TUGAS PERTEMUAN: 9

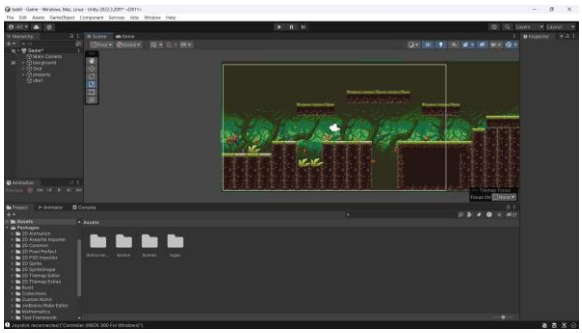
GAME ANIMATION

NIM	:	2118121
Nama	:	Acmad saiful udin
Kelas	:	D
Asisten Lab	:	Berchmans Bayu Bin Jaya (2218034)

9.1 Tugas 9 :Mengimplementasikan Game Animation

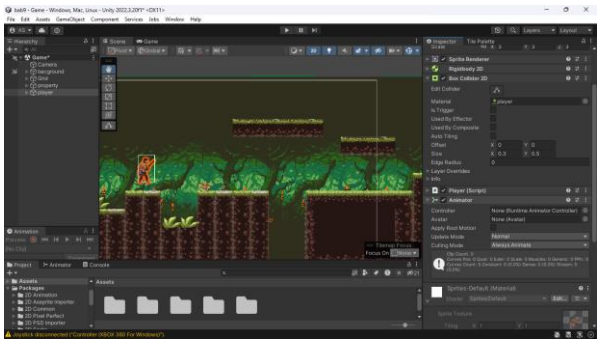
A. Mengimplementasikan Game Animation

1. Buka project Unity yang telah di buat sebelumnya.



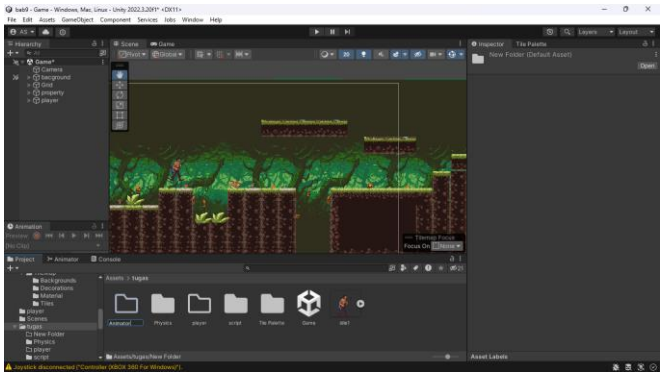
Gambar 9.1 Tampilan *Project Unity*

2. Pada karakter di Inspector, klik `Add Component` dan tambahkan `Animator.



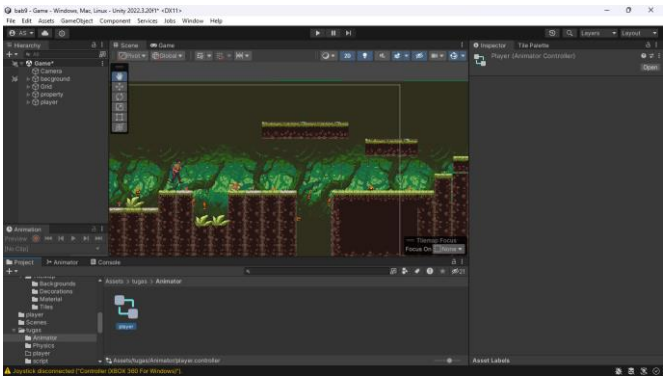
Gambar 9.2 Add Component Animator

3. Buat folder baru di dalam `Tugas` dan beri nama `Animator`.



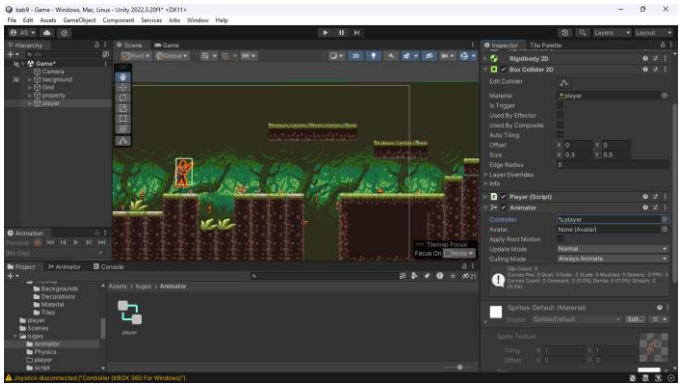
Gambar 9.3 New folder

4. Di dalam folder `Animator`, buat file `Animator Controller` dengan cara klik kanan kemudian `Create` pilih `Animator Controller` dan beri nama `Player`.



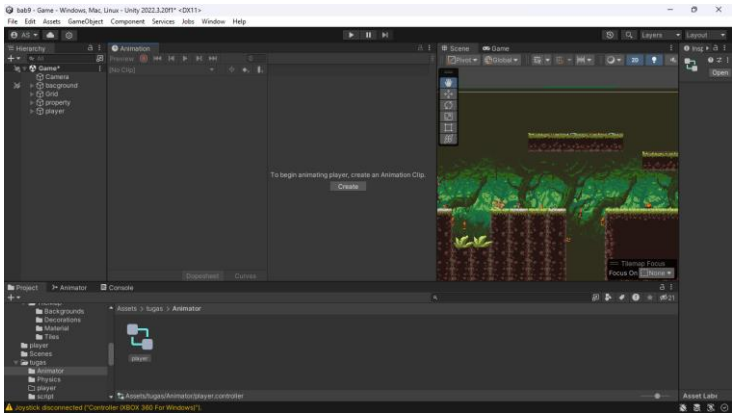
Gambar 9.4 *Animator Controller*

5. Pada komponen `Animator` di Inspector player, ubah `Controller` menjadi `Player`.



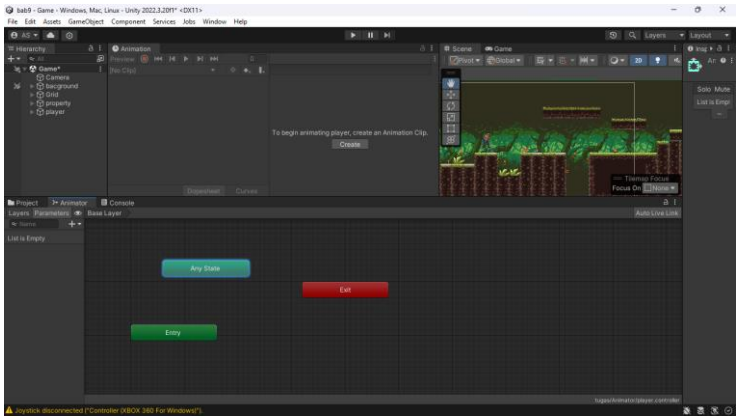
Gambar 9.5 *Component Animator*

6. Tambahkan menu panel `Animation` dengan menekan `Ctrl + 6`.



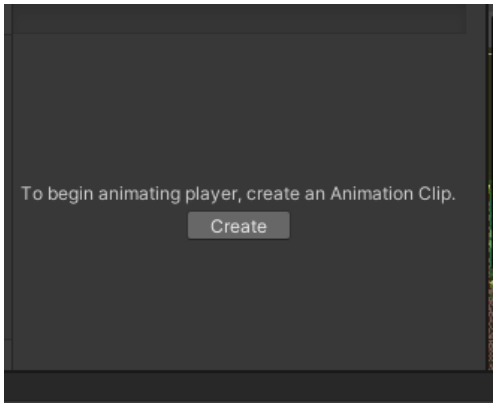
Gambar 9.6 Menu Panel Animation

7. Tambahkan juga menu panel Animator.



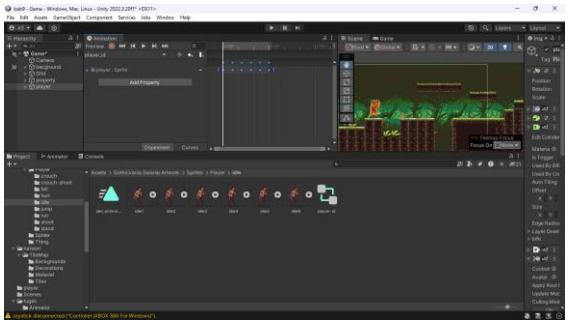
Gambar 9.7 Panel Animator

8. Pada menu panel `Animation`, klik `Create`.



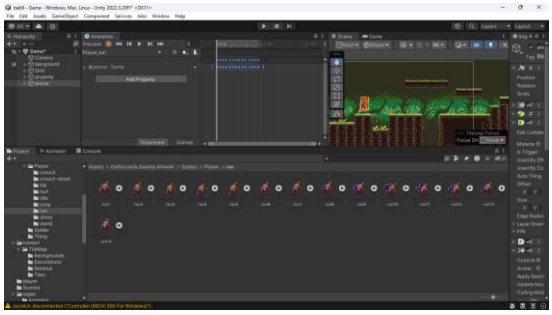
Gambar 9.8 Create Animation Clip

9. Simpan folder `Animation` dan beri nama `Player_idle`, kemudian masukkan asset `character_idle` dengan cara drag and drop ke tab `animation`.



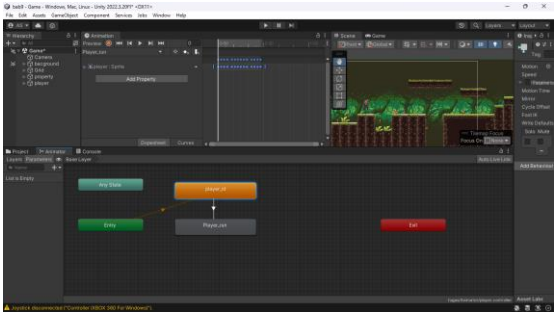
Gambar 9.9 Player_idle Animation

10. Tambahkan animasi baru dan beri nama `Player_run`, lakukan hal yang sama seperti langkah sebelumnya.



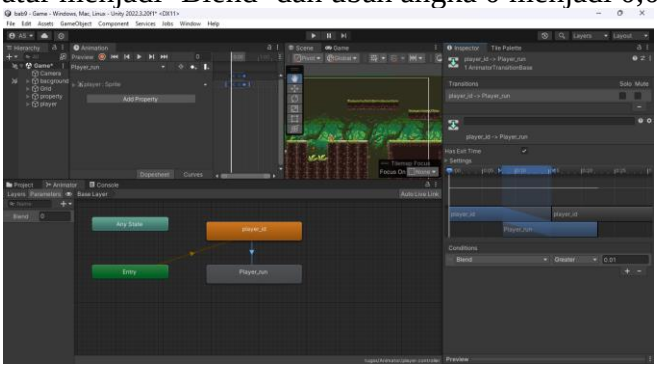
Gambar 9.10 Player_run Animation

11. Buka panel `Animator`, buat transisi antara `Player_idle` dan `Player_run`. Klik kanan pada `Player_idle` pilih `Make Transition` dan tarik ke `Player_run`. Masuk ke tab `Parameter` buat tipe data float dan beri nama `Blend`.



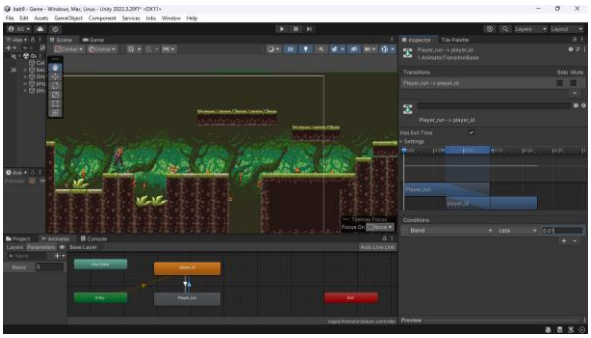
Gambar 9.11 *Transition Player_idle* dengan *Player_run*

12. Klik tanda panah putih, pada bagian `Conditions` klik icon `+` kemudian atur menjadi `Blend` dan ubah angka 0 menjadi 0,01.



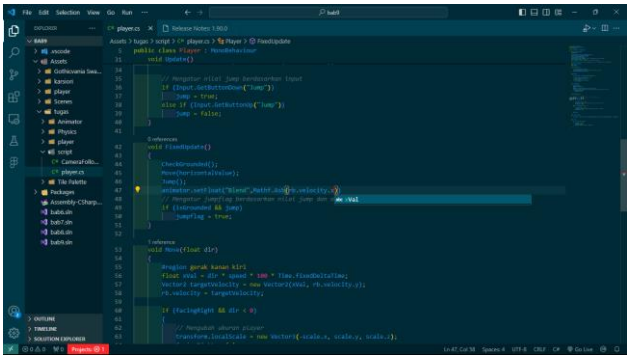
Gambar 9.12 Conditions Blend

13. Lakukan hal yang sama untuk transisi dari `Player_run` ke `Player_idle`. Tambahkan `Blend` dan setting ke `Less` ubah ukuran menjadi 0,01.



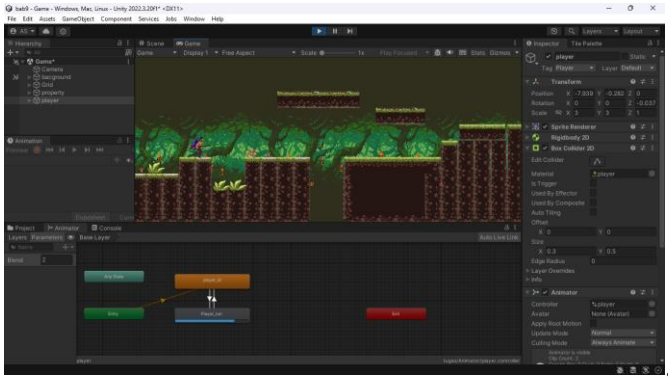
Gambar 9.13 Player_run transition to Player_idle

14. Pada script player tambahkan source code.



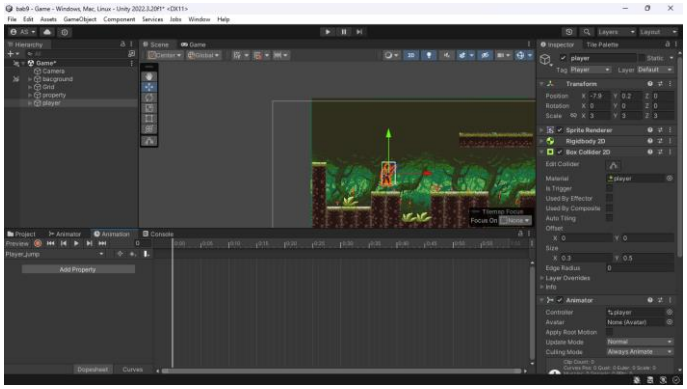
Gambar 9.14 perbarui source code

15. Jika dijalankan, player akan memiliki animasi ketika berhenti maupun berlari.



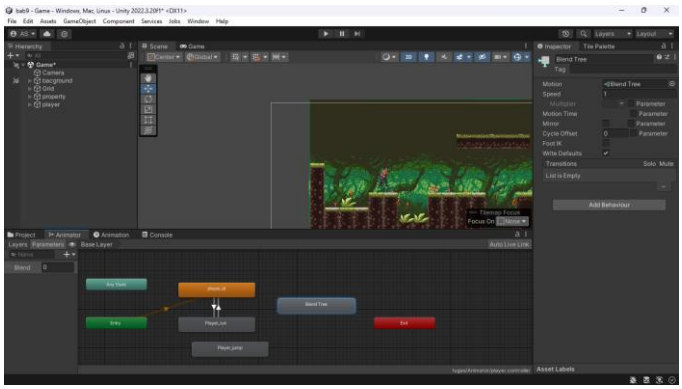
Gambar 9.15 *Animation run*

16. Pada panel animasi, buat clip baru dan beri nama `Player_jump`.



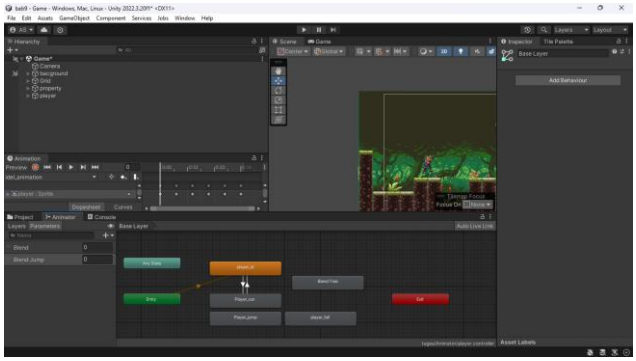
Gambar 9.16 *Player_jump*

17. Pada window `Animator`, buat `From New Blend Tree`.



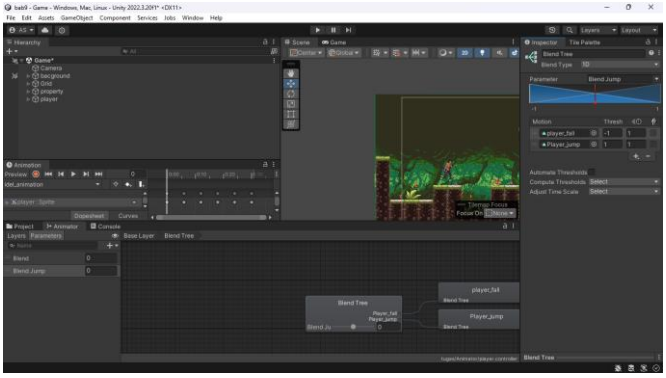
Gambar 9.17 *New Blend Tree*

18. Pada menu `Parameters`, tambahkan parameter tipe float dengan nama `Blend Jump`.



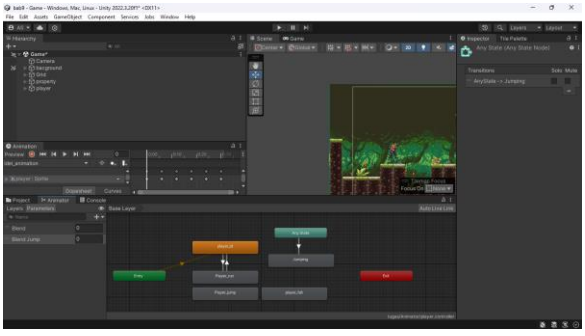
Gambar 9.18 *Blend Jump*

19. Klik dua kali pada `Blend Tree Jumping`, tekan pada `Blend Tree`. Di Inspector, ubah `Parameters` menjadi `Blend Jump`, ubah juga motionnya dengan `Player Fall` dan `Run`.



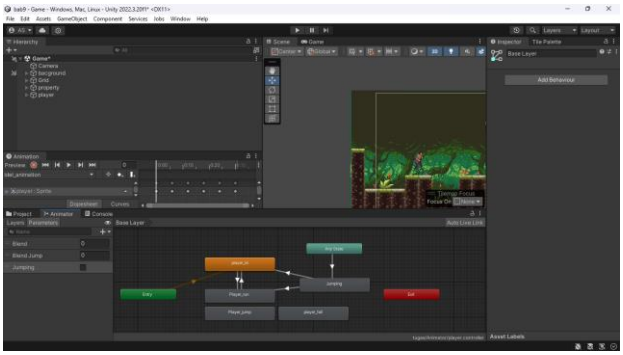
Gambar 9.19 *Inspector Blend Tree*

20. Di `Base Layer`, klik kanan `Any State`, buat transisi ke `Jumping`.



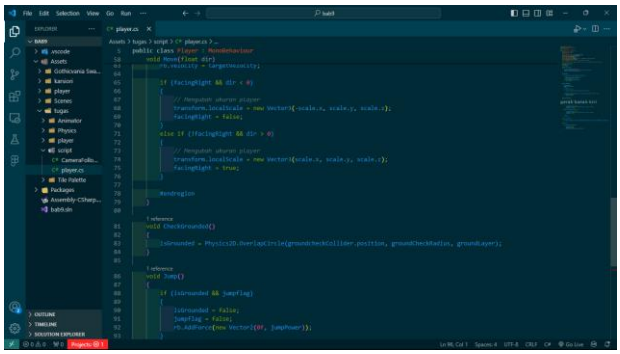
Gambar 9.20 *base layer*

21. Buat transisi dari jumping ke player_run dan player_id.



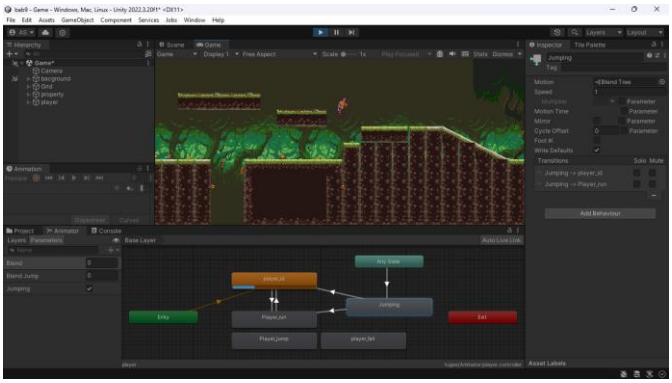
Gambar 9.21 transisi

22. Pada script player perbarui source code.



Gambar 9.22 source code

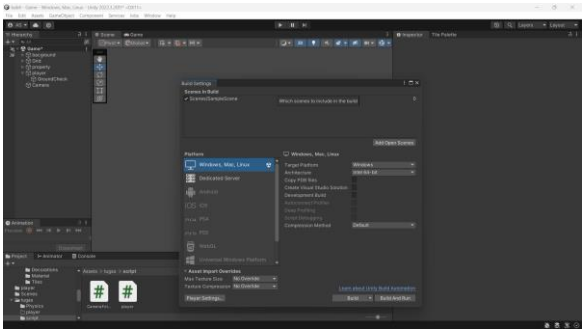
23. Play project game dan cek animasi jumping, run, dan fall.



Gambar 9.23 Play project

B. Render

1. Untuk proses merender pada file kemudian pilih Build Settings atau menekan Ctrl + Shift + B, setelah masuk Setting Build pilih PC, Mac dan Linux. Jangan lupa pada scenes in build mencentang project kita, kalo belum ada tekan Add Open Scenes.



Gambar 9.24 Render Settings

2. Setelah itu kita klik Build dan kita pilih project jadinya akan disimpan dimana. Dan tunggu hasilnya.

Gambar 9.25 Render Settings

C. Link Github Pengumpulan

https://github.com/acmadsaiful090/2118121_PRAK_ANIGAME/tree/main/BAB%209

KUIS

Kuis bab 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);
        }
    }
```

Analisa :

Untuk penjelasan kode diatas yaitu pada baris animator.SetBool ("isJumping",) ; dan animator.SetBool("isJumping",);, nilai boolean tidak diberikan. Fungsi SetBool membutuhkan dua argumen: nama parameter dan nilai boolean, sehingga kita harus memberikan nilai true atau false. Selain itu, penggunaan rb.AddForce(Vector2.up * jumpForce, ForceMode2D.Impulse); salah karena variabel jumpForce tidak didefinisikan dalam. Seharusnya menggunakan jumpPower, sesuai dengan deklarasi variabel sebelumnya. Terakhir, terdapat kesalahan logika di HandleMovementInput, di mana kondisi else if (move > 0) tidak akan pernah tercapai karena kondisi if (move != 0) sudah menangani semua kasus di mana move tidak sama dengan 0. Seharusnya, kita menggunakan if terpisah untuk menangani move > 0 dan move < 0.