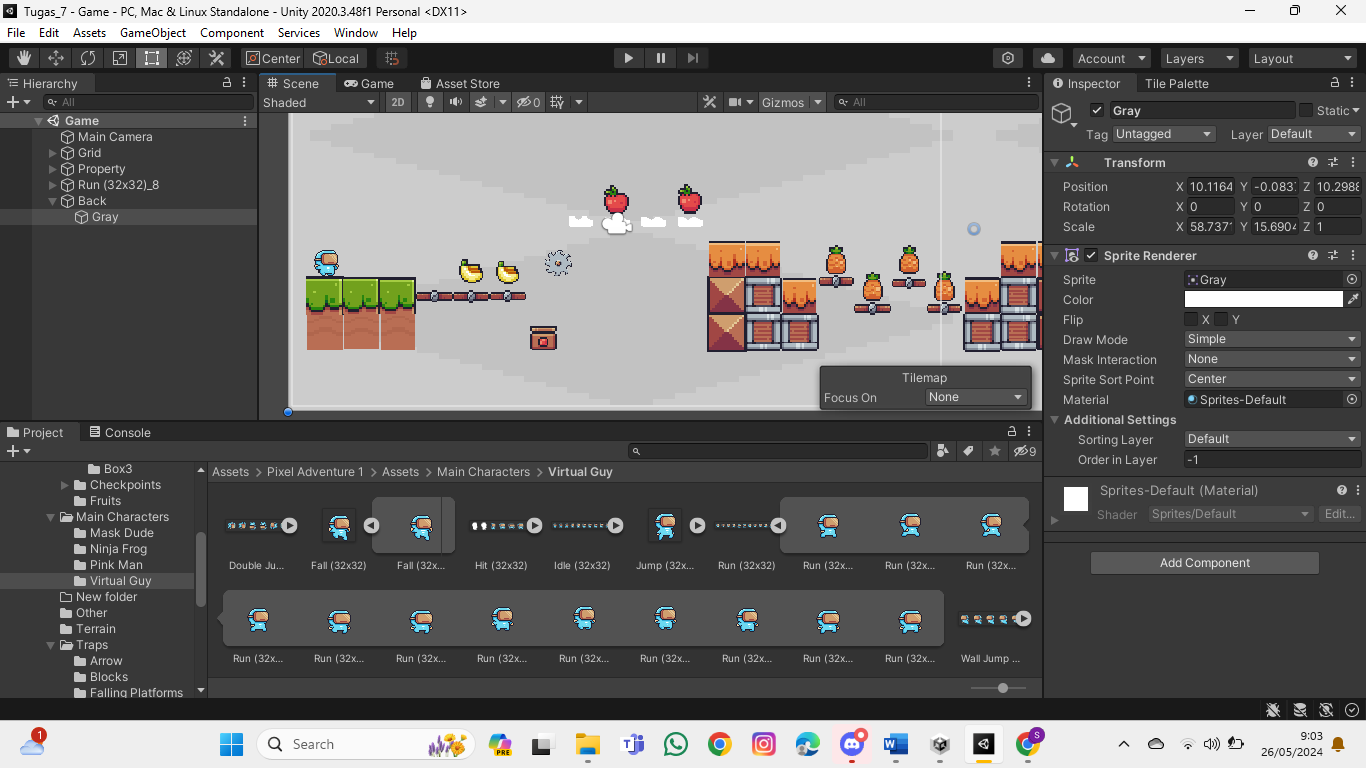
# 8 Camera & Character Movement

|  |  |  |
| --- | --- | --- |
| **NIM** | : | 2118018 |
| **Nama** | : | Syalsia Fatiha Yunkanabilla |
| **Kelas** | : | A |
| **Asisten Lab** | : | Natasya Octavia (2118034) |

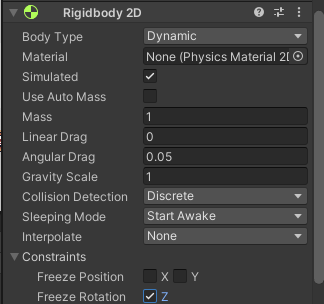
## 1.1 Tugas 1 : Membuat Pergerakan Player

1. **Membuat Pergerakan Player**
2. Buka aplikasi unity yang telah dibuat sebelumnya pada bab7 kemudian tambahkan karakter player run. dimana telah dilakukan import asset yg didownload sebelumnya



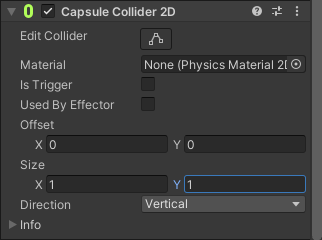
### 8.1 Tampilan Unity

1. Kemudian klik pada player run pada hierarchy kemudian pada inspector tambahkan komponen baru yaitu rigidbody 2 collider dan centang box pada freeze rotation z.



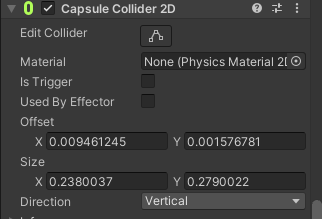
### 8.2 Add Component

1. Langkah selanjutnya, tambahkan capsule collider 2D pada karakter run kemudian edit collider.



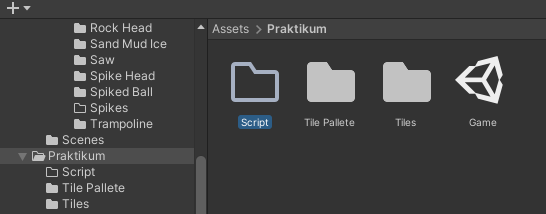
### 8.3 Capsule Collider 2D

1. Kemudian sesuaikan ukuran garis oval sesuai dengan karakter atau dapat juga dengan menggunakan offset X dan Y nya.



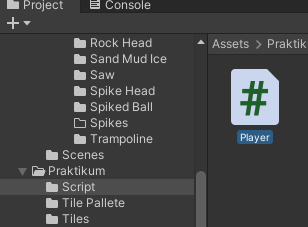
### 8.4 Mengatur Offset

1. Kemudian pada folder praktikum klik kanan lalu pilih create>folder dan ubah nama menjadi script.



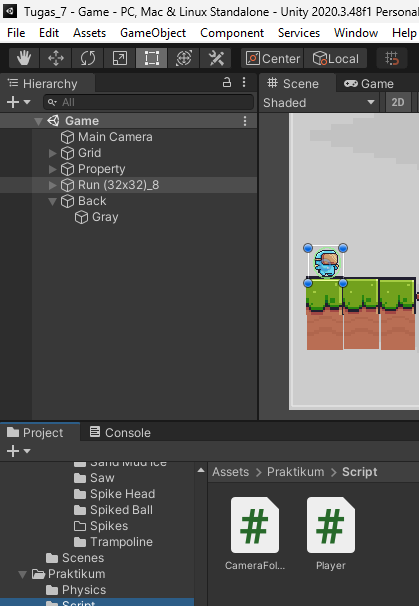
### 8.5 Create Script

1. Kemudian pada folder script klik kanan kemudian pilih C# kemudian ubah namanya menjadi player.



### 8.6 Create C#

1. Kemudian drag & drop script player kedalam karakter run player agar dapat dimasukkan script.

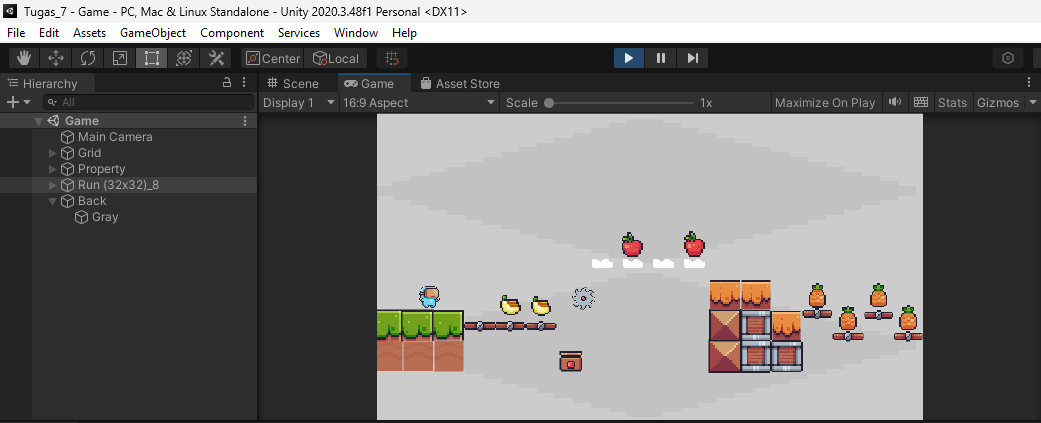


### 8.7 Drag & Drop Script Player

1. Kemudian masukkan source code berikut untuk membuat player dapat bergerak ke kanan dan ke kiri.

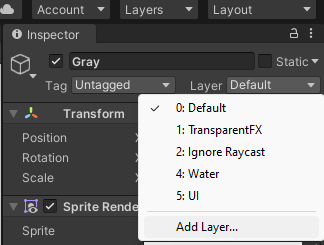
|  |
| --- |
| using System.Collections;  using System.Collections.Generic;  using UnityEngine;  public class Player : MonoBehaviour  {  Rigidbody2D rb;  [SerializeField] float speed = 1;  float horizontalValue;  bool facingRight;  private void Awake()  {  rb = GetComponent<Rigidbody2D>();  }  void Update ()  {  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  }  } |

1. Kemudian play untuk mengecek apakah karakter dapat bergerak ke kanan atau kekiri atau belum.



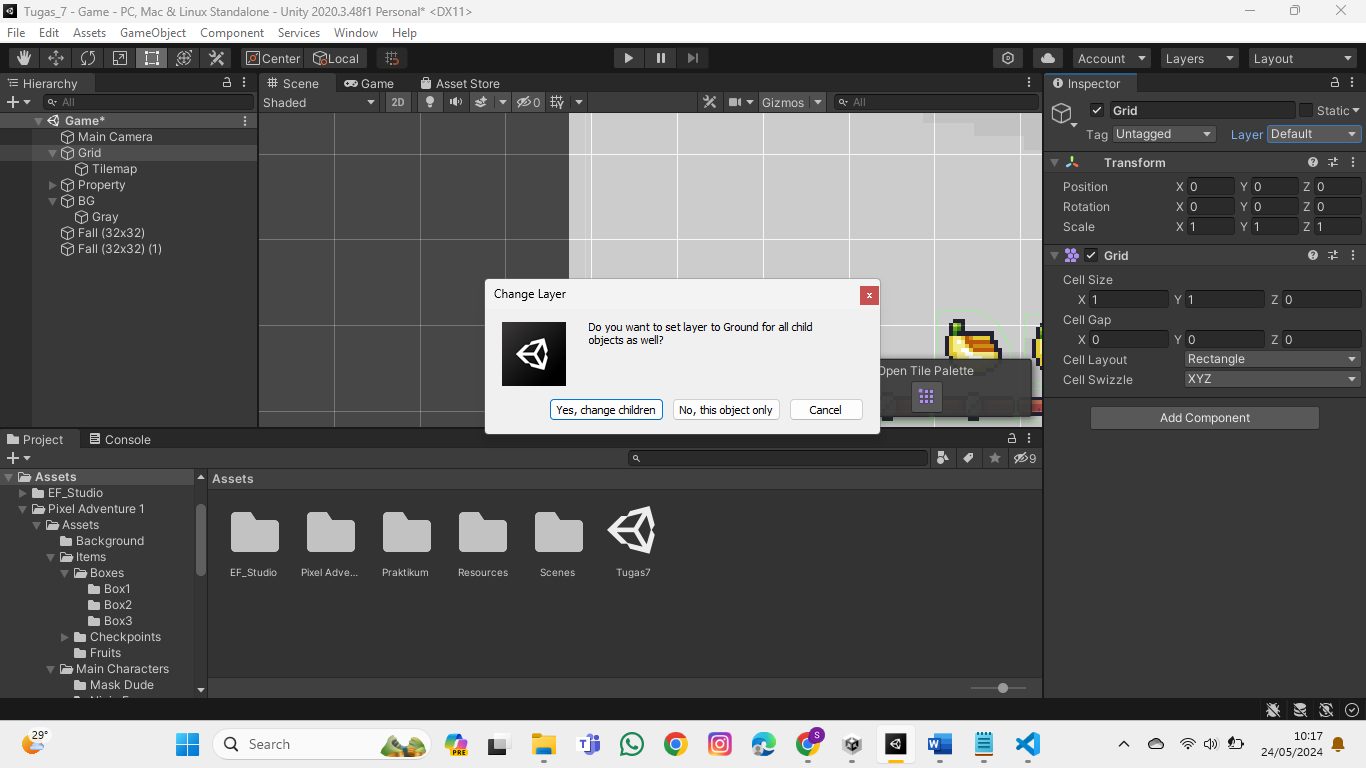
### 8.8 Play Karakter

1. Setelah itu untuk membuat karakter loncat maka pada hierarcy grid menuju inspector kemudian pada layer pilih add layer kemudian pada user 6 isikan ground.



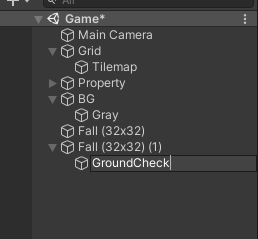
### 8.9 Add Layer

1. Kemudian pada layer ubah menjadi ground, jika muncul pop up saat mengubah layer menjadi ground maka pilih yes saja.



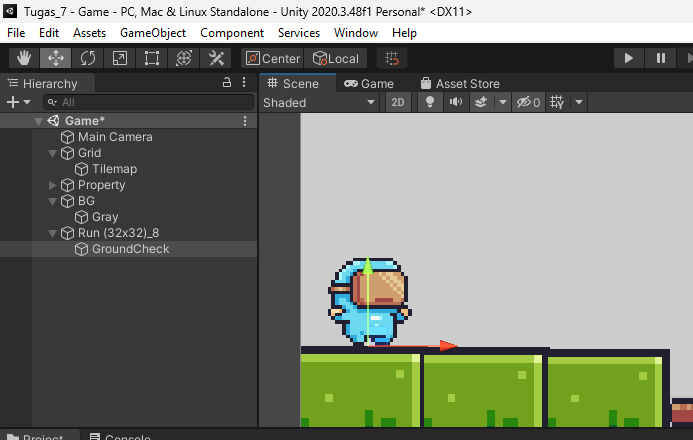
### 8.10 Mengubah Layer

1. Selanjutnya pada tampilan hierarchy karakter player klik kanan kemudian pilih Create Empty kemudian ubah nama menjadi GroundCheck.



### 8.11 Create Empty GroundCheck

1. Kemudian pada hierarcy GroundCheck gunakan move tools dan ubah ke arah bawah karakter player



### 8.12 Move Tools

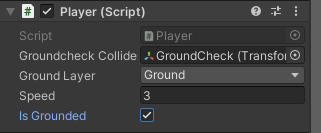
1. Kemudian kembali ke script player dan tambahkan source code berikut ini.

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

1. Kemudian tambahkan script pada void GroundCheck dibawah void fixedUpdate seperti dibawah ini.

|  |
| --- |
| void FixedUpdate()  {  GroundCheck();  Move(horizontalValue);  }  void GroundCheck()  {  isGrounded = false;  Collider2D[] colliders = Physics2D.OverlapCircleAll(groundcheckCollider.position, groundCheckRadius, groundLayer);  if (colliders.Length > 0)  isGrounded = true;  } |

1. Lalu klik karakter player kemudian pada inspector scroll ke bawah menuju player (script) tekan ikon GroundCheckCollide ubah menjadi GroundCheck (Transform) kemudian pada GroundLayer ubah menjadi Ground.



### 8.13 GroundCheck(Transform)

1. Kemudian tambahkan source code berikut ini agar karakter player dapat melompat.

|  |
| --- |
| [SerializeField] float jumpPower = 100;  bool jump; |

1. Selanjutnya, tambahkan script pada bagian voidUpdate seperti tampilan dibawah ini.

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

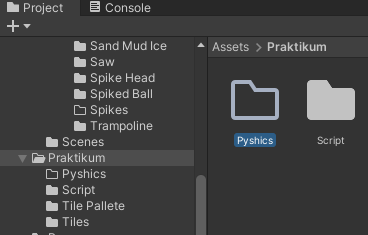
1. Tambahkan juga parameter move dengan tampilan source code seperti berikut.

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

1. Kemudian tambahkan source code pada karakter player pada void move dengan source code berikut.

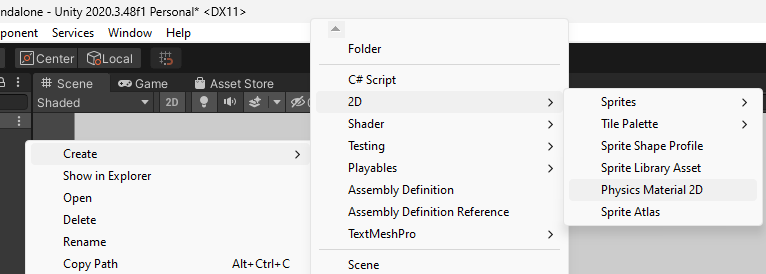
|  |
| --- |
| bool jumpflag  if(isGrounded && jumpflag)  {  isGrounded = false;  jumpflag = false;  rb.AddForce(new Vector2(0f, jumpPower));  } |

1. Kemudian buatlah folder baru dengan nama physics pada folder praktikum seperti dibawah ini



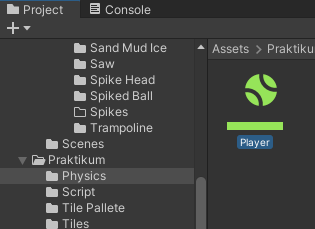
### 8.14 Folder Physics

1. Langkah selanjutnya adalah pada folder physics klik kanan create> 2D> Physics Material 2D seperti berikut.



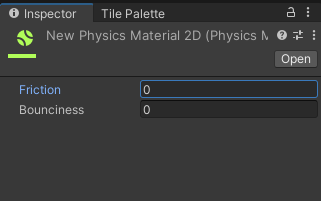
### 8.15 Create Physics Material 2D

1. Kemudian pada tampilan physics akan tampil player seperti berikut ini.



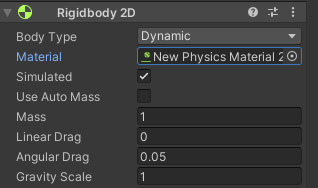
### 8.16 Player

1. Kemudian pada player klik lalu menuju inspector player ubah friction dan bounciness menjadi 0.



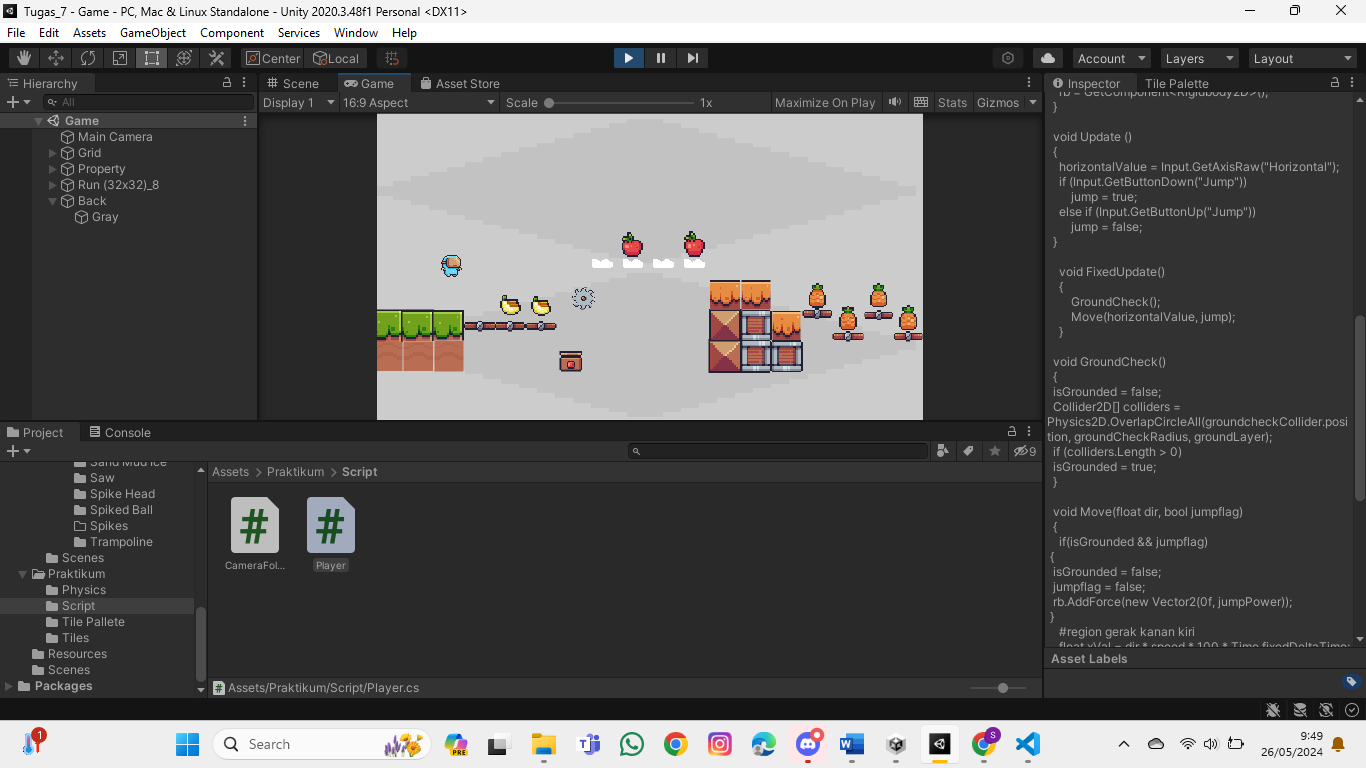
### 8.17 Friction & Bounciness

1. Kemudian pada Rigidbody 2D Collider klik ikon pada materialnya ubah menjadi New Physics Material 2D kemudian sesuaikan seperti tampilan dibawah ini.



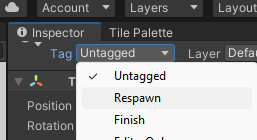
### 8.18 New Physics Material 2D

1. Kemudian tekan play untuk mengecek apakah karakter player dapat melompat atau belum.



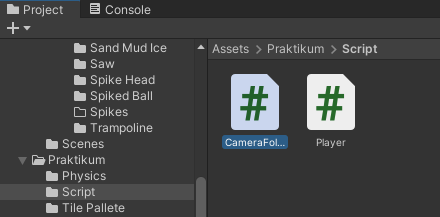
### 8.19 Tampilan Game

1. **Camera Movement**
2. Pada hierarcy camera klik kemudian menuju ke inspector lalu ubah tag menjadi untagged



### 8.20 Untagged Camera

1. Kemudian buat script pada folder praktikum dengan nama Camera Follow

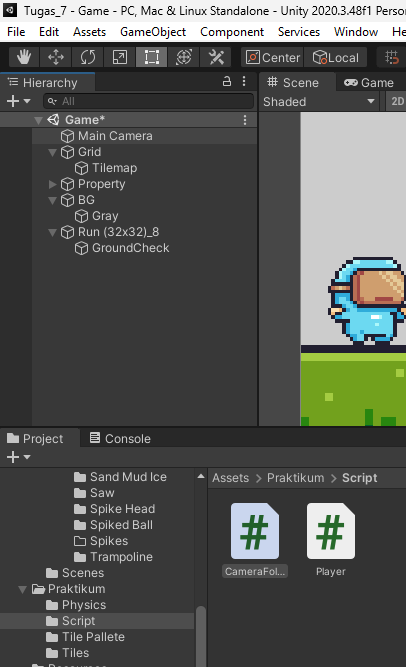


### 8.21 Script CameraFollow

1. Kemudian tambahkan script source code berikut pada 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. Apabila source code pada script CameraFollow telah dimasukkan maka langkah berikutnya adalah drag & drop script CameraFollow ke dalam hierarcy karakter objek.



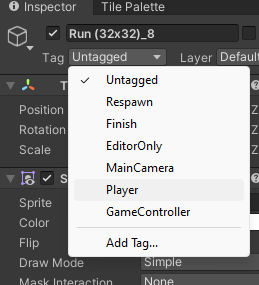
### 8.22 Tampilan Drag & Drop

1. Kemudian pada maincamera klik dan pergi menuju inspector kemudian pada CameraFollow (script) sesuaikan max X Y dan Min X Y.



### 8.23 CameraFollow Script

1. Selanjutnya pada karakter player pergi ke inspector dan ubah tag dari untagged menjadi player.



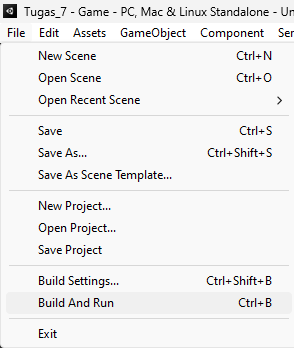
### 8.24 Tag Player

1. Kemudian tekan play untuk menampilkan hasil dari camera movement yang telah dibuat.



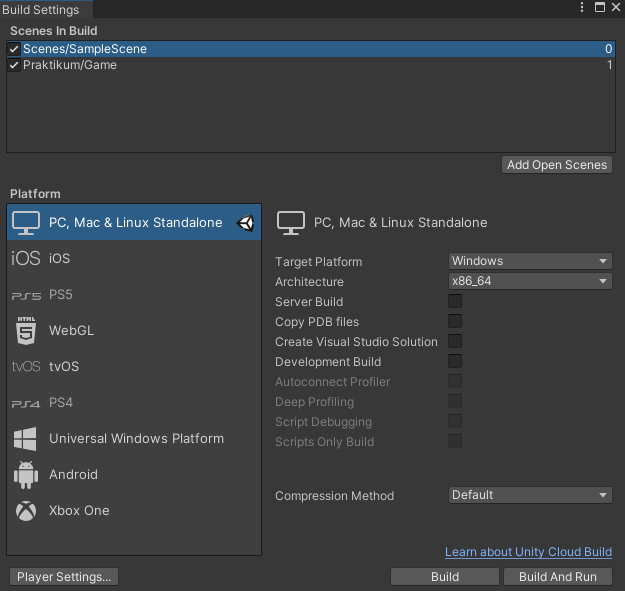
### 8.25 Hasil CameraMovement

1. **Render**
2. Selanjutnya melakukan render dengan klik pada file kemudian pilih build settings



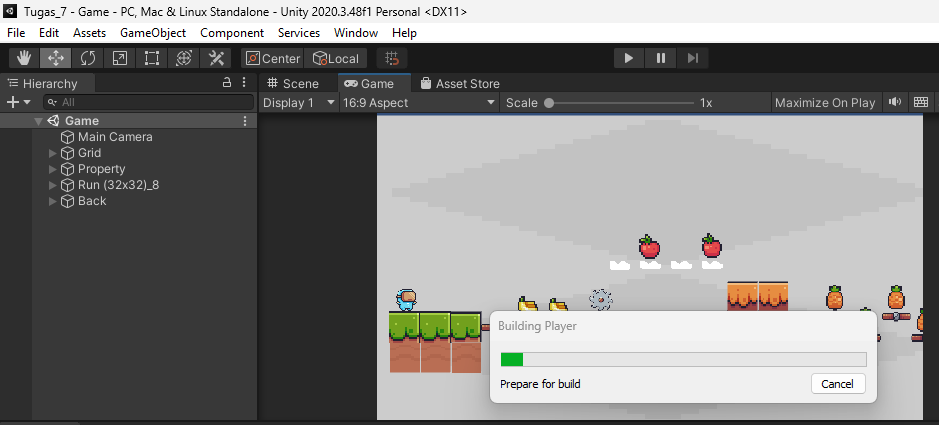
### 8.26 File

1. Kemudian atur tampilan build settings menjadi tampilan seperti berikut ini agar dapat dirender.



### 8.27 Build Settings

1. Selanjutnya lakukan build and run untuk merender game tersebut.



### 8.28 Build and Run

1. Langkah terakhir adalah menampilkan hasil dari build render pada game tersebut.



### 8.29 Hasil Tampilan Render