



TUGAS PERTEMUAN: 10

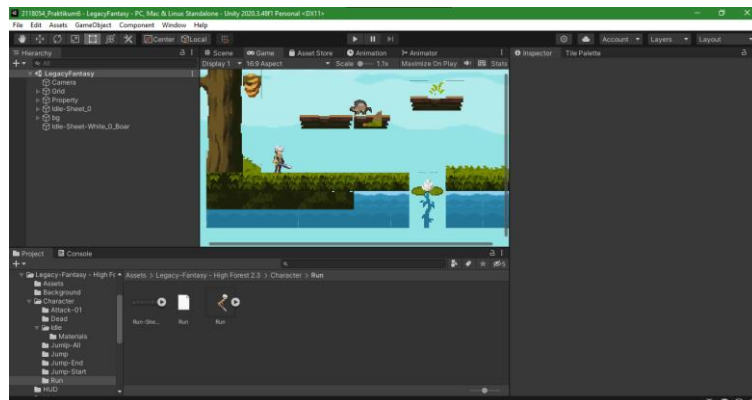
Respawn and AI Enemy Attack

NIM	:	2118054
Nama	:	M. Farish Ramadhan Ferdinansyah
Kelas	:	B
Asisten Lab	:	Difa Fisabilillah (2118052)
Baju Adat	:	
Referensi	:	

10.1 Tugas 10 : Membuat Respawn & AI Enemy Attack

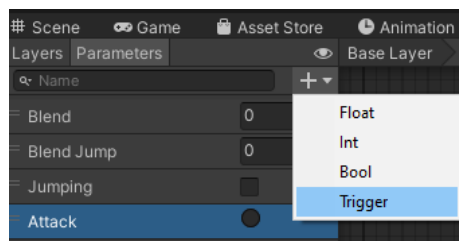
A. Membuat Mekanisme Attack

1. Buka *project Unity* sebelumnya bab 9.



Gambar 10.1 Tampilan *Project Unity*

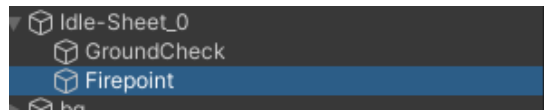
2. Kemudian pada menu *Tab Animator* tambahkan *Parameter Trigger*, *Rename* menjadi *Attack*.



Gambar 10.2 Tampilan *Parameter Attack*

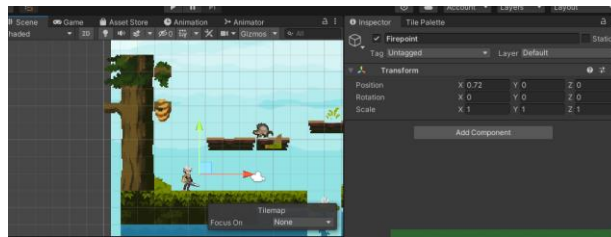


3. Lalu membuat *Layer Game Object* didalam HeroIdle1, klik kanan pilih *Create Empty* kemudian rename menjadi *Firepoint*.



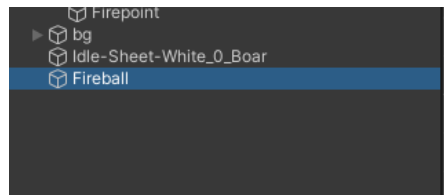
Gambar 10.3 Tampilan *FirePoint*

4. Pada menu *Hierarchy* klik *Firepoint* lalu ke *inspector*, ubah *Icon* menjadi dan atur letak titik di depan *player*.



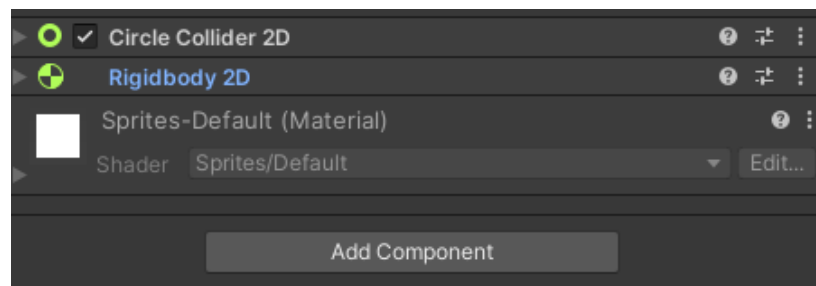
Gambar 10.4 Tampilan atur letak titik

5. Lalu tambahkan *item-feedback-1*, di *folder Sprites > Fx>item-feedback-1*, rename menjadi *fireball*.



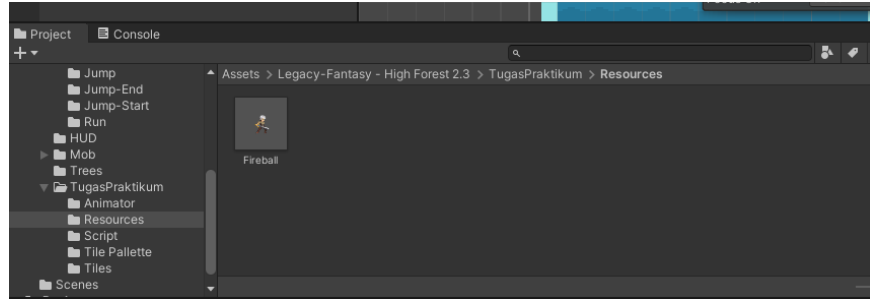
Gambar 10.5 Tampilan *Fireball*

6. Klik *fireball* dan pergi ke *inspector* tambahkan *component Circle Collider 2D* dan *Rigidbody 2D*.



Gambar 10.6 Tampilan *Circle Collider & Rigidbody*

7. Kemudian buat folder baru beri nama *Resources* di menu *project*, kemudian *drag & drop fireball* ke dalam *folder Resources*, kemudian hapus *fireball* pada *Hierarchy*.



Gambar 10.7 Tampilan *Folder Resources*

8. Pada Script player tambahkan script dibawah ini :

```
public class Player : MonoBehaviour
{
    public Animator animator;
    public GameObject bullet;
    public Transform firePoint;
```

Lalu tambahkan script fungsi `fixedUpdate` & `Void Update` dibawah ini

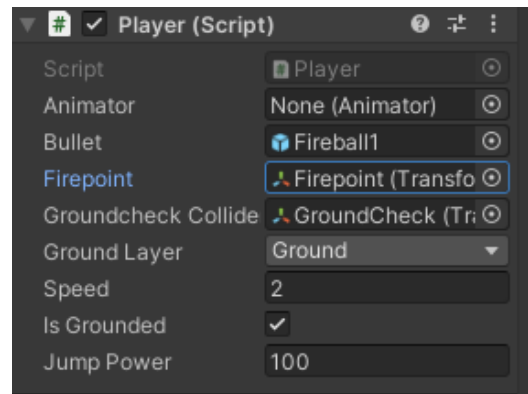
```
IEnumerator Attack()
{
    animator.SetTrigger("Attack");
    yield return new WaitForSeconds(0.25f);
    float direction = facingRight? 1f:-1f;

    GameObject fireball = Instantiate(bullet,
        firePoint.position, Quaternion.identity);
    fireball.GetComponent<Rigidbody2D>().velocity =
        new Vector2(direction * 10f, 0);
    Destroy(fireball, 2f);
}

# Tambahkan pada Function void Update

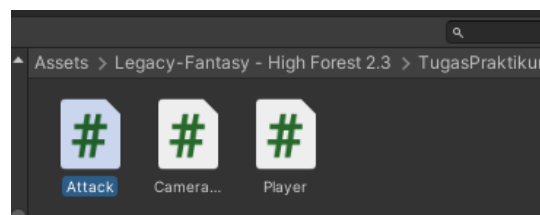
if (Input.GetKeyDown(KeyCode.C))
{
    StartCoroutine(Attack());
}
```

9. Pada *inspector* player ubah *bullet* menjadi *fireball* dan *firePoint* adalah titik tembak pertama.



Gambar 10.8 Tampilan *FirePoint Player*

10. Kemudian buat *script Attack* pada *folder Script*

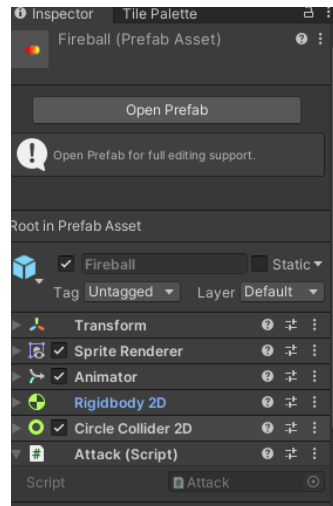


Gambar 10.9 Tampilan *Script Attack*

11. Lalu *Script Attack* dibawah ini

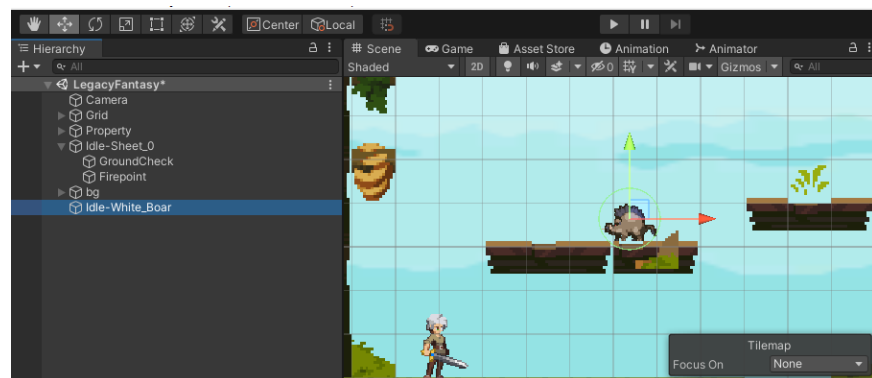
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
public class Attack : MonoBehaviour
{
    private void OnTriggerEnter2D(Collider2D collision)
    {
        if (collision.gameObject.CompareTag("Enemy"))
        {
            Destroy(gameObject);
            Destroy(collision.gameObject);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

12. Kemudian di *folder resource* tambahkan *Script Attack* di *Prefab fireball*, dengan cara klik *fireball* kemudian di *inspector* arahkan *Script Attack* kedalam *Inspector*.



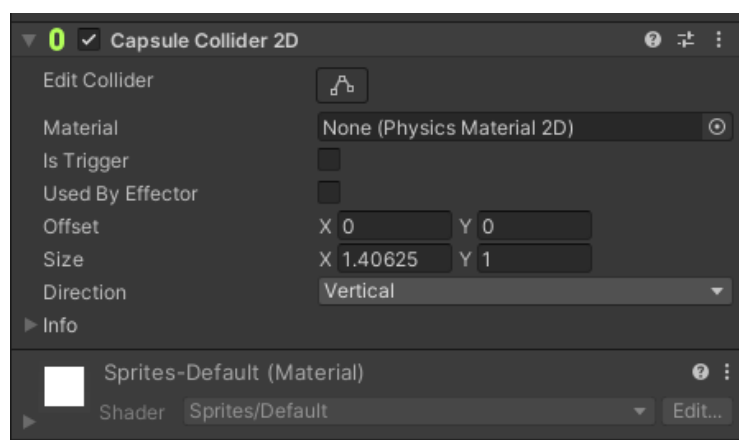
Gambar 10.10 Tampilan *Prefab Fireball*

13. Tambahkan *Enemy Idle-White_Boar* pada *hierarchy* di *folder Sprites*.



Gambar 10.11 Tampilan *Idle-White_Boar*

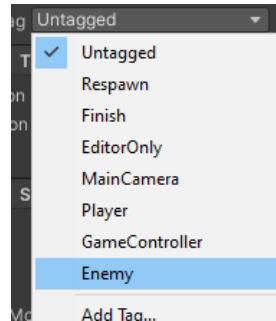
14. Klik pada *ghost-halo-1*, pergi ke *inspector* tambahkan *Collider 2D* untuk mendeteksi.





Gambar 10.12 Tampilan *Capsule Collider 2D*

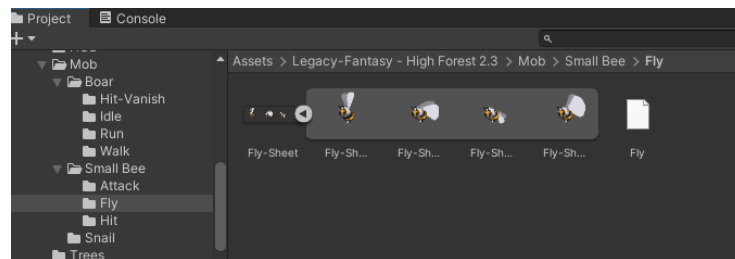
15. Tambahkan *tag enemy* dengan cara pilih *Add tag*, kemudian *add tag to the list*, lalu tuliskan *enemy*.



Gambar 10.13 Tampilan *Tag Enemy*

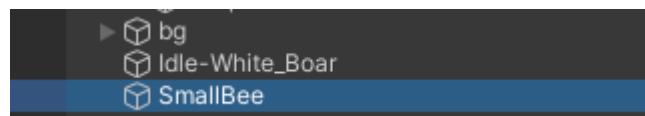
B. Enemy Behavior NPC

1. Cari sebuah *sprite pack* bernama *mob* dan buka *folder* bernama “Small Bee”



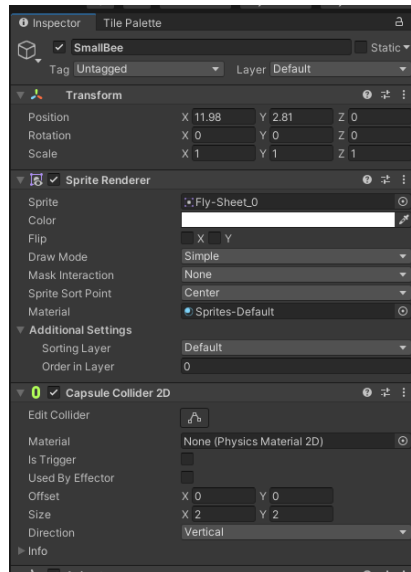
Gambar 10.15 Tampilan skeleton

2. Tambahkan “SmallBee” ke *Hierarchy*



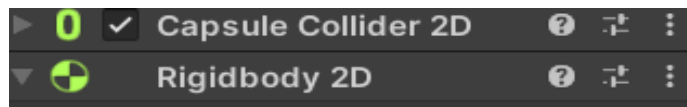
Gambar 10.16 Tampilan Skeleton

3. Pada *inspector* atur *transform Scale* menjadi seperti berikut



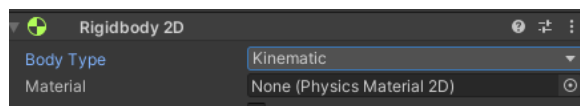
Gambar 10.17 Tampilan *Setting Transform*

4. Tambahkan *component* bernama *Capsule Collider 2D* & *Rigidbody 2D* pada *Skeleton*



Gambar 10.18 Tampilan *Capsule & Rigidbody* pada *Skeleton*

5. Atur sedikit *Collider* dan pada *Body Type* ubah menjadi *Kinematic*



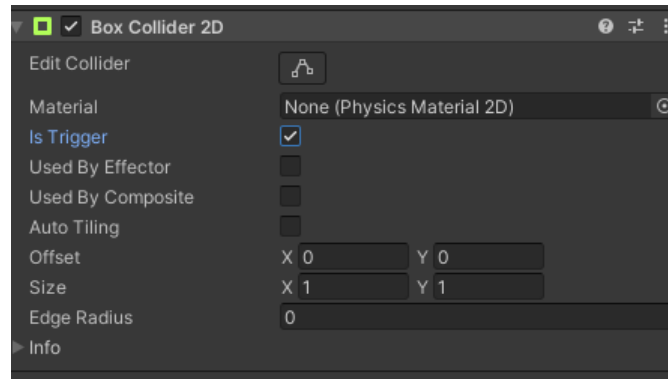
Gambar 10.19 Tampilan *Kinematic*

6. *Create Empty object* pada *Hierarchy*, *rename* menjadi *Boundary*



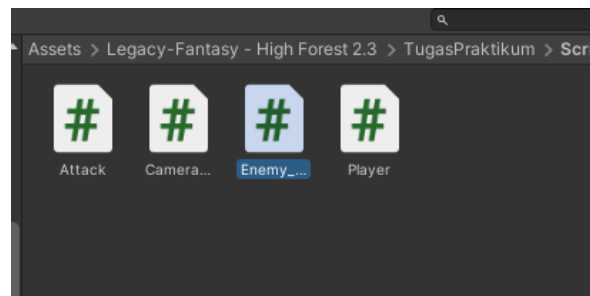
Gambar 10.20 Tampilan *Boundary*

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



Gambar 10.21 Tampilan *Box Collider 2D*

8. Buat file *Script* didalam *folder script* beri nama “Enemy_Behavior” kemudian *drag* masukan ke dalam game object “Skeleton”.



Gambar 10.22 Tampilan *Enemy Behavior*

9. 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()
```

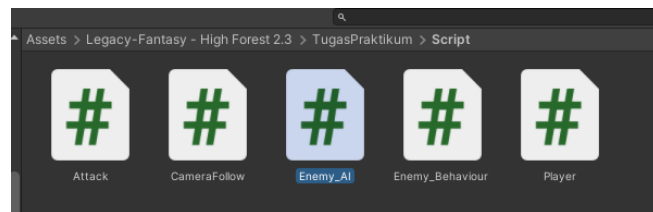



```
{
    return transform.localScale.x > Mathf.Epsilon;
}

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

C. Enemy AI

1. Buat Script “Enemy_AI” pada folder *Praktikum Script*



Gambar 10.24 Tampilan *Enemy AI*

2. 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
    private bool facingRight = true; // Menunjukkan
    apakah musuh menghadap ke kanan

    // Use this for initialization
    void Start()
    {
        // Mencari pemain berdasarkan tag
        player =
GameObject.FindGameObjectWithTag("Player").transfo
rm;
        // 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);
    FacePlayer(); // Memutar musuh untuk
menghadap pemain
}
else
{
    // Musuh kembali ke posisi awal
    transform.position =
Vector2.MoveTowards(transform.position,
initialPosition, speed * Time.deltaTime);
    FaceInitialPosition(); // Memutar musuh
untuk menghadap posisi awal jika diperlukan
}
}

// Memutar musuh untuk menghadap pemain
void FacePlayer()
{
    if (player.position.x > transform.position.x &&
facingRight)
    {
        Flip();
    }
    else if (player.position.x <
transform.position.x && !facingRight)
    {
        Flip();
    }
}

// Memutar musuh untuk menghadap posisi awal jika
diperlukan
void FaceInitialPosition()
{
    if (initialPosition.x < transform.position.x &&
facingRight)
    {
        Flip();
    }
    else if (initialPosition.x >
transform.position.x && !facingRight)
    {
        Flip();
    }
}

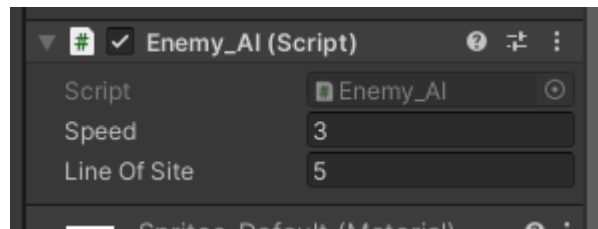
// Membalik orientasi musuh
void Flip()
{
    facingRight = !facingRight;
    Vector3 scaler = transform.localScale;
    scaler.x *= -1;
}
```



```
transform.localScale = scaler;
}

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

3. Pada *Inspector Enemy_AI*, atur *Speed* juga *Line of Site* untuk menentukan jarak dan *speed* pada enemy



Gambar 10.25 Tampilan menentukan jarak dan *Speed*

D. Respawn

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

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

2. Tambahkan kode dibawah untuk mengatur posisi *respawn* sesuai dengan posisi awal permainan

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

    respawn_loc = transform.position;
}
```

3. Tambahkan didalam *void update Player.cs* agar nyawa player dibawah 0 maka akan melakukan *respawn*.

```
If (nyawa < 0)
{
    Playagain();
}
```



4. Tambahkan kode dibawah ini jika player jatuh dibawah platform maka 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;
    }
}
```

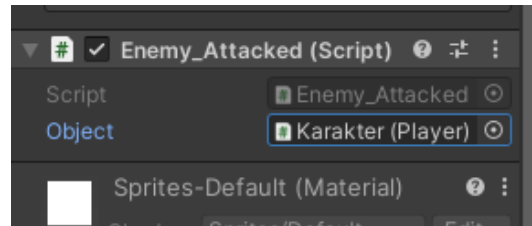
6. Tambahkan *file script*(Enemy_Attacked.cs) dan isikan *source code* dibawah ini

```
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--;

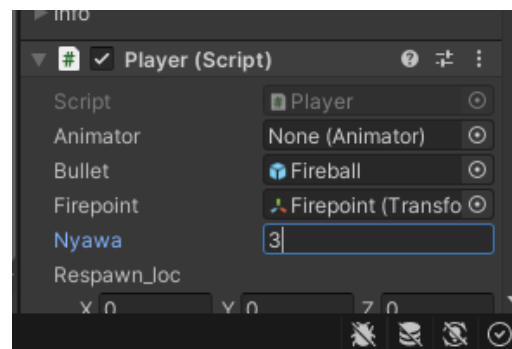
            if (Object.nyawa < 0)
            {
                Object.play_again = true;
            }
        }
    }
}
```

7. Tambahkan *script enemy attack*, arahkan *object* pada Idle-White_Boar



Gambar 10.26 Tampilan *Script Enemy Attack*

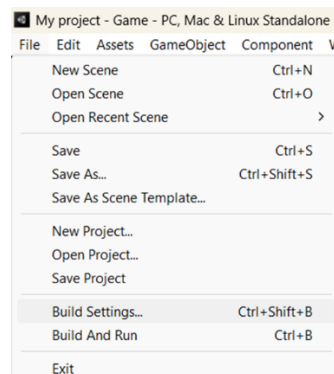
8. Klik game *object* Player, pergi ke *Inspector* dan ubah nilai nyawa menjadi 3 pada *Player (Script)*



Gambar 10.27 Tampilan *Player Nyawa*

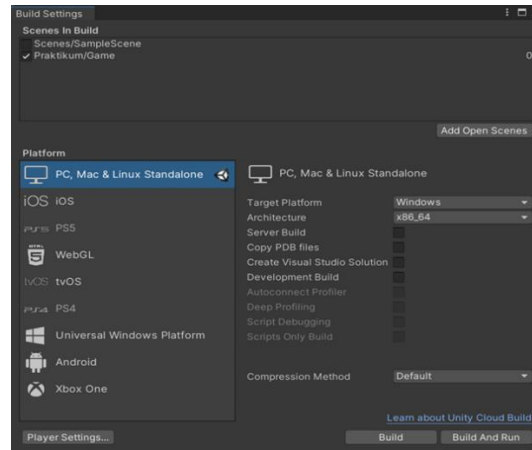
E. Render

1. Pergi ke menu *file* pilih *Build Setting* (Ctrl + Shift + B)



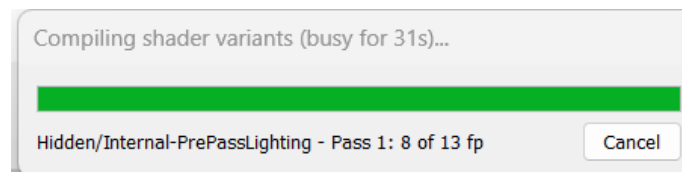
Gambar 10.28 Tampilan *Menu File*

2. Pada *Setting Build* ini pilih PC, Mac & Linux, Tekan *Build*, pastikan pada menu *Scene in Build* berada pada *project* Tugas Kalian



Gambar 10.29 Tampilan *Setting Build*

3. Pilih dimana *project* disimpan dan tunggu hasilnya



Gambar 10.30 Tampilan Tunggu hasil Project

F. Link Pengumpulan Github

Link : https://github.com/MichaelKevv/2118069_PRAK_ANIGAME

KUIS

Lengkapi Source Code dibawah ini:

Soal kuis Bab 10

```
using UnityEngine;

public class PlayerAttack : MonoBehaviour
{
    public float attackRange = 2.0f; // Ganti int dengan float
    untuk jarak serangan
    public int attackDamage = 10; // Perbaiki nama variabel
    dari "attacDamage" ke "attackDamage"

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

    void PerformMeleeAttack()
    {

```



```
RaycastHit hit;
if (Physics.Raycast(transform.position,
transform.forward, out hit, attackRange))
{
    // Periksa apakah objek yang terkena memiliki
    komponen EnemyHealth
    EnemyHealth enemyHealth =
hit.transform.GetComponent<EnemyHealth>();
    if (enemyHealth != null)
    {
        // Mengurangi health musuh
        enemyHealth.TakeDamage(attackDamage);
    }
}
}
```

Analisa :

Pada kode yang telah diperbaiki, fungsi utama dari PlayerAttack yaitu untuk mengelola serangan melee pemain dalam game Unity. Sewaktu pemain menekan tombol yang ditentukan (Fire1), metode PerformMeleeAttack dipanggil. Metode ini akan menggunakan Physics.Raycast untuk memancarkan sinar dari posisi pemain ke arah depan. Apabila sinar tersebut mengenai objek dalam jarak yang ditentukan (attackRange), selanjutnya akan memeriksa apakah objek yang terkena memiliki komponen EnemyHealth. Jika iya, maka metode TakeDamage dari EnemyHealth dipanggil dengan parameter attackDamage, yang mengurangi kesehatan (currentHealth) dari musuh tersebut. Jika kesehatan musuh mencapai nol atau kurang, metode Die dipanggil untuk memusnahkan atau menonaktifkan musuh.

Kuis 10

berikan tanda merah yang menyebabkan source code error

```
void HandleJumpInput()
{
    if (Input.GetKeyDown(KeyCode.Space))
    {
        animator.SetBool("isJumping", );
        rb.AddForce(Vector2.up * jumpForce,
ForceMode2D.Impulse);
    }
    else if (Input.GetKey(KeyCode.Space))
    {
        animator.SetBool("isJumping",);
    }
}

void HandleMovementInput()
```



```
{
    float move = Input.GetAxis("Horizontal");

    if (move != 1)
    {
        animator.SetBool("isIdle", true);
        transform.Translate(Vector3.left * move *
Time.deltaTime);
    }
    else
    {
        animator.SetBool("isWalking", false);
    }

    if (move != 0)
    {
        transform.localScale = new Vector3(-4, 1, 1);
    }
    else if (move > 0)
    {
        transform.localScale = new Vector3(1, 2, 1);
    }
}
```

Analisa :

Pada source code diatas, penyebab error pertama pada “**animator.SetBool("isJumping",);**” hal ini terjadi karena seharusnya terdapat nilai boolean (True atau False) yang mengakibatkan error krn sintaknya tidak lengkap. Selanjutnya yaitu “**if (move != 1)**” yang mana seharusnya bernilai 0 agar player bisa bergerak dan tidak hanya diam, ketiga “**transform.Translate(Vector3.left * move * Time.deltaTime);**” seharusnya script tersebut di taruk dibagian else agar vektor bisa bergerak ke arah lainnya, keempat “**animator.SetBool("isWalking", false);**” karena seharusnya set walking ke true agar bisa bergerak dan tidak diam, kelima “**(-4, 1, 1);**” dan “**(1, 2, 1);**” karena seharusnya bernilai (-1,1,1) agar negative x for left agar tidak menimpa kondisi else if yang akan dijalankan lalu seharusnya bernilai (1,1,1) agar positive x for right