PRAKTIKUM ANIMASI DAN GAME PERTEMUAN :9

Unity 2D Game Animation & Respawn

NIM	**:**	1918039
NAMA	**:**	M.Sofian Attasauri
MATERI	**:**	Unity 2D Game Animation & Respawn
KELAS	**:**	D
TUGAS	**:**	Membuat Mekanisme Respawn

9.1 Tujuan

1. Praktikan dapat mengetahui *Game* 2D dan *Game* 3D.
2. Praktikan dapat mengetahui jenis *Game.*
3. Praktikan dapat menerapkan *Game* pada Unity.

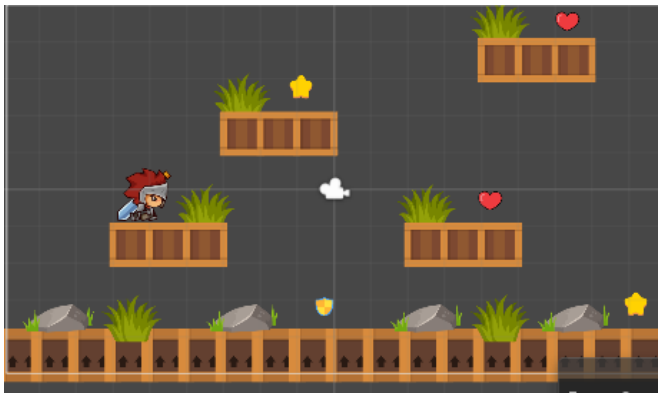
9.2 Alat dan Bahan

1. Laptop/pc.
2. Modul Praktikum Animasi dan *Game* 2024.
3. Unity 2017.

9.3 Langkah-Langkah Membuat Tugas

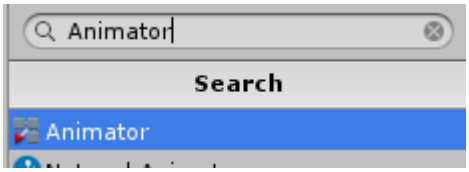
A. Membuat Mekanisme Respawn

1. Buat file *projek Unity* Tugas 8



Gambar 9.1 Buka projek tugas 8

2. Klik *warrior*, pergi ke *Inspector*, pilih *Add Component*, cari Animator

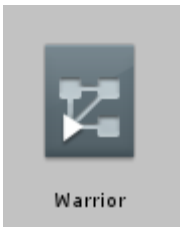


Gambar 9.2 Menambahkan component Animator

3. Buat sebuah folder baru dengan nama "Animator"

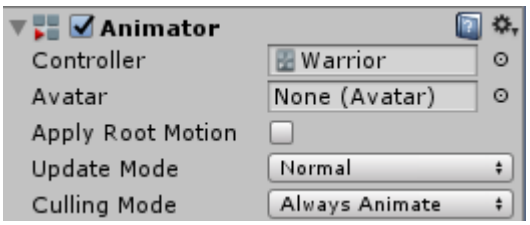
Gambar 9.3 Membuat folder Animator

4. Klik kanan folder Animator>*Create*>*Animation Controller*, dan ubah namanya menjadi "*warrior*"



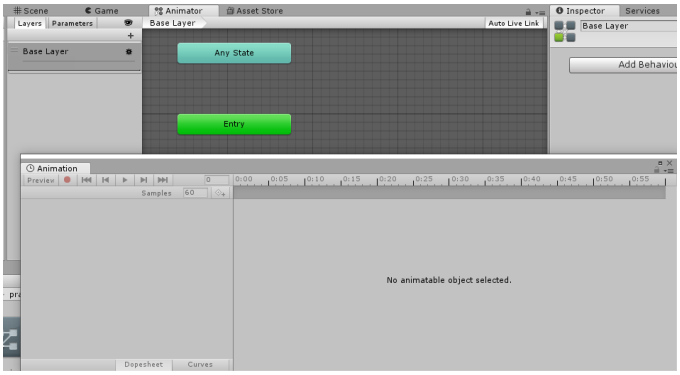
Gambar 9.4 Membuat *Warrior* Animator

5. Drag *warrior*, kemudian masukkan kedalam Controller komponen Animator



Gambar 9.5 Mendrag *warrior* Animator

6. Klik *warrior*, kemudian pergi ke tab Animation, pilih Create

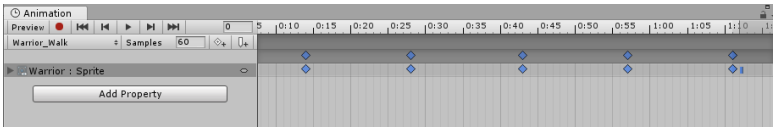


Gambar 9.6 Membuat suatu *Animation*

7. Drag and Drop *warrior* lalu regangkan, pada *timeline* tekan *Ctrl+A* di *keyboard*, klik bagian kotak kecil disamping *keyframe* terakhir dan geser sampai waktu 0:50

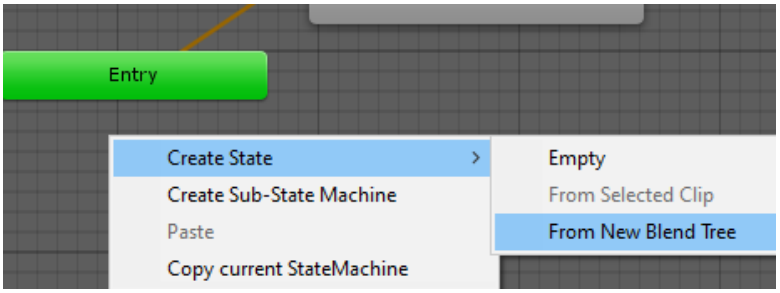
Gambar 9.7 *Drag and Drop warrior*

8. Pilih *warrior enemywalk* kemudian drag kedalam *timeline warrior_walk Animation* lalu geser ke *keyframe* waktu 1:10



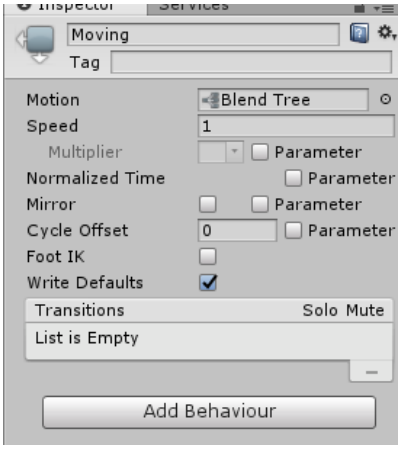
Gambar 9.8 Menambahkan *warrior enemywalk*

9. Klik kanan pada area sekitar *Animator*, pilih *Create State>From New Blend Tree*



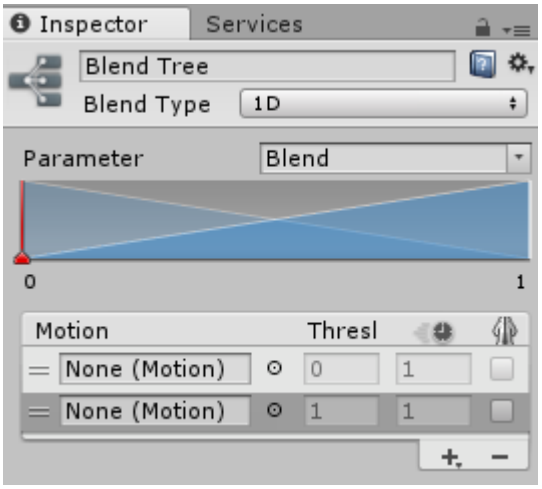
Gambar 9.9 Membuat *Blend Tree*

10. Klik *Blend Tree*, pergi ke *Inspector* dan ubah namanya menjadi *Moving*



Gambar 9.10 Mengubah nama menjadi *Moving*

11. Klik 2X *Blend Tree* "*Moving*", pergi ke *Inspector*, tekan icon + dan pilih *Add Motion Field*, tambahkan satu lagi *Motion Field*



Gambar 9.11 Menambahkan *Motion field*

12. Klik bagian icon bulat di samping *None (Motion)*, kemudian akan muncul *Windows Motion*, pilih *warrior_Idle* dan yang *motion* satunya pilih *warrior_Walk*

Gambar 9.12 Menambahkan sebuah *motion*

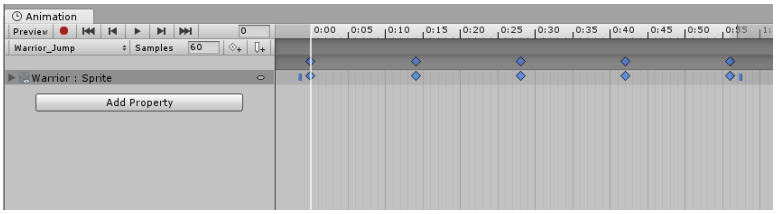
13. Lalu pada *script warrior* tambahkan sebuah *Source* Berikut

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

```
public class Warrior : MonoBehaviour {

    Animator animator;
    Rigidbody2D rb;
    [SerializeField] Transform groundcheckCollider;
    [SerializeField] LayerMask groundLayer;
    public int nyawa;
    Vector2 play;
    public bool play_again;
    const float groundCheckRadius = 0.2f;
    [SerializeField] bool isGrounded;
    [SerializeField] float speed = 1;
    [SerializeField] float jumpPower = 100;
    float horizontalValue;
    bool facingRight;
    bool jump;
    private void Awake()
    {
        rb = GetComponent<Rigidbody2D>();
        animator = GetComponent<Animator>();

    }
    // Update is called once per frame
    void Update()
    {
        horizontalValue                                    =
Input.GetAxisRaw("Horizontal");
        if (Input.GetButtonDown("Jump"))
        {
            jump = true;
            animator.SetBool("Jumping", true);
        }
        else if (Input.GetButtonUp("Jump"))
        {
            jump = false;
            animator.SetFloat("Blend_Jump",
rb.velocity.y);
        }
        if (nyawa < 0)
        {
            playagain();
        }
    }
    void playagain()
    {
        if (play_again == true)
        {
            nyawa = 3;
            transform.position = play;
            play_again = false;
        }
    }

    void FixedUpdate()
    {
        GroundCheck();
        Move(horizontalValue, jump);
    }
```

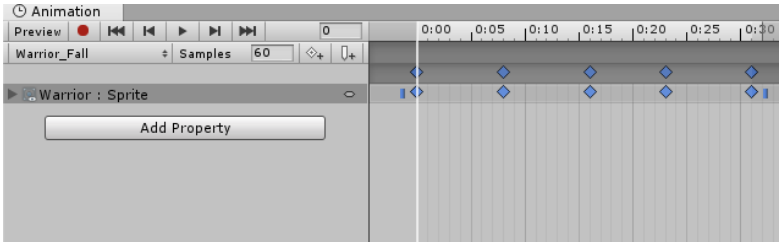
```
    void GroundCheck()
    {
        isGrounded = false;
        Collider2D[] colliders =

Physics2D.OverlapCircleAll(groundcheckCollider.position
, groundCheckRadius,
       groundLayer);
        if (colliders.Length > 0)
            isGrounded = true;
        animator.SetBool("Jumping", !isGrounded);
    }

    void Move(float dir, bool jumpflag)
    {
        if (isGrounded && jumpflag)
        {
            isGrounded = false;
            jumpflag = false;
            rb.AddForce(new Vector2(0f, jumpPower));
        }

        #region gerak kanan kiri
        float   xVal   =   dir   *   speed   *   100   *
Time.fixedDeltaTime;
        Vector2  targetVelocity  =  new  Vector2(xVal,
rb.velocity.y);
        rb.velocity = targetVelocity;
        if (facingRight && dir < 0)
        {
            transform.localScale  =  new  Vector3(-1,  1,
1);
            facingRight = false;
        }
        else if (!facingRight && dir > 0)
        {
            transform.localScale = new Vector3(1, 1, 1);
            facingRight = true;
        }
        animator.SetFloat("Blend",
Mathf.Abs(rb.velocity.x));
        #endregion
    }
}
```

14. *Drag and Drop enemyjump* lalu geser *keyframe* ke waktu 0:55



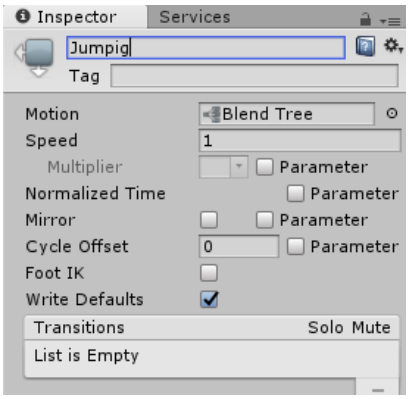
Gambar 9.13 Menambahkan *enemyjump*

15. Drag and Drop enemyfall ke *layer Animation*



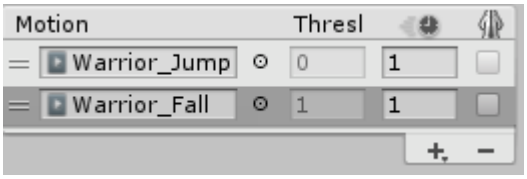
Gambar 9.14 Menambahkan *enemyfall*

16. Buat *Blend Tree*, pergi ke *Inspector* dan ubah namanya menjadi *Jumping*



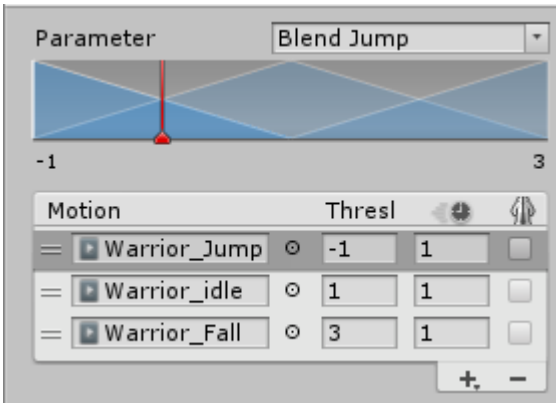
Gambar 9.15 Mengubah nama menjadi *Jumping*

17. Klik bagian icon bulat di samping *None (Motion)*, pilih *warrior_Jump* dan yang *motion* satunya pilih *warrior_Fall*



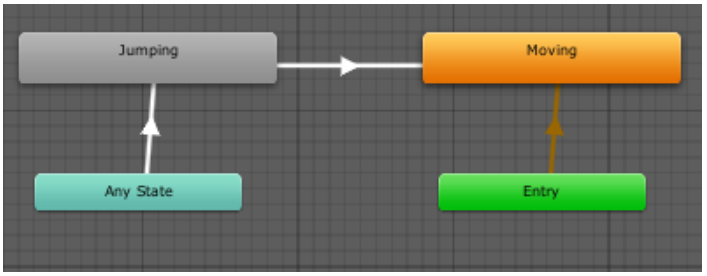
Gambar 9.16 Menambahkan sebuah *motion*

18. Klik *Blend Tree*, pada *Inspector* ubah parameternya menjadi *Blend_Jump*



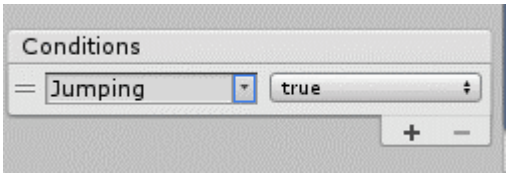
Gambar 9.17 Mengubah parameter *Jumping*

19. Klik kanan lalu pilih *Make Transition* dan buatlah *transation* seperti ini



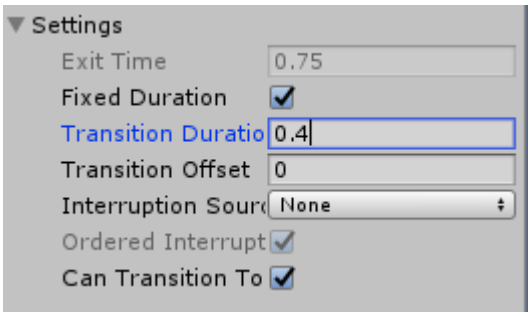
Gambar 9.18 Membuat *Transation*

20. Klik panah yang mengarah ke *Jumping*, pada *inspector* tambahkan *condition*, pilih *condition Jumping* dan ubah nilainya menjadi *true*



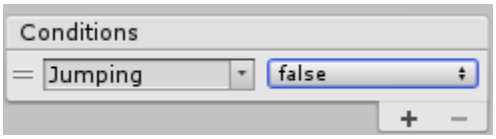
Gambar 9.19 Menambahkan kondisi *Jumping*

21. Klik *Settings* dan ubah nilai *Transition Duration* menjadi 0.4 dan hilangkan centang *Has Exit Time*



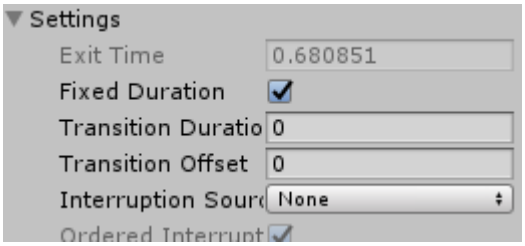
Gambar 9.20 Mengubah *Transition Duration*

22. Klik panah yang mengarah ke *Moving*, pada *inspector* tambahkan *condition*, pilih *condition Jumping* dan ubah nilainya menjadi *false*



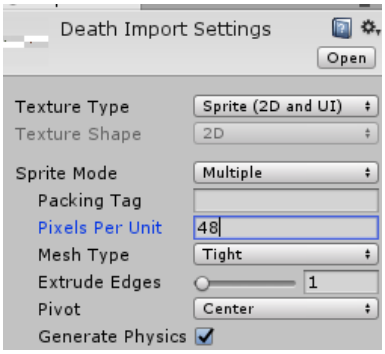
Gambar 9.21 Menambahkan kondisi *Jumping*

23. Klik *Settings* dan ubah nilai *Transition Duration* menjadi 0 dan hilangkan centang *Has Exit Time*



Gambar 9.22 Mengubah *Transition Duration*

24. Untuk membuat sebuah *Enemy (musuh)*, cari sebuah *sprite pack* bernama "Enemy", klik Run dan ubah nilai *Pixel* per Unit menjadi 48 pada *Inspector*.

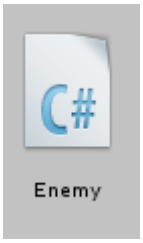


Gambar 9.23 Mengubah *Pixel* Per unit

25. *Drag* enemy tersebut ke dalam editor dan simpan *file* anim tersebut kedalam folder Animator dan beri nama menjadi "*Enemy*" Lalu ubah nama pada *hierarchy* menjadi "*Enemy*"

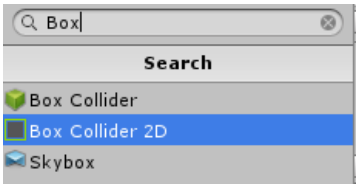
Gambar 9.24 Menambahkan *Enemy*

26. Buat sebuah *file script* didalam folder *Script* beri nama, kemudian *drag* dan masukkan ke dalam game *object* "*Enemy*"



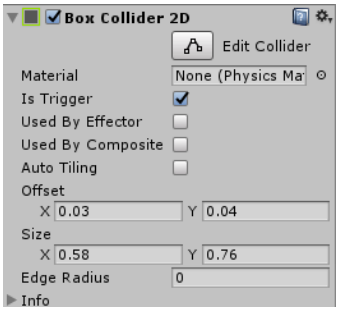
Gambar 9.25 Membuat *Script*

27. Tambahkan sebuah komponen bernama *Box Colider 2D* dalam *inspector* game objek *Enemy*



Gambar 9.26 Menambahkan *Box Collider 2D*

28. Atur sedikit *collider* tersebut seperti ukurannya diubah jika terlalu besar, centang bagian *Is Trigger*

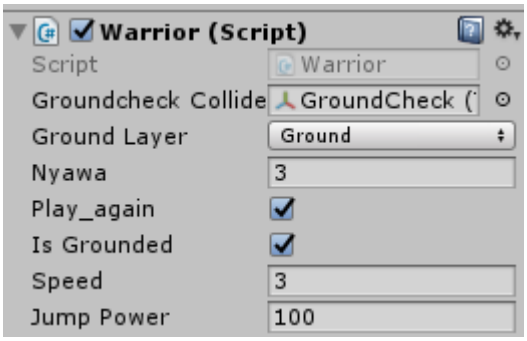


Gambar 9.27 Mengatur sedikit *collider*

29. Buka *file script* (*Enemy.cs*) dan isikan *source code* dibawah ini

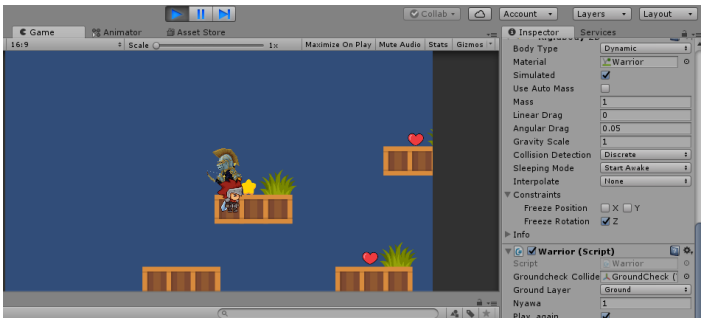
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Enemy : MonoBehaviour {

    Warrior wr;
    // Use this for initialization
    void Start()
    {
        //Warrior adl nama dari game object yg ada di
hierarchy
        wr =
GameObject.Find("Warrior").GetComponent<Warrior>();
    }
    // Update is called once per frame
    void Update()
    {
    }
    void OnTriggerEnter2D(Collider2D other)
    {
        if (other.transform.tag == "Player")
        {
            wr.nyawa--;
        }
        if (wr.nyawa < 0)
        {
            wr.play_again = true;
        }
    }
}
```

30. Klik game *object warrior*, pergi ke *Inspector* dan ubah nilai Nyawa menjadi 3 pada *warrior(Script)*



Gambar 9.28 Mengubah nilai nyawa

31. Jika di *Play*, Ketika *enemy*not berjalan dan mengenai si enemy maka nyawa akan berkurang satu



Gambar 9.29 Tampilan game ketika di Run

9.4 Kesimpulan

1. metode bone based untuk menganimasikannya adalah dengan menggerakkan setiap objek yang ada pada *warrior* pada seriap frame
2. Animation windows di Unity digunakan untuk memodifikasi klip animasi langsung dalam unity, didalamya terdapat klip animasi yang memunginkan untuk dibuat klip lebih dari satu, dalam satu klip juga terdapat timeline yang dapat digunakan untuk mengatur frame animasi.
3. Animator adalah komponen game yang berfungsi untuk mengendalikan state dari objek yang ingin di animasikan, state ini contohnya dari state diam, jalan, melompat, menyerang dll.

Nilai	Asisten Laboratorium
	(M.Rafi Faddilani) 2118144