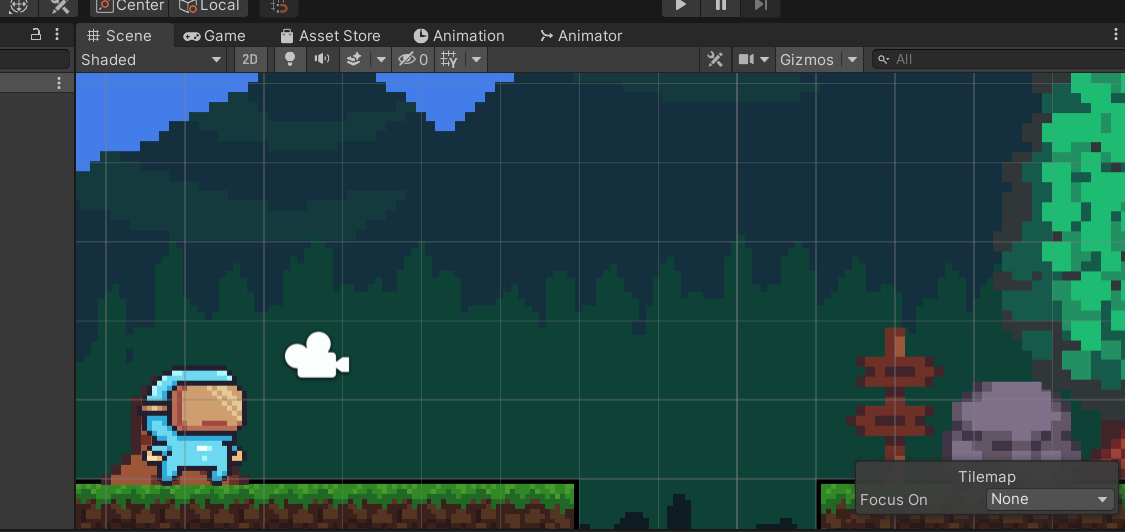
# 10 RESPAWN AND AI ENEMY ATTACK

|  |  |  |
| --- | --- | --- |
| **NIM** | : | 2118041 |
| **Nama** | : | Andi Naufal Yutaka |
| **Kelas** | : | D |
| **Asisten Lab** | : | Mohammad Akbar Ilham (2118091) |

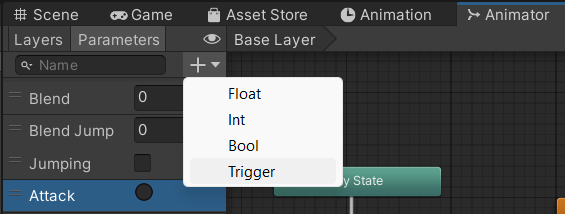
## Tugas 10: Respawn and AI Enemy Attack

1. **Attack, Respawn, and Enemy AI**
2. Buka tugas sebelumnya yaitu tugas 9.



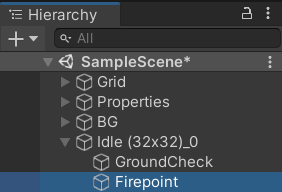
### 10.1 Membuka tugas9

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



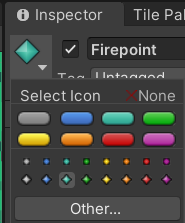
### 10.2 Menambahkan parameter Trigger dengan nama *Attack*

1. Membuat Layer baru pada *Hierarchy* *Player*-nya dan Rename menjadi *Firepoint*.



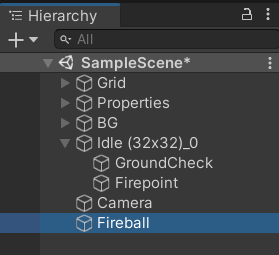
### 10.3 Menambahkan layer

1. Membuat Firepoint pada Hierarchy, lalu ubah *icon* menjadi titik dan posisikan di depan *player*.



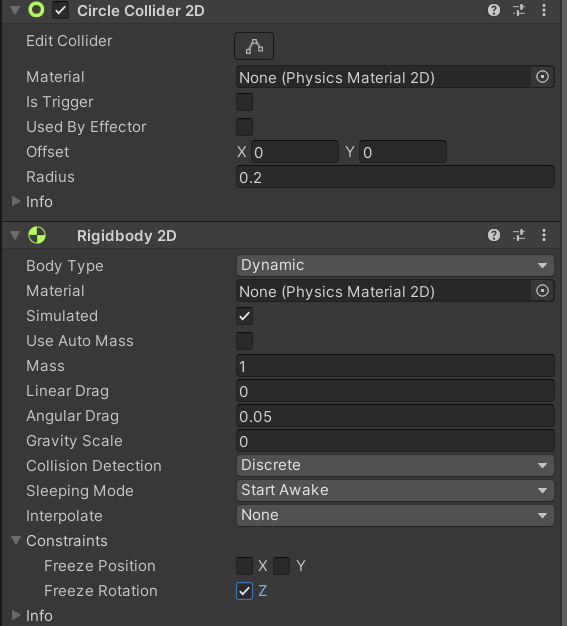
### 10.4 Membuat model *Firepoint*

1. Tambahkan *Fireball* menggunakan *Asset* anda.



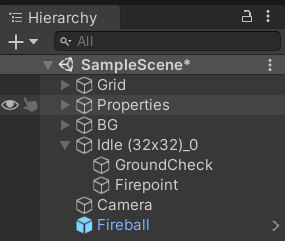
### 10.5 Menambahkan *Fireball*

1. Menambahkan *Component* *Circle Collider 2d*, dan *Riggidbody 2D* pada *Fireball*.



### 10.6 Menambahkan *Component*

1. Buat Folder baru *Resources* di menu Project, Lalu *drag and drop fireball* kedalam *folder Resources* yang tadi sudah dibuat, dan hapus *fireball* pada *Hierarchy*.

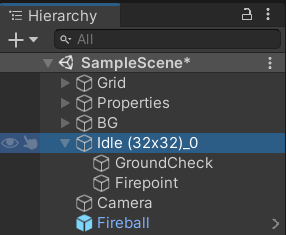


### 10.7 *Fireball* pada *Hierarchy*

1. Tambahkan *Source* *code* dibawah ini kedalam *Script* *Player*.

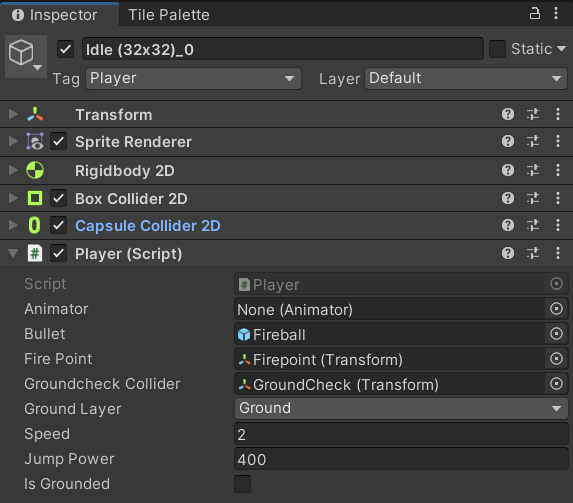
|  |
| --- |
| IEnumerator Attack()  {    animator.SetTrigger("Attack");  yield return new WaitForSeconds(0.25f);    float direction = 1f;    GameObject fireball = Instantiate(bullet, firePoint.position, Quaternion.identity);  fireball.GetComponent<Rigidbody2D>().velocity = new Vector2(direction \* 10f, 0);    Destroy(fireball, 2f);  }  if (Input.GetKeyDown(KeyCode.C))  {  StartCoroutine(Attack());  } |

1. Lalu klik *Player* sebelum masuk ke *Inspector*



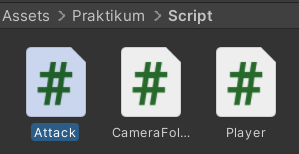
### 10.8 *Player Hierarchy*

1. Pada Inspector Player, Ubah seperti ini, Dimana Bullet berisi object yang akan ditembak sedangkan fire point adalah titik tembak pertama.



### 10.9 *Inspector* *Player*

1. Buat *Script* baru dengan nama *Attack* pada *folder Script*.



### 10.10 *Attack script*

Lalu masukan *source code* berikut

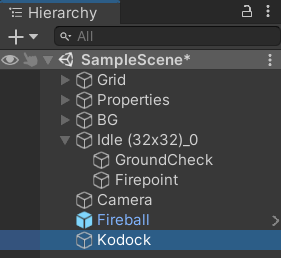
|  |
| --- |
| using System.Collections;  using System.Collections.Generic;  using UnityEngine;  public class Attack : MonoBehaviour  {  private void OnTriggerEnter2D(Collider2D collision)  {  if (collision.gameObject.CompareTag("Enemy"))  {  Destroy(gameObject);  Destroy(collision.gameObject);  }  }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

1. Tambahkan *Script Attack* di *Prefab fireball* dengan cara di *Drag n Drop*.



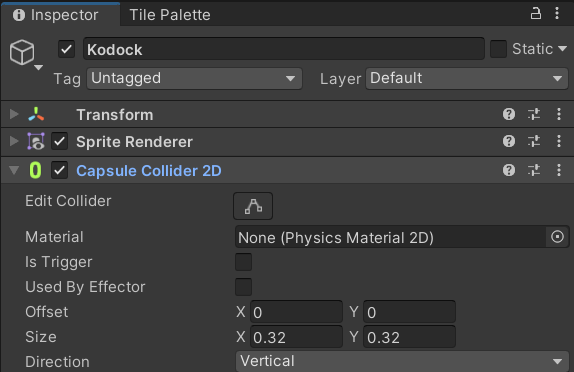
### 10.11 *Script on fireball*

1. Tambahkan *Enemy* lalu *drag and drop* kedalam *Hierarchy*.



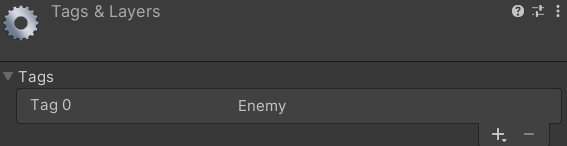
### 10.12 Menambahkan *Enemy*

1. Tambahkan *component Collider 2D* pada *Enemy.*



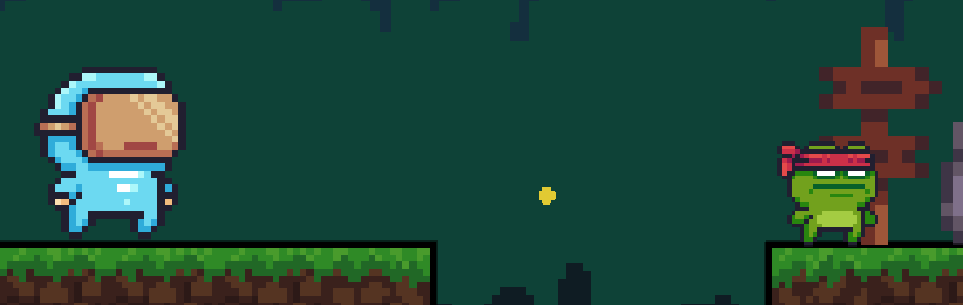
### 10.13 *Component Collider 2D*

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



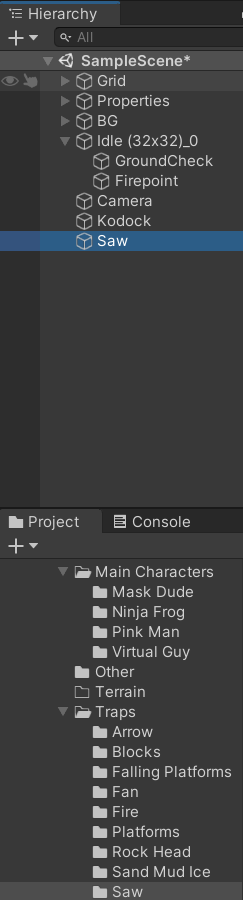
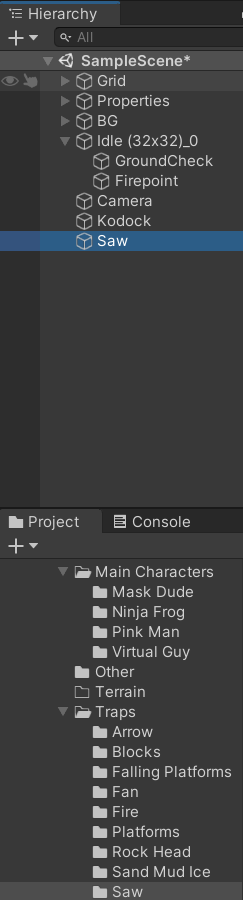
### 10.14 *Enemy Tag*

1. Lalu *Test* apakah saat peluru mengenai *Enemy* dia akan menghilang, kalau menghilang berarti sudah benar.



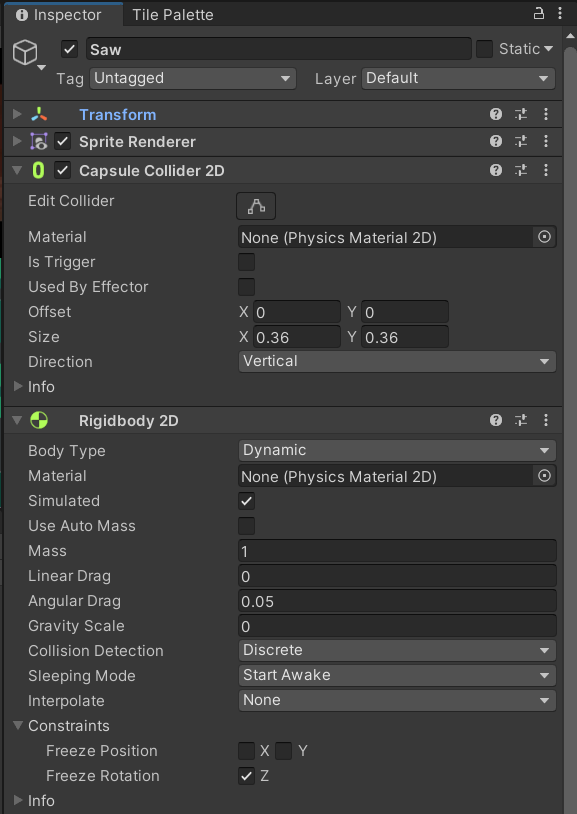
### 10.15 *Testing*

1. Lalu dilanjut dengan membuat *Enemy Behavior* NPC, dengan cara mencari *Enemy* di *Asset* yang anda punya dan pada *inspector* atur *scale* sesuai selera*.*



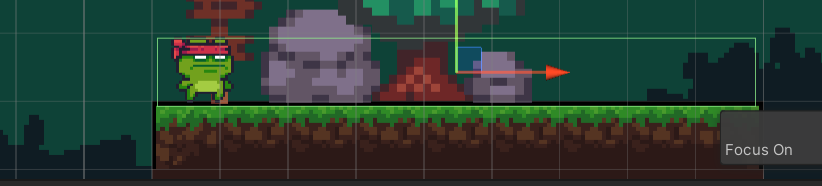
### 10.16 *Enemy NPC*

1. Tambahkan sebuah komponen bernama Capsule Colider 2D dan Rigidbody 2D dalam inspector NPC yang tadi sudah di masukan, lalu pada Rigidbody 2d ubah *Body Type* menjadi *Kinematic*.



### 10.17 Menambahkan *Components*

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

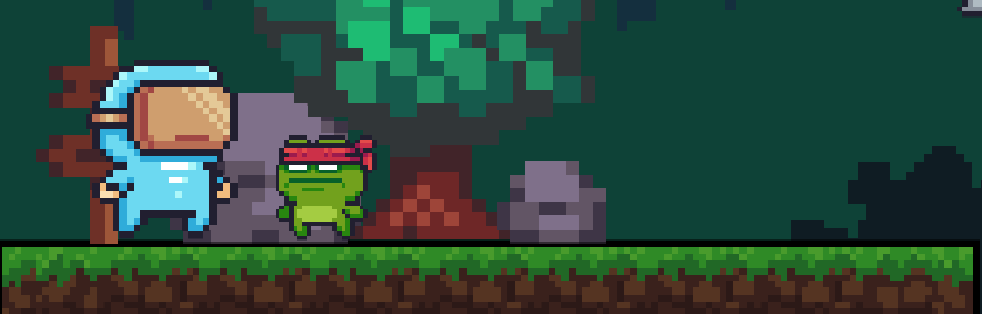


### 10.18 Menambahkan *Boundary*

1. Tambahkan Script dengan nama “Enemy\_Behavior” lalu drag dan drop ke “Kodock”.

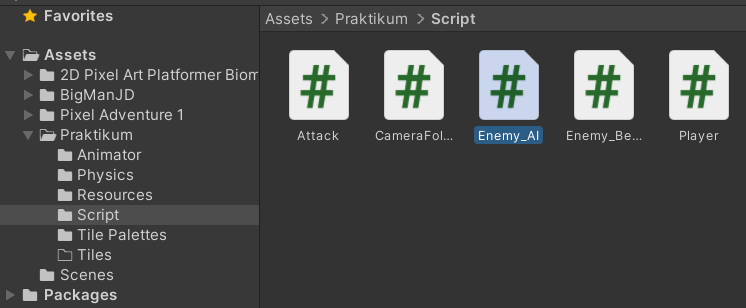
|  |
| --- |
| using System.Collections;  using System.Collections.Generic;  using UnityEngine;  public class Enemy\_Behavior : MonoBehaviour  {  [SerializeField] float moveSpeed = 1f;  Rigidbody2D rb;  void Start()  {  rb = GetComponent<Rigidbody2D>();  }  void Update()  {  if (isFacingRight())  {  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. Jika dijalankan maka *Kodock* akan berjalan menyesuaikan dengan ukuran Boundary.



### 10.19 Animasi *Kodock* Berjalan

1. Kemudian lanjut dengan pembuatan *Enemy* *AI* yang diterapkan di “Saw” yang tadi sudah di tambahkan ke *Hierarchy,* tambahkan *Script* *Enemy*\_*AI* di *folder* *Script*.



### 10.20 *Create New Script*

1. 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;  void Start()  {  player = GameObject.FindGameObjectWithTag("Player").transform;  initialPosition = GetComponent<Transform>().position;  }  void Update()  {  float distanceToPlayer = Vector2.Distance(player.position, transform.position);  if (distanceToPlayer < lineOfSite)  {  transform.position = Vector2.MoveTowards(this.transform.position, player.position, speed \* Time.deltaTime);  }  else  {  transform.position = Vector2.MoveTowards(transform.position, initialPosition, speed \* Time.deltaTime);  }  }    private void OnDrawGizmosSelected()  {  Gizmos.color = Color.red;  Gizmos.DrawWireSphere(transform.position, lineOfSite);  }  } |

1. Pada *Inspector* *Enemy\_Ai*, Atur *Speed* juga *Line of Site* untuk menentukan jarak dan kecepatan pada enemy saat mengejar *Player*.



### 10.21 Mengatur *Saw*

1. Coba *Running Game*, jika Saw mengikuti pergerakan *Player* berarti sudah benar.



### 10.22 *Testing Game*

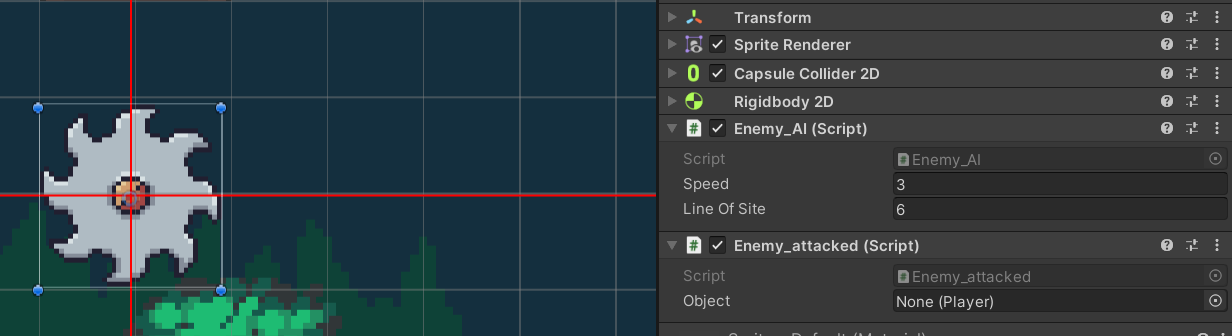
1. Lalu dilanjut dengan pembuatan *Respawn* yang berguna jika *player* mati, maka akan kembali ke titik *respawn* yang sudah di*set* dengan cara sedikit mengubah di script (Player.cs) menjadi seperti berikut.

|  |
| --- |
| using System.Collections;  using System.Collections.Generic;  using UnityEngine;  public class Player : MonoBehaviour  {  public Animator animator;  public GameObject bullet;  public Transform firePoint;  public int nyawa;  [SerializeField] Vector3 respawn\_loc;  public bool play\_again;  Rigidbody2D rb;  [SerializeField] Transform groundcheckCollider;  [SerializeField] LayerMask groundLayer;  const float groundCheckRadius = 0.2f;  [SerializeField] float speed = 4;  [SerializeField] float jumpPower = 100;  float horizontalValue;  [SerializeField] bool isGrounded;  bool facingRight = true;  bool jump;  private void Awake()  {  rb = GetComponent<Rigidbody2D>();  animator = GetComponent<Animator>();  respawn\_loc = transform.position;  }  void playagain()  {  if (play\_again == true)  {  nyawa = 3;  transform.position = respawn\_loc;  play\_again = false;  }  }  void Update()  {  horizontalValue = Input.GetAxisRaw("Horizontal");  if (Input.GetButtonDown("Jump")){  animator.SetBool("Jumping", true);  jump = true;  }  else if (Input.GetButtonUp("Jump")){  jump = false;  }  if (Input.GetKeyDown(KeyCode.J))  {  StartCoroutine(Attack());  }  if (nyawa < 0);  {  playagain();  }  if (transform.position.y < -10)  {  play\_again = true;  playagain();  }  }  void FixedUpdate()  {  GroundCheck();  Move(horizontalValue, jump);    animator.SetFloat("Blend", Mathf.Abs(rb.velocity.x));  animator.SetFloat("Blend Jump", rb.velocity.y);  }  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);  }  void GroundCheck()  {  isGrounded = false;  Collider2D[] colliders = Physics2D.OverlapCircleAll(groundcheckCollider.position, groundCheckRadius, groundLayer);  if (colliders.Length > 0){  isGrounded = true;  }  animator.SetBool("Jumping", !isGrounded);  }  void Move(float dir, bool jumpflag)  {  if (isGrounded && jumpflag)  {  isGrounded = false;  jump = false; // Reset the jump flag  rb.AddForce(new Vector2(0f, jumpPower));  }  #region gerak kanan kiri  float xVal = dir \* speed \* 100 \* Time.fixedDeltaTime;  Vector2 targetVelocity = new Vector2(xVal, rb.velocity.y);  rb.velocity = targetVelocity;  if (facingRight && dir < 0)  {  // ukuran player  transform.localScale = new Vector3(-6.3f, 6.3f, 6.3f);  facingRight = false;  }  else if (!facingRight && dir > 0)  {  // ukuran player  transform.localScale = new Vector3(6.3f, 6.3f, 6.3f);  facingRight = true;  }  #endregion  }  } |

1. Tambahkan file script dengan nama “Enemy\_Attacked.cs” dan isikan source code dibawah ini.

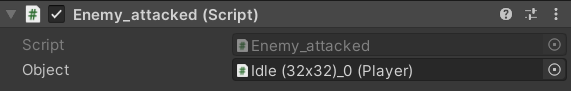
|  |
| --- |
| using System.Collections;  using System.Collections.Generic;  using UnityEngine;  public class Enemy\_attacked : MonoBehaviour  {  [SerializeField] private Player Object;  void Start()  {  if (Object == null)  {  Object = GameObject.FindWithTag("Player").GetComponent<Player>();  }  }  void OnTriggerEnter2D(Collider2D other)  {  if (other.CompareTag("Player"))  {  Object.nyawa--;  if (Object.nyawa < 0)  {  Object.play\_again = true;  }  }  }  } |

1. Lalu tambahkan *script Enemy\_attacked* kedalam Object Saw.



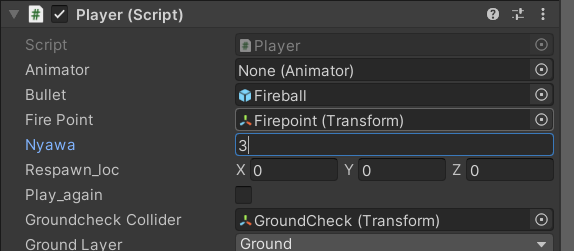
### 10.23 *Script Enemy\_Attacked*

1. Lalu ubah *object*nya dengan memilih *Player*-nya.



### 10.24 *Change the Object*

1. Lalu klik *player* dan menuju ke *Inspector*, lalu ubah Nyawa menjadi 3 seperti pada *Script*(*Player*)*.*



### 10.25 Ubah nyawa

1. Lalu tes, jika *Player* mengenai *Saw* sebanyak 3 kali maka nyawa akan berkurang 1 dan jika nyawa kurang dari 0 maka *GameOver*.



### 10.26 *Running Game*

1. **Kuis Pertemuan 10**

Lengkapi Source code dibawah ini :

|  |
| --- |
| 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  }  }  } |

Source code yang sudah dilengkapi yang kurang

|  |
| --- |
| 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))  {  EnemyHealth enemyHealth = hit.transform.GetComponent<EnemyHealth>();  if (enemyHealth != null)  {  enemyHealth.TakeDamage(attackDamage);  }  }  }  } |

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);  }  } |

Dan ini merupakan source code yang sudah diperbaiki

|  |
| --- |
| void HandleJumpInput()  {  if (Input.GetKeyDown(KeyCode.Space))  {  animator.SetBool("isJumping", true); // Set to true when jumping starts  rb.AddForce(Vector2.up \* jumpForce, ForceMode2D.Impulse);  }  else if (Input.GetKey(KeyCode.Space))  {  animator.SetBool("isJumping", true); // Set to true while the space key is held down  }  else  {  animator.SetBool("isJumping", false); // Set to false when the space key is released  }  }  void HandleMovementInput()  {  float move = Input.GetAxis("Horizontal");  if (move == 0)  {  animator.SetBool("isIdle", true);  animator.SetBool("isWalking", false);  }  else  {  animator.SetBool("isIdle", false);  animator.SetBool("isWalking", true);    transform.Translate(Vector3.right \* move \* Time.deltaTime); // Corrected to Vector3.right  }  if (move < 0)  {  transform.localScale = new Vector3(-1, 1, 1); // Correct scaling for left direction  }  else if (move > 0)  {  transform.localScale = new Vector3(1, 1, 1); // Correct scaling for right direction  }  } |

Analisa :

Pada bagian pertama, fungsi *PerformMeleeAttack* yang ada pada *PlayerAttack* script mengalami beberapa kekurangan. Kesalahan utamanya adalah tidak adanya kode untuk mengurangi *health* musuh setelah terkena serangan. Hal ini disebabkan oleh tidak adanya pemeriksaan komponen musuh (*EnemyHealth*) yang mengelola health mereka. Selain itu, dalam *script* *HandleJumpInput*, kesalahan utama adalah tidak adanya nilai *boolean* pada *animator*.*SetBool*("*isJumping*", );. Kondisi ini menyebabkan *error* kompilasi karena *SetBool* membutuhkan dua parameter: nama parameter animasi dan nilai *boolean* yang akan diatur. Pada *script* *HandleMovementInput*, terdapat beberapa kesalahan *logika*. Pertama, kondisi *if (move != 1)* yang seharusnya memeriksa apakah nilai *move* tidak sama dengan nol (0) untuk mendeteksi gerakan. Kedua, arah *transform.Translate(Vector3.left \* move \* Time.deltaTime);* yang tidak sesuai dengan arah gerakan yang benar.

Setelah perbaikan, pada fungsi *PerformMeleeAttack*, telah ditambahkan logika untuk memeriksa apakah objek yang terkena adalah musuh dan mengurangi *health* mereka menggunakan komponen *EnemyHealth*. *Script HandleJumpInput* telah diperbaiki dengan menambahkan nilai *boolean* yang sesuai (*true atau false*) pada *animator*.*SetBool*, sehingga sekarang dapat berfungsi dengan benar. Pada *script* *HandleMovementInput*, *kondisi if (move != 1)* telah diperbaiki menjadi *if (move == 0)* untuk mendeteksi jika tidak ada pergerakan. Selain itu, arah gerakan telah diperbaiki dengan menggunakan *Vector3.right* untuk mencerminkan arah yang benar berdasarkan nilai *move*. Logika animasi juga telah diperbaiki dengan mengatur *animator.SetBool("isWalking", true)* ketika ada gerakan dan *animator.SetBool("isIdle", false)* ketika tidak ada. Terakhir, pengaturan *transform.localScale* telah disesuaikan untuk mencerminkan arah karakter yang menghadap kiri atau kanan dengan benar, berdasarkan nilai *move*.