# PRAKTIKUM ANIMASI DAN GAME PERTEMUAN :10

# Enemy AI & Attack

| | | |
|---|---|---|
| **NIM** | **:** | 1918039 |
| **NAMA** | **:** | M.Sofian Attasauri |
| **MATERI** | **:** | Enemy AI & Attack |
| **KELAS** | **:** | D |
| **TUGAS** | **:** | Enemy AI & Attack |

## 10.1 Tujuan

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

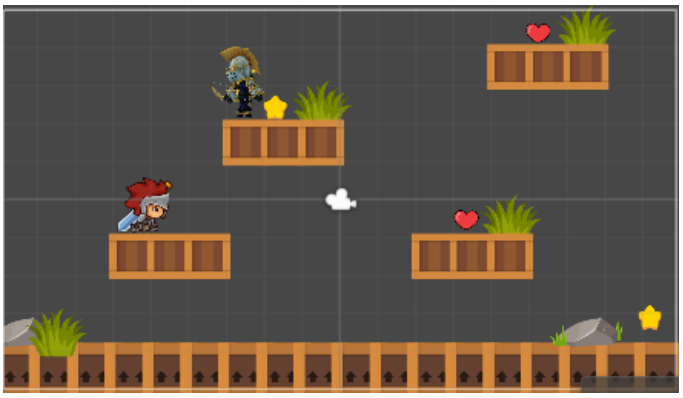
## 10.2 Alat dan Bahan

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

## 10.3 Langkah-Langkah Membuat Tugas

### A. Membuat Mekanisme Serang

1. Buat file *projek Unity* Tugas 9



Gambar 10.1 Buka projek tugas 9

2. Buat File *Script* dalam folder *Script* dengan nama" *PlayerCombat*.cs"


Gambar 10.2 Membuat *Script PlayerCombat*

3. Tambahkan *Source code* berikut dalam folder script *PlayerCombat*.

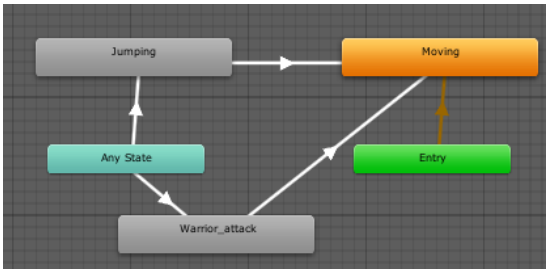
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
public class PlayerCombat : MonoBehaviour
{
    public Animator animator;
    public Transform attackPoint;
    public LayerMask enemyLayers;
    public float attackRange = 0.5f;
    public int attackDamage = 35;
    // Update is called once per frame
    void Update()
    {
        if (Input.GetKeyDown(KeyCode.C))
        {
            Attack();
        }
    }
    void Attack()
    {
        //play attck animation
        animator.SetTrigger("Attack");
        // detect enemy in range of attck
        Collider2D[] hitEnemies =
       Physics2D.OverlapCircleAll(attackPoint.position,
   attackRange,
       enemyLayers);
        //damage ke musuh
        foreach (Collider2D enemy in hitEnemies)
        {

   enemy.GetComponent<EnemyHealth>().TakeDamage(attackD
   amage);
        }
    }
    private void OnDrawGizmosSelected()
    {
        if (attackPoint == null)
            return;
        Gizmos.DrawWireSphere(attackPoint.position,
   attackRange);
    }
```

```
    }
```

4. Buat sebuah *clip* animasi baru di tab *Animation* dan isikan nama menjadi "*Warrior_attack*.anim" lalu tambahkan *Astroattack* dan atur timenya sampai 1:10

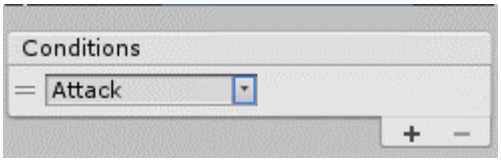

Gambar 10.3 Menambahkan *Warrior Attack*

5. Buatlah sebuah *transtition* untuk *warrior attack*



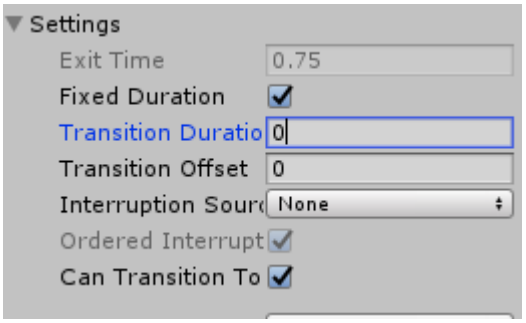

Gambar 10.4 Membuat *Transition Warrior Attack*

6. Klik panah yang mengarah ke karaket_*attack*, pergi ke *Inspector* dan tambahkan *condition* dan pilih *condition* Attack



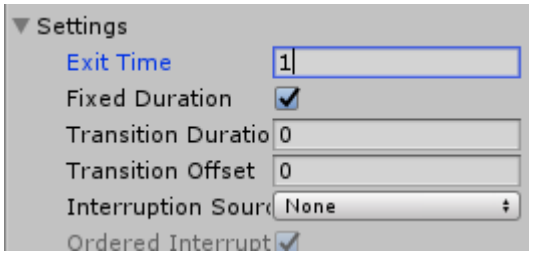

Gambar 10.5 Mengubah *Condition Attack*

7. Klik *setting* dan ubah *Transition Duration* menjadi 0, hilangkan centang *Has Exit Time*, untuk yang lain sesuaikan saja



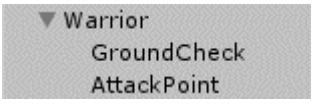

Gambar 10.6 Mengatur *Transition Duration*

8. Klik panah yang mengarah dari *Warrior_attack* ke *Moving*, klik *setting*, ubah *Exit Time* menjadi 1 dan *Transition Duration* menjadi 0




Gambar 10.7 Mensetting *Warrior Attack*

9. Pergi ke *Hierarchy*, klik kanan *warrior* dan pilih *Create Empty*, dan ubah Namanya menjadi *AttackPoint*

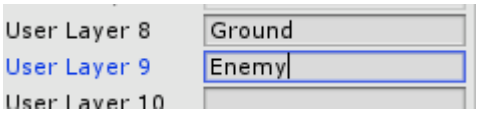



Gambar 10.8 Membuat *Attack Point*

10. Klik game object *warrior*, pergi ke Inspector dan ubah komponen Player Combat seperti berikut

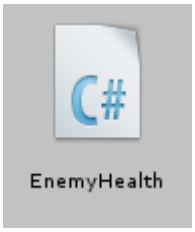

Gambar 10.9 Mengubah komponen *player combat*

11. klik *game object Enemy*, klik *Layer Default* dan pilih *Add Layer*, isikan "*Enemy*" pada user Layer 9




Gambar 10.10 Membuat layer *Enemy*

12. Buat sebuah *file Script* beri nama *EnemyHealth*.cs, simpan didalam folder *script*




Gambar 10.11 Membuat Script *EnemyHealt*

13. Tambahkan *Source Code* seperti dibawah ini

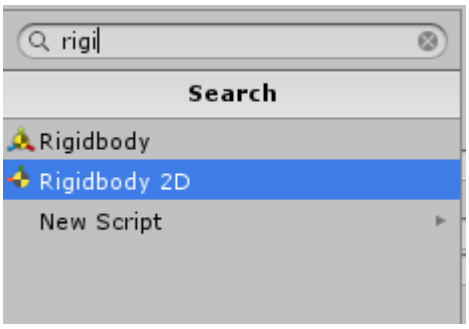
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
public class EnemyHealth : MonoBehaviour
{
    public int maxHealth = 100;
    // Use this for initialization
    void Start()
    {
    }
    // Update is called once per frame
    void Update()
    {
    }
    public void TakeDamage(int damage)
    {
        maxHealth -= damage;
        //hurt anim
        if (maxHealth <= 0)
        {
            Wafat();
        }
    }
    void Wafat()
    {
        Destroy(gameObject);
    }
}
```

14. Jika *diplay* lalu menekan *keybord* C maka *warrior* akan menyerang dan apibila serangan terkena ke *enemy* maka *maxhealtnya* akan berkurang

Gambar 10.12 Tampilan Serang

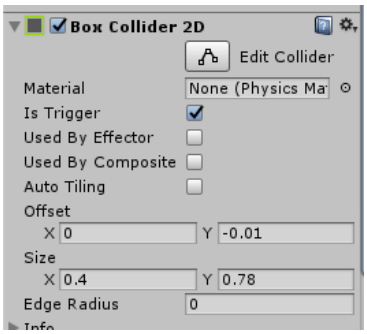
## B. Membuat Enemy AI

1. Klik *game object Enemy*, kemudian *Add Componenet Rigidbody 2D*



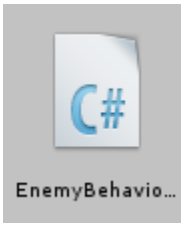

Gambar 10.13 Menambahkan *Rigibody 2D*

2. Atur sedikit ukuran *box collider* Enemy agar sedikit menyentuh dengan *tilemap* dan buat agak sedikit ramping




Gambar 10.14Mengatur *Box Collider*

3. Buat sebuah *file script* baru di dalam folder *script* dan namakan *EnemyBehaviour.cs*.




Gambar 10.15 Membuat *script EnemyBehaviour*

4. Isikan *source code* berikut di dalam *file script EnemyBehaviour,cs*

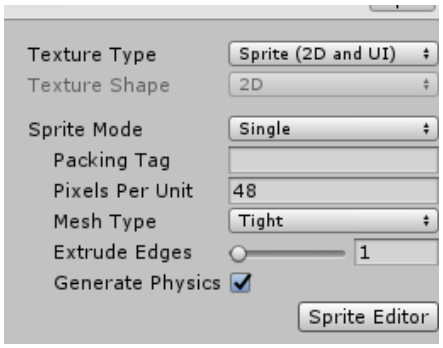
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
public class EnemyBehaviour : MonoBehaviour
{
    [SerializeField]
    float moveSpeed = 1f;
    Rigidbody2D rb;

    // Use this for initialization
    void Start()
```

```
    {
        rb = GetComponent<Rigidbody2D>();
    }
    // Update is called once per frame
    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(-
(Mathf.Sign(rb.velocity.x)),
       transform.localScale.y);
    }
}
```

5. Klik game *object Enemy* dan ubah *Move Speed* menjadi 2 atau bisa disesuaikan

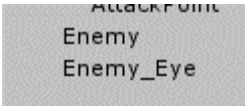

Gambar 10.16 Mengatur *Move Speed*

6. Cari *Flying eye* di folder *Assets Monster* Ubah nilai *Pixels* Per Unit menjadi 48



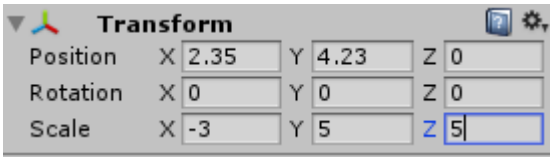

Gambar 10.17 Mengatur *Pixels* per unit

7. Ubah Namanya di *Hierarchy* menjadi *Enemy_Eye*



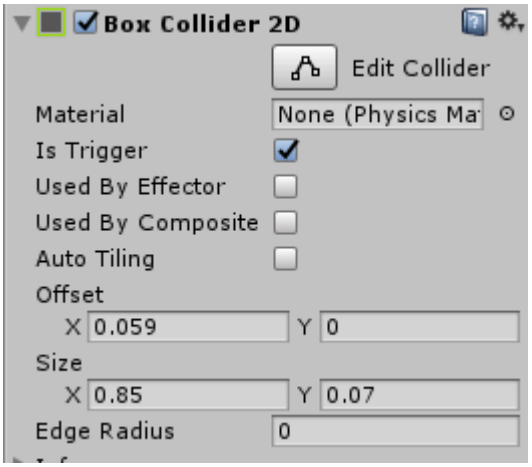

Gambar 10.18 Mengubah nama *Enemy_Eye*

8. Klik *game object Enemy_Eye*, ubah *Layernya* menjadi *Enemy* kemudian sesuaikan nilai *scalenya*



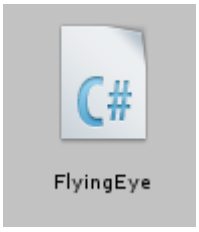

Gambar 10.19 Mengatur nilai *scale*

9. Tetap di *Inspector Enemy_Eye*, tambahkan sebuah komponen bernama *Box collider 2D* dan centang *Is Trigger* dan Atur juga ukuran *box collidernya*




Gambar 10.20 Mengatur *Box Collider 2D*

10. Buat sebuah *file script* bernama *FlyingEye.cs* dan simpan dalam folder *Script*




Gambar 10.21 Membuat *script FlyingEye*

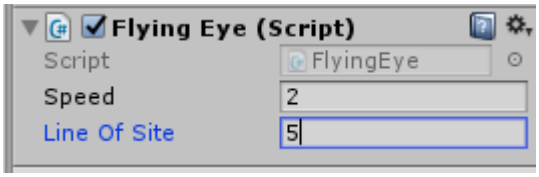
11. Isikan *source code* dibawah ini

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

```
using UnityEngine;

public class FlyingEye : MonoBehaviour {

    public float speed;
    public float lineOfSite;
    private Transform player;
    private Vector2 currentpos;
    Warrior wr;
    // Use this for initialization
    void Start()
    {
        player                                          =
   GameObject.FindGameObjectWithTag("Player").transfor
   m;
        currentpos                                      =
   GetComponent<Transform>().position;
        wr                                              =
   GameObject.Find("Warrior").GetComponent<Warrior>();
    }
    // Update is called once per frame
    void Update()
    {
        float            jarakdariplayer            =
   Vector2.Distance(player.position,
       transform.position);
        if (jarakdariplayer < lineOfSite)
        {
            transform.position =

   Vector2.MoveTowards(this.transform.position,
   player.position, speed *
           Time.deltaTime);
        }
        else
        {
            transform.position                          =
```

```
    Vector2.MoveTowards(transform.position,
            currentpos, speed * Time.deltaTime);
        }
    }
    void OnTriggerEnter2D(Collider2D other)
    {
        if (other.transform.tag == "Player")
        {
            wr.nyawa--;
        }
        if (wr.nyawa < 0)
        {
            wr.play_again = true;
        }
    }
    private void OnDrawGizmosSelected()
    {
        Gizmos.color = Color.red;
        Gizmos.DrawWireSphere(transform.position,
    lineOfSite);
    }
}
```

12. klik game *object Enemy_Eye* dan ubah nilai *speed* dan *Line of Site* sesuai keinginan




Gambar 10.22 Mengatur *speed flying eye*

13. Jika di *play* apabila *warrior* memasuki wilayah *Line of Site Enemy*, maka *Flying eye* akan mengikutinya,


Gambar 10.23 Tampilan *Running*

## 10.4 Kesimpulan

1. Dalam Video *Game*, kecerdasan buatan digunakan untuk membuat perilaku cerdas yang biasanya terletak pada *non*-player *characters* ( *NPCs*), dan seringnya mensimulasikan seperti kecerdasan manusia.
2. *layer-experience modelling*: memahami kemampuan dan kondisi emosional pemain, agar menyesuaikan *game* dengan benar
3. *Procedural-content generation*: membuat elemen dari sebuah lingkungan game seperti kondisi lingkungan, tingkatan, dan bahkan music dengan secara otomatis.

| Nilai | Asisten Laboratorium |
|---|---|
| | (M.Rafi Faddilani)<br>2118144 |