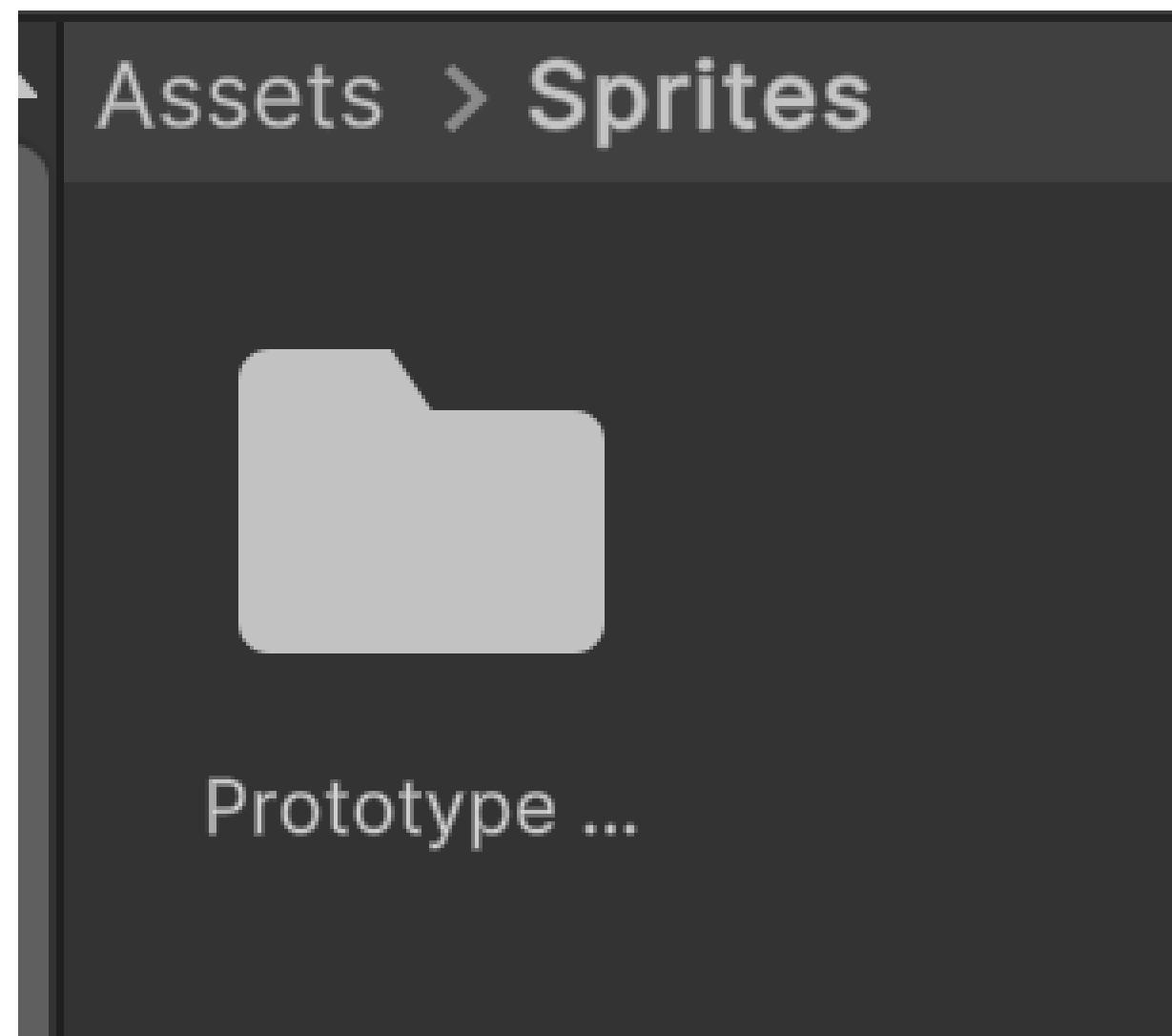
We then Downloaded the assets that came from the book

▼	Prototype Gnome	9/6/17, 6:17 AM
-	Prototype Ar...ith Gold.png	9/6/17, 6:17 AM
-	Prototype Ar...ld.png.meta	9/6/17, 6:17 AM
-	Prototype Arm Holding.png	9/6/17, 6:17 AM
-	Prototype Ar...ng.png.meta	9/6/17, 6:17 AM
-	Prototype Arm Loose.png	9/6/17, 6:17 AM
-	Prototype Ar...se.png.meta	9/6/17, 6:17 AM
-	Prototype Body.png	9/6/17, 6:17 AM
-	Prototype Body.png.meta	9/6/17, 6:17 AM
-	Prototype Head.png	9/6/17, 6:17 AM
-	Prototype Head.png.meta	9/6/17, 6:17 AM
-	Prototype Leg Dangle.png	9/6/17, 6:17 AM
-	Prototype Le...le.png.meta	9/6/17, 6:17 AM
-	Prototype Leg Rope.png	9/6/17, 6:17 AM
-	Prototype Le...pe.png.meta	9/6/17, 6:17 AM
	-- Folder	
	3 KB	PNG image
	1 KB	Document
	312 bytes	PNG image
	1 KB	Document
	312 bytes	PNG image
	1 KB	Document
	4 KB	PNG image
	1 KB	Document
	5 KB	PNG image
	1 KB	Document
	1 KB	PNG image
	1 KB	Document
	1 KB	PNG image

STEP 4

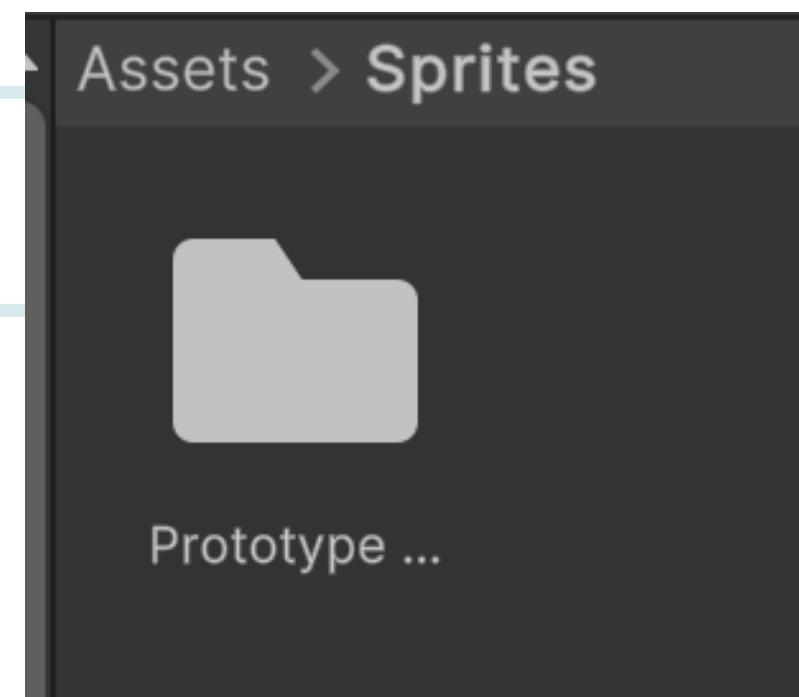
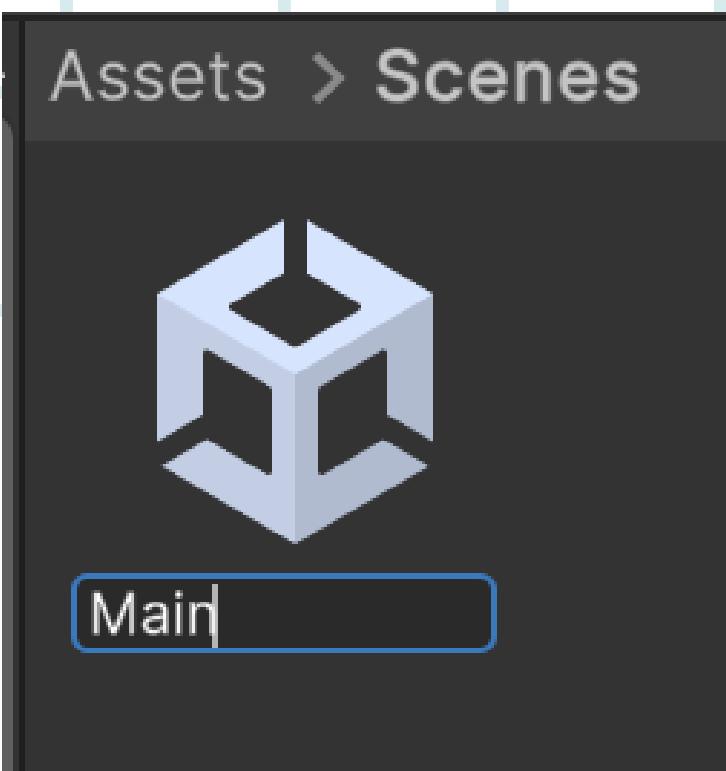
We then make sure
that the Scene has a
name which is Main

We then transferred the Prototype
Gnome Folder Asset to the Sprites folder
in the asset tab



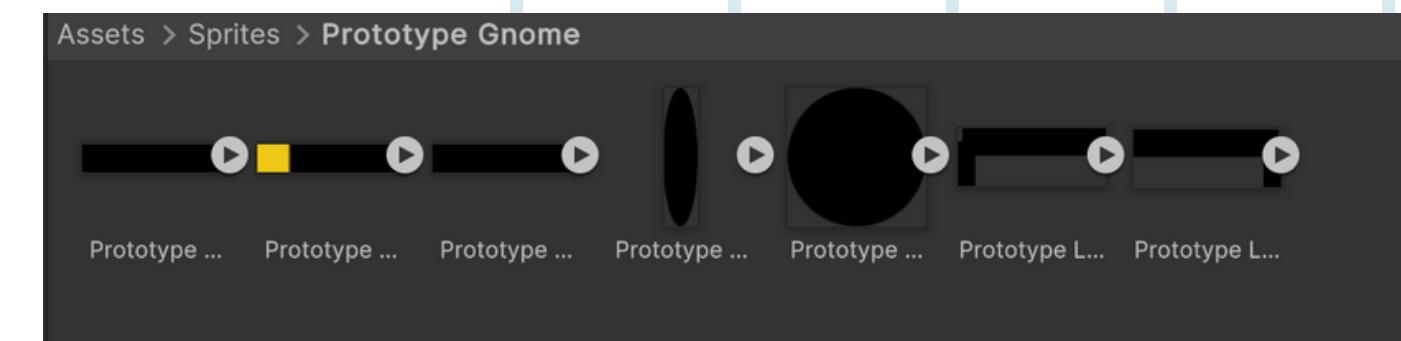
STEP 5

We then make sure
that the Scene has a
name which is Main



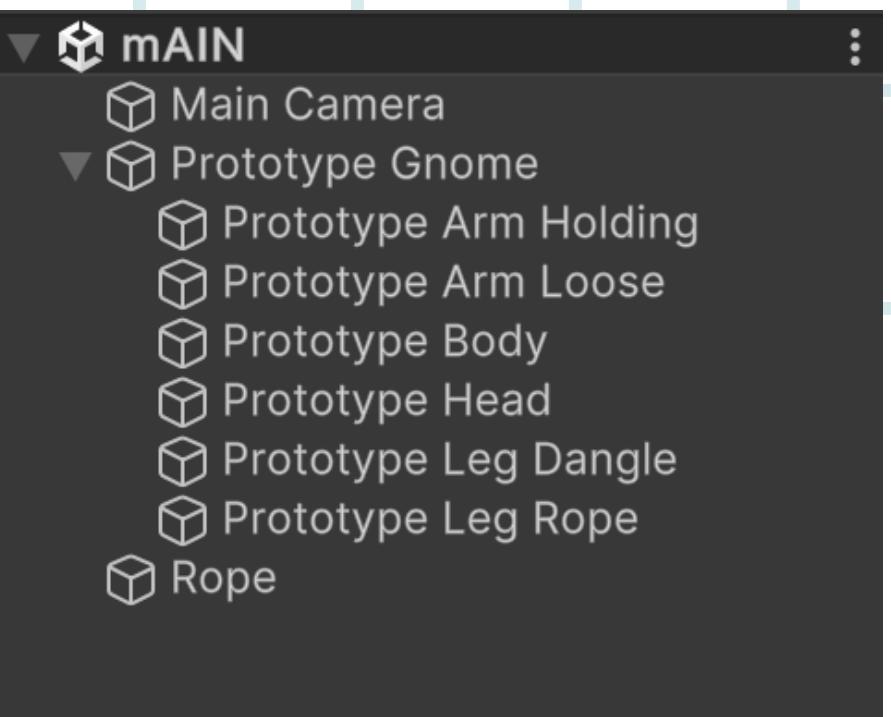
We then transferred the Prototype
Gnome Folder Asset to the Sprites folder
in the asset tab

It Will look like this

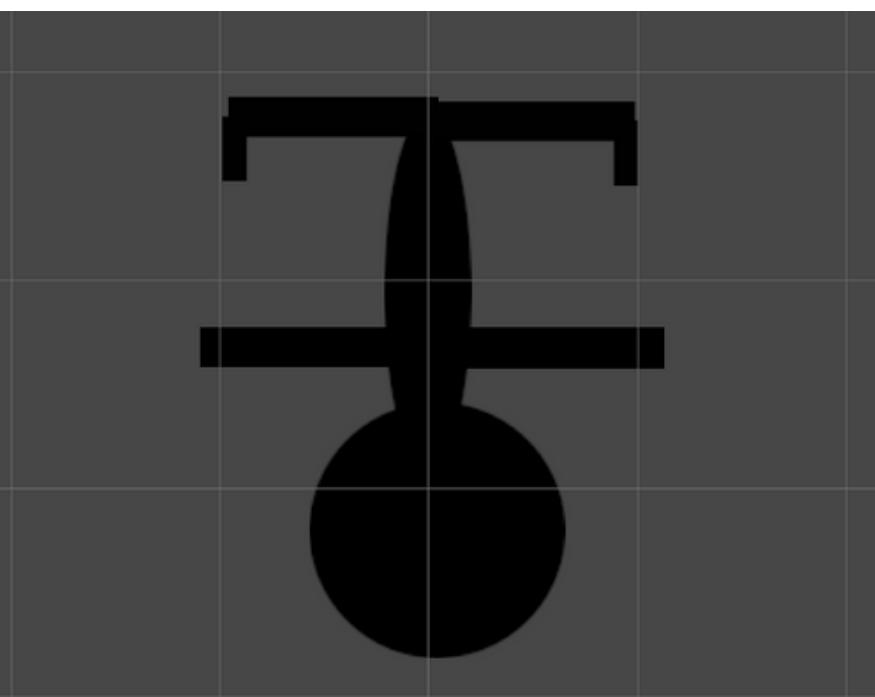


STEP 6

We then created this objects,
The prototype Gnome has the
prototypes assets inside of it



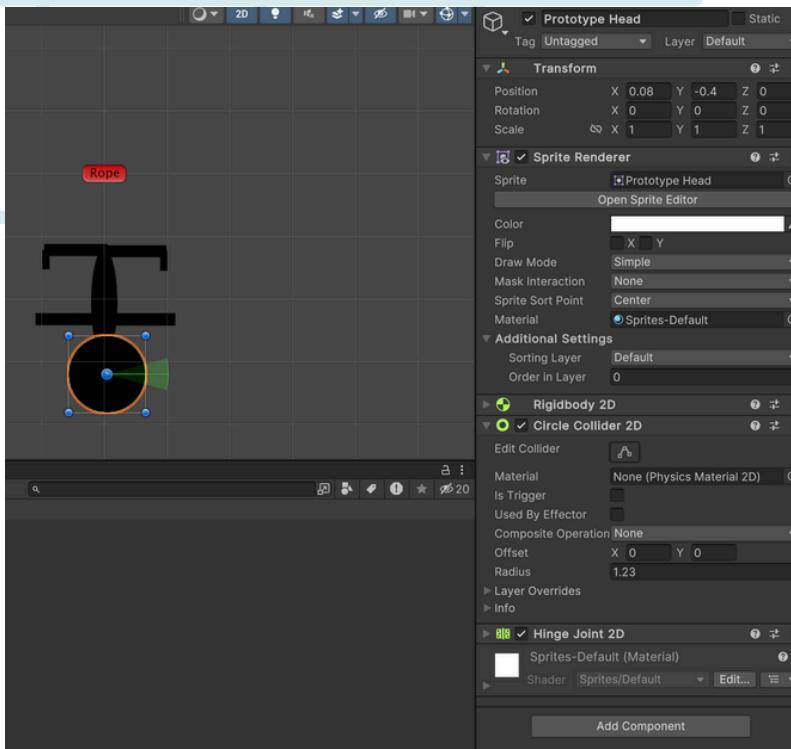
We then moved the parts to make it look
like this



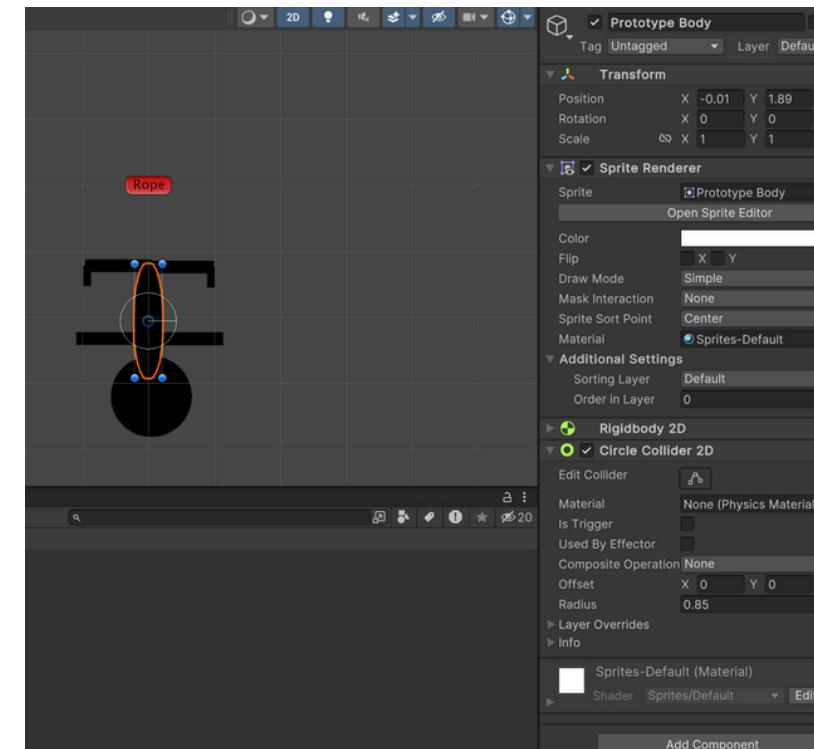
STEP 7

We clicked each part of the body and added components

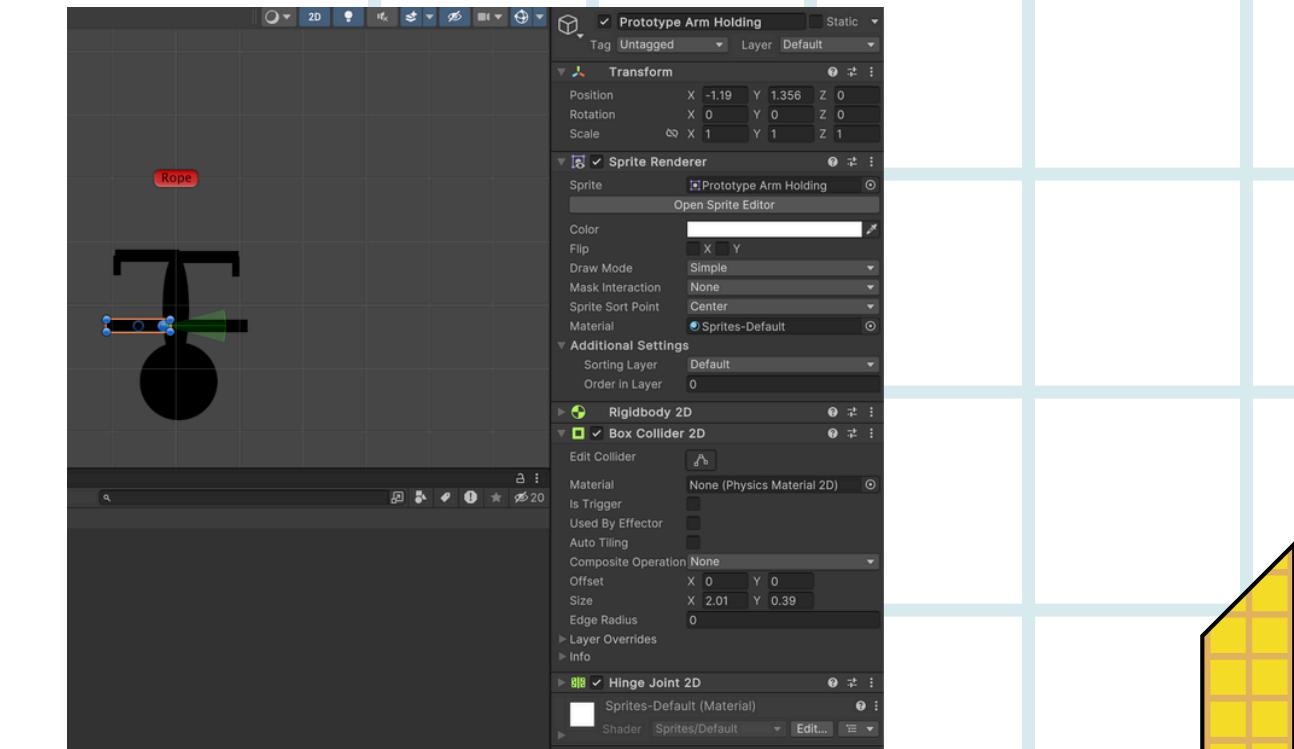
Prototype Head



Prototype Body

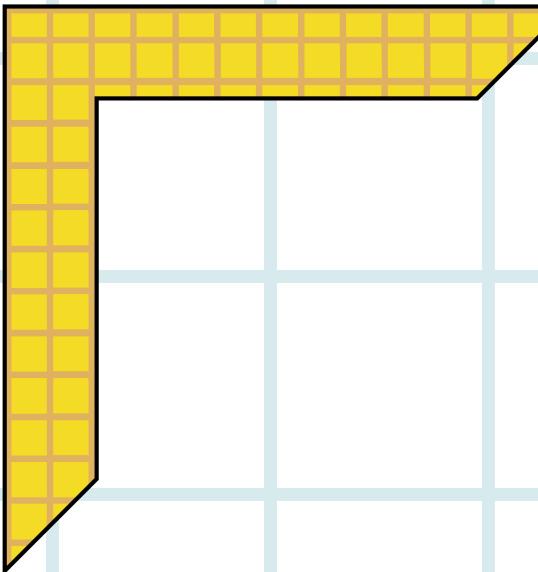


Prototype Arm Holding

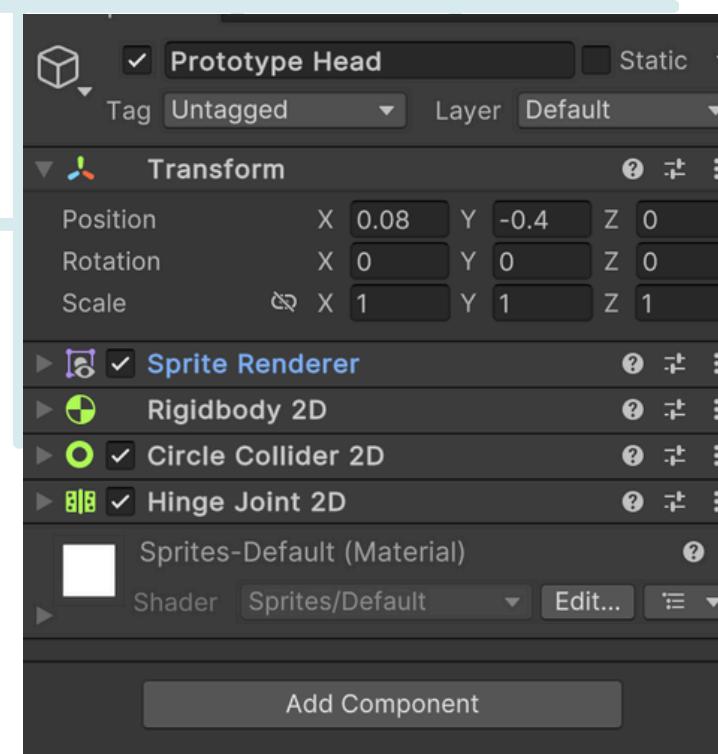


STEP 7

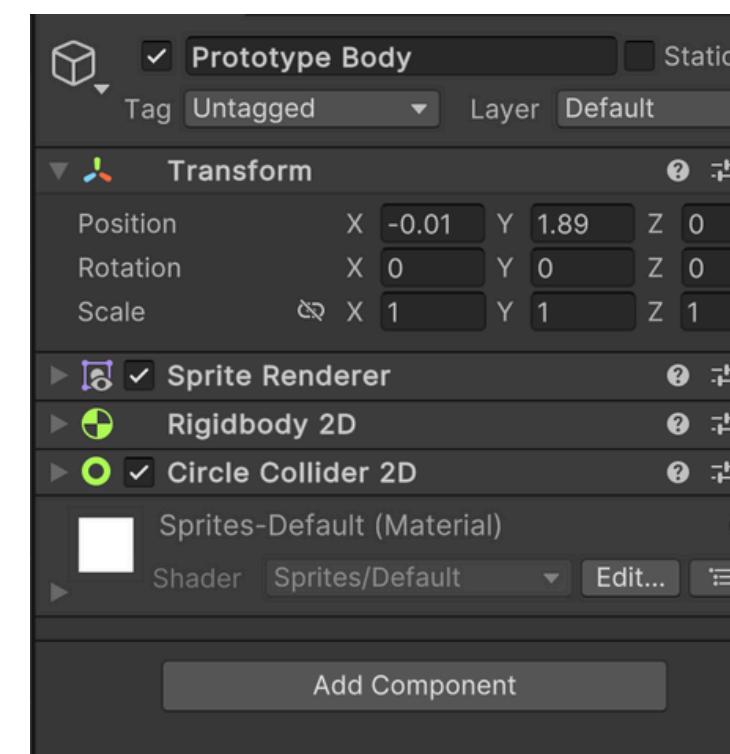
We clicked each part of the body and added components



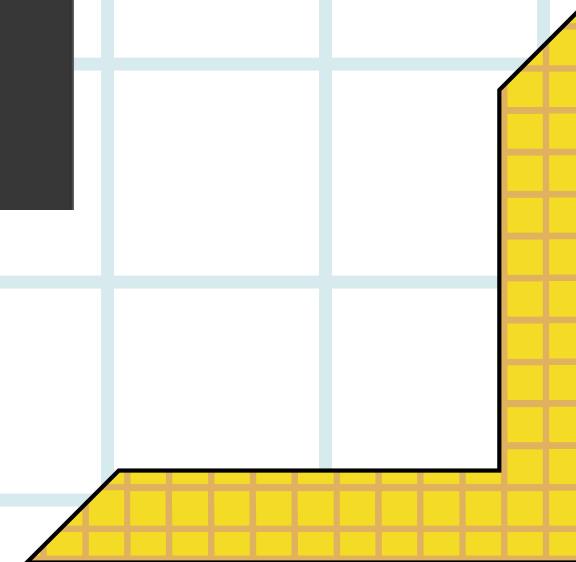
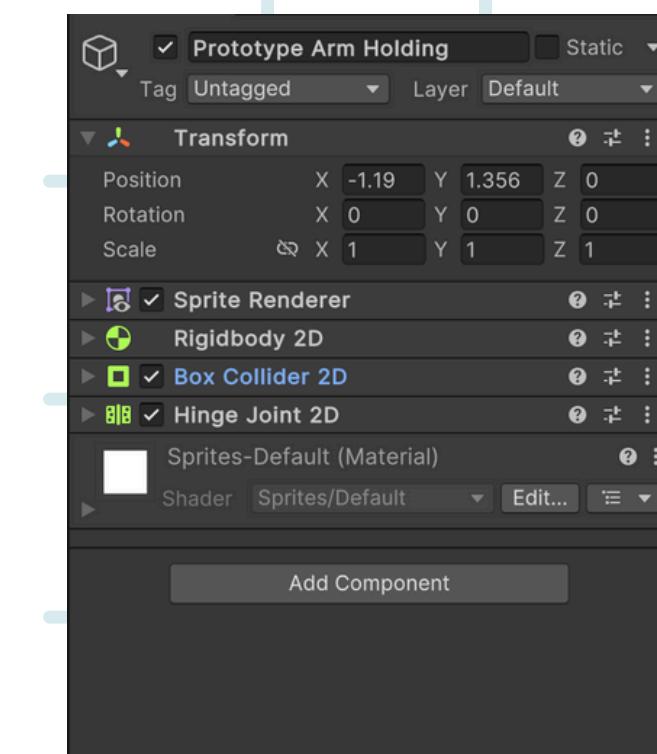
Prototype Head



Prototype Body



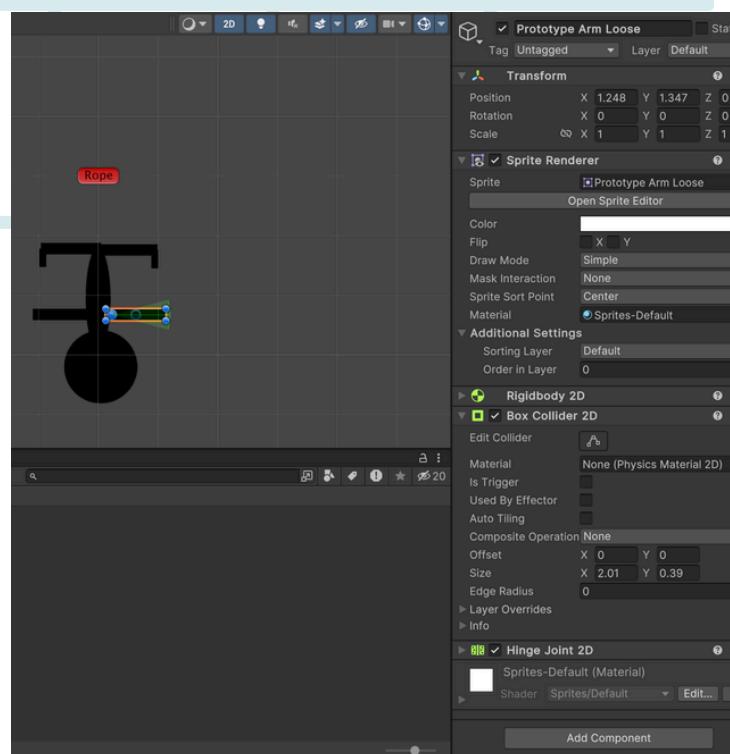
Prototype Arm Holding



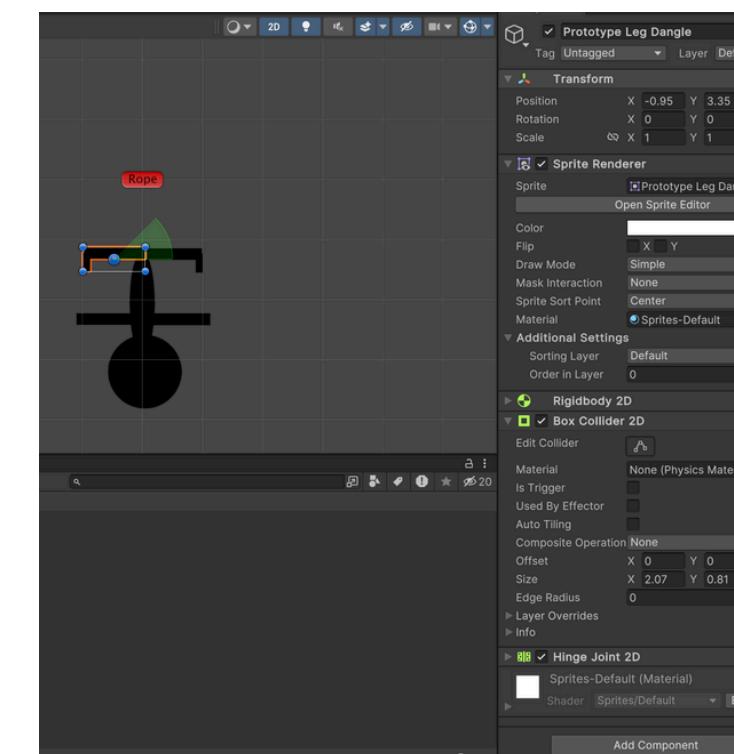
STEP 7

We clicked each part of the body and added components

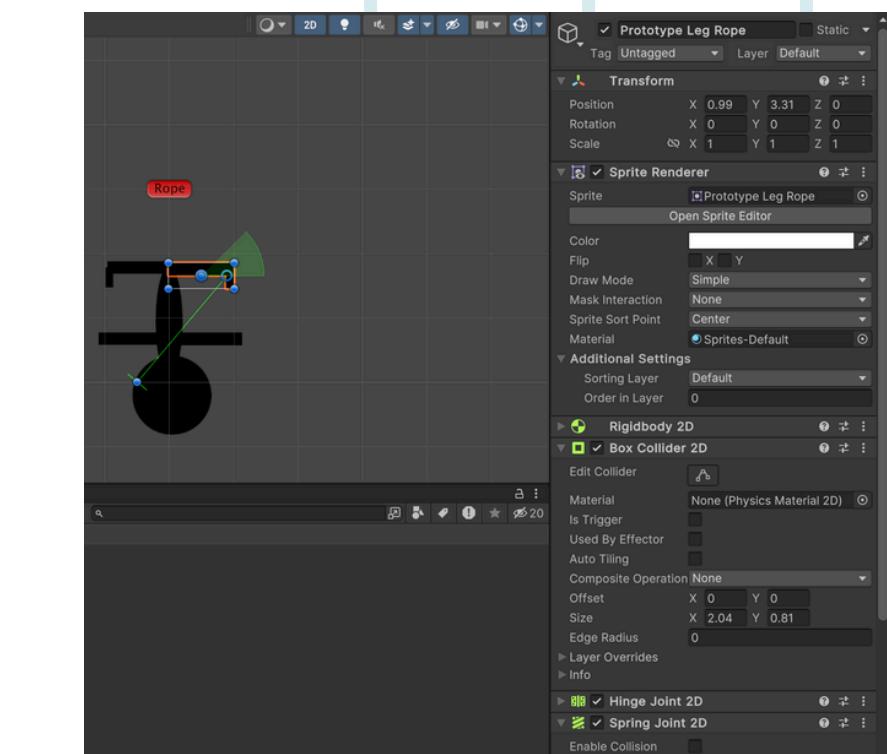
Prototype Arm Loose



Prototype Leg Dangle

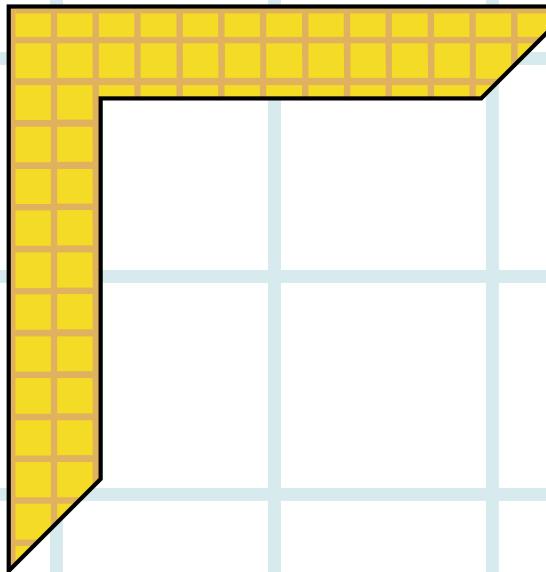


Prototype Leg Rope

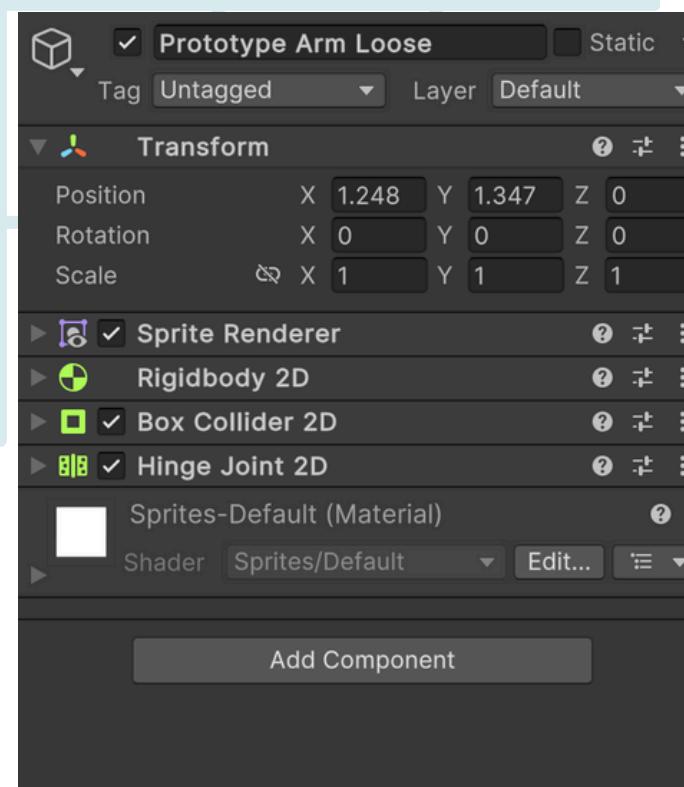


STEP 7

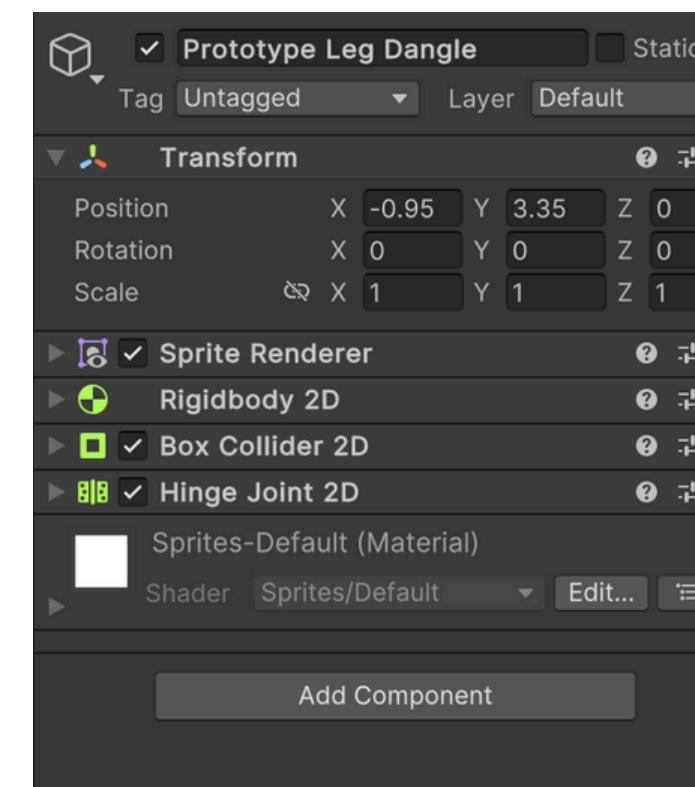
We clicked each part of the body and added components



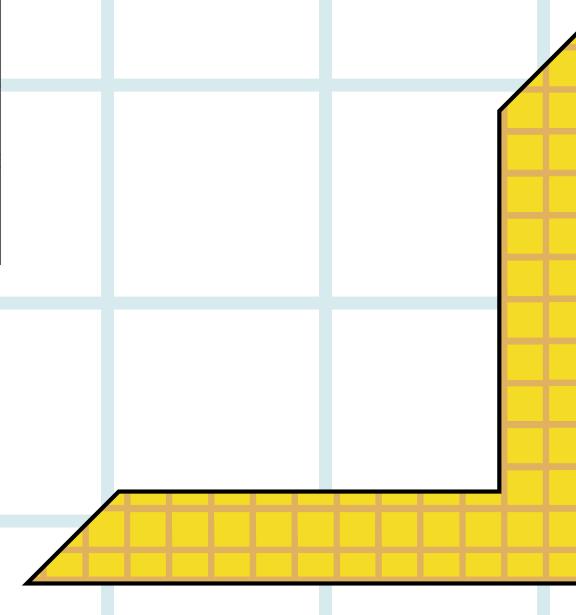
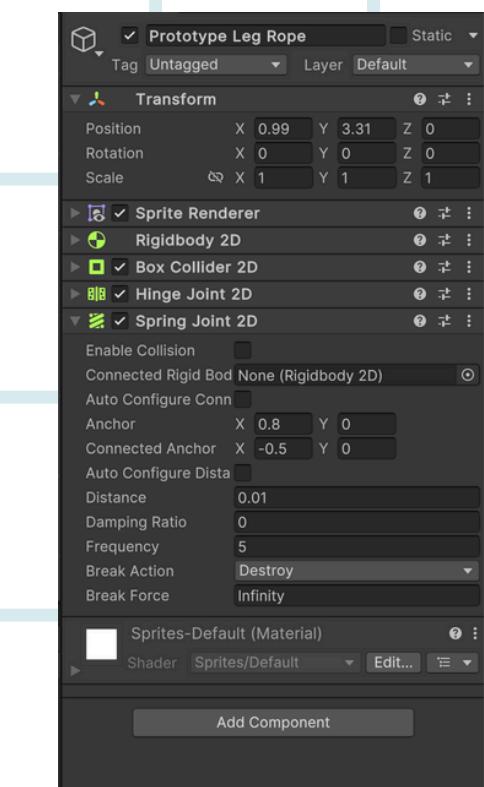
Prototype Arm Loose



Prototype Leg Dangle

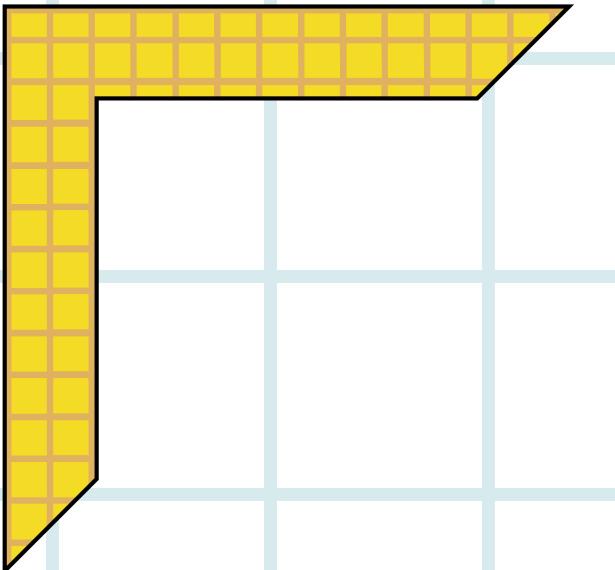


Prototype Leg Rope

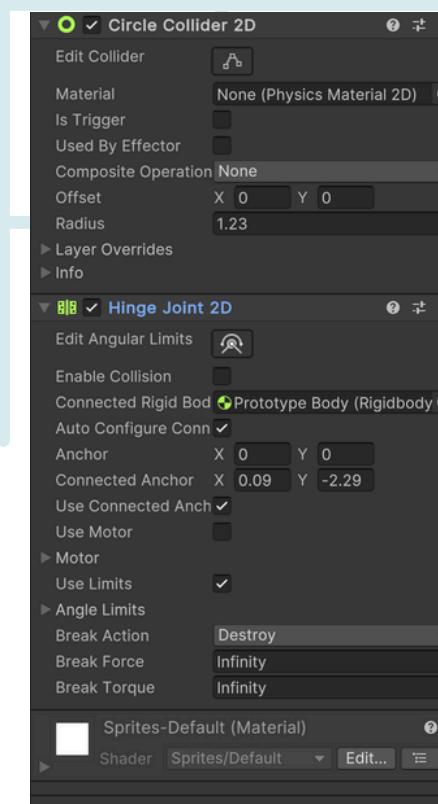
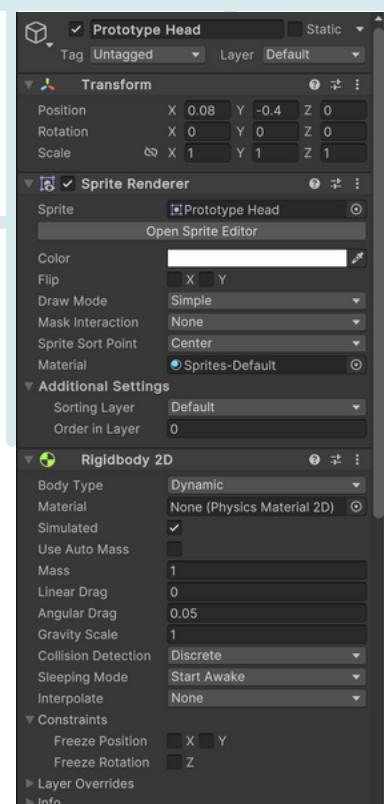


STEP 7.5

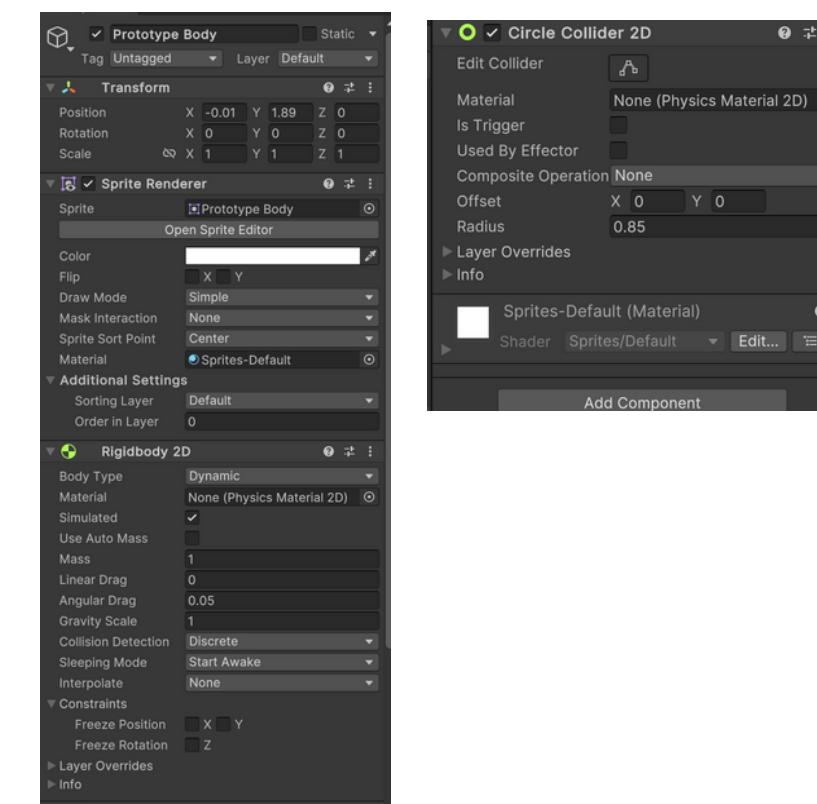
And Each Component with
these settings



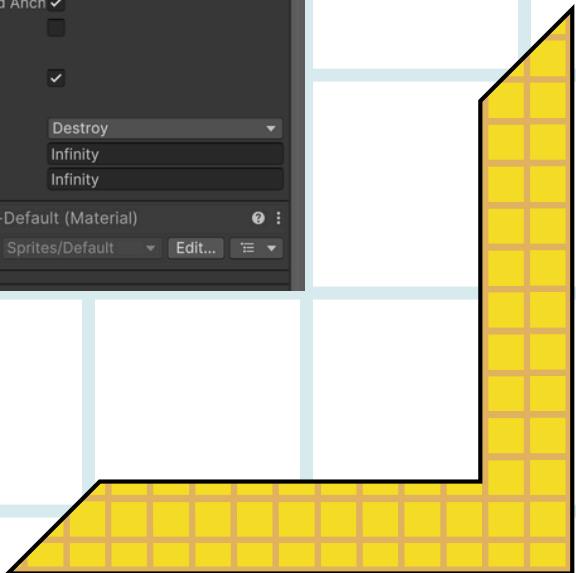
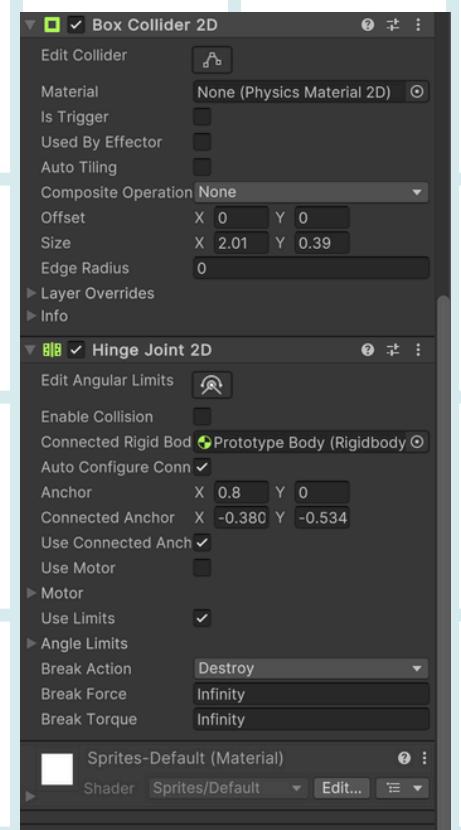
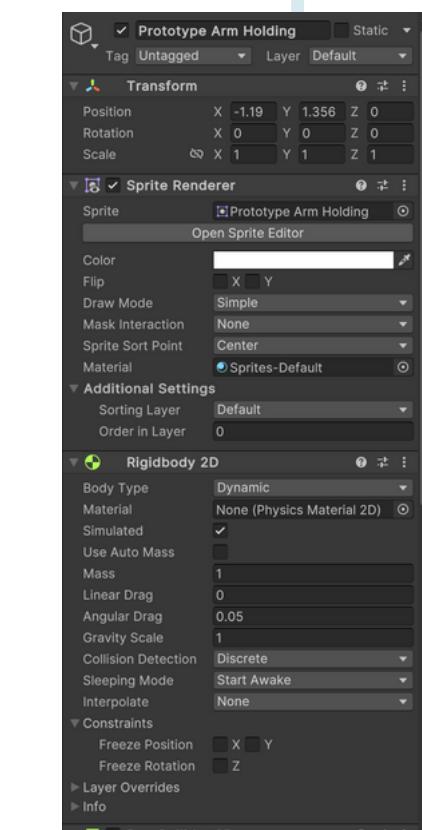
Prototype Head



Prototype Body

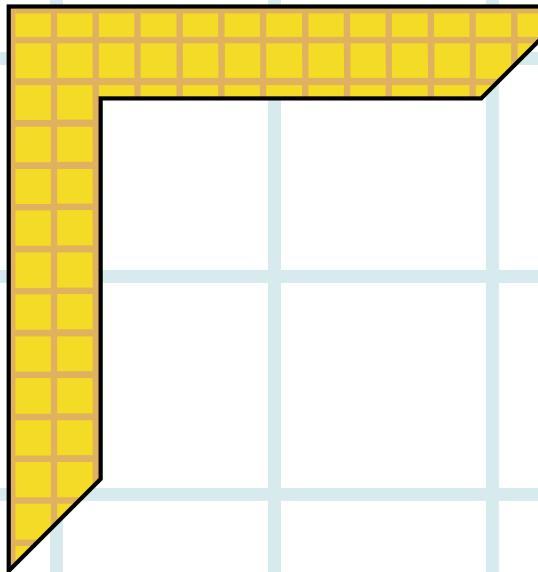


Prototype Arm Holding

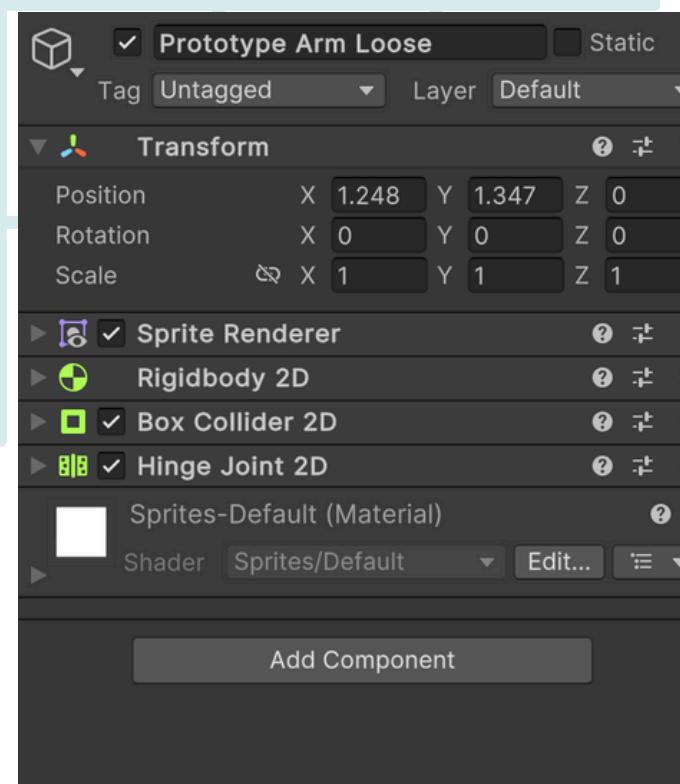


STEP 7.5

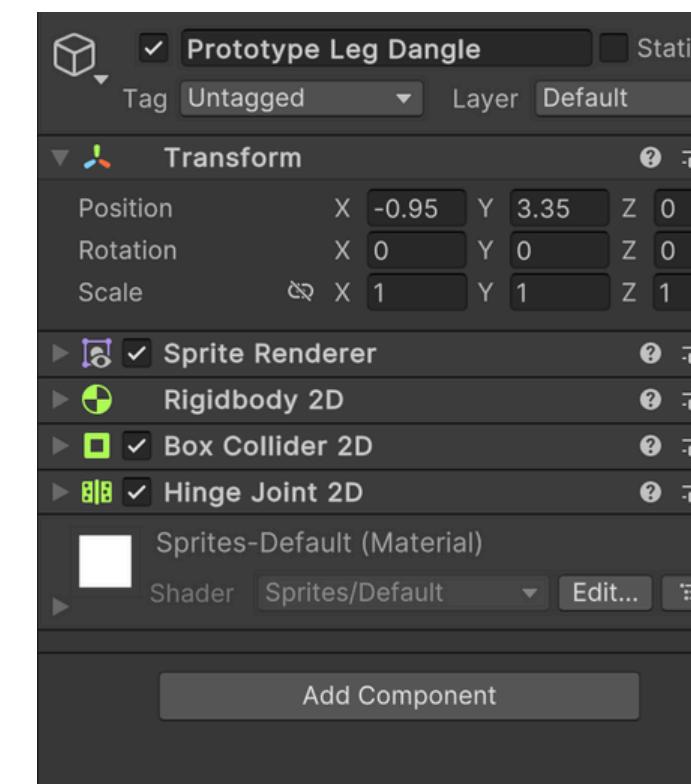
We clicked each part of the body and added components



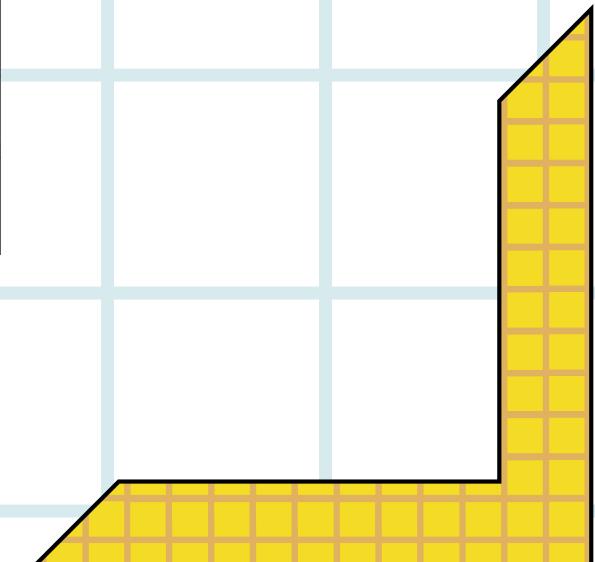
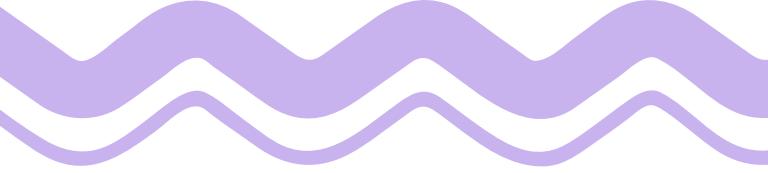
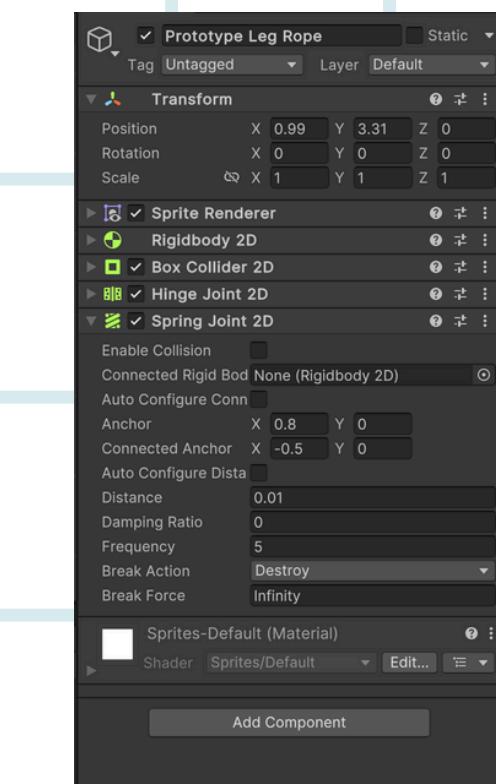
Prototype Arm Loose



Prototype Leg Dangle



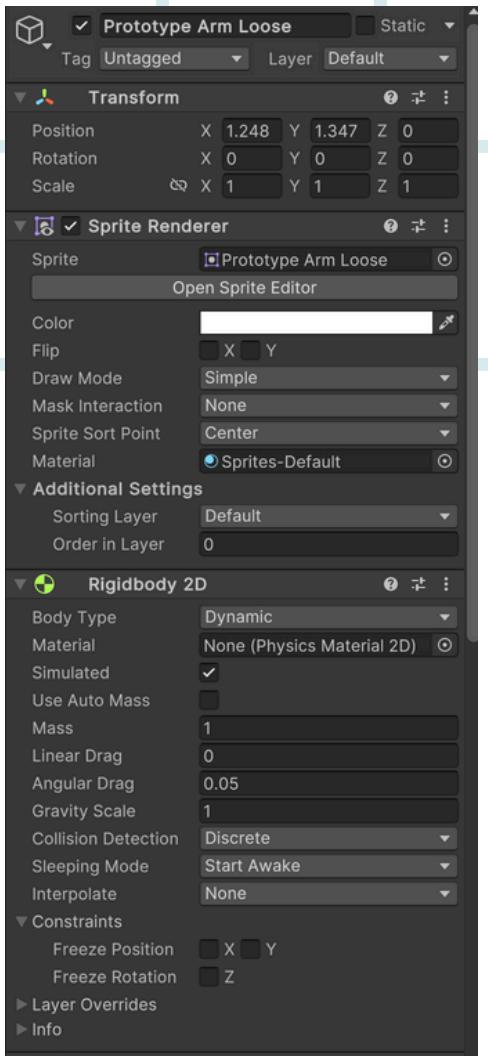
Prototype Leg Rope



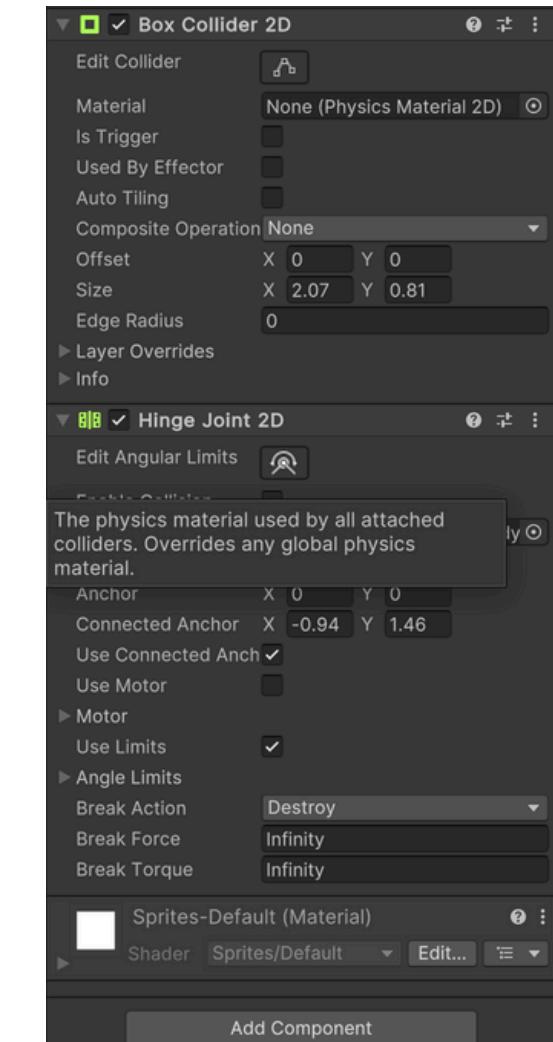
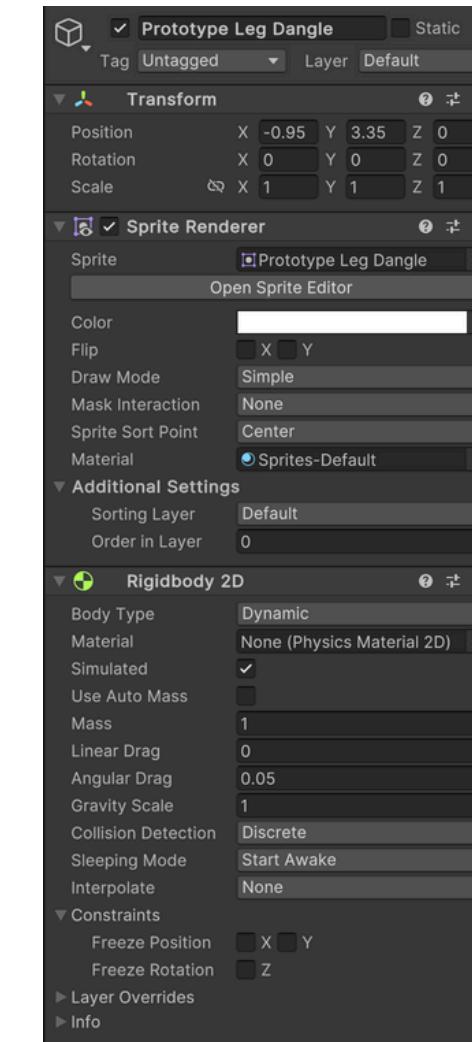
STEP 7.5

We clicked each part of the body and added components

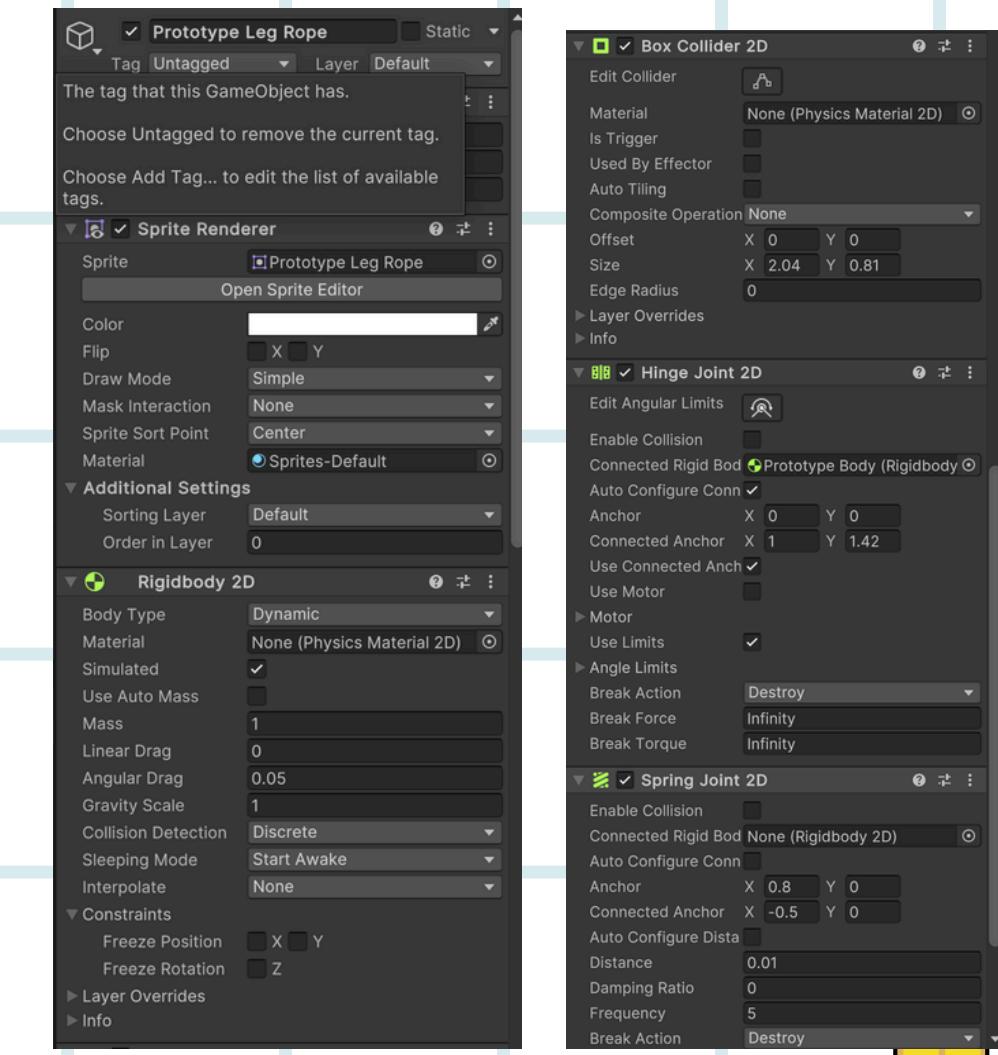
Prototype Arm Loose



Prototype Leg Dangle



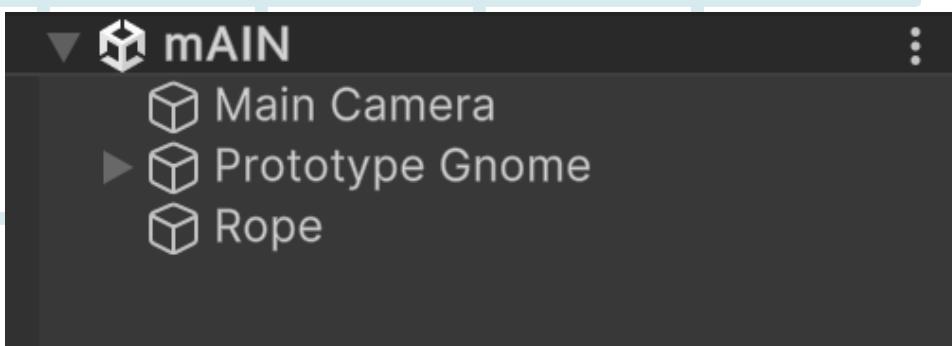
Prototype Leg Rope



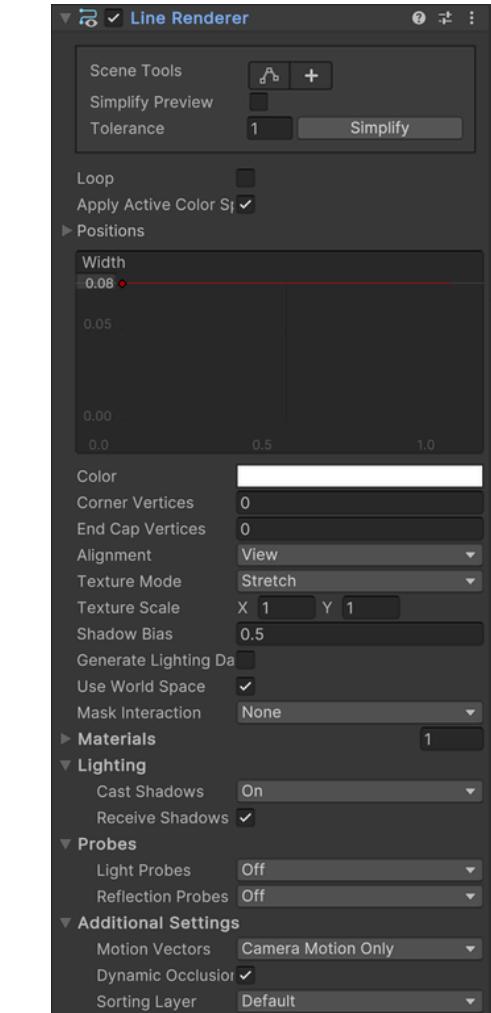
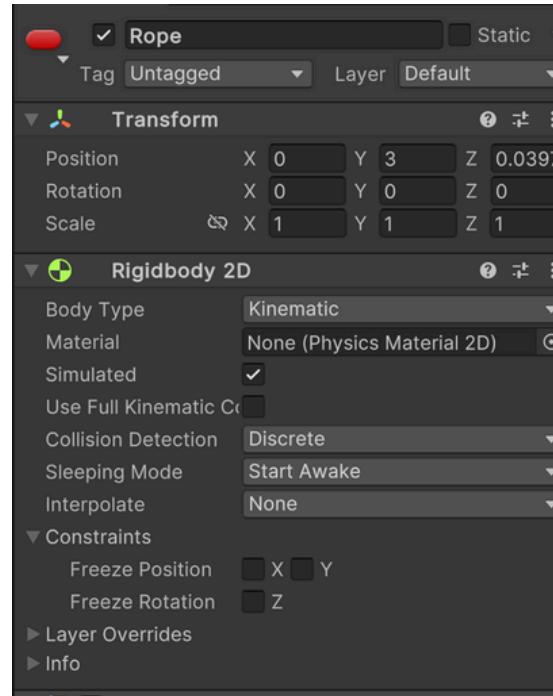
STEP 8

Configure the rope physics

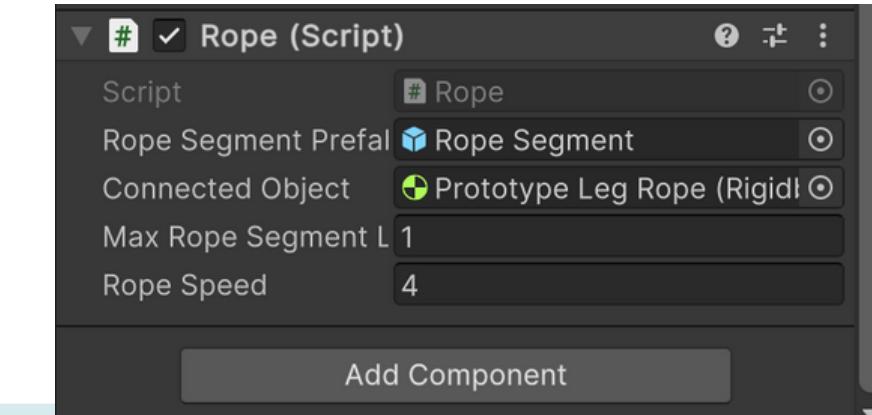
Add Rope Object



Add these settings



Add this C# code to make it work



With these code

```
using UnityEngine;
using System.Collections;
using System.Collections.Generic;
// The connected rope.
public class Rope : MonoBehaviour
{
    // The Rope Segment prefab to use.
    public GameObject ropeSegmentPrefab;
    // Contains a list of Rope Segment objects.
    List<GameObject> ropeSegments = new List<GameObject>();
    // Are we currently extending or retracting the rope?
    public bool isIncreasing { get; set; }
    public bool isDecreasing { get; set; }
    // The rigidbody object that the end of the rope
    // should be attached to.
    public Rigidbody2D connectedObject;
    // The maximum length a rope segment should be
    // if we
    // need to extend by more than this, create a new
    // rope
    // segment).
    public float maxRopeSegmentLength = 1.0f;
    // How quickly we should pay out new rope.
    public float ropeSpeed = 4.0f;
    // The LineRenderer that renders the actual rope.
    LineRenderer lineRenderer;
}
```

STEP 9

Connect evrything and run
the game



PROTOTYPE

These Steps are to create a prototype of the game Gnomes Well, Where we learned that this simple components and sprites are what games are and the endless possibilities we can do to improve the game.

CREATING BUTTONS

Create another cs file names
singleton



and add this code

```
# Singleton (Mono Script)  
Assembly Information  
Filename Assembly-CSharp.dll  
  
using UnityEngine;  
using System.Collections;  
// This class allows other objects to refer to a single  
// shared object. The GameManager and InputManager  
classes  
// use this.  
// To use this, subclass like so:  
// public class MyManager : Singleton<MyManager> {}  
// You can then access the single shared instance of the  
// class like so:  
// MyManager.instance.DoSomething();  
public class Singleton<T> : MonoBehaviour  
where T : MonoBehaviour  
{  
    // The single instance of this class.  
    private static T _instance;  
    // The accessor. The first time this is called, _instance  
    // will be set up. If an appropriate object can't be  
    found,  
    // an error will be logged.  
    public static T instance  
    {  
        get  
        {  
            // If we haven't already set up _instance...  
            if (_instance == null)  
            {  
                // Try to find the object.  
                _instance = FindObjectOfType<T>();  
                // Log if we can't find it.  
                if (_instance == null)  
                {  
                    Debug.LogError("Can't find " +  
                        typeof(T) + "!");  
                }  
            }  
            // Return the instance so that it can be used!  
            return _instance;  
        }  
    }  
}
```

CREATING BUTTONS

Create an object named inoutmanager and put this code



The screenshot shows the "Import Settings" dialog for the "Input Manager (Mono Script)" asset. It displays the assembly information and the script code. The script is a MonoBehavior that translates accelerometer data into sideways motion.

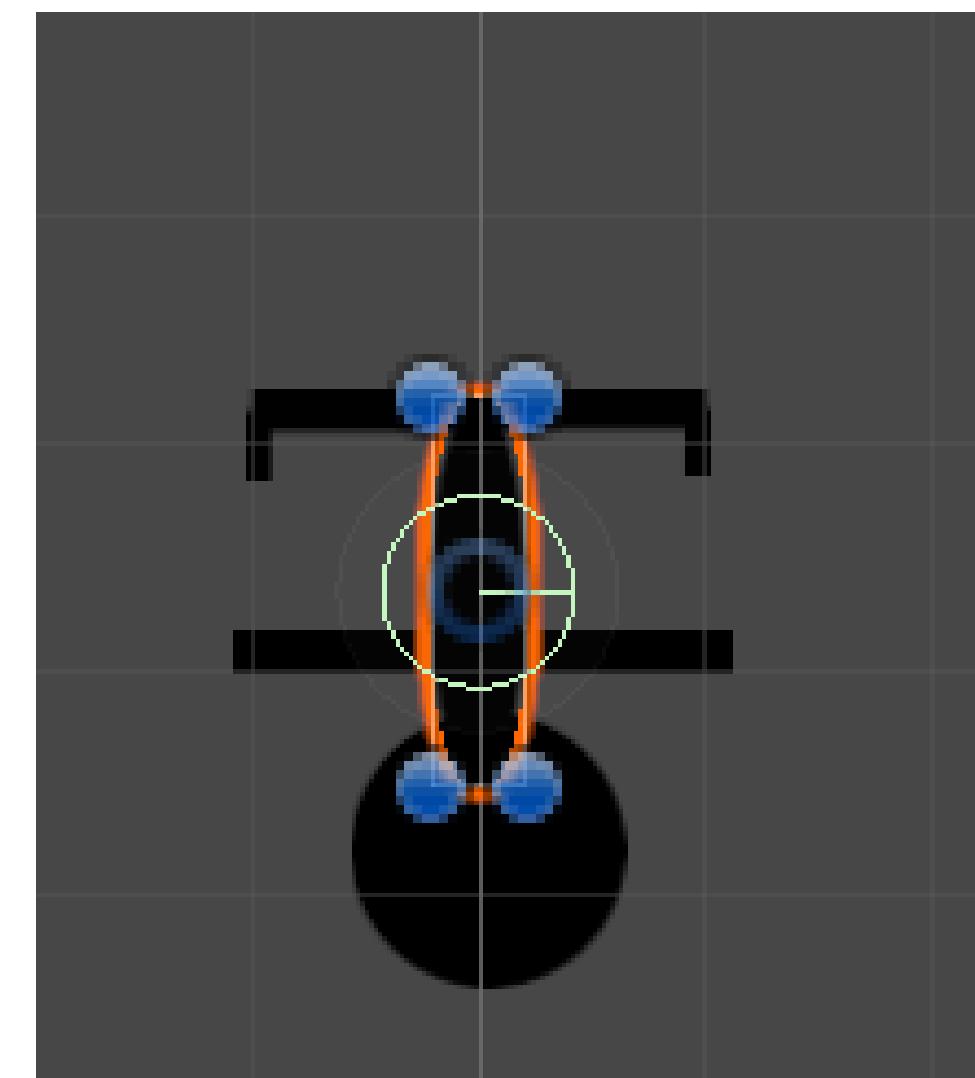
```
Input Manager (Mono Script) Import Settings

Imported Object
# Input Manager (Mono Script)

Assembly Information
Filename Assembly-CSharp.dll

using UnityEngine;
using System.Collections;
// Translates the accelerometer data into sideways motion
// info.
public class InputManager : Singleton<InputManager>
{
    // How much we're moving. -1.0 = full left, +1.0 = full
    // right
    private float _sidewaysMotion = 0.0f;
    // This property is declared as read-only, so that other
    // classes can't change it

    public float sidewaysMotion
    {
        get
        {
            return _sidewaysMotion;
        }
    }
    // Every frame, store the tilt
    void Update()
    {
        Vector3 accel = Input.acceleration;
        _sidewaysMotion = accel.x;
    }
}
```



The screenshot shows the "Import Settings" dialog for the "Swinging (Mono Script)" asset. It displays the assembly information and the script code. The script is a MonoBehaviour that applies sideways forces to a rigidbody based on the InputManager's sideways motion.

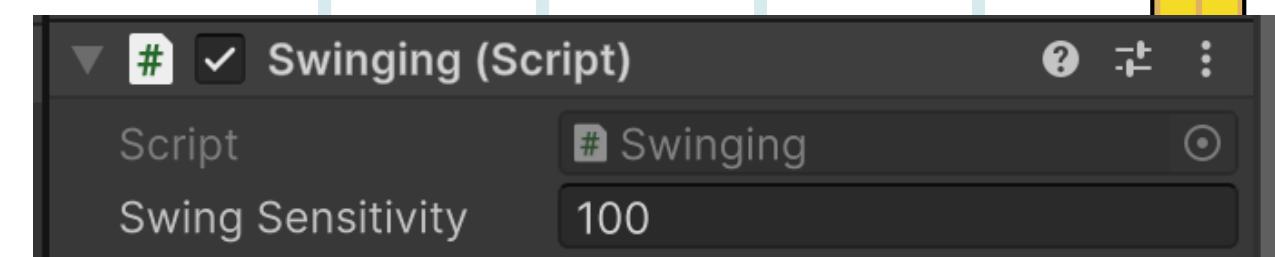
```
Swinging (Mono Script) Import Settings

Imported Object
# Swinging (Mono Script)

Assembly Information
Filename Assembly-CSharp.dll

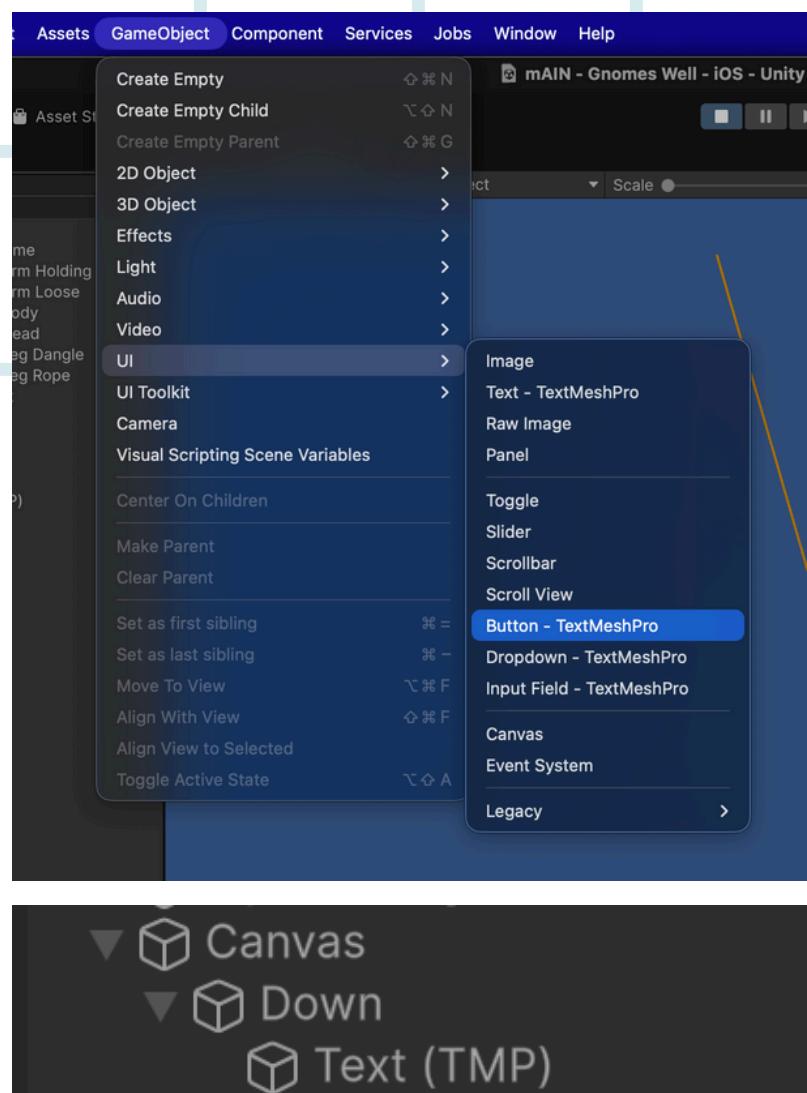
using UnityEngine;
using System.Collections;
// Uses the input manager to apply sideways forces to an
// object. Used to make the gnome swing side-to-side.
public class Swinging : MonoBehaviour
{
    // How much should we swing? Bigger numbers = more
    // swing
    public float swingSensitivity = 100.0f;

    // Use FixedUpdate instead of Update, in order to play
    // better with the physics engine
    void FixedUpdate()
    {
        // If we have no rigidbody (anymore), remove this
        // component
        if (GetComponent<Rigidbody2D>() == null)
        {
            Destroy(this);
            return;
        }
        // Get the tilt amount from the InputManager
        float swing =
        InputManager.instance.sidewaysMotion;
        // Calculate a force to apply
        Vector2 force =
        new Vector2(swing * swingSensitivity, 0);
        // Apply the force
        GetComponent<Rigidbody2D>().AddForce(force);
    }
}
```

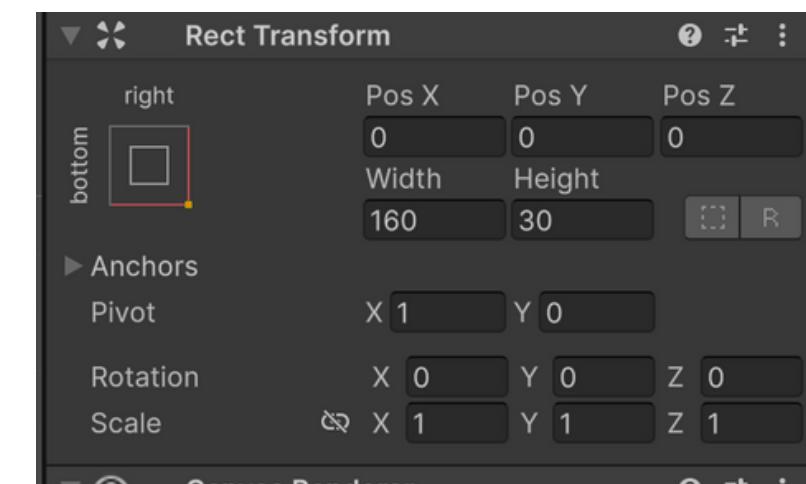


CREATING BUTTONS

Add new gameobject called button and rename it to Down



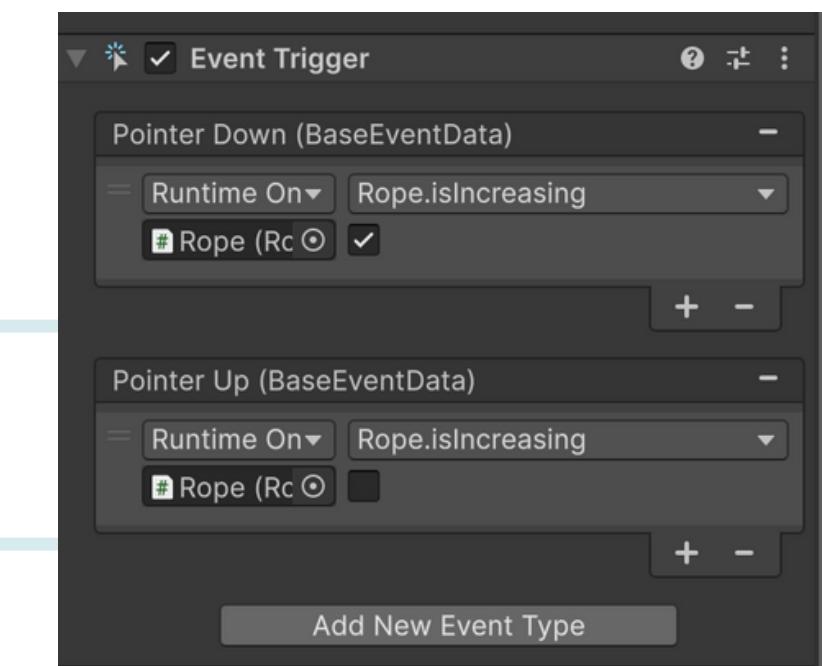
Click down and
configure this



Change the text child of the
button to Down

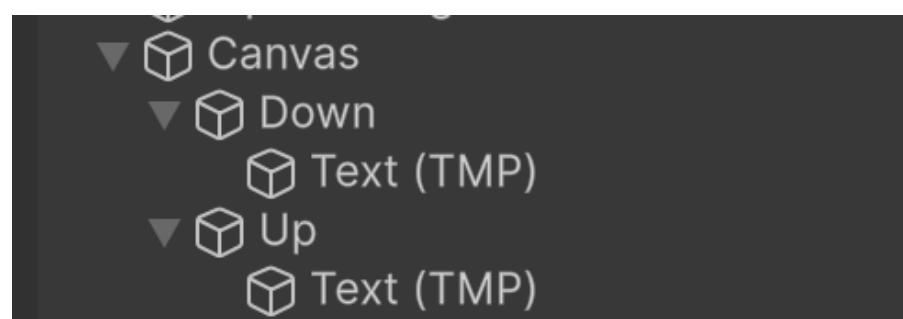
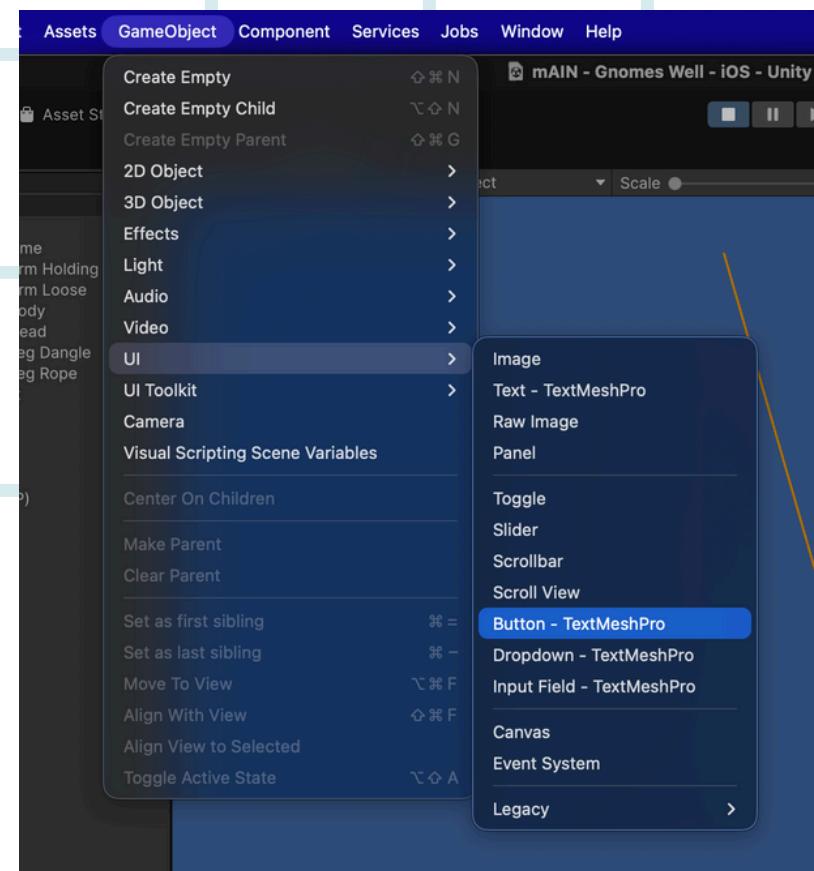


Add Event Trigger
component and add
this event type

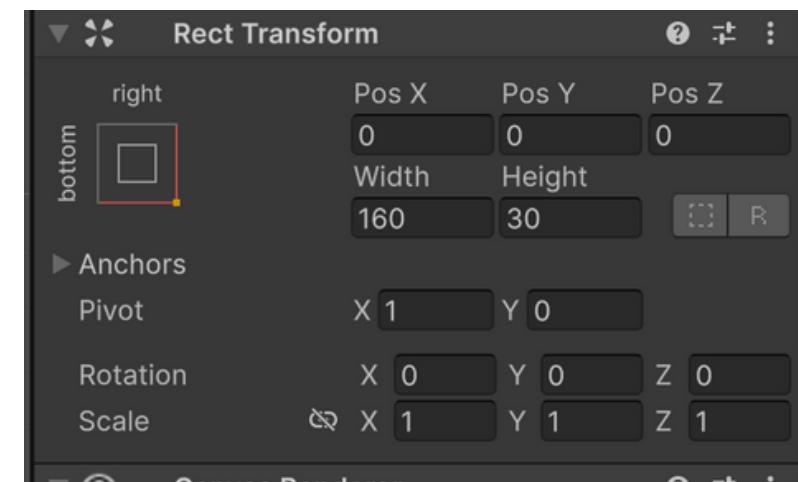


CREATING BUTTONS

Add new gameobject called button and rename it to Down and up



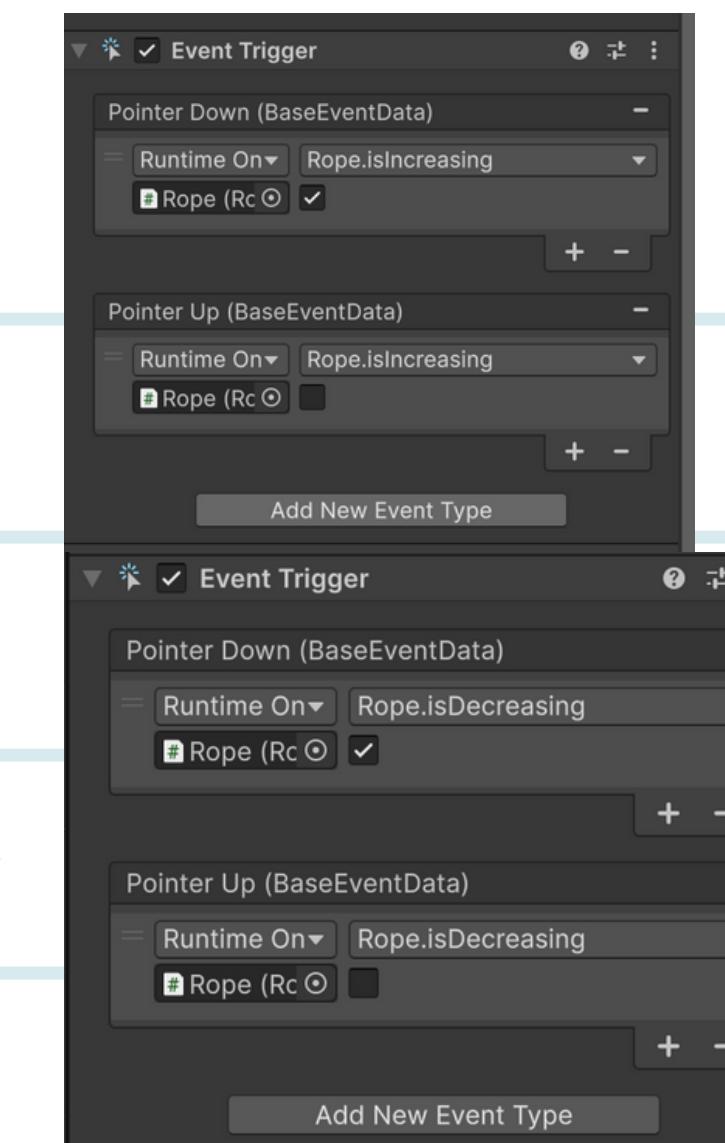
Click down and configure this



Change the text child of the button to Down and up



Add Event Trigger component and add this event type for both up and down



down

up

MAKING THE CAMERA FOLLOW

ADD this code in the camerafollow.cs

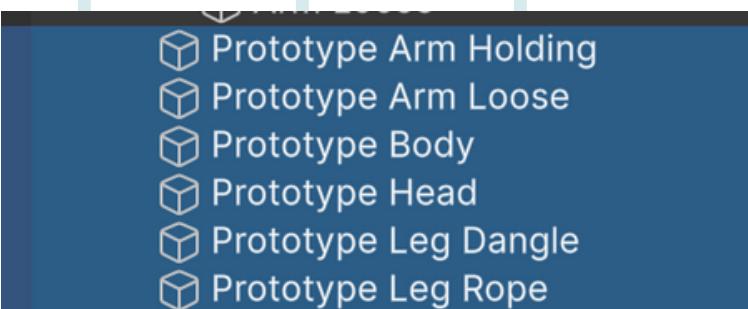
```
1  using UnityEngine;
2
3  // Adjusts the camera to always match the Y-position of a
4  // target object, within certain limits.
5  public class CameraFollow : MonoBehaviour
6  {
7      // The object we want to match the Y position of.
8      public Transform target;
9
10     // The highest point the camera can go.
11     public float topLimit = 10.0f;
12
13     // The lowest point the camera can go.
14     public float bottomLimit = -10.0f;
15
16     // How quickly we should move toward the target.
17     public float followSpeed = 0.5f;
18
19     // After all objects have updated position, work out where
20     // this camera should be
21     void LateUpdate()
22     {
23         // If we have a target...
24         if(target != null)
25         {
26             // Get its position
27             Vector3 newPosition = this.transform.position;
28             // Work out where this camera should be
29             newPosition.y = Mathf.Lerp(newPosition.y,
30
31                 target.position.y, followSpeed);
32
33             // Clamp this new location to within our
34             // limits
35             newPosition.y =
36                 Mathf.Min(newPosition.y, topLimit);
37             newPosition.y =
38                 Mathf.Max(newPosition.y, bottomLimit);
39
40             // Update our location
41             transform.position = newPosition;
42         }
43
44         // When selected in the editor, draw a line from the top
45         // limit to the bottom.
46         void OnDrawGizmosSelected()
47         {
48             Gizmos.color = Color.yellow;
49             Vector3 topPoint =
50                 new Vector3(this.transform.position.x,
51                 topLimit, this.transform.position.z);
52             Vector3 bottomPoint =
53                 new Vector3(this.transform.position.x,
54                 bottomLimit, this.transform.position.z);
55             Gizmos.DrawLine(topPoint, bottomPoint);
56         }
57     }
58
59     public void DestroyGnome(DamageType type)
60     {
61         // The object that the camera should follow.
62         public Rigidbody2D cameraFollowTarget;
63
64         // The joint that the camera should follow.
65         public Joint2D cameraFollowJoint;
66
67         // The object we want to match the Y position of.
68         public Rigidbody2D ropeBody;
69
70         // The joint that connects the two objects.
71         public Joint2D armHoldingTreasure;
72
73         // The sprite renderer holding the treasure.
74         public SpriteRenderer holdingArm;
75
76         // The object that holds the treasure.
77         public GnomeObject flameDeathPrefab;
78
79         public float delayBeforeReleasingPost = 3.0f;
80         public float delayBeforeReleasingPost = 0.25f;
81
82         bool dead = false;
83         float holdingTreasureValue;
84
85         public float holdingTreasure;
86
87         get
88         {
89             return _holdingTreasure;
90         }
91         set
92         {
93             if (dead == true)
94             {
95                 return;
96             }
97             _holdingTreasure = value;
98             if (holdingArm != null)
99             {
100                 if (_holdingTreasure)
101                     holdingArm.sprite =
102                         armHoldingTreasure;
103                 else
104                     holdingArm.sprite =
105                         armHoldingEmpty;
106             }
107         }
108
109         public enum DamageType
110         {
111             Slicing,
112             Burning
113         }
114
115         public void ShowDamageEffect(DamageType type)
116         {
117             switch (type)
118             {
119                 case DamageType.Burning:
120                     if (flameDeathPrefab != null)
121                         Instantiate(
122                             flameDeathPrefab,
123                             cameraFollowTarget.position,
124                             cameraFollowTarget.rotation);
125                     break;
126                 case DamageType.Slicing:
127                     if (flameDeathPrefab != null)
128                         Instantiate(
129                             flameDeathPrefab,
130                             cameraFollowTarget.position,
131                             cameraFollowTarget.rotation);
132             }
133         }
134
135         // Add a RemoveAfterDelay component to this object
136         var remove = gameObject.AddComponent<RemoveAfterDelay>();
137         remove.Delay = 1.0f;
138         remove.OnStart();
139         remove.OnDestroy();
140
141         // Create a new joint
142         var joint = new Joint2D();
143         joint.type = JointType.Fixed;
144         joint.connectedBody = target;
145         joint.connectedAnchor = target.position;
146         joint.localAnchorA = Vector2.zero;
147         joint.localAnchorB = Vector2.zero;
148         joint.enableCollision = false;
149         joint.spring = 0.0f;
150         joint.damper = 0.0f;
151         joint.bounciness = 0.0f;
152         joint.frequency = 0.0f;
153         joint.mass = 0.0f;
154         joint.stiffness = 0.0f;
155
156         // Attach the joint to the camera
157         cameraFollowTarget.gameObject.AddComponent<Joint2D>().joint = joint;
158
159         // Set the target
160         cameraFollowTarget.target = target;
161
162         // Set the rotation
163         cameraFollowTarget.rotation = target.rotation;
164
165         // Set the position
166         cameraFollowTarget.position = target.position;
167
168         // Set the scale
169         cameraFollowTarget.localScale = target.localScale;
170
171         // Set the color
172         cameraFollowTarget.color = target.color;
173
174         // Set the material
175         cameraFollowTarget.material = target.material;
176
177         // Set the render mode
178         cameraFollowTarget.renderMode = target.renderMode;
179
180         // Set the texture
181         cameraFollowTarget.texture = target.texture;
182
183         // Set the type
184         cameraFollowTarget.type = target.type;
185
186         // Set the visibility
187         cameraFollowTarget.visibility = target.visibility;
188
189         // Set the depth
190         cameraFollowTarget.depth = target.depth;
191
192         // Set the shadow casting
193         cameraFollowTarget.receiveShadows =
194             target.receiveShadows;
195
196         // Set the shadow receiving
197         cameraFollowTarget.castShadows =
198             target.castShadows;
199
200         // Set the physics
201         cameraFollowTarget.physicsMaterial =
202             target.physicsMaterial;
203
204         // Set the physics settings
205         cameraFollowTarget.physicsShapeType =
206             target.physicsShapeType;
207
208         // Set the physics settings
209         cameraFollowTarget.physicsShapeType =
210             target.physicsShapeType;
211
212         // Set the physics settings
213         cameraFollowTarget.physicsShapeType =
214             target.physicsShapeType;
215
216         // Set the physics settings
217         cameraFollowTarget.physicsShapeType =
218             target.physicsShapeType;
219
220         // Set the physics settings
221         cameraFollowTarget.physicsShapeType =
222             target.physicsShapeType;
223
224         // Set the physics settings
225         cameraFollowTarget.physicsShapeType =
226             target.physicsShapeType;
227
228         // Set the physics settings
229         cameraFollowTarget.physicsShapeType =
230             target.physicsShapeType;
231
232         // Set the physics settings
233         cameraFollowTarget.physicsShapeType =
234             target.physicsShapeType;
235
236         // Set the physics settings
237         cameraFollowTarget.physicsShapeType =
238             target.physicsShapeType;
239
240         // Set the physics settings
241         cameraFollowTarget.physicsShapeType =
242             target.physicsShapeType;
243
244         // Set the physics settings
245         cameraFollowTarget.physicsShapeType =
246             target.physicsShapeType;
247
248         // Set the physics settings
249         cameraFollowTarget.physicsShapeType =
250             target.physicsShapeType;
251
252         // Set the physics settings
253         cameraFollowTarget.physicsShapeType =
254             target.physicsShapeType;
255
256         // Set the physics settings
257         cameraFollowTarget.physicsShapeType =
258             target.physicsShapeType;
259
260         // Set the physics settings
261         cameraFollowTarget.physicsShapeType =
262             target.physicsShapeType;
263
264         // Set the physics settings
265         cameraFollowTarget.physicsShapeType =
266             target.physicsShapeType;
267
268         // Set the physics settings
269         cameraFollowTarget.physicsShapeType =
270             target.physicsShapeType;
271
272         // Set the physics settings
273         cameraFollowTarget.physicsShapeType =
274             target.physicsShapeType;
275
276         // Set the physics settings
277         cameraFollowTarget.physicsShapeType =
278             target.physicsShapeType;
279
280         // Set the physics settings
281         cameraFollowTarget.physicsShapeType =
282             target.physicsShapeType;
283
284         // Set the physics settings
285         cameraFollowTarget.physicsShapeType =
286             target.physicsShapeType;
287
288         // Set the physics settings
289         cameraFollowTarget.physicsShapeType =
290             target.physicsShapeType;
291
292         // Set the physics settings
293         cameraFollowTarget.physicsShapeType =
294             target.physicsShapeType;
295
296         // Set the physics settings
297         cameraFollowTarget.physicsShapeType =
298             target.physicsShapeType;
299
300         // Set the physics settings
301         cameraFollowTarget.physicsShapeType =
302             target.physicsShapeType;
303
304         // Set the physics settings
305         cameraFollowTarget.physicsShapeType =
306             target.physicsShapeType;
307
308         // Set the physics settings
309         cameraFollowTarget.physicsShapeType =
310             target.physicsShapeType;
311
312         // Set the physics settings
313         cameraFollowTarget.physicsShapeType =
314             target.physicsShapeType;
315
316         // Set the physics settings
317         cameraFollowTarget.physicsShapeType =
318             target.physicsShapeType;
319
320         // Set the physics settings
321         cameraFollowTarget.physicsShapeType =
322             target.physicsShapeType;
323
324         // Set the physics settings
325         cameraFollowTarget.physicsShapeType =
326             target.physicsShapeType;
327
328         // Set the physics settings
329         cameraFollowTarget.physicsShapeType =
330             target.physicsShapeType;
331
332         // Set the physics settings
333         cameraFollowTarget.physicsShapeType =
334             target.physicsShapeType;
335
336         // Set the physics settings
337         cameraFollowTarget.physicsShapeType =
338             target.physicsShapeType;
339
340         // Set the physics settings
341         cameraFollowTarget.physicsShapeType =
342             target.physicsShapeType;
343
344         // Set the physics settings
345         cameraFollowTarget.physicsShapeType =
346             target.physicsShapeType;
347
348         // Set the physics settings
349         cameraFollowTarget.physicsShapeType =
350             target.physicsShapeType;
351
352         // Set the physics settings
353         cameraFollowTarget.physicsShapeType =
354             target.physicsShapeType;
355
356         // Set the physics settings
357         cameraFollowTarget.physicsShapeType =
358             target.physicsShapeType;
359
360         // Set the physics settings
361         cameraFollowTarget.physicsShapeType =
362             target.physicsShapeType;
363
364         // Set the physics settings
365         cameraFollowTarget.physicsShapeType =
366             target.physicsShapeType;
367
368         // Set the physics settings
369         cameraFollowTarget.physicsShapeType =
370             target.physicsShapeType;
371
372         // Set the physics settings
373         cameraFollowTarget.physicsShapeType =
374             target.physicsShapeType;
375
376         // Set the physics settings
377         cameraFollowTarget.physicsShapeType =
378             target.physicsShapeType;
379
380         // Set the physics settings
381         cameraFollowTarget.physicsShapeType =
382             target.physicsShapeType;
383
384         // Set the physics settings
385         cameraFollowTarget.physicsShapeType =
386             target.physicsShapeType;
387
388         // Set the physics settings
389         cameraFollowTarget.physicsShapeType =
390             target.physicsShapeType;
391
392         // Set the physics settings
393         cameraFollowTarget.physicsShapeType =
394             target.physicsShapeType;
395
396         // Set the physics settings
397         cameraFollowTarget.physicsShapeType =
398             target.physicsShapeType;
399
400         // Set the physics settings
401         cameraFollowTarget.physicsShapeType =
402             target.physicsShapeType;
403
404         // Set the physics settings
405         cameraFollowTarget.physicsShapeType =
406             target.physicsShapeType;
407
408         // Set the physics settings
409         cameraFollowTarget.physicsShapeType =
410             target.physicsShapeType;
411
412         // Set the physics settings
413         cameraFollowTarget.physicsShapeType =
414             target.physicsShapeType;
415
416         // Set the physics settings
417         cameraFollowTarget.physicsShapeType =
418             target.physicsShapeType;
419
420         // Set the physics settings
421         cameraFollowTarget.physicsShapeType =
422             target.physicsShapeType;
423
424         // Set the physics settings
425         cameraFollowTarget.physicsShapeType =
426             target.physicsShapeType;
427
428         // Set the physics settings
429         cameraFollowTarget.physicsShapeType =
430             target.physicsShapeType;
431
432         // Set the physics settings
433         cameraFollowTarget.physicsShapeType =
434             target.physicsShapeType;
435
436         // Set the physics settings
437         cameraFollowTarget.physicsShapeType =
438             target.physicsShapeType;
439
440         // Set the physics settings
441         cameraFollowTarget.physicsShapeType =
442             target.physicsShapeType;
443
444         // Set the physics settings
445         cameraFollowTarget.physicsShapeType =
446             target.physicsShapeType;
447
448         // Set the physics settings
449         cameraFollowTarget.physicsShapeType =
450             target.physicsShapeType;
451
452         // Set the physics settings
453         cameraFollowTarget.physicsShapeType =
454             target.physicsShapeType;
455
456         // Set the physics settings
457         cameraFollowTarget.physicsShapeType =
458             target.physicsShapeType;
459
460         // Set the physics settings
461         cameraFollowTarget.physicsShapeType =
462             target.physicsShapeType;
463
464         // Set the physics settings
465         cameraFollowTarget.physicsShapeType =
466             target.physicsShapeType;
467
468         // Set the physics settings
469         cameraFollowTarget.physicsShapeType =
470             target.physicsShapeType;
471
472         // Set the physics settings
473         cameraFollowTarget.physicsShapeType =
474             target.physicsShapeType;
475
476         // Set the physics settings
477         cameraFollowTarget.physicsShapeType =
478             target.physicsShapeType;
479
480         // Set the physics settings
481         cameraFollowTarget.physicsShapeType =
482             target.physicsShapeType;
483
484         // Set the physics settings
485         cameraFollowTarget.physicsShapeType =
486             target.physicsShapeType;
487
488         // Set the physics settings
489         cameraFollowTarget.physicsShapeType =
490             target.physicsShapeType;
491
492         // Set the physics settings
493         cameraFollowTarget.physicsShapeType =
494             target.physicsShapeType;
495
496         // Set the physics settings
497         cameraFollowTarget.physicsShapeType =
498             target.physicsShapeType;
499
500         // Set the physics settings
501         cameraFollowTarget.physicsShapeType =
502             target.physicsShapeType;
503
504         // Set the physics settings
505         cameraFollowTarget.physicsShapeType =
506             target.physicsShapeType;
507
508         // Set the physics settings
509         cameraFollowTarget.physicsShapeType =
510             target.physicsShapeType;
511
512         // Set the physics settings
513         cameraFollowTarget.physicsShapeType =
514             target.physicsShapeType;
515
516         // Set the physics settings
517         cameraFollowTarget.physicsShapeType =
518             target.physicsShapeType;
519
520         // Set the physics settings
521         cameraFollowTarget.physicsShapeType =
522             target.physicsShapeType;
523
524         // Set the physics settings
525         cameraFollowTarget.physicsShapeType =
526             target.physicsShapeType;
527
528         // Set the physics settings
529         cameraFollowTarget.physicsShapeType =
530             target.physicsShapeType;
531
532         // Set the physics settings
533         cameraFollowTarget.physicsShapeType =
534             target.physicsShapeType;
535
536         // Set the physics settings
537         cameraFollowTarget.physicsShapeType =
538             target.physicsShapeType;
539
540         // Set the physics settings
541         cameraFollowTarget.physicsShapeType =
542             target.physicsShapeType;
543
544         // Set the physics settings
545         cameraFollowTarget.physicsShapeType =
546             target.physicsShapeType;
547
548         // Set the physics settings
549         cameraFollowTarget.physicsShapeType =
550             target.physicsShapeType;
551
552         // Set the physics settings
553         cameraFollowTarget.physicsShapeType =
554             target.physicsShapeType;
555
556         // Set the physics settings
557         cameraFollowTarget.physicsShapeType =
558             target.physicsShapeType;
559
560         // Set the physics settings
561         cameraFollowTarget.physicsShapeType =
562             target.physicsShapeType;
563
564         // Set the physics settings
565         cameraFollowTarget.physicsShapeType =
566             target.physicsShapeType;
567
568         // Set the physics settings
569         cameraFollowTarget.physicsShapeType =
570             target.physicsShapeType;
571
572         // Set the physics settings
573         cameraFollowTarget.physicsShapeType =
574             target.physicsShapeType;
575
576         // Set the physics settings
577         cameraFollowTarget.physicsShapeType =
578             target.physicsShapeType;
579
580         // Set the physics settings
581         cameraFollowTarget.physicsShapeType =
582             target.physicsShapeType;
583
584         // Set the physics settings
585         cameraFollowTarget.physicsShapeType =
586             target.physicsShapeType;
587
588         // Set the physics settings
589         cameraFollowTarget.physicsShapeType =
590             target.physicsShapeType;
591
592         // Set the physics settings
593         cameraFollowTarget.physicsShapeType =
594             target.physicsShapeType;
595
596         // Set the physics settings
597         cameraFollowTarget.physicsShapeType =
598             target.physicsShapeType;
599
599
600         // Set the physics settings
601         cameraFollowTarget.physicsShapeType =
602             target.physicsShapeType;
603
604         // Set the physics settings
605         cameraFollowTarget.physicsShapeType =
606             target.physicsShapeType;
607
608         // Set the physics settings
609         cameraFollowTarget.physicsShapeType =
610             target.physicsShapeType;
611
612         // Set the physics settings
613         cameraFollowTarget.physicsShapeType =
614             target.physicsShapeType;
615
616         // Set the physics settings
617         cameraFollowTarget.physicsShapeType =
618             target.physicsShapeType;
619
620         // Set the physics settings
621         cameraFollowTarget.physicsShapeType =
622             target.physicsShapeType;
623
624         // Set the physics settings
625         cameraFollowTarget.physicsShapeType =
626             target.physicsShapeType;
627
628         // Set the physics settings
629         cameraFollowTarget.physicsShapeType =
630             target.physicsShapeType;
631
632         // Set the physics settings
633         cameraFollowTarget.physicsShapeType =
634             target.physicsShapeType;
635
636         // Set the physics settings
637         cameraFollowTarget.physicsShapeType =
638             target.physicsShapeType;
639
639
640         // Set the physics settings
641         cameraFollowTarget.physicsShapeType =
642             target.physicsShapeType;
643
644         // Set the physics settings
645         cameraFollowTarget.physicsShapeType =
646             target.physicsShapeType;
647
648         // Set the physics settings
649         cameraFollowTarget.physicsShapeType =
650             target.physicsShapeType;
651
652         // Set the physics settings
653         cameraFollowTarget.physicsShapeType =
654             target.physicsShapeType;
655
656         // Set the physics settings
657         cameraFollowTarget.physicsShapeType =
658             target.physicsShapeType;
659
659
660         // Set the physics settings
661         cameraFollowTarget.physicsShapeType =
662             target.physicsShapeType;
663
664         // Set the physics settings
665         cameraFollowTarget.physicsShapeType =
666             target.physicsShapeType;
667
668         // Set the physics settings
669         cameraFollowTarget.physicsShapeType =
670             target.physicsShapeType;
671
672         // Set the physics settings
673         cameraFollowTarget.physicsShapeType =
674             target.physicsShapeType;
675
676         // Set the physics settings
677         cameraFollowTarget.physicsShapeType =
678             target.physicsShapeType;
679
679
680         // Set the physics settings
681         cameraFollowTarget.physicsShapeType =
682             target.physicsShapeType;
683
684         // Set the physics settings
685         cameraFollowTarget.physicsShapeType =
686             target.physicsShapeType;
687
688         // Set the physics settings
689         cameraFollowTarget.physicsShapeType =
690             target.physicsShapeType;
691
692         // Set the physics settings
693         cameraFollowTarget.physicsShapeType =
694             target.physicsShapeType;
695
695
696         // Set the physics settings
697         cameraFollowTarget.physicsShapeType =
698             target.physicsShapeType;
699
700         // Set the physics settings
701         cameraFollowTarget.physicsShapeType =
702             target.physicsShapeType;
703
704         // Set the physics settings
705         cameraFollowTarget.physicsShapeType =
706             target.physicsShapeType;
707
708         // Set the physics settings
709         cameraFollowTarget.physicsShapeType =
710             target.physicsShapeType;
711
711
712         // Set the physics settings
713         cameraFollowTarget.physicsShapeType =
714             target.physicsShapeType;
715
716         // Set the physics settings
717         cameraFollowTarget.physicsShapeType =
718             target.physicsShapeType;
719
720         // Set the physics settings
721         cameraFollowTarget.physicsShapeType =
722             target.physicsShapeType;
723
724         // Set the physics settings
725         cameraFollowTarget.physicsShapeType =
726             target.physicsShapeType;
727
727
728         // Set the physics settings
729         cameraFollowTarget.physicsShapeType =
730             target.physicsShapeType;
731
732         // Set the physics settings
733         cameraFollowTarget.physicsShapeType =
734             target.physicsShapeType;
735
736         // Set the physics settings
737         cameraFollowTarget.physicsShapeType =
738             target.physicsShapeType;
739
739
740         // Set the physics settings
741         cameraFollowTarget.physicsShapeType =
742             target.physicsShapeType;
743
744         // Set the physics settings
745         cameraFollowTarget.physicsShapeType =
746             target.physicsShapeType;
747
748         // Set the physics settings
749         cameraFollowTarget.physicsShapeType =
750             target.physicsShapeType;
751
751
752         // Set the physics settings
753         cameraFollowTarget.physicsShapeType =
754             target.physicsShapeType;
755
756         // Set the physics settings
757         cameraFollowTarget.physicsShapeType =
758             target.physicsShapeType;
759
759
760         // Set the physics settings
761         cameraFollowTarget.physicsShapeType =
762             target.physicsShapeType;
763
764         // Set the physics settings
765         cameraFollowTarget.physicsShapeType =
766             target.physicsShapeType;
767
768         // Set the physics settings
769         cameraFollowTarget.physicsShapeType =
770             target.physicsShapeType;
771
771
772         // Set the physics settings
773         cameraFollowTarget.physicsShapeType =
774             target.physicsShapeType;
775
776         // Set the physics settings
777         cameraFollowTarget.physicsShapeType =
778             target.physicsShapeType;
779
779
780         // Set the physics settings
781         cameraFollowTarget.physicsShapeType =
782             target.physicsShapeType;
783
784         // Set the physics settings
785         cameraFollowTarget.physicsShapeType =
786             target.physicsShapeType;
787
787
788         // Set the physics settings
789         cameraFollowTarget.physicsShapeType =
790             target.physicsShapeType;
791
792         // Set the physics settings
793         cameraFollowTarget.physicsShapeType =
794             target.physicsShapeType;
795
795
796         // Set the physics settings
797         cameraFollowTarget.physicsShapeType =
798             target.physicsShapeType;
799
800         // Set the physics settings
801         cameraFollowTarget.physicsShapeType =
802             target.physicsShapeType;
803
803
804         // Set the physics settings
805         cameraFollowTarget.physicsShapeType =
806             target.physicsShapeType;
807
808         // Set the physics settings
809         cameraFollowTarget.physicsShapeType =
810             target.physicsShapeType;
811
811
812         // Set the physics settings
813         cameraFollowTarget.physicsShapeType =
814             target.physicsShapeType;
815
816         // Set the physics settings
817         cameraFollowTarget.physicsShapeType =
818             target.physicsShapeType;
819
819
820         // Set the physics settings
821         cameraFollowTarget.physicsShapeType =
822             target.physicsShapeType;
823
824         // Set the physics settings
825         cameraFollowTarget.physicsShapeType =
826             target.physicsShapeType;
827
827
828         // Set the physics settings
829         cameraFollowTarget.physicsShapeType =
830             target.physicsShapeType;
831
832         // Set the physics settings
833         cameraFollowTarget.physicsShapeType =
834             target.physicsShapeType;
835
835
836         // Set the physics settings
837         cameraFollowTarget.physicsShapeType =
838             target.physicsShapeType;
839
839
840         // Set the physics settings
841         cameraFollowTarget.physicsShapeType =
842             target.physicsShapeType;
843
844         // Set the physics settings
845         cameraFollowTarget.physicsShapeType =
846             target.physicsShapeType;
847
847
848         // Set the physics settings
849         cameraFollowTarget.physicsShapeType =
850             target.physicsShapeType;
851
852         // Set the physics settings
853         cameraFollowTarget.physicsShapeType =
854             target.physicsShapeType;
855
855
856         // Set the physics settings
857         cameraFollowTarget.physicsShapeType =
858             target.physicsShapeType;
859
859
860         // Set the physics settings
861         cameraFollowTarget.physicsShapeType =
862             target.physicsShapeType;
863
864         // Set the physics settings
865         cameraFollowTarget.physicsShapeType =
866             target.physicsShapeType;
867
867
868         // Set the physics settings
869         cameraFollowTarget.physicsShapeType =
870             target.physicsShapeType;
871
872         // Set the physics settings
873         cameraFollowTarget.physicsShapeType =
874             target.physicsShapeType;
875
875
876         // Set the physics settings
877         cameraFollowTarget.physicsShapeType =
878             target.physicsShapeType;
879
879
880         // Set the physics settings
881         cameraFollowTarget.physicsShapeType =
882             target.physicsShapeType;
883
884         // Set the physics settings
885         cameraFollowTarget.physicsShapeType =
886             target.physicsShapeType;
887
887
8
```

CONFIGURING THE GAME

Add this to the RemoveAfterDelay.cs

```
RemoveAfterDelay > M Remove()
1 // Removes an object after a certain delay.
2 using UnityEngine;
3 using System.Collections;
4 public class RemoveAfterDelay : MonoBehaviour
{
    // How many seconds to wait before removing.
8    public float delay = 1.0f;
9    void Start()
10    {
11        // Kick off the 'Remove' coroutine.
12        StartCoroutine("Remove");
13    }
14    IEnumerator Remove()
15    {
16        // Wait 'delay' seconds, and then destroy the
17        // gameobject attached to this object.
18        yield return new WaitForSeconds(delay);
19        Destroy(gameObject);
20        // Don't say Destroy(this) - that just destroys this
21        // RemoveAfterDelay script.
22    }
}
```

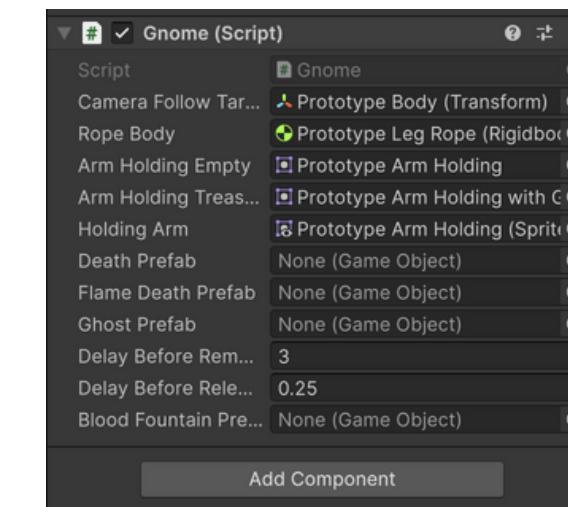
Select all the body parts and put the
BodyPart.cs



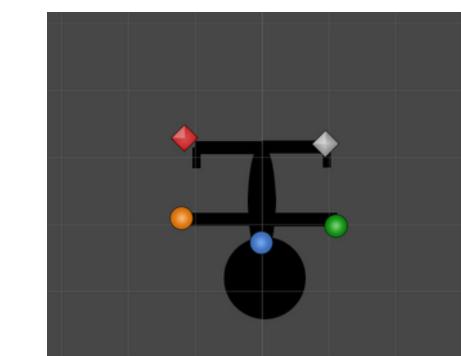
Select Prototype Gnome and put the
Gnome.cs



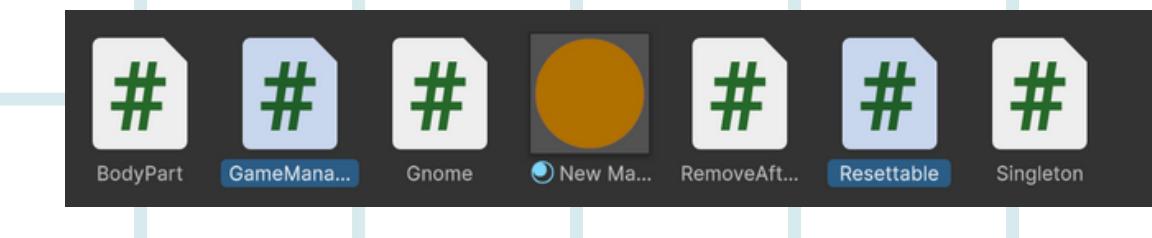
Configure Gnome.cs



Configure the 5 Blood Fountain
Objects and put it in their
respective places



Create 2 More cs names
GameManager and resettable



Put this in the Resettable

```
using UnityEngine;
using UnityEngine.Events;
// Contains a UnityEvent that can be used to reset the state
// of this object.
public class Resettable : MonoBehaviour
{
    // In the editor, connect this event to methods that should
    // run when the game resets.
    public UnityEvent onReset;
    // Called by the GameManager when the game resets.
    public void Reset()
    {
        // Kicks off the event, which calls all of the
        // connected methods.
        onReset.Invoke();
    }
}
```

CONFIGURING THE GAME

Add this code to the game manager

```
delayAfterDeath
-----
public void CreateNewGnome()
{
    // Remove the current gnome, if there is one
    RemoveGnome();
    // Create a new Gnome object, and make it be our
    // currentGnome
    GameObject newGnome =
        (GameObject)Instantiate(gnomePrefab,
        startingPoint.transform.position,
        Quaternion.identity);
    currentGnome = newGnome.GetComponent<Gnome>();
    // Make the rope visible
    rope.gameObject.SetActive(true);
    // Connect the rope's trailing end to whichever
    // rigidbody the Gnome object wants (e.g., his foot)
    rope.connectedObject = currentGnome.ropeBody;
    // Reset the rope's length to the default
    rope.ResetLength();
    // Tell the cameraFollow to start tracking the new
    // Gnome object
    cameraFollow.target = currentGnome.cameraFollowTarget;
}

void RemoveGnome()
{
    // Don't actually do anything if the gnome is invincible
    if (gnomeInvincible)
        return;
    // Hide the rope
    rope.gameObject.SetActive(false);
    // Stop tracking the gnome
    cameraFollow.target = null;
    // If we have a current gnome, make that no longer be
    // the player
    if (currentGnome != null)
    {
        // This gnome is no longer holding the treasure
        currentGnome.holdingTreasure = false;
        // Mark this object as not the player (so that
        // colliders won't report when the object
        // hits them)
        currentGnome.gameObject.tag = "Untagged";
        // Find everything that's currently tagged
        // "Player" and remove that tag
        foreach (Transform child in
            currentGnome.transform)
        {
            child.gameObject.tag = "Untagged";
        }
        // Mark ourselves as not currently having a
        // gnome
        currentGnome = null;
    }
}

// Kills the gnome.
void KillGnome(Gnome.DamageType damageType)
{
    // If we have an audio source, play "gnome died"
    // sound
    var audio = GetComponent<AudioSource>();
    if (audio)
    {
        audio.PlayOneShot(this.gnomeDiedSound);
    }
    // Show the damage effect
}
```

```
CreateNewGnome()
-----
if (gnomeInvincible == false)
{
    // Tell the gnome that it died
    currentGnome.DestroyGnome(damageType);
    // Remove the Gnome
    RemoveGnome();
    // Reset the game
    StartCoroutine(ResetAfterDelay());
}

// Called when gnome dies.
IEnumerator ResetAfterDelay()
{
    // Wait for delayAfterDeath seconds, then call Reset
    yield return new WaitForSeconds(delayAfterDeath);
    Reset();
}

// Called when the player touches a trap
public void TrapTouched()
{
    KillGnome(Gnome.DamageType.Slicing);
}

// Called when the player touches a fire trap
public void FireTrapTouched()
{
    KillGnome(Gnome.DamageType.Burning);
}

// Called when the gnome picks up the treasure.
public void TreasureCollected()
{
    // Tell the currentGnome that it should have the
    // treasure.
    currentGnome.holdingTreasure = true;
}

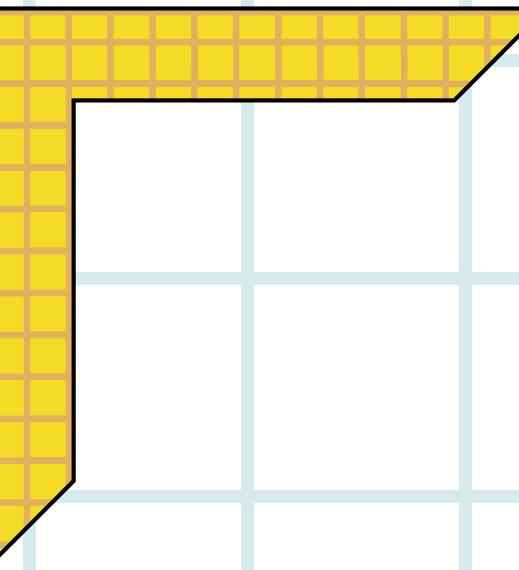
// Called when the player touches the exit.
public void ExitReached()
{
    // If we have a player, and that player is holding
    // treasure, game over!
    if (currentGnome != null &&
        currentGnome.holdingTreasure == true)
    {
        // If we have an audio source, play the "game
        // over" sound
        var audio = GetComponent< AudioSource >();
        if (audio)
        {
            audio.PlayOneShot(this.gameOverSound);
        }
        // Pause the game
        Time.timeScale = 0.0f;
        // Turn off the Game Over menu, and turn on the
        // "game over" screen!
        if (gameOverMenu)
        {
            gameOverMenu.gameObject.SetActive(true);
        }
        if (gameplayMenu)
        {
            gameplayMenu.gameObject.SetActive(false);
        }
    }
}

// Called when the Menu button is tapped, and when the
// Resume Game button is tapped.
public void SetPaused(bool paused)
{
    // If we're paused, stop time and enable the menu (and
    // disable the game overlay)
    if (paused)
    {
        Time.timeScale = 0.0f;
        mainMenu.gameObject.SetActive(true);
        gameplayMenu.gameObject.SetActive(false);
    }
    else
    {
        // If we're not paused, resume time and disable
        // the menu (and enable the game overlay)
        Time.timeScale = 1.0f;
        mainMenu.gameObject.SetActive(false);
        gameplayMenu.gameObject.SetActive(true);
    }
}

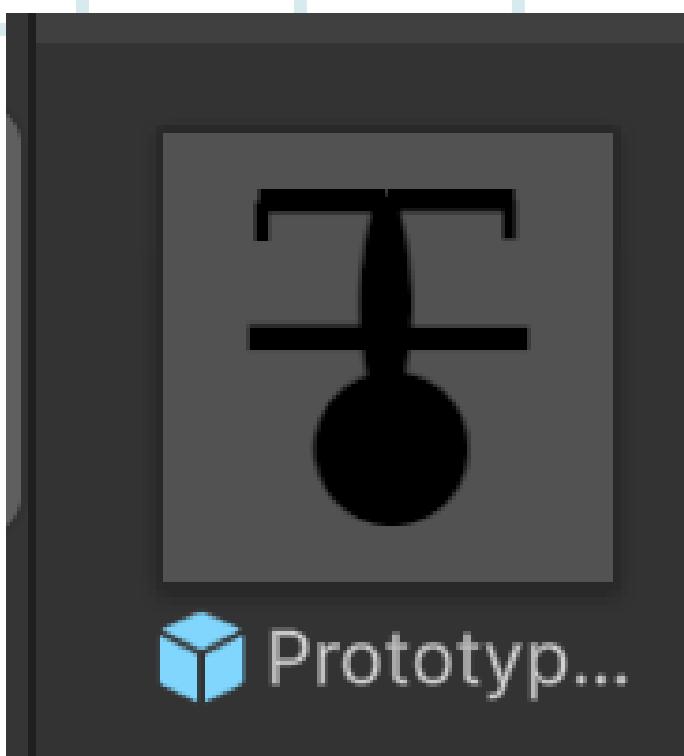
// Called when the Restart button is tapped.
public void RestartGame()
{
    // Immediately remove the gnome (instead of killing it)
    Destroy(currentGnome.gameObject);
    currentGnome = null;
    // Now reset the game to create a new gnome.
    Reset();
}
```

```
Reset()
-----
if (gameOver)
{
    // If we're not invincible, reset the game and make
    // the gnome not be the current player.
    if (gnomeInvincible == false)
    {
        // Tell the gnome that it died
        currentGnome.DestroyGnome(damageType);
        // Remove the Gnome
        RemoveGnome();
        // Reset the game
        StartCoroutine(ResetAfterDelay());
    }
}
else
{
    // If we're not paused, resume time and disable
    // the menu (and enable the game overlay)
    Time.timeScale = 1.0f;
    mainMenu.gameObject.SetActive(true);
    gameplayMenu.gameObject.SetActive(false);
}
```

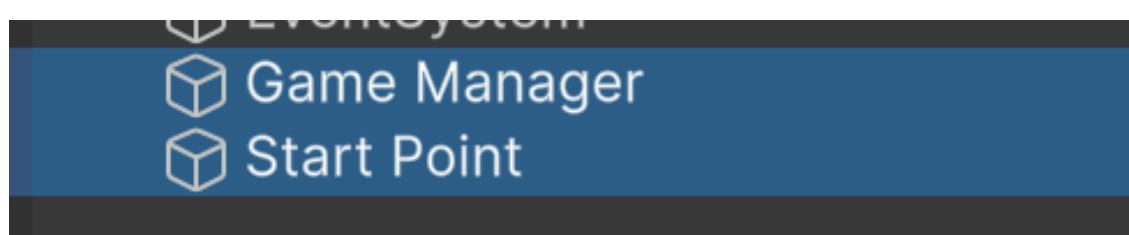
UPDATE THE GNOME



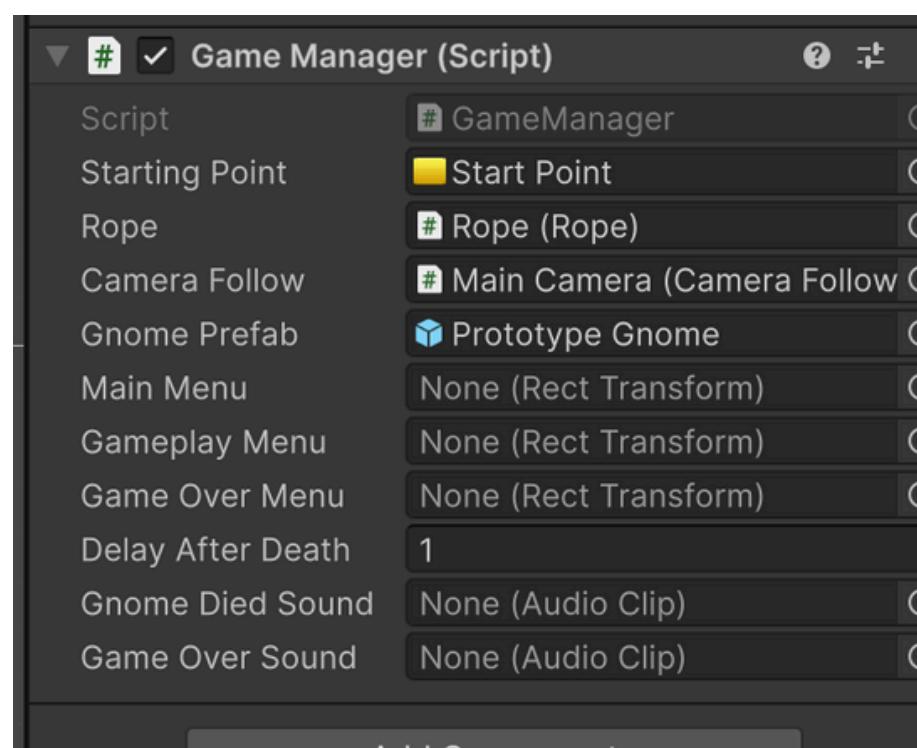
make the Prototype Gnome in a prefab
by dragging it into the Gnome asset
folder



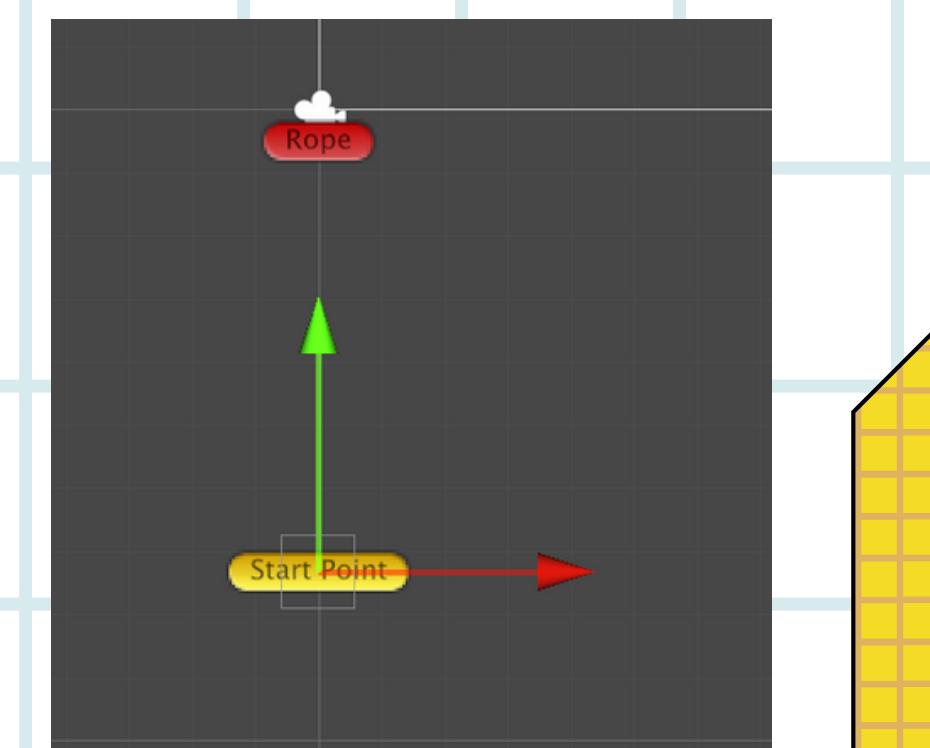
Create 2 game object named Game
manager and Start Point



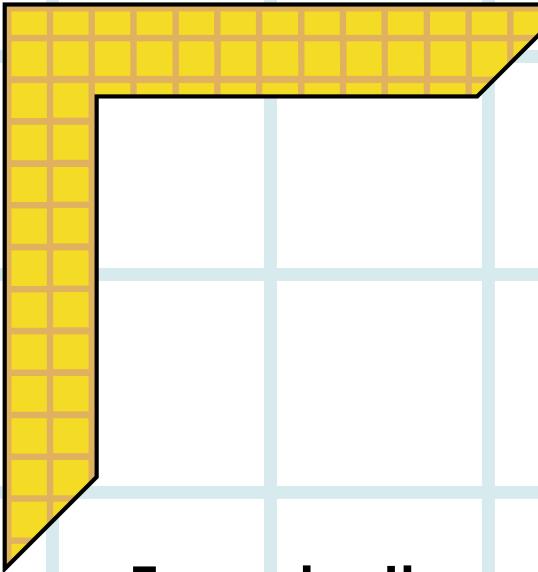
Put the Gamemanager.cs into the game
manager object and set this setting



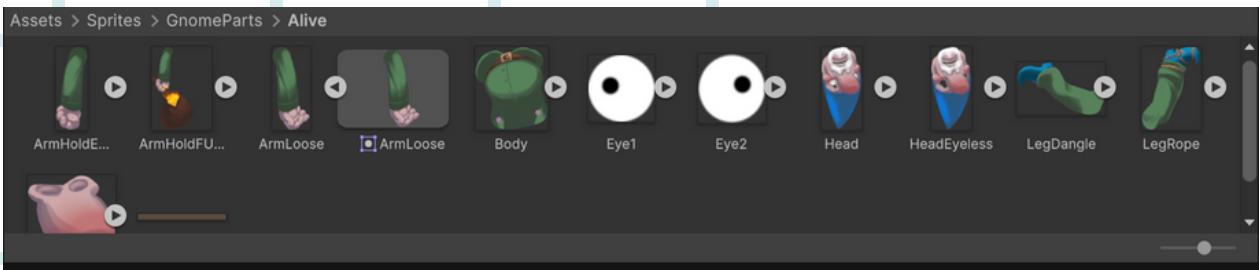
Make Sure to put the Start point object
in the gnome itself



UPDATE THE GNOME



To make the gnome more polished lets add the sprites



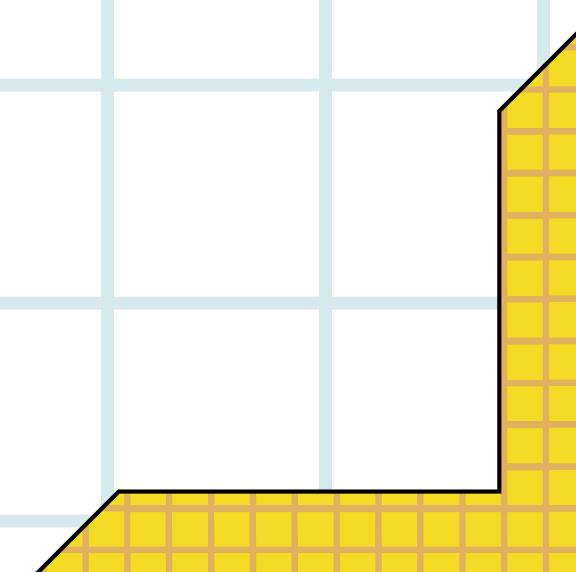
And replace the prefab with the actual sprites



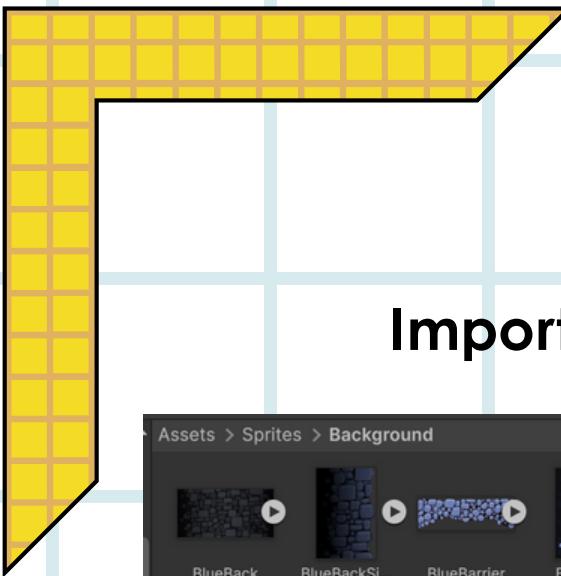
Add a collider object to each part except the body

- ▼ Prototype Arm Holding
 Collider
- ▼ Prototype Arm Loose
 Collider
- Prototype Body
- ▼ Prototype Head
 Collider
- ▼ Prototype Leg Dangle
 Collider
- ▼ Prototype Leg Rope
 Collider

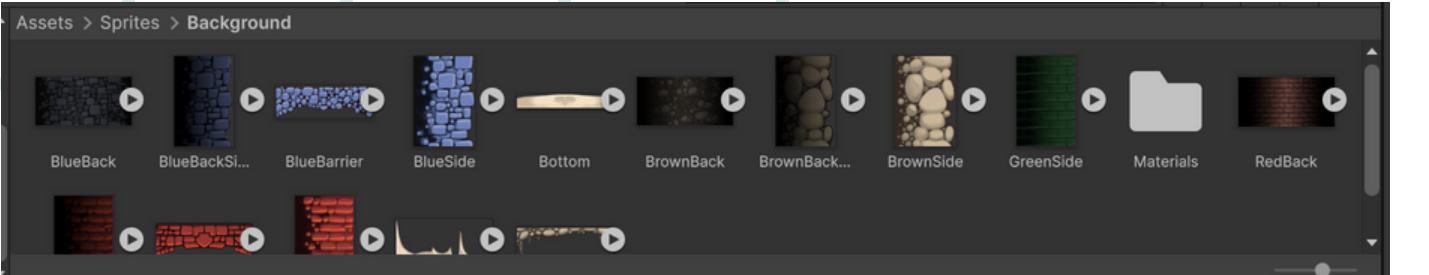
and inside the collider add. polygoncollider and make it like this



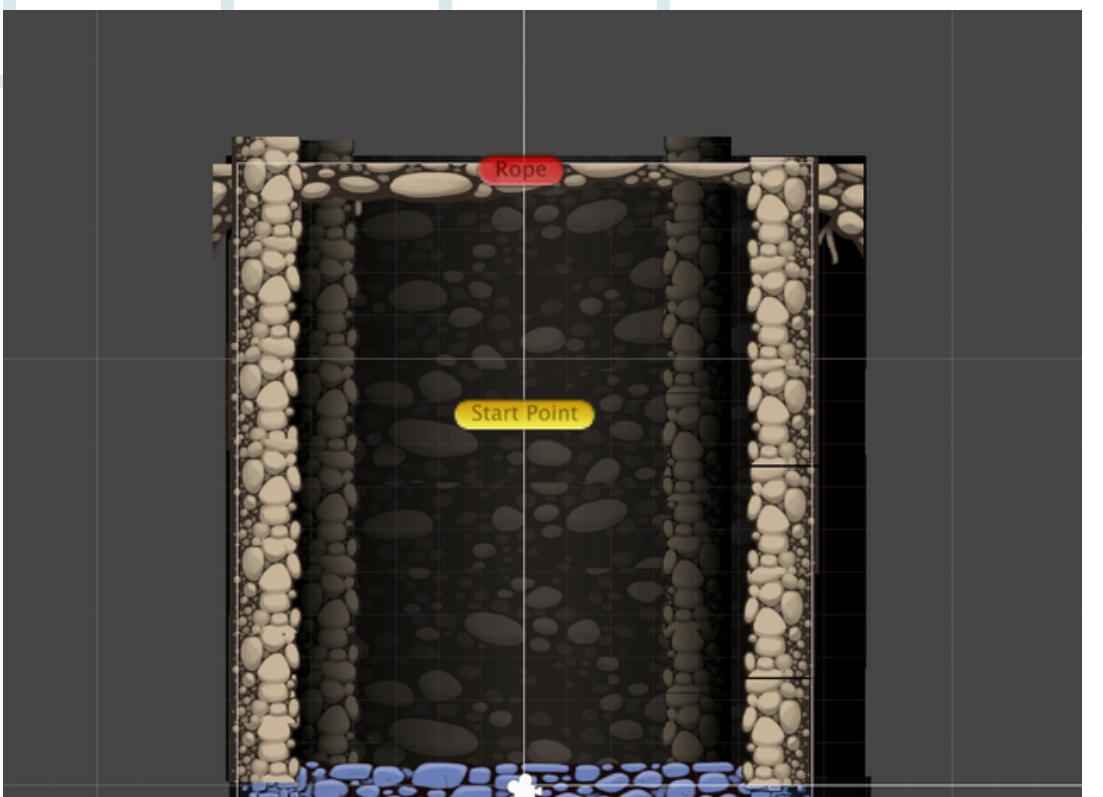
MAKE THE LEVEL



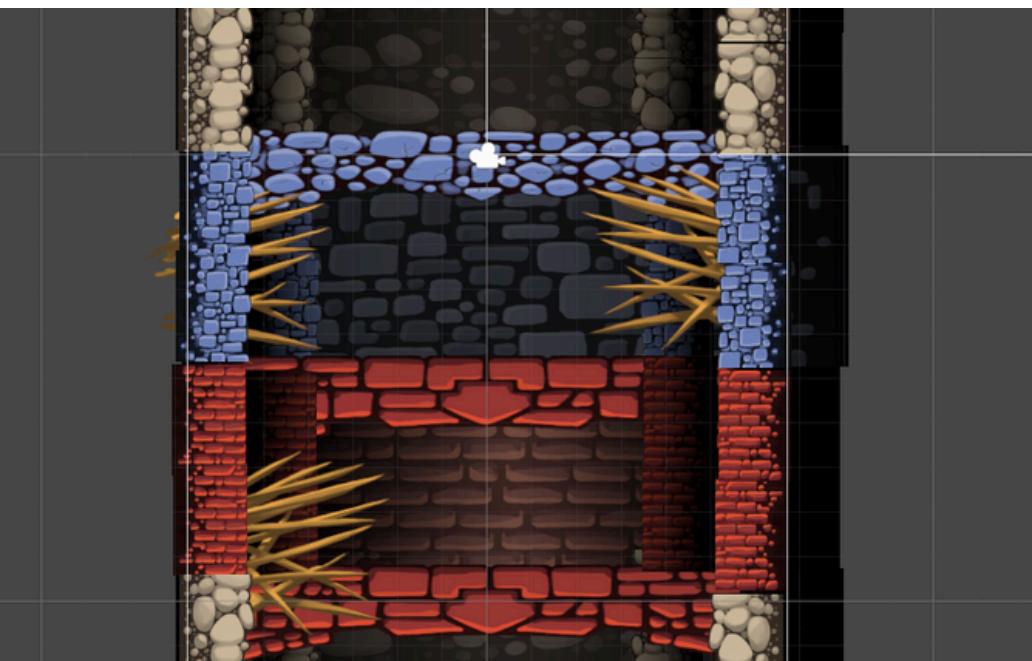
Import the background assets



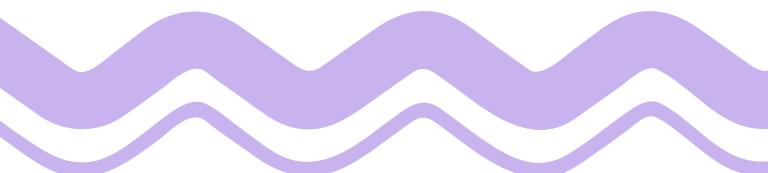
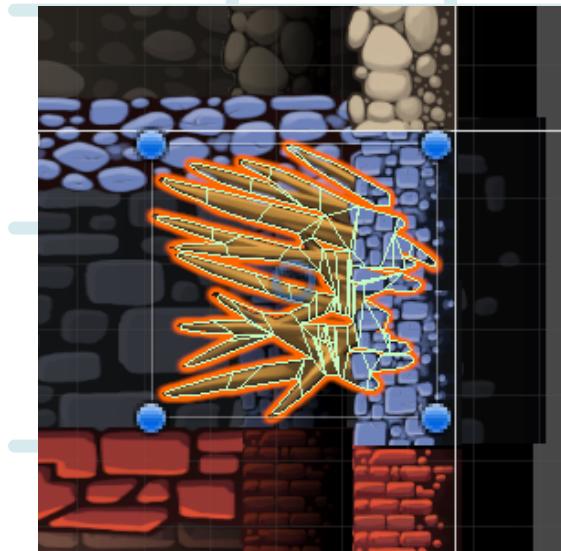
and arrange them like this



put spikes too for the death

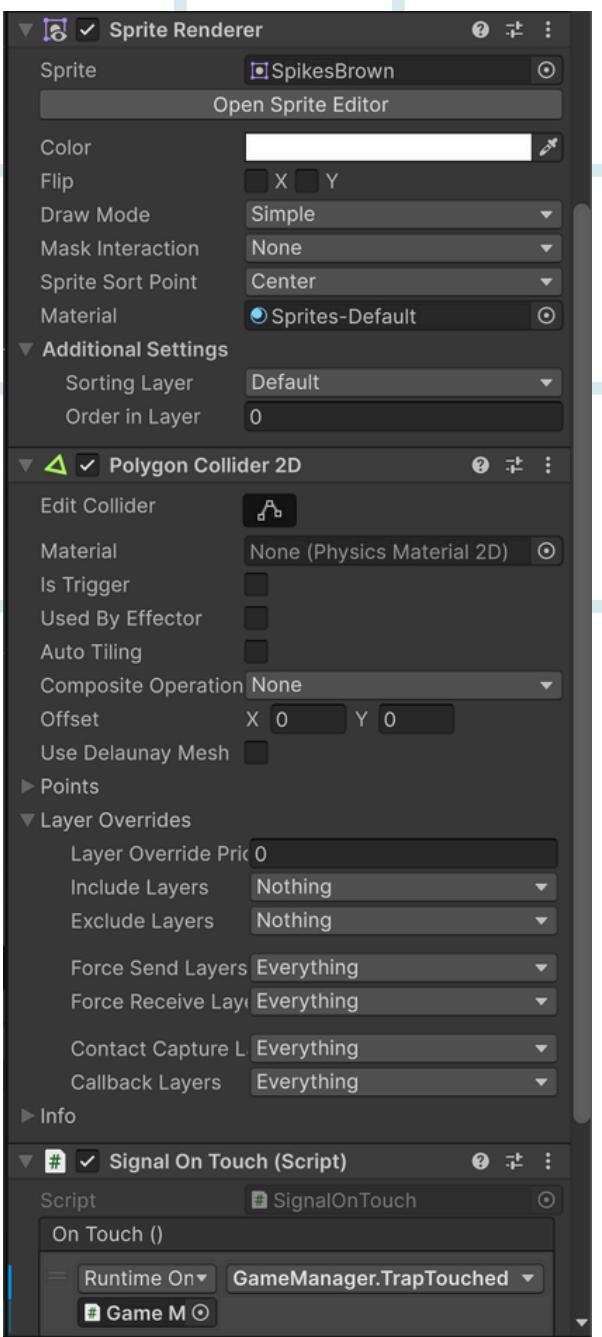


To configure it



MAKE THE LEVEL

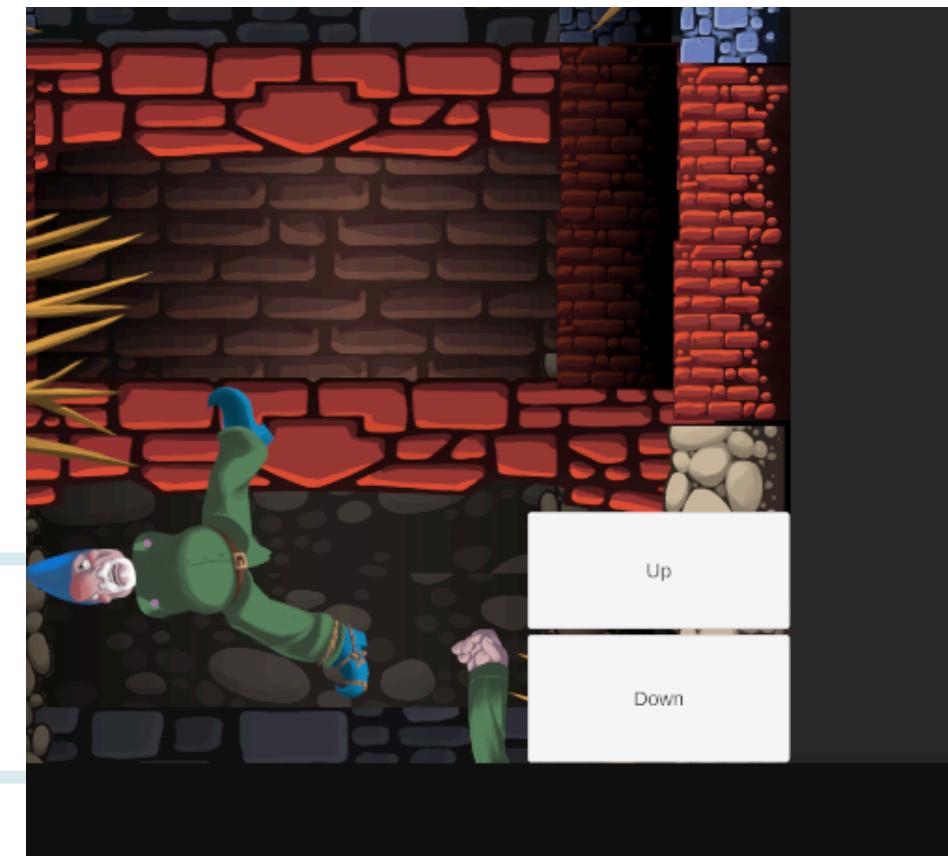
Add these components to the spike



Create a new script called
SignalOntouch.cs

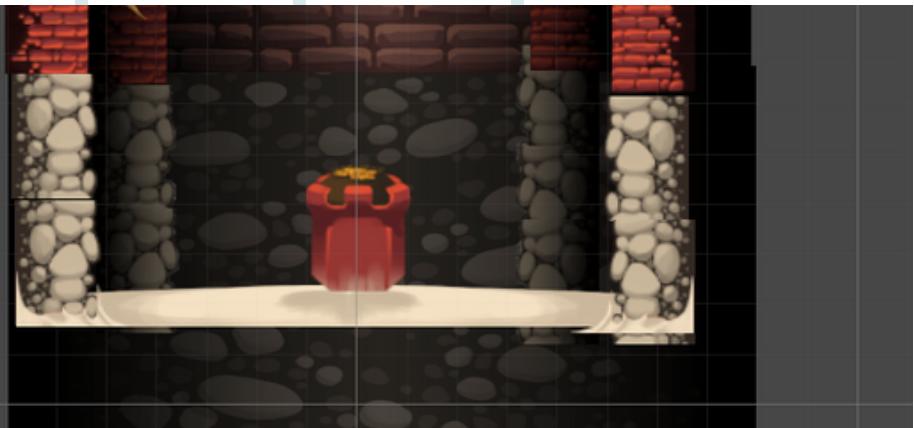
```
1  using UnityEngine;
2  using UnityEngine.Events;
3  // Invokes a UnityEvent when the Player collides with this
4  // object.
5  [RequireComponent(typeof(Collider2D))]
6  public class SignalOnTouch : MonoBehaviour
7  {
8
9
10 // The UnityEvent to run when we collide.
11 // Attach methods to run in the editor.
12 public UnityEvent onTouch;
13
14 // If true, attempt to play an AudioSource when we collide.
15 public bool playAudioOnTouch = true;
16 // When we enter a trigger area, call SendSignal.
17 void OnTriggerEnter2D(Collider2D collider)
18 {
19     SendSignal(collider.gameObject);
20 }
21
22 // When we collide with this object, call SendSignal.
23 void OnCollisionEnter2D(Collision2D collision)
24 {
25     SendSignal(collision.gameObject);
26 }
27
28 // Checks to see if this object was tagged as Player, and
29 // invoke the UnityEvent if it was.
30 void SendSignal(GameObject objectThatHit)
31 {
32     // Was this object tagged Player?
33     if (objectThatHit.CompareTag("Player"))
34     {
35         // If we should play a sound, attempt to play it
36         if (playAudioOnTouch)
37         {
38             var audio = GetComponent< AudioSource >();
39             // If we have an audio component,
40             // and this component's parents|
41             // are active, then play
42             if (audio &&
43                 audio.gameObject.activeInHierarchy)
44                 audio.Play();
45         }
46     }
47 }
```

When the gnome touches the spike it will
die and fall off

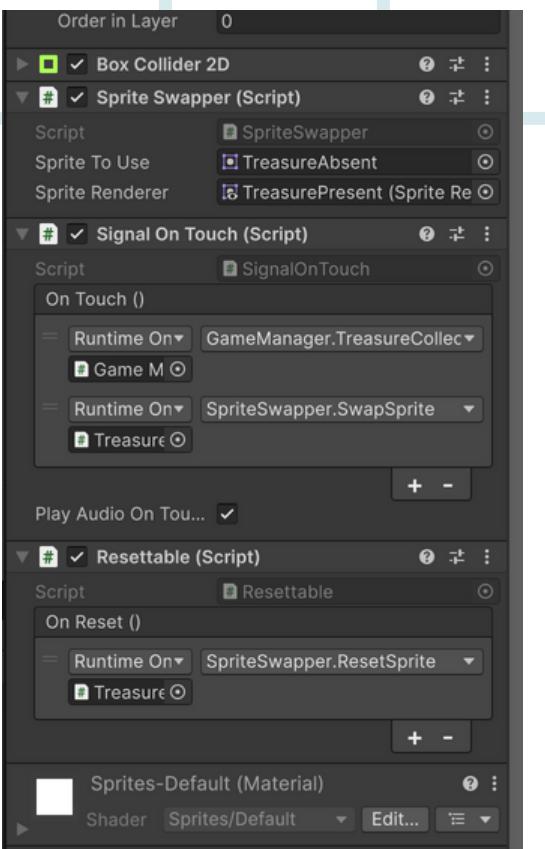


CREATING BUTTONS

Lets now create the Treasure



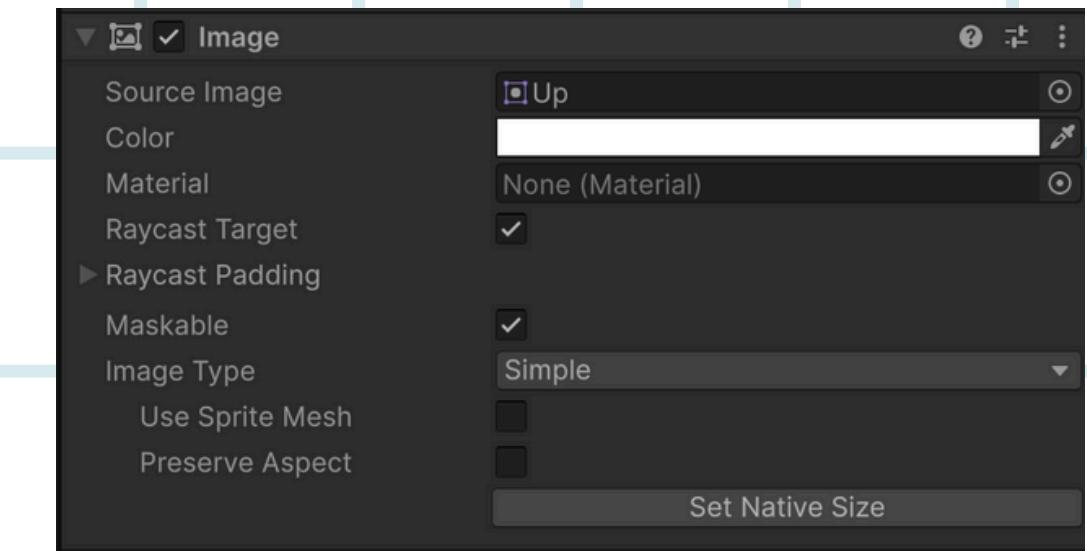
Add these components



Lets now update the ui and add other functions of ui



Click the 2 up and down button and remove the text and add this component and add the image from the asset folder

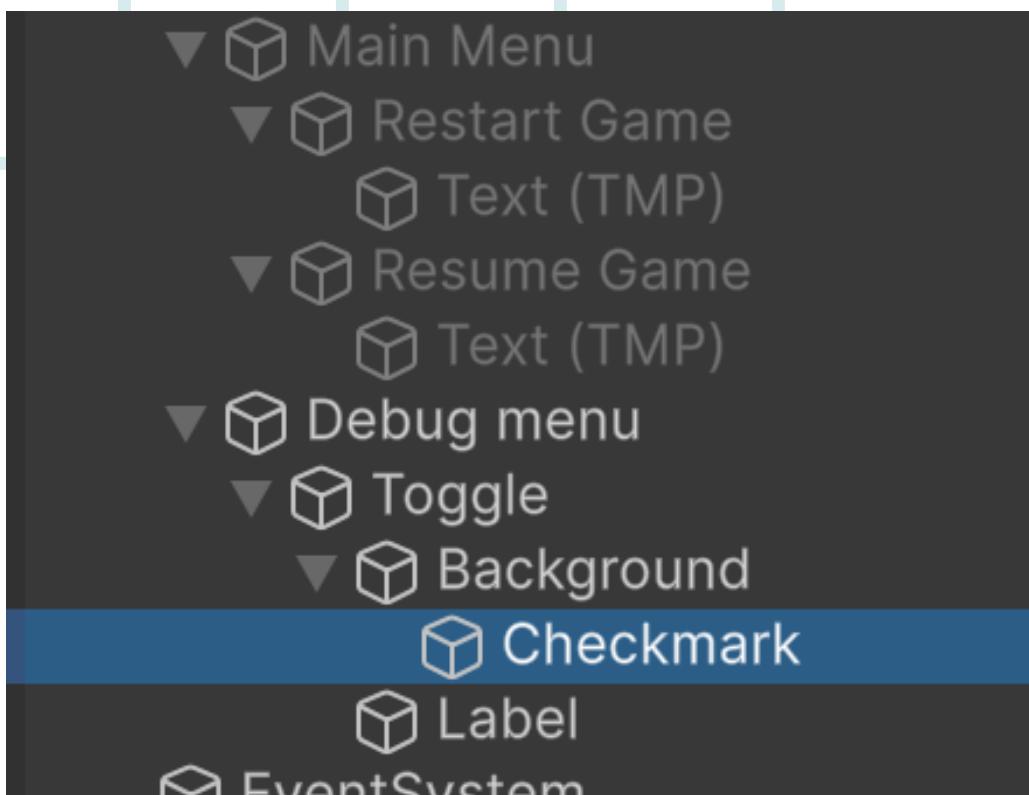


Do the same with the down button

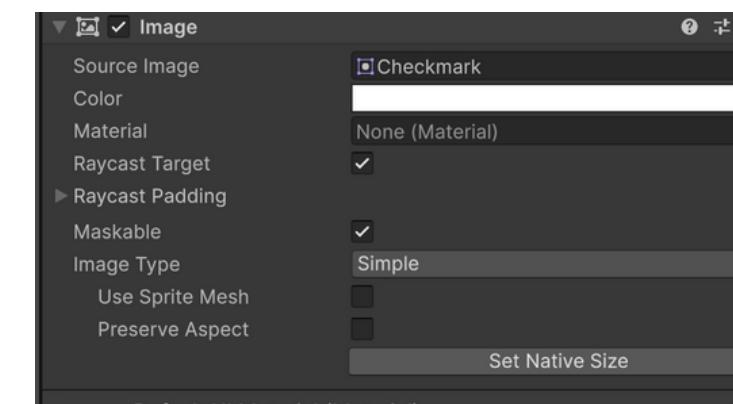
CREATE MECHANICS

Lets now make the
Menu and invincible
button

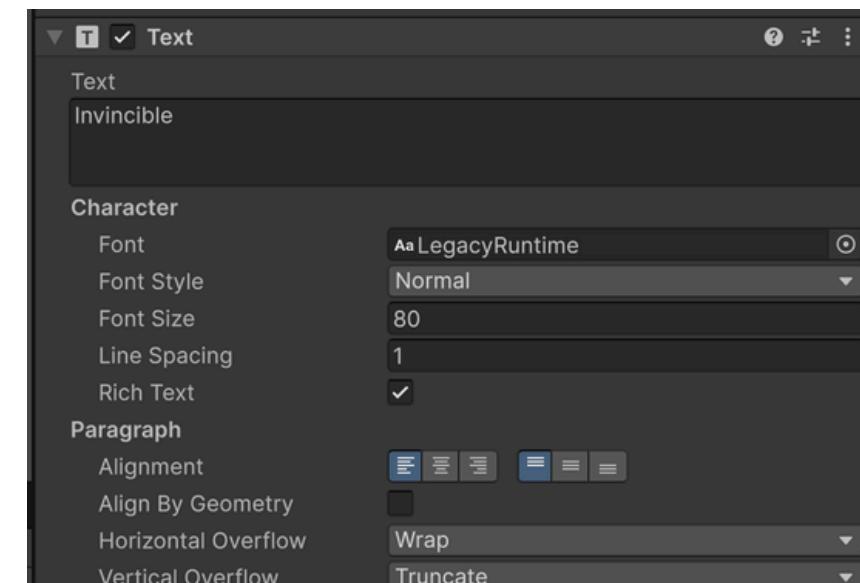
Make this objects with parent
and child objects like this



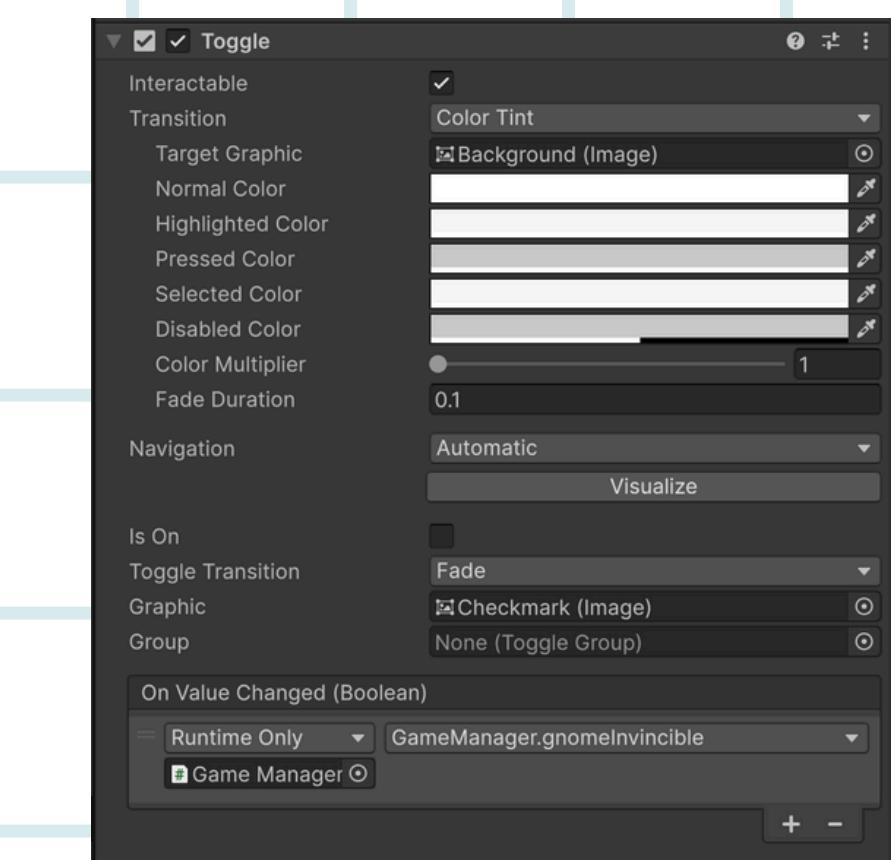
Add these on the checkmark
object



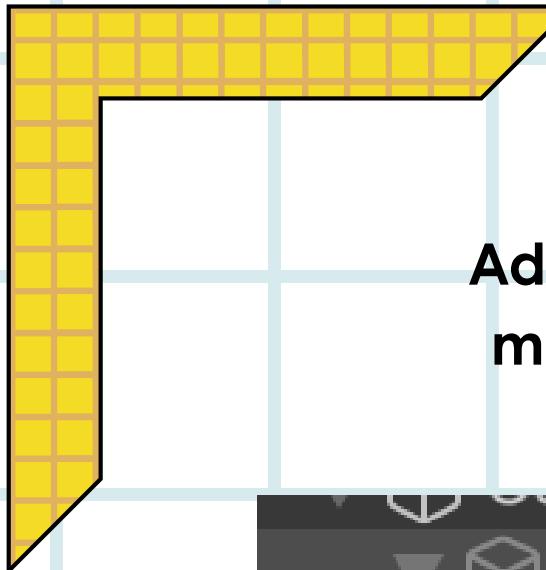
And modify level to this



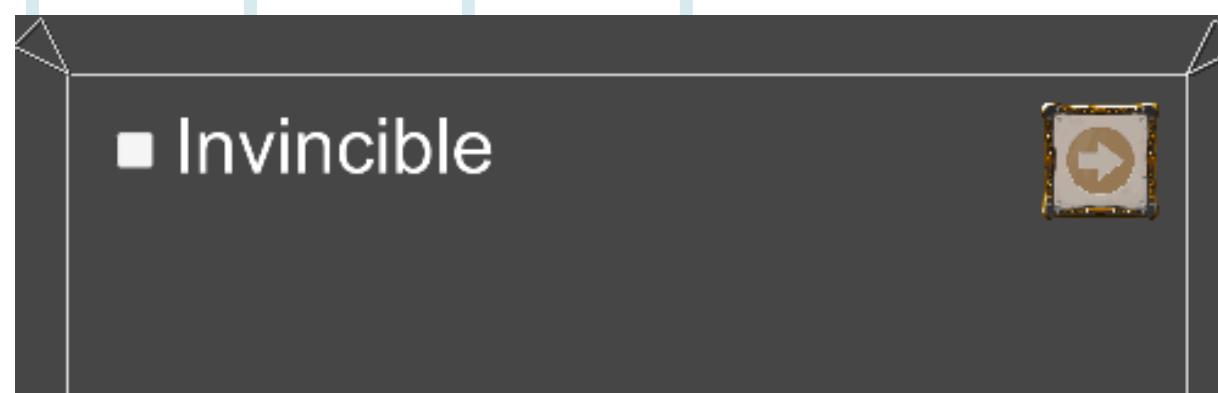
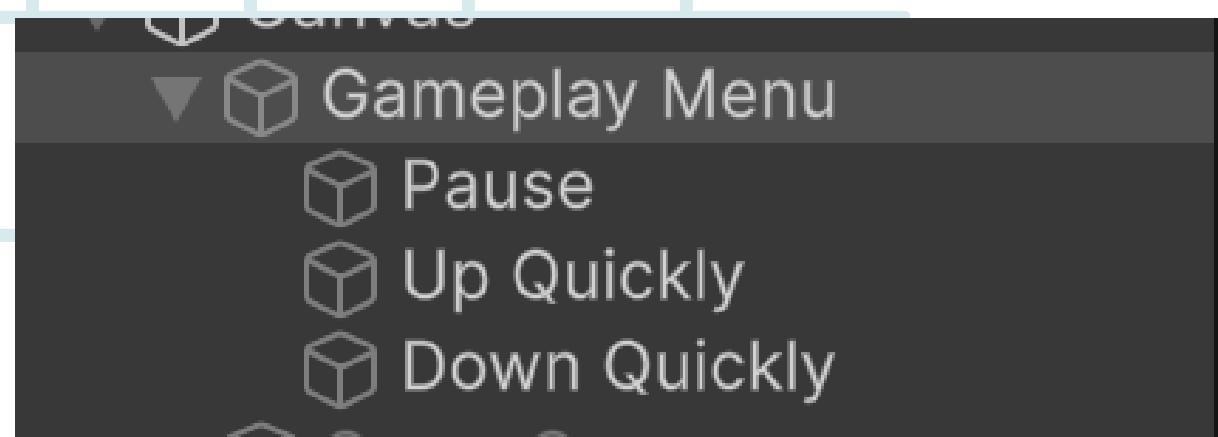
And put these on the toggle
object



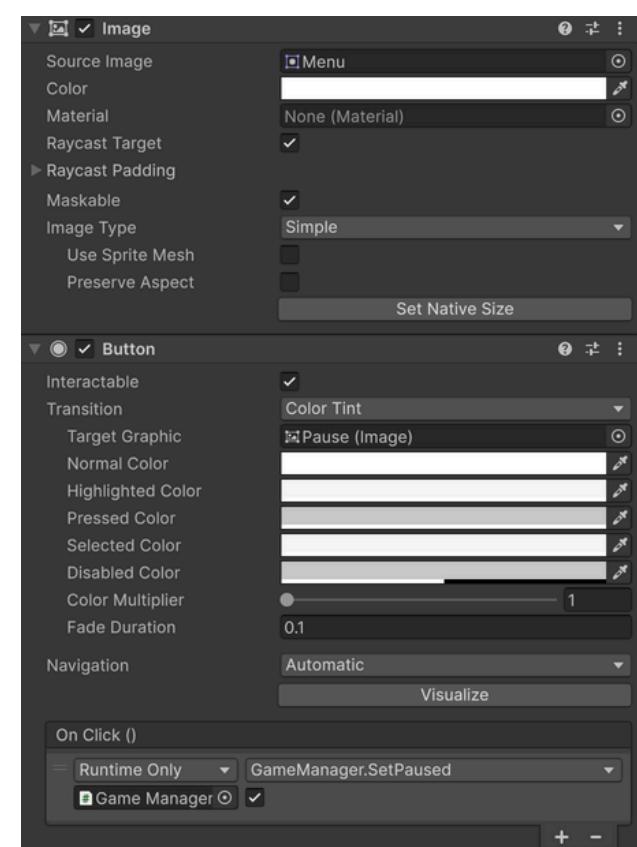
CREATE MECHANICS



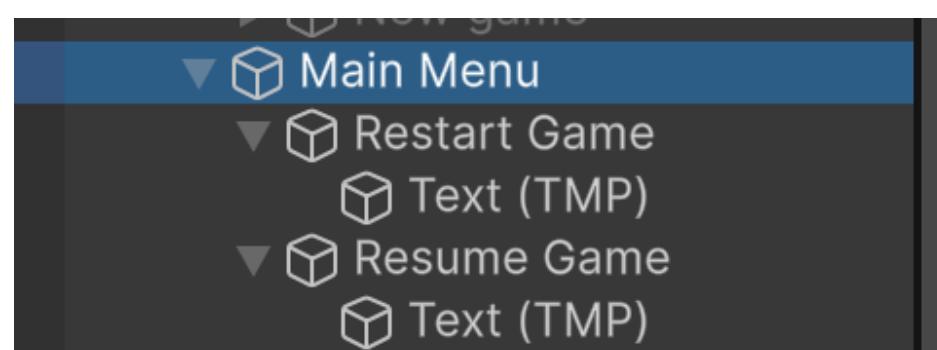
Add pause to the gameplay menu and put it on the top right



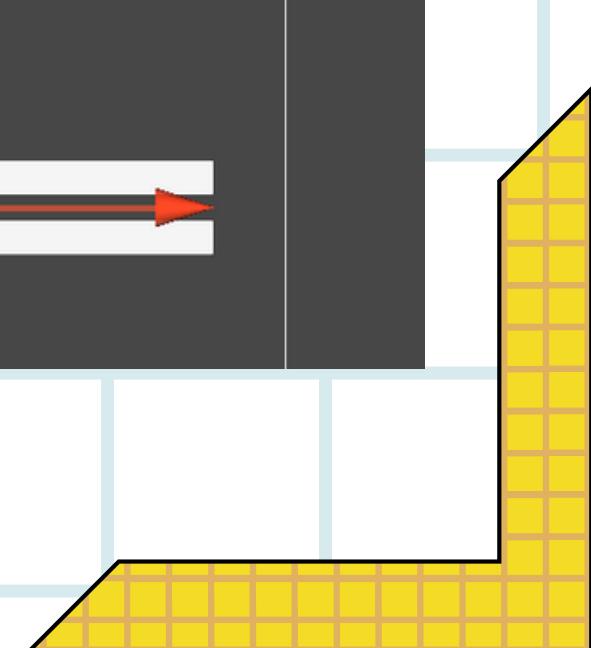
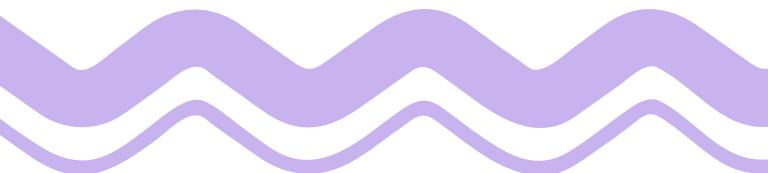
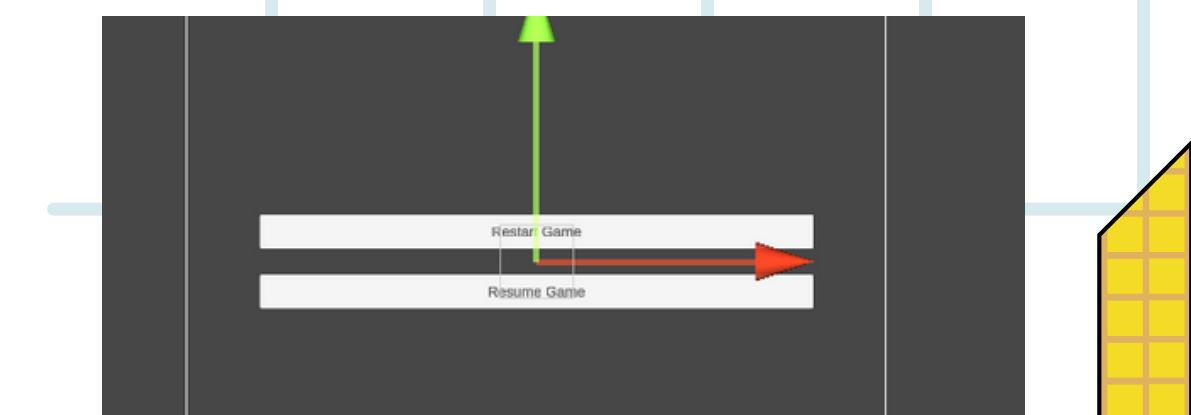
Add these components to the object pause



make another objects called main menu and add these objects under it

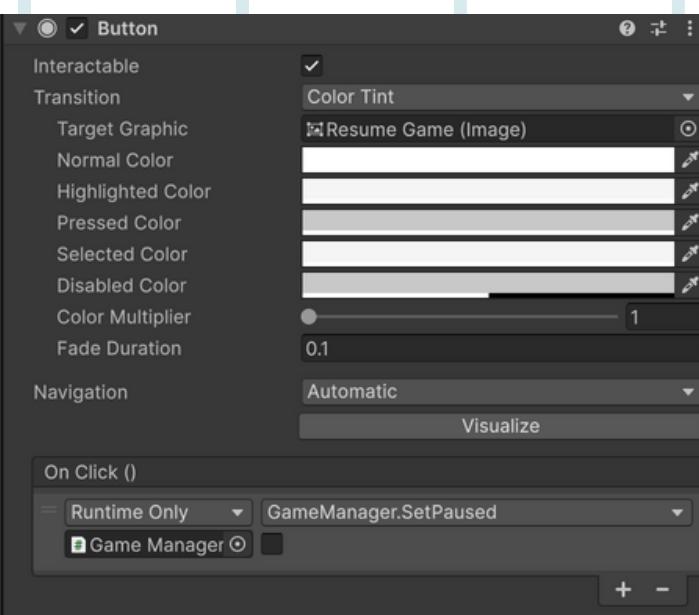
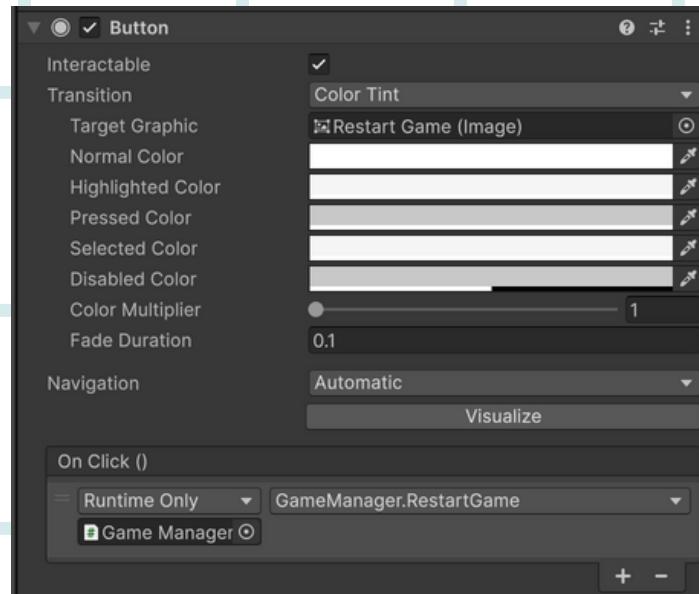


and Make them Look like this

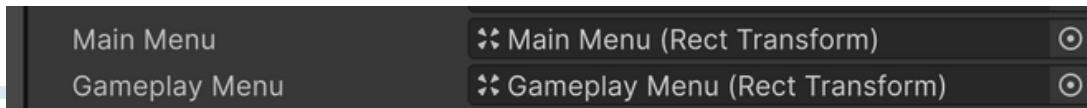


CREATE GAMEOVER

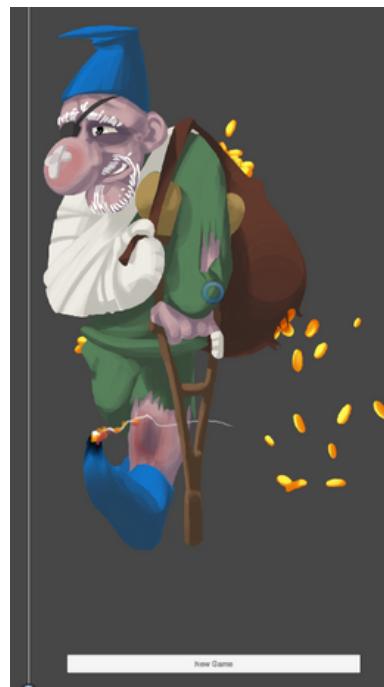
Add these components respectively



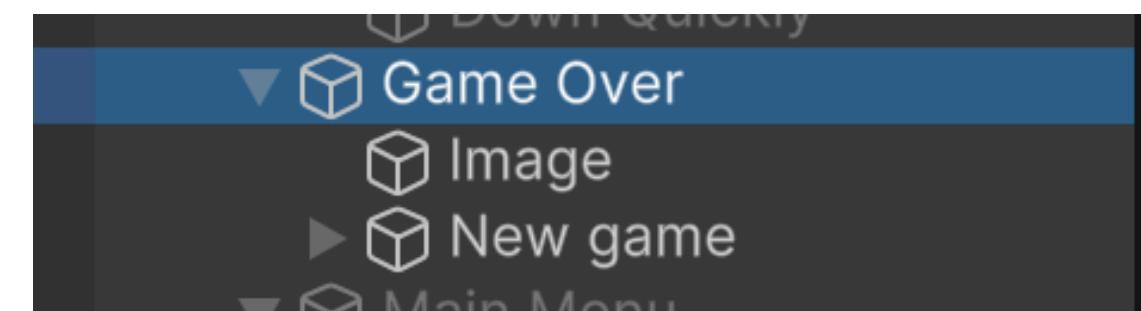
And set the game manager to main menu and gameplay menu



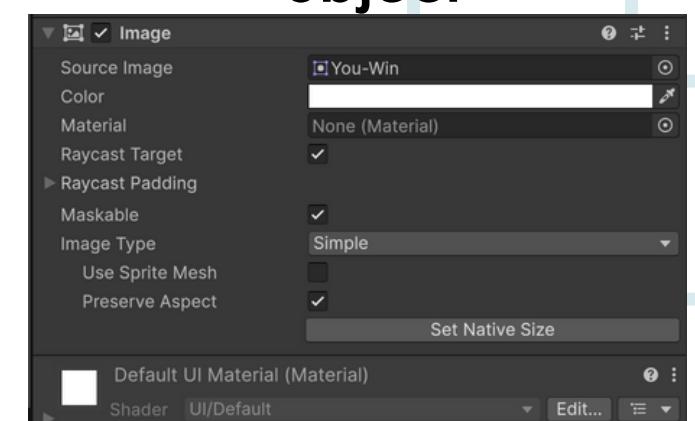
Lets create the game over menu



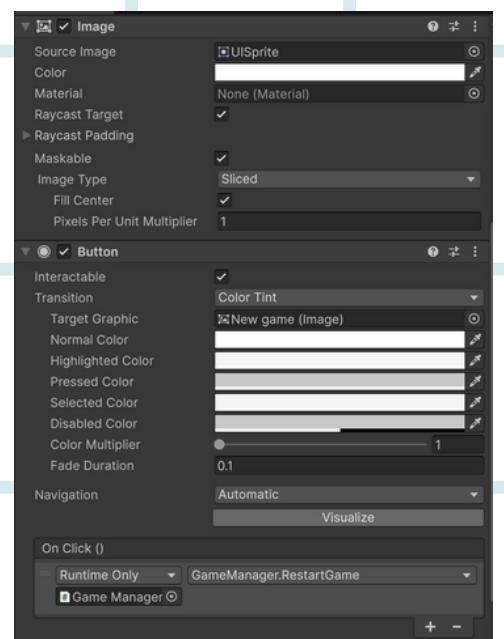
Create these 3 objects



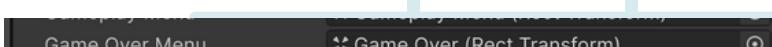
Put these on the image object



Put these on the new game object



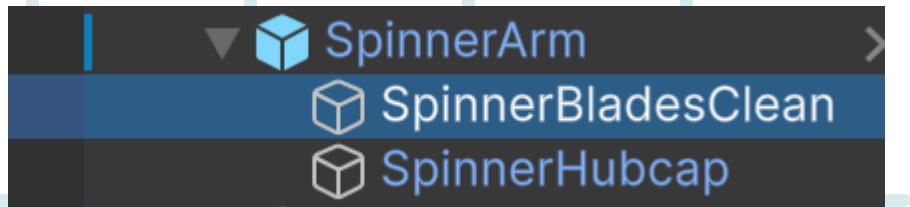
and configure game manager



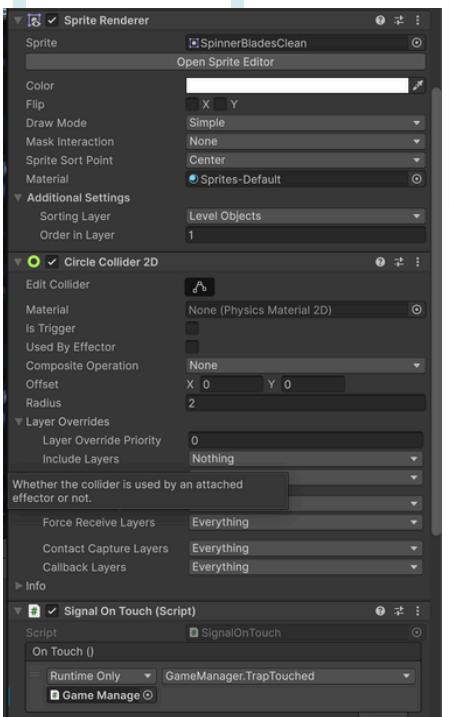
ADD MORE TRAPS

Lets add another trap called spinner

Make these objects

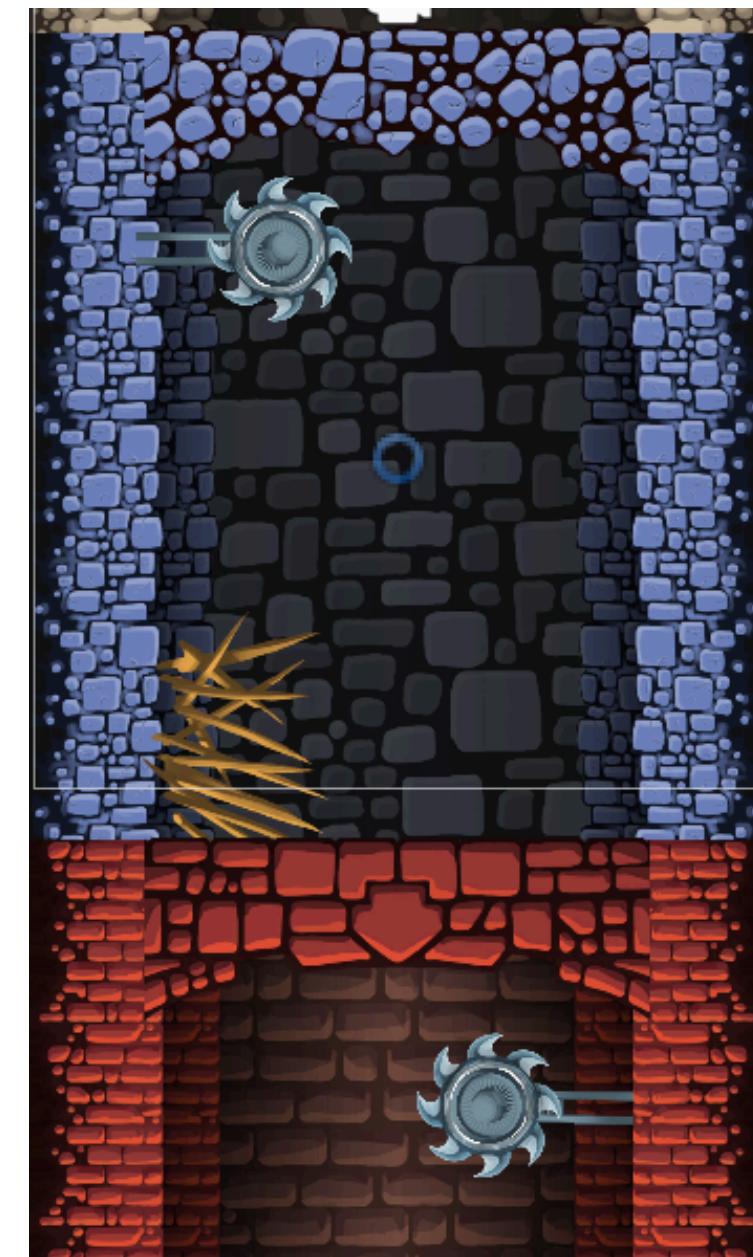
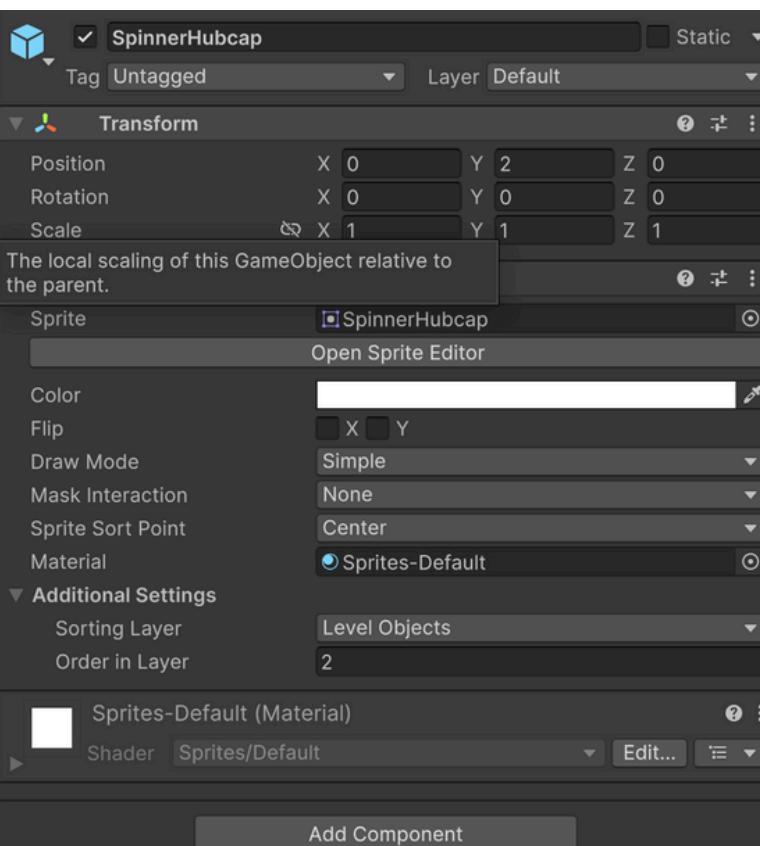


Add this components to spinnerbladesclean



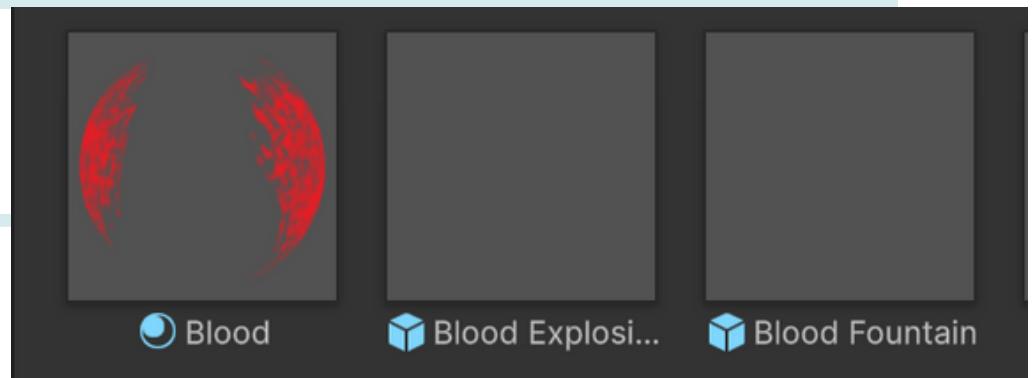
copy this setting to the spinner hub cap

And put it anywhere on the map as you like

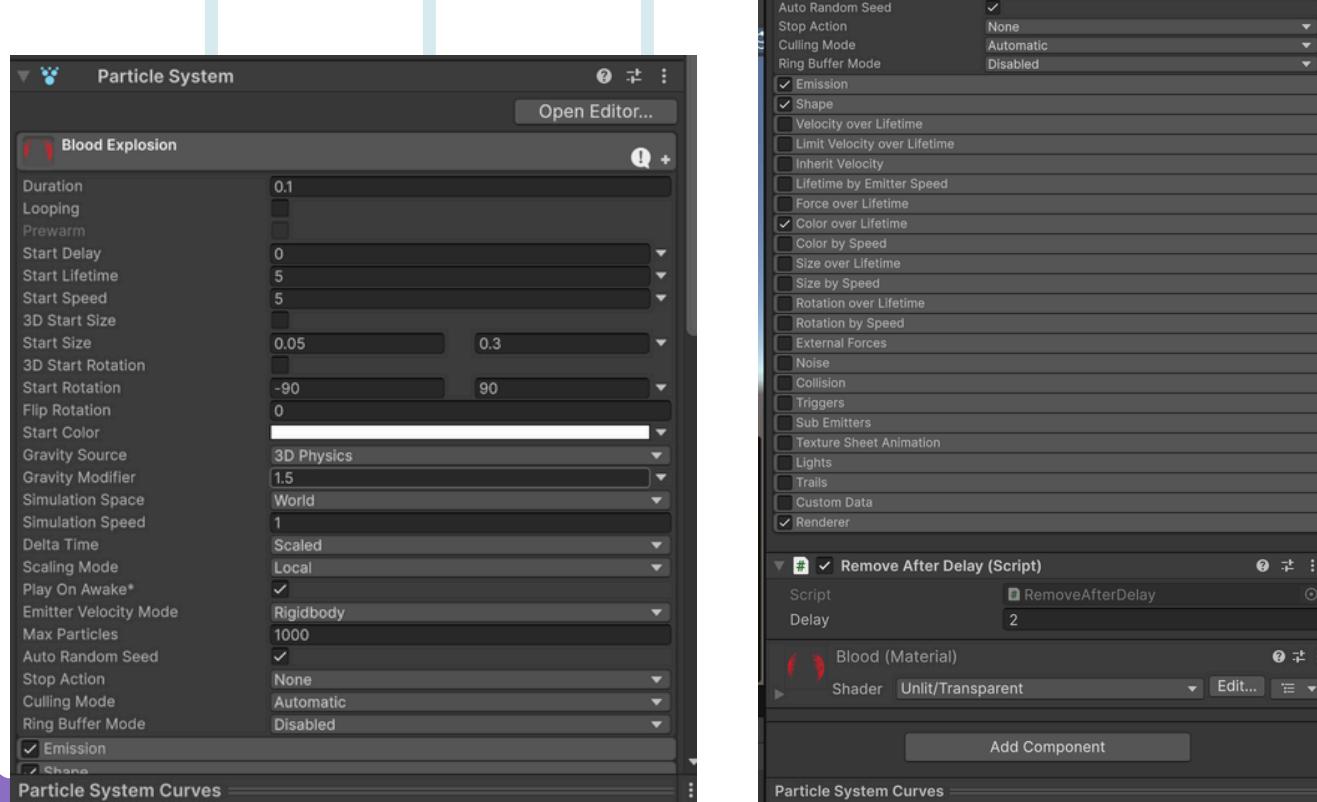


CREATING BUTTONS

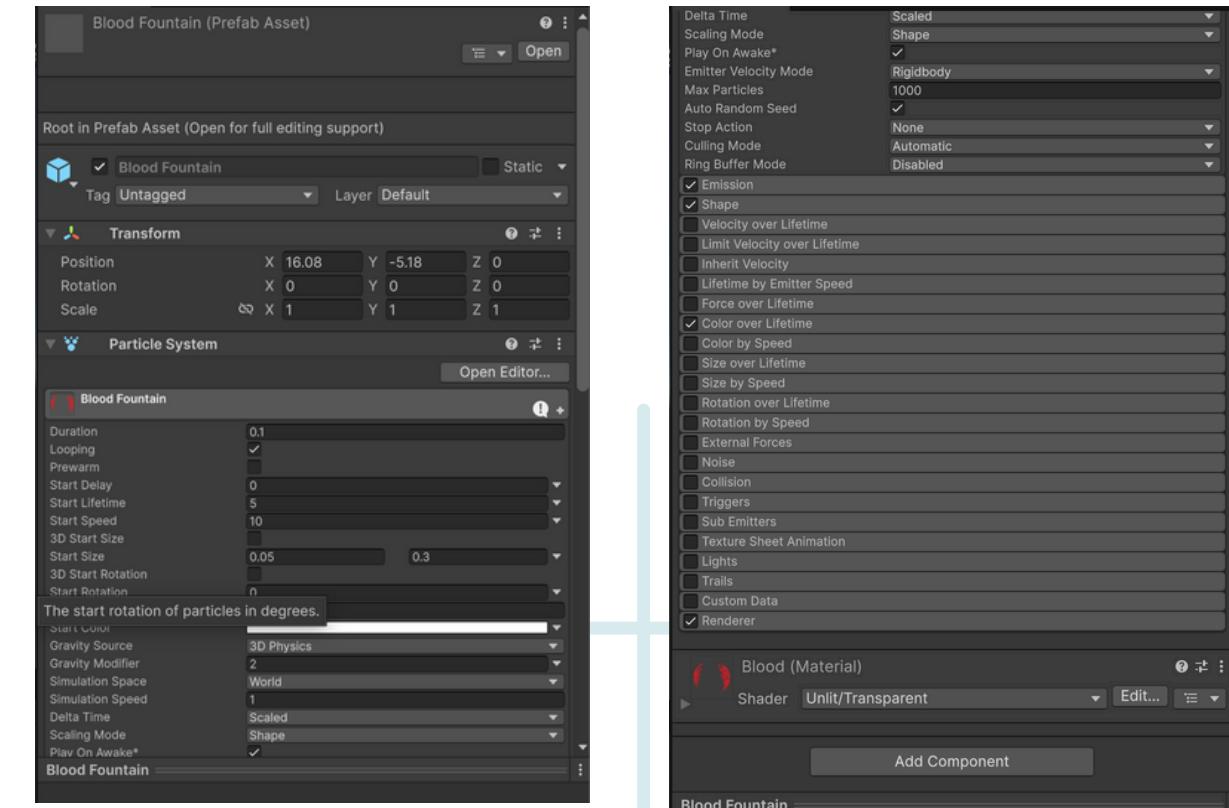
Create this prefabs called
blood explosion and bloof
fountain into a particle system



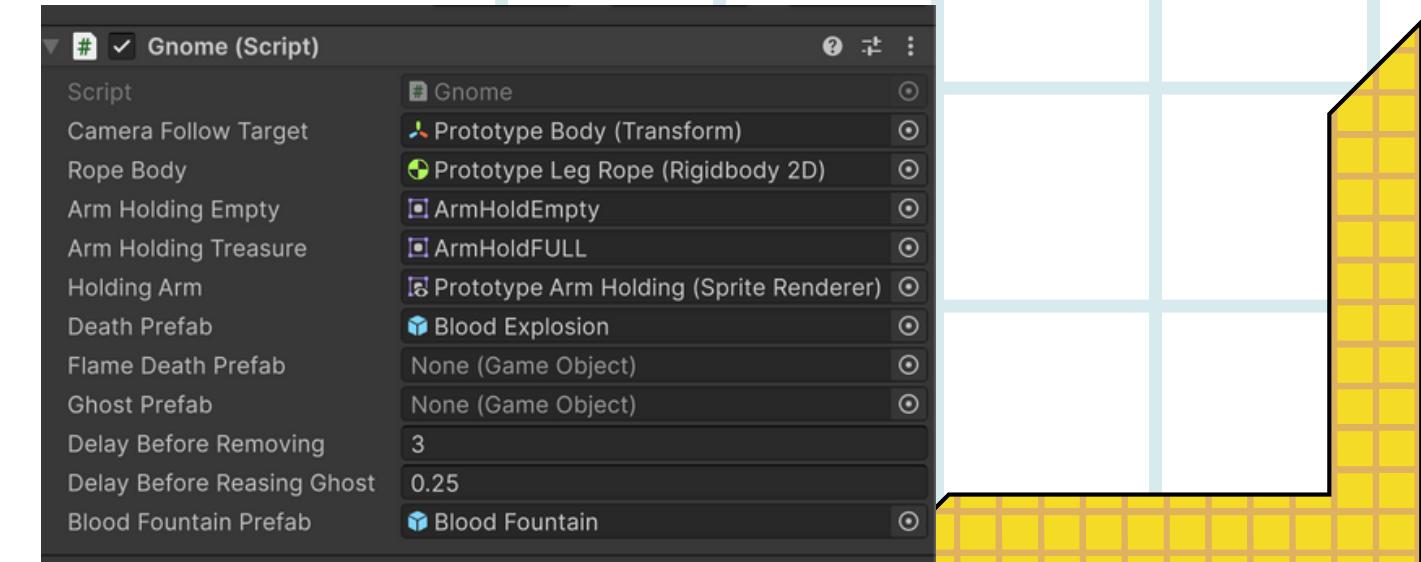
Copy these settings to blood explosion



Copy these settings to blood fountain

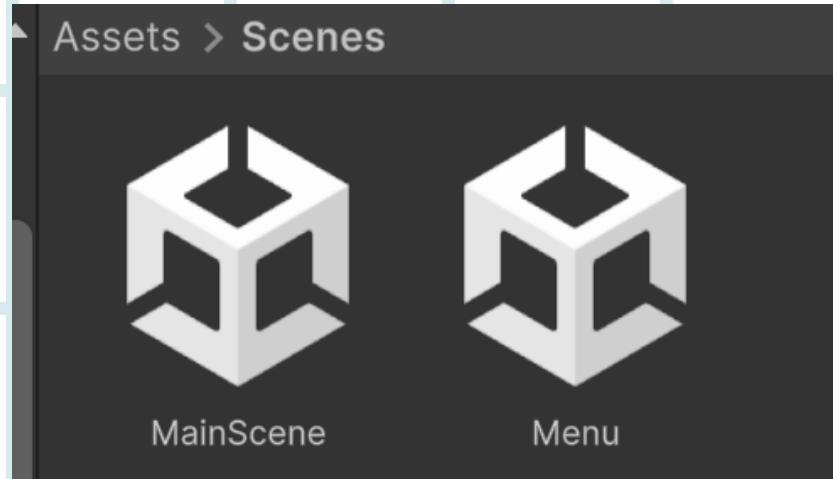


Update the gnome script in th Gnome prefab

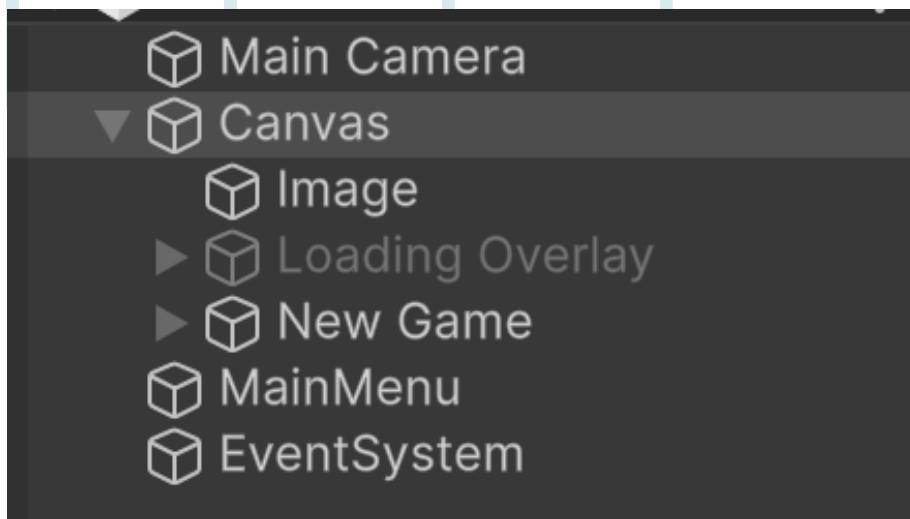


CREATE THE MENU

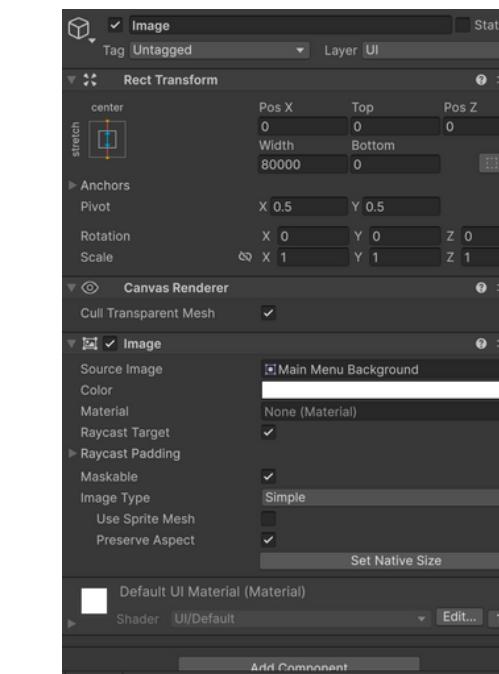
Create a new scene Called
Menu



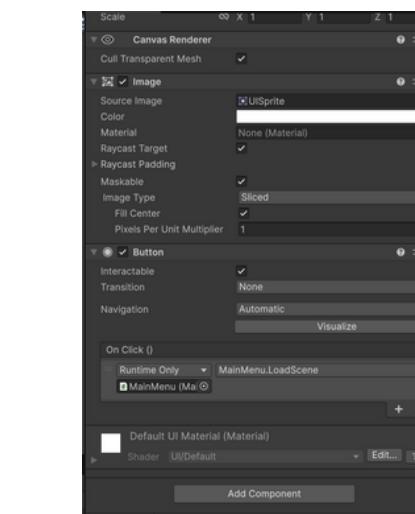
And Make this objects



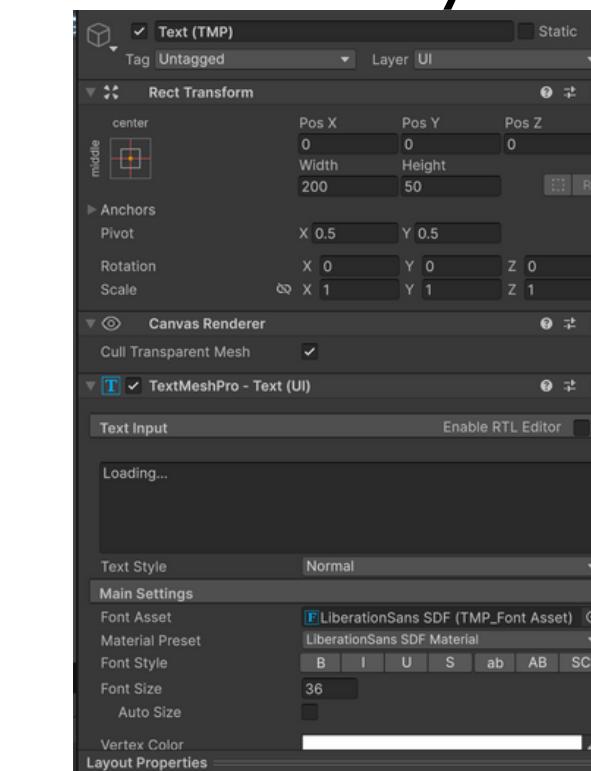
Make these image settings



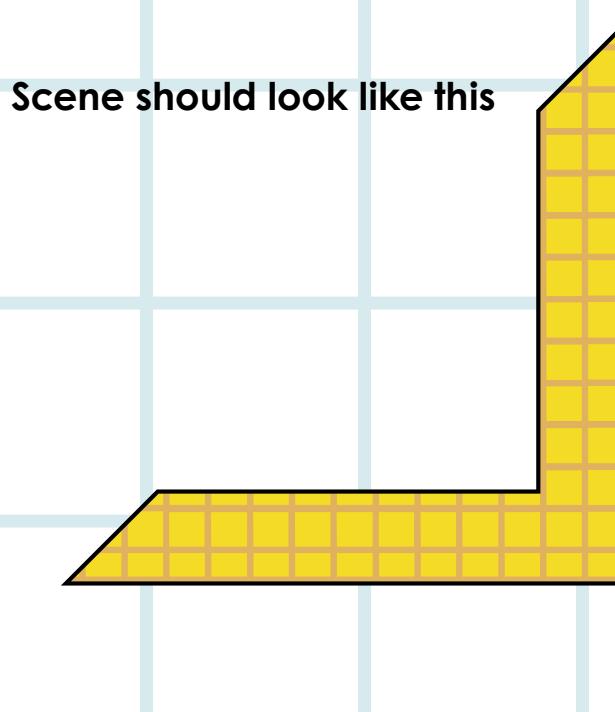
Like in the pause game add a
new game button and add
these buttons



make these in the loading
overlay text

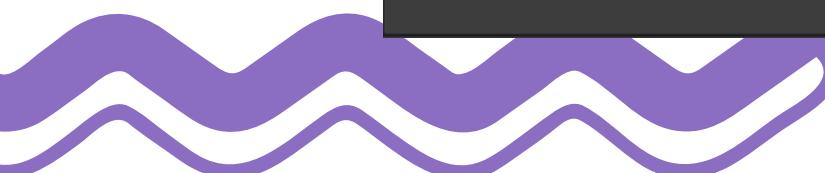


The Scene should look like this



CREATING THE MENU

Make Main menu code like this and put it on the main menu object



Main Menu (Mono Script) Import Settings

Open Execution Order...

Loading Overlay None (Rect Transform)

Default references will only be applied in edit mode.

Imported Object

Main Menu (Mono Script)

Assembly Information

Filename Assembly-CSharp.dll

```
using UnityEngine;
using UnityEngine.SceneManagement;

public class MainMenu : MonoBehaviour
{
    public RectTransform loadingOverlay;

    private AsyncOperation _sceneLoadingOperation;

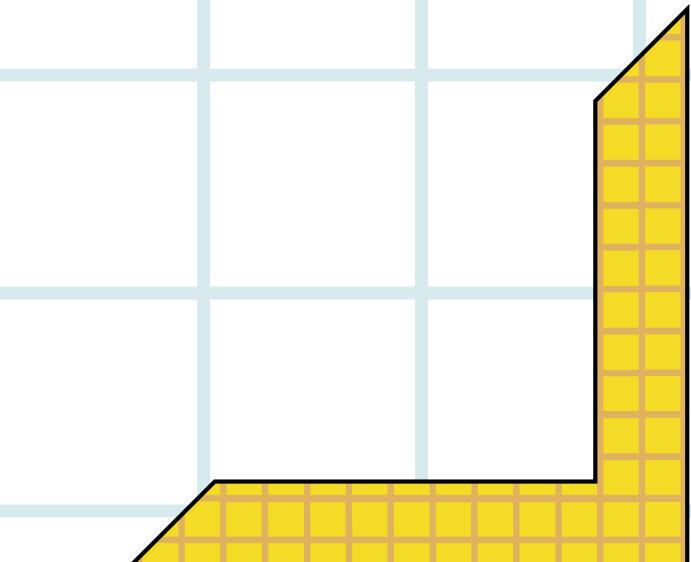
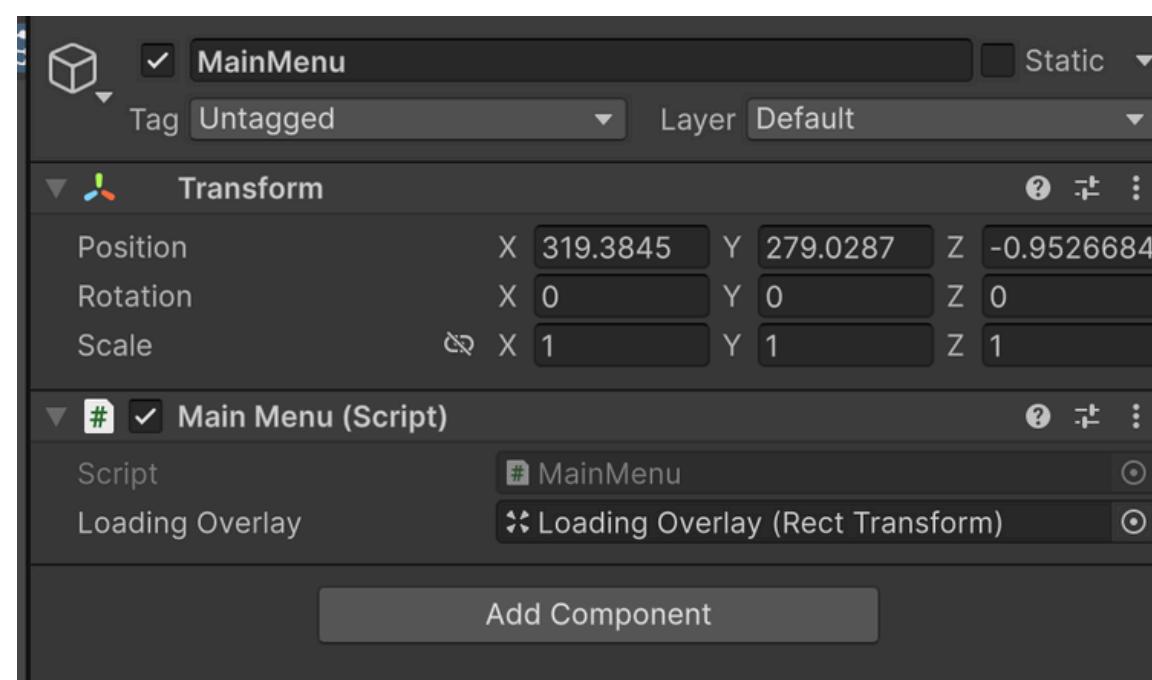
    private void Start()
    {
        loadingOverlay.gameObject.SetActive(false);

        _sceneLoadingOperation = SceneManager.LoadSceneAsync(Scenes.MAIN_SCENE);
        _sceneLoadingOperation.allowSceneActivation = false;
    }

    public void LoadScene()
    {
        loadingOverlay.gameObject.SetActive(true);

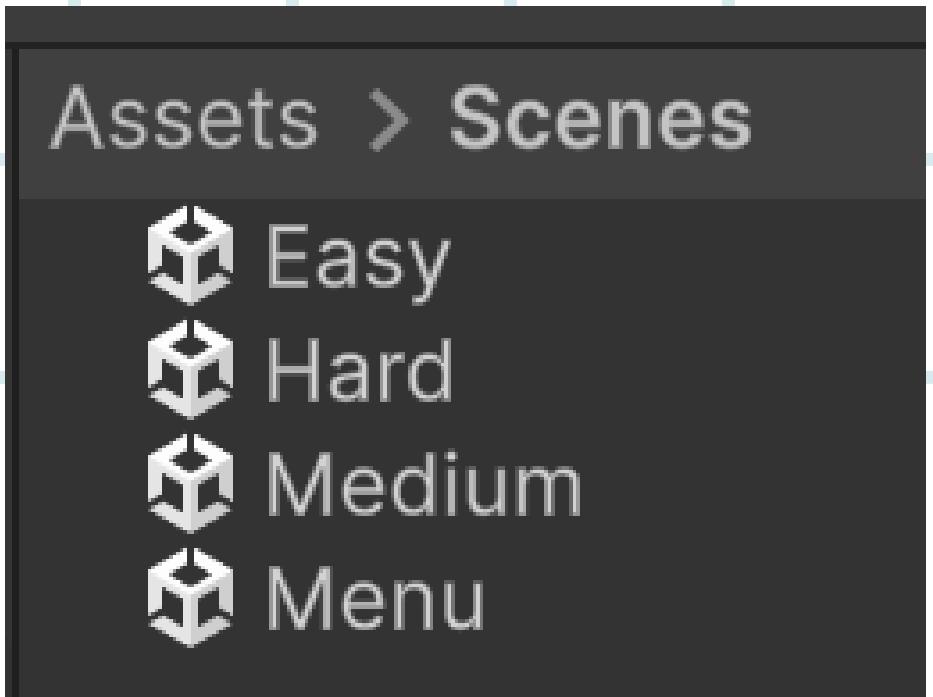
        _sceneLoadingOperation.allowSceneActivation = true;
    }
}
```

Add mainmenu.cs to the mainmenu object



CREATING LEVELS

Duplicate the main scene and create all these scenes



Create Menu.cs for the Menu Scene

```
# Menu (Mono Script)
Assembly Information
Filename Assembly-CSharp.dll

using UnityEngine;
using UnityEngine.SceneManagement;

public class Menu : MonoBehaviour
{
    // Difficulty buttons
    public void PlayEasy()
    {
        SceneManager.LoadScene("Easy");
    }

    public void PlayMedium()
    {
        SceneManager.LoadScene("Medium");
    }

    public void PlayHard()
    {
        SceneManager.LoadScene("Hard");
    }

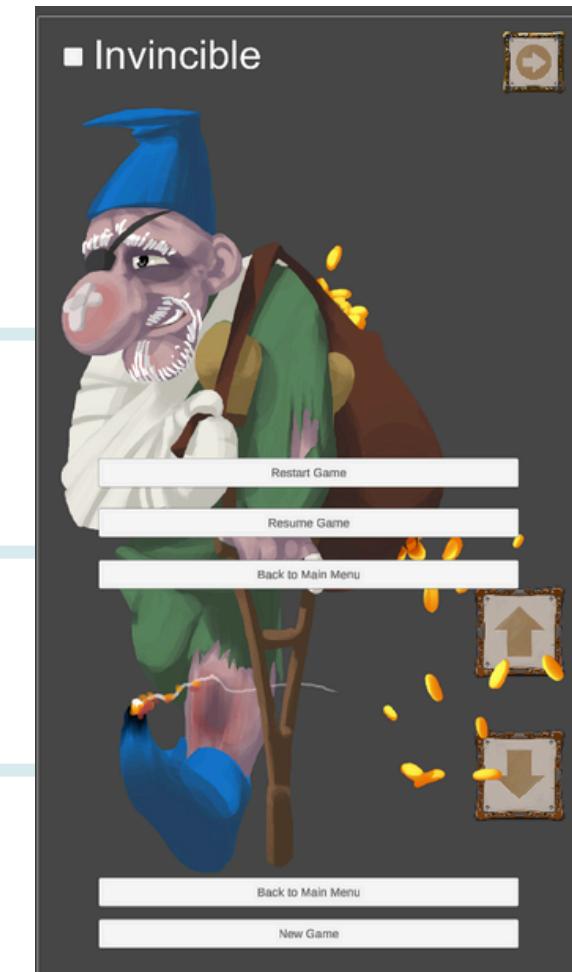
    // Back to Menu button (for gameplay scenes)
    public void BackToMenu()
    {
        Time.timeScale = 1f; // Ensure game is unpause
        SceneManager.LoadScene("Menu");
    }

    // Quit button
    public void QuitGame()
    {
        Debug.Log("Quit Game");
        Application.Quit();
    }
}
```

Make buttons and arrange Menu Scene Like this



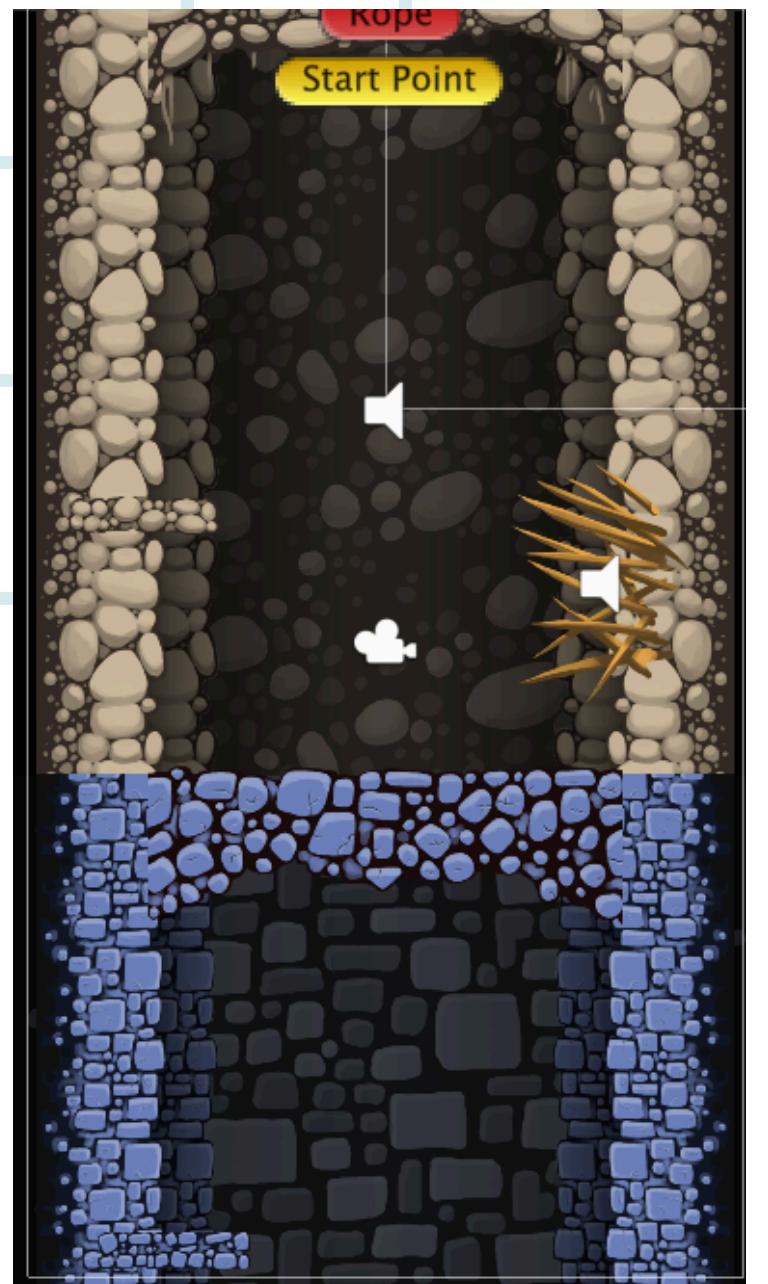
And Arrange the Duplicated Scene



CREATING LEVELS

And Update the Duplicated levels to match the difficulty

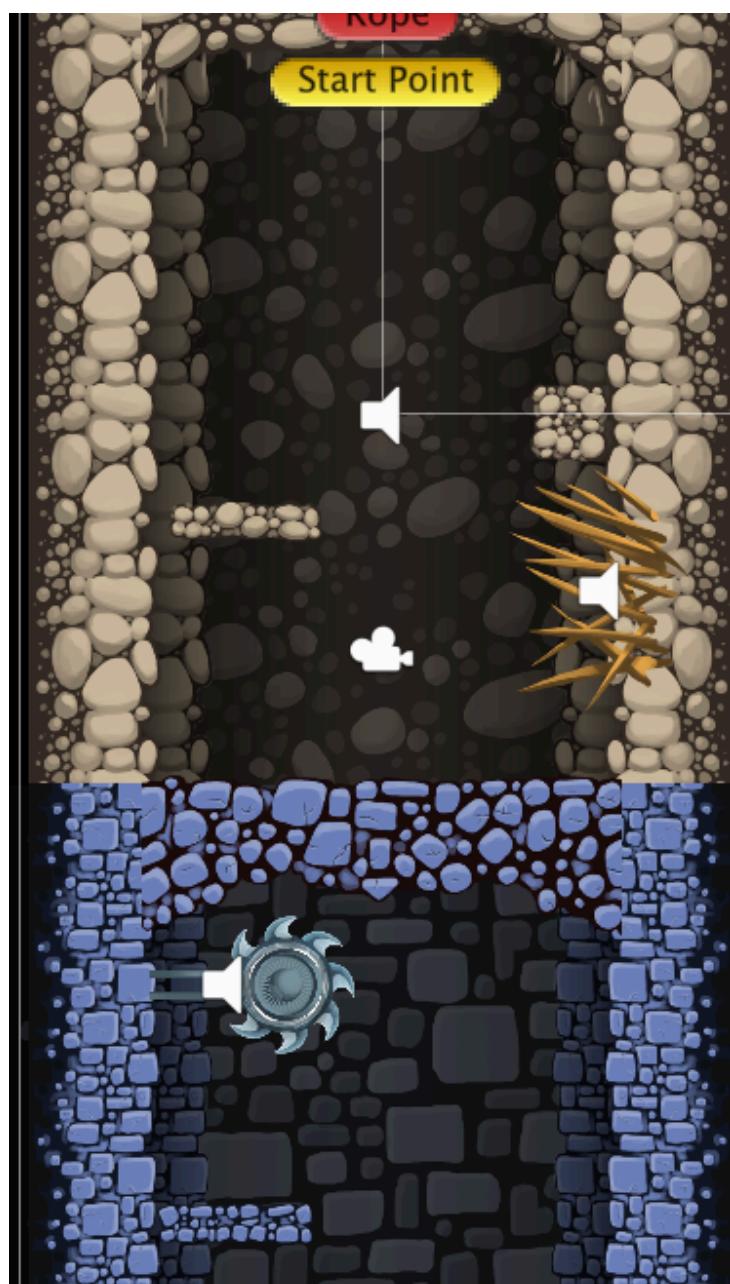
Easy



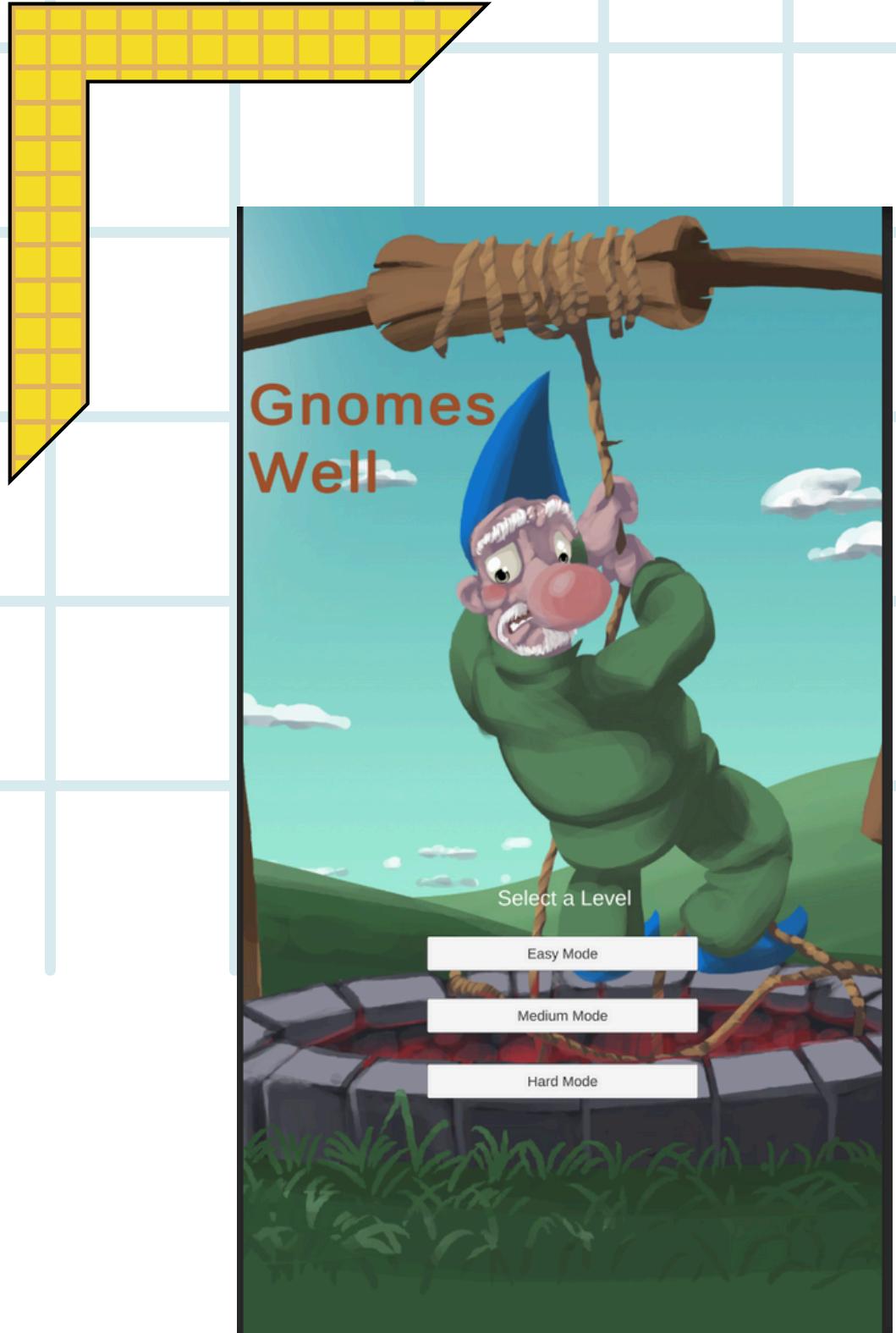
Medium



Medium



FINAL OUTPUT



After All the steps that we've done we should have a decent 2D game output that runs with unity which are these pictures, in the left with the **Menu scene** and in the right the **Game scene**.

