# TUGAS PERTEMUAN: 8

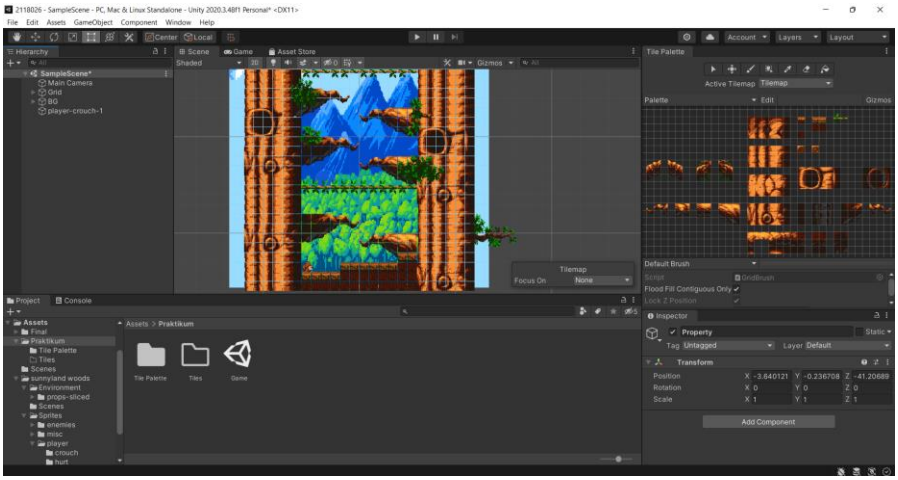
# CAMERA & CHARACTER MOVEMENT

| | | |
|---|---|---|
| **NIM** | : | 2118026 |
| **Nama** | : | Bryan Ifan Etikamena |
| **Kelas** | : | B |
| **Asisten Lab** | : | Aprillia Dwi Dyah S (2118143) |

## 8.1 Tugas 8 : Membuat Character Movement, Detect Ground, Jumping, & Camera Movement Tidak Termasuk Animasi Karakter

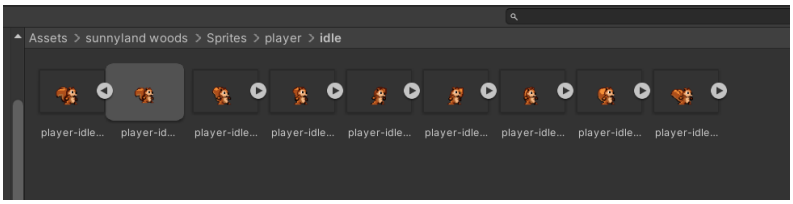
### A. Langkah-langkah Membuat Character Movement

1. Buka *file* projek *Unity* sebelumnya pada bab 7 untuk digunakan kembali.



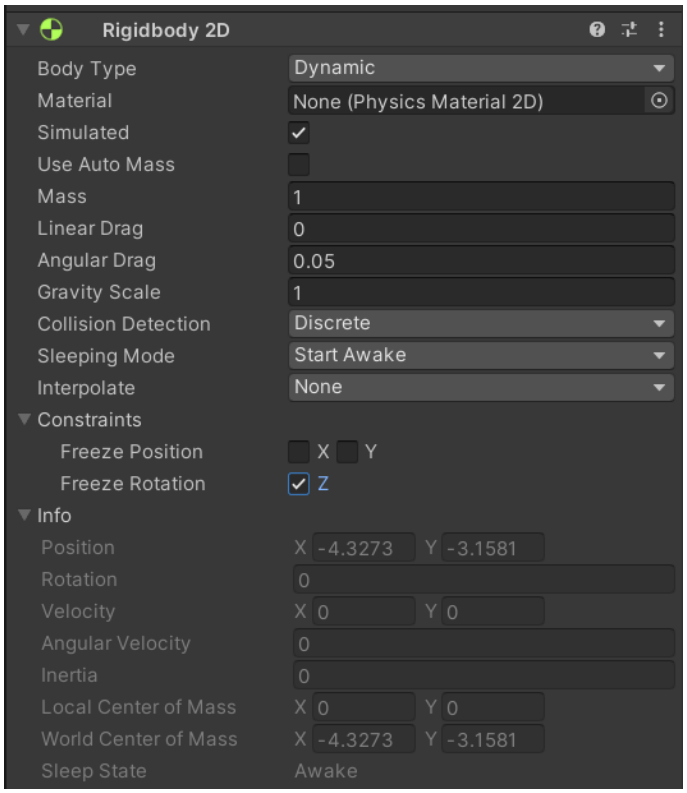
Gambar 8.1 Membuka *Project Unity*

2. Tambahkan *player* bernama *player-id*, pilih yang *idle*, *Import* ke dalam Hirarki.




Gambar 8.2 Menambahkan *Player*

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




Gambar 8.3 Menambahkan *Component Rigidbody* 2D

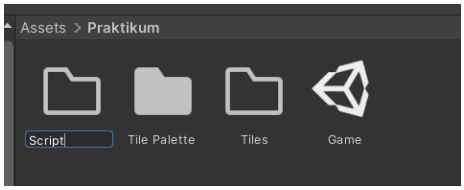
4. Lalu tambahkan komponen *Capsule Colider* di *player-idle-1*, lalu klik *icon* sebelah kanan *edit collider*.


Gambar 8.4 Menambahkan *Component Capsule Colider*

5. Lalu cocokan garis oval degan karakternya atau bisa diinputkan *Offset* X, Y dan juga *size* X, Y nya.

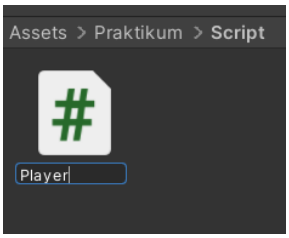
Gambar 8.5 Mencocokan Garis Oval

6. Buka *Folder* praktikum, lalu bikin *folder* baru bernama *Script*.



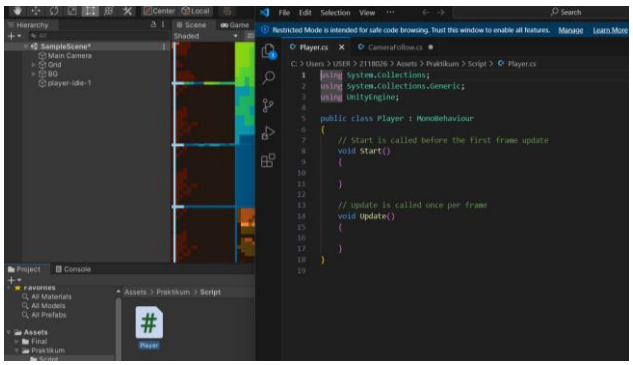

Gambar 8.6 Membuat *Folder Script*

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




Gambar 8.7 Membuat C# *Script*

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

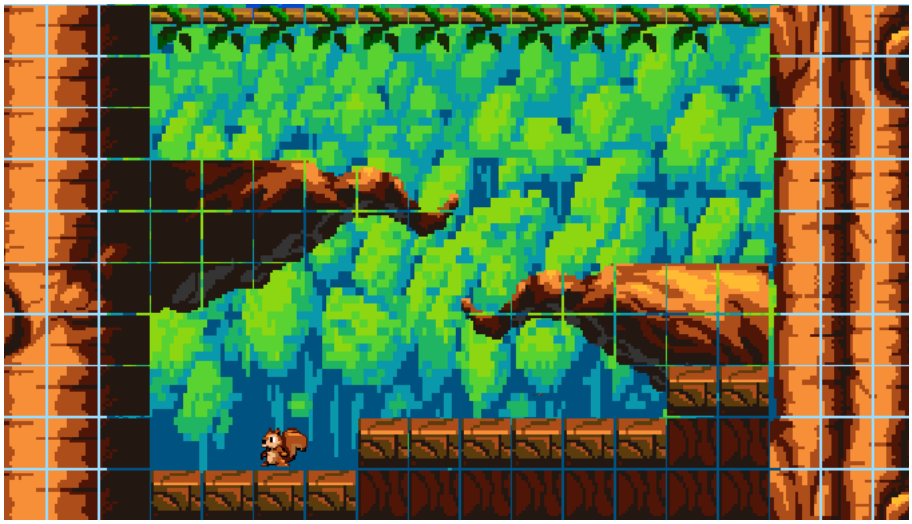


Gambar 8.8 Membuka *Text Editor*

9. Masukan *source code* di bawah ini, pastikan nama *public class* harus sama dengan nama *file* yang dibuat.

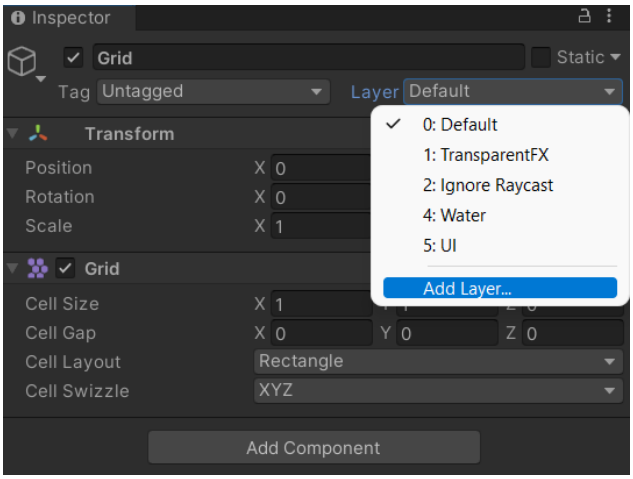
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

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
  }
}
```

10. Untuk mencoba *Source code* di atas berhasil, Tekan di *keyboard* “a” atau “*left arrow*” untuk ke arah kiri, tekan “d” atau “*right arrow*” untuk ke arah kanan.



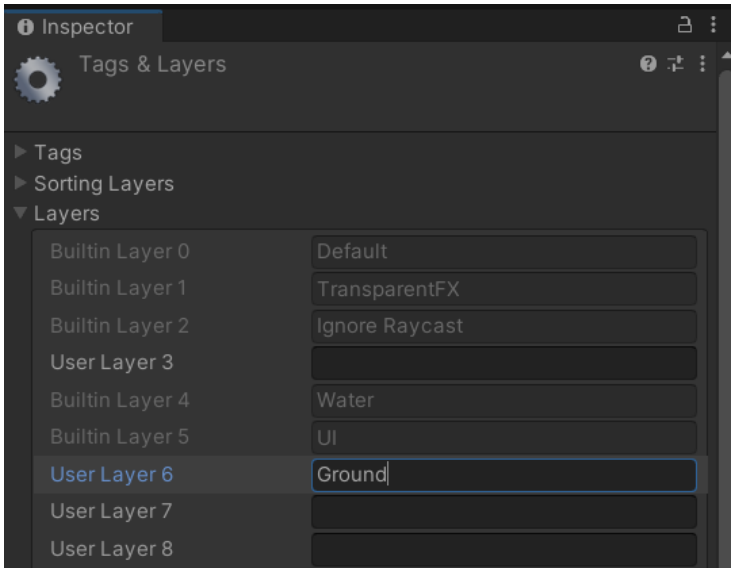
Gambar 8.9 Mencoba *Source Code*

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



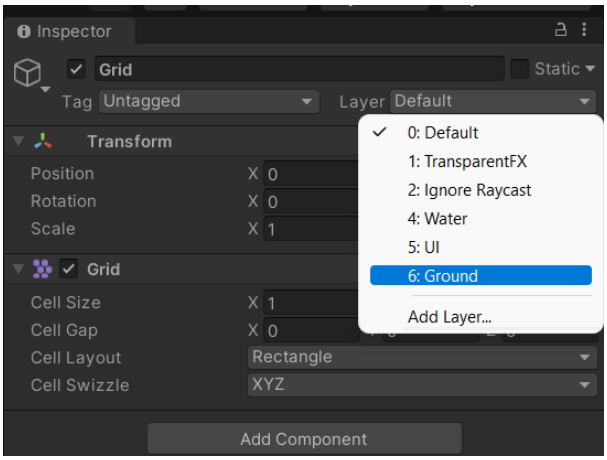

Gambar 8.10 Membuat *GorundCheck*

12. Lalu isi *"Ground"* pada *User Layer* 6.



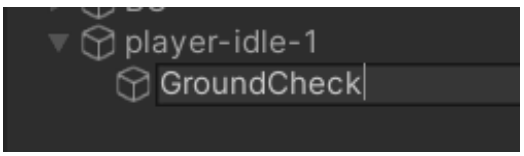

Gambar 8.11 Menambahkan *GorundCheck*

13. Ubah *Layer* menjadi *Ground*, jika muncul *pop* uip *Change Layer*, klik *yes* saja.




Gambar 8.12 Mengubah *Layer* Menjadi *Ground*

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




Gambar 8.13 Menambahlan *GorundCheck*

15. Klik pada Hirarki *GorundCheck*, lalu gunakan "*Move Tools*" untuk memindahkan ke bagian bawah *Player* seperti gambar berikut.

Gambar 8.14 Memindahkan *GorundCheck* Ke Bawah *Player*

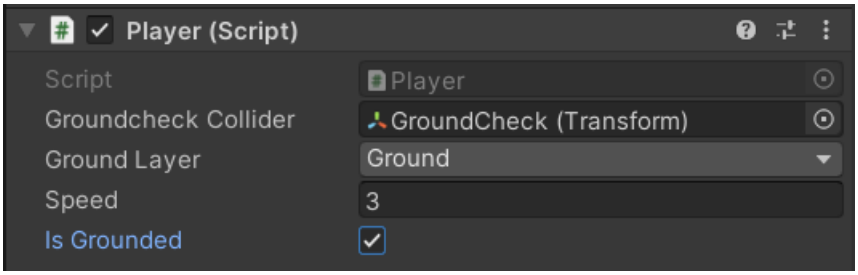
16. Kembali ke *script Player* tambahkan *source code* seperti ini.

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

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

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

17. Buat *void ground check* di bawah *void fixedUpdate* & tambahkan *GorunCheck*(); 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;
  }
```

18. 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 8.15 Mengatur *Effect Player Script*

19. Lalu untuk membuat *player* melompat tambahkan *script* berikut.

```
Player.cs [SerializeField] float jumpPower = 100;
bool jump;
```

20. Tambahkan juga *script* berikut di bagian *void update.*

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

21. Tambahkan juga *jump* pada parameter *Move*.

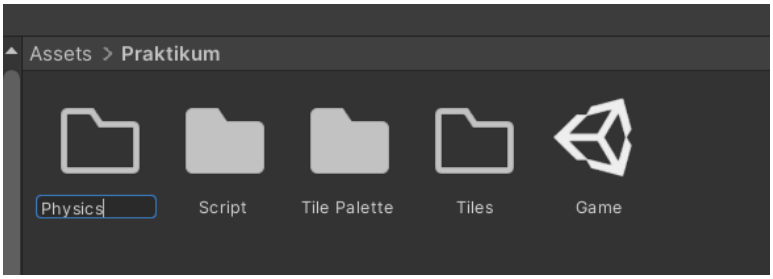
```
void FixedUpdate()
{
  GroundCheck();
  Move(horizontalValue);
}
```

Gambar 8.16 Mengklik *Tilemap*

22. Tambahkan *script* berikut pada *void Move.*

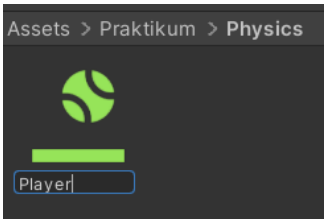
```
bool jumpflag

if(isGrounded && jumpflag)
 {
  isGrounded = false;
  jumpflag = false;
  rb.AddForce(new Vector2(0f, jumpPower));
}
```

23. Buat folder baru di praktikum bernama “Physics”.



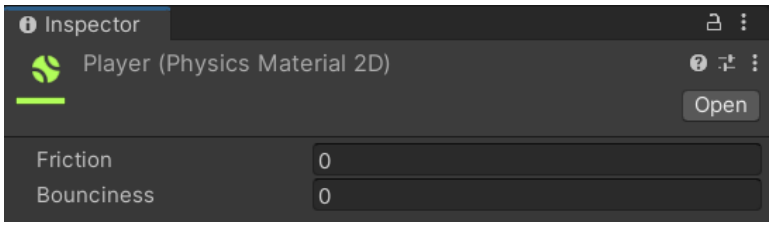

Gambar 8.17 Membuat *Folder Physiscs*

24. Di dalam *folder Pyshics create* > 2d > physical material 2d, beri nama “*Player*”.



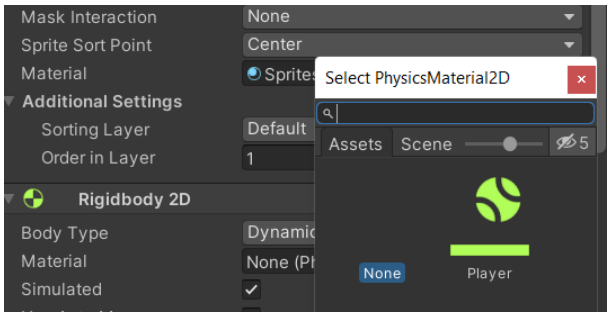

Gambar 8.18 Membuat *Physical Material* 2D

25. Klik *Player* (*Physics Material* 2D), di bagian menu *inspector*, *friction & bounces* ubah menjadi 0.



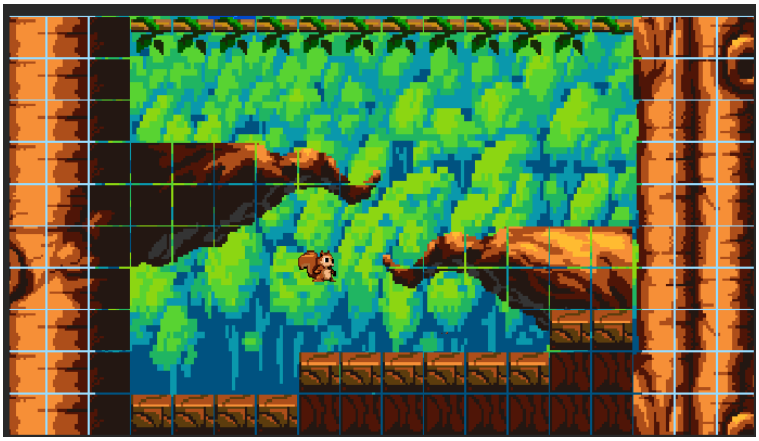

Gambar 8.19 Mengubah *Friction & Bounces* Menjadi 0

26. 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 8.20 Menambahkan *Physhics Material* 2d

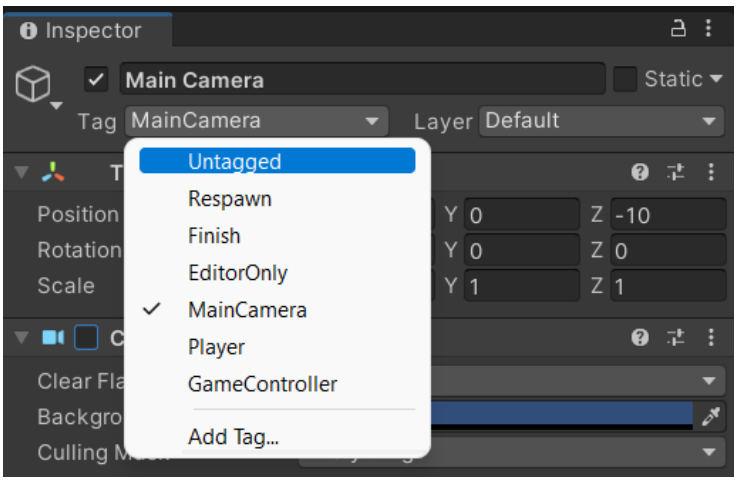
27. Tekan *play*, maka *player* bisa melompat dengan menekan spasi.



Gambar 8.21 *Player* Melompat

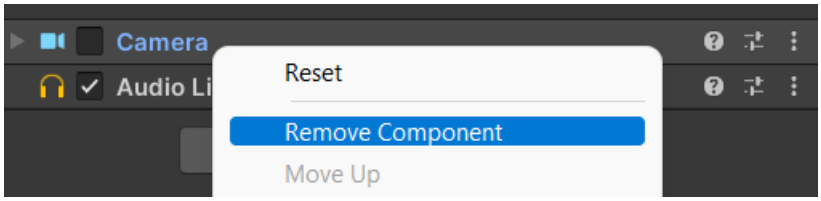
## B. Langkah-langkah Membuat Camera Movement

1. Pada Hirarki *Property* Ubah *Inspector* pada *tag Main camera* Menjadi *untaged.*



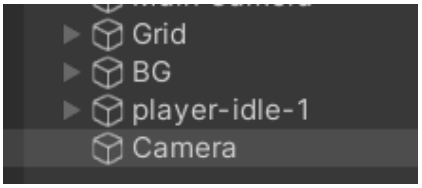

Gambar 8.22 *Untag Main Camera*

2. Pada *Effect Camera* pilih *Remove Component.*



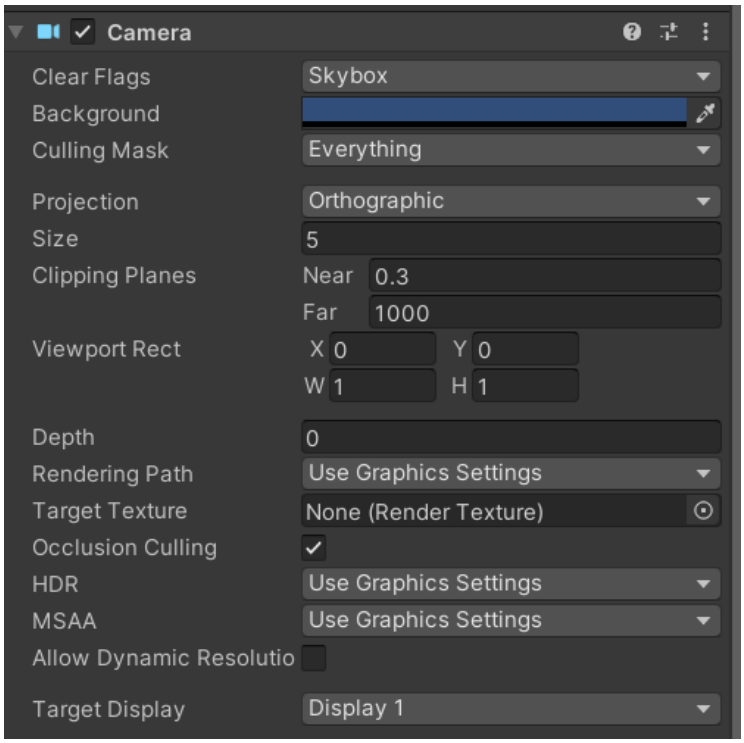

Gambar 8.23 *Remove Camera*

3. *Create Empty* pada Hirarki dan *Rename* Menjadi *Camera*.




Gambar 8.24 Membuat *Camera*

4. Sesuaikan *Setting Layer Camera* seperti gambar di bawah ini.




Gambar 8.25 Mengatur *Layer Camera*

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

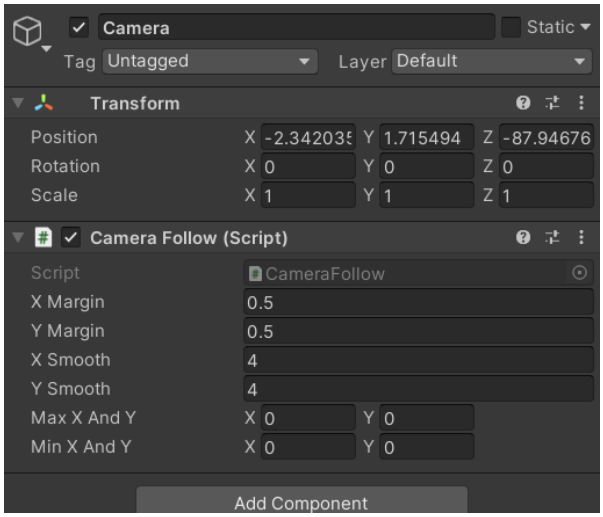

Gambar 8.26 Membuat *Folder Script CameraFollow*

6. Lalu tuliskan script berikut ini.

```
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);
    }
}
```

7. *Drag & drop script CameraFollow* Ke dalam *Layer Camera*.

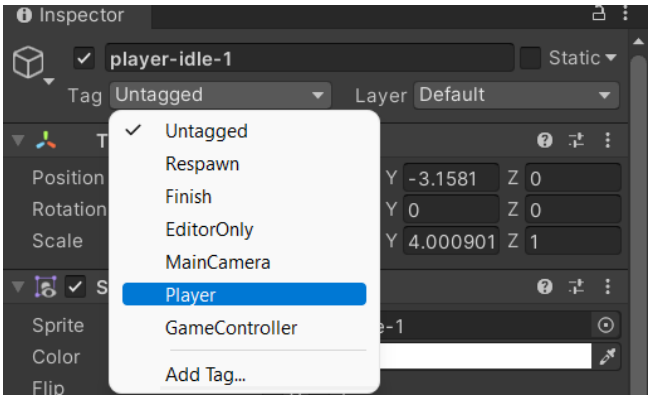



Gambar 8.27 Menambahkan *Script CameraFollow* Ke *Camera*

8. Lalu klik pada *camera*, buka *inspector* Pada bagian *Camera Follow* (*Script*) Ubah Bagian *Max* X dan *Max* Y nya.

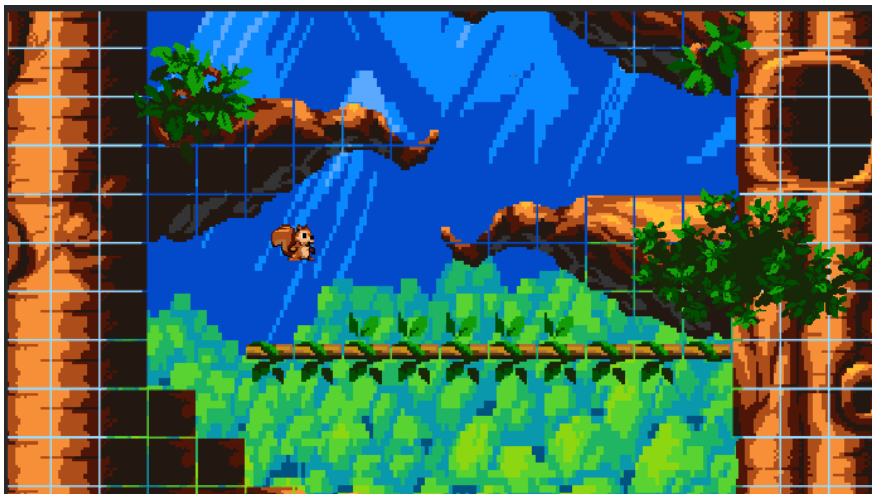

Gambar 8.28 Mengubah Bagian *Max* X dan *Max* Y

9. Ubah tag di player-idle-1 Untagged menjadi *"Player"*.




Gambar 8.29 Mengubah *Tag Player*

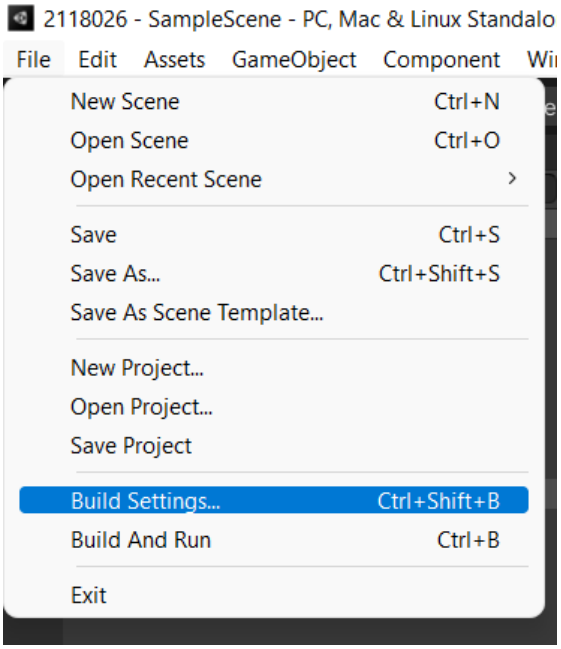
10. Tekan *play* untuk menjalankan, maka sekarang kamera akan mengikuti pergerakan karakter.



Gambar 8.30 Menjalankan *Game*

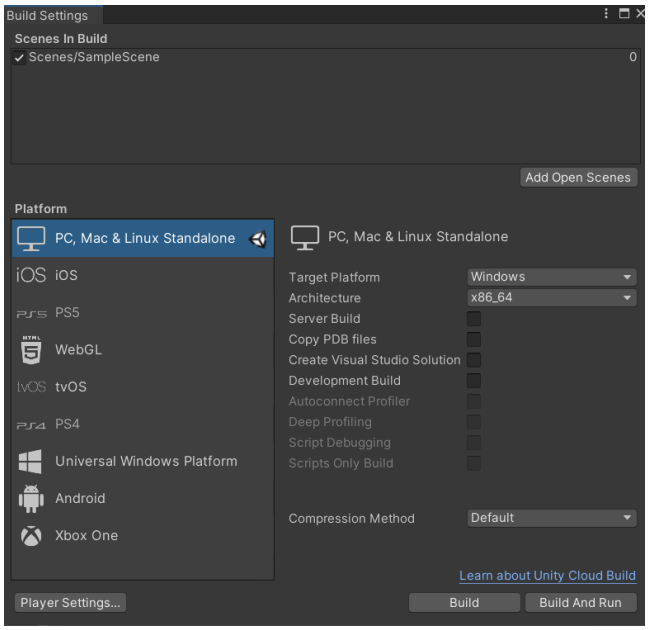
## C. Langkah-langkah Render

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



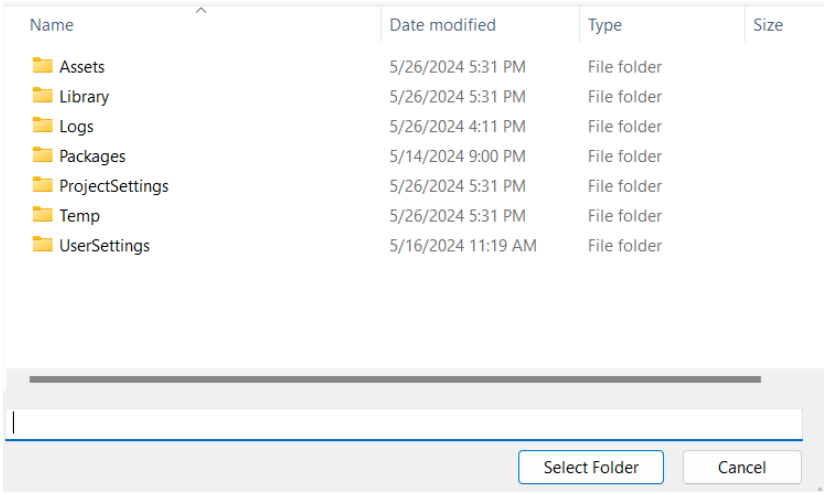

Gambar 8.31 Menu *File*

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



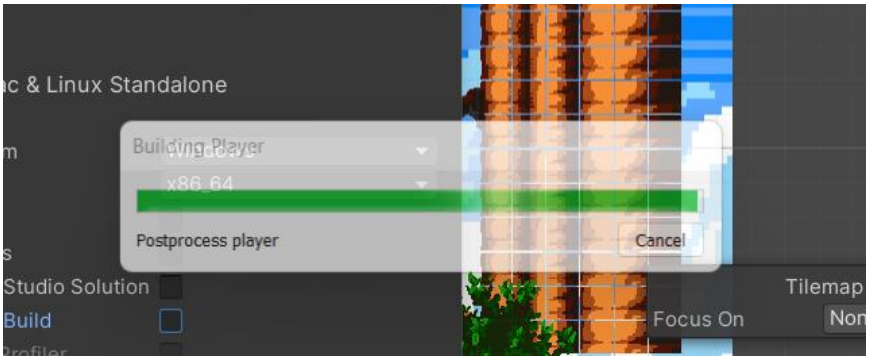

Gambar 8.32 Memilih *Setting Build*

3. Pilih di mana *Project* disimpan, dan tunggu hasilnya.



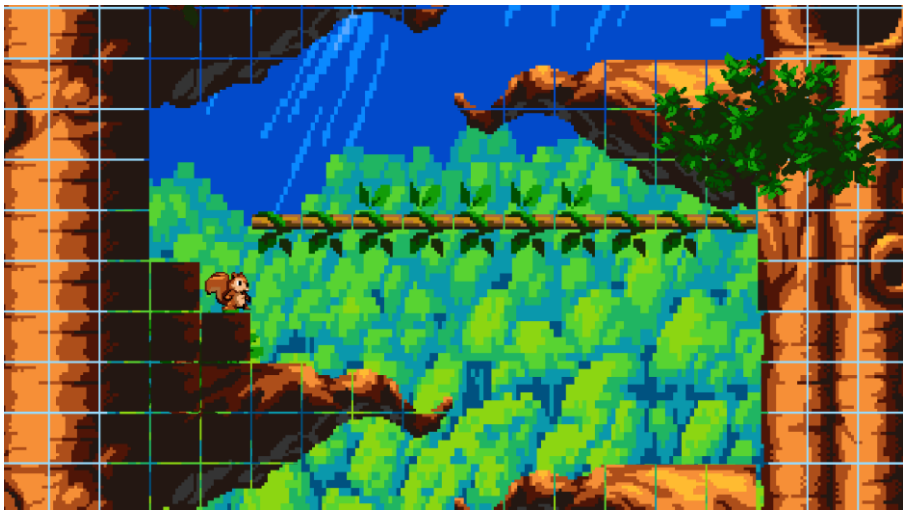

Gambar 8.33 Memilih Lokasi Penyimpanan *Project*

4. Tunggu.




Gambar 8.34 Menunggu Menunggu Hasil *Render*

5. Hasil *Render*.



Gambar 8.35 Hasil *Render*

## 8.2 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);
    }
}
```

Analisis :

Tigas baris awal pada source code di atas digunakan untuk menggunakan library yang ada pada unity. Bari pertama untuk menggunakan koleksi yang ada selain generic. Baris kedua untuk menggunakan koleksi generic. Dan baris ketiga untuk menggunakan library fungsi fungsi yang ada pada unity. Selanjutnya terdapat sebuah kelas yang bersifat public yang berarti dapat diakses oleh yang lain bernama Camera Follow Dimana kelas ini mengikuti atau mewarisi MonoBehaviour. Selanjutnya di dalam kelas ini terdapat variable player dengan tipenya transform dan variable ini bersifat private. Kemudian terdapat juga method update yang bersifat void atau tidak mengembalikan nilai. Isi dari method ini adalah perintah untuk membuat kamera mengikuti pemain secara horizontal. Dimana player.position.x ini perintah untuk mengambil posisi player, transform.position.y ini mempertahankan posisi y pada kamera terhadap player dan transform.position.z mempertahankan posisi z pada kamera terhadap player.