



TUGAS PERTEMUAN: 8

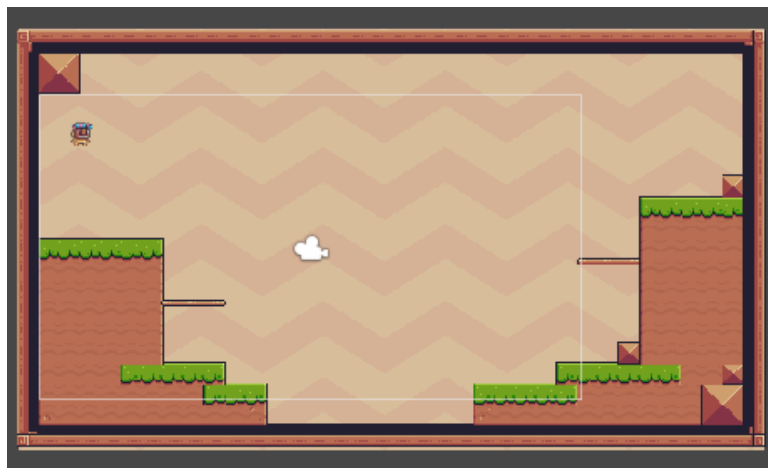
CAMERA & CHARACTER MOVEMENT

NIM	:	2118027
Nama	:	RAFID ARTUR PRASETYA
Kelas	:	C
Asisten Lab	:	APRILLIA DWI DYAH S. (2118143)

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

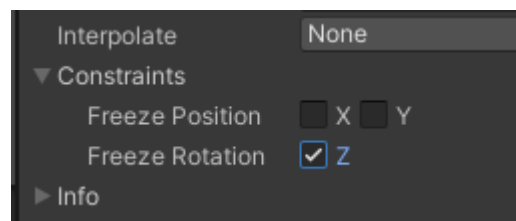
A. Membuat Pergerakan Layar

1. Buka project sebelumnya yaitu pada bab 7.



Gambar 8.1 Open File Project Bab 7

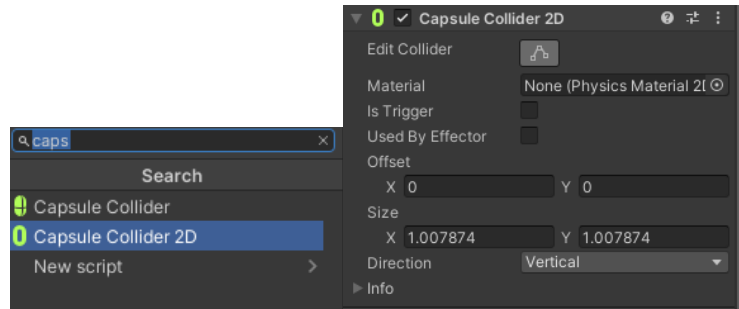
2. Klik character, kemudian menuju ke Inspector dan tambahkan komponen baru yaitu Rigidbody 2D. Centang pada bagian Freeze Rotation Z.



Gambar 8.2 Add Rigidbody 2D Component

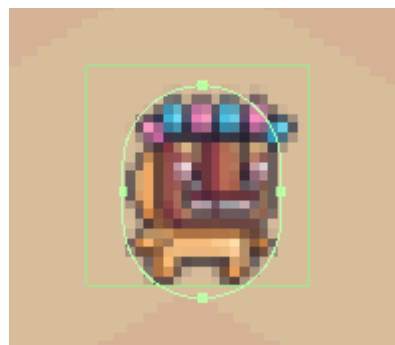


3. Tambahkan juga komponen yaitu Capsule Collider 2D.



Gambar 8.3 Add Capsule Collider 2D Component

4. Sesuaikan ukurannya dengan objek karakter yang digunakan.



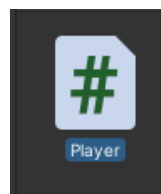
Gambar 8.4 Menyesuaikan Ukuran Capsule Collider

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



Gambar 8.5 Menambahkan Folder Script

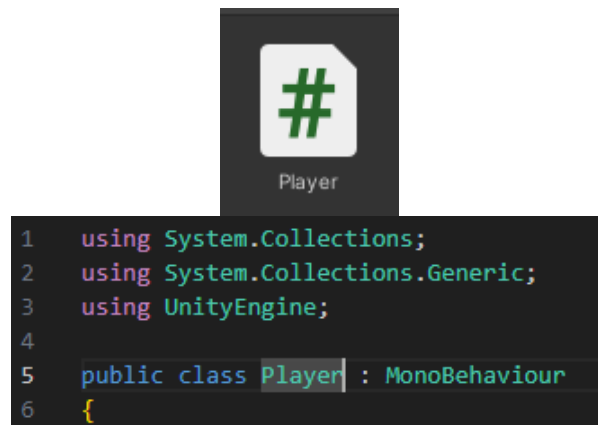
6. Masuk ke dalam folder Script, kemudian buat file C# Script dan beri nama “Player”.



Gambar 8.6 Membuat file C# Player



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



Gambar 8.7 Drag and Drop File C# Player ke character

8. 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
}
```

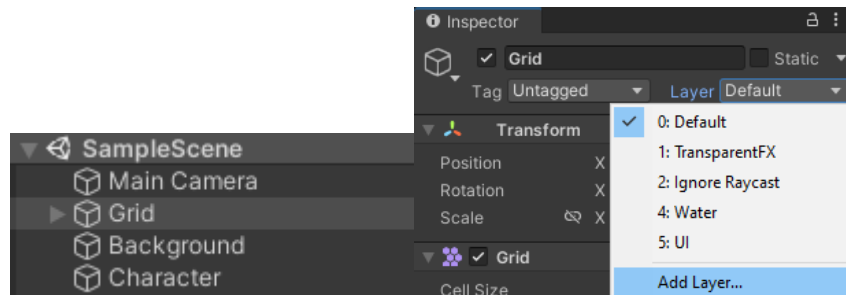
9. Untuk mencoba source code berhasil atau tidak, maka Play terlebih dahulu kemudian tekan “a” atau “left arrow” untuk ke arah kiri, tekan “d” atau “right arrow” untuk ke arah kanan.



Gambar 8.8 Pengujian Source Code

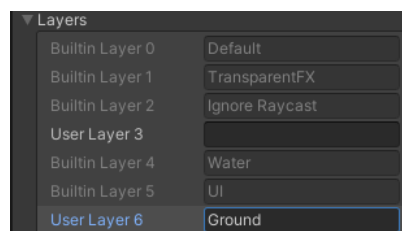


10. Selanjutnya karakter akan dibuat bisa melompat dengan menggunakan spasi. Caranya yaitu membuat GroundCheck dengan klik Grid pada hirarki, kemudian pada Inspector terdapat Layer, pilih Add Layer.



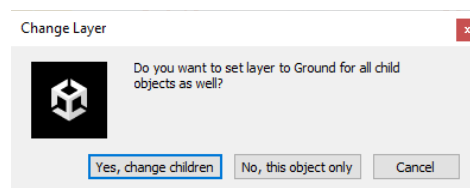
Gambar 8.9 Add Layer pada Grid

11. Isikan User Layer 6 dengan “Ground”.



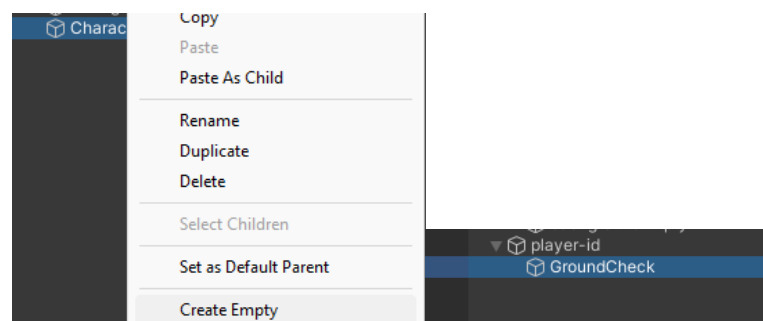
Gambar 8.10 Membuat Layer 6 Menjadi Ground

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



Gambar 8.11 Klik Yes

13. Klik kanan pada character, kemudian Create Empty. Dan beri nama menjadi GroundCheck.



Gambar 8.12 Membuat GroundCheck



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



Gambar 8.13 Memindahkan Ground Check ke Bagian Bawah Player

15. Tambahkan source code berikut pada script Player.

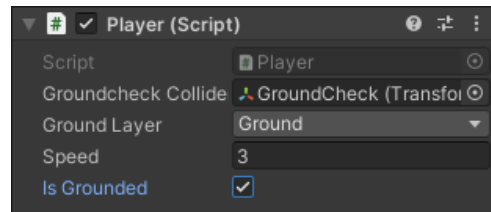
```
[SerializeField] Transform groundcheckCollider;  
[SerializeField] LayerMask groundLayer;  
  
const float groundCheckRadius = 0.2f; // +  
[SerializeField] float speed = 1;  
float horizontalValue;  
  
[SerializeField] bool isGrounded; // +  
bool facingRight;
```

16. Buat void ground check di bawah void fixedUpdate & tambahkan GroundCheck(); 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;  
}
```



17. Klik character 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 8.14 Setting Player (Script)

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

```
[SerializeField] float jumpPower = 100;

bool jump;
```

19. Tambahkan script berikut di bagian void update.

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

20. Menambahkan jump pada parameter Move.

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

Gambar 8.15 Menambahkan Jump pada Parameter Move

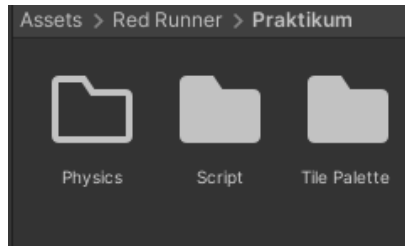
21. Tambahkan script pada void Move.

```
bool jumpflag

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

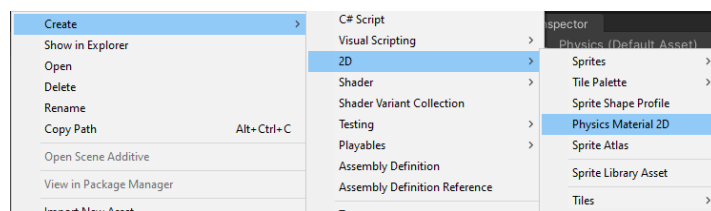


22. Buat folder baru Bernama Physics pada folder Praktikum.



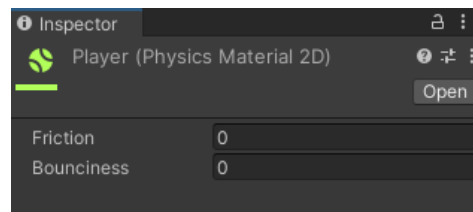
Gambar 8.16 Membuat Folder Physics

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



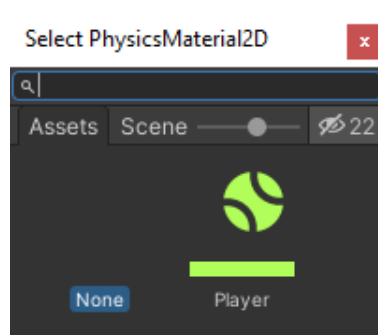
Gambar 8.17 Membuat Physics Material 2D

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



Gambar 8.18 Mengubah Nilai Friction dan Bounciness

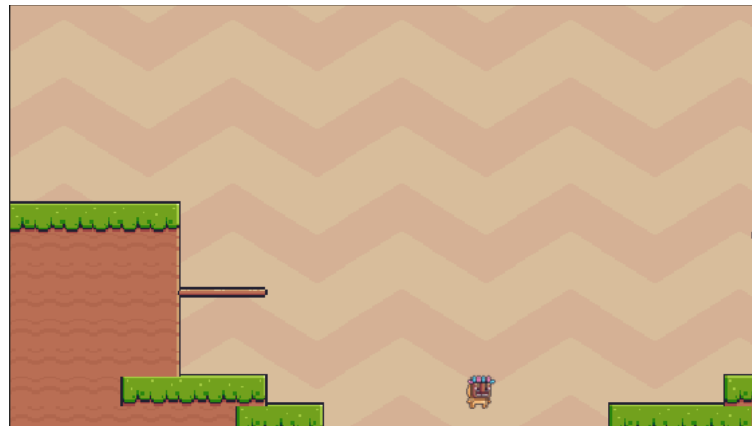
25. Pada hirarki pilih character, di Inspector pada komponen Rigidbody 2D klik icon untuk membuka box select physics material 2D, lalu pilih asset Player yang sudah dibuat sebelumnya.



Gambar 8.19 Memilih Asset Player pada Rigidbody 2D



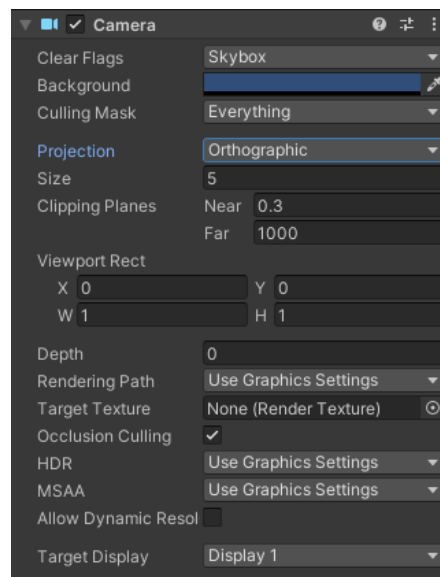
26. Tekan play, kemudian tekan spasi agar Player bisa melompat.



Gambar 8.20 Player Melompat

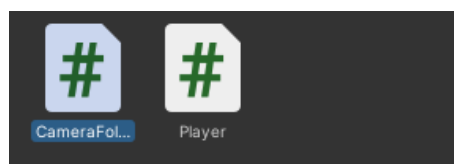
B. Camera Movement

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



Gambar 8.21 Setting Inspector Kamera

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



Gambar 8.22 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;
    }

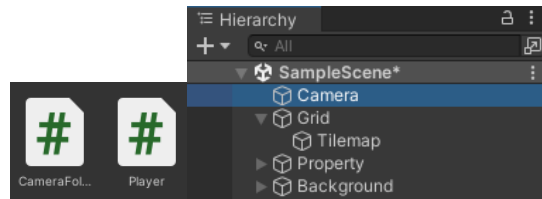
    void FixedUpdate()
    {
        TrackPlayer();
    }

    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);
    }
}
```



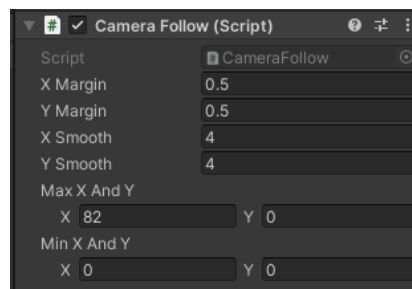
```
}
```

4. Drag and drop script CameraFollow ke dalam layer Camera.



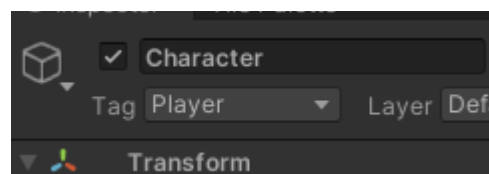
Gambar 8.23 Drag & Drop Script CameraFollow

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



Gambar 8.24 Mengubah Nilai Max X And Y

6. Ubah tag di character Untagged menjadi Player.



Gambar 8.25 Mengubah Tag



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



Gambar 8.26 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 mengambil value posisi dari player berdasarkan posisi sumbu x, y dan z. dengan mendefinisikan variable player dengan tipe data transform dengan nama class CameraFollow.