TUGAS PERTEMUAN: 8

MEMBUAT CAMERA & CHARACTER MOVEMENT

NIM	:	1918082
Nama	:	Dimas Fariski Setyawan Putra
Kelas	:	E
Asisten Lab	:	M. Rafi Faddilani (2118114)
Baju Adat	:	-
Referensi	:	-

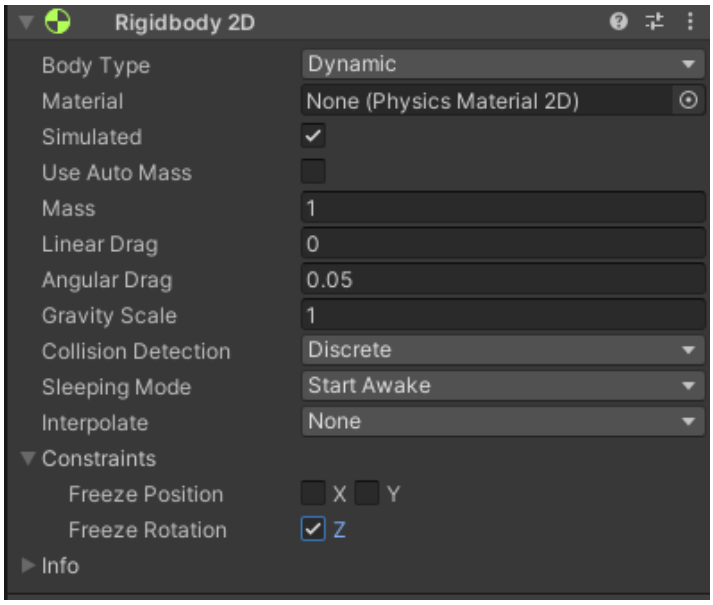
1.1 Tugas 1 : Membuat Character Movement

A. Membuat Character Movement

1. Buka file project sebelumnya

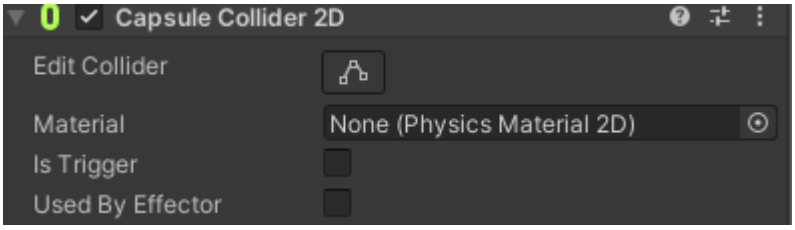
Gambar 1.1 Membuka Projek

2. Klik player-idle-1 tambahkan Component Rigidbody 2D, sesuaikan settingannya seperti gambar berikut, Centang pada Freeze Rotation Z



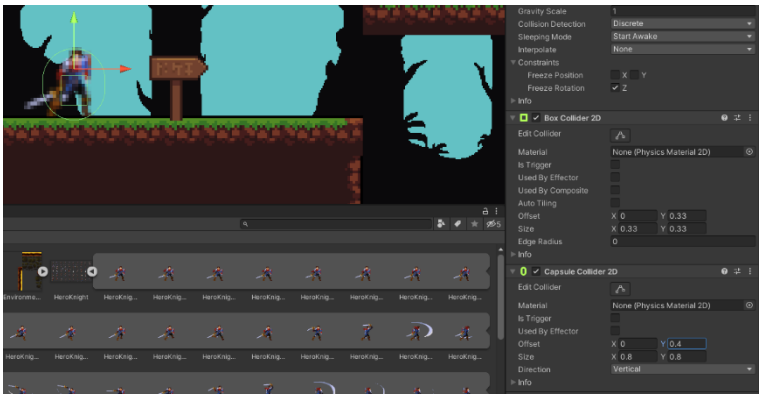
Gambar 1.2 Menambahkan Rigidbody 2D

3. Lalu tambahkan komponen Capsule Colider di player-idle-1, lalu klik icon sebelah kanan edit collider



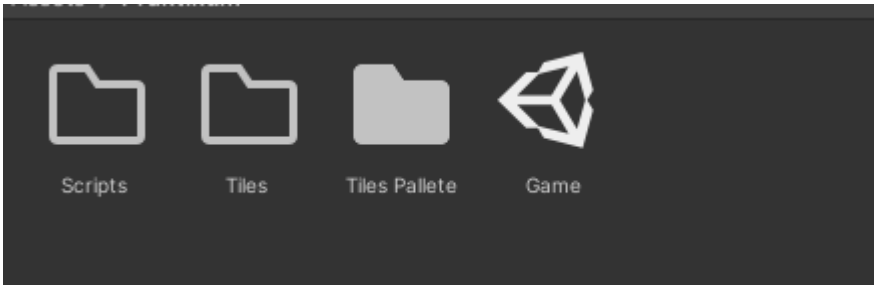
Gambar 1.3 Menambahkan Capsule Collider 2D

4. Lalu cocokan garis oval degan karakternya atau bisa di inputkan Offset X, Y dan juga Size X, Y nya



Gambar 1.4 Mengubah Offset Capsule Collider 2D

5. Buka Folder praktikum, lalu bikin folder baru bernama Script

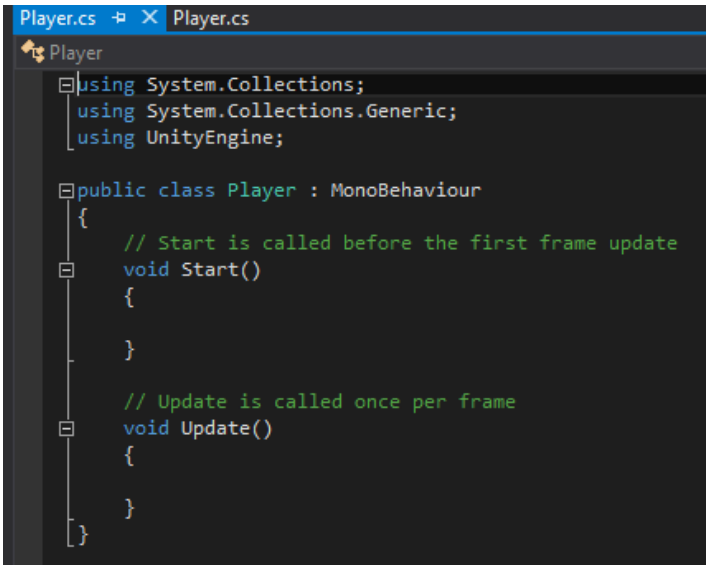


Gambar 1.5 Membuat Folder Script

6. Masuk kedalam folder Script, lalu buat C# Script, beri nama Player

Gambar 1.6 Membuat Script Player

7. Drag & drop script player kedalam Hirarki player-idle-1, lalu klik 2x pada script player maka akan masuk kedalam text editor seperti ini



Gambar 1.7 Memasukkan Script Ke Hirarki

8. Masukan source code dibawah ini, pastikan nama public class harus sama dengan nama file yang dibuat.

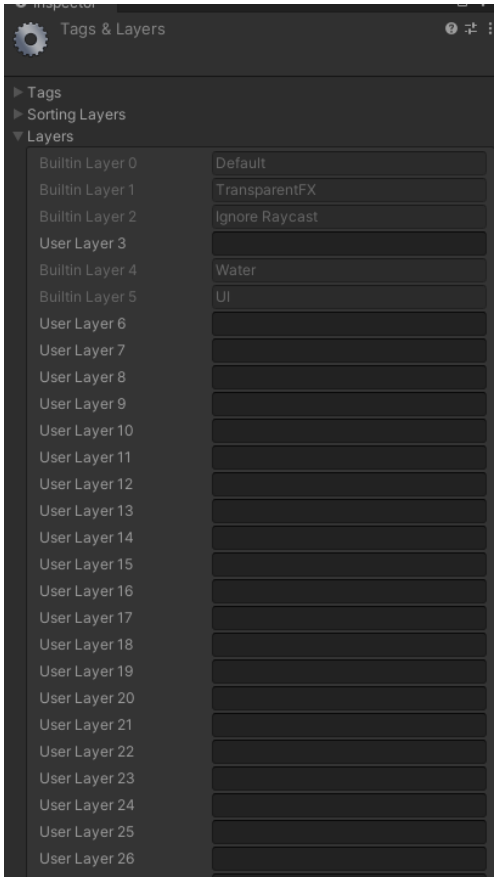
Player.cs
`using System.Collections;` `using System.Collections.Generic;`

```
using UnityEngine;

public class Player : MonoBehaviour
{
  Rigidbody2D rb;

  [SerializeField] float speed = 1;
  float horizontalValue;
  bool facingRight;

  private void Awake()
  {
    rb = GetComponent<Rigidbody2D>();
  }

  void Update ()
  {
    horizontalValue = Input.GetAxisRaw("Horizontal");
  }

  void FixedUpdate()
  {
    Move(horizontalValue);
  }

  void Move(float dir)
  {
    #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(-1, 1, 1);
      facingRight = false;
    }

    else if (!facingRight && dir > 0)
    {
      // ukuran player
      transform.localScale = new Vector3(1, 1, 1);
      facingRight = true;
    }

    #endregion
  }
}
```

9. Untuk mencoba Source code diatas berhasil, Tekan dikeyboard “a” atau “left arrow” untuk ke arah kiri, tekan “d” atau “right arrow” untuk ke arah kanan

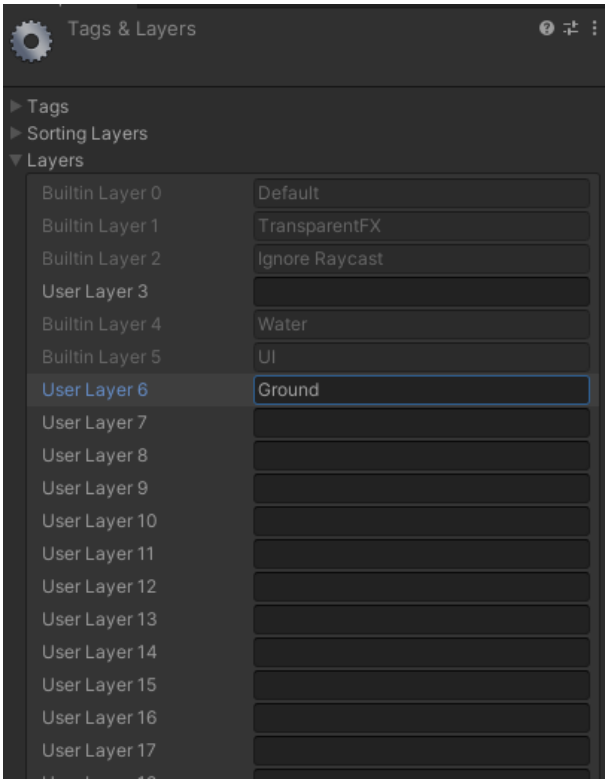
Gambar 1.8 Hasil Menggerakkan Karakter

10. Untuk membuat player loncat menggunakan spasi, kita perlu membuat GorundCheck dengan cara, klik Grid pada Hierarchy, pergi ke inspector, pilih Layer, Klik Add Layer



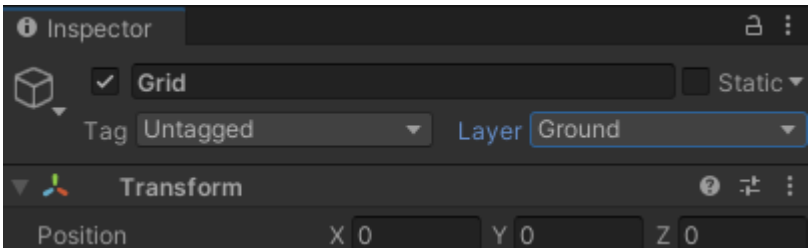
Gambar 1.9 Menambahkan Layer

11. Lalu isi “Ground” pada User Layer 6



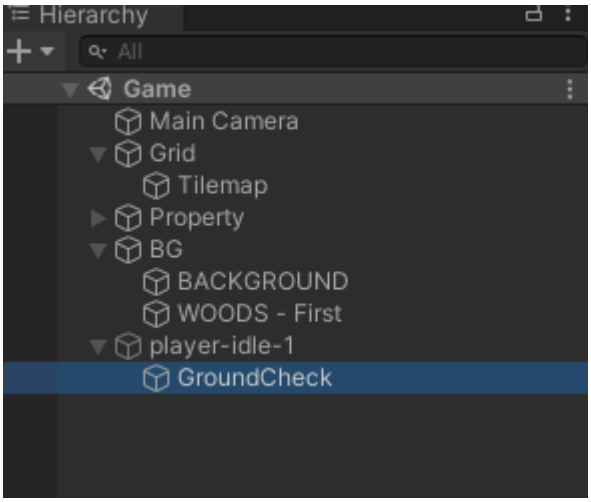
Gambar 1.10 Menambahkan Ground Layer

12. Ubah Layer menjadi Ground, jika muncul pop up Change Layer, klik yes saja



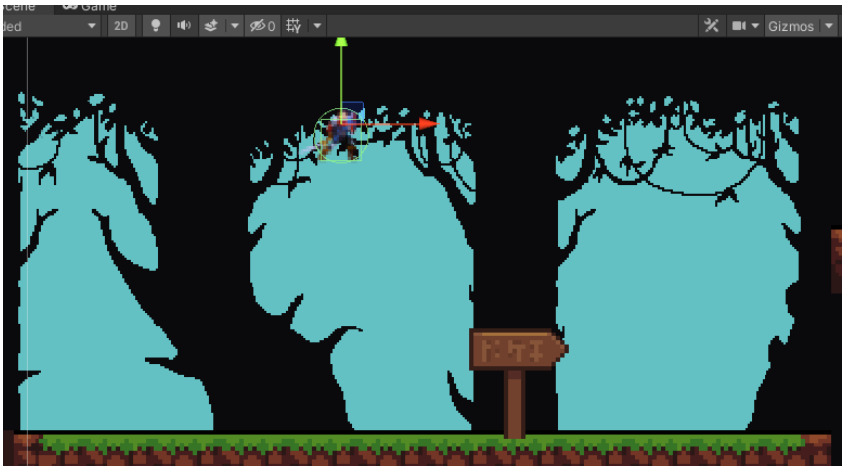
Gambar 1.11 Mengubah Layer

13. Klik kanan pada player-idle-1, lalu Create empty, beri nama GorundCheck



Gambar 1.12 Membuat Object GroundCheck

14. Klik pada Hirarki GorundCheck, lalu gunakan “Move Tools” untuk memindahkan ke bagian bawah Player seperti gambar berikut



Gambar 1.13 Memindahkan Karakter

15. Kembali ke script Player tambahkan source code seperti ini

```
public class Player : MonoBehaviour
{
    Rigidbody2D rb;

    [SerializeField]
    Transform groundcheckCollider;
    [SerializeField]
    LayerMask groundLayer;

    const float groundCheckRadius = 0.2f; // +
    [SerializeField]
    float speed = 1;
    float horizontalValue;

    [SerializeField]
    bool isGrounded; // +
    bool facingRight;
```

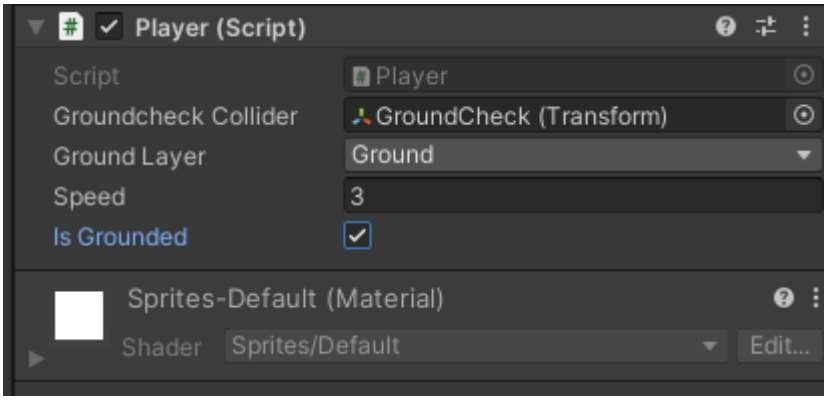
Gambar 1.14 Membuat Variable Player

16. Buat void ground check dibawah void fixedUpdate & tambahkan GroundCheck(); pada void fixedUpdate

```
void FixedUpdate()
{
    GroundCheck();
    Move(horizontalValue);
}

void GroundCheck()
{
    isGrounded = false;
    Collider2D[] colliders = Physics2D.OverlapCircleAll(groundcheckCollider.position, groundCheckRadius, groundLayer);
    if (colliders.Length > 0)
        isGrounded = true;
}
```

Gambar 1.15 Mengubah Script Player

17. Klik player-idle-1, lalu ke inspector ke effect Player script di bagian "Goruncheck collider" tekan icon lalu pilih yang GorundCheck Transform, dan pada Ground Layer pilih Ground



Gambar 1.16 Mengubah Groundcheck Collider

18. Lalu untuk membuat player melompat tambahkan script berikut

```
public class Player : MonoBehaviour
{
    Rigidbody2D rb;

    [SerializeField] Transform groundcheckCollider;
    [SerializeField] LayerMask groundLayer;

    const float groundCheckRadius = 0.2f; // +
    [SerializeField] float speed = 1;
    float horizontalValue;
    [SerializeField] float jumpPower = 100;

    bool jump;

    [SerializeField] bool isGrounded; // +
    bool facingRight;
```

Gambar 1.17 Menambahkan Variable Melompat

19. Tambahkan juga script berikut di bagian void update

```
void Update()
{
    horizontalValue = Input.GetAxisRaw("Horizontal");
    if (Input.GetButtonDown("Jump"))
        jump = true;
    else if (Input.GetButtonUp("Jump"))
        jump = false;
}
```

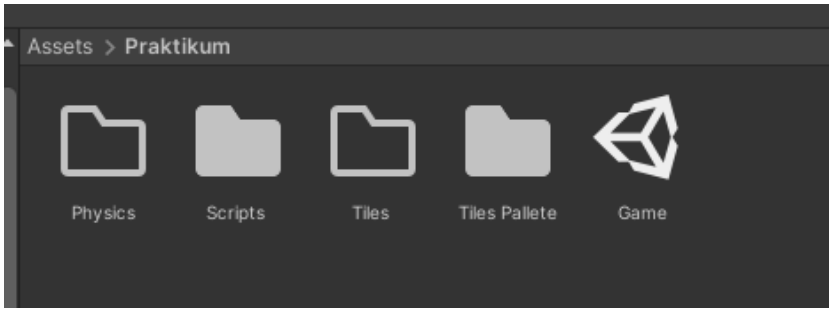
Gambar 1.18 Menambahkan Script Melompat

20. Tambahkan juga jump pada parameter Move

```
void FixedUpdate()
{
    GroundCheck();
    Move(horizontalValue, jump);
}

void GroundCheck()
{
    isGrounded = false;
    Collider2D[] colliders = Physics2D.OverlapCircleAll(groundcheckCollider.position, groundCheckRadius, groundLayer);
    if (colliders.Length > 0)
        isGrounded = true;
}

void Move(float dir)
{
    #region gerak kanan kiri

    if(isGrounded && jumpflag)
     {
      isGrounded = false;
      jumpflag = false;
      rb.AddForce(new Vector2(0f, jumpPower));
     }
    float xVal = dir * speed * 100 * Time.fixedDeltaTime;
    Vector2 targetVelocity = new Vector2(xVal, rb.velocity.y);
    rb.velocity = targetVelocity;
```

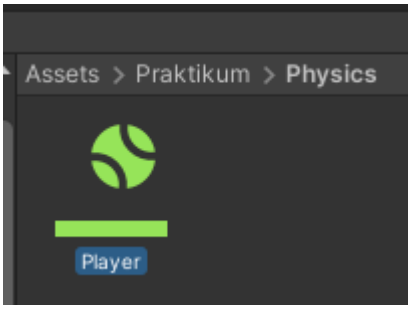
Gambar 1.19 Mengubah Parameter Move

21. Buat folder baru di praktikum bernama “Physics”



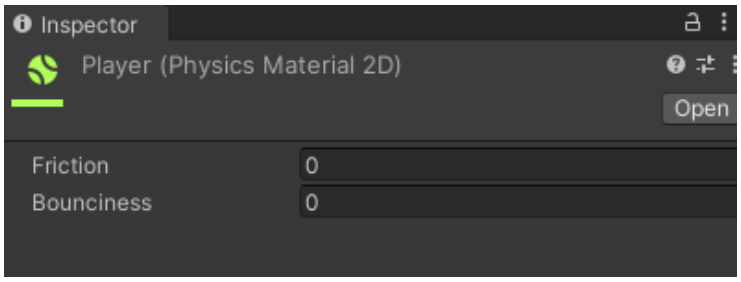
Gambar 1.20 Menambahkan Folder Physics

22. Didalam folder Pyshics create > 2d > physical material 2d , berinama “Player”



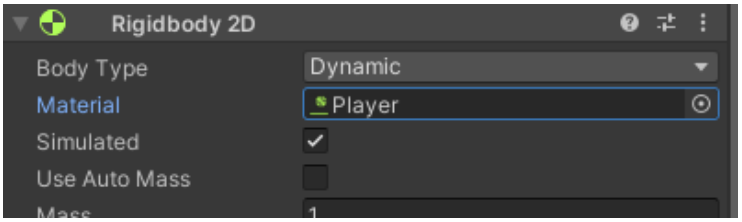
Gambar 1.21 Menambahkan Material Physics

23. Klik Player (Physics Material 2D), dibagian menu inspector, friction & bounces ubah menjadi 0



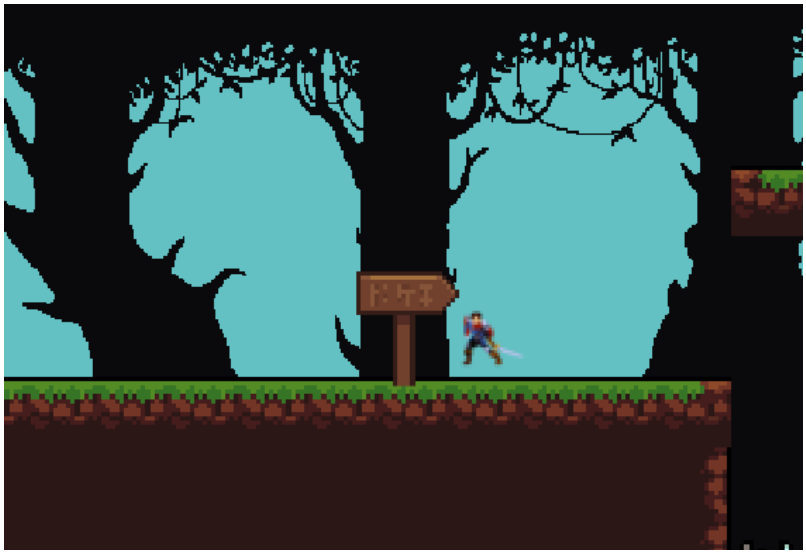
Gambar 1.22 Mengubah Konfigurasi Physics

24. Klik Hierarchy pilih layer player idle 1, pada Inspector Cari Rigidbody 2D lalu klik icon untuk membuka box select physhics material 2d , lalu pilih asset Player yang sudah kita buat tadi



Gambar 1.23 Mengubah Material RigidBody 2D

25. Tekan play, maka player bisa melompat dengan menekan spasi

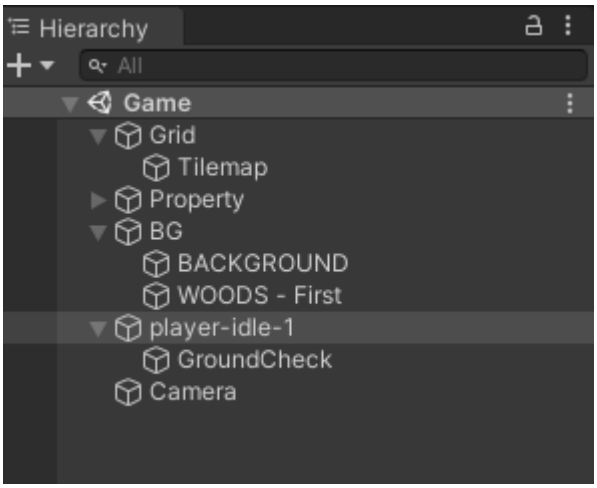


Gambar 1.24 Hasil Tampilan Karakter Melompat

1.2 Tugas 2 : Membuat Camera Movement

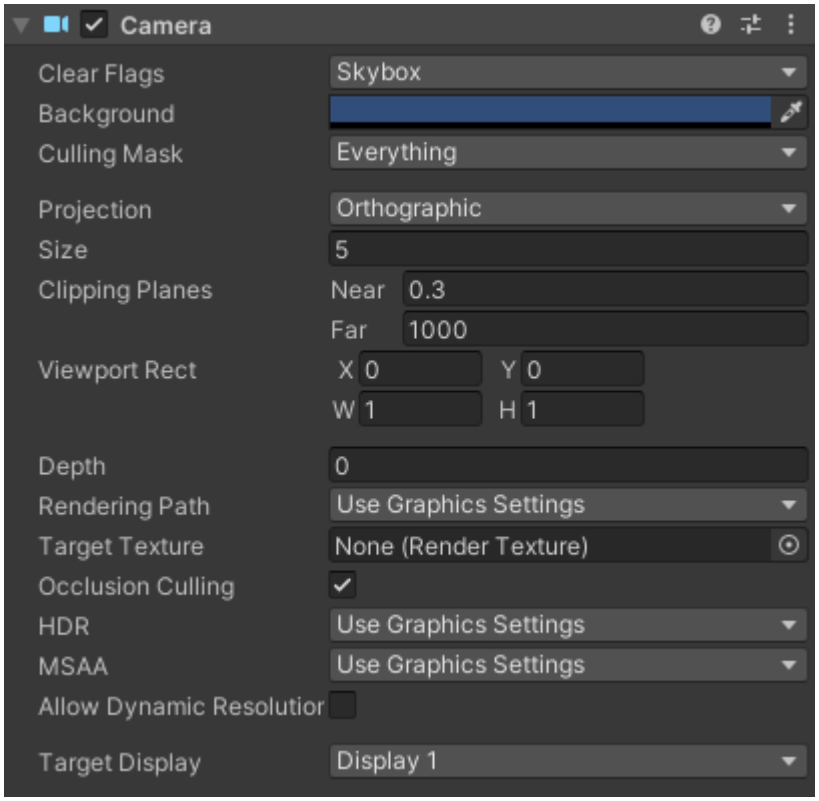
A. Membuat Camera Movement

1. Create Empty pada Hirarki, dan Rename Menjadi Camera



Gambar 1.25 Menambahkan Hirarki Camera

2. Tambahkan komponen Camera dan konfigurasikan seperti berikut



Gambar 1.26 Menambahkan Komponen Camera

3. Buat file script baru di folder Script dengan nama ”CameraFollow”

CameraFollow.cs

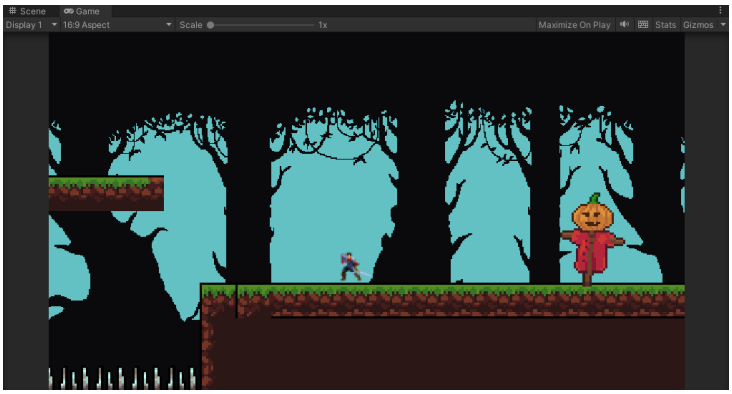
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class CameraFollow : MonoBehaviour
{

    public float xMargin = 0.5f;
    public float yMargin = 0.5f;
    public float xSmooth = 4f;
    public float ySmooth = 4f;
    public Vector2 maxXAndY;
    public Vector2 minXAndY;
    private Transform player;

    void Awake()
    {
```

```
        player                                                  =
GameObject.FindGameObjectWithTag("Player").transform;
    }

    bool CheckXMargin()
    {
        return     Mathf.Abs(transform.position.x     -
player.position.x) > xMargin;
    }

    bool CheckYMargin()
    {
        return     Mathf.Abs(transform.position.y     -
player.position.y) > yMargin;
    }

    void FixedUpdate()
    {
        TrackPlayer();
    }

    void TrackPlayer()
    {
        float targetX = transform.position.x;
        float targetY = transform.position.y;
        if (CheckXMargin())
            targetX  =  Mathf.Lerp(transform.position.x,
player.position.x,
            xSmooth * Time.deltaTime);
        if (CheckYMargin())
            targetY  =  Mathf.Lerp(transform.position.y,
player.position.y,
            ySmooth * Time.deltaTime);
            targetX  =  Mathf.Clamp(targetX,  minXAndY.x,
maxXAndY.x); targetY =
            Mathf.Clamp(targetY,            minXAndY.y,
maxXAndY.y); transform.position = new
            Vector3(targetX,                   targetY,
transform.position.z);}}
```

4. Drag & drop script CameraFollow Kedalam Layer Camera dan konfigurasikan seperti berikut

Gambar 1.27 Menkonfigurasi CameraFollow

5. Tekan play untuk menjalankan, maka sekarang kamera akan mengikuti pergerakan karakter

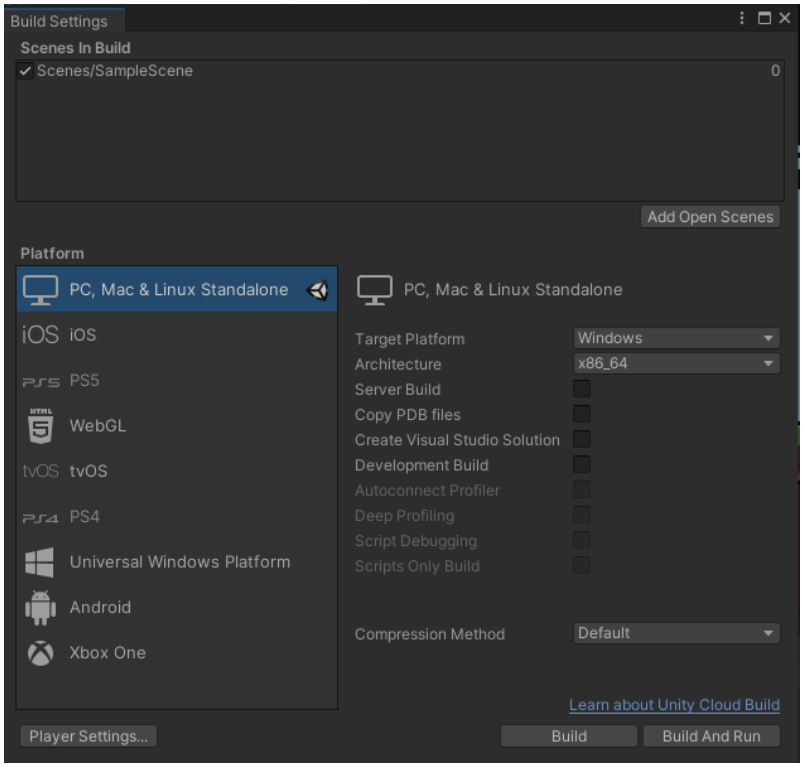


Gambar 1.28 Hasil Tampilan Camera Movement

1.3 Tugas 2 : Render

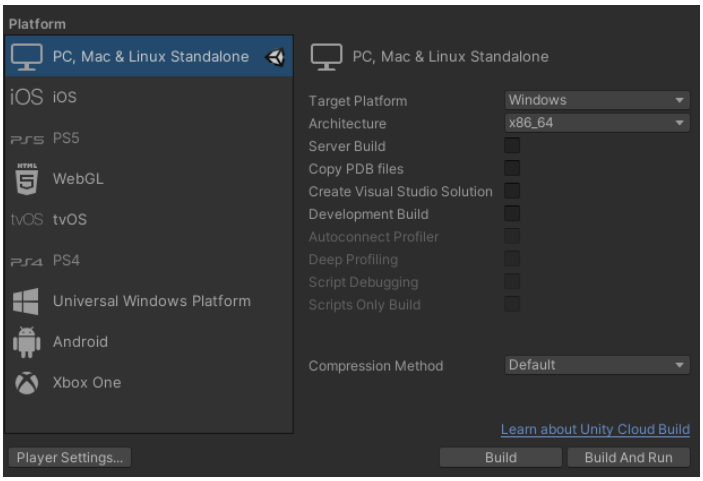
A. Render

1. Pergi ke menu File kemudian pilih Build Setting (Ctrl + Shift + B)



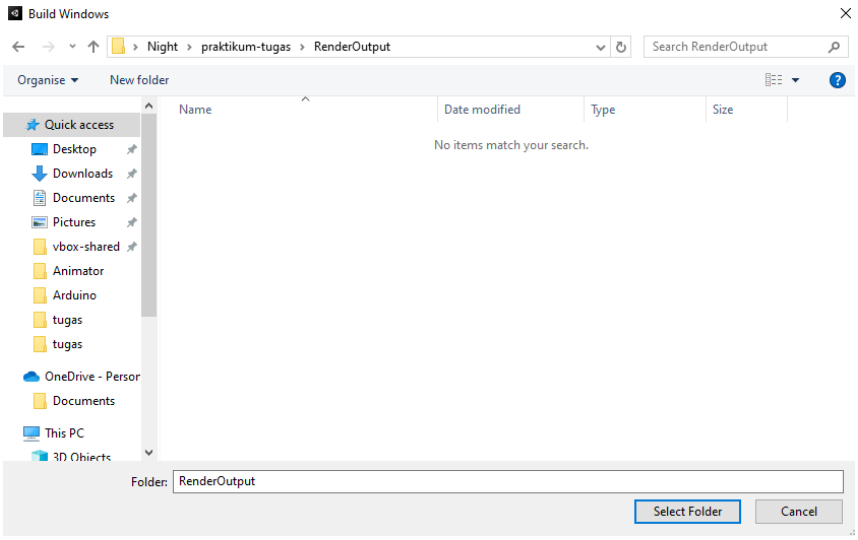
Gambar 1.29 Membuka Build Settings

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



Gambar 1.30 Memilih Build Settings

3. Pilih lokasi render



Gambar 1.31 Memilih Lokasi Render

4. Hasil Render

Gambar 1.32 Hasil Render

B. Kuis BAB 8

Kuis CameraFollow

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

public class CameraFollow : MonoBehaviour
{
[SerializeField] private Transform player;

void Update(){
transform.position = new Vector3(player.position.x,
transform position.y, transform.position.z);
}}
```

Penjelasan:

Pada program di atas akan menggunakan library System dan UnityEngine, kemudian akan mendeklarasikan class CameraFollow dengan *inheritance* dari MonoBehaviour. Setelah itu akan mendeklarasikan variable class Transform dengan nama player dan jenis enkapsulasi *private.* Setelah itu terdapat method Update. Pada method tersebut akan merubah posisi kamera yang mengikuti isi dari Vector3(x, y, z) dimana posisi x merupakan posisi dari karakter. Dengan kata lain, kamera akan mengikuti posisi karakter secara horizontal saja.

C. Link Github Pengumpulan

https://github.com/a-blue-moon/animasi-dan-game/tree/main/BAB%208