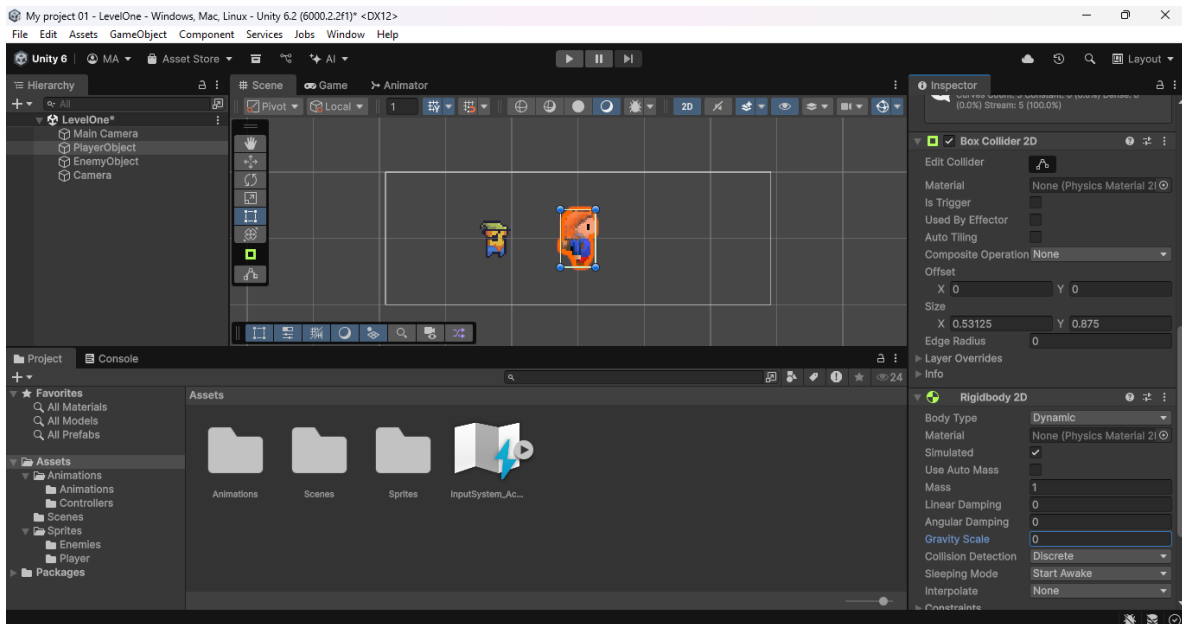
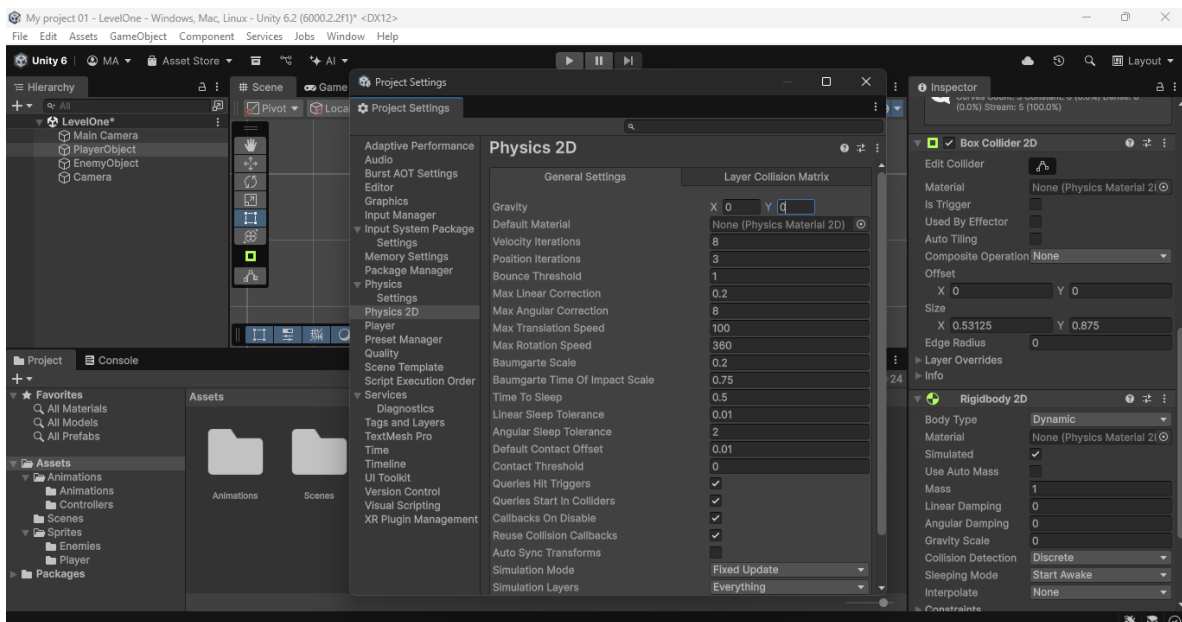


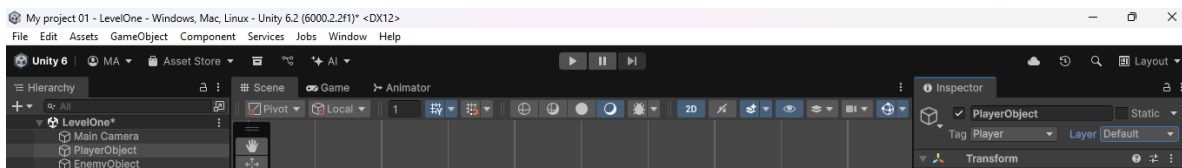
Se agrega Box Collider 2D y Rigidbody 2D para player y enemy



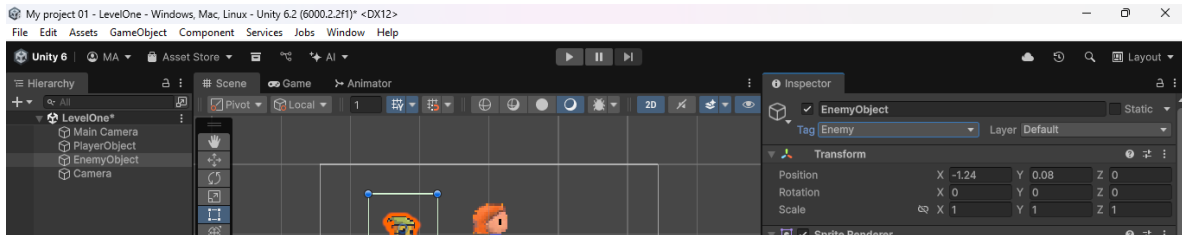
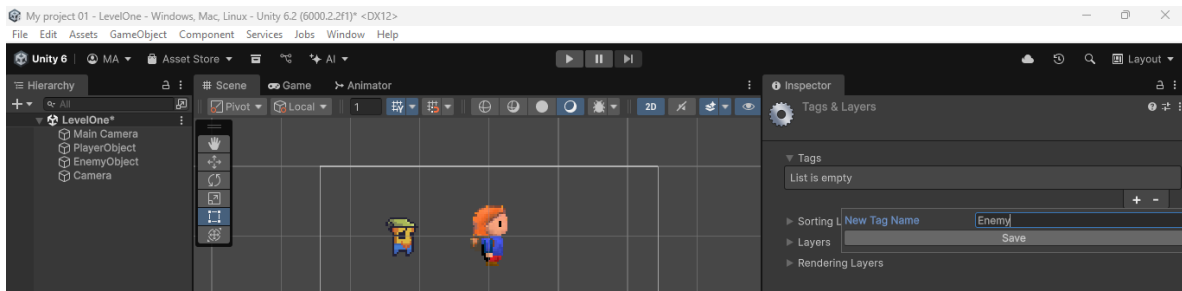
Cambia Gravity Y de -9.81 a 0 (porque usamos vista top-down)



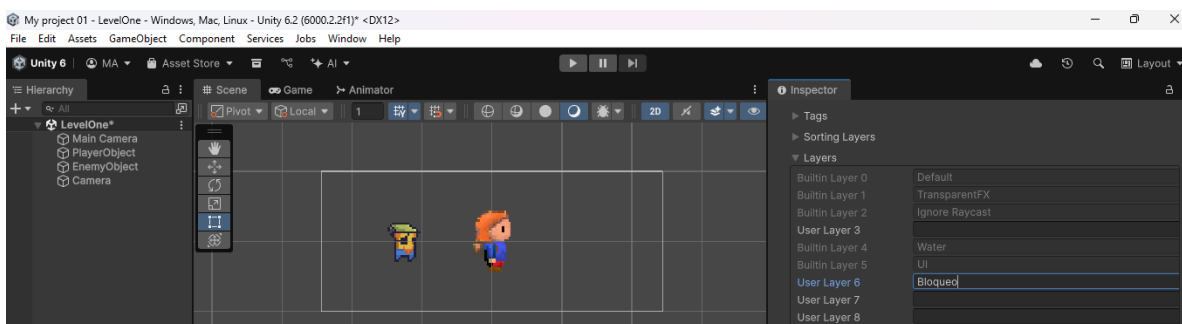
En el Inspector arriba a la izquierda, en el menú Tag, selecciona "Player"



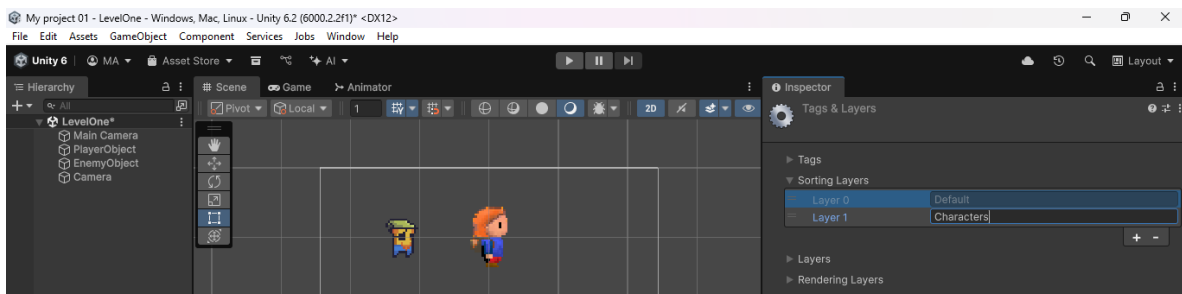
Crear una etiqueta a enemy y nombrarla “enemy”



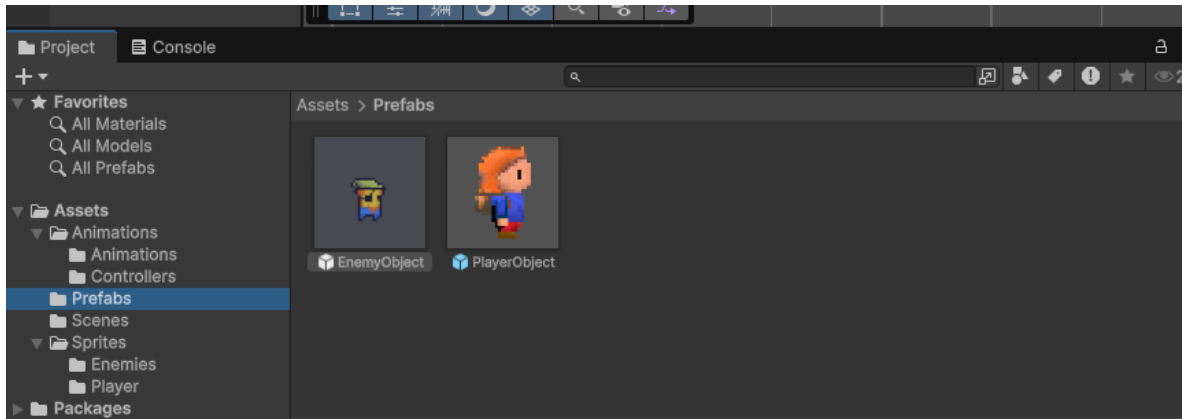
A player se le agrago una capa llamada “bloqueo”



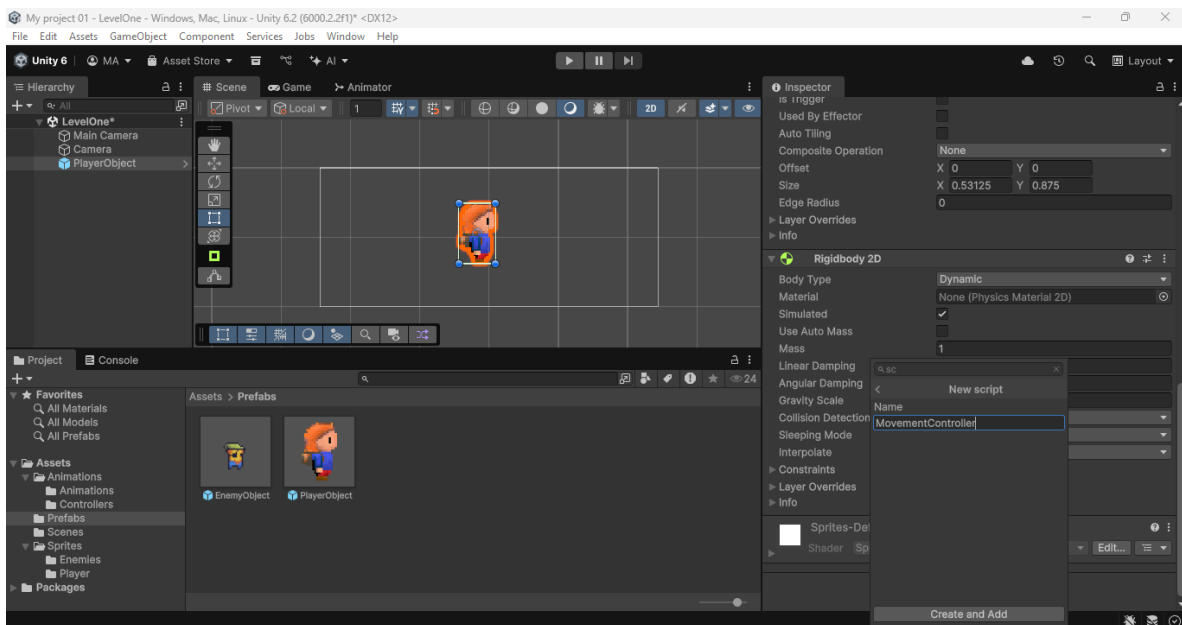
Crea una capa llamada "Characters" y agregarla para player y enemy



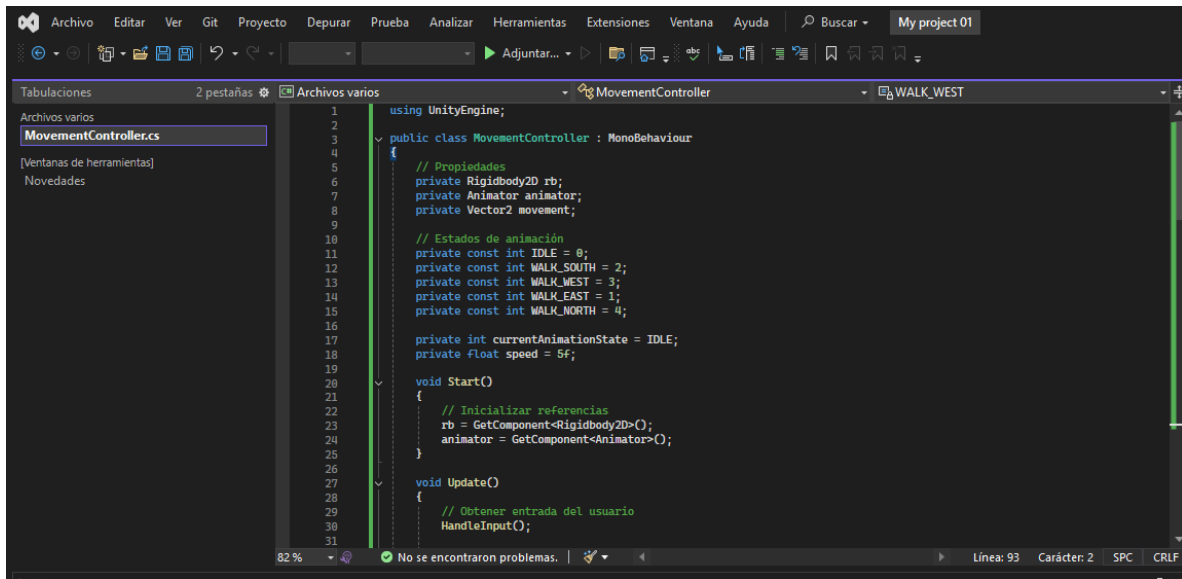
Se creo la carpeta prefabs y se arrastraron los playerObject y enemyObject y eliminar las que están en Hierarchy



Se vuelven arrastrar player y enemy a hierarchy y crear un scrip para cada uno

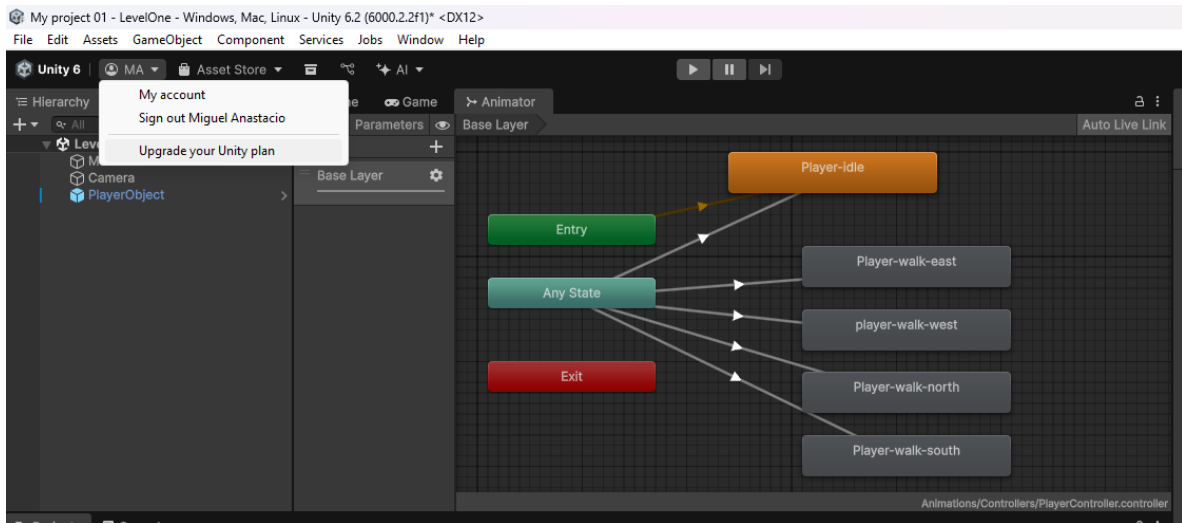


Programar el movementController.cs

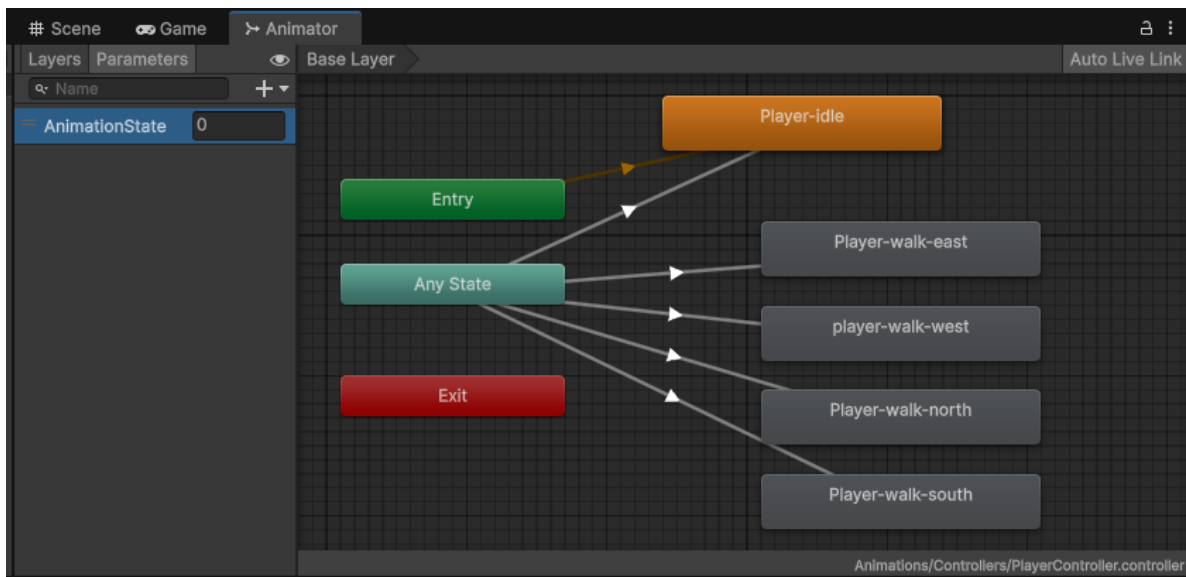


The screenshot shows the Visual Studio IDE with the 'MovementController.cs' script open. The script is a C# class that inherits from 'MonoBehaviour'. It includes properties for 'Rigidbody2D', 'Animator', and 'Vector2'. It also defines constants for animation states: 'IDLE', 'WALK_SOUTH', 'WALK_WEST', 'WALK_EAST', and 'WALK_NORTH'. The 'Start()' method initializes the 'Rigidbody2D' and 'Animator' components. The 'Update()' method calls 'HandleInput()'.

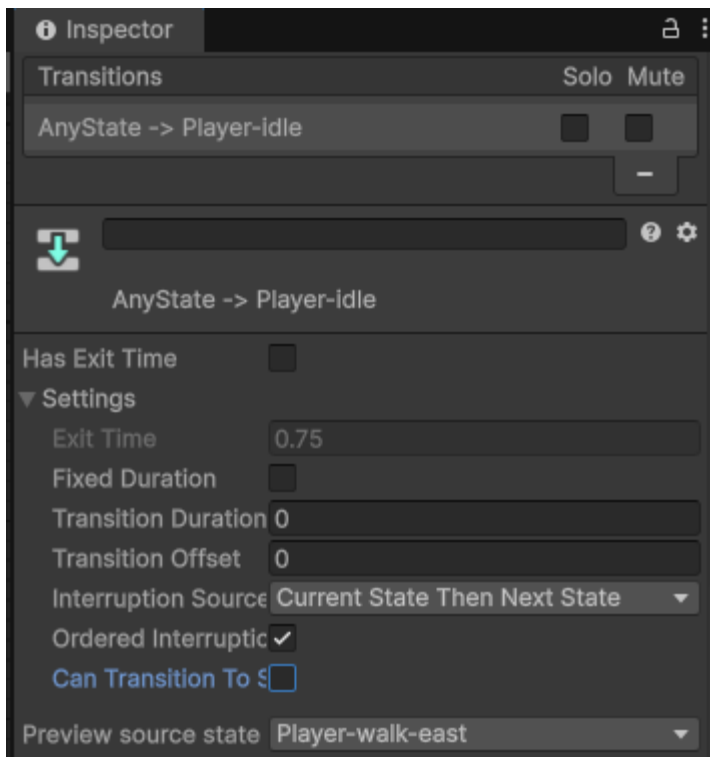
```
1 using UnityEngine;
2
3 public class MovementController : MonoBehaviour
4 {
5     // Propiedades
6     private Rigidbody2D rb;
7     private Animator animator;
8     private Vector2 movement;
9
10    // Estados de animación
11    private const int IDLE = 0;
12    private const int WALK_SOUTH = 2;
13    private const int WALK_WEST = 3;
14    private const int WALK_EAST = 1;
15    private const int WALK_NORTH = 4;
16
17    private int currentAnimationState = IDLE;
18    private float speed = 5f;
19
20    void Start()
21    {
22        // Inicializar referencias
23        rb = GetComponent<Rigidbody2D>();
24        animator = GetComponent<Animator>();
25    }
26
27    void Update()
28    {
29        // Obtener entrada del usuario
30        HandleInput();
31    }
```



Se creo un parámetro int con el nombre AnimationState y para cada uno



Y para cada flecha blanca al seleccionar pondremos estos parámetros



Para cada una se pondrá una condición que es la siguiente

Transición	Condición
Any State to player-walk-east	AnimationState Equals = 1
Any State to player-walk-west	AnimationState Equals = 3
Any State to player-walk-north	AnimationState Equals = 4
Any State to player-walk-south	AnimationState Equals = 2
Any State to player-idle	AnimationState Equals = 0