TUGAS PERTEMUAN: 10

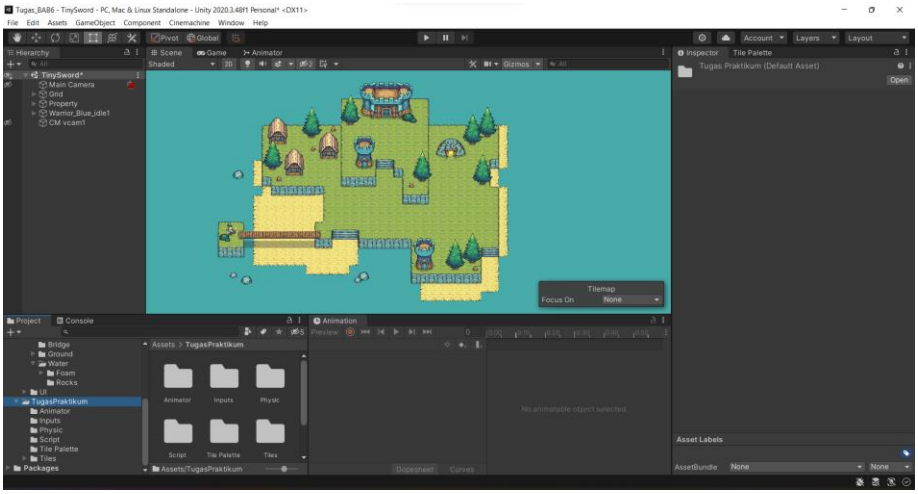
RESPAWN & AI ENEMY ATTACK

NIM	:	2118064
Nama	:	Ilham Maulana Prasetyo
Kelas	:	B
Asisten Lab	:	Bagas Anardi Surya W (2118004)

10.1 Tugas 1 : Membuat Mekanisme Attack, Enemy AI, dan Respawn

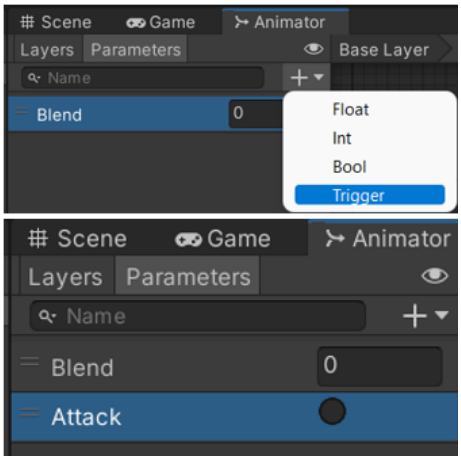
A. Mekanisme Attack

1. Buka Project Unity Tugas Bab 9 sebelumnya.



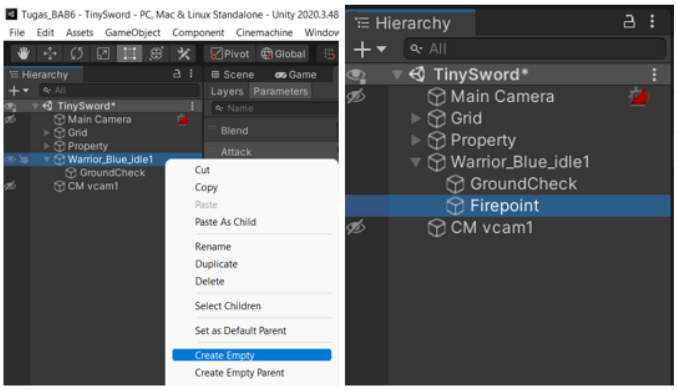
Gambar 10.1 Membuka Project Unity

2. Pada menu Tab *Animator* tambahkan parameter Trigger, dan beri nama *Attack.*



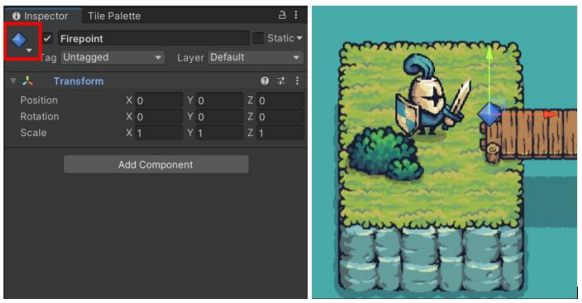
Gambar 10.2 Menambahkan Parameter Trigger Attack

3. Selanjutnya pada *Hierarchy* buat Layer *GameObject* baru didalam Warrior_Blue_idle1 dengan nama *Firepoint*.



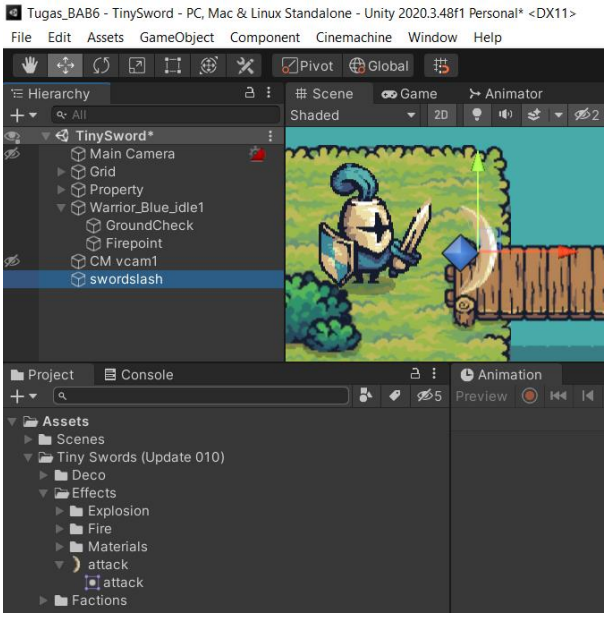
Gambar 10.3 Membuat Layer Firepoint

4. Pada menu *Hierarchy* klik *Firepoint* untuk *setting* pada *Inspector*, ubah Icon menjadi titik, atur letak titik didepan player.



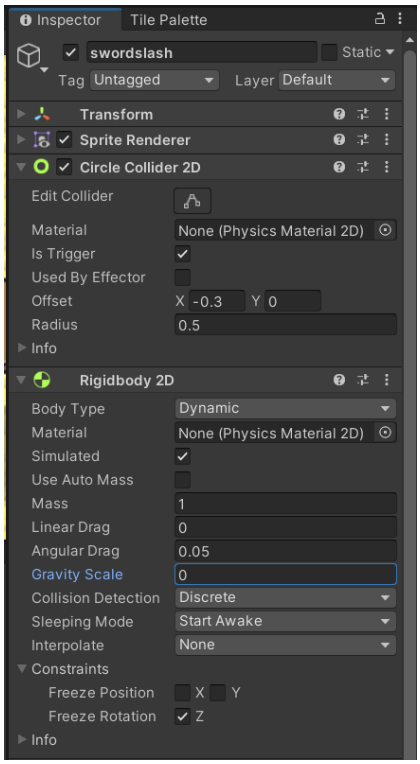
Gambar 10.4 Mengubah Icon Firepoint

5. Pada menu *Hierarchy* tambahkan attack, di folder Effects > attack, *rename* menjadi *swordslash*.



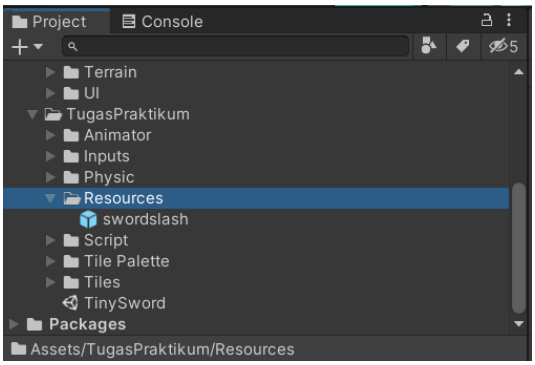
Gambar 10.5 Menambahkan Asset attack

6. Klik *swordslash* untuk menambahkan *Component Circle Collider* 2D, dan Riggidbody 2D, setting sesuai gambar dibawah ini.



Gambar 10.6 Menambahkan *Circle Collider* 2D

7. Buat folder baru bernama *Resources* di folder TugasPraktikum, kemudian *drag and drop swordslash* kedalam folder *Resources*. Kemudian hapus *swordslash* pada *Hierarchy*.

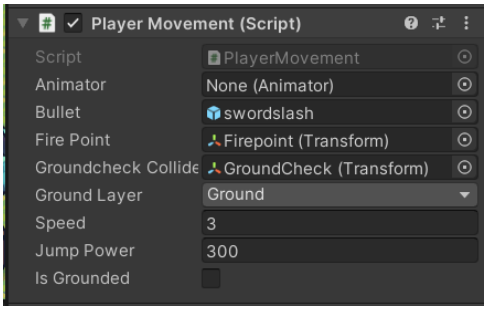


Gambar 10.7 Membuat Folder *Resources*

8. Pada Script PlayerMovement tambahkan script dibawah ini.

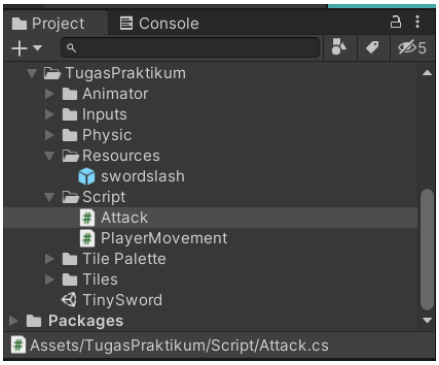
```
#Tambahkan pada class PlayerMovement
public GameObject bullet;
public Transform firePoint;


#Tambahkan dibawah fungsi fixedUpdate
IEnumerator Attack()
{
  animator.SetTrigger("Attack");
  yield return new WaitForSeconds(0.25f);

  float direction = facingRight ? 1f : -1f;

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

  if (direction < 0)
  {
    Vector3 theScale = swordslash.transform.localScale;
    theScale.x *= -1;
    swordslash.transform.localScale = theScale;
  }
  else if (direction > 0)
  {
    Vector3 theScale = swordslash.transform.localScale;
    theScale.x *= 1;
    swordslash.transform.localScale = theScale;
  }

  Destroy(swordslash, 2f);
}


#Tambahkan pada fungsi Update
if (Input.GetKeyDown(KeyCode.Mouse0))
{
  StartCoroutine(Attack());
  animator.SetBool("Attacking", true);
}
else if (Input.GetKeyUp(KeyCode.Mouse0))
{
  animator.SetBool("Attacking", false);
}
```

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 Tampilan *Inspector* Player

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

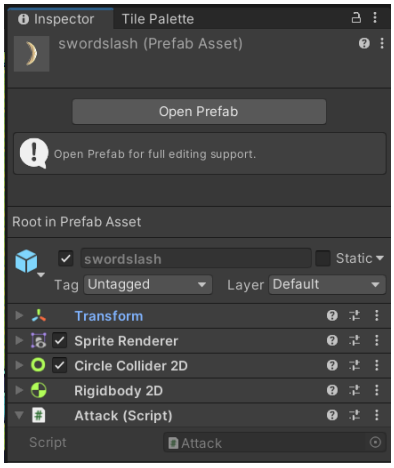


Gambar 10.9 Membuat *Script Attack*

11. Tambahkan *Script Attack* dibawah ini.

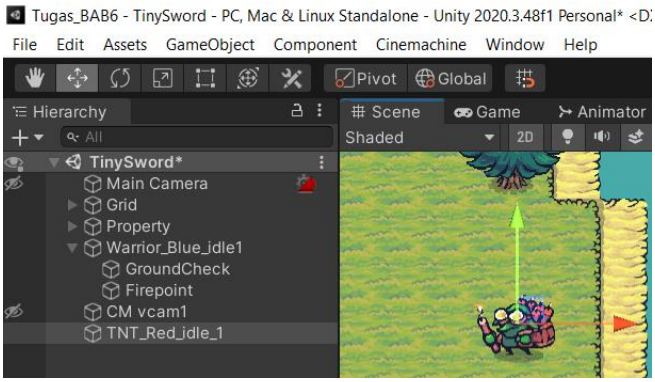
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

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 *swordslash*, dengan cara klik *swordslash* kemudian arahkan *Script Attack* kedalam *Inspector*.



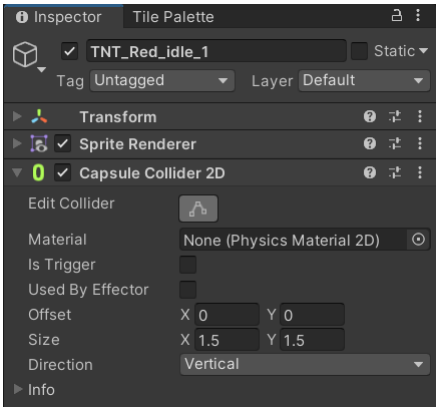
Gambar 10.10 Menambahkan *Component Script Attack*

13. Tambahkan *enemy* Goblins TNT_Red_idle_1 pada *hierarchy* di folder Faction > Goblins.



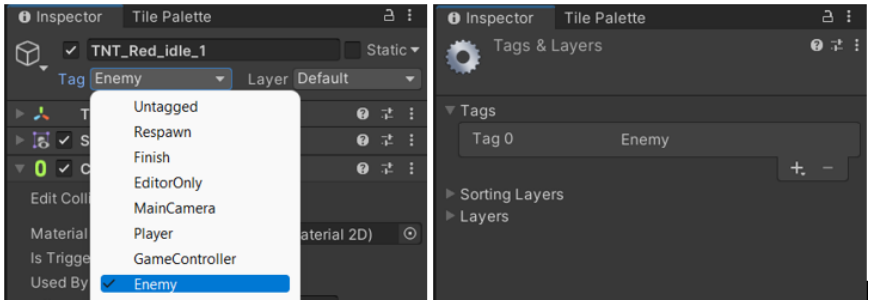
Gambar 10.11 Menambahkan *Enemy* Goblin TNT

14. Kemudian klik pada TNT_Red_idle_1, lalu pada menu tab *Inspector* tambahkan Capsule Collider 2D untuk mendeteksinya.



Gambar 10.12 Menambahkan Capsule Collider 2D

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



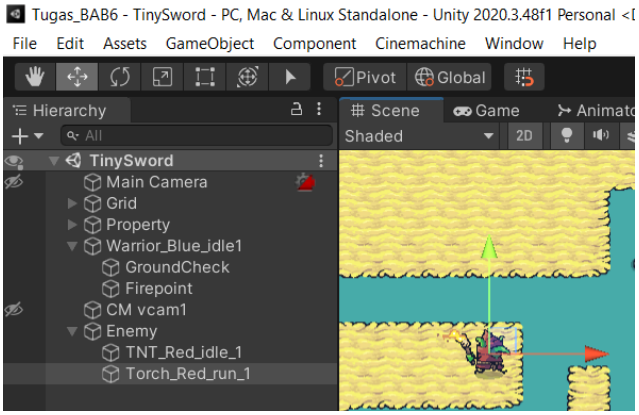
Gambar 10.13 Menambahkan *Tag Enemy*

16. Tembak Enemy dengan menekan tombol kiri mouse untuk menghancurkan musuh.

Gambar 10.14 Tampilan Mekanisme *Attack*

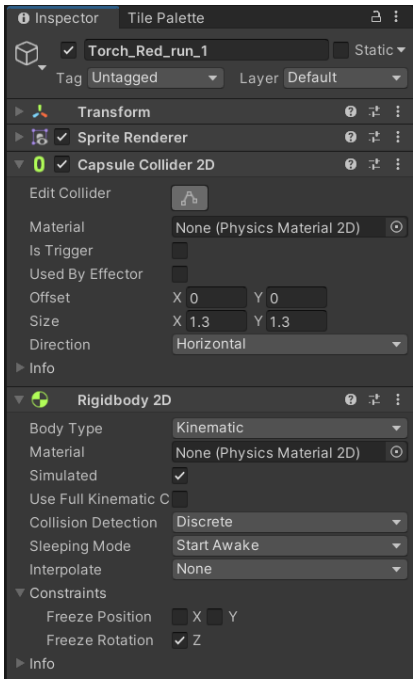
B. Enemy Behavior NPC

1. Tambahkan *enemy* Goblins Torch_Red_run_1 pada *hierarchy* di folder Faction > Goblins



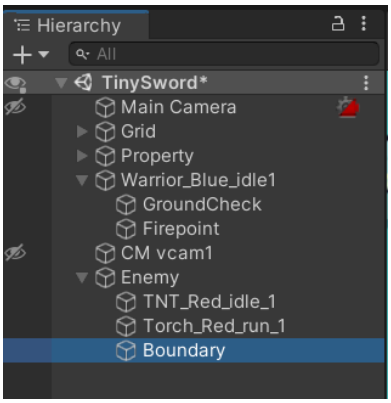
Gambar 10.15 Menambahkan *Enemy* Goblin *Torch*

2. Klik pada Torch_Red_run_1, lalu pada menu tab *Inspector* tambahkan Capsule Collider 2D dan Rigidbody 2D, lalu atur seperti gambar ini.



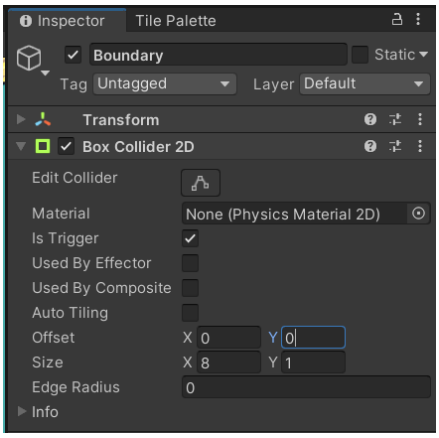
Gambar 10.16 Menambahkan Capsule Collider 2D

3. *Create Empty object* pada hierarchy, Rename menjadi Boundary.



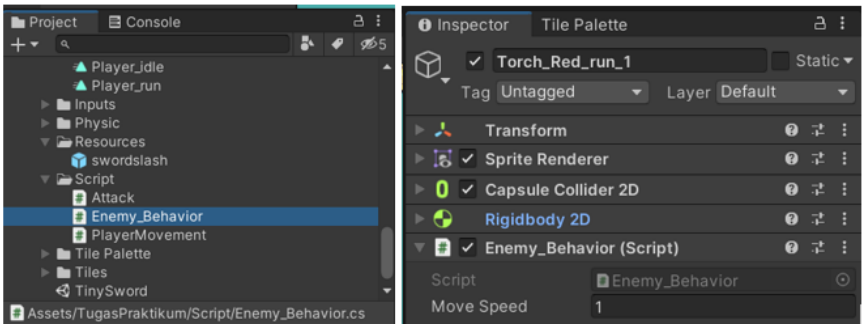
Gambar 10.17 Membuat Boundary

4. Tambahkan Box Collider 2D pada Boundary, centang pada *Is Trigger* lalu atur size dan offside seperti gambar dibawah ini.



Gambar 10.18 Menambahkan Box Collider 2D

5. Buat sebuah file *script* didalam folder Script TugasPraktikum lalu beri nama "Enemy_Behavior", kemudian *drag* dan masukkan ke dalam *game object* Torch_Red_run_1.



Gambar 10.19 Membuat Script Enemy_Behavior

6. Pada *Script* Enemy_Behavior 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);
```

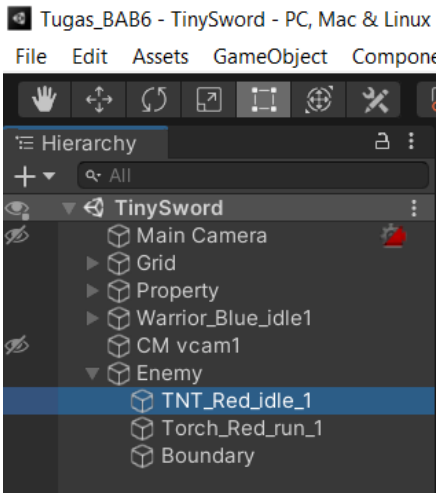
```
    }
    else
    {
      rb.velocity = new Vector2(-moveSpeed, 0f);
    }
  }

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

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

7. Jalankan game untuk melihat *enemy behavior*.

Gambar 10.20 Tampilan Hasil *Enemy Behavior*

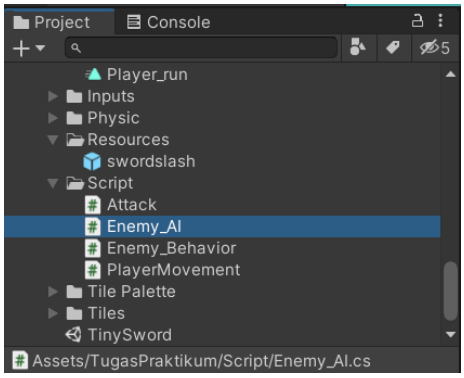
C. Enemy AI

1. Pada *hierarchy* cari TNT_Red_idle_1 dan tekan object.



Gambar 10.21 TNT_Red_idle_1

2. Buat sebuah file *script* didalam folder Script TugasPraktikum lalu beri nama “Enemy_AI”, kemudian *drag* dan masukkan ke dalam *game object* TNT_Red_idle_1.



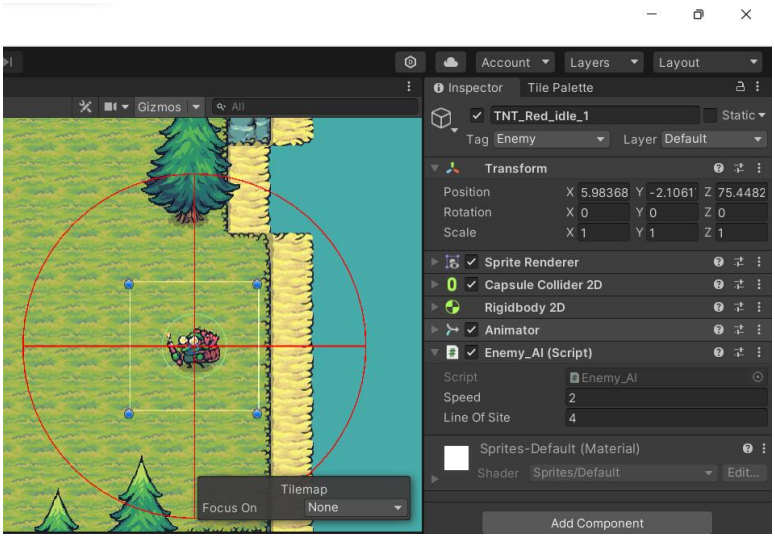
Gambar 10.22 Membuat Script Enemy_AI

3. Pada *Script* Enemy_AI tambahkan *script* dibawah ini.

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

public class Enemy_AI : MonoBehaviour
{
  public float speed;
  public float lineOfSite;
  private Transform player;
  private Vector2 initialPosition;
  private Animator animator;
  private bool facingRight = true;

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

  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);
      animator.SetBool("isRunning", true);
      animator.SetBool("isReturning", false);
```

```
      FaceDirection(player.position.x -
transform.position.x);
    }
    else
    {
      // Musuh kembali ke posisi awal
      transform.position =
Vector2.MoveTowards(transform.position,
initialPosition, speed * Time.deltaTime);
      animator.SetBool("isReturning", true);

      FaceDirection(initialPosition.x -
transform.position.x);

      // Jika sudah kembali ke posisi semula, hentikan
animasi run
      if (Vector2.Distance(transform.position,
initialPosition) < 0.1f)
      {
        animator.SetBool("isRunning", false);
      }
    }
  }

  // Menghadapi arah berdasarkan arah gerakan
  private void FaceDirection(float direction)
  {
    if (direction > 0 && !facingRight)
    {
      Flip();
    }
    else if (direction < 0 && facingRight)
    {
      Flip();
    }
  }

  // Membalikkan arah karakter
  private void Flip()
  {
    facingRight = !facingRight;
    Vector3 theScale = transform.localScale;
    theScale.x *= -1;
    transform.localScale = theScale;
  }

  // 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* dan *Line of Site* untuk menentukan kecepatan dan jarak pada *enemy*.



Gambar 10.23 Mengatur *Speed* dan *Line of Site*

5. Jalankan *game*, maka *enemy* TNT_Red_idle_1 akan mengikuti gerakan player.

Gambar 10.24 Tampilan Hasil Enemy AI

D. Respawn

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

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

Gambar 10.25 Menambahkan Variabel Nyawa

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

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

    respawn_loc = transform.position;
}
```

Gambar 10.26 Menambahkan Posisi Respawn

3. Tambahkan kode dibawah ini di dalam void Update agar ketika nyawa player dibawah 0 maka akan melakukan *respawn*.

```
void Update()
{
    GroundCheck();
    horizontalValue = Input.GetAxisRaw("Horizontal");
    verticalValue = Input.GetAxisRaw("Vertical");
    if (Input.GetButtonDown("Jump")) ···
    else if (Input.GetButtonUp("Jump")) ···

    //menyerang
    if (Input.GetKeyDown(KeyCode.Mouse0)) ···
    else if (Input.GetKeyUp(KeyCode.Mouse0)) ···

    //play again
    if (nyawa < 0)
    {
        playagain();
    }
}
```

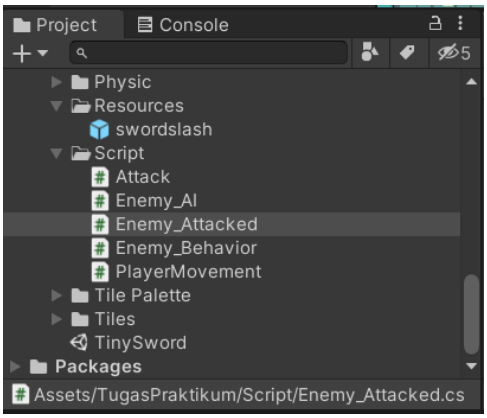
Gambar 10.27 Menambahkan Kode Nyawa Habis

4. Tambahkan fungsi playagain() dibawah fungsi Awake.

```
private void Awake() ···

1 reference
void playagain()
{
    if (play_again == true)
    {
        nyawa = 5;
        transform.position = respawn_loc;
        play_again = false;
    }
}
```

Gambar 10.28 Menambahkan Fungsi Respawn

5. Buat sebuah file *script* didalam folder Script TugasPraktikum lalu beri nama “Enemy_Attacked”

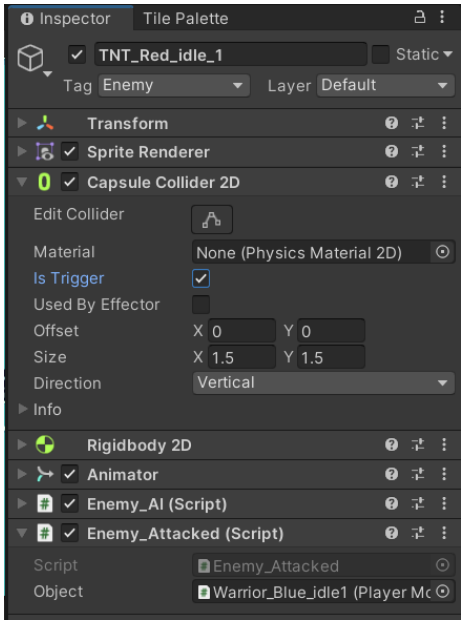


Gambar 10.29 Membuat Script Enemy_Attacked

6. Pada *Script* Enemy_Attacked tambahkan *script* dibawah ini.

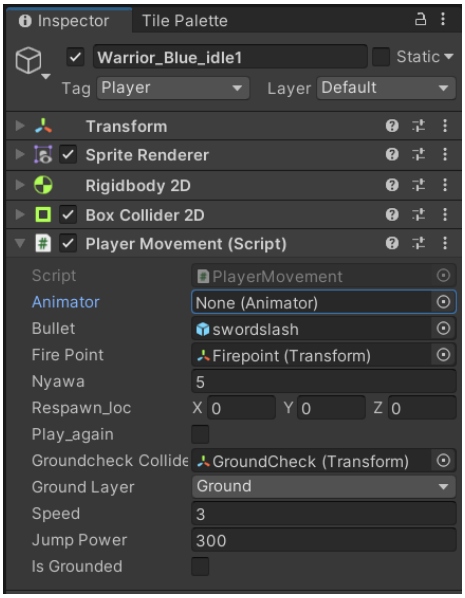
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
public class Enemy_Attacked : MonoBehaviour
{
  [SerializeField] private PlayerMovement Object;
  void Start()
  {
    if (Object == null)
    {
      Object =
GameObject.FindWithTag("Player").GetComponent<PlayerMov
ement>();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

7. Pada *hierarchy* klik TNT_Red_idle_1, pergi ke *Inspector* tambahkan komponen *script* Enemy_Attacked, arahkan Object pada Warrior_Blue_idle1. Centang juga *Is Trigger* pada Capsule Collider 2D.



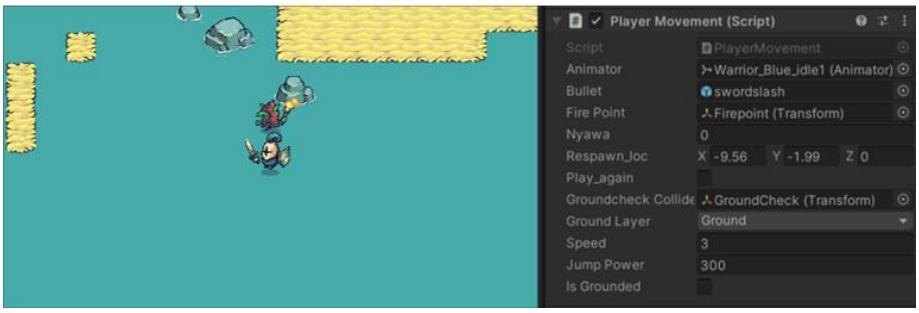
Gambar 10.30 Menambahkan Komponen Script Enemy_Attacked

8. Pada *hierarchy* klik Warrior_Blue_idle1, pergi ke *Inspector* dan ubah nilai Nyawa menjadi 5 pada komponen *script* PlayerMovement.



Gambar 10.31 Mengubah Nilai Nyawa Player

9. Jika di play, Player mengenai atau menyentuh Enemy sebanyak 5 kali maka nyawa akan berkurang 1 dan jika nyawa kurang dari 0 maka akan respawn ke titik awal.



Gambar 10.32 Tampilan Hasil Respawn

E. Link Github

https://github.com/ilham-pras/2118064_PRAK_ANIGAME.git

10.2 Kuis

Melangkapi source code dibawah ini:

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

Analisa:

Terdapat dua variabel publik attackRange dan attackDamage yang digunakan untuk menentukan jangkauan serangan dan jumlah kerusakan yang dihasilkan dari serangan. Dalam metode Update memeriksa apakah pemain menekan tombol “Fire1”, jika ditekan maka akan memanggil metode PerformMeleeAttack.

Pada metode PerformMeleeAttack menggunakan Raycast untuk menyerang, jika raycast mengenai sebuah collider, akan memeriksa apakah objek yang terkena serangan memiliki komponen “EnemyHealth”. Jika memiliki maka akan dipanggil metode “TakeDamage” pada script “EnemyHealth” yang fungsinya untuk mengurangi nilai health pada objek. Jika nilai health kurang lebih sama dengan 0 maka objek akan dihancurkan.