Pose Blender Lite - Documentation

Ver. 1.0.1
Made by: ChiefValentine, BSS

What Does It Do?

The **Pose Blender Lite** suite offers a range of tools designed to enhance character animation and interaction in your Unity projects. It enables more natural transitions and greater control over your character's pose by providing the following key features:

- **Pose Blender Lite:**
 Seamlessly blend between animations by overriding specific bone transforms at runtime. This component lets you sample an animation frame and overlay that frame onto selected bones, creating smooth and natural transitions. You can also manually adjust rotation offsets and blend weights for precise control over the final pose.

- **IK Controller:**
 Address the limitations of traditional animation by applying inverse kinematics (IK). This controller automatically adjusts hand or arm positions to match your character's movements, ensuring that interactions with the environment appear realistic.

- **PoseRecorder:**
 Easily capture and store any desired pose or animation. With a single click, record the current state and generate a new AnimationData ScriptableObject (AnimationDataSO), making it simple to reuse or modify poses as needed.

- **Camera Stabilizer:**
 Enhance camera movement by synchronizing it with your character's animations. This tool stabilizes the camera by considering the head and spine transforms, ensuring smooth and responsive visual tracking throughout the scene.

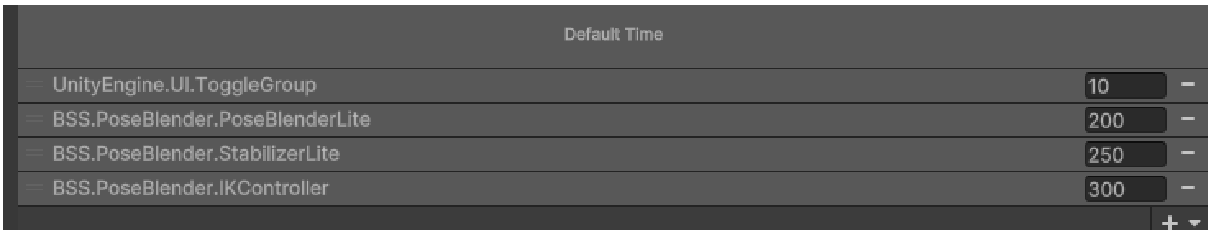
How It Works:
 Pose Blender Lite applies its updates during the Late Update phase, ensuring that all modifications—from pose blending to IK corrections—are applied after the base animations are processed. This timing guarantees that your adjustments are smoothly integrated into the final output.

Getting Started

Setting up Pose Blender Lite is simple and can be broken down into a few easy steps:

Step 0: Prepare Your Script Execution Order

- Go to **Edit - Project Settings - Script Execution Order**
- Set the Execution order to be as followed:
    - First PoseBlenderLite
    - Stabilizer Lite
    - IKController

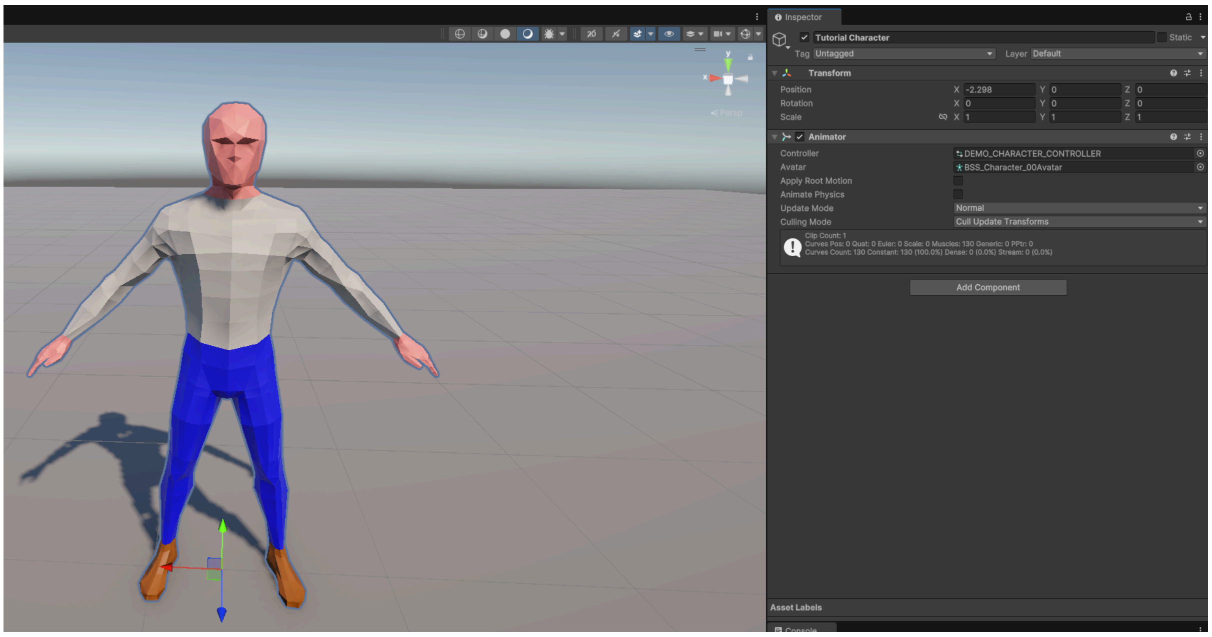


Default Time

UnityEngine.UI.ToggleGroup	10
BSS.PoseBlender.PoseBlenderLite	200
BSS.PoseBlender.StabilizerLite	250
BSS.PoseBlender.IKController	300

Step 1: Prepare Your Character

- **Drag Your Character into the Scene:**
  Ensure your character has an Animation Controller with at least one animation.



Inspector

Tutorial Character — Static
Tag: Untagged — Layer: Default

Transform

	X	Y	Z
Position	-2.298	0	0
Rotation	0	0	0
Scale	1	1	1

Animator

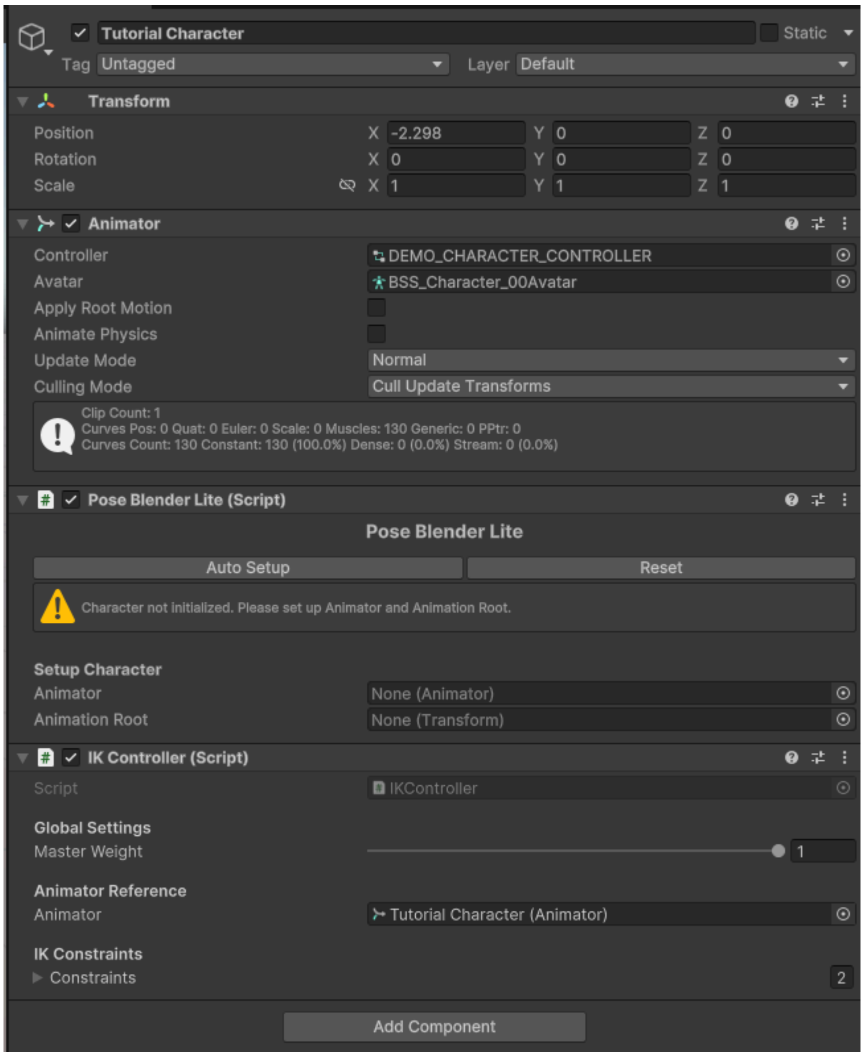
Controller	DEMO_CHARACTER_CONTROLLER
Avatar	BSS_Character_00Avatar
Apply Root Motion	☐
Animate Physics	☐
Update Mode	Normal
Culling Mode	Cull Update Transforms

```
Clip Count: 1
Curves Pos: 0 Quat: 0 Euler: 0 Scale: 0 Muscles: 130 Generic: 0 PPtr: 0
Curves Count: 130 Constant: 130 (100.0%) Dense: 0 (0.0%) Stream: 0 (0.0%)
```

Step 2: Add and Configure Pose Blender Lite

2a. Add the Component

- **Add Pose Blender Lite:**
  Drag the Pose Blender Lite component onto your character.



Tutorial Character Static
Tag Untagged Layer Default

Transform

Position	X -2.298	Y 0	Z 0
Rotation	X 0	Y 0	Z 0
Scale	X 1	Y 1	Z 1

Animator

Controller	DEMO_CHARACTER_CONTROLLER
Avatar	BSS_Character_00Avatar
Apply Root Motion	☐
Animate Physics	☐
Update Mode	Normal
Culling Mode	Cull Update Transforms

```
Clip Count: 1
Curves Pos: 0 Quat: 0 Euler: 0 Scale: 0 Muscles: 130 Generic: 0 PPtr: 0
Curves Count: 130 Constant: 130 (100.0%) Dense: 0 (0.0%) Stream: 0 (0.0%)
```

Pose Blender Lite (Script)

Pose Blender Lite

Auto Setup Reset

Character not initialized. Please set up Animator and Animation Root.

Setup Character

Animator	None (Animator)
Animation Root	None (Transform)

IK Controller (Script)

Script	IKController

Global Settings

Master Weight	1

Animator Reference

Animator	Tutorial Character (Animator)

IK Constraints

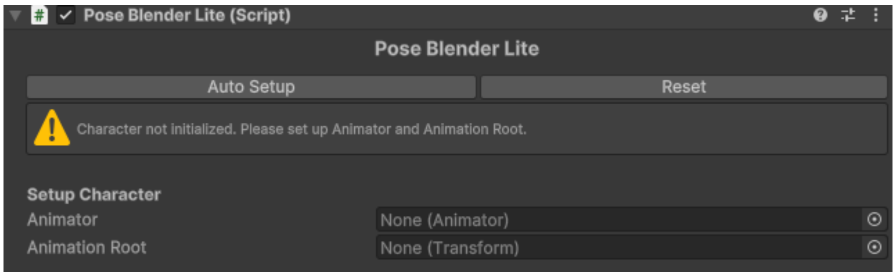
Constraints	2

Add Component

Note: This automatically adds the IK Controller.

2b. Configure Pose Blender Lite

- **Auto Setup:**
  Click **Auto Setup** to automatically assign the Animator and Animation Root.
- **Manual Setup:**
  Alternatively, drag the references manually into the designated fields.
- **Reset:**
  Use the **Reset** button to clear the current Animator and Animation Root settings if needed.



Character not initialized. Please set up Animator and Animation Root.

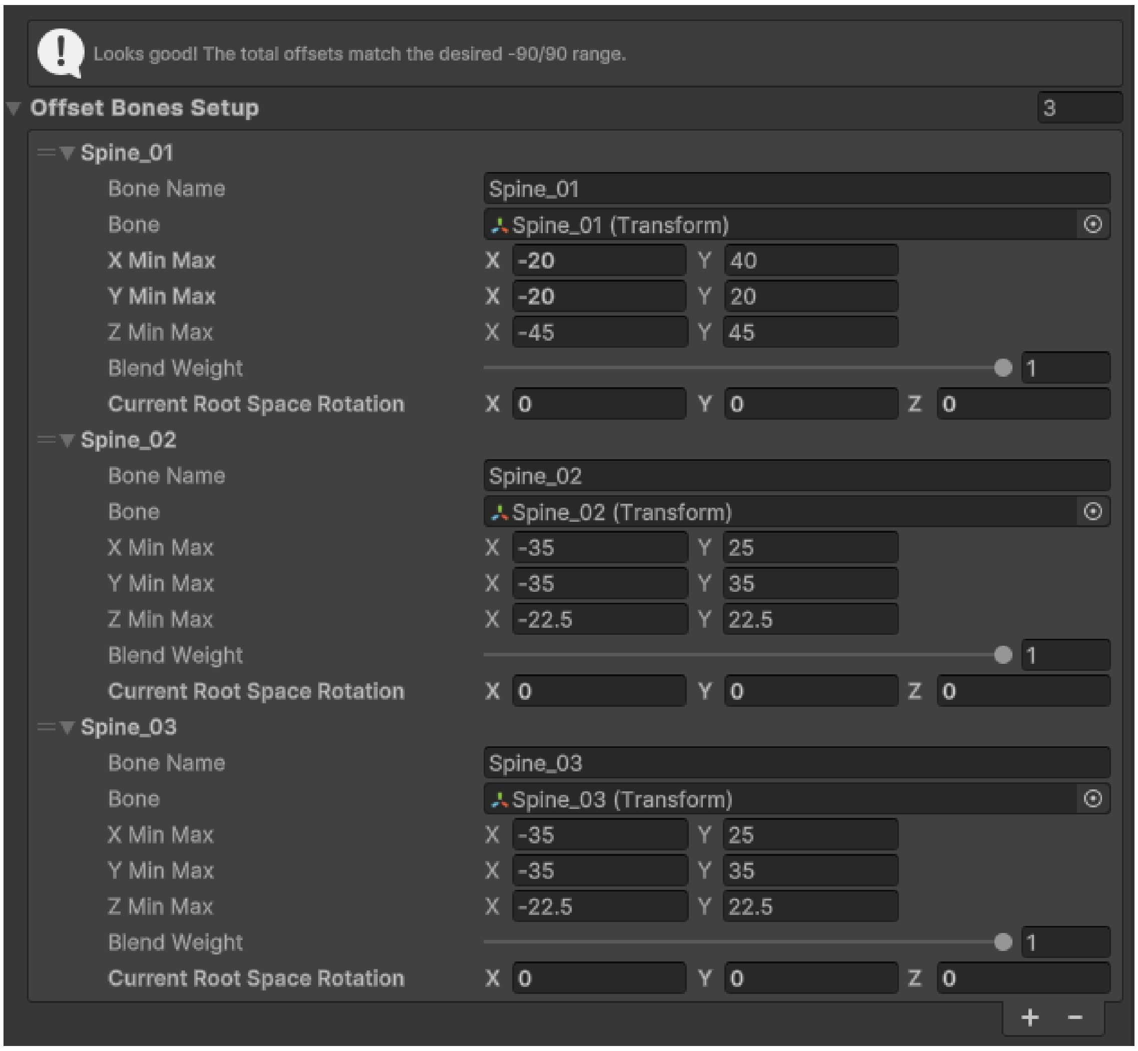
Setup Character
Animator: None (Animator)
Animation Root: None (Transform)

2c. Configure Offset Bones

The offset bones are rotated according to your character's offsets. They must adhere to a specific rotation constraint:

- **Rotation Constraint:**
  - All rotations are based on a maximum of **90°**.
  - **Important:** The sum of the individual **X Min** values must equal **-90°** and the sum of the **X Max** values must equal **90°** (and similarly for Y and Z).
  - A helper message will appear in the inspector to guide you through this setup.

- **Demo Reference:**
  The demo character uses three offset bones (spine_01, spine_02, and spine_03). You can copy these settings or set up your own—just remember to reference your character's bones for correct behavior.



Looks good! The total offsets match the desired -90/90 range.

Offset Bones Setup 3

Spine_01

Bone Name	Spine_01
Bone	Spine_01 (Transform)
X Min Max	X -20 Y 40
Y Min Max	X -20 Y 20
Z Min Max	X -45 Y 45
Blend Weight	1
Current Root Space Rotation	X 0 Y 0 Z 0

Spine_02

Bone Name	Spine_02
Bone	Spine_02 (Transform)
X Min Max	X -35 Y 25
Y Min Max	X -35 Y 35
Z Min Max	X -22.5 Y 22.5
Blend Weight	1
Current Root Space Rotation	X 0 Y 0 Z 0

Spine_03

Bone Name	Spine_03
Bone	Spine_03 (Transform)
X Min Max	X -35 Y 25
Y Min Max	X -35 Y 35
Z Min Max	X -22.5 Y 22.5
Blend Weight	1
Current Root Space Rotation	X 0 Y 0 Z 0

- **Feedback:**
  When the offset bones are configured correctly, you will see a confirmation message in the inspector.

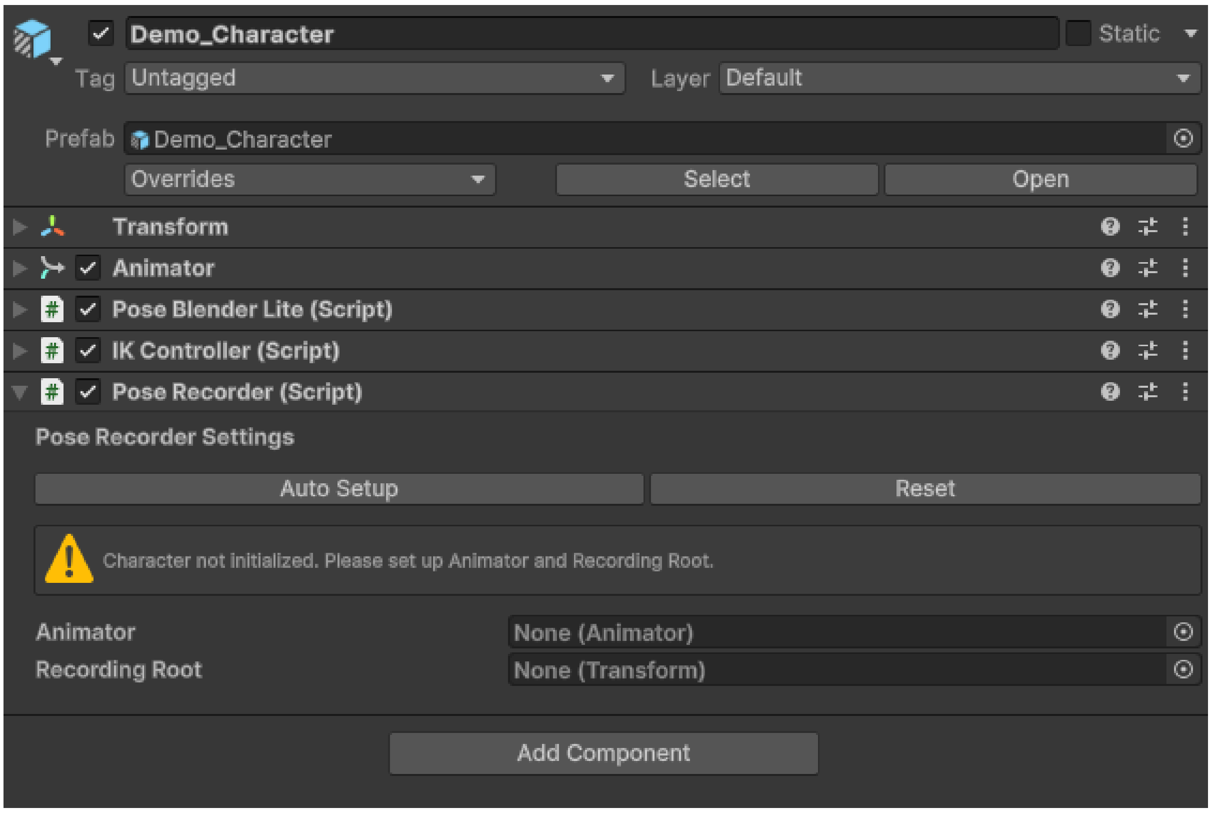
- **Preview in Editor:**
  Enable the preview mode to see real-time adjustments using the character offsets.

Step 3: Creating Animation Data

Before you can set up an overlay pose, you need to capture the animation data that represents the desired pose.

3a. Add the PoseRecorder to Your Character

- Ensure PoseRecorder setup matches Pose Blender Lite (assigned Animator and Animation Root).
- Verify **POSE_CONTROLLER** and **POSE_OVERRIDE_CONTROLLER** assignments.

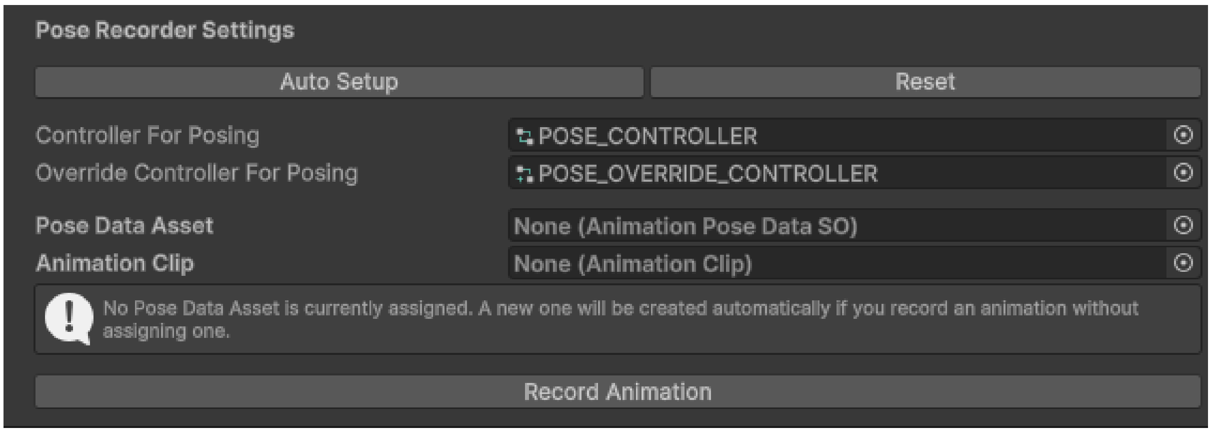


Demo_Character
Static
Tag: Untagged
Layer: Default
Prefab: Demo_Character
Overrides
Select
Open
Transform
Animator
Pose Blender Lite (Script)
IK Controller (Script)
Pose Recorder (Script)
Pose Recorder Settings
Auto Setup
Reset

Character not initialized. Please set up Animator and Recording Root.

Animator: None (Animator)
Recording Root: None (Transform)

Add Component



Pose Recorder Settings
Auto Setup
Reset

Controller For Posing: POSE_CONTROLLER
Override Controller For Posing: POSE_OVERRIDE_CONTROLLER
Pose Data Asset: None (Animation Pose Data SO)
Animation Clip: None (Animation Clip)

No Pose Data Asset is currently assigned. A new one will be created automatically if you record an animation without assigning one.

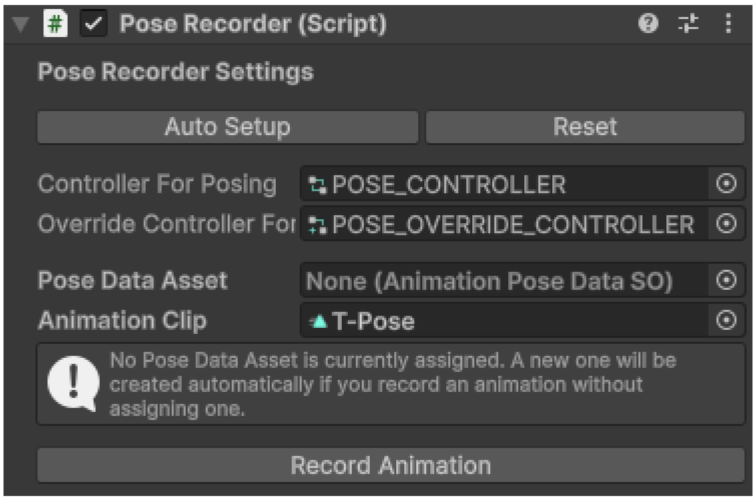
Record Animation

- You are now ready to record Pose Data

3b. Record a Pose

- Select the animation clip you want to sample.
    - *Make sure this animation is compatible with your current character.*

- Click 'Record Animation' to capture animation frames.

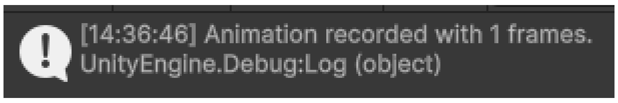


Controller For Posing: POSE_CONTROLLER
Override Controller For: POSE_OVERRIDE_CONTROLLER
Pose Data Asset: None (Animation Pose Data SO)
Animation Clip: T-Pose

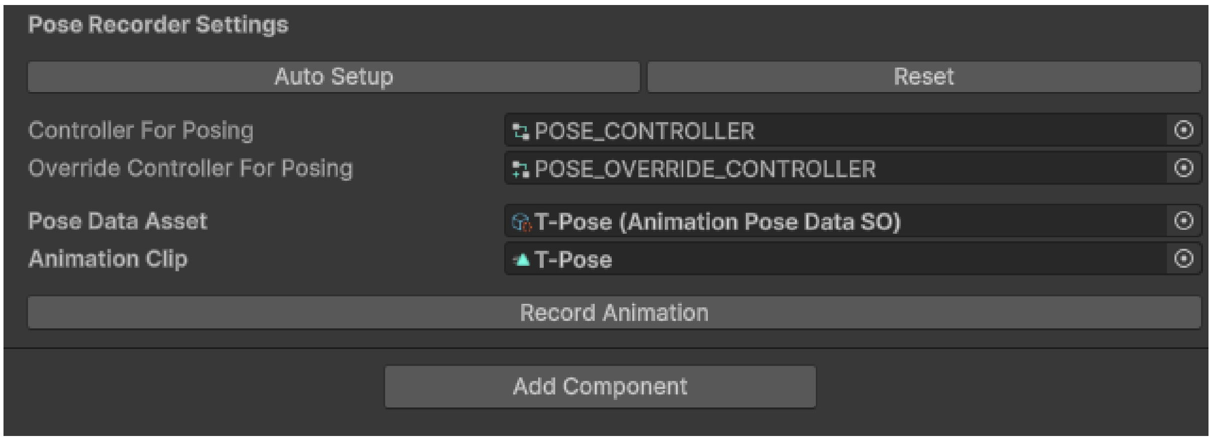
No Pose Data Asset is currently assigned. A new one will be created automatically if you record an animation without assigning one.

- Select save location for new Pose Data asset.
    - *If a Pose Data Asset has been referenced the Pose Data will be overwritten.*

- Console confirms the number of frames captured; the overlay pose uses only the first frame.



```
[14:36:46] Animation recorded with 1 frames.
UnityEngine.Debug:Log (object)
```

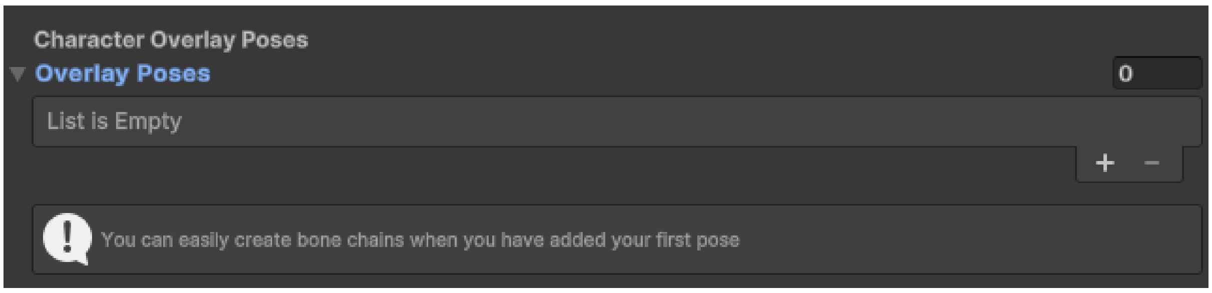


Controller For Posing: POSE_CONTROLLER
Override Controller For Posing: POSE_OVERRIDE_CONTROLLER
Pose Data Asset: T-Pose (Animation Pose Data SO)
Animation Clip: T-Pose

Step 4: Setting Up the Pose Overlay

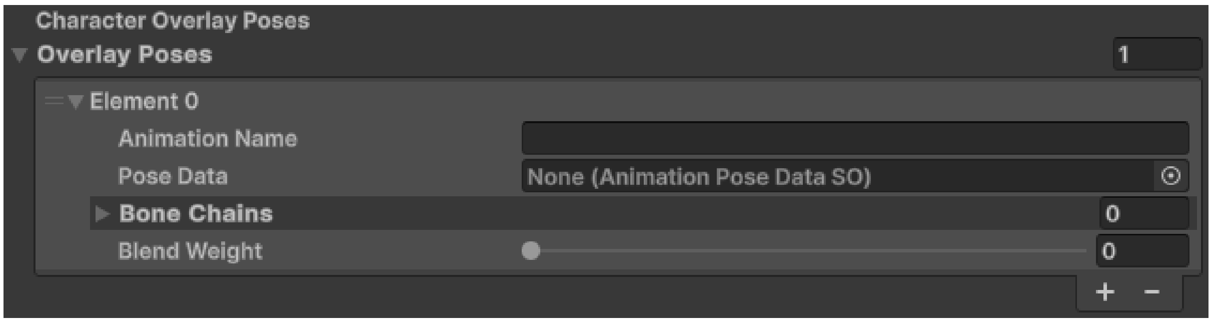
With your Animation Data ready, you can now apply it as an overlay pose to blend with your character's base animation.

1. **Add an Overlay Pose:**



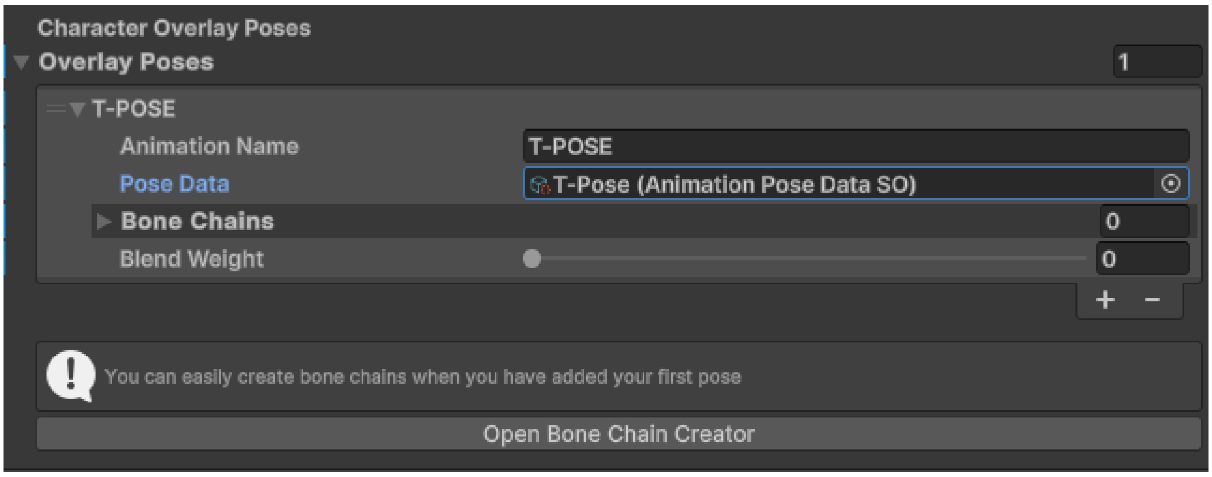
Character Overlay Poses
Overlay Poses 0
List is Empty
You can easily create bone chains when you have added your first pose

   - Add a new element to the overlay poses list.



Character Overlay Poses
Overlay Poses 1
Element 0
Animation Name
Pose Data None (Animation Pose Data SO)
Bone Chains 0
Blend Weight 0

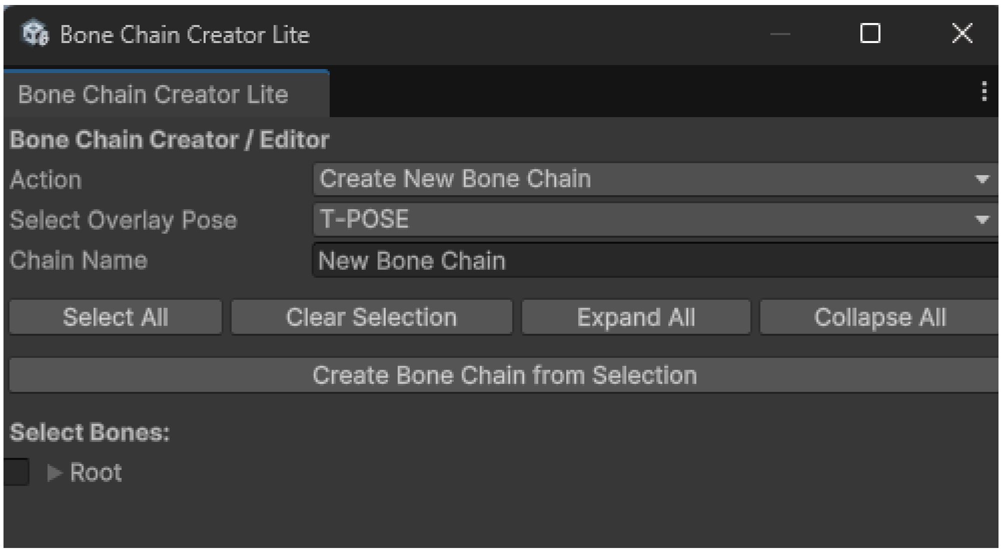
   - Assign a descriptive name to the overlay and link the newly created Pose Data asset.



Character Overlay Poses
Overlay Poses 1
T-POSE
Animation Name T-POSE
Pose Data T-Pose (Animation Pose Data SO)
Bone Chains 0
Blend Weight 0
You can easily create bone chains when you have added your first pose
Open Bone Chain Creator

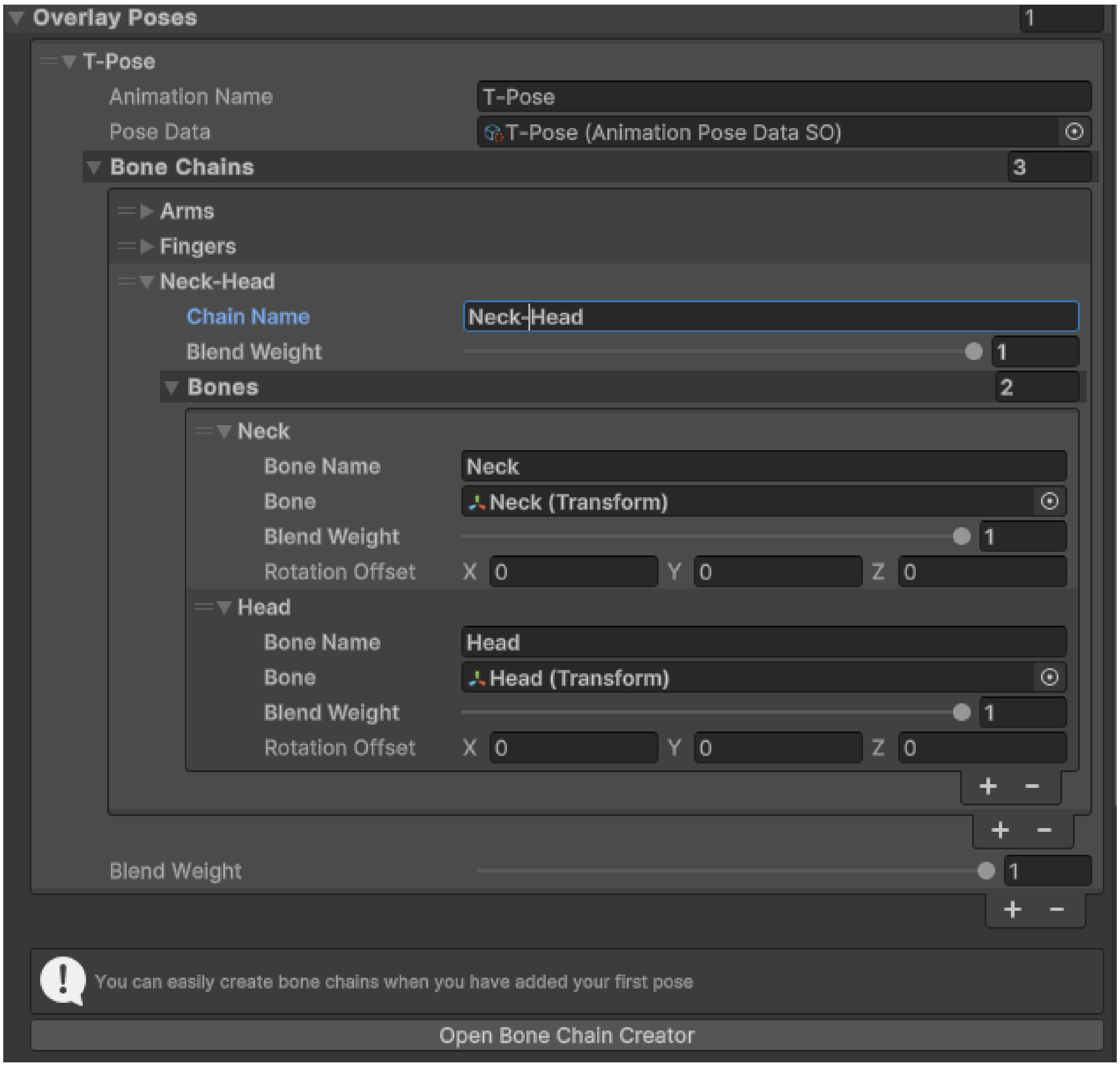
2. **Configure Bone Chains:**

   ○ Click **Open Bone Chain Creator** to launch the Bone Chain Wizard.



Bone Chain Creator / Editor
Action: Create New Bone Chain
Select Overlay Pose: T-POSE
Chain Name: New Bone Chain
Select Bones:

- Use the wizard to create and edit bone chains
  - Choose the action you would like to do
  - Select the pose to apply it to.
  - Give the chain a name.
  - Select the bones you want the chain to have.
  - Hit ‘Create Bone Chain from Selection’
  - Your bone chain will now be added to your overlay pose.



Overlay Poses 1

T-Pose

Animation Name: T-Pose

Pose Data: T-Pose (Animation Pose Data SO)

Bone Chains 3

Arms

Fingers

Neck-Head

Chain Name: Neck-Head

Blend Weight: 1

Bones 2

Neck

Bone Name: Neck

Bone: Neck (Transform)

Blend Weight: 1

Rotation Offset: X 0 Y 0 Z 0

Head

Bone Name: Head

Bone: Head (Transform)

Blend Weight: 1

Rotation Offset: X 0 Y 0 Z 0

Blend Weight: 1

You can easily create bone chains when you have added your first pose

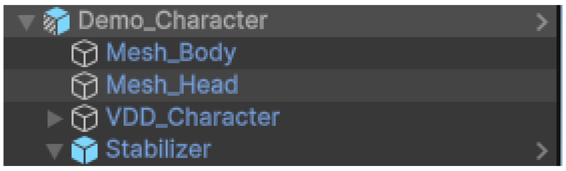
- If the blend weight is set to 1, all bones in that chain will fully rotate toward their Pose Data rotation.

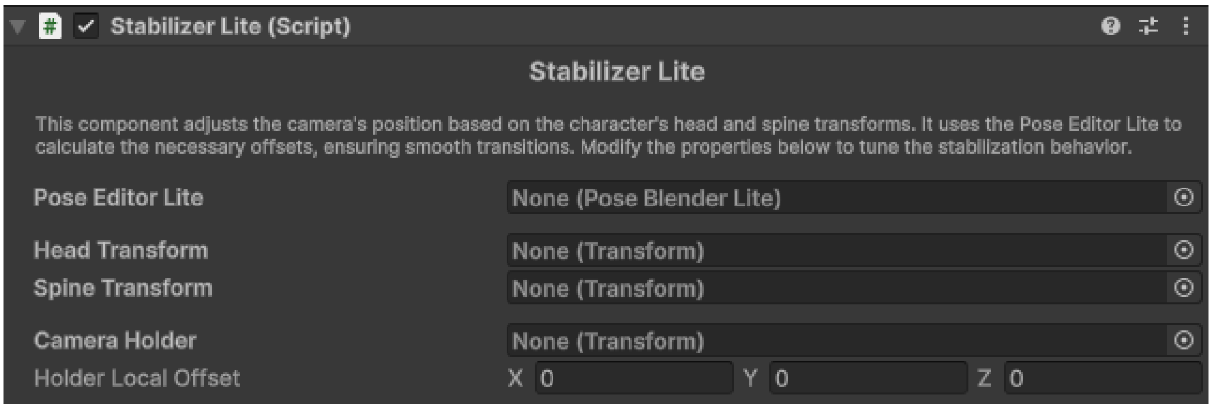
- Adjust individual blend weights as needed for more natural blends between the current animation and the pose.

- If the overlay isn’t working immediately, it’s likely because the bone chains that the pose affects haven’t been assigned.

Step 5: Setting Up Stabilizer Lite

Stabilizer Lite is used to stabilize your camera. Add the Camera Stabilizer to your character, make sure your character is using Pose Blender Lite.




Stabilizer Lite (Script)

Stabilizer Lite

This component adjusts the camera's position based on the character's head and spine transforms. It uses the Pose Editor Lite to calculate the necessary offsets, ensuring smooth transitions. Modify the properties below to tune the stabilization behavior.

Pose Editor Lite: None (Pose Blender Lite)
Head Transform: None (Transform)
Spine Transform: None (Transform)
Camera Holder: None (Transform)
Holder Local Offset: X 0 Y 0 Z 0

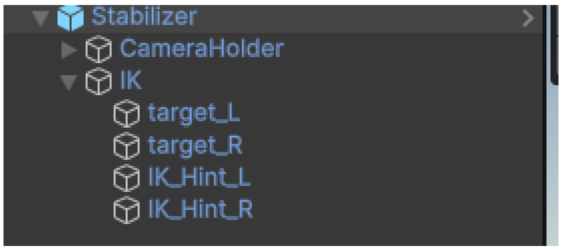
Configuring the Camera Stabilizer:

- Ensure **Stabilizer Lite** references your character’s **Pose Blender Lite**.
- Assign the head transform to position the **CameraHolder** accurately.
- Assign the spine transform to the uppermost spine bone for optimal stabilization results.
- **CameraHolder** should be a child of the stabilizer. Attached to the **CameraHolder** transform is the **Camera** object.
- **Holder Local Offset** lets you offset the camera holder local position.

Step 6: Setting Up IK

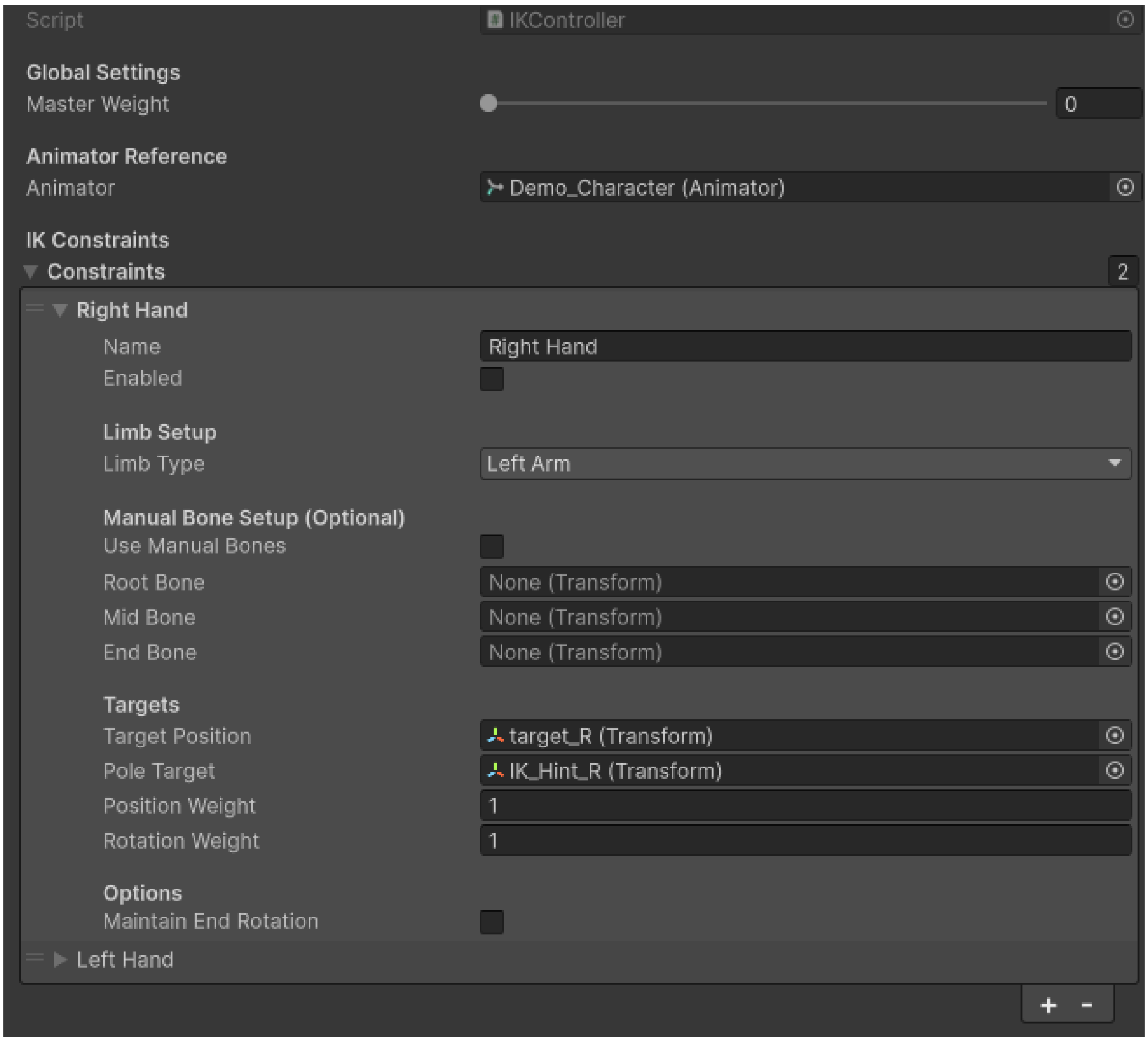
When Pose Blender Lite is added, the IK Controller is automatically included. Ensure proper execution order by adding both scripts to the Unity script execution order. The IK Controller must run after Pose Blender Lite.

- ***IK SETUP INDICATIVE ONLY -***



Configuring IK:

- Add IK constraints to the provided list.
- Name the limb and select its type.
- For Humanoid characters, manual bone setup is optional but available.
- Assign IK targets and hints appropriately. The demo character demonstrates this by synchronizing IK targets and hints with Pose Blender Lite rotations, ensuring natural and coherent movements.



Script: IKController

Global Settings
Master Weight: 0

Animator Reference
Animator: Demo_Character (Animator)

IK Constraints
Constraints: 2

Right Hand
- Name: Right Hand
- Enabled: ☐

Limb Setup
- Limb Type: Left Arm

Manual Bone Setup (Optional)
- Use Manual Bones: ☐
- Root Bone: None (Transform)
- Mid Bone: None (Transform)
- End Bone: None (Transform)

Targets
- Target Position: target_R (Transform)
- Pole Target: IK_Hint_R (Transform)
- Position Weight: 1
- Rotation Weight: 1

Options
- Maintain End Rotation: ☐

Left Hand

PoseBlenderLite Methods

Public Methods

- **SetLookOffsets(float vertical, float horizontal, float leaning):**

  Sets the look offsets for vertical, horizontal, and leaning angles, clamping each to the range –90 to 90 degrees.

  Example:

  ```
  poseBlenderLite.SetLookOffsets(30f, 15f, 5f);
  ```

- **AutoSetupCharacter():**

  Automatically configures the character by assigning the animation root (if not set) and retrieving the Animator component from the GameObject. Marks the component as initialized if successful.

  Example:

  ```
  poseBlenderLite.AutoSetupCharacter();
  ```

- **ResetCharacter():**

  Resets the character configuration by clearing the animation root and animator references, marking the component as uninitialized.

  Example:

  ```
  poseBlenderLite.ResetCharacter();
  ```

- **GetRoot():**
  Searches among the child transforms for one named "Root" or "root" and returns it. Returns null if no matching transform is found.
  Example:

  ```
  Transform root = poseBlenderLite.GetRoot();
  ```

IKController Methods

Public Methods

- **SetMasterWeight(float weight)**

  Sets the overall master IK weight, clamped between 0 and 1. This weight scales the influence of all IK constraints.

  Usage Example:

  ikController.SetMasterWeight(0.75f);

- **SetLayerWeightByName(string layerName, float blendWeight)**

  Finds the IK constraint with the specified name and sets both its position and rotation weights to the given blend weight.

  Usage Example:

  ikController.SetLayerWeightByName("Left Hand", 0.8f);

- **SetLayerWeightByName(string layerName, float positionBlendWeight, float rotationBlendWeight)**

  Finds the IK constraint with the specified name and sets its position and rotation weights independently.

  Usage Example:

- ikController.SetLayerWeightByName("Right Hand", 0.8f, 0.6f);