TUGAS PERTEMUAN: 8

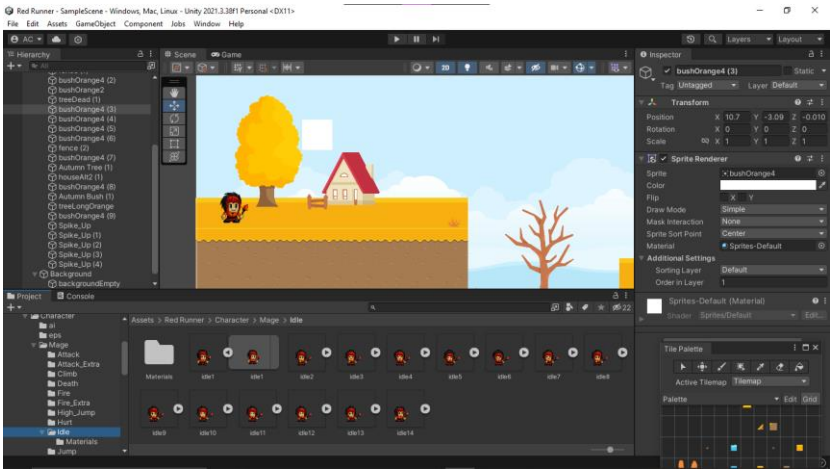
CAMERA & CHARACTER MOVEMENT

NIM	:	2118114
Nama	:	Ardhea Dwi Cahyani
Kelas	:	C
Asisten Lab	:	Nayaka Apta Nayottama (2218102)

1.1 Tugas 1 : Membuat Langkah-langkah pada Penerapan Camera dan Character Movement

A. Membuat Pergerakan Layar

1. Buka project sebelumnya yaitu pada bab 7.

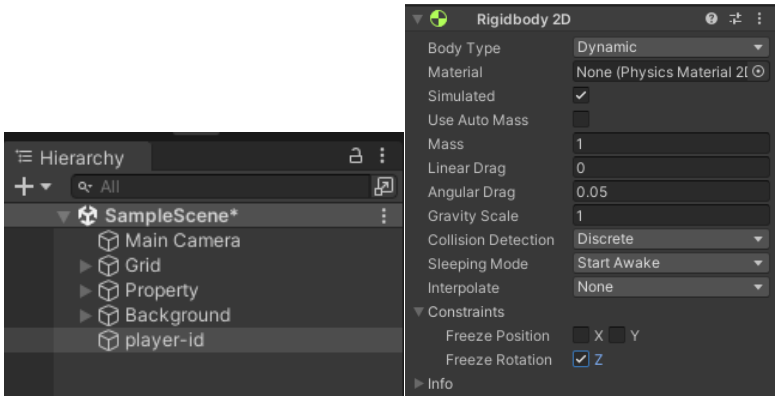


Gambar 8.1 Open File Project Bab 7

2. Pilih karakter idle1, kemudian import ke dalam hirarki. Kemudian rename menjadi player-id.

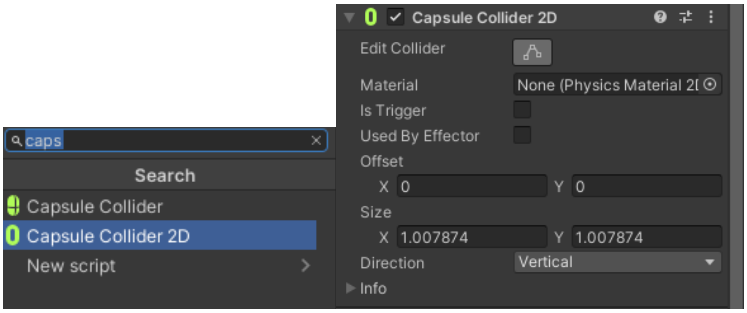
Gambar 8.2 Import Karakter ke Hirarki

3. Klik player-id, kemudian menuju ke Inspector dan tambahkan komponen baru yaitu Rigidbody 2D. Centang pada bagian Freeze Rotation Z.



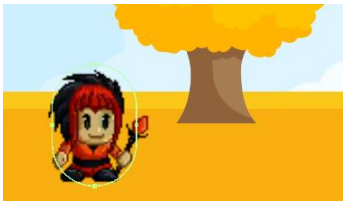
Gambar 8.3 Add Rigidbody 2D Component

4. Tambahkan juga komponen yaitu Capsule Collider 2D.



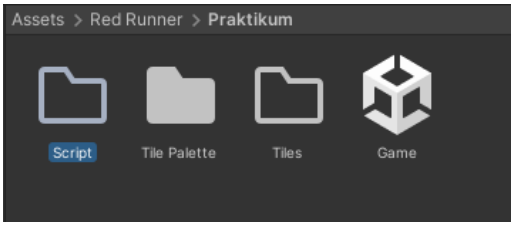
Gambar 8.4 Add Capsule Collider 2D Component

5. Sesuaikan ukurannya dengan objek karakter yang digunakan.



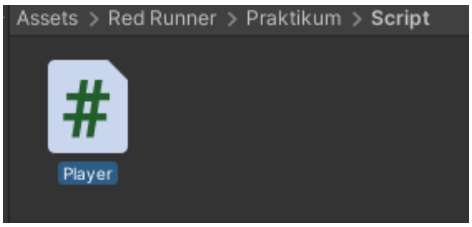
Gambar 8.5 Menyesuaikan Ukuran Capsule Collider

6. Tambahkan folder baru yaitu “Script” pada folder Praktikum.



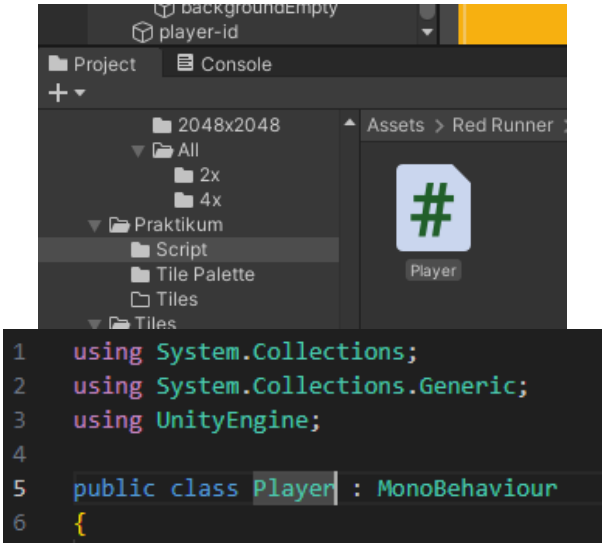
Gambar 8.6 Menambahkan Folder Script

7. Masuk ke dalam folder Script, kemudian buat file C# Script dan beri nama “Player”.



Gambar 8.7 Membuat file C# Player

8. Drag and drop file C# ke player_id yang ada di hirarki. Setelah itu klik dua kali pada file script Player, maka akan diarahkan ke Text Editor.



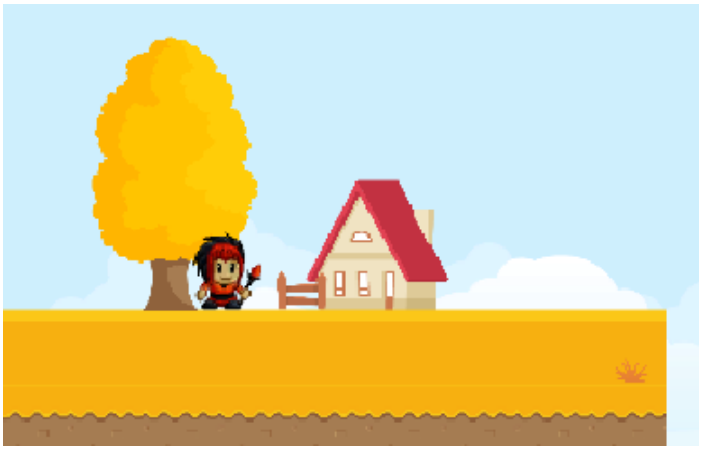
Gambar 8.8 Drag and Drop File C# Player ke player-id

9. Masukkan source code di bawah ini untuk memberikan animasi berjalan ke kanan dan ke kiri pada karakter.

```
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 ()
```

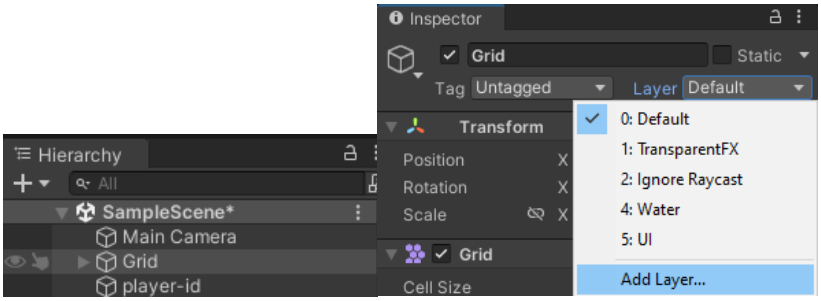
```
  {
    horizontalValue = Input.GetAxisRaw("Horizontal");
  }

  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 berhasil atau tidak, maka Play terlebih dahulu kemudian tekan “a” atau “left arrow” untuk ke arah kiri, tekan “d” atau “right arrow” untuk ke arah kanan.



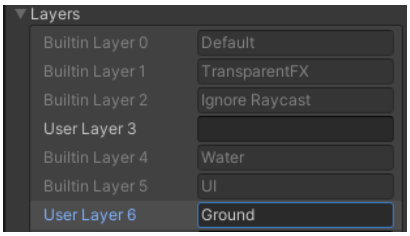
Gambar 8.9 Pengujian Source Code

11. Selanjutnya karakter akan dibuat bisa melompat dengan menggunakan spasi. Caranya yaitu membuat GroundCheck dengan klik Grid pada hirarki, kemudian pada Inspector terdapat Layer, pilih Add Layer.



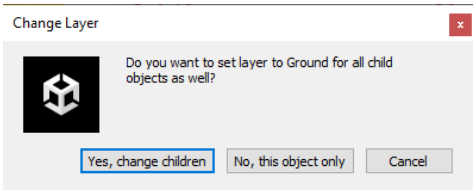
Gambar 8.10 Add Layer pada Grid

12. Isikan User Layer 6 dengan “Ground”.



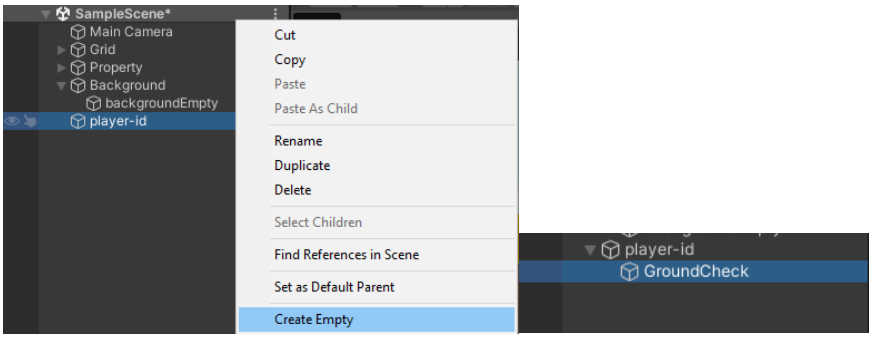
Gambar 8.11 Membuat Layer 6 Menjadi Ground

13. Jika muncul pop up seperti ini maka klik Yes, change children. Kemudian ubah Layer menjadi Ground.



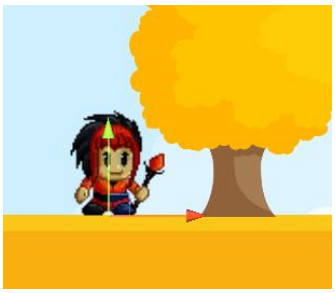
Gambar 8.12 Klik Yes

14. Klik kanan pada player-id, kemudian Create Empty. Dan beri nama menjadi GroundCheck.



Gambar 8.13 Membuat GroundCheck

15. Klik pada hirarki GroundCheck, kemudian gunakan Move Tools untuk memindahkan ke bagian bawah Player.



Gambar 8.14 Memindahkan Ground Check ke Bagian Bawah Player

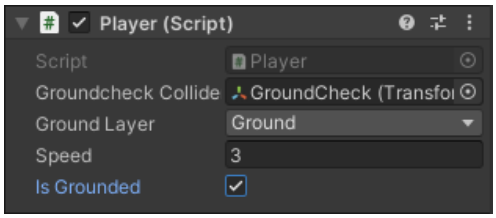
16. Tambahkan source code berikut pada script Player.

```
[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 GroundCheck(); pada void fixedUpdate.

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

void GroundCheck()
{
isGrounded = false;
Collider2D[]                colliders                =
hysics2D.OverlapCircleAll(groundcheckCollider.position,
roundCheckRadius, groundLayer);
if (colliders.Length > 0)
isGrounded = true;
}
```

18. Klik player-id pada hirarki, kemudian ke Inspector, di bagian paling bawah akan muncul Player (Script). Ubah Groundcheck Collide, Ground Layer, Speed dan centang pada bagian Is Grounded seperti pada gambar di bawah ini.



Gambar 8.15 Setting Player (Script)

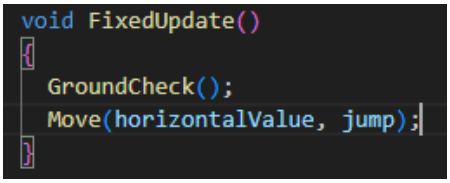
19. Untuk membuat Player bisa melompat, tambahkan script berikut.

```
[SerializeField] float jumpPower = 100;

bool jump;
```

20. Tambahkan script berikut di bagian void update.

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

21. Menambahkan jump pada parameter Move.

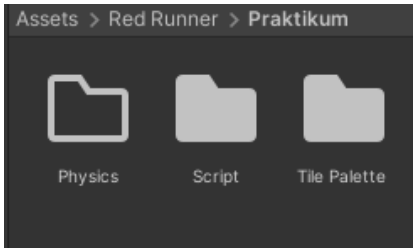


Gambar 8.16 Menambahkan Jump pada Parameter Move

22. Tambahkan script pada void Move.

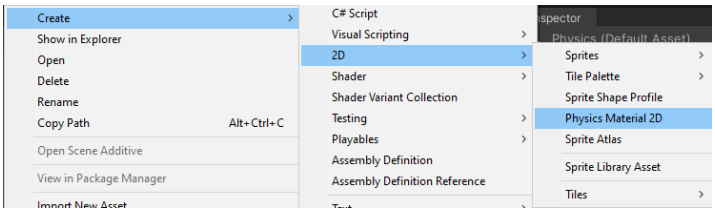
```
bool jumpflag

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

23. Buat folder baru Bernama Physics pada folder Praktikum.



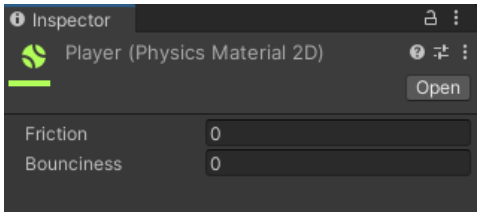
Gambar 8.17 Membuat Folder Physics

24. Dalam folder Physics create kemudian 2D, Physics Material 2D dan berinama "Player".



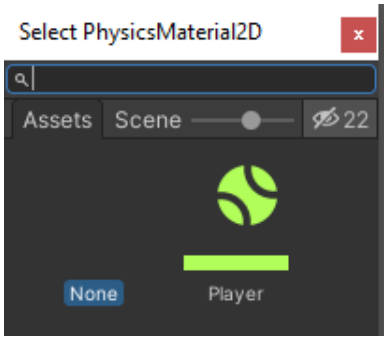
Gambar 8.18 Membuat Physics Material 2D

25. Pada Inspector, friction dan Bounciness diubah menjadi 0.



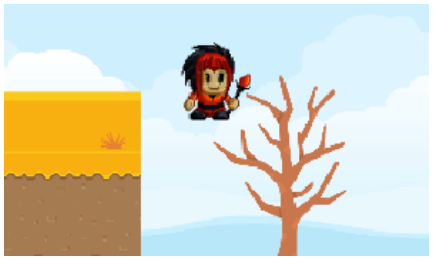
Gambar 8.19 Mengubah Nilai Friction dan Bounciness

26. Pada hirarki pilih player-id, di Inspector pada komponen Rigidbody 2D klik icon untuk membuka box select physics material 2D, lalu pilih asset Player yang sudah dibuat sebelumnya.



Gambar 8.20 Memilih Asset Player pada Rigidbody 2D

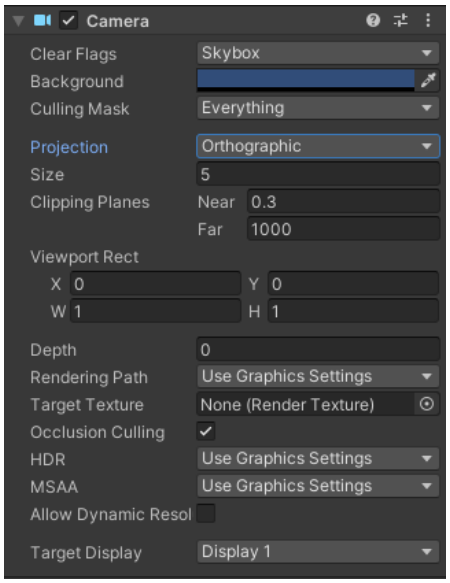
27. Tekan play, kemudian tekan spasi agar Player bisa melompat.



Gambar 8.21 Player Melompat

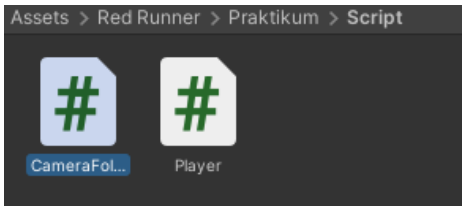
B. Camera Movement

1. Create empty pada HIrarki kemudian rename menjadi Camera. Pada Inspector ubah Projection menjadi Orthographic.



Gambar 8.22 Setting Inspector Kamera

2. Membuat file script baru pada folder Script dan beri nama CameraFollow.



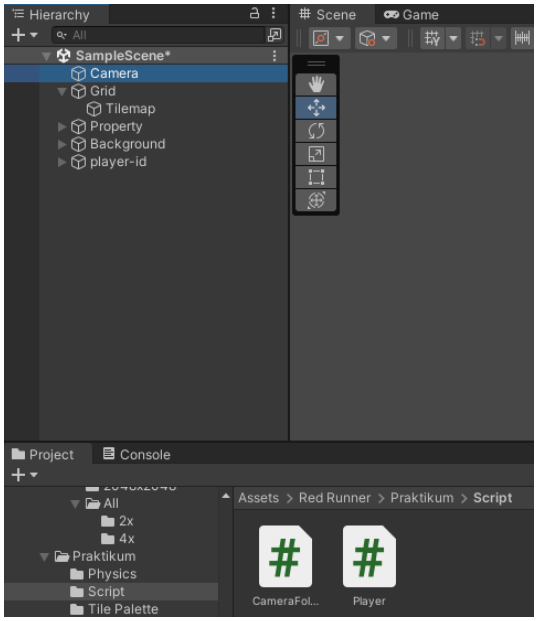
Gambar 8.23 Membuat File Script CameraFollow

3. Masukkan script berikut.

```
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 and drop script CameraFollow ke dalam layer Camera.

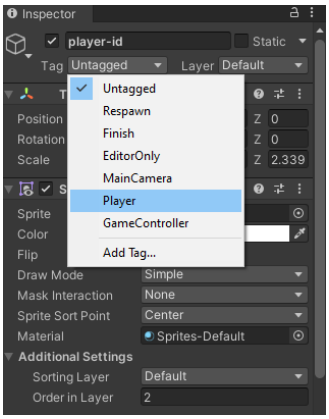


Gambar 8.24 Drag & Drop Script CameraFollow

5. Klik pada layer Camera, buka pada Inspector. Pada Camera Follow (Script) ubah nilai Max X And Y menjadi 82.

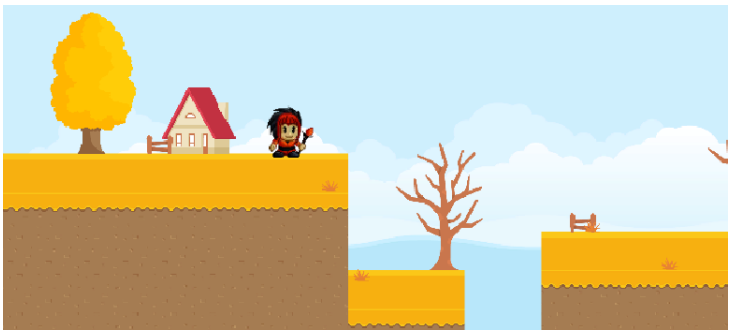
Gambar 8.25 Mengubah Nilai Max X And Y

6. Ubah tag di player-id Untagged menjadi Player.



Gambar 8.26 Mengubah Tag

7. Klik Play untuk melanjutkan dan kamera akan mengikuti pergerakan dari Player.



Gambar 8.27 Hasil Play

C. 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 code di atas digunakan untuk menerapkan camera movement. Void update digunakan untuk memastikan jika metode update() dipanggil setiap frame. Transform.position digunakan untuk mengakses posisi yang diambil dari sudut pandang kamera. Sementara itu, posisi X dari Player ditetapkan ke posisi X dari kamera, dan posisi Y dan Z tetap dijaga agar tetap sama seperti sebelumnya.