TUGAS PERTEMUAN: 9

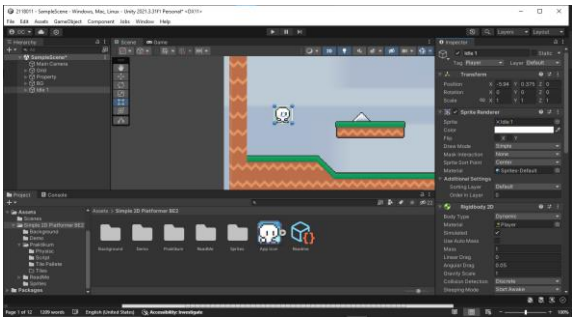
Game Animation

NIM	:	2118011
Nama	:	Dewa Chandra Agung Wibawa
Kelas	:	C
Asisten Lab	:	Naufal Dhiaurrafif (2218059)

1.1 Tugas 1 : Character Animation

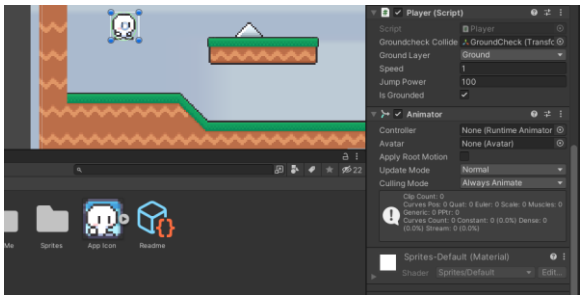
A. Membuat Character Movement

1. Pertama buka *project* yang telah dikerjakan di materi sebelumnya.



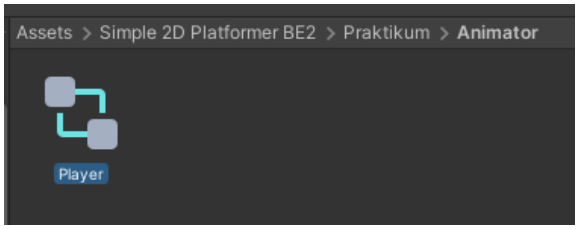
Gambar 1.1 Membuka Project

2. Setelah itu klik pada karakter dan tambahkan komponen. Cari komponen Bernama animator. Jika sudah maka hasilnya seperti gambar dibawah ini.



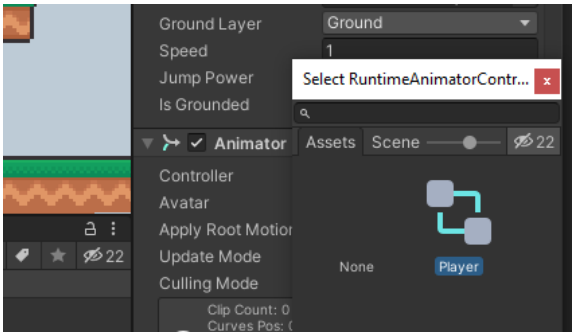
Gambar 1.2 Menambahkan Komponen Animator

3. Setelah itu buat *folder* Bernama animator pada *folder* praktikum. Lalu buat *file animator controller* pada *folder* tersebut. Ubah nama *file* tersebut menjadi “Player”.



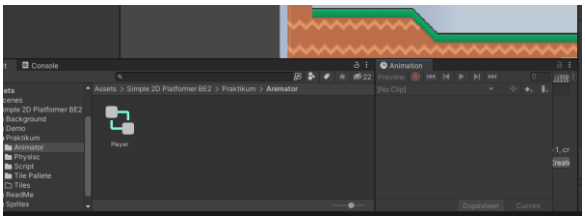
Gambar 1.3 Menambahkan File Animator Collector

4. Setelah itu klik karakter. Pergi ke komponen animator yang telah dibuat. Setelah itu *setting controller* menjadi *player*.Jika sudah maka hasilnya akan seperti gambar dibawah ini.



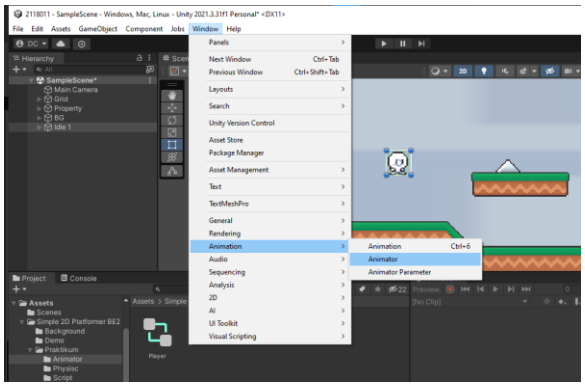
Gambar 1.4 Merubah Setting Controller pada Animator

5. Berikutnya buka *panel animation*. Tekan CTRL + 6 untuk membukanya. Setelah itu letakkan disamping dari bar *console* dan *project*. Jika sudah maka seperti gambar dibawah ini.



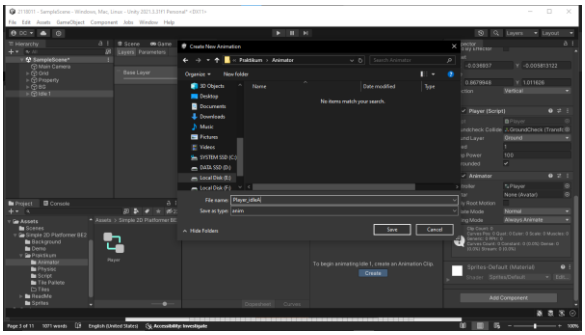
Gambar 1.5 Membuat Membuka Panel Animator

6. Setelah itu tambahkan *menu panel animator*. Klik *window*, lalu pilih *animation*, setelah itu pilih *animator*.



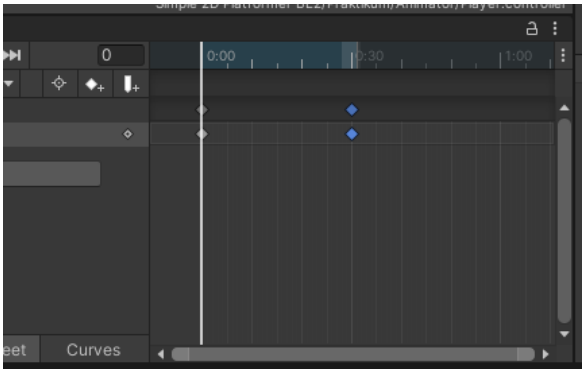
Gambar 1.6 Menambahkan Menu Panel Animator

7. Lalu buat animasi pada *player*. Klik *player* pada *hierarchy*. Lalu ke menu panel *animation* dan pilih *create*. Simpan pada *folder* Animator yang telah dibuat dan beri nama "Player_idle"



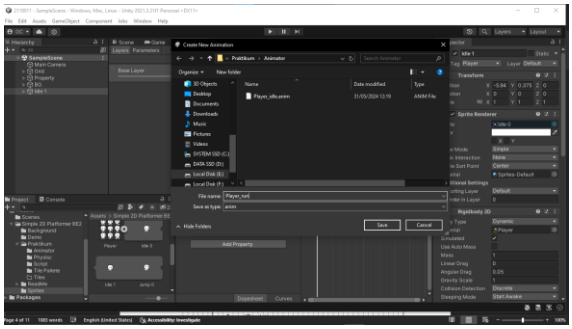
Gambar 1.7 Mulai Mmebuat Animasi

8. Berikutnya, cari file idle 0. Jika sudah menemukan klik pada idle 0. Lalu *drag and drop* pada animation. Jika sudah maka geser kotak kecil *timeline* sampai *frame* 0:30, agar animasi tidak terlalu cepat.



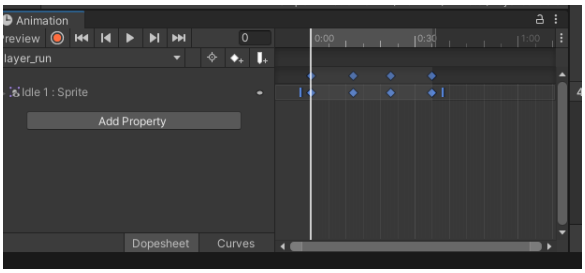
Gambar 1.8 Mengatur Timeline Idle

9. Setelah itu buat animasi baru lagi untuk bagian lari. Klik pada bagian player_idle. Lalu pilih *create new clip*. Setelah itu beri nama "*Player_run*". Simpan *file* pada *folder* animator.



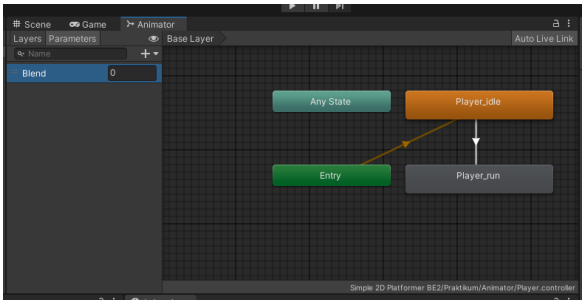
Gambar 1.9 Membuat Animasi Lari pada Karakter

10. Lalu cari *file* tempat gambar karakter berlari. Jika sudah menemukan lakukan *drag and drop* pada *panel* animator. Setelah itu Tarik *timeline*nya agar ke waktu 0:35.



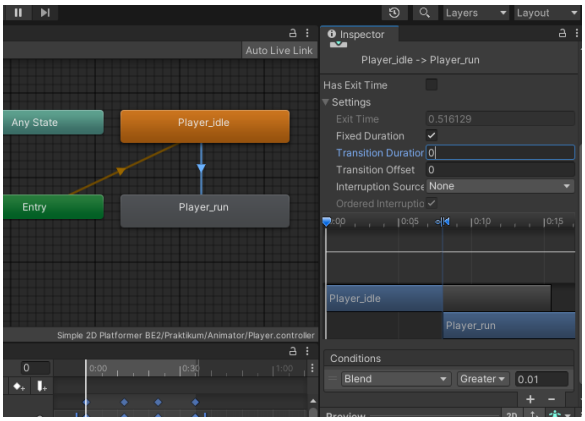
Gambar 1.10 Mengantur Timeline Lari

11. Lalu pergi ke *menu* animator yang telah dibuka. Setelah itu tambahkan sebuah transisi dari *player idle* ke *player run*. Jika sudah masuk ke tab parameter dan tambahkan tipe data *float* dan ubah namanya menjadi "Blend".



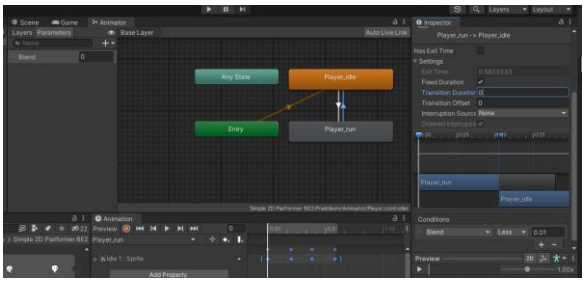
Gambar 1.11 Menambahkan Transisi dan Tipe Data

12. Selanjutnya klik panah putih (transisi yang telah dibuat). Lalu pergi ke *inspector*. Tambahkan kondisi *blend* dan beri *value*nya menjadi 0.01. Jika sudah hilangkan centang pada *has exit time* dan beri *value* pada *transistion duration* menjadi 0.



Gambar 1.12 Mengatur Transisi Idle ke Run

13. Setelah itu buat transisi dari *Run* ke *Idle*. Jika sudah atur serupa dengan seperti Langkah sebelumnya. Yang memedakan adalah dibagian kondisi, *blend* berstatus *less*.



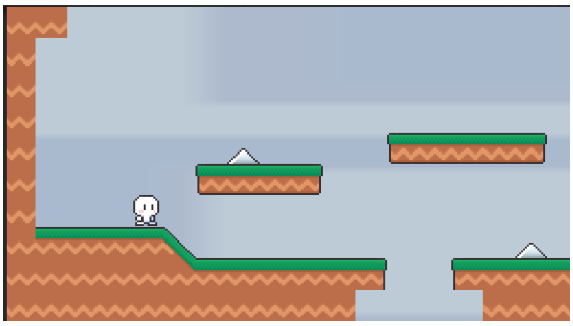
Gambar 1.13 Mengatur Transisi Run ke Idle

14. Selanjutnya klik *player phyisics*, lihat bagian *inspector*. Jika sudah ubah menjadi 0 pada bagian *friction* dan *bounces*nya.

```
public Animator animator;

  private void Awake()
  {
    rb = GetComponent<Rigidbody2D>();
    animator = GetComponent<Animator>();
  }

  void FixedUpdate()
  {
    GroundCheck();
    Move(horizontalValue, jump);

    animator.SetFloat("Blend",
Mathf.Abs(rb.velocity.x));
```

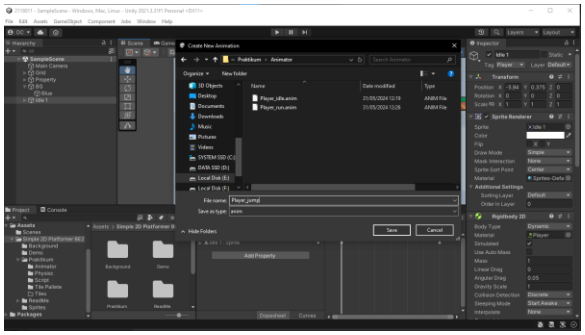
```
    }
```

Hasil Tampilan:



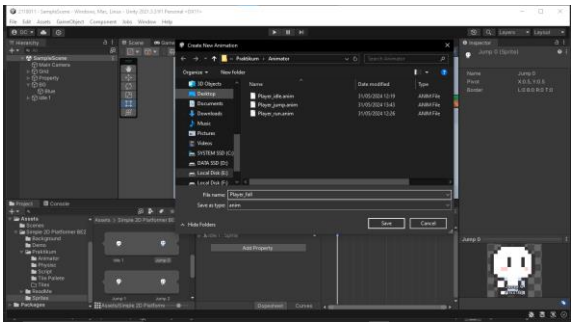
Gambar 1.14 Animasi Berjalan dan Berhenti

15. Selanjutnya buat untuk bagian animasi melompat. Pilih *create new clip*. Setelah itu beri nama “Palyer_Jump”. Letakkan di bagian *folder* praktikum. Dan *drag and drop* animasinya.



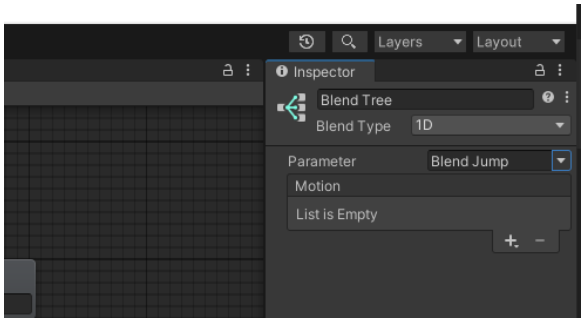
Gambar 1.15 Menambahkan File Player Jump

16. Setelah itu lakukan hal yang sama. Buat file baru untuk karakter ketika mendarat dari melompat.



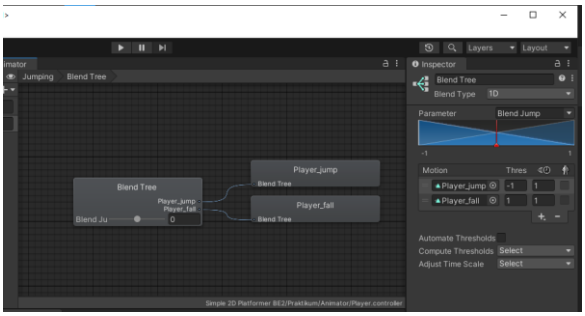
Gambar 1.16 Menambahkan File Player Fall

17. Lalu *state* baru. Pilih *create state* dan pilih *from new blend tree*. Lalu pada animator klik *blend tree* dan ubah Namanya menjadi “Jumping” . Jika sudah pada parameter buat tipe data *float* baru. Beri nama “Blend Jump”. Setelah itu tekan *blend tree*, dan ubah parameternya menjadi *blend jump*.



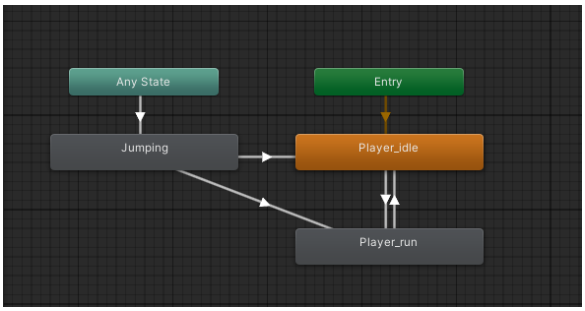
Gambar 1.17 Menambahkan Animasi Melompat

18. Lalu pada menu *inspector*, pilih *add motion field*. Buat dua *motion field*. Jika sudah, isi motion dengan *player fall* dan *player jump*. Jika sudah hilangkan centang pada *automate tresholds* dan berikan nvalue -1 untuk bagian *player fall*.



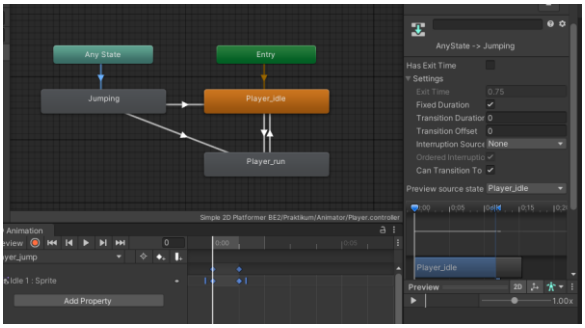
Gambar 1.18 Mengatur Treshold

19. Setelah itu kemabli ke *menu base layer*. Jika sudah maka bentuk transasi yang baru. Bentuk seperti gambar dibawah ini.



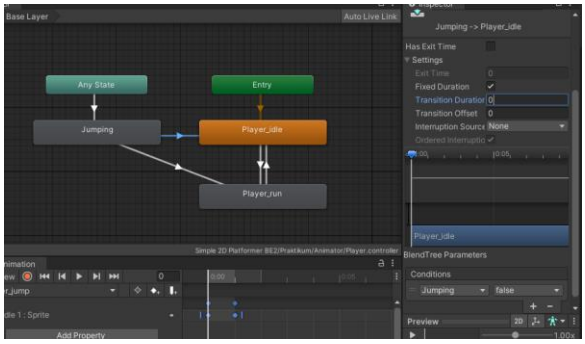
Gambar 1.19 Membentuk Transisi

20. Selanjutnya klik panah putih pada bagian yang ke jumping. Tambahkan kondisi Jumping, dan ubah *value*nya menjadi *true*. Lalu klik *settings* dan ubah nilai *transistion*nya menjadi 0 dan hilangkan centang *has exit time*.



Gambar 1.20 Megatur Trasnsisi Jumping

21. Selanjutnya lakukan hal yang sama pada tanda panah *jumping* ke *idle* dan *run*. Pada tanda panah *idle* kondisinya diberi nilai *false*. Sedangkan pada *run* diberi nilai *true*.



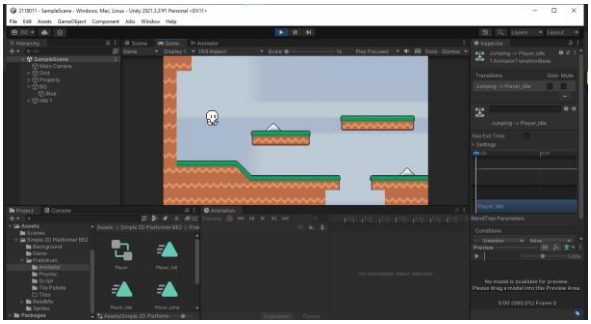
Gambar 1.21 Mengatur Transisi Run dan Idle

22. Selanjutnya tambahkan sedikit source code dibawah ini. Lalu coba terlebih dahulu untuk melihat hasilnya. Jika sudah dicek dan aman, maka bisa dirender.

Source code

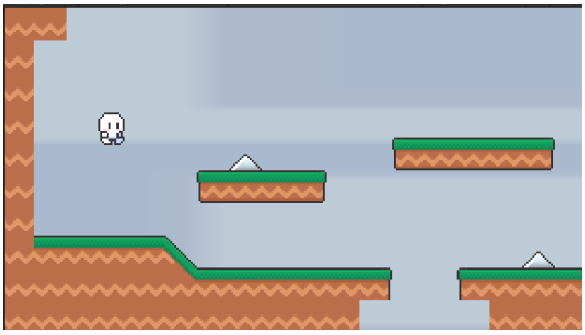
```
  void Update ()
  {
    horizontalValue = Input.GetAxisRaw("Horizontal");
    if (Input.GetButtonDown("Jump"))
    {
      animator.SetBool("Jumping", true);
      jump = true;
    }
    else if (Input.GetButtonUp("Jump"))
    jump = false;
  }

  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);
  }
```

Hasil Tampilan



Gambar 1.22 Mengatur Transisi Run dan Idle

23. Maka Setelah di render hasilnya akan seperti gambar dibawah ini.



Gambar 1.23 Hasil Render

B. Kuis Pertemuan 9

```
    void HandleJumpInput()
    {
        if (Input.GetKeyDown(KeyCode.Space))
        {
            animator.SetBool("isJumping", );
            rb.AddForce(Vector2.up      *      jumpForce,
ForceMode2D.Impulse);
        }
        else if (Input.GetKey(KeyCode.Space))
        {
            animator.SetBool("isJumping",);
        }
    }

    void HandleMovementInput()
    {
        float move = Input.GetAxis("Horizontal");

        if (move != 1)
        {
            animator.SetBool("isIdle", true);
            transform.Translate(Vector3.left   *   move   *
Time.deltaTime);
        }
        else
        {
            animator.SetBool("isWalking", false);
        }

        if (move != 0)
        {
            transform.localScale = new Vector3(-4, 1, 1);
        }
        else if (move > 0)
        {
            transform.localScale = new Vector3(1, 2, 1);
        }
    }
```

Penjelasan Source code

Pada source code terdapat kesalahan ketika menerima nilai bool. Seharusnya dalam tipe data bool terdapat 2 parameter, dimana parameter kedua ini pakah bernilai false atau true.