# TUGAS PERTEMUAN: 10

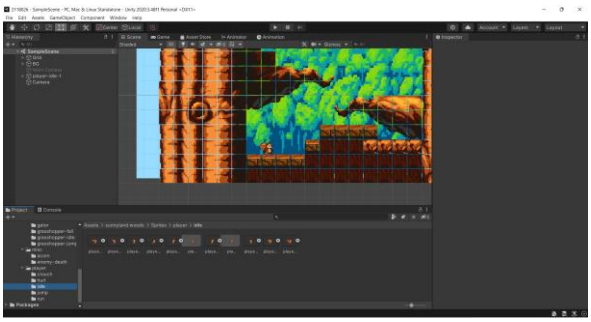
# RESPAWN AND ENEMY ATTACK

| | | |
|---|---|---|
| **NIM** | : | 2118026 |
| **Nama** | : | Bryan Ifan Etikamena |
| **Kelas** | : | B |
| **Asisten Lab** | : | Aprillia Dwi Dyah S (2118143) |

## 10.1 Tugas 10 : Membuat Respawn And Enemy Attack

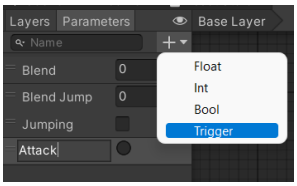
### A. Langkah-langkah Membuat Mekanisme Attack

1. Buka *file* projek *Unity* sebelumnya pada bab 9 untuk digunakan kembali.



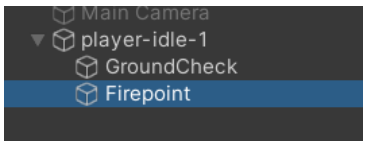
Gambar 10.1 Membuka *Project Unity*

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



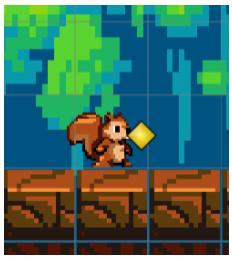

Gambar 10.2 Menambahkan Parameter Baru

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.*



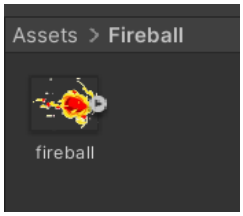

Gambar 10.3 Membuat *Game Object* Baru

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



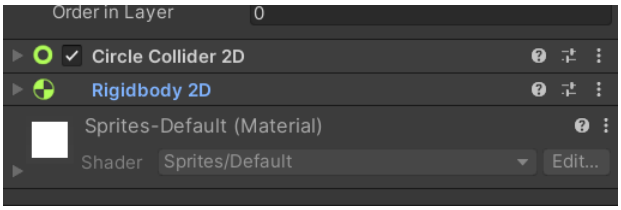
Gambar 10.4 Mengubah *Icon*

5. Pada menu *Hierarchy* Tambahkan *fireball*, di *folder Asset*.



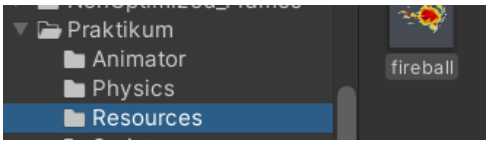

Gambar 10.5 Menambahkan *Fireball*

6. Klik *fireball* untuk menambahkan *Component Circle Collider* 2d, dan *Riggidbody* 2D, *Setting* sesuai gambar di bawah ini.




Gambar 10.6 Menambahkan *Component*

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



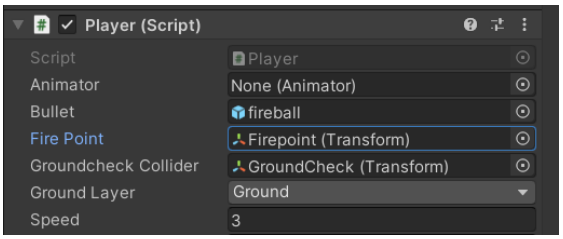

Gambar 10.7 Membuat *Folder* Baru

8. Pada *Script Player* Tambahkan *Script* di bawah ini

```
#Pada class Player
// Deklarasi Variabel
public Animator animator;
public GameObject bullet;
public Transform firepoint;

#Tambahkan dibawah fungsi fixedUpdate
IEnumerator Attack()
```

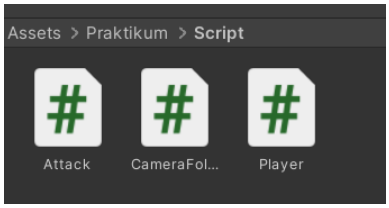
```
{
      animator.SetTrigger("Attack");
      yield return new WaitForSeconds(0.25f);

      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 di bawah ini, Dimana *Bullet* berisi *object* yang akan ditembak sedangkan *fire point* adalah titik tembak pertama.




Gambar 10.8 Mengatur *Inspector Player*

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



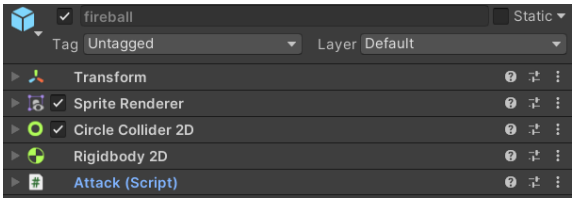

Gambar 10.9 Membuat *Script Attack*

11. Tambahkan *Script Attack* di bawah ini

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

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

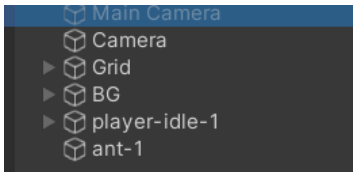
```
                Destroy(gameObject);
                Destroy(collision.gameObject);
            }
        }
}
```

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



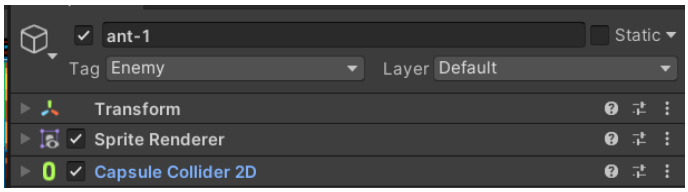

Gambar 10.10 Menambahkan *Script Attack*

13. Tambahkan *Enemy ant-1* pada *hierarchy* di *folder Sprites, ant-1*



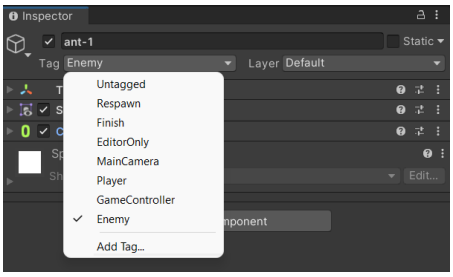

Gambar 10.11 Menambahkan *Enemy*

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



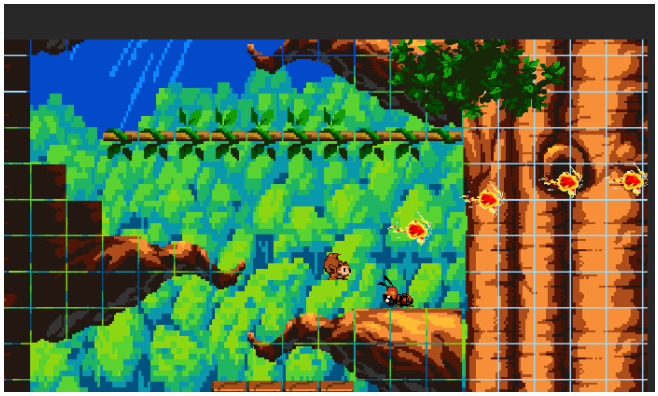

Gambar 10.12 Menambahkan *collider 2D*

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




Gambar 10.13 Menambahkan *Tag Enemy*

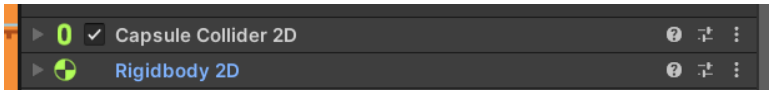
16. *Play* dan coba untuk menembak dengan menekan Tombol C.



Gambar 10.14 Mencoba Menembak

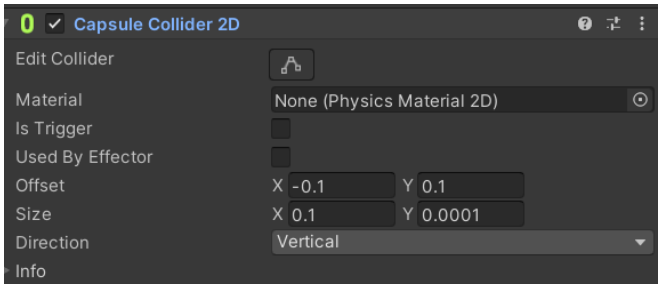
## B. Langkah-langkah Membuat Enemy Behavior NPC

1. Klik enemy ant yang sudah dibuat sebelumnya. Pada *inspector* tambahkan sebuah komponen bernama *Capsule Colider* 2D dan *Rigidbody*




Gambar 10.15 Menambahkankan *Component*

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

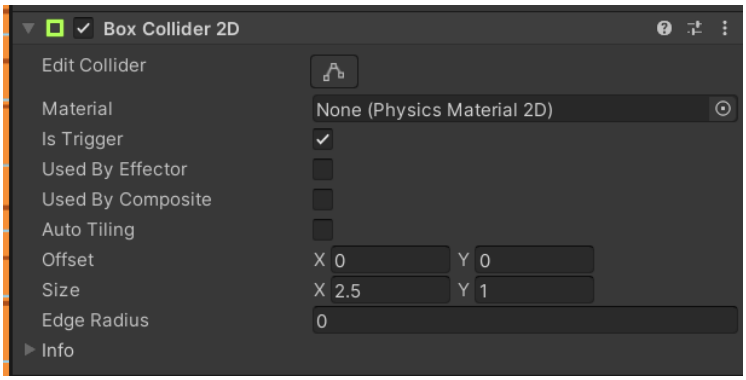



Gambar 10.16 Mengatur *Collider*

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

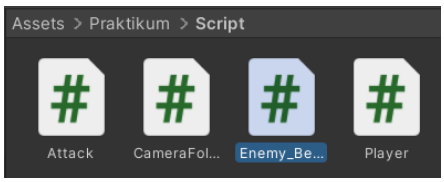

Gambar 10.17 Membuat *Empty Object*

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




Gambar 10.18 Menambahkan *Box Collider* 2D

5. Buat sebuah *file script* di dalam *folder Script* beri nama "Enemy_Behavior", kemudian *drag* dan masukkan ke dalam *game object* "ant-1"



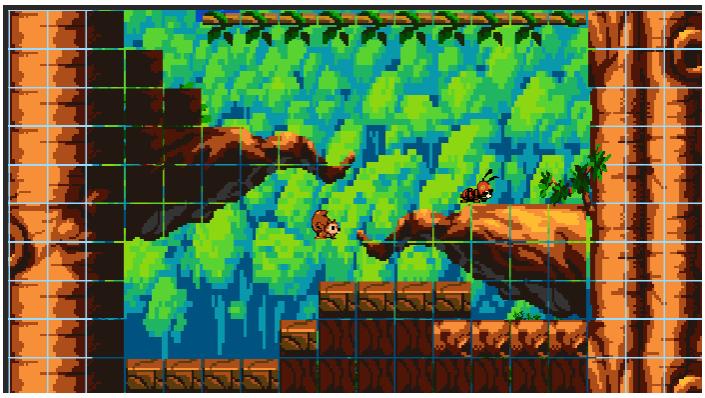

Gambar 10.19 Membuat *File Script Baru*

6. Tambahkan *Script* di bawah 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);
        }
    }
```

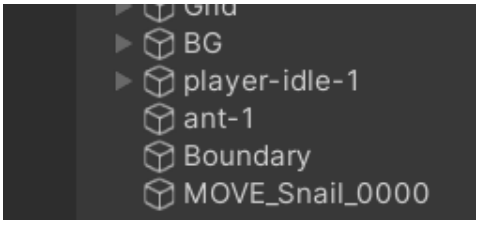
```
    private bool isFacingRight()
    {
        return transform.localScale.x > Mathf.Epsilon;
    }

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

7. Jalankan Program untuk mengecek apakah *enemy* sudah sesuai atau belum.



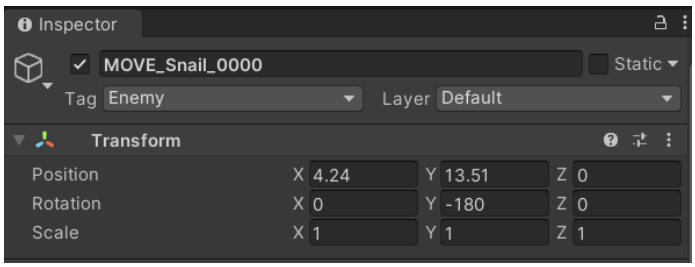
Gambar 10.20 Menjalankan Program

8. Tambahkan lagi *enemy* dengan cara *drag and drop MOVE_Snail_0000* ke *hirarchy*.



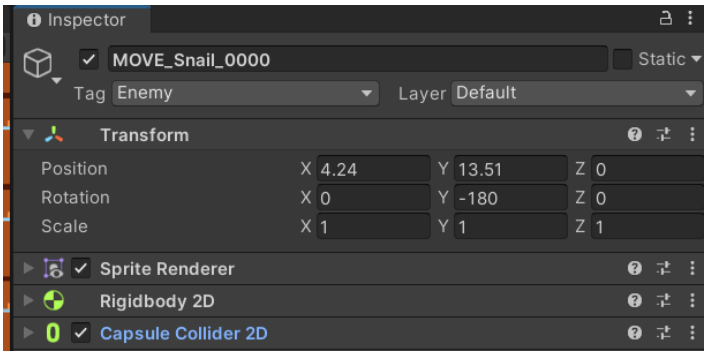

Gambar 10.21 Menambahkan *Enemy* Baru

9. Masuk ke bagian *inspector* lalu atur posisi dan ukuran objek tersebut.



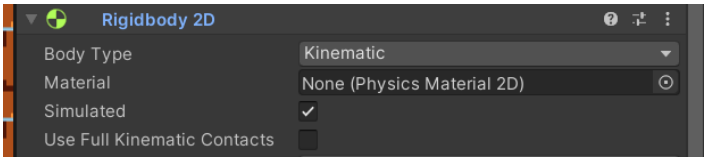

Gambar 10.22 Mengatur Ukuran Objek

10. Tambahkan sebuah komponen bernama *Capsule Colider* 2D dan *Rigidbody* dalam *inspector game* objek *MOVE_Snail_0000* ke.



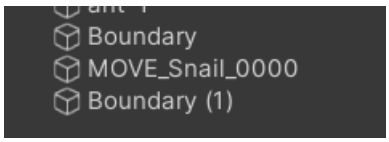

Gambar 10.23 Menambahkan *Component*

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



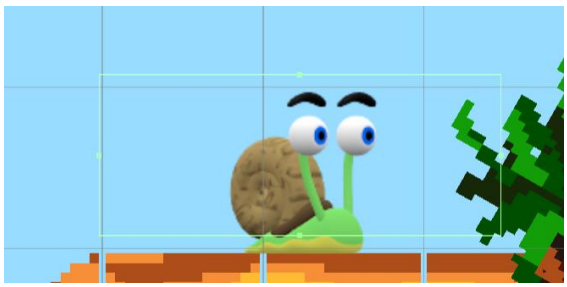

Gambar 10.24 Mengatur *Component*

12. *Duplicate object boundery* yang sudah dibuat sebelumnya pada *hirarchy*.



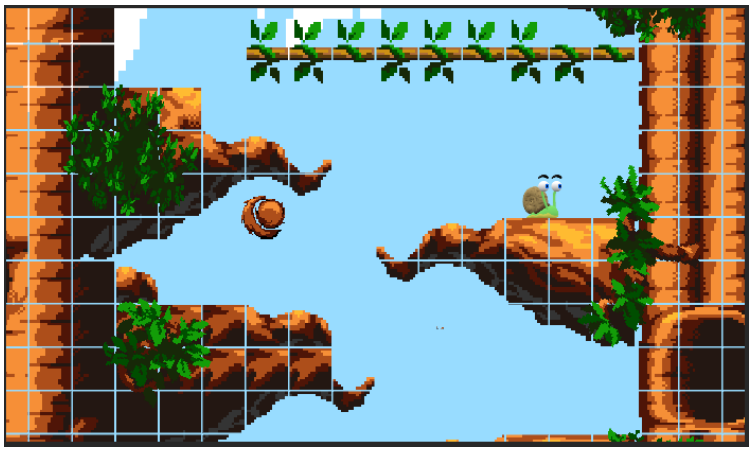

Gambar 10.25 Menggandakan *Boundery*

13. Kemudian atur posisi collider *boundery* di objeknya .



Gambar 10.26 Mengatur *Boundery*

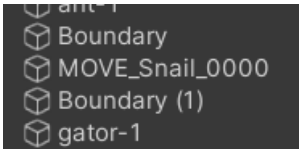
14. Jalankan Program untuk mengecek apakah enemy sudah sesuai atau belum.



Gambar 10.27 Menjalankan Program

## C. Langkah-langkah Membuat Enemy AI

1. Cari sebuah *sprite pack* bernama *'enemy'* dan buka *folder* bernama gator-1. Tambahkan gator-1 pada *Hierarchy*




Gambar 10.28 Menambahkan *Enemy*

2. Buat *Script Enemy_AI* pada *folder* Praktikum – *Script*


Gambar 10.29 Membuat *Script* Baru

3. 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);
    }
}
```

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


Gambar 10.30 Mengatur *Enemy_Ai*

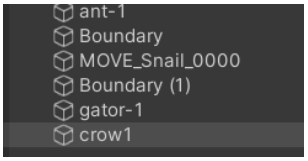
5. Atur posisi gator-1 tersebut

Gambar 10.31 Mengtur Posisi *Enemy*

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

Gambar 10.32 Menjalankan *Game*

7. Cari lagi sebuah *sprite pack* bernama *'Crow'* dan buka *folder* bernama *crow1*. Tambahkan *crow1* pada *Hierarchy*




Gambar 10.33 Menambahkan *Enemy*

8. Tambahkan *Script Enemy_AI* pada crow1 lalu Atur *Speed* juga *Line of Site* untuk menentukan jarak dan *speed* pada *enemy*


Gambar 10.34 Mengatur *Enemy_Ai*

9. Atur posisi gator-1 tersebut

Gambar 10.35 Mengtur Posisi *Enemy*

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

Gambar 10.36 Menjalankan *Game*

## D. Langkah-langkah Membuat Respawn

1. Buka *file script* (*Player.cs*) tambahkan variabel nyawa seperti di bawah ini

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

2. Tambahkan kode di bawah ini untuk mengatur posisi *respawn* sesuai dengan posisi awal permainan dimulai

```
private void Awake ()
{
    rb = GetComponent<Rigidbody2D>();
    animator = GetComponent<Animator>();

    respawn_loc = transform.position;
}
```

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

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

4. Tambahkan juga kode berikut di bawah *code* sebelumnya agar ketika *player* jatuh di bawah *platform* akan melakukan *respawn*

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

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

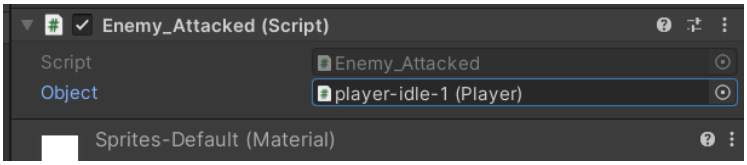
```
Void playagain()
{
   if (play_again == true)
   {
      nyawa = 3;
      transform.position = respawn_loc;
      play_again = false;
   }
}
```

6. Tambahkan *file script* (*Enemy_Attacked.cs*) dan isikan *source code* di bawah 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--;
```

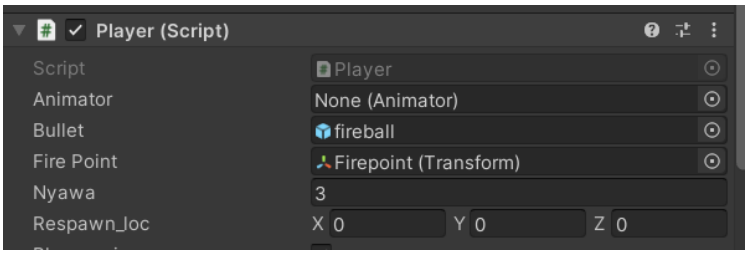
```
                if (Object.nyawa < 0)
                {
                    Object.play_again = true;
                }
            }
        }
}
```

7. Pada hierarchy setiap *enemy* Tambahkan *Script enemy attacked*, arahkan *object* pada *player-idle-1*



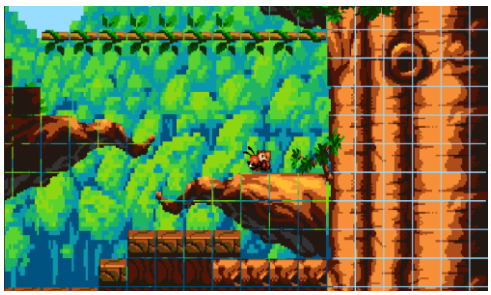

Gambar 10.37 Menambahkan *Script Enemy Attack*

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




Gambar 10.38 Mengubah Nilai Nyawa

9. Jika di *play*, *Player* mengenai atau menyentuh setiap *enemy* sebanyak 3 kali maka nyawa akan berkurang 1 dan jika nyawa kurang dari 0 maka akan *respawn* ke titik awal



Gambar 10.39 Menjalankan Program

## 10.2 Kuis Pertemuan 10

```
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))
        {
            // Lengkapi kode di sini untuk mengenai musuh dan
mengurangi health mereka
        }
    }
}
```

Analisis :

Berdasarkan source code di atas, maka ada beberapa yang harus diubah dan ditambahkan. Yang diubah misalnya tipe data atackRange yang sebelumnya dari int ke float. Dan juga tambahan untuk source code voidnya. Pada void PlayerAtttack, attackRange digunakan untuk menentukan jarak serangan jarak dekat. attackDamage digunakan untuk menentukan jumlah damage yang diberikan. Pada metode Update, dicek apakah tombol "Fire1" ditekan, dan jika ya, kita memanggil PerformMeleeAttack. PerformMeleeAttack menggunakan Physics.Raycast untuk mendeteksi objek di depan pemain dalam jarak serangan. Jika objek yang terkena memiliki komponen EnemyHealth, maka kita mengurangi health musuh tersebut dengan memanggil metode TakeDamage. Pada EnemyHealth, maxHealth digunakan untuk menentukan jumlah health maksimal musuh. currentHealth digunakan untuk menyimpan jumlah health saat ini. Pada metode Start, currentHealth diinisialisasi dengan nilai maxHealth. TakeDamage berguna untuk mengurangi currentHealth sebesar jumlah damage yang diterima, dan memeriksa apakah health telah

mencapai nol. Jika ya, metode Die dipanggil. Metode Die digunakan untuk memusnahkan objek musuh menggunakan Destroy(gameObject).

Source code untuk PlayerAttack

```
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))
        {
            // Memeriksa apakah objek yang terkena adalah
musuh
            EnemyHealth enemyHealth =
hit.collider.GetComponent<EnemyHealth>();
            if (enemyHealth != null)
            {
                // Mengurangi health musuh
                enemyHealth.TakeDamage(attackDamage);
            }
        }
    }
}
```

Source code untuk EnemyHealth

```
using UnityEngine;

public class EnemyHealth : MonoBehaviour
{
    public int maxHealth = 100;
    private int currentHealth;

    void Start()
    {
        currentHealth = maxHealth;
    }

    public void TakeDamage(int amount)
    {
        currentHealth -= amount;
        if (currentHealth <= 0)
        {
            Die();
```

```
        }
    }

    void Die()
    {
        // Logika kematian musuh (misalnya, memusnahkan
objek atau memicu animasi kematian)
        Destroy(gameObject);
    }
}
```