**PRAKTIKUM ANIMASI DAN GAME PERTEMUAN :9**

**Unity 2D Game Animation & Respawn**

|  |  |  |
| --- | --- | --- |
| **NIM** | **:** | 1818002 |
| **Nama** | **:** | Joao Baptista De Carvalho Gama |
| **Kelas** | **:** | E |
| **Asisten Lab** | **:** | YEDIJA ADYA VESAKA (2118056) |
| **Referensi** | **:** |  |

# 9.1 Tujuan

1. Praktikan dapat mengetahui *Game* 2D dan *Game* 3D.
2. Praktikan dapat mengetahui jenis *Game.*
3. Praktikan dapat menerapkan *Game* pada Unity.

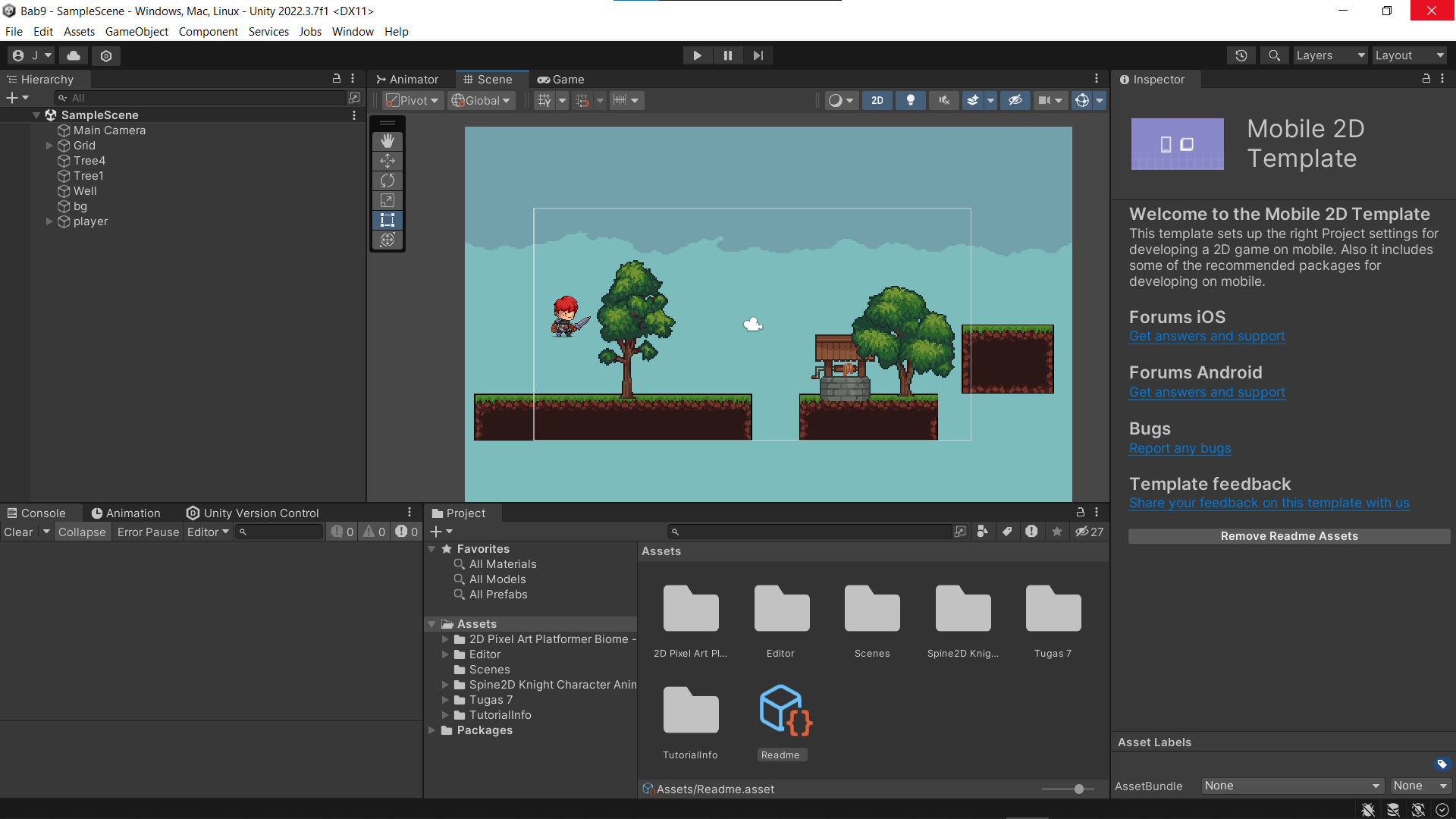
# 9.2 Alat dan Bahan

1. Laptop/pc.
2. Modul Praktikum Animasi dan *Game* 2024.
3. Unity 2017.

# 9.3 Langkah-Langkah Membuat Tugas

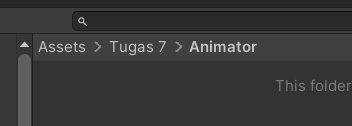
**A. Membuat Mekanisme Respawn**

* 1. Buat file *projek Unity* Tugas 8



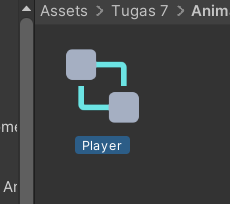
Gambar 9.1 Buka projek tugas 8

* 1. Klik karakter, pergi ke *Inspector*, pilih *Add Component*, cari Animator,Buat sebuah folder baru dengan nama “Animator”



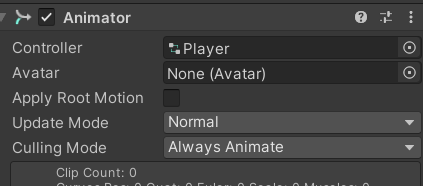
Gambar 9.3 Membuat folder Animator

1. Klik kanan folder Animator>*Create*>*Animation Controller*, dan ubah namanya menjadi “Player”



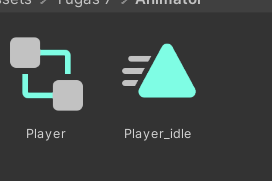
Gambar 9.4 Membuat Karakter Animator

1. Drag karakter, kemudian masukkan kedalam Controller komponen Animator



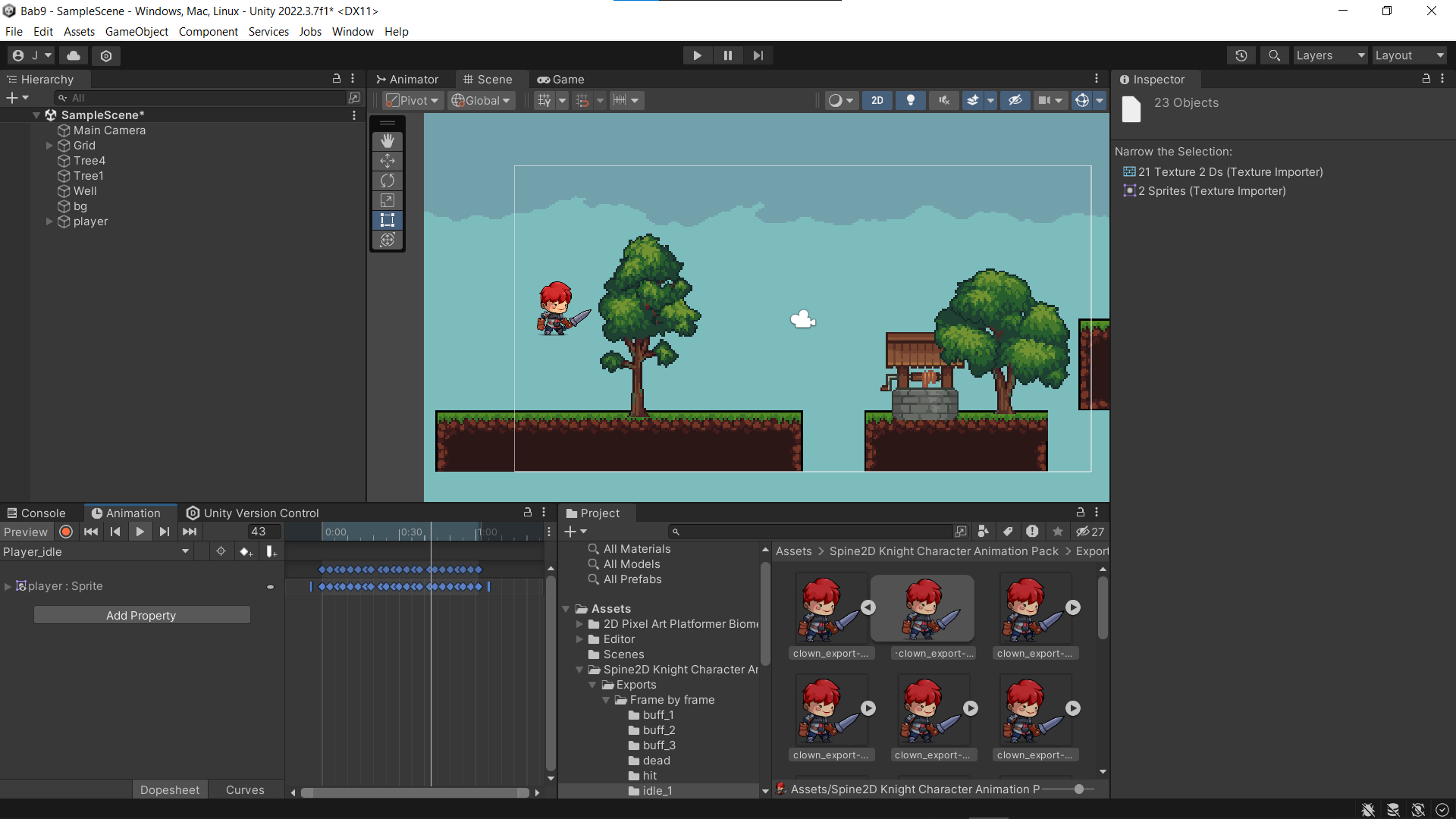
Gambar 9.5 Mendrag karakter Animator

1. Klik karakter, kemudian pergi ke tab Animation, pilih Create



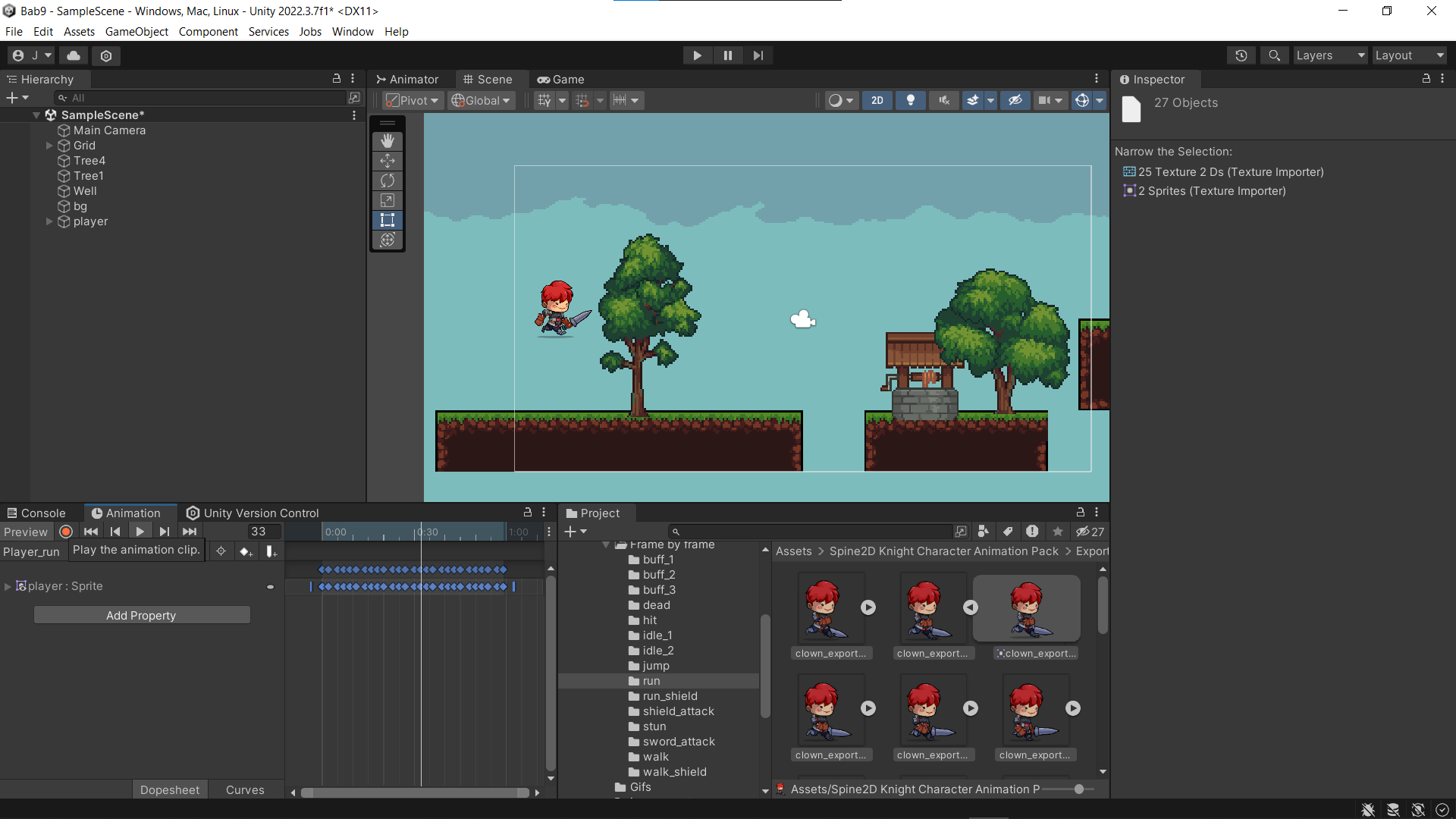
Gambar 9.6 Membuat suatu *Animation*

1. Drag and Drop karakter lalu regangkan, pada *timeline* tekan *Ctrl+A* di *keyboard*, klik bagian kotak kecil disamping *keyframe* terakhir dan geser sampai waktu 01:00



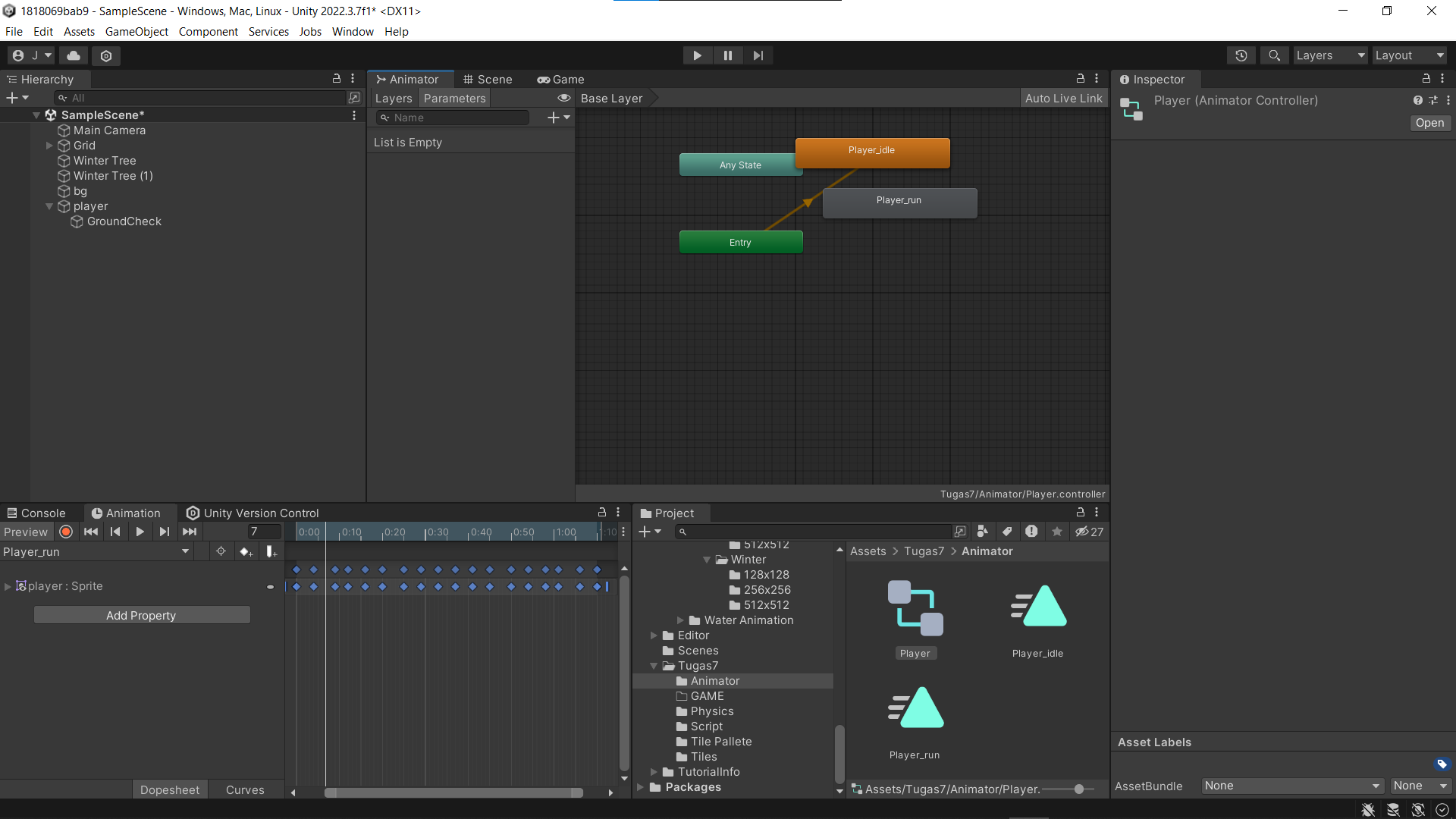
Gambar 9.7 *Drag* *and Drop* karakter

1. Buat animasi baru, Klik pada “Player\_idle” kemudian pilih Create New Clip, dan beri nama “Player\_run”, Simpan pada Folder Animator Pilih karakter *run* kemudian drag kedalam *timeline* Player\_run *Animation* lalu geser ke *keyframe* waktu 1:10



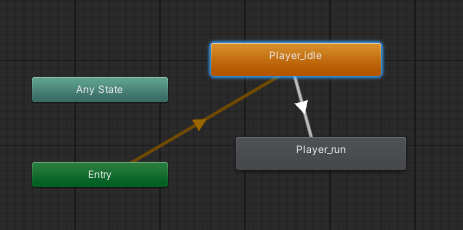
Gambar 9.8 Menambahkan karakter *playerrun*

1. Klik kanan pada area sekitar *Animator*, pilih *Create State>From New Blend Tree*



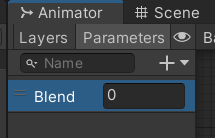
Gambar 9.9 Membuat *Blend Tree*

1. Kemudian buat transisi antara player\_idle dan player\_run dengan cara klik kanan pada player\_idle dan pilih Make Transition dan tarik ke player\_run



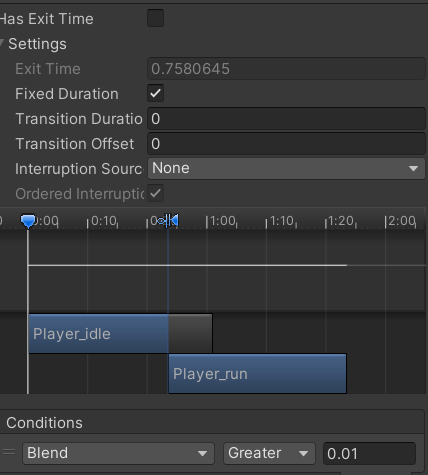
Gambar 9.11 Menambahkan *Motion field*

1. Masuk ke tab parameter, tambahkan tipe data float dengan cara tekan icon tambah dan ubah namanya menjadi “Blend”



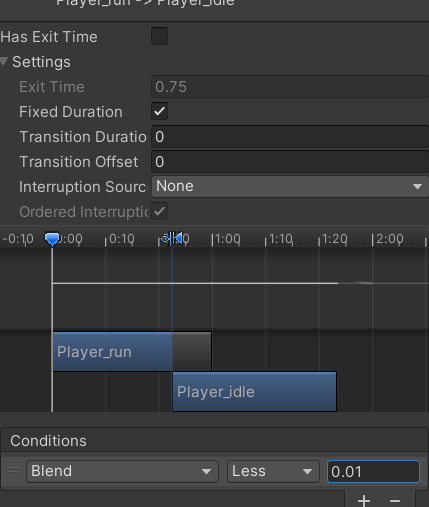
Gambar 9.12 Menambahkan sebuah *motion*

1. Klik panah putih tersebut, pada bagian conditions klik icon tambah kemudian atur menjadi “Blend”, Atur nilai conditions blend tersebut menjadi 0.01 Pada bagian Settings, hilangkan centang pada Has Exit Time dan atur nilai Transition Duration menjadi 0



Gambar 9.12 Menambahkan sebuah *motion*

1. Buat transisi juga dari player\_run ke player\_idle dengan cara klik kanan pada player\_run dan pilih Make Transition. Tambahkan parameter transisi dengan tipe data Float. Klik ikon tambah dan rename menjadi “Blend”. Setelah itu, ubah operator dari Greater menjadi Less dan atur nilainya menjadi 0.01. Pada bagian Settings, hilangkan centang pada Has Exit Time dan atur nilai Transition Duration menjadi 0

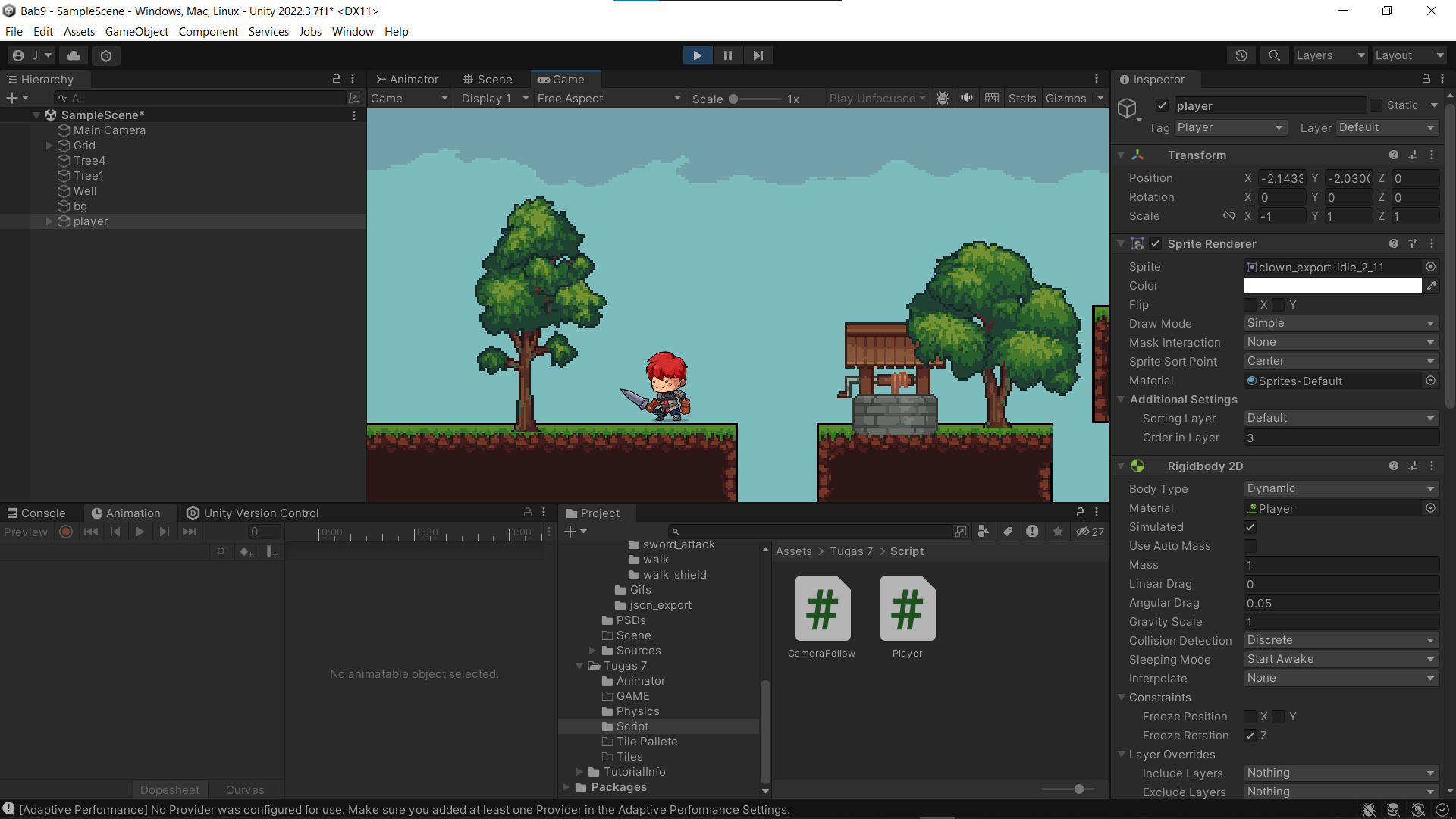


Gambar 9.12 Menambahkan sebuah *motion*

1. Agar animasi dapat sesuai ketika berjalan, buka script Player dan tambahkan source code berikut pada class Player

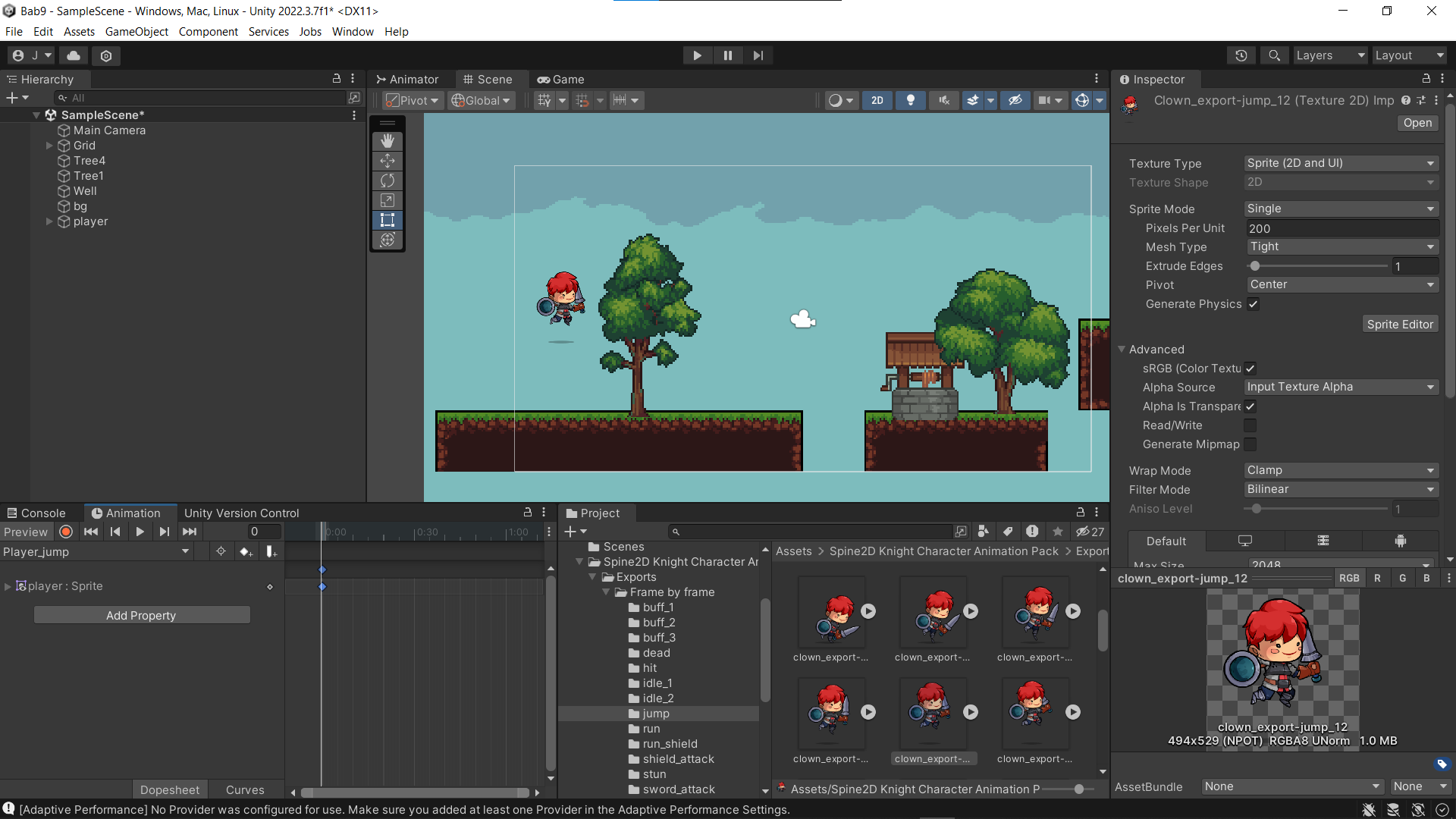
|  |
| --- |
| using System.Collections;  using System.Collections.Generic;  using UnityEngine;  public class Player : MonoBehaviour  {  public Animator animator;  Rigidbody2D rb;  [SerializeField]  Transform groundcheckCollider;  [SerializeField]  LayerMask groundLayer;  const float groundCheckRadius = 0.2f;  [SerializeField]  bool isGrounded;  [SerializeField]  float speed = 1;  [SerializeField]  float jumpPower = 100;  float horizontalValue;  bool facingRight;  bool jump;  private void Awake()  {  rb = GetComponent<Rigidbody2D>();  animator = GetComponent<Animator>();  }  // Update is called once per frame  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);  animator.SetFloat("Blend", Mathf.Abs(rb.velocity.x));  }  void GroundCheck()  {  isGrounded = false;  Collider2D[] colliders = Physics2D.OverlapCircleAll(groundcheckCollider.position, groundCheckRadius,  groundLayer);  if (colliders.Length > 0)  isGrounded = 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)  {  transform.localScale = new Vector3(-1, 1, 1);  facingRight = false;  }  else if (!facingRight && dir > 0)  {  transform.localScale = new Vector3(1, 1, 1);  facingRight = true;  }  #endregion  }  } |

1. Jika dijalankan maka player dapat memiliki animasi ketika berhenti ataupun ketika berjalan



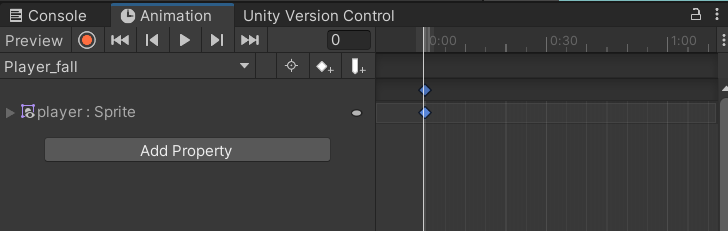
Gambar 9.13 Menambahkan *astrowalk*

1. Kemudian buat animasi baru tekan tulisan “Player\_run” kemudian pilih Create New Clip, dan beri nama “Player\_jump”, Pada folder player buka jump lalu pilih gambar player-jump-1, kemudian drag ke tab Animation.



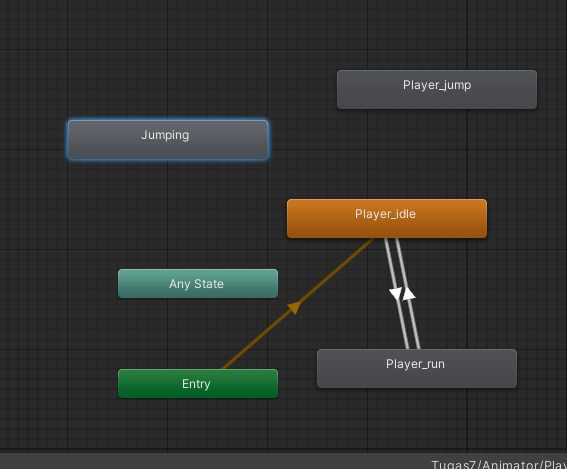
Gambar 9.13 Menambahkan *astrowalk*

1. Buat animasi baru dengan cara tekan tulisan “Player\_jump" kemudian pilih Create New Clip, dan beri nama “Player\_fall”, Pada tab Project buka folder karakter lalu pilih Idle dan pilih gambar playerfall, kemudian drag ke tab Animation.



Gambar 9.14 Menambahkan *astrofall*

1. Buat *Blend Tree*, pergi ke *Inspector* dan ubah namanya menjadi *Jumping*



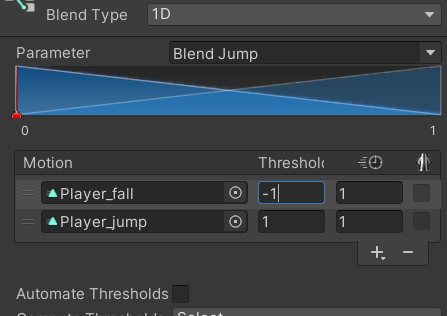
Gambar 9.14 Menambahkan *blend tree jumping*

1. Pada menu Parameteres tambahkan parameter tipe data Float tekan icon + dan ubah namanya menjadi “Blend Jump”, Pada menu Animator, Klik dua kali pada Blend Tree “Jumping”, Tekan pada Blend Tree, Klik 2X Blend Tree “Jumping”, pada inspector ubah parameter menjadi “Blend Jump”



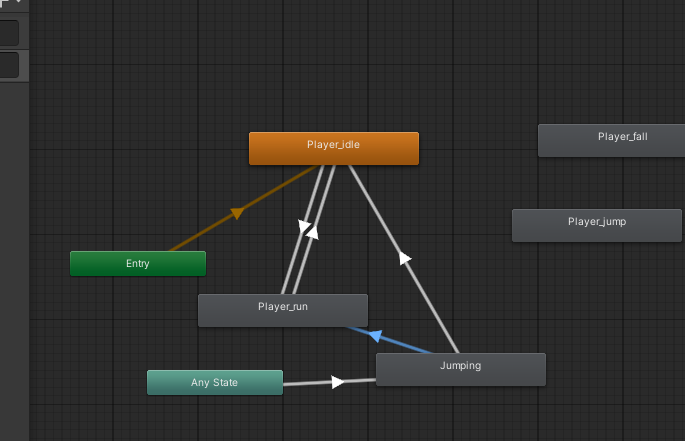
Gambar 9.17 Mengubah parameter *Jumping*

1. Buka menu Inspector, tekan icon + dan pilih Add Motion Field. Tambahkan dua Motion Field, Klik bagian icon None (Motion), maka akan muncul Windows Motion, Tambahkan Sesuai dengan urutan, Hilangkan centang “Automate Thresholds” dan atur nilai Threshold seperti berikut



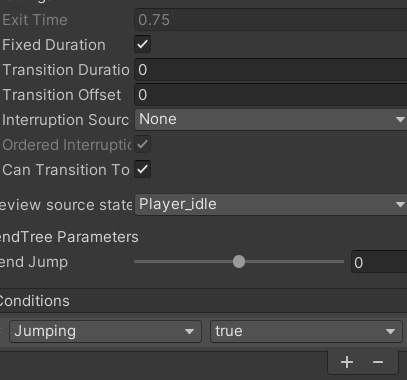
Gambar 9.17 Mengubah parameter *Jumping*

1. Kembali ke Base Layer, klik kanan Any State, pilih Make Transition dan arahkan panahnya ke Jumping, Klik kanan Jumping, pilih Make Transition dan arahkan panahnya ke Player\_idle dan Player\_run



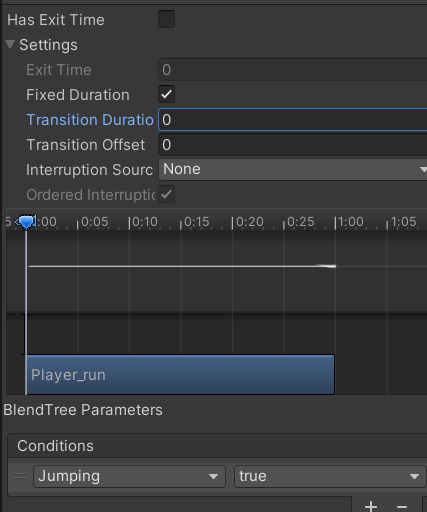
Gambar 9.17 Mengubah parameter *Jumping*

1. Tambahkan parameter transisi dengan tipe data Bool tekan icon + dan ubah namanya menjadi “Jumping”, Klik panah yang mengarah ke Jumping, pada inspector tambahkan condition, pilih condition Jumping dan ubah nilainya menjadi true, Klik Settings dan ubah nilai Transition Duration menjadi 0 dan hilangkan centang Has Exit Time



Gambar 9.17 Mengubah parameter *Jumping*

1. Klik panah yang mengarah ke Player\_idle dan Player\_run, pada inspector tambahkan condition, pilih condition Jumping, pada arah panah ke player\_idle ubah menjadi false, pada arah panah ke player\_run ubah menjadi true, Klik Settings dan ubah nilai Transition Duration menjadi 0 dan hilangkan centang Has Exit Time

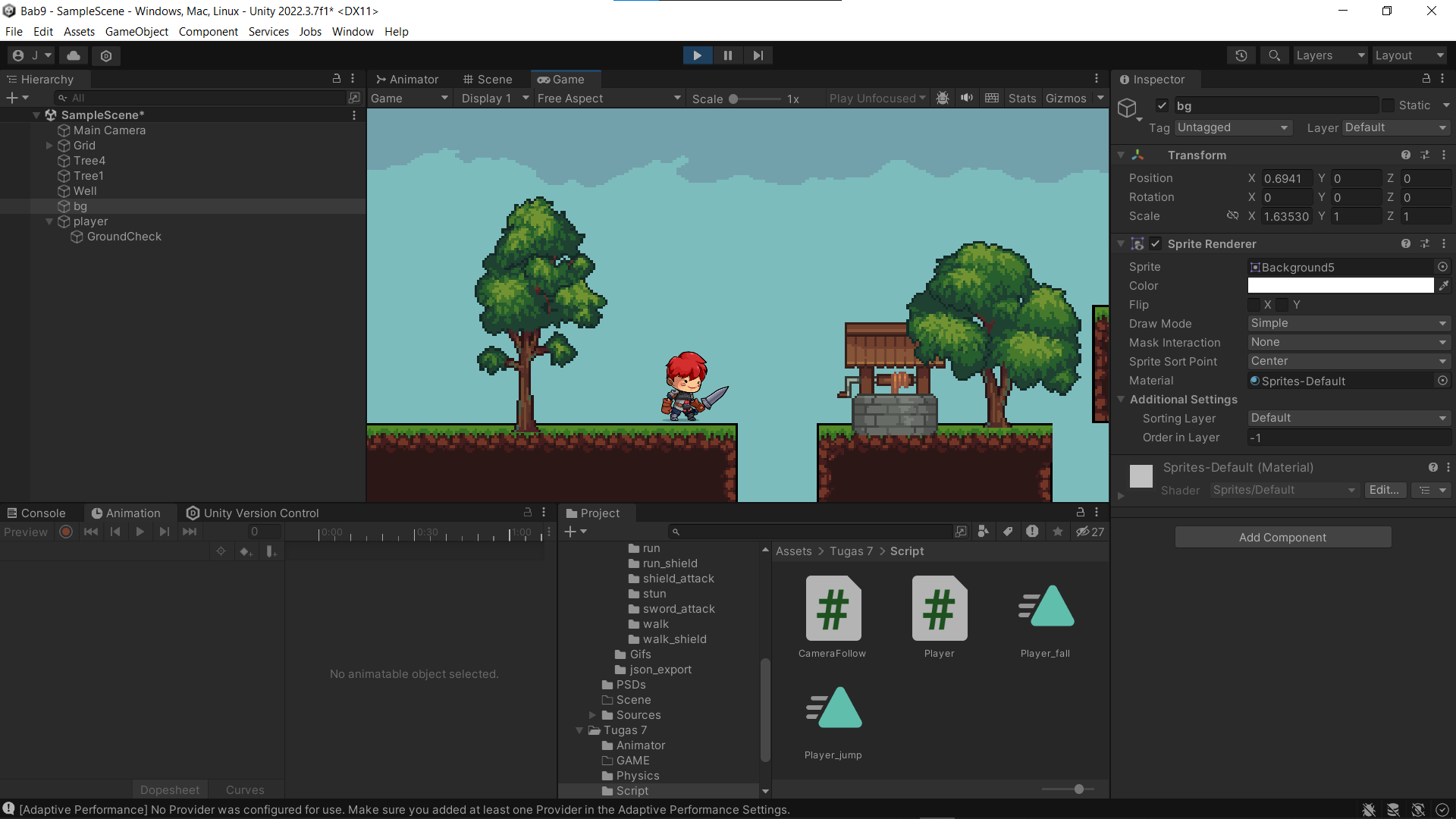


Gambar 9.17 Mengubah parameter *Jumping*

1. Tambahkan Source code seperti ini

|  |
| --- |
| using System.Collections;  using System.Collections.Generic;  using UnityEngine;  public class Player : MonoBehaviour  {  public Animator animator;  Rigidbody2D rb;  [SerializeField]  Transform groundcheckCollider;  [SerializeField]  LayerMask groundLayer;  const float groundCheckRadius = 0.2f;  [SerializeField]  bool isGrounded;  [SerializeField]  float speed = 1;  [SerializeField]  float jumpPower = 100;  float horizontalValue;  bool facingRight;  bool jump;  private void Awake()  {  rb = GetComponent<Rigidbody2D>();  animator = GetComponent<Animator>();  }  // Update is called once per frame  void Update()  {  horizontalValue = Input.GetAxisRaw("Horizontal");  if (Input.GetButtonDown("Jump")) {  jump = true;  animator.SetBool("Jumping", true);  }  else if (Input.GetButtonUp("Jump")) {  jump = false;  animator.SetFloat("Blend Jump", rb.velocity.y);  }  }  void FixedUpdate()  {  GroundCheck();  Move(horizontalValue, jump);  animator.SetFloat("Blend", Mathf.Abs(rb.velocity.x));  animator.SetFloat("Blend Jump", rb.velocity.y);  }  void GroundCheck()  {  isGrounded = false;  Collider2D[] colliders = Physics2D.OverlapCircleAll(groundcheckCollider.position, groundCheckRadius,  groundLayer);  if (colliders.Length > 0){  isGrounded = true;  }  animator.SetBool("Jumping", !isGrounded);  }  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)  {  transform.localScale = new Vector3(-1, 1, 1);  facingRight = false;  }  else if (!facingRight && dir > 0)  {  transform.localScale = new Vector3(1, 1, 1);  facingRight = true;  }  #endregion  }  } |

1. Jika di play maka karakter sudah bisa bergerak dengan animasi



Gambar 9.17 Mengubah parameter *Jumping*