Universidad Nacional de San Agustin
Facultad de Ingenieria, Produccion y Servicios
Escuela Profesional de Ciencia de la Computacion

ANIMACIONES

Integrantes: Kelvin Paul Pucho Zevallos
Angelo Aldo Perez Rodriguez
Luis Armando Sihuinta Perez
Josnick Chayña Batallanes

Profesor: Diego Alonso Iquira Becerra

Fecha de realización: 7 de junio de 2022
Fecha de entrega: 9 de julio de 2022

Arequipa - Perú

Índice de Contenidos

1. Animación 3D **1**
1.1. Creación de Escena 1
1.2. Animator 1
1.3. Codigos 4

2. Animación 2D - Sprites **7**

3. Animación de menús **11**

4. Animación 2D - Dinosaurio Zombie **12**
4.1. Entorno 12
4.2. Zombie 13
4.3. Dinosaurio 17

Referencias **22**

Índice de Figuras

1. Vista del Unity 1
2. Vista del Golem en escena 1
3. Vista del Animator 2
4. Vista del Blend Tree 2
5. Vista del Inspector 3
6. Sprites 2d : Vista general del proyecto 8
7. Sprites 2d : SpriteSheet 8
8. Sprites 2d : Animator 9
9. Sprites 2d : Personaje 9
10. Sprites 2d : Controlador 10
11. Sprites 2d : Vista de cámara 10
12. 11
13. Animaciones realizadas. 11
14. Escena de créditos finales. 12
15. Draw Mode: Tiled 12
16. Zombie 13
17. Zombie: Box - Rigidbody 14
18. Animación de zombie 15
19. Animaciones del dinosaurio 17
20. Dinosaurio - Animator 18

Índice de Códigos

1. CameraController.cs 4

2. RockManMove.cs 4
3. CameraController.cs 15
4. CameraController.cs 18

1. Animación 3D

Se estableció una animación de golem mediante un animator e interacción con el usuario usando eventos en el script para poder cambiar de estado de animación.

1.1. Creación de Escena

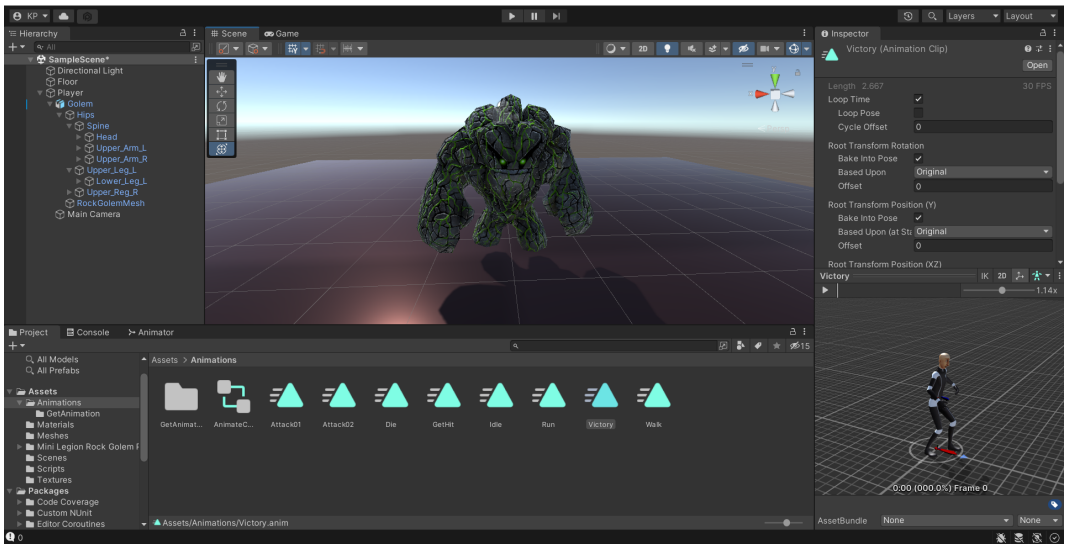

Figura 1: Vista del Unity

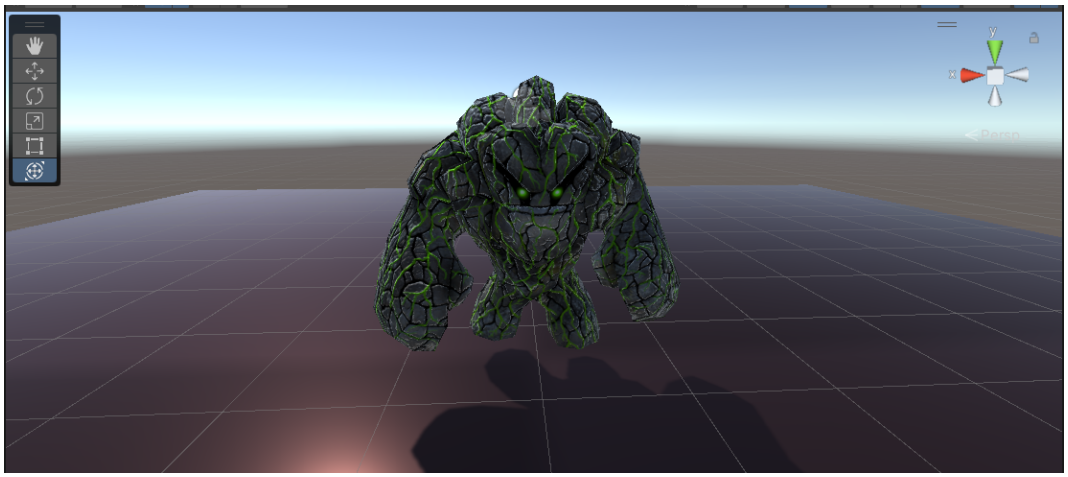

Figura 2: Vista del Golem en escena

1.2. Animator

Se estableció 5 estados en las cuales 3 se uso de manera blend tree el cual se establece a partir de una valor el cambio de animación mas fluida. Los dos estados restantes son de ataque y victoria el

cual se habilita y deshabilita con el script.

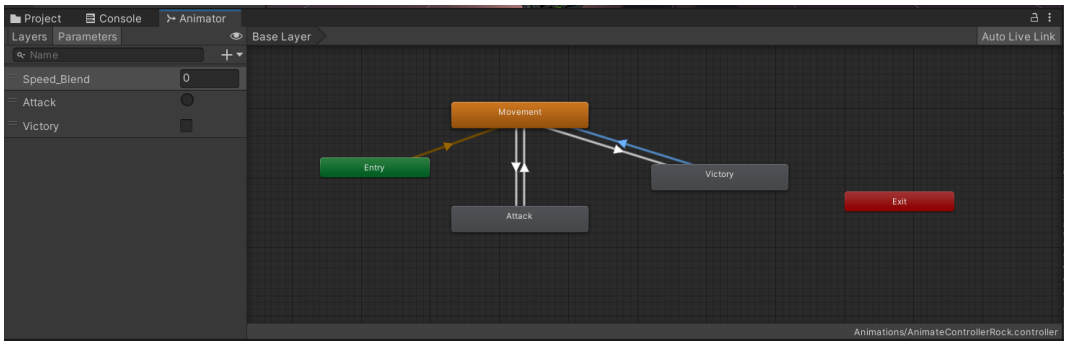

Figura 3: Vista del Animator

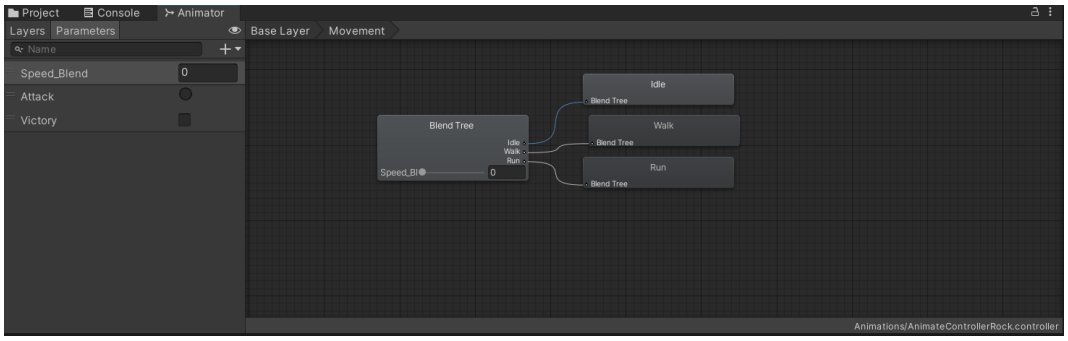

Figura 4: Vista del Blend Tree

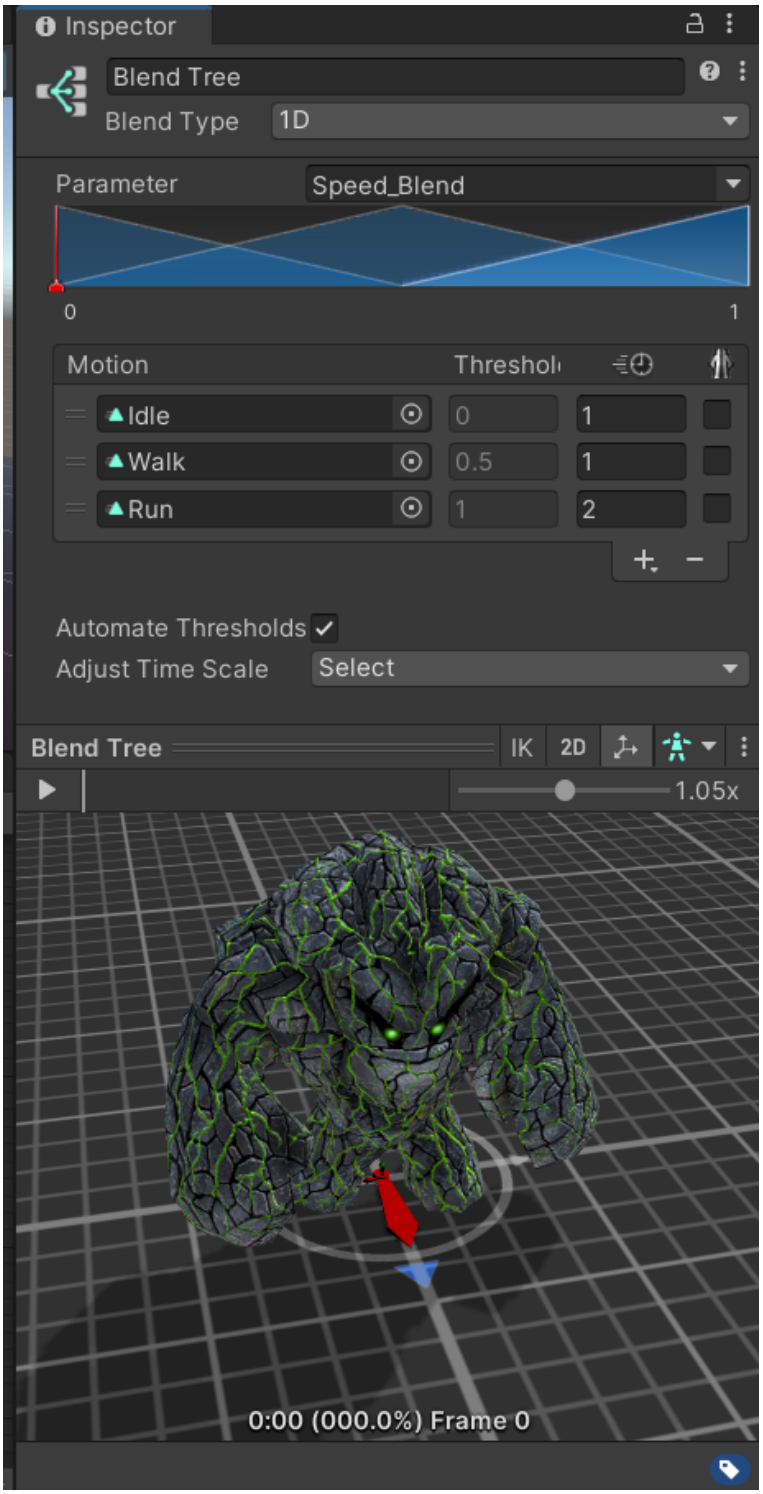

Figura 5: Vista del Inspector

1.3. Codigos

Script para controlar la cámara con el personaje.

Código 1: CameraController.cs

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

public class CameraController : MonoBehaviour
{
    [SerializeField] private float mouseSensitivity;

    private Transform parent;
    // Start is called before the first frame update
    void Start()
    {
        parent = transform.parent;
        Cursor.lockState = CursorLockMode.Locked;
    }

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

    private void Rotate()
    {
        float mouseX = Input.GetAxis("Mouse X");
        mouseX = mouseX * mouseSensitivity * Time.deltaTime;
        parent.Rotate(Vector3.up, mouseX);
    }
}
```

Script para controlar el movimiento del personaje y el establecer el cambio de estados en el animmator.

Código 2: RockManMove.cs

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

public class RockManMove : MonoBehaviour
{
    // Start is called before the first frame update
    [SerializeField] private float walkSpeed;
    [SerializeField] private float moveSpeed;
    [SerializeField] private float runSpeed;

```

```
12  private Vector3 moveDirection;
13  private Vector3 velocity;
14  private CharacterController controller;
15  private Animator animator;
16  [SerializeField] private Transform rockModel;
17
18  [SerializeField] private bool isGrounded;
19  [SerializeField] private float groundCheckDistance;
20  [SerializeField] private LayerMask groundMask;
21  [SerializeField] private float gravity;
22  [SerializeField] private float jumpHeight;
23
24
25
26  void Start()
27  {
28      controller = GetComponent<CharacterController>();
29      animator = GetComponentInChildren<Animator>();
30  }
31
32  // Update is called once per frame
33  void Update()
34  {
35      onMove();
36      if (Input.GetKeyDown(KeyCode.Mouse0))
37      {
38          StartCoroutine(Attack());
39      }
40
41  }
42
43  private void onMove()
44  {
45      isGrounded = Physics.CheckSphere(transform.position, groundCheckDistance, groundMask);
46      float moveZ = Input.GetAxis("Vertical");
47      Debug.Log(moveZ);
48
49      if (moveZ < 0.0f) rockModel.localScale = new Vector3(0.5f, 0.5f, -0.5f);
50      else if (moveZ > 0.0f) rockModel.localScale = new Vector3(0.5f, 0.5f, 0.5f);
51      moveDirection = new Vector3(0, 0, moveZ);
52      moveDirection = transform.TransformDirection(moveDirection); // para modificar el axis en
     ↪ modo local
53
54
55      if (isGrounded && velocity.y < 0)
56      {
57          velocity.y = -2f;
58
59      }
60
61      if (isGrounded)
```

```
62      {
63
64
65          if (moveDirection != Vector3.zero && !Input.GetKey(KeyCode.LeftShift))
66          {
67              Walk();
68          }
69          else if (moveDirection != Vector3.zero && Input.GetKey(KeyCode.LeftShift))
70          {
71              Run();
72          }
73          else if (moveDirection == Vector3.zero)
74          {
75              Idle();
76          }
77          moveDirection *= moveSpeed;
78
79          if (Input.GetKey(KeyCode.Space))
80          {
81              Jump();
82          }
83
84          if (Input.GetKey(KeyCode.LeftControl))
85          {
86              Victory();
87          }
88
89      }
90      //animator.SetBool("Victory", false);
91
92      //controller.Move(moveDirection * Time.deltaTime);
93
94      velocity.y += gravity * Time.deltaTime;
95      moveDirection.y = velocity.y;
96      controller.Move(moveDirection * Time.deltaTime);
97  }
98
99  private void Idle()
100 {
101     moveSpeed = 0;
102     animator.SetFloat("Speed_Blend", 0, 0.1f, Time.deltaTime);
103     animator.SetBool("Victory", false);
104 }
105
106 private void Walk()
107 {
108     moveSpeed = walkSpeed;
109     animator.SetFloat("Speed_Blend", 0.5f, 0.1f, Time.deltaTime);
110 }
111
112 private void Run()
```

```
113     {
114         moveSpeed = runSpeed;
115         animator.SetFloat("Speed_Blend", 1, 0.1f, Time.deltaTime);
116     }
117 
118     private void Jump()
119     {
120         velocity.y = Mathf.Sqrt(jumpHeight * -2 * gravity);
121         animator.SetFloat("Speed_Blend", 0, 0.01f, Time.deltaTime);
122     }
123     private void Victory()
124     {
125         animator.SetBool("Victory", true);
126         //moveSpeed = 0;
127     }
128 
129 
130     private IEnumerator Attack()
131     {
132         animator.SetTrigger("Attack");
133         yield return new WaitForSeconds(0.9f);
134     }
135 }
```

2. Animación 2D - Sprites

Los sprites son imágenes o animaciones bidimensionales superpuestas en una escena. Son los elementos no estáticos dentro de un juego 2D, moviéndose independientemente del fondo. A menudo se usan para representar personajes controlados por el jugador, accesorios, unidades enemigas, etc., los sprites pueden estar compuestos de múltiples mosaicos o sprites más pequeños.

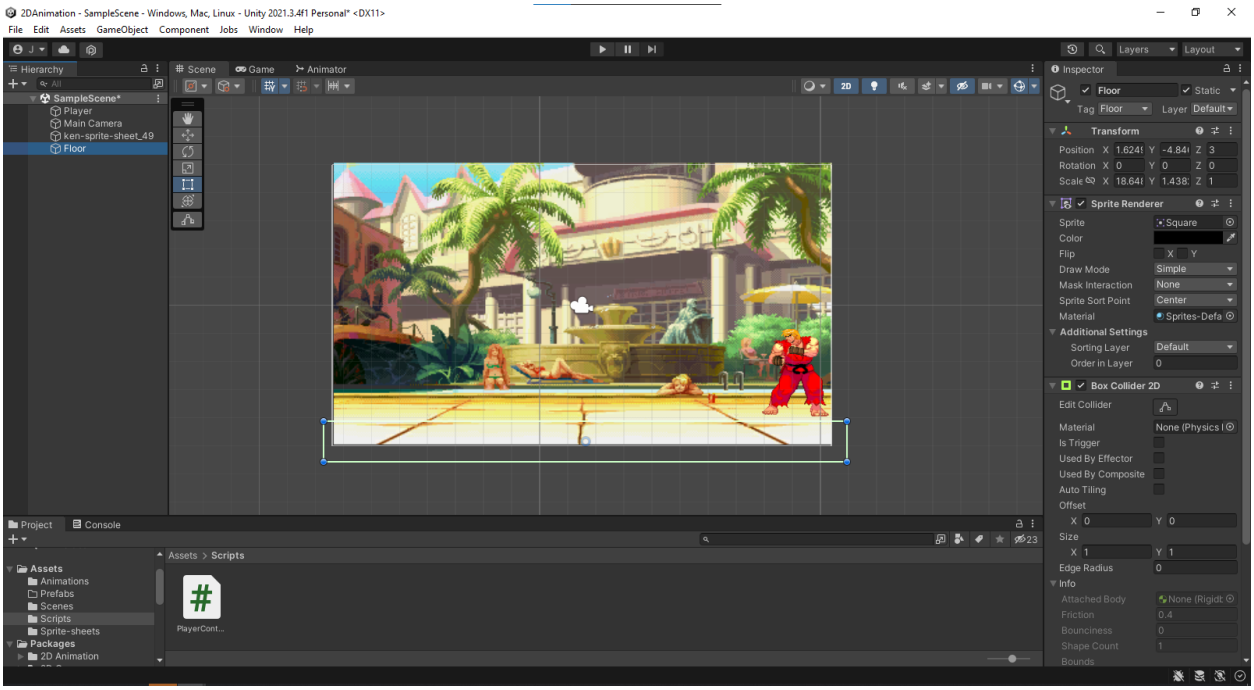

Figura 6: Sprites 2d : Vista general del proyecto

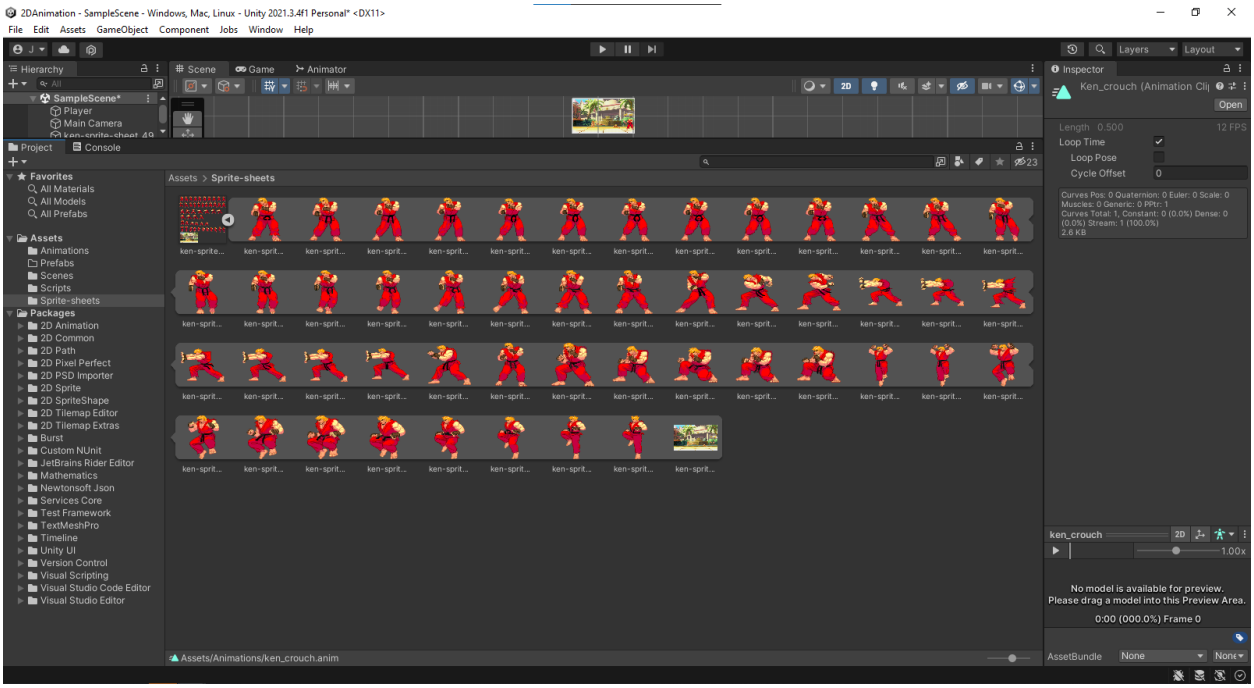

Figura 7: Sprites 2d : SpriteSheet

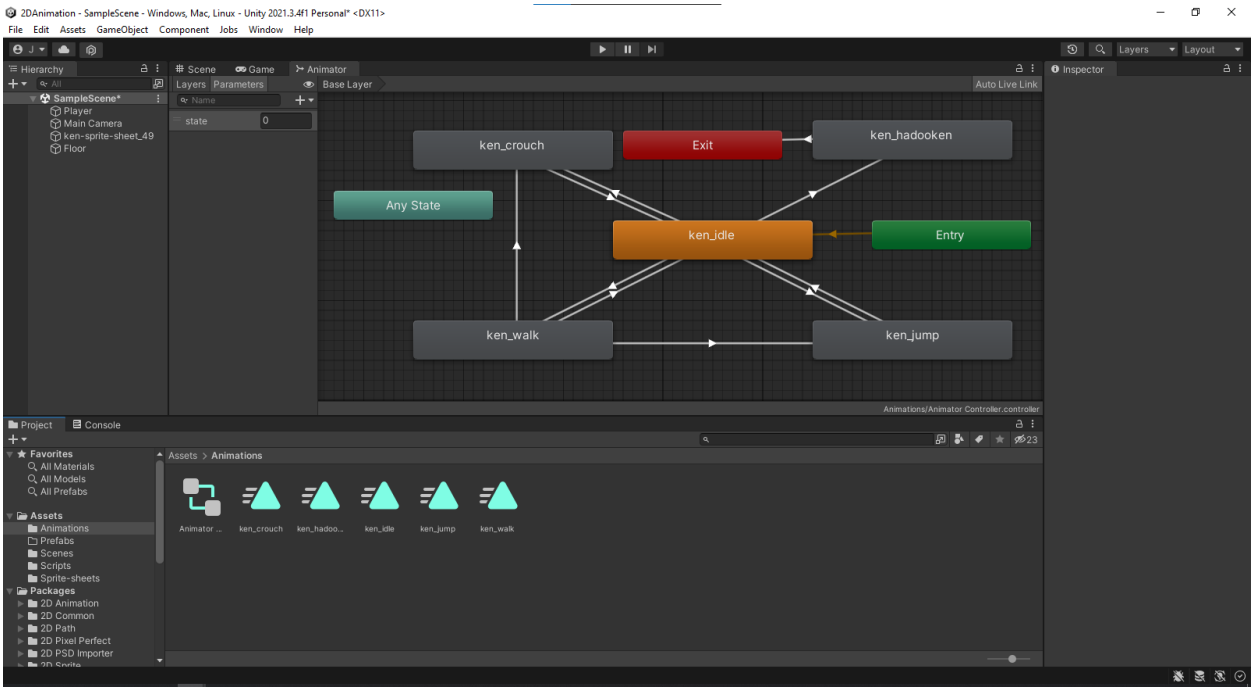

Figura 8: Sprites 2d : Animator

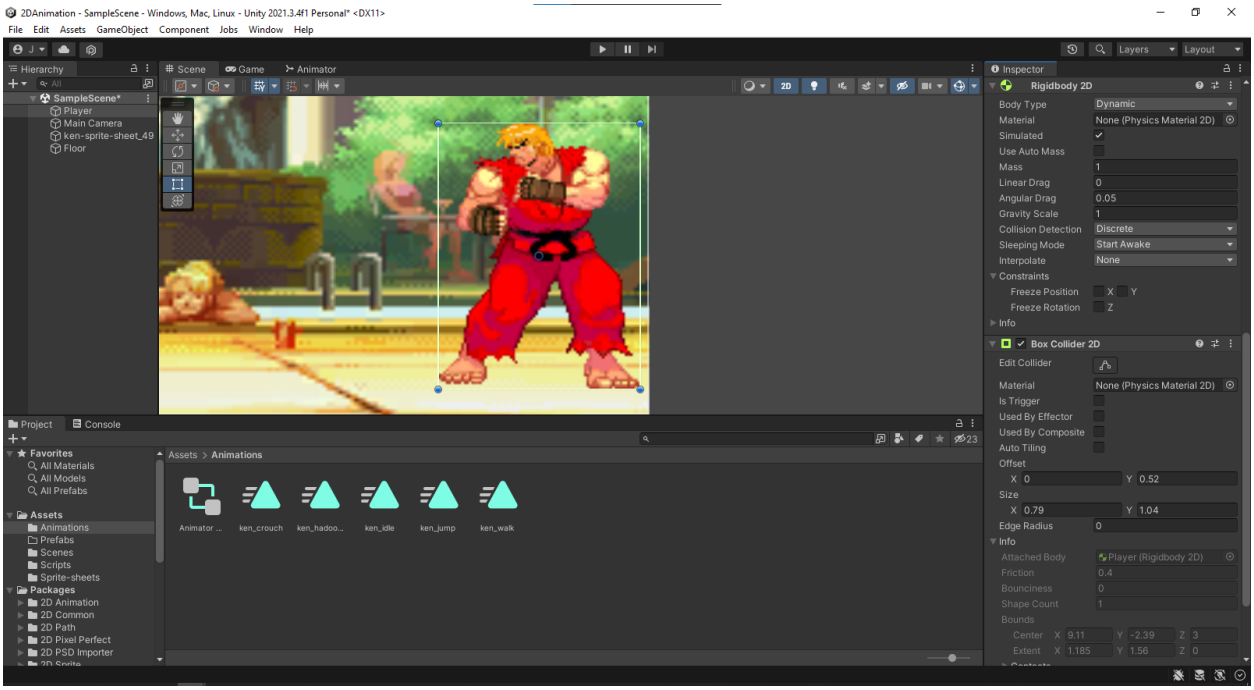

Figura 9: Sprites 2d : Personaje

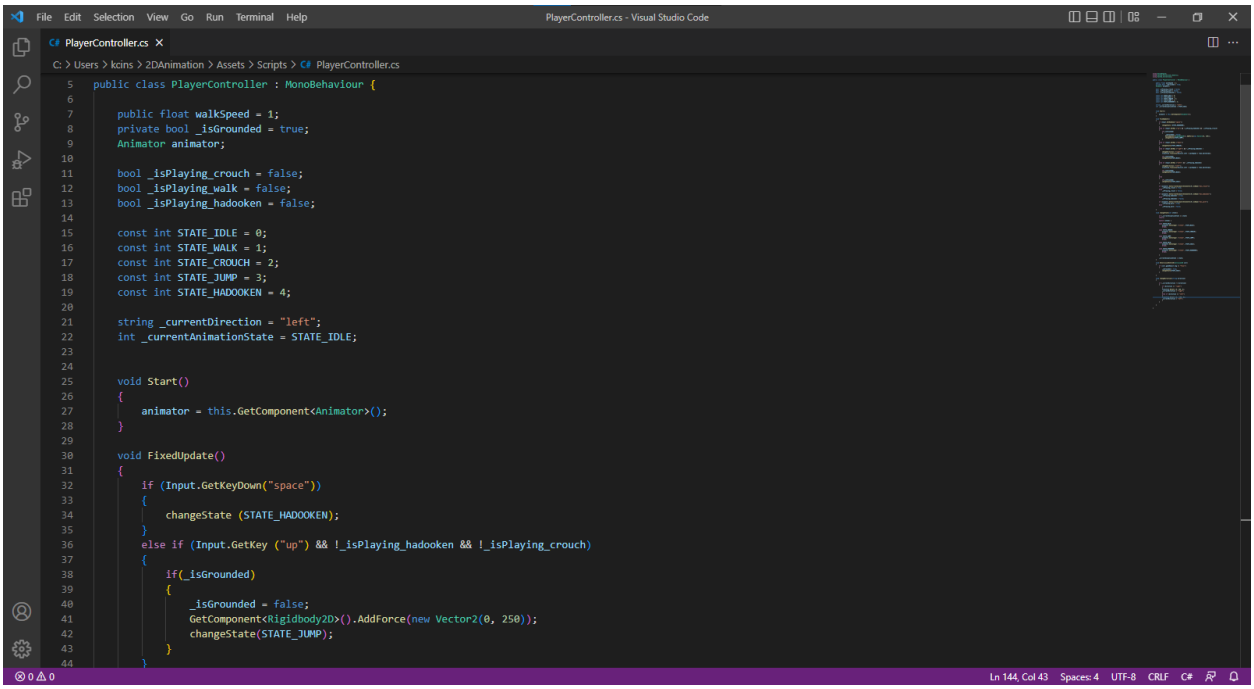

Figura 10: Sprites 2d : Controlador

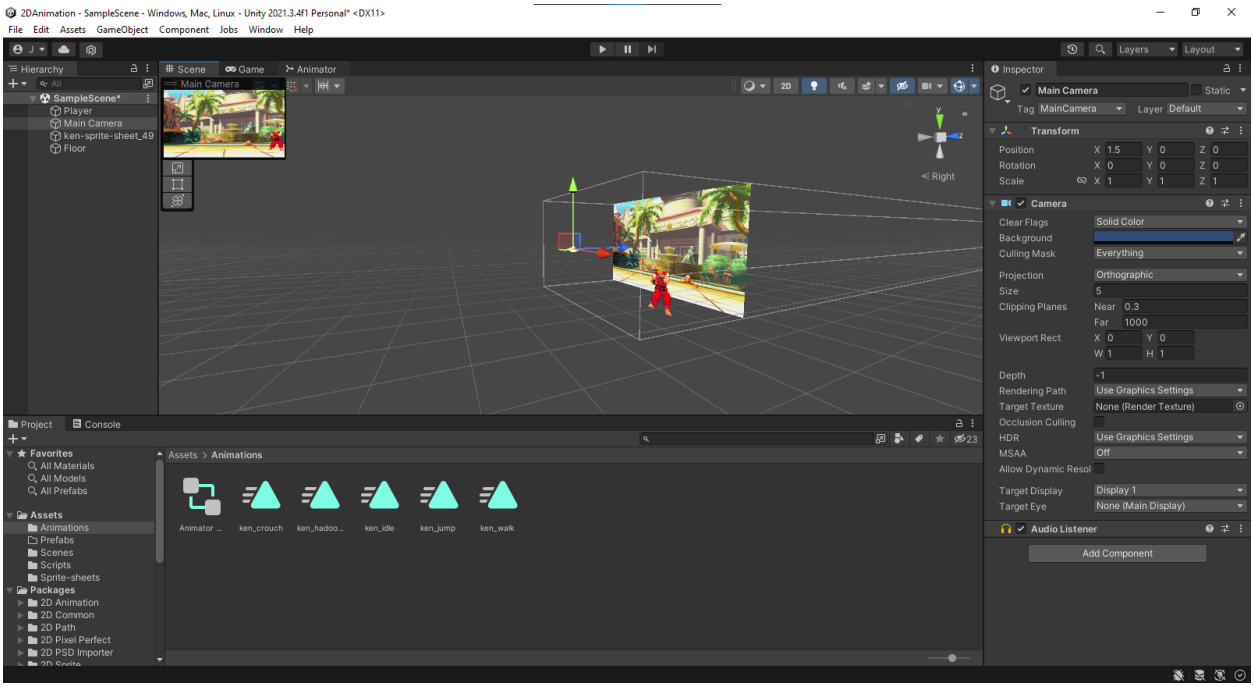

Figura 11: Sprites 2d : Vista de cámara

3. Animación de menús

Se realizaron 2 escenas, uno como menú principal y otro como créditos que muestra a los integrantes del grupo. La transición entre ambas escenas se realiza mediante botones en cada escena.

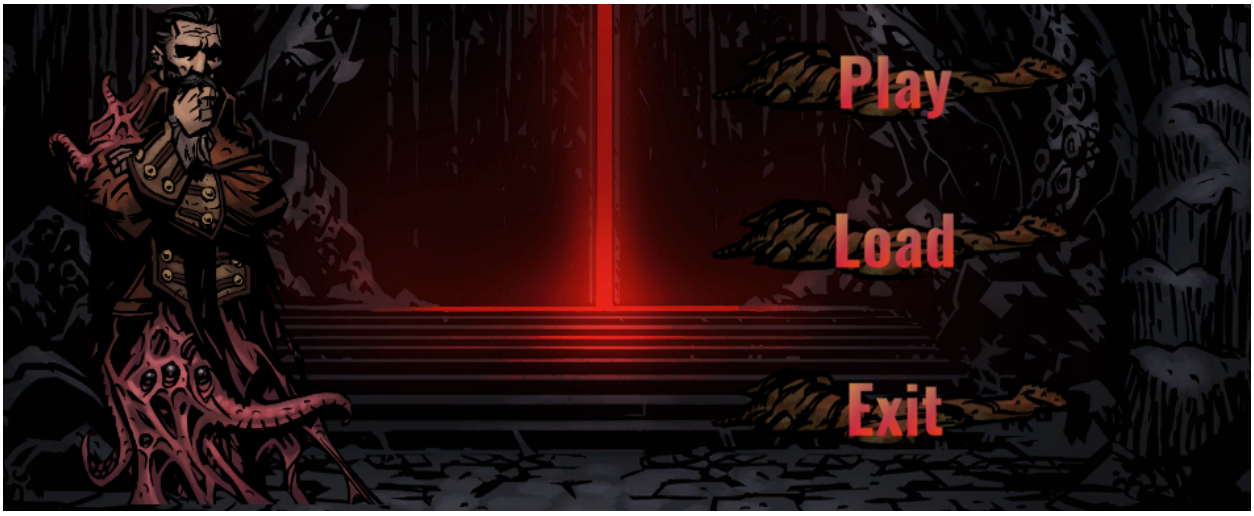

Figura 12

Para las animaciones ;

- El fondo se mantiene moviéndose ligeramente .
- Los botones tienen una animación para que crezcan al pasar el ratón sobre ellas.
- El hombre posee una animación que durante unos pocos frames cambia a una forma mas grotesca.
- Para los créditos es una secuencia vertical que muestran los integrantes.

Figura 13: Animaciones realizadas.

Para la realización del menú se usaron sprites de un juego llamado Darkest Dungeon.

Figura 14: Escena de créditos finales.

4. Animación 2D - Dinosaurio Zombie

Creamos una animación 2D de un entorno sencillo, los personajes son un dinosaurio y un zombie los cuales nos servirán para mostrar las animaciones creadas y el tema del laboratorio

4.1. Entorno

El entorno solo se coloca un fondo, para que sea el paisaje y luego para el suelo colocamos un sprite de un piso de tierra y pasto, lo cual servira para la movilidad del enemigo zombie y dinosaurio. Lo unico a recalcar es que ese sprite se agrando por Draw Mode Tiled para que no pareciera que lo agrandamos a mano y asi quede feo en el resultado final

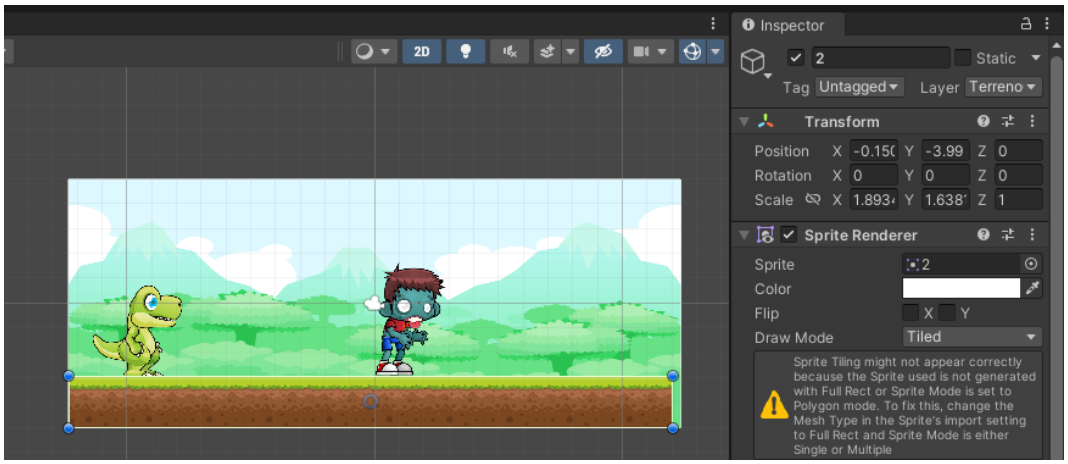

Figura 15: Draw Mode: Tiled

4.2. Zombie

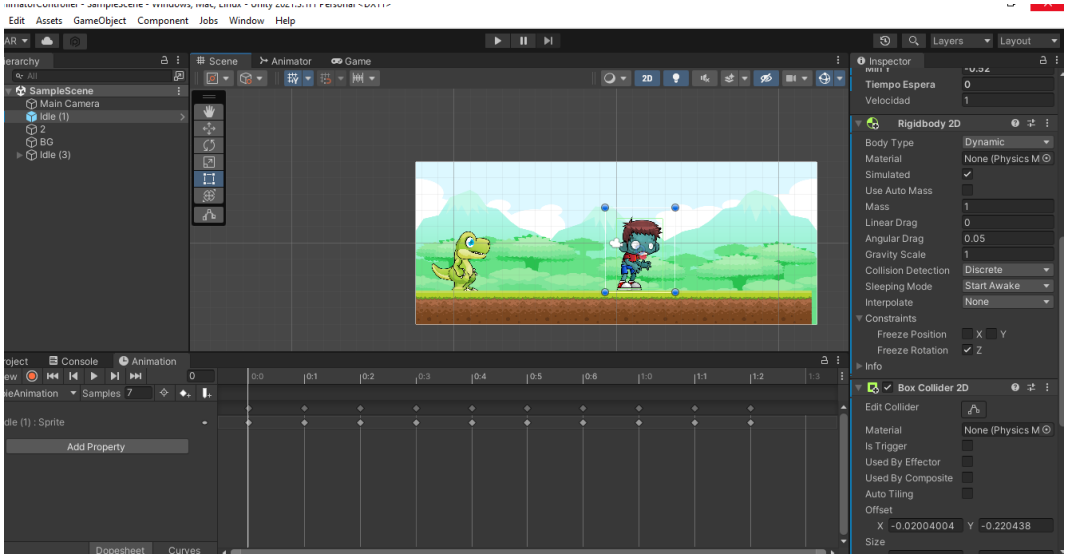

Figura 16: Zombie

El zombie va tener propiedades como el rigidbody y box collider para que pueda caminar por el piso.

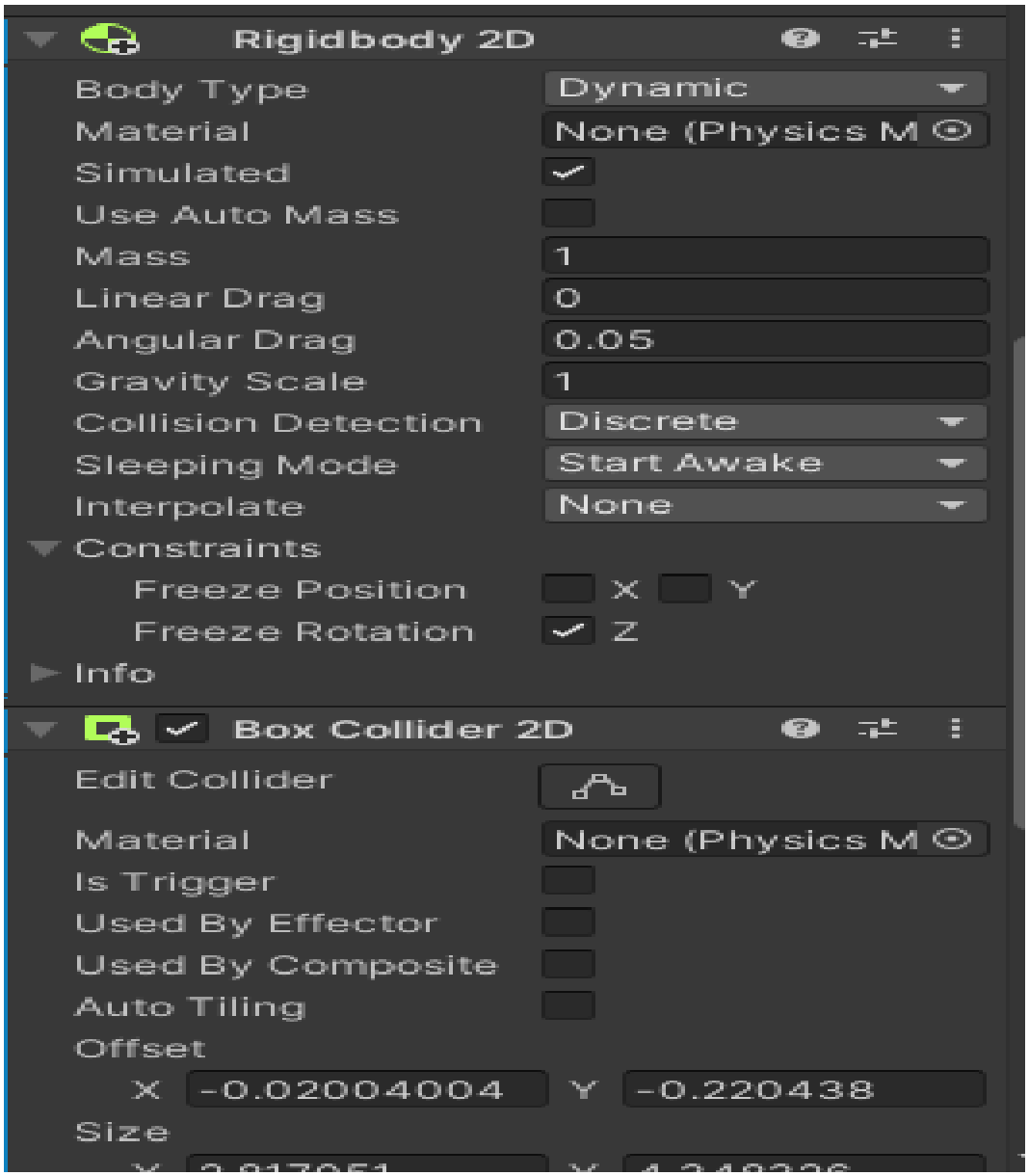

Figura 17: Zombie: Box - Rigidbody

Luego tambien vamos animar los movimientos del zombie lo cual haremos una patrulla pero para esto debemos crear una animación con los sprites que tenemos del zombie

Figura 18: Animación de zombie

Luego de animarlos podemos crear el código para que el zombie patrulle

Código 3: CameraController.cs

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

public class movimientoX : MonoBehaviour
{
    public float minX;
   public float maxX;
    public float minY;
   public float TiempoEspera = 2f;
   public float Velocidad = 1f;
    private GameObject _LugarObjetivo;
    void Start()
    {
        UpdateObjetivo();
      StartCoroutine("Patrullar");
    }

    // Update is called once per frame
    void Update()
    {
```

```
22
23    }
24
25    private void UpdateObjetivo()
26    {
27      // Si es la primera vez iniciar el patrullaje para la izquierda
28      if (_LugarObjetivo == null) {
29        _LugarObjetivo = new GameObject("Sitio_objetivo");
30        _LugarObjetivo.transform.position = new Vector2(minX, minY);
31        transform.localScale = new Vector3(-1, 1, 1);
32        return;
33      }
34
35      // iniciar el patrullaje para la derecha
36      if (_LugarObjetivo.transform.position.x == minX) {
37        _LugarObjetivo.transform.position = new Vector2(maxX, minY);
38        transform.localScale = new Vector3(1, 1, 1);
39      }
40
41      // Cambio de sentido de derecha a izquierda
42      else if (_LugarObjetivo.transform.position.x == maxX) {
43        _LugarObjetivo.transform.position = new Vector2(minX, minY);
44        transform.localScale = new Vector3(-1, 1, 1);
45      }
46    }
47
48    private IEnumerator Patrullar()
49    {
50      // Co-rutina para mover el enemigo
51      while(Vector2.Distance(transform.position, _LugarObjetivo.transform.position) > 0.05f) {
52        // Se desplazará hasta el sitio objetivo
53        Vector2 direction = _LugarObjetivo.transform.position - transform.position;
54        float xDirection = direction.x;
55
56        transform.Translate(direction.normalized * Velocidad * Time.deltaTime);
57
58        yield return null;
59      }
60
61      // En este punto, se alcanzó el objetivo, se establece nuestra posición en la del objetivo.
62      Debug.Log("Se alcanzo el Obejitvo");
63      transform.position = new Vector2(_LugarObjetivo.transform.position.x, transform.position.y);
64
65      // Esperamos un momento antes de volver a movernos
66      Debug.Log("Esperando " + TiempoEspera + " segundos");
67      yield return new WaitForSeconds(TiempoEspera);
68
69      Debug.Log("Se espera lo necesario para que termine y vuelva a empezar movimiento");
70      UpdateObjetivo();
71      StartCoroutine("Patrullar");
72    }
```

```
73 }
```

En este código solo damos un X-máximo y X-mínimo para que patrulle el zombie y damos una velocidad y cambio de sprite (espejo)

4.3. Dinosaurio

En la parte del Dinosaurio se realizo lo mismo que el zombie, lo que agregamos es más animaciones y crear el animator, como la siguiente imagen.

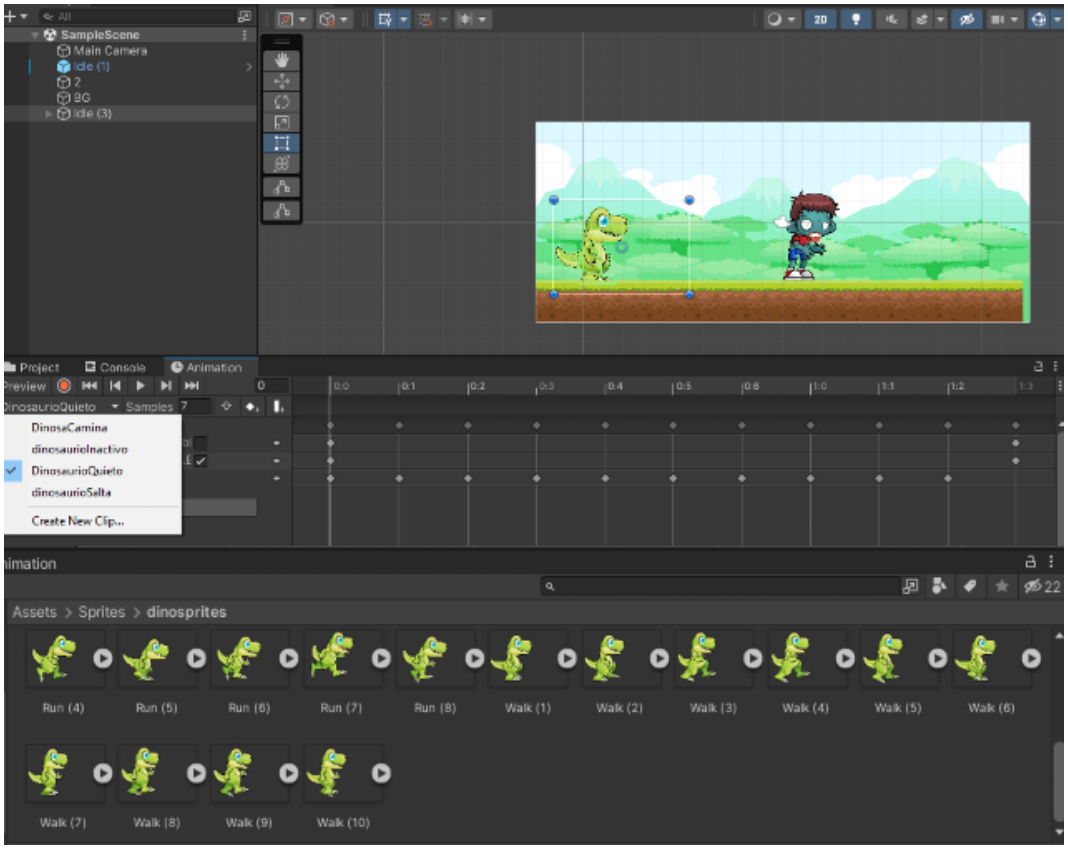

Figura 19: Animaciones del dinosaurio

Cuando tenemos un todas las animaciones, esto creara automaticamente los animators lo cual podremos dar estados y con eso podemos ayudarnos con los parámetros que van a dar condiciones para irnos de un estado a otro.

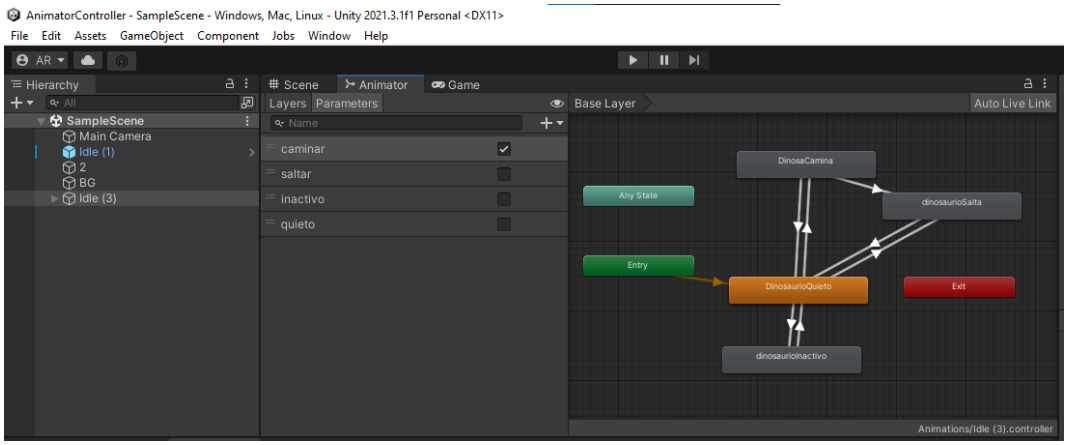

Figura 20: Dinosaurio - Animator

Y finalmente en el código, podemos hacer esas animaciones con los parametros que creamos, en el código podemos hacer los tipicos movimientos de los personajes derecha-izquierda-salto junto con ellos las animaciones para que se muevan,esto podemos verlo en el siguiente código

Código 4: CameraController.cs

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

public class movimiento : MonoBehaviour
{
    private movimiento objeto;
    private bool flip = false; //personaje inicia con sprite positivo hacia la derecha HACEMOS falso el giro desde el inicio

    private Animator _animator;//Esto permitirá utilizar las animaciones para cada estado

    public float velocidad = 2f;//Velocidad por defecto para el movimiento del jugador
    public float FuerzaSalto = 5;//Fuerza que utiliza para saltar desde el suelo

    public Rigidbody2D rb;//Necesitaremos el rigidbody de nuestro personaje para moverlo

    float h;//Capturará acción en los botones para movimiento horizontal
    public bool SeMueve;//falso o true para acción de h
    float v;//Capturará acción en los botones para movimiento vertical, en este caso mirar arriba y agacharse
    public bool jump;//Capturará acción del botón salto

    public Transform poloTierra;//Capturamos el nivel de los pies del personaje para cuando existe contacto con el suelo
    public bool estaEnTierra;//Definiremos cuando tocamos suelo o cuando estamos en el aire u otra superficie

    //Identificamos el contacto con la superficie o layer elegido como suelo
    public float radius = 0.3f;
```

```
27    public LayerMask sueloLayer;
28
29    public bool permitirMov;
30    public bool permitirSalto;
31
32    void Awake()
33    {
34      objeto = this;
35    }
36
37    // Start is called before the first frame update
38    void Start()
39    {
40       rb = GetComponent<Rigidbody2D>();
41       _animator = GetComponent<Animator>();
42    }
43
44    // Update is called once per frame
45    void Update()
46    {
47       h = Input.GetAxisRaw("Horizontal");
48       v = Input.GetAxis("Vertical");
49       jump = Input.GetButton("Jump");
50
51      //Capturamos como FALSE O TRUE si hay PULSACION por parte del USUARIO para
     ↪ movimiento en X
52       SeMueve = (h != 0f);
53
54       //Pedimos permiso para movernos SIEMPRE
55       permitirMov = aprobarMov();
56
57       //Este metodo se encarga de voltear el sprite personaje para que mire a la izq o a la der
58       flipSprite();
59
60       //mientras se OPRIMA el movimiento en direccion X se ejecuta  LA ANIMACIóN CAMINAR
61       Anima_Caminar();
62
63
64       //Actualzaremos las animaciones según sea el contacto con el suelo o alguna superficie
65       Anima_Update01();
66    }
67
68    void FixedUpdate(){
69       if(permitirMov)
70           rb.velocity = new Vector2(h * velocidad , rb.velocity.y);
71    }
72
73    void OnDestroy(){
74       objeto = null;
75    }
76
```

```
77    private void inactivo(){
78    _animator.SetBool("inactivo", true);
79    _animator.SetBool("quieto", false);
80  }
81  //Retornamos al personaje activo a espera de movimiento
82  private void activo(){
83    _animator.SetBool("inactivo", false);
84    _animator.SetBool("quieto", true);
85  }
86
87  //Actualizar las animaciones segun sea el estado del personaje sobre el LAYER suelo
88  private void Anima_Update01(){
89    //Asi capturamos un BOOL para saber si nuestro PERSONAJE hace CONTACTO O
       ↪ COLISIONA con una LAYER elejida
90    estaEnTierra = Physics2D.OverlapCircle(poloTierra.position, radius, sueloLayer);
91    if(estaEnTierra){
92       _animator.SetBool("saltar", false);
93
94       if(_animator.GetBool("quieto"))
95          _animator.SetBool("inactivo", false);
96
97    }
98    if(!estaEnTierra){
99       // _animator.SetBool("Caer", true);
100      permitirMov=true;
101   }
102   if((Input.GetButtonDown("Jump"))&&(estaEnTierra)&&(permitirSalto)){
103         _animator.SetBool("quieto", true);
104         Saltar();
105   }
106 }
107
108 //Funcion preparar salto con animación
109 public void Saltar(){
110   if(!estaEnTierra){
111     return;
112   }
113   else{
114      permitirSalto=true;
115
116      _animator.SetBool("saltar", true);
117
118
119      rb.velocity = new Vector2(rb.velocity.x, FuerzaSalto);
120
121   }
122 }
123
124 //Objetivo de esta funcion es restringir el desplazamiento si se esta agachado o mirando arriba
125 private bool aprobarMov(){
126   //Permitiremos saltar si estamos agachados
```

```
127    if(Input.GetButtonDown("Jump")){
128        permitirSalto=true;
129        return true;
130    }
131 return true;
132 }
133 
134 //Metodo Para Voltear la imagen en funcion del eje horizontal
135 private void flipSprite(){
136 
137        if ((h < 0)&&(flip == false)){
138            activo();
139            transform.localScale = new Vector3(-transform.localScale.x, transform.localScale.y, 1);
140            flip = true;
141        }
142        if ((h > 0)&& (flip == true)){
143            activo();
144            transform.localScale = new Vector3(-transform.localScale.x, transform.localScale.y, 1);
145            flip = false;
146        }
147 }
148 
149 //Metodo para detectar la direccion y proceder con la ANIMACIóN CAMINAR solo sobre SUELO
150 private void Anima_Caminar(){
151    if (((h > 0) || (h < 0))){
152        activo();
153        permitirSalto=true;
154        _animator.SetBool("caminar", true);
155    }
156    else
157        _animator.SetBool("caminar", false);
158 }
159 }
```

Finalmente podemos apreciar lo que se explico en el vídeo

Referencias

- **Grabación de Explicación:** https://drive.google.com/file/d/11H_xXR8rE59CMLGWR-2rOmIEgNo_LiuA/view
- **Links tutorial para Animacion-3D**
  - https://www.youtube.com/watch?v=qc0xU2Ph86Q&t=2236s
  - https://www.youtube.com/watch?v=_5pxcUykXcA
  - https://www.youtube.com/watch?v=qc0xU2Ph86Q&t=2237s
- **Link Tutorial Animación 2D Zombie - Dinosaurio**
  - https://www.youtube.com/watch?v=dBfIMaIN2Jc
  - https://www.youtube.com/watch?v=J9Ljl5KdgSk
  - https://www.youtube.com/c/ElCanaldeAlejo-Co
- **Links tutorial para Animacion-2D sprites**
  - https://www.youtube.com/watch?v=jPF24aL4-MQ&t=1s
  - https://www.pinterest.com/pin/531284087267014558/
  - https://www.youtube.com/watch?v=jjvjbKrIgiw
- **Links tutorial del Animación para Menús**
  - https://www.youtube.com/watch?v=jh3zD-wGBnw