
SUPER
MARIO
TM


1988
SUPER
MARIO BROS. 2
1989
SUPER
MARIO LAND
1990
SUPER
MARIO BROS. 3
1991
SUPER
MARIO WORLD
1992
SUPER
MARIO LAND 2:
6 GOLDEN COINS
1996
SUPER
MARIO 64
2002
SUPER
MARIO SUNSHINE

What are the characteristics of Super Mario Bros?

Key characteristics (requirements)

- Side scroller: Mario/Camera moves right
- Gravity effect of jump
- Collects coins and optional powerups
- Enemies (Goombas) try to take lives.
- Time limit in which Mario must reach the end of the level

Paper design!

BG1 2000H
BG2 2400H
BG3 2800H
BG4 2C00H

1	0	3	2
5	4	7	6

BIT

BG PLANNING SHEET

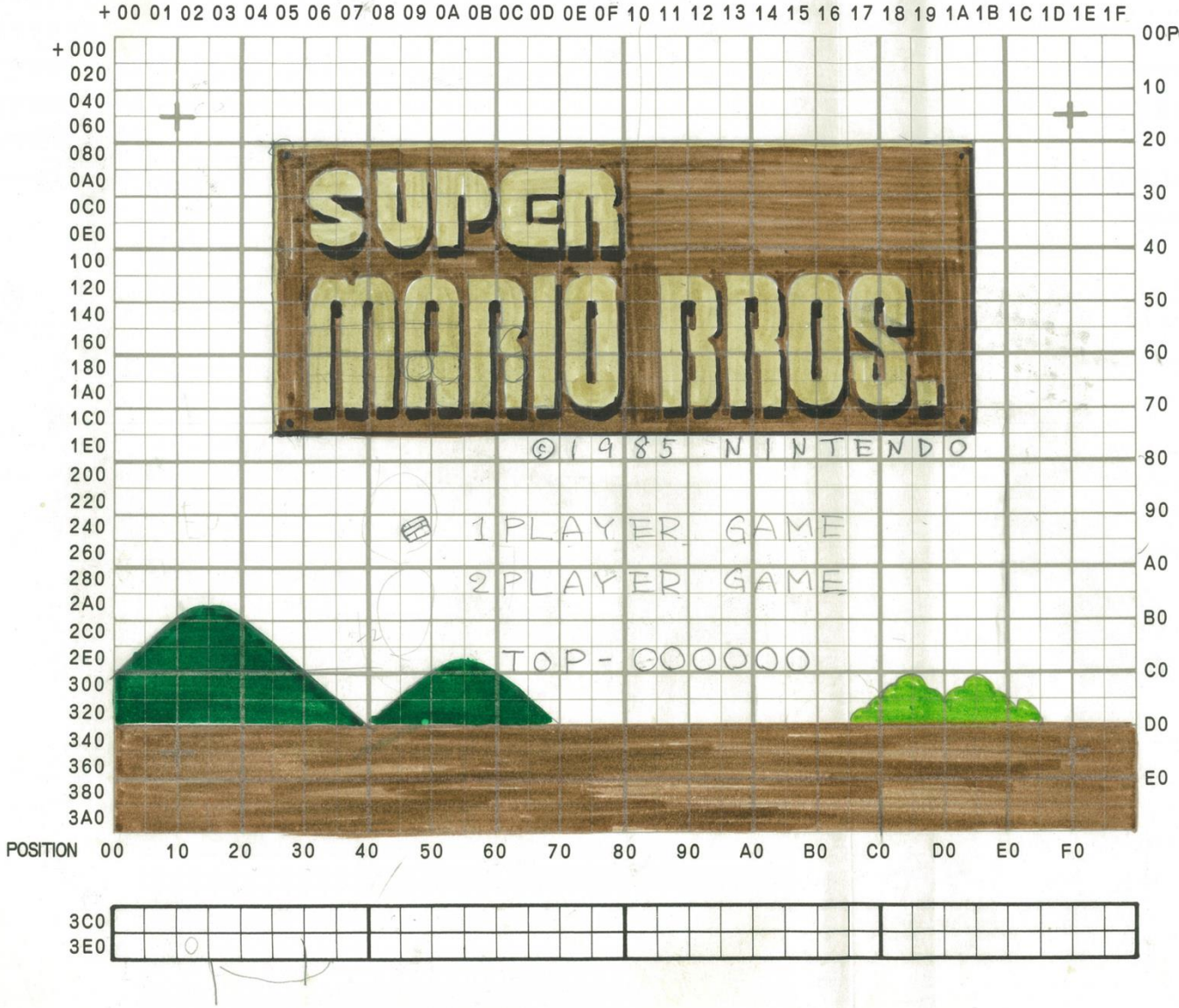
PHASE

年 月 日
AM PM 時 分

デザイナー

プログラマー

MEMO: 地上基本BGのOを使ってもいい



COLOR GENERATOR DATA TABLE

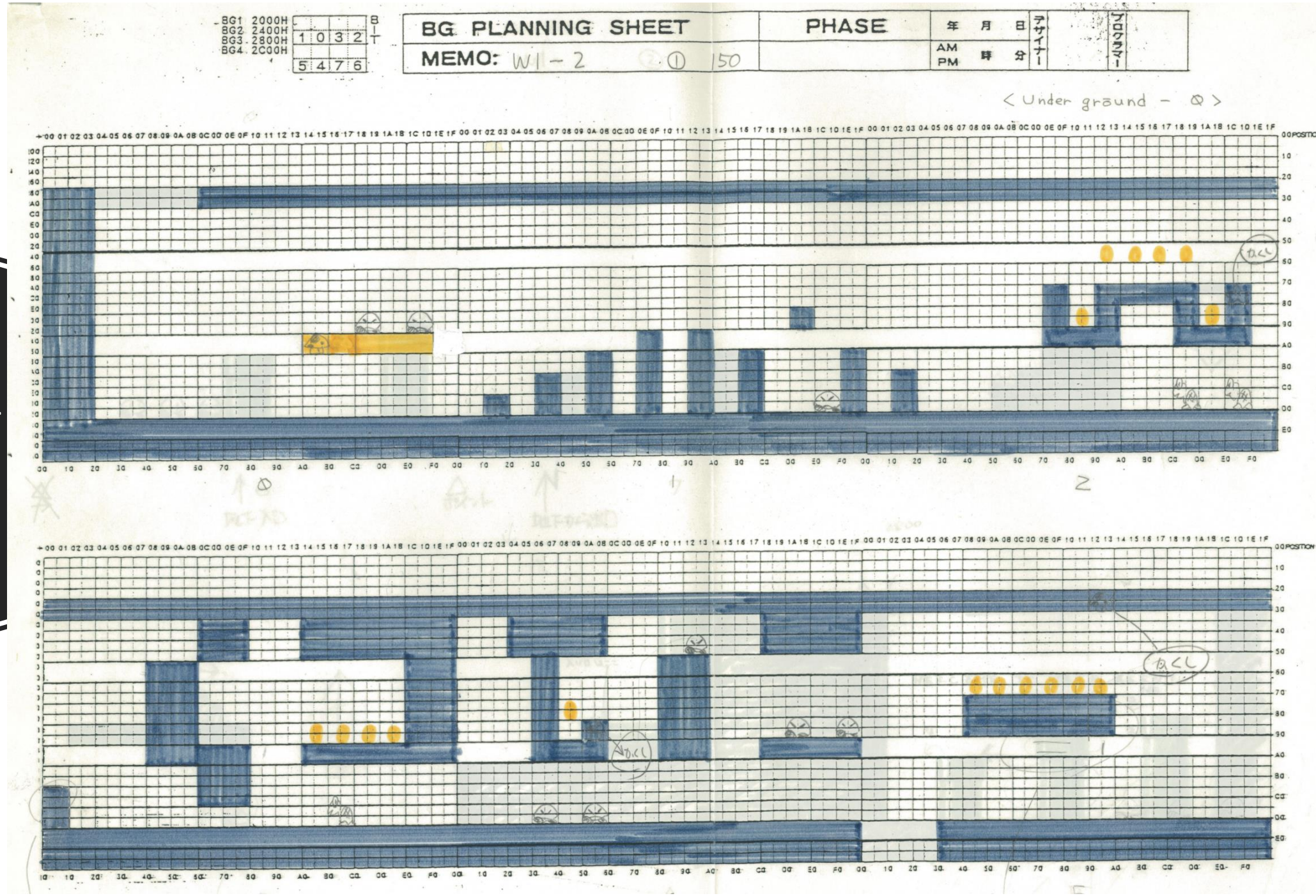
B.G. COLOR

FRAME COLOR	CL D1	CL D0	CHAR m'	CHAR m	C.G. Addr	C.G. DATA	VIDEO COLOR	REMARKS
NONE	0	0	0	0	3F00H			
			0	1	01			
			1	0	02			
			1	1	03			
BROWN	0	1	0	0	04			
			0	1	05			
			1	0	06			
			1	1	07			
RED	1	0	0	0	08			
			0	1	09			
			1	0	0A			
			1	1	0B			
ORANGE	1	1	0	0	0C			
			0	1	0D			
			1	0	0E			
			1	1	0F			

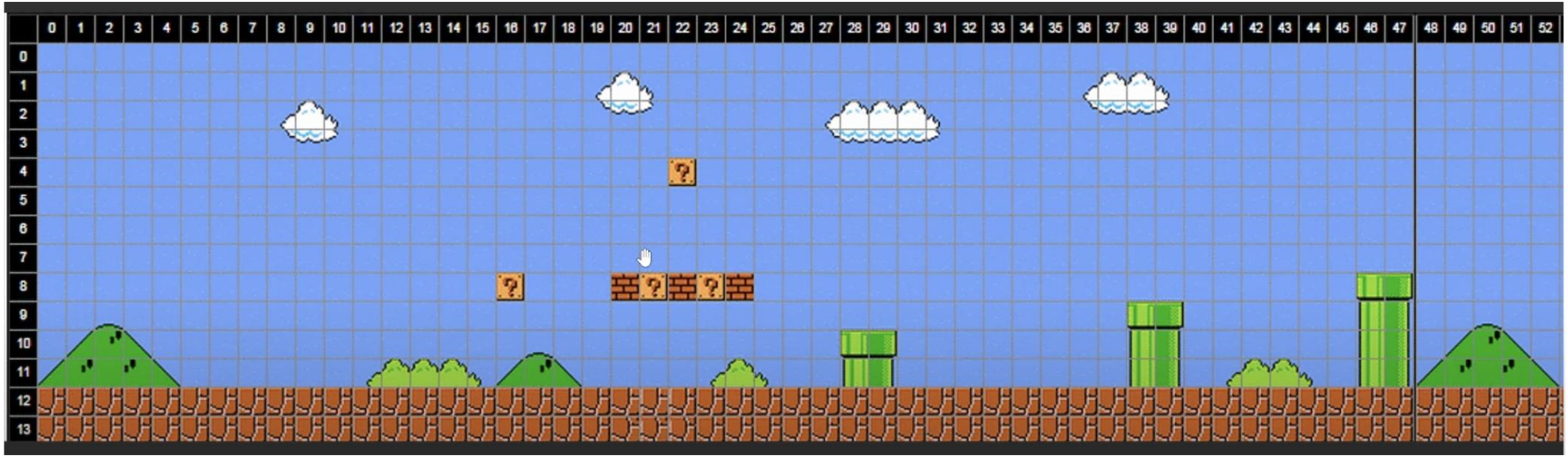
OBJ COLOR

CL D1	CL D0	CHAR m'	CHAR m	C.G. Addr	C.G. DATA	VIDEO COLOR	REMARKS
0	0	0	0	3F10H			
		0	1	11			
		1	0	12			
		1	1	13			
0	1	0	0	14			
		0	1	15			
		1	0	16			
		1	1	17			
1	0	0	0	18			
		0	1	19			
		1	0	1A			
		1	1	1B			
1	1	0	0	1C			
		0	1	1D			
		1	0	1E			
		1	1	1F			

Paper design

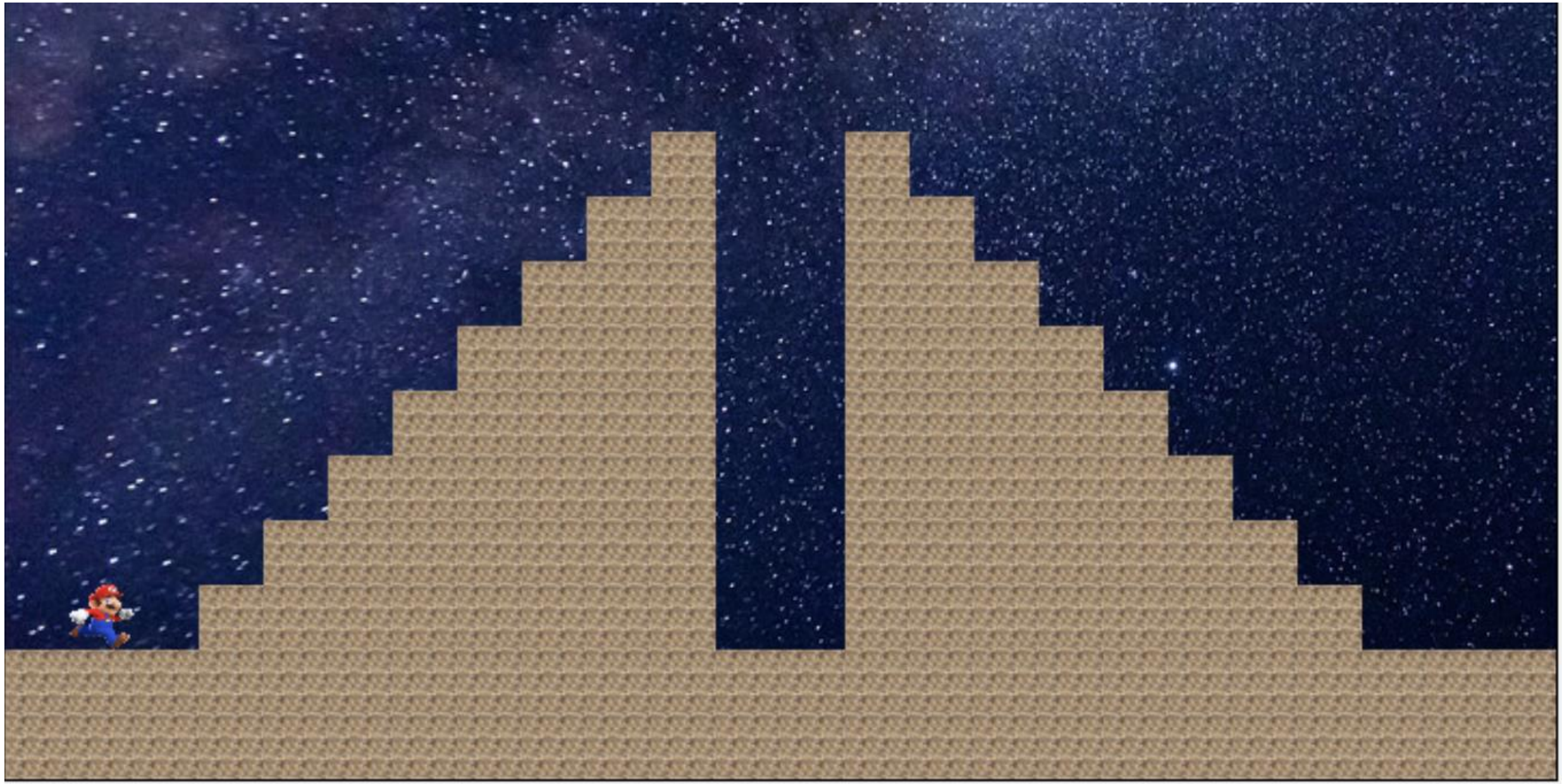


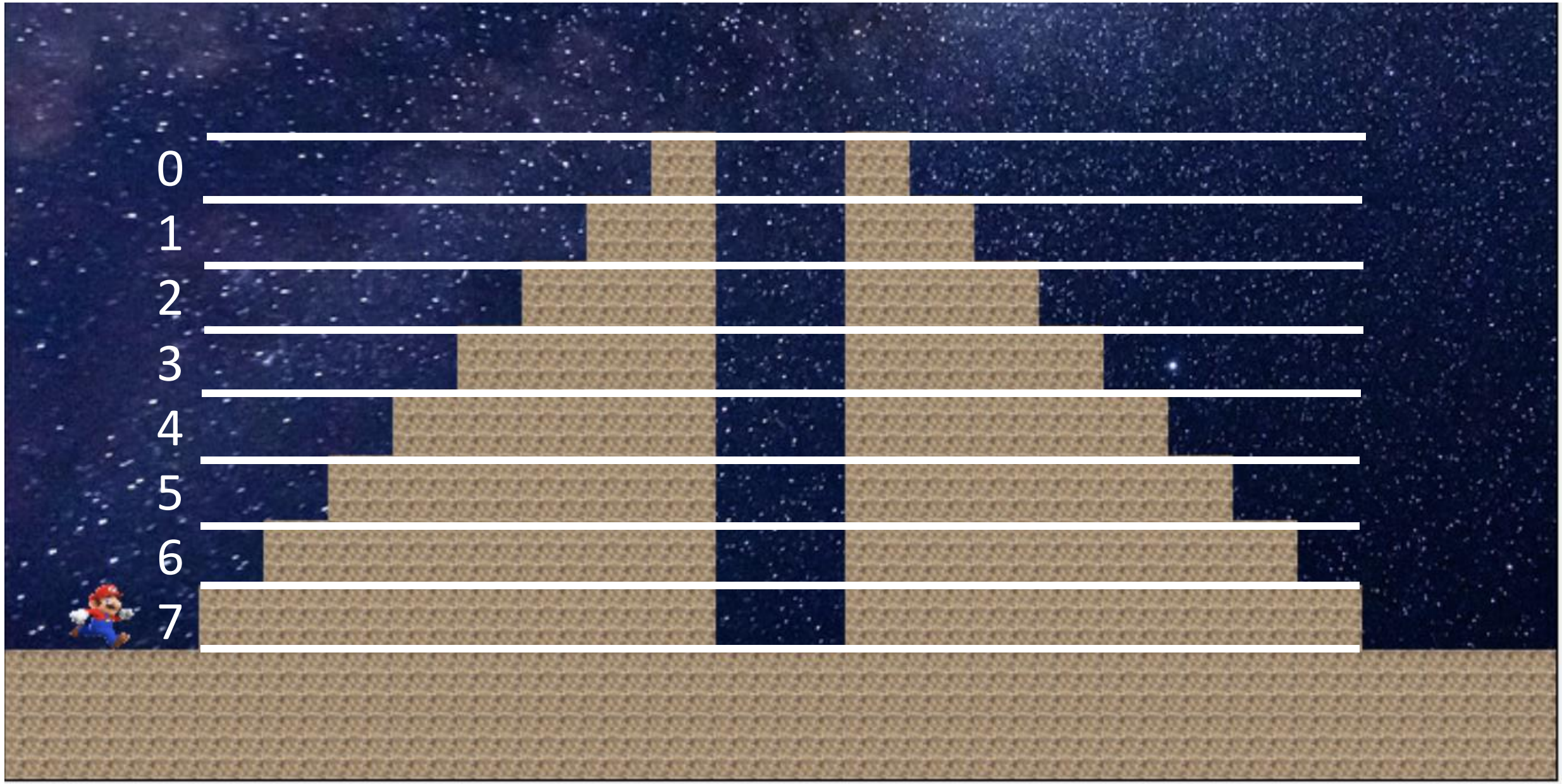
Unity tilemap

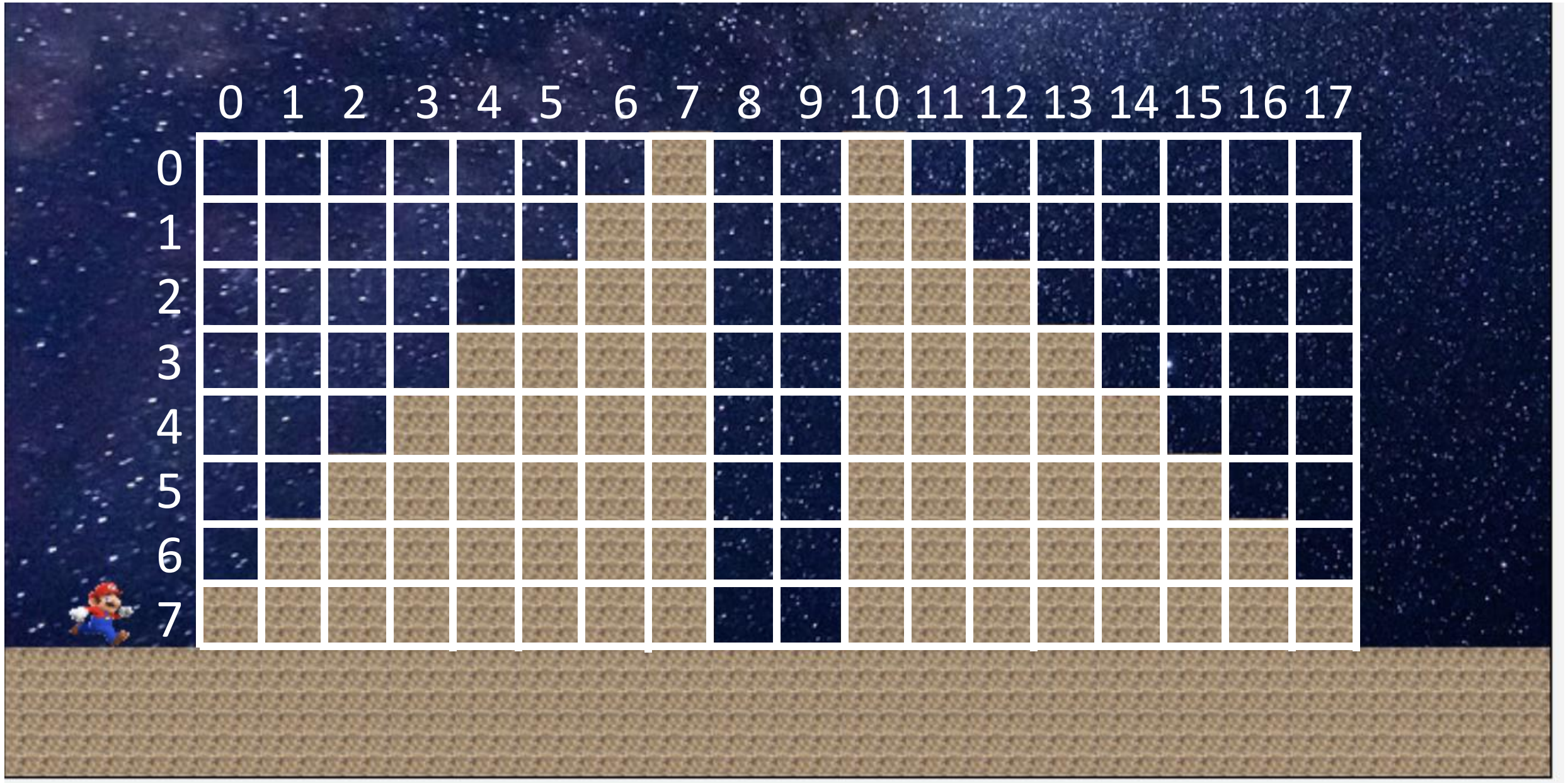


How do we design a solution to this?

- Mario would move right (optionally left if levels allows)
  - Move coordinates (x) when respond to left and right key press
- The array of blocks
  - 2D array of graphics?
- Moving camera with Mario
  - An x coordinate to keep a track of this?
  - Compare x/y coordinates?
- Jump - gravity
  - Change the x and y coordinates accordingly

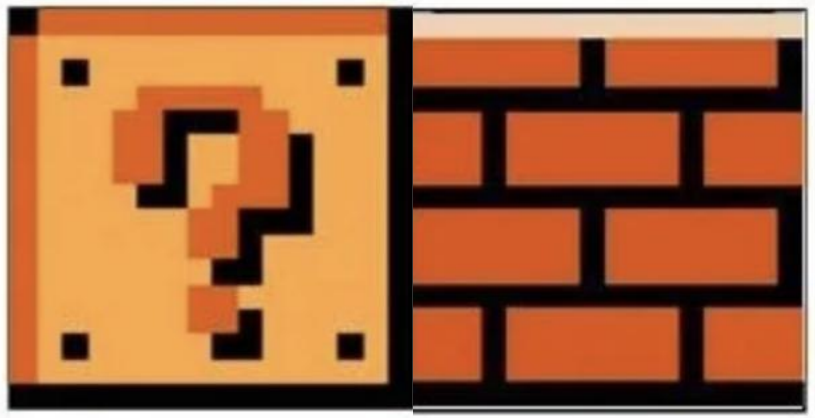


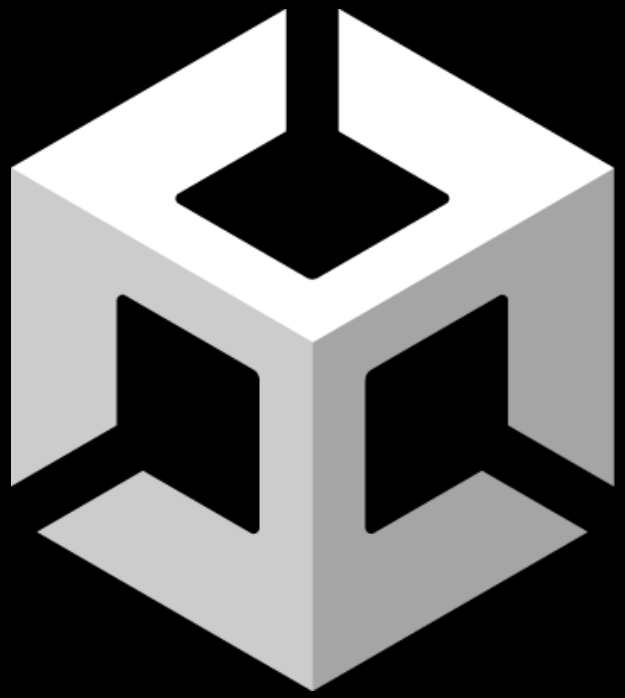

0
1
2
3
4
5
6
7


0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
0
1
2
3
4
5
6
7

What about OOP?

- Should we build some of these entities as classes?



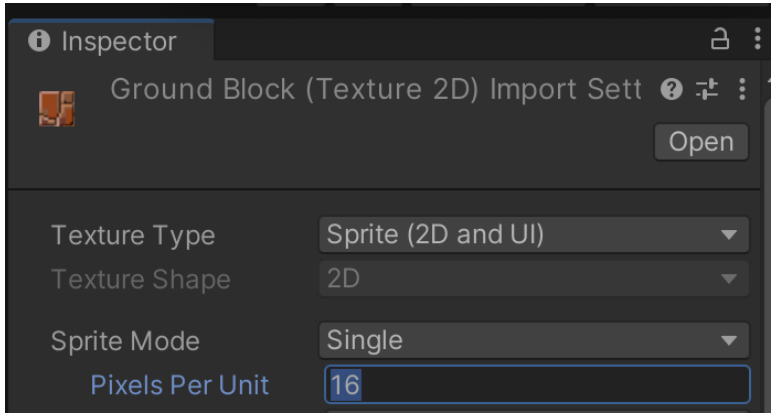


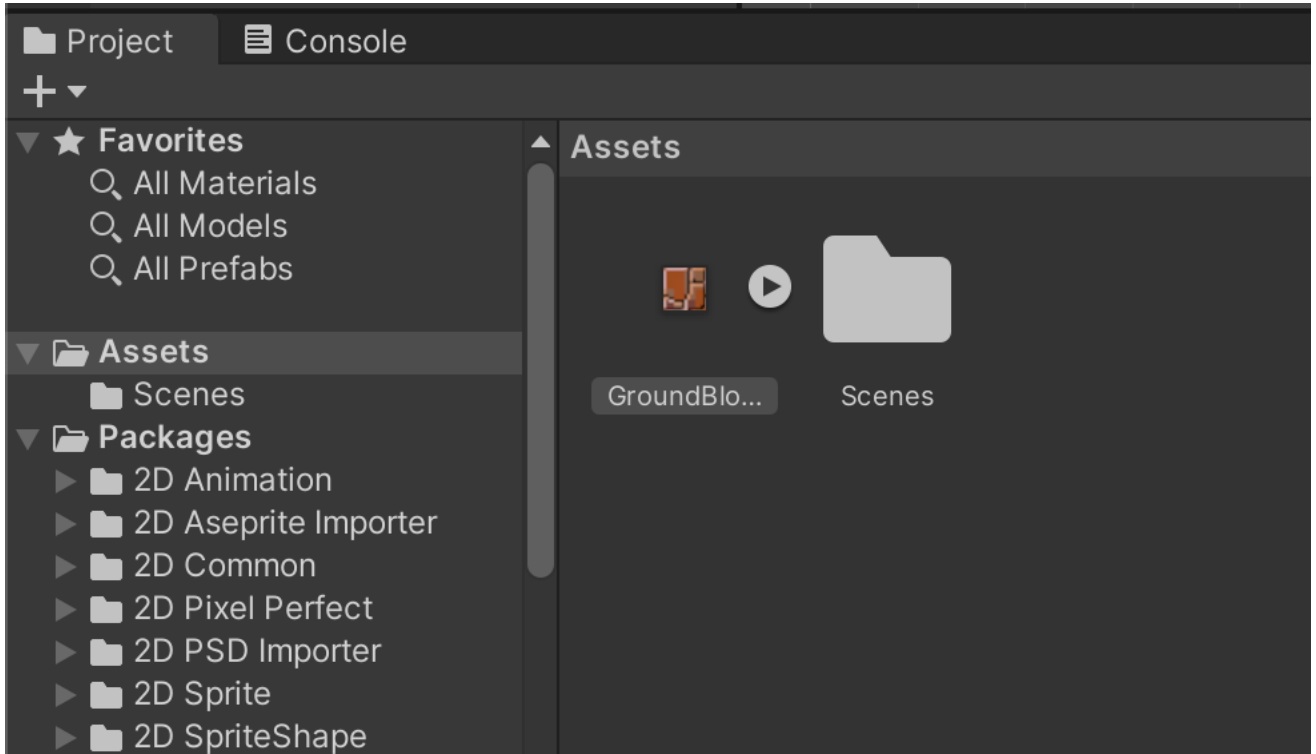
Unity®

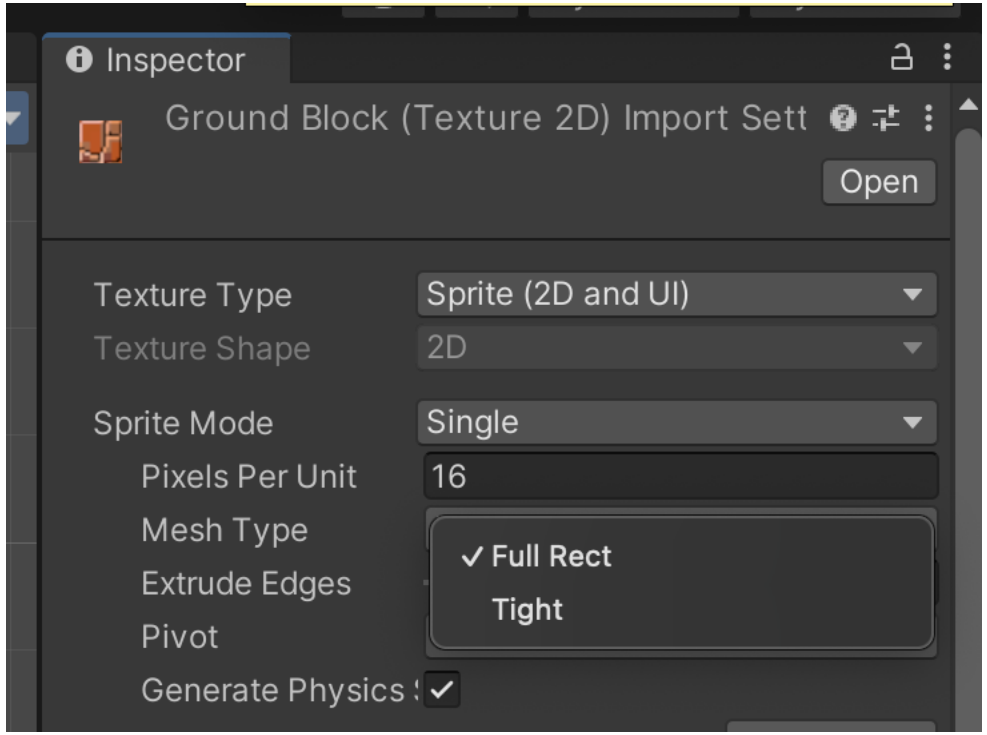
Steps / Requirements

1. **Level Design in Unity**
2. Mario Movement (left + right)
3. Object detection / collision
4. Jumping over blocks / gravity
5. Enemies (Goombas) alternate directions
6. Jumping on Goombas – further collisions
7. Decoration – animation – walking
8. Camera to track Mario (once tile map created)

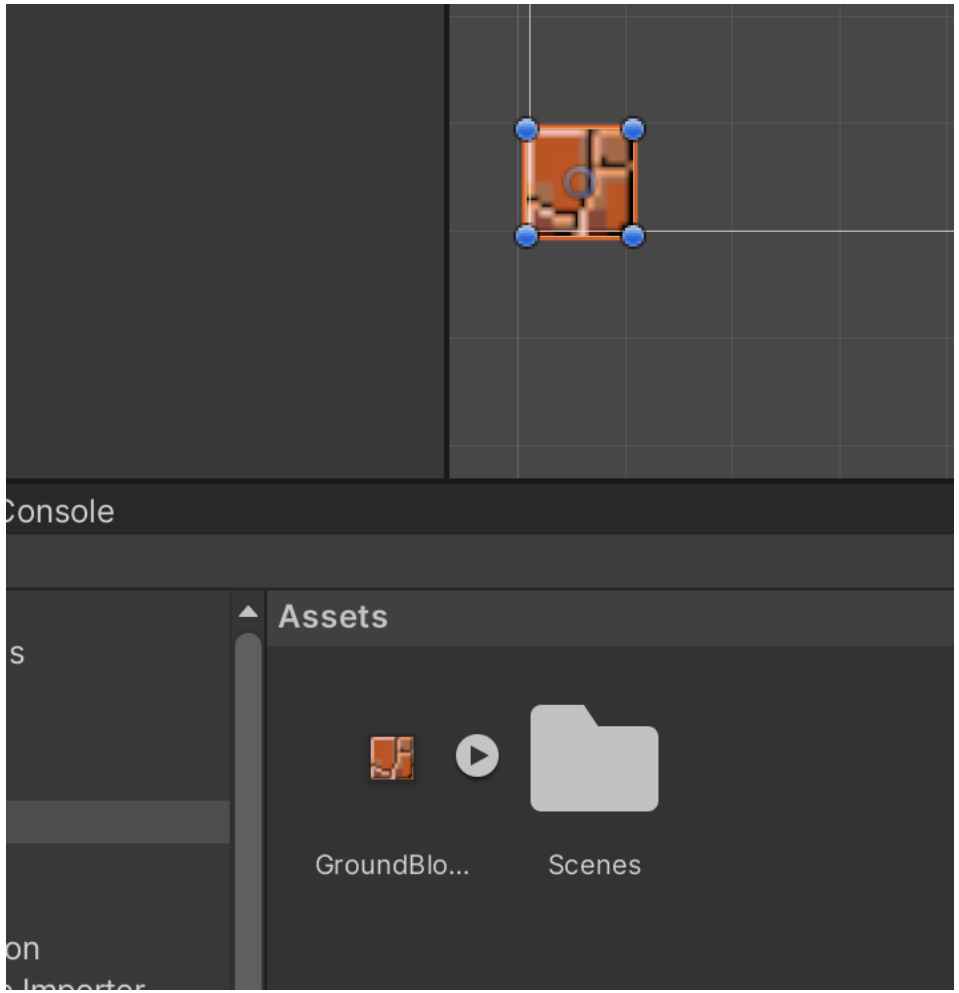
Level Design - load in Ground Block Pixels Per Unit = 16

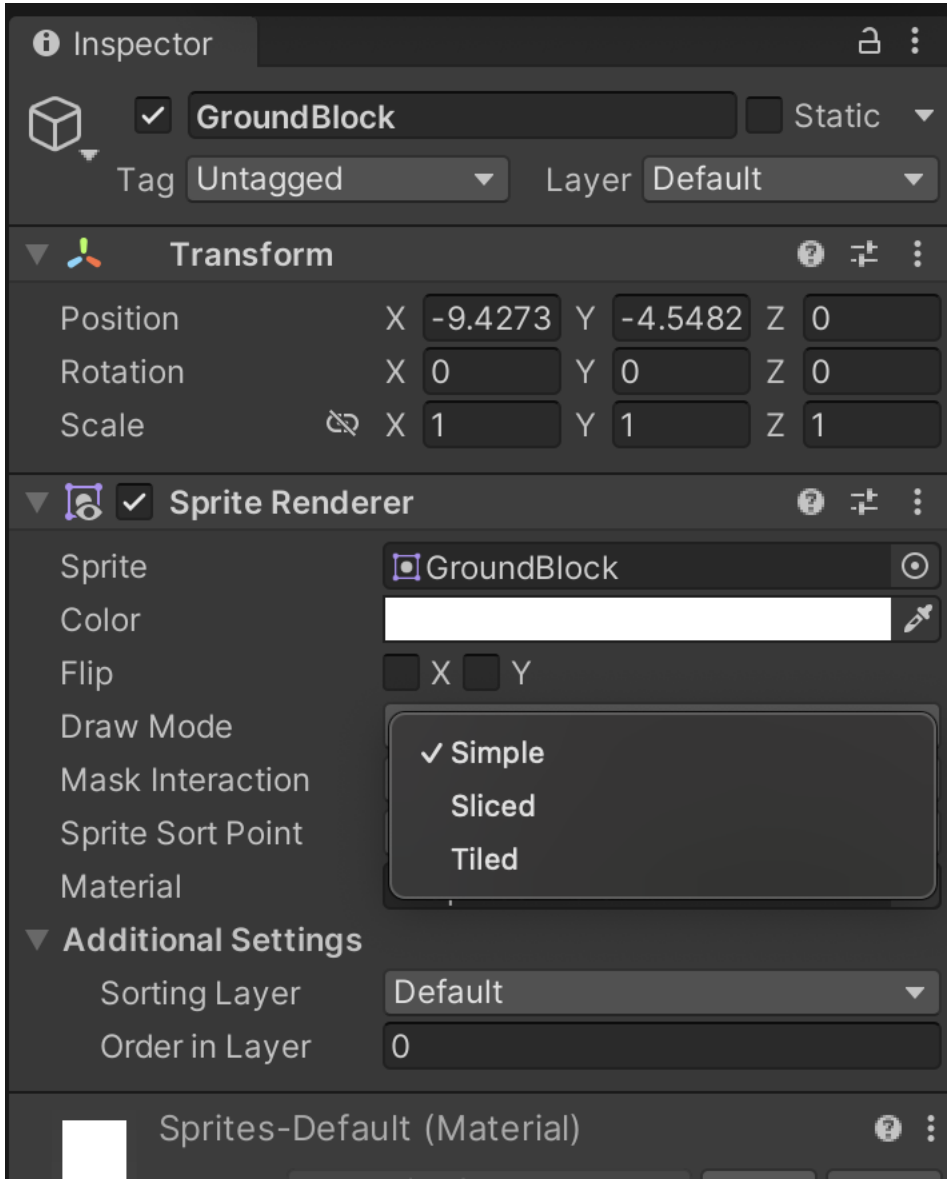




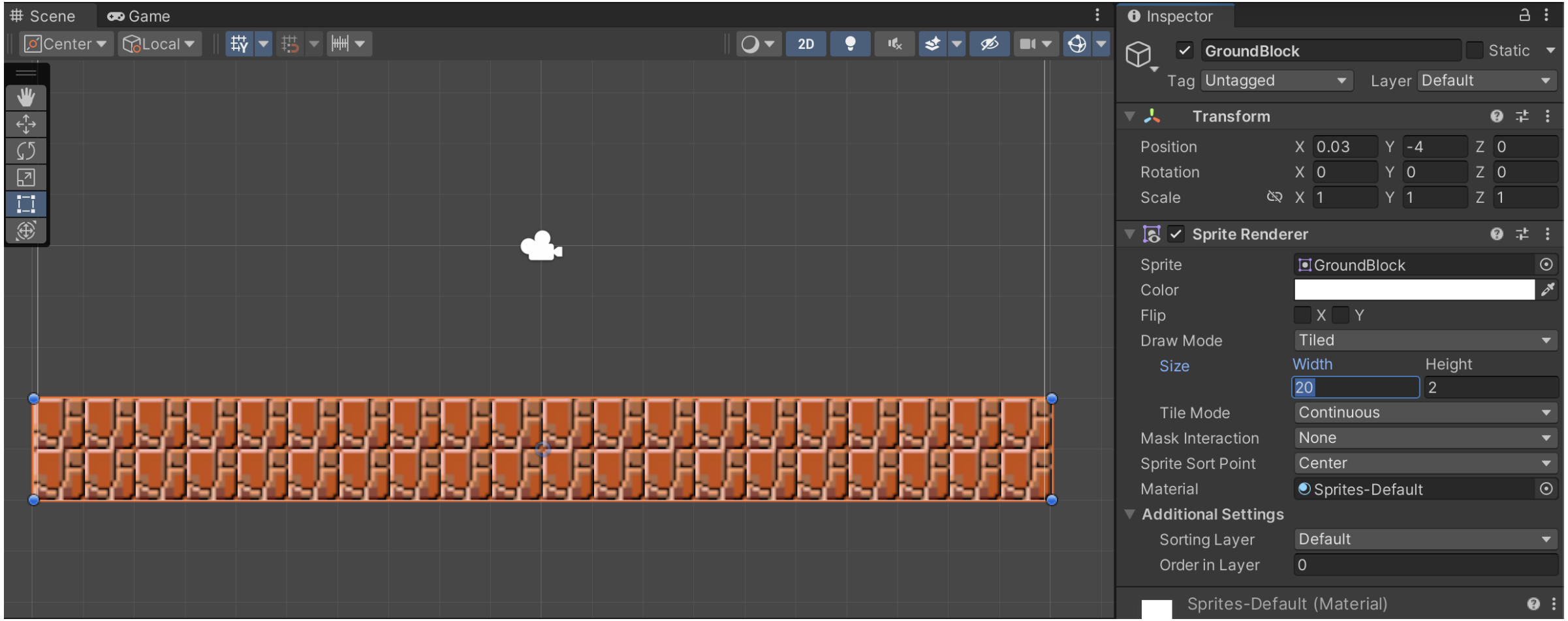


Select ‘Tiled’

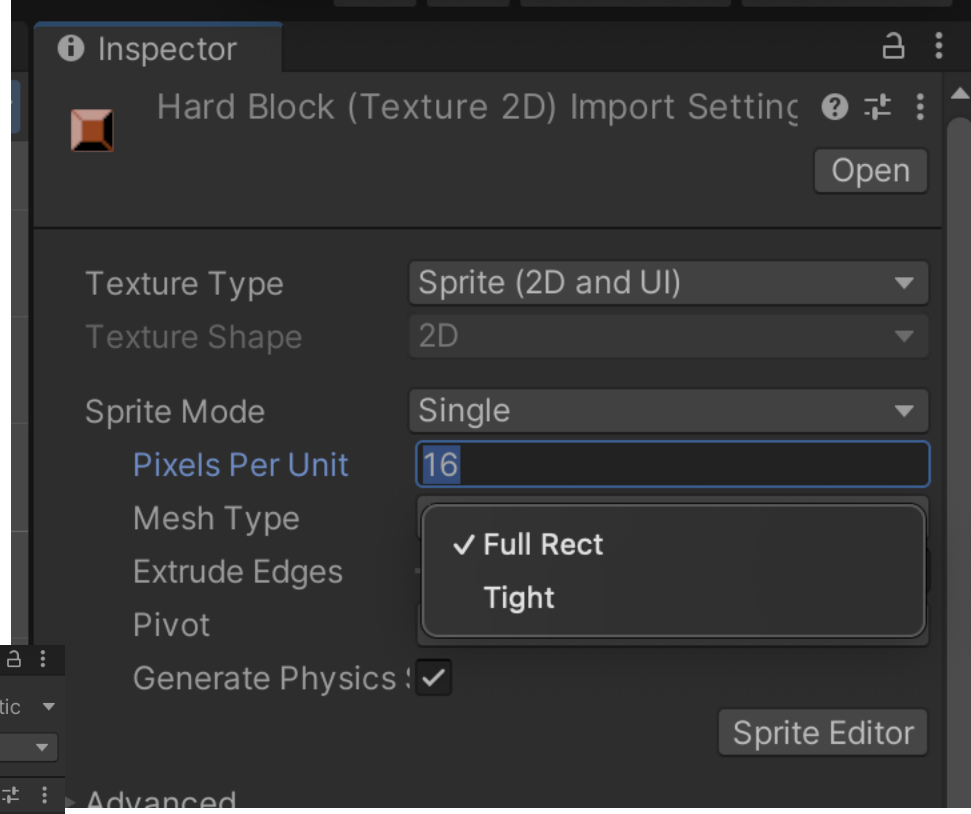


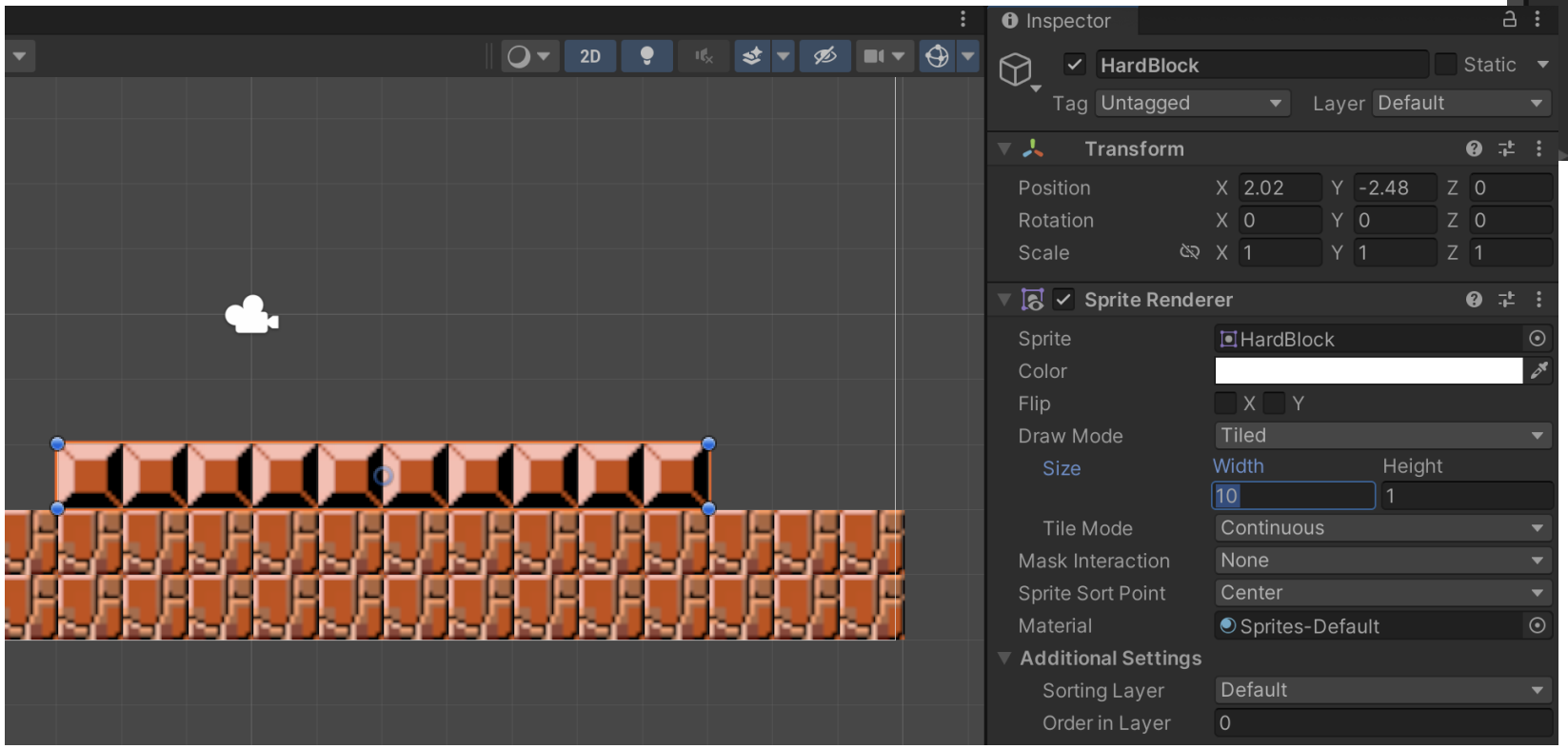


Tiled Draw Mode – 20 tiles wide x 2 high

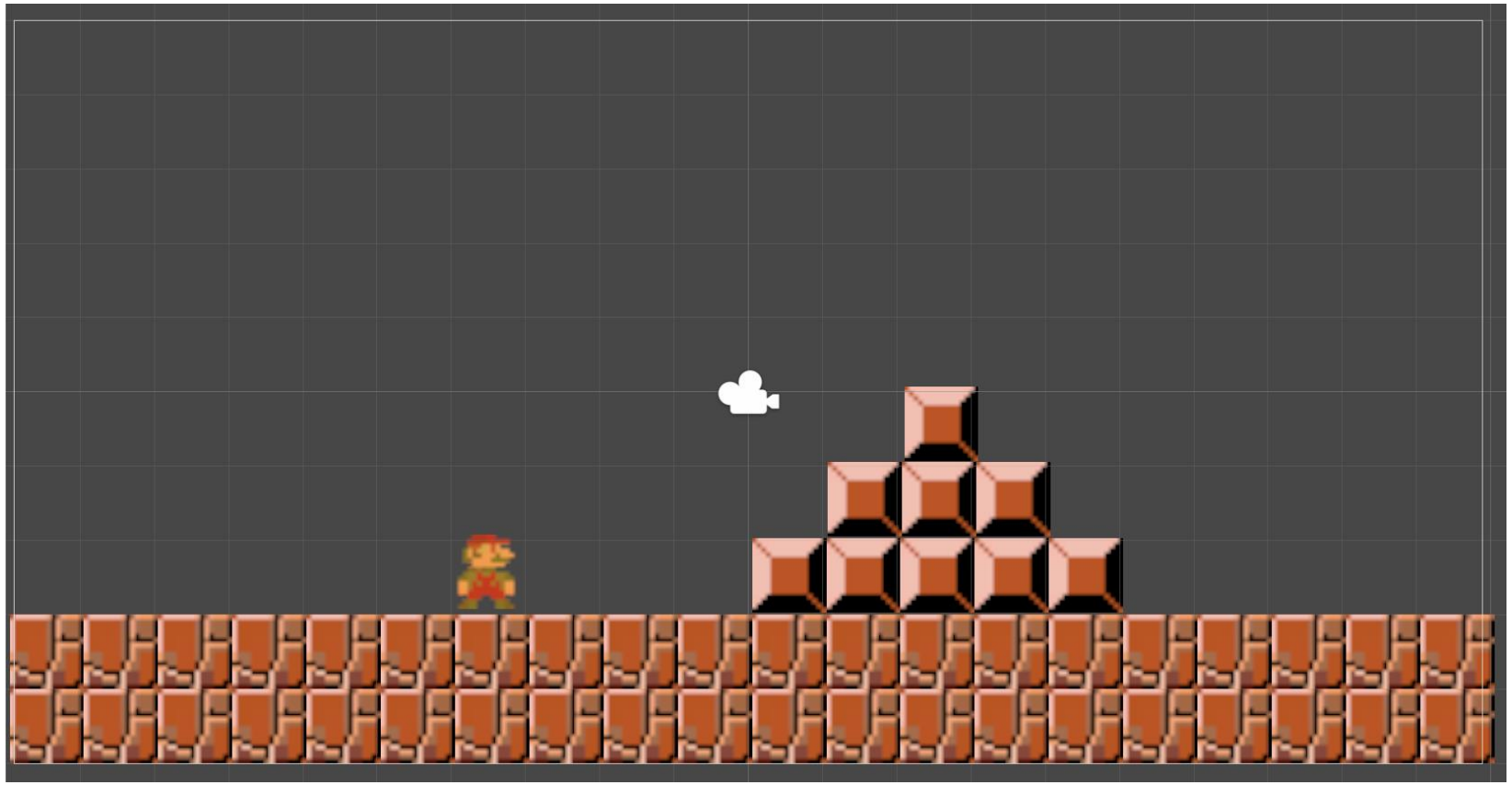


In Unity – Select Tiled





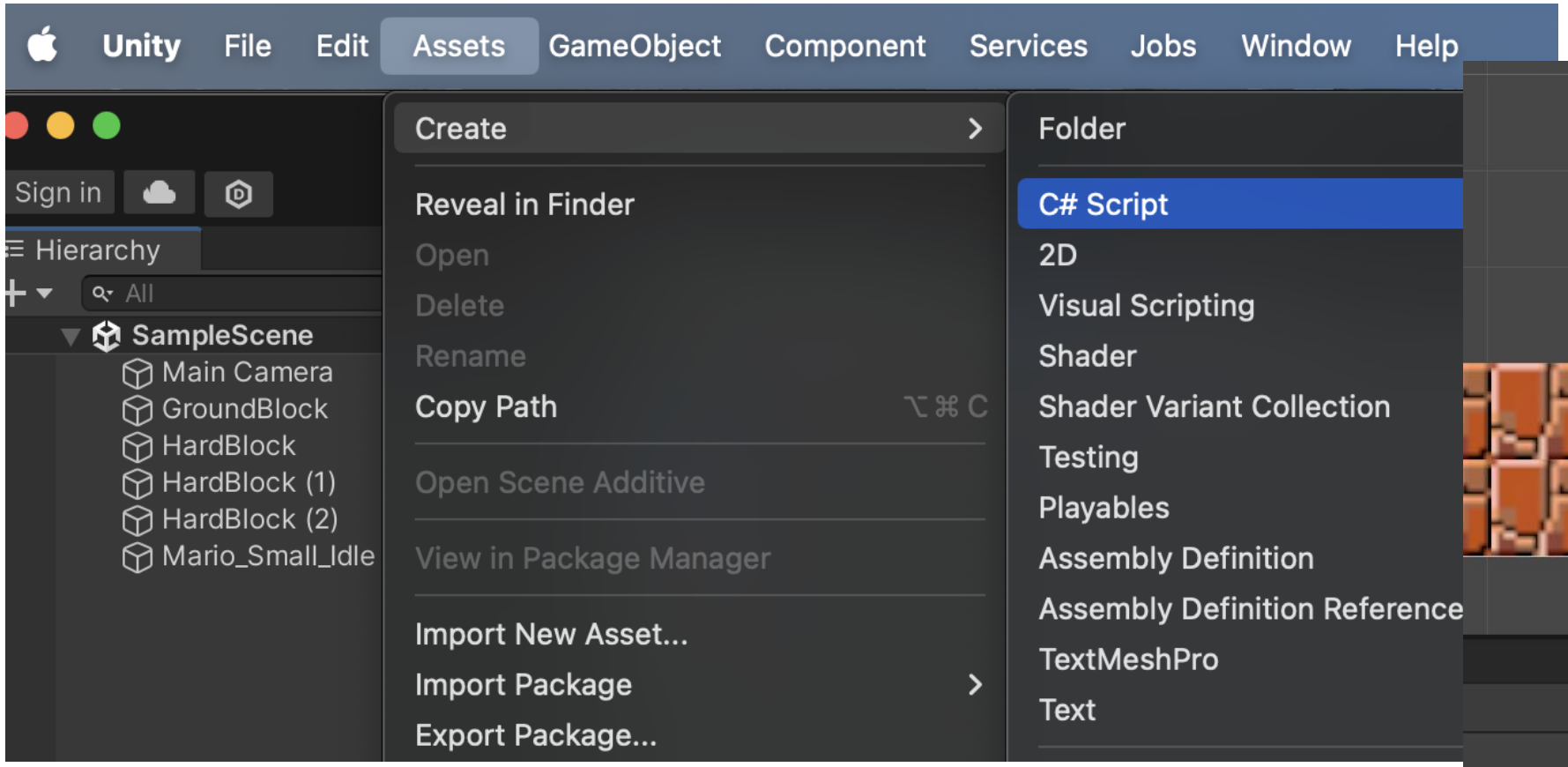
Simple Level Design

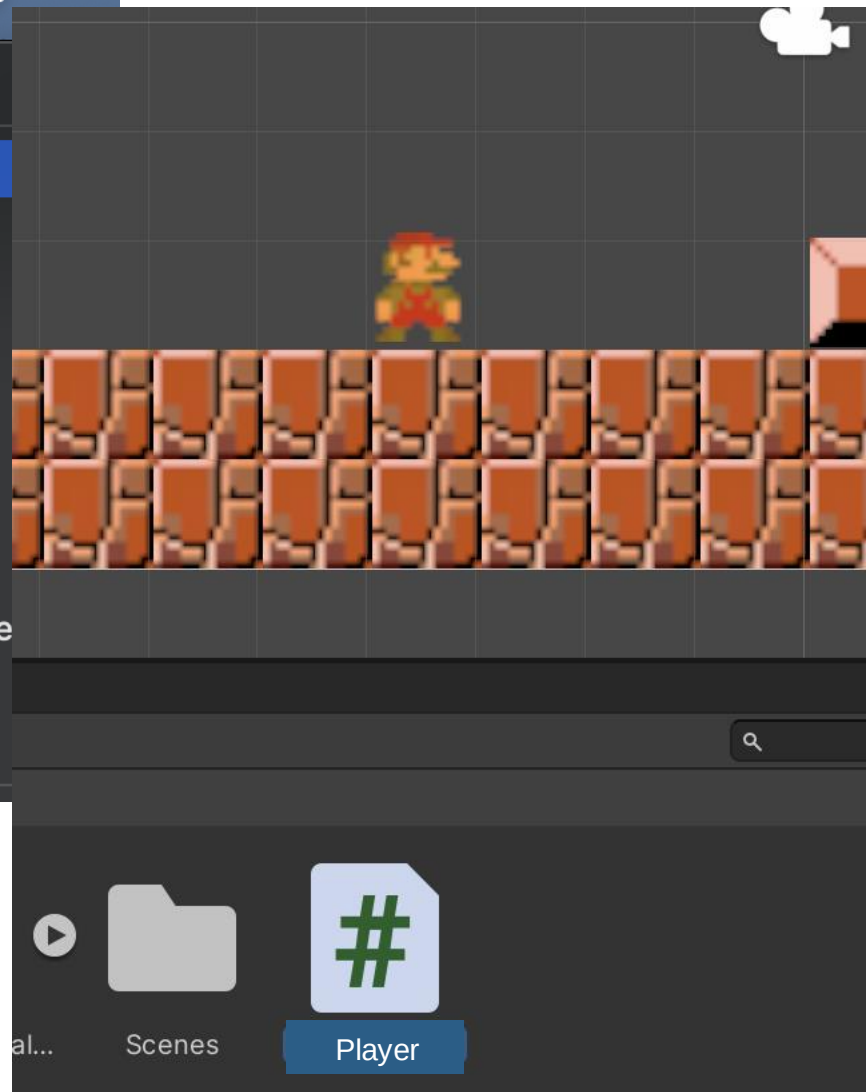


Steps / Requirements

1. Level Design in Unity ✅
2. **Mario Movement (left + right)**
3. Object detection / collision
4. Jumping over blocks / gravity
5. Enemies (Goombas) alternate directions
6. Jumping on Goombas – further collisions
7. Decoration – animation – walking
8. Camera to track Mario (once tile map created)

2. Mario Movement



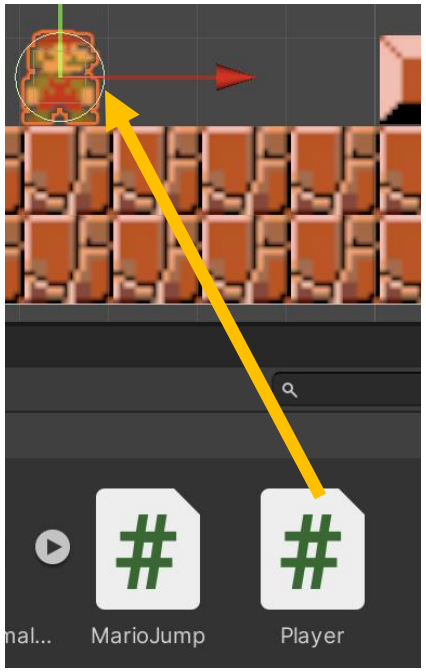


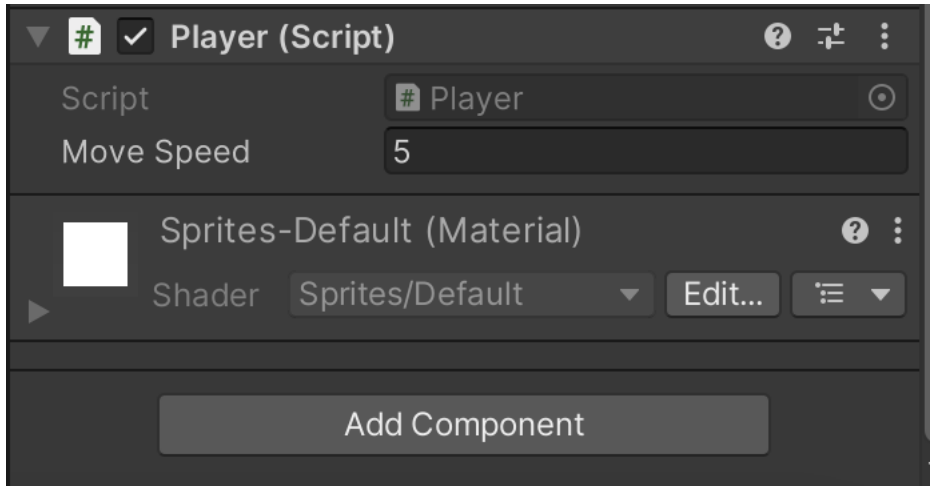
Player.cs

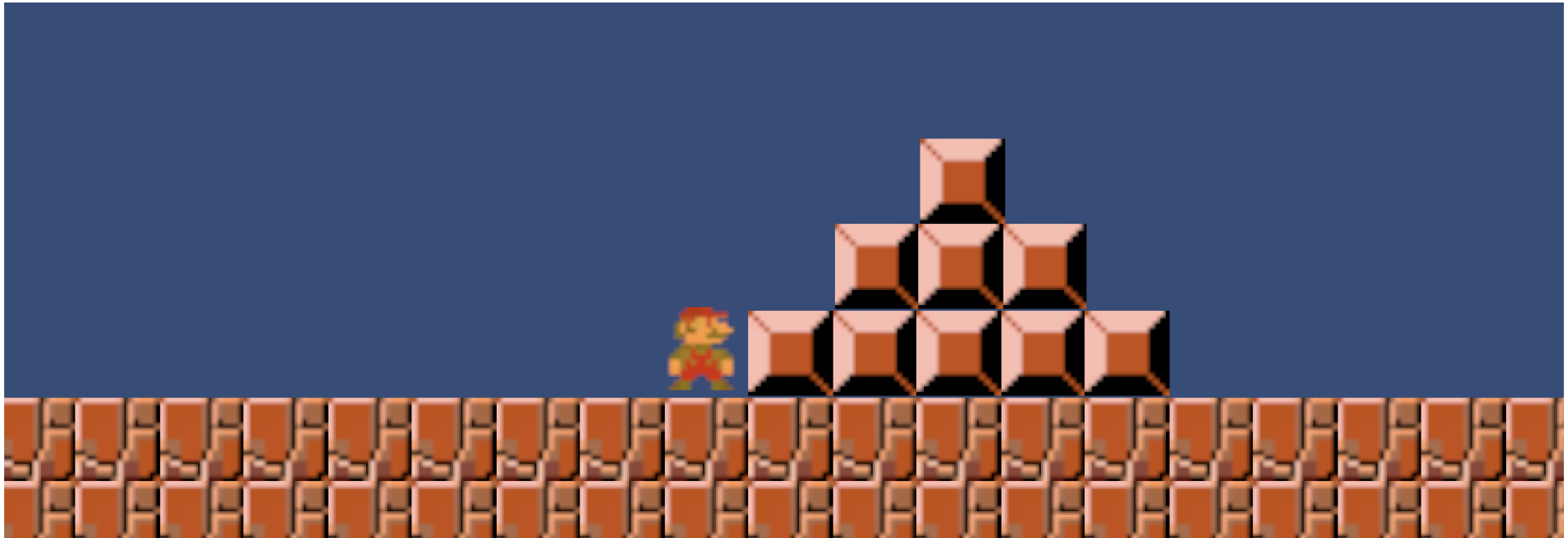
Users > nick > Documents > test_p_1 > Assets > Player.cs

```
1   using System.Collections;
2   using System.Collections.Generic;
3   using UnityEngine;
4
5   public class Player : MonoBehaviour
6   {
7       public float moveSpeed = 5f;
8
9       // Start is called before the first frame update
10      void Start()
11      {
12
13      }
14
15      // Update is called once per frame
16      void Update()
17      {
18          // Get horizontal input (-1 for left, 1 for right)
19          float horizontalInput = Input.GetAxis("Horizontal");
20
21          // Calculate movement vector
22          Vector3 movement = new Vector3(horizontalInput, 0f, 0f) * moveSpeed * Time.deltaTime;
23
24          // Move the sprite
25          transform.position += movement;
26      }
27  }
28
```

2. Mario Movement



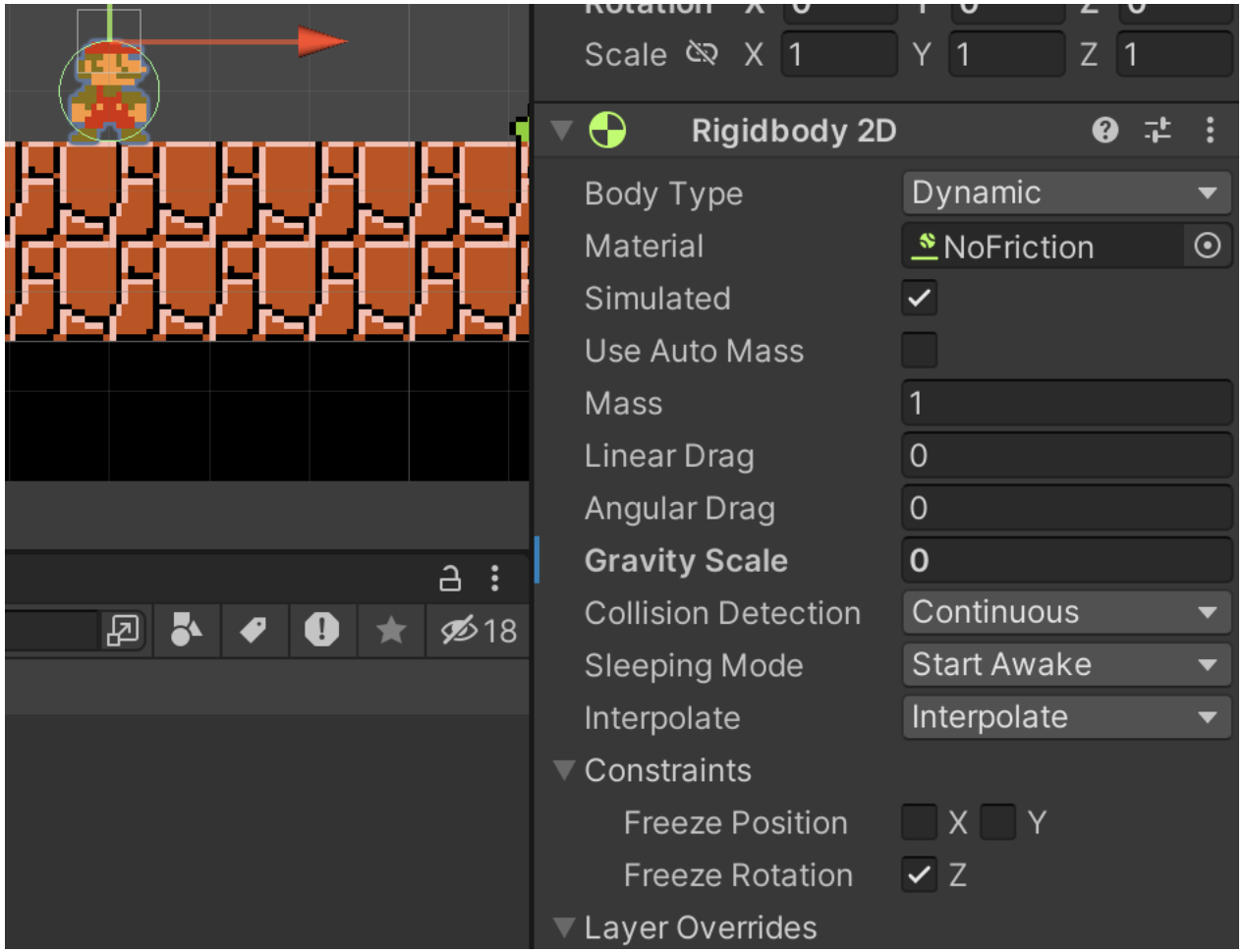


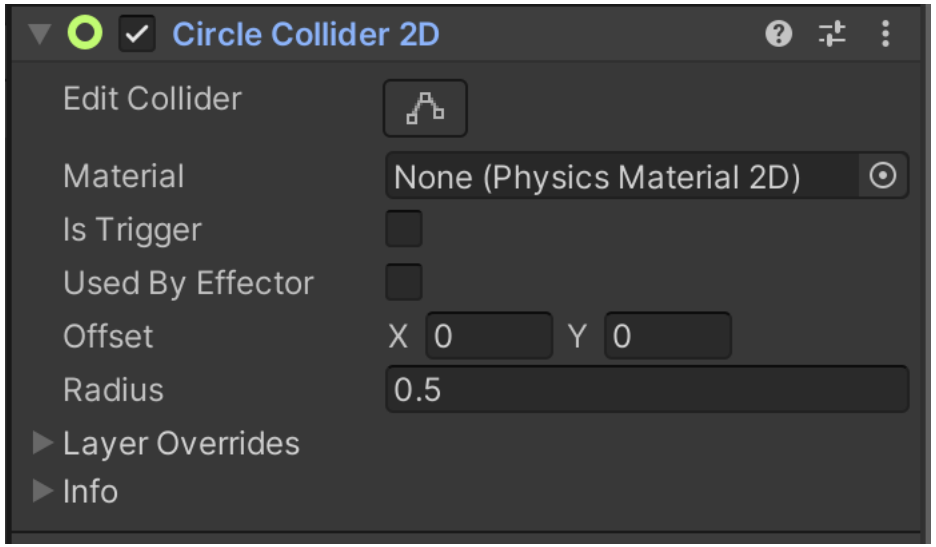


Steps / Requirements

1. Level Design in Unity ✅
2. Mario Movement (left + right) ✅
3. **Object detection / collision**
4. Jumping over blocks / gravity
5. Enemies (Goombas) alternate directions
6. Jumping on Goombas – further collisions
7. Decoration – animation – walking
8. Camera to track Mario (once tile map created)

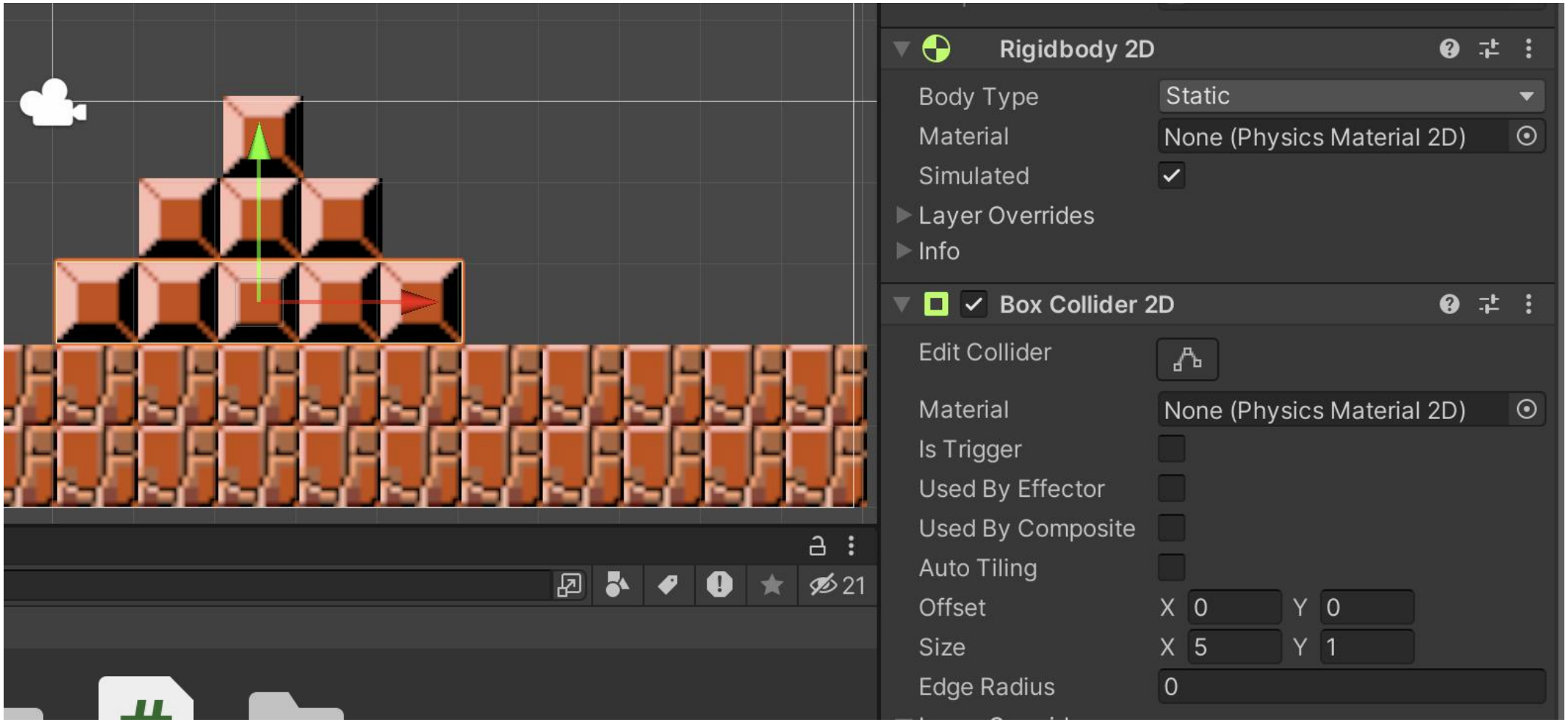
3. Object Detection – Colliders for Mario





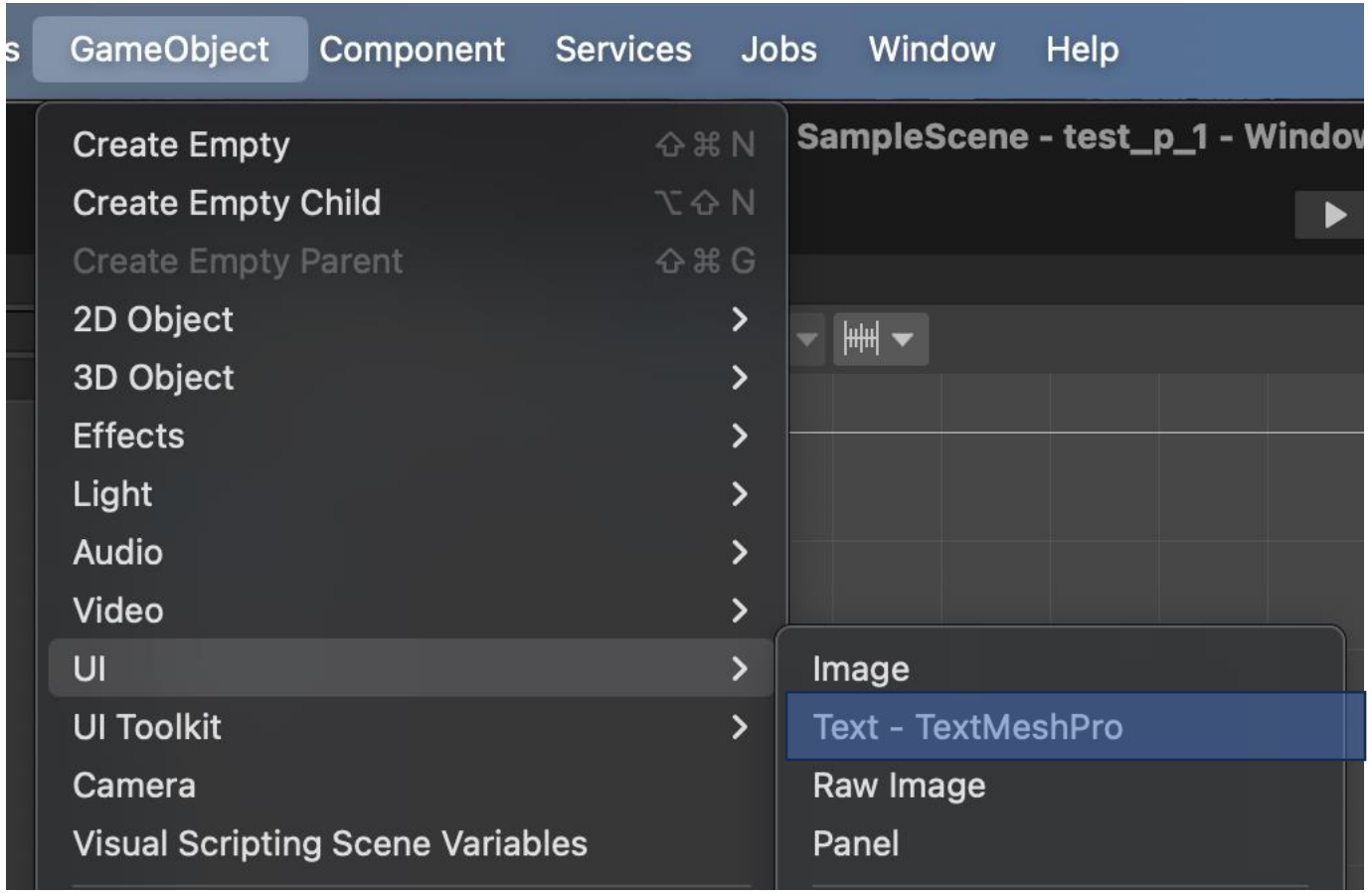
Click on Mario – add new Component -> Physics -> Rigidbody 2D
Click on Mario – add new Component -> Physics -> Circle Collider 2D

3. Object Detection – Colliders for Block



Click on Block – add new Component -> Physics -> Rigidbody 2D
Click on Block – add new Component -> Physics -> Box Collider 2D

3. Collision text – for visual cues



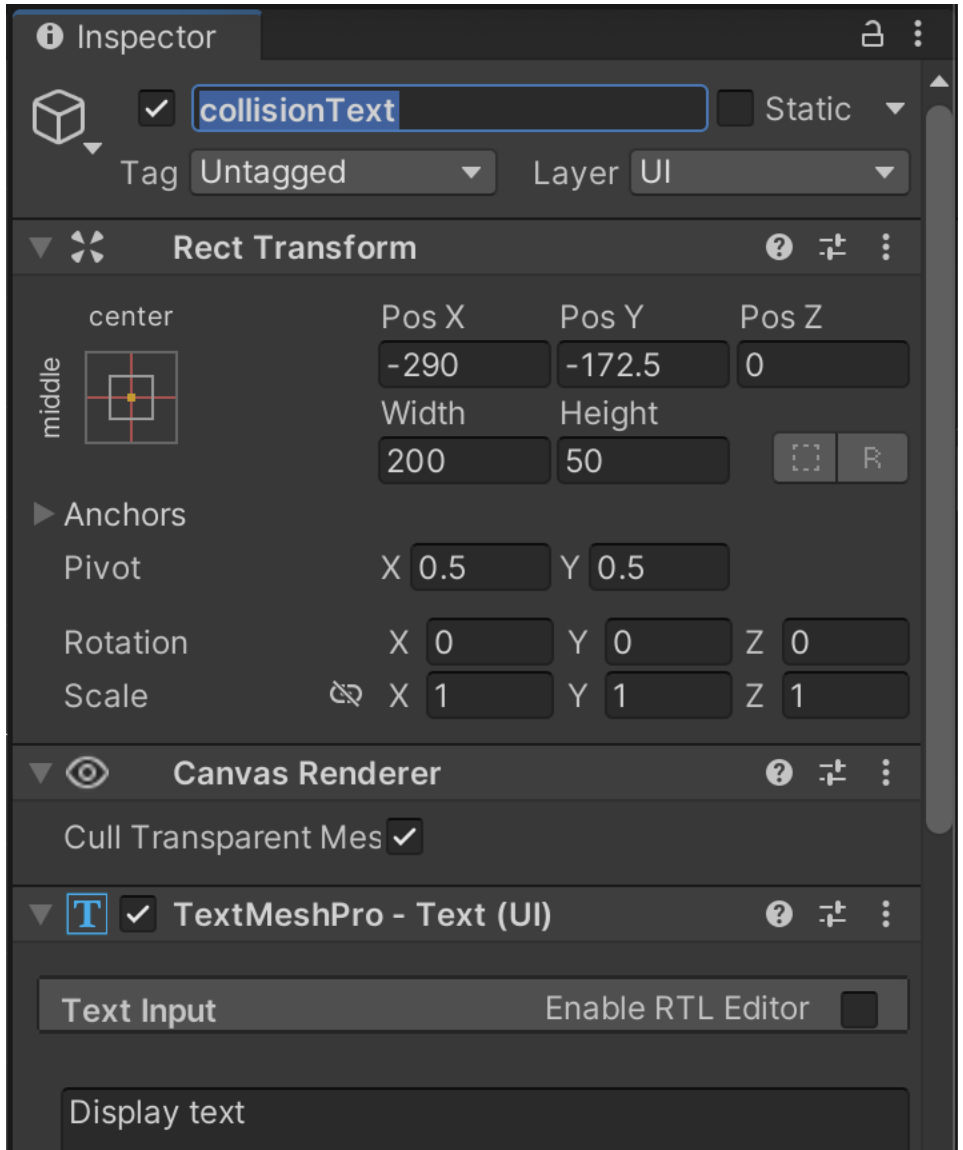
GameObject -> UI -> Text - TextMeshPro

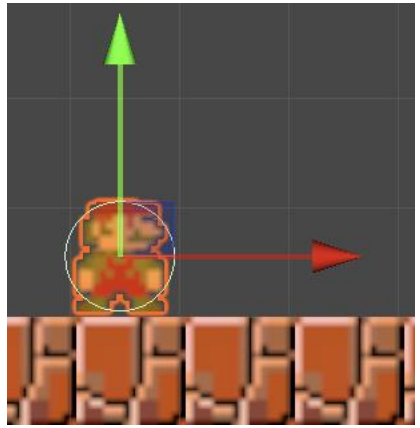
3. Collision code in our C# Script - for visual cues

```
using UnityEngine;

using TMPro; // For TextMeshPro

public class Player : MonoBehaviour
{
    public float moveSpeed = 5f;

    public TextMeshProUGUI collisionText;
```

```
void OnCollisionEnter2D(Collision2D collision)
{
    collisionText.text = "Collided with: " + collision.gameObject.name;
    // Print a message to the Console when a 2D collision occurs
    Debug.Log("Collided with: " + collision.gameObject.name);
}
```

3. Update parameters in the Inspector





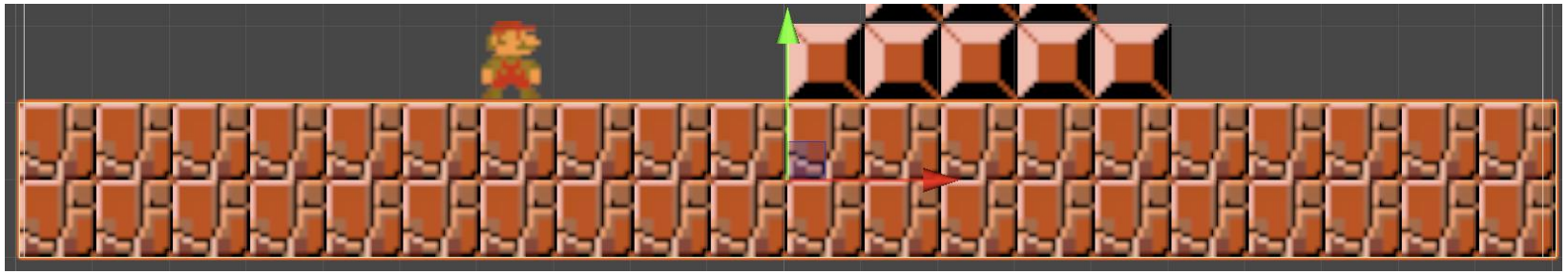
3. Change the TextMesh.text upon collision!

Steps / Requirements

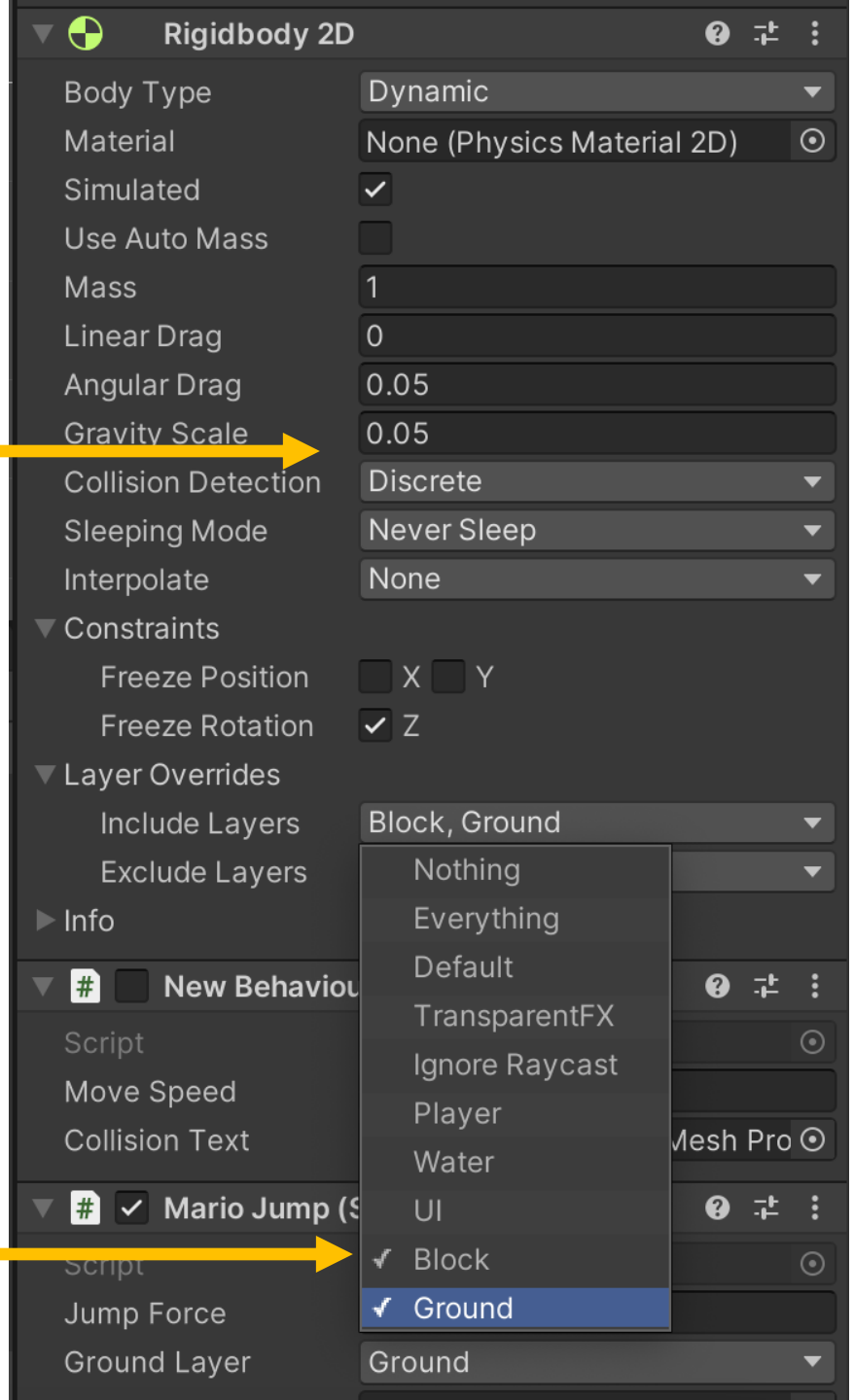
1. Level Design in Unity ✅
2. Mario Movement (left + right) ✅
3. Object detection / collision ✅
4. **Jumping over blocks / gravity**
5. Enemies (Goombas) alternate directions
6. Jumping on Goombas – further collisions
7. Decoration – animation – walking
8. Camera to track Mario (once tile map created)

4. Gravity / Jump – Mario

Start by setting Mario's RigidBody 2D Gravity to a low value > 0 but < 1 so you can check whether he falls through the floor or not…

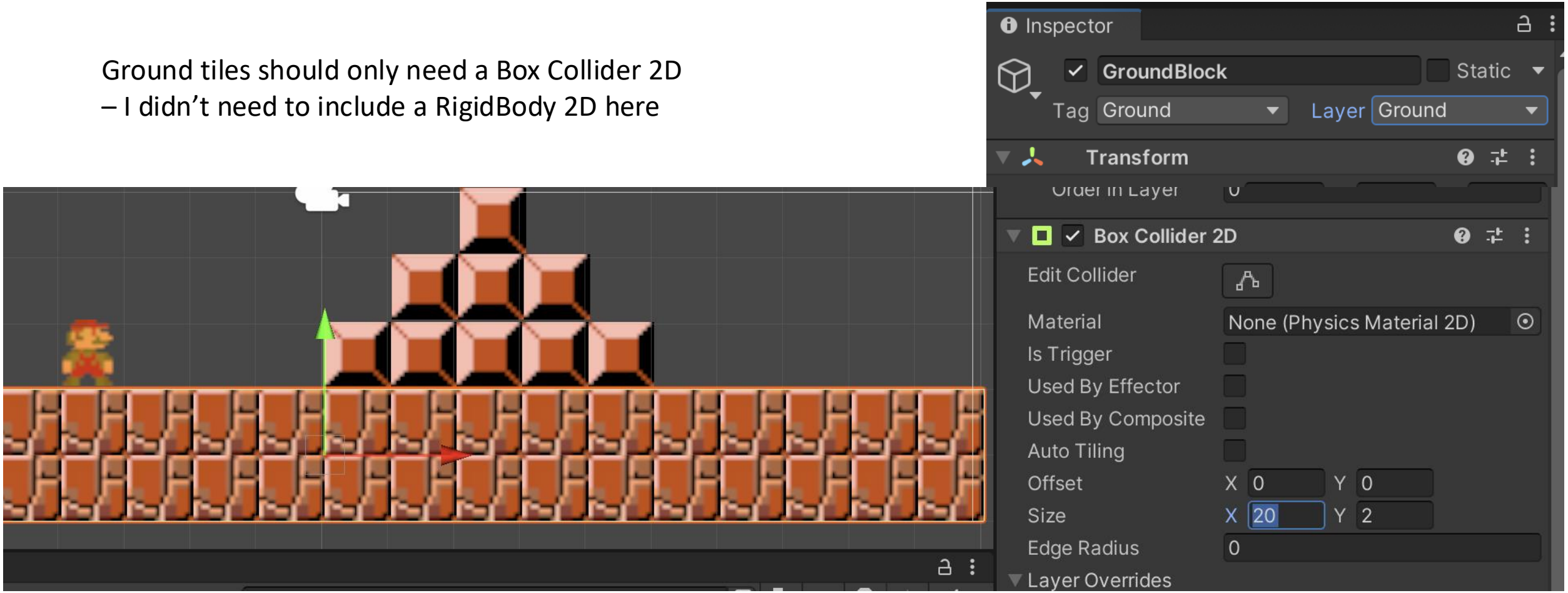


Also make sure to 'INCLUDE' Ground tiles/layers you have



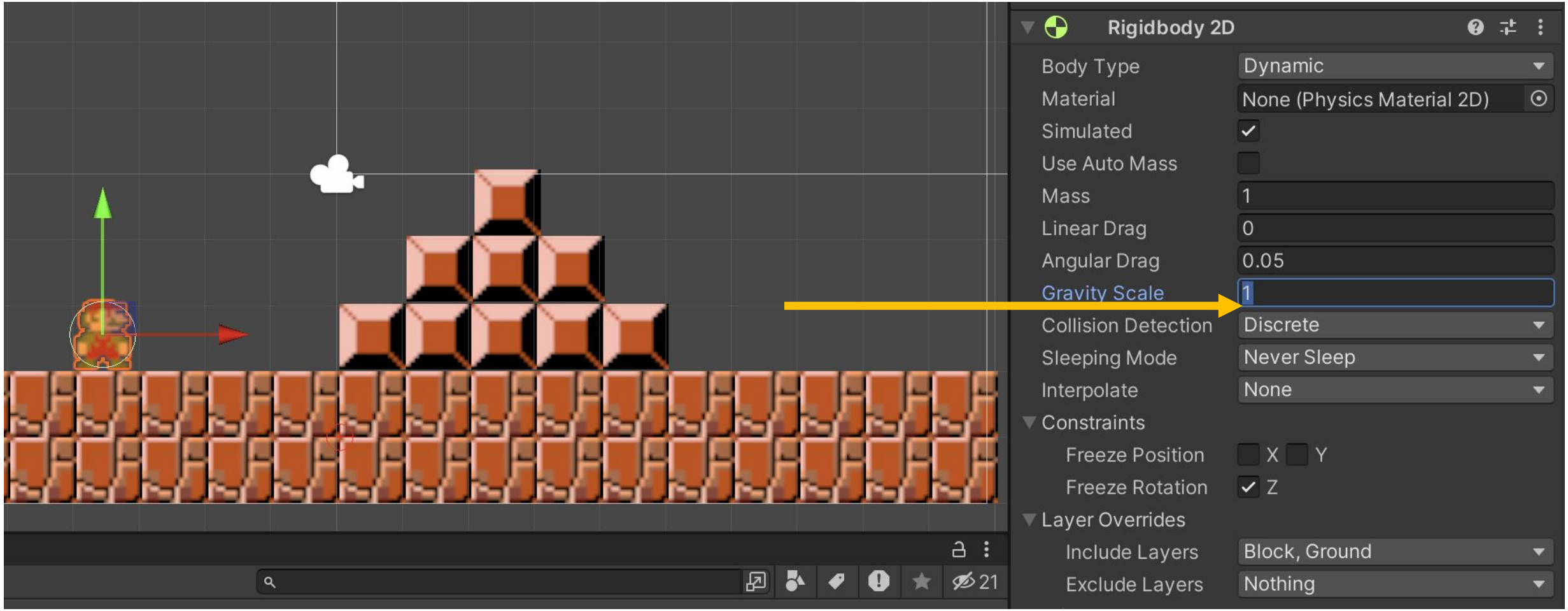
4. Gravity / Jump – Check Ground tiles

Ground tiles should only need a Box Collider 2D
– I didn't need to include a RigidBody 2D here



4. Gravity / Jump – Mario

If Mario doesn't fall through the floor – then increase the Gravity Scale to 1 (or higher)



4

Player.cs | MarioJump.cs

Users > nick > Documents > test_p_1 > Assets > MarioJump.cs

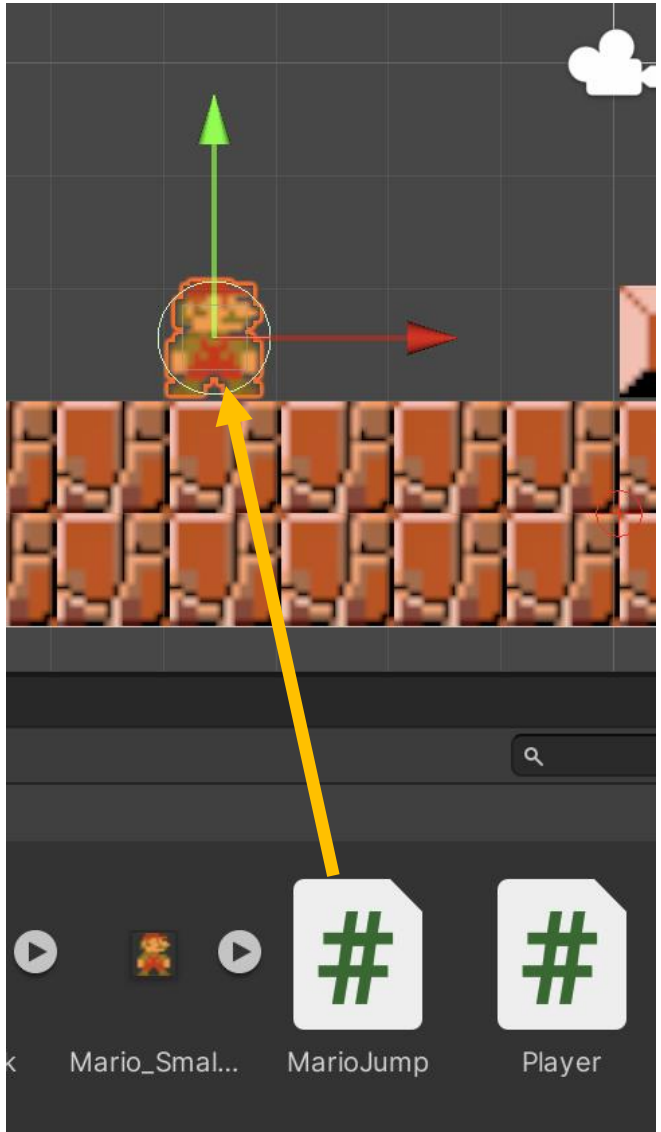
```
using UnityEngine;
using TMPro; // For TextMeshPro

public class MarioJump : MonoBehaviour
{
    private Rigidbody2D rb;
    public float jumpForce = 10f; // Force of the jump
    public LayerMask groundLayer; // Layer representing ground
    public Transform groundCheck; // Empty GameObject to check if on the ground
    public float groundCheckRadius = 0.2f; // Radius of the ground check circle

    private bool isGrounded;

    void Start()
    {
        rb = GetComponent<Rigidbody2D>();
    }

    void Update()
    {
        // Check if Mario is on the ground
        isGrounded = Physics2D.OverlapCircle(groundCheck.position, groundCheckRadius, groundLayer);

        // Jump logic
        if (Input.GetButtonDown("Jump") && isGrounded)
        {
            Jump();
        }
    }
```

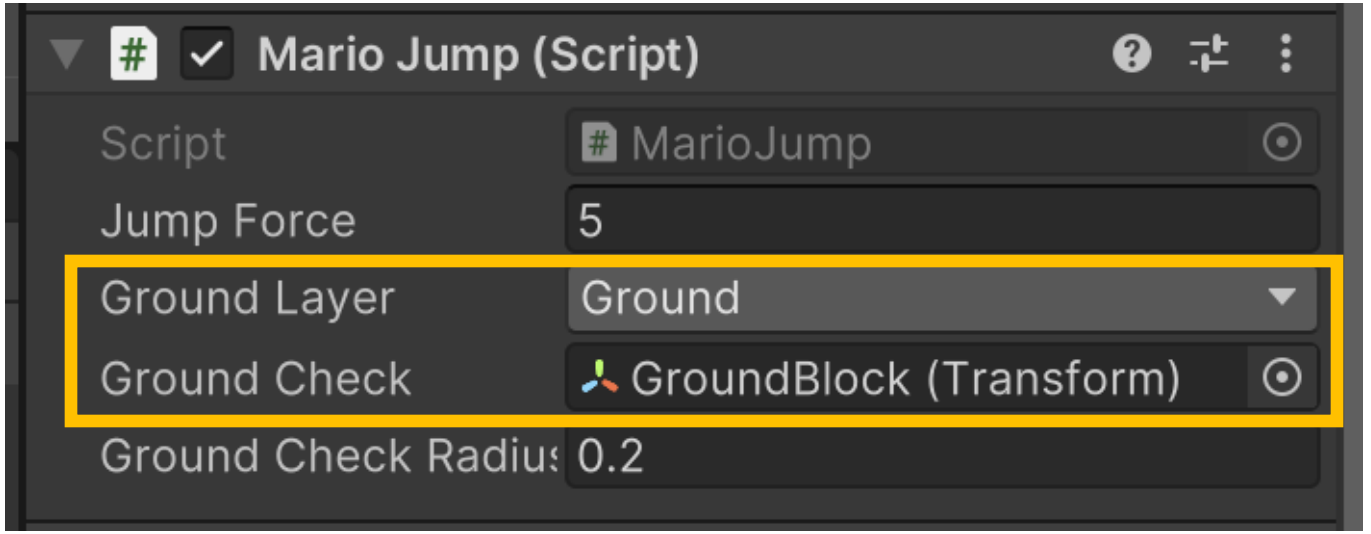
Player.cs | MarioJump.cs

Users > nick > Documents > test_p_1 > Assets > MarioJump.cs

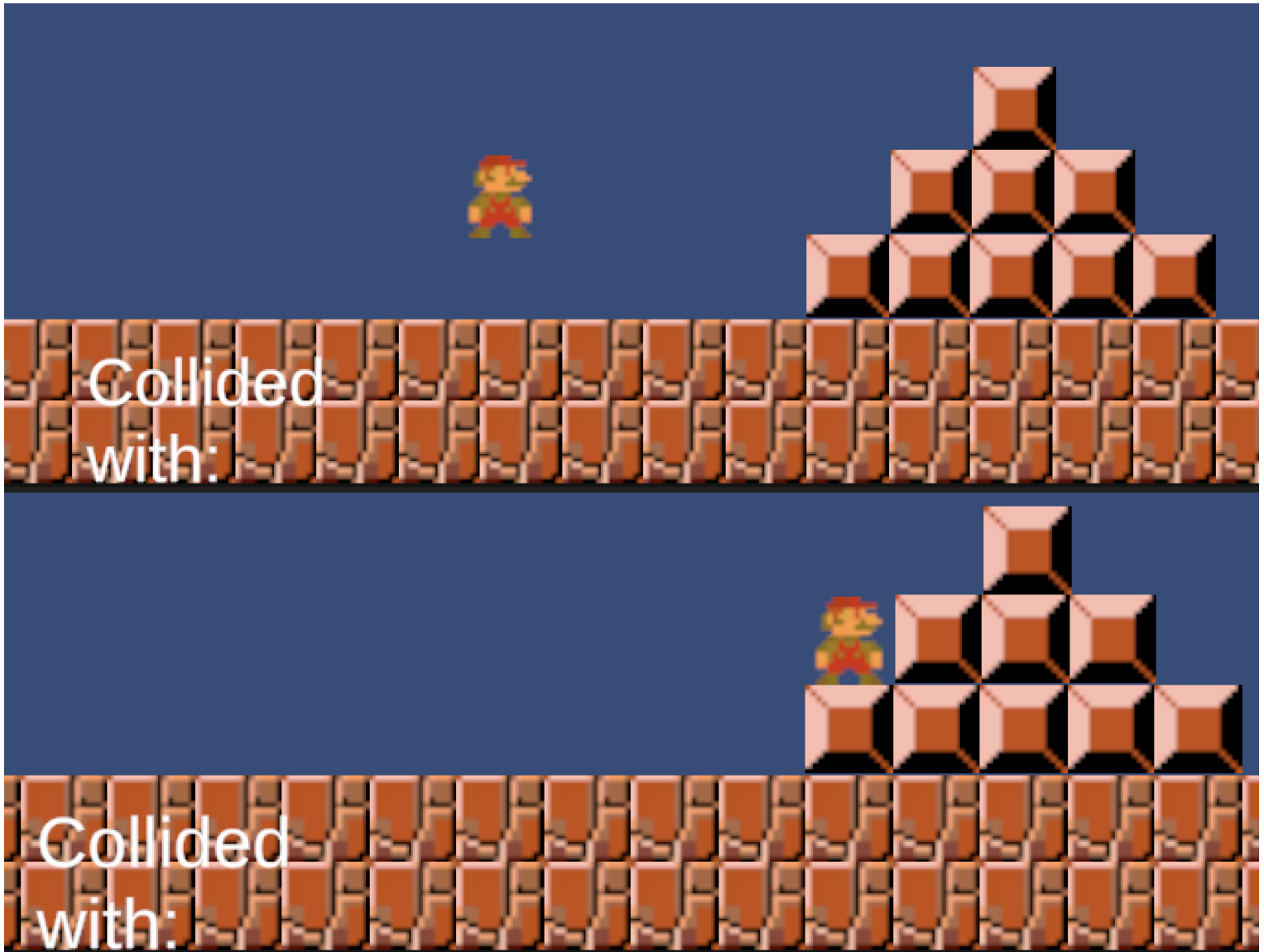
```
5   {
18
19      void Update()
20      {
21          // Check if Mario is on the ground
22          isGrounded = Physics2D.OverlapCircle(groundCheck.position, groundCheckRadius, groundLayer);
23
24          // Jump logic
25          if (Input.GetButtonDown("Jump") && isGrounded)
26          {
27              Jump();
28          }
29      }
30
31      void Jump()
32      {
33          rb.velocity = new Vector2(rb.velocity.x, jumpForce); // Apply upward velocity
34      }
35
36      private void OnDrawGizmosSelected()
37      {
38          // Draw ground check radius for debugging
39          if (groundCheck != null)
40          {
41              Gizmos.color = Color.red;
42              Gizmos.DrawWireSphere(groundCheck.position, groundCheckRadius);
43          }
44      }
45  }
```

4. Gravity / Jump – Mario





4. Gravity / Jump – Mario



Steps / Requirements

1. Level Design in Unity ✅
2. Mario Movement (left + right) ✅
3. Object detection / collision ✅
4. Jumping over blocks / gravity ✅
5. **Enemies (Goombas) alternate directions**
6. Jumping on Goombas – further collisions
7. Decoration – animation – walking
8. Camera to track Mario (once tile map created)