RAGDOLL MECANIM MIXER

Guide

VIDEO GUIDE

https://assetstore.altinqiran.kz/ramecan-mixer/
nurzzz27@gmail.com

CONTENTS

UPDATES

Version	Date	Changes
1.01	28.03.2020	PostProcess package updated Joint Drive Damper issue fixed Mixer states saving issue fixed

DEMO SCENE

Animations

Since the demo scene uses animations from another asset, you need to download this animation package. Asset Store: [RPG Character Mecanim Animation Pack FREE] (Thanks Explosive).

When importing, check only the «Unarmed» folder, as in 1.1 figure.

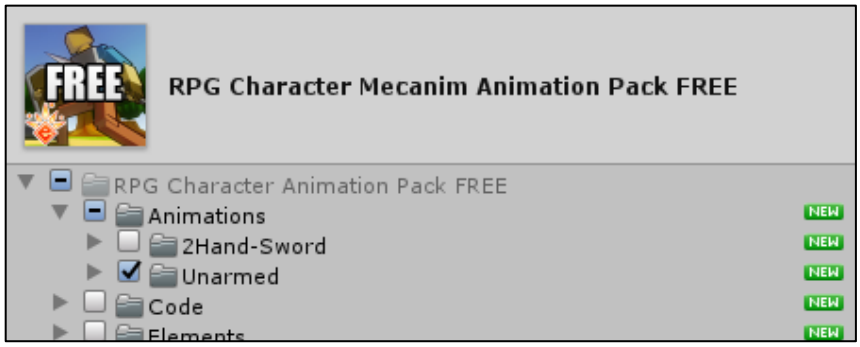


Figure 1.1

After importing, you must restart Unity for the animations to properly communicate with the project. After that, you can start the demo scene.

Post effects

In order to make the same picture as in the video, you need to import PostProcessing package from Package Manager.

After importing, the «PostProcessing» object will have a «Post Processing Volume» component with an empty “Profile” parameter. There you need to put the finished profile, which is shown in 1.2 figure.

Figure 1.2

CREATING A RAGDOLL USING ETHAN CHARACTER FROM STANDARD ASSETS

1. «Ragdoll Constructor» component and selection of active bones.

The component is responsible for creating a special ragdoll for working with asset. This constructor is different from what is built into the engine. It can be found along the path shown in figure 2.1.

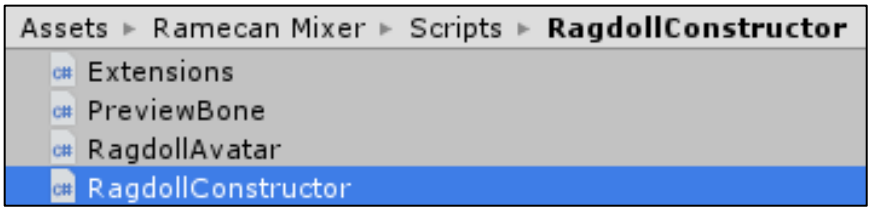


Figure 2.1

Ethan has quite a few bones, and they will all be displayed as a hierarchy tab as in figure 2.2.

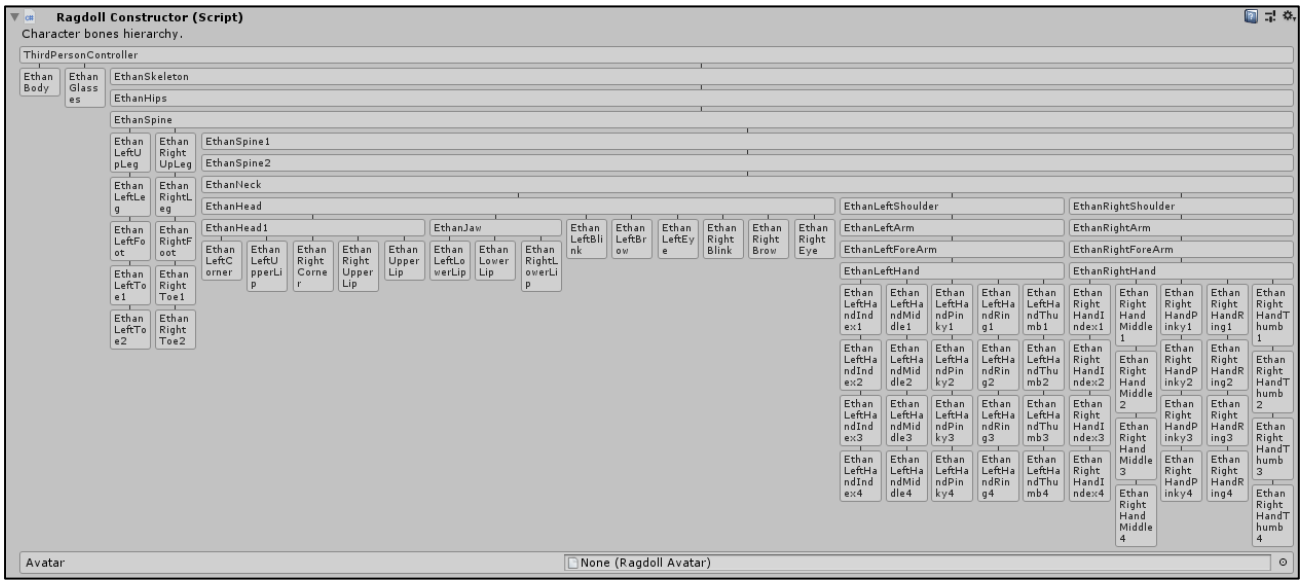


Figure 2.2

It is necessary to choose exactly the bones from which the ragdoll will be built. But first you need to install an avatar. You can use one of the ready-made ones. They are located in the «Ragdoll Avatars» folder. In figure 2.2, below, there is a special parameter. It is necessary to place the selected avatar in it.

After installing the finished avatar, another hierarchy of bones will appear in the inspector. If the constructor determines the same name between the bones of the character and the bones of the avatar, then he will automatically connect them. And if the bones of the avatar remain red, then you need to establish connections manually.

To do this, hold the mouse on the avatar's bones and drag the connecting wire to the corresponding character's bone, as in figure 2.3.

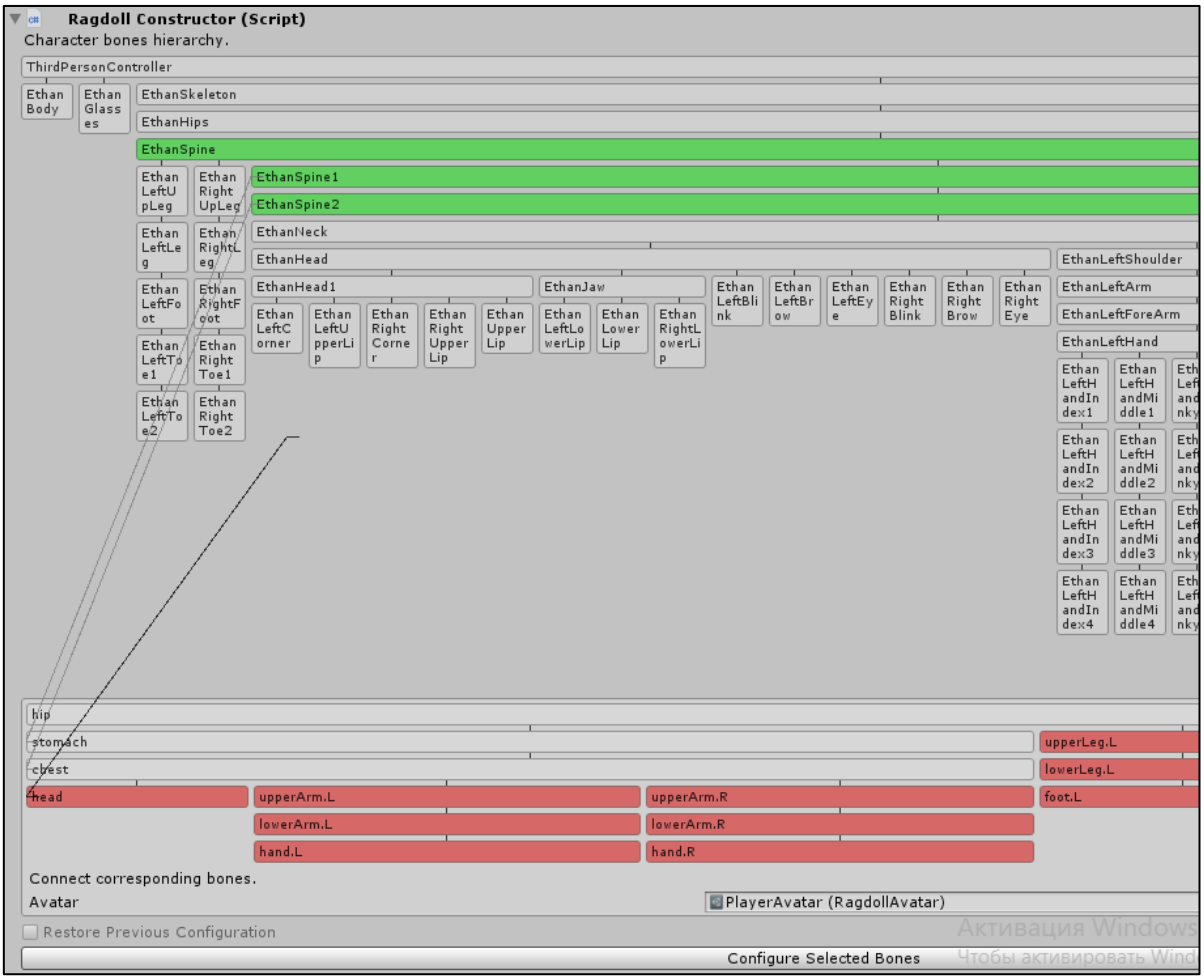


Figure 2.3

But since there is a ready-made avatar for Ethan, it is more convenient to take it. Then the connections will be configured automatically, according to the names of the bones.

Active bones are highlighted in green, from which a ragdoll will be created. By clicking on the character's bones, you can turn them on and off.

If you want to create a new avatar, then just right-click in the «Project» tab and select «Create - Ragdoll Avatar» in the context menu that opens, as in figure 2.4. It is almost at the very top of the list.

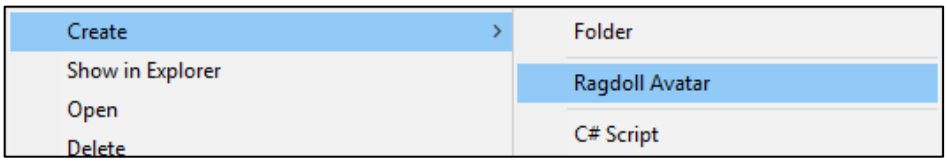


Figure 2.4

After all the desired bones are selected and highlighted in green, you can begin to configure the bones. To do this, click on the “Configure Selected Bones” button, shown at the bottom in figure 2.3.

2. Bone configuration.

At this stage, the basic work of bone configuring takes place. When you click on a bone, its settings appear at the bottom of the component, and the scene will show the collider or joint of the bone. This can be seen in 2.5 figure. You can change the bone parameters both in the inspector tab and on the scene.

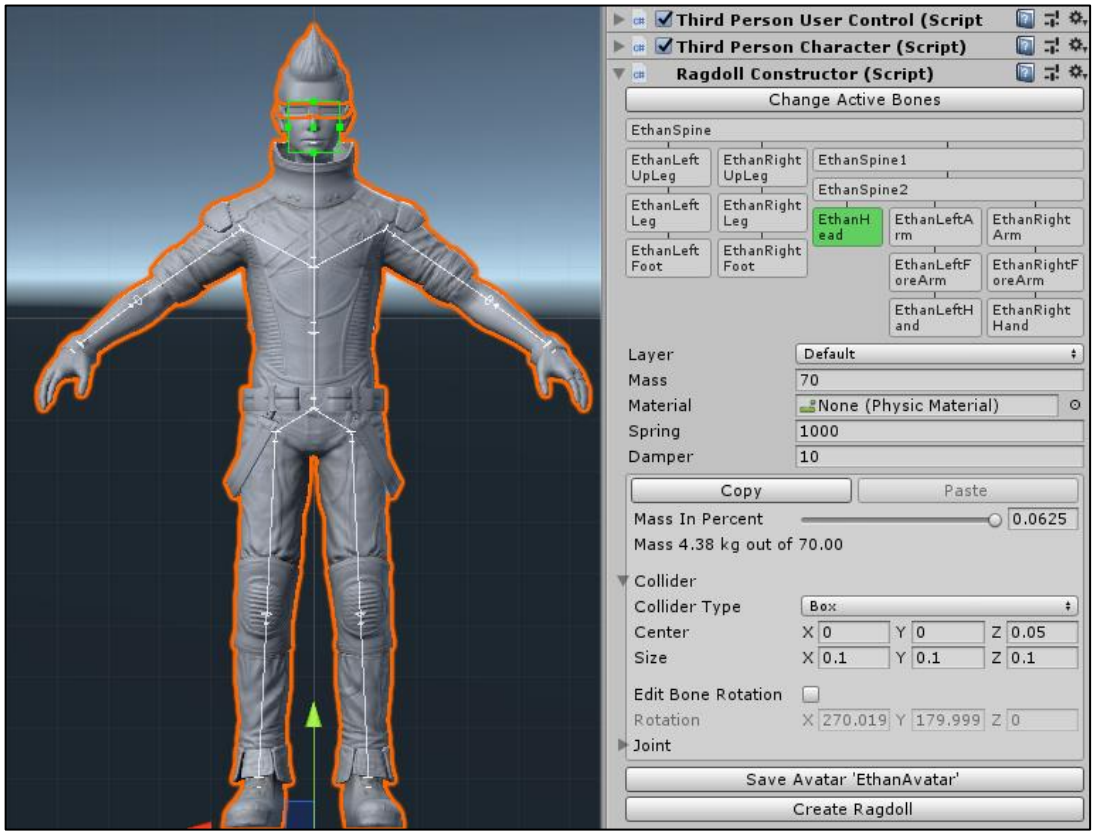


Figure 2.5

After setting up one bone, you can easily copy these parameters to another bone. This may be useful when setting up the same or mirrored body parts. The skeleton of a ragdoll is slightly different from the skeleton of a character in terms of bone orientation. It is designed so that when copying mirror bones, most of the settings are identical. But there are two parameters that need to be changed (not necessarily) manually when mirroring bones. The first, «Collider – Rotation» – this parameter is not copied and remains the same as it was. The second, «Collider – Center» - this parameter is copied, but one of the axes (usually X) will have to be inverted by adding or removing the minus(-) sign before the value of the parameter. But in most cases, you can do without fixing these parameters.

At this stage, it is desirable to select the layer in which the bones will be located. I suggest you create a special layer of «Ragdoll» and select it. You can also set the layer after creating the ragdoll.

To create a new layer, you need to click «Layer - Add Layer» at the top of the inspector. And then select the created layer in the ragdoll constructor.

At the bottom of 2.5 figure there is a button «Save Avatar» intended for saving all made settings, separately, in an avatar, for the subsequent use at creation of a ragdoll for other characters.

After making the settings, all you must do is click on the «Create Ragdoll» button to create all the necessary objects and components of the Ragdoll.

Checking the ragdoll and modifying the «Third Person Character» code.

After clicking on the button of creating a ragdoll on the scene, the «ThirdPersonController Container» object is created. This is done in order not to lose the connected character and his ragdoll. They are inside the container. Also created object «Ragdoll», inside which there are all the created bones with components. The difference from the standard Ragdoll is that the Ragdoll exists separately from the character himself, not in his own bones. All of this is shown in figure 2.6.

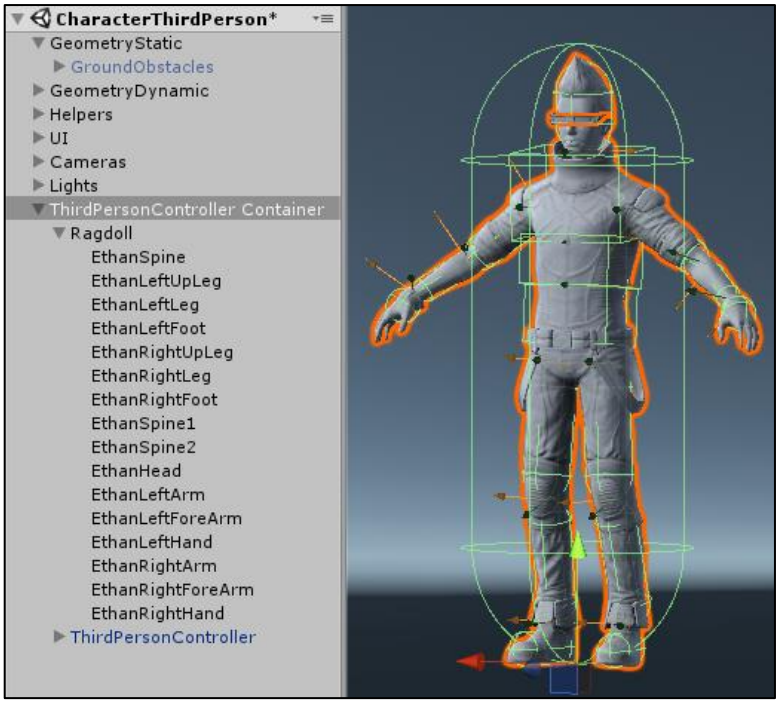


Figure 2.6

In fact, the ragdoll is ready, it can be used. But in this case, «Third Person Character» code has a Raycast ray that bumps into the colliders of the ragdoll. That's why the character sits down because he thinks he's in a limited space. It's very easy to fix. First, you should open the «Third Person Character» code and add the «layerMask» variable as in 2.7 figure.

```
17          [SerializeField] float m_GroundCheckDistance = 0.1f;
18          [SerializeField] LayerMask layerMask;
19
20          Rigidbody m_Rigidbody;
```

Figure 2.7

This variable will be responsible for which layers will be visible to the raycast. There are three uses of the raycast in the code, and you will need to add the «layerMask» parameter, just like in 1.8 figure.

```
93      if (Physics.SphereCast(crouchRay, m_Capsule.radius * k_Half, crouchRayLength, layerMask, QueryTriggerInteraction.Ignore))
111     if (Physics.SphereCast(crouchRay, m_Capsule.radius * k_Half, crouchRayLength, layerMask, QueryTriggerInteraction.Ignore))
212 if (Physics.Raycast(transform.position + (Vector3.up * 0.1f), Vector3.down, out hitInfo, m_GroundCheckDistance, layerMask))
```

Figure 2.8

Now all that remains to be done is to uncheck the checkbox next to the ragdoll layer in the «Third Person Character» component of the inspector.

Component «Ramecan Mixer» performs all the work on mixing the physics of ragdoll and animation mecanim. It is automatically added to the character after the creation of the ragdoll in the constructor. And a constructor component can be removed if you don't need it anymore. But it is necessary to specify that after removal, the ragdoll can be changed only directly, through corresponding components of bones.

By reducing size of the collider of the character, you can achieve the interaction of the ragdoll with the surrounding objects, as in figure 2.9.

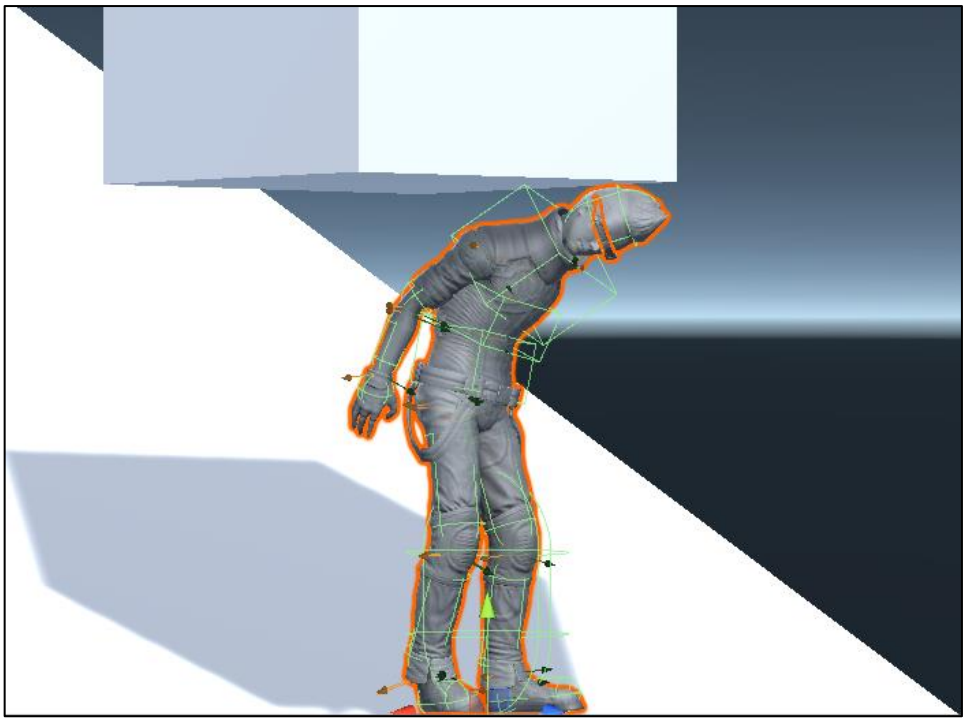


Figure 2.9

WORKING WITH RAGDOLL STATES IN RAMECAN MIXER

The «Ramecan Mixer» component is a form for working with bone states. As it is possible to see in figure 3.1, in the top part there is a possibility to add, remove and change states, and a little bit more low there is a hierarchy of bones of a ragdoll. And when you click on the bone, the current state of the bone parameters appears at the very bottom. You can choose several bones and change them simultaneously.

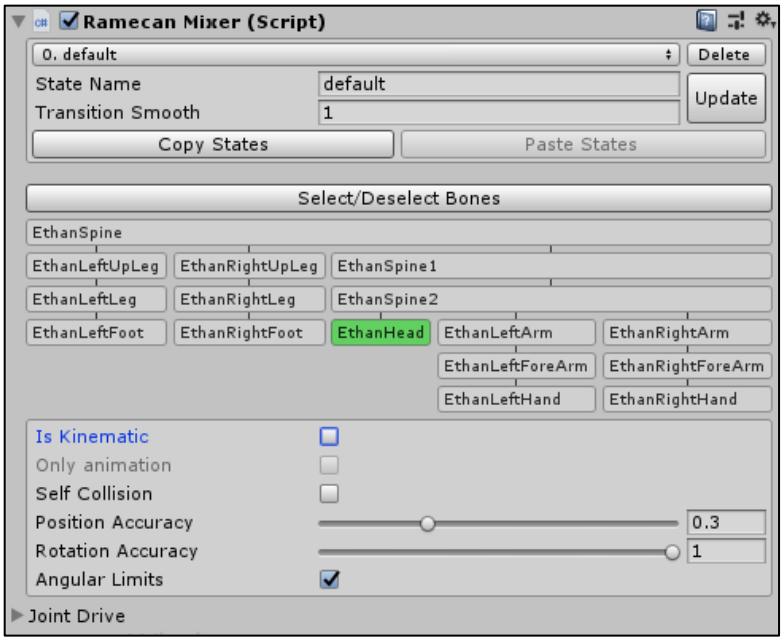


Figure 3.1

When states change, the bone parameters change according to the selected state. But first you need to add a new one. To do this, simply enter its name, for example, «dead». Then the button next to it will change to «Add» and you need to press it. Okay. The next parameter is the smooth state change. The lower the value, the faster and more sharper the transition will be. If the value is «0», the transition will be instantaneous.

When dead is selected, press the «Select/Deselect Bones» button to select all bones. You will see that the parameters associated with rotation have become inactive. This is because the root bone is highlighted and there is no joint on it. If you click on the root bone, it will be deselected, and the joints will become active. Here you need to change «Rotation Accuracy» to 0.1. Now select all the bones again, and change «Position Accuracy» to 0, so that the character is not standing vertically in space. You can also enable «Self Collision». To save the state, press the «Update» button.

If the scene is started, the selected state is automatically activated. You can try to change the state and see how the character reacts. If you change the state manually, it changes instantly, without a smooth transition. We will change the state through the code, it will be just as smooth as necessary.

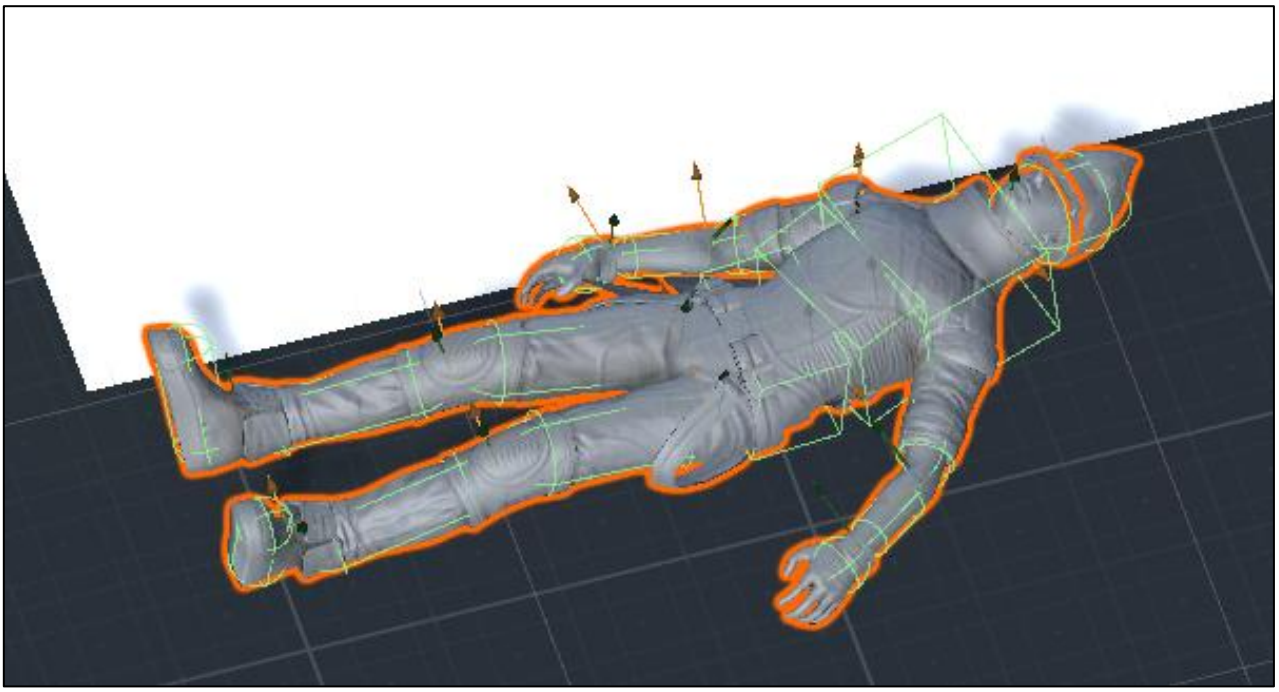
Let's make a simple script that will change the states by pressing the button. To change the state, you only need one method – «BeginStateTransition». You need to enter the name of the required state or an id number into it.

```
using RagdollMecanimMixer;
using UnityEngine;

public class ChangeStateExample : MonoBehaviour {

    private RamecanMixer ramecanMixer;

        void Start () {
        ramecanMixer = GetComponent<RamecanMixer>();
        }

        void Update () {
        if (Input.GetKeyDown(KeyCode.UpArrow))
            ramecanMixer.BeginStateTransition("default");

        if (Input.GetKeyDown(KeyCode.DownArrow))
            ramecanMixer.BeginStateTransition("dead");
    }
}
```

All we need to do is add this script to the character and run the scene. Now, when you click on the up and down arrows will change the state.

This was an elementary example of the use of ragdoll states. With it, you can create the kind of ragdoll behavior you need. The Asset also includes a demo scene that shows you one of the possible uses.

I wish you a successful and pleasant use of "Ragdoll Mecanim Mixer"!



IMPORTANT NOTES

1. **Culling Mode** parameter of character's Animator must be set to **Always Animate**. In this case animation will play even if the character is outside the field of view of the camera.

Figure 4.1

2. In some cases, you may need to use different layers for the character and his ragdoll. A ragdoll layer can be assigned in advance. **Ragdoll Constructor** component has a corresponding parameter.

Note that after importing the asset, the character and his ragdoll bones have empty layers. This is because layers are not exportable.

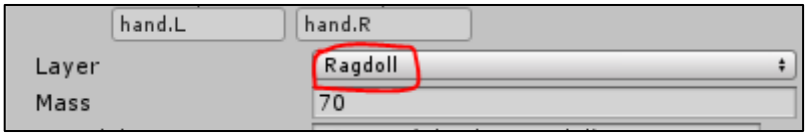


Figure 4.2

3. To move a character, you should move the character's main object, not its container. When creating a ragdoll, the constructor creates a container for the character and its ragdoll, as in figure 4.3. Ragdoll is located separately to improve physical parameters. A container is created in order not to lose related objects.

Figure 4.3