Games Programming Workshop 02

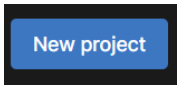
Contents

Games Programming Workshop 01 1
Exercise 1 2

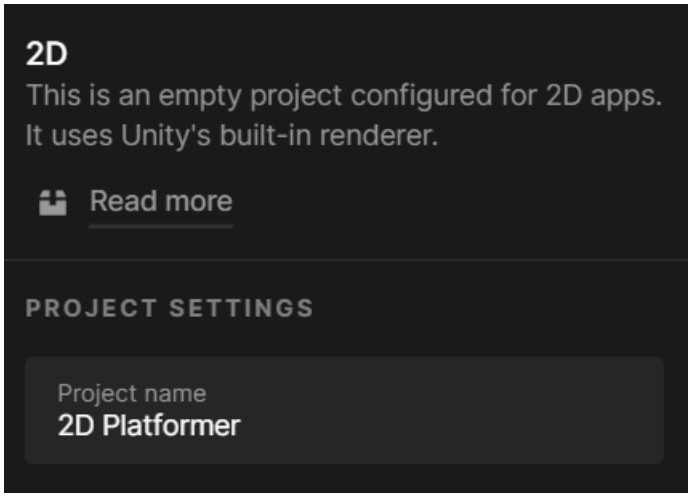
Exercise 1

In this exercise, you will create a 2D ‘Gravity Switching’ Platformer. We will add objects to the game that repeat what we did in Workshop 1, and we will continue use basic geometric objects as placeholders for more complex sprites.

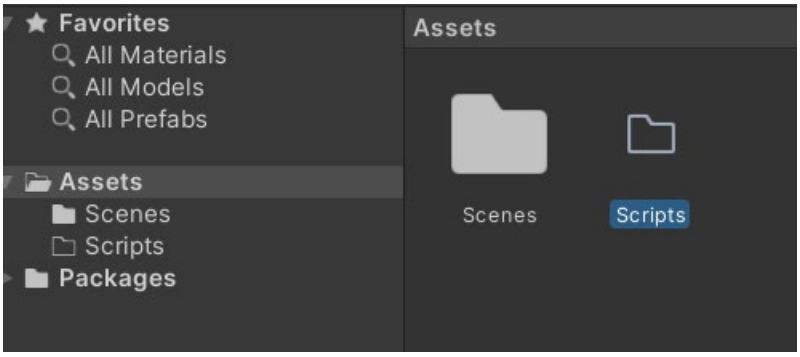
1) In the Unity Hub app,



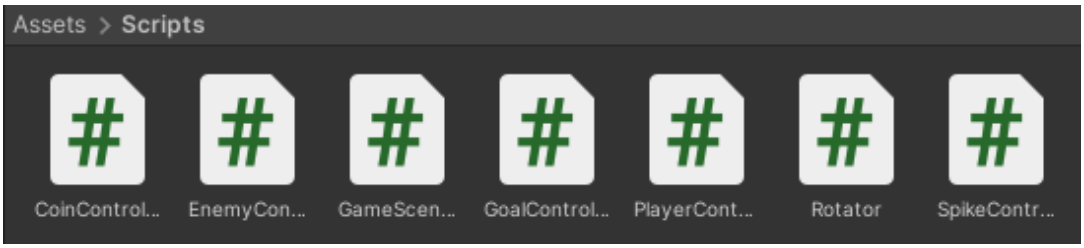
2) Create a new 2D (Core) Project. Name the project whatever you like, though I have called mine 2D Gravity Platformer (If you are on a Uni PC, save this to the K: Drive)



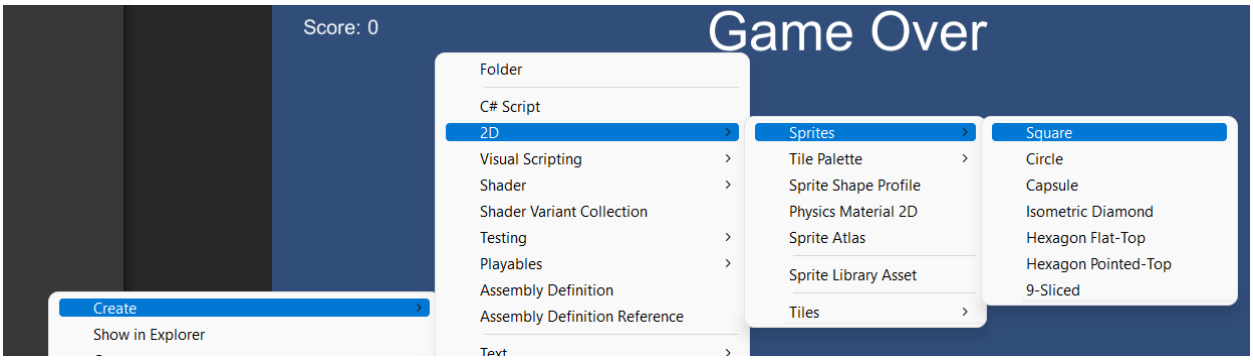
3) Right click in the Assets folder in the Content Manager and choose Create > Folder



4) Add scripts



5) Create a new folder called Graphics and then add a 2D Square



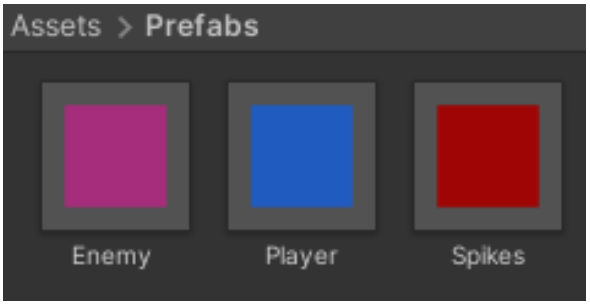
6) Create a **Physics** folder

7) Add a **Physics 2D Material** from the **2D** right-click menu

8) Create a Prefab folder and add a Prefab for Enemy, Player and Spike

9) Add a **Sprite Renderer** component to each Prefab and set the sprite to **Square**

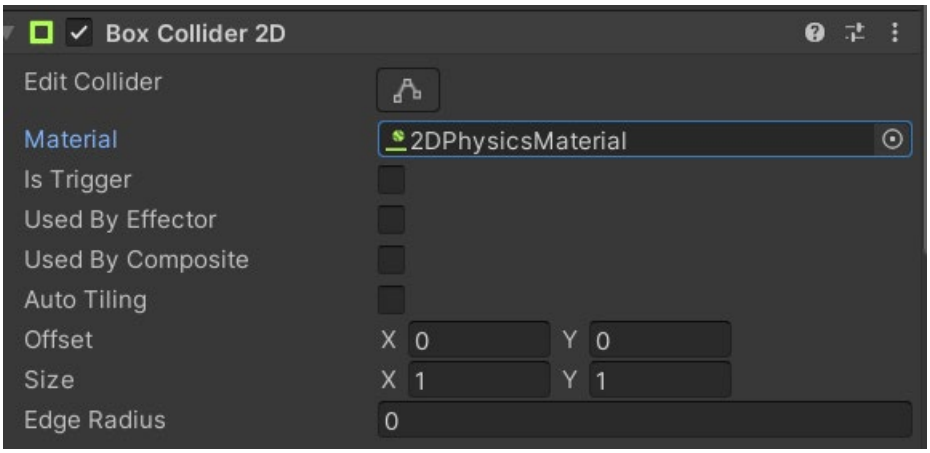
10) Set the colour of each Prefab to be different



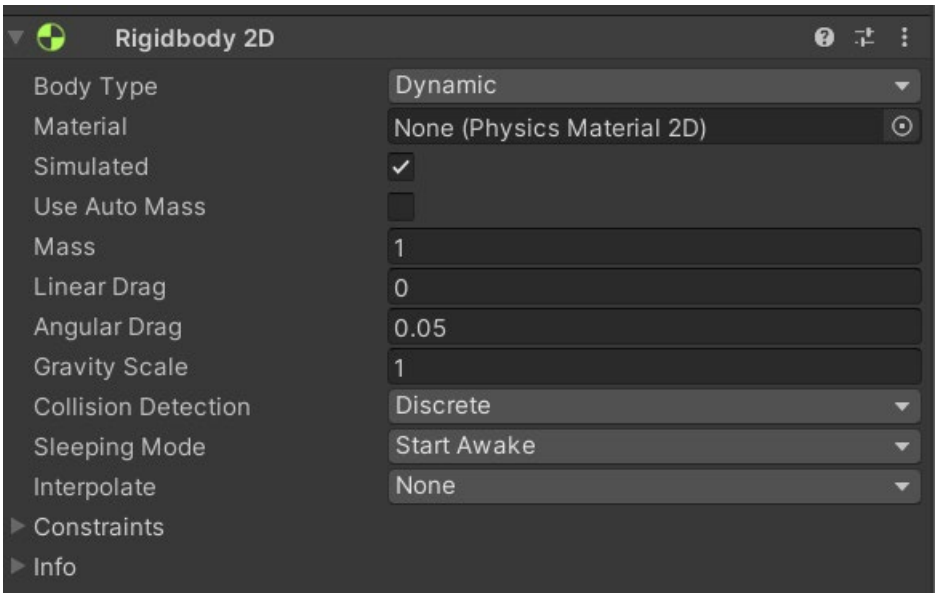
11) Add **Enemy Controller script** to Enemy Prefab

12) Add **Box Collider 2D** to the Enemy Prefab

13) Add this material to the **Box Collider 2D** in Enemy

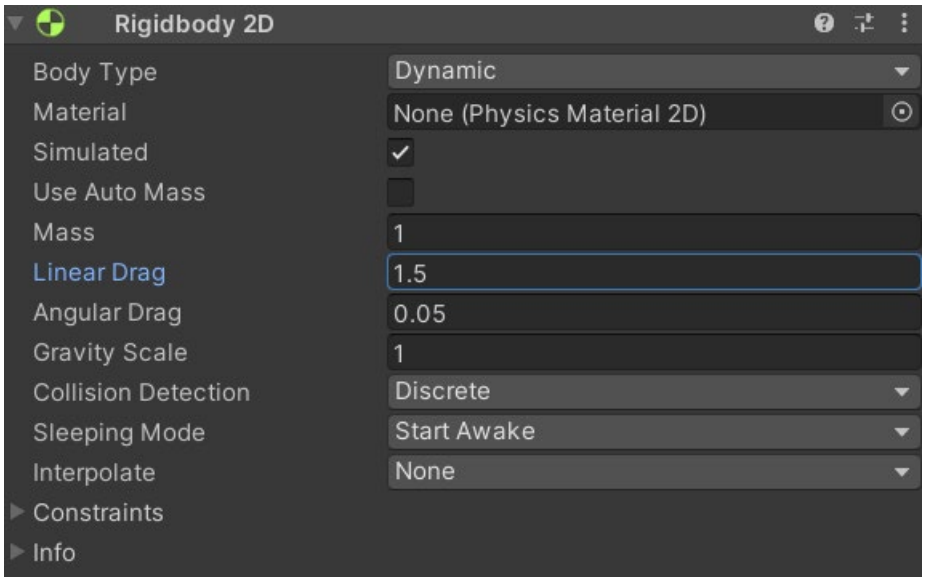


14) Add a **Rigidbody 2D** to the Enemy

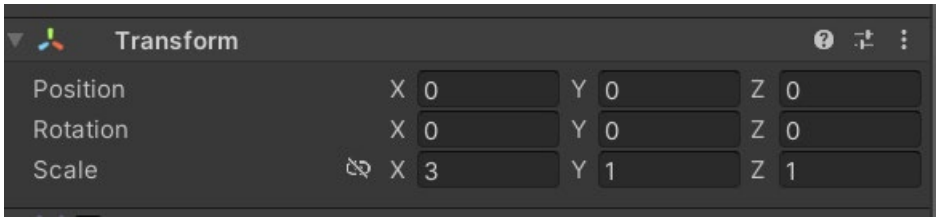


15) Repeat all these steps for the **Player**

16) Change **Transform Scale** of **Player** to x0.5, y1, z1 to make it skinny



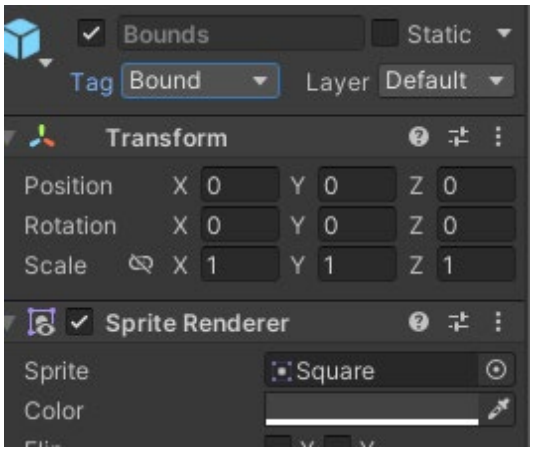
17) Change **Transform** of **Spike** to x3, y1, z1 to make it wide



18) Add the **Spike Controller script** and **Box Collider 2D**, but this time do not add a Rigidbody as the spikes are stationary and will not need physics.

19) Add a new prefab to the Prefabs folder and call it **Bounds.**

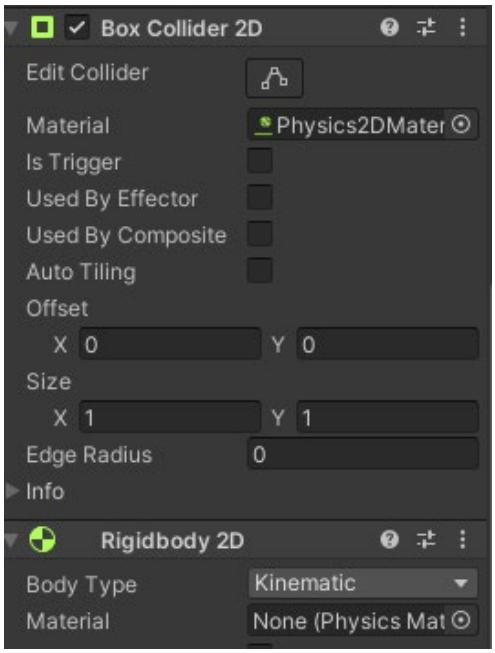
20) Add a new Tag called Bound and tag the Bounds prefab with this.



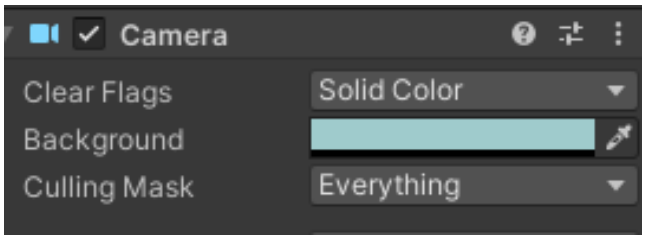
21) Add a Sprite Renderer component and assign the **Square** graphic, setting the colour to dark grey.

22) Add a **Box Collider 2D** and assign the **2d Physics material**

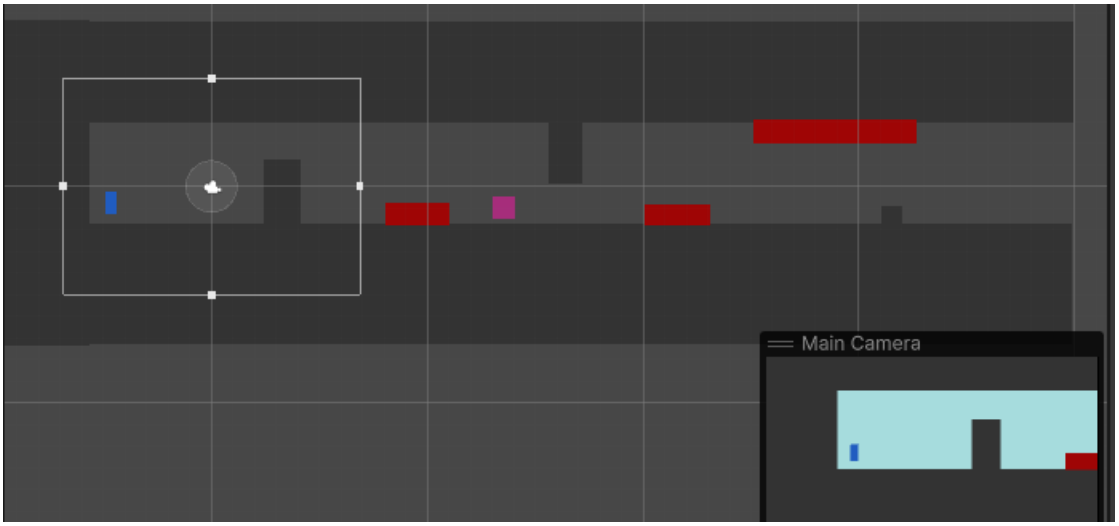
23) Add a **Rigidbody 2D** and change the **Body Type** to **Kinematic**



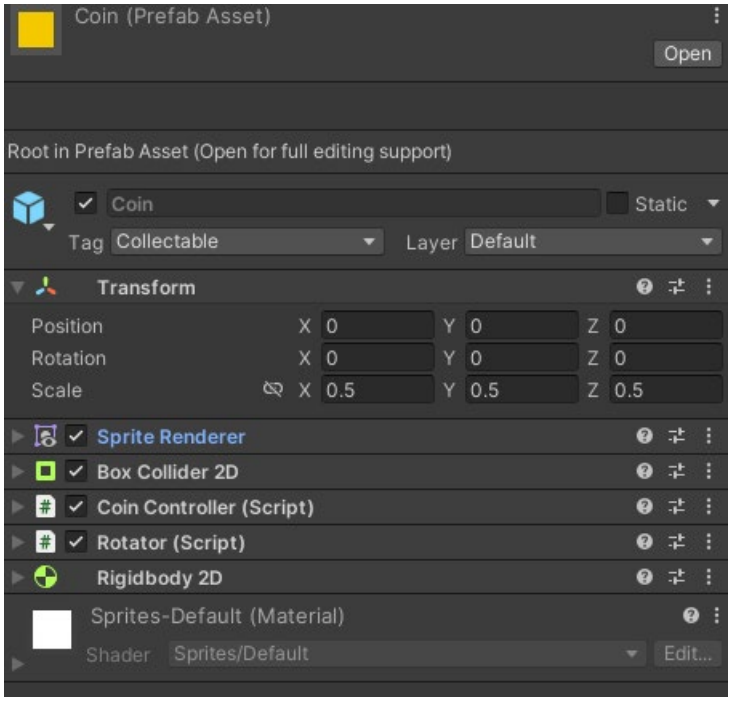
24) Select the Main Camera object in the hierarchy and set the Background property to a light sky blue colour.



25) Drag instances of the Bounds prefab into the scene and resize them to build the level. As this will be a gravity switching game your test level should be quite long. Also add the Player, an Enemy and some Spikes to the level layout.

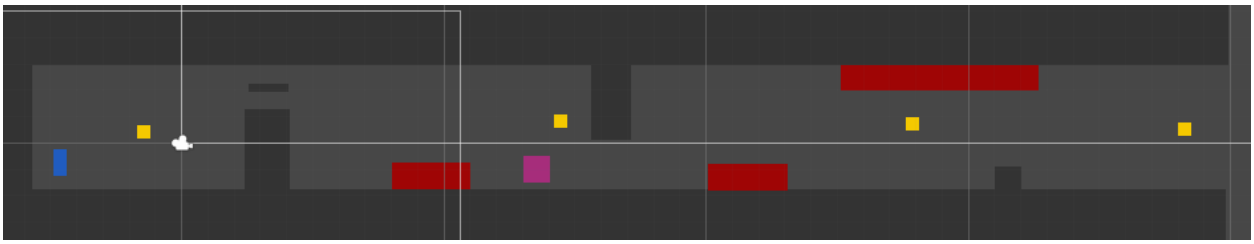


26) Create a Coin Prefab and Add the **Sprite Renderer** and **Square**. Transform scaled 0.5. Add a **Box Collider 2D,** then add the **CoinController** and **Rotator** scripts**.**

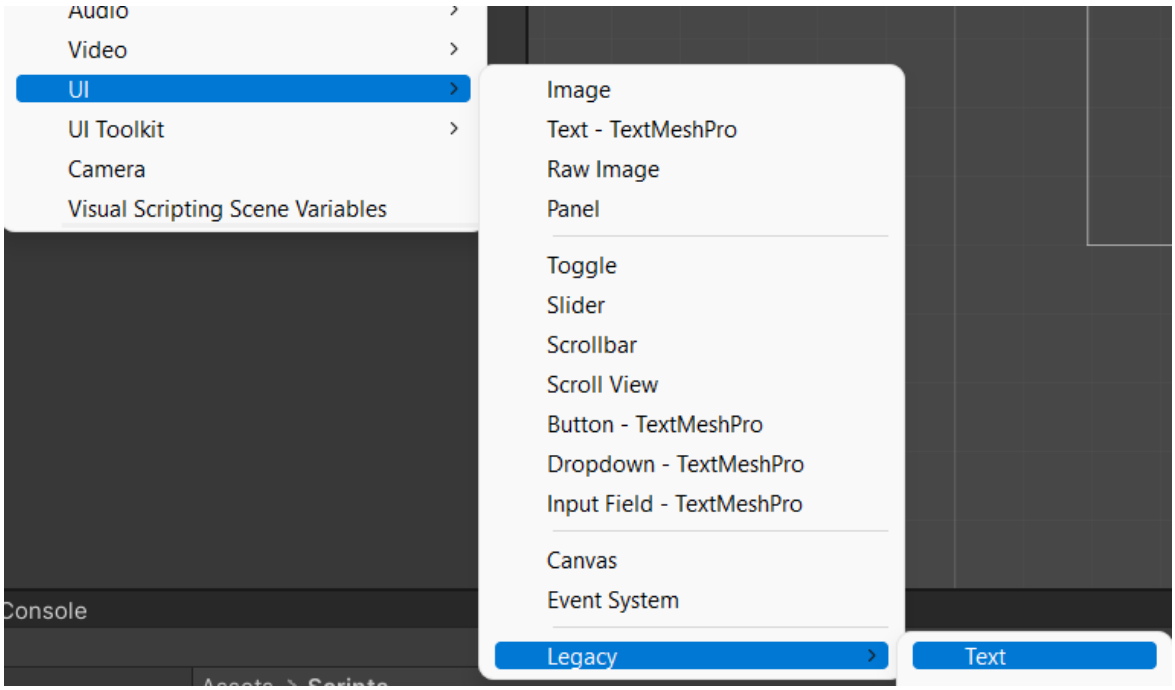


27) Se the Coin **Rigidbody 2D** to have a **Mass** of 0.0001 and a **Gravity Scale** of 0. This will prevent it from falling to the ground and also stop the player from bouncing off it.

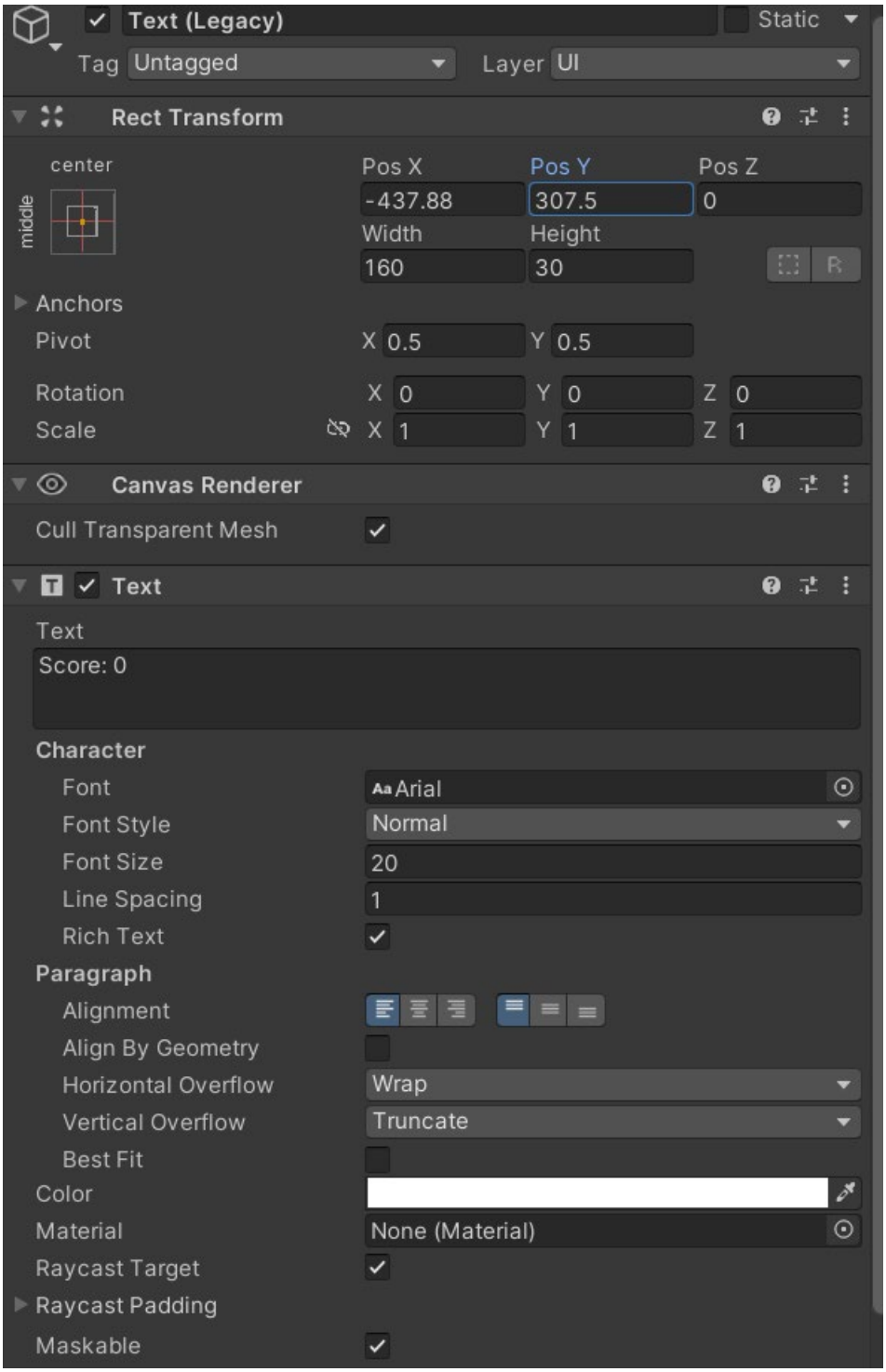
28) Add Coins to your level



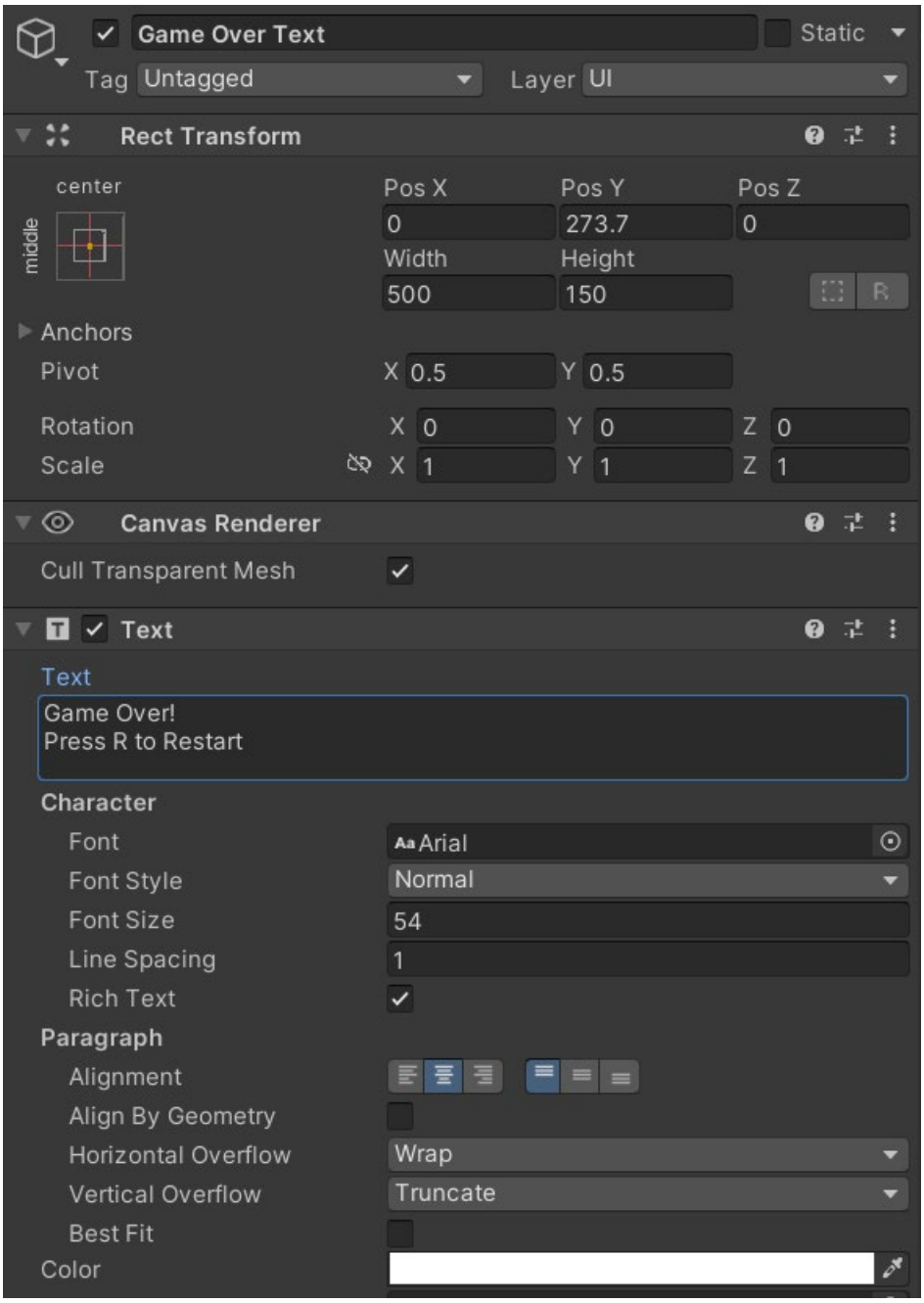
29) Add **Legacy Text UI** component to the Hierarchy



30) Position the text for **Score** (In game Mode), rename to **Score Text**


Text (Legacy)
Static
Tag Untagged
Layer UI
Rect Transform
center
middle
Pos X
Pos Y
Pos Z
-437.88
307.5
0
Width
Height
160
30
Anchors
Pivot
X 0.5
Y 0.5
Rotation
X 0
Y 0
Z 0
Scale
X 1
Y 1
Z 1
Canvas Renderer
Cull Transparent Mesh
Text
Text
Score: 0
Character
Font
Arial
Font Style
Normal
Font Size
20
Line Spacing
1
Rich Text
Paragraph
Alignment
Align By Geometry
Horizontal Overflow
Wrap
Vertical Overflow
Truncate
Best Fit
Color
Material
None (Material)
Raycast Target
Raycast Padding
Maskable

31) Duplicate as **Game Over Text** and position in the middle of the screen.



22) Open the **PlayerController** script in **VS Code**

```
using System;
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class PlayerController : MonoBehaviour
{
    public delegate void OnHitSpikeAction();
    public delegate void OnHitCoinAction();

    public OnHitSpikeAction OnHitSpike;
    public OnHitCoinAction OnHitCoin;

    float speed = 1000;

    Vector3 leftBound;
    Vector3 rightBound;
    bool canJump;

    // Start is called before the first frame update
    void Start()
    {
        if (Physics2D.gravity.y > 0)
        {
            Physics2D.gravity *= -1;
        }
    }

    // Update is called once per frame
    void Update()
    {
        ProcessInput();
    }

    private void ProcessInput()
    {
        if(Input.GetKey("left") || Input.GetKey("a"))
            this.GetComponent<Rigidbody2D>().AddForce(Vector2.left * speed *
Time.deltaTime);

        if(Input.GetKey("right") || Input.GetKey("d"))
```

```
            this.GetComponent<Rigidbody2D>().AddForce(Vector2.right * speed *
Time.deltaTime);

        if(Input.GetKeyDown("up") || Input.GetKey("w"))
            InvertGravity();
    }

    private void InvertGravity()
    {
        Physics2D.gravity *= -1;
    }

    public void OnCollisionEnter2D(Collision2D collision)
    {
        if(collision.gameObject.GetComponent<SpikeController>() != null)
        {
            if (OnHitSpike != null)
            {
                OnHitSpike();
            }
        }
        else if(collision.gameObject.GetComponent<EnemyController>() != null)
        {
            if(OnHitSpike != null)
            {
                OnHitSpike();
            }
        }

        else if(collision.gameObject.GetComponent<CoinController>() != null)
        {
            GameObject.Destroy(collision.gameObject);

            if(OnHitCoin != null)
            {
                OnHitCoin();
            }
        }
    }
}
```

23) Open the **EnemyController** in **VSCode**

```
24)using System.Collections;
25)using System.Collections.Generic;
26)using UnityEngine;
27)
28)public class EnemyController : MonoBehaviour
29){
30)
31)    public float PatrolSpeed = 300f;
32)    public float PatrolDuration = 3f;
33)
34)    float patrolTimer;
35)
36)    Vector2 direction;
37)
38)    // Start is called before the first frame update
39)    void Start()
40)    {
41)        this.direction = Vector2.left;
42)    }
43)
44)    // Update is called once per frame
45)    void Update()
46)    {
47)        patrolTimer += Time.deltaTime;
48)
49)        if(patrolTimer >= PatrolDuration)
50)        {
51)            this.GetComponent<Rigidbody2D>().velocity = Vector2.zero;
52)
53)            patrolTimer = 0;
54)            direction *= -1;
55)        }
56)
57)        this.GetComponent<Rigidbody2D>().AddForce(this.direction *
   PatrolSpeed * Time.deltaTime);
58)    }
59)}
60)
```

22) Open the **Rotator** script in **VSCode**

```
23)using System.Collections;
24)using System.Collections.Generic;
25)using UnityEngine;
26)
27)public class Rotator : MonoBehaviour
28){
29)
30)    public float X_Speed = 0f;
31)    public float Y_Speed = 0f;
32)    public float Z_Speed = 400f;
33)    // Start is called before the first frame update
34)    void Start()
35)    {
36)
37)    }
38)
39)    // Update is called once per frame
40)    void Update()
41)    {
42)        this.transform.localEulerAngles += new Vector3(
43)            this.transform.localEulerAngles.x + X_Speed,
44)            this.transform.localEulerAngles.y + Y_Speed,
45)            this.transform.localEulerAngles.z + Z_Speed
46)        ) * Time.deltaTime;
47)    }
48)}
49)
```

50) Back in Unity, set the Rotator Script on the Coin Prefab to have X_Speed=0, Y_Speed=0, Z_Speed = 200

51) Also, try adding the Rotator script to a Bounds object and rotate it on the Z axis to create a spinning obstacle.

52) Open the **GamesceneManager** in **VSCode**

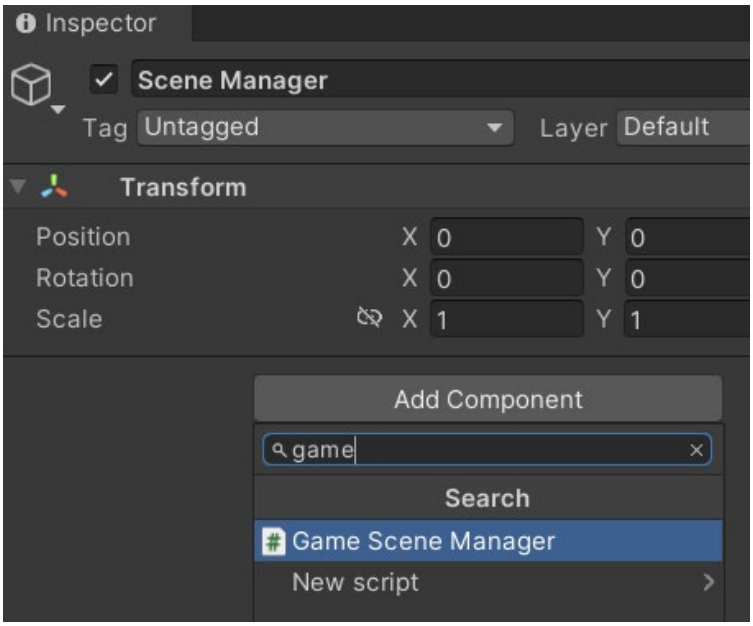
```
53)using System.Collections;
54)using System.Collections.Generic;
55)using UnityEngine;
56)using UnityEngine.UI;
57)using UnityEngine.SceneManagement;
58)using System;
59)
60)public class GameSceneManager : MonoBehaviour
61){
62)    public Camera MainCamera;
63)    public Text ScoreText;
64)    public Text GameOverText;
65)    public PlayerController player;
66)    int score;
67)    float gameTimer;
68)    bool gameOver;
69)

70)    // Start is called before the first frame update
71)    void Start()
72)    {
73)        Time.timeScale = 1; //start time
74)        player.OnHitSpike += OnGameOver;
75)        player.OnHitCoin += OnHitCoin;
76)        ScoreText.enabled = true;
77)        GameOverText.enabled = false;
78)
79)    }
80)
81)    // Update is called once per frame
82)    void Update()
83)    {
84)        MainCamera.transform.position = new Vector3(
85)        Mathf.Lerp(MainCamera.transform.position.x,
   player.transform.position.x, Time.deltaTime * 10),
86)        Mathf.Lerp(MainCamera.transform.position.y,
   player.transform.position.y, Time.deltaTime * 10),
87)        MainCamera.transform.position.z);
88)        if(gameOver)
89)        {
90)            if(Input.GetKeyDown("r"))
91)            {
```

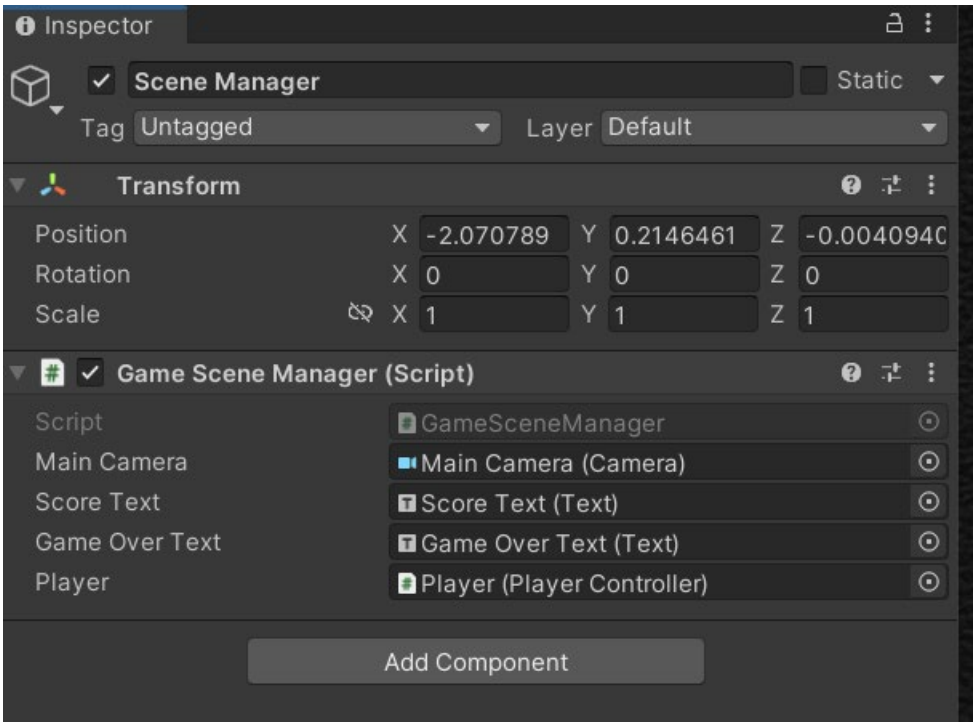
```
92)                 SceneManager.LoadScene
   (SceneManager.GetActiveScene().name);
93)             }
94)             return; // Skip the following lines if GameOver
95)         }
96)
97)         ScoreText.text = "Score: " + score;
98)         if(player.transform.position.y < -10)
99)             OnGameOver();
100)        }
101)
102)            private void OnHitCoin()
103)        {
104)            this.score += 100;
105)        }
106)        private void OnGameOver()
107)        {
108)            gameOver = true;
109)            ScoreText.enabled = false;
110)            GameOverText.enabled = true;
111)            GameOverText.text = "Game Over!\nPress R to Resart";
112)
113)            Time.timeScale = 0; //stop time
114)        }
115)
116)    }
```

117) Back in **Unity**, Add a new Empty GameObject to the Scene Hierarchy and drag the Main Camera into it.

118) Add the Game Scene Manager script component to the Scene Manager object



119) Set the Public Properties to the objects in your scene



Extensions

- Extend the level layout
- Add a prefab for a Goal, which completes the level when you collide with it.