TUGAS PERTEMUAN: 10

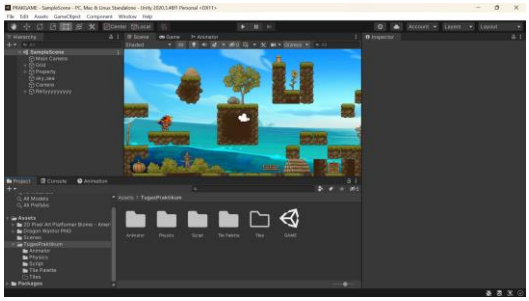
RESPAWN AND AI ENEMY ATTACK

NIM	:	2118132
Nama	:	Madelbertha Fridolin Puka
Kelas	:	D
Asisten Lab	:	Abidzar Al Giffari (2218013)

10.1 Tugas 10: Membuat Enemy Attcak

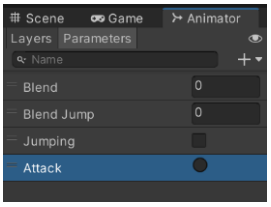
A. Membuat Mekanisme Attack

1. Buka kembali project Bab 9 kemarin, untuk melanjutkan tugas bab 10.



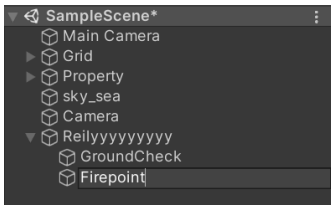
Gambar 10.1 Tampilan Project Bab 9

2. Kemudian pada menu *Tab Animator*, tambahkan parameter *Trigger*, *rename* menjadi *Attcak*



Gambar 10.2Tampilan Penambahan Attack

3. Setelah menambahkan parameter *Attack*, Langkah selanjutnya adalah membuat Layer *Game object* baru didalam folder player, Klik kanan pilih *Create Empty* lalu Rename menjadi *Firepoint*

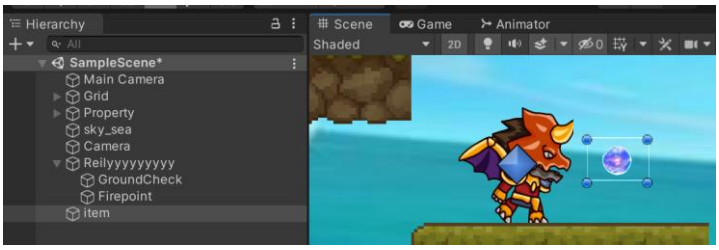


Gambar 10.3Tampilan Penambahan Firepoint

4. Pada menu *Hierarchy* klik Firepoint untuk *setting* pada *Inspector*, Ubah *Icon* Menjadi titik, atur letak titik didepan *player*

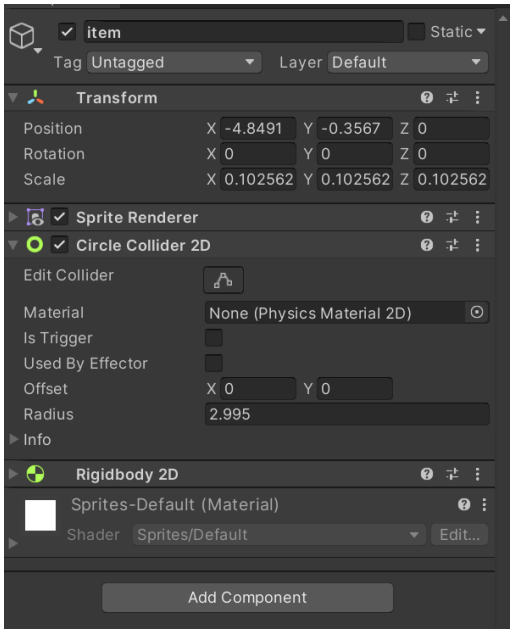
Gambar 10.4Tampilan *Setting* Firepoint

5. Pada menu *Hierarchy* Tambahkan *file* item.



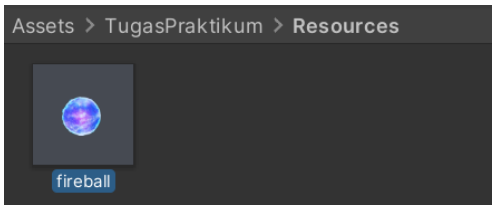
Gambar 10.5Tampilan Penambahan *File* Item.

6. Klik item untuk menambahkan *Component Circle Collider* 2D, dan *Riggidbody* 2D, *Setting* sesuai gambar dibawah ini



Gambar 10.6Tampilan *Setting* Item

7. Buat Folder baru *Resources* di menu Project, kemudian *drag and drop fireball* kedalam folder Resources, dan hapus *fireball* pada *Hierarchy*



Gambar 10.7Tampilan Folder Resources

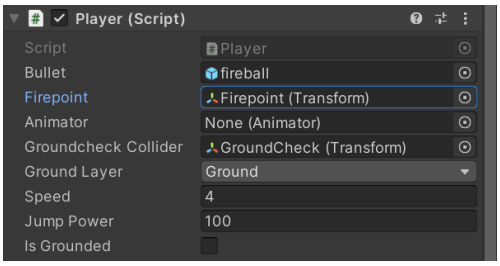
8. Pada *Script Player* ubahlah *Script* seperti dibawah ini

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
public class Player : MonoBehaviour
{
    public GameObject bullet;
    public Transform Firepoint;
    public Animator animator;
    public int nyawa;
    [SerializeField] Vector3 respawn_loc;
    public bool play_again;

    Rigidbody2D rb;
    [SerializeField] Transform groundcheckCollider;
    [SerializeField] LayerMask groundLayer;
    const float groundCheckRadius = 0.2f; // +
    [SerializeField] float speed = 1;
    [SerializeField] float jumpPower = 100;
    float horizontalValue;
    [SerializeField] bool isGrounded; // +
    bool facingRight;
    bool jump;
    Vector3 scaleValue = new Vector3(0.2280304f,
0.2473785f, 0.321396f); // Inisialisasi nilai awal
    private void Awake()
    {
        rb = GetComponent<Rigidbody2D>();
        animator = GetComponent<Animator>();
        respawn_loc = transform.position;
    }
    void playagain()
    {
        if (play_again == true){
            nyawa = 3;
            transform.position = respawn_loc;
            play_again = false;
        }
    }
    void Update()
    {
        horizontalValue                                    =
Input.GetAxisRaw("Horizontal");
        if (Input.GetButtonDown("Jump"))
        {
```

```
             jump = true;
             animator.SetBool("Jumping", true);
        }

        else if (Input.GetButtonUp("Jump"))
              jump = false;
        if (Input.GetKeyDown(KeyCode.C))
        {
             StartCoroutine(Attack());
        }
        if (nyawa < 0)
        {
            playagain();
        }
        if(transform.position.y < -10){
            play_again = true;
            playagain();
        }
    }
    void FixedUpdate()
    {
        GroundCheck();
        Move(horizontalValue, jump);
        animator.SetFloat("Blend",
Mathf.Abs(rb.velocity.x));
        animator.SetFloat("Blend Jump", rb.velocity.x);
    }
    IEnumerator Attack()
    {

        animator.SetTrigger("Attack");
        yield return new WaitForSeconds(0.25f);

        float direction = 1f;

        GameObject   fireball   =   Instantiate(bullet,
Firepoint.position, Quaternion.identity);
        fireball.GetComponent<Rigidbody2D>().velocity
= new Vector2(direction * 10f, 0);

        Destroy(fireball, 2f);
    }
    void GroundCheck()
    {
        isGrounded = false;
        Collider2D[]            colliders            =
Physics2D.OverlapCircleAll(groundcheckCollider.positio
n, 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)
        {
            // Ubah nilai scaleValue sesuai kebutuhan
            scaleValue = new Vector3(-0.2280304f,
0.2473785f, 0.321396f);
            facingRight = false;
        }
        else if (!facingRight && dir > 0)
        {
            // Ubah nilai scaleValue sesuai kebutuhan
            scaleValue = new Vector3(0.2280304f,
0.2473785f, 0.321396f);
            facingRight = true;
        }
        //    Terapkan    nilai    scaleValue    ke
transform.localScale
        transform.localScale = scaleValue;
        #endregion
    }
}
```

9. Pada *Inspector Player*, Ubah seperti dibawah ini, Dimana *Bullet* berisi object yang akan ditembak sedangkan *fire point* adalah titik tembak pertama



Gambar 10.8Tampilan *Inspector Player*

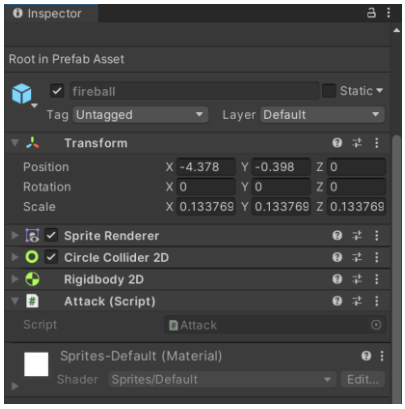
10. Buat *Script Attack* pada folder *Script*

Gambar 10.9 Tampilan *Script Attack*

11. Tambahkan *Script Attack* dibawah ini

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

public class Attack : MonoBehaviour
{
    private void OnTriggerEnter2D(Collider2D collision)
    {
        if (collision.gameObject.CompareTag("Enemy"))
        {
            Destroy(gameObject);
            Destroy(collision.gameObject);
        }
    }
}
```

12. Didalam folder Resource Tambahkan *Script Attack* di *Prefab fireball*, dengan cara Klik *fireball* kemudian pada menu *Inspector* arahkan *Script Attack* kedalam *Inspector*

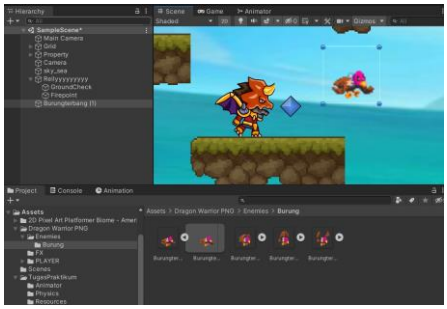


Gambar 10.10 Tampilan *Inspector* Fireball

13. Kita akan menggunakan 2 *Enemy Attack* yaitu Burung terbang dan Simer. Kedua *enemy* ini akan kita buat dengan animasinya masing-masing. Buatlah *File* Enemy pada Folder Animator yang ada pada TugasPraktikum.

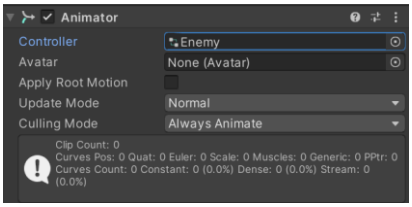
Gambar 10.11 Tampilan *Animator*

14. Setelah itu buka folder Enemies lalu pilih Burung. Setelah masuk kedalam foldernya, pilih burung dengan nama Burungterbang(1) dan masukan kedalam *hierarchy*



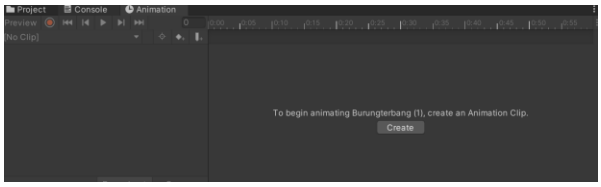
Gambar 10.12 Tampilan *Hierarchy*

15. Pada bagian *Animator*, ubahlan *controller* menjadi Enemy



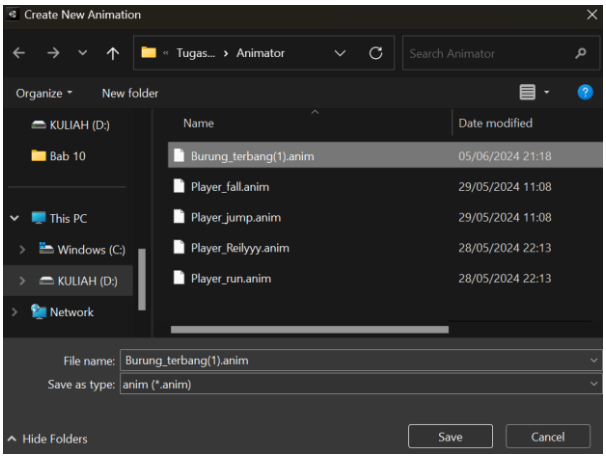
Gambar 10.13 Tampilan *Animator*

16. Bukalah bagian *Animation* lalu pada bagian kanan, pilih *Create*.



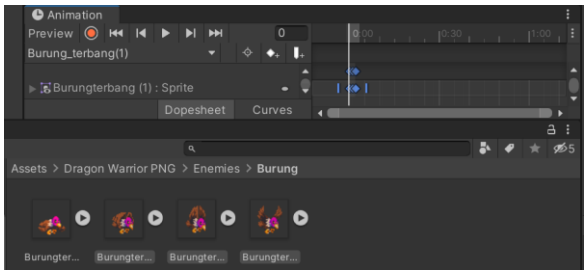
Gambar 10.14 Tampilan *Animation*

17. Simpan Animation dengan nama Burung_terbang(1)



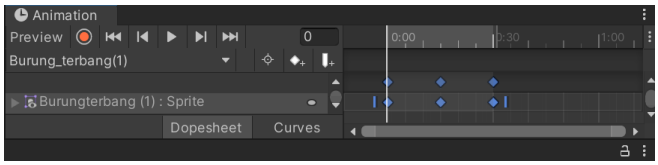
Gambar 10.15 Tampilan *Create New Animation*

18. Setelah itu pilih Burungterbang(2)-Burungterbang(4) lalu drag and drog kedalam animation



Gambar 10.16 Tampilan *Animation*

19. Atur supaya *animation*-nya berada pada frame 0-30

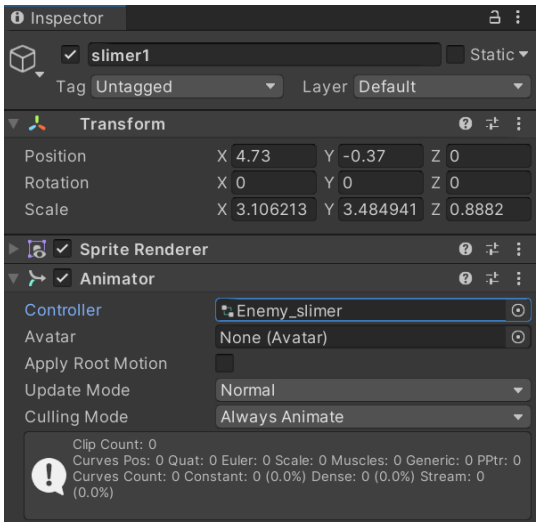


Gambar 10.17 Tampilan *Frame*

20. Selanjutnya tambahkan lagi *enemy* yang kedua yaitu *enemy* Slimer.

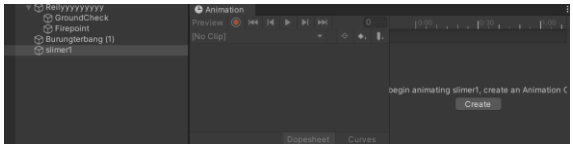
Gambar 10.18 Penambahan *Enemy* Slimer

21. Pada *inspector* Slimer, ubah *controller* yang ada pada Animator diganti menjadi Enemy_slimer.



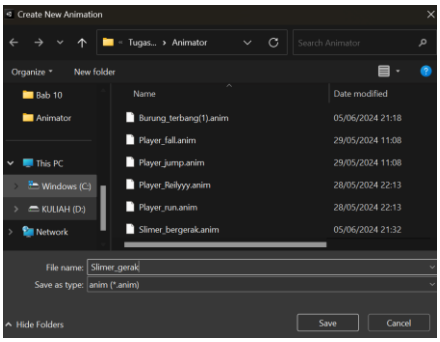
Gambar 10.19 Tampilan *Controller Animator*

22. Pada halaman *Animation* pilih *Create*



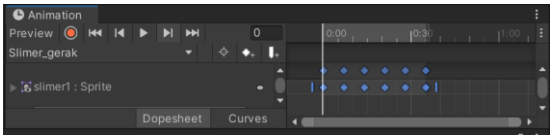
Gambar 10.20 Tampilan *Create Animation*

23. Setelah itu pada halaman Create New Animation, masukan nama menjadi Slimer_gerak



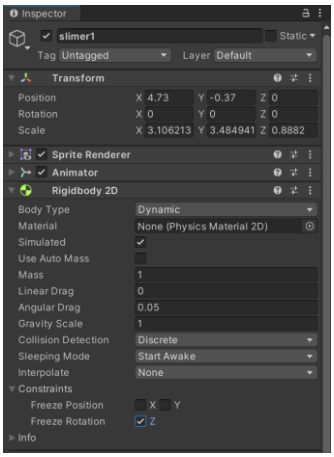
Gambar 10.21 Tampilan *Create New Animation*

24. Pada halaman *animation*, masukan Slimer kedalam *frame*. Lalu atur waktunya menjadi 35



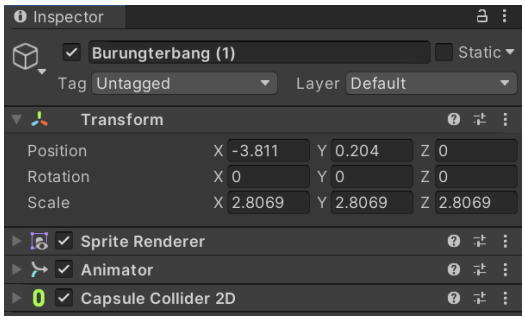
Gambar 10.22 Tampilan *Animation* Slimer

25. Selanjutnya pada *inspector* slimer, tambahkan kompenen *Capsule Collider* 2D *Rigidbody* 2D lalu pada *simulated* dan *freeze rotation* Z diberikan centang.



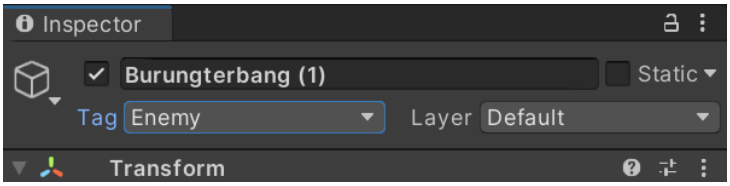
Gambar 10.23 Tampilan *Add Component*

26. Selanjutnya pada Burungterbang(1) tambahkan juga *capsule collider* 2D



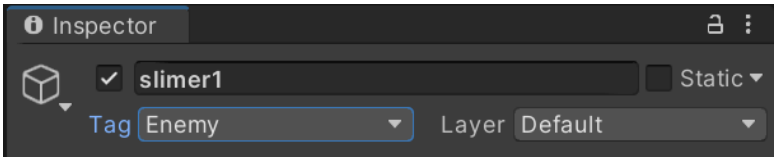
Gambar 10.24 Tampilan *Inspector* Burungterbang(1)

27. Pada bagian *tag* ubahlah menjadi *Enemy*



Gambar 10.25 Tampilan *Inspector* Burungterbang(1)

28. Pada Slimer1, ubahlah *inspector Tag* menjadi *Enemy*



Gambar 10.26 Tampilan *Inspector* slimer1

29. Tembak Enemy dengan menekan Tombol C untuk menghancurkan musuh

Gambar 10.27 Tampilan Hasil

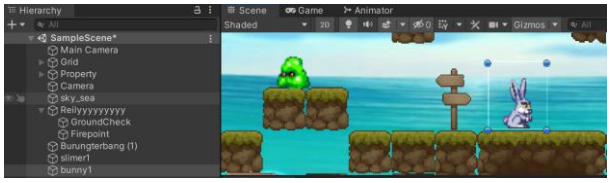
B. Membuat Enemy AI

1. Enemy Behavior NPC

   a. Cari sebuah *sprite* pack Bernama enemy dan buka folder bernama "Kelinci"

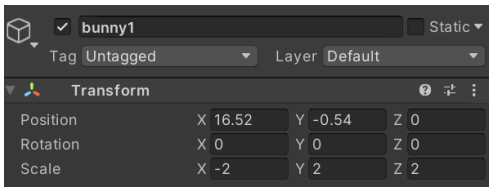
Gambar 10.28 Tampilan Folder Kelinci

   b. Tambahkan Kelinci kedalam *Hierarchy*



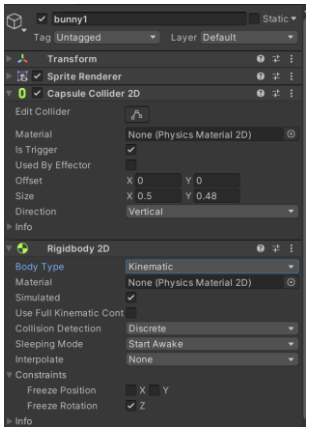
Gambar 10.29 Tampilan *Hierarchy*

   c. Pada *inspector* atur *transform scale* menjadi seperti berikut



Gambar 10.30 Tampilan *Inspector* bunny1

   d. Tambahkan sebuah komponen bernama *Capsule Colider* 2D dan *Rigidbody* dalam *inspector* game objek Kelinci

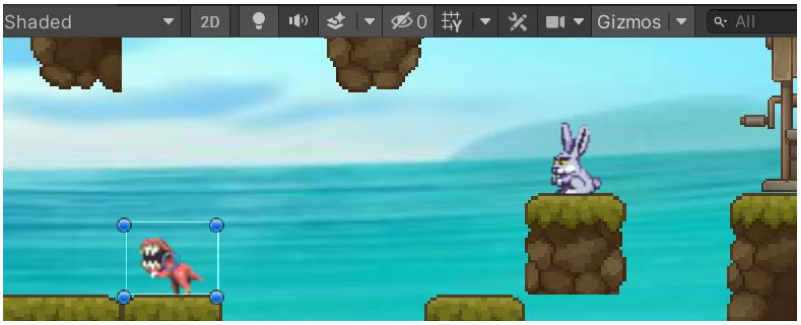


Gambar 10.31 Tampilan *Add Component*

e. Atur sedikit *collider* tersebut seperti ukurannya diubah jika terlalu besar, dan pada *Body Type* Ubah menjadi *Kinematic*

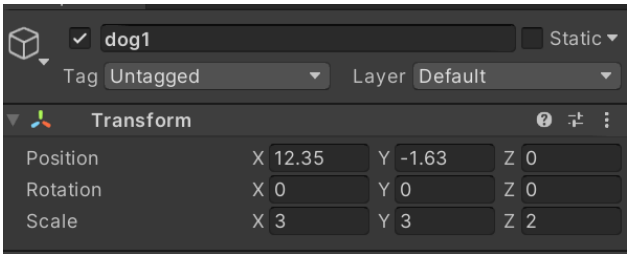
Gambar 10.32 Tampilan *Settings*

f. Tambahkan lagi *enemy* baru yang bernama Dog



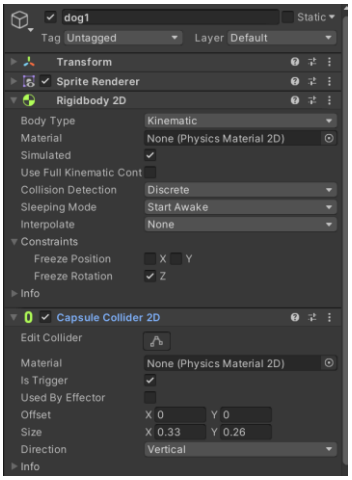
Gambar 10.33 Tampilan Penambahan Dog

g. Pada *inspector* dog, ubahlah scale menjadi seperti berikut ini



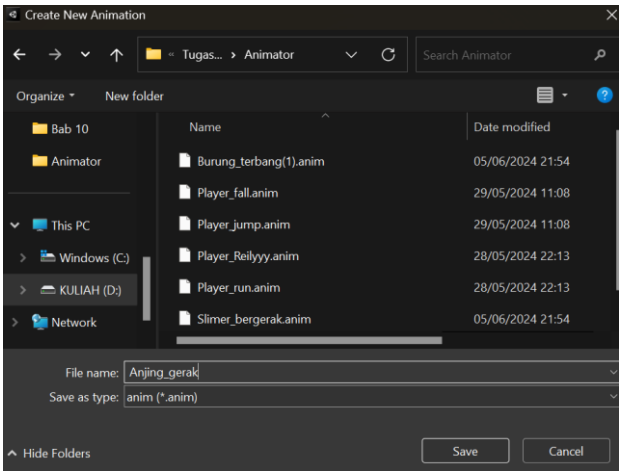
Gambar 10.34 Tampilan *Inspector* Dog

h. Lalu tambahkan juga komponen rigidbody 2D dan capsule collider 2D dan aturlah seperti berikut ini



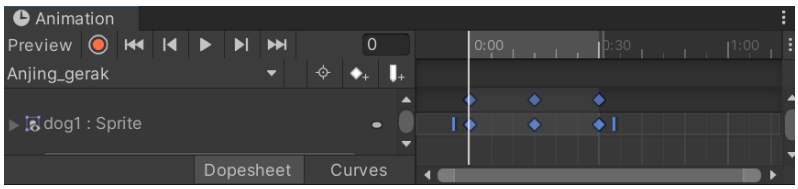
Gambar 10.35 Tampilan *Add Component*

i. Untuk membuat animasi pada enemy, buatlah animator untuk dog dan berilah nama Anjing_gerak



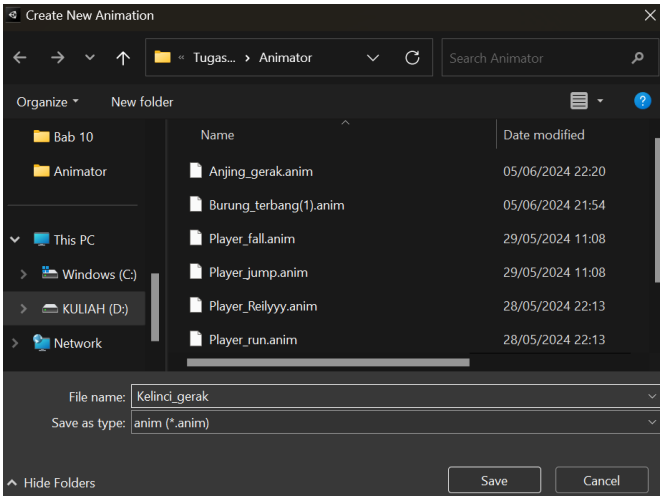
Gambar 10.36 Tampilan *Create New Animation*

j. Pada bagian *animator*, masukan dog yang ada pada folder Anjing ke dalam *Animation* lalu pada *frame* buatlah menjadi 0-30



Gambar 10.37 Tampilan *Animation* Anjing_gerak

k. Buatlah animasi baru untuk enemy kelinci. Lalu berikan nama Kelinci_gerak

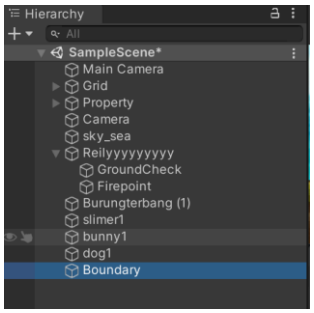


Gambar 10.38 Tampilan *Create New Animation*

l. Pada *animation* tambahkan bunny kedalam halaman *animation* dan atur *frame* menjadi 0-30

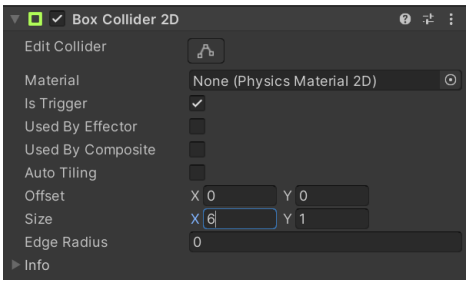
Gambar 10.39 Tampilan *Animation*

m. *Create Empty object* pada *Hierarchy*, *Rename* Menjadi *Boundary*



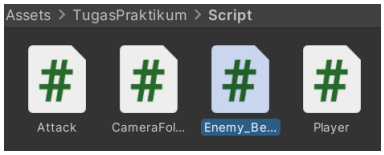
Gambar 10.40 Tampilan *Hierarchy*

n. Tambahkan *Box Collider* 2D pada *Boundary*, centang pada *Is Trigger* lalu atur sesuai keinginan pada *size* dan *offside*



Gambar 10.41 Tampilan *Settings Boc Collider* 2D

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



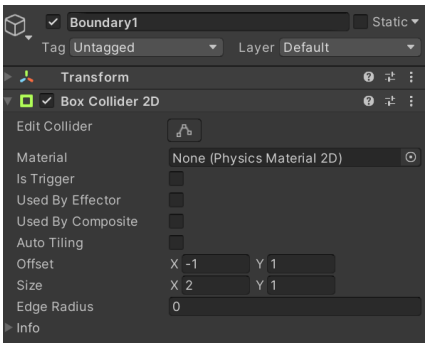
Gambar 10.42 Tampilan *Script* Enemy_Behavior

p. Tambahkan *Script* dibawah ini

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

```
public class Enemy_Behavior : MonoBehaviour
{
    [SerializeField] float moveSpeed = 1f;
    Rigidbody2D rb;

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

    void Update()
    {
        if (isFacingRight())
        {
            rb.velocity = new Vector2(moveSpeed,
0f);
        }
        else
        {
            rb.velocity = new Vector2(-moveSpeed,
0f);
        }
    }

    private bool isFacingRight()
    {
        return transform.localScale.x >
Mathf.Epsilon;
    }

    private void OnTriggerExit2D(Collider2D
collision)
    {
        transform.localScale = new Vector2(-
transform.localScale.x, transform.localScale.y);
    }
}
```

q. Masukan Enemy_Behavior ke dalam Bunny dan dog. Setelah itu buatlah boundary1 yang akan digunakan untuk Bunny. Setelah itu atur *inspector* Boundary1



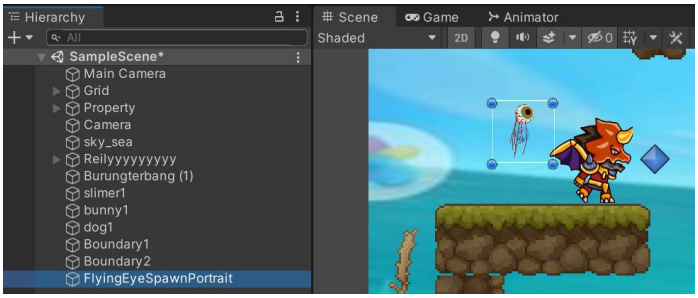
Gambar 10.43 Tampilan *Inspector* Boundary1

r. Jalankan Programam

Gambar 10.44 Tampilan Akhir

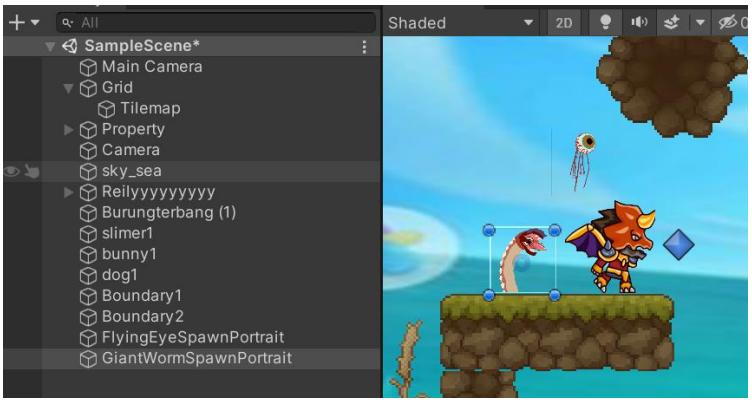
2. Enemy AI

a. Cari sebuah *sprite pack* player lalu tambahkan 'FlyingEyeSpawnPortrait' pada *Hierarchy*



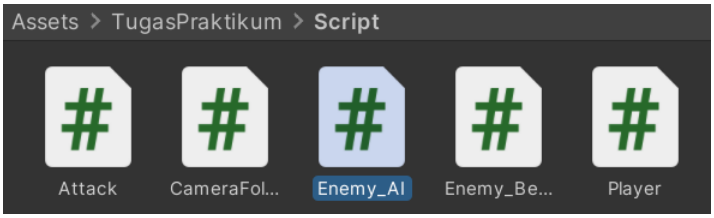
Gambar 10.45 Tampilan *Player Jump*

b. Tambahkan juga *enemy* 'GianWormSpawnPortrait' kedalam *Hierarchy*



Gambar 10.46 Tampilan Penambahan *Enemy*

c. Buat *Script* Enemy_AI pada folder Praktikum - *Script*



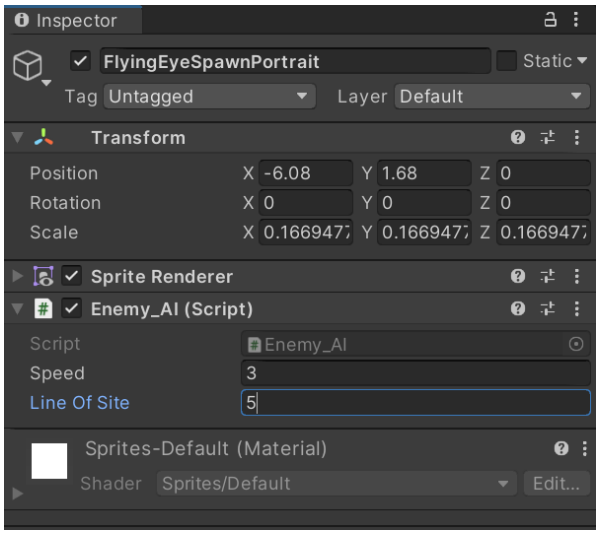
Gambar 10.47 Tampilan Folder Script

d. Tambahkan *Script* pada Enemy_Ai seperti dibawah ini

```
using UnityEngine;

public class Enemy_AI : MonoBehaviour
{
    public float speed; // Kecepatan gerakan musuh
    public float lineOfSite; // Jarak penglihatan
musuh
    private Transform player; // Transform dari
pemain
    private Vector3 initialPosition; // Posisi awal
musuh

    // Use this for initialization
    void Start()
    {
        // Mencari pemain berdasarkan tag
        player                                    =
GameObject.FindGameObjectWithTag("Player").transfor
m;
        // Menyimpan posisi awal musuh
        initialPosition = transform.position;
    }

    // Update is called once per frame
    void Update()
    {
        // Menghitung jarak antara musuh dan pemain
        float           distanceToPlayer          =
Vector3.Distance(player.position,
transform.position);

        // Jika pemain berada dalam jarak penglihatan
musuh
        if (distanceToPlayer < lineOfSite)
        {
            // Musuh bergerak menuju pemain
            transform.position                    =
Vector3.MoveTowards(transform.position,
player.position, speed * Time.deltaTime);
        }
        else
        {
            // Musuh kembali ke posisi awal
            transform.position                    =
Vector3.MoveTowards(transform.position,
initialPosition, speed * Time.deltaTime);
```

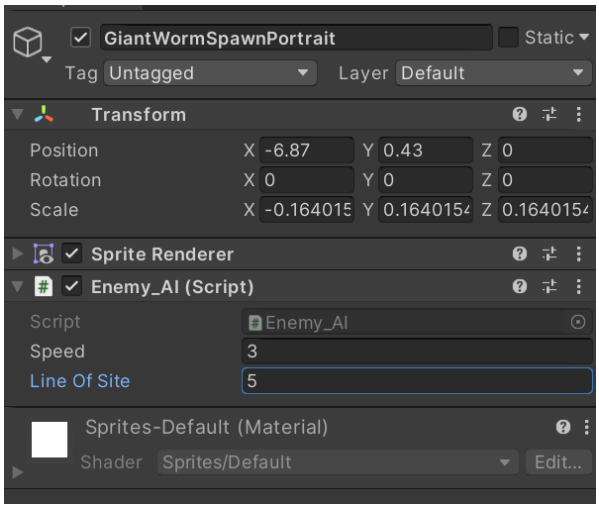
```
        }
    }

    // Untuk menggambar jarak penglihatan musuh di
editor
    private void OnDrawGizmosSelected()
    {
        Gizmos.color = Color.red;
        Gizmos.DrawWireSphere(transform.position,
lineOfSite);
    }
}
```

e. Pada *Inspector* Enemy_Ai, Atur *Speed* juga *Line of Site* untuk menentukan jarak dan *speed* pada *enemy*



Gambar 10.48 Tampilan *Drag Player*

f. Ubahlah juga untuk *Enemy* yang kedua



Gambar 10.49 Tampilan *Inspector Enemy* Ke-2

g. *Running* Game, maka *enemy* AI akan mengikuti Gerakan *Player*

Gambar 10.50 Tampilan Hasil *Running*

3. Respawn

a. Buka file *script* (Player.cs) tambahkan variabel nyawa seperti dibawah ini

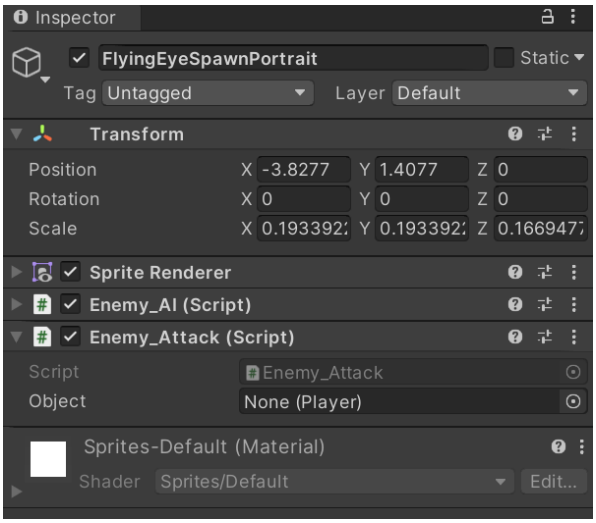
Gambar 10.51 Tampilan Folder Script

b. Tambahkan file *script* (Enemy_Attacked.cs) dan isikan *source code* dibawah ini

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

public class Enemy_Attack : MonoBehaviour
{
    [SerializeField] private Player Object;

    void Start()
    {
        if (Object == null)
        {
            Object                                          =
GameObject.FindWithTag("Player").GetComponent<Playe
r>();
        }
    }

    void OnTriggerEnter2D(Collider2D other)
    {
        if (other.CompareTag("Player"))
        {
            Object.nyawa--;

            if (Object.nyawa < 0)
```

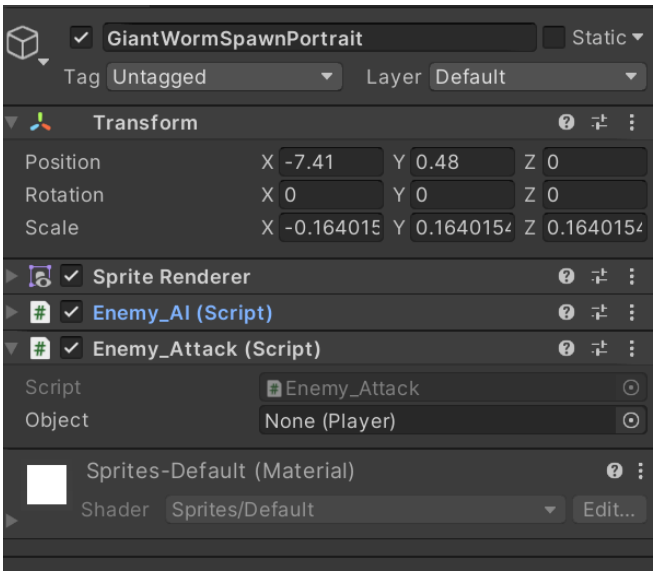
```
                {
                    Object.play_again = true;
                }
            }
        }
    }
```

c. Pada *hierarchy* FlyingEyeSpawnPortrait Tambahkan *Script enemy attack*, arahkan object pada player.



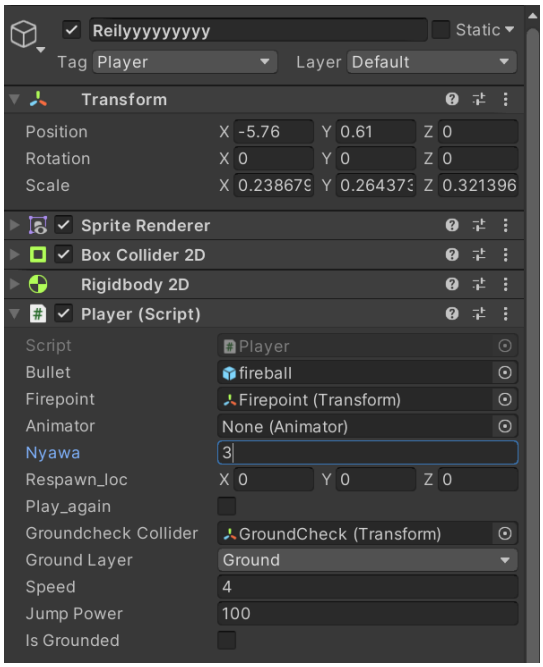
Gambar 10.52 Tampilan *Inspector*

d. Pada *hierarchy* GiantWormSpawnPotrait Tambahkan *Script enemy attack*, arahkan object pada *player*.



Gambar 10.53 Tampilan *Inspector*

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



Gambar 10.54 Tampilan Nyawa

f. Jika di *play*, *Player* mengenai atau menyentuh Anjing dan Kelinci sebanyak 3 kali maka nyawa akan berkurang 1 dan jika nyawa kurang dari 0 maka akan *reswpawn* ke titik awal

Gambar 10.55 Tampilan Hasil *Enemy Attack*

KUIS

Soucer Code

```
using UnityEngine;

public class PlayerAttack : MonoBehaviour
{
    public float attackRange = 2.0f;
    public int attackDamage = 10;

    void Update()
    {
        if (Input.GetButtonDown("Fire1"))
        {
            PerformMeleeAttack();
        }
    }

    void PerformMeleeAttack()
    {
        RaycastHit hit;
        if                  (Physics.Raycast(transform.position,
transform.forward, out hit, attackRange))
        {
            // Cek apakah objek yang terkena adalah musuh
            Enemy enemy = hit.transform.GetComponent<Enemy>();
            if (enemy != null)
            {
                // Kurangi health musuh
                enemy.TakeDamage(attackDamage);
            }
        }
    }
}
```

Analisa

Pada codingan diatas, terdapat fungsi *Update* yang digunakan untuk memeriksa apakah tombol *Fire1* atau tombol kiri *mouse* ditekan dengan menggunakan *Input.GetButtonDown("Fire1").* Jika tomobol ditekan maka fungsi *PerformMeleeAttack* akan dipanggil.

Selanjutnya fungsi *PerformMeleeAttack* melakukan operasi serangan. Yang pertama akan dideklarasikan variabel *hit* yang akan menyimpan informasi tentang objek yang terkena *raycast*. Kemudian jika *raycast* mengenai sesuatu, kita memeriksa apakah objek terkena *raycast* kalau mengenai maka nilai pada *attack demage* akan berkurang.