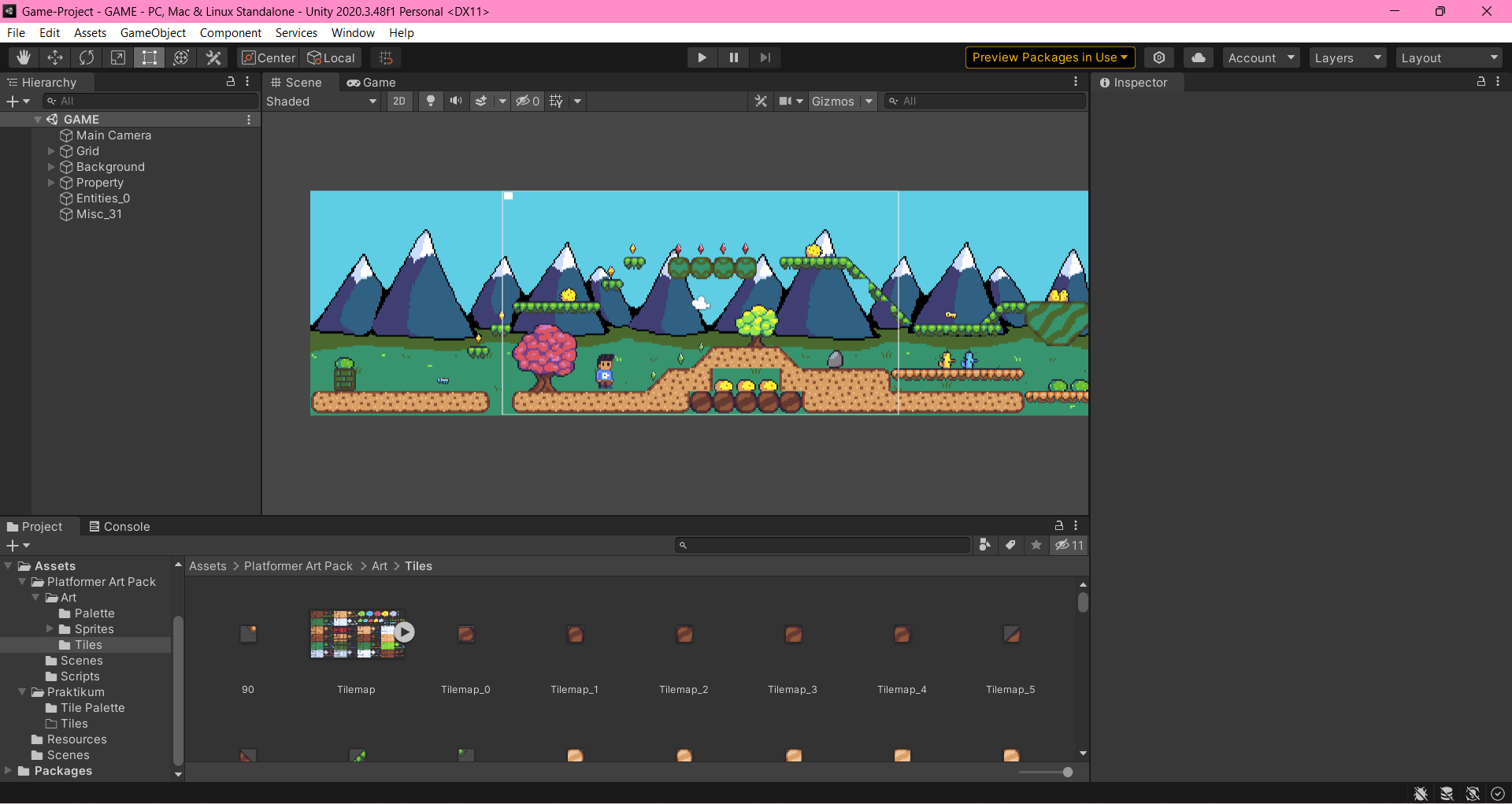
# 8 CAMERA & CHARACTER MOVEMENT

|  |  |  |
| --- | --- | --- |
| **NIM** | : | 2118060 |
| **Nama** | : | CORNELIA LUBA TARA BORO |
| **Kelas** | : | B |
| **Asisten Lab** | : | BAGAS ANARDI SURYA WIJAYA (2118004) |

## Tugas 8 : Membuat Camera dan Character Movement

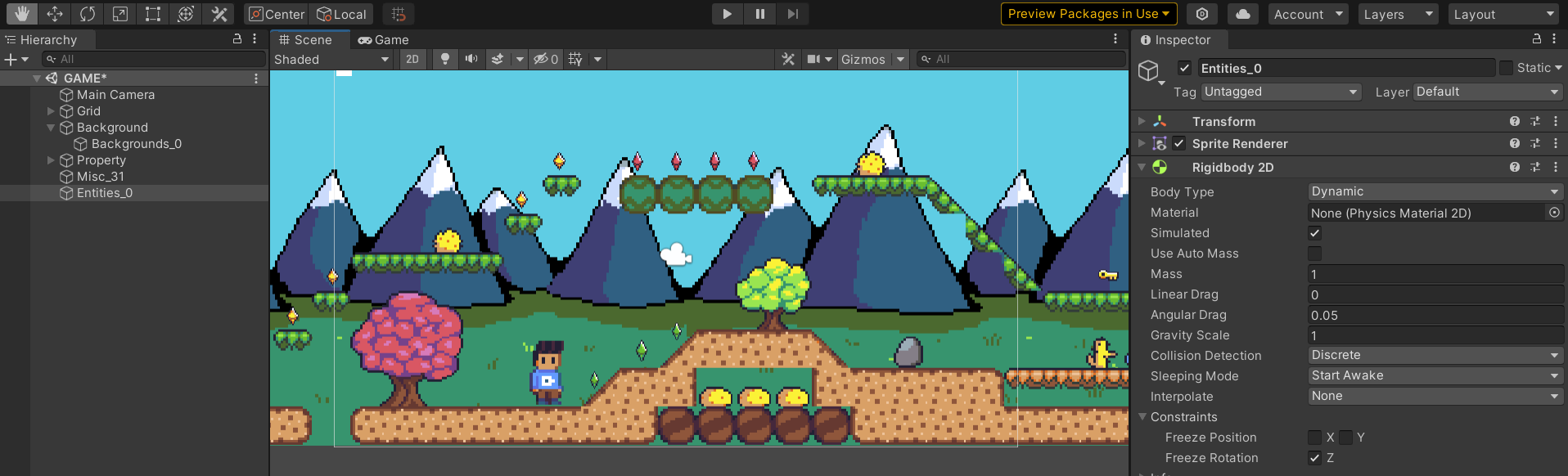
Membuat *Camera* dan *Character Movement*

1. **Membuat Camera dan Character Movement**
2. Buka *project* yang sudah dikerjakan pada bab sebelumnya yang telah terimport asset.



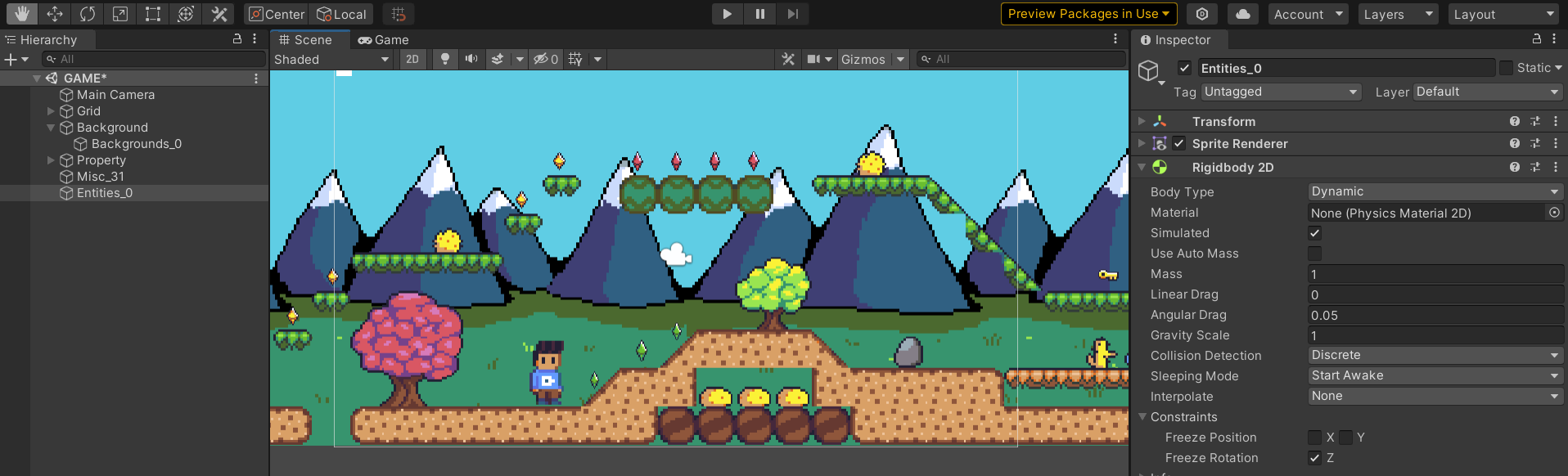
* 1. Buka *Project*

1. Tambahkan karakter *Player* ke dalam area permainan.



* 1. Menambahkan *Player*

1. Klik pada Entities\_0 lalu tambahkan komponen pada *Inspector*. Tambahkan komponen *Rigidbody* 2D sehingga karakter tersebut dapat diberikan fungsi-fungsi fisika.



* 1. Menambahkan Komponen

1. Selanjutnya tambahkan komponen lainnya, yaitu *Capsule* *Collider* 2D, sehingga karakter dapat mendeteksi objek lainnya pada saat bersentuhan dengan objek lainnya.



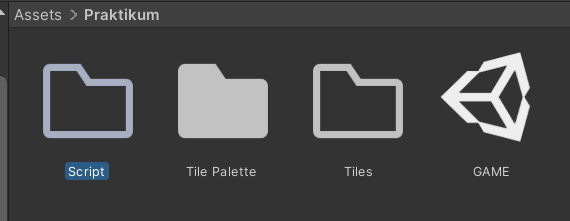
* 1. Menambahkan Komponen

1. Kemudian pilih edit *collider*, lalu sesuaikan *capsule* dengan ukuran karakter.



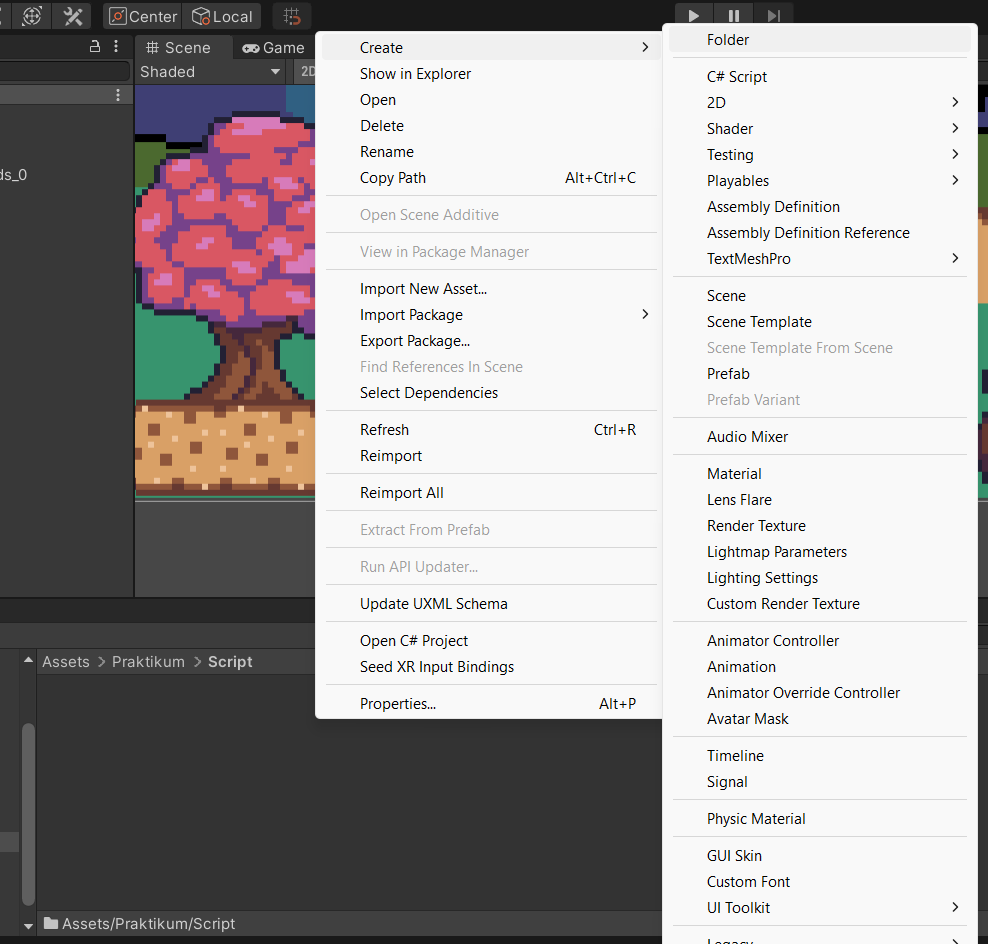
* 1. Edit *Capsule* *Collider*

1. Selanjutnya, buat folder baru didalam folder praktikum dengan cara klik kanan pada folder praktikum lalu pilih *create*>folder dan beri nama *Script* untuk folder tersebut.



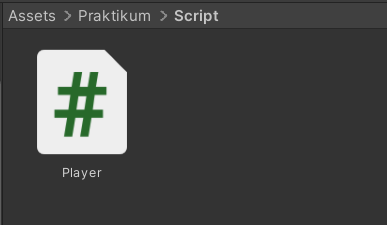
* 1. Buat Folder Baru

1. Selanjutnya, di dalam folder *Script* tambahkan *file* C# *Script* dengan cara klik kanan, lalu pilih *create*>C# *Script*.



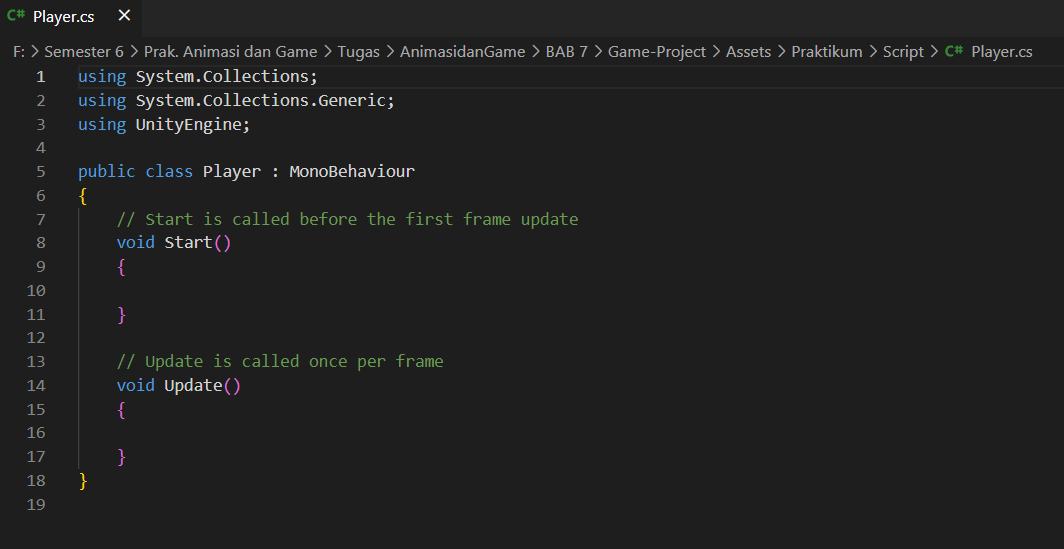
* 1. Membuat *File* C# *Script*

1. Beri nama *Player* pada *script* tersebut, kemudian *drag* *and drop* *file* tersebut ke *layer* Entities\_0.



* 1. Membuat Filde C# *Script*

1. Selanjutnya klik dua kali pada *file* *Player*, maka akan langsung diarahkan ke *tab code editor*.

****

* 1. Code Editor *Player*.cs

1. Selanjutnya tambahkan *script* dibawah ini untuk membuat karakter dapat bergerak ke kiri dan kana sesuai dengan input *keyboard*.

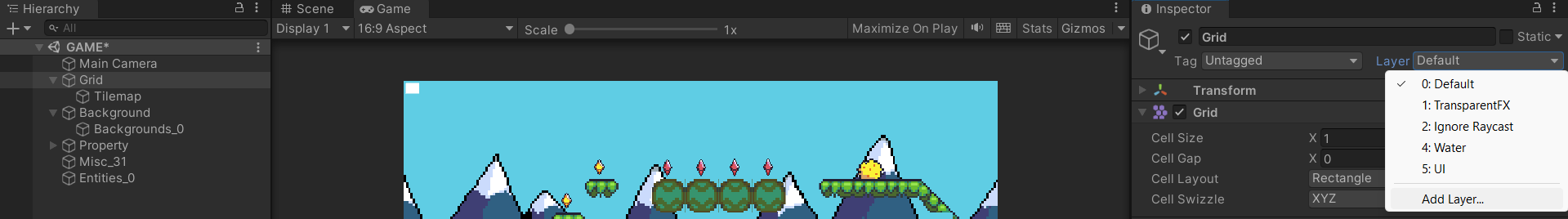
|  |
| --- |
| using System.Collections;  using System.Collections.Generic;  using UnityEngine;  public class *Player* : MonoBehaviour{  *Rigidbody*2D rb;  [SerializeField] float speed = 1;  float horizontalValue;  bool facingRight;  private void Awake() {  rb = GetComponent<*Rigidbody*2D>(); }  void 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. Maka jika game di *play*, karakter sudah dapat bergerak ke kiri dan kanan sesuai dengan inputan yang diberikan melalui *keyboard*.

****

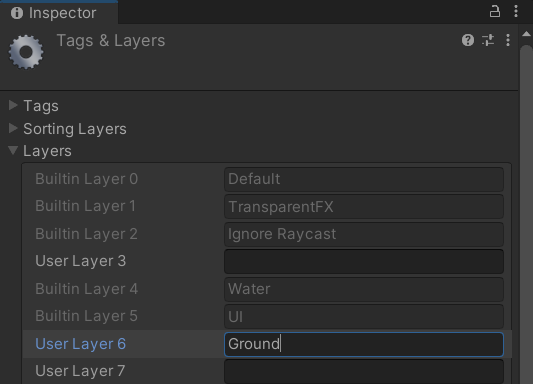
* 1. Game *Play*

1. Selanjutnya adalah membuat agar karakter dapat melompat. Langkah pertama adalah pilih *layer* *grid* lalu pada tambahkan *layer* baru pada *Inspector*.

****

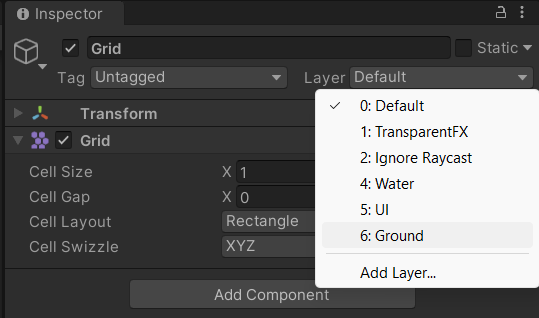
* 1. Menambahkan *Layer* Baru

1. Tambahkan *Ground* pada *User* *Layer* 6.

****

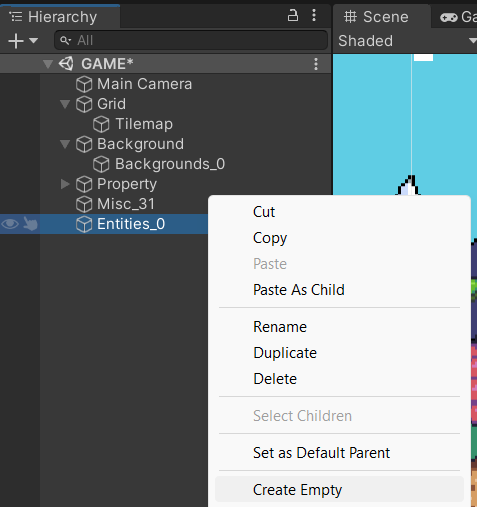
* 1. Menambahkan *Layer* Baru

1. Kembali ke bagian *Inspector*, lalu ubah *layer* yang sebelumnya *default* menjadi *Ground*.

****

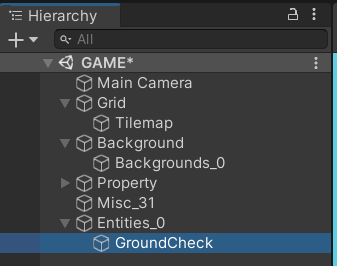
* 1. Mengganti *Layer*

1. Selanjutnya klik kanan pada *layer* Entities\_0 pada *hierarchy* lalu pilih *create empty*.

****

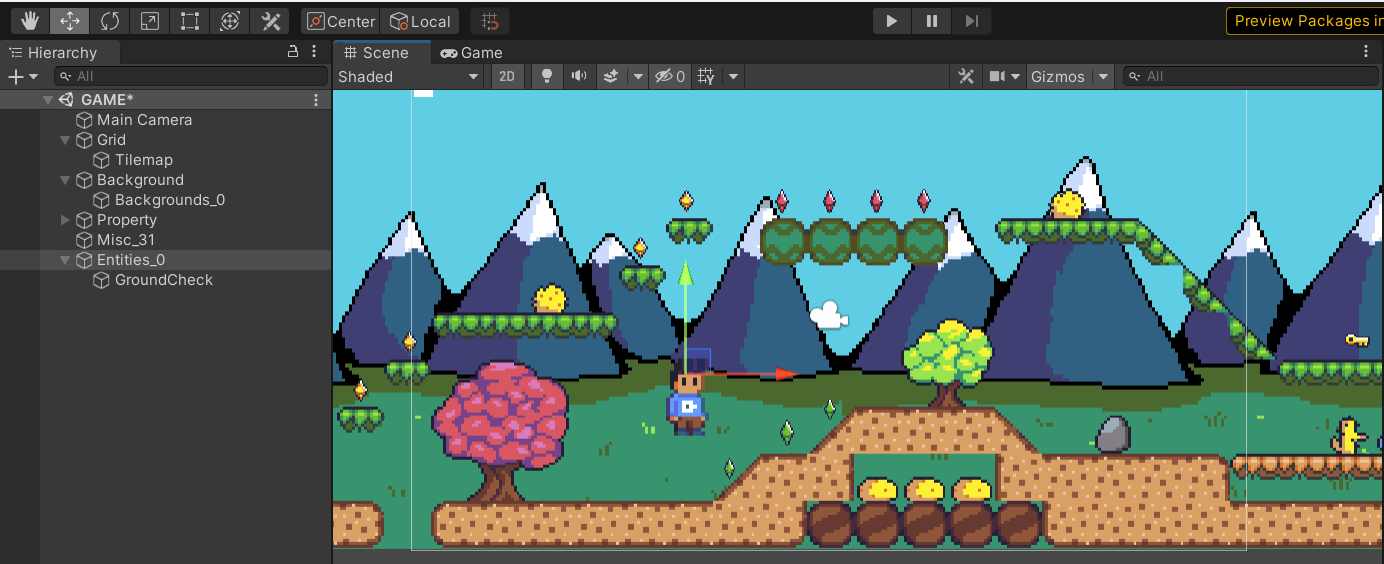
* 1. *Create Empty*

1. Beri nama *GroundCheck*.

****

* 1. *Create Empty*

1. Kemudian dengan menggunakan *move tool*, atur posisi karakter seperti gambar dibawah ini.

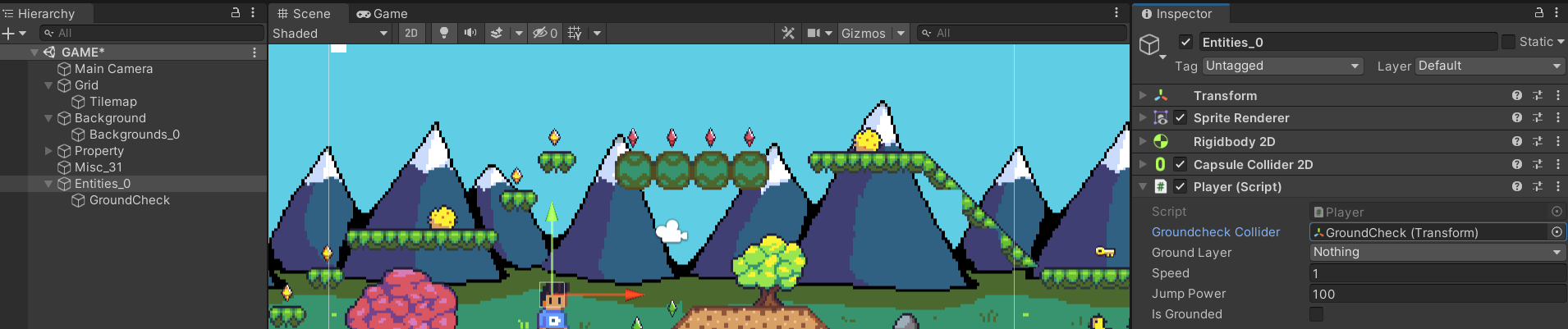
****

* 1. Mengatur Posisi Karakter

1. Selanjutnya, ubah *source code* seperti dibawah ini untuk membuat karakter bisa melompat jika diberikan input spasi pada *keyboard*.

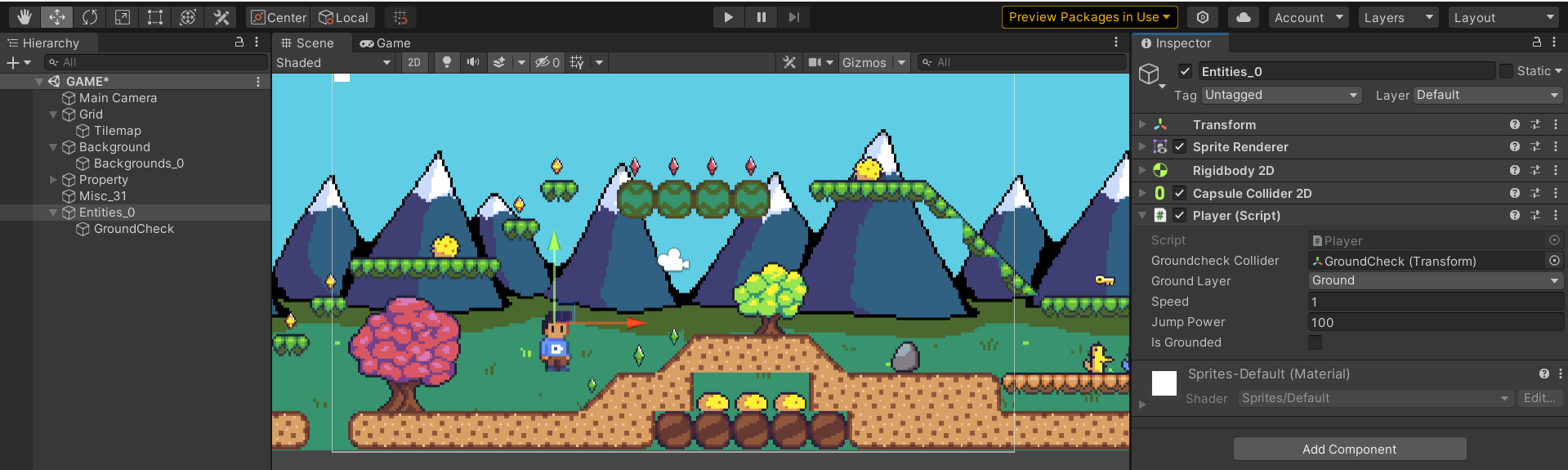
|  |
| --- |
| using System.Collections;  using System.Collections.Generic;  using UnityEngine;  public class *Player* : MonoBehaviour{  *Rigidbody*2D rb;    [SerializeField] Transform *Ground*checkCollider;    [SerializeField] *Layer*Mask *GroundLayer*;    const float *Ground*CheckRadius = 0.2f; // +    [SerializeField] float speed = 1;    [SerializeField] float jumpPower = 100;    float horizontalValue;    [SerializeField] bool is*Ground*ed; // +    bool facingRight;    bool jump;    private void Awake()  {      rb = GetComponent<*Rigidbody*2D>();  }    void Update () {      horizontalValue = Input.GetAxisRaw("Horizontal");      if (Input.GetButtonDown("Jump"))          jump = true;      else if (Input.GetButtonUp("Jump"))          jump = false;  }      void FixedUpdate()    {  *Ground*Check();          Move(horizontalValue, jump);}  void *Ground*Check(){      is*Ground*ed = false;      Collider2D[] colliders = *Physics*2D.OverlapCircleAll(*Ground*checkCollider.position, *Ground*CheckRadius, *GroundLayer*);      if (colliders.Length > 0)      is*Ground*ed = true;}    void Move(float dir, bool jumpflag  {      if(is*Ground*ed && jumpflag)      {   is*Ground*ed = 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(-1, 1, 1);        facingRight = false;}      else if (!facingRight && dir > 0) {        // ukuran *Player*        transform.localScale = new Vector3(1, 1, 1);        facingRight = true;}      #endregion}} |

1. Klik *layer* Entities\_0 lalu ubah *GroundCheck Collider* pada *Inspector* seperti gambar dibawah ini.

****

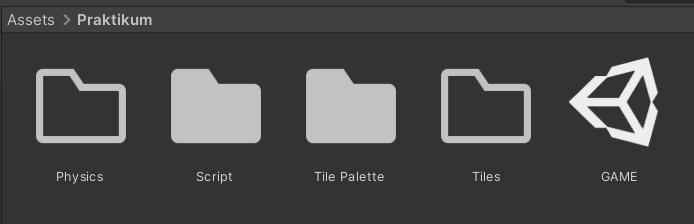
* 1. Mengganti *GroundCheck Collider*

1. Setelah itu ubah *Ground* *Layer* menjadi *Ground*.

****

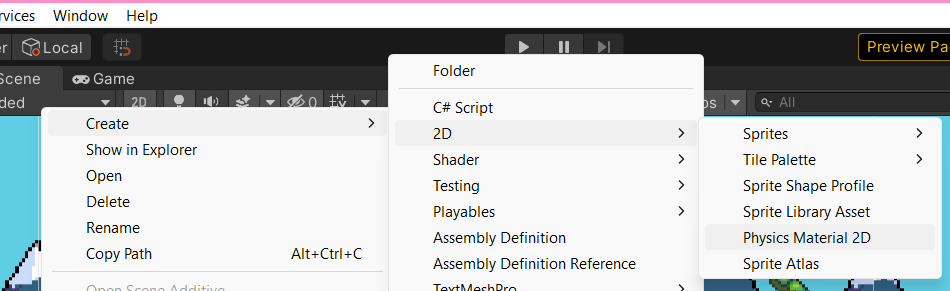
* 1. Mengganti *Ground* *Layer*

1. Buat Folder baru di dalam folder praktikum dan beri nama *Physics*.

****

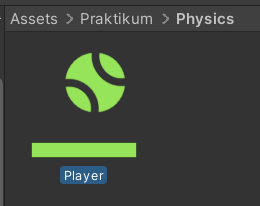
* 1. Membuat Folder Baru

1. Lalu di dalam folder *Physics* buat *file* baru seperti gambar dibawah ini.

****

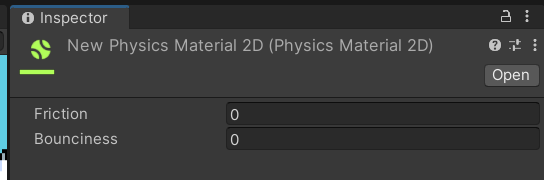
* 1. Menambahkan *File* Baru

1. Selanjutnya beri nama *file* tersebut dengan nama *Player*.

****

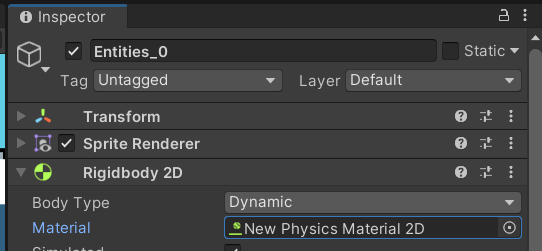
* 1. Menambahkan *File* Baru

1. Pada bagian *Inspector* *Player* ubah nilai menjadi seperti gambar dibawah ini.

****

* 1. Mengatur *Inspector* *Player*

1. Selanjutnya pilih *layer* Entities\_0 ubah materian pada komponen *Rigidbody* 2D pada *Inspector* menjadi *Player* yang sudah dibuat sebelumnya.

****

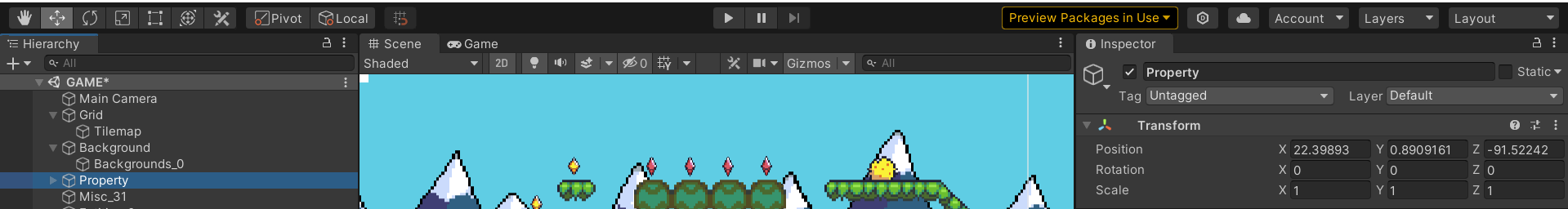
* 1. Mengganti Material *Rigidbody* 2D

1. Selanjutnya *play* game, maka karakter sudah dapat melompat jika terdapat input spasi melalui *keyboard*.

****

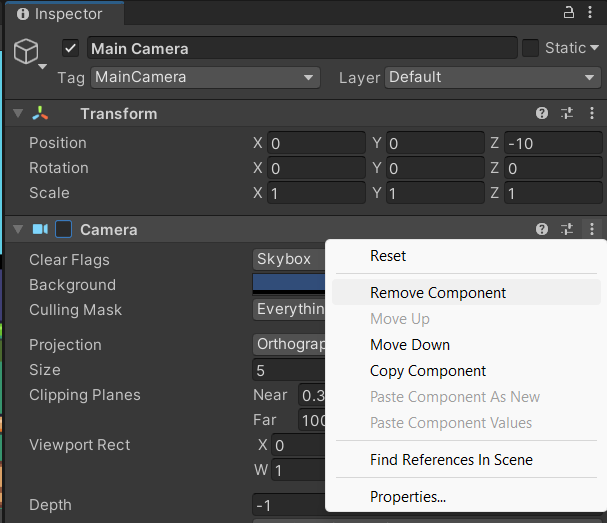
* 1. Game *Play*

1. Selanjutnya adalah membuat *Camera* *Movement*. Langkah pertama adalah ubah *tag* *layer* *property* menjadi *untagged*.

****

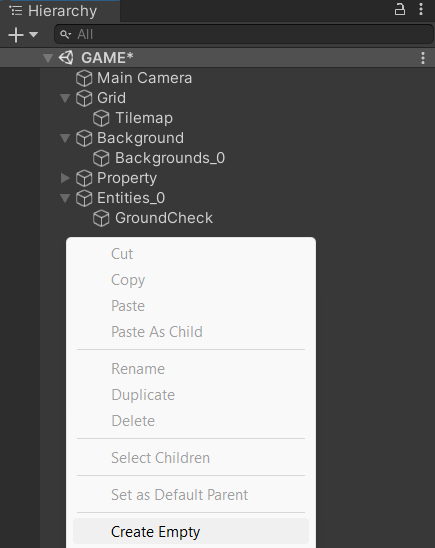
* 1. Ubah Tag *Layer* *Property*

1. Hapus centang pada komponen kamera lalu pilih *remove component* untuk menghapus component *Camera*.

****

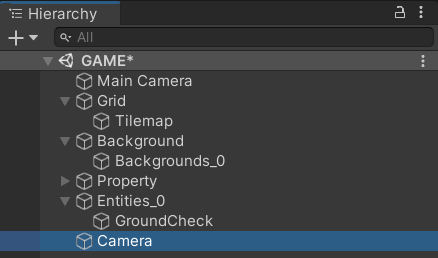
* 1. Menghapus *Component*

1. Klik kanan pada *hierarchy* kemudian pilih *create empty* untuk membuat *layer* baru.

****

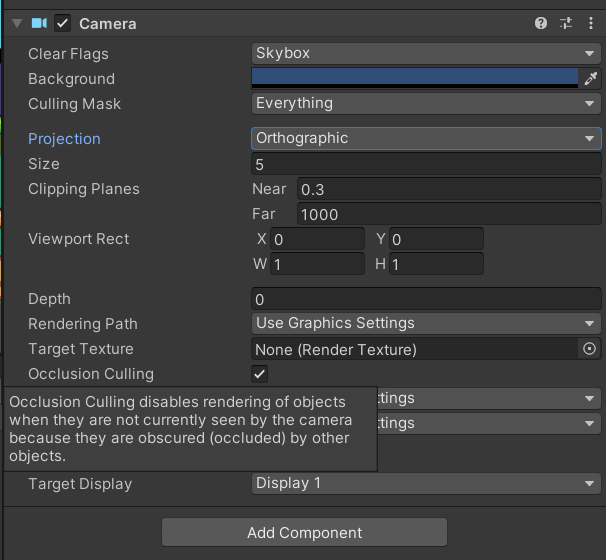
* 1. Membuat *Layer* Baru

1. Ganti nama *layer* baru tersebut menjadi *Camera*.

****

* 1. Membuat *Layer* *Camera*

1. Pilih *Layer* *Camera*, lalu ubah pengaturan pada *Inspector* sesuai dengan gambar dibawah ini.

****

* 1. Mengatur *Inspector* *Camera*

1. Buat *file* C# *Script* pada folder *Script*, lalu beri nama *CameraFollow* lalu *drag* *and drop* *script* ke dalam *layer* *Camera*.

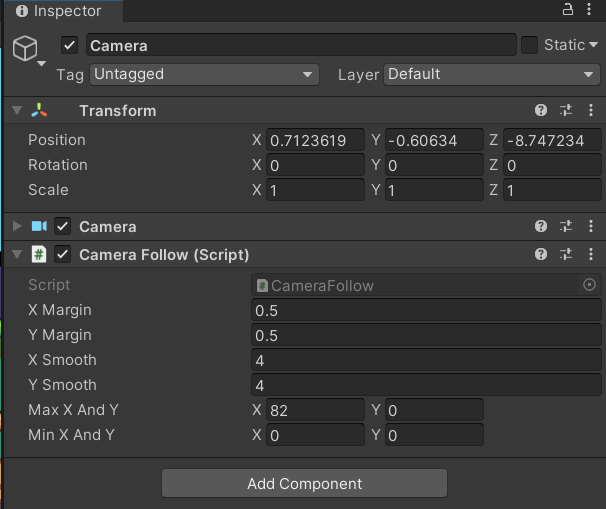
****

* 1. Menambahkan *File* C# *Script*

1. Tambahkan *source code* dibawah ini ke dalam *file* *Camera*Follow.cs

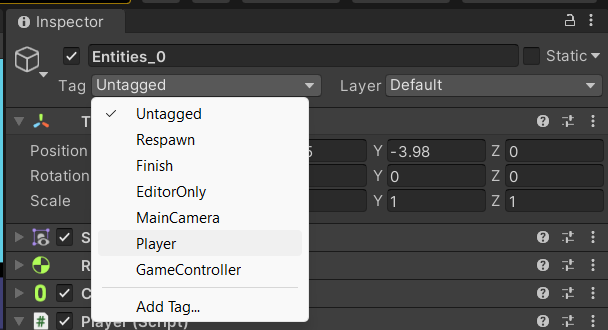
|  |
| --- |
| using System.Collections;  using System.Collections.Generic;  using UnityEngine;  public class *Camera*Follow : MonoBehaviour  { public float xMargin = 0.5f;  public float yMargin = 0.5f;  public float xSmooth = 4f;  public float ySmooth = 4f;  public Vector2 maxXAndY;  public Vector2 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() {  Track*Player*(); }  void Track*Player*() {  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. Buka *Inspector* *layer* *Camera*, ubah pengaturannya seperti dibawah ini.

****

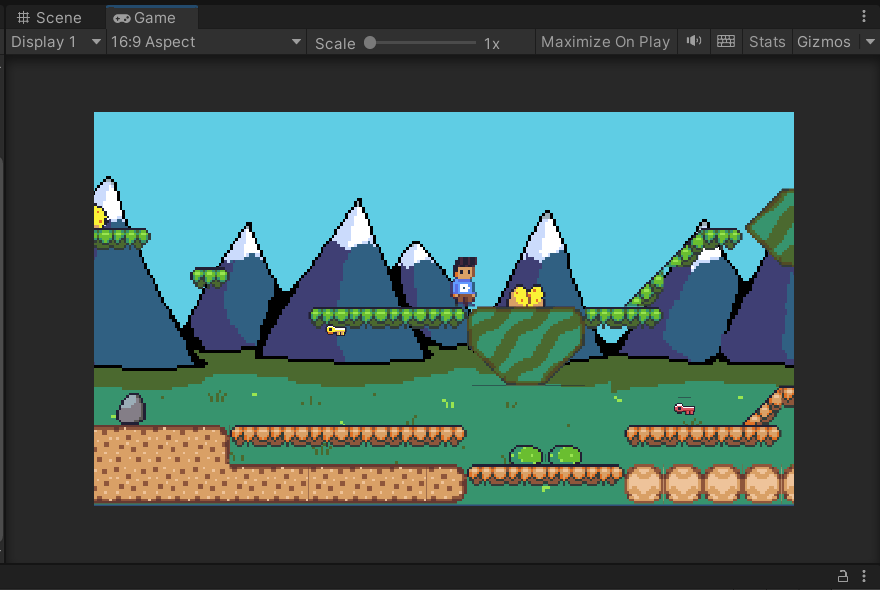
* 1. Ubah *Inspector* *Camera*

1. Lalu pada *inspector* *layer* Entities\_0 ubah *tag* menjadi *Player*.

****

* 1. Ubah *Tag* *Layer* Entities\_0

1. Maka, ketika game di *play* sudah terdapat *Camera* *Movement* yang mengikuti gerakan karakter.

****

* 1. Hasil Akhir Game

1. **KUIS**
2. Player.cs

|  |
| --- |
| using System.Collections;  using System.Collections.Generic;  using UnityEngine; |

Penjelasan

*Source code* diatas adalah deklarasi library yang akan digunakan pada fungsi-fungsi. *System collections* menyediakan class UI yang dapat digunakan seperti *object, array*, dll. Selanjutnya *unity engine* memungkinkan untuk menggunakan berbagai fitur dari *Unity*, seperti komponen, *grapichs*, dll.

|  |
| --- |
| public class Player : MonoBehaviour{  Rigidbody2D rb;  [SerializeField] Transform groundcheckCollider;  [SerializeField] LayerMask groundLayer;  const float groundCheckRadius = 0.2f; // +  [SerializeField] float speed = 1;  [SerializeField] float jumpPower = 100;  float horizontalValue;  [SerializeField] bool isGrounded; // +  bool facingRight;  bool jump; |

Penjelasan

Deklarasi *class* dengan nama *Player* sesuai dengan yang sudah dibuat pada *Unity* dan mewariskan *class* dari *MonoBehaviour* contohnya adalah fungsi *Update*. Selanjutnya adalah deklarasi variabel rb dengan tipe data *Rigidbody* 2D yang akan digunakan pada fungsi-fungsi yang akan didefinisikan selanjutnya. Kemudian juga terdapat deklarasi varibael-variabel lainnya. Atribut *SerializeField* artinya nilai pada variabel tersebut dapat diatur melalui *inspector* pada *unity*.

|  |
| --- |
| private void Awake(){rb = GetComponent<Rigidbody2D>();} |

Penjelasan

Fungsi ini akan dijalankan pertama yaitu menginisalisasi variabel rb untuk dapat mengakses dan melakukan perubahan terhadap komponen Rigidbody 2D.

|  |
| --- |
| void Update () {  horizontalValue = Input.GetAxisRaw("Horizontal");  if (Input.GetButtonDown("Jump"))  jump = true;  else if (Input.GetButtonUp("Jump"))  jump = false;}  void FixedUpdate(){  GroundCheck();  Move(horizontalValue, jump); } |

Penjelasan

Fungsi *Update* berfungsi untuk membaca dan menyimpan input gerakan dan mendeteksi dan menggerakkan *player* untuk melompat jika tombol lompat di klik pada *keyboard*. Fungsi *FixedUpdate* berfungsi untuk mengecek apakah karakter berada ditanah atau tidak dengan menggunakan fungsi *GroundCheck* dan akan menggerakan karakter sesuai dengan inputan pada *keyboard*.

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

Penjelasan

Fungsi *GroundCheck* berfungsi untuk mengecek apakah karakter menyentuh tanah atau tidak lalu disimpan pada variabel *isGround*. Oleh sebab itu, diinisialisasi awal nilai *isGround* adalah *false* yang artinya karakter sedang berada di tanah, kemudian jika nilai *collider* lebih dari nol, maka artinya karakter sedang tidak menyentuh tanah dan nilai *isGround* diubah menjadi *true.*

|  |
| --- |
| 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(-1, 1, 1);  facingRight = false;}  else if (!facingRight && dir > 0) {  // ukuran player  transform.localScale = new Vector3(1, 1, 1);  facingRight = true;}  #endregion} |

Penjelasan

Fungsi ini berfungsi untuk mengerakan karakter baik ke kanan, ke kiri ataupun melompat Berdasarkan inputan melalui keyboard serta berfungsi untuk membalikkan arah tampilan karakter menyesuaikan dengan arah gerakan karakter.

1. CameraFollow.cs

|  |
| --- |
| 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);  }  } |

Penjelasan :

*Code* diatas berfungsi untuk membuat kamera mengikuti gerakan pemain. Kamera tidak bergerak setiap karakter melakukan gerakan, namun kamera hanya akan bergerak jika posisi karakter sudah bergerak cukup jauh dari posisi kamera sehingga menciptakan efek gerakan yang *smooth*.

1. **Link Github Pengumpulan**

[GitHub - CorneliaBoro/AnimasidanGame](https://github.com/CorneliaBoro/AnimasidanGame)