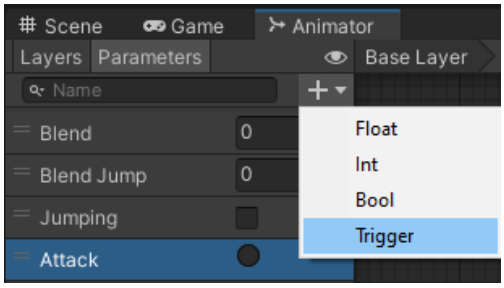
TUGAS PERTEMUAN: 10

RESPAWN DAN AI ENEMY ATTACK

NIM	:	2118030
Nama	:	Putra Prasetya Utama
Kelas	:	D
Asisten Lab	:	Aprillia Dwi Dyah S. (2118143)

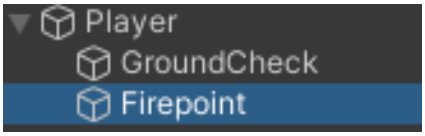
10.1 Tugas 1 : Langkah-langkah Membuat Player Menembak 2 Arah dan Ancang-Ancang Sebelum Menembak

1. Buka *project* sebelumnya, lalu ke *tab Animator* tambahkan parameter *Tringger* dan beri nama *Attack*.



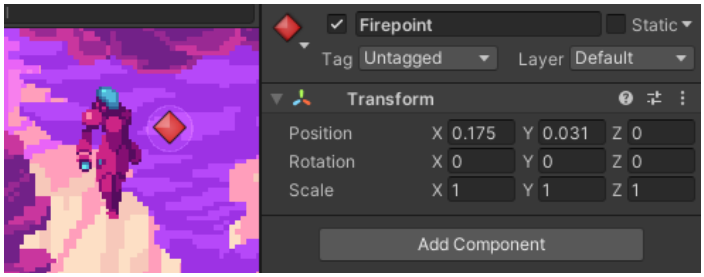
Gambar 10.1 Membuat Parameter *Attack*

2. Masuk ke *Player* pada hirarki lalu buat *Create Empty* dan ubah namanya menjadi *Firepoint*.



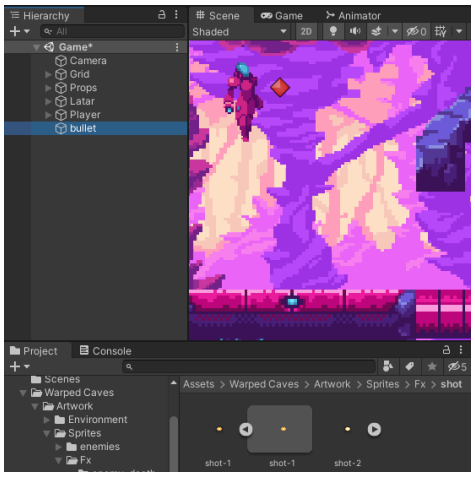
Gambar 10.2 Membuat *Firepoint*

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



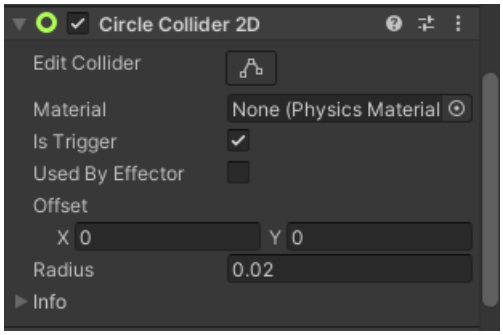
Gambar 10.3 Mengatur Posisi Untuk Peluru Keluar

4. Pergi ke folder shot pada aset. Lalu *drag and drop shot*-1 kedalam hirarki. Ubah namanya menjadi peluru.



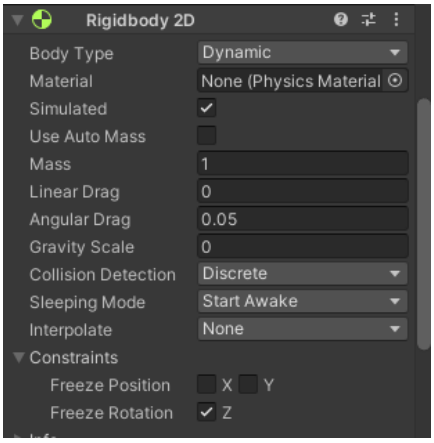
Gambar 10.4 *Drag and Drop Shot*-1 Ke Hirarki

5. Klik peluru lalu tambahkan *Component Circle Collider* 2D. centang *is trigger*.



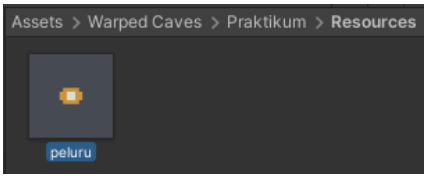
Gambar 10.5 Menambahkan Komponen *Circle Collider* 2D

6. Lalu tambahkan komponen *Riggidbody* 2D dan atur *gravity scale* menjadi 0 dan centang *freeze rotation* Z.



Gambar 10.6 Mengatur Rigidbody 2D

7. Buat folder baru didalam folder Praktikum dan beri nama *Resources*. *Drag and drop bullet* yang ada di hirarki kedalam folder *Resources*. Lalu hapus peluru yang ada di hirarki

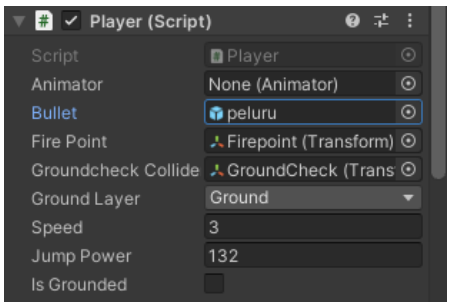


Gambar 10.7 Menambahkan Folder *Resources*

8. Pada *script Player* tambahkan *source code* dibawah ini.

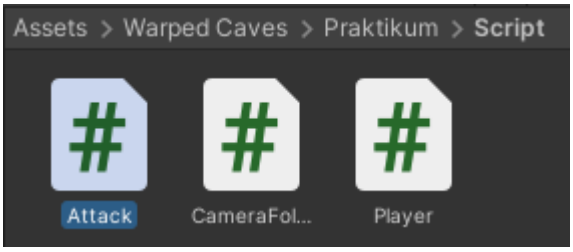
```
//tambahkan source code berikut untuk deklarasi variabel
public Animator animator;
public GameObject bullet;
public Transform firePoint;

//tambahkan dibawah fungsi FixedUpdate
IEnumerator Attack(){
  animator.SetTrigger("Attack");
  yield return new WaitForSeconds(0.25f);

  // agar player dapat menembak ke kanan dan kiri
  float direction = facingRight ? 1f : -1f;

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

  Destroy(peluru, 2f);
}
//tambahkan didalam Void Update
if (Input.GetKeyDown(KeyCode.C)){
   StartCoroutine(Attack());
}
```

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



Gambar 10.8 Mengubah Script Pada Inspector Player

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

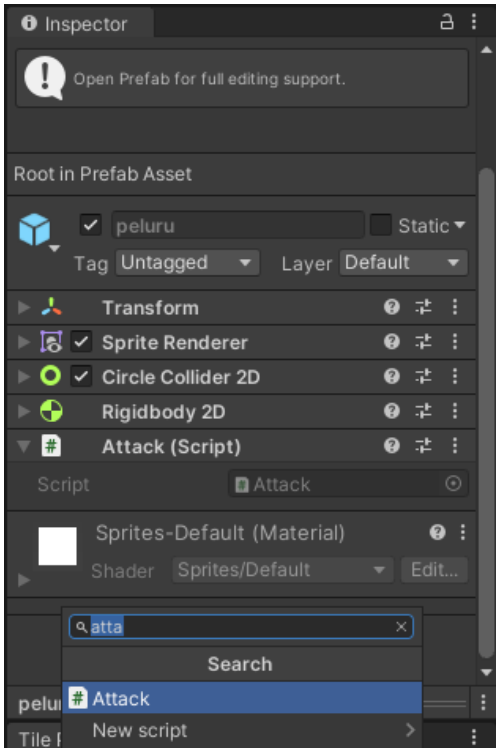


Gambar 10.9 Menambahkan Script Attack

11. Tambahkan *source code* berikut pada *script Attack* yang sudah dibuat.

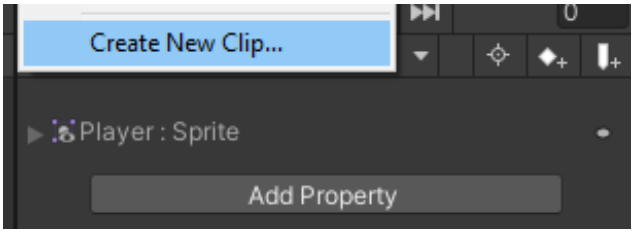
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

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

12. Klik peluru pada folder *Resources* lalu pergi ke *Inspector* lalu tambahkan komponen *script Attack*.



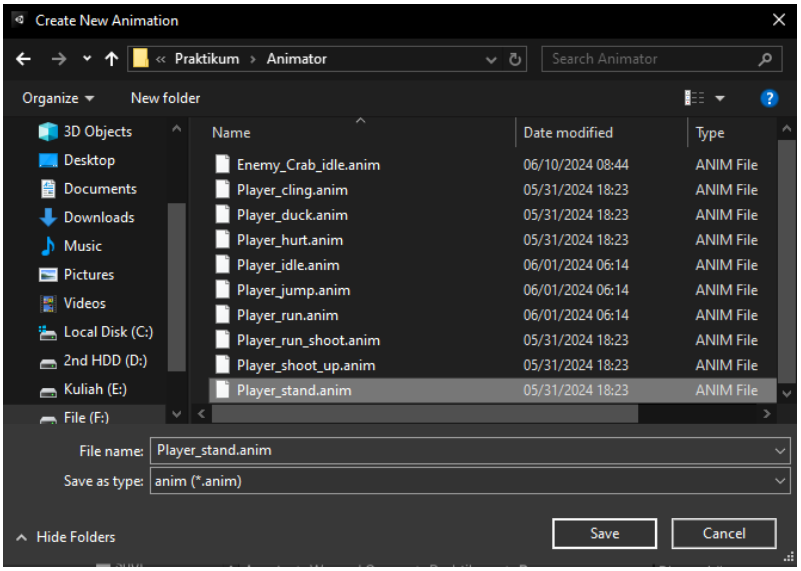
Gambar 10.10 Menambahkan Komponen Script *Attack*

13. Klik *player* lalu pergi ke panel *Animation* lalu *Create New Clip*.



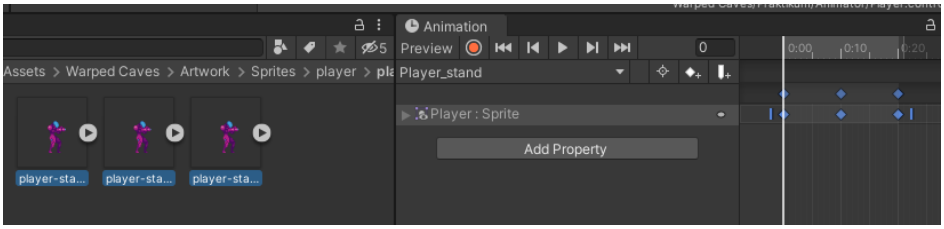
Gambar 10.11 Membuat *Clip* Baru

14. Lalu beri nama *Player_stand* dan simpan di folder *Animator*.



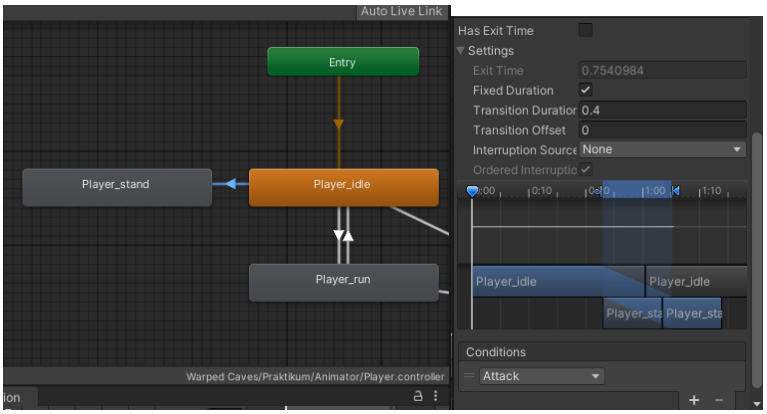
Gambar 10.12 Memberi Nama Clip *Player_stand*

15. Lalu cari folder *player-stand*. Pilih semua animasinya dan lakukan *drag and drop* ke *clip* yang sudah dibuat sebelumnya. Atur menjadi 0.20 detik.



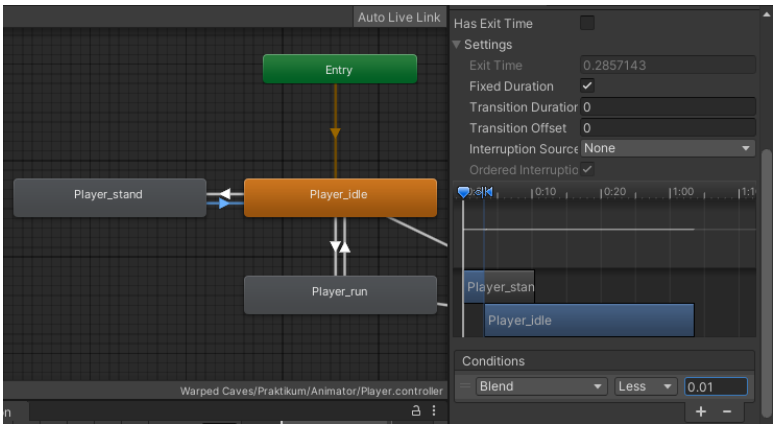
Gambar 10.13 Menambahkan Animasi Untuk Menembak

16. Lalu pergi ke *tab Animator* dan buat transisi dari *Player_idle* ke *Player*_stand. Lalu atur seperti gambar dibawah ini.



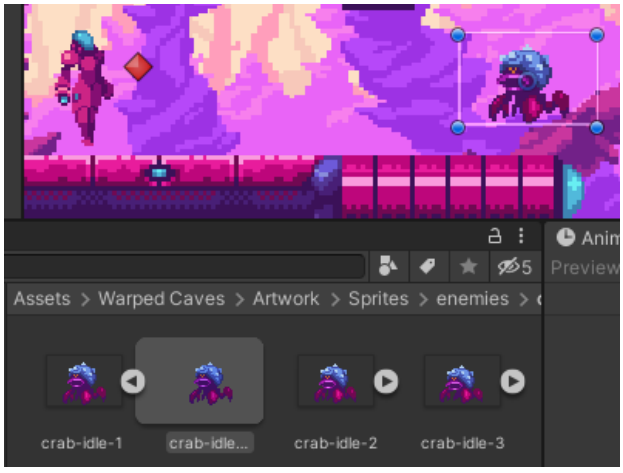
Gambar 10.14 Mengatur Transisi *Idle* ke *Stand*

17. Buat transisi lagi dari *Player_stand* ke *Player_idle*. Lalu atur buat kondisi *Blend* dan *Less* dengan *value* 0.01.



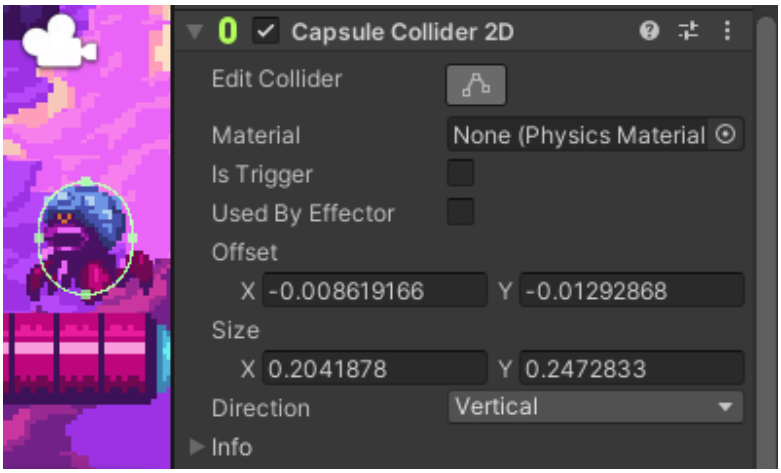
Gambar 10.15 Mengatur Transisi *Stand* ke *Idle*

18. Cari folder *enemies* lalu tambahkan musuh didepan *player*.



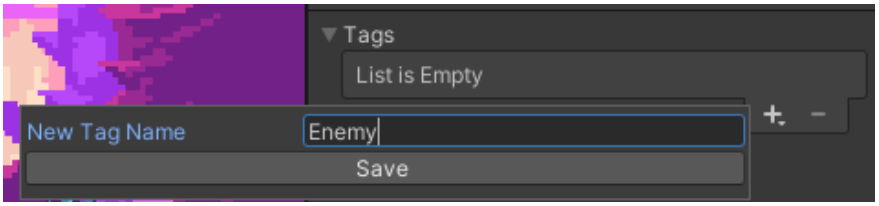
Gambar 10.16 Menambahkan Musuh Crab

19. Lalu klik *crab-idle*-1 dan tambahkan komponen *Collider* 2D apapun itu lalu atur *box* sesuai dengan musuh.



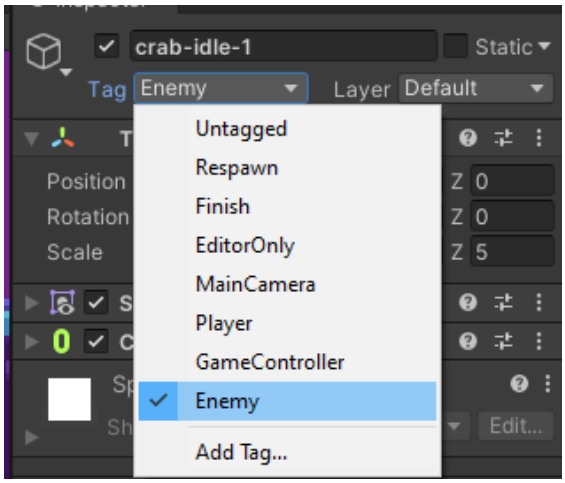
Gambar 10.17 Menambahkan Komponen *Collider* 2D

20. Tambahkan *Tag Enemy* dengan cara Pilih *Add Tag*, kemudian *add tag to the list*, Tuliskan *Enemy*.



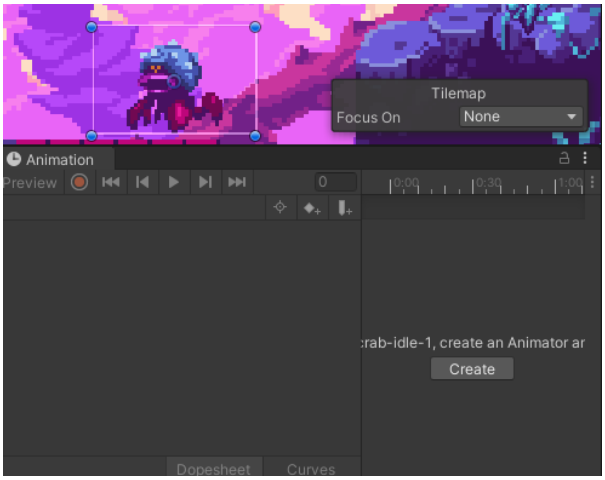
Gambar 10.18 Menambahkan *Tag Enemy*

21. Lalu atur *tag crab-idle-1* menjadi *Enemy* yang sudah dibuat sebelumnya.



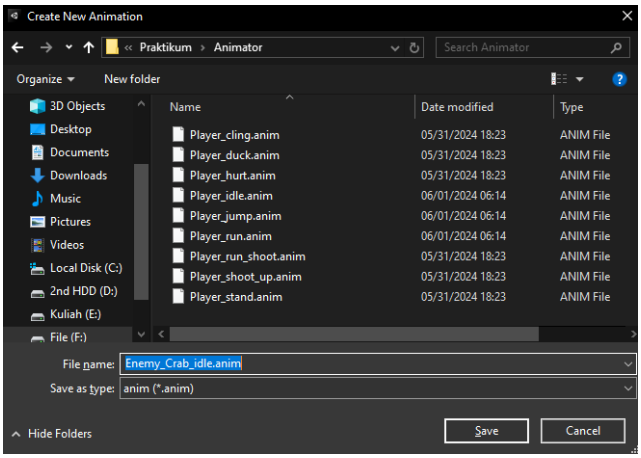
Gambar 10.19 Merubah Tag *crab-idle-1*

22. Lalu klik pada *crab-idle*-1. Pergi ke panel *Animation* dan klik *Create*.



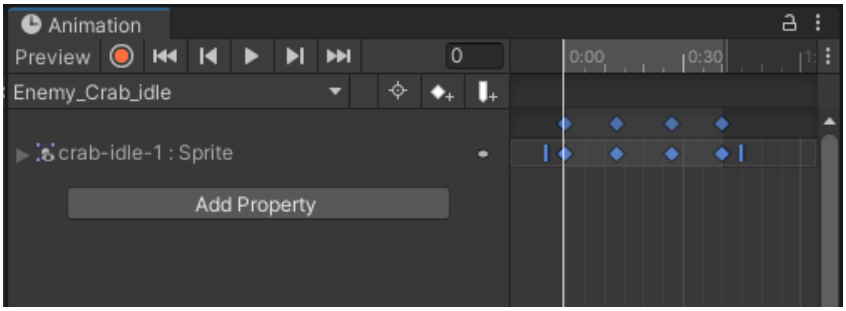
Gambar 10.20 Menambahkan Animation Baru Ke *crab-idle*-1

23. Simpan ke folder *Animator* dan beri nama Enemy_Crab_idle.anim



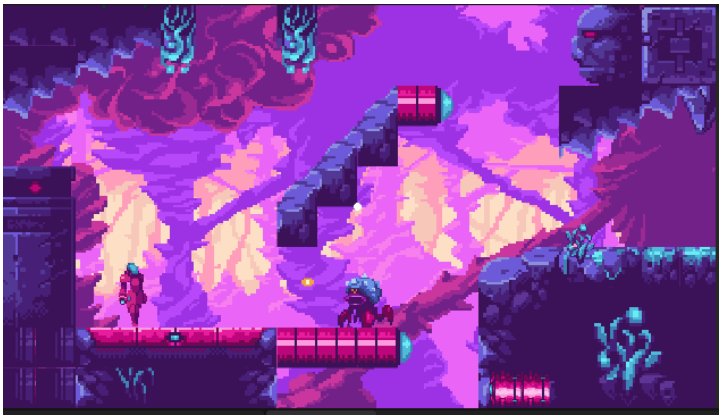
Gambar 10.21 Menyimpan Animasi Idle Enemy

24. Lalu pergi ke folder *enemies* bagian *crab-idle*. Lalu pilih semuanya dan *drag and drop* ke panel *Animation*. Lalu atur agar menjadi 0.40 detik.



Gambar 10.22 Mengatur Animasi *Idle* Dari *Enemy Crab*

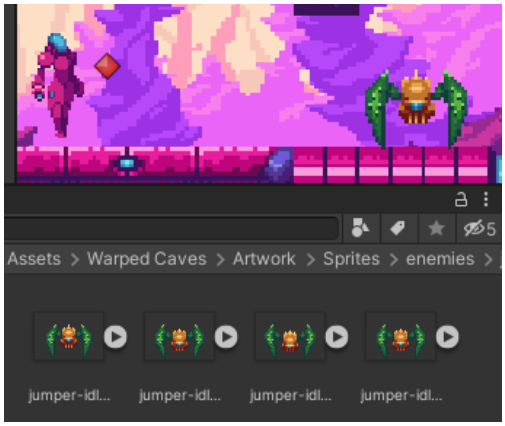
25. Coba jalankan dengan menekan *play*.



Gambar 10.23 Menjalankan *Game*

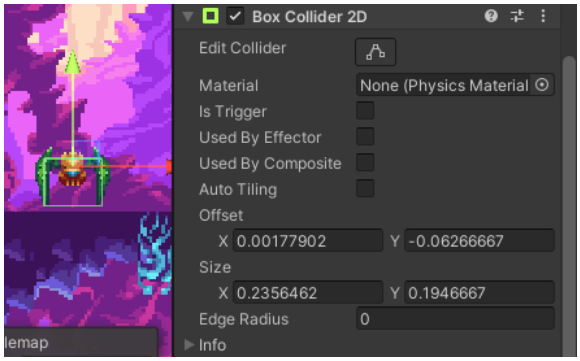
10.2 Tugas 2 : Langkah-langkah Membuat Animasi Enemy Dengan Implementasi 2 Enemy Behaviour dan 2 Enemy AI

1. Cari folder yang bernama *jumper-idle*. Lalu masukkan ke hirarki. Perbesar ukurannya jika terasa kecil.



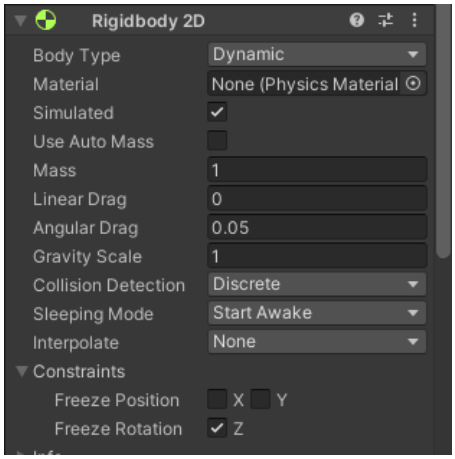
Gambar 10.24 Menambahkan *Enemy Jumper*

2. Tambahkan *Collider* 2D dan sesuaikan dengan *enemy jumper*.



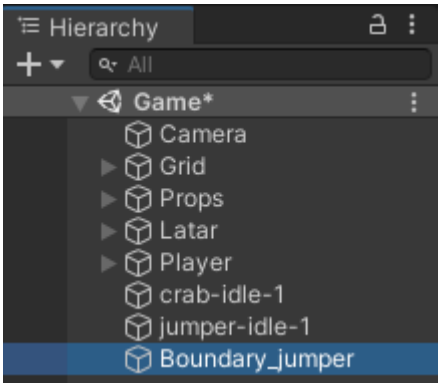
Gambar 10.25 Menambahkan Komponen *Collider* 2D

3. Tambahkan komponen *Rigidbody* 2D dan atur seperti dibawah ini.



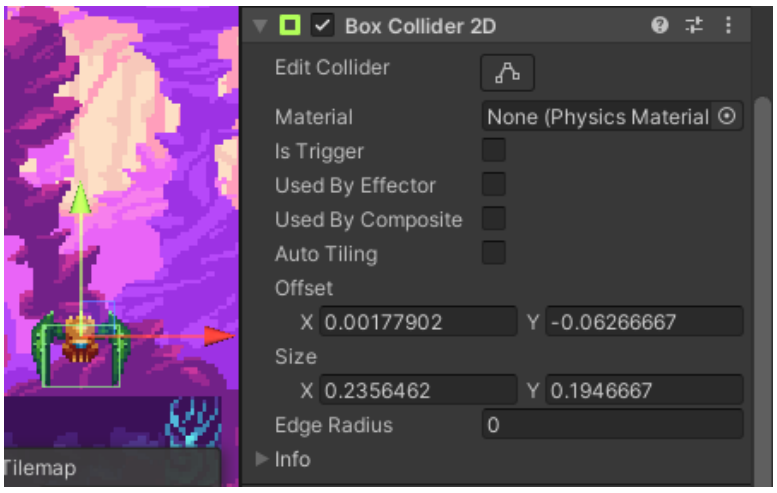
Gambar 10.26 Menambahkan Komponen *Rigidbody* 2D

4. *Create Empty object* pada *Hierarchy* ubah namanya menjadi *Boundary_jumper*.



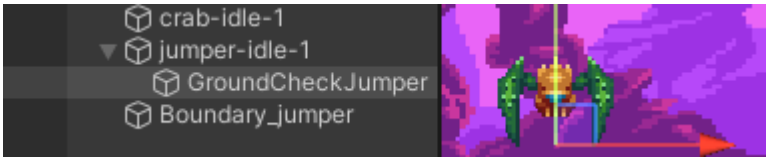
Gambar 10.27 Menambahkan *Boundary_jumper*

5. Tambahkan *Box Collider* 2d pada *Boundary*.



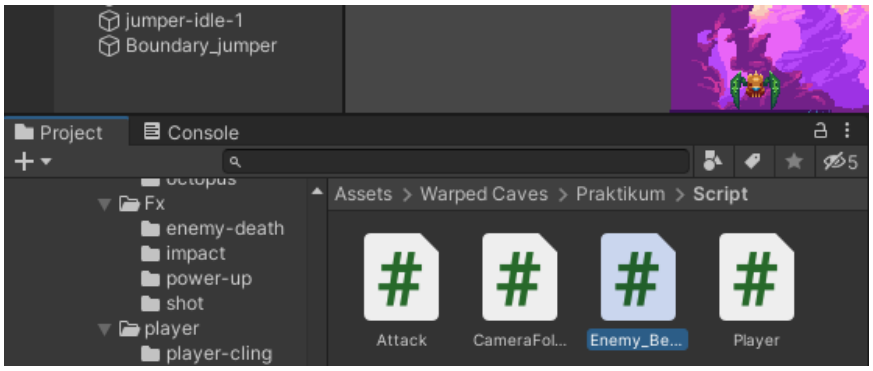
Gambar 10.28 Mengatur *Box Collider* 2D

6. Buat *Create Empty* didalamnya *jumper-idle*-1 lalu ubah namanya menjadi *GroundCheckJumper*. Setelah itu letakkan dibawah seperti gambar berikut.



Gambar 10.29 Membuat *GroundCheckJumper*

7. Buat *script* dengan nama *Enemy_Behavior_Jumper* dan masukkan ke *enemy jumper_idle_1*.



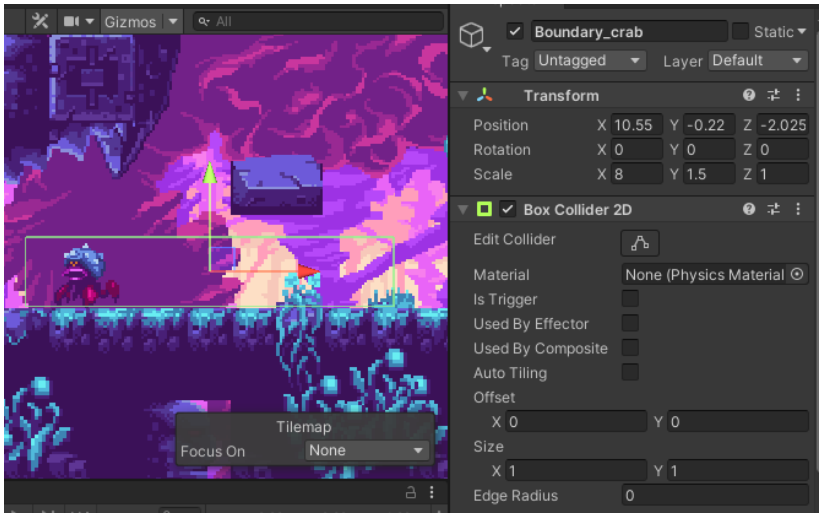
Gambar 10.30 Memberi *Script* Pada *jumper_idle_1*

8. Tambahkan *script* dibawah ini agar *enemy jumper* dapat melompat.

```
using System.Collections;
using UnityEngine;

public class Enemy_Behavior_Jumper : MonoBehaviour{
    [SerializeField] float jumpForce = 5f;
    [SerializeField] float jumpInterval = 2f;
    [SerializeField] float moveDistance = 1f;
    Rigidbody2D rb;

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

    void Update(){
    }

    IEnumerator JumpRoutine(){
        while (true){
            yield return new
WaitForSeconds(jumpInterval);
            float moveDirection = Random.Range(-1f, 1f);
            Vector2 jumpDirection = new
Vector2(moveDirection * moveDistance, jumpForce);
            rb.AddForce(jumpDirection,
ForceMode2D.Impulse);
```

```
        }
    }
}
```

9. Setelah itu buat lagi *Boundary_crab* dan berikan komponen *Box Collider* 2D dan *Rigidbody* 2D.

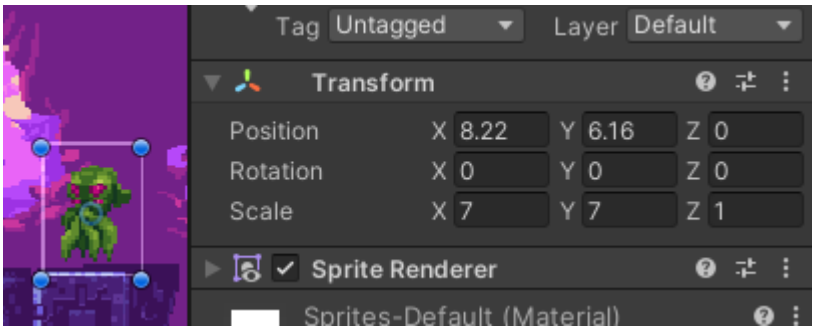


Gambar 10.31 Mengatur *Boundary_crab*

10. Buat *script* baru dan beri nama *Enemy_Behavior_Crab*. Lalu tambahkan *source code* dibawah ini. *Drag and drop script* tersebut ke *crab-idle*-1.

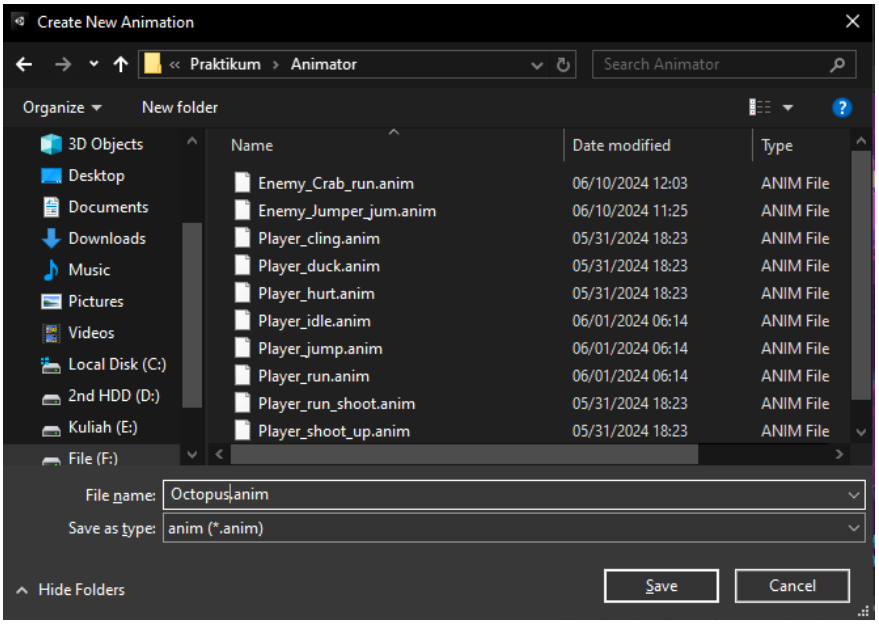
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Enemy_Behavior_Crab : 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);
    }
}
```

11. Cari folder yang bernama *octopus*. Lalu *drag and drop* kedalam hirarki. Atur ukurannya.



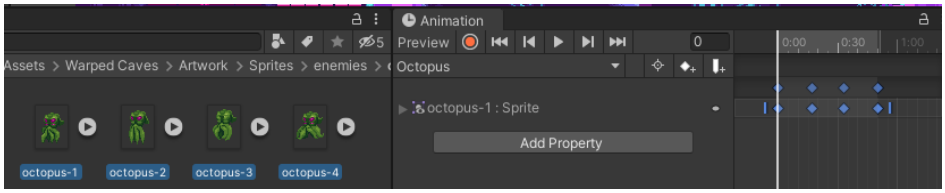
Gambar 10.32 Menambahkan *Enemy Octopus*

12. Pada panel *Animation* klik *create* lalu beri nama Octopus dan simpan kedalam folder *Animator*.



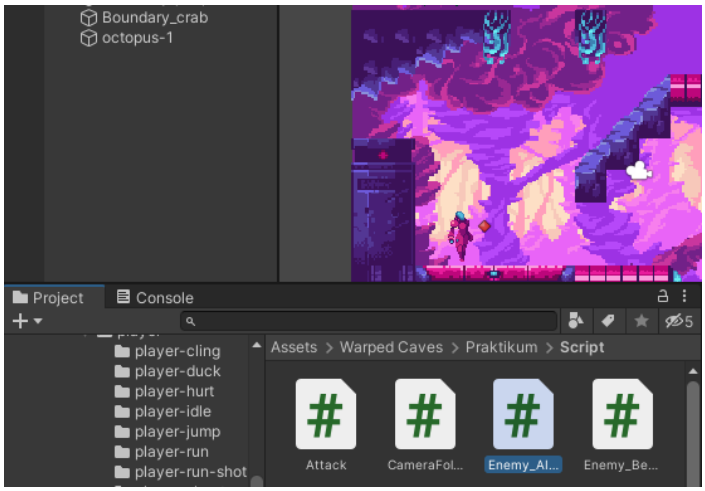
Gambar 10.33 Membuat Animasi *Enemy Octopus*

13. Lalu *drag and drop* semua *Octopus* kedalam *timeline*. Lalu atur menjadi 0.50.



Gambar 10.34 Menambahkan Animasi *Octopus*

14. Buat *script* baru dan beri nama *Enemy_AI_Octopus* didalam folder *script*. Lalu *drag and drop script* ke *octopus*-1.



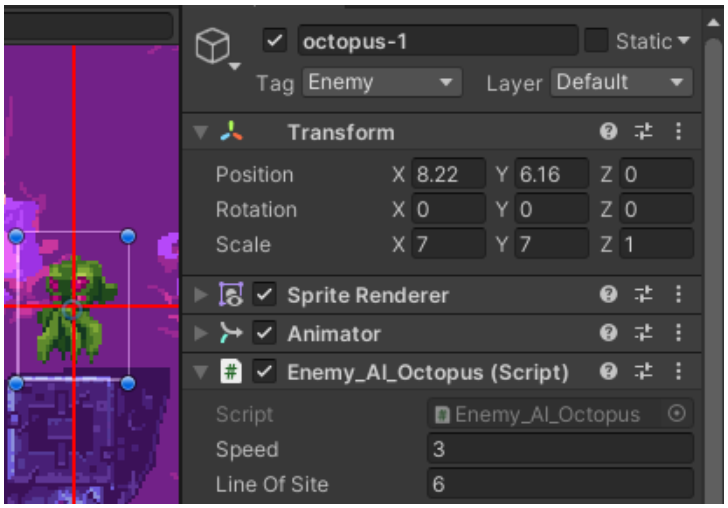
Gambar 10.35 Membuat *Script* AI Octopus

15. Tambahkan *source code* berikut didalam *script Enemy_AI_Octopus*

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

public class Enemy_AI_Octopus : MonoBehaviour{
    public float speed;
    public float lineOfSite;
    private Transform player;
    private Vector2 initialPosition;

    void Start(){
        player                                          =
GameObject.FindGameObjectWithTag("Player").transform;
        initialPosition                                 =
GetComponent<Transform>().position;
    }

    void Update(){
        float             distanceToPlayer              =
Vector2.Distance(player.position, transform.position);
        Vector2 targetPosition;

        if (distanceToPlayer < lineOfSite){
            targetPosition = player.position;
        }else{
            targetPosition = initialPosition;
        }

        transform.position                              =
Vector2.MoveTowards(this.transform.position,
targetPosition, speed * Time.deltaTime);

        if (targetPosition.x > transform.position.x){
            transform.localScale = new Vector3(-7, 7, 1);
```

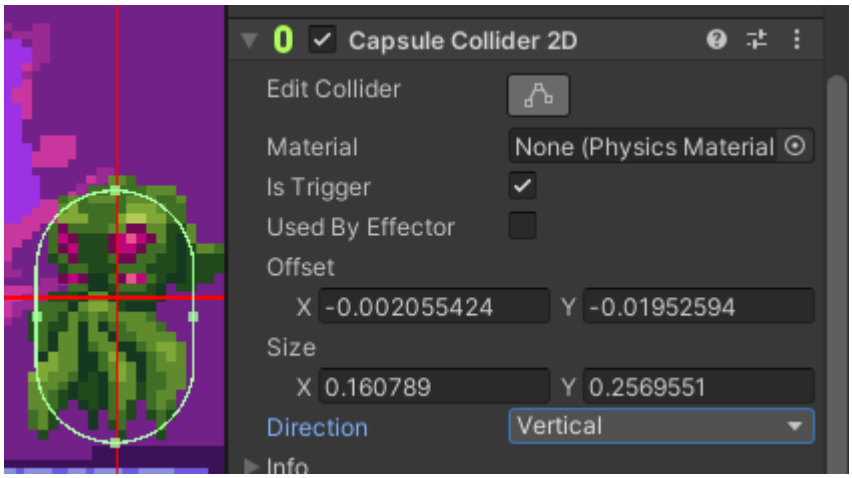
```
        }else        if        (targetPosition.x        <
transform.position.x){
            transform.localScale = new Vector3(7, 7, 1);
        }
    }

    private void OnDrawGizmosSelected(){
        Gizmos.color = Color.red;
        Gizmos.DrawWireSphere(transform.position,
lineOfSite);
    }
}
```

16. Pada Inspector *Enemy_AI_Octopus*, Atur *Speed* juga *Line of Site* untuk menentukan jarak dan *speed* pada *enemy*. Atur juga *tag* menjadi *enemy*.



Gambar 10.36 Mengatur *Speed* dan *Line Of Site*

17. Tambahkan komponen *capsule collider* 2d.



Gambar 10.37 Menambahkan Komponen *Capsule Collider* 2D

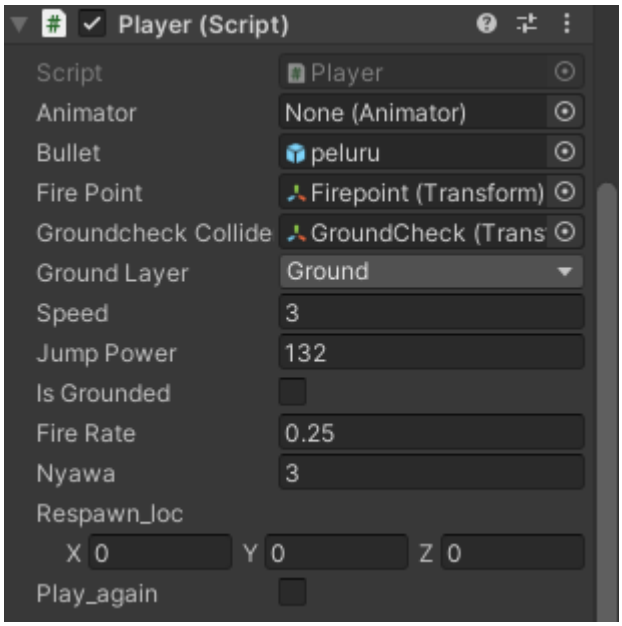
18. Lalu buka *script player* dan tambahkan berikut

```
// menambahkan variabel
public int nyawa;
[SerializeField] Vector3 respawn_loc;
public bool play_again;

// menambahkan posisi respawn didalam void Awake()
respawn_loc = transform.position;

//menambahkan respawn di void Update()
if(nyawa < 0){
    playagain();
}
if (transform.position.y < -10){
  play_again = true;
  playagain();
}

//menambahkan fungsi playagain()
void playagain(){
  if(play_again == true){
    nyawa = 3;
    transform.position = respawn_loc;
    play_again = false;
  }
}
```

19. Buat *script* baru dengan nama *Enemy_Attacked* lalu isikan *source code* dibawah ini.

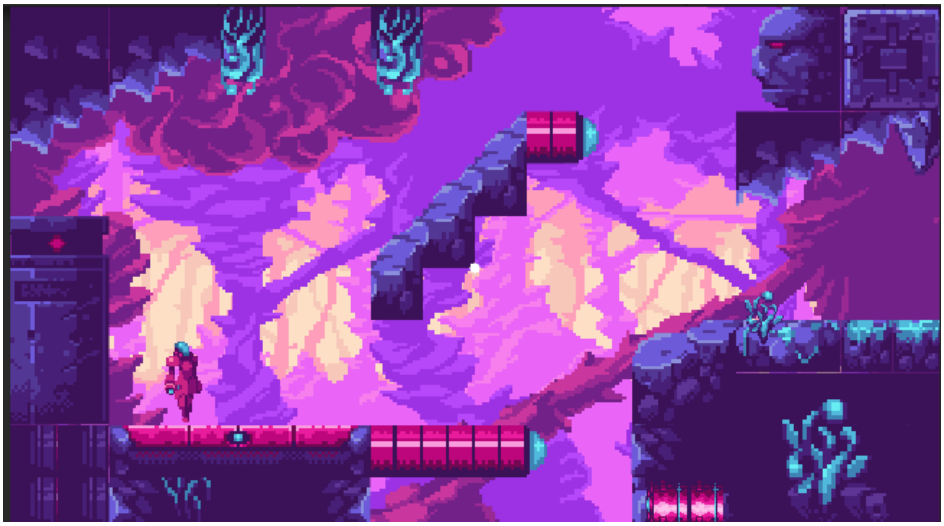
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Enemy_Attacked : MonoBehaviour{
    [SerializeField] private Player Object;
    void Start(){
        if (Object == null)
        {
            Object                                          =
GameObject.FindWithTag("Player").GetComponent<Player>();
        }
    }
    void OnTriggerEnter2D(Collider2D other){
        if (other.CompareTag("Player")){
            Object.nyawa--;
            if (Object.nyawa < 0){
                Object.play_again = true;
            }
        }
    }
}
```

20. Tambahkan *script Enemy_Attacked* ke *Octopus*. Lalu klik *Player* dan ubah nyawa pada bagian *script* menjadi 3.



Gambar 10.38 Mengubah Nyawa Menjadi 3

21. Coba jalankan.



Gambar 10.39 Menjalankan Game

10.3 Kuis

```
using UnityEngine;

public class PlayerAttack : MonoBehaviour{
    public int atackRange = 2.0f;
    public int attacDamage = 10;

    void Update(){
        if (InputGetButtonDown("Fire1")){
```

```
            PerformMeleeAttack();
        }
    }

    void PerformMeleeAttack(){
        RaycastHit hit;
        if (Physics.Raycast(transform.position,
transform.forward, out hit, attackRange)){
            EnemyHealth enemyHealth =
hit.transform.GetComponent<EnemyHealth>();
            if (enemyHealth != null){
                enemyHealth.TakeDamage(attackDamage);
            }
        }
    }
}
```

Penjelasan:

Pada *source code* diatas pada fungsi *PerformMeleeAttack* ini menggunakan deklarasi *Raycast* untuk mendeteksi serangan dari musuh. Lalu jika *Raycast* terkena serangan maka akan memeriksa *EnemyHealth* lalu jika terdeteksi bahwa *enemyHealt* tidak sama dengan *null* maka akan memberikan *demage* dengan memanggil *TakeDamage* dari komponen *EnemyHealth*. Untuk bagian *EnemyHealth* dapat dilihat pada *source code* dibawah ini.

```
using UnityEngine;

public class EnemyHealth : MonoBehaviour{
    public int maxHealth = 100;
    private int currentHealth;
    void Start(){
        currentHealth = maxHealth;
    }
    public void TakeDamage(int damage){
        currentHealth -= damage;
        if (currentHealth <= 0){
            Die();
        }
    }
    void Die(){
        Destroy(gameObject);
    }
}
```

Penjelasan:

Terdapat *source code* untuk *EnemyHealth* yang terdapat *MonoBehaviour* untuk penerapan di *unity*. Lalu dibuat variabel baru untuk nyawa maksimal yaitu 100 dengan tipe data *integer*. Lalu dibuat fungsi *Start*

untuk mengatur *currentHealth* agar sama dengan *maxHealth*. Lalu terdapat fungsi *TakeDamage* dengan parameter *damage* dengan tipe data *integer*. Pada fungsi *TakeDamage* ini digunakan untuk mengurangi nyawa musuh ketika menerima *damage*. Lalu di cek apakah *currentHealth* <= 0 maka *enemy* akan mati dengan memanggil fungsi. Pada fungsi *Die* ini digunakan untuk menghapus atau menghancurkan objek atau musuh dari *game* nya.