TUGAS PERTEMUAN: 10

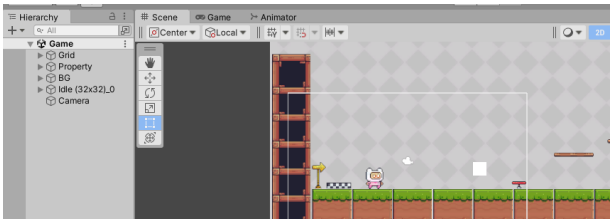
Respawn and AI Enemy Attack

NIM	:	2118101
Nama	:	Marie Pangestu
Kelas	:	C
Asisten Lab	:	Rifal Rifqi Rhomadon (2218106)

1.1 Tugas 1 : Respawn and AI Enemy Attack

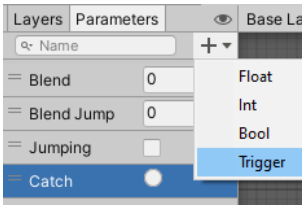
A. Mekanisme Attack

1. Buka project bab 9.



Gambar 10.1 Buka Project Bab 9

2. Lalu pada animator, buat parameter baru dengan nama ‘Catch’ bertipe data trigger.

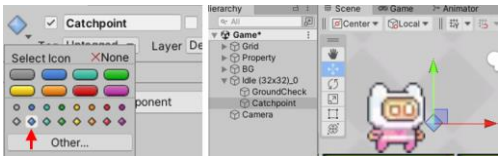


Gambar 10.2 Add Catch Trigger Parameter

3. Buat layer *Game object* baru didalam karakter, Klik kanan pilih *Create Empty* lalu Rename menjadi *Catchpoint*

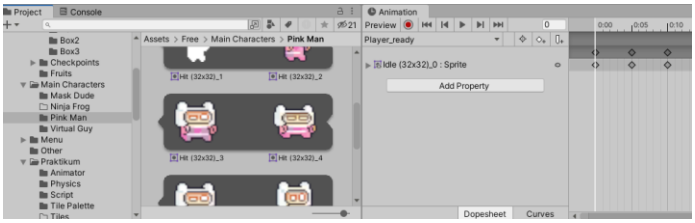
Gambar 10.3 Create Empty Catch Point

4. Pada menu *Hierarchy* klik Catchpoint untuk setting pada Inspector, Ubah *Icon* Menjadi titik, atur letak titik didepan player



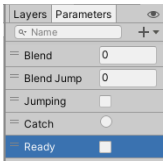
Gambar 10.4 Icon Catchpoint

5. Tekan player pada hierarchy, lalu tambahkan object untuk animasi ancang-ancang dengan nama ‘Player_ready’.



Gambar 10.5 Add Player_Ready Animation

6. Buat parameter baru dengan nama ‘Ready’ bertipe data Boolean.

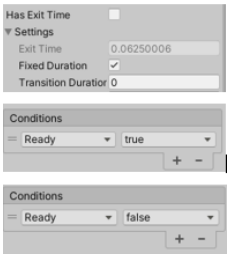


Gambar 10.6 New Ready Bool Parameter

7. Buat transisi untuk animasi Player_ready

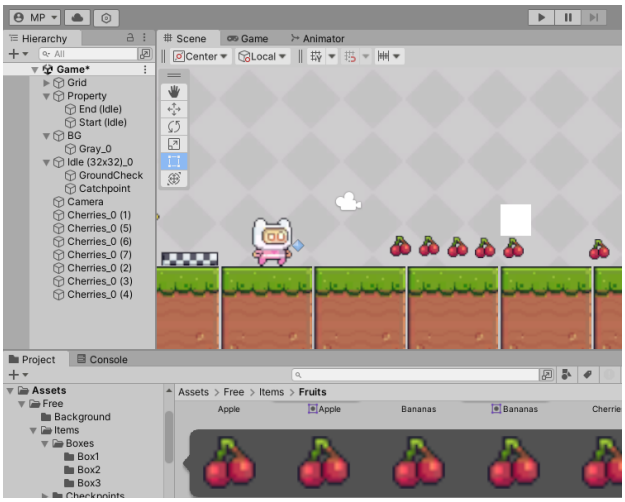
Gambar 10.7 Make Transition for Ready

8. Kondisi parameter ‘Ready’ yaitu true bertipe data Boolean transisi dari run ke ready. Dan buat transisi sebaliknya dengan kondisi parameter ‘Ready’ bernilai false. Unchecklist has exit time dan ubah nilai transition duration menjadi 0.



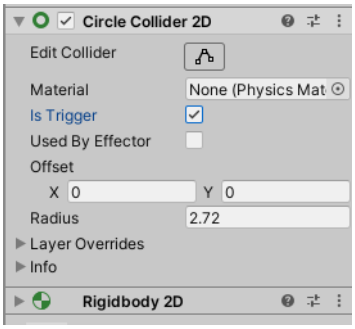
Gambar 10.8 Condition Ready Parameter on Transition

9. Kemudian tambahkan object buah yang akan ditangkap oleh karakter ke dalam hierarchy dari folder Free > Item > Fruits> drag and drop (pastikan mengubah sprite mode dan pixel per units. Ubah tag pada buah menjadi ‘Fruit’.



Gambar 10.9 Add Fruit

10. Tambahkan component pada karakter buah yaitu circle collider 2D (checklist ‘Is Trigger’) dan rigidbody 2D(gravity scale = 0)



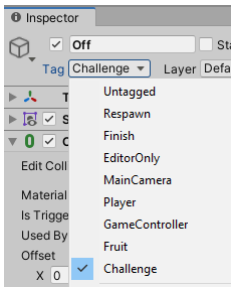
Gambar 10.10 Add Circle Collider 2D Dan Rigidbody 2D Fruit

11. Tambahkan kode berikut dibawah kode void FixedUpdate()

```
private void OnTriggerEnter2D(Collider2D other)
    {
        if (other.gameObject.CompareTag("Fruit"))
        {
            //   Mengaktifkan   trigger   'catch'   pada
   animator
            animator.SetTrigger("Catch");
            // Mengatur boolean 'Ready' menjadi true
            animator.SetBool("Ready", true);

            //  Memulai  coroutine  untuk  menghancurkan
   buah  dan  menonaktifkan  boolean  'Ready'  setelah
   animasi ancang-ancang
```

```
StartCoroutine(CatchFruit(other.gameObject));

        }
    }
    IEnumerator CatchFruit(GameObject fruit)
    {
        // Menunggu sebentar untuk memberikan waktu bagi
animasi ancang-ancang
        yield return new WaitForSeconds(0.2f);
        // Menghancurkan objek buah
        Destroy(fruit);

        // Menonaktifkan boolean 'Ready' setelah animasi
ancang-ancang
        animator.SetBool("Ready", false);

    }
```

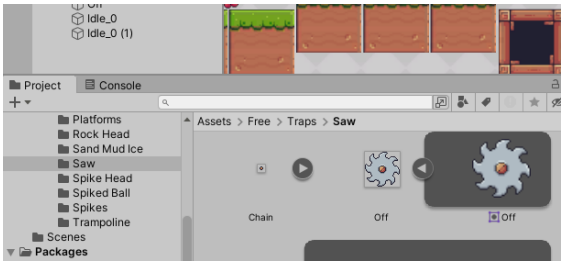
12. Buat tag ‘Challenge’ untuk memberi tag rintangan.



Gambar 10.11 Tag Challenge

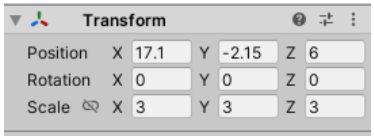
B. Enemy Behavior

1. Cari komponen saw, lalu drag and drop menuju hierarchy.



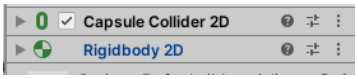
Gambar 10.12 Add Saw Challenge

2. Pada inspector saw, ubah transform.



Gambar 10.13 New Parameter Blend

3. Tambahkan komponen capsule collider 2D(checklist ‘Is Trigger’) dan rigidbody 2D(body type : Kinematic) dan sesuaikan offset serta size.

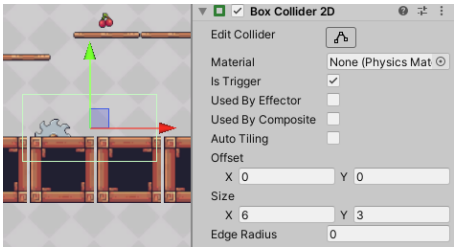


Gambar 10.14 Add Component Saw Challenge

4. Kemudian klik kanan pada hierarchy, beri nama ‘Boundary’.

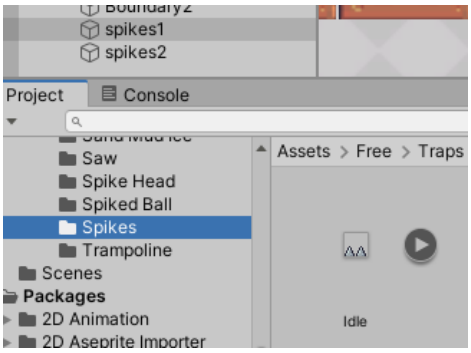
Gambar 10.15 Create Empty Boundary

5. Pada inspector boundary, tambahkan komponen box collider 2D, checklist ‘Is Trigger’ dan sesuaikan ukuran dengan jarak gerak saw sesuai keinginan.



Gambar 10.16 Box Collider 2D Boundary

6. Cari komponen spikes, lalu drag and drop menuju hierarchy, dan ulangi Langkah 2-4 pada sub Enemy AI.



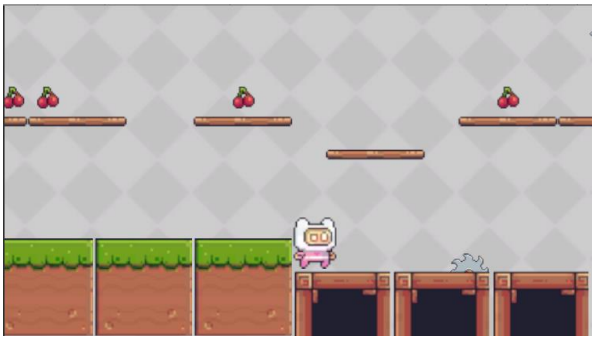
Gambar 10.17 Add Spikes Challenge

7. Buat file script dengan nama ‘Enemy_Behavior’ dan masukkan kode berikut dan berikan pada object challenge saw dan spikes.

```
public class Enemy_Behavior : MonoBehaviour
{
    [SerializeField] float moveSpeed = 1.2f;
    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);
    }
}
```

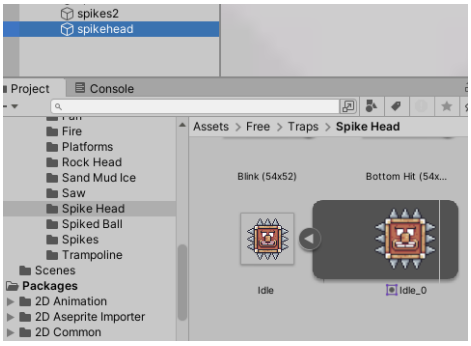
8. Lakukan uji coba jalankan game apakah saw sudah dapat berjalan mengikuti ukuran boundary.



Gambar 10.18 Hasil Running Enemy Behavior

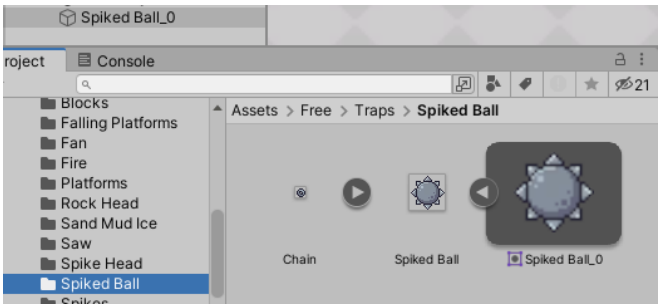
C. ENEMY AI

1. Tambahkan object spike head kedalam hierarchy dan atur posisi sedemikian rupa.



Gambar 10.19 Add Spike Head Challenge

2. Tambahkan object spike ball kedalam hierarchy dan atur posisi sedemikian rupa



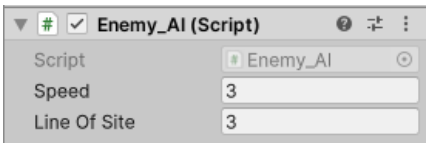
Gambar 10.20 Add Spike Ball Challenge

3. Buat script baru dengan nama 'Enemy_AI' dan masukkan kode berikut.

```
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);
    }
}
```

4. Pada Inspector Enemy_Ai, Atur Speed juga Line of Site untuk menentukan jarak dan speed pada enemy.



Gambar 10.21 Atur Speed dan Line of Site Script Enemy AI

5. Jalankan game, maka spike head dan spike ball akan mengikuti arah gerak player saat player dalam jangkauan.

Gambar 10.22 Hasil Running Enemy AI

D. Respawn

1. Buka script Player.cs tambahkan kode berikut.

```
//tambahkan pada bagian inisialisasi
    public int nyawa;
    [SerializeField] Vector3 respawn_loc;
    public bool play_again;
```

```
  private void Awake()
  {
    rb = GetComponent<Rigidbody2D>();
    animator = GetComponent<Animator>();
    respawn_loc = transform.position;
  }
```

```
void Update ()
  {
    ...
    if (nyawa<0)
```

```
    {
      playagain();
    }
    if (transform.position.y < -10)
    {
      play_again = true;
      playagain();
    }
  }
```

```
//tambahkan dibawah void awake
  void playagain()
  {
    nyawa = 3;
    transform.position = respawn_loc;
    play_again = false;
  }
```

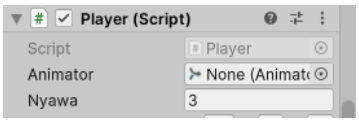
2. Buat script baru dengan nama ‘Enemy_Attack’ dan masukkan kode berikit.

```
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;
            }
        }
    }
}
```

3. Kemudian tambahkan script Enemy_attack pada seluruh challenge yang sudah dibuat dan ubah object sasaran menjadi nama player.

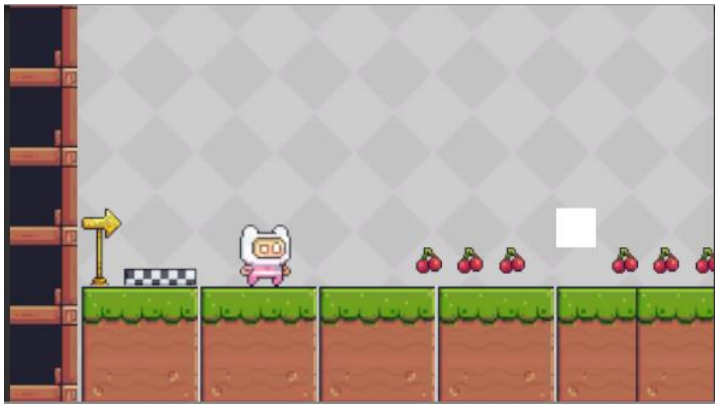
Gambar 10.21 Object Enemy Attack

4. Klik game object Player, pergi ke Inspector dan ubah nilai Nyawa menjadi 3 pada Player(Script).



Gambar 10.22 Nyawa Player

5. Jalankan permainan.



Gambar 10.23 Hasil Running Respawn

1.2 Kuis

```
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))
        {
          // Mengecek objek yang terkena memiliki komponen
            EnemyHealth EnemyHealth enemyHealth =
hit.transform.GetComponent<EnemyHealth>();
            if (enemyHealth != null)
            {
                // Mengurangi health musuh
                enemyHealth.TakeDamage(attackDamage);
            }
        }
    }
}
```

Analisa :

Kode PlayerAttack di atas bertujuan untuk memungkinkan pemain melakukan serangan melee saat tombol "Fire1" ditekan. Serangan ini menggunakan raycast untuk mendeteksi musuh dalam jarak tertentu dan mengurangi health musuh yang terkena.

1.2 Kuis 2

```
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 fungsi HandleJumpInput nilai Boolean is jumping harus terisi dengan kondisi true atau false. Lalu pada fungsi Handle MovemnetInput kondisi percabangan dengan nilai move=0 bukan 1. Dan pada logika gerak player pada vector3 harus memiliki nilai yang sama pada parameternya (a, b, c) agar konsisten.