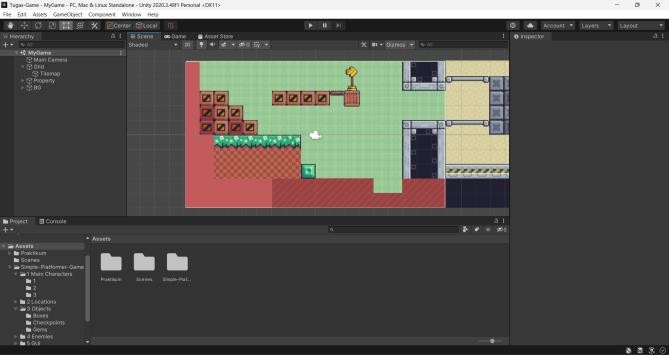
TUGAS PERTEMUAN: 8 CAMERA & CHARACTER MOVEMENT

|  |  |  |
| --- | --- | --- |
| **NIM** | : | 1918027 |
| **Nama** | : | Dirgarianda R C Rumabar |
| **Kelas** | : | D |
| **Asisten Lab** | : | M. RAFI FADDILANI (2118114) |

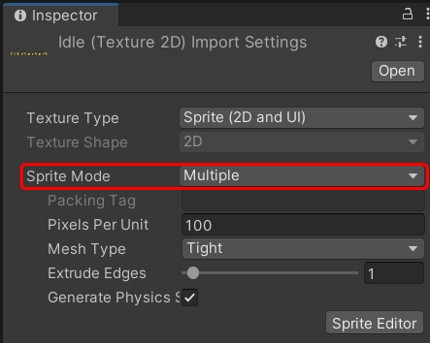
# Tugas 1 : Membuat Camera & Character Movement

* + 1. **Membuat Karakter Bergerak**
       1. Buka *project* Unity sebelumnya yang telah ditambahkan *tilemap*.



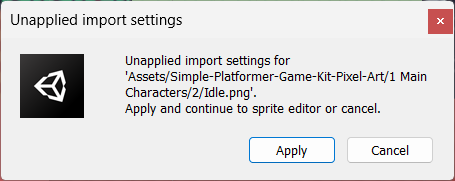
## Gambar 8.1 Membuka File Project

* + - 1. Pilih karekter dengan animasi *idle* pada *folder Main Characters*. Lalu ubah *Sprite Mode* menjadi *Multiple*.



## Gambar 8.2 Mengubah Sprite Mode

* + - 1. Kemudian masuk ke dalam *Sprite Editor*, jika tampil *window* seperti berikut pilih *Apply*.



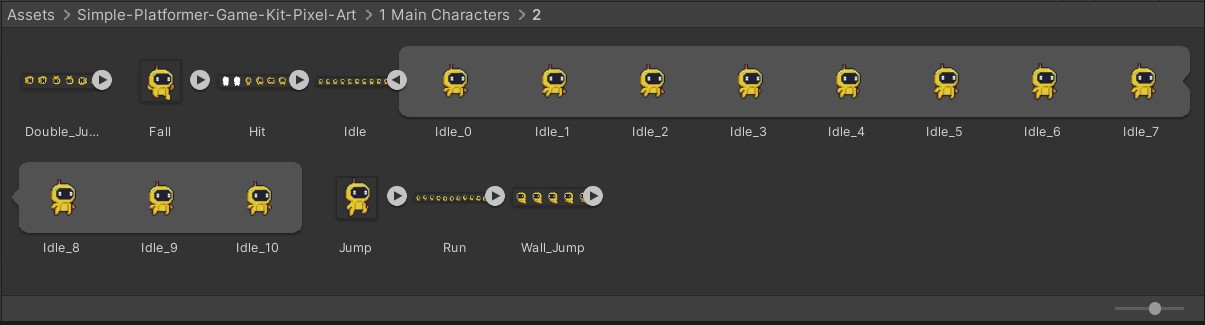
## Gambar 8.3 PopUp Unapplied Import

* + - 1. Kemudian lakukan *slicing* karakter dengan memilih *Type* menjadi

*Automatic* lalu pilih *Slice*.

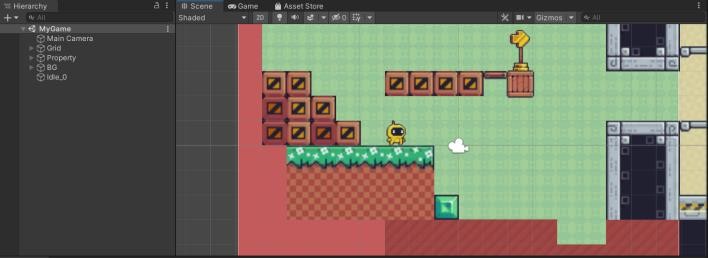
## Gambar 8.4 Melakukan Slicing

* + - 1. Kemudian pilih *Apply*, maka gambar akan terpisah seperti pada gambar berikut.



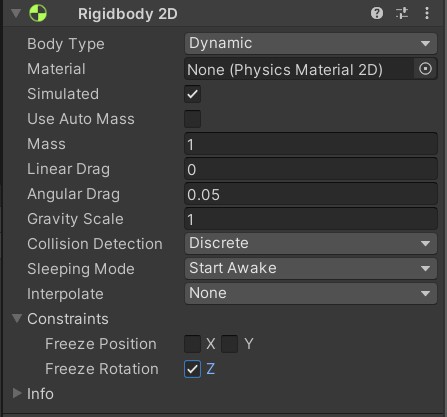
## Gambar 8.5 Hasil Slicing

* + - 1. Tambahkan karakter bernama Idle\_0, Import ke dalam hirarki dan posisikan karakter di posisi awal.



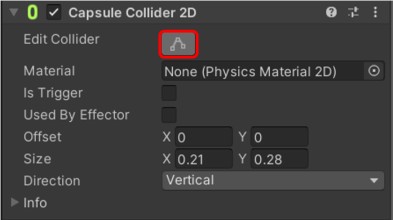
## Gambar 8.6 Menambahkan Karakter

* + - 1. Klik karakter Idle\_0, tambahkan komponen *Rigidbody* 2D. Sesuaikan pengaturannya seperti gambar berikut, centang pada *Freeze Rotation* Z.



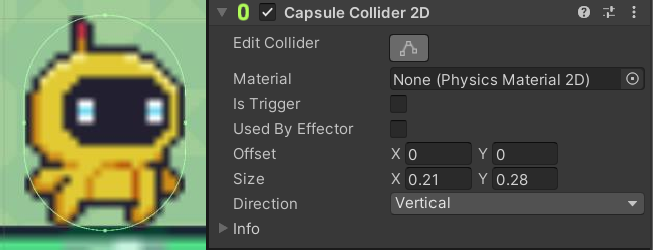
## Gambar 8.7 Menambah Komponen Rigidbody 2D

* + - 1. Kemudian tambahkan komponen *Capsule Collider* 2D pada Idle\_0, lalu klik *icon* di samping kanan *Edit Collider*.

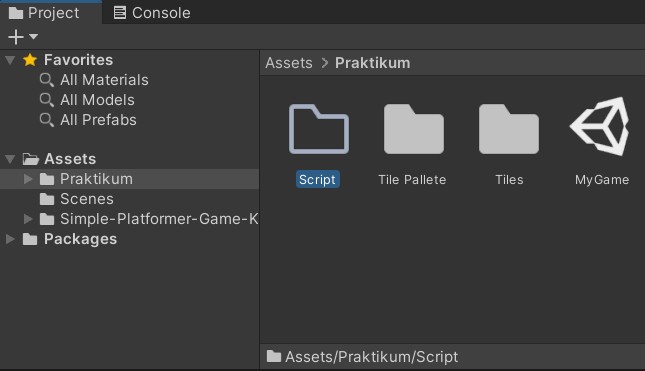


## Gambar 8.8 Menambah Komponen Capsuel Collider 2D

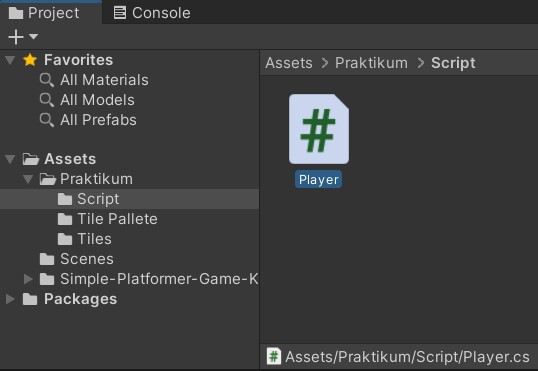
* + - 1. Lalu sesuaikan garis oval dengan karakter atau bisa di-*input Offset* X,Y dan juga *Size* X,Y.



## Gambar 8.9 Membuat Pallete Baru

* + - 1. Buka *folder* Praktikum, lalu buat *folder* baru dengan nama *Script*.

## Gambar 8.10 Membuat Folder Baru

* + - 1. Masuk ke dalam *folder Script*, lalu buat C# *Script* dengan nama *Player*.

## Gambar 8.11 Membuat File Baru

* + - 1. Masukkan *source code* di bawah dan pastikan nama *class* sama dengan nama *file*.

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 bergerak kanan kiri float xVal = dir \* speed \* 100 \*

Time.fixedDeltaTime;

Vector2 targetVelocity = new Vector2(xVal, rb.velocity.y);

rb.velocity = targetVelocity;

if (facingRight && dir < 0)

{

transform.localScale = new Vector3(-4, 4, 4); facingRight = false;

}

else if (!facingRight && dir > 0)

{

transform.localScale = new Vector3(4, 4, 4); facingRight = true;

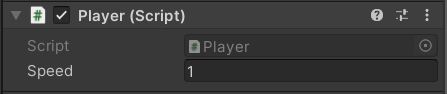
}

#endregion

}

}

* + - 1. Kemudian *drag* & *drop script* Player ke dalam hirarki Idle\_0, maka pada Idle\_0 akan ditambahkan *script*.



## Gambar 8.12 Menambahkan Script

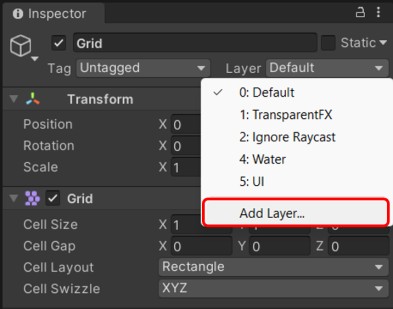
* + - 1. Coba jalankan *project* untuk mencoba *source code* di atas. Tekan *key* ‘A’ atau ‘*Left Arrow*’ untuk ke arah kiri dan tekan *key* ‘D’ atau ‘*Right Arrow*’ untuk ke arah kanan.



Gambar 8.13 Mencoba Menggerakkan Karakter

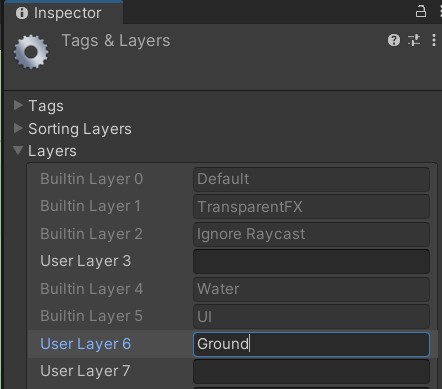
# Membuat Detect Ground

* + - 1. Klik *Grid* pada hirarki, lalu pilih *Layer* pada *Inspector*, klik *Add Layer*.



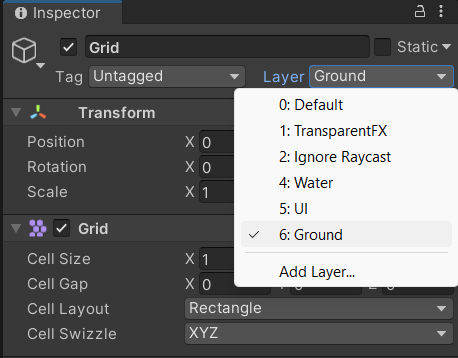
## Gambar 8.14 Menambahkan Layer

* + - 1. Kemudian pada *User Layer* 6 isi “Ground”.



## Gambar 8.15 Memberi Nama Layer Baru

* + - 1. Lalu pada hirarki *Grid*, ubah layer menjadi *Ground*. Kemudian jika muncul *pop up Change Layer*, pilih *Yes*.



## Gambar 8.16 Mengubah Layer

* + - 1. Pada Idle\_0 klik kanan, lalu *Create Empty* dengan nama *GroundCheck*.

## Gambar 8.17 Menambahkan Hirarki Baru

* + - 1. Klik pada hirarki *GroundCheck*, lalu gunakan *Move Tools* untuk memindahkan panah ke bagian bawah karakter seperti pada gambar berikut.



## Gambar 8.18 Memindahkan Panah Pada Karakter

* + - 1. Kembali ke *script Player* ubah *source code* setelah inisialisasi Rigidbody2D rb; menjadi seperti berikut.

Rigidbody2D rb;

[SerializeField] Transform groundcheckCollider; // source code baru

[SerializeField] LayerMask groundLayer; // source code baru

const float groundCheckRadius = 0.2f; // source code baru

[SerializeField] float speed = 1; float horizontalValue;

[SerializeField] bool isGrounded; // source code baru bool facingRight;

* + - 1. Ubah *source code* pada *function FixedUpdate* dan tambahkan *function GroundCheck* seperti berikut.

void FixedUpdate()

{

GroundCheck(); Move(horizontalValue);

}

void GroundCheck()

{

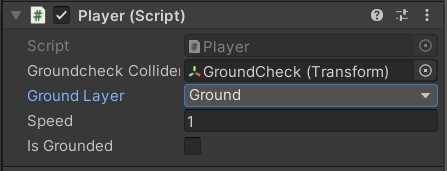
isGrounded = false; Collider2D[] colliders =

Physics2D.OverlapCircleAll(groundcheckCollider.position

, groundCheckRadius, groundLayer); if (colliders.Length > 0) isGrounded = true;

}

* + - 1. Klik Idle\_0, lalu pada *inspector* ke bagian *Player Script*, ubah bagian *Groundcheck Collider* dengan menekan *icon* dan pilih *GroundCheck* (*Transform*) dan pada *Ground Layer* pilih *Ground*.



Gambar 8.19 Mengatur Pada Bagian Script

# Membuat Fitur Karakter Melompat

* + - 1. Kembali ke *script Player* ubah *source code* setelah inisialisasi Rigidbody2D rb; menjadi seperti berikut.

Rigidbody2D rb;

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

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

code baru

float horizontalValue;

[SerializeField] bool isGrounded; bool facingRight;

bool jump; // source code baru

* + - 1. Ubah pada bagian *function Update* seperti berikut.

void Update ()

{

horizontalValue = Input.GetAxisRaw("Horizontal"); if (Input.GetButtonDown("Jump"))

jump = true;

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

}

* + - 1. Ubah pada bagian *function FixedUpdate* seperti berikut.

void FixedUpdate()

{

GroundCheck(); Move(horizontalValue, jump);

}

* + - 1. Ubah pada bagian *function Move* seperti berikut.

void Move(float dir, bool jumpflag)

{

if(isGrounded && jumpflag)

{

isGrounded = false;

jumpflag = false;

rb.AddForce(new Vector2(0f, jumpPower));

}

#region bergerak kanan kiri float xVal = dir \* speed \* 100 \*

Time.fixedDeltaTime;

Vector2 targetVelocity = new Vector2(xVal, rb.velocity.y);

rb.velocity = targetVelocity;

if (facingRight && dir < 0)

{

transform.localScale = new Vector3(-4, 4, 4); facingRight = false;

}

else if (!facingRight && dir > 0)

{

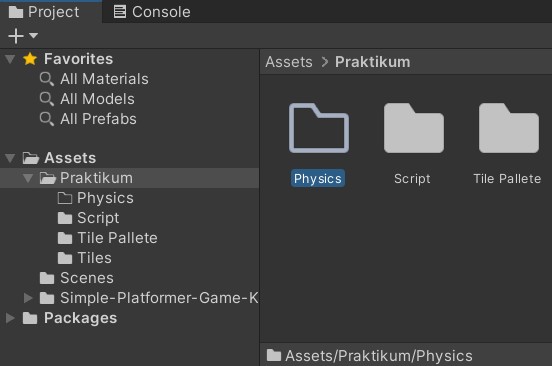
transform.localScale = new Vector3(4, 4, 4); facingRight = true;

}

#endregion

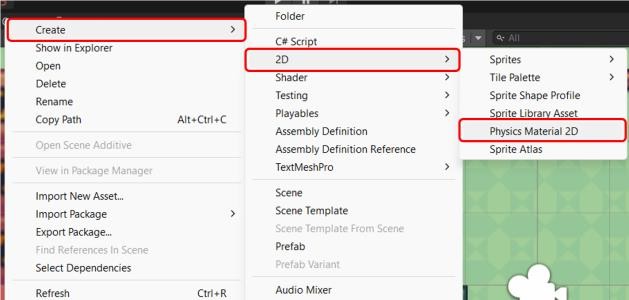
}

* + - 1. Pada *folder* Praktikum, buat *folder* baru dengan nama “*Physics*”.



## Gambar 8.20 Membuat Folder Baru

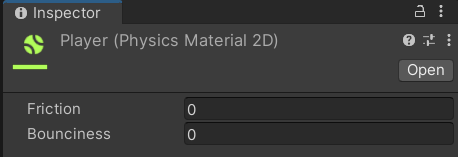
* + - 1. Masuk ke dalam *folder Physics*, lalu klik kanan pilih *Create*, kemudian 2D dan pilih *Physics Material* 2D serta beri nama *Player*.



## Gambar 8.21 Membuat Physics Material 2D

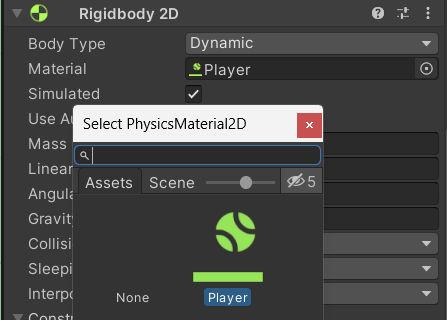
* + - 1. Kemudian klik *Player* (*Physics Material* 2D), ubah *Friction* dan

*Bounciness* menjadi 0 pada menu *Inspector*.



Gambar 8.22 Mengatur *Friction* Dan *Bounciness*

## Lalu pilih Idle\_0 pada hirarki, cari Rigidbody 2D pada menu *Inspector* lalu klik *icon* yang ada pada bagian Material, lalu ubah dengan *Physics Material* yang sebelumnya dibuat.



Gambar 8.23 Mengubah Material

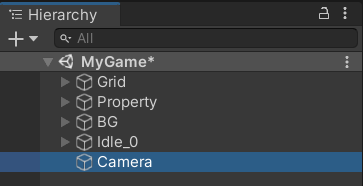
## Untuk mencobanya tekan *play*, kemudian agar karakter melompat tekan spasi.



Gambar 8.24 Mencoba Karakter Melompat

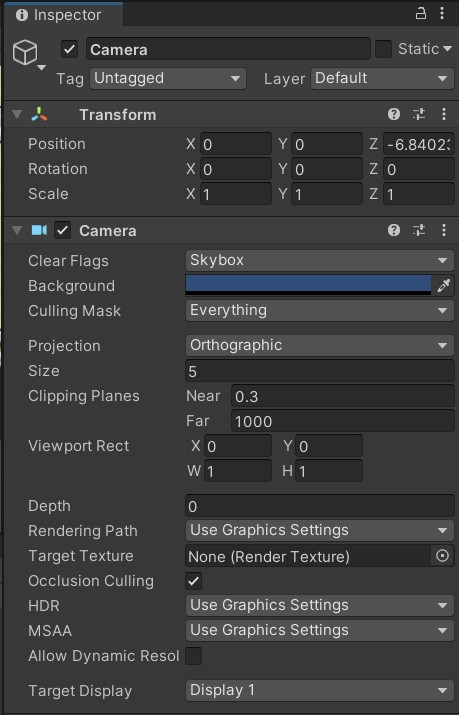
# Membuat Camera Movement

## Hapus hirarki *Main Camera*, lalu buat hirarki baru dan beri nama

*Camera*.

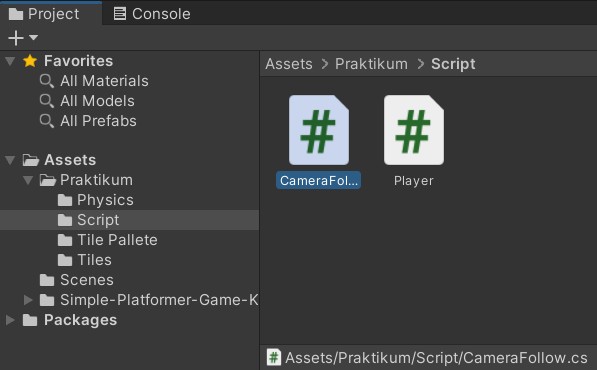
## Gambar 8.25 Membuat Hirarki Baru

* + - 1. Kemudian tambahkan komponen *Camera* pada hirarki *Camera*, lalu sesuaikan pengatur seperti gambar di bawah.



## Gambar 8.26 Menambahkan Komponen Camera

* + - 1. Buat *file script* baru pada *folder Script* dengan nama *CameraFollow*.



## Gambar 8.27 Membuat File Script

* + - 1. Lalu tulis *source code* berikut ini pada *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);

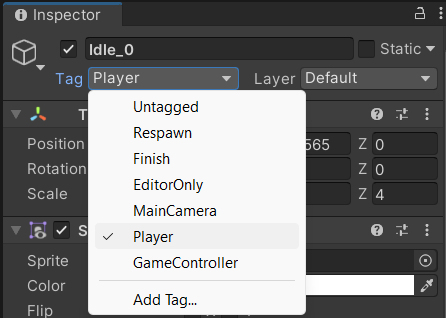
}

}

* + - 1. Selanjutnya *drag* & *drop script Camera Follow* ke hirarki *Camera*, lalu ubah nilai Max X pada menu *Inspector* hirarki *Camera*.



## Gambar 8.28 Menambahkan Script

* + - 1. Ubah *tag* pada Idle\_0 menjadi *Player*.

## Gambar 8.29 Mengubah Tag

* + - 1. Tekan *play* untuk menjalankan, maka kamera dapat mengikuti pergerakan dari karakter.



Gambar 8.30 Mencoba Camera Movement

# Link Github Pengumpulan

https://github.com/reydjav/1918027\_PRAK\_ANIGAME.git

# Kuis : Menjelaskan Source Code 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:

Pertama pada baris kode tersebut dilakukan inisialisasi *library* yang akan digunakan. Kemudian terdapat *class* dengan nama CameraFollow untuk menampung baris kode yang berisi *update* dari *player*. Di dalam *class* tersebut

## pertama-tama dideklarasi sebuah *field* bertipe *Transform* dengan nama *player* yang ditandai dengan atribut ‘[SerializeField]’. Kemudian dilanjutkan dengan sebuah *function* untuk membuat kamera mengikuti pergerakan secara horizontal *player*.