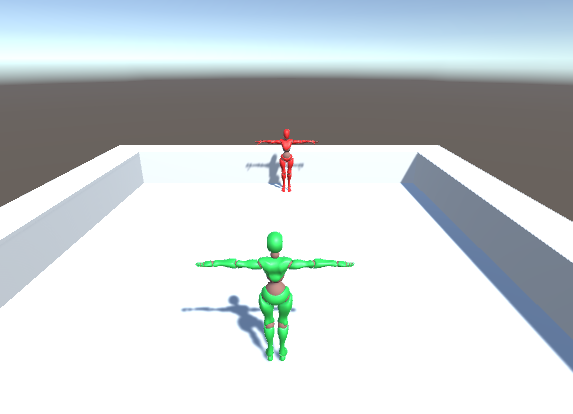
**Lab 5 : Setting Up Animations and Game Mechanics**

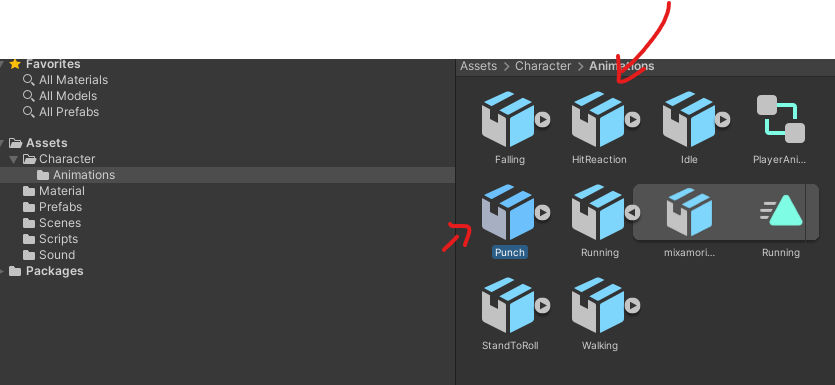
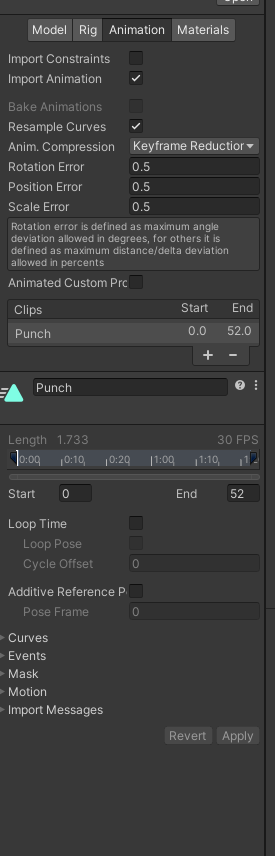
Follow these step-by-step instructions to complete the lab tasks.

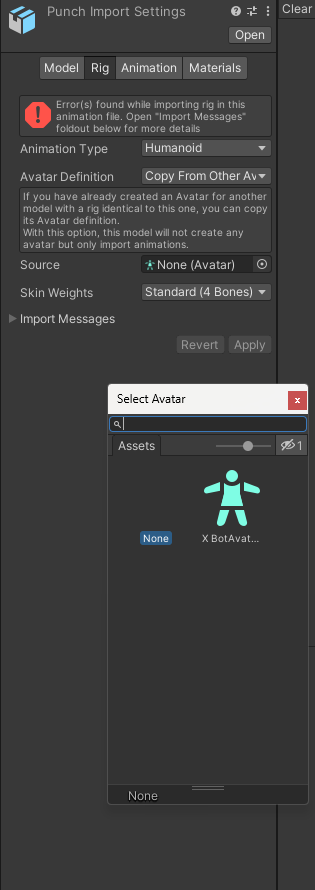
Open the git assignment :

https://classroom.github.com/a/FMtLDcBd



### **Task 1: Setup the Punch and Hit Reaction Animation**

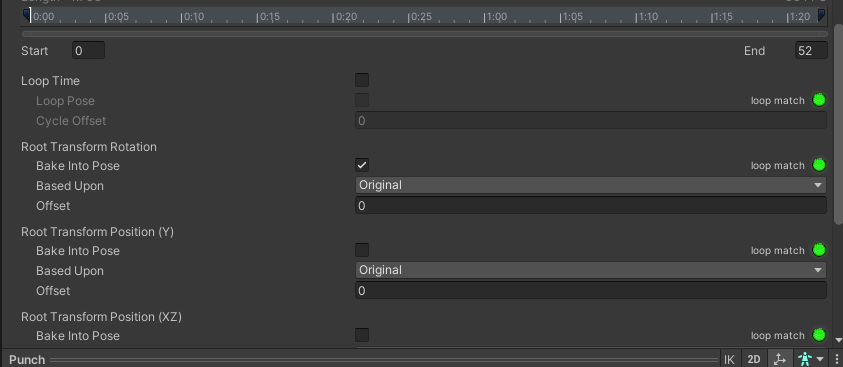
1. **Set Up Animations:**
   * Select each animation file.
   * In the **Inspector**, go to the **Rig** tab and set **Animation Type** to **Humanoid**.
   * Set **Avatar Definition** to **Copy From Other Avatar**.
   * Select the enemy avatar or create one if not existing.Click **Apply**.



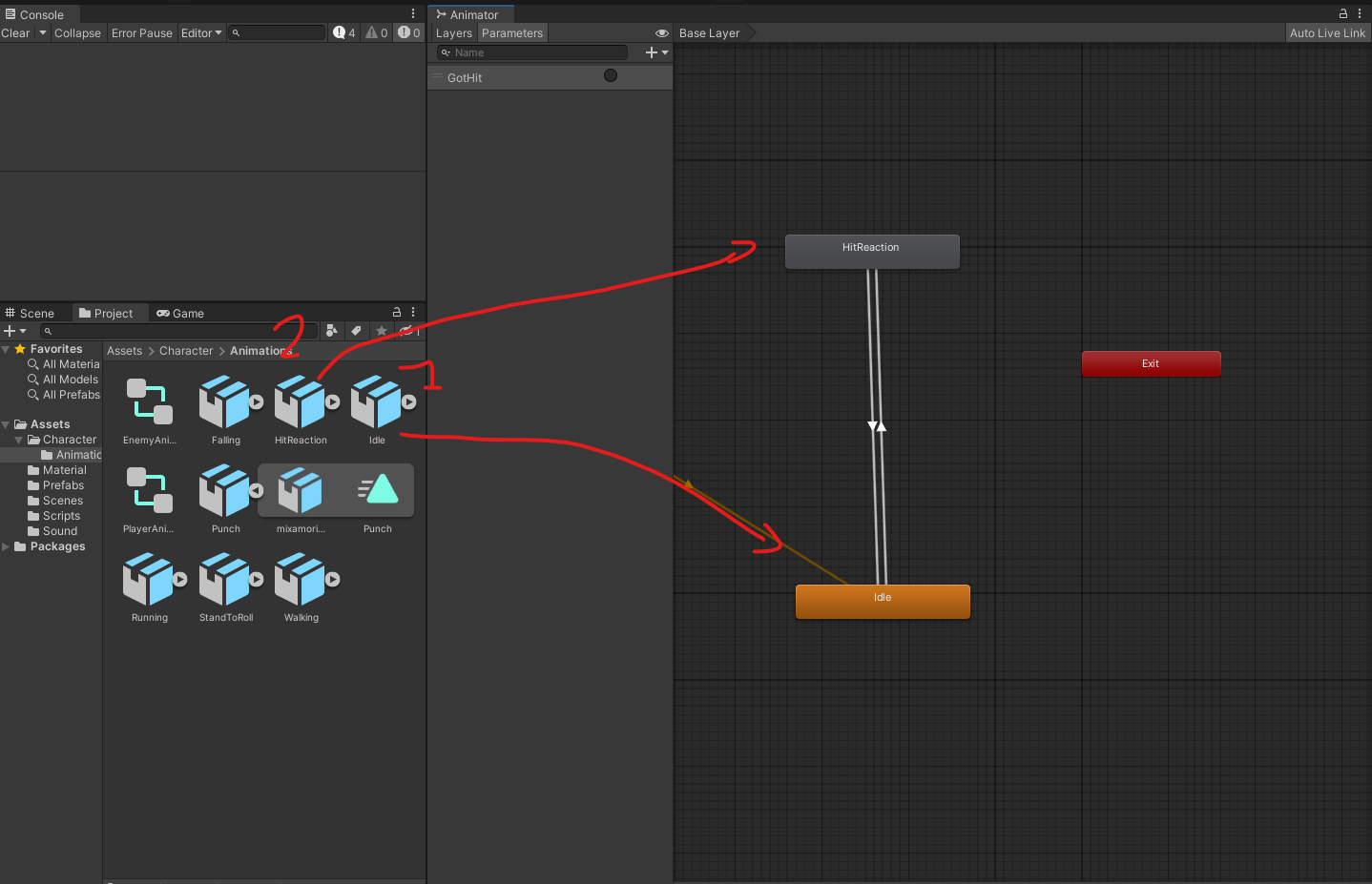
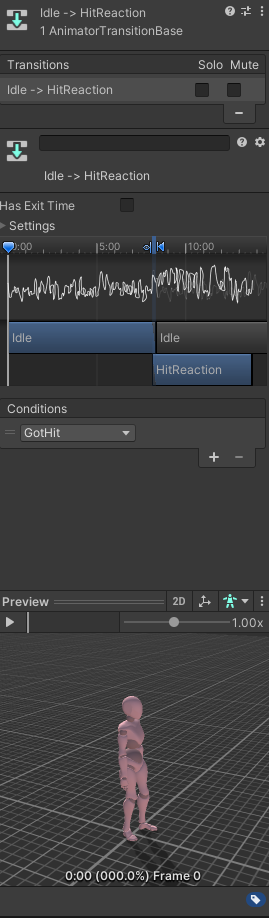
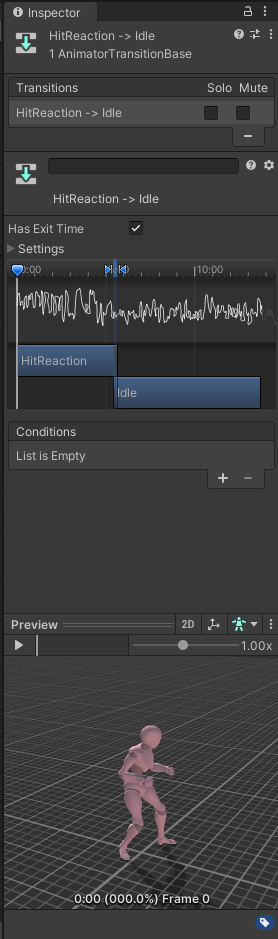
Do the same thing for the HitReaction Animation

Reajust the initial rotation in the Punch animation :

Modify the punch animation to bake into the pose the rotation



### **Task 2: Create an Animator for the Enemy**

1. **Create Animator Controller:**
   * Right-click in the **Project** window and select **Create > Animator Controller**.
   * Name it EnemyAnimator.
2. **Assign Animator to Enemy:**
   * Select your enemy GameObject in the **Hierarchy**.
   * In the **Inspector**, click **Add Component** and choose **Animator**.
   * Drag and drop the EnemyAnimator onto the **Controller** field of the Animator component.
3. **Open Animator Window:**
   * Select EnemyAnimator.
   * Go to **Window > Animator** to open the Animator window.
4. **Add Idle Animation:**
   * Drag the idle animation clip from the **Project** window into the Animator.
   * Set it as the **Default State** by right-clicking and selecting **Set as Layer Default State**.
   * 
5. **Add Hit Reaction Animation:**
   * Drag the hit reaction animation clip into the Animator.
6. **Create Transition with Trigger:**
   * Select the transition from **Idle** to **Hit Reaction**.
   * In the **Inspector**, under **Conditions**, add a new **Trigger** parameter (e.g., GotHit).
   * Set the condition for the transition to the GotHit trigger.
   * 
7. **Set Transition Back to Idle:**
   * Create a transition from **Hit Reaction** back to **Idle**.
   * Ensure there are no conditions for this transition.

### **Task 4: Modify the Player Animator to Add the Attack ( Punch ) Animations**

1. **Open Player Animator Controller:**
   * Select your player's Animator Controller (e.g., PlayerAnimator).
   * Open the Animator window.
2. **Add Attack Animations:**
   * Drag the attack animation clips into the Animator.
3. **Create Parameters:**
   * In the Animator window, click the **Parameters** tab.
   * Add a **Trigger** parameter for the attack (e.g., doPunch)
4. **Set Up Transitions:**
   * Create transitions from the appropriate states (e.g., from **Idle/walk/run**) to the attack animation.
   * Set the condition for each transition to its corresponding trigger.
5. **Transition Back to Idle or Movement:**
   * Create transitions from each attack animation back to **Idle/walk/run** or **Movement**.
   * Ensure there are no conditions for these transitions.
6. Modify the PlayerAnimationManager to add your Trigger :

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class PlayerAnimationManager : MonoBehaviour

{

private Animator animator;

private PlayerMovement movement;

private Rigidbody rb;

public void Start()

{

animator = GetComponent<Animator>();

movement = GetComponent<PlayerMovement>();

rb = GetComponent<Rigidbody>();

}

public void Update()

{

animator.SetFloat("CharacterSpeed", rb.velocity.magnitude);

animator.SetBool("IsFalling",!movement.isGrounded);

if (Input.GetButtonUp("Fire1"))

{

animator.SetTrigger("doRoll");

}

if (Input.GetButtonUp("Fire2"))

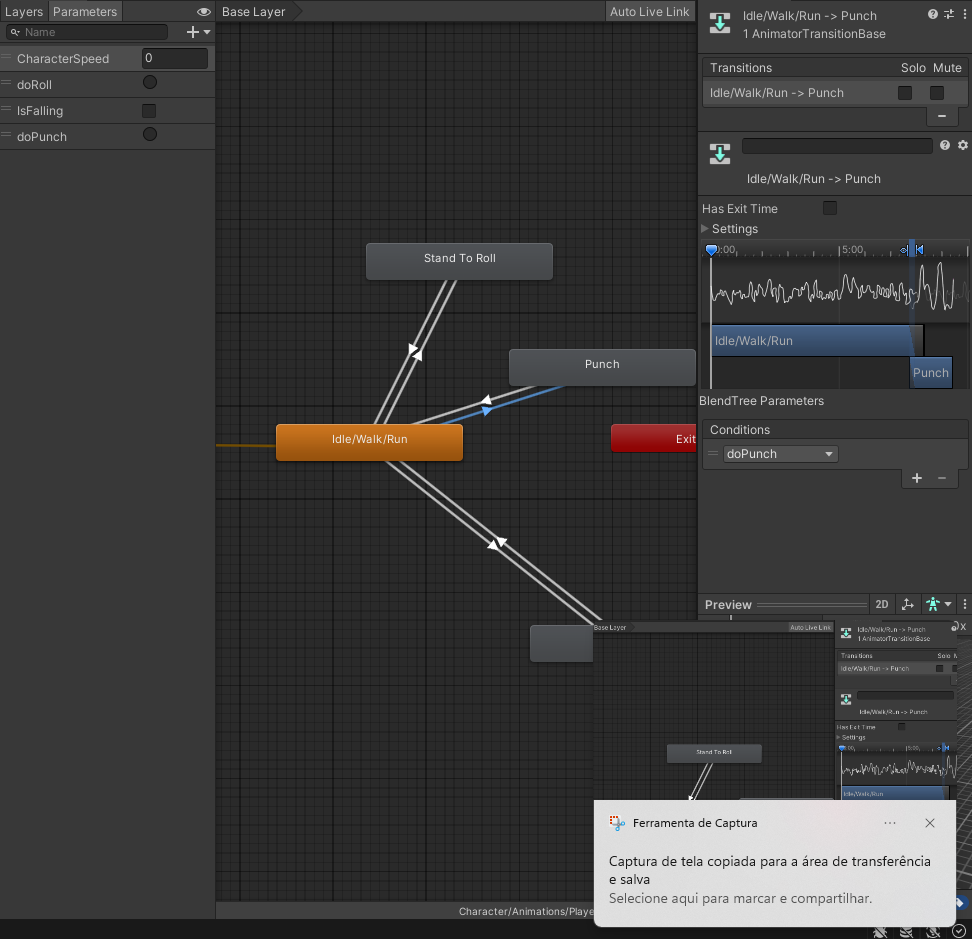
{

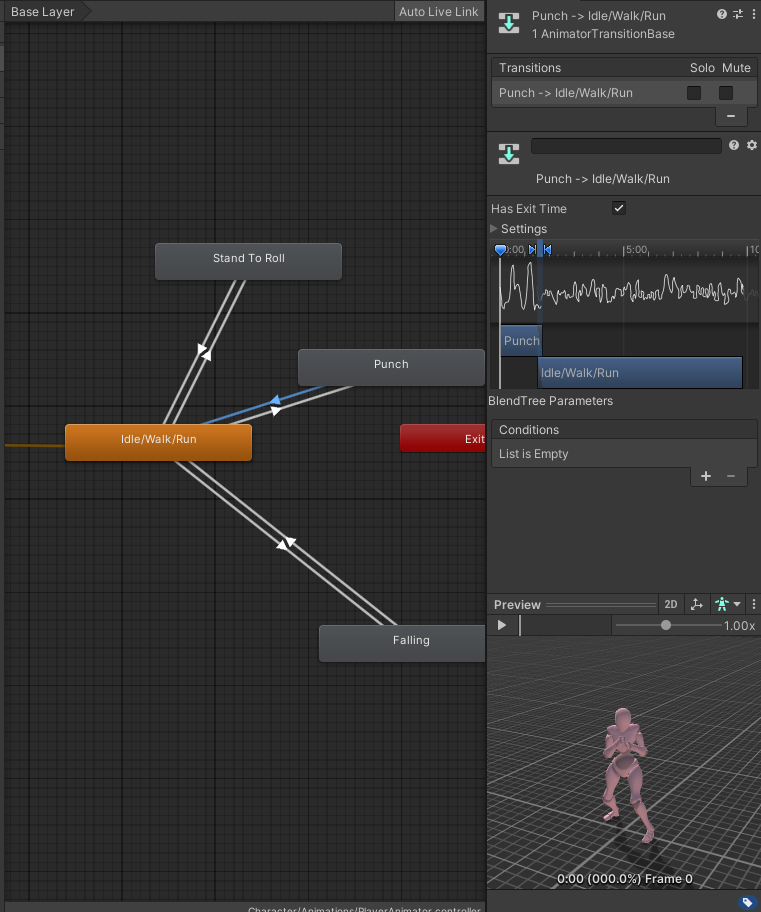
animator.SetTrigger("doPunch");

}

}

}





### **Task 6: Add to the Colliders to the fists**

### **Access the Character's Bone Hierarchy**

1. **Expand the Character's Hierarchy:**
   * In the **Hierarchy** window, expand your character's GameObject by clicking the arrow next to its name.
   * Continue expanding the child objects until you reach the hand bones.
2. **Locate the Hand Bones:**
   * Typically, the path to the hand bones is:
     + **CharacterName**
       - **Hips**
         * **Spine**

**Chest**

**UpperChest** (if available)

**LeftShoulder** / **RightShoulder**

**LeftArm** / **RightArm**

**Left ForeArm/RightForearm**

**LeftHand/RigthHand,**

### **Step 4: Add Colliders to the Hands**

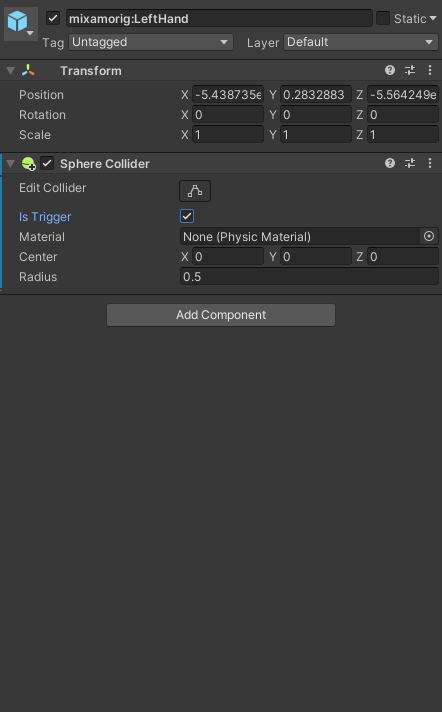
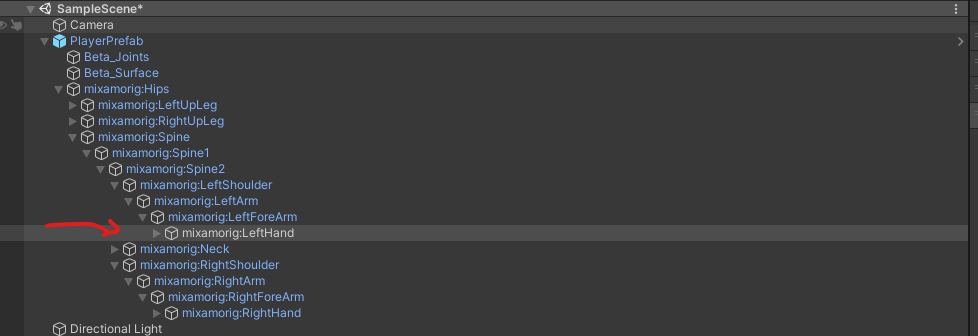
#### **For the Right Hand:**

1. **Select the RightHand Bone:**
   * **Click on the RightHand GameObject in the Hierarchy.**
   * **Note : If you want to add a weapon to a Character you would add it as a child of the RightHand object**
2. **Add a Collider Component:**
   * **In the Inspector, click on Add Component.**
   * **Type Capsule Collider in the search bar and select it.**
3. **Adjust the Collider:**
   * **Center: Adjust the Center values to align the collider with the hand mesh.**
   * **Radius and Height: Modify the Radius and Height to encompass the hand.**
   * **Direction: Set the Direction (X, Y, or Z) based on the orientation of the hand bone.**
4. **Make sure it is a Trigger**

****

#### **For the Left H**

1. **Select the LeftHand Bone:**
   * **Click on the LeftHand GameObject in the Hierarchy.**
2. **Add a Collider Component:**
   * **Repeat the same steps as for the right hand.**
3. **Adjust the Collider:**
   * **Configure the collider to fit the left hand, similar to how you did for the right hand.**



### **4.3 Implement the Hit Detection Logic**

Open the script and write the following code:

using UnityEngine;

public class HitDetector : MonoBehaviour

{

private void OnTriggerEnter(Collider other)

{

/\*if (other.CompareTag("Enemy"))

{

EnemyController enemyController = other.GetComponent<EnemyController>();

if (enemyController != null)

{

enemyController.GotHit();

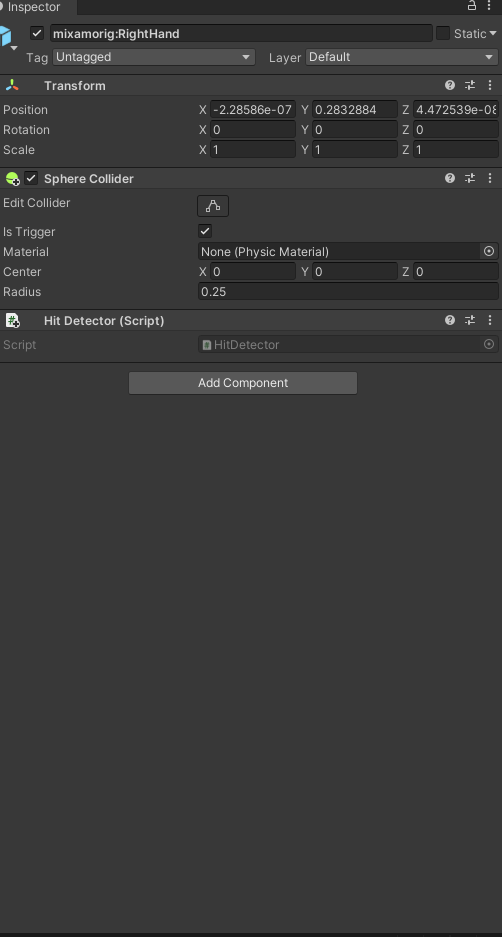
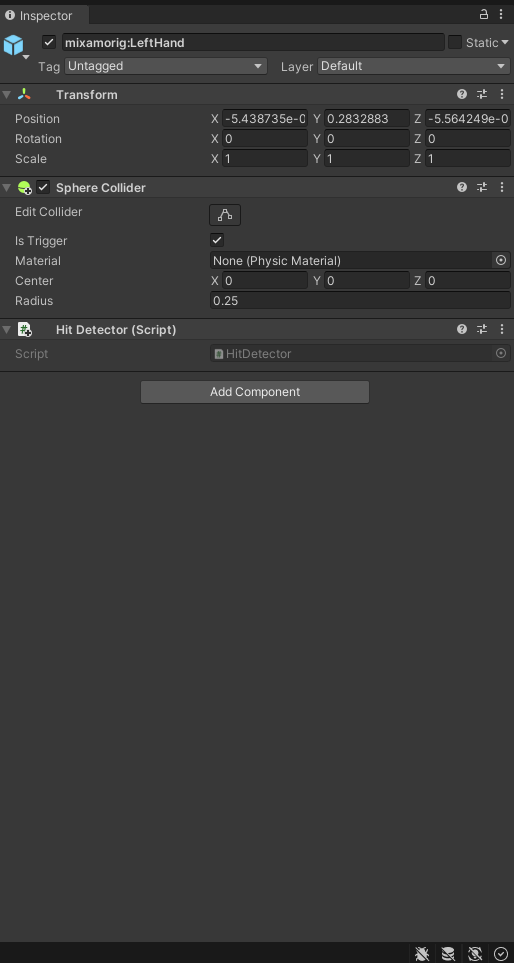
}

}\*/

}

}

Add this script to both fists



## **Step 5: Synchronizing Hit Detection with Animations**

### **5.1 Disabling the Collider by Default**

Create a **PlayerHitboxManagerScript** script, modify the Start method to disable the colliders initially we will want to add this script on the Player :

public Collider[] attackColliders;

void Start()

{

foreach( Collider attackCollider in attackColliders)

{

attackCollider.enabled = false; // Disable collider at start

}

}

### **5.2 Creating Functions to Enable/Disable the Collider**

Add the following methods to the script:

public void EnableHitbox()

{

foreach( Collider attackCollider in attackColliders)

{

attackCollider.enabled = true;

}

}

public void DisableHitbox()

{

foreach( Collider attackCollider in attackColliders)

{

attackCollider.enabled = false;

}

}

The full script should look like this :

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class PlayerHitBoxManager : MonoBehaviour

{

// Start is called before the first frame update

public Collider[] attackColliders;

void Start()

{

foreach( Collider attackCollider in attackColliders)

{

attackCollider.enabled = false; // Disable collider at start

}

}

public void EnableHitbox()

{

foreach( Collider attackCollider in attackColliders)

{

attackCollider.enabled = true;

}

}

public void DisableHitbox()

{

foreach( Collider attackCollider in attackColliders)

{

attackCollider.enabled = false;

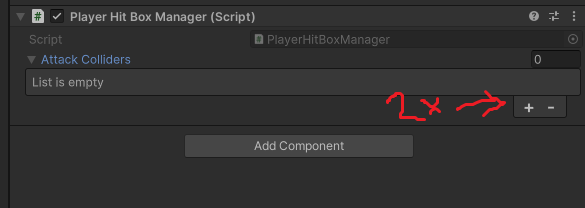
}

}

}

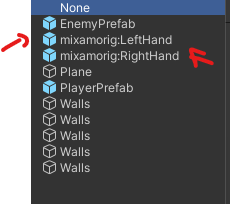
Add the script to the Player

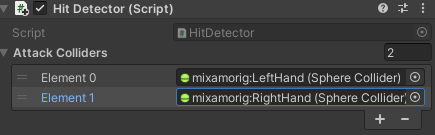




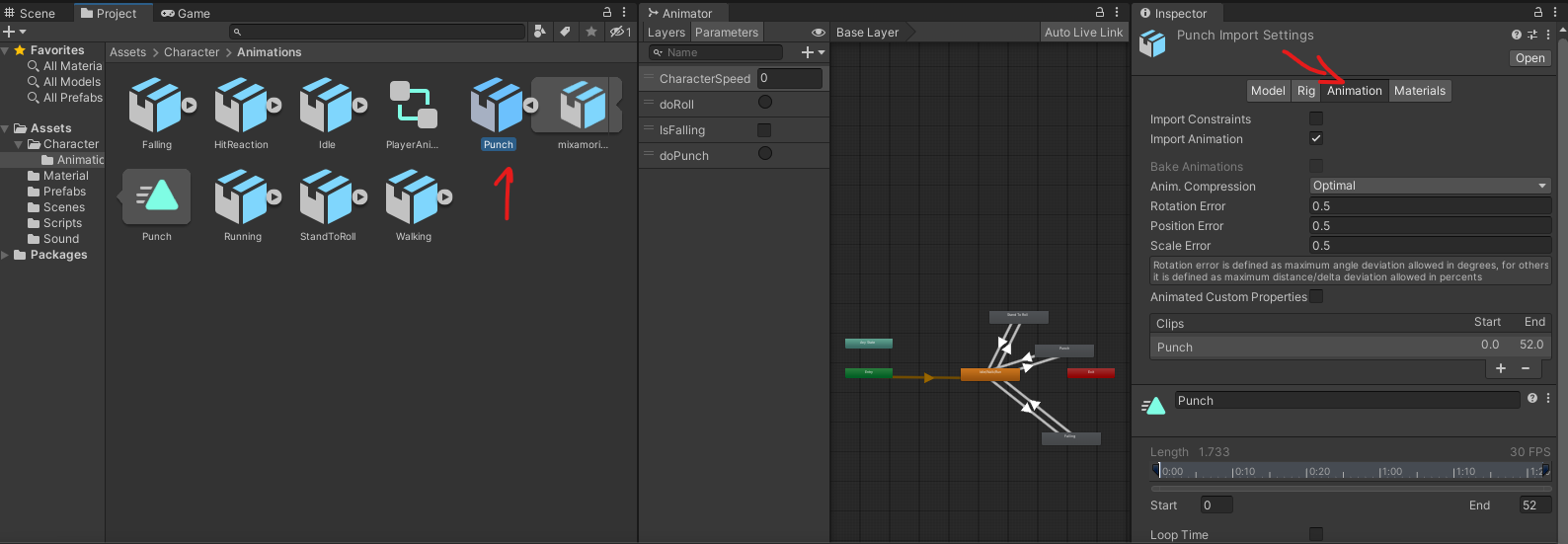
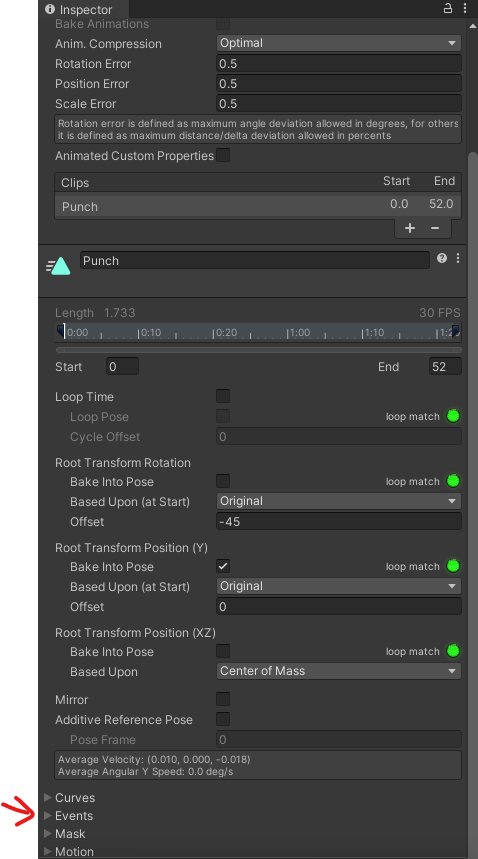
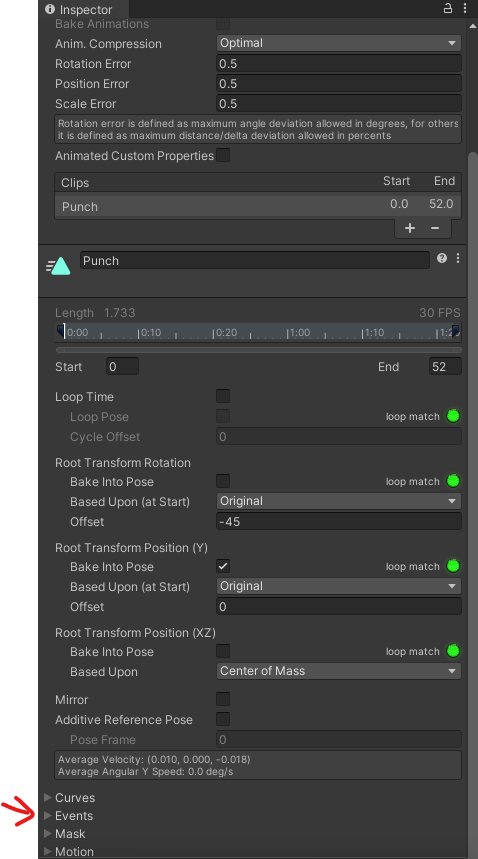
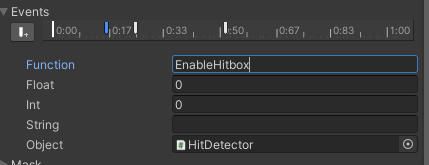
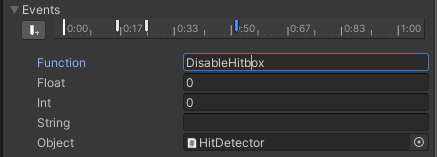
Click the button on the right

Select the two hand colliders you created

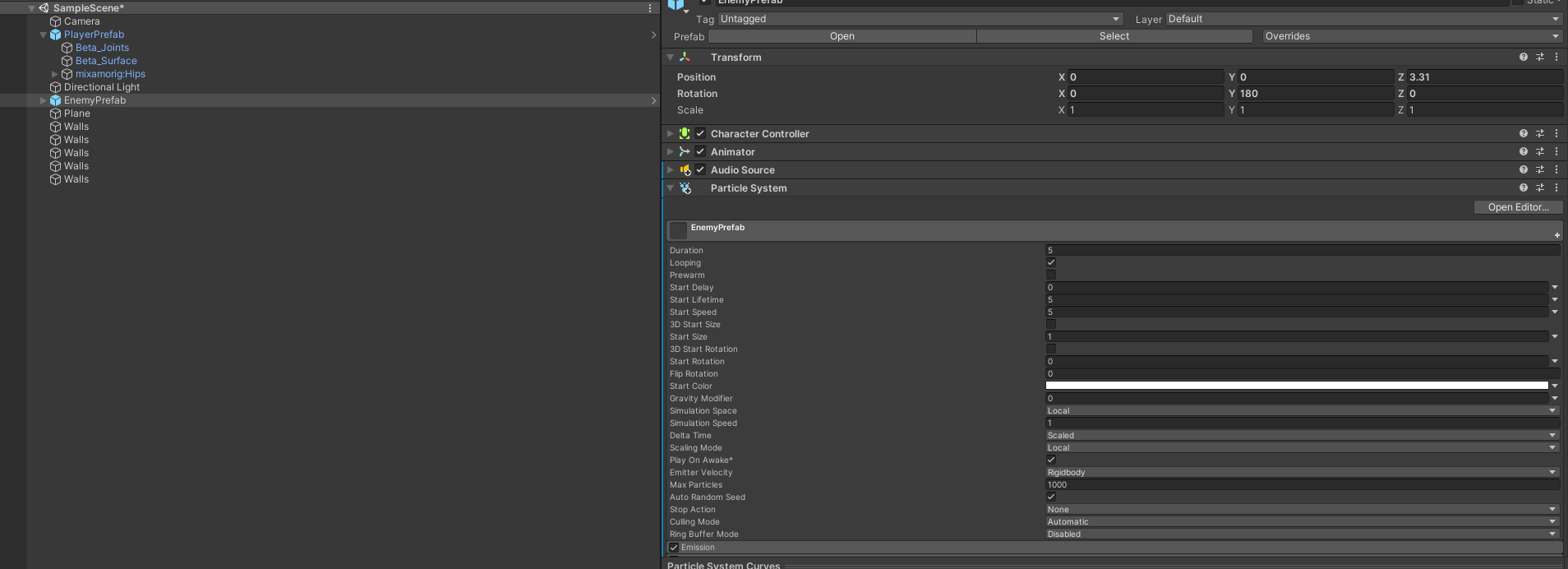
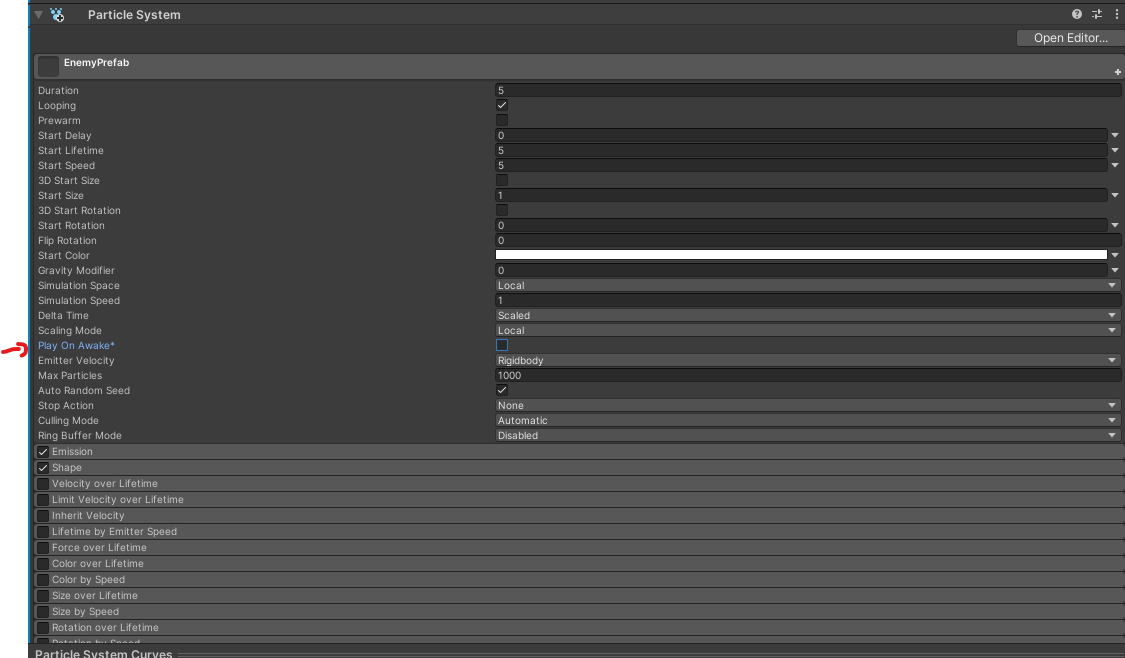
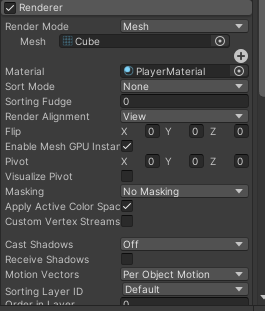




### **5.3 Adding Animation Events**

* Select the Punch animation in the project Hierachy
* In the animation Tab go down to the section called Events
* .
* 
* Click the **Add Event** button (looks like a white marker).
* In the **Inspector**, set the function to EnableHitbox.
* 
* Move to the frame where the attack should stop dealing damage.
* Add another event and set the function to DisableHitbox.
* 

### **Task 7: Add a Particle System Component to the Enemy GameObject**

1. **Select Enemy GameObject:**
   * In the **Hierarchy**, select your enemy GameObject.
2. **Add Particle System:**
   * In the **Inspector**, click **Add Component**.
   * Choose **Particle System**.
3. **Configure Particle System:**
   * Adjust the particle system settings as desired (e.g., shape, emission rate, lifetime).
   * Uncheck **Play On Awake** to prevent it from playing automatically.
   * Set the renderer to cubes and assign it a material  
     

### **Increment the Score in the Game Manager Singleton**

1. **Create GameManager Singleton:**
   * In the **Project** window, create a new C# script named GameManager.
2. **Edit the GameManager Script:**

using UnityEngine;

using UnityEngine.SceneManagement; // Add this namespace

public class GameManager : MonoBehaviour

{

public static GameManager Instance;

private int score = 0;

public int targetScore = 4; // Score to reach before changing scenes

void Awake()

{

// Singleton pattern

if (Instance == null)

{

Instance = this;

DontDestroyOnLoad(gameObject); // Optional

}

else

{

Destroy(gameObject);

}

}

public void IncrementScore()

{

score++;

Debug.Log("Score: " + score);

if (score >= targetScore)

{

LoadNewScene();

}

}

void LoadNewScene()

{

SceneManager.LoadScene("VictoryScene"); // Use your new scene's name

}

}

The dont destroy on load will make sure the game manager stays active between scene loading.

1. **Add GameManager to Scene:**
   * Create an empty GameObject in the **Hierarchy**.
   * Name it GameManager.
   * Attach the GameManager script to it.
2. **Ensure Enemy Calls IncrementScore():**
   * Confirm that the GetHit() method in EnemyController calls GameManager.Instance.IncrementScore();.

### **Task 7: When the Enemy Gets Hit, Play the Reaction Animation, Particle System, and Sound**

1. **Create Hit Sound:**
   * Import the hit sound effect into the **Project** window.
2. **Add Audio Source to Enemy:**
   * Select the enemy GameObject.
   * Click **Add Component** and choose **Audio Source**.
   * Uncheck **Play On Awake**.
3. **Create Enemy Script:**
   * In the **Project** window, create a new C# script named EnemyController.
4. **Edit the Enemy Script:**

using UnityEngine;

public class EnemyController : MonoBehaviour

{

private Animator animator;

private ParticleSystem hitParticles;

private AudioSource audioSource;

void Start()

{

animator = GetComponent<Animator>();

hitParticles = GetComponent<ParticleSystem>();

audioSource = GetComponent<AudioSource>();

}

public void GotHit()

{

animator.SetTrigger("GotHit");

hitParticles.Play();

audioSource.Play();

GameManager.Instance.IncrementScore();

}

}

Uncomment the commented code in the Hit detector

using UnityEngine;

public class HitDetector : MonoBehaviour

{

private void OnTriggerEnter(Collider other)

{

if (other.CompareTag("Enemy"))

{

EnemyController enemyController = other.GetComponent<EnemyController>();

if (enemyController != null)

{

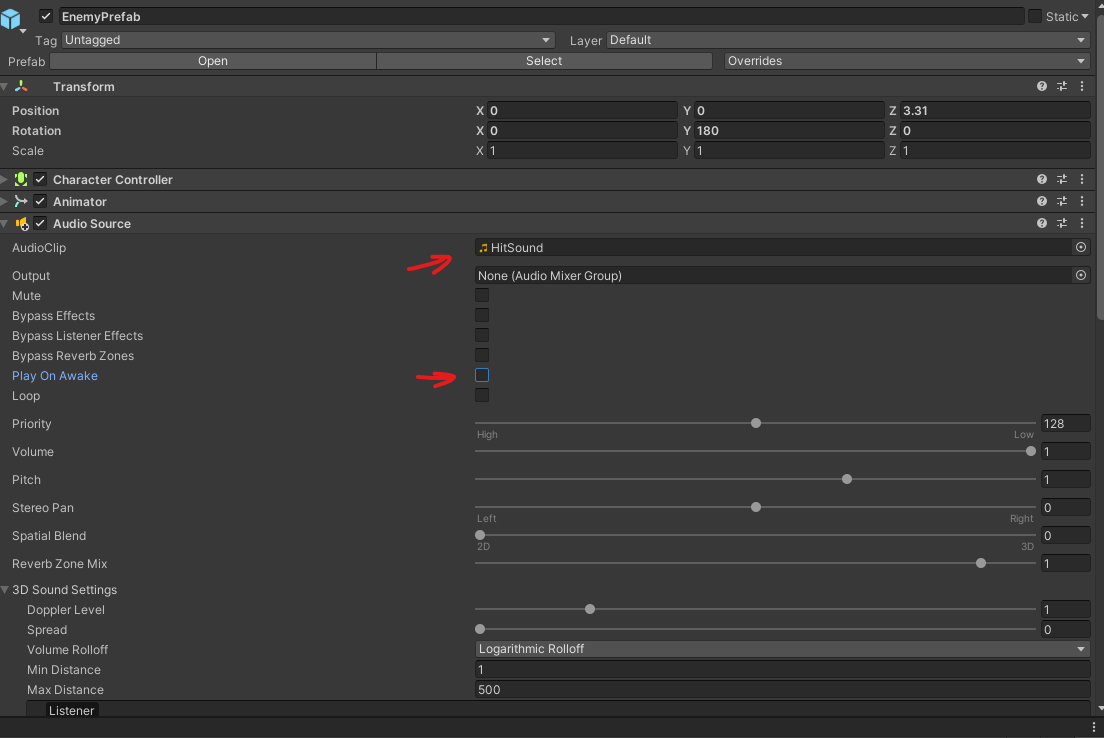
enemyController.GotHit();

}

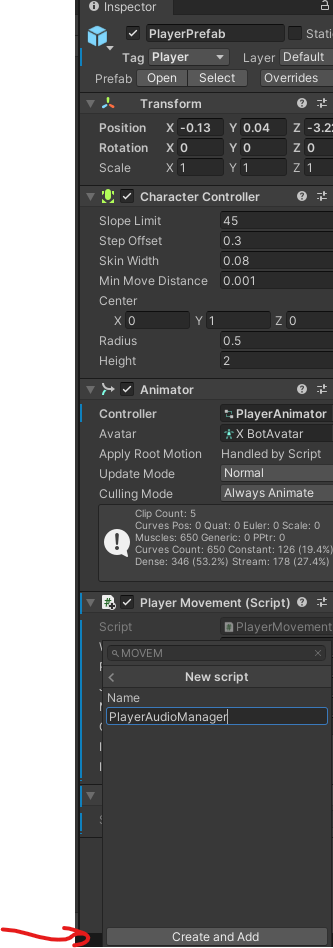
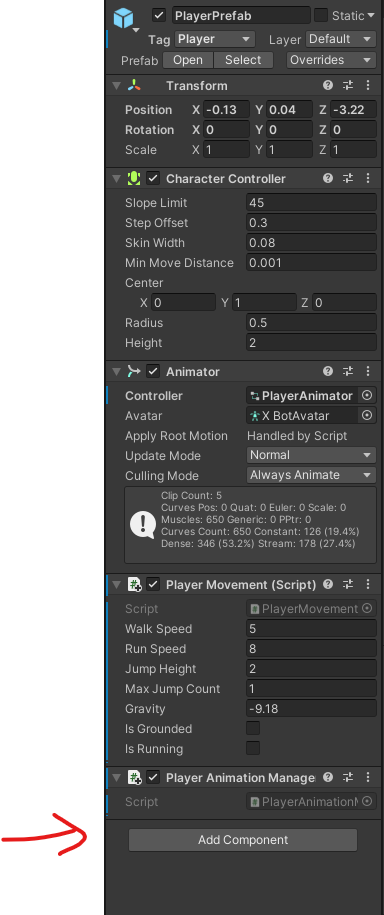
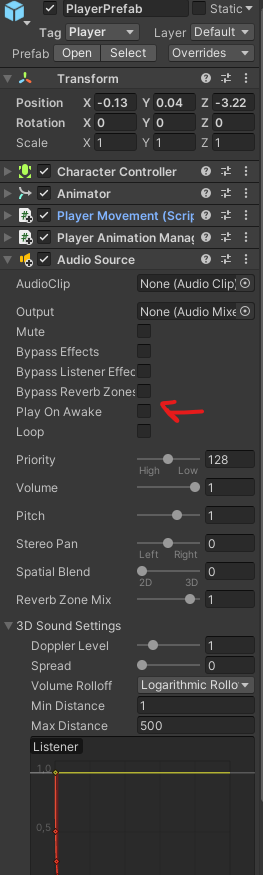
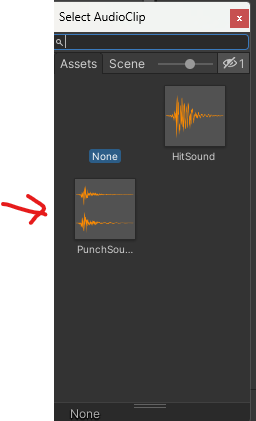
}

}

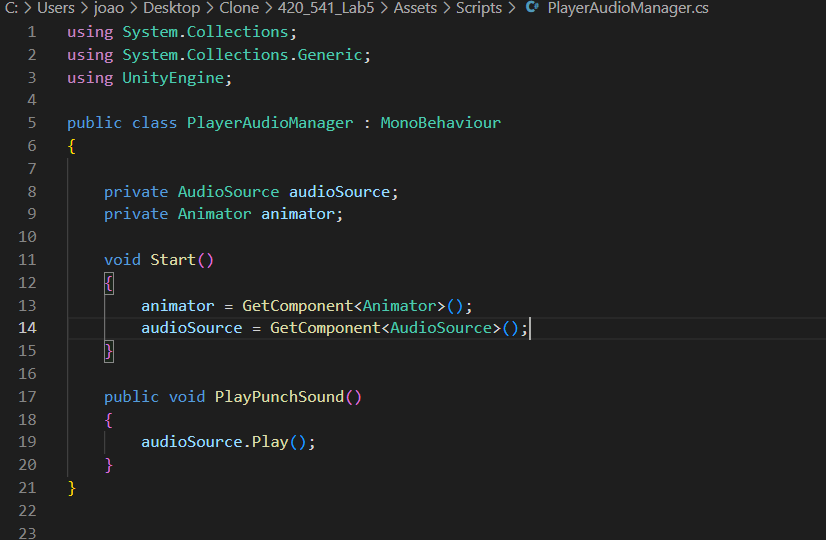
}

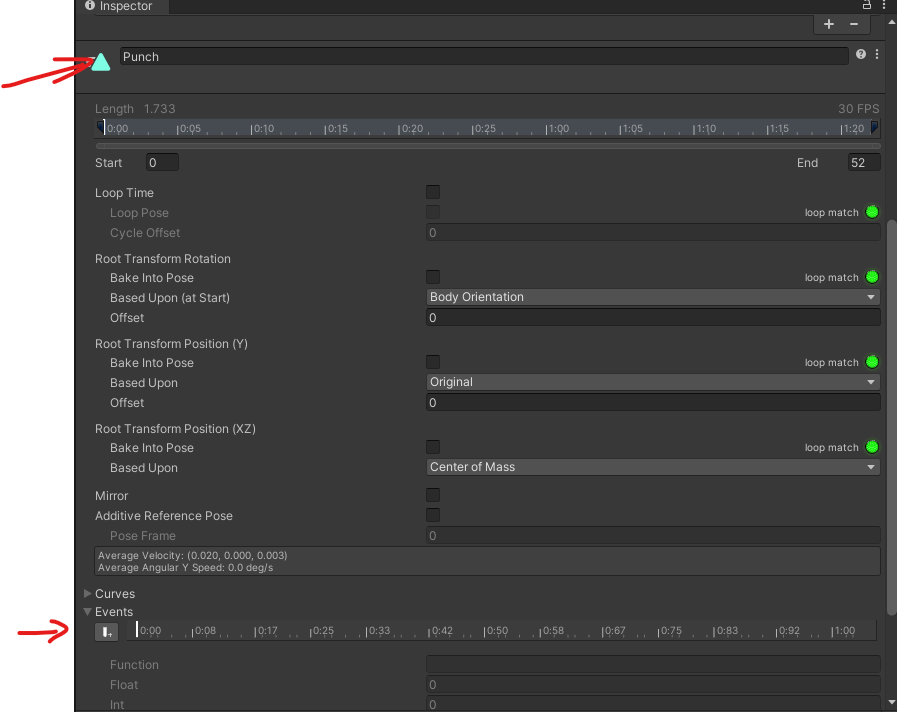
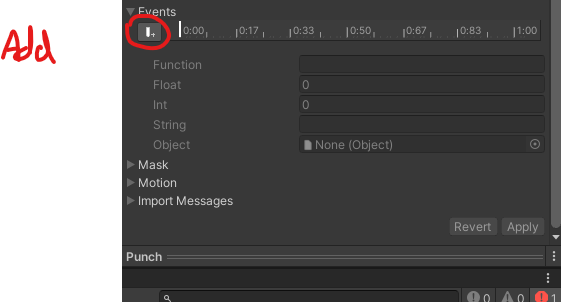
1. **Assign Hit Sound:**
   * In the **Inspector**, assign the hit sound to the Audio Source component.
   * 
2. **Set Up Animator Trigger:**
   * Ensure the **GotHit** trigger is set up in the enemy's Animator Controller.

### **Task 7: Add Animation Event to Play Sounds When You Punch**

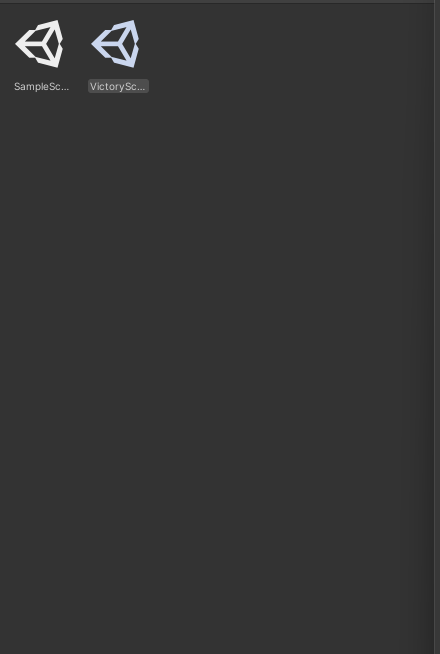
1. **Create a script on the player called PlayerAudioManager **
2. **Create Audio Source:**
   * Select the player GameObject.
   * In the **Inspector**, click **Add Component** and choose **Audio Source**.
   * Uncheck **Play On Awake**.
   * 
3. **Add Audio Clip Reference:**
   * In the script, add a reference to the Audio Source:
   * By click the little circle and then selecting the PunchSound
   * 

Now lets code the PlayerAudioManager:

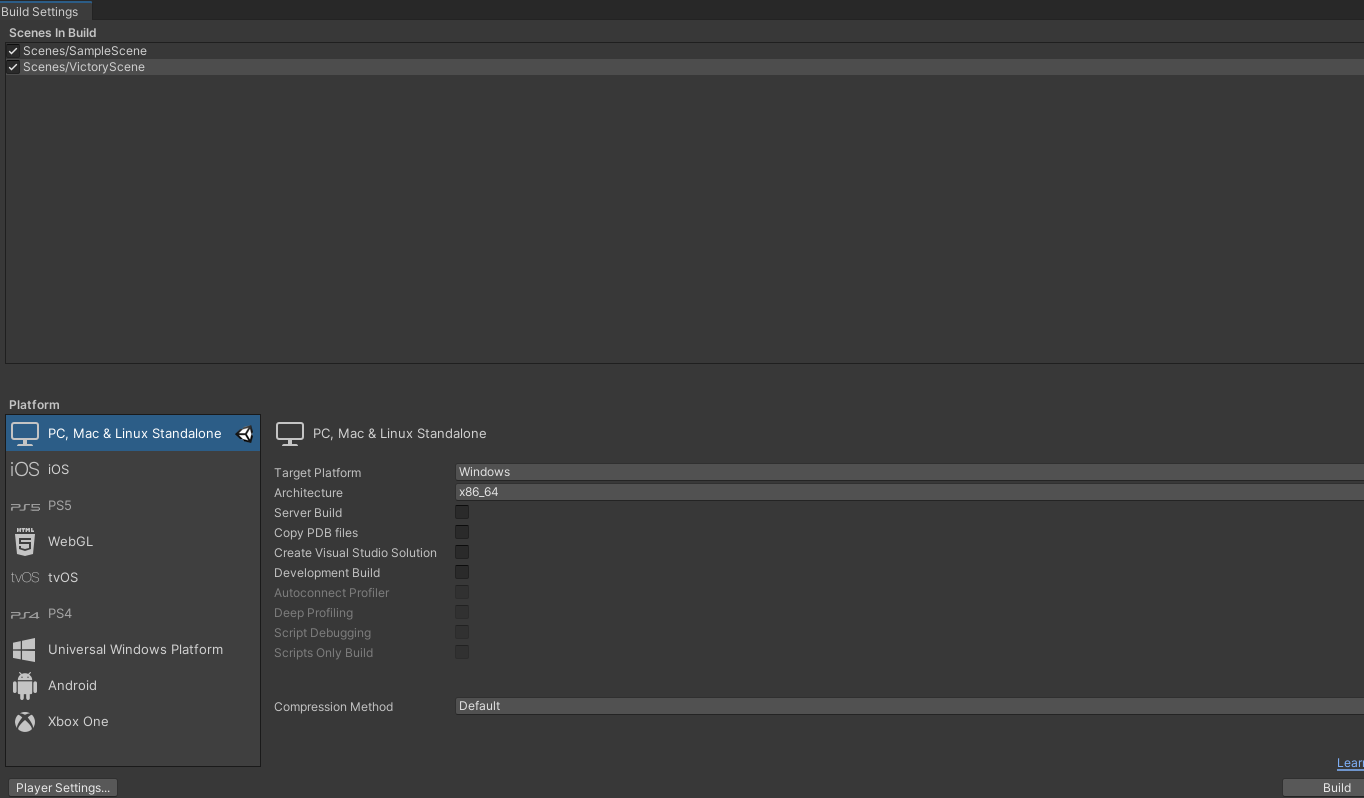


1. **Add Animation Events to Punch Animations:**
   * Select the punch attack animation clip.
   * 
   * Open the Animation window.
   * Move the playhead to the frame where the punch occurs.
   * Click **Add Event**.
   * In the event, type PlayPunchSound.
   * 

### **Task 9: Create a New Scene and Save It**

1. **Create New Scene:**
   * Go to **File > New Scene**.
2. **Set Up Scene (Optional):**
   * Add any GameObjects or settings needed for the new scene.
3. **Save the Scene:**
   * Go to **File > Save As**.
   * Name the scene (e.g., VictoryScene.unity) and save it in your **Scenes** folder.
   * 

### **Task 11: Add Your New Scene to the Build Settings**

1. **Open Build Settings:**
   * Go to **File > Build Settings**.
   * 
2. **Add Open Scenes:**
   * With your new scene open, click **Add Open Scenes**.
   * Ensure both your main game scene and the new scene are listed in the **Scenes In Build**.
3. **Verify Scene Order:**
   * Check that the scenes are in the correct order or adjust as needed.

Submit your project on github.