Ragdoll dismemberment system manual

Summary

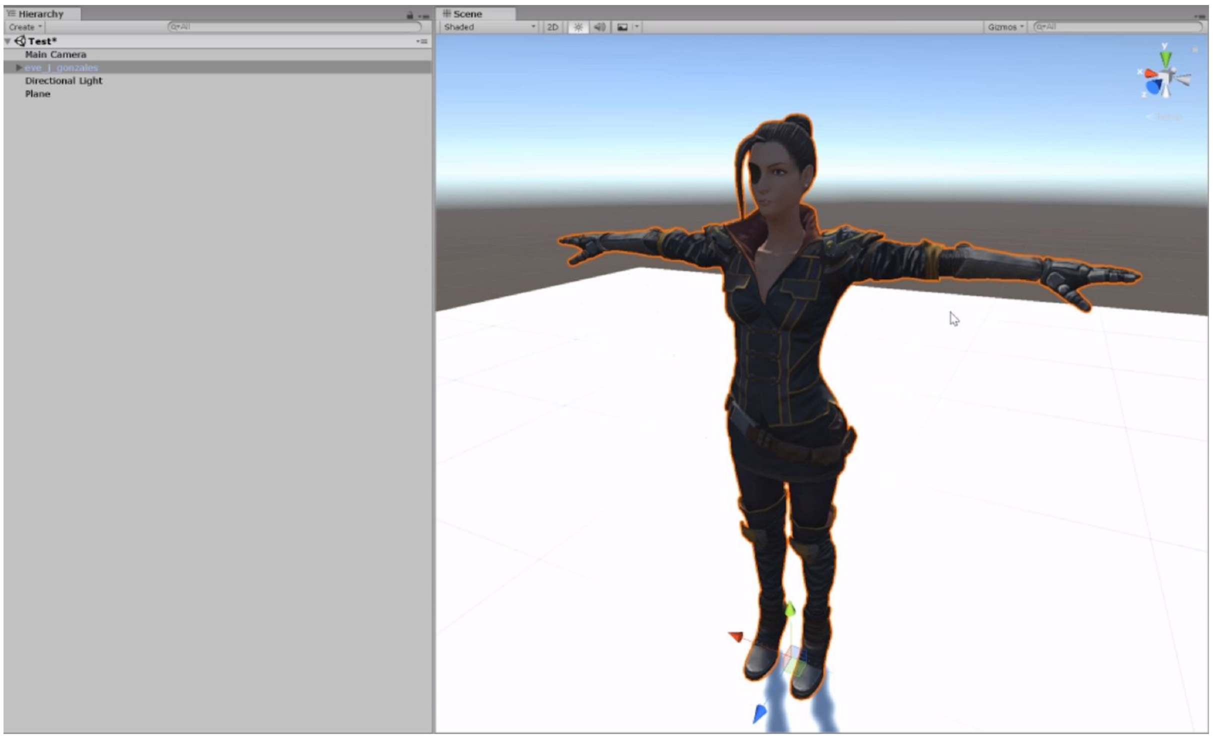
This system allows to dismember any skinned mesh without additional preparation in external 3D software.
Features:

- This system is suitable for games with suit customization (when character consists from several skinned meshes, that can be equipped/unequipped in runtime).
- LOD group auto-duplication for fragments (if skinned mesh has LOD group)
- Hierarchycal dismemberment (a dismembered arm can be dismembered again in elbow joint)
- Supports any kind of ragdoll (human, robot, animal etc.)
- Effects customization (broken electronics for robots, green meat for aliens etc.)

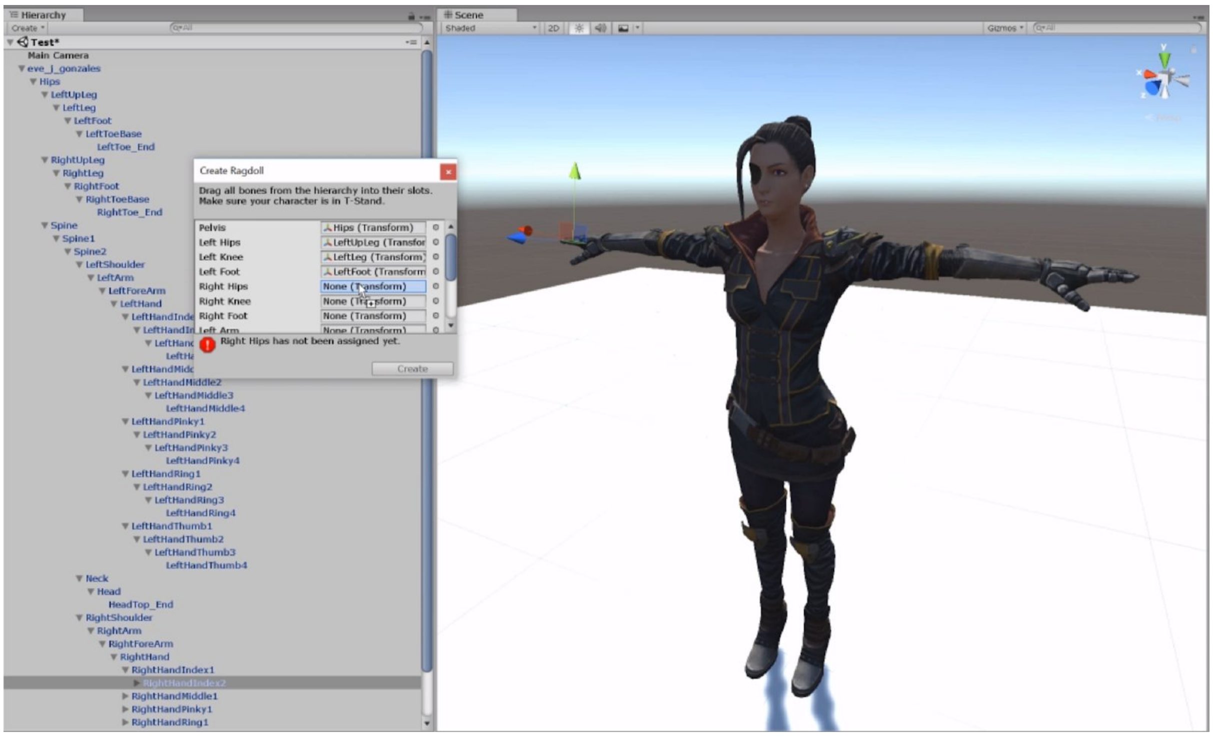
Demo
Video tutorial
If you have questions, please contact us info@animus.digital
More info on our official site animus.digital

How to configure ragdoll for dismemberment

1. Import skinned mesh FBX file to Unity and drag it to the scene.



2. Make a ragdoll from skinned mesh (for example, using built-in Unity ragdoll wizard).



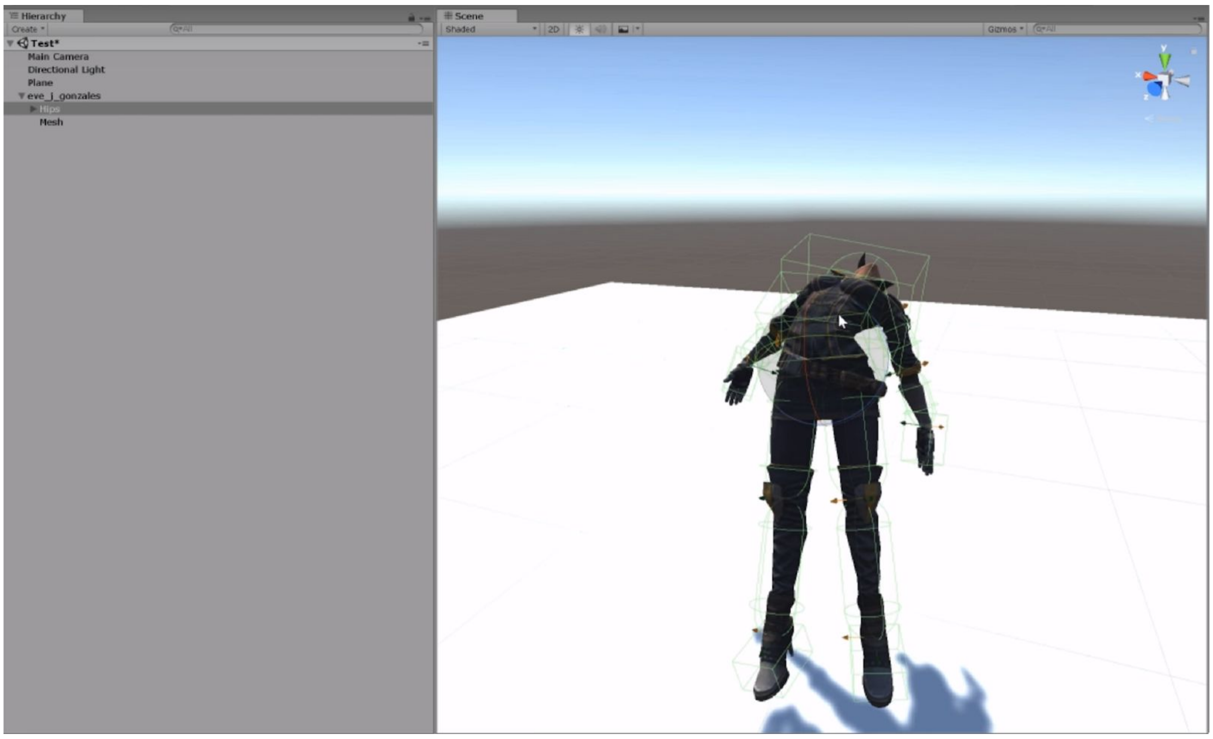
Create Ragdoll

Drag all bones from the hierarchy into their slots.
Make sure your character is in T-Stand.

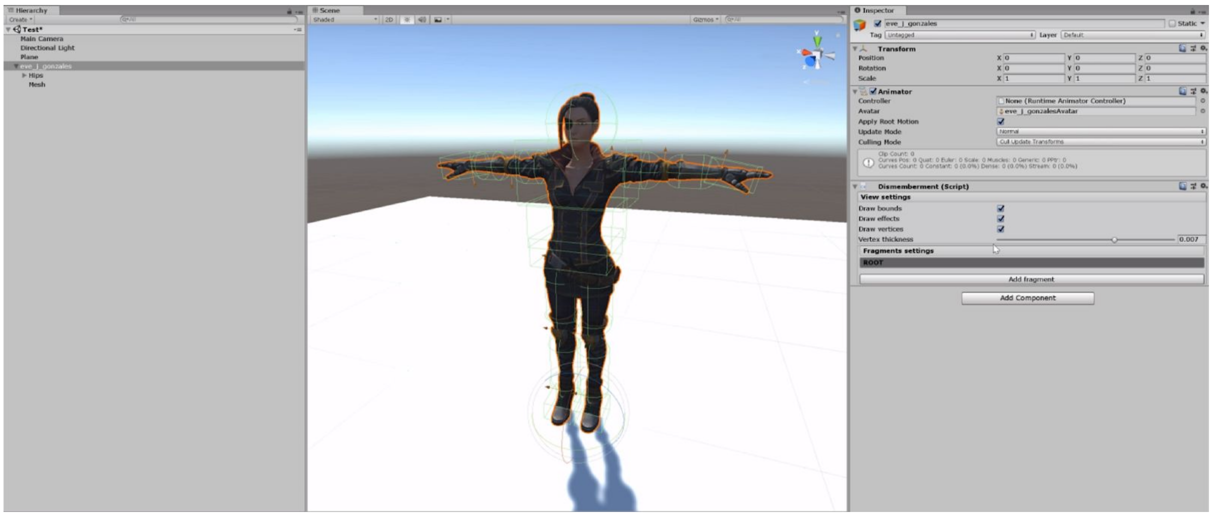
Pelvis	Hips (Transform)
Left Hips	LeftUpLeg (Transfor
Left Knee	LeftLeg (Transform)
Left Foot	LeftFoot (Transform
Right Hips	None (Transform)
Right Knee	None (Transform)
Right Foot	None (Transform)
Left Arm	None (Transform)

Right Hips has not been assigned yet.

Create

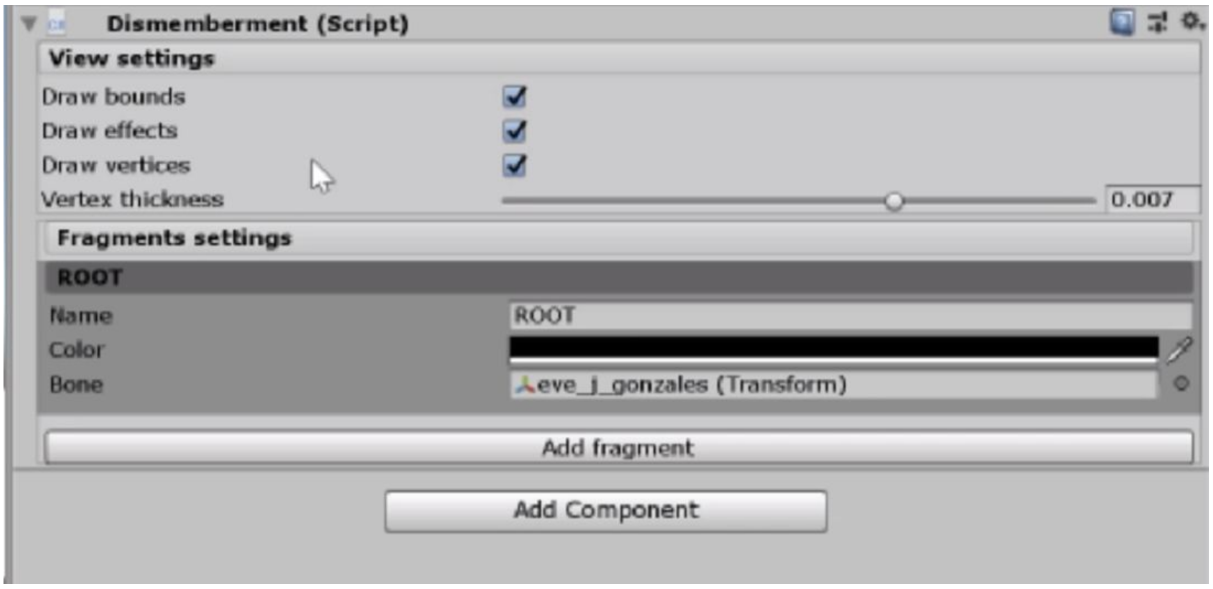


3. Attach **RagdollDismembermentVisual** component to root gameobject of skinned mesh



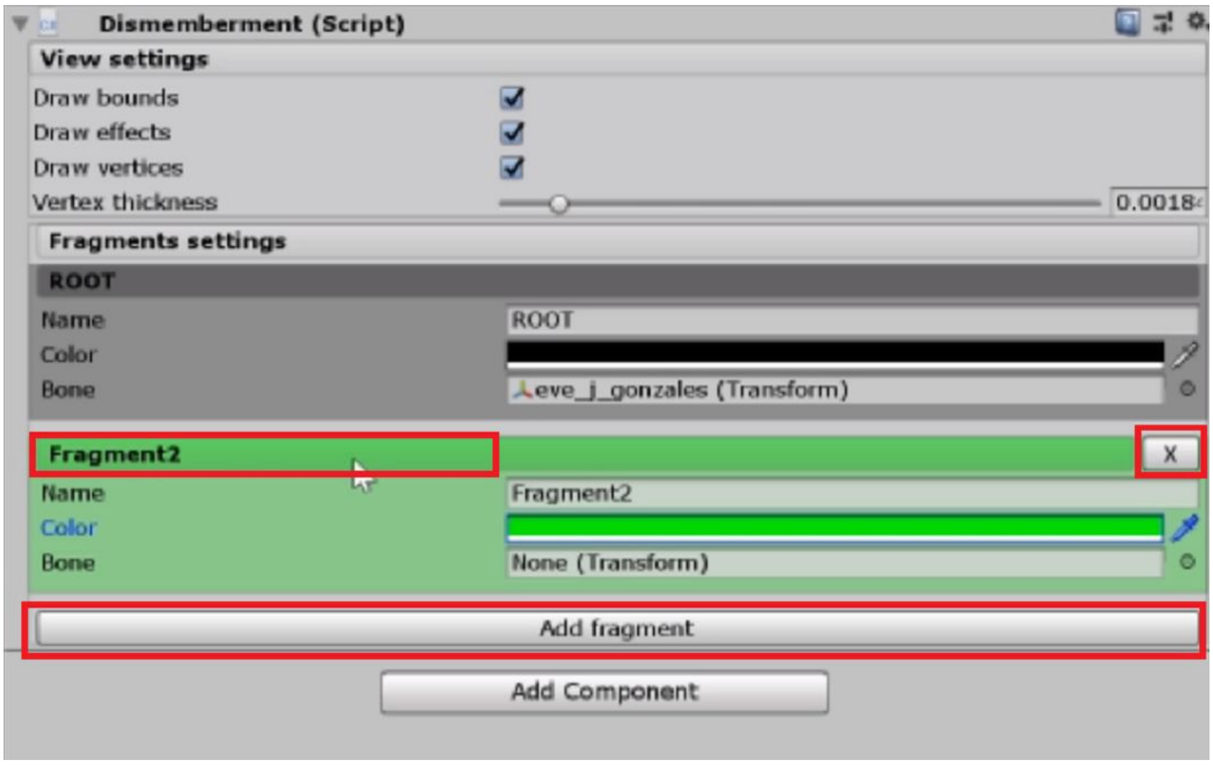
```
eve_j_gonzales                         Static
Tag Untagged          Layer Default
Transform
Position        X 0    Y 0    Z 0
Rotation        X 0    Y 0    Z 0
Scale           X 1    Y 1    Z 1
Animator
Controller        None (Runtime Animator Controller)
Avatar            eve_j_gonzalesAvatar
Apply Root Motion ✓
Update Mode       Normal
Culling Mode      Cull Update Transforms
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%)
Dismemberment (Script)
View settings
Draw bounds       ✓
Draw effects      ✓
Draw vertices     ✓
Vertex thickness  0.007
Fragments settings
ROOT
Add fragment
Add Component
```

4. Now take a look on dismemberment editor window. All settings are divided into View settings and Fragments settings. View settings allows to change visual representation of the editor. Fragments settings is the main editor part and it is responsible for all fragment properties (like bounds, effects etc).



```
Dismemberment (Script)
View settings
Draw bounds       ✓
Draw effects      ✓
Draw vertices     ✓
Vertex thickness  0.007
Fragments settings
ROOT
Name    ROOT
Color
Bone    eve_j_gonzales (Transform)
Add fragment
Add Component
```

5. The root fragment is created by default and covers entire mesh. Only name, color and bone transform can be edited. Root fragment can't be deleted.
6. To add fragment, press "Add fragment" button. To remove fragment, press "X" button in top-right corner of fragment panel. Click on the name of fragment to show/hide it's properties.


Dismemberment (Script)

View settings

Draw bounds	✓
Draw effects	✓
Draw vertices	✓
Vertex thickness	0.0018

Fragments settings

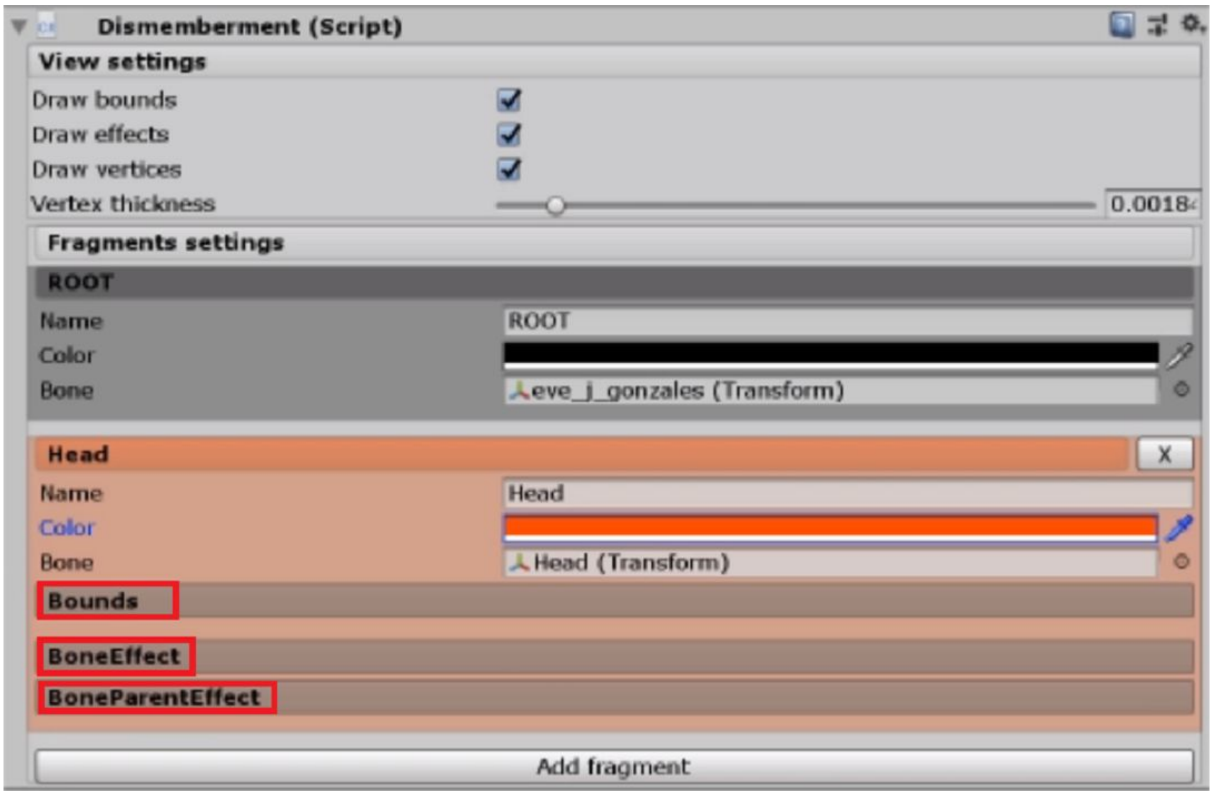
ROOT

Name	ROOT
Color	■
Bone	eve_j_gonzales (Transform)

Fragment2

Name	Fragment2
Color	🟩
Bone	None (Transform)

7. When bone transform is assigned, bounds and effects' subpanels will appear.


Dismemberment (Script)

View settings

Draw bounds	✓
Draw effects	✓
Draw vertices	✓
Vertex thickness	0.0018

Fragments settings

ROOT

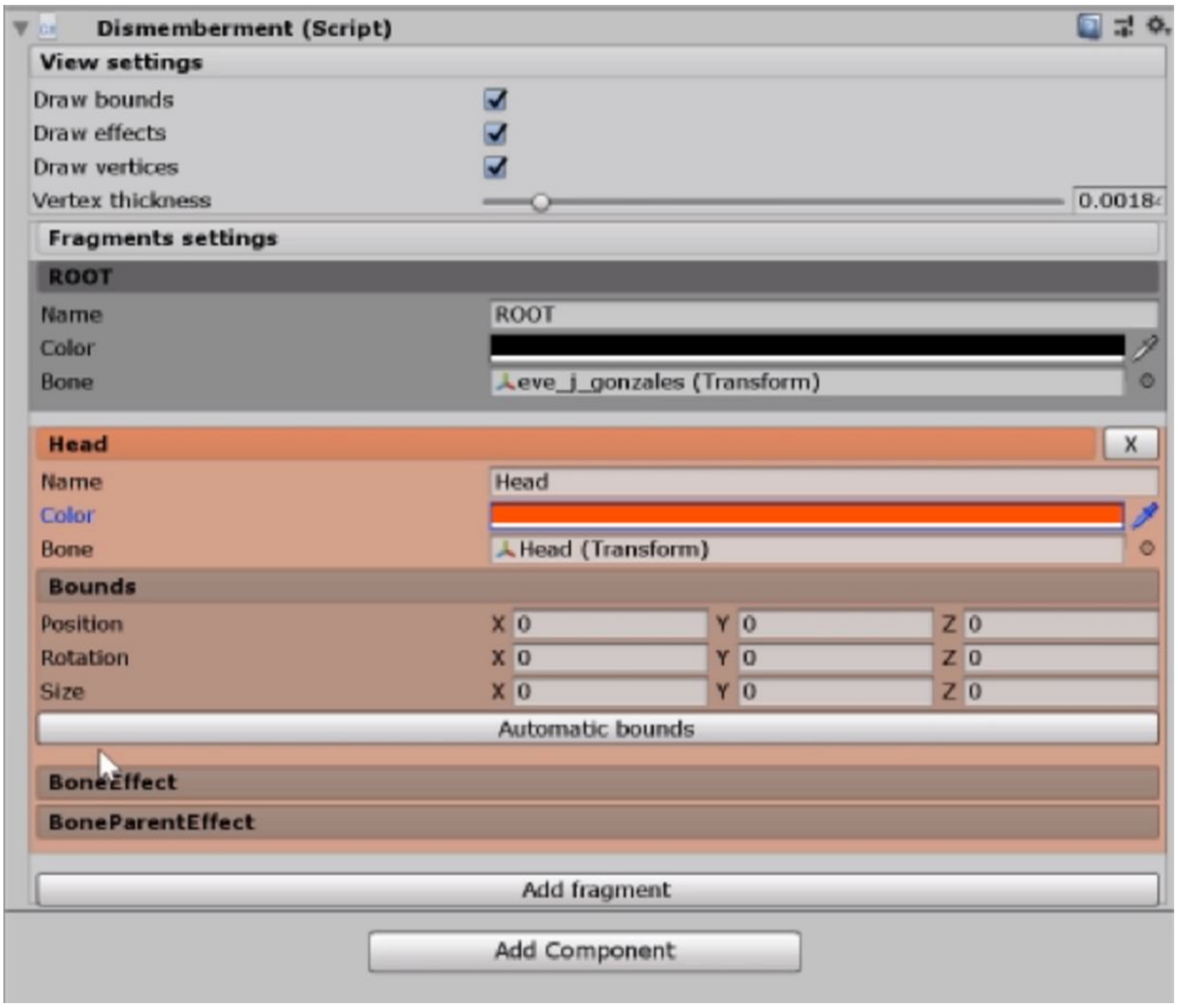
Name	ROOT
Color	■
Bone	eve_j_gonzales (Transform)

Head

Name	Head
Color	🟧
Bone	Head (Transform)

8. In bounds subpanel, there are position, rotation and scale parameters. Click on "Automatic bounds" button, and system will try to calculate bounds, which will encapsulate collider, attached to

bone transform.



Dismemberment (Script)

View settings

Draw bounds	✓
Draw effects	✓
Draw vertices	✓
Vertex thickness	0.0018

Fragments settings

ROOT

Name	ROOT
Color	■
Bone	eve_j_gonzales (Transform)

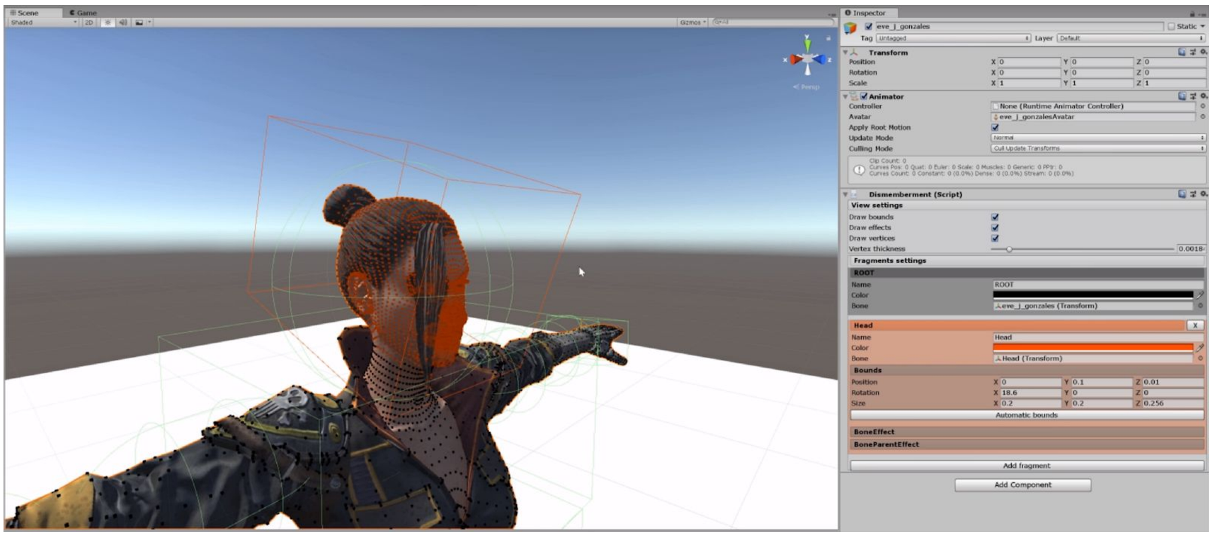
Head

Name	Head
Color	🟧
Bone	Head (Transform)

Bounds

	X	Y	Z
Position	0	0	0
Rotation	0	0	0
Size	0	0	0

9. Fragment bounds must include all vertices, which will belong to dismembered fragment. Also, for better effect matching, we recommend to orient bounds along slice edge loop. For example, head bounds will look something like this



eve_j_gonzales

Tag: Untagged Layer: Default

Transform

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

Animator

Controller	None (Runtime Animator Controller)
Avatar	eve_j_gonzalesAvatar
Apply Root Motion	✓
Update Mode	Normal
Culling Mode	Cull Update Transforms

```
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%)
```

Dismemberment (Script)

View settings

Draw bounds	✓
Draw effects	✓
Draw vertices	✓
Vertex thickness	0.0018

Fragments settings

ROOT

Name	ROOT
Color	■
Bone	eve_j_gonzales (Transform)

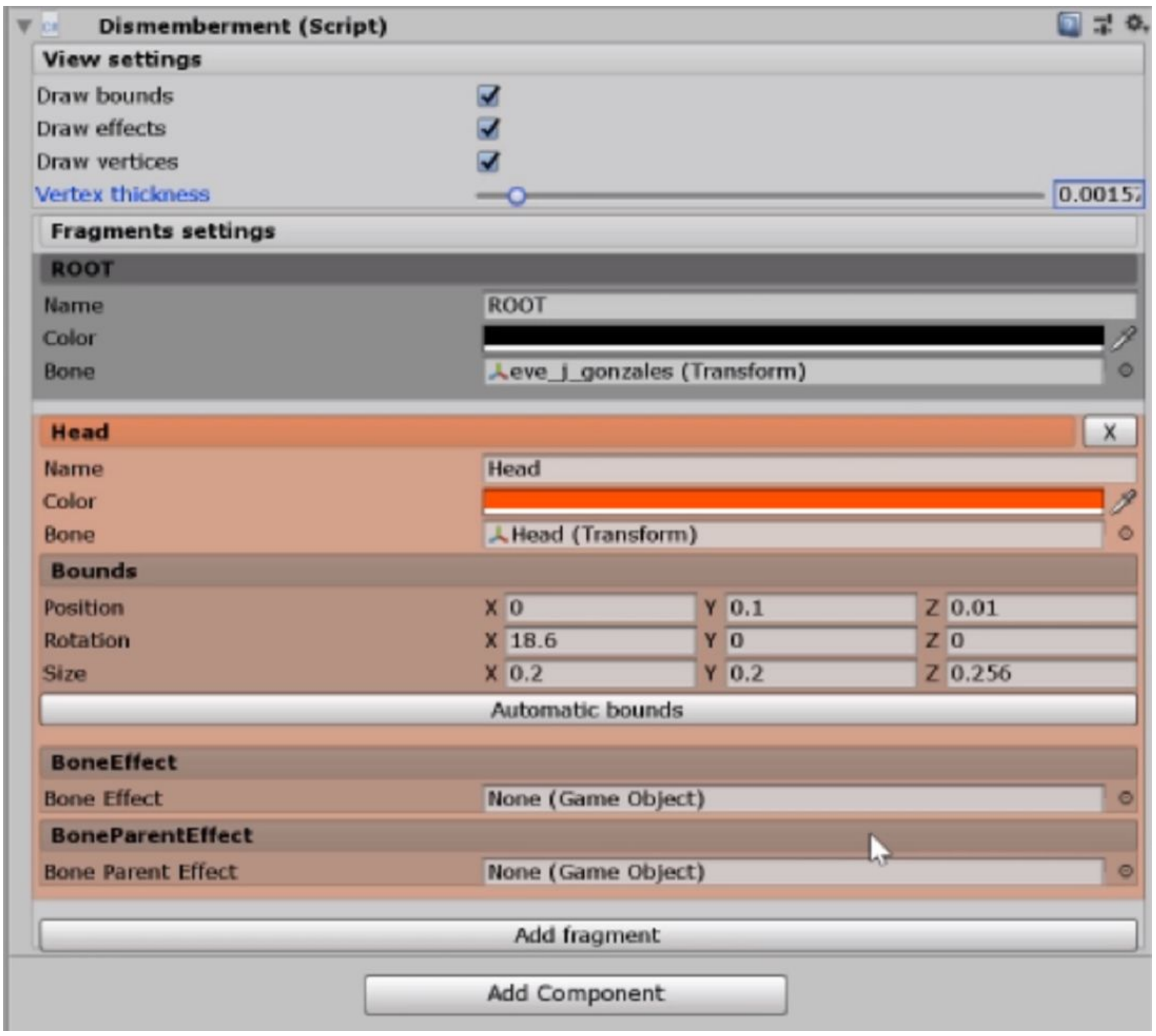
Head

Name	Head
Color	🟧
Bone	Head (Transform)

Bounds

	X	Y	Z
Position	0	0.1	0.01
Rotation	18.6	0	0
Size	0.2	0.2	0.256

10. Every fragment has slot for bone effect and slot for parent bone effect. Effects are needed to fill gaps in sliced meshes.



Dismemberment (Script)

View settings

Draw bounds	✓
Draw effects	✓
Draw vertices	✓
Vertex thickness	0.0015;

Fragments settings

ROOT

Name	ROOT
Color	■
Bone	eve_j_gonzales (Transform)

Head

Name	Head
Color	🟧
Bone	Head (Transform)

Bounds

	X	Y	Z
Position	0	0.1	0.01
Rotation	18.6	0	0
Size	0.2	0.2	0.256

BoneEffect

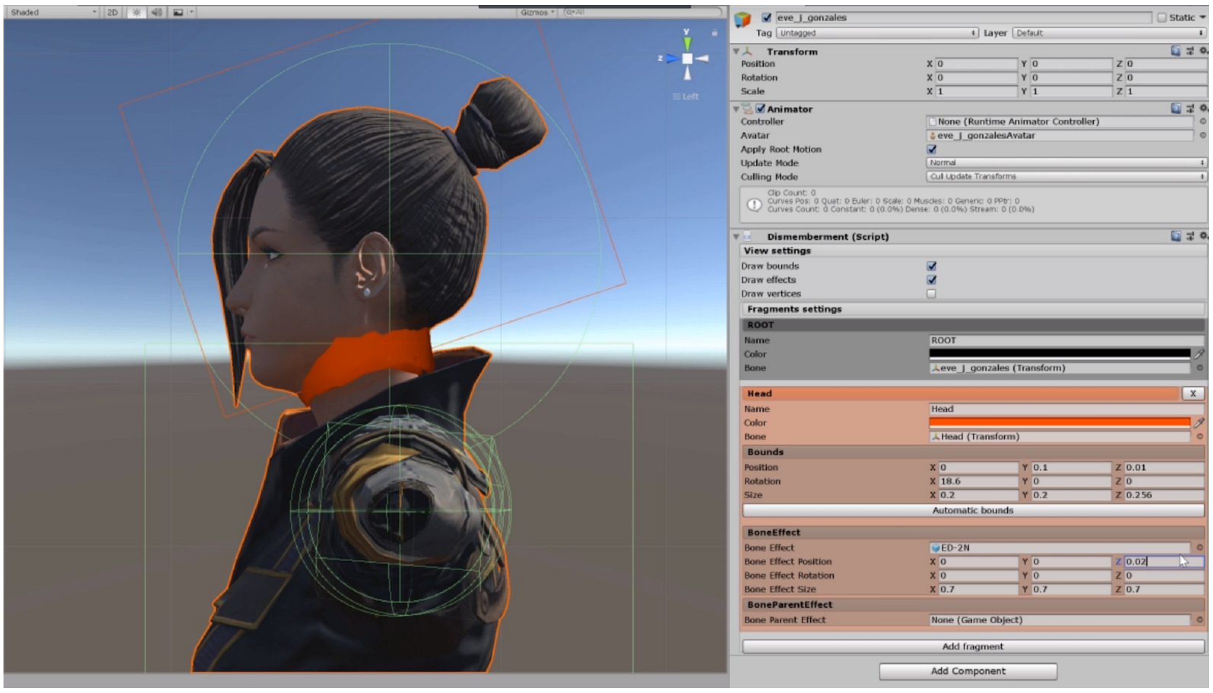
Bone Effect	None (Game Object)

BoneParentEffect

Bone Parent Effect	None (Game Object)

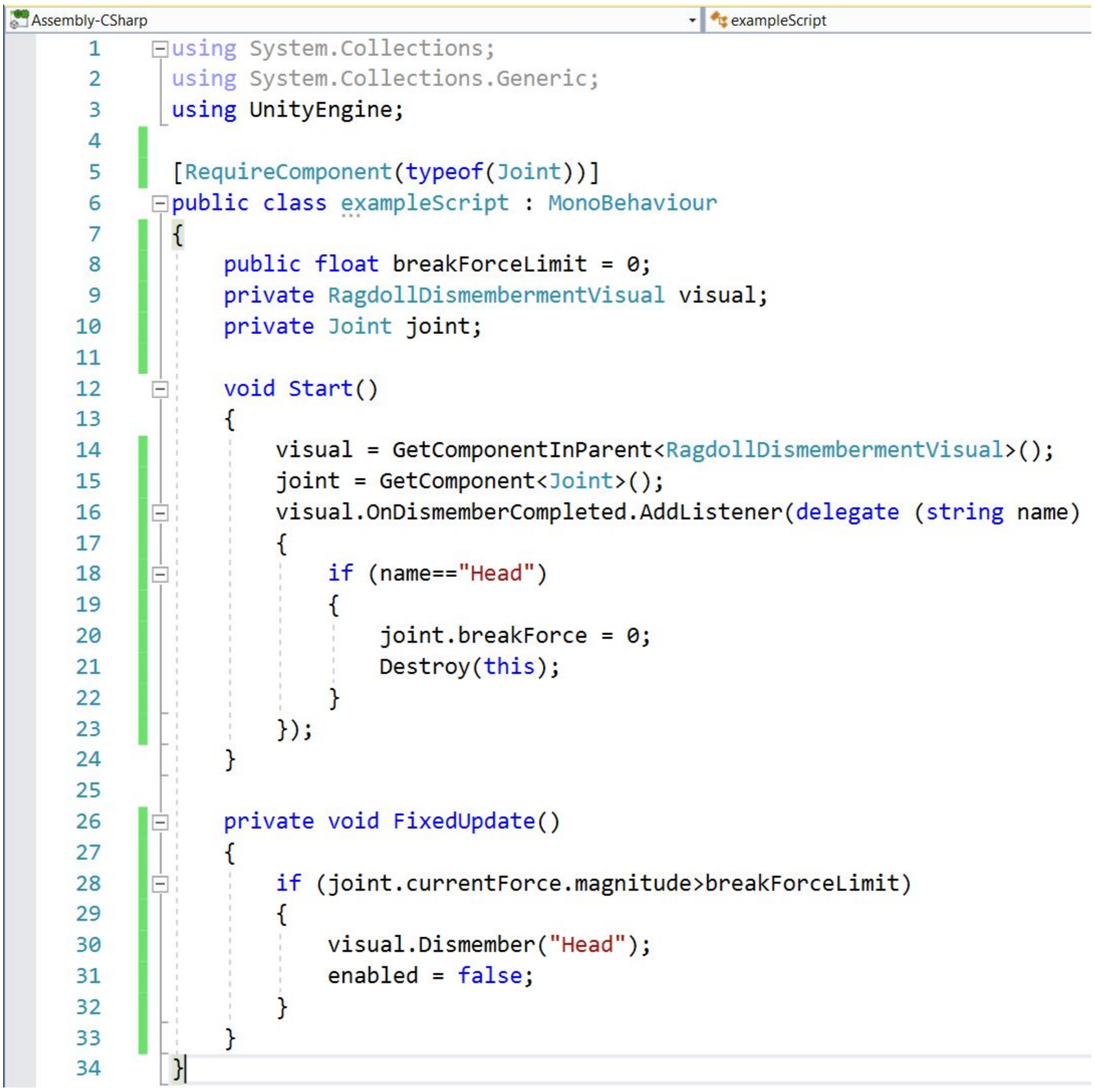
11. When effect is assigned, position, rotation and scale parameters will become editable. An effect will be displayed as mesh with the

same color as corresponding fragment.



```
eve_j_gonzales                                   Static
Tag  Untagged                 Layer  Default
Transform
Position            X 0        Y 0        Z 0
Rotation            X 0        Y 0        Z 0
Scale               X 1        Y 1        Z 1
Animator
Controller          None (Runtime Animator Controller)
Avatar              eve_j_gonzalesAvatar
Apply Root Motion   ✓
Update Mode         Normal
Culling Mode        Cull Update Transforms
  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%)
Dismemberment (Script)
View settings
Draw bounds         ✓
Draw effects        ✓
Draw vertices       ☐
Fragments settings
ROOT
Name                ROOT
Color               ■
Bone                eve_j_gonzales (Transform)
Head                                                 X
Name                Head
Color               🟧
Bone                Head (Transform)
Bounds
Position            X 0        Y 0.1      Z 0.01
Rotation            X 18.6     Y 0        Z 0
Size                X 0.2      Y 0.2      Z 0.256
                 Automatic bounds
BoneEffect
Bone Effect         ED-2N
Bone Effect Position   X 0     Y 0        Z 0.02
Bone Effect Rotation   X 0     Y 0        Z 0
Bone Effect Size       X 0.7   Y 0.7      Z 0.7
BoneParentEffect
Bone Parent Effect  None (Game Object)
                 Add fragment
                 Add Component
```

12. After fragment bounds and effects are configured, it's time to look at scripting part. To dismember fragment, call **void Dismember(string name)** method from Dismemberment component. Dismember operation is performing asyncronously. If you need to perform some logic after dismember operation, then add listener to **UnityEvent<string> OnDismemberCompleted** event.
For example, a script below is attached to head and checks joint currentForce in FixedUpdate. If currentForce exceeds certain limit, script calls **Dismember(**“Head”**)**, which starts dismember operation of fragment named Head. When operation of head dismemberment is done, **OnDismemberCompleted** event will be invoked with string parameter “Head”, which indicates that operation of head dismemberment is completed. Event listener of example script performs check, whether dismembered fragment is Head. If it’s true, then script breaks joint (by setting breakForce to zero), and destroys itself.



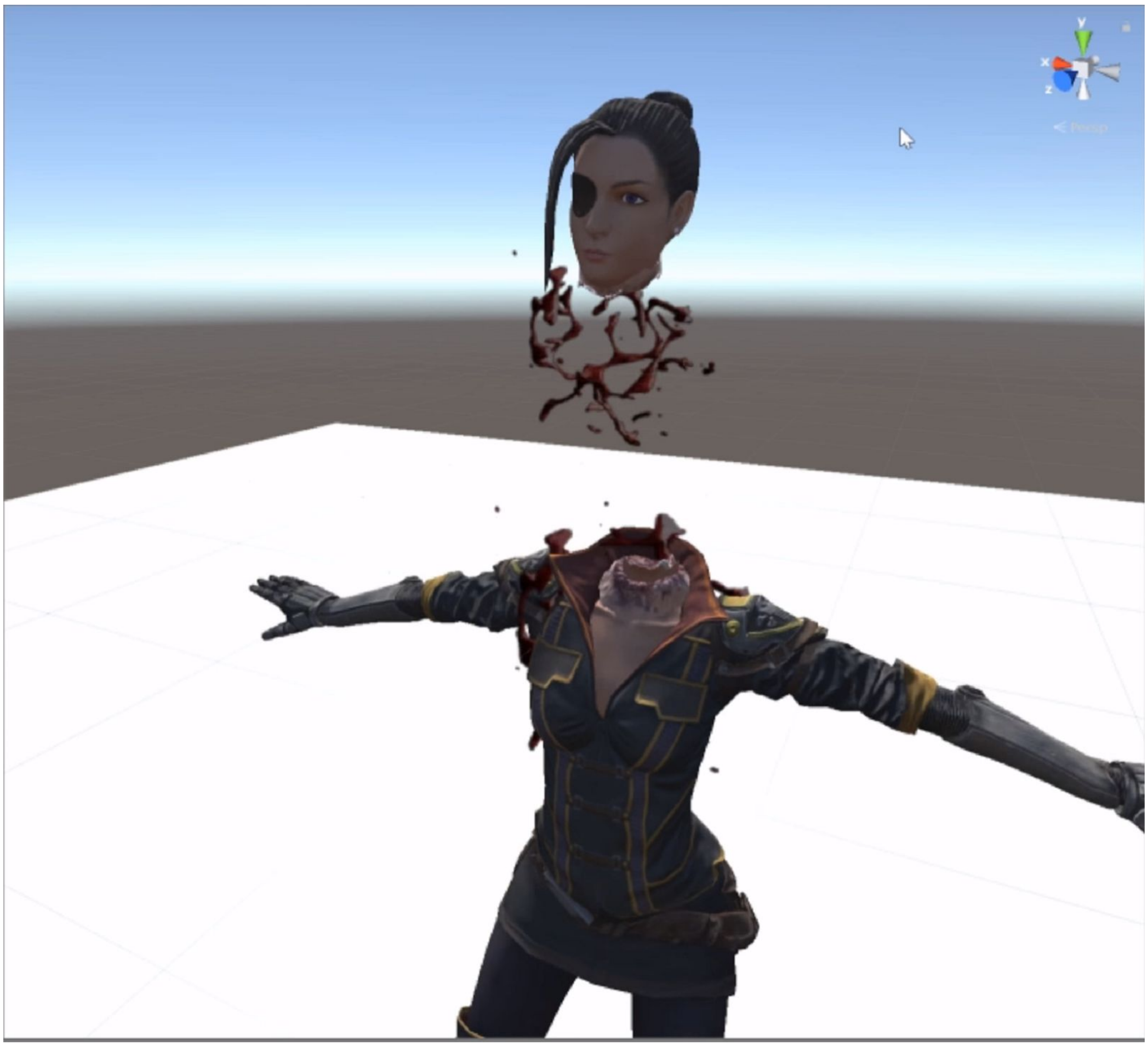
```csharp
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

[RequireComponent(typeof(Joint))]
public class exampleScript : MonoBehaviour
{
    public float breakForceLimit = 0;
    private RagdollDismembermentVisual visual;
    private Joint joint;

    void Start()
    {
        visual = GetComponentInParent<RagdollDismembermentVisual>();
        joint = GetComponent<Joint>();
        visual.OnDismemberCompleted.AddListener(delegate (string name)
        {
            if (name=="Head")
            {
                joint.breakForce = 0;
                Destroy(this);
            }
        });
    }

    private void FixedUpdate()
    {
        if (joint.currentForce.magnitude>breakForceLimit)
        {
            visual.Dismember("Head");
            enabled = false;
        }
    }
}
```

13. That's all. When everything is configured, a dismemberment will look something like this



Tips and tricks

1. Effects can be mirrored by simply changing scale sign on negative. It allows to make effects only for one side of a character.
2. It is recommended to fragment mesh corresponding to ragdoll bodyparts, as it allows to associate dismembering with OnJointBreak event. It is also possible to join several bodyparts

into one fragment (for example LowerArm fragment can include lower arm and hand colliders).