# TUGAS PERTEMUAN: 8

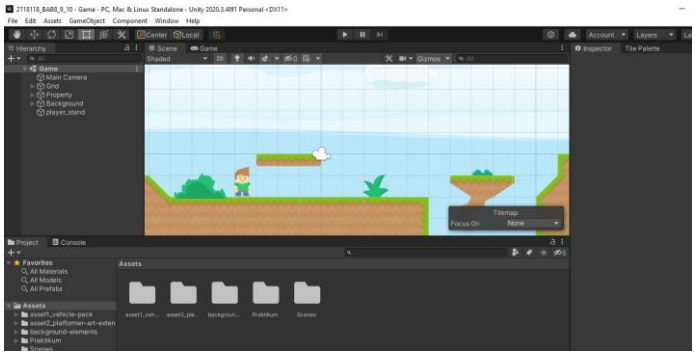
# CAMERA AND CHARACTER MOVEMENT

| | | |
|---|---|---|
| **NIM** | : | 2118118 |
| **Nama** | : | Legming Dwi Anggraini |
| **Kelas** | : | C |
| **Asisten Lab** | : | Nayaka Apta Nayottama (2218102) |

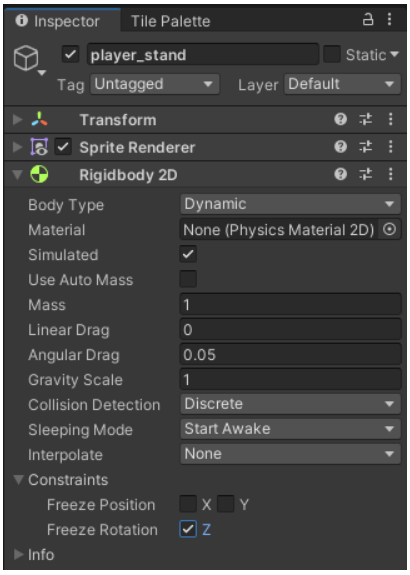
## 1.1 Tugas 1 : Membuat Character Movement, Detect Ground, Jumping, & Camera Movement Tidak Termasuk Animasi Karakter

### A. Membuat Prgerakan Player

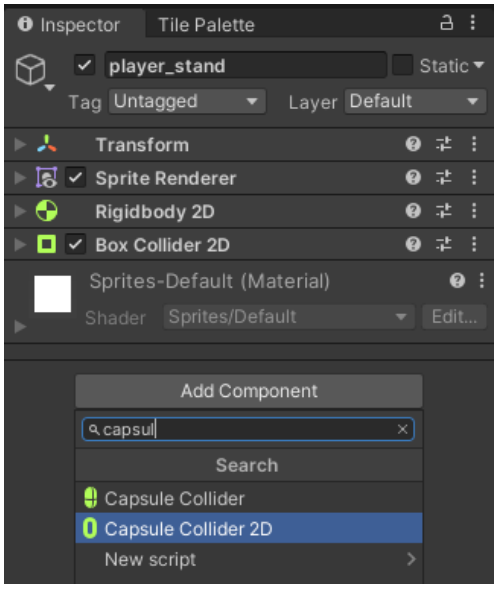
1. Buka project yang sudah dibuat pada bab sebelumnya, berikut tampilannya.



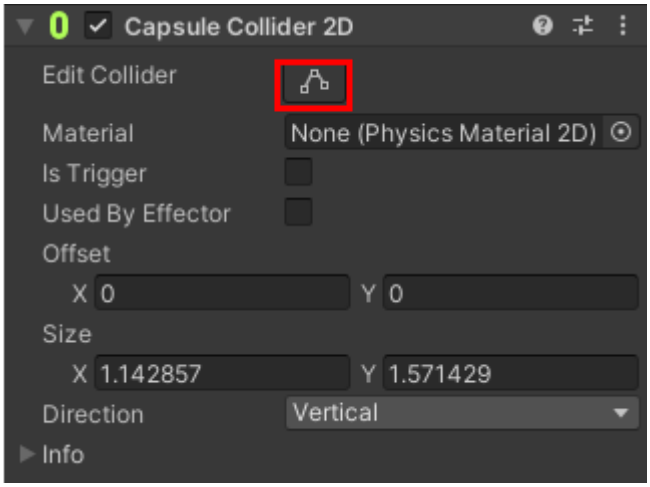
2. Tambahkan komponen RigidBody 2D untuk asset player_stand dan checklist bagian freeze rotation untuk sumbu Z



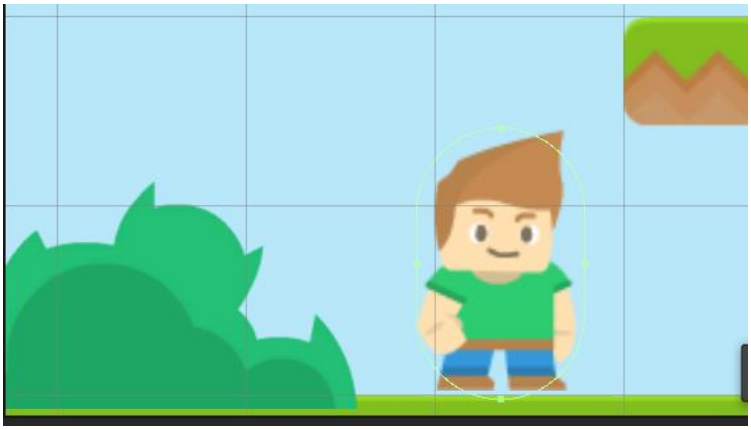
3. Tambahkan juga komponen Capsule Collider 2D pada asset player juga



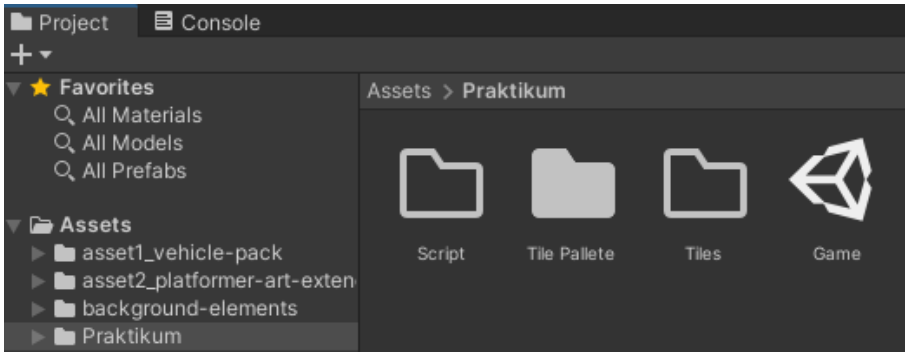

4. Klik pada bagian edit collider



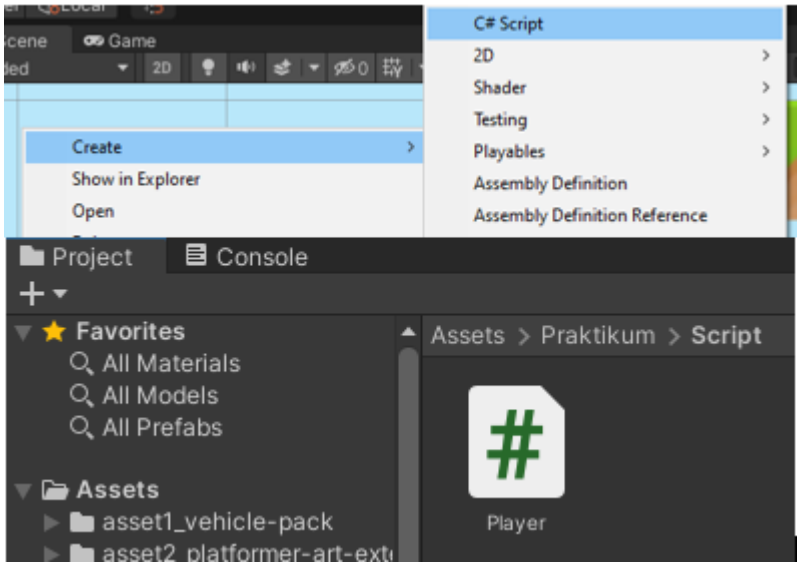

5. Sesuaikan ukuran capsule-nya dengan ukuran player seperti berikut



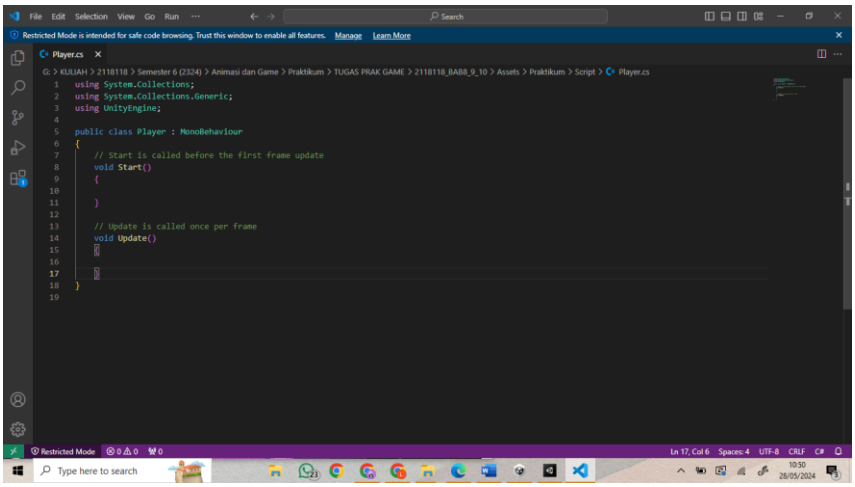
6. Buat folder baru bernama Script di dalam folder Praktikum




7. Pada folder script buat file C# Script dengan cara klik kanan pada folder kemudian pilih Create lalu C# Script dan beri nama Player



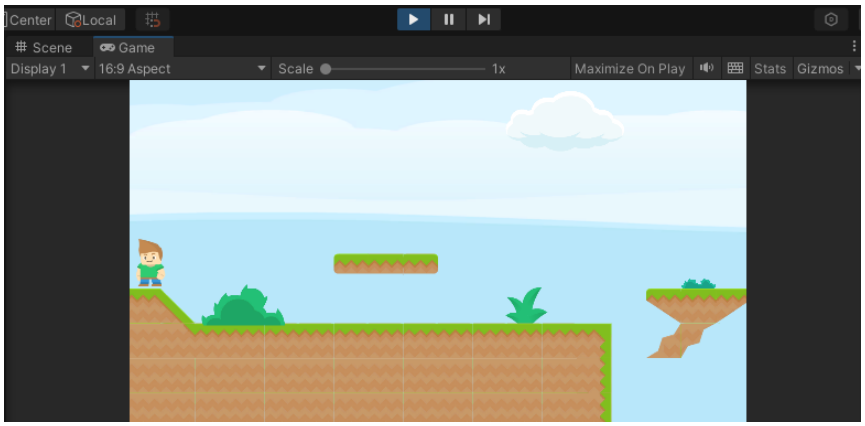

8. Drag and drop script player ke dalam asset player pada jendela hierarchy. Kemudian, klik 2 kali pada file script player, maka akan otomatis menuju ke halaman editor script.



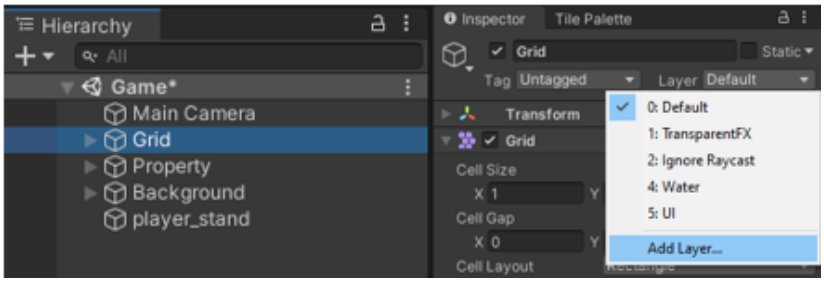
9. Masukkan script berikut pada file player

```
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 mencobanya, bisa klik button play dan gunakan keyboard arrow kanan kiri untuk menggerakan player

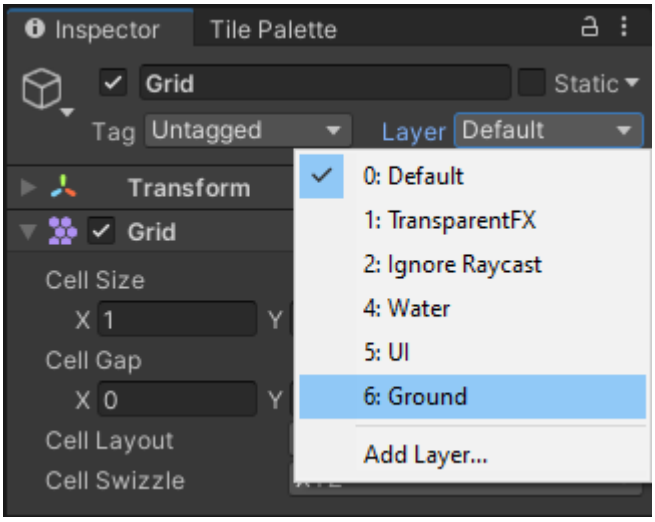



11. Klik layer grid, lalu pada jendela inspector pilih add layer

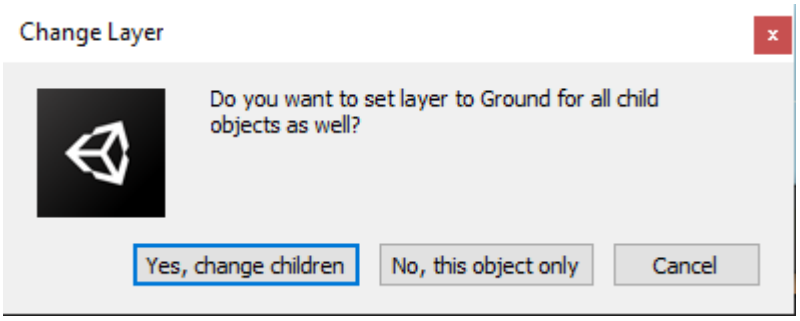



12. Isi user layer 6 dengan nama Ground

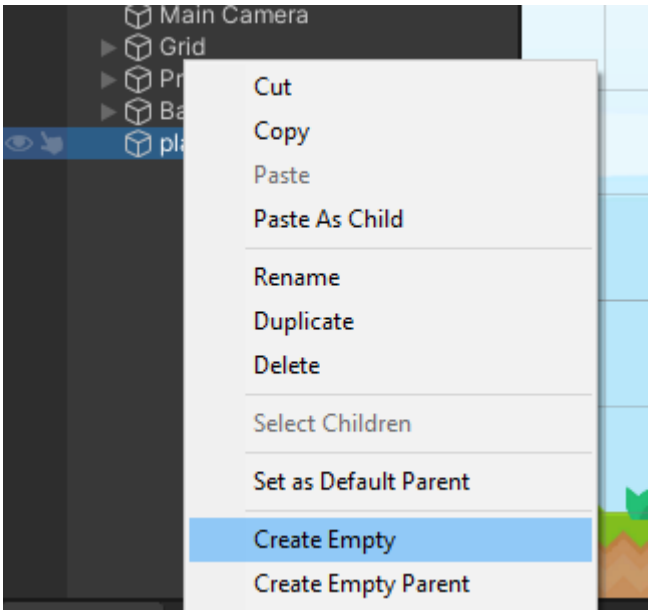
13. Klik lagi layer grid, kemudian pilih layer ground pada jendela inspector



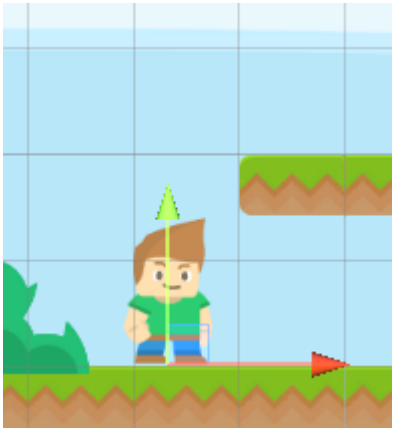

14. Jika muncul message seperti berikut, klik yes.




15. Buat layer baru di dalam layer player dengan nama GroundCheck



16. Klik GroundCheck, lalu gunakan Move Tools untuk memindahkannya ke bawah seperti berikut



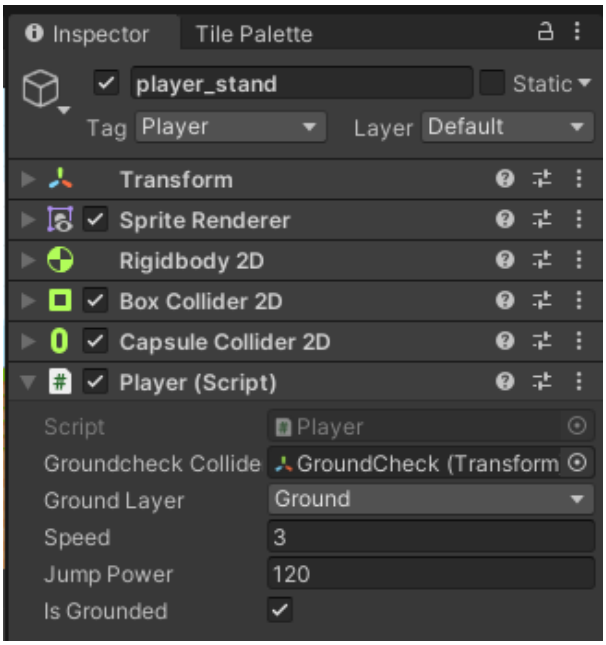
17. Tambahkan source code berikut pada file script player

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

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

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

18. Buat fungsi void GroundCheck() dengan isi sebagai berikut, lalu panggil fungsi tersebut 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;
  }
```

19. Pada jendela inspector untuk layer player, di bagian player script sesuaikan sebagai berikut




20. Tambahkan source code berikut pada file script player

```
[SerializeField] float jumpPower = 100;
bool jump;
```

21. Tambahkan source berikut pada void Update()

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

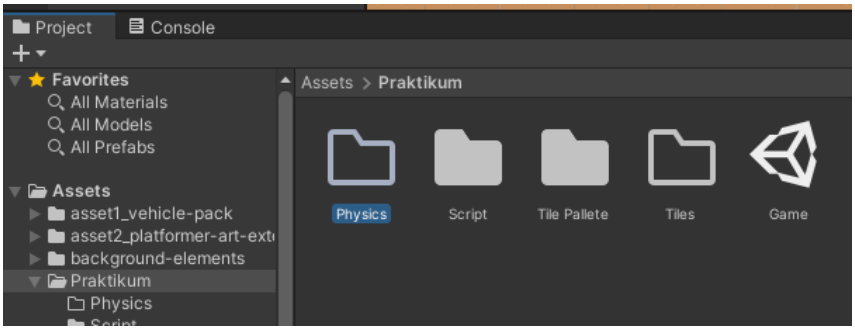
22. Pada void FixedUpdate(), bagian Move tambahkan parameter jump

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

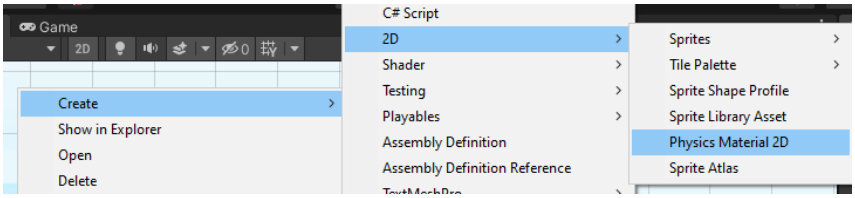
23. Tambahkan parameter 'bool jumpflag' pada void Move() dan tambahkan isinya dengan source code berikut.

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

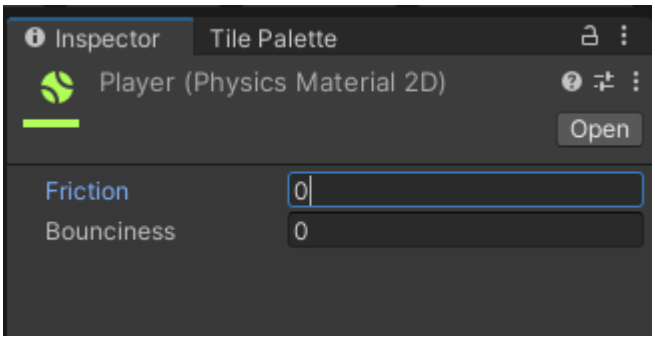
24. Pada folder praktikum create folder baru dengan nama Physics



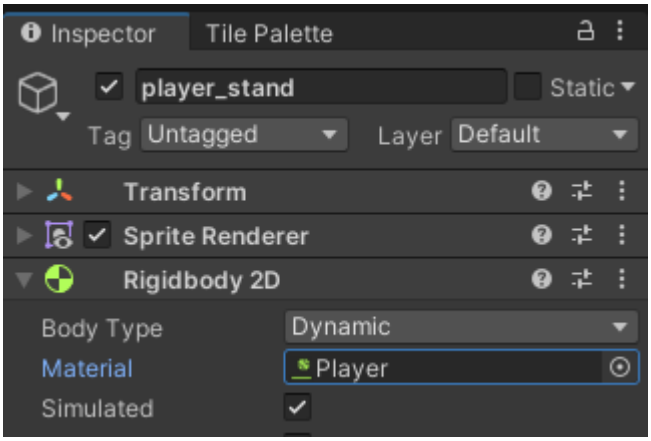

25. Pada folder physics create 2D pilih physics material 2D



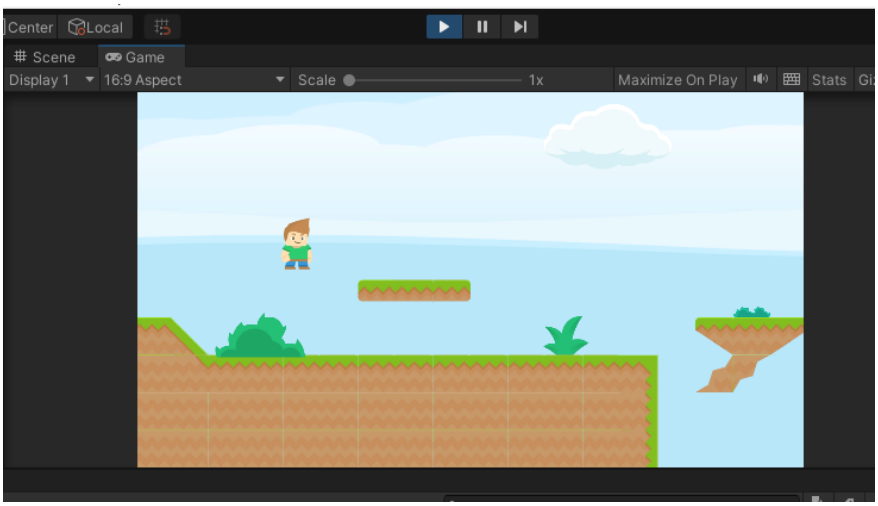

26. Klik material tersebut dan ubah bagian friction serta bouncies menjadi 0.




27. Kembali pada layer asset player, pada komponen rigid body pilih material yang sudah dibuat tadi yaitu player

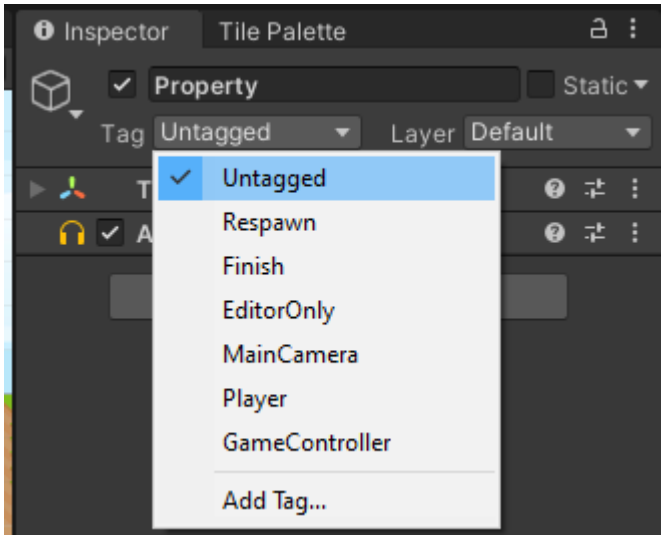


28. Berikut merupakan tampilan player jump jika berhasil.

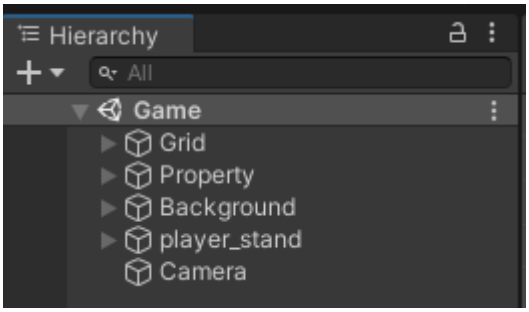



## B. Camera Movement

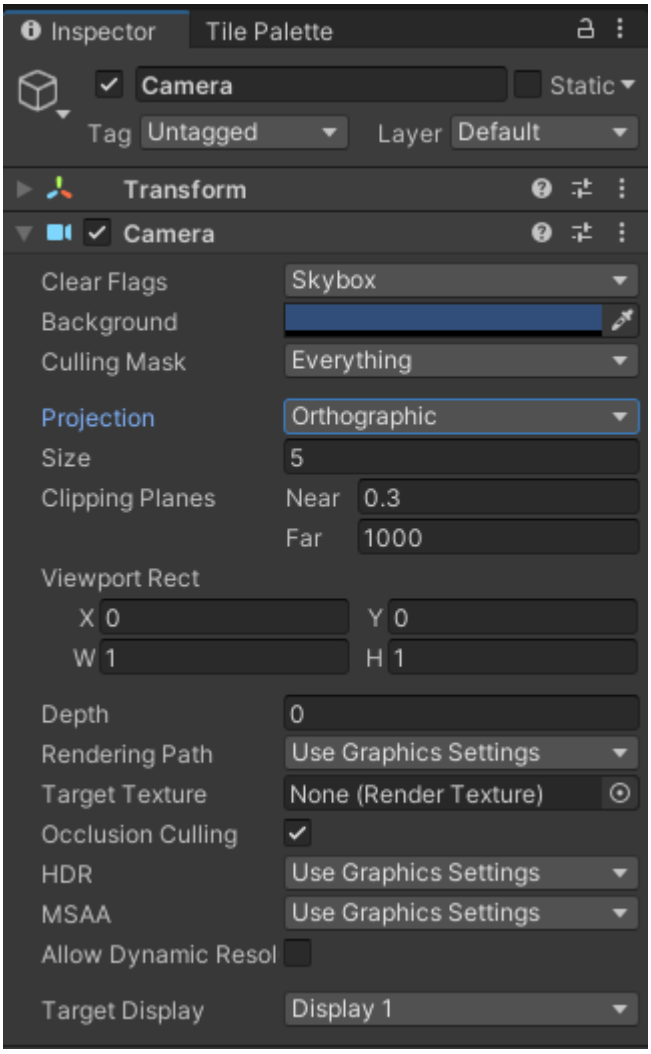
1. Pada layer property ubah tag menjadi untagged, serta hapus komponen camera.



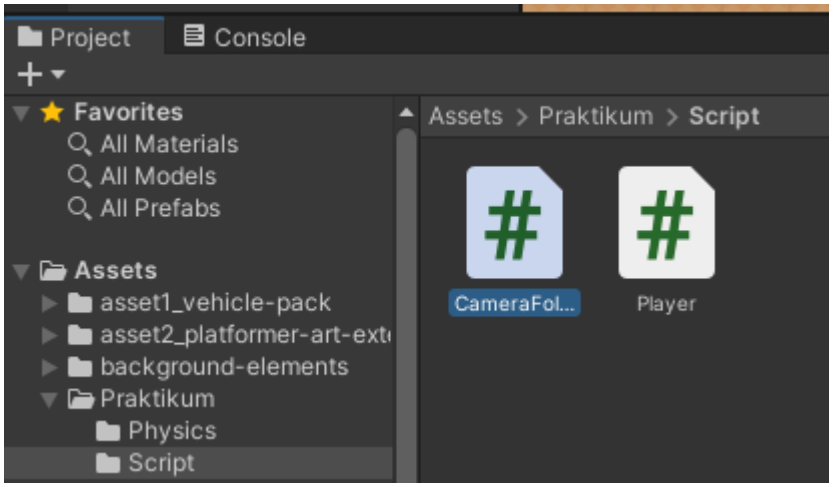

2. Buat layer baru pada jendela hierarchy dengan nama Camera



3. Pada layer camera sesuaikan komponen camera seperti berikut




4. Buat file script baru pada folder script dengan nama CameraFollow.

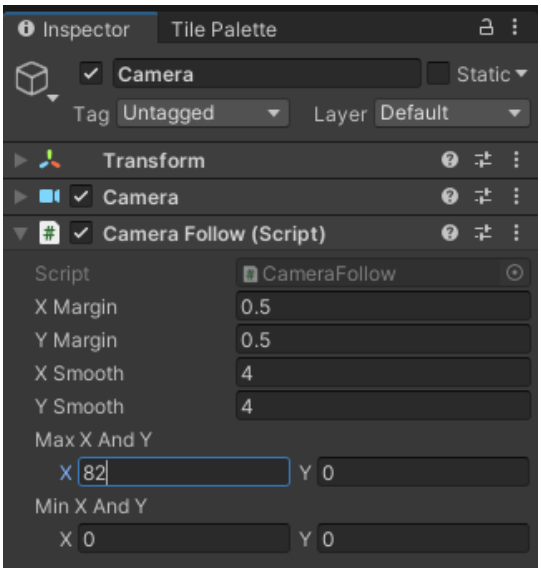



5. Masukkan source code berikut pada file script CameraFollow

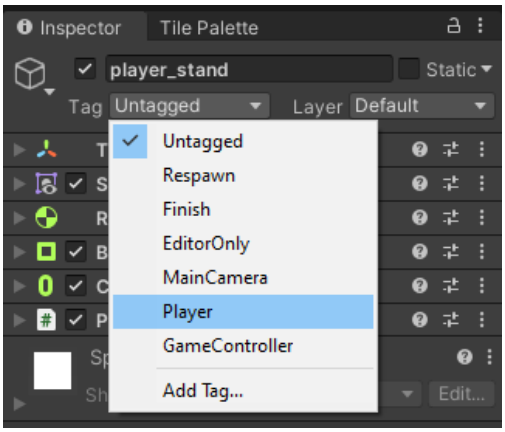
```
using System.Collections;
using System.Collections.Generic;
```

```
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);
    }
}
```

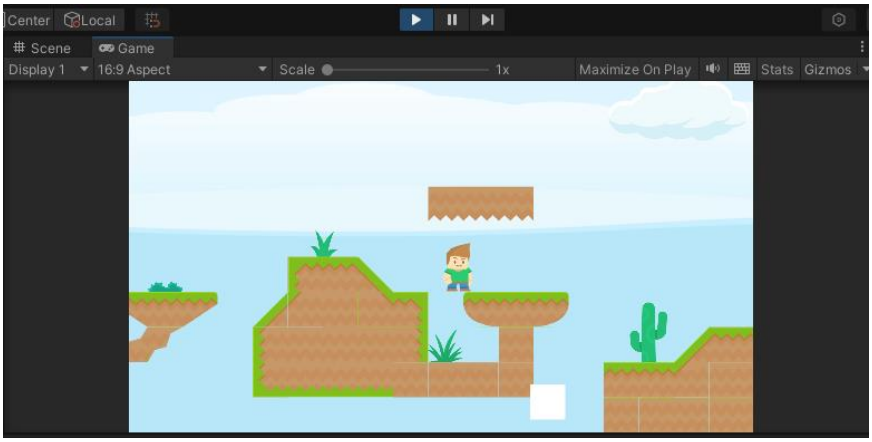
6. Drag and drop file script CameraFollow pada layer camera. Kemudian, pada jendela inspector ubah bagian Max X dan Y seperti berikut.




7. Untuk layer asset player ubag tag menjadi Player

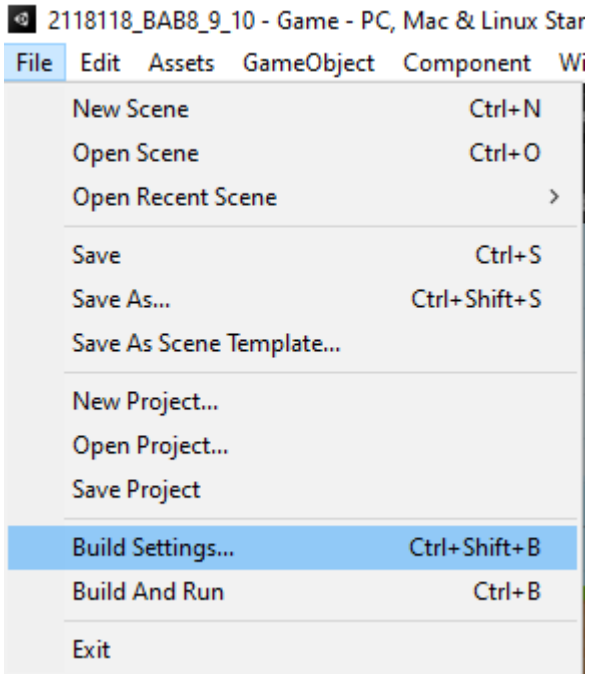



8. Berikut merupakan hasil tampilan dari camera movement

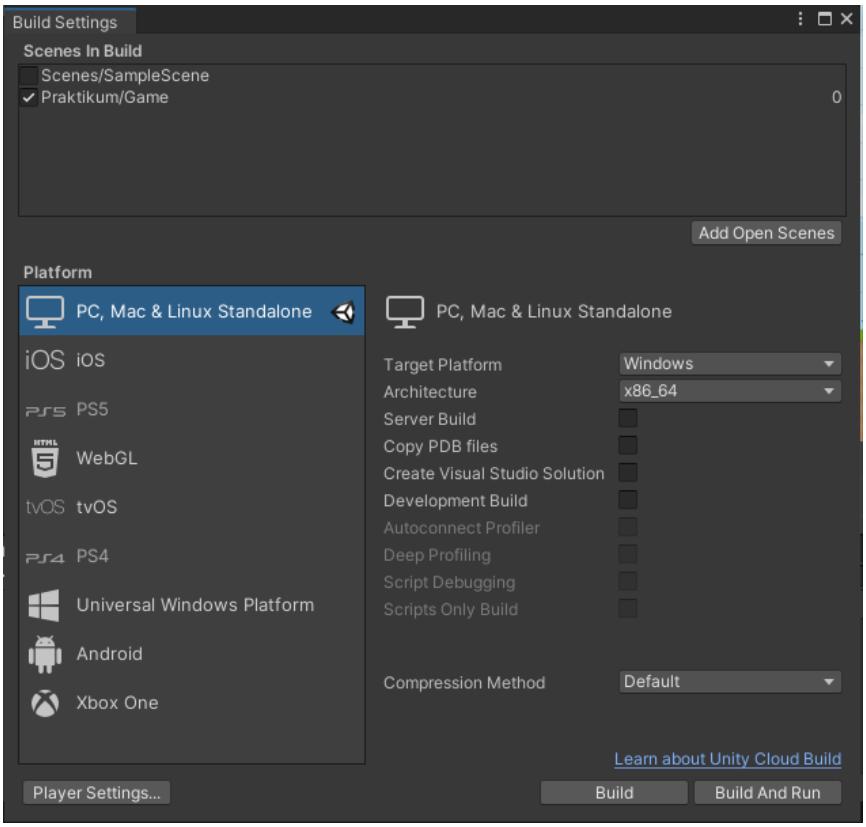


## C. Rendering

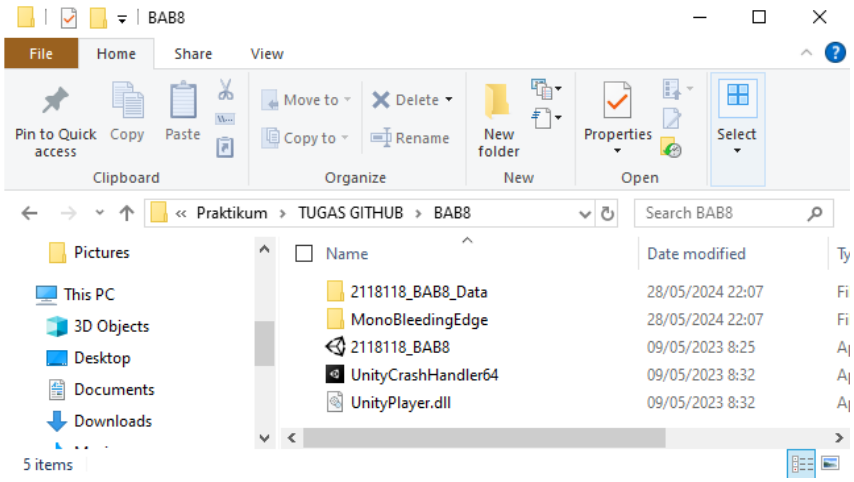
1. Klik menu file lalu pilih build setting atau gunakan shortcut Ctrl+Shift+B




2. Atur scene mana yang akan dirender dan juga spesfifikasinya. Jika sudah, klik button build



3. Jika sudah berhasil render, maka akan tersimpan file-file berikut.




## D. 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 source :

Script digunakan untuk mengatur kamera agar mengikuti pergerakan pemain pada sumbu x. Script ini mendefinisikan sebuah kelas bernama CameraFollow yang merupakan turunan dari MonoBehaviour, sebuah kelas dasar untuk semua script yang dapat dipasang pada objek dalam Unity. Di dalam kelas ini, terdapat variabel player bertipe Transform yang ditandai dengan atribut [SerializeField], yang memungkinkan variabel diatur melalui Inspector di Unity editor, meskipun variabel tersebut bersifat privat. Pada void Update(), yang dipanggil setiap frame, posisi kamera diatur ulang. Kamera akan mengambil posisi x dari pemain (player.position.x) sementara posisi y dan z kamera tetap tidak berubah (transform.position.y dan transform.position.z). Dengan demikian, kamera akan mengikuti pergerakan pemain hanya pada sumbu x, memastikan bahwa pemain selalu berada dalam pandangan kamera saat bergerak secara horizontal.

## E. Link Github Pengumpulan

https://github.com/Legming-DA/2118118_PRAK_ANIGAME.git