MODUL 14

CONTROLLING 3D ANIMATIONS

A. TUJUAN

- Mahasiswa dapat membuat Configuring a character's Avatar and Idle Animation.
- Mahasiswa dapat membuat Moving character with root motion and Blend Trees.
- Mahasiswa dapat membuat Mixing animations with Layers and Masks.
- Mahasiswa dapat membuat Organizing States Into Sub State Machines.
- Mahasiswa dapat membuat Transfoming the Character Controller via Script.

B. PETUNJUK

1. Awali setiap kegiatan praktikum dengan berdoa
2. Baca dan pahami tujuan, dasar teori, dan latihan-latihan praktikum dengan baik
3. Kerjakan tugas-tugas praktikum dengan baik, sabar dan jujur
4. Tanyakan kepada dosen apabila ada hal-hal yang kurang jelas

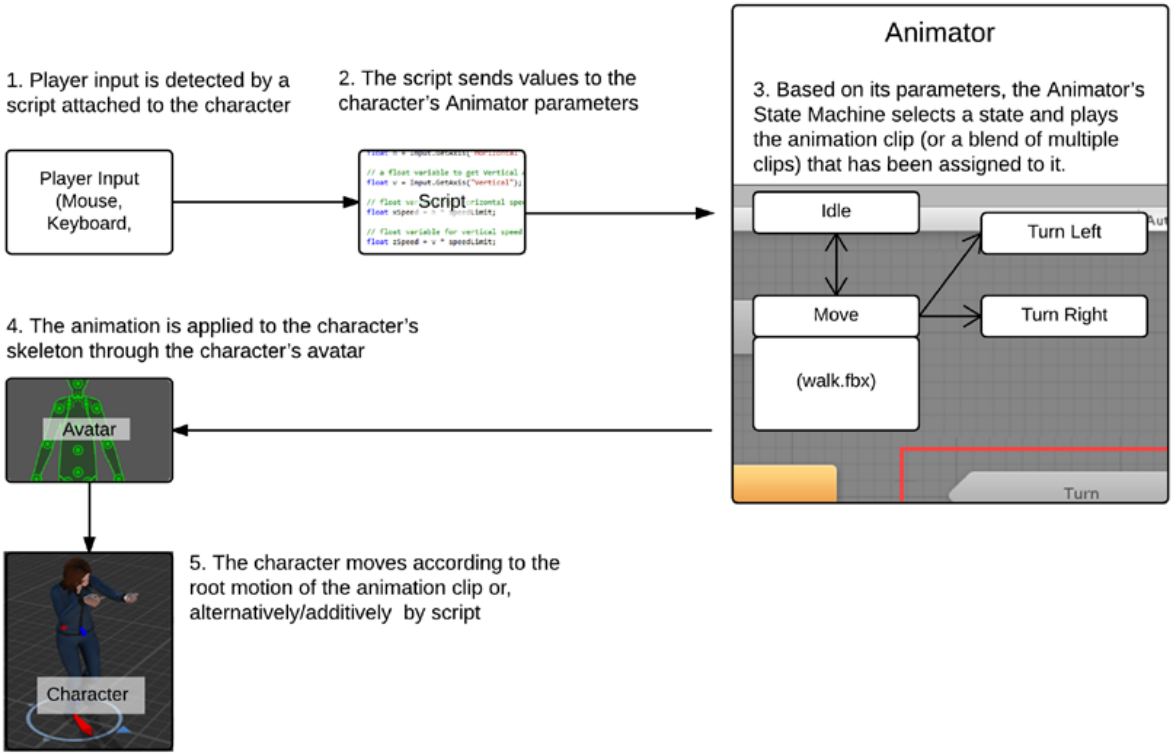
C. ALOKASI WAKTU : 3 jam pelajaran

D. DASAR TEORI

The Mecanim animasi dapat berevolusi bagaimana membuat karakter animasi dan dapat dikendalikan dalam *Unity*. Dalam modul ini, dapat belajar dengan memanfaatkan *flexibility, power, friendly, highly visual interface*.

Gambaran besarnya

Mengontrol karakter yang dapat dimainkan dengan sistem Mecanim mungkin terlihat seperti *complex task*, tapi sebenarnya sangat mudah. Untuk lebih jelasnya dapat dilihat pada gambar dibawah:



pada bab ini, mendapatkan setidaknya pemahaman dasar system *Mecanim*. Untuk lebih lengkap tentang subjek ini, lihatlah Animasi Karakteritas Jamie Dean dengan *Mecanim*, juga diterbitkan oleh *Packt Publishing*. Semua project akan menggunakan paket gerak *Mixamo*. *Mixamo* adalah solusi lengkap untuk *character production, rigging, dan animation*. Padahal karakter di Penggunaannya dirancang dengan perangkat lunak pembuatan karakter Mixamo yang disebut *Fuse* dan dicocokkan dengan *Mixamo Auto-rigger*. Anda dapat mengetahui lebih banyak tentang *Mixamo* dan produk mereka di *Unity's Asset Store.*

(https://www.assetstore.unity3d.com/en/#!/publisher/150) atau akun mereka Situs web di https://www.mixamo.com/.

Perlu diketahui bahwa meskipun Mixamo menawarkan *character dan animation clips* akan menggunakan *unprepared animation clips*. Alasannya adalah untuk membuat lebih percaya diri saat berhadapan dengan aset yang diperoleh dengan metode dan sumber lain.

E. LATIHAN PRAKTIKUM

Configuring a Character's Avatar and Idle Animation

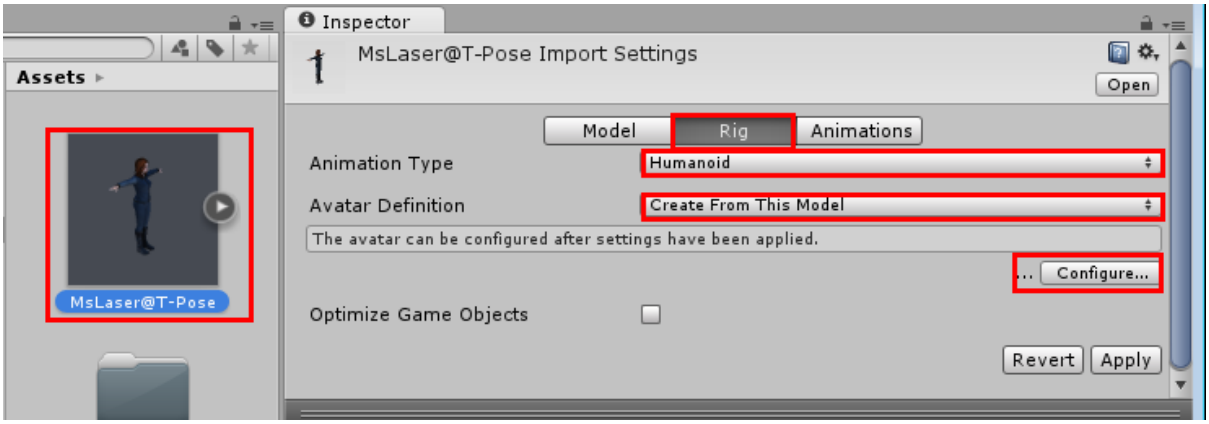
Sebuah fitur yang dapat membuat *Mecanim* begitu *flexibel* dan *power* adalah mempunyai kemampuan dengan cepat untuk dapat kembali ke *animation clips* dari satu karakter ke karakter lainnya. Hal ini dimungkinkan melalui

penggunaan *Avatars*, yang pada dasarnya adalah lapisan antara *rig* asli karakter dan the Unity's Animator System.

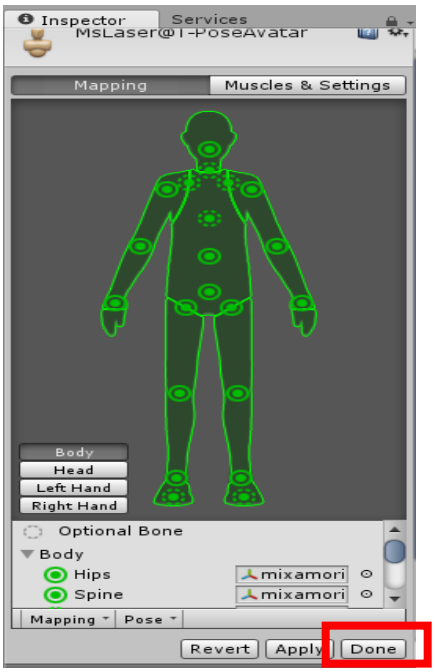
Dalam project ini, kita akan belajar cara mengkonfigurasi kerangka *Avatar* dengan karakter *rigged*:

1. Untuk membuat project ini, memerlukan file **MsLaser@T-Pose.fbx** dan **Swat@rifle_aiming_idle.fbx**, yang terdapat di dalam folder 1362_07_code
2. Impor file **MsLaser@T-Pose.fbx** dan **Swat@rifle_aiming_idle.fbx** ke project anda.

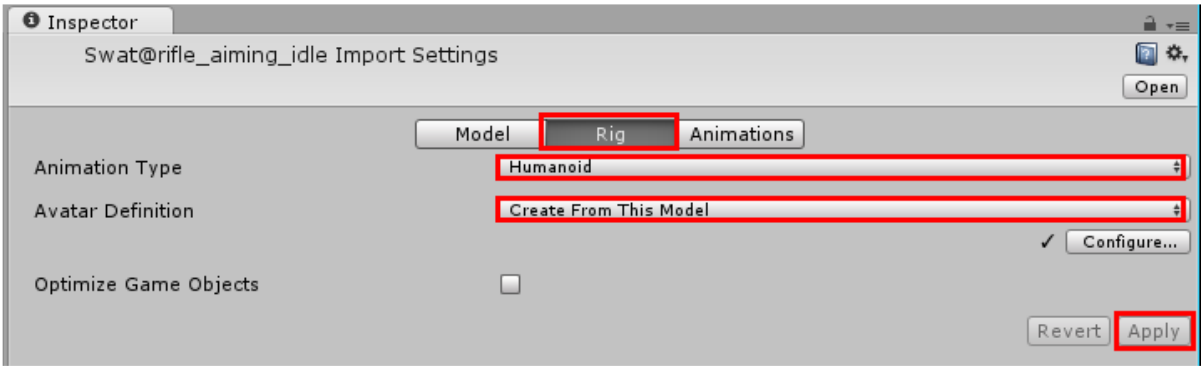
3. Pilih dari tampilan Project, model **MsLaser @ T-Pose**.
4. Dalam tampilan **Inspector**, Import **MsLaser @ T-Pose** dan setting **Inspector** pada bagin **Rig**. Ubah **Animation Type** menjadi **Humanoid**. Setelah itu pada **Avatar Definition** rubah ke **Create From This Model**. Akhirnya, klik tombol **Configure**....



5. Pada tampilan **Inspector** akan menampilkan **Avatar** yang baru saja dibuat. Amatilah bagaimana Unity memetakan tulang karakter dengan benar ke dalam strukturnya dan menetapkan. misalnya, tulang **mixamoRig: LeftForeArm** sebagai **Lengan Bawah Avatar**. Tentu saja, bisa menugaskan kembali tulang jika diperlukan. Untuk saat ini, cukup klik tombol Done untuk menutup tampilan.

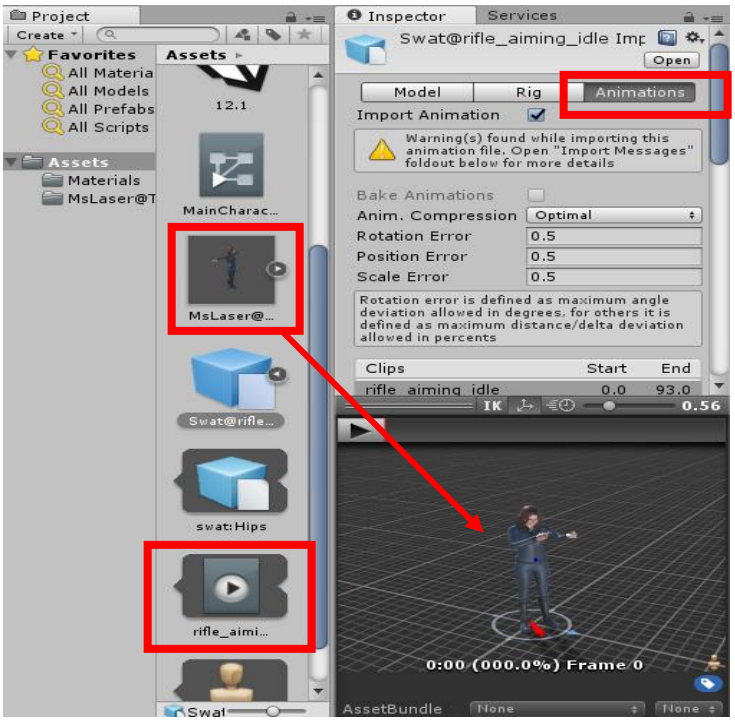


6. Sekarang Avatar sudah siap, selanjutnya konfigurasikan animasi dengan **Idle State**. Dengan cara di tampilan Project → Asset, pilih file **Swat@rifle_aiming_idle**. Pilih bagian **Rig**, ubahlah **Animation Type** menjadi **Humanoid** dan **Avatar Definition** menjadi **Create From This Model**. Setelah itu pilih button **Apply**.

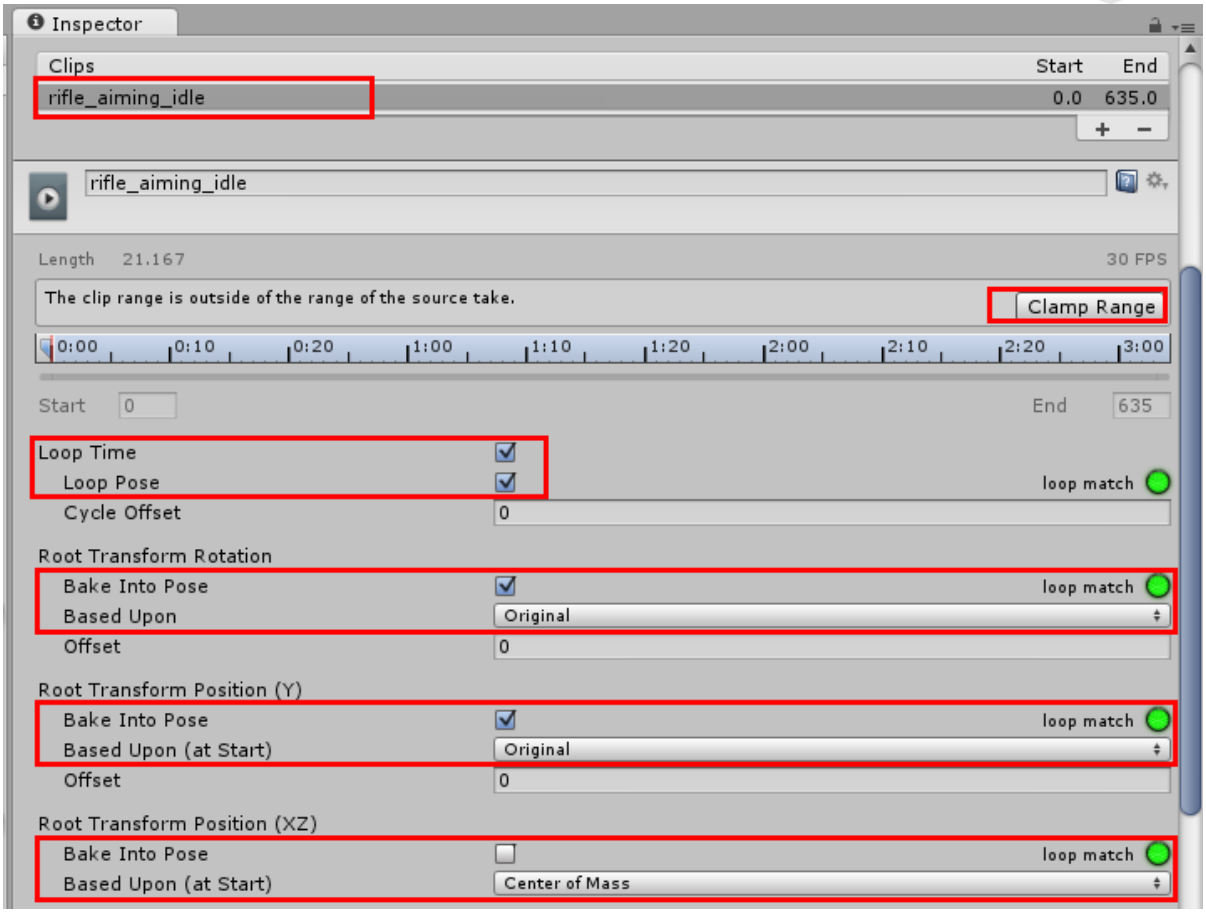


7. Selanjutnya pilih bagian **Animation** (di sebelah kanan Rig) pada **Inspector**. Pilih **Swat@rifle_aiming_idle → rifle_aiming_idle**. Didalam area **Preview** (di bagian bawah Inspector) akan menampilkan pesan sebagai **Model Tidak Tersedia (No Model)** untuk

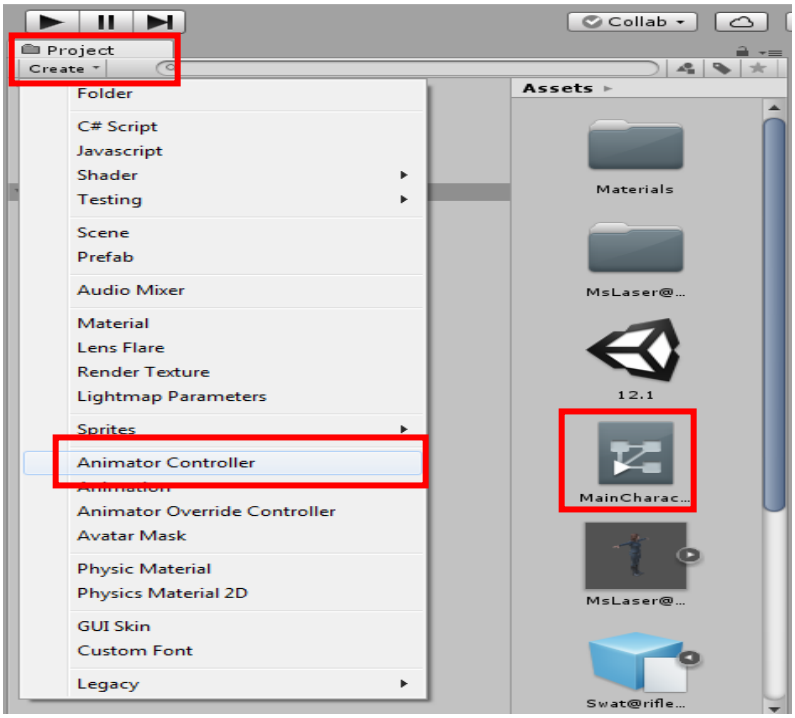
preview. maka **drag MsLaser@T-Pose** ke dalam **Area Preview** untuk memperbaiki project.



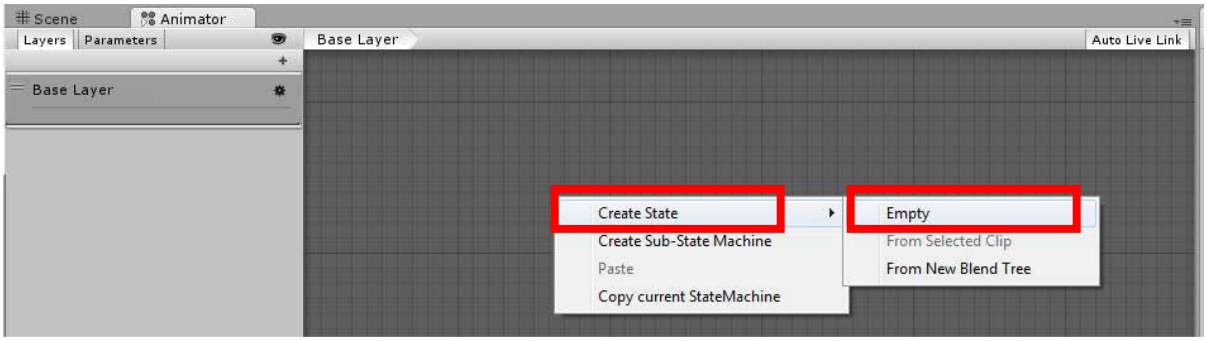
8. Dengan **rifle_aiming_idle** pilih dari daftar **Clips**, **centanglah** pilihan **Loop Time and Loop Pose**. Juga, klik pada tombol **Clamp Range** untuk mengatur garis waktu ke waktu sebenarnya dari klip animasi, Kemudian di bawah **Root Transform Rotation**, centanglah **Bake Into Pose**, dan pilih **Based Upon → Original**. Di bawah posisi **Root Transform Position(Y)** centanglah **Bake Into Pose**, dan pilih **Baked Upon (at Start) → Original**. Di bawah **Root Transform Position (XZ)** tidak dicentang pada **Bake Into Pose** dan pilih **Based Upon (at Start) → Center Of Mass**. Selanjutnya klik **Apply** untuk mengkonfirmasi perubahannya.



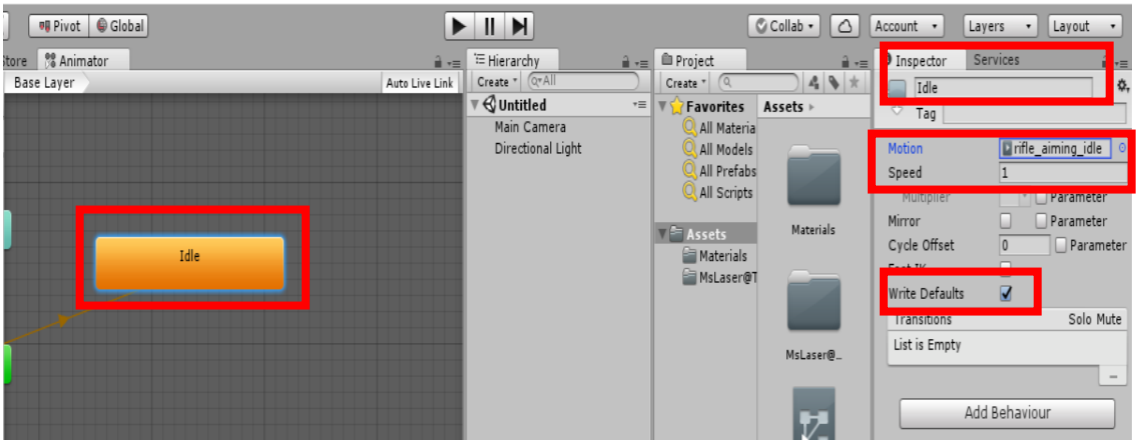
9. Untuk mengakses *animation clips* dan memainkannya diperlukan membuat kontroler. Maka lakukanlah dengan mengklik tombol **Create** dari tampilan **Project** dan kemudian memilih pilihan **Animator Controller** rubahlah dengan nama sebagai **MainCharacter**.



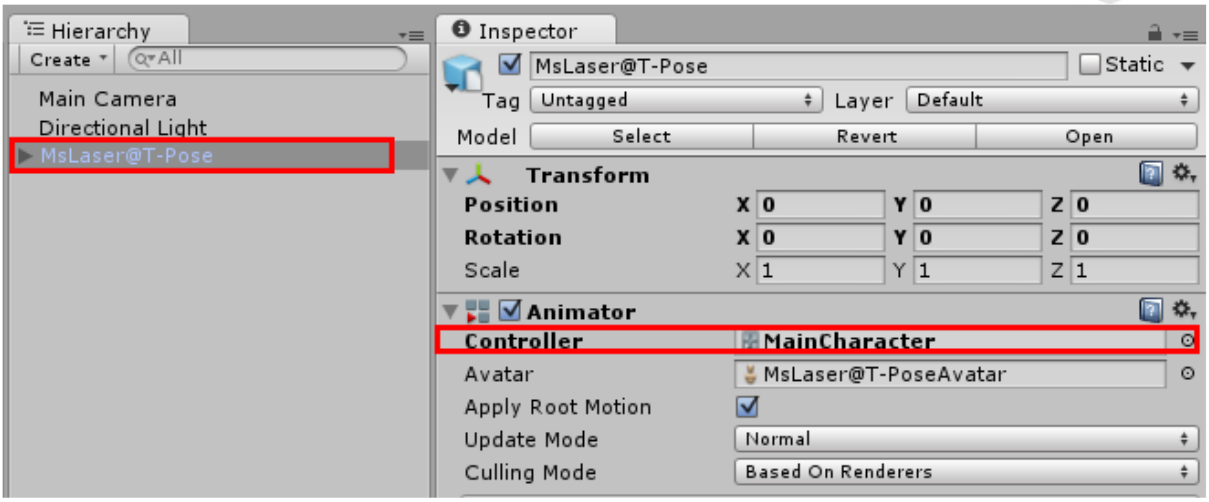
10. Klik dua kali pada **Animator Controller** untuk membuka tampilan **Animator**.
11. Dari tampilan **Animator**, **klik kanan** pada **grid** untuk membuka menu konteks. Lalu, pilih **Create State** → **Empty** dan sebuah kotak baru bernama **New State** akan muncul. Kotak **New State** tersebut berwarna orange, menunjukkan bahwa itu adalah keadaan **default**.



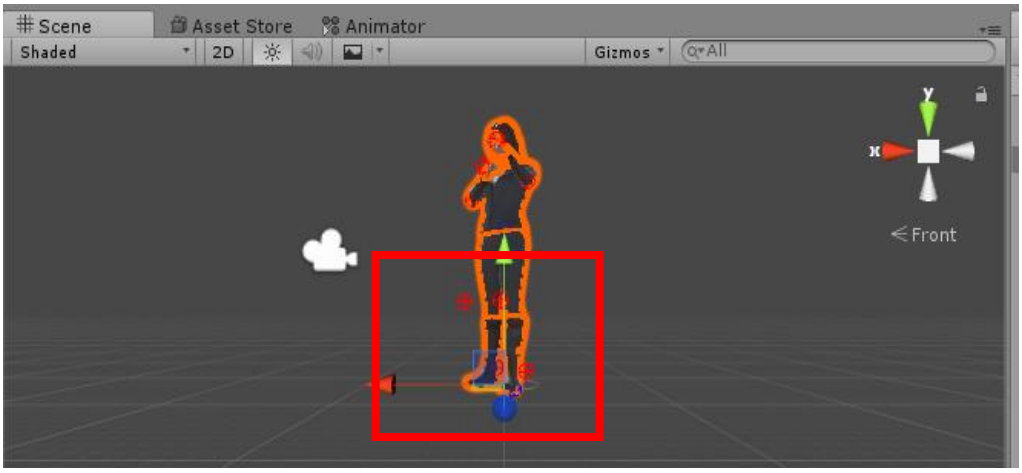
12. Pilih **New State** dalam tampilan **Inspector** ganti namanya menjadi **Idle**. Juga, di **Motion** pilih **rifle_aiming_idle** dengan memilihnya dari daftar.



13. Drag model **MsLaser@T-Pose** dari tampilan **Project** ke tampilan **Hierarchy** dan Letakkan di tempat scene.
14. Pilih **MsLaser@T-Pose** dari tampilan **Hierarchy** dan amati **Animator-nya** komponen dalam tampilan **Inspector**. Kemudian, tetapkan **MainCharacter** yang baru dibuat **Controller** ke bidang **Controller**-nya.



15. Mainkan adegan Anda untuk melihat karakter animasi yang benar.



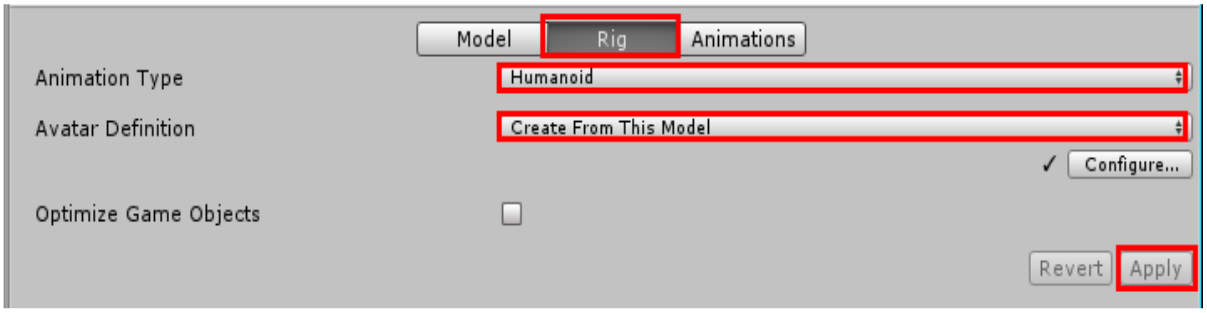
Moving your character with root motion and Blend Trees

The Mecanim Animation mampu menerapkan Root Motion pada karakter. Selain itu, karakter benar-benar bergerak sesuai dengan klip animasi, sebagai menerjemahkan model karakter sambil bermain di tempat *animation cycle*. Project ini dapat membuat sebagian besar klip animasi *Mixamo* yang sempurna untuk digunakan dengan pada *The Mecanim*. Fitur lain dari sistem animasi adalah *Blend Trees*, yang bisa memadukan klip animasi *smoothly dan easily*. Dalam project ini, memanfaatkan fitur untuk membuat produk dengan karakter berjalan / lari maju dan mundur, dan juga kanan dan kiri pada kecepatan yang berbeda. kita akan belajar cara Memindahkan karakter dengan *Root Motion dan Blend Trees*:

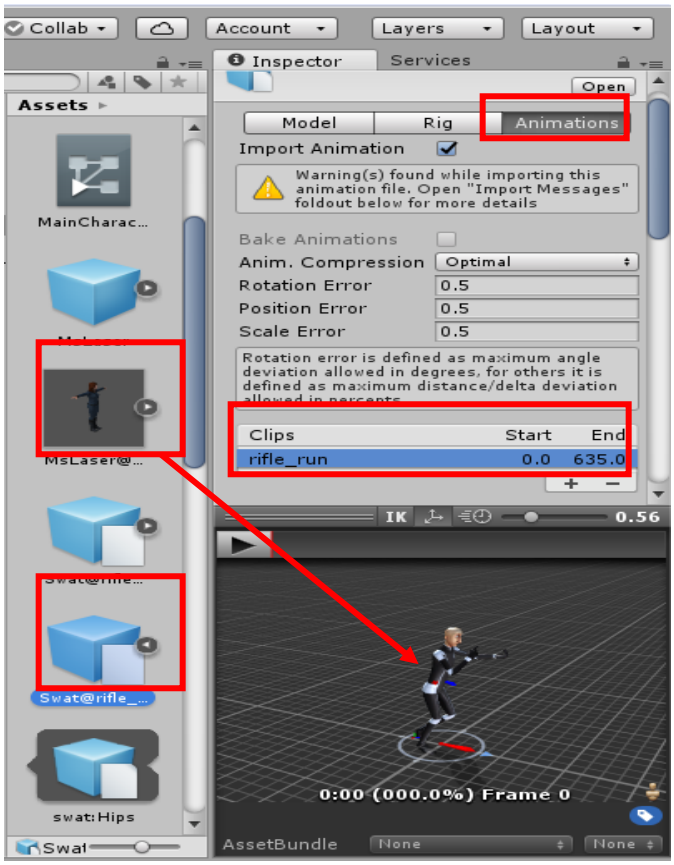
1. Untuk project ini dibutuhkan **Import Character_02.unityPackage** ke dalam sebuah project baru. Juga, **impor Swat@rifle_run, Swat@run_backwards, Swat@strafe,**

Swat@strafe_2, Swat@strafe_left, Swat@strafe_right, Swat@walking, dan Swat@Berjalan_backwards .fbx file.

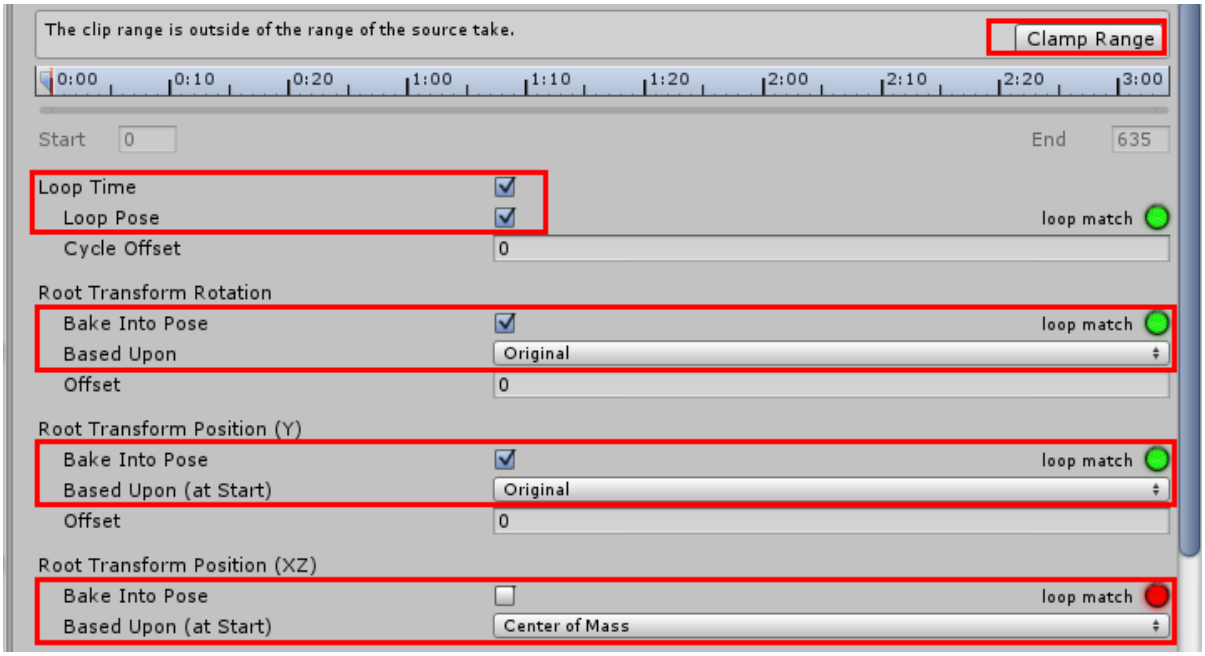
2. Kita perlu mengkonfigurasi **animation clips** maka dari tampilan project pilih **Swat@rifle_run**
3. Pilih bagian **Rig** didalam **inspector**. Ubah **Animation Type** menjadi **Humanoid** dan **Avatar Definition** menjadi **Creat From This Model**. Setelah itu pilih dengan button klik **Apply**.



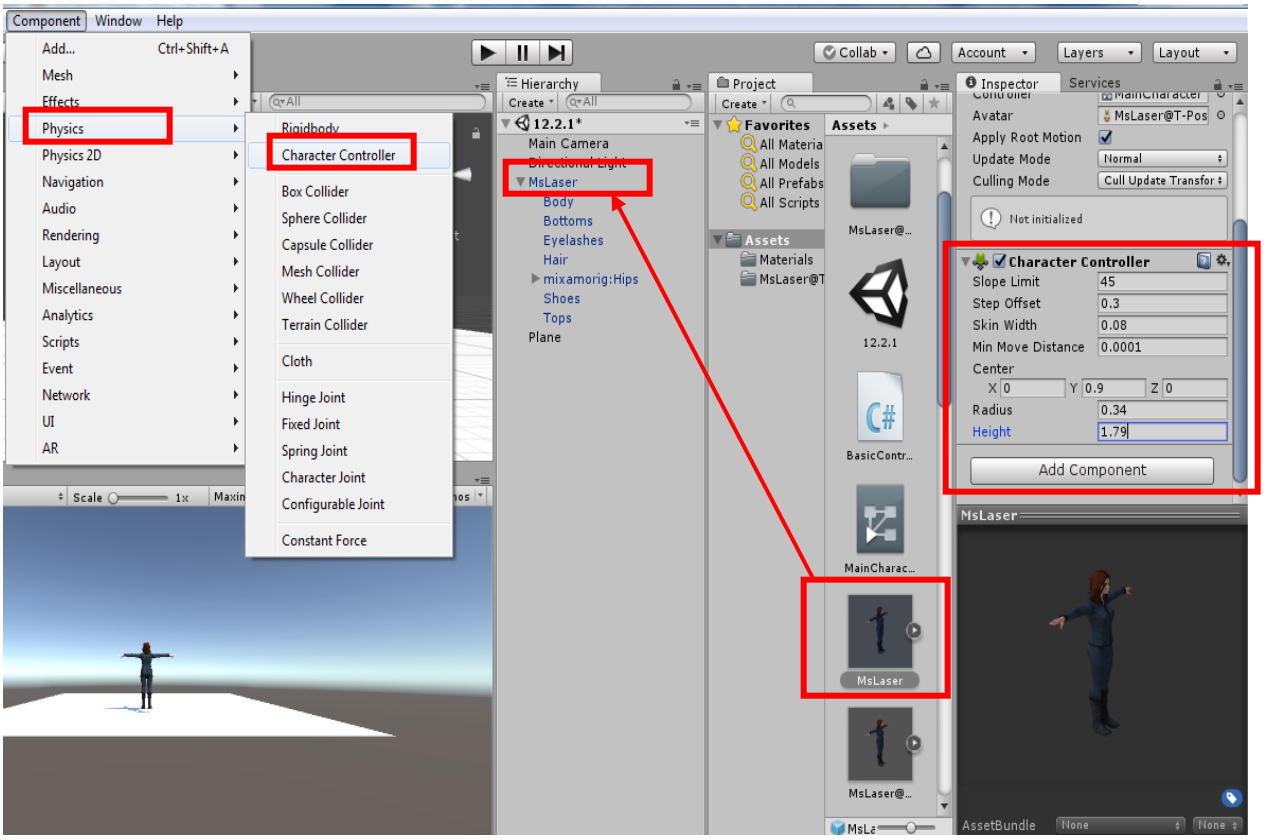
4. Selanjutnya aktifkan bagian **Animations** (di sebelah kanan Rig) pada **Inspector** dan pilih **rifle_run** (Dari daftar **Clips**). *The **Preview Area*** (di bagian bawah tampilan Inspector) akan ditampilkan pesan sebagai **No Model** maka silahkan drag model **MsLaser @ T-Pose** ke **Preview Area** untuk mengisi karakter.



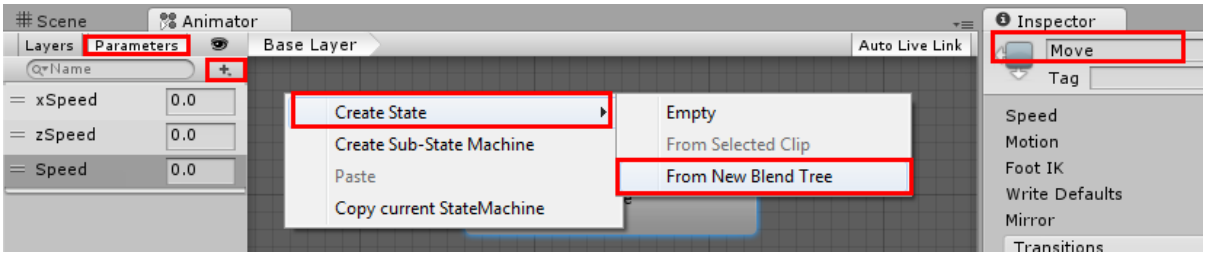
5. Selanjutnya dalam Inspector pilih **rifle_run** dari daftar **Clips** Dan periksa opsi **Loop Time and Loop Pose**. Juga, pilih button pada **Clamp Range** untuk menyesuaikan garis waktu dengan waktu aktual animation clips.
6. Kemudian di bawah **Root Transform Rotation** centang **Bake Into Pose** dan pilih **Baked Upon (at Start) → Original**. Di bawah Root **Transform Position (Y)** centang **Bake Into Pose** dan pilih **Baked Upon | Original**. Di bawah **Root Transform Position (XZ)** biarkanlah **Bake Into Pose** tidak dicentang dan pilih **Baked Upon (at Start) | Center of Mass**, setelah itu klik **Apply** untuk mengkonfirmasi perubahannya.



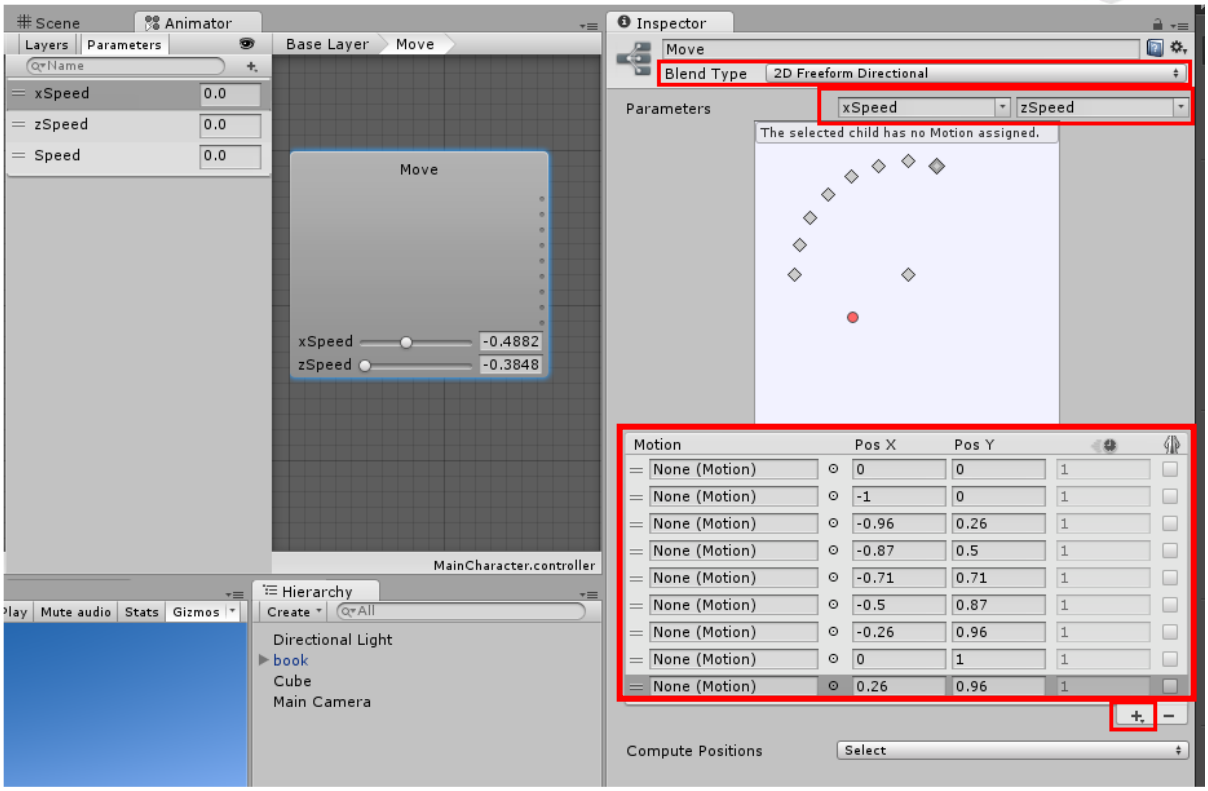
7. Ulangi langkah 3 sampai 6 untuk masing-masing **animation clips** berikut ini: **Swat @ run_backwards, Swat @ strafe, Swat @ strafe_2, Swat @ strafe_left, Swat @ strafe_right, Swat @ walking, dan Swat @ walking_backwards.**
8. Dari tampilan **Project**, pilih **MsLaser** dan drag ke **Hierarchy** untuk memunculkan di tempat scene.
9. Dari tampilan **Hierarchy**, pilih **MsLaser** dan lampirkan pengontrol komponen karakter untuk itu pilih menu **Component → Physics → Character Controller**. Kemudian, atur **Skin Width** menjadi **0,0001**, dan **Center** sebagai **X: 0, Y: 0.9, Z: 0;** Juga berubah **Radius** menjadi **0,34** dan **Height** menjadi **1,79**.



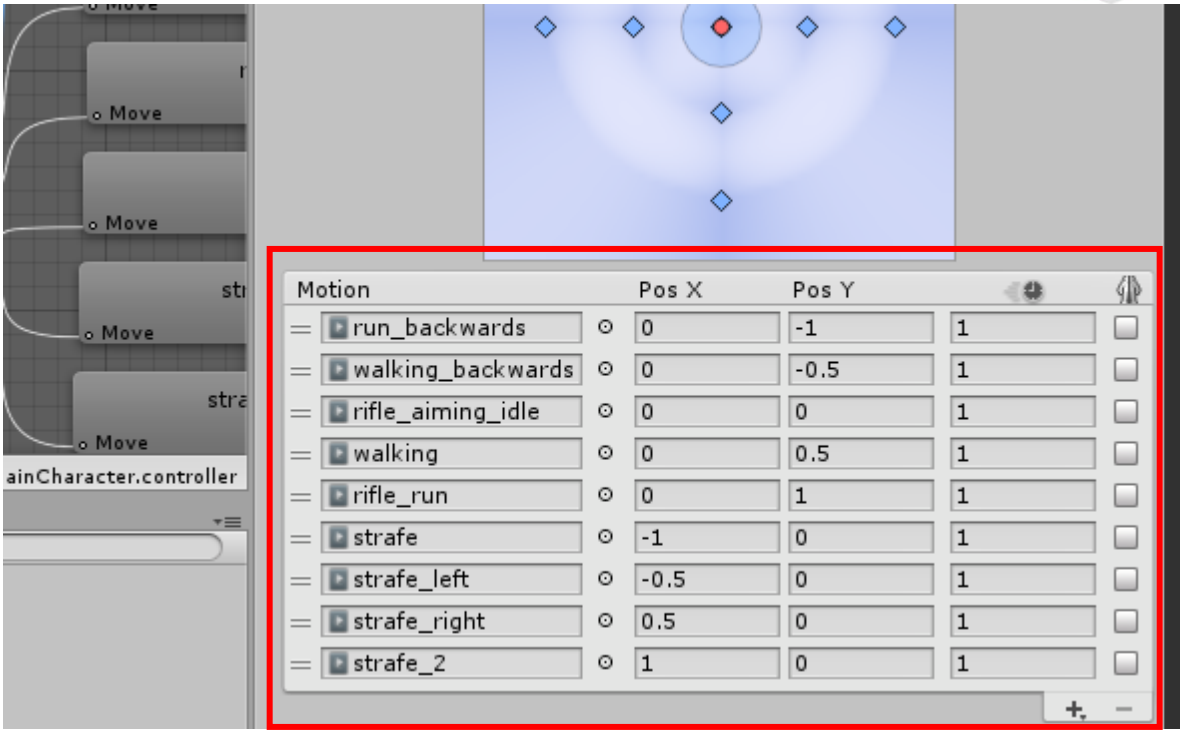
10. Dalam tampilan **Project**, buka **MainCharacter** controller.
11. Di sudut kiri atas tampilan **Animator**, aktifkan bagian **Parameters** dan Gunakan tanda **+** untuk membuat tiga **Parameter** baru (**Float**) bernama **xSpeed, zSpeed**, Dan **Speed**.
12. Kita memiliki status **Idle** untuk karakter, tapi kita membutuhkan yang baru. Klik kanan pada Area **Gridded** dan dari menu konteks arahkan ke **Create State → From New Blend Tree**. Ganti namanya dari tampilan **Inspector** ke **Move**.



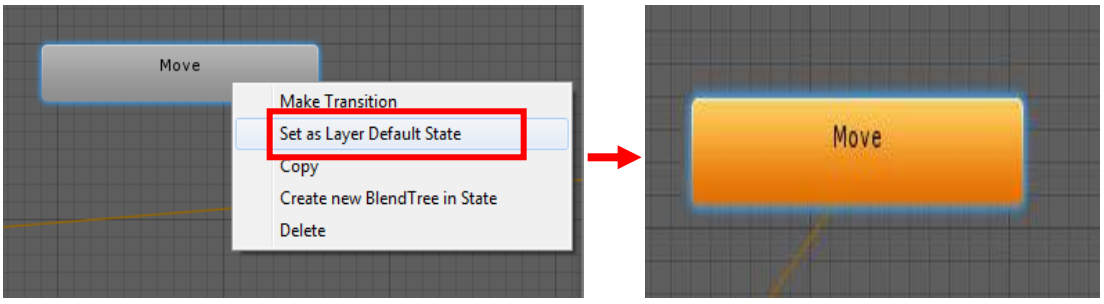
13. Klik dua kali pada **Move** dan akan terlihat *blend tree* kosong yang dimiliki didalam tampilan **Inspector**, ubah namanya menjadi **Move**. Lalu ganti **Blend Type** ke **2D Freeform Directional** juga mengatur **xSpeed** dan **zSpeed** di **Tab Parameter.** Akhirnya, dengan menggunakan tanda **+** dari bagian bawah daftar **Motion**, tambahkan **Sembilan** bidang **Add Motion Field** baru.



14. Sekarang isilah daftar **Motion** dengan clips gerak berikut dan masing-masing **Pos Nilai X dan Pos Y: run_backwards, 0, -1; Walking_backwards, 0, -0,5; senapan_ Aiming_idle, 0, 0; Berjalan, 0, 0,5; Rifle_run, 0, 1; Strafe, -1, 0; Strafe_left, -0,5, 0; Strafe_right, 0.5, 0; Strafe_2, 1, 0**. Anda dapat mengisi daftar **Motion** dengan memilihnya Dari daftar atau jika ada lebih dari satu clips dengan nama yang sama, Anda dapat menariknya Dari tampilan Project ke slot (dengan memperluas ikon model yang sesuai).



15. Klik dua kali pada area **gridded** untuk beralih dari **Move blend tree** ke **Base Layer**.
16. Karena kita memiliki *clip Motion* **rifle_aiming_idle** dalam **Move** *blend tree*, Kita bisa mengganti status **Idle** asli. Klik kanan pada kotak **Move** state pilih **Set as Layer Default State** akan menjadi default state baru dengan berubah warna oranye.



17. Sekarang, kita harus membuat script yang benar-benar akan mengubah masukan pemain menjadi variabel yang diciptakan untuk mengendalikan animasi. Dari tampilan Project, buat **Script C#** baru dan beri nama **BasicController**.
18. Buka script anda dan ganti semuanya dengan kode berikut:

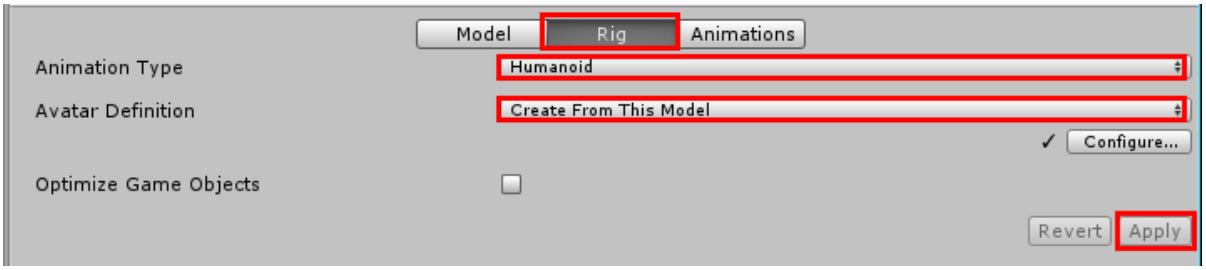
```
using UnityEngine;
using System.Collections;
public class BasicController: MonoBehaviour {
    private Animator anim;
    private CharacterController controller;
    public float transitionTime = .25f;
    private float speedLimit = 1.0f;
    public bool moveDiagonally = true;
    public bool mouseRotate = true;
    public bool keyboardRotate = false;
    void Start () {
    controller = GetComponent<CharacterController>();
    anim = GetComponent<Animator>();
  }
    void Update () {
    if(controller.isGrounded){
    if (Input.GetKey (KeyCode.RightShift) ||Input.GetKey
    (KeyCode.LeftShift))
    speedLimit = 0.5f;
    else
    speedLimit = 1.0f;
    float h = Input.GetAxis("Horizontal");
    float v = Input.GetAxis("Vertical");
    float xSpeed = h * speedLimit;
    float zSpeed = v * speedLimit;
    float speed = Mathf.Sqrt(h*h+v*v);
    if(v!=0 && !moveDiagonally)xSpeed = 0;
    if(v!=0 && keyboardRotate)
    this.transform.Rotate(Vector3.up * h, Space.World);
    if(mouseRotate)
    this.transform.Rotate(Vector3.up * (Input.GetAxis("Mouse X")) * Mathf.Sign(v),
Space.World);
    anim.SetFloat("zSpeed", zSpeed, transitionTime, Time.deltaTime);
    anim.SetFloat("xSpeed", xSpeed, transitionTime, Time.deltaTime);
    anim.SetFloat("Speed", speed, transitionTime, Time.deltaTime);
   }
  }
 }
```

19. Simpan script dan drag ke **MsLaser** di tampilan Hierarchy. Kemudian, tambahkan **Plane** (opsi menu **GameObject → 3D Object → Plane**) dan letakkan di dalamnya di bawah karakter.
20. Mainkan project Anda dan uji permainannya. Anda akan bisa mengendalikan karakter Anda dengan Tombol panah (atau tombol WASD). Menjaga tombol Shift ditekan akan memperlambatnya.

Mixing animations with Layers and Masks

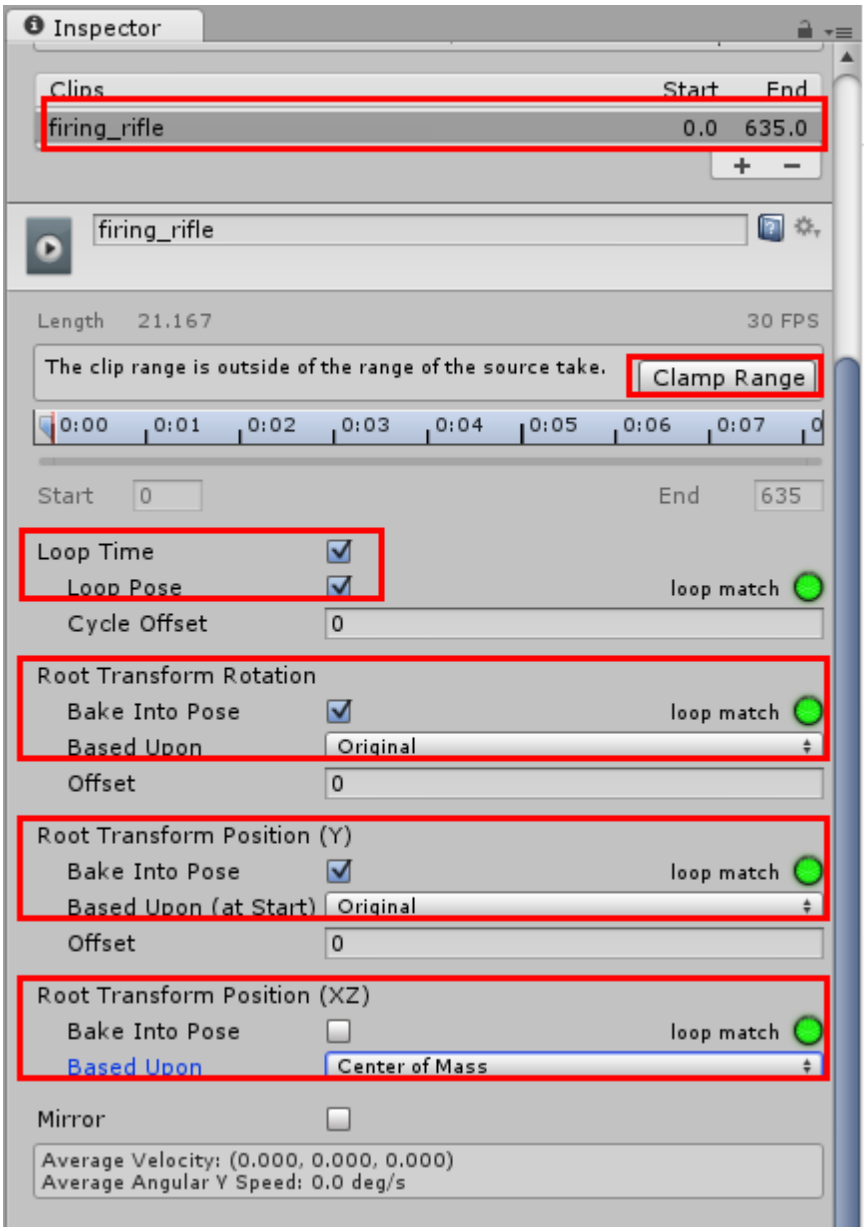
Mixing animations adalah cara terbaik untuk menambahkan kompleksitas pada karakter animasi anda tanpa membutuhkan jumlah besar *animation clips*. Dengan menggunakan *Layers and Masks* kita bisa memadukan perbedaan Animasi dengan memainkan clip untuk bagian tubuh tertentu dari karakter. Dalam project ini, akan diterapkan teknik pada karakter animasi untuk triggering animation clips untuk menembak senapan dan melempar granat dengan tubuh bagian atas karakter. Kami akan melakukan ini sambil menjaga tubuh bagian bawah *moving or idle*, sesuai masukan pemain. Untuk dapat mengetahui hasilnya kerjakan praktek dibawah ini:

1. Untuk project ini, telah disiapkan ***Unity Package*** bernama **Mixing** yang berisi adegan dasar yang memiliki karakter animasi. ***The package*** bisa ditemukan di dalam 1362_07_03 Folder bersama dengan *animation clip* yang disebut **Swat@firing_rifle.fbx dan Swat @ toss_ Grenade.fbx**.
2. Buat project baru dan impor **Mixing Unity Package** lalu dari **Project** bukalah level **mecanimPlayground**.
3. Impor file **Swat@firing_rifle.fbx dan Swat@toss_grenade.fbx** ke project.
4. Kita perlu mengkonfigurasi animation clips dari tampilan **Project** pilih **Swat@klip animasi firing_rifle**.
5. Aktifkan bagian **Rig** ubahlah **Animation Type** menjadi **Humanoid**, dan **Avatar Definition** menjadi **Create From This Model**. Selanjutnya tekan button dengan mengklik **Apply**.



6. Sekarang aktifkan bagian **Animation** dan pilih **firing_rifle** (dari daftar **Clip**). Klik pada button **Clamp Range** untuk mengatur garis waktu dan periksa **Loop Time** Dan centang **Loop Pose**. Di bawah **Root Transform Rotation** centang **Bake Into Pose** dan Pilih **Baked Upon → Original**. Pada **Root Transform Position(Y)** centang **Bake Into Pose** dan pilih **Baked Upon (at Start) → Original**. Di bawah **Root Transform Position (XZ)**, biarkan **Bake**

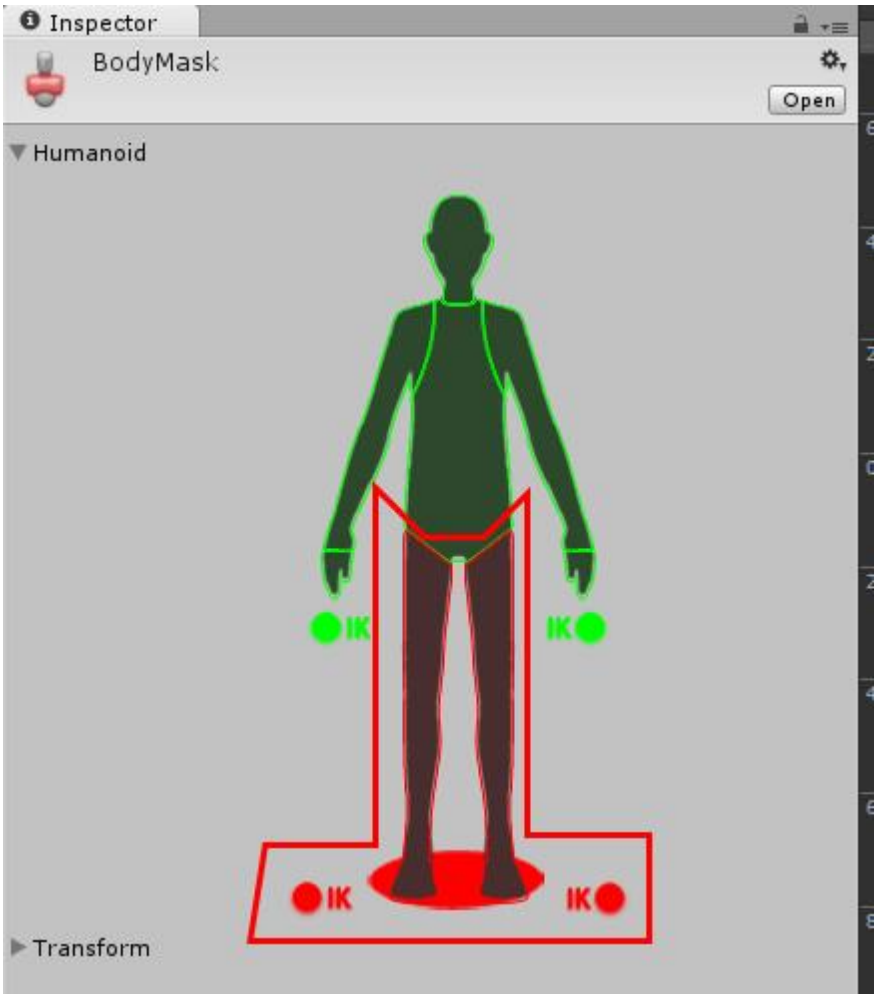
Into Pose tidak dicentang juga **Based Upon → Center Of Mass**. Klik **Apply** untuk mengkonfirmasi perubahan.



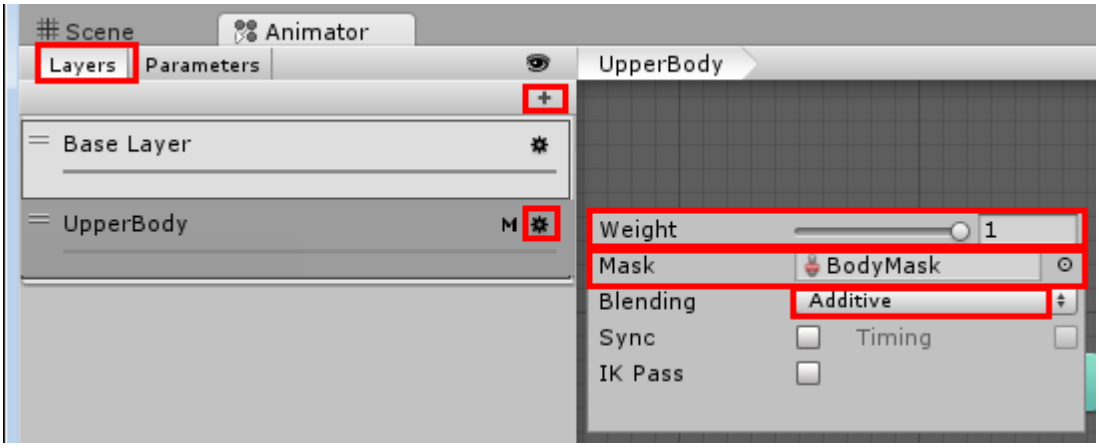
7. Pilih klip animasi **Swat@toss_grenade** klik bagian **Rig** di **Inspector**. Kemudian ubah **Animation Type** menjadi **Humanoid** dan **Avatar Definition** ke **Create From this Model**. Konfirmasikan dengan mengklik **Apply**.
8. Sekarang pilih bagian **Animation** pada **Inspector**. Pilih **toss_grenade** (dari **Clips List**) klik pada tombol **Clamp Range** untuk menyesuaikan garis waktu dan tidak dicentang pada **Loop Time and Loop Pose**. Di bawah **Root Transform Rotation** centang **Bake Into Pose** dan pilih **Baked Upon (at Start)** → Original. Di bawah **Root Transform Position (Y)** centang **Bake Into Pose** dan pilih **Baked Upon (at Start)** → **Original**. Di bawah **Root**

Transform Position (XZ) biarkan Bake Into Pose tidak dicentang. Klik **Apply** untuk mengkonfirmasi perubahan.

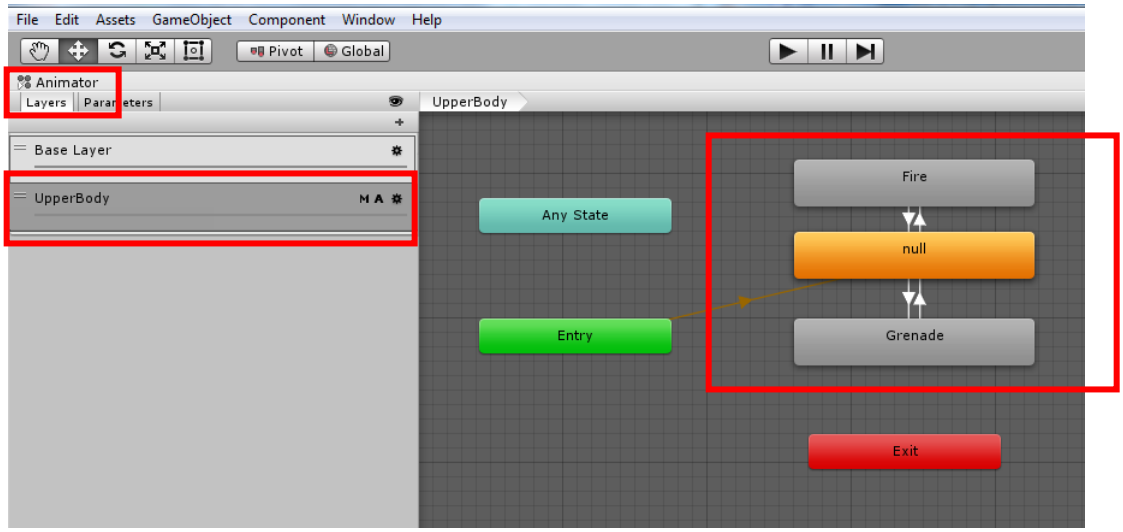
9. Mari kita buat sebuah **Mask**. Dari tampilan **Project**, klik pada tombol **Create** dan tambahkan sebuah **Avatar Mask** untuk project ini serta ganti nama sebagai **BodyMask**.
10. Pilih **BodyMask** dan di tampilan **Inspector** luaskan bagian **Humanoid** untuk tidak memilih the character legs, base, IK Spots atau mengubah garis besar merah.



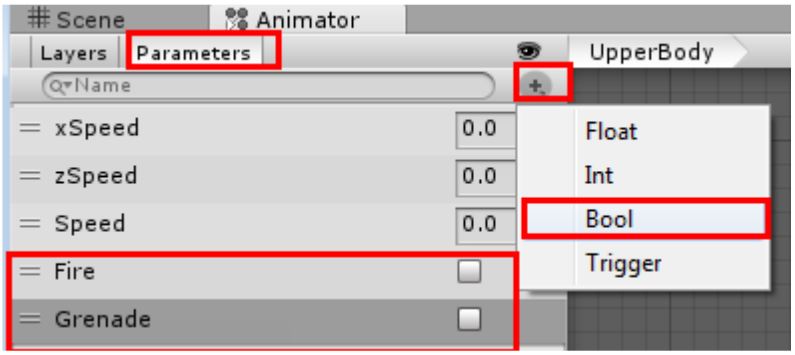
11. Dari tampilan **Hierarchy** pilih karakter **MsLaser**. Lalu dari **Animator** komponen dalam tampilan **Inspector** klik dua kali pada *controller* **MainCharacter**.
12. Pada tampilan **Animator** buat layer baru dengan mengklik tanda **+** di kiri atas **Layers** tab, di atas **Base Layer**.
13. Beri nama layer baru sebagai **UpperBody** dan klik ikon **gear** untuk pengaturannya. Kemudian, Ubah **Weight** ke 1 dan pilih **BodyMask** di slot **Mask** juga ganti **Blending** to **Additive**.



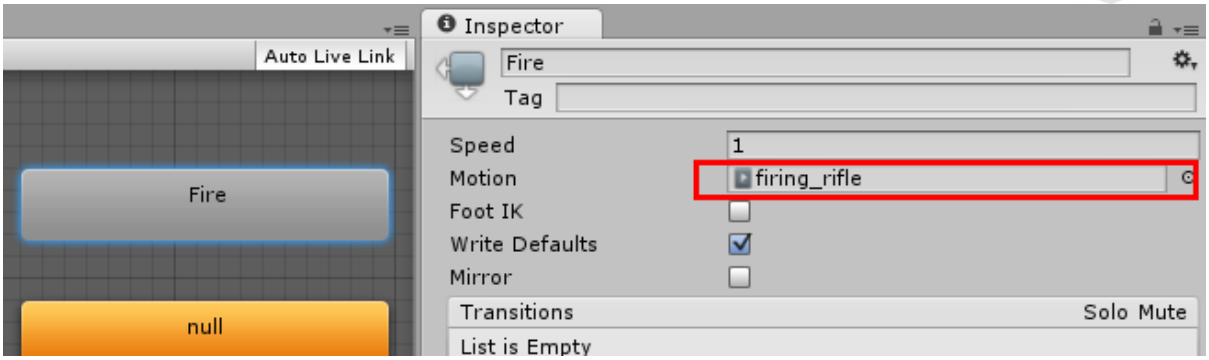
14. Di tampilan **Animator** dengan layer **UpperBody** yang dipilih, buatlah tiga baru **Empty States** (dengan mengklik kanan pada area **gridded** dan menavigasi menu **Create State → Empty**). Nama default (oranye) state **null** dan dua lainnya sebagai **Fire** dan **Grenade**.



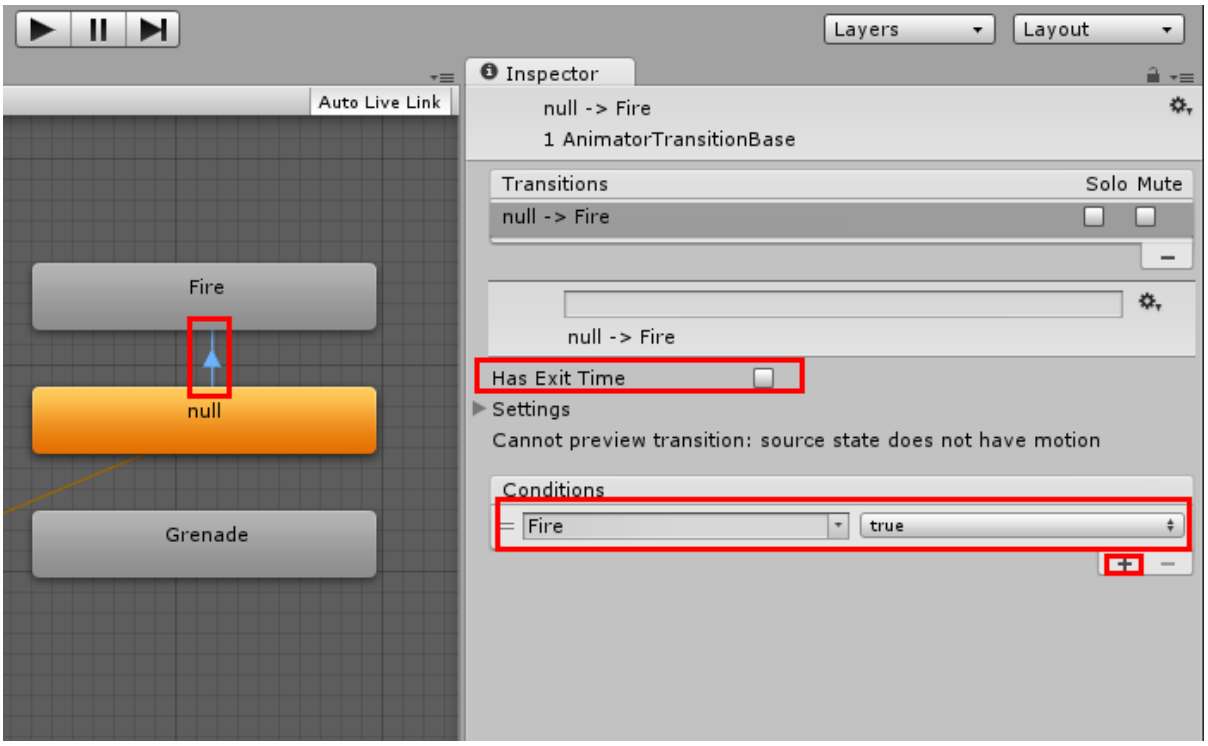
15. Sekarang akses tab **Parameters** dan tambahkan dua parameter baru dari tipe Boolean: **Fire** dan **Grenade**.



16. Pilih **Fire** dan dalam tampilan **Inspector** tambahkan animasi clip **firing_rifle** ke **Motion** *field*

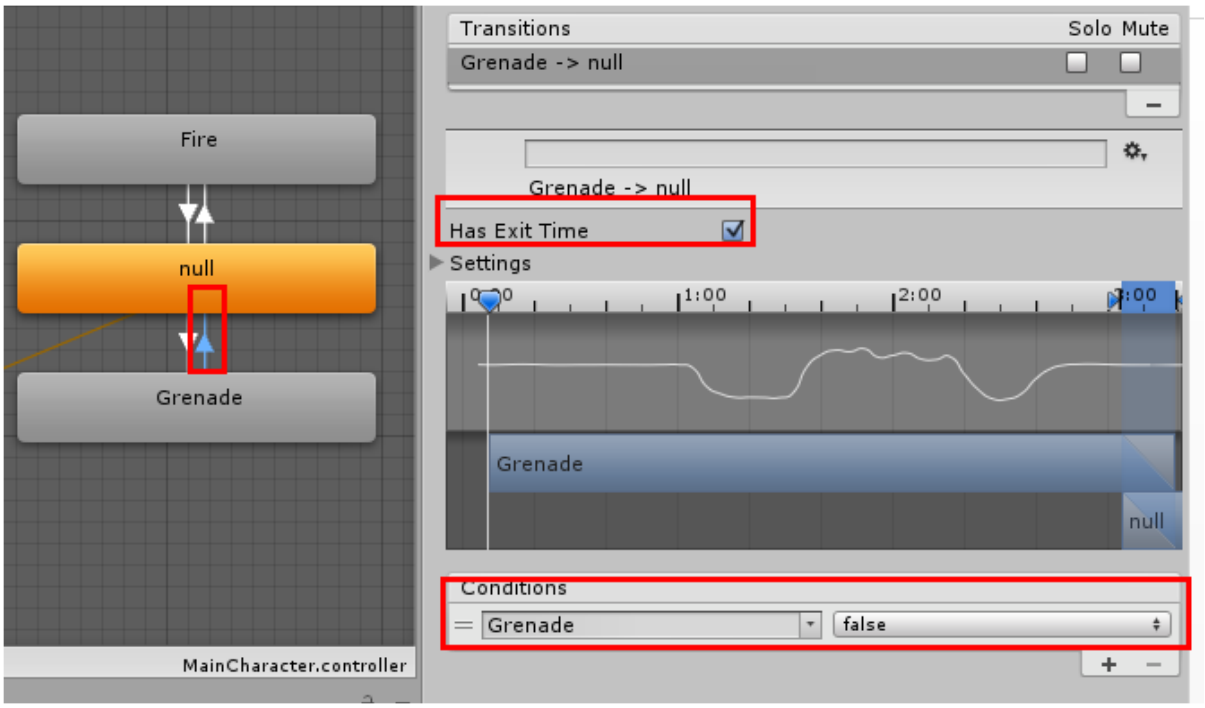


17. Sekarang pilih **Grenade** dan di tampilan **Inspector** tambahkan **toss_grenade** animasi clip ke **Motion** *field*.
18. Klik kanan pada kotak negara null dan dari menu pilih Make Transition. Kemudian Tarik panah putih ke Fire.
19. Pilih panah (akan berubah menjadi biru). Dari tampilan Inspector hapus centang pada **Has Exit Time,** Kemudian akses daftar **Conditions**, klik tanda **+** untuk menambahkan yang Conditions baru dan menetapkannya sebagai **Fire** dan **True**.



20. Buatlah transisi dari **null** ke **Grenade**. Pilih panah-nya (akan berubah menjadi biru) dari tampilan **Inspector**, hapus centang pada opsi **Has Exit Time**. Lalu, akses daftar **Conditions** klik tanda **+** untuk menambahkan kondisi baru dan tetapkan sebagai **Grenade** dan **True**.

21. Sekarang, buatlah transisi dari **Fire** ke **null** dan dari **Grenade** menjadi **null**. Kemudian, pilih panah yang masuk dari **Fire** ke **null** dan di kotak **Conditions** pilih **Fire** dan **false**. Biarkan opsi **Has Exit Time** dicentang.
22. Selanjutnya pilih panah yang masuk dari **Grenade** ke **null**. Dalam kotak **Conditions** pilih **grenade** dan **false** dan Biarkan opsi **Has Exit Time** dicentang.



23. Dari tampilan **Project** drag karakter **MsLaser** ke **Hierarchy**. Temukan dalam **Project** script C# **Basic Controller** dan buka skripnya.
24. Segera sebelum akhir fungsi **Update ()**, tambahkan kode berikut ini:

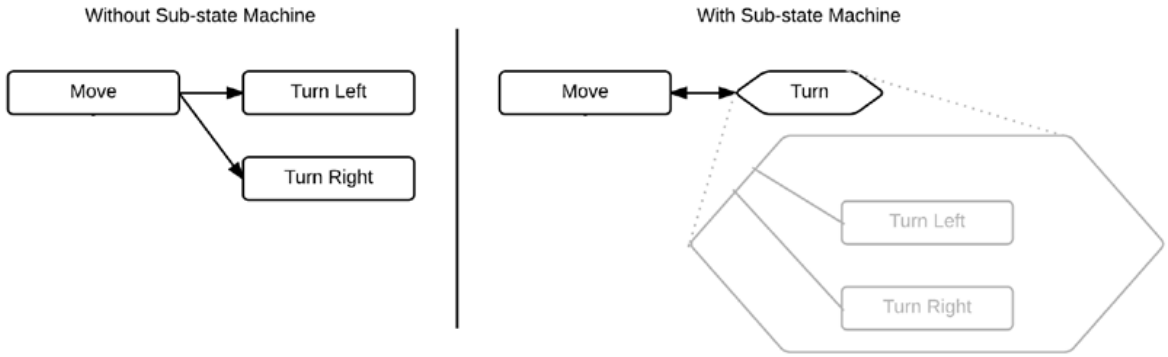
```
        if (Input.GetKeyDown(KeyCode.F))
        {
            anim.SetBool("Grenade", true);
        }
        else
        {
            anim.SetBool("Grenade", false);
        }
        if (Input.GetButtonDown("Fire1"))
        {
            anim.SetBool("Fire", true);
        }
        if (Input.GetButtonUp("Fire1"))
        {
            anim.SetBool("Fire", false);
        }

    }
```

25. Simpan script dan mainkan project Anda akan bisa memicu **firing_rifle** dan **Toss_grenade** animasi dengan mengklik tombol **fire** dan menekan tombol **F**. Amati bagaimana kaki karakter tetap merespons keadaan animasi **Move**.

Organizing States into Sub-state Machines

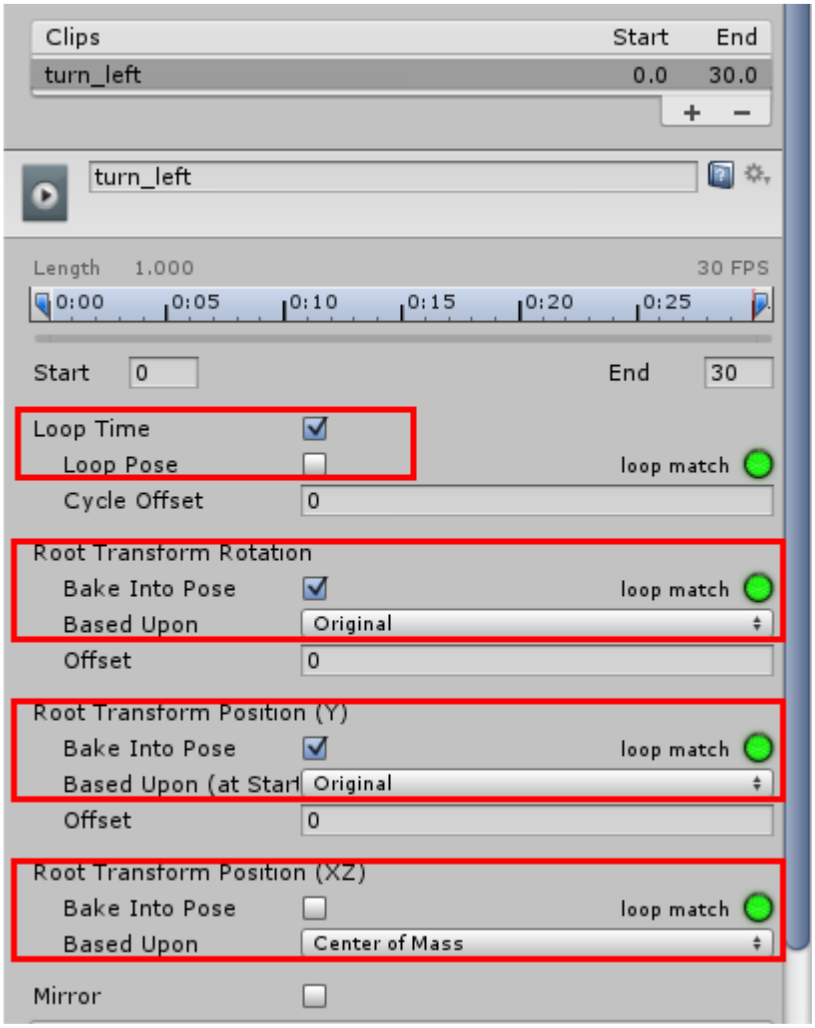
Setiap kali di area *Animator* terlalu *clustered* untuk itu bisa dipikirkan untuk mengaturnya *Animation State* menjadi Sub-State Machines. Dalam project ini akan menggunakan teknik mengatur *Animation State* untuk mengubah karakter. Juga sejak *animasi clips* yang disediakan jangan sertakan *Root Motion*, gunakan kesempatan untuk menggambarkan bagaimana mengatasi kekurangan *Root Motion* melalui script dan menggunakannya untuk mengubah karakter 45 derajat ke kiri dan ke kanan.



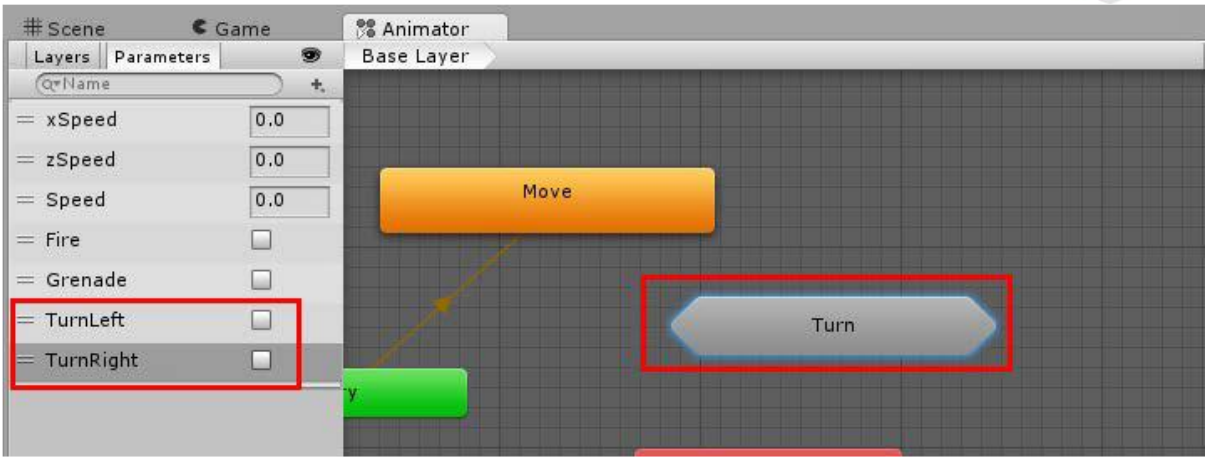
Kerjakan langkah-langkah project dibawah ini untuk mengetahui hasilnya:

1. Untuk project ini telah disiapkan *Unity Package* bernama **Turning**, berisi sebuah adegan dasar yang memiliki karakter animasi. Paketnya bisa ditemukan di dalam folder 1362_07_04, beserta clip animasi yang disebut **Swat@turn_right_45_degrees.fbx dan Swat@turn_left.fbx.**
2. Buat proyek baru dan impor **Turning** *Unity Package* lalu dari Project Lihat dan buka **mecanimPlayground** level.
3. Impor File **Swat@turn_right_45_degrees.fbx** dan **Swat@turn_left.fbx** ke dalam proyek
4. Mengkonfigurasikan klip animasi dengan memilih file **Swat @ turn_left** dari Tampilan **Project.**
5. Pilihlah bagian **Rig** didalam **Inspector** dan gantilah **Animation Type** menjadi **Humanoid** juga **Avatar Definition** menjadi **Create From this Model**. Setelah itu baru klik button **Apply**.

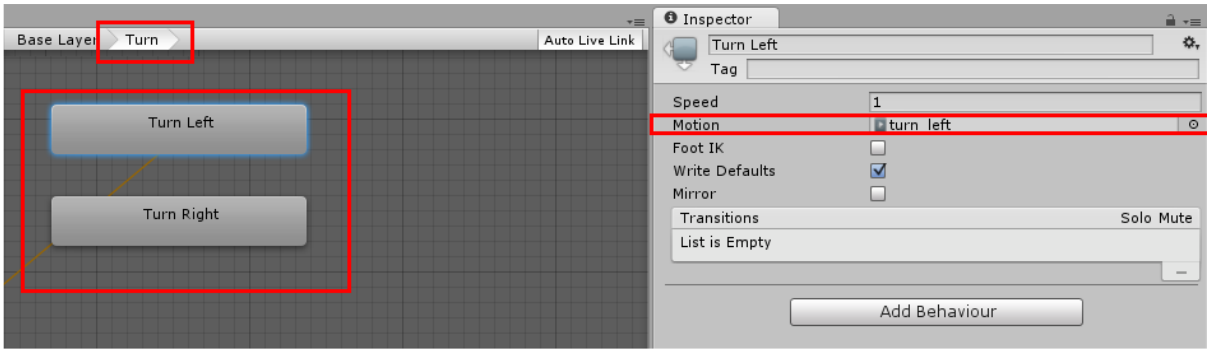
6. Sekarang aktifkan bagian **Animations** pilih clip **turn_left** (dari daftar **Clip**), klik Pada tombol **Clamp Range** untuk menyesuaikan garis waktu dan periksa opsi **Loop Time**. Di bawah **Root Transform Rotation** centang **Bake Into Pose** dan arahkan ke **Baked Upon → Original**. Di bawah **Root Transform Position (Y)** centang **Bake Into Pose** dan Pilih **Baked Upon (at Start) → Original**. Di bawah **Root Transform Position (XZ)** biarkan tidak dicentang pada **Bake Into Pose** dan pilih **Center of Mass**. Selanjutnya klik **Aplly** untuk mengkonfirmasi perubahan.



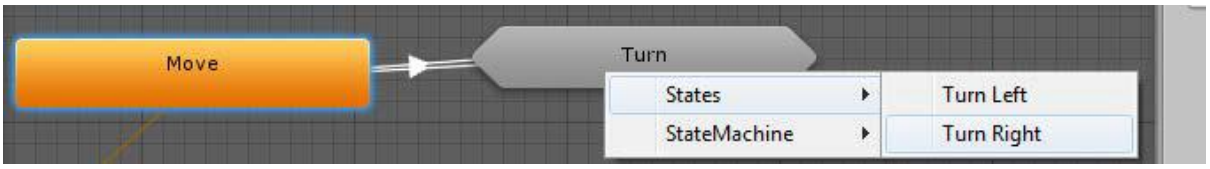
7. Ulangi langkah 5 dan 6 untuk **Swat @ turning_right_45_degrees**.
8. Dari tampilan **Hierarchy** pilih karakter **MsLaser** lalu dari komponen **Animator** dalam tampilan Inspector buka **MainCharacter**.
9. Dari sudut kiri atas tampilan **Animator** aktifkan bagian Parameters dan gunakan tanda **+** untuk membuat dua **Parameter baru (Boolean)** bernama **TurnLeft** dan **TurnRight**.
10. Klik kanan pada area **gridded**. Dari menu konteks pilih **Create Sub-State Machine**. Dari tampilan **Inspector** ubahlah namanya menjadi **Turn**.



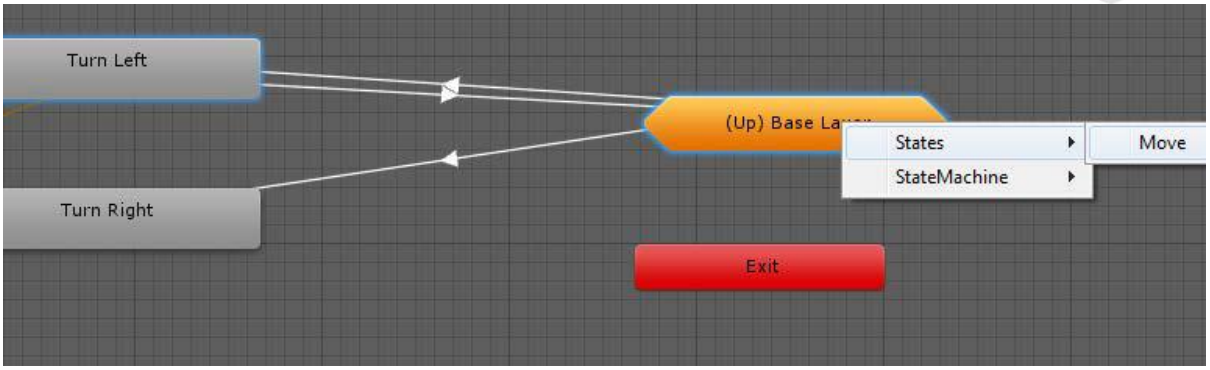
11. Klik dua kali pada sub-state **Turn** selanjutnya Klik kanan pada area **gridded**, pilih **Create State → Empty** dan tambahkan new state. Ganti nama menjadi **Turn Left** juga tambahkan **state** lain bernama **Turn Right**.
12. Dari tampilan **Inspector** isilah **Turn Left** dengan clip motion **turn_left**. Kemudian mengisi **Turn Right** dengan **turning_right_45_degrees**.



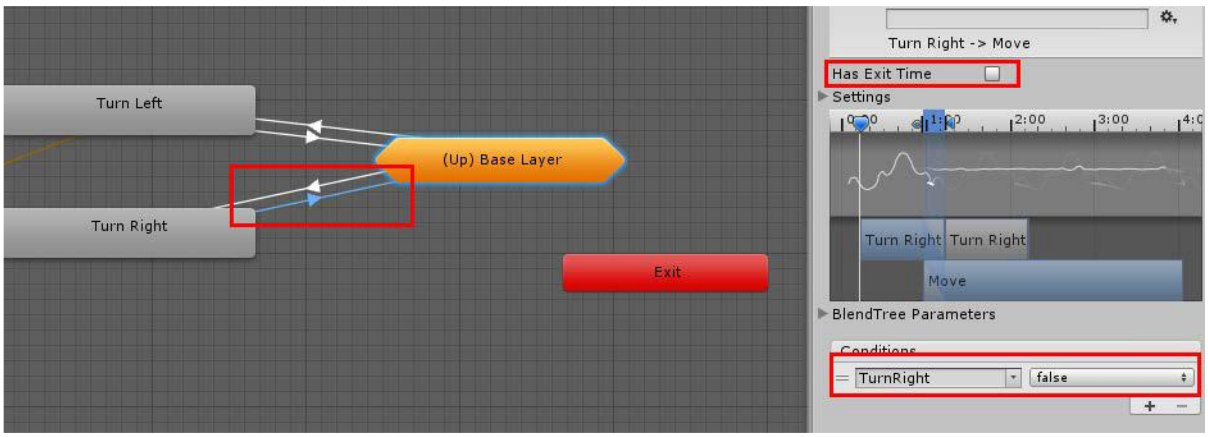
13. Keluar dari **turn sub-state** kembali ke **Base Layer**. Dengan mengklik kanan setiap bagian **State** dan memilih pilihan **Make Transition**, buat transisi antara **Move** ke **Turn Left** dan **Move** ke **Turn Right**.



14. Masukkan **Turn** sub-state machine kemudian buat transisi dari **Turn Left** dan **Turn Right** Langsung ke state **Move**.



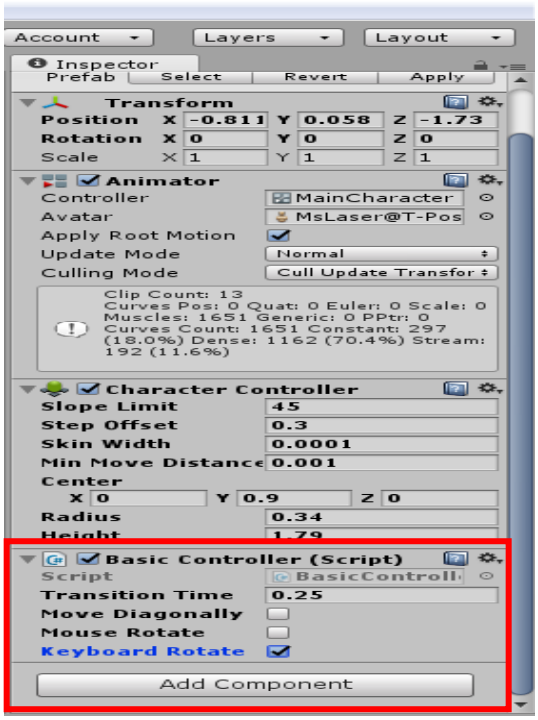
15. Pilih panah yang menuju **form Turn Right** ke **(Up) Base Layer** dan akan menjadi biru. Dari Tampilan Inspector hapus centang pada opsi **Has Time Exit** lalu akses daftar **Conditions**, klik tanda **+** untuk menambahkan **Conditions** baru dan atur sebagai **TurnRight** dan **false**.



16. Pilih panah yang masuk dari **(Up) Base Layer** ke **Turn Right** dan dari tampilan **Inspector** hapus centang opsi **Has Exit Time**. Kemudian, akses daftar **Conditions** klik tombol **+** tanda untuk menambahkan *new conditions* dan atur sebagai **TurnRight** dan **true**.
17. Ulangi langkah 15 dan 16 dengan panah yang berada di antara **(Up) Base Layer** dan **Turn Left,** menggunakan kondisi **TurnLeft** sebagai syarat.
18. Dari tampilan **Project** buka script **BasicController** dari **Project**.
19. Isikan script dibawah setelah **if (controller.isGrounded)** {line, tambahkan:

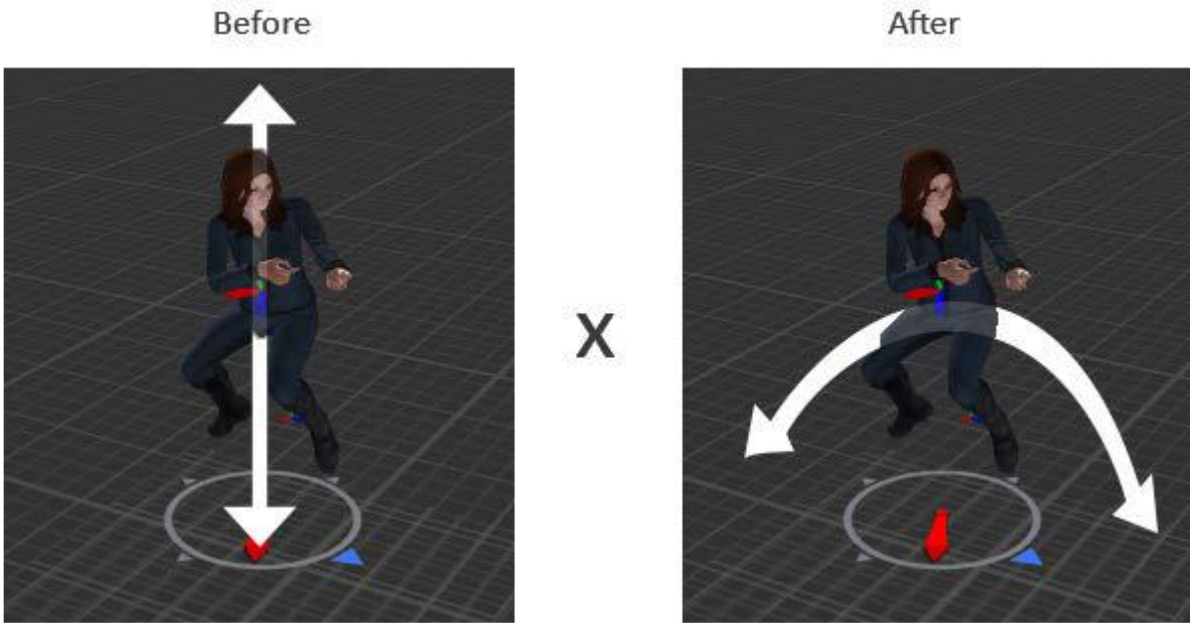
```
if (Input.GetKey(KeyCode.Q))
            {
                anim.SetBool("TurnLeft", true);
                transform.Rotate(Vector3.up *
(Time.deltaTime * -45.0f),
                Space.World);
            }
            else
            {
                anim.SetBool("TurnLeft", false);
            }
            if (Input.GetKey(KeyCode.E))
            {
                anim.SetBool("TurnRight", true);
                transform.Rotate(Vector3.up *
(Time.deltaTime * 45.0f), Space.
                World);
            }
            else
            {
                anim.SetBool("TurnRight", false);
            }
```

20. **Save script** kemudian drag karakter **MsLaser** dari tampilan **Inspector** ke Hierarchy dan drag komponen **Basic Controller**. Lihat pada **Inspector** ada **Move Diagonal** dan **Mouse Rotate** pilih untuk tidak dicentang, selain itu pilihan **Keyboard Rotate** dicentang. Akhirnya jalankan **project** anda akan bisa berbelok ke kiri dan kanan dengan menggunakan tombol **Q** dan **E**.



Transforming the Character Controller Via Script

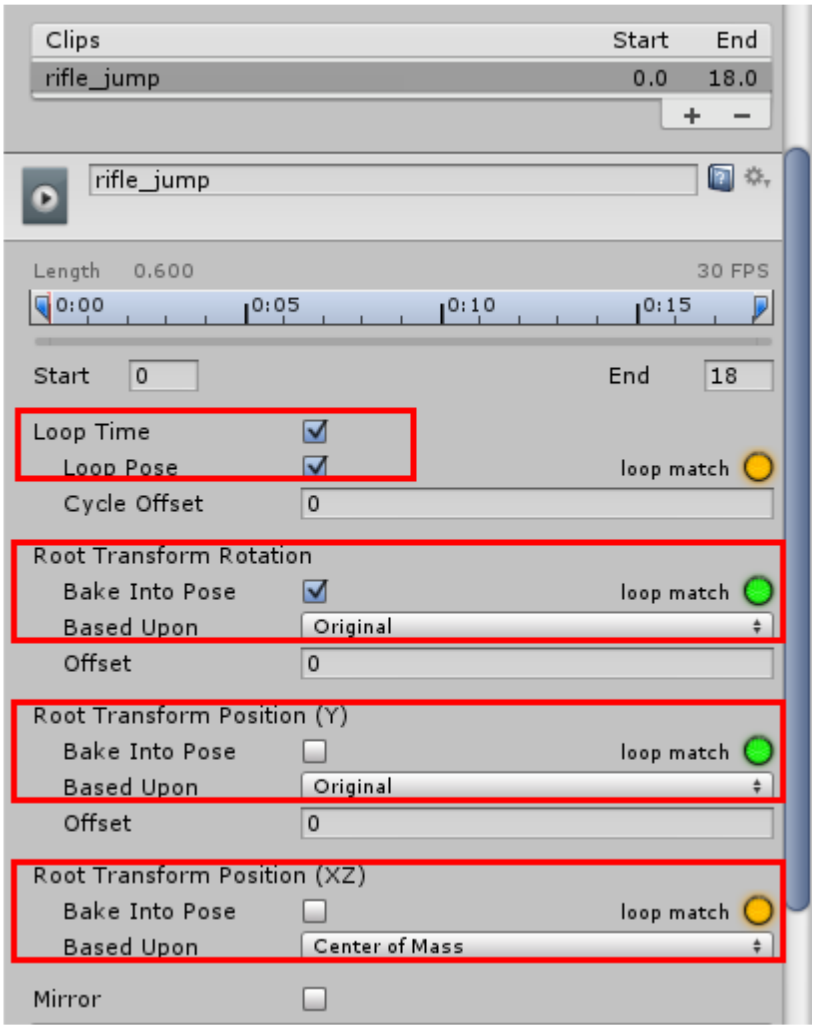
Menerapkan *Motion Root* pada karakter mungkin cara yang sangat praktis dan akurat untuk animasi. Namun, sesekali mungkin perlu mengendalikan satu atau dua aspek secara manual gerakan karakter atau mungkin Ingin gerakan karakter itu terpengaruh oleh variabel lain. Dalam kasus ini, Anda akan membutuhkannya untuk menimpa gerakan *root* melalui script. Untuk mengilustrasikan masalah ini, project ini memanfaatkan klip animasi untuk melompat yang awalnya memindahkan karakter hanya pada sumbu Y. Untuk membuatnya bergerak maju atau mundur dengan melompat, kita akan belajar bagaimana mengakses kecepatan karakter untuk menginformasikan arah lompatan via script.



Untuk menerapkan Root Motion melalui script, ikuti langkah-langkah ini:

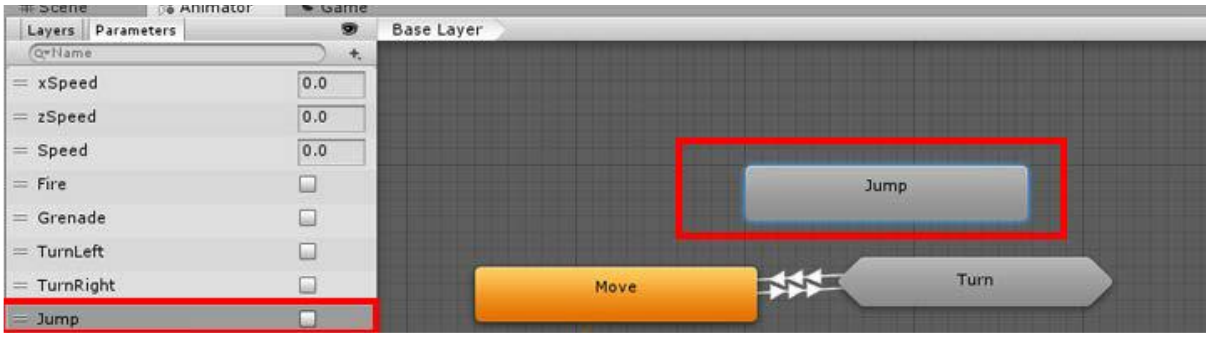
1. Untuk **project** ini telah disiapkan *Unity Package* bernama **Jumping** yang berisi adegan dasar yang memiliki karakter animasi. **Package** bisa ditemukan di dalam Folder **1362_07_05** bersama dengan clip animasi yang disebut **Swat @ rifle_jump**.
2. Buat *project* baru dan impor package **Jumping l**alu dari tampilan **Project** buka level **mecanimPlayground**.
3. Impor file **Swat@rifle_jump.fbx** ke **Project**.
4. Perlu konfigurasi clip animasi dengan cara dari tampilan **Project** pilih file **Swat @ rifle_jump**

5. Aktifkan bagian **Rig** yang ada pada **Inspector,** ubahlah **Animation Type** menjadi **Humanoid** dan **Avatar Definition** ke **Create From This Model.** Selanjutnya mengklik **Apply** yang ada dibawah **Inspector**.
6. Sekarang aktifkan bagian Animation pilih clip **rifle_jump** (dari daftar clip), Klik pada tombol **Clamp Range** untuk mengatur garis waktu dan periksa **Loop Time** juga pilih **Loop Pose**. Di bawah **Root Transform Rotation** centang **Bake Into Pose** dan Pilih **Baked Upon (at Start) → Original**. Di bawah **Root Transform Position (Y)** biarkan untuk tidak dicentang pada **Bake Into Pose** dan pilih **Baked Upon (at Start) → Original**. Di bawah **Root Transform Position (XZ)** biarkan untuk tidak dicentang pada bagian **Bake Into Pose**. Klik **Apply** untuk konfirmasi perubahan.

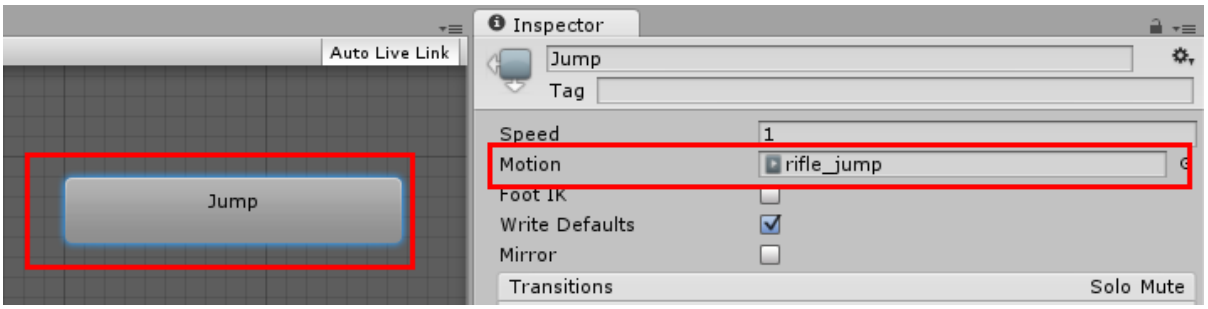


7. Dari tampilan **Hierarchy** pilih karakter **MsLaser** lalu pilih dikomponen **Animator** dalam tampilan **Inspector** buka pengontrol **MainCharacter**.
8. Dari sudut kiri atas tampilan **Animator** aktifkan bagian **Parameters** dan gunakan tanda **+** untuk membuat **Parameter** baru **(Boolean)** bernama **Jump**.

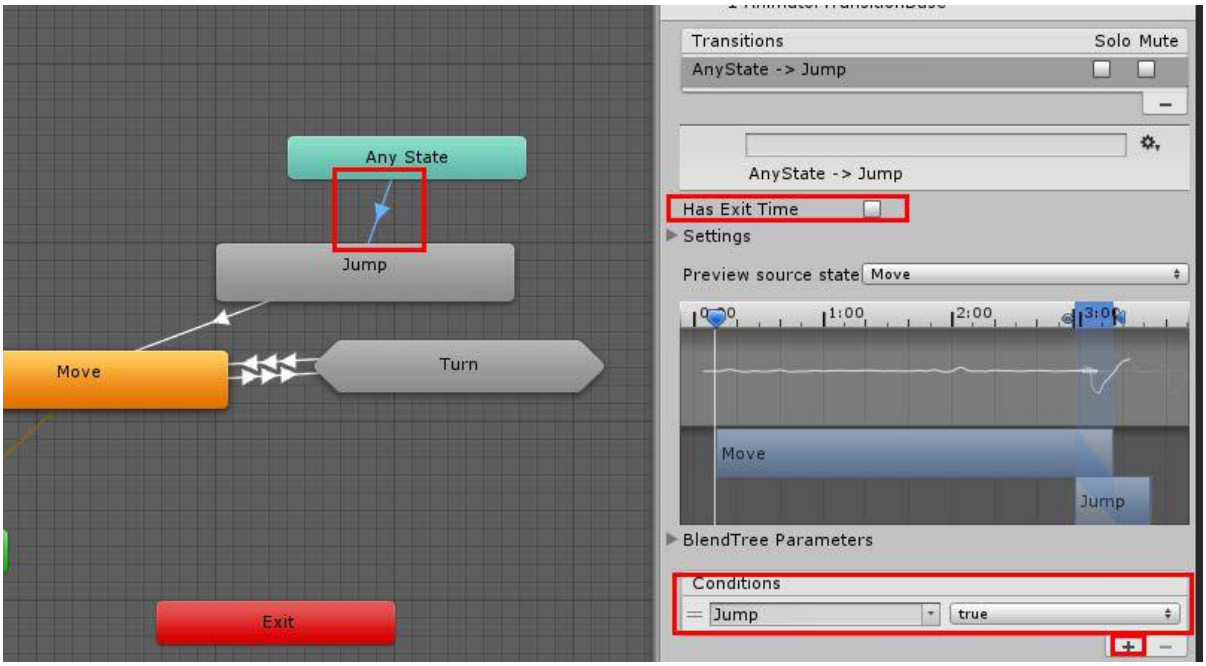
9. Klik kanan pada area **gridded** dan dari menu konteks pilih **Create State** → Empty selanjutnya ganti namanya dari tampilan Inspektur dengan **Jump**.



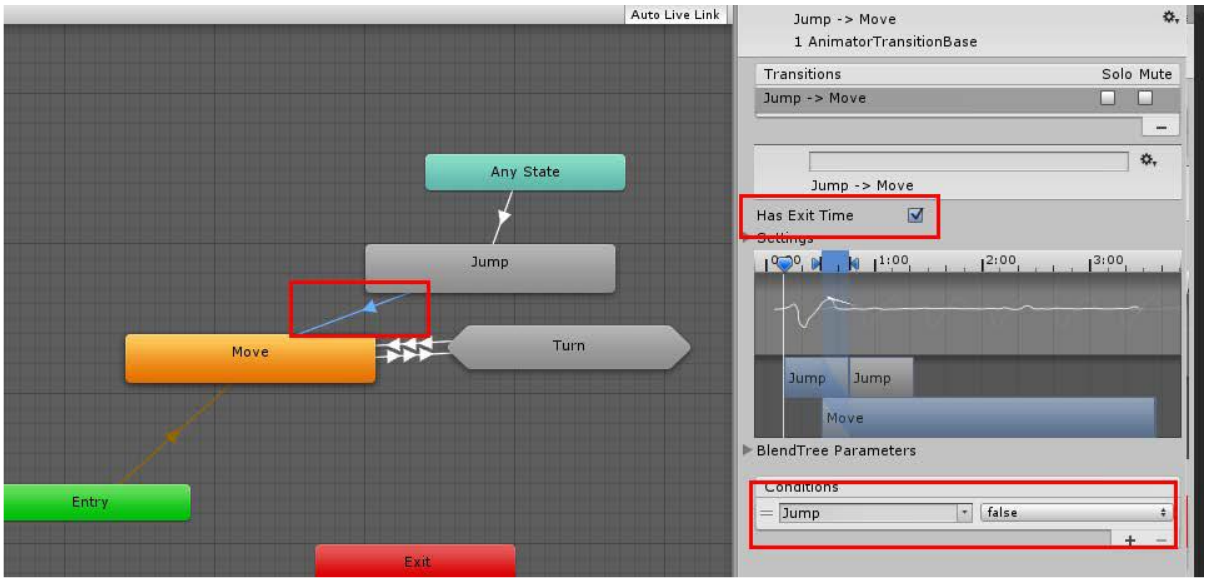
10. Pilih status **Jump** kemudian dari tampilan **Inspector** pilih dengan isi clip gerak **rifle_jump**.



11. Cari dan klik kanan di Any State yang ada pada **Animator** kemudian pilih opsi **Make Transition**, Buat transisi dari **Any State** ke **Jump**. Pilih transisi dan hapus centang **Has Exit Time** dan gunakan variabel **Jump** sebagai **Conditions (true)**.
12. Sekarang, buat transisi dari **Jump** to **Move**.



13. Konfigurasikan transisi antara **Jump** dan **Move** biarkan Has Exit Time tidak dicentang dan gunakan variabel **Jump** sebagai **Conditions (false)**.



14. Dari tampilan **Hierarchy** pilih karakter **MsLaser** lalu dari lihat Inspector buka script dari komponen **BasicController**.
15. Tepat sebelum fungsi **Start ()**, tambahkan kode berikut:

```
public float jumpHeight = 3f;
    private float verticalSpeed = 0f;
    private float xVelocity = 0f;
    private float zVelocity = 0f;
```

16. Di dalam fungsi **Update ()**, temukan baris yang berisi kode berikut:

```
if(controller.isGrounded){
```

Dan tambahkan baris berikut dengan segera setelah itu:

```
if (Input.GetKey(KeyCode.Space))
            {
                anim.SetBool("Jump", true);
                verticalSpeed = jumpHeight;
            }
```

17. Akhirnya tambahkan fungsi baru sebelum akhir kode **}**:

```
void OnAnimatorMove()
    {
        Vector3 deltaPosition = anim.deltaPosition;
        if (controller.isGrounded)
        {
            xVelocity = controller.velocity.x;
            zVelocity = controller.velocity.z;
        }
        else
        {
            deltaPosition.x = xVelocity * Time.deltaTime;
            deltaPosition.z = zVelocity * Time.deltaTime;
            anim.SetBool("Jump", false);
        }
        deltaPosition.y = verticalSpeed * Time.deltaTime;
        controller.Move(deltaPosition);
        verticalSpeed += Physics.gravity.y * Time.deltaTime;
        if ((controller.collisionFlags & CollisionFlags.Below)
        != 0)
        {
            verticalSpeed = 0;
        }
    }
```

18. Save script Anda dan mainkan projectnya. Project akan bisa melompat-lompat menggunakan Space. Amati bagaimana kecepatan karakter mempengaruhi arah lompatan.

TUGAS PRAKTIKUM

1. Analisis dan buat catatan untuk mendiskripsikan setiap hasil project diatas apa saja yang membedakan satu project dengan project lainnya.
2. Pada project nomer 1 buatlah MsLaser bergerak dengan tombol WASD.
3. Presentasikan setiap hasil project ke dosen via video record secara singkat.

--- SELAMAT BELAJAR ---