# PRAKTIKUM ANIMASI DAN GAME PERTEMUAN :8

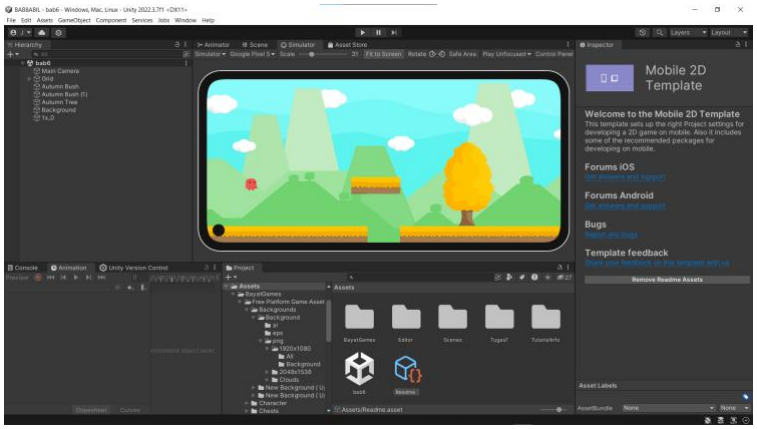
# Camera & Character Movement

| | | |
|---|---|---|
| **NIM** | **:** | 2018048 |
| **NAMA** | **:** | Mustaqdimin Salsabil Haq |
| **KELAS** | **:** | D |
| **Asisten Lab** | **:** | Difa Fisabilillah (2118052) |

## 8.1 Tugas 1 : Membuat

### A. Membuat Pergerakan Player

1. Buat file *projek Unity* Tugas 7

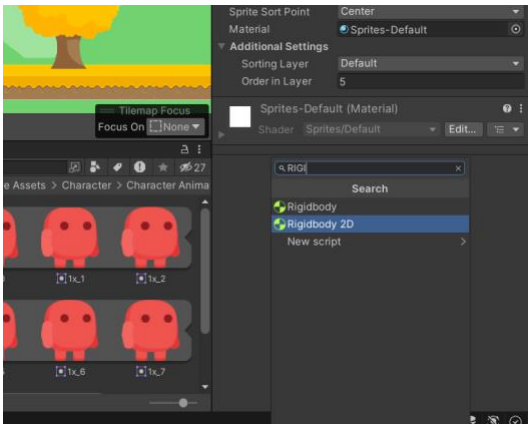



Gambar 8.1 Buka Project Tugas 7

2. Tambahkan player, pilih yang idle, Import kedalam Hirarki.

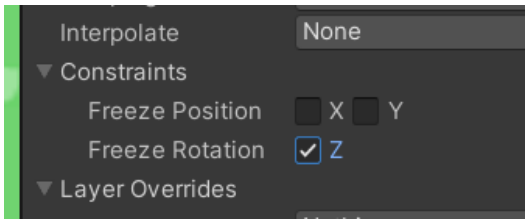
Gambar 8.2 Tampilan menambahkan karakter

3. Klik karakter tersebut, pergi ke *Inspector* dan klik *Add Component*, kemudian cari komponen bernama *RigidBody2D*, komponen tersebut berguna untuk memberikan efek gravitasi pada objek



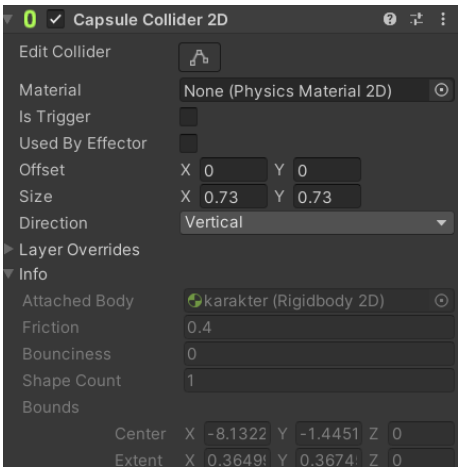

Gambar 8.3 komponen *RigidBody 2d*

4. sesuaikan settingannya seperti gambar berikut, Centang pada Freeze Rotation Z



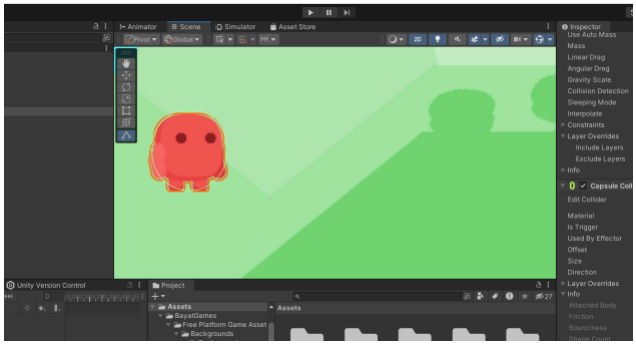

Gambar 8.4 Centang Rotation Z

5. Lalu tambahkan komponen Capsule Colider di karakter, lalu klik icon sebelah kanan edit colider



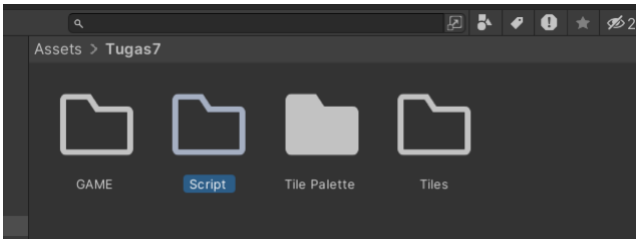

Gambar 8.5 Menambahkan komponen *capsule collider 2d*

6. Lalu cockan garis oval degan karakternya atau bisa di inputkan Offset X, Y dan juga Size X, Y nya



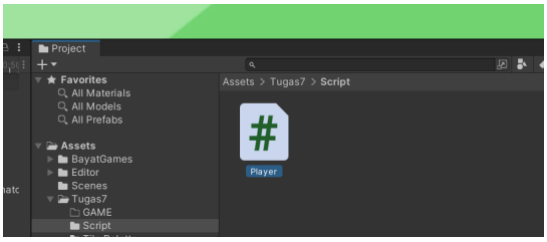
Gambar 8.6 Mengatur gari oval

7. Buka Folder Tugas7, lalu bikin folder baru bernama Script



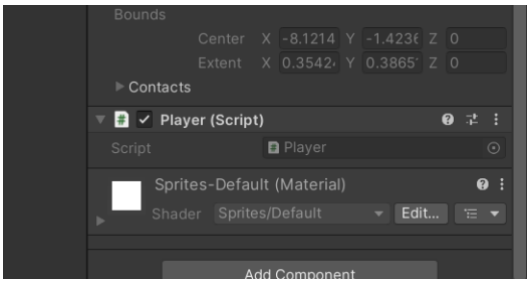

Gambar 8.7 Membuat folder script

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




Gambar 8.8 Membuat script

9. Drag & drop script player kedalam Hirarki player-idle-1, lalu klik 2x pada script player maka akan masuk kedalam text editor seperti ini




Gambar 8.9 Memasukan Script Player

10. Lalu pada script Player masukkan Source Berikut

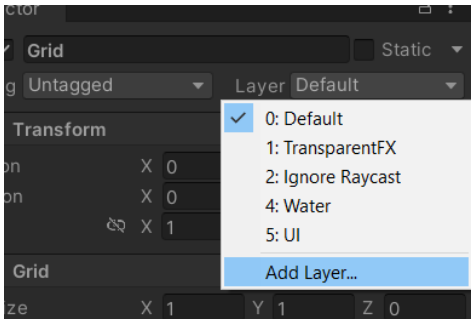
```
using System.Collections;
```

```
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
  }
}
```

11. 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.

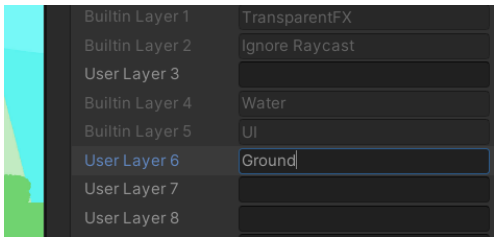
Gambar 8.10 Menjalankan program

12. Untuk membuat player loncat menggunakan spasi, kita perlu membuat GorundCheck dengan cara, klik Grid pada Hierarchy, pergi ke inspector, pilih Layer, Klik Add Layer



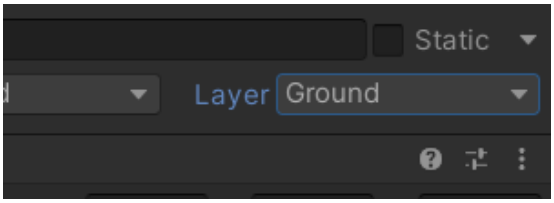

Gambar 8.11 Add Layer

13. Isikan *User Layer* 6 dengan nama *Ground*



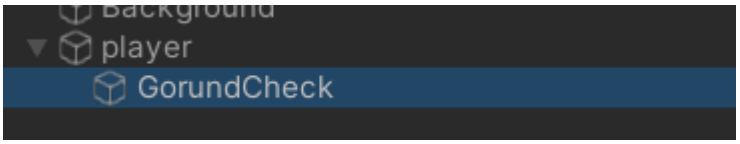

Gambar 8.12 Mengisi *layer 6* menjadi *Ground*

14. Ganti *Layer Default* menjadi *Ground*




Gambar 8.13 Menggati layer menjadi *Ground*

15. Klik kanan *Warrior* kemudian pilih *Create Empty* dan ubah namanya menjadi *GroundCheck*




Gambar 8.14 Membuat *GroundCheck*

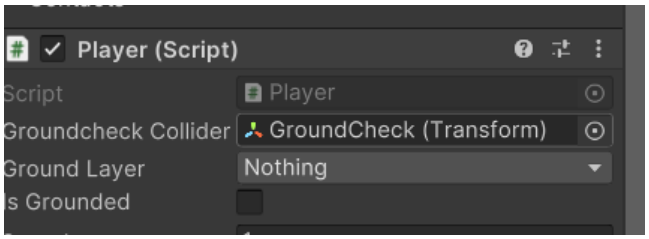
16. Kembali ke script Player tambahkan source code seperti ini

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

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

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

17. Buat void ground check dibawah void fixedUpdate & tambahkan GorundCheck(); pada void fixedUpdate

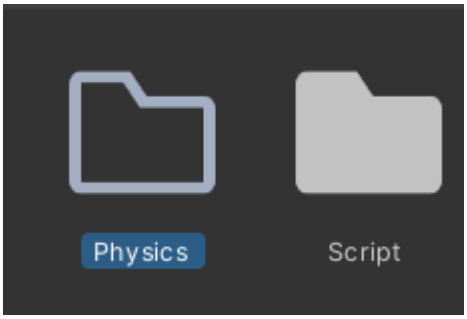
```
  void FixedUpdate()
  {
  GroundCheck();
  Move(horizontalValue);
  }

  void GroundCheck()
  {
  isGrounded = false;
  Collider2D[]                colliders                =
Physics2D.OverlapCircleAll(groundcheckCollider.positio
n, groundCheckRadius, groundLayer);
  if (colliders.Length > 0)
  isGrounded = true;
  }
```

18. drag *GroundCheck* arahkan kedalam *Groundcheck* Collider yang ada pada *Inspector*



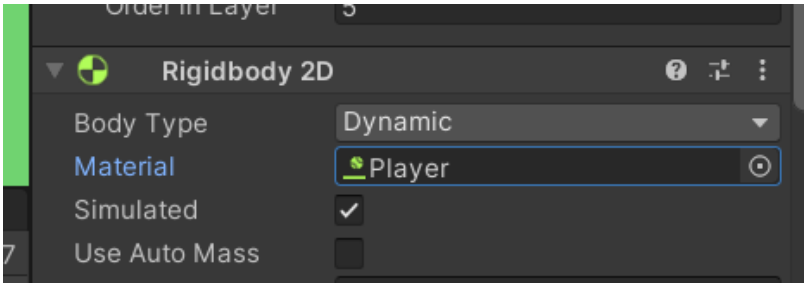

Gambar 8.15 Penyesuaian *GroundCheck*

19. Membuat folder Physics



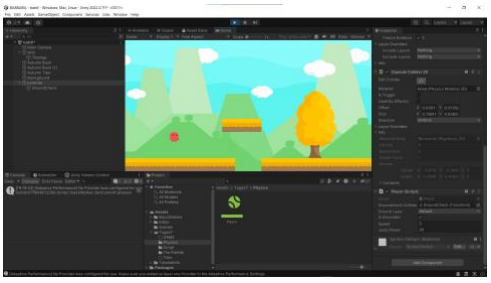

Gambar 8.16 Membuat folder Physics

20. Klik kanan folder "*Physics*" pilih *Create>Physics Material 2D* dan beri nama "karakter" lalu ubah nilai Friction dan Bouncies menjadi 0




Gambar 8.17 Tampilan *Physics* karakter

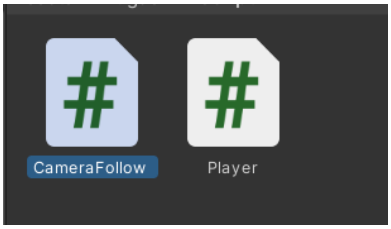
21. Tekan play, maka player bisa melompat dengan menekan spasi



Gambar 8.18 Tampilan *Physics* karakter

## B. Camera Movement

1. Tambahkan script dalam folder Script dan beri nama *"CameraFollow.cs"*



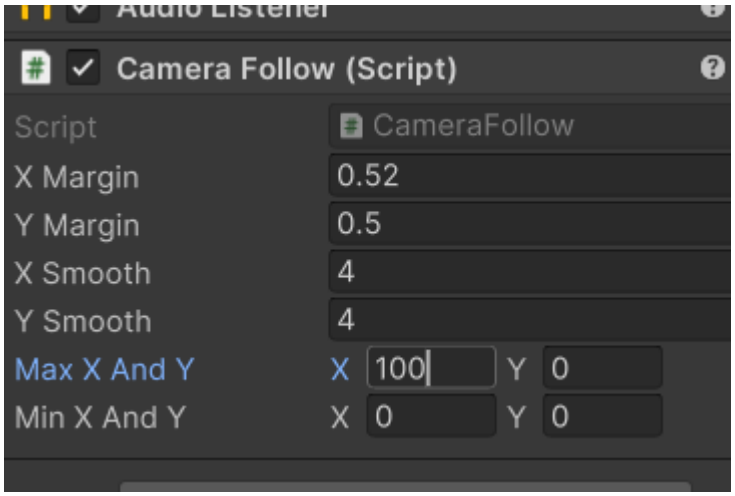

Gambar 8.19 Membuat dan Menambahkan *Source Code*

2. Tambahkan *Source Code* berikut 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,
```

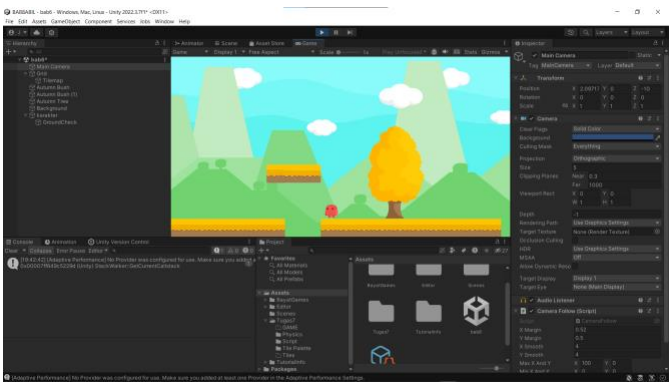
```
player.position.y,
            ySmooth * Time.deltaTime);
         targetX = Mathf.Clamp(targetX, minXAndY.x,
maxXAndY.x); targetY =
        Mathf.Clamp(targetY, minXAndY.y, maxXAndY.y);
transform.position = new
        Vector3(targetX, targetY,
transform.position.z);
    }
}
```

3. Pergi ke *Inspector Main* Camera kemudian lakukan setting *Camera Follow* seperti gambar berikut




Gambar 8.20 Mensetting *Camera Follow*

4. Jika di *Play*, maka camera akan mengikuti pergerakan setiap karakter.



Gambar 8.21 Tampilan game ketika di Run

## 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,
player.Position.y, player.Position.z);
}
}
```

**Analisa :**

Source code diatas adalah code yang digunakan untuk mengikuti pergerakan kamera pada objek pemain dalam game Unity. Pertama mengimport library selanjutnya Kelas CameraFollow akan mewarisi kelas MonoBehaviour dari Unity. Ini berarti kelas ini dapat digunakan sebagai komponen pada objek game dalam Unity. Properti [SerializeField] private Transform player; adalah properti yang menunjukkan objek pemain yang akan diikuti oleh kamera. Properti ini ditandai dengan atribut [SerializeField] agar dapat diakses dan diatur melalui editor Unity. Metode Update() adalah metode yang dipanggil setiap frame dalam game. Di dalam metode ini, posisi kamera diatur agar mengikuti posisi pemain. Posisi kamera diubah menggunakan transform.position dengan menggunakan posisi pemain (player.position) sebagai referensi.

## D. Link Github

https://github.com/Mustaqdimin/2018048_PRAK_ANIGAME/tree/2223c4bcb1bc68ebc79e3936dc8b630ef3c6c154/BAB8