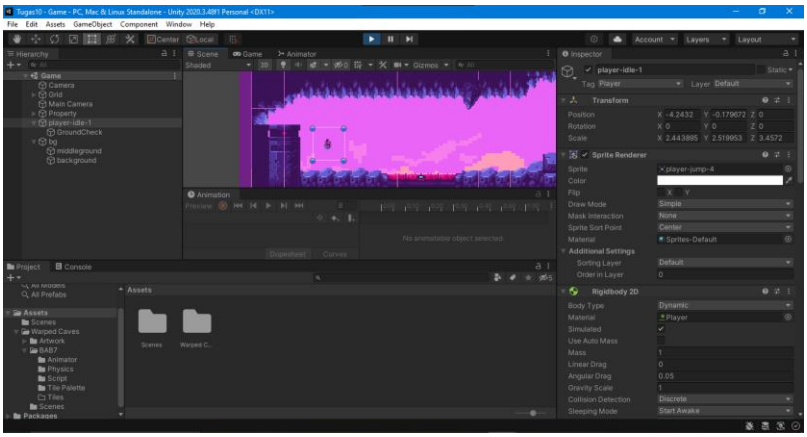
TUGAS PERTEMUAN: 10

Respawn and AI Enemy Attack

NIM	:	2118089
Nama	:	Shiva Divanti Natasya
Kelas	:	B
Asisten Lab	:	Nur Aria Hibnastiar (2118078)

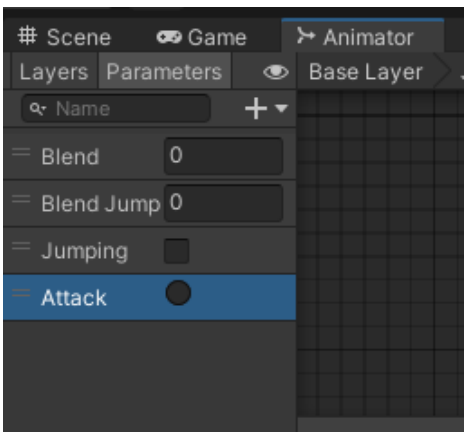
8.1 Tugas 1 : Membuat AI Enemy Attack dan Respawn Serta Kuis

1. Buka *Project* Bab 9 terlebih dahulu.



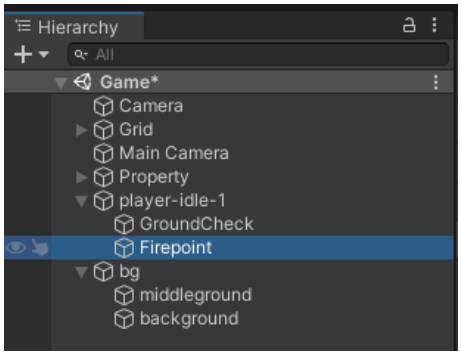
Gambar 10.1 Tampilan Bab 9

2. Kemudian pada *menu Tab Animator* Tambahkan Parameter *Trigger*, *Rename* Menjadi *Attack*.



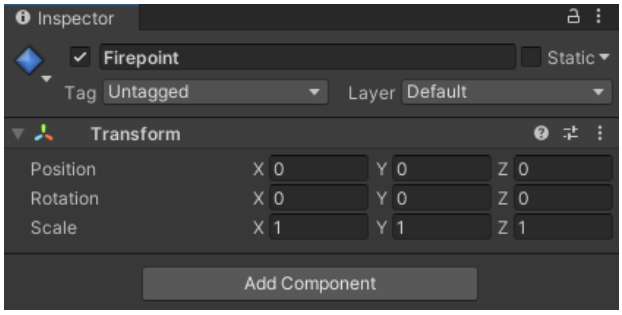
Gambar 10.2 Tampilan Add Parameter

3. Setelah menambahkan parameter *Attack*, Langkah selanjutnya adalah membuat Layer *Game object* baru didalam *player-idle-1*, Klik kanan pilih *Create Empty* lalu Rename menjadi *Firepoint*.



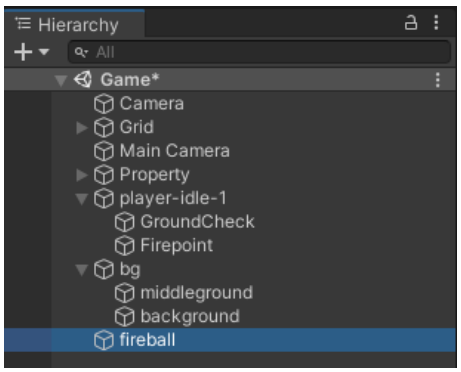
Gambar 10.3 Tampilan Add Firepoint

4. Pada menu Hierarchy Tambahkan item-feedback-1, di folder Sprites > Fx > item-feedback-1 , *rename* menjadi *fireball*.



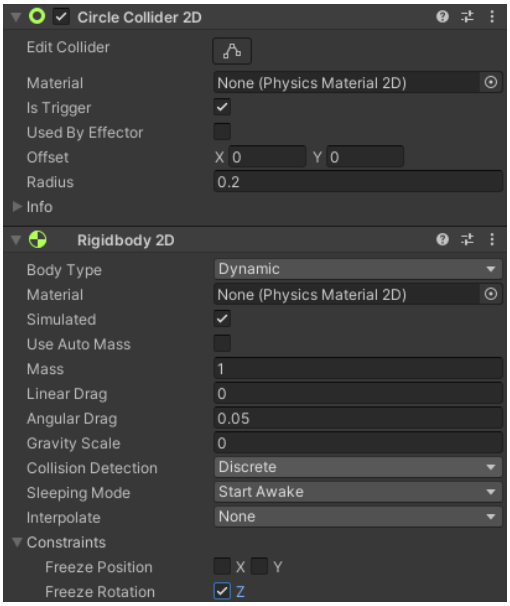
Gambar 9.4 Tampilan Mengatur Icon

5. Pada *menu Hierarchy* Tambahkan *item-feedback*-1, di folder Sprites > Fx > *item-feedback*-1 , *rename* menjadi *fireball*.



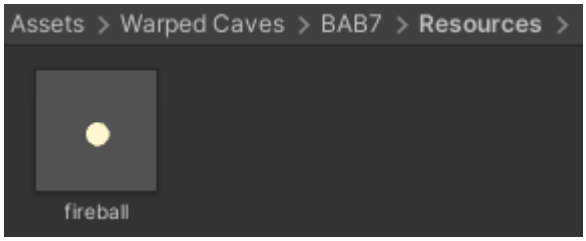
Gambar 10.5 Menambahkan Item Feedback

6. Klik *item-feedback*-1 untuk menambahkan *Component Circle Collider* 2d, dan *Riggidbody* 2D, *Setting* sesuai gambar dibawah ini.



Gambar 10.6 Add Circle Collider Dan Rigidbody

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



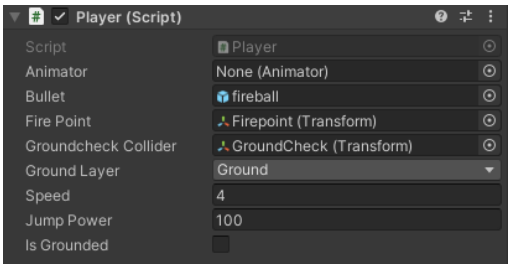
Gambar 10.7 Buat Folder Resource

8. Pada *Script Player* Tambahkan *Script* dibawah ini.

```
#Pada class Player
// Deklarasi variable
Public Animator animator;
Public GameObject bullet;
Public Transform Firepoint;

#Tambahkan dibawah fungsi fixedUpdate
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);
```

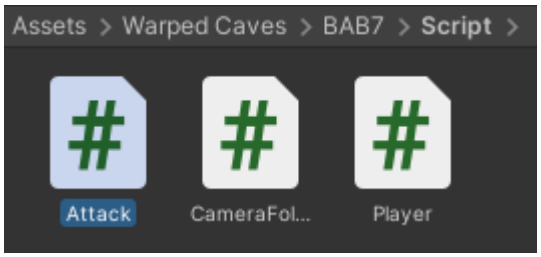
```
     Destroy(fireball, 2f);
}
#Tambakan pada Function Void Update

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 10.9 Add Bullet Objek

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

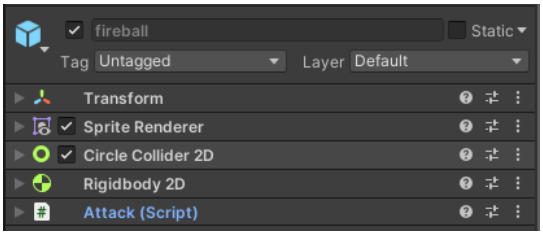


Gambar 10.10 Membuat Script Attack

11. Tambahkan *Script Attack* dibawah ini.

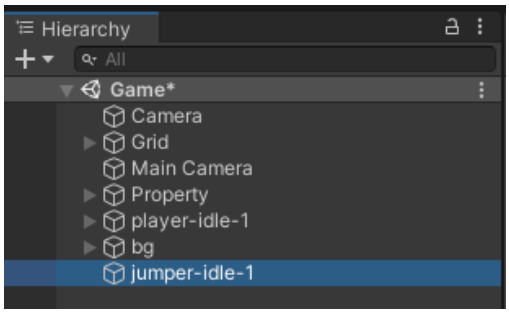
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

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*.



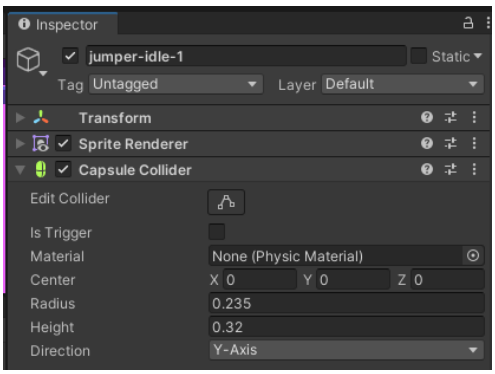
Gambar 10.12 Add Script Attack

13. Tambahkan *Enemy jumper-idle-1* pada *hierarchy* di folder *Sprites*, *enemies->jumper-idle-1*.



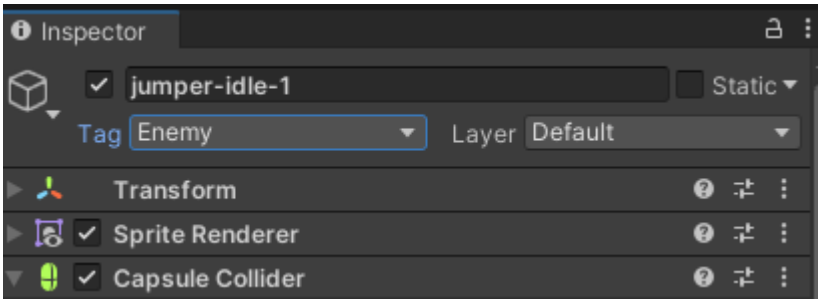
Gambar 10.13 Add Enemy *jumper-idle-1*

14. Kemudian klik pada *jumper-idle-1*, lalu pada *menu tab inspector* tambahkan *collider* 2D untuk mendeteksinya.



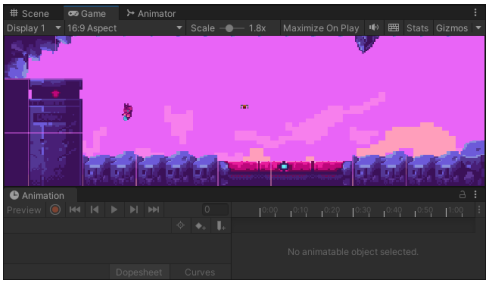
Gambar 10.14 Add Capsule *collider*

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



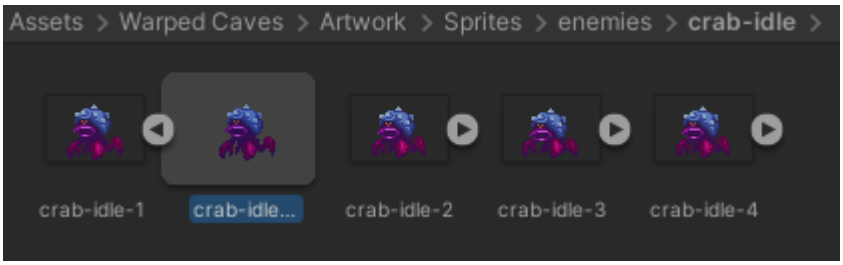
Gambar 10.15 Add Tag Enemy

16. Tembak *Enemy* dengan menekan Tombol C untuk menghancurkan musuh.



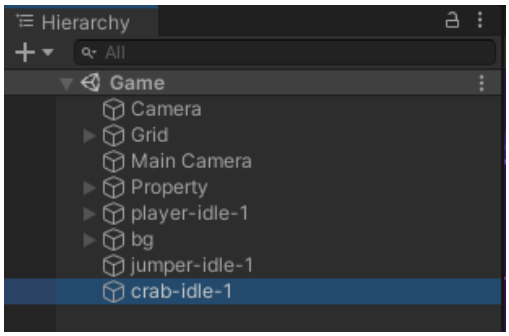
Gambar 10.16 Tampilan Menembak Lawan

17. Cari sebuah *sprite pack* bernama *enemy* dan buka folder bernama "crab-idle".



Gambar 10.17 Tampilan crab-idle

18. Tambahkan "crab-idle-1" ke *Hierarchy*.

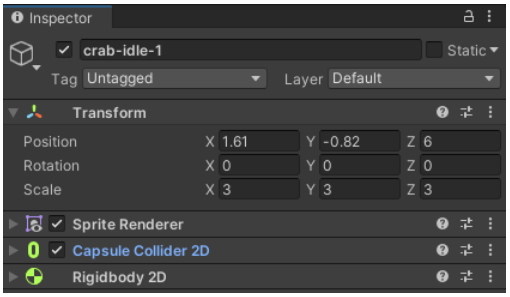


Gambar 10.18 Add crab-idle

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

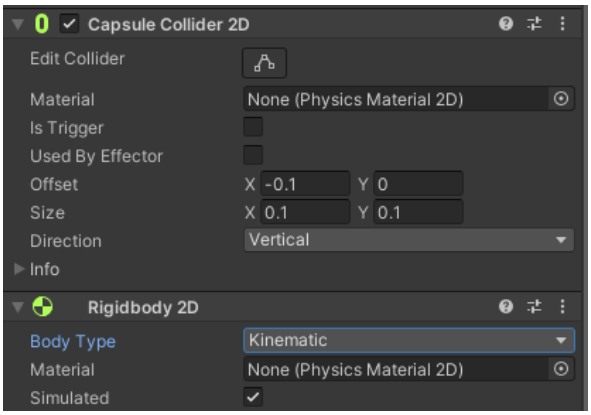
Gambar 10.19 Setting Transform Scale

20. Tambahkan sebuah komponen bernama *Capsule Colider* 2D dan *Rigidbody* dalam *inspector game* objek crab-idle-1.



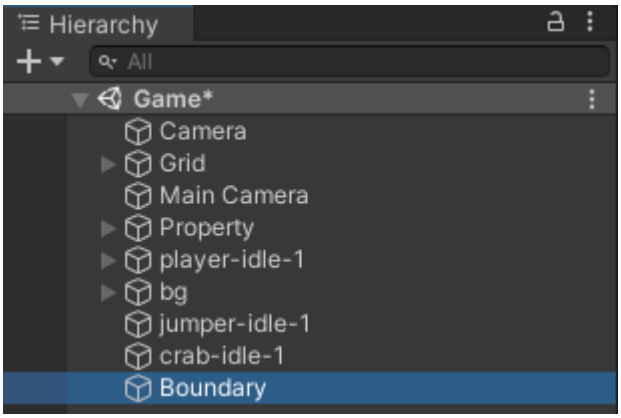
Gambar 10.20 Add Component

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



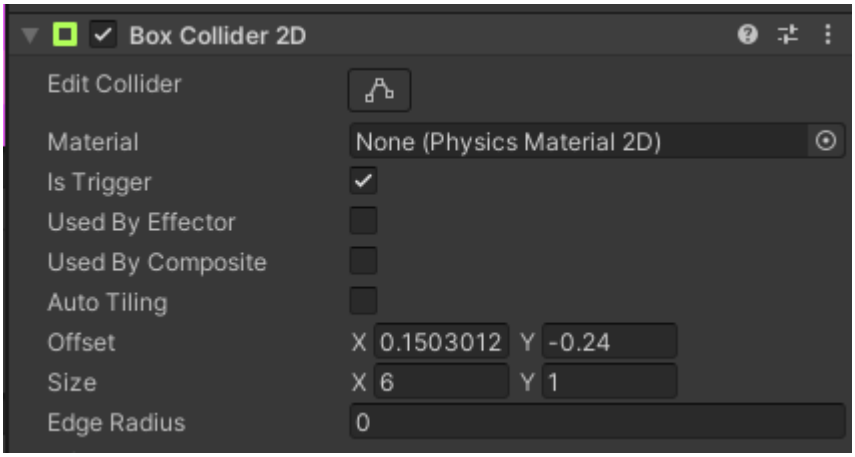
Gambar 10.21 Menyeting Komponen

22. *Create Empty object* pada *Hierarchy*, *Rename* menjadi *Boundary*.



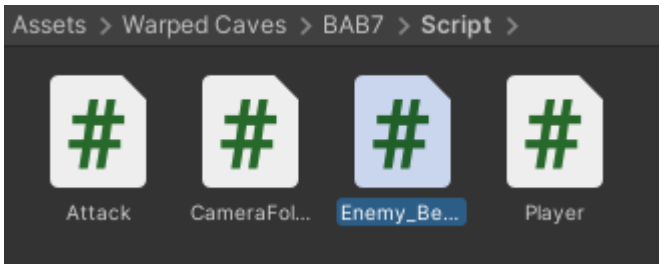
Gambar 10.22 Create Empty Object

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



Gambar 10.23 Add Box Collider

24. Buat sebuah *file script* didalam folder *Script* beri nama "Enemy_Behavior", kemudian *drag* dan masukkan ke dalam game *object* "crab-idle-1".



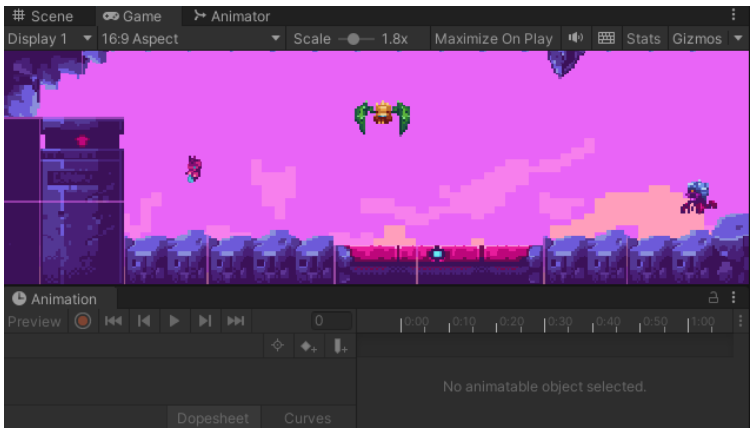
Gambar 10.24 Membuat Script

25. 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);
    }
    }
```

26. Jalankan Program.



Gambar 10.25 Hasil Tampilan Lawan

27. Buat *Script Enemy_AI* pada folder BAB7 – Script.

Gambar 10.26 Membuat Script AI

28. Tambahkan *Script* dibawah ini.

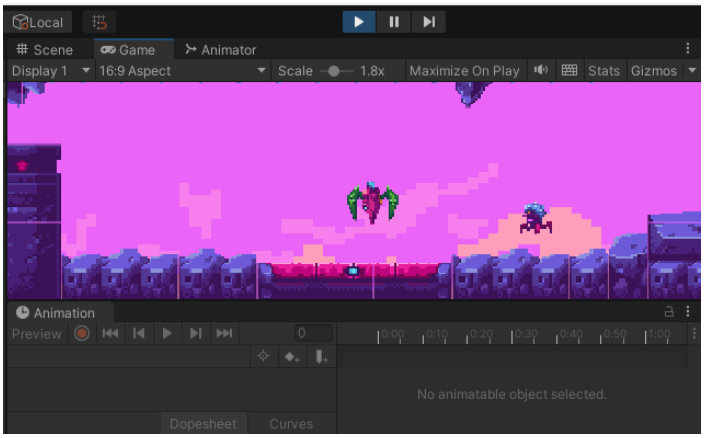
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Enemy_AI : MonoBehaviour
{
```

```
    public float speed; // Kecepatan gerakan musuh
    public float lineOfSite; // Jarak penglihatan musuh
    private Transform player; // Transform dari pemain
    private Vector2 initialPosition; // Posisi awal musuh

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

    // Update is called once per frame
    void Update()
    {
        // Menghitung jarak antara musuh dan pemain
        float              distanceToPlayer             =
Vector2.Distance(player.position, transform.position);

        // Jika pemain berada dalam jarak penglihatan musuh
        if (distanceToPlayer < lineOfSite)
        {
            // Musuh bergerak menuju pemain
            transform.position                          =
Vector2.MoveTowards(this.transform.position,
player.position, speed * Time.deltaTime);
        }
        else
        {
            // Musuh kembali ke posisi awal
            transform.position                          =
Vector2.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);
    }
}
```

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

Gambar 10.27 Mengatur Speed Line

30. Running Game, maka *eagle* akan mengikuti Gerakan Player.



Gambar 10.30 Hasil Running Game

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

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

32. Tambahkan kode dibawah ini untuk mengatur posisi *respawn* sesuai dengan posisi awal permainan dimulai.

```
respawn_loc = transform.position;
```

33. Tambahkan kode dibawah ini di dalam *void update* Player.cs agar ketika nyawa *player* dibawah 0 maka akan melakukan *respawn*.

```
  if (nyawa < 0)
          {
              playagain();
          }
```

34. Tambahkan juga kode berikut dibawah *code* sebelumnya agar ketika *pliiayer* jatuh dibawah *platform* akan melakukan *respawn*.

```
if (transform.position.y < -10)
        {
            play_again = true;
            playagain();
        }
```

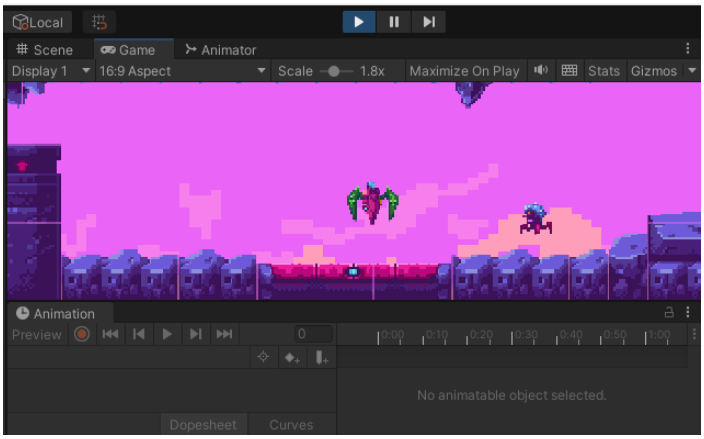
35. Tambahkan fungsi *playagain*() dalam *script* Player.cs.

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

36. Pada *hierarchy* crab-idle-1 Tambahkan *Script enemy attack*, arahkan *object* pada *player-idle-1*.

```
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;
            }
        }
    }
}
```

37. Ubah nilai Nyawa menjadi 3 pada *Player*(*Script*). Jika di *play*, *Player* mengenai atau menyentuh *Enemy_idle1* sebanyak 3 kali maka nyawa akan berkurang 1 dan jika nyawa kurang dari 0 maka akan *reswpawn* ke titik awal.



Gambar 10.31 Hasil Akhir Running

A. Kuis

```
using UnityEngine;

public class PlayerAttack : MonoBehaviour
{
    public float attackRange = 2.0f; // Perbaikan: tipe
data harus float untuk jarak serangan
    public int attackDamage = 10;    // Perbaikan: salah
ketik "attacDamage" menjadi "attackDamage"

    void Update()
    {
        if (Input.GetButtonDown("Fire1")) // Perbaikan:
"InputGetButtonDown" menjadi "Input.GetButtonDown"
        {
            PerformMeleeAttack();
        }
    }

    void PerformMeleeAttack()
    {
        RaycastHit hit;
        if (Physics.Raycast(transform.position,
transform.forward, out hit, attackRange))
        {
            // Menambahkan kode untuk mengenai musuh dan
mengurangi health mereka
            EnemyHealth enemyHealth =
hit.collider.GetComponent<EnemyHealth>();
            if (enemyHealth != null)
            {
                enemyHealth.TakeDamage(attackDamage);
            }
        }
    }
}
```

Analisa :

Pertama, tipe data untuk `attackRange` diubah dari `int` menjadi `float` agar dapat menangani nilai desimal, yang sesuai untuk mengukur jarak serangan. Kemudian, kesalahan pengetikan pada variabel `attacDamage` diperbaiki menjadi `attackDamage` untuk konsistensi penamaan dan menghindari error. Selain itu, fungsi `InputGetButtonDown` yang salah ketik diperbaiki menjadi `Input.GetButtonDown`, yang merupakan metode yang benar dari API Unity untuk mendeteksi input tombol. Terakhir, ditambahkan logika dalam metode `PerformMeleeAttack` untuk memeriksa apakah `Raycast` mengenai musuh dengan komponen `EnemyHealth`, dan jika iya, maka health musuh akan dikurangi menggunakan metode `TakeDamage`.

A. Link Github

https://github.com/ShivaDina/2118089_PRAK_ANIGAME