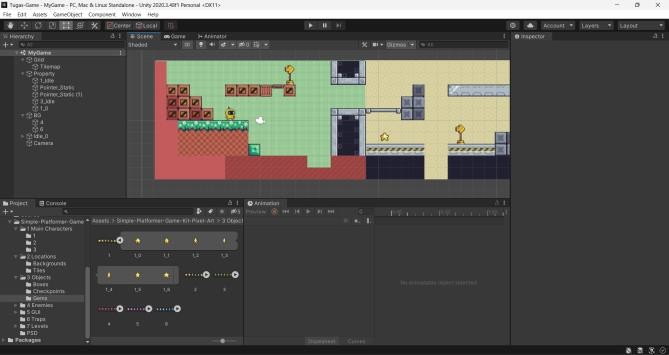
TUGAS PERTEMUAN: 10 RESPAWN AND AI ENEMY ATTACK

|  |  |  |
| --- | --- | --- |
| **NIM** | : | 1918027 |
| **Nama** | : | Dirgarianda R C Rumabar |
| **Kelas** | : | D |
| **Asisten Lab** | : | M. RAFI FADDILANI (2118114) |

# Tugas 1 : Membuat Respawn And AI Enemy Attack

**A. Mempersiapkan Mekanisme Menyerang**

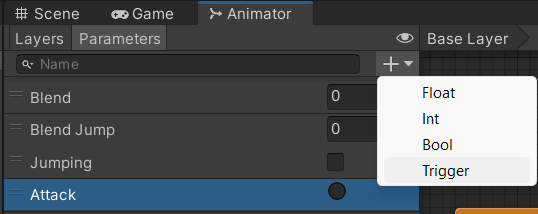
* + 1. Buka *project* Unity sebelumnya yang telah ditambahkan *game animation*.



Gambar 10.1 Membuka File Project

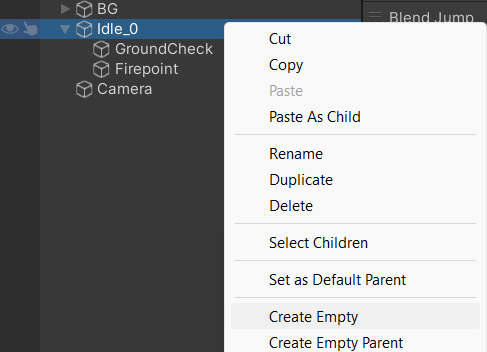
* + 1. Buka tab Animator, tambahkan parameter *Trigger* dengan nama

*Attack*.



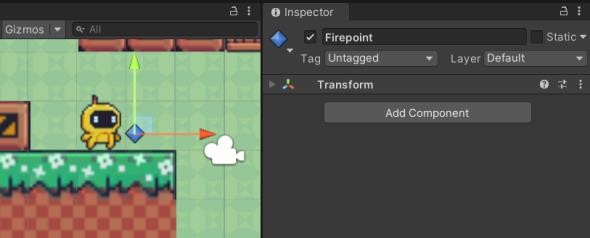
Gambar 10.2 Menambah Parameter

* + 1. Selanjutnya buat *layer game object* baru di dalam Idle\_0, klik kanan pilih *Create Empty* dan ubah nama menjadi *Firepoint*.



Gambar 10.3 Membuat Game Object Baru

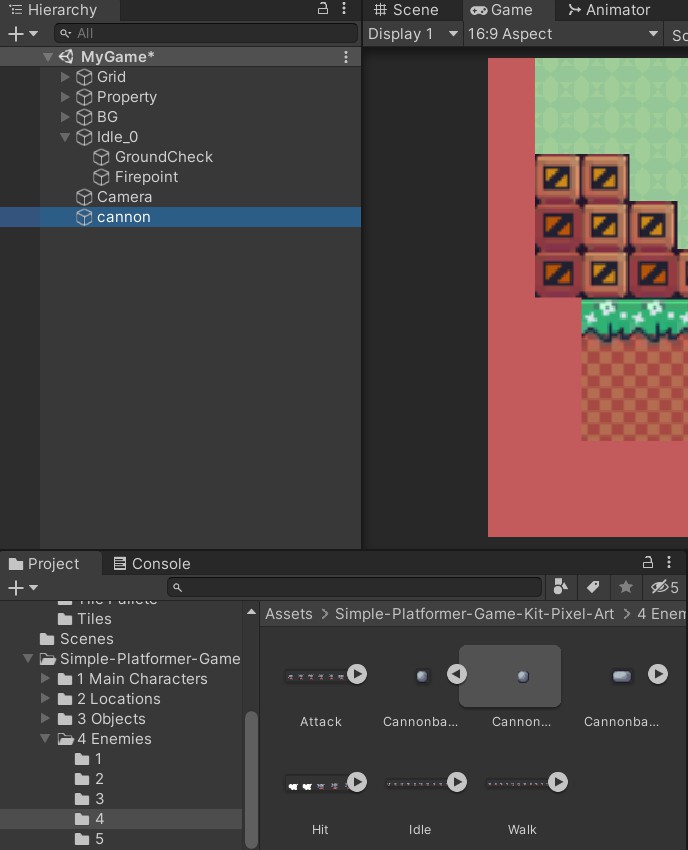
* + 1. Kemudian atur *icon Firepoint* menjadi titik melalui *tab Inspector* dan letakkan di depan *player*.



Gambar 10.4 Memindahkan Firepoint

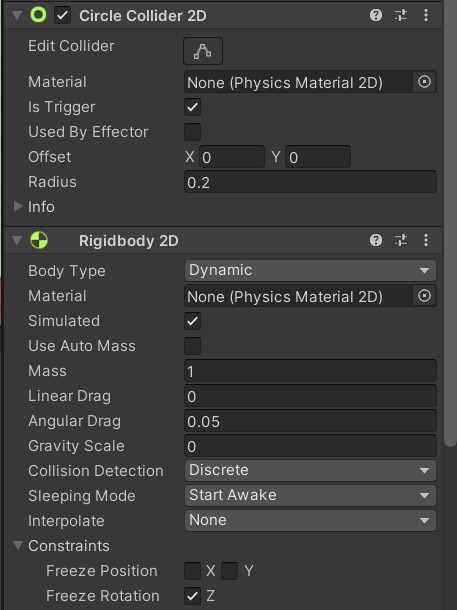
* + 1. Lalu tambahkan satu *item* sebagai peluru pada *folder Enemies* lalu *folder*

4 pilih Cannonball 1 dan ubah namanya menjadi *cannon*.



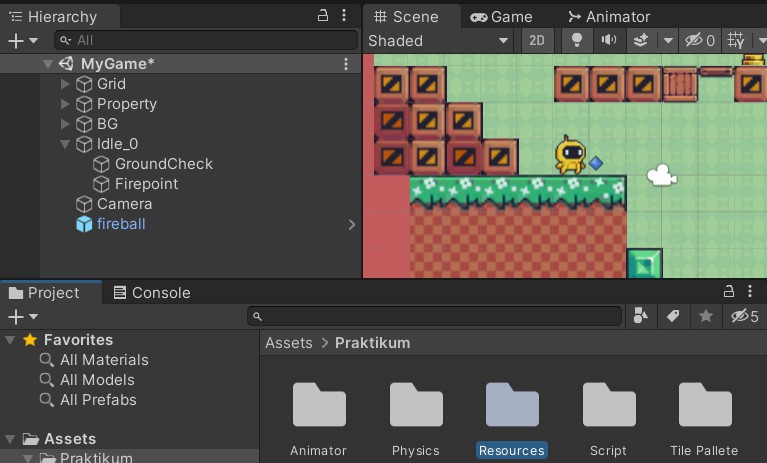
Gambar 10.5 Menambahkan Peluru

* + 1. Kemudian tambahkan komponen *Circle Collider* 2D dan *Rigidbody* 2D serta atur seperti berikut.



Gambar 10.6 Menambahkan Komponen

* + 1. Buat *folder* baru *Resources* pada *folder project*, kemudian *drag and drop fireball* ke dalam *folder Resources* dan hapus *fireball* pada *Hierarchy*.



Gambar 10.7 Membuat Folder Resources

* + 1. Pada *script Player* tambahkan *script* berikut.

public Animator animator;

// Deklarasi objek bullet dan firePoint serta variabel direction

public GameObject bullet; public Transform firePoint; float direction;

void Update ()

{

horizontalValue = Input.GetAxisRaw("Horizontal");

if (Input.GetKeyDown(KeyCode.C))

{

StartCoroutine(Attack());

}

if (Input.GetButtonDown("Jump"))

{

animator.SetBool("Jumping", true); jump = true;

}

else if (Input.GetButtonUp("Jump")) jump = false;

}

// Letakkan setelah fungsi FixedUpdate IEnumerator Attack()

{

animator.SetTrigger("Attack");

yield return new WaitForSeconds(0.25f);

GameObject cannon = Instantiate(bullet, firePoint.position, Quaternion.identity);

cannon.GetComponent<Rigidbody2D>().velocity = new Vector2(direction \* 10f, 0);

Destroy(cannon, 2f);

}

void Move(float dir, bool jumpflag)

{

if(isGrounded && jumpflag)

{

isGrounded = false; jumpflag = false;

rb.AddForce(new Vector2(0f, jumpPower));

}

#region bergerak kanan kiri float xVal = dir \* speed \* 100 \*

Time.fixedDeltaTime;

Vector2 targetVelocity = new Vector2(xVal, rb.velocity.y);

rb.velocity = targetVelocity;

if (facingRight && dir < 0)

{

transform.localScale = new Vector3(-4, 4, 4); facingRight = false;

direction = -1f;

}

else if (!facingRight && dir > 0)

{

transform.localScale = new Vector3(4, 4, 4); facingRight = true;

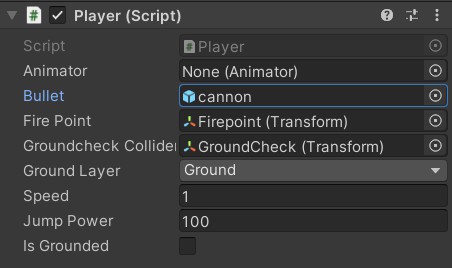
direction = 1f;

}

#endregion

}

* + 1. Pada *tab Inspector Player*, ubah *Bullet* dengan objek yang akan ditembakan sedangkan *Fire Point* sebagai titik tembak pertama.



Gambar 10.8 Mengatur Player

* + 1. Buat *script Attack* pada *folder Script* dan tambahkan *script* berikut.

using System.Collections;

using System.Collections.Generic; using UnityEngine;

public class Attack : MonoBehaviour

{

private void OnTriggerEnter2D(Collider2D collision)

{

if (collision.gameObject.CompareTag("Enemy"))

{

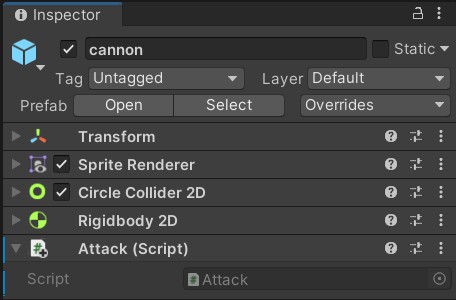
Destroy(gameObject); Destroy(collision.gameObject);

}

}

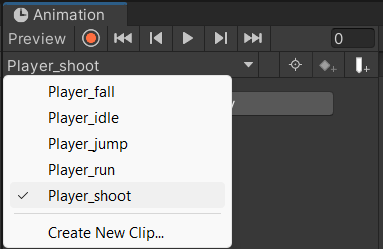
}

* + 1. Pada *folder Resources* tambahkan *script Attack* ke dalam *Prefab fireball*, dengan cara klik *fireball* kemudian pada *tab Inspector* arahkan *script Attack* ke dalam *Inspector*.



Gambar 10.9 Menambahkan Script Attack

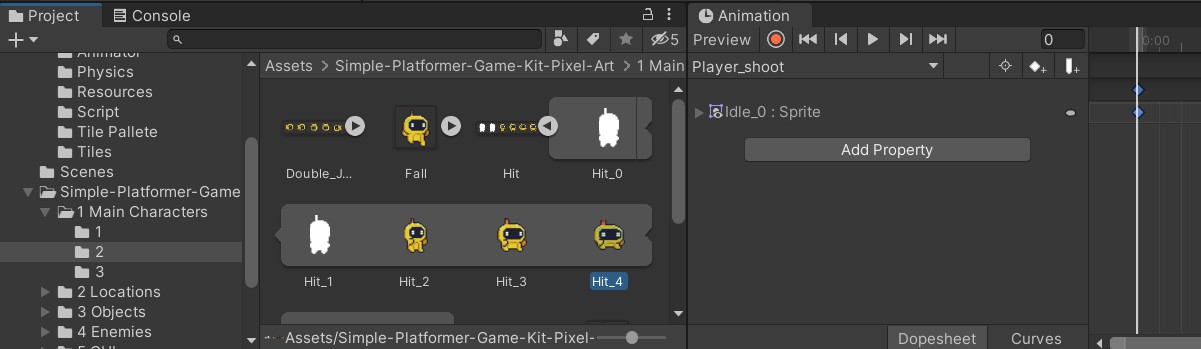
* + 1. Tambahkan animasi ketika menembak, klik *Player* dan buat *clip* baru dengan nama Player\_shoot.



Gambar 10.10 Menambahkan Clip Baru

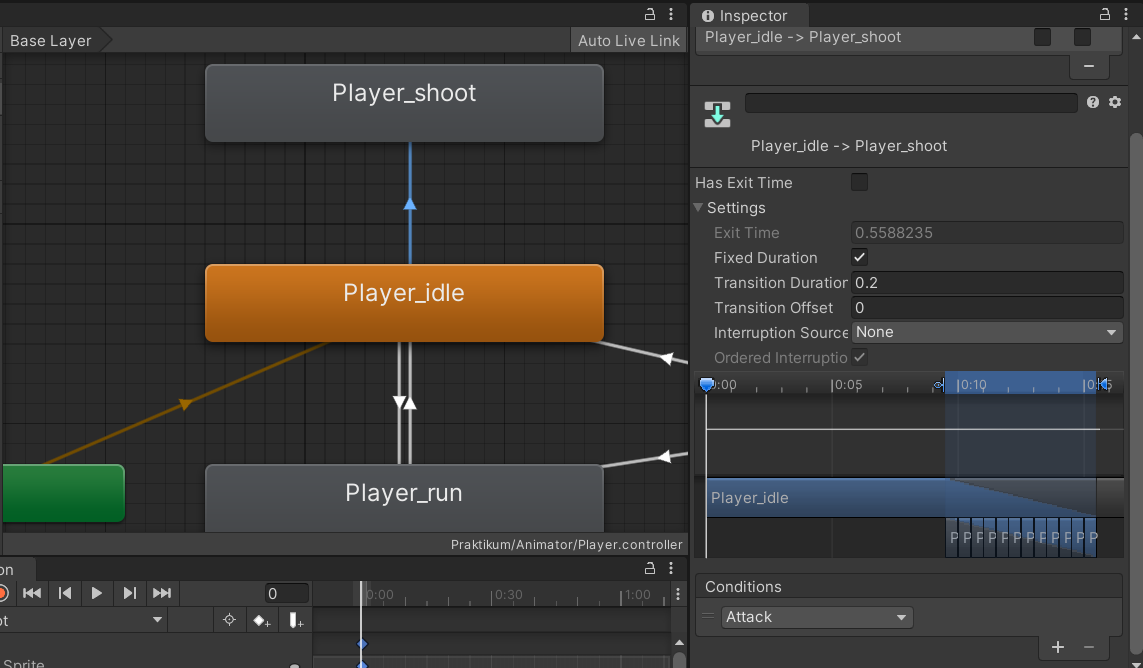
* + 1. Lalu *drag and drop* animasi yang ingin digunakan ke dalam *timeline*

animasi.



Gambar 10.11 Menambahkan Animasi

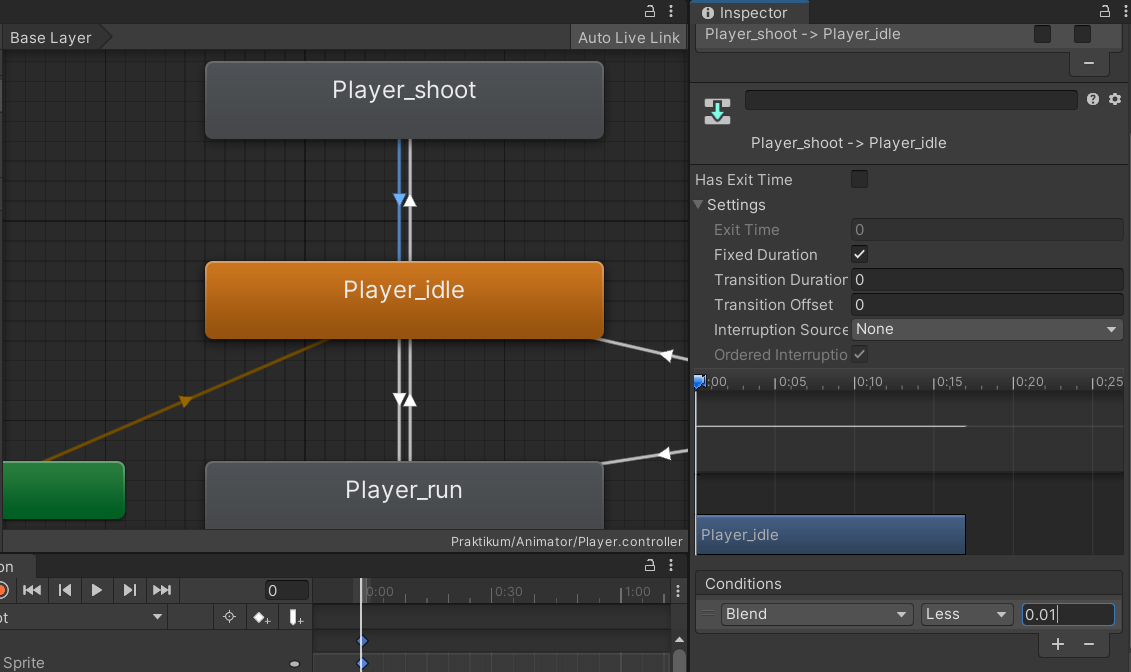
* + 1. Kemudian buat transisi dari Player\_idle ke Player\_shoot, buat kondisi menjadi *Attack* dan beri durasi transisi 0.2 serta atur seperti berikut.



Gambar 10.12 Transisi Idle Ke Shoot

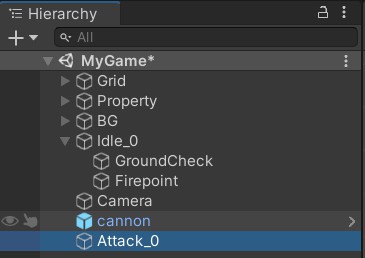
* + 1. Selanjutnya buat transisi dari Player\_shoot ke Player\_idle, buat kondisi

*Blend* dan *Less* dengan *value* 0.01 serta atur seperti berikut.



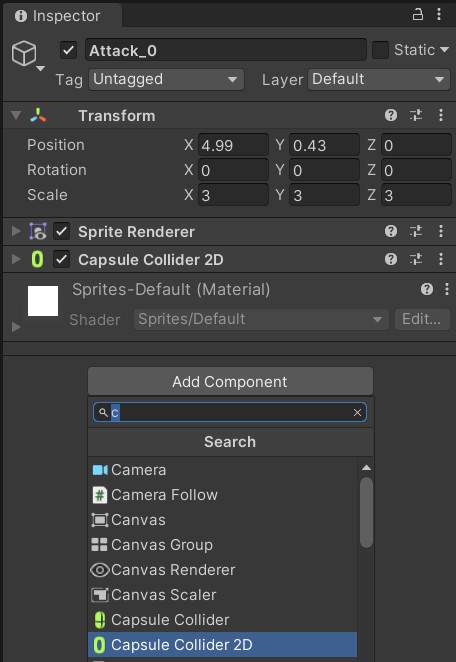
Gambar 10.13 Transisi Shoot Ke Idle

* + 1. Tambahkan *Enemy* yang ada pada *folder Enemies* lalu *folder* 5, pilih Attack\_0.



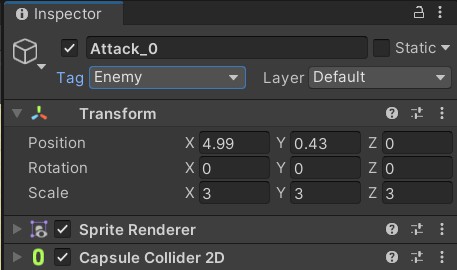
Gambar 10.14 Menambahkan Enemy

* + 1. Klik pada Attack\_0, lalu pada *tab Inspector* tambahkan *Capsule Collider* 2D.



Gambar 10.15 Menambahkan Capsule Collider 2D

* + 1. Pilih *Tag* dan tambahkan *tag* baru dengan nama *Enemy*, kemudian tambahkan *tag Enemy* pada Attack\_0.



Gambar 10.16 Mengubah Tag

* + 1. Coba jalankan *game* dan tembak dengan menekan tombol C untuk menghancurkan musuh.

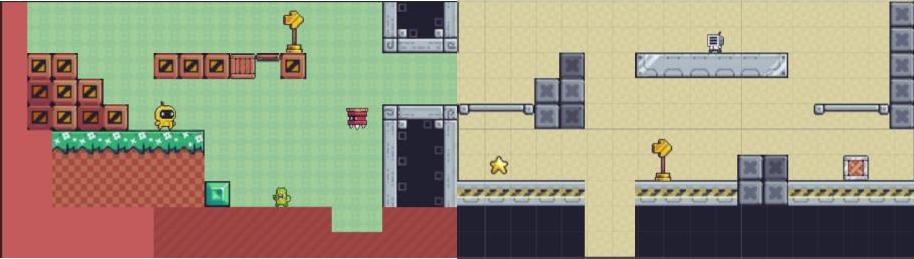


Gambar 10.17 Menjalankan Game

# Membuat Enemy Behavior

* 1. Tambahkan *sprite Run* ke dalam hirarki dari *folder Enemies* pada

*folder* 1 dan 2.



Gambar 10.18 Menambahkan Sprite

* 1. Pada kedua *enemy* tersebut tambahkan komponen *Capsule Collider*

2D dan *Rigidbody* 2D. Lalu atur pada *enemy* 1 sebagai berikut.



Gambar 10.19 Pengaturan Enemy 1

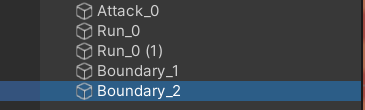
* 1. Kemudian untuk *enemy* 2 sebagai berikut.



Gambar 10.20 Pengaturan Enemy 2

* 1. Buat 2 *empty object* pada hirarki dengan nama *Boundary*\_1 dan

*Boundary*\_2.



Gambar 10.21 Membuat Empty Object

* 1. Tambahkan *Box Collider* 2D pada tiap *Boundary*, centang *Is Trigger* dan sesuaikan *size* dan *offset* sebagai batas pergerakan *enemy* serta rotasi sumbu Y pada *enemy* sebesar 180.



Gambar 10.22 Menambahkan Box Collider

* 1. Lalu buat sebuah *script* baru dengan nama Enemy\_Behavior dan tambahkan *script* berikut.

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)

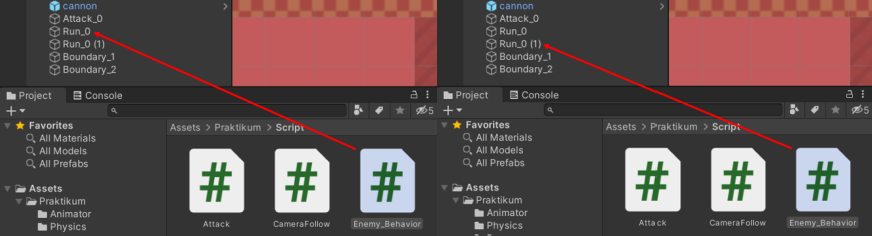
{

transform.localScale = new Vector2(- transform.localScale.x, transform.localScale.y);

}

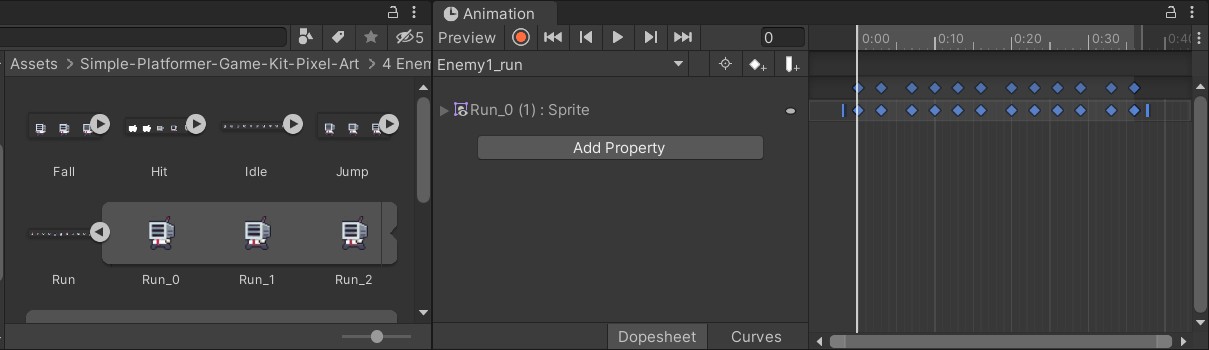
}

* 1. Kemudian *drag and drop script* pada *enemy* 1 dan *enemy* 2.



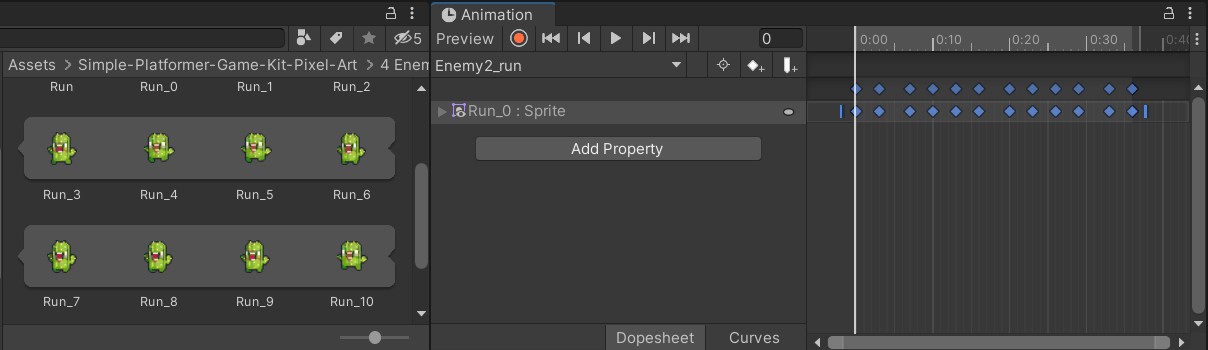
Gambar 10.23 Menambahkan Script

* 1. Lalu tambahkan animasi berjalan pada *enemy* 1 dengan membuat *clip* baru dan tambahkan animasi *sprite* berjalan ke dalam *timeline* serta atur waktunya.



Gambar 10.24 Menambah Animasi Enemy 1

* 1. Selanjutnya tambahkan animasi berjalan pada *enemy* 2 dengan membuat *clip* baru dan tambahkan animasi *sprite* berjalan ke dalam *timeline* serta atur waktunya.



Gambar 10.25 Menambah Animasi Enemy 2

* 1. Kemudian jalankan *game* dan pastikan *enemy* 1 dan 2 berjalan.



Gambar 10.26 Menjalankan Game

# Membuat Enemy AI

* 1. Cari sebuah *sprite* bernama 2\_0 dan tambahkan ke dalam hirarki.



Gambar 10.27 Menambahkan Sprite

* 1. Buat *script* Enemy\_AI dan tambahkan *script* berikut.

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()

{

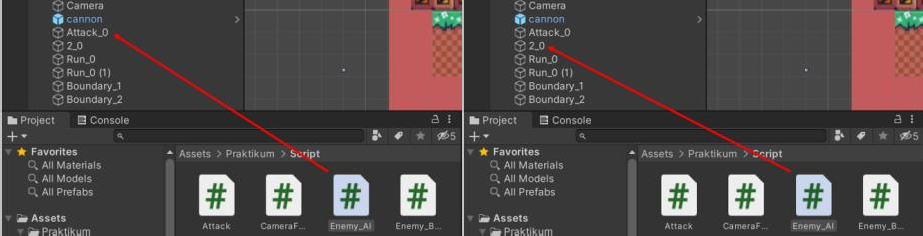
Gizmos.color = Color.red; Gizmos.DrawWireSphere(transform.position,

lineOfSite);

}

}

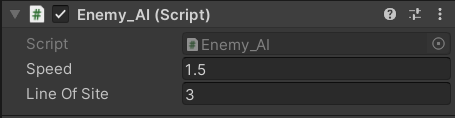
* 1. Kemudian *drag and drop script* Enemy\_AI ke *enemy* 3 dan 4.



Gambar 10.28 Menambahkan Script

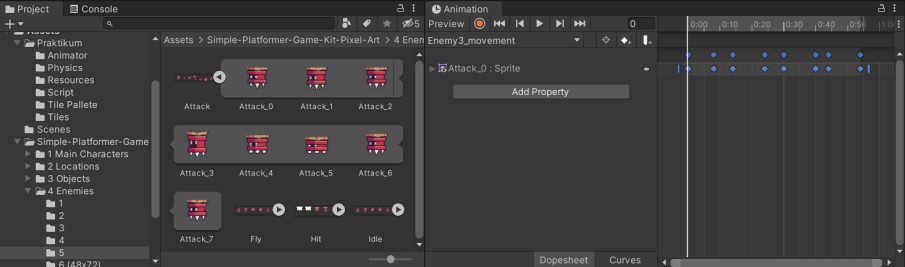
* 1. Lalu atur *Speed* dan *Line of Site* untuk menentukan jarak terhadap

*player* dan kecepatan.



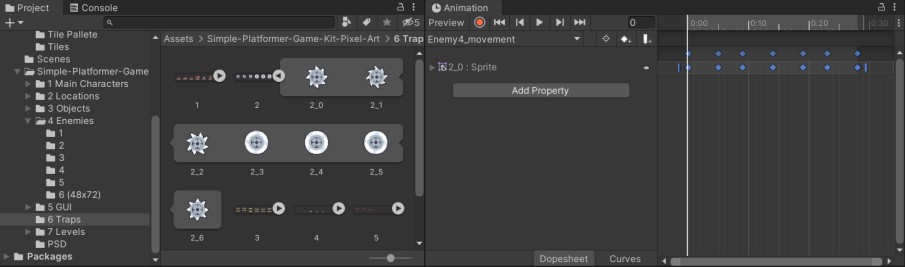
Gambar 10.29 Mengatur Kecepatan Dan Jarak

* 1. Lalu tambahkan animasi bergerak pada *enemy* 3 dengan membuat *clip* baru dan tambahkan animasi *sprite* bergerak ke dalam *timeline* serta atur waktunya.



Gambar 10.30 Menambah Animasi Enemy 3

* 1. Selanjutnya tambahkan animasi bergerak pada *enemy* 4 dengan membuat *clip* baru dan tambahkan animasi *sprite* bergerak ke dalam *timeline* serta atur waktunya.



Gambar 10.31 Menambah Animasi Enemy 4

# Respawn

* 1. Tambahkan *script* berikut ke dalam *script Player* untuk mengembalikan *player* ke tempat awal ketika jatuh atau nyawa kurang dari 0.

public int nyawa;

[SerializeField] Vector3 respawn\_loc; public bool play\_again;

private void Awake()

{

rb = GetComponent<Rigidbody2D>(); animator = GetComponent<Animator>();

// Menentukan tempant respawn respawn\_loc = transform.position;

}

void Update ()

{

horizontalValue = Input.GetAxisRaw("Horizontal");

if (Input.GetKeyDown(KeyCode.C))

{

StartCoroutine(Attack());

}

if (Input.GetButtonDown("Jump"))

{

animator.SetBool("Jumping", true); jump = true;

}

else if (Input.GetButtonUp("Jump")) jump = false;

if (nyawa < 0)

{

playagain();

}

if (transform.position.y < -10)

{

play\_again = true; playagain();

}

}

void playagain()

{

if (play\_again == true)

{

nyawa = 3;

transform.position = respawn\_loc; play\_again = false;

}

}

* 1. Kemudian buat *script* baru dengan nama Enemy\_Attack dan tambahkan *script* berikut.

using System.Collections;

using System.Collections.Generic; using UnityEngine;

public class Enemy\_Attack : 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;

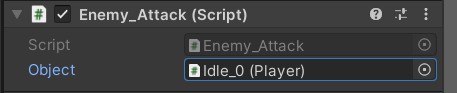
}

}

}

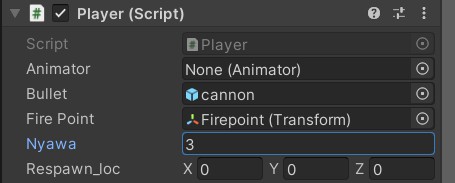
}

* 1. Lalu *drag and drop script* Enemy\_Attack ke semua musuh dan arahkan *object* pada Idle\_0.



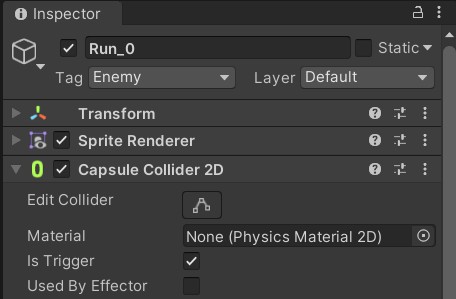
Gambar 10.32 Menambahkan Script

* 1. Pergi ke *tap Inspector game object Player* dan ubah nilai Nyawa menjadi 3 pada *Player* (*Script*).



Gambar 10.33 Mengubah Nilai Nyawa

* 1. Kemudian pastikan bahwa setiap musuh menggunakan *Tag Enemy* dan memiliki komponen *Capsule Collider* 2D dengan *Is Trigger check*.



Gambar 10.34 Tab Inspector Enemy

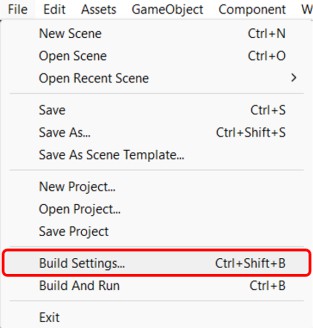
* 1. Lalu coba jalankan *game*, maka ketika *player* menyentuh musuh, nyawa akan berkurang 1 dan jika nyawa kurang dari 0 atau *player* jatuh maka akan *respawn* ke titik awal.



Gambar 10.35 Menjalankan Game

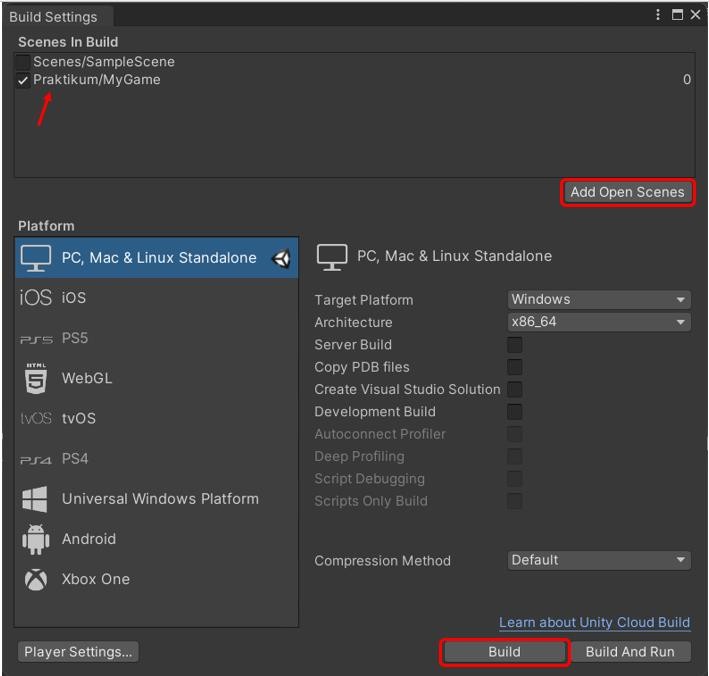
# Render Game

* 1. Lalu untuk merender *game*, pilih *Build Settings* pada menu *File*.



Gambar 10.36 Menu File

* 1. Selanjutnya *Add Open Scenes* dan pilih *Scene* pada *project* yang ingin dirender, lalu pilih *Build* dan simpan kedalam *folder* yang diinginkan.



Gambar 10.37 Build Settings

# Link Github Pengumpulan

https://github.com/reydjav/1918027\_PRAK\_ANIGAME.git

# Kuis : Melengkapi Source Code

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))

{

Health enemyHealth = hit.transform.GetComponent<Health>();

if (enemyHealth != null)

{

enemyHealth.TakeDamage(attackDamage);

}

}

}

}

Penjelasan:

Mula-mula akan diperiksa apakah objek yang terkena serangan memiliki komponen *Health*. Jika objek tersebut memiliki komponen *Health* dengan nilai tidak *null*, maka fungsi *TakeDamage* dipanggil pada objek tersebut untuk mengurangi *Health* dengan jumlah *attackDamage*. Sehingga dapat dipastikan hanya objek dengan komponen *Health* yang akan menerima serangan.