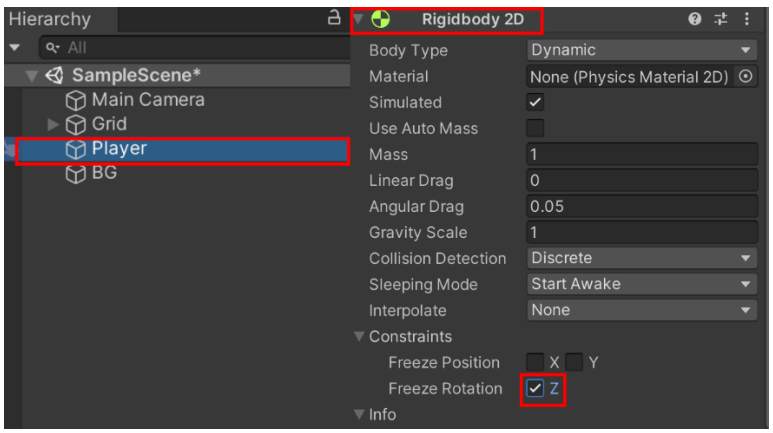
TUGAS PERTEMUAN: 8

Camera & Character Movement

NIM	:	2118105
Nama	:	Yonanda Haryono
Kelas	:	C
Asisten Lab	:	Rifal Rifqi Rhomadon

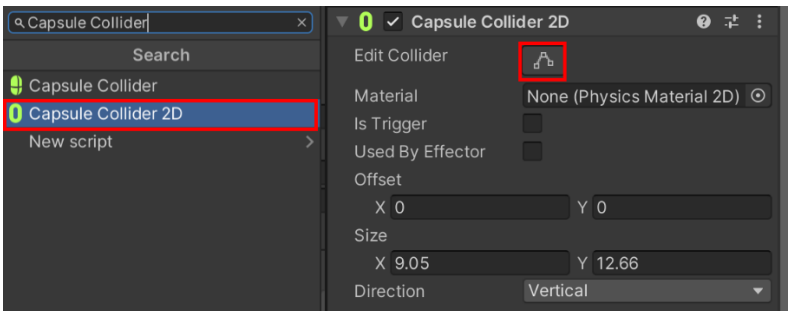
8.1 Tugas 8 : Membuat Character Movement, Detect Ground, Jumping, & Camera Movement Tidak Termasuk Animasi Karakter

1. Buka file projek Unity sebelumnya pada bab 7 untuk digunakan Kembali. Tambahkan player, Import kedalam hierarchy. Klik player tambahkan Component Rigidbody 2D, sesuaikan settingannya seperti gambar berikut, Centang pada Freeze Rotation Z.



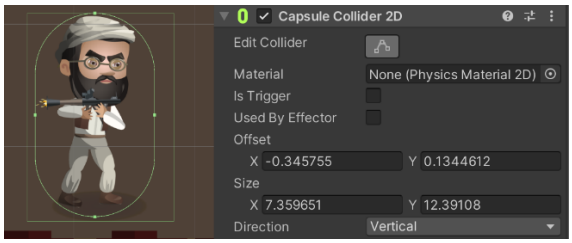
Gambar 8.1 Tambahkan Rigidbody 2D

2. Lalu tambahkan komponen Capsule Collider 2D di player, lalu klik icon sebelah kanan edit collider



Gambar 8.2 Menambahkan Capsule Collider 2D

3. Lalu sesuaikan dengan garis oval pada karakter atau bisa inputkan Offset X, Y dan juga Size X, Y nya

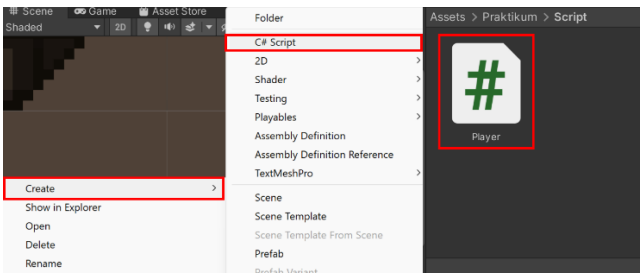


Gambar 8.3 Mengatur Garis Oval Pada Karakter

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

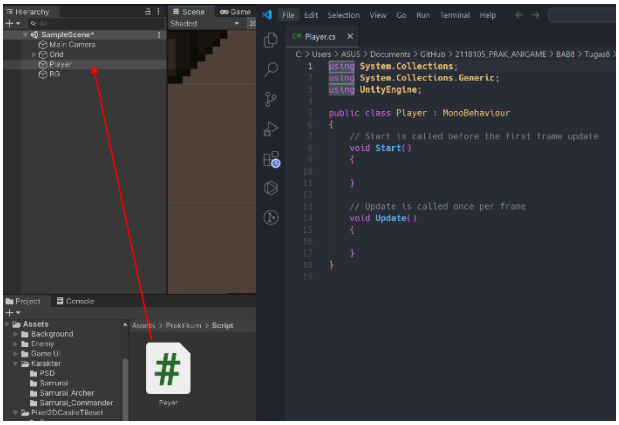
Gambar 8.4 Menambahkan Folder Script

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



Gambar 8.5 Membuat File Script Dengan Nama Player

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

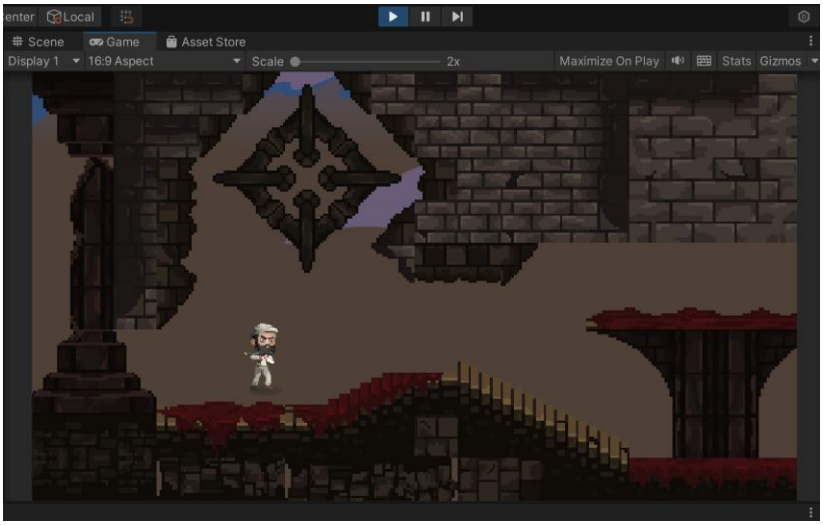


Gambar 8.6 Drag & Drop File Script Player

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

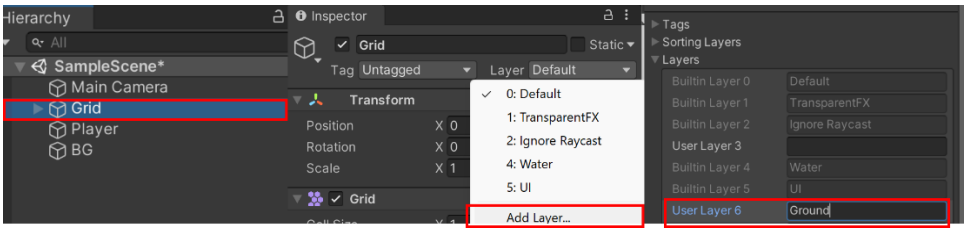
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

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(-0.13f, 0.13f,
0.13f);
      facingRight = false;
    }

    else if (!facingRight && dir > 0)
    {
      // ukuran player
      transform.localScale = new Vector3(0.13f, 0.13f,
0.13f);
      facingRight = true;
    }

    #endregion
  }
}
```

8. 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



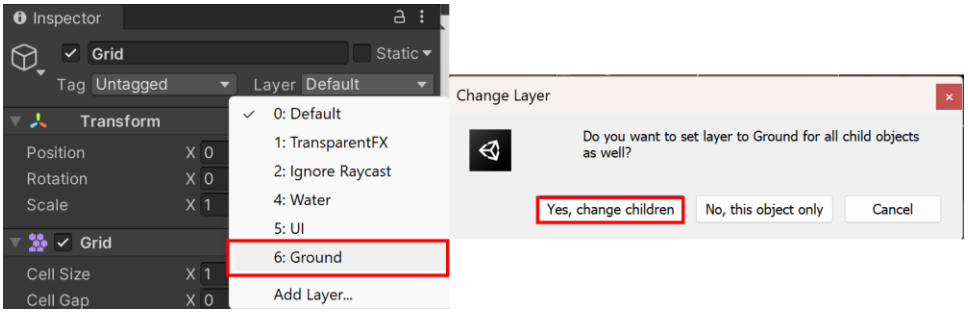
Gambar 8.7 Tampilan Saat Game Dijalankan

9. Untuk membuat player loncat menggunakan spasi, kita perlu membuat GorundCheck dengan cara, klik Grid pada Hierarchy, pergi ke inspector, pilih Layer, Klik Add Layer. Lalu isi “Ground” pada User Layer 6



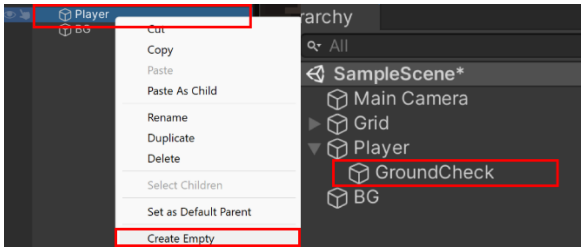
Gambar 8.8 Menambahkan Layer Ground Pada Hirrarchy Grid

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



Gambar 8.9 Mengubah Layer to Ground

11. Klik kanan pada player, lalu Create empty, beri nama GroundCheck



Gambar 8.10 Menambahkan GroundCheck

12. Klik pada Hirarki GroundCheck, lalu gunakan "Move Tools" untuk memindahkan ke bagian bawah Player seperti gambar berikut.

Gambar 8.11 Meletakkan Posisi GroundCheck

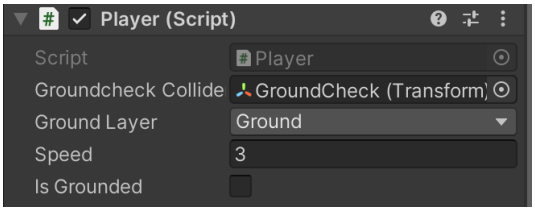
13. Kembali ke script Player tambahkan source code seperti ini

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

  const float groundCheckRadius = 0.2f; // +
  [SerializeField] float speed = 1;
  float horizontalValue;
  [SerializeField] bool isGrounded; // +
  bool facingRight;
```

14. Buat void ground check dibawah 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;
  }
```

15. Klik player, lalu ke inspector ke effect Player script di bagian "GroundCheck collider" tekan icon lalu pilih yang GroundCheck Transform, dan pada Ground Layer pilih Ground



Gambar 8.12 Tampilan Menambahkan GroundCheck Collider

16. Lalu untuk membuat player melompat tambahkan script berikut

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

float horizontalValue;

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

17. 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;
  }
```

18. Tambahkan juga jump pada parameter Move

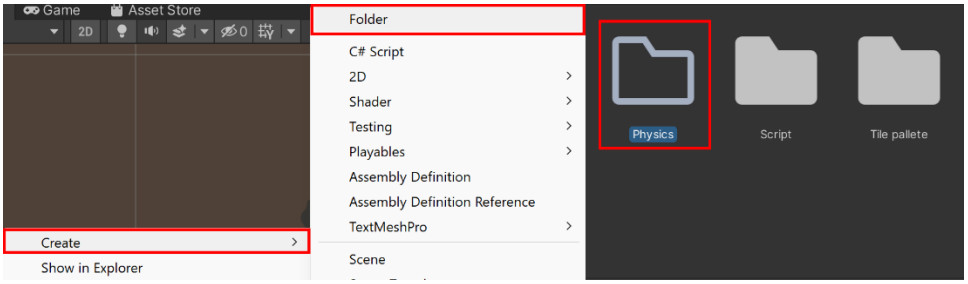
```
void FixedUpdate()
  {
    GroundCheck();
    Move(horizontalValue, jump);
  }
```

19. Tambahkan script berikut pada void Move

```
void Move(float dir, bool jumpflag)
  {
    if(isGrounded && jumpflag)
    {
        isGrounded = false;
        jumpflag = false;
        rb.AddForce(new Vector2(0f, jumpPower));
    }
    #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(-0.13f, 0.13f,
0.13f);
      facingRight = false;
    }

    else if (!facingRight && dir > 0)
    {
      // ukuran player
      transform.localScale = new Vector3(0.13f, 0.13f,
0.13f);
      facingRight = true;
    }

    #endregion
  }
```

20. Buat folder baru di praktikum bernama “Physics”

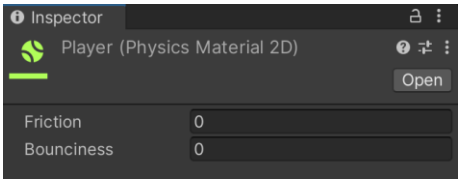


Gambar 8.13 Menambahkan Folder Physics

21. Didalam folder Pyshics create pilih 2d, pilih physical material 2d, berinama “Player”

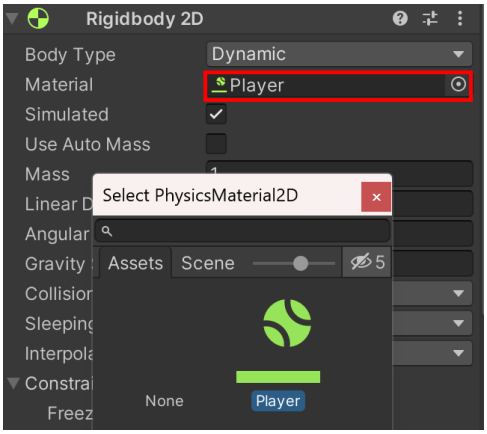
Gambar 8.14 Menambahkan Physics Material 2D

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



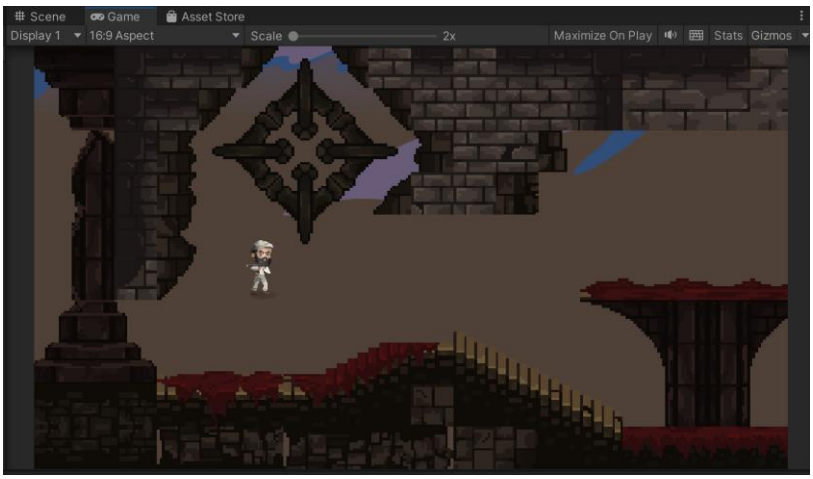
Gambar 8.15 Ubah Friction & Bounces

23. 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



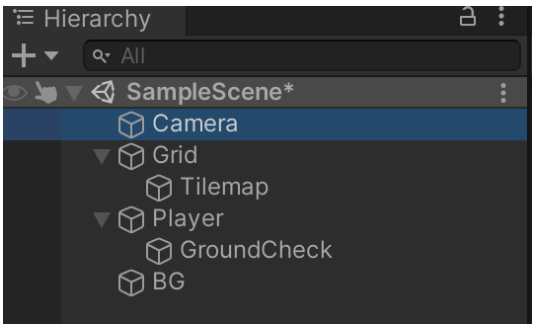
Gambar 8.16 Menambahkan Material Pada RigidBody 2D

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



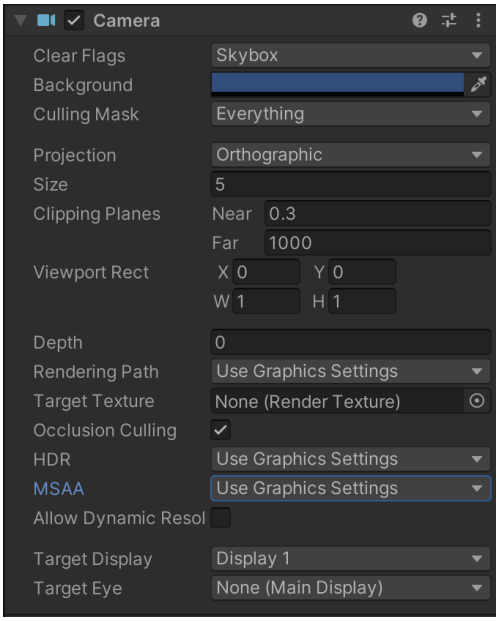
Gambar 8.17 Tampilan Ketika Karakter Melompat

25. Pada Hirarki, Main Camera Rename Menjadi Camera



Gambar 8.18 Tampilan Hirarki Camera

26. Sesuaikan Setting Layer Camera seperti gambar dibawah ini



Gambar 8.19 Tampilan Setting Layer Camera

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

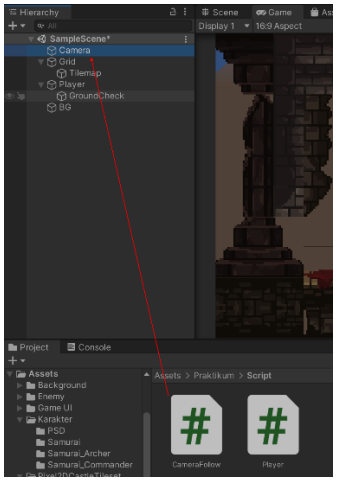
Gambar 8.20 Tampilan Membuat Script CameraFollow

28. 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;
    }
```

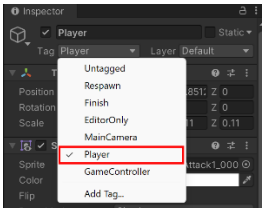
```
    bool CheckXMargin()
    {
        return       Mathf.Abs(transform.position.x       -
player.position.x) > xMargin;
    }

    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);
    }
}
```

29. Drag & drop script CameraFollow Kedalam Layer Camera



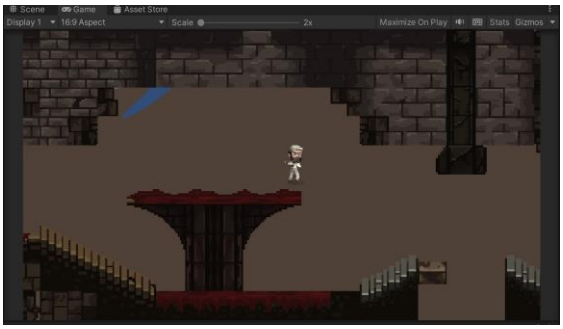
Gambar 8.21 Tampilan Drag & Drop Script CameraFollow

30. Ubah tag di player Untagged menjadi ”Player”



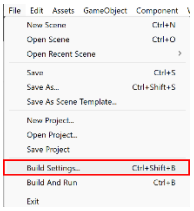
Gambar 8.22 Tampilan Mengubah Tag Menjadi Player

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



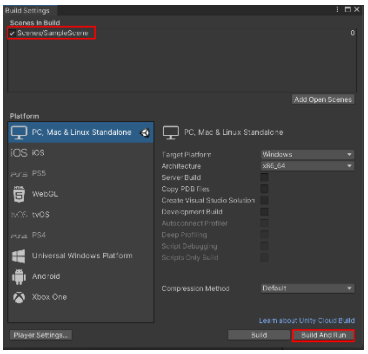
Gambar 8.23 Tampilan Ketika Script CameraFollow Berhasil

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



Gambar 8.24 Tampilan Build Settings

33. Pada Setting Build ini pilih PC, Mac & Linux, Tekan Build, pastikan pada menu Scene in Build berada pada project Tugas, Kemudian tekan build and run



Gambar 8.25 Tampilan Untuk Build & Run

8.2 Kuis CameraFollow

Menjelaskan Source code dibawah ini

```
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

Skrip CameraFollow ditulis dalam C# dan memerlukan penggunaan UnityEngine. Objek kamera memiliki referensi ke posisi pemain melalui variabel player yang ditentukan sebagai Transform dan diatur melalui SerializeField, sehingga dapat diatur dari editor Unity tanpa harus menjadikan variabel tersebut publik. Di dalam metode Update, yang dipanggil setiap frame, posisi kamera diperbarui menjadi posisi baru di mana komponen x dari posisi kamera diatur agar selalu sama dengan komponen x dari posisi pemain, sementara komponen y dan z tetap tidak berubah. Dengan cara ini, kamera akan selalu mengikuti pemain secara horizontal saat pemain bergerak, menjaga sumbu vertikal dan kedalaman tetap konstan.