# 8 CAMERA & CHARACTER MOVEMENT

|  |  |  |
| --- | --- | --- |
| **NIM** | : | 2118006 |
| **Nama** | : | Mikhael Christian |
| **Kelas** | : | A |
| **Asisten Lab** | : | Naufal Dhiaurrafif (2218059) |

## Tugas 8 : Membuat Langkah-Langkah

1. **Langkah-Langkah Membuat Pergerakan Player**
2. Buka *Project Unity* sebelumnya kemudian klik Astronaut-idle tambahkan *Component* *Rigidbody* 2D, sesuaikan settingannya seperti gambar berikut.

**A screenshot of a computer

Description automatically generated**

### 8.1 Tambahkan *Player* ke *Hierarchy*

1. Kemudian tambahkan *Capsule* *Colider* 2D di Astronaut-idle, lalu klik *icon* sebelah kanan *Edit* *Colider*.

A screenshot of a computer

Description automatically generated

### 8.2 Tambahkan *Capsule Collider* 2D

1. Lalu cocokan garis oval dengan karakter.

A screenshot of a video game

Description automatically generated

### 8.3 Atur Garis Oval

1. Buka folder Praktikum, lalu bikin folder baru bernama *Script*. Masuk ke dalam folder *Script*, lalu buat C# *Script*, beri nama “Player”.

A screenshot of a computer

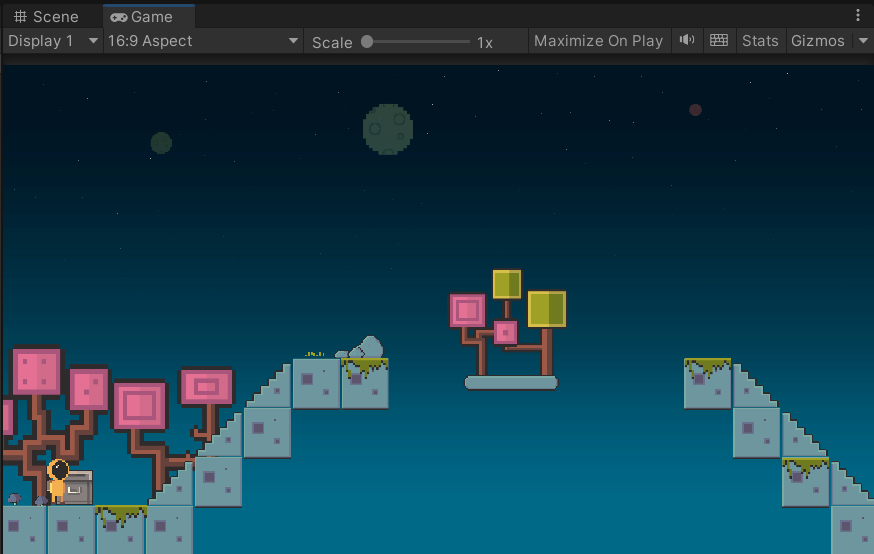
Description automatically generated

### 8.4 Buat C# *Script*

1. *Drag* & *drop* *script* *player* ke dalam *Hierarchy* Astronaut-idle, lalu klik 2x pada *script* dan masukkan *source code* di bawah ini.

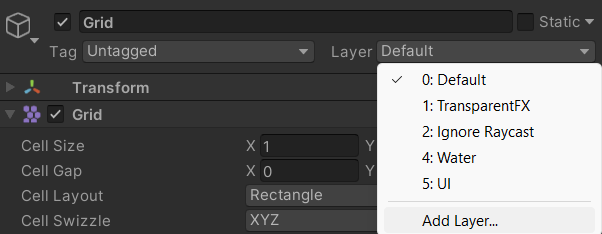
|  |
| --- |
| using System.Collections;  using System.Collections.Generic;  using UnityEngine;  public class Player : MonoBehaviour  {      Rigidbody2D rb;      [SerializeField] float speed = 1;      float horizontalValue;      bool facingRight;      private void Awake()      {          rb = GetComponent<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. Untuk mencoba *Source* *code* di atas berhasil, tekan *keyboard* “a” atau “*left arrow*” untuk ke arah kiri, tekan “d” atau “*right arrow*” untuk ke arah kanan.



### 8.6 *Test Source Code*

1. Untuk membuat *player* loncat menggunakan spasi, kita perlu membuat *GorundCheck* dengan cara, klik *Grid* pada *Hierarchy*, pergi ke *inspector*, pilih *Layer*, klik *Add Layer*. Kemudian isi “Ground” pada *User Layer* 6.



### 8.6 Tambahkan *Layer* Baru

1. Ubah *Layer* menjadi *Ground*, jika muncul *pop up Change Layer*, klik *yes*.

A screenshot of a computer

Description automatically generated

### 8.7 Pilih *Layer Ground*

1. Klik kanan pada Astronaut-idle, lalu *Create Empty*, beri nama “GroundCheck”.

A close up of a sign

Description automatically generated

### 8.8 Buat *GroundCheck*

1. Klik pada *Hierarchy* *GroundCheck*, lalu gunakan *Move Tools* untuk memindahkan ke bagian bawah *Player* seperti gambar berikut.

A screenshot of a video game

Description automatically generated

### 8.9 Pindahkan *GroundCheck*

1. 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; |

1. 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.positi on,groundCheckRadius, groundLayer);    if (colliders.Length > 0)       isGrounded = true;  } |

1. Klik Astronaut-idle, lalu ke *inspector* ke *Effect Player script* di bagian “Groundcheck Collider” tekan *icon* lalu pilih yang *GroundCheck* *Transform*, dan pada *Ground* *Layer* pilih *Ground*.

A screenshot of a computer

Description automatically generated

### 8.10 Atur *Script Player* pada *Inspector*

1. Lalu untuk membuat *player* melompat tambahkan *script* berikut.

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

1. Tambahkan juga *script* berikut di bagian *void* *update*.

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

1. Tambahkan juga *jump* pada parameter *Move*.

A computer screen with white text

Description automatically generated

### 8.11 Tambahkan *Script*

1. Tambahkan *script* berikut pada *void* *Move*.

A computer screen shot of text

Description automatically generated

### 8.12 Tambahkan *Script*

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

1. Buat folder baru di praktikum bernama “Physics”, kemudian di dalam folder “Pyshics” *Create* > 2D > *Physical Material* 2D, berinama “Player”.

A screenshot of a computer

Description automatically generated

### 8.13 Tambahkan *Physical Material* 2D

1. Klik *Player* (*Physics Material* 2D), di bagian menu *Inspector*, *Friction* & *Bounciness* ubah menjadi 0.

A screenshot of a computer

Description automatically generated

### 8.14 Atur *Physics*

1. Klik *Hierarchy* pilih *layer* Astronaut-idle, pada *Inspector* Cari *Rigidbody* 2D lalu klik *icon* untuk membuka *box* *Select* *Physics* *Material* 2D , lalu pilih *asset* *Player* yang sudah dibuat.

A screenshot of a computer

Description automatically generated

### 8.15 Pilih *Physics* *Player*

1. Tekan *play*, maka *player* bisa melompat dengan menekan spasi.

A screenshot of a video game

Description automatically generated

### 8.16 Tekan *Play*

1. **Langkah-Langkah Membuat Camera Movement**
2. Pada *Hierarchy* *Props* ubah *Inspector* pada *Tag Main camera* menjadi *untagged*.

A screenshot of a computer

Description automatically generated

### 8.17 Atur *Inspector Props*

1. *Create Empty* pada *Hierarchy*, dan *Rename* menjadi “Camera”, kemudian sesuaikan *Setting Layer Camera* seperti di bawah ini.

A screenshot of a computer

Description automatically generated

### 8.18 Atur *Inspector Camera*

1. Buat file *script* baru di folder *Script* dengan nama ”CameraFollow”, lalu tuliskan *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);      }  } |

1. *Drag & drop script* CameraFollow ke dalam *Layer Camera*. Lalu klik pada camera, buka inspector setting menjadi berikut.

A screenshot of a computer

Description automatically generated

### 8.19 Atur *Inspector Camera*

1. Ubah *Tag* di Astronaut-idle *Untagged* menjadi ”Player”

A screenshot of a computer

Description automatically generated

### 8.20 Atur *Tag* Astronaut-idle

1. Tekan *play* untuk menjalankan, maka sekarang kamera akan mengikuti pergerakan karakter.

A screenshot of a video game

Description automatically generated

### 8.21 Tekan *Play*

1. **Langkah-Langkah Render**
2. Pergi ke menu *File* kemudian pilih *Build Setting* (*Ctrl* + *Shift* + B). Pada *Setting* *Build* ini pilih *PC, Mac & Linux*, tekan *Build*, pastikan pada menu *Scene* in *Build* berada pada *project* yang sedang dikerjakan.

A screenshot of a computer

Description automatically generated

### 8.22 Atur *Build Settings*

1. Kemudian pilih dimana *Project* disimpan dan tunggu.

A screenshot of a computer

Description automatically generated

### 8.23 Tampilan Folder *Game*

1. **Kuis**

|  |
| --- |
| 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 merupakan script agar kamera dapat mengikuti gerakan player. Kamera tersebut akan mengikuti gerakan player pada sumbu x. Namun terdapat beberapa kesalahan penulisan, seperti pada penulisan “Void” seharusnya menjadi “void” dan seharusnya tidak ada spasi atau jarak pada parameter Vector3. Seharusnya “player.position.x”.