# Games Programming Workshop 01

Contents

[Games Programming Workshop 01 1](#_Toc117034187)

[Exercise 1 2](#_Toc117034188)

# Exercise 1

In this exercise, you will create a simple 2D Platformer In the style of Super Mario Bros, but with basic geometric objects as placeholders for more complex sprites..

1. In the Unity Hub app,

A blue screen with white text

Description automatically generated with medium confidence

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

Graphical user interface, application

Description automatically generated

1. Right click in the Assets folder in the Content Manager and choose Create > Folder

Graphical user interface, application

Description automatically generated

1. Add scripts

Graphical user interface, application, Teams

Description automatically generated

1. Create a new folder called assets and then add a 2D Square



1. Create a Prefab folder and add a Prefab for Enemy, Player and Spike
2. Add a sprite renderer component to each Prefab and set the sprite to Square
3. Set colour of each Prefab to be different

A picture containing text

Description automatically generated

1. Add Enemy Controller script to enemy Prefab
2. Add Box Collider 2D to the Enemy Prefab
3. Create a Physics folder
4. Add a Physics 2D material
5. Add this to the Box Collider 2D in Enemy

Graphical user interface, text, application

Description automatically generated

1. Add a Rigidbody 2D to the Enemy

Graphical user interface, text, application

Description automatically generated

1. Repeat all steps for Player and change the Linear Drag to 1.5

Graphical user interface, application

Description automatically generated

1. Change Transform of Spike to 3x1

Graphical user interface, application

Description automatically generated

1. Add the Spike Controller script and Box Collider 2D
2. Add a new instance of Square to the scene and call it Bounds, set the colour to dark grey
3. Add Rigidbody 2D and change the Body Type to Kinematic
4. Add a Box Collider 2D
5. Duplicate and build level

Box and whisker chart

Description automatically generated with medium confidence

1. Tag all of the Bounds with a new tag called Bound
2. Add the Orb Component. Scaled 0.5
3. Add Legacy Text UI component to the Hierarchy



1. Position the text (In game Mode)

Graphical user interface

Description automatically generated

1. Duplicate as Game Over Text

Graphical user interface

Description automatically generated

1. Open EnemyController in VS Code
2. using System.Collections;
3. using System.Collections.Generic;
4. using UnityEngine;
5. public class EnemyController : MonoBehaviour
6. {
7. float patrolTime = 2.0f; //alternate diection every tick
8. float patrolForce = 400f; //Patrol Speed
9. float patrolTimer;
10. Vector2 direction;
12. // Start is called before the first frame update
13. void Start()
14. {
15. this.direction = Vector2.left;  //Set initial direction
16. }
17. // Update is called once per frame
18. void Update()
19. {
20. patrolTimer += Time.deltaTime;
21. if(patrolTimer >= patrolTime)
22. {
23. this.GetComponent<Rigidbody2D>().velocity = Vector2.zero;
24. patrolTimer = 0;
25. direction \*= -1;
26. }
27. this.GetComponent<Rigidbody2D>().AddForce(this.direction \* patrolForce \* Time.deltaTime);
28. }
29. }
30. Open GamesceneManager in VS Code
31. using System.Collections;
32. using System.Collections.Generic;
33. using UnityEngine;
34. using UnityEngine.UI;
35. using UnityEngine.SceneManagement;
36. public class GameSceneManager : MonoBehaviour
37. {
38. public Camera MainCamera;
39. public Text ScoreText;
40. public Text GameOverText;
41. public PlayerController player;
42. int score;
43. float gameTimer;
44. bool gameOver;
46. // Start is called before the first frame update
47. void Start()
48. {
49. Time.timeScale = 1; //start time
50. player.OnHitEnemy += OnHitEnemy;
51. player.OnHitSpike += OnGameOver;
52. player.OnHitOrb += OnGameWin;
53. ScoreText.enabled = true;
54. gameOverText.enabled = false;
55. }
56. // Update is called once per frame
57. void Update()
58. {
59. MainCamera.transform.position = new Vector3(
60. Mathf.Lerp(MainCamera.transform.position.x, player.transform.position.x, Time.deltaTime \* 10),
61. Mathf.Lerp(MainCamera.transform.position.y, player.transform.position.y, Time.deltaTime \* 10),
62. MainCamera.transform.position.z
63. );
64. if(gameOver) {
65. if(Input.GetKeyDown("r")) {
66. SceneManager.LoadScene (SceneManager.GetActiveScene().name);
67. }
68. return; // Skip the following lines if GameOver
69. }
70. ScoreText.text = "Score: " + score;
71. if(player.transform.position.y < -10)
72. OnGameOver();
73. }
74. void OnHitEnemy() {
75. this.score += 100;
76. }
77. void OnGameOver() {
78. gameOver = true;
79. ScoreText.enabled = false;
80. GameOverText.enabled = true;
81. GameOverText.text = "Game Over!\nPress R to Resart";
82. Time.timeScale = 0; //stop time
83. }
84. void OnGameWin() {
85. gameOver = true;
86. ScoreText.enabled = false;
87. GameOverText.enabled = true;
88. GameOverText.text = "Stage Clear!\nPress R to Restart";
89. Time.timeScale = 0; //stop time
90. }
91. }
92. Open PlayerController in VSCode

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class PlayerController : MonoBehaviour

{

    public delegate void OnHitSpikeAction();

    public delegate void OnHitEnemyAction();

    public delegate void OnHitOrbAction();

    public OnHitEnemyAction OnHitEnemy;

    public OnHitSpikeAction OnHitSpike;

    public OnHitOrbAction OnHitOrb;

    float speed = 1000;

    float jumpSpeed = 400;

    Vector3 leftBound;

    Vector3 rightBound;

    bool canJump;

    // Start is called before the first frame update

    void Start()

    {

    }

    // Update is called once per frame

    void Update()

    {

        ProcessInput();

    }

    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("space") || Input.GetKeyDown("w"))

        {

            Jump();

        }

    }

    void Jump(bool force = false)

    {

        if(canJump || force)

        {

            canJump = false;

            this.GetComponent<Rigidbody2D>().AddForce(Vector2.up \* jumpSpeed);

        }

    }

    public void OnCollisionEnter2D(Collision2D collision)

    {

        if(collision.gameObject.tag == "Bound")

        {

            canJump = true;

        }

        if(collision.gameObject.GetComponent<SpikeController>() != null)

        {

            if(OnHitSpike != null)

            {

                OnHitSpike();

            }

        }

        else if(collision.gameObject.GetComponent<EnemyController>() != null)

        {

            EnemyController enemy = collision.gameObject.GetComponent<EnemyController>();

            if(this.transform.position.y > enemy.transform.position.y + enemy.GetComponent<BoxCollider2D>().size.y / 2)

            {

                GameObject.Destroy(collision.gameObject);

                Jump(true);

                if (OnHitEnemy != null)

                {

                    OnHitEnemy();

                }

            }

            else {

                if(OnHitSpike != null)

                {

                    OnHitSpike();

                }

            }

        } else if (collision.gameObject.GetComponent<OrbController>() != null) {

            if(OnHitOrb != null)

            {

                OnHitOrb();

            }

        }

    }

}

1. Delete contents of SpikeController

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class SpikeController : MonoBehaviour

{

}

1. Add a new Empty GameObject to the Scene Hierarchy and drag the Main Camera into it.

Graphical user interface, text, application

Description automatically generated

1. Add the Game Scene Manager script component to the Scene Manager object



1. Set the Public Properties to the objects in your scene

Graphical user interface, application

Description automatically generated

Extensions

* Experiment with the physics settings in the Player Rigidbody 2D Component (Such as drag)
* Add a rotator component to the Orb

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class Rotator : MonoBehaviour

{

    public float speed = 200;

    // Update is called once per frame

    void Update()

    {

        this.transform.localEulerAngles = new Vector3(

            this.transform.localEulerAngles.x,

            this.transform.localEulerAngles.y + speed \* Time.deltaTime,

            this.transform.localEulerAngles.z

        );

    }

}

* Expand the level design