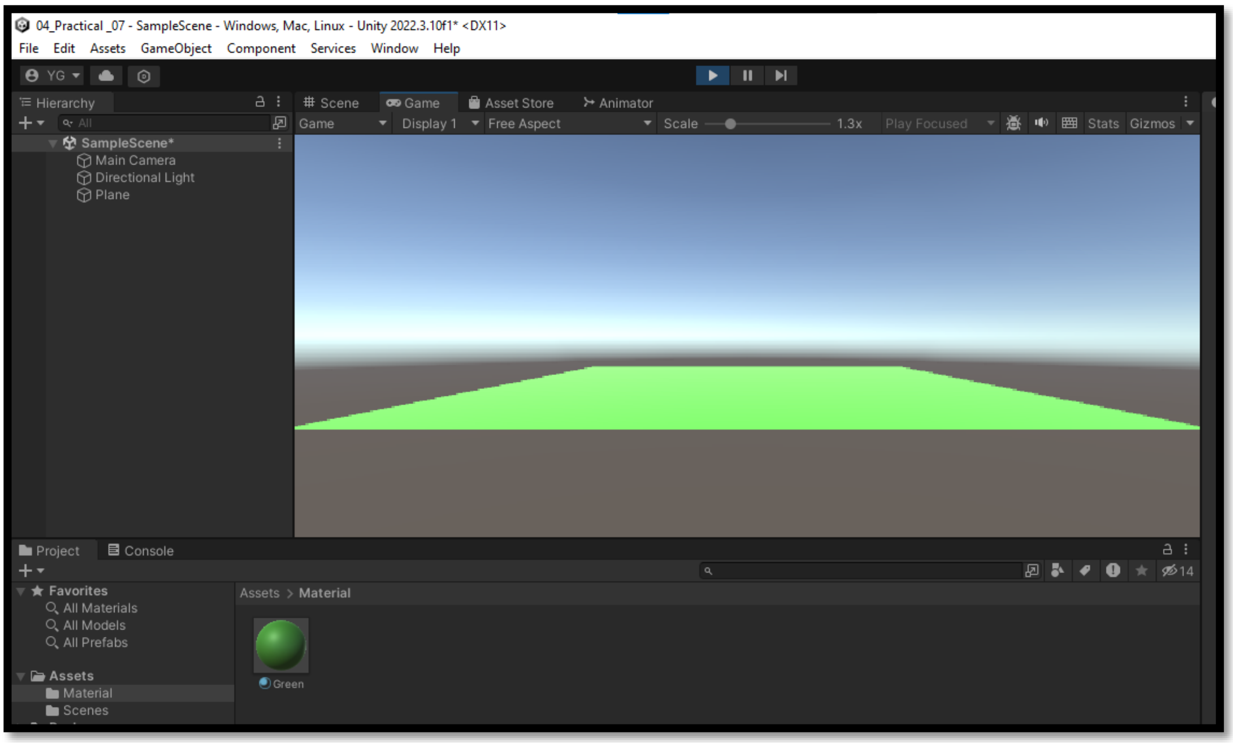
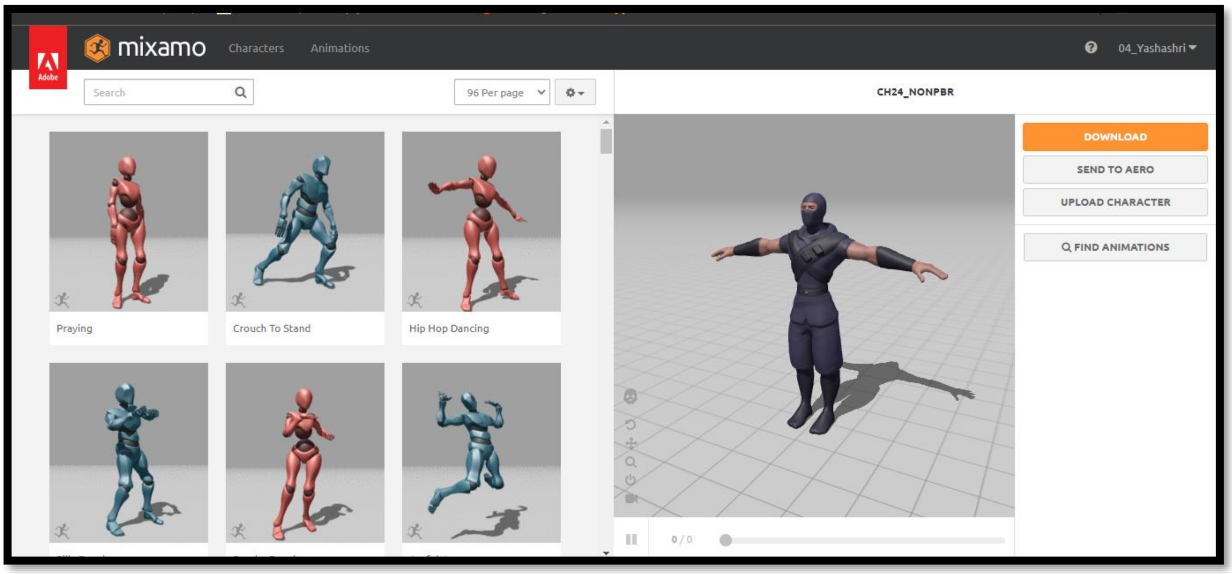
Practical No. 07

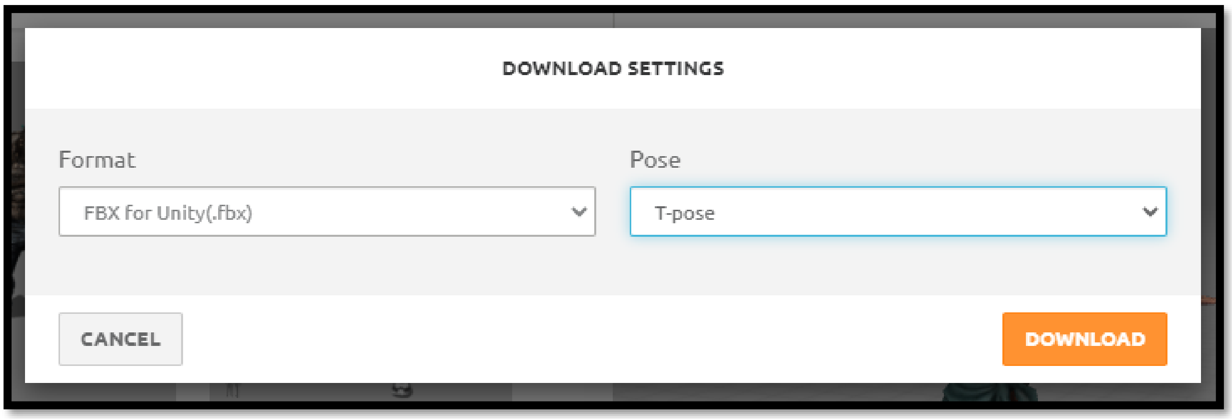
Aim: Implement animation layers in Unity.

Step 1: Firstly, add a Plane for our character to stand on.


04_Practical _07 - SampleScene - Windows, Mac, Linux - Unity 2022.3.10f1* <DX11>
File Edit Assets GameObject Component Services Window Help
Hierarchy
SampleScene*
Main Camera
Directional Light
Plane
Scene Game Asset Store Animator
Game Display 1 Free Aspect Scale 1.3x Play Focused Stats Gizmos
Project Console
Favorites
All Materials
All Models
All Prefabs
Assets
Material
Scenes
Assets > Material
Green

Step 2: Download character of your choice from Mixamo.



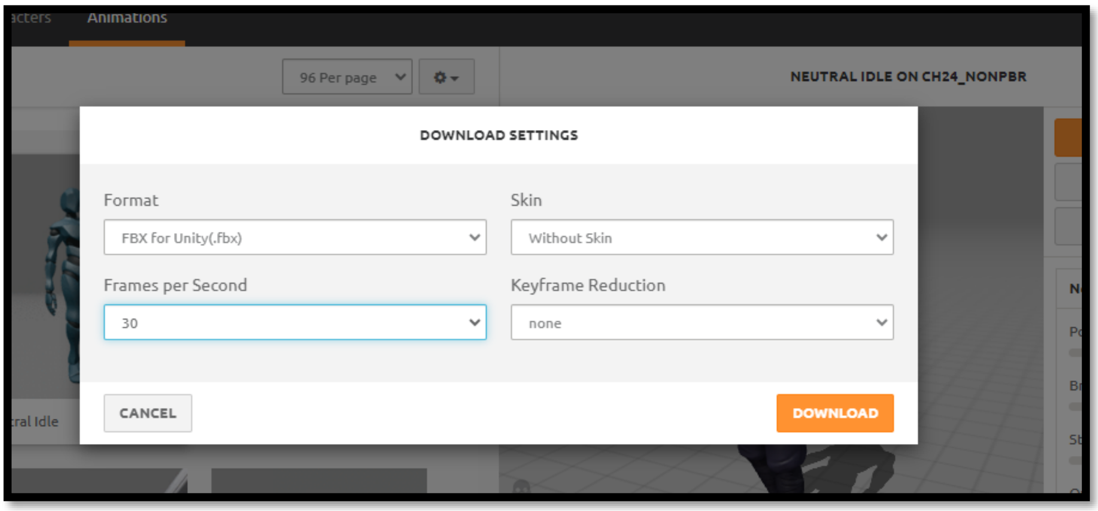

DOWNLOAD SETTINGS

Format: FBX for Unity(.fbx)

Pose: T-pose

Step 3: Download Animations Idle, Fast run, Injured Idle, Injured Run without skin and add it in asset folder create a folder Mixamo and import the character and animations in your mixamo folder

Idle Animation


NEUTRAL IDLE ON CH24_NONPBR

DOWNLOAD SETTINGS

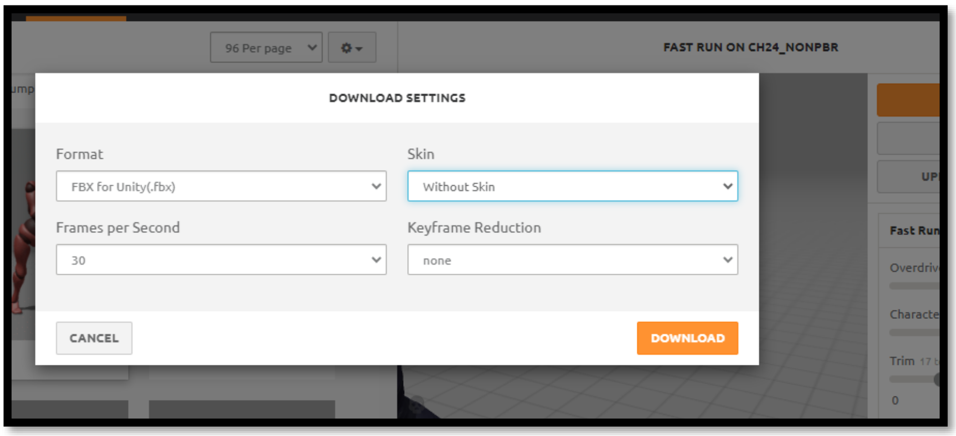
Format: FBX for Unity(.fbx)

Skin: Without Skin

Frames per Second: 30

Keyframe Reduction: none

Fast Run Animation


FAST RUN ON CH24_NONPBR

DOWNLOAD SETTINGS

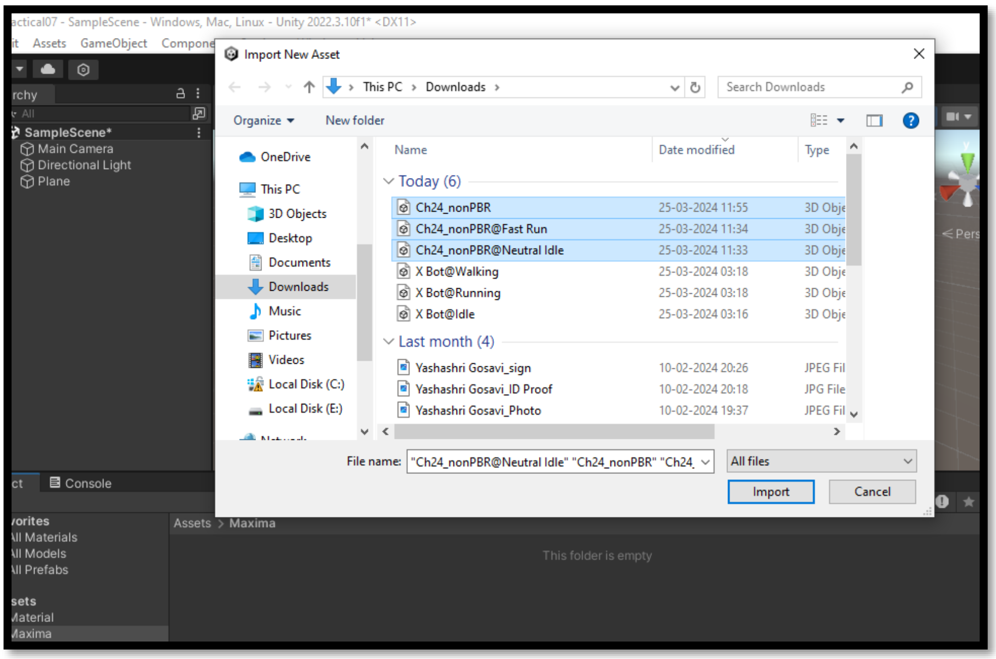
Format: FBX for Unity(.fbx)

Skin: Without Skin

Frames per Second: 30

Keyframe Reduction: none

Import the asset.in Maxima Folder.

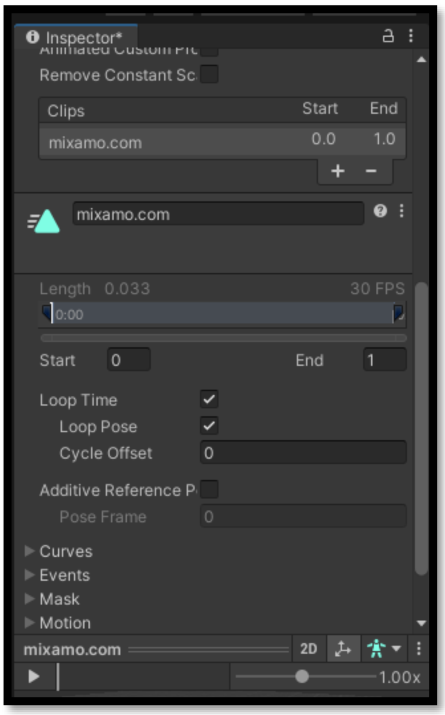


Name	Date modified	Type
Today (6)		
Ch24_nonPBR	25-03-2024 11:55	3D Obje
Ch24_nonPBR@Fast Run	25-03-2024 11:34	3D Obje
Ch24_nonPBR@Neutral Idle	25-03-2024 11:33	3D Obje
X Bot@Walking	25-03-2024 03:18	3D Obje
X Bot@Running	25-03-2024 03:18	3D Obje
X Bot@Idle	25-03-2024 03:16	3D Obje
Last month (4)		
Yashashri Gosavi_sign	10-02-2024 20:26	JPEG Fil
Yashashri Gosavi_ID Proof	10-02-2024 20:18	JPG File
Yashashri Gosavi_Photo	10-02-2024 19:37	JPEG Fil

File name: "Ch24_nonPBR@Neutral Idle" "Ch24_nonPBR" "Ch24_

All files

Step 4: Select Animation from the asset and Enable loop time & loop pose for all the animations.



Remove Constant Sc

Clips	Start	End
mixamo.com	0.0	1.0

mixamo.com

Length 0.033 30 FPS

Start 0 End 1

Loop Time ✓

Loop Pose ✓

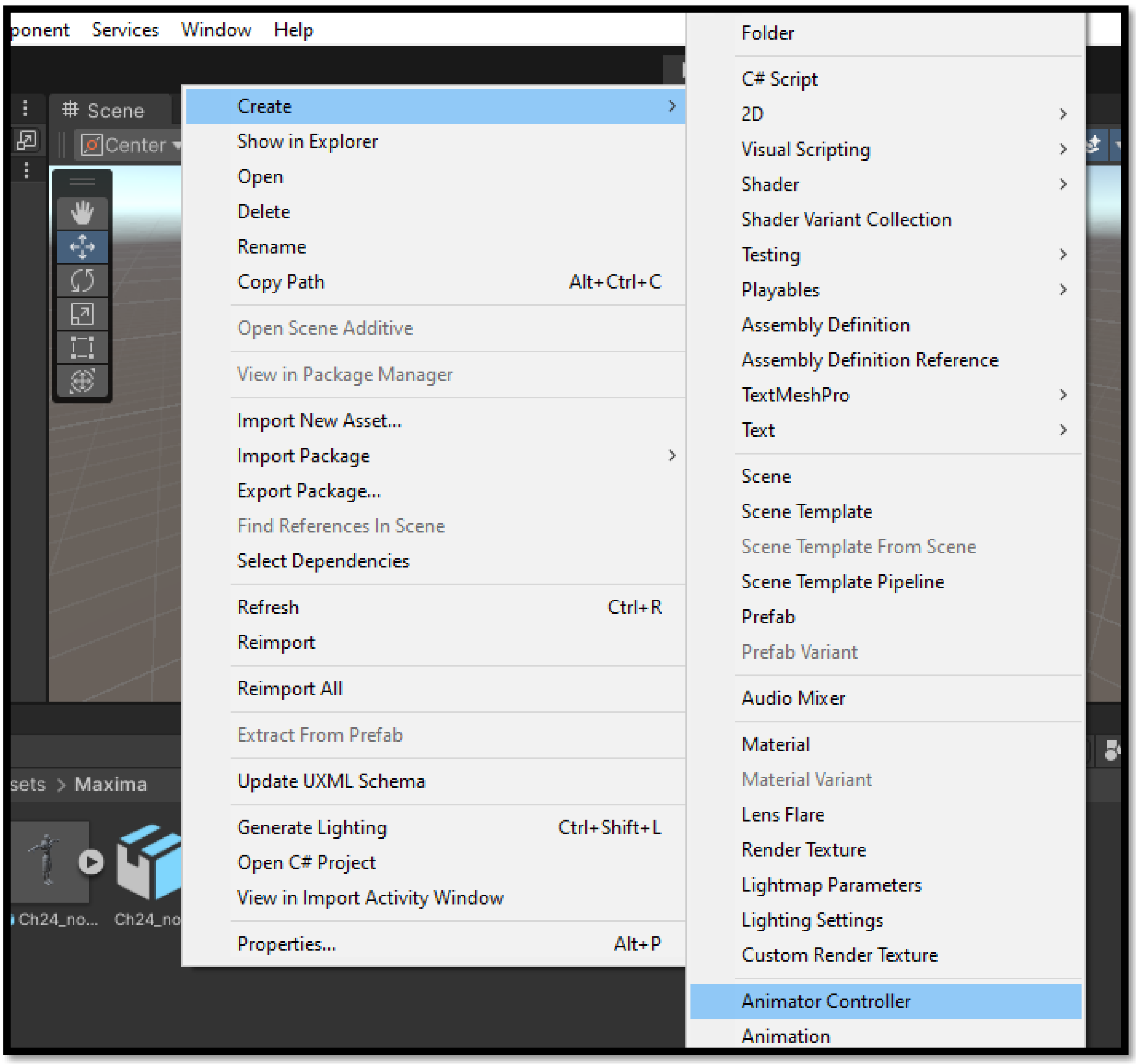
Cycle Offset 0

Additive Reference P

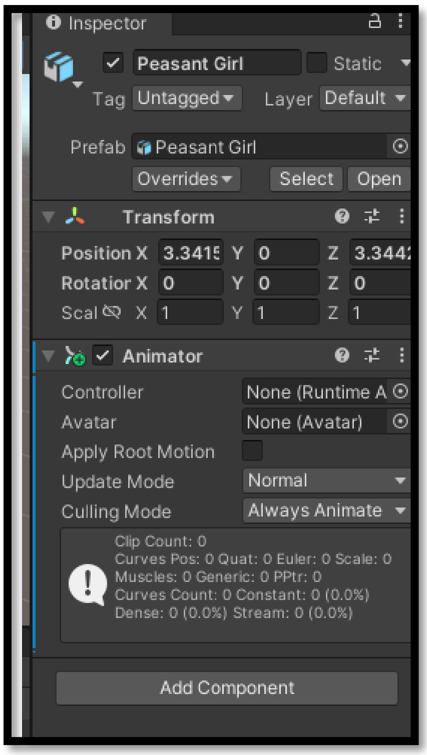
Pose Frame 0

mixamo.com

Step 5: Create and add an Animator Controller to your character [Right-click on Asset folder ➞ click on Create ➞ Animator Controller ➞ Rename it accordingly and add it to your character].



Add Component Animator



Inspector

Peasant Girl | Static

Tag Untagged | Layer Default

Prefab Peasant Girl

Overrides | Select | Open

Transform

Position X 3.3415 Y 0 Z 3.3442

Rotation X 0 Y 0 Z 0

Scale X 1 Y 1 Z 1

Animator

Controller None (Runtime A

Avatar None (Avatar)

Apply Root Motion

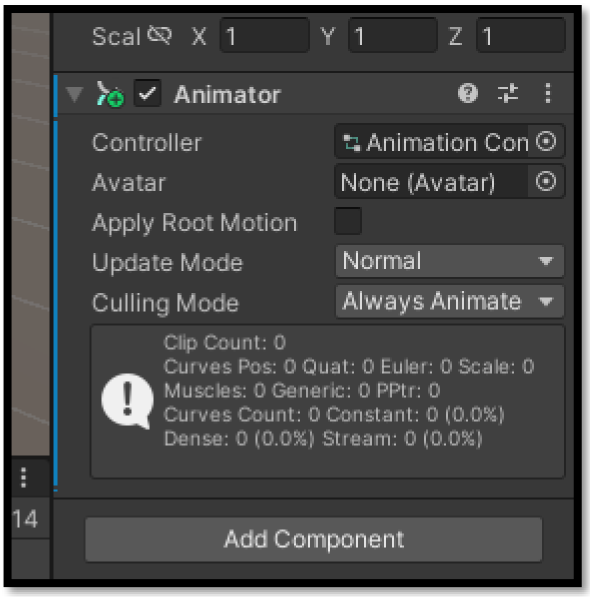
Update Mode Normal

Culling Mode Always Animate

```
Clip Count: 0
Curves Pos: 0 Quat: 0 Euler: 0 Scale: 0
Muscles: 0 Generic: 0 PPtr: 0
Curves Count: 0 Constant: 0 (0.0%)
Dense: 0 (0.0%) Stream: 0 (0.0%)
```

Add Component

Drag and drop animation controller in animator.



Scale X 1 Y 1 Z 1

Animator

Controller Animation Con

Avatar None (Avatar)

Apply Root Motion

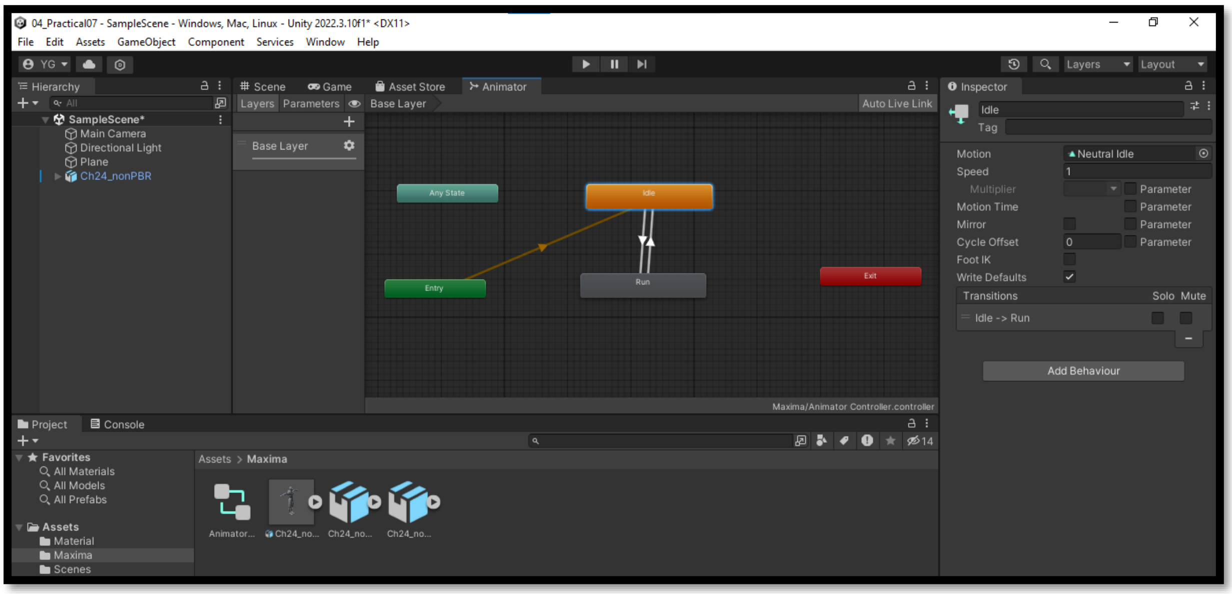
Update Mode Normal

Culling Mode Always Animate

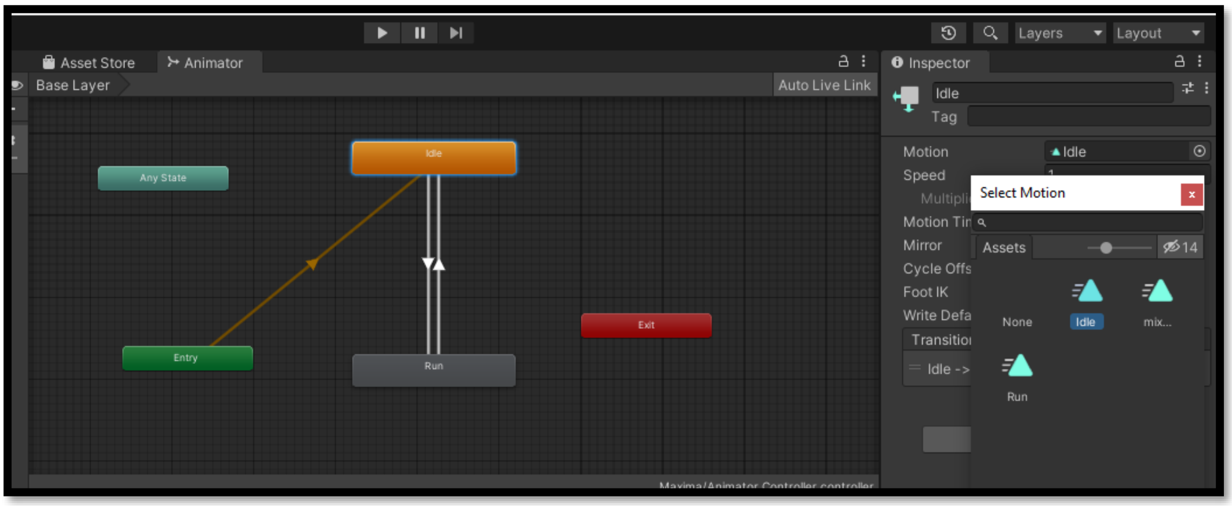
```
Clip Count: 0
Curves Pos: 0 Quat: 0 Euler: 0 Scale: 0
Muscles: 0 Generic: 0 PPtr: 0
Curves Count: 0 Constant: 0 (0.0%)
Dense: 0 (0.0%) Stream: 0 (0.0%)
```

Add Component

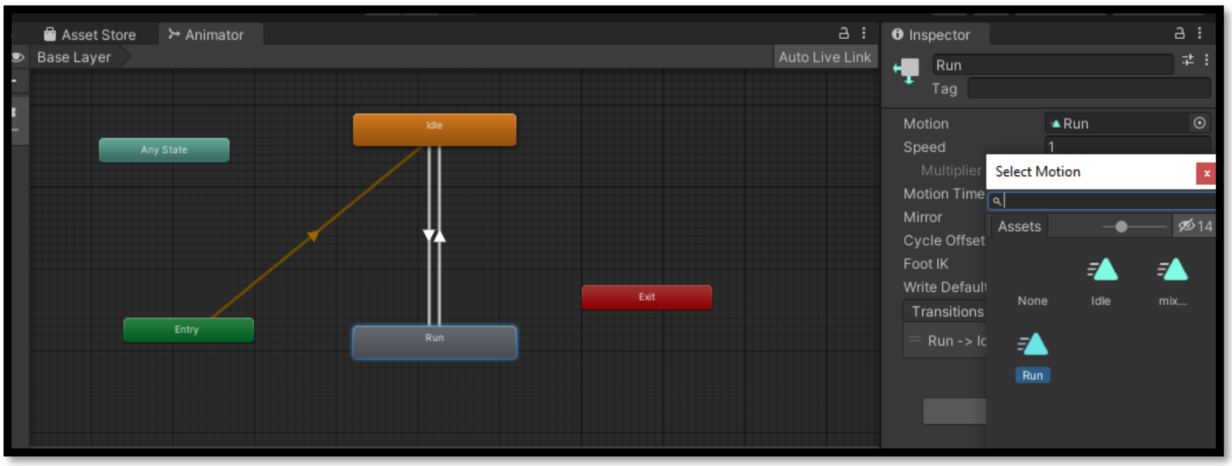
Step 6: Click on Animator Window ➞ Right click and Click on New State.



Idle State 👇

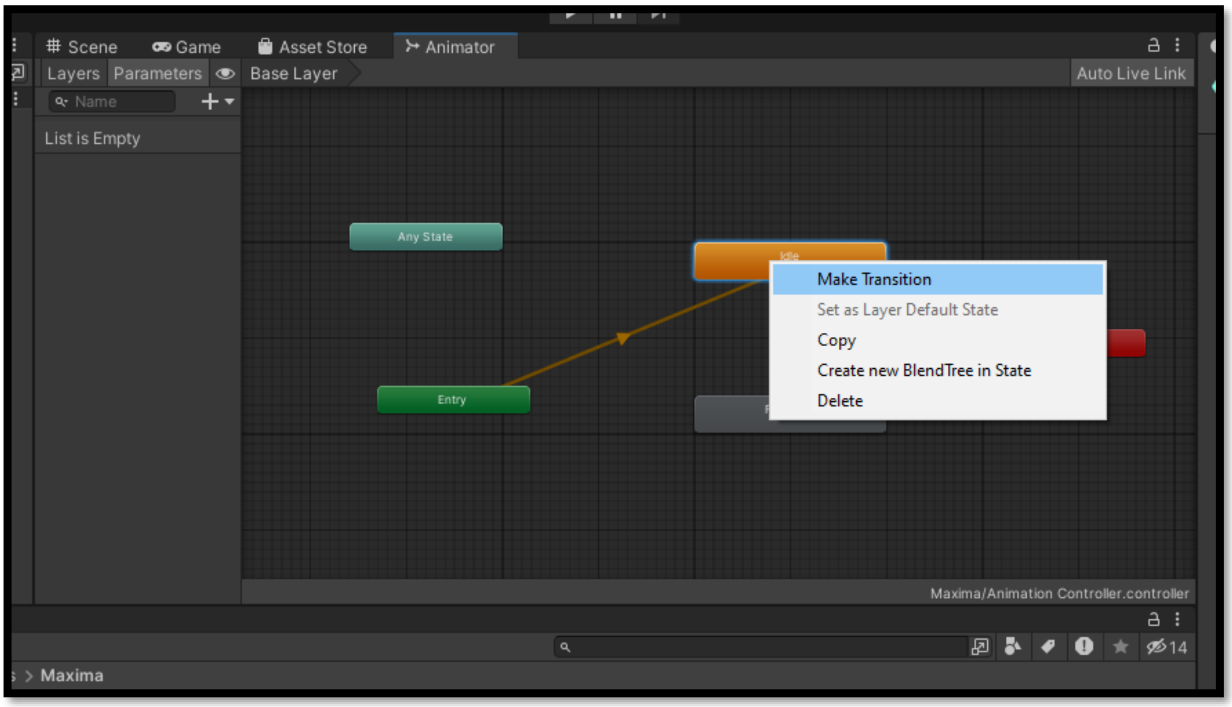


Run State 👇

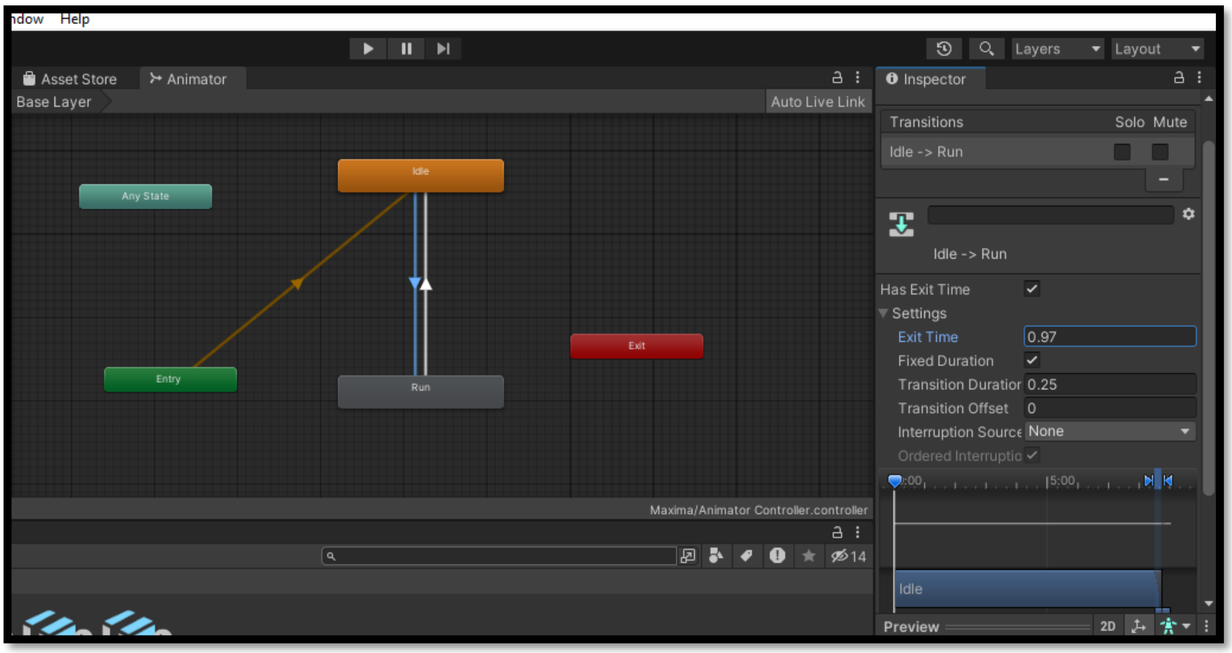


Step 7: Now add a transition from Idle state to Run state

Right-click on Idle state ➞ Click on make Transition and connect it to Run state and similarly Make Transition from Run State to Idle State.



Step 8: Now click on Transition from Idle state to Run state, and let the checkbox be enabled for Has Exit Time [Exit time 0.97 indicates that the transition to the next state will occur when the current animation has played up to 97% of its duration].



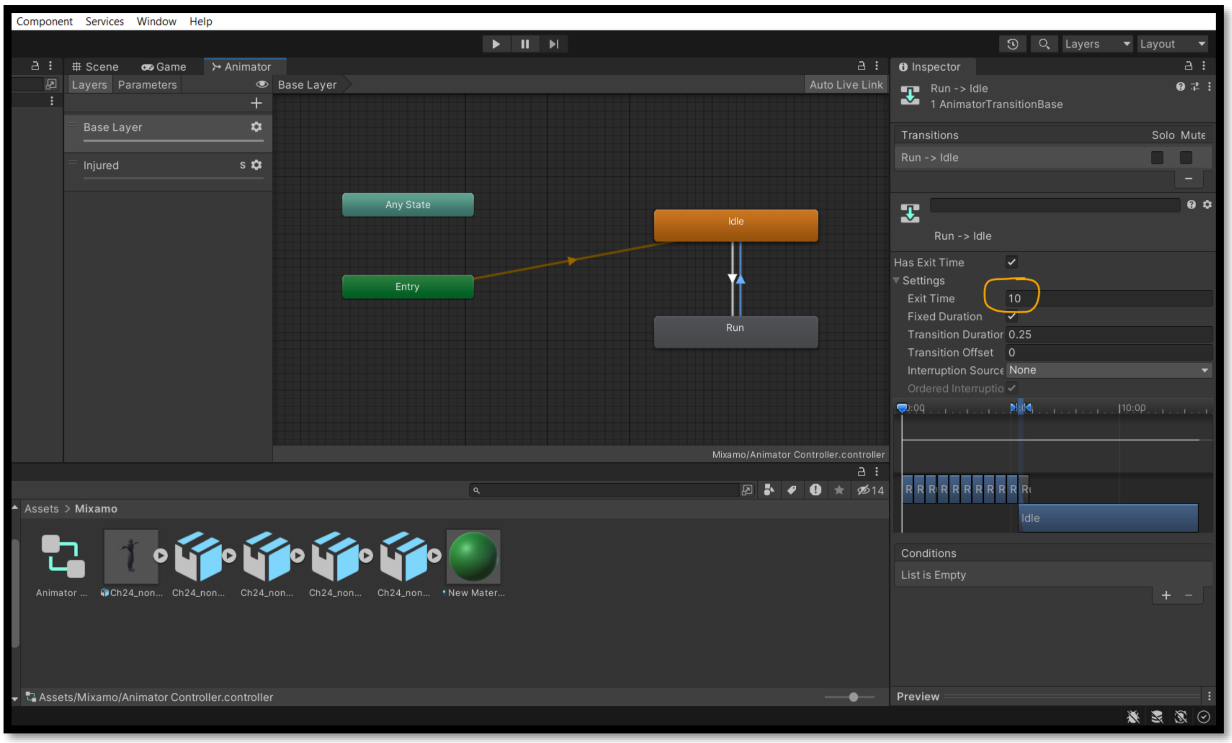
Transitions Solo Mute
Idle -> Run

Idle -> Run

Has Exit Time ✓
Settings
Exit Time 0.97
Fixed Duration ✓
Transition Duratior 0.25
Transition Offset 0
Interruption Source None
Ordered Interruptio ✓

0:00 15:00
Idle

Step 9: Adding animation layer for Injured animations.



Has Exit Time ✓

Settings

Exit Time 10

Fixed Duration ✓

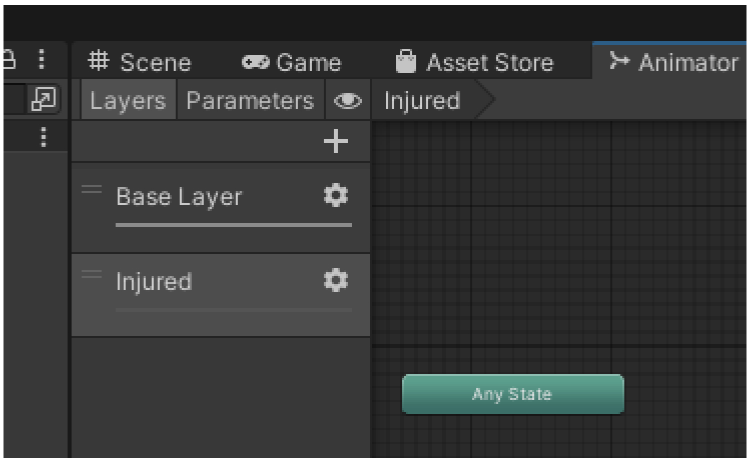
Transition Duration 0.25

Transition Offset 0

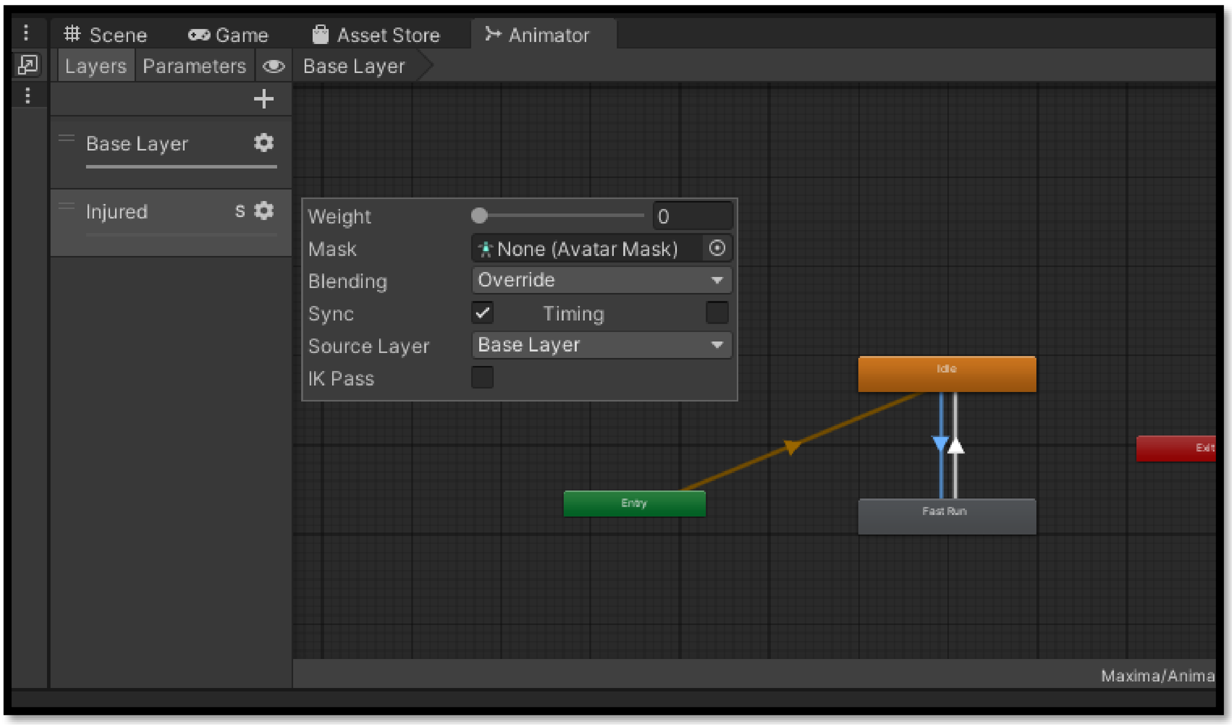
Interruption Source None

Ordered Interruption ✓

Step 10: In layers tab below base layer add a layer, rename this layer as Injured.

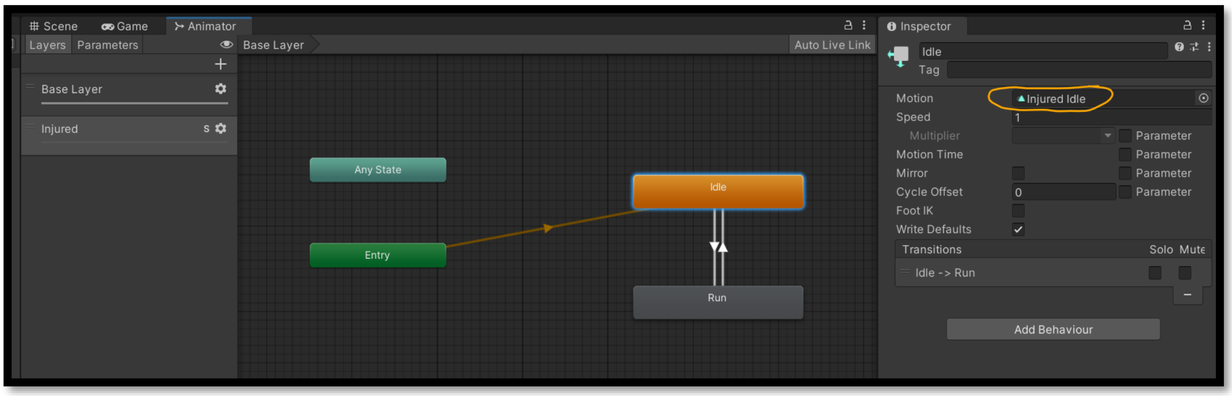


Step 11: Click on the setting icon and enable sync, this will sync the base layer states with the newly created layer Injured

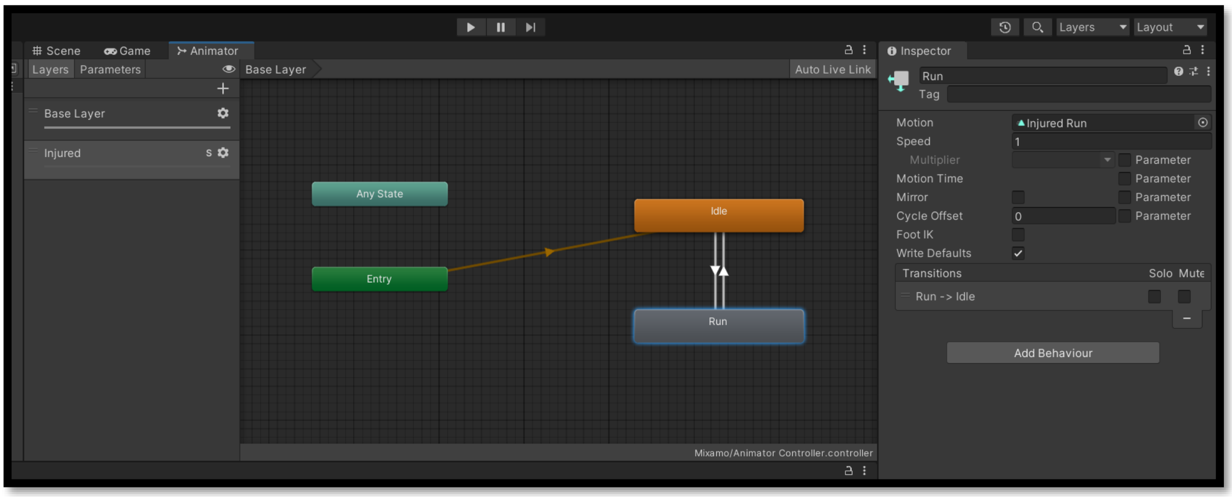


Weight 0
Mask None (Avatar Mask)
Blending Override
Sync ✓ Timing
Source Layer Base Layer
IK Pass

Step 12: Select Idle state from Injured Layer and in motion add Injured Idle animation.

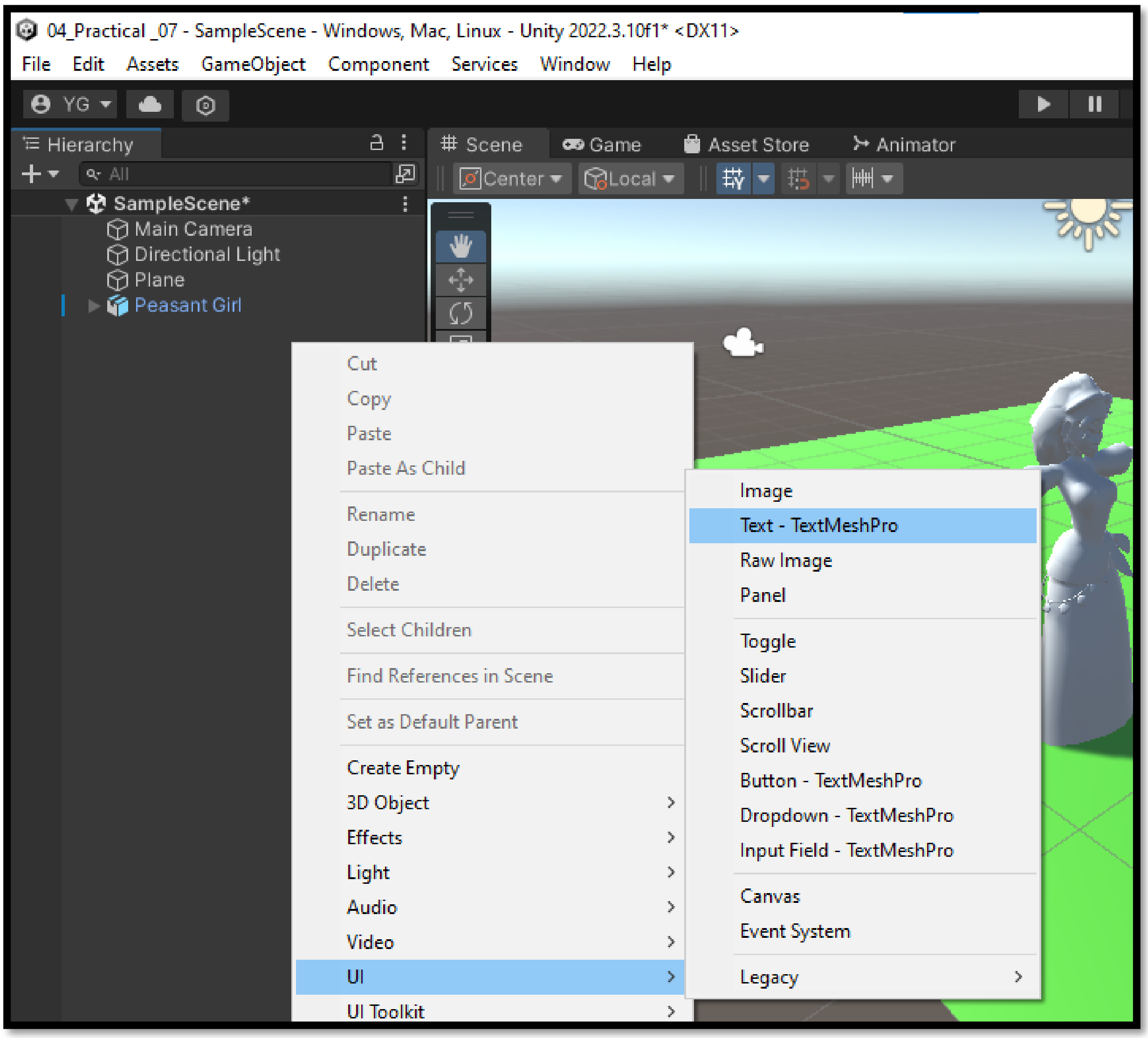


Idle
Tag
Motion Injured Idle
Speed 1
Multiplier Parameter
Motion Time Parameter
Mirror Parameter
Cycle Offset 0 Parameter
Foot IK
Write Defaults ✓
Transitions Solo Mute
Idle -> Run

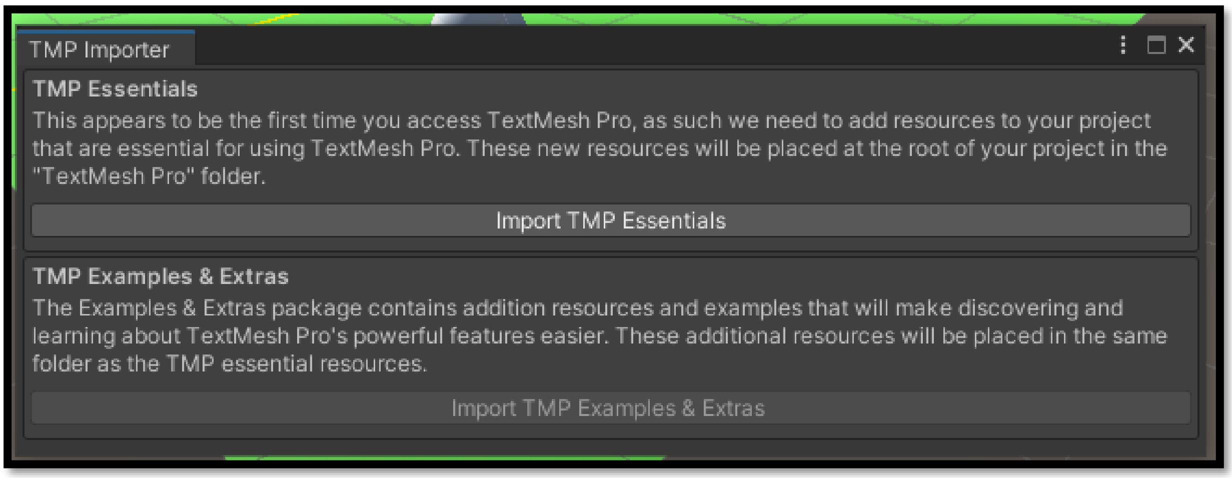


Run
Tag
Motion Injured Run
Speed 1
Multiplier Parameter
Motion Time Parameter
Mirror Parameter
Cycle Offset 0 Parameter
Foot IK
Write Defaults ✓
Transitions Solo Mute
Run -> Idle

Step 13: Adding UI to display the health of the player.



Click on the import


TMP Importer

TMP Essentials

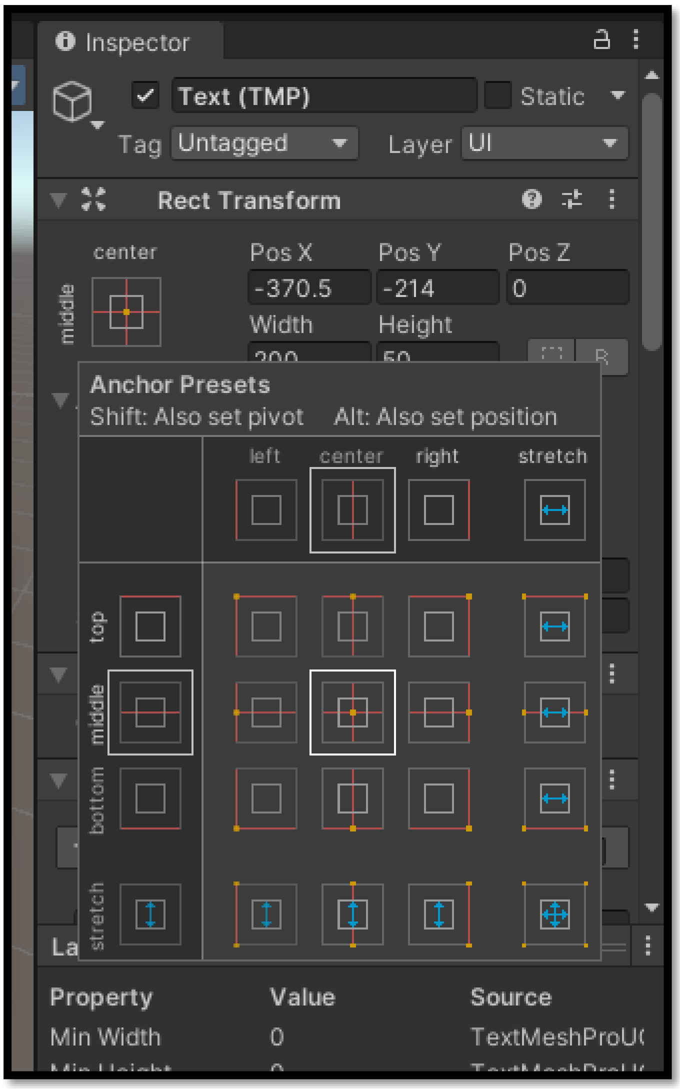
This appears to be the first time you access TextMesh Pro, as such we need to add resources to your project that are essential for using TextMesh Pro. These new resources will be placed at the root of your project in the "TextMesh Pro" folder.

Import TMP Essentials

TMP Examples & Extras

The Examples & Extras package contains addition resources and examples that will make discovering and learning about TextMesh Pro's powerful features easier. These additional resources will be placed in the same folder as the TMP essential resources.

Import TMP Examples & Extras


Inspector

Text (TMP) Static

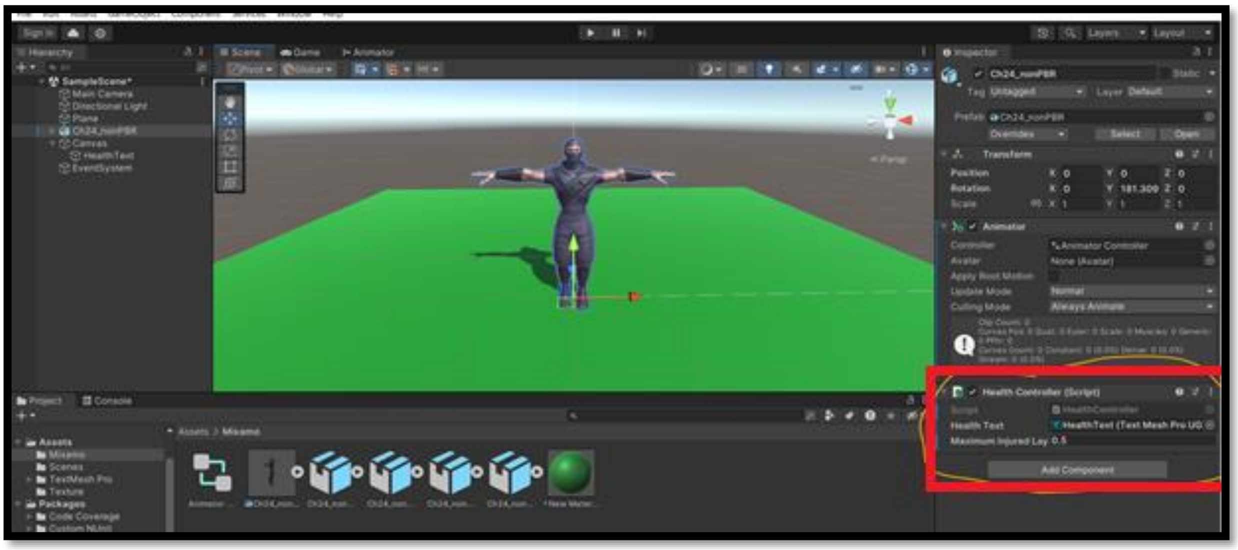
Tag Untagged Layer UI

Rect Transform

center	Pos X	Pos Y	Pos Z
middle	-370.5	-214	0
	Width	Height	
	200	50	

Property	Value	Source
Min Width	0	TextMeshProU(
Min Height	0	TextMeshProU(

Step 14: Select your character and click on Add component ➞ add script ➞ remane the script to Health Controller.


Ch24_nonPBR

Tag Untagged Layer Default

Transform

	X	Y	Z
Position	0	0	0
Rotation	0	181.309	0
Scale	1	1	1

Animator

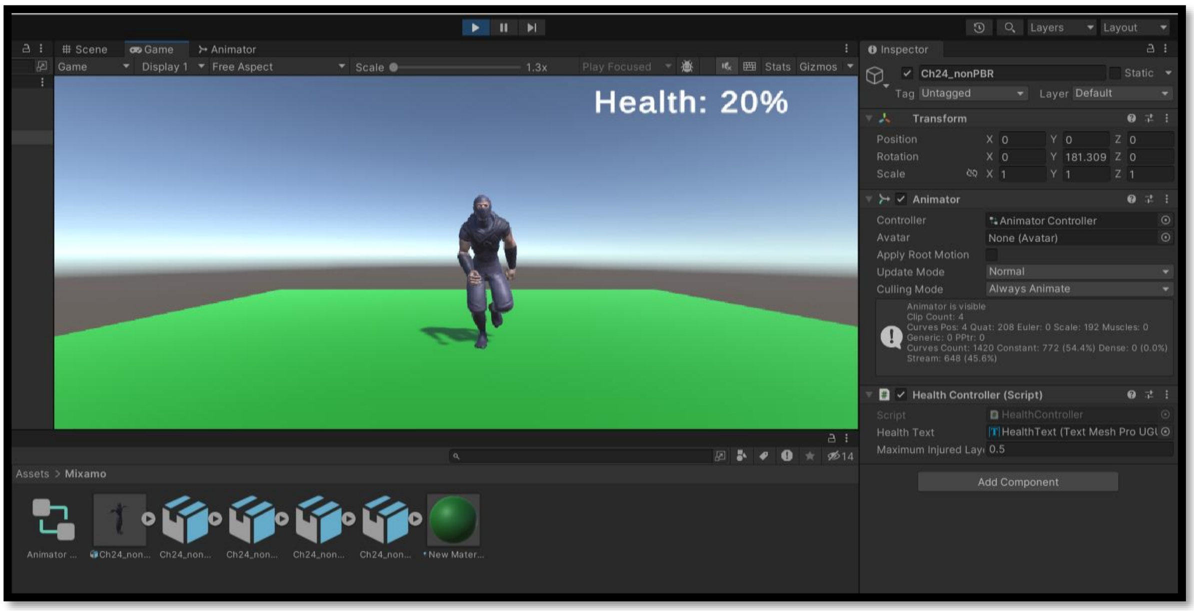
Controller: Animator Controller
Avatar: None (Avatar)
Update Mode: Normal
Culling Mode: Always Animate

Health Controller (Script)

Script: HealthController
Health Text: HealthText (Text Mesh Pro UGUI)
Maximum Injured Lay 0.5

Step 15: Double click on script and add the below code to drop health of character once space bar is clicked.

Code:
```
using System.Collections;
using System.Collections.Generic;
using TMPro;
using UnityEngine;

public class HealthController : MonoBehaviour
{
    // Serialized field allows private variables to be exposed in the Unity Editor for easy
tweaking.
    [SerializeField]
    private TextMeshProUGUI healthText; // Reference to UI text for displaying health
information

    [SerializeField]
    private float maximumInjuredLayerWeight; // Maximum weight for the "Injured"
animation layer

    private float maximumHealth = 100; // Maximum health value
    private float currentHealth; // Current health value

    private Animator animator; // Reference to the Animator component for animation control
    private int injuredLayerIndex; // Index of the "Injured" animation layer
    private float layerWeightVelocity; // Velocity for smooth animation transitions

    // Start is called before the first frame update
    void Start()
    {
        // Initialize current health to maximum health
        currentHealth = maximumHealth;

        // Get reference to the Animator component attached to this game object
        animator = GetComponent<Animator>();

        // Get the index of the "Injured" animation layer in the Animator
        injuredLayerIndex = animator.GetLayerIndex("Injured");
    }

    // Update is called once per frame
    void Update()
    {
        // Check if the spacebar is pressed
        if (Input.GetKeyDown(KeyCode.Space))
        {
            // Reduce current health by 10% of maximum health
            currentHealth -= maximumHealth / 10;
        }
        // Ensure current health doesn't go below 0
        if (currentHealth < 0)
        {
            currentHealth = maximumHealth; // Reset health to maximum if it's below 0
        }

        // Calculate health percentage
        float healthPercentage = currentHealth / maximumHealth;
        // Update UI text to display health percentage
        healthText.text = $"Health: {healthPercentage * 100}%";
        // Get the current weight of the "Injured" animation layer
        float currentInjuredLayerWeight = animator.GetLayerWeight(injuredLayerIndex);

        // Calculate the target weight based on health percentage and maximum injured layer
weight
        float targetInjuredLayerWeight = (1 - healthPercentage) *
maximumInjuredLayerWeight;

        // Smoothly transition the weight of the "Injured" animation layer
        animator.SetLayerWeight(
            injuredLayerIndex,
            Mathf.SmoothDamp(
                currentInjuredLayerWeight,
                targetInjuredLayerWeight,
                ref layerWeightVelocity,
                0.2f // Smoothing duration
            )
        );
    }
}
```
Output:



Ch24_nonPBR
Tag Untagged Layer Default

Transform
Position X 0 Y 0 Z 0
Rotation X 0 Y 181.309 Z 0
Scale X 1 Y 1 Z 1

Animator
Controller: Animator Controller
Avatar: None (Avatar)
Apply Root Motion
Update Mode: Normal
Culling Mode: Always Animate

```
Animator is visible
Clip Count: 4
Curves Pos: 4 Quat: 208 Euler: 0 Scale: 192 Muscles: 0
Generic: 0 PPtr: 0
Curves Count: 1420 Constant: 772 (54.4%) Dense: 0 (0.0%)
Stream: 648 (45.6%)
```

Health Controller (Script)
Script: HealthController
Health Text: HealthText (Text Mesh Pro UGU
Maximum Injured Lay 0.5