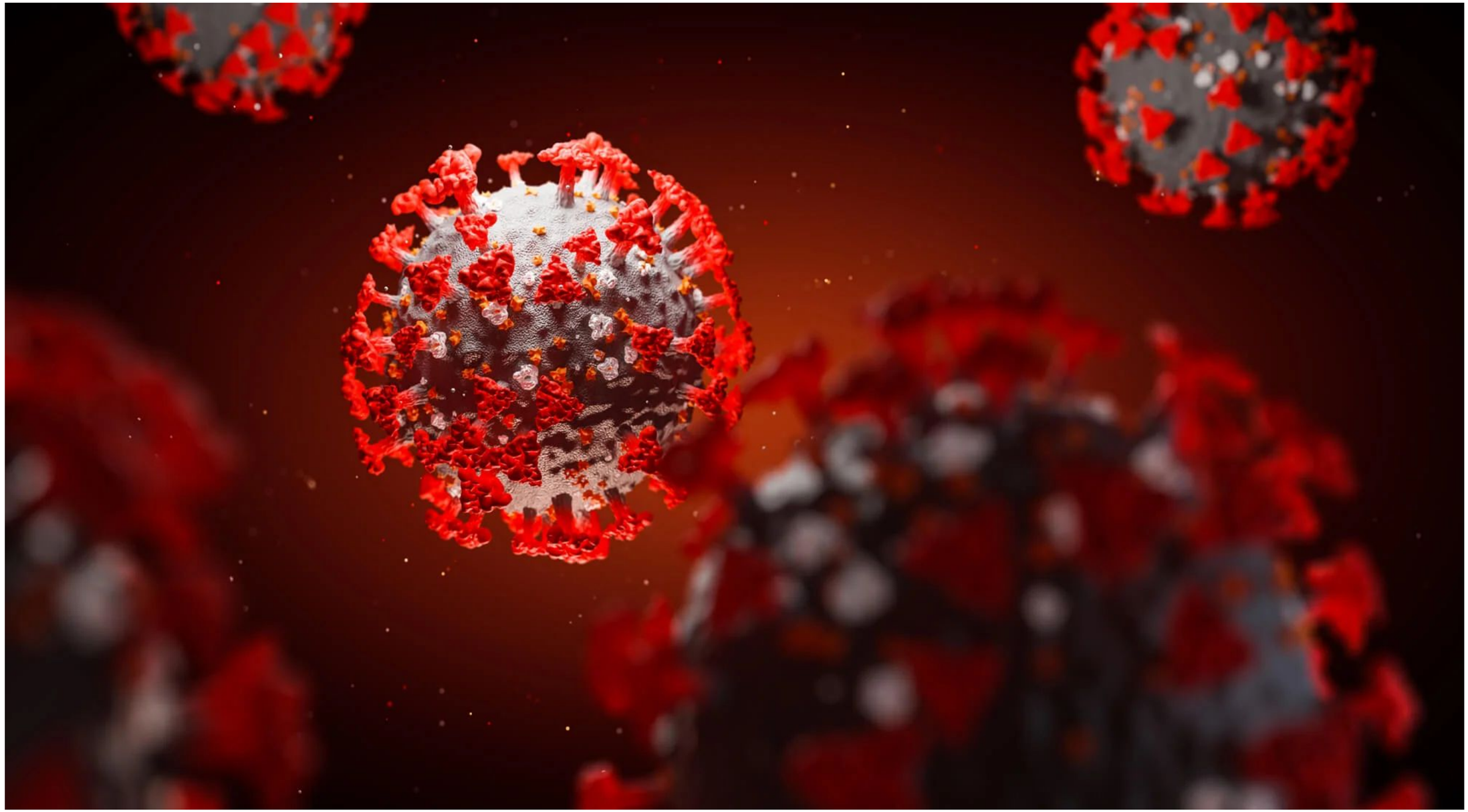
VR Post-Covid, Where to start ? OpenXR & Unity Input

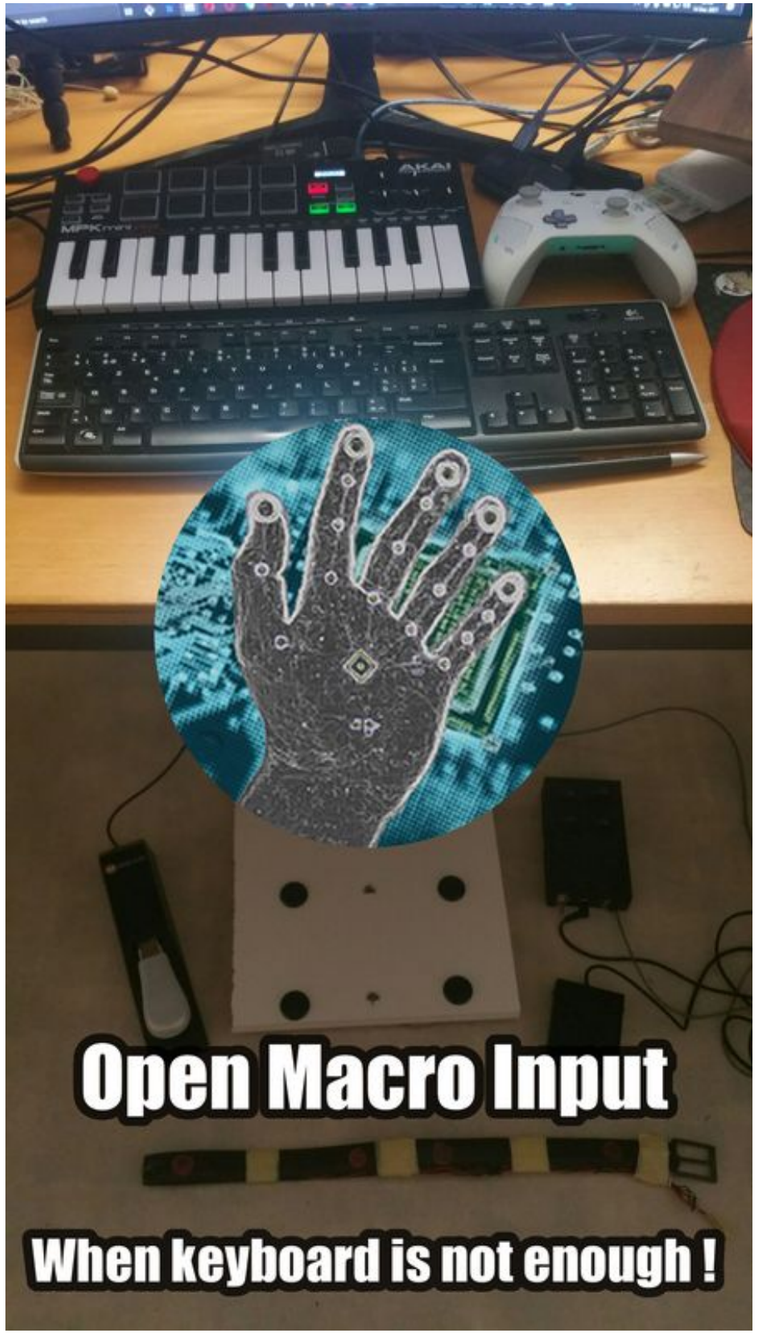

GitHub
SCAN ME
Find all the link, code and, source here:
https://github.com/EloiStree/2022_06_18_HackInTheWood_Topic_LynxAndOpenXR
HACK IN THE WOODS

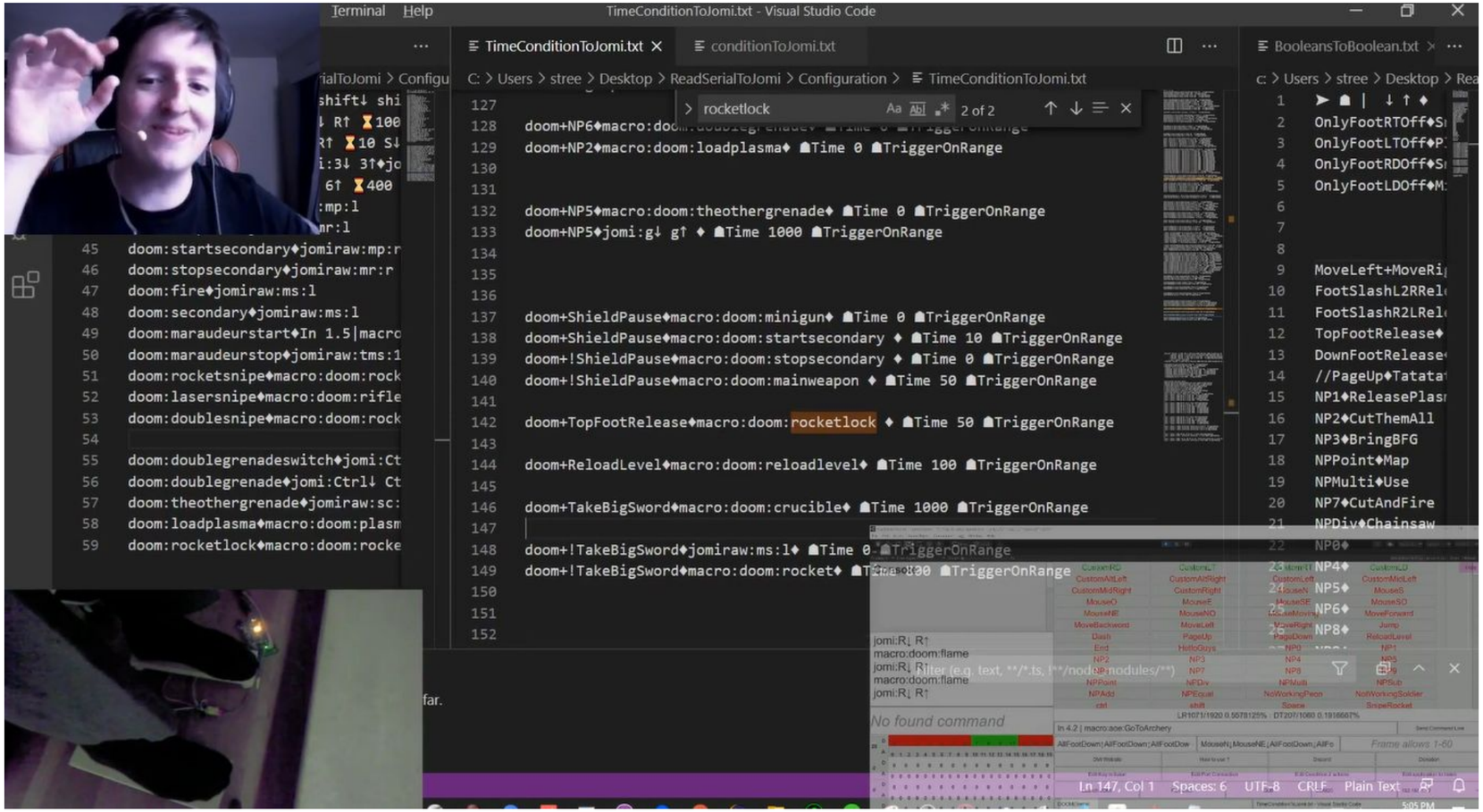
Disclaimer

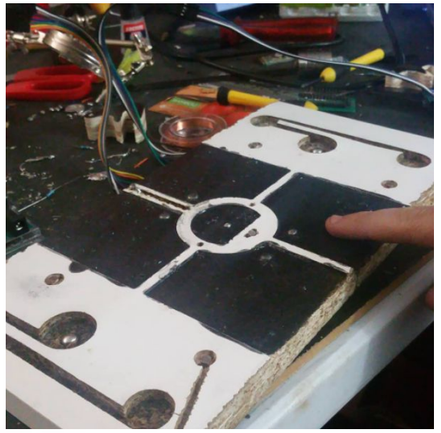
HACK IN THE WOODS



HACK IN THE WOODS







HACK IN THE WOODS


One device to rule them all, one device to find them,
One device to bring them all, and in the adsness bind them;
In the Land of Meta where the shadows lie.


AND NOW
WE WAIT


friday
saturday

AND NOW...
... WE WAIT
memegenerator.net

STILL WAITING

<🌲/>
HACK IN THE
WOODS

Oculus App Lab | Game List

Add your Game

Best | New | Trending | Free | Sale

Gorilla Tag
Action, Casual, Social
★★★★½ (19993) FREE

Gym Class - Basketball
Simulation, Social, Sports
★★★★★ (12660) FREE

Sport Mode
Action, Shooter, Simulation
★★★★½ (4688) $9.99

Pavlov Shack Beta
Action, Shooter, Social
★★★★½ (7352) FREE

Deisim
Casual, Simulation, Strategy

iB Cricket - App Lab
Casual, Simulation, Sport

Frenzy VR
Action, Fighting, Shooter

PowerBeatsVR
Action, Music, Sport

SIDEQUEST

Search games...

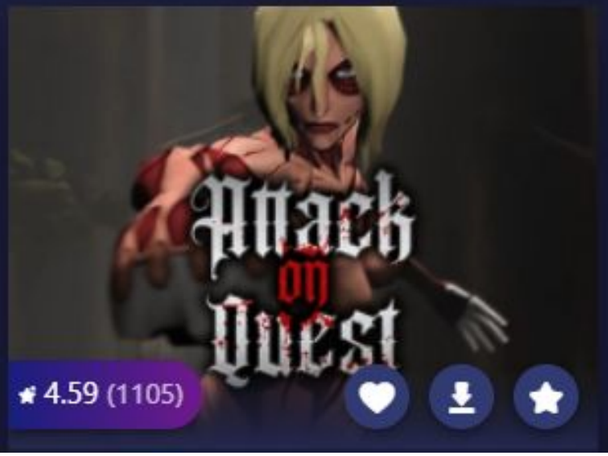
SIGN IN SIGN UP

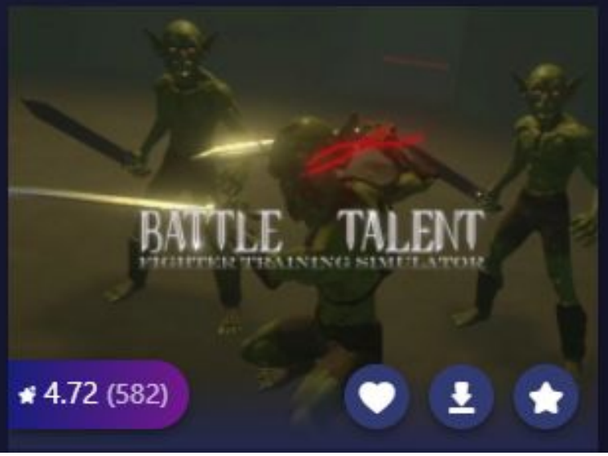
HOME GAMES SUPPORT PROMOTE GET SIDEQUEST GIVEAWAYS

Browse Categories

category view | all apps | adventure | app lab | building | climbing | combat | custom homes | early access | educational | escape | flying | fitness | fps | ga

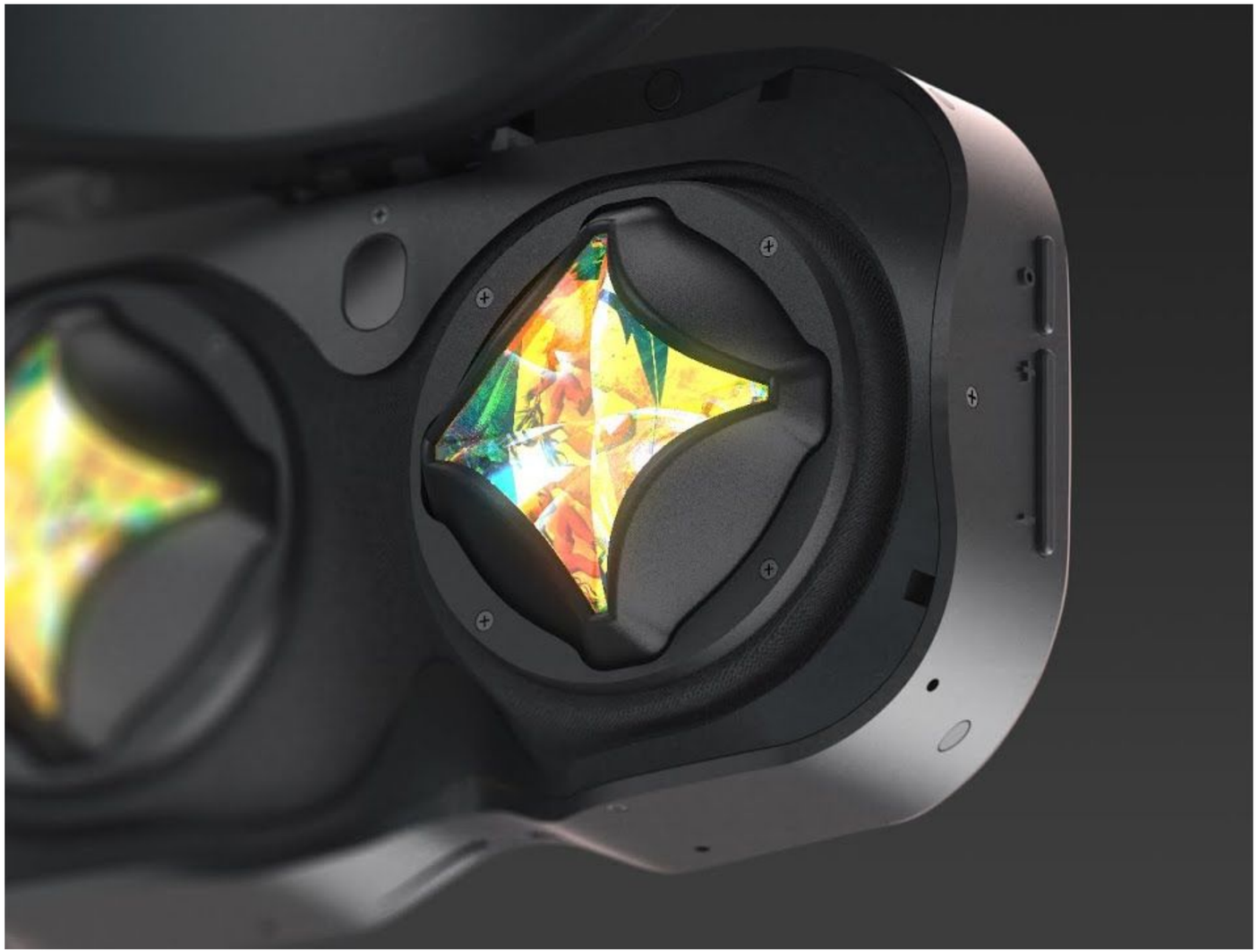
Top Hot New Updated



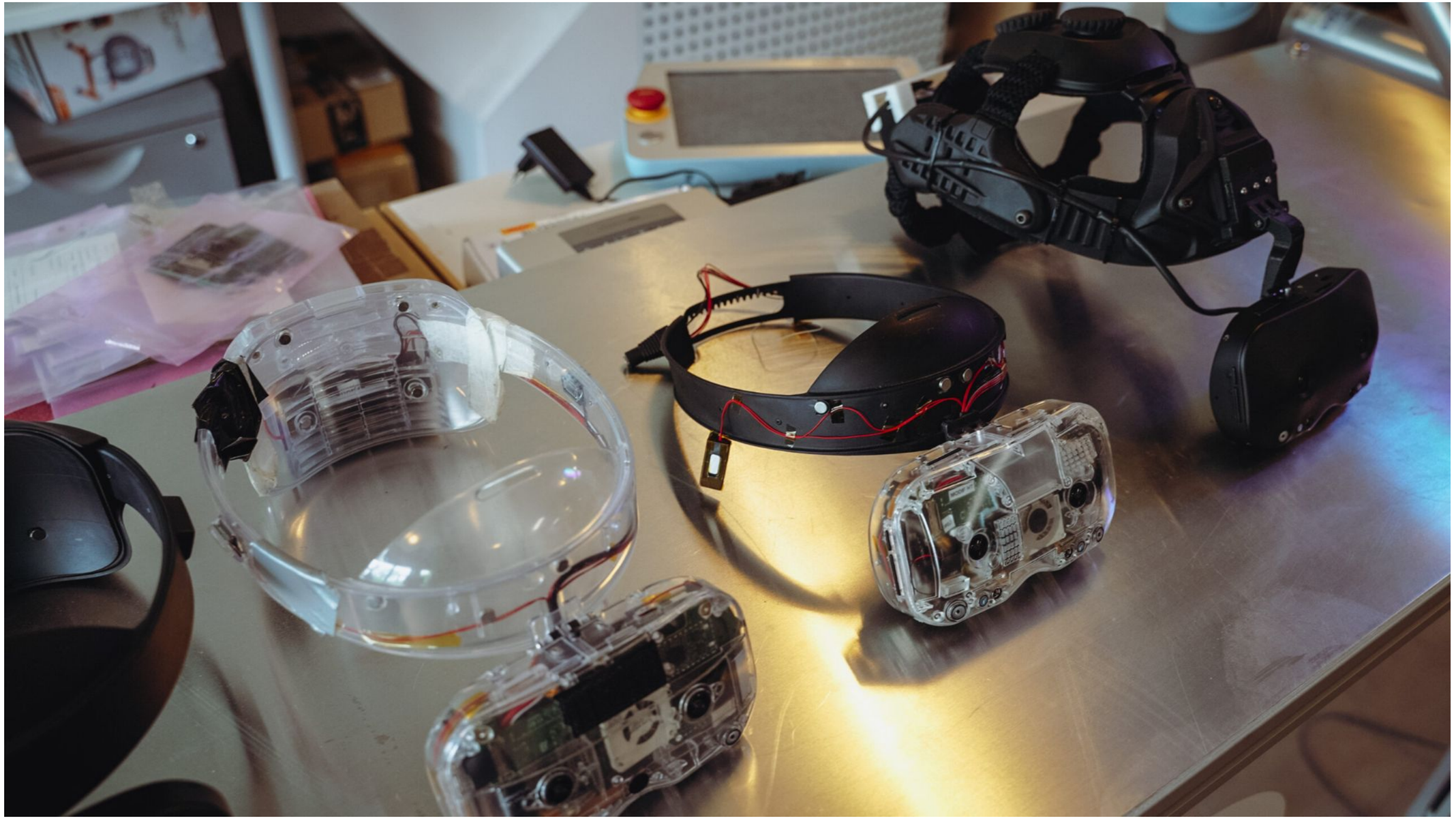


Look to my coming on the first light of the september month, at dawn look to the east.

Stan Larroque & the Lynx R1



<
/>
HACK IN THE
WOODS



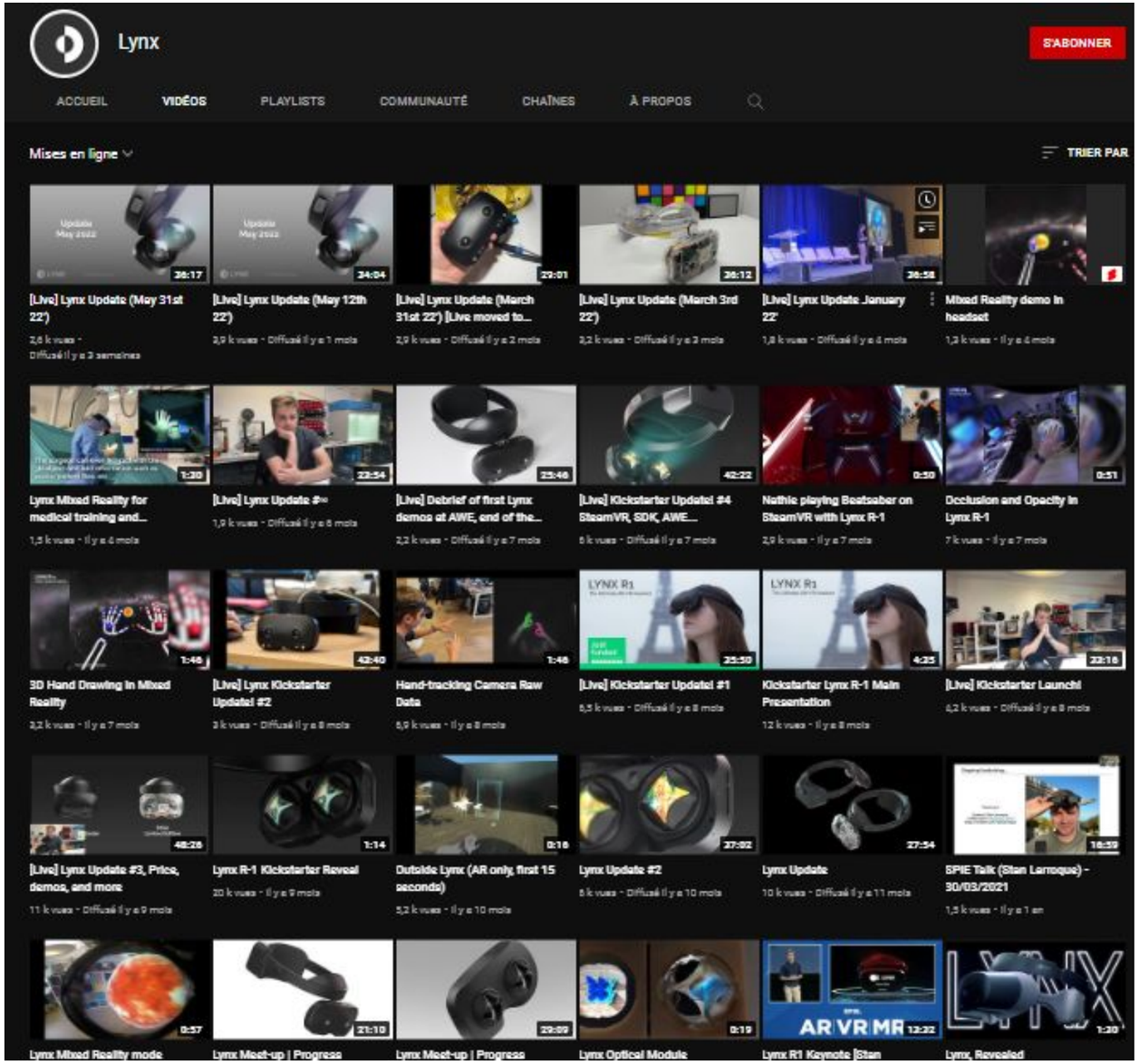
Lynx

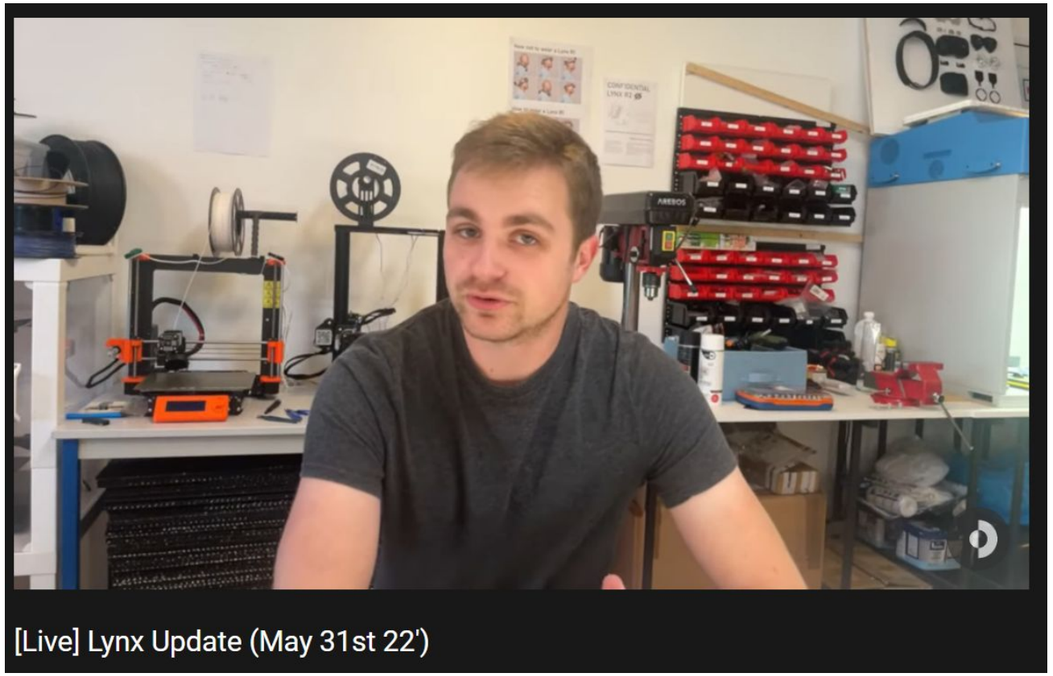
Coups de cœur Paris, France Matériel

725 281 €
engagés sur 300 000 €

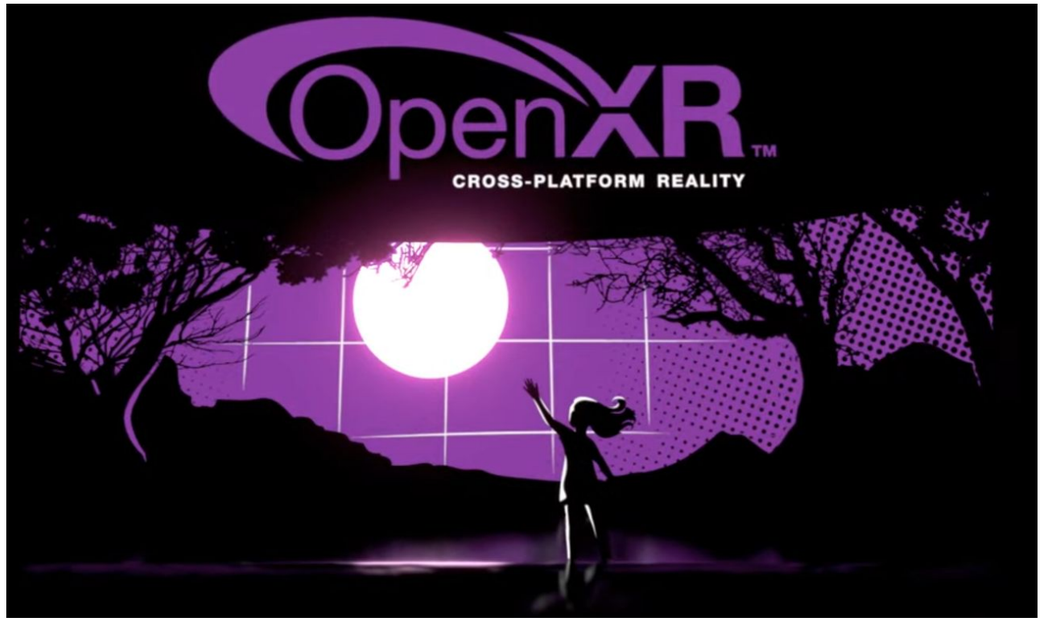
1 216
contributeurs

HACK IN THE WOODS





HACK IN THE WOODS



SideQuest
MAKE IT YOUR OWN

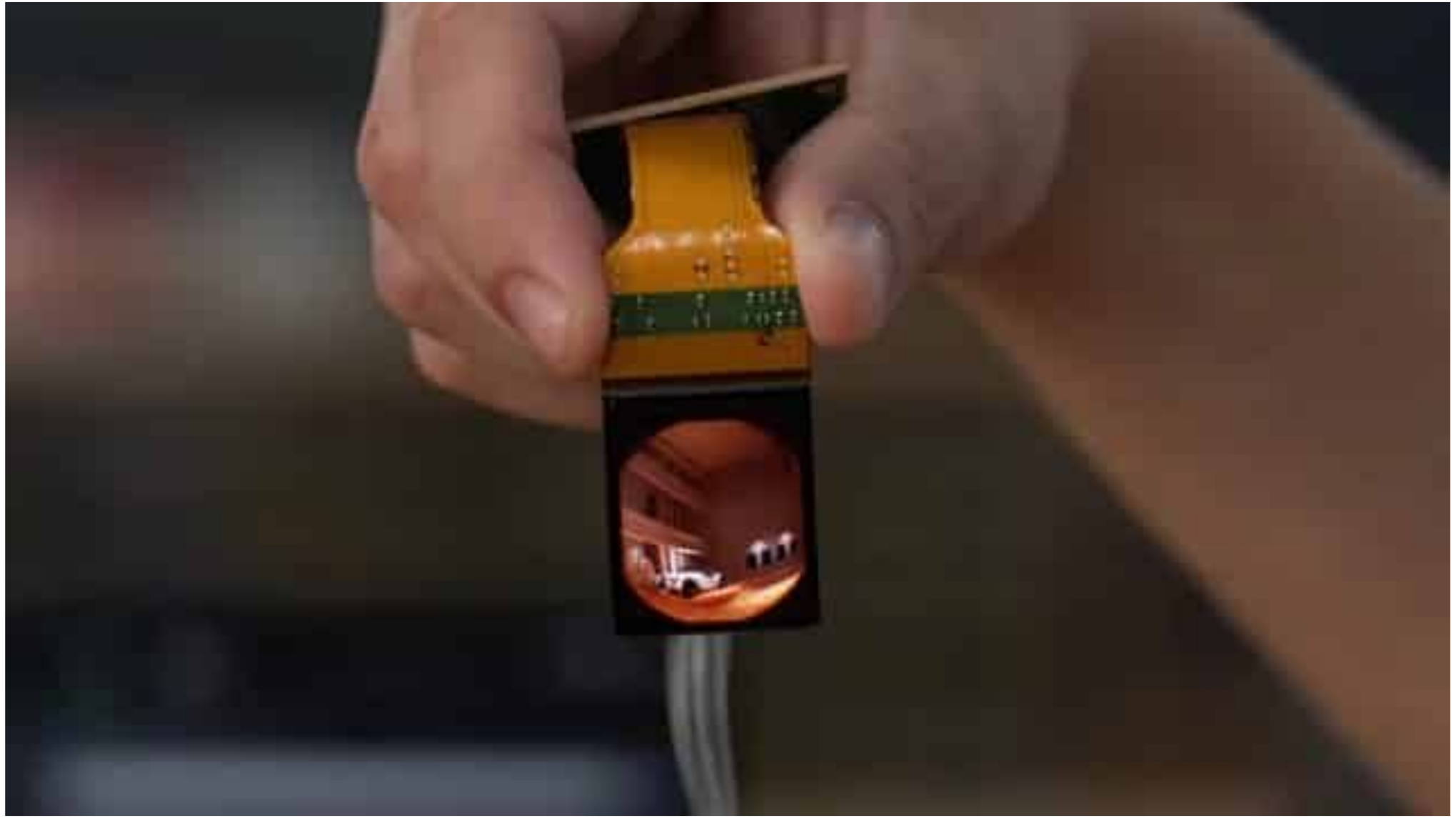
Unity®

ultraleap

“It is your device, it belongs to you.“

Micro Oled

HACK IN THE WOODS



60 pixel per degree For the next big Mark screen

HACK IN THE WOODS

The BEST VR Headset in the WORLD - I CAN'T GO BACK!

HACK IN THE WOODS

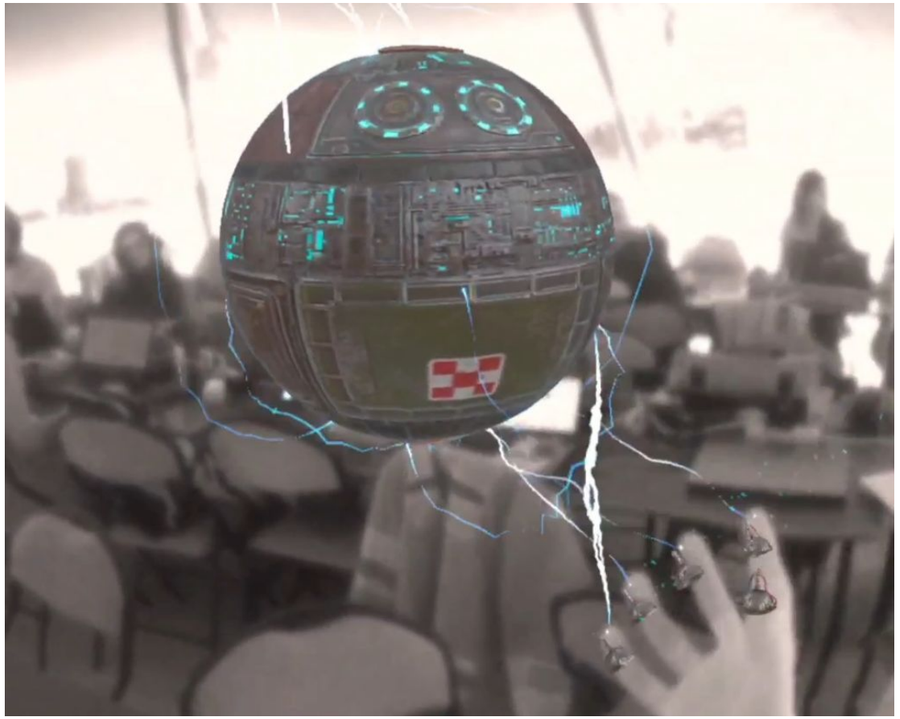
How is this even possible? Arpara 5k Micro OLED Review

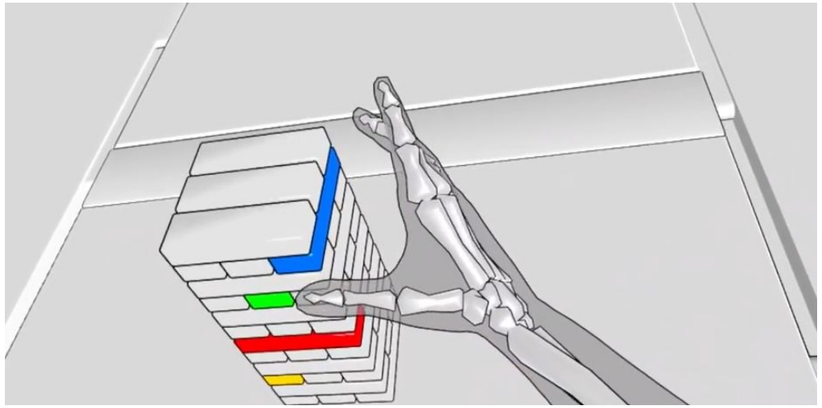
HACK IN THE WOODS

2880x2720 Peripheral Display

1920x1080 uOLED

Hands are amazing in video





HACK IN THE WOODS

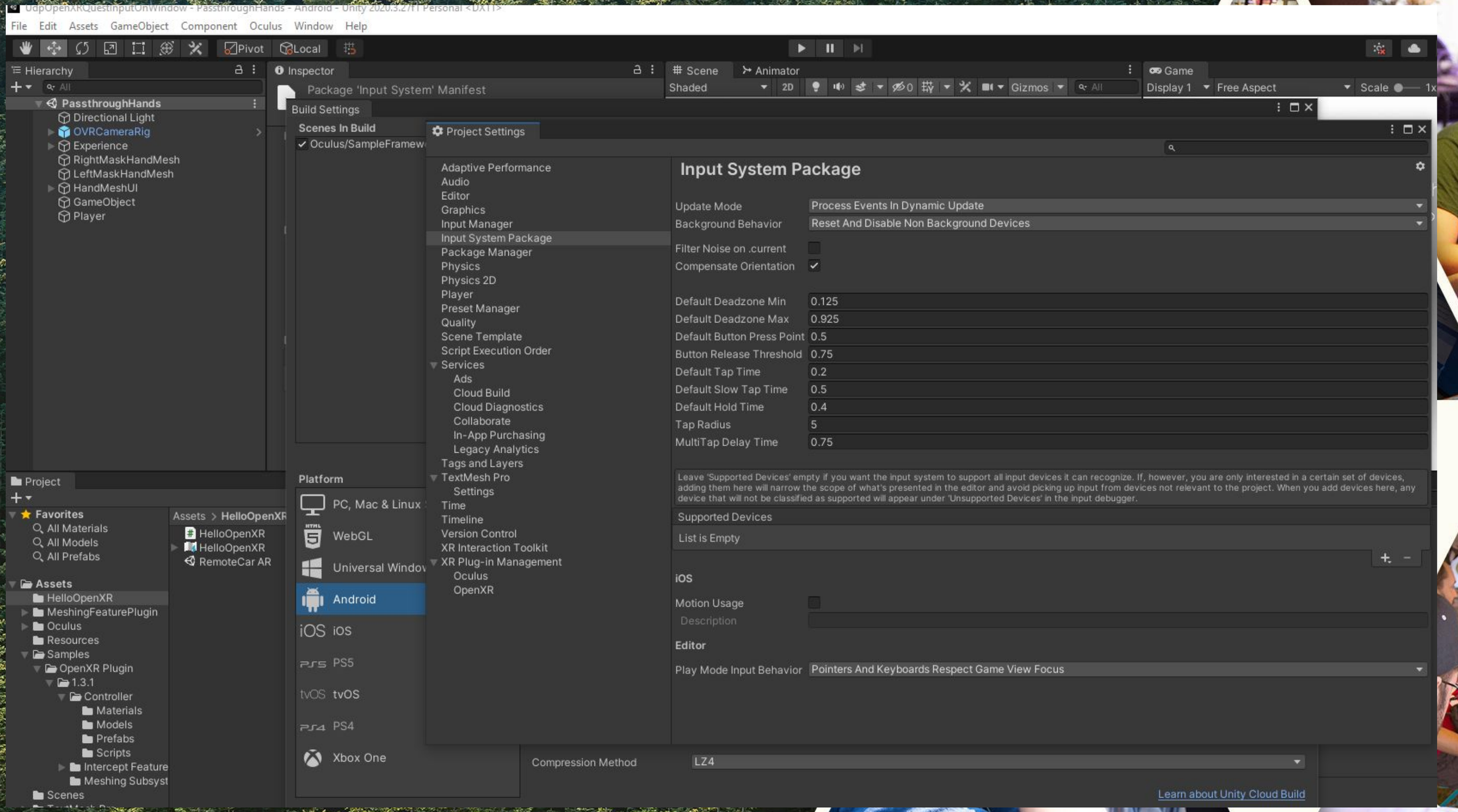
Unity Input System

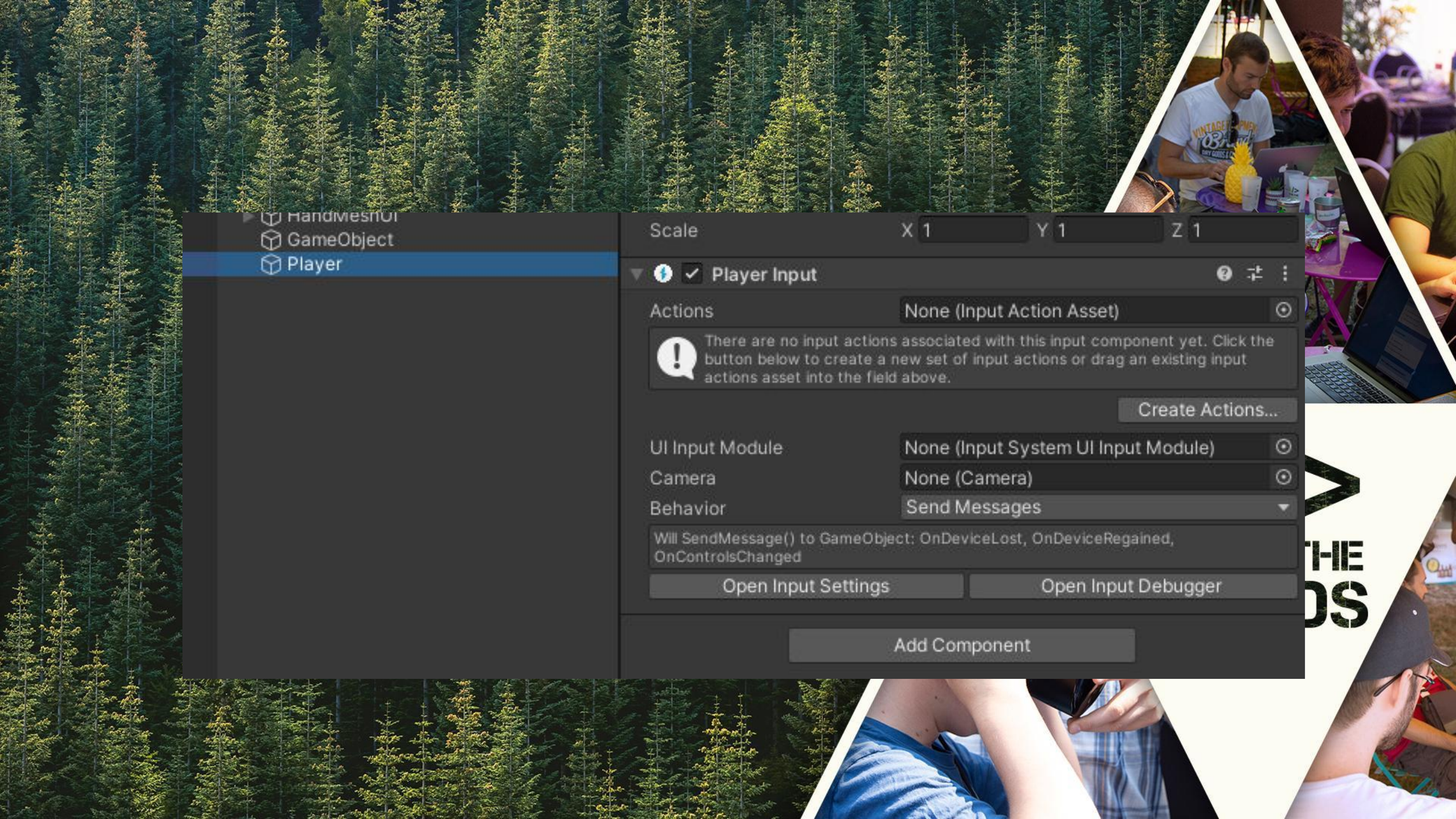
HACK IN THE WOODS

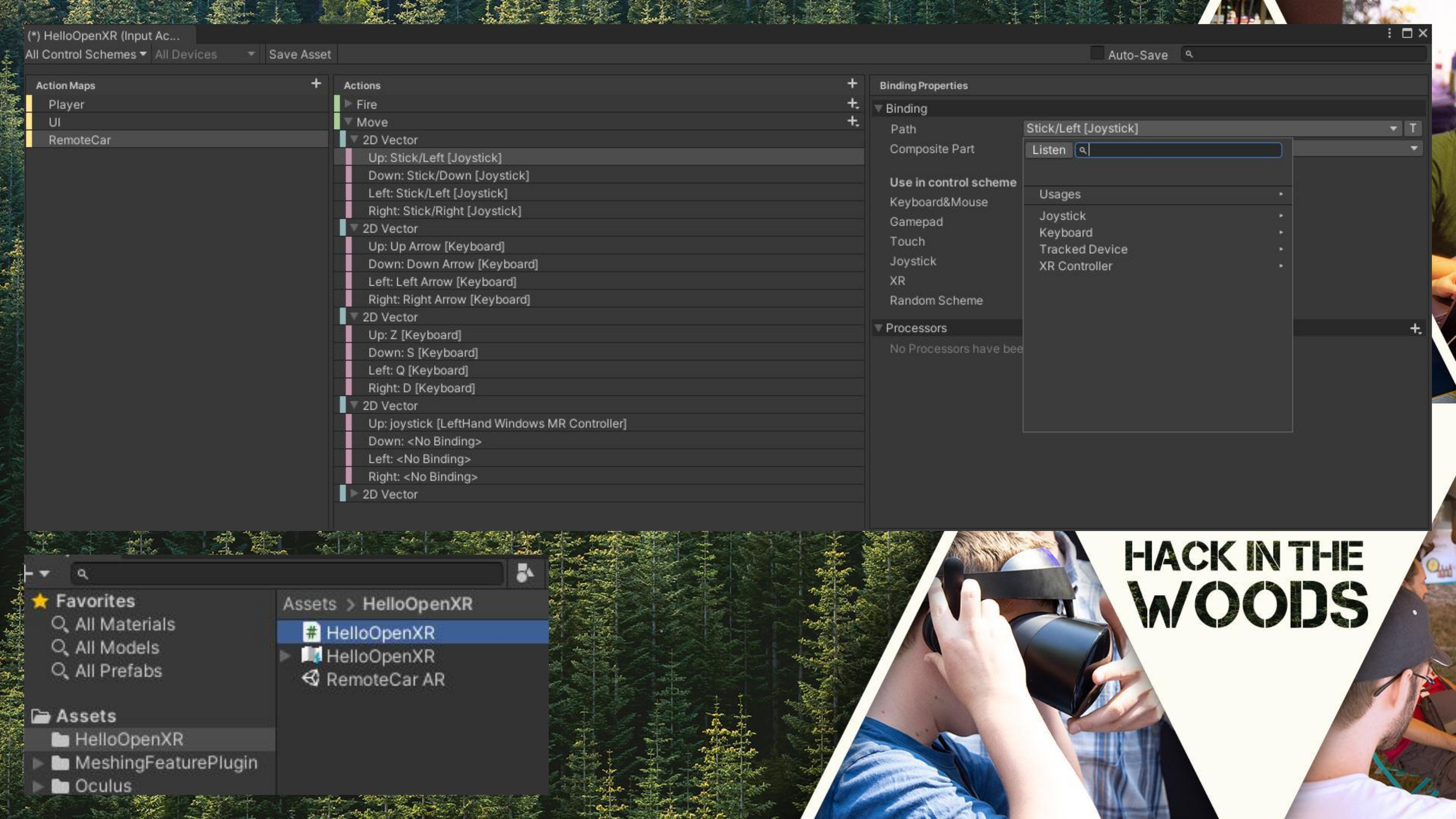
HACK IN THE WOODS

HACK IN THE WOODS

HACK IN THE WOODS


UdpOpenXRQuestInputOnWindow - PassthroughHands - Android - Unity 2020.3.27f1 Personal <DX11>
File Edit Assets GameObject Component Oculus Window Help
Pivot Local
Hierarchy
PassthroughHands
Directional Light
OVRCameraRig
Experience
RightMaskHandMesh
LeftMaskHandMesh
HandMeshUI
GameObject
Player
Inspector
Package 'Input System' Manifest
Scene Animator
Shaded 2D Gizmos
Game
Display 1 Free Aspect Scale 1x
Build Settings
Scenes In Build
Oculus/SampleFramew
Project Settings
Adaptive Performance
Audio
Editor
Graphics
Input Manager
Input System Package
Package Manager
Physics
Physics 2D
Player
Preset Manager
Quality
Scene Template
Script Execution Order
Services
Ads
Cloud Build
Cloud Diagnostics
Collaborate
In-App Purchasing
Legacy Analytics
Tags and Layers
TextMesh Pro
Settings
Time
Timeline
Version Control
XR Interaction Toolkit
XR Plug-in Management
Oculus
OpenXR
Input System Package
Update Mode Process Events In Dynamic Update
Background Behavior Reset And Disable Non Background Devices
Filter Noise on .current
Compensate Orientation
Default Deadzone Min 0.125
Default Deadzone Max 0.925
Default Button Press Point 0.5
Button Release Threshold 0.75
Default Tap Time 0.2
Default Slow Tap Time 0.5
Default Hold Time 0.4
Tap Radius 5
MultiTap Delay Time 0.75
Leave 'Supported Devices' empty if you want the input system to support all input devices it can recognize. If, however, you are only interested in a certain set of devices, adding them here will narrow the scope of what's presented in the editor and avoid picking up input from devices not relevant to the project. When you add devices here, any device that will not be classified as supported will appear under 'Unsupported Devices' in the input debugger.
Supported Devices
List is Empty
iOS
Motion Usage
Description
Editor
Play Mode Input Behavior Pointers And Keyboards Respect Game View Focus
Platform
PC, Mac & Linux
WebGL
Universal Windov
Android
iOS
PS5
tvOS
PS4
Xbox One
Compression Method LZ4
Learn about Unity Cloud Build
Project
Favorites
All Materials
All Models
All Prefabs
Assets
HelloOpenXR
MeshingFeaturePlugin
Oculus
Resources
Samples
OpenXR Plugin
1.3.1
Controller
Materials
Models
Prefabs
Scripts
Intercept Feature
Meshing Subsyst
Scenes
Assets > HelloOpenXR
HelloOpenXR
HelloOpenXR
RemoteCar AR


HandMeshUI
GameObject
Player
Scale
X 1
Y 1
Z 1
Player Input
Actions
None (Input Action Asset)
There are no input actions associated with this input component yet. Click the button below to create a new set of input actions or drag an existing input actions asset into the field above.
Create Actions...
UI Input Module
None (Input System UI Input Module)
Camera
None (Camera)
Behavior
Send Messages
Will SendMessage() to GameObject: OnDeviceLost, OnDeviceRegained, OnControlsChanged
Open Input Settings
Open Input Debugger
Add Component

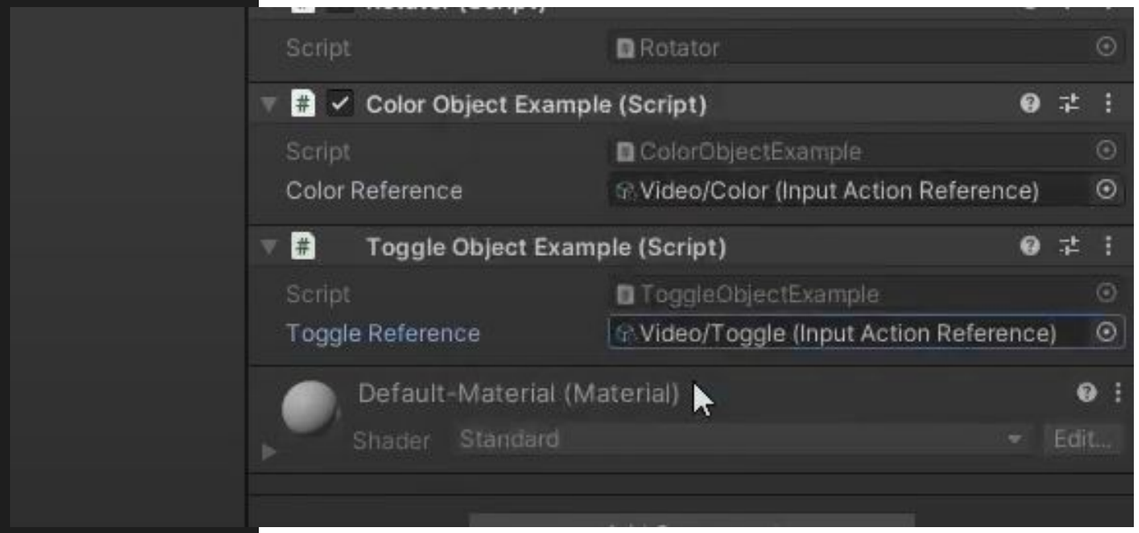

(*) HelloOpenXR (Input Ac...
All Control Schemes
All Devices
Save Asset
Auto-Save
Action Maps
Player
UI
RemoteCar
Actions
Fire
Move
2D Vector
Up: Stick/Left [Joystick]
Down: Stick/Down [Joystick]
Left: Stick/Left [Joystick]
Right: Stick/Right [Joystick]
2D Vector
Up: Up Arrow [Keyboard]
Down: Down Arrow [Keyboard]
Left: Left Arrow [Keyboard]
Right: Right Arrow [Keyboard]
2D Vector
Up: Z [Keyboard]
Down: S [Keyboard]
Left: Q [Keyboard]
Right: D [Keyboard]
2D Vector
Up: joystick [LeftHand Windows MR Controller]
Down: <No Binding>
Left: <No Binding>
Right: <No Binding>
2D Vector
Binding Properties
Binding
Path
Stick/Left [Joystick]
Composite Part
Listen
Use in control scheme
Keyboard&Mouse
Gamepad
Touch
Joystick
XR
Random Scheme
Usages
Joystick
Keyboard
Tracked Device
XR Controller
Processors
No Processors have bee
Favorites
All Materials
All Models
All Prefabs
Assets
HelloOpenXR
MeshingFeaturePlugin
Oculus
Assets > HelloOpenXR
HelloOpenXR
HelloOpenXR
RemoteCar AR
HACK IN THE
WOODS

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

public class ActionBaseMoveRemoteCar : MonoBehaviour
{
    public HelloOpenXR m_helloXR;
    public Transform m_targetToMove;
    public float m_lateralRotation=90;
    public float m_frontalSpeed=3;
    public Vector2 m_move2D;
    void Awake()
    {
        m_helloXR = new HelloOpenXR();
        m_helloXR.RemoteCar.Enable();
        m_helloXR.RemoteCar.Move.performed += ListenToMoveRequest;
        m_helloXR.RemoteCar.Move.canceled += ListenStopMoving;
    }
    public void Update()
    {
        m_targetToMove.Translate(
            Vector3.forward * m_move2D.y * Time.deltaTime * m_frontalSpeed, Space.Self);
        m_targetToMove.Rotate(
            Vector3.up * m_move2D.x * Time.deltaTime * m_lateralRotation, Space.Self);
    }

    private void ListenStopMoving(InputAction.CallbackContext obj)=>
        m_move2D = Vector2.zero;
    private void ListenToMoveRequest(InputAction.CallbackContext obj)=>
        m_move2D = obj.ReadValue<Vector2>();
}
```

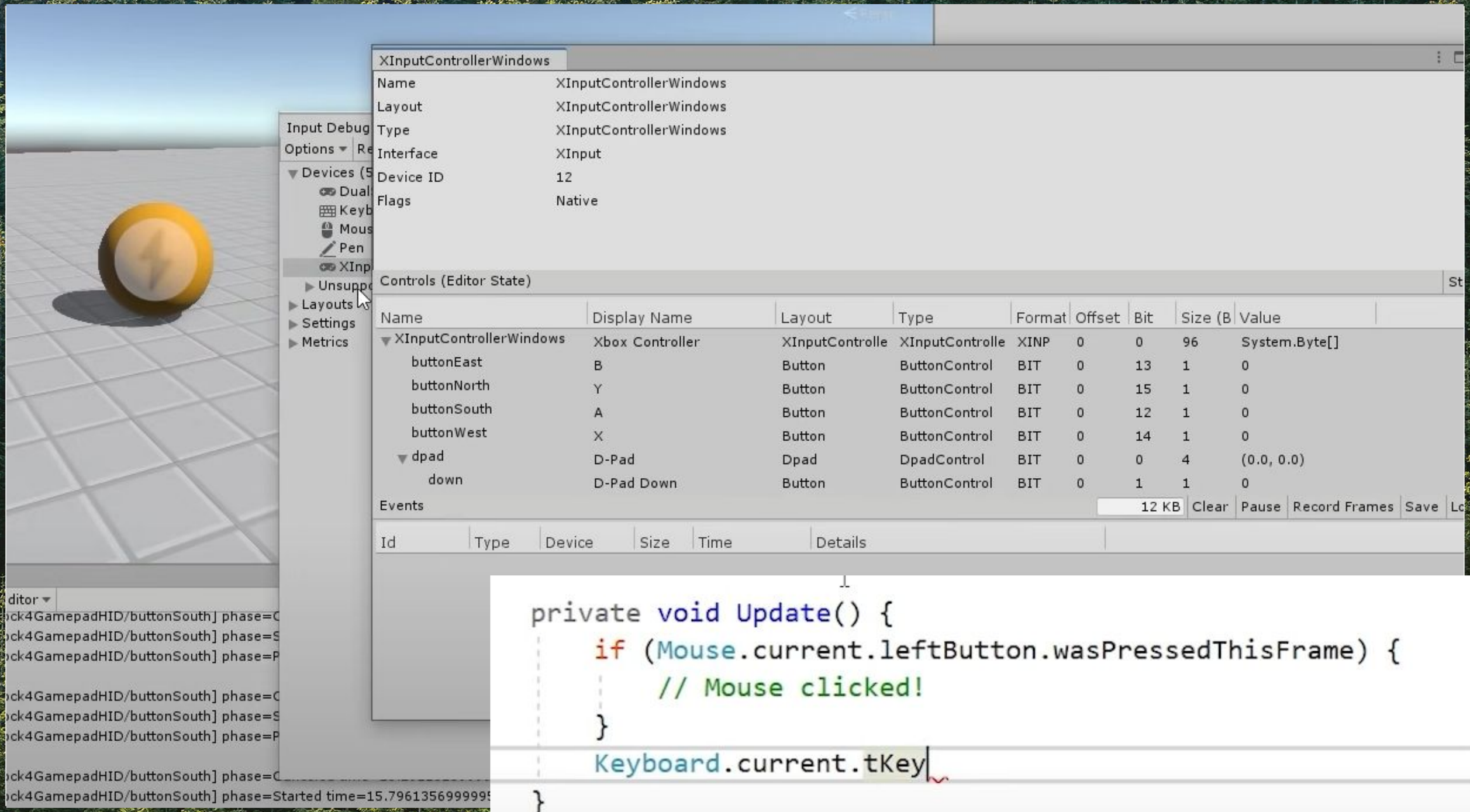
```
using UnityEngine;
using UnityEngine.InputSystem;
public class ReportMousePosition : MonoBehaviour
{
    void Update()
    {
        Vector2 mousePosition = Mouse.current.position.ReadValue();
        if(Keyboard.current.anyKey.wasPressedThisFrame)
        {
            Debug.Log("A key was pressed");
        }
        if (Gamepad.current.aButton.wasPressedThisFrame)
        {
            Debug.Log("A button was pressed");
        }
    }
}
```

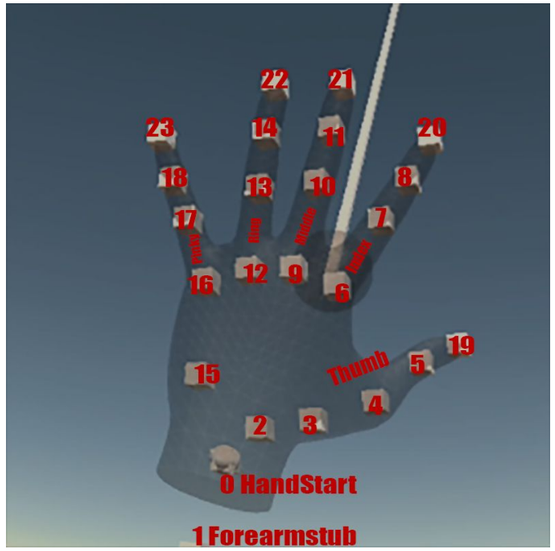
```
Unity Script | 0 references
public class ColorObjectExample : MonoBehaviour
{
    public InputActionReference colorReference = null;

    private MeshRenderer meshRenderer = null;

    Unity Message | 0 references
    private void Awake()
    {
        meshRenderer = GetComponent<MeshRenderer>();
    }

    Unity Message | 0 references
    private void Update()
    {
        float value = colorReference.action.ReadValue<float>();
        UpdateColor(value);
    }

    1 reference
    private void UpdateColor(float value)
    {
        meshRenderer.material.color = new Color(value,)
    }
}
```

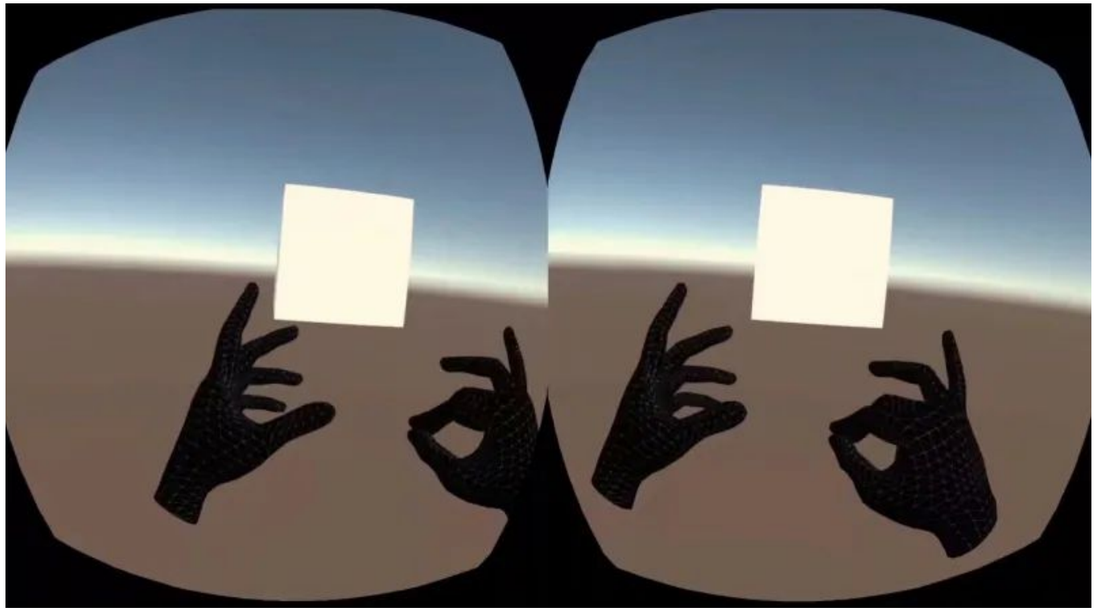
▲ 2 of 3 ▼ Color(float r, float g, float b)
Constructs a new Color with given r,g,b components and
g: Green component.

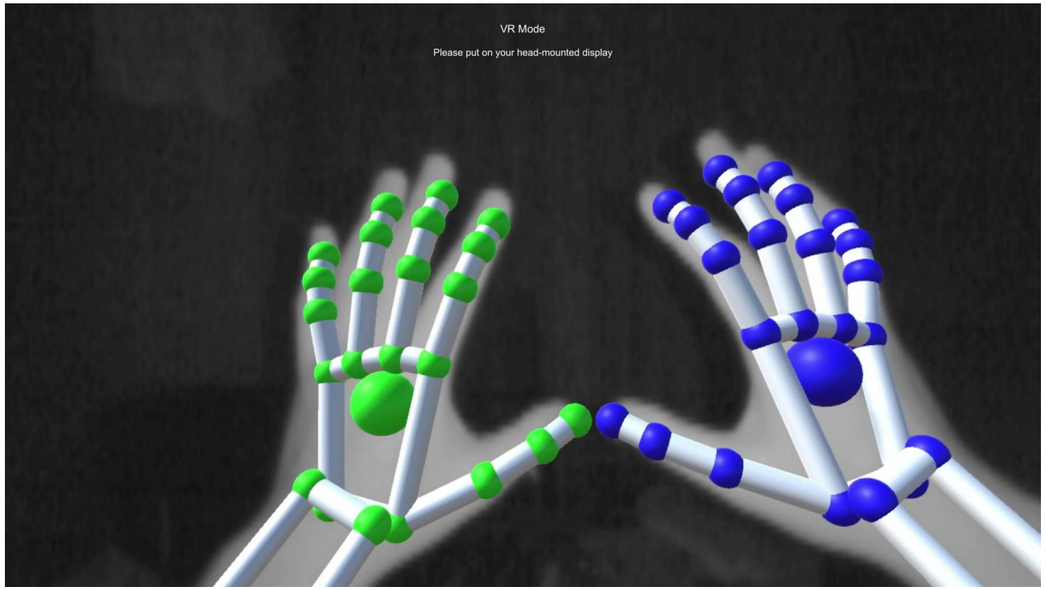


HACK IN THE WOODS

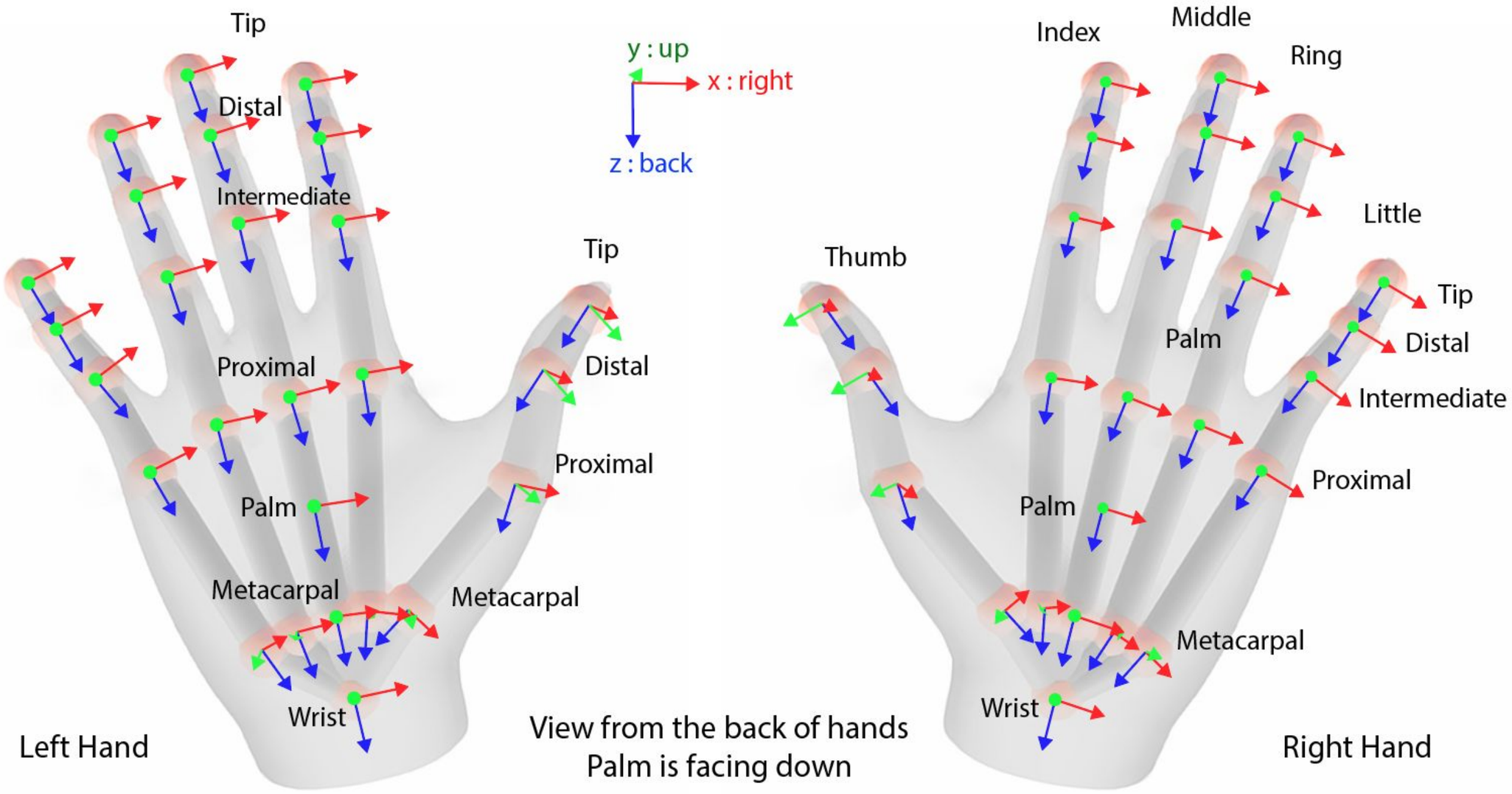

XInputControllerWindows
Name XInputControllerWindows
Layout XInputControllerWindows
Type XInputControllerWindows
Interface XInput
Device ID 12
Flags Native
Controls (Editor State)
Name | Display Name | Layout | Type | Format | Offset | Bit | Size (B | Value
XInputControllerWindows | Xbox Controller | XInputControlle | XInputControlle | XINP | 0 | 0 | 96 | System.Byte[]
buttonEast | B | Button | ButtonControl | BIT | 0 | 13 | 1 | 0
buttonNorth | Y | Button | ButtonControl | BIT | 0 | 15 | 1 | 0
buttonSouth | A | Button | ButtonControl | BIT | 0 | 12 | 1 | 0
buttonWest | X | Button | ButtonControl | BIT | 0 | 14 | 1 | 0
dpad | D-Pad | Dpad | DpadControl | BIT | 0 | 0 | 4 | (0.0, 0.0)
down | D-Pad Down | Button | ButtonControl | BIT | 0 | 1 | 1 | 0
Events
12 KB Clear Pause Record Frames Save
Id | Type | Device | Size | Time | Details
Input Debug
Options
Devices
Layouts
Settings
Metrics
ditor
ck4GamepadHID/buttonSouth] phase=Started time=15.7961356999995
private void Update() {
if (Mouse.current.leftButton.wasPressedThisFrame) {
// Mouse clicked!
}
Keyboard.current.tKey
}


22
21
23
14
11
20
18
13
10
8
17
7
12
9
6
16
15
Thumb
5
19
4
2
3
0 HandStart
1 Forearmstub

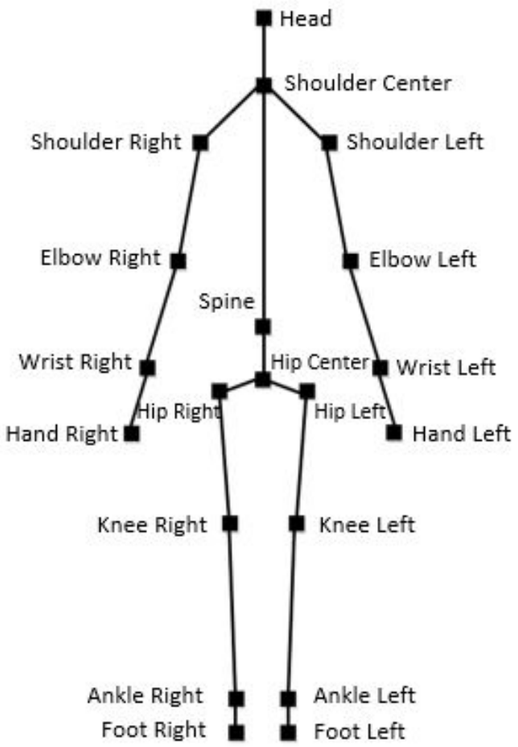


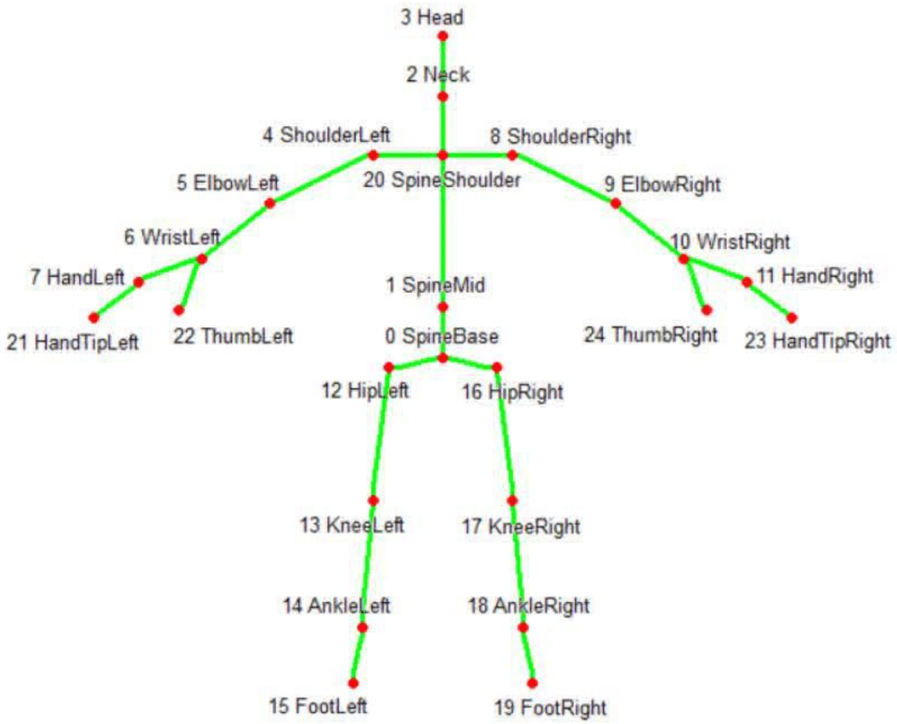

VR Mode
Please put on your head-mounted display

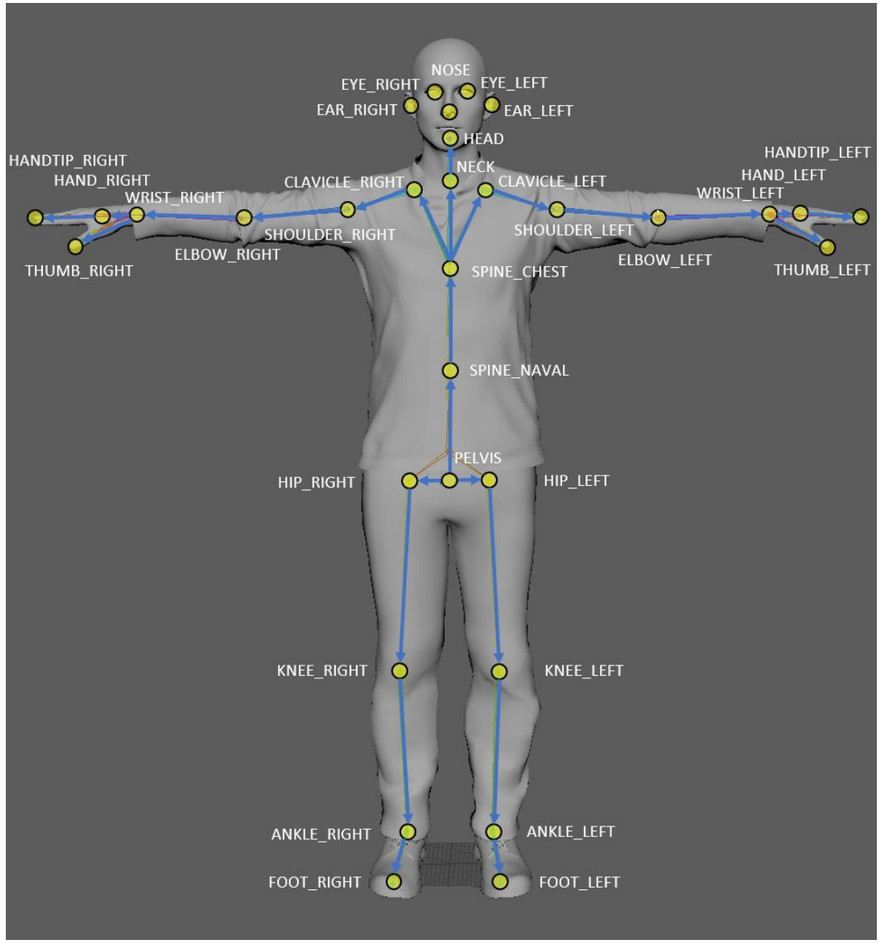
<
/>
HACK IN THE
WOODS

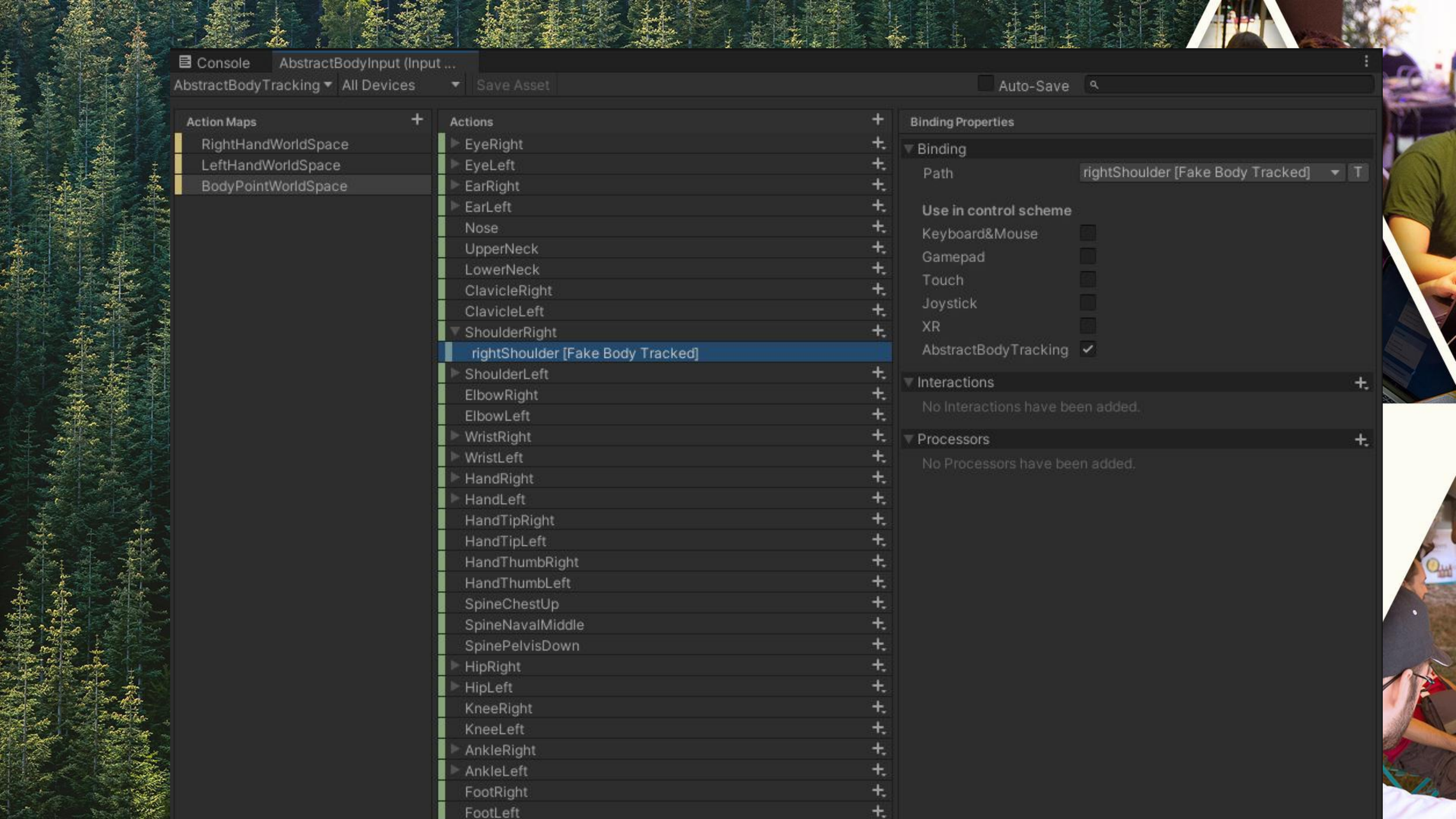


HACK IN THE WOODS

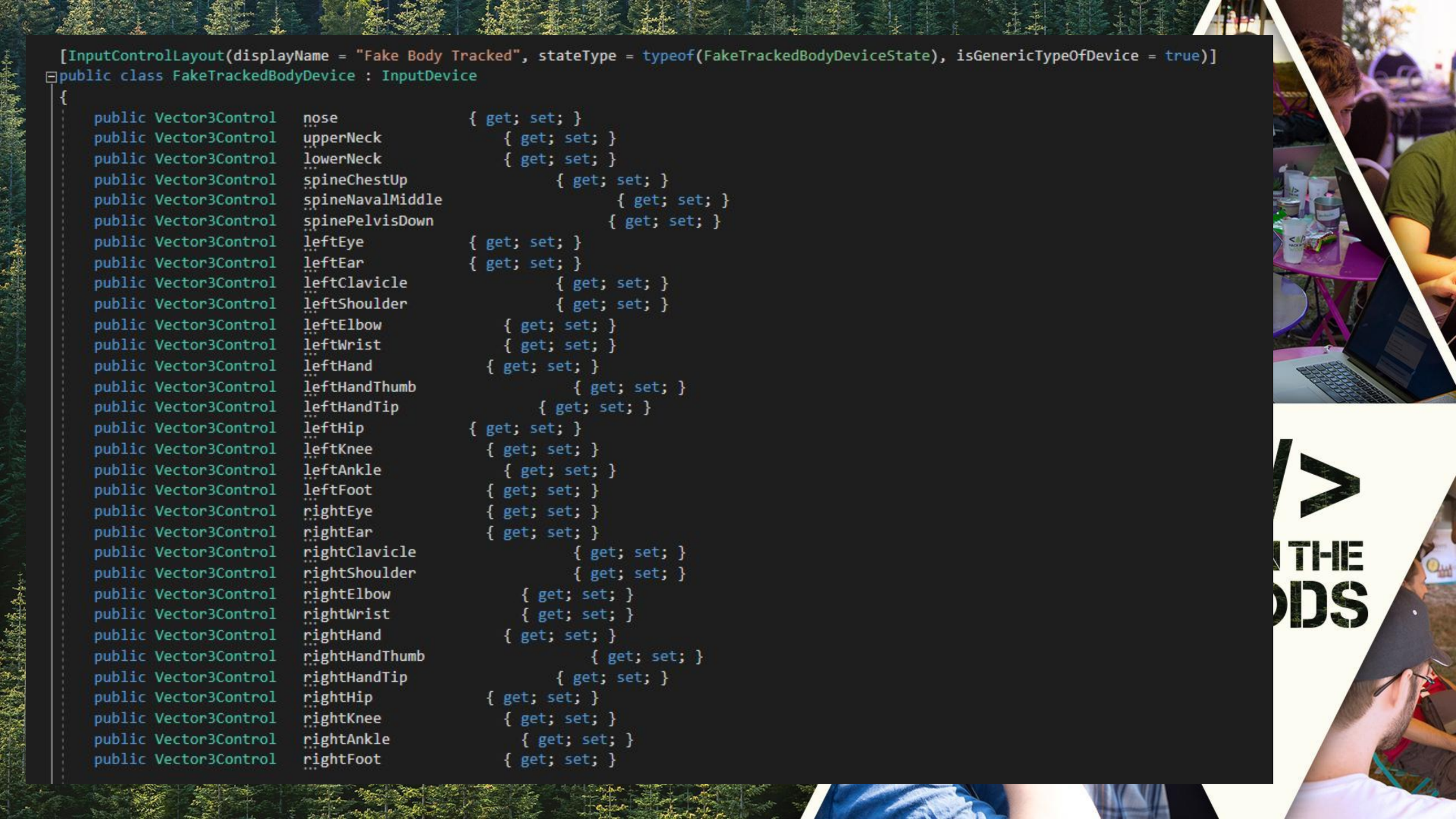







Console
AbstractBodyInput (Input ...
AbstractBodyTracking
All Devices
Save Asset
Auto-Save
Action Maps
RightHandWorldSpace
LeftHandWorldSpace
BodyPointWorldSpace
Actions
EyeRight
EyeLeft
EarRight
EarLeft
Nose
UpperNeck
LowerNeck
ClavicleRight
ClavicleLeft
ShoulderRight
rightShoulder [Fake Body Tracked]
ShoulderLeft
ElbowRight
ElbowLeft
WristRight
WristLeft
HandRight
HandLeft
HandTipRight
HandTipLeft
HandThumbRight
HandThumbLeft
SpineChestUp
SpineNavalMiddle
SpinePelvisDown
HipRight
HipLeft
KneeRight
KneeLeft
AnkleRight
AnkleLeft
FootRight
FootLeft
Binding Properties
Binding
Path
rightShoulder [Fake Body Tracked]
Use in control scheme
Keyboard&Mouse
Gamepad
Touch
Joystick
XR
AbstractBodyTracking
Interactions
No Interactions have been added.
Processors
No Processors have been added.

```
public struct FakeTrackedBodyDeviceState : IInputStateTypeInfo
{
    public FourCC format => new FourCC('F', 'H', 'T', 'B');
    [InputControl(noisy = true, dontReset = true, layout = "Vector3")] public Vector3  nose;
    [InputControl(noisy = true, dontReset = true, layout = "Vector3")] public Vector3  upperNeck;
    [InputControl(noisy = true, dontReset = true, layout = "Vector3")] public Vector3  lowerNeck;
    [InputControl(noisy = true, dontReset = true, layout = "Vector3")] public Vector3  spineChestUp;
    [InputControl(noisy = true, dontReset = true, layout = "Vector3")] public Vector3  spineNavalMiddle;
    [InputControl(noisy = true, dontReset = true, layout = "Vector3")] public Vector3  spinePelvisDown;
    [InputControl(noisy = true, dontReset = true, layout = "Vector3")] public Vector3  leftEye;
    [InputControl(noisy = true, dontReset = true, layout = "Vector3")] public Vector3  leftEar;
    [InputControl(noisy = true, dontReset = true, layout = "Vector3")] public Vector3  leftClavicle;
    [InputControl(noisy = true, dontReset = true, layout = "Vector3")] public Vector3  leftShoulder;
    [InputControl(noisy = true, dontReset = true, layout = "Vector3")] public Vector3  leftElbow;
    [InputControl(noisy = true, dontReset = true, layout = "Vector3")] public Vector3  leftWrist;
    [InputControl(noisy = true, dontReset = true, layout = "Vector3")] public Vector3  leftHand;
    [InputControl(noisy = true, dontReset = true, layout = "Vector3")] public Vector3  leftHandThumb;
    [InputControl(noisy = true, dontReset = true, layout = "Vector3")] public Vector3  leftHandTip;
    [InputControl(noisy = true, dontReset = true, layout = "Vector3")] public Vector3  leftHip;
    [InputControl(noisy = true, dontReset = true, layout = "Vector3")] public Vector3  leftKnee;
    [InputControl(noisy = true, dontReset = true, layout = "Vector3")] public Vector3  leftAnkle;
    [InputControl(noisy = true, dontReset = true, layout = "Vector3")] public Vector3  leftFoot;
    [InputControl(noisy = true, dontReset = true, layout = "Vector3")] public Vector3  rightEye;
    [InputControl(noisy = true, dontReset = true, layout = "Vector3")] public Vector3  rightEar;
    [InputControl(noisy = true, dontReset = true, layout = "Vector3")] public Vector3  rightClavicle;
    [InputControl(noisy = true, dontReset = true, layout = "Vector3")] public Vector3  rightShoulder;
    [InputControl(noisy = true, dontReset = true, layout = "Vector3")] public Vector3  rightElbow;
    [InputControl(noisy = true, dontReset = true, layout = "Vector3")] public Vector3  rightWrist;
    [InputControl(noisy = true, dontReset = true, layout = "Vector3")] public Vector3  rightHand;
    [InputControl(noisy = true, dontReset = true, layout = "Vector3")] public Vector3  rightHandThumb;
    [InputControl(noisy = true, dontReset = true, layout = "Vector3")] public Vector3  rightHandTip;
    [InputControl(noisy = true, dontReset = true, layout = "Vector3")] public Vector3  rightHip;
    [InputControl(noisy = true, dontReset = true, layout = "Vector3")] public Vector3  rightKnee;
    [InputControl(noisy = true, dontReset = true, layout = "Vector3")] public Vector3  rightAnkle;
    [InputControl(noisy = true, dontReset = true, layout = "Vector3")] public Vector3  rightFoot;


}
```

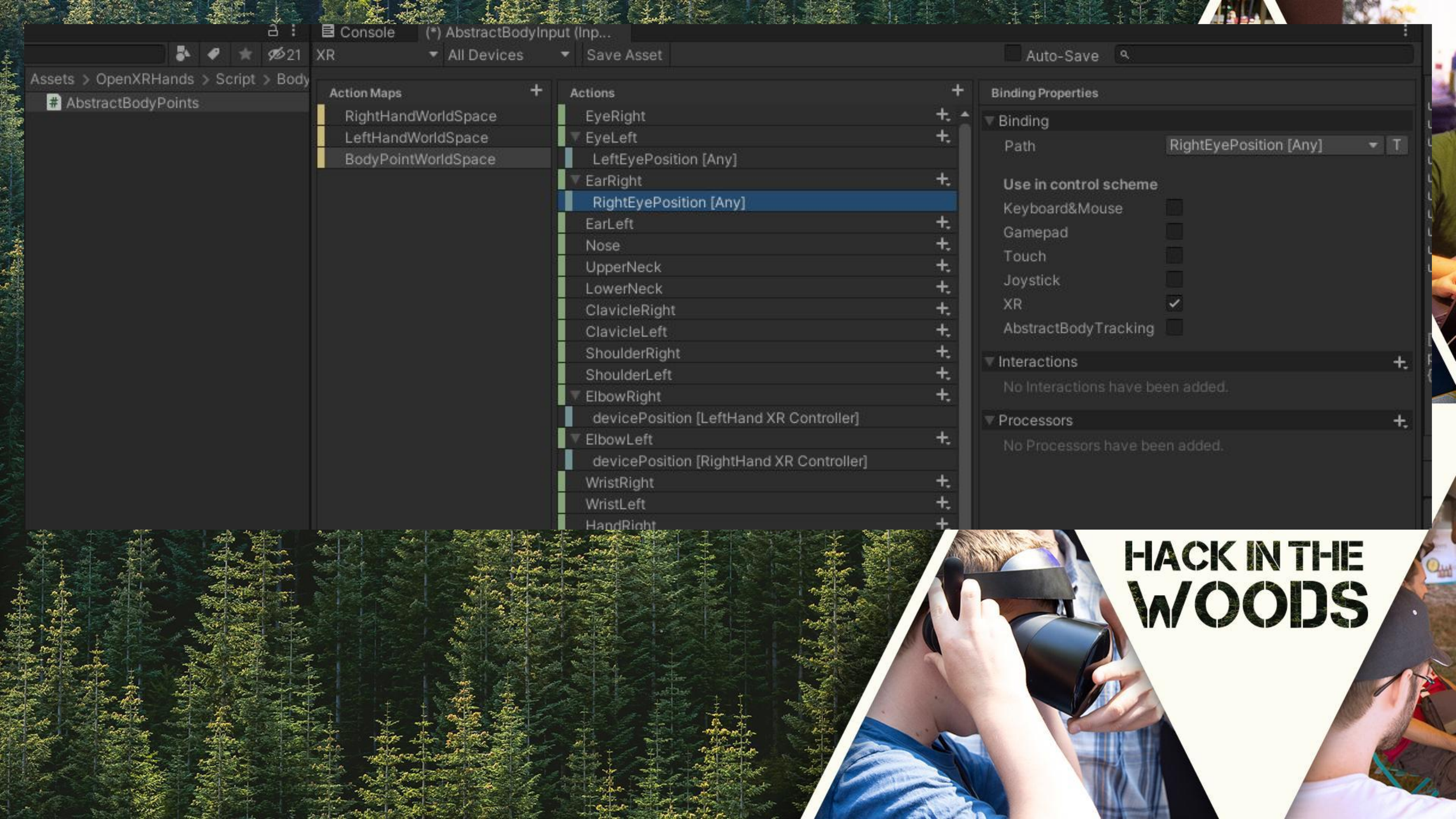


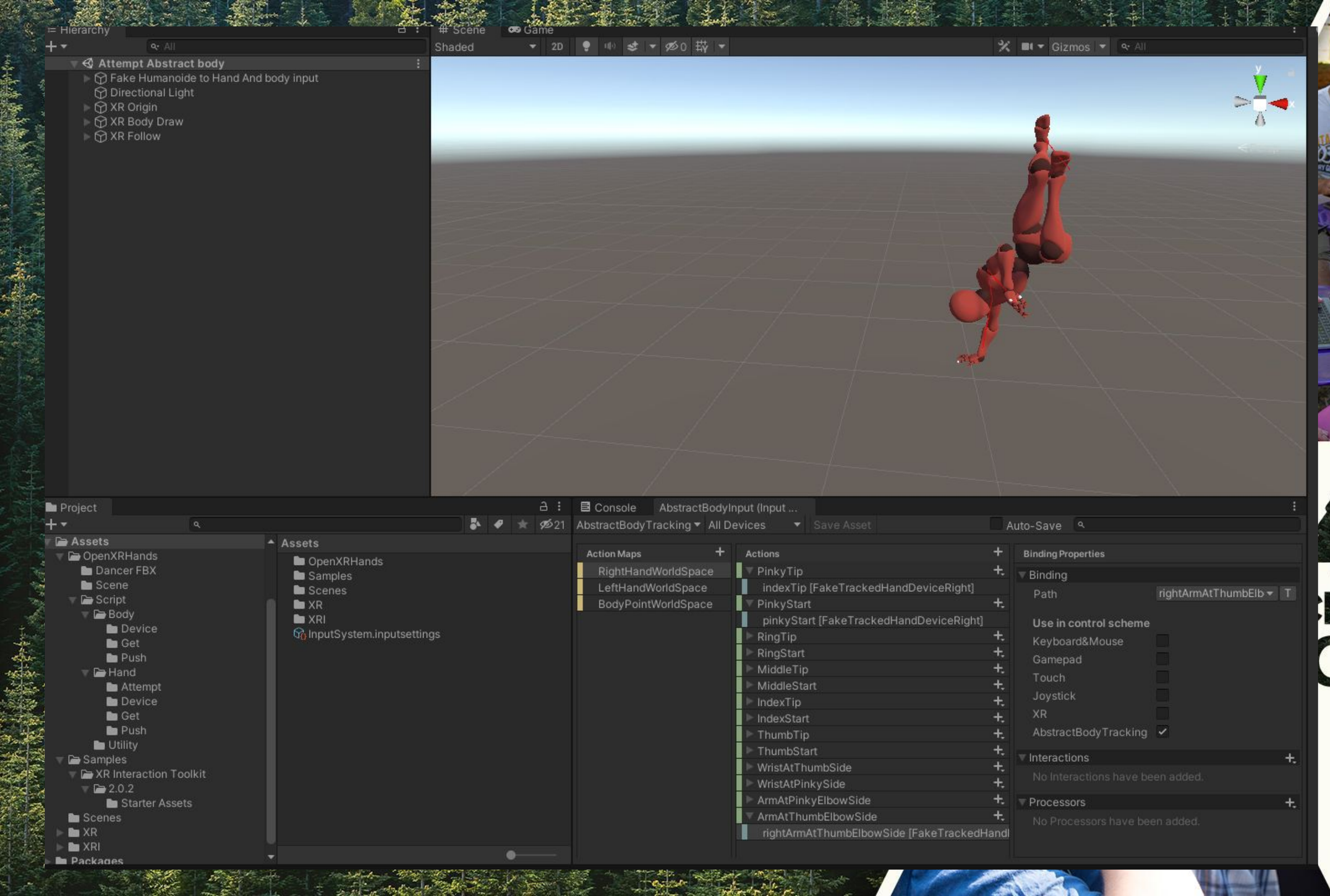
```
[InputControlLayout(displayName = "Fake Body Tracked", stateType = typeof(FakeTrackedBodyDeviceState), isGenericTypeOfDevice = true)]
public class FakeTrackedBodyDevice : InputDevice
{
    public Vector3Control   nose              { get; set; }
    public Vector3Control   upperNeck            { get; set; }
    public Vector3Control   lowerNeck            { get; set; }
    public Vector3Control   spineChestUp               { get; set; }
    public Vector3Control   spineNavalMiddle                 { get; set; }
    public Vector3Control   spinePelvisDown                { get; set; }
    public Vector3Control   leftEye           { get; set; }
    public Vector3Control   leftEar           { get; set; }
    public Vector3Control   leftClavicle               { get; set; }
    public Vector3Control   leftShoulder               { get; set; }
    public Vector3Control   leftElbow            { get; set; }
    public Vector3Control   leftWrist            { get; set; }
    public Vector3Control   leftHand           { get; set; }
    public Vector3Control   leftHandThumb                { get; set; }
    public Vector3Control   leftHandTip             { get; set; }
    public Vector3Control   leftHip           { get; set; }
    public Vector3Control   leftKnee           { get; set; }
    public Vector3Control   leftAnkle            { get; set; }
    public Vector3Control   leftFoot           { get; set; }
    public Vector3Control   rightEye           { get; set; }
    public Vector3Control   rightEar           { get; set; }
    public Vector3Control   rightClavicle                { get; set; }
    public Vector3Control   rightShoulder                { get; set; }
    public Vector3Control   rightElbow             { get; set; }
    public Vector3Control   rightWrist             { get; set; }
    public Vector3Control   rightHand            { get; set; }
    public Vector3Control   rightHandThumb                 { get; set; }
    public Vector3Control   rightHandTip               { get; set; }
    public Vector3Control   rightHip           { get; set; }
    public Vector3Control   rightKnee            { get; set; }
    public Vector3Control   rightAnkle             { get; set; }
    public Vector3Control   rightFoot            { get; set; }
```

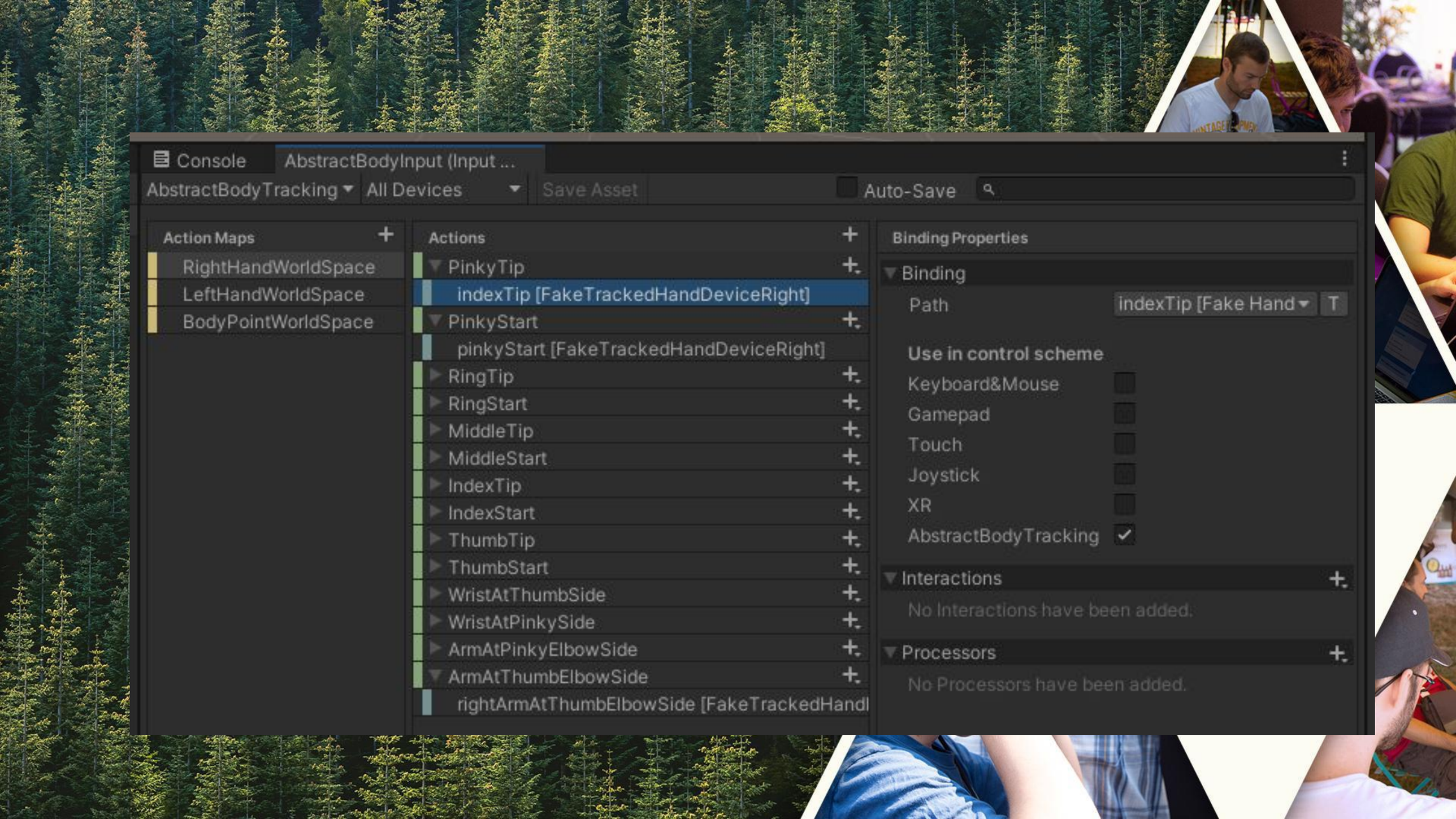
```
[UnityEditor.MenuItem("Tools/Add My body device")]
public static void InitilizedFromMenu()
{
    CheckThatDeviceIsRegistered();
}
ndif
protected override void FinishSetup()
{
    base.FinishSetup();

    nose                = GetChildControl<Vector3Control>("nose");
    upperNeck           = GetChildControl<Vector3Control>("upperNeck");
     lowerNeck          = GetChildControl<Vector3Control>("lowerNeck");
     spineChestUp       = GetChildControl<Vector3Control>("spineChestUp");
     spineNavalMiddle   = GetChildControl<Vector3Control>("spineNavalMiddle");
     spinePelvisDown    = GetChildControl<Vector3Control>("spinePelvisDown");
     leftEye            = GetChildControl<Vector3Control>("leftEye");
     leftEar            = GetChildControl<Vector3Control>("leftEar");
     leftClavicle       = GetChildControl<Vector3Control>("leftClavicle");
     leftShoulder       = GetChildControl<Vector3Control>("leftShoulder");
     leftElbow          = GetChildControl<Vector3Control>("leftElbow");
     leftWrist          = GetChildControl<Vector3Control>("leftWrist");
     leftHand           = GetChildControl<Vector3Control>("leftHand");
     leftHandThumb      = GetChildControl<Vector3Control>("leftHandThumb");
     leftHandTip        = GetChildControl<Vector3Control>("leftHandTip");
     leftHip            = GetChildControl<Vector3Control>("leftHip");
     leftKnee           = GetChildControl<Vector3Control>("leftKnee");
     leftAnkle          = GetChildControl<Vector3Control>("leftAnkle");
     leftFoot           = GetChildControl<Vector3Control>("leftFoot");
     rightEye           = GetChildControl<Vector3Control>("rightEye");
     rightEar           = GetChildControl<Vector3Control>("rightEar");
     rightClavicle      = GetChildControl<Vector3Control>("rightClavicle");
     rightShoulder      = GetChildControl<Vector3Control>("rightShoulder");
     rightElbow         = GetChildControl<Vector3Control>("rightElbow");
     rightWrist         = GetChildControl<Vector3Control>("rightWrist");
     rightHand          = GetChildControl<Vector3Control>("rightHand");
     rightHandThumb     = GetChildControl<Vector3Control>("rightHandThumb");
     rightHandTip       = GetChildControl<Vector3Control>("rightHandTip");
     rightHip           = GetChildControl<Vector3Control>("rightHip");
     rightKnee          = GetChildControl<Vector3Control>("rightKnee");
     rightAnkle         = GetChildControl<Vector3Control>("rightAnkle");
     rightFoot          = GetChildControl<Vector3Control>("rightFoot");


}
```

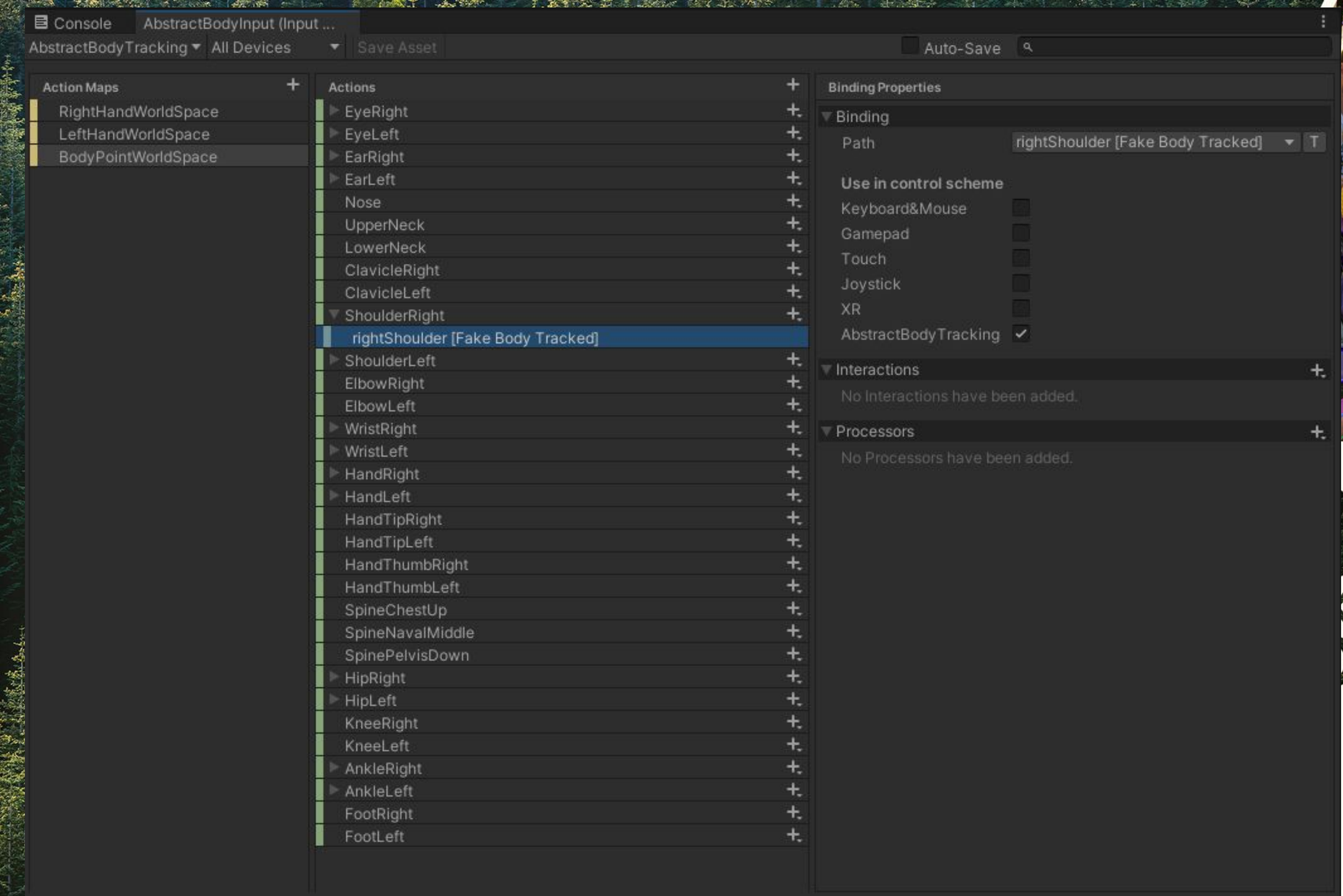
```
static FakeTrackedBodyDevice()
{
    CheckThatDeviceIsRegistered();

}
private static bool m_wasInitialized = false;
private static FakeTrackedBodyDevice body;
private static void CheckThatDeviceIsRegistered()
{
    FakeTrackedBodyDevice[] devices = InputSystem.devices.Where(k => k is FakeTrackedBodyDevice).Select(k => (FakeTrackedBodyDevice)k).ToArray();
    if (m_wasInitialized)
        return;
    InputSystem.RegisterLayout<FakeTrackedBodyDevice>(
       matches: new InputDeviceMatcher()
       .WithInterface("FakeTrackedBodyDeviceState"));
    if (devices.Length < 1)
    {
        body = (FakeTrackedBodyDevice) InputSystem.AddDevice(new InputDeviceDescription()
        {
            interfaceName = "FakeTrackedBodyDeviceState",
            product = "Body Tracked Mockup"
        });
    }
    InputSystem.QueueStateEvent(body, new FakeTrackedBodyDeviceState() { });
    m_wasInitialized = true;
}
[RuntimeInitializeOnLoadMethod]
public static void Initialize()
{
    CheckThatDeviceIsRegistered();
}
public static FakeTrackedBodyDevice GetBodyDevice()
{
    if (body == null) CheckThatDeviceIsRegistered();
    return body;
}
}
```

```
public class PushFakeHandsInputTransformMono : MonoBehaviour
{

    public FakeTrackedHandDeviceStateLeft m_fakeHandLeft;
    public FakeTrackedHandDeviceStateRight m_fakeHandRight;
    private FakeTrackedHandDeviceLeft m_deviceLeft;
    private FakeTrackedHandDeviceRight m_deviceRight;
    public TrackedHandsInfoTransform m_toPush;
    public enum PushType { ByQueueStateEvent, BySingleton}
    public PushType m_pushType;

    public void LateUpdate()
    {
        PushInfo();
    }

    private void PushInfo()
    {
        if (m_pushType == PushType.ByQueueStateEvent)
        {
            //if (m_deviceLeft == null)
            {
                m_deviceLeft = FakeTrackedHandDevice.GetLeftDevice();
            }
            //if (m_deviceRight == null)
            {
                m_deviceRight = FakeTrackedHandDevice.GetRightDevice();
            }


            //...
            InputSystem.QueueStateEvent(m_deviceLeft, m_fakeHandLeft, -1);
            InputSystem.QueueStateEvent(m_deviceRight, m_fakeHandRight, -1);
```


Console
(*) AbstractBodyInput (Inp...
XR
All Devices
Save Asset
Auto-Save
Assets > OpenXRHands > Script > Body
AbstractBodyPoints
Action Maps
RightHandWorldSpace
LeftHandWorldSpace
BodyPointWorldSpace
Actions
EyeRight
EyeLeft
LeftEyePosition [Any]
EarRight
RightEyePosition [Any]
EarLeft
Nose
UpperNeck
LowerNeck
ClavicleRight
ClavicleLeft
ShoulderRight
ShoulderLeft
ElbowRight
devicePosition [LeftHand XR Controller]
ElbowLeft
devicePosition [RightHand XR Controller]
WristRight
WristLeft
Binding Properties
Binding
Path
RightEyePosition [Any]
Use in control scheme
Keyboard&Mouse
Gamepad
Touch
Joystick
XR
AbstractBodyTracking
Interactions
No Interactions have been added.
Processors
No Processors have been added.
HACK IN THE
WOODS


Hierarchy
Attempt Abstract body
Fake Humanoide to Hand And body input
Directional Light
XR Origin
XR Body Draw
XR Follow
Scene
Game
Shaded
2D
Gizmos
Project
Assets
OpenXRHands
Dancer FBX
Scene
Script
Body
Device
Get
Push
Hand
Attempt
Device
Get
Push
Utility
Samples
XR Interaction Toolkit
2.0.2
Starter Assets
Scenes
XR
XRI
Packages
Assets
OpenXRHands
Samples
Scenes
XR
XRI
InputSystem.inputsettings
Console
AbstractBodyInput (Input ...
AbstractBodyTracking
All Devices
Save Asset
Auto-Save
Action Maps
RightHandWorldSpace
LeftHandWorldSpace
BodyPointWorldSpace
Actions
PinkyTip
indexTip [FakeTrackedHandDeviceRight]
PinkyStart
pinkyStart [FakeTrackedHandDeviceRight]
RingTip
RingStart
MiddleTip
MiddleStart
IndexTip
IndexStart
ThumbTip
ThumbStart
WristAtThumbSide
WristAtPinkySide
ArmAtPinkyElbowSide
ArmAtThumbElbowSide
rightArmAtThumbElbowSide [FakeTrackedHandI
Binding Properties
Binding
Path
rightArmAtThumbElb
Use in control scheme
Keyboard&Mouse
Gamepad
Touch
Joystick
XR
AbstractBodyTracking
Interactions
No Interactions have been added.
Processors
No Processors have been added.


Console
AbstractBodyInput (Input ...
AbstractBodyTracking
All Devices
Save Asset
Auto-Save
Action Maps
RightHandWorldSpace
LeftHandWorldSpace
BodyPointWorldSpace
Actions
PinkyTip
indexTip [FakeTrackedHandDeviceRight]
PinkyStart
pinkyStart [FakeTrackedHandDeviceRight]
RingTip
RingStart
MiddleTip
MiddleStart
IndexTip
IndexStart
ThumbTip
ThumbStart
WristAtThumbSide
WristAtPinkySide
ArmAtPinkyElbowSide
ArmAtThumbElbowSide
rightArmAtThumbElbowSide [FakeTrackedHandI
Binding Properties
Binding
Path
indexTip [Fake Hand
T
Use in control scheme
Keyboard&Mouse
Gamepad
Touch
Joystick
XR
AbstractBodyTracking
Interactions
No Interactions have been added.
Processors
No Processors have been added.

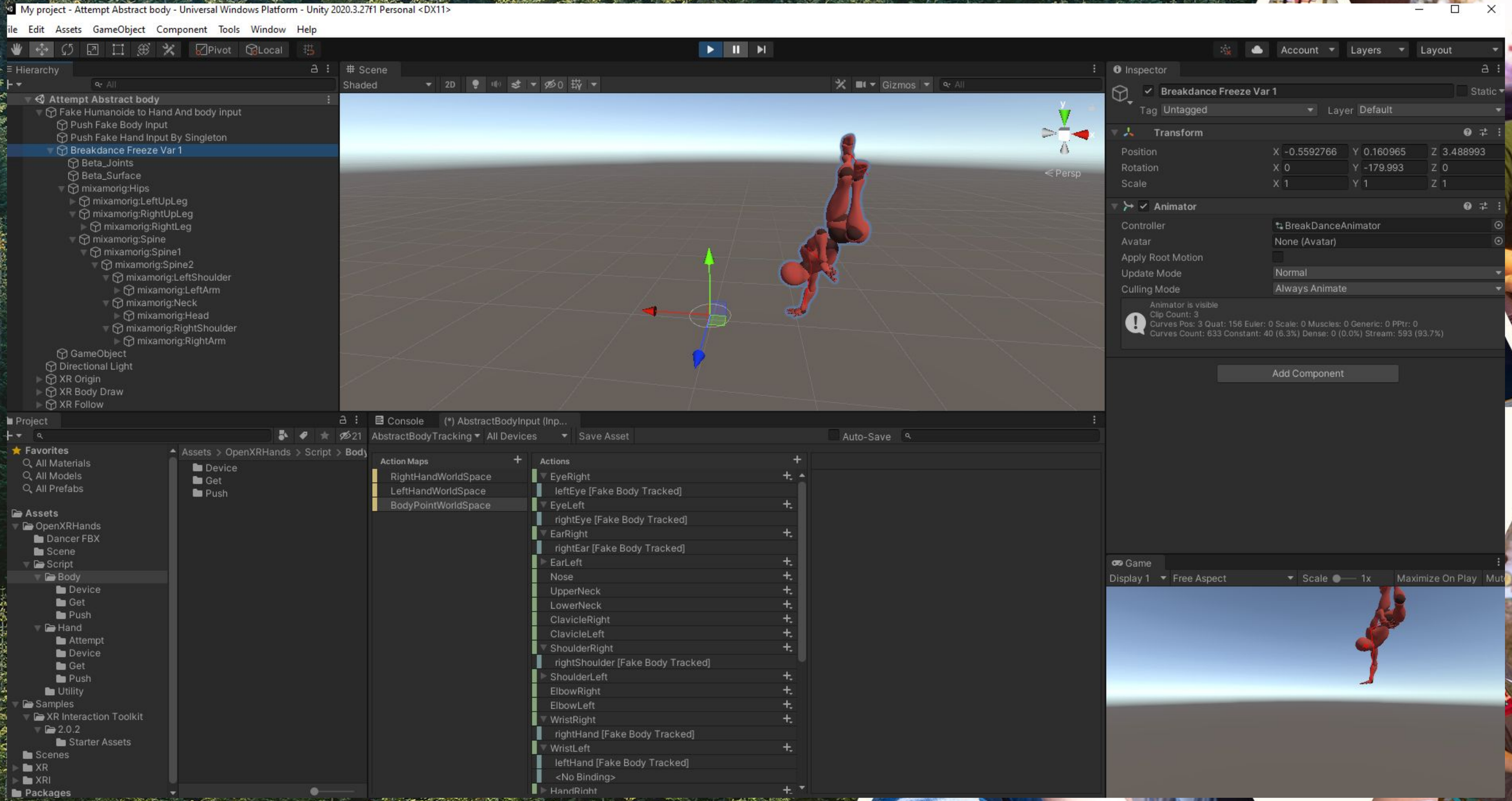

Console
AbstractBodyInput (Input ...
AbstractBodyTracking
All Devices
Save Asset
Auto-Save
Action Maps
RightHandWorldSpace
LeftHandWorldSpace
BodyPointWorldSpace
Actions
EyeRight
EyeLeft
EarRight
EarLeft
Nose
UpperNeck
LowerNeck
ClavicleRight
ClavicleLeft
ShoulderRight
rightShoulder [Fake Body Tracked]
ShoulderLeft
ElbowRight
ElbowLeft
WristRight
WristLeft
HandRight
HandLeft
HandTipRight
HandTipLeft
HandThumbRight
HandThumbLeft
SpineChestUp
SpineNavalMiddle
SpinePelvisDown
HipRight
HipLeft
KneeRight
KneeLeft
AnkleRight
AnkleLeft
FootRight
FootLeft
Binding Properties
Binding
Path
rightShoulder [Fake Body Tracked]
T
Use in control scheme
Keyboard&Mouse
Gamepad
Touch
Joystick
XR
AbstractBodyTracking
Interactions
No Interactions have been added.
Processors
No Processors have been added.

```cpp
using UnityEngine.InputSystem.Utilities;
using UnityEngine.InputSystem.XR;

[System.Serializable]
public class TrackedBodyInfo<T>
{
    public T nose;
    public T upperNeck;
    public T lowerNeck;
    public T spineChestUp;
    public T spineNavalMiddle;
    public T spinePelvisDown;
    public T leftEye;
    public T leftEar;
    public T leftClavicle;
    public T leftShoulder;
    public T leftElbow;
    public T leftWrist;
    public T leftHand;
    public T leftHandThumb;
    public T leftHandTip;
    public T leftHip;
    public T leftKnee;
    public T leftAnkle;
    public T leftFoot;
    public T rightEye;
    public T rightEar;
    public T rightClavicle;
    public T rightShoulder;
    public T rightElbow;
    public T rightWrist;
    public T rightHand;
    public T rightHandThumb;
    public T rightHandTip;
    public T rightHip;
    public T rightKnee;
    public T rightAnkle;
    public T rightFoot;
    public T GetInfoOfBody(in AbstractBodyInputUtility.BodyAnchor anchor)
    { return AbstractBodyInputUtility.GetInfoOfBody<T>(this, in anchor); }
    public void SetInfoOfHand(in T info, in AbstractBodyInputUtility.BodyAnchor anchor)
    => AbstractBodyInputUtility.SetInfoOfBody<T>(this, in info, in anchor);
}
```

```cpp
[System.Serializable]
public class TrackedBodyInfoTransform : TrackedBodyInfo<Transform> { }

[System.Serializable]
public class TrackedBodyInfoVector3 : TrackedBodyInfo<Vector3> { }
[System.Serializable]
public class TrackedBodyInfoBool : TrackedBodyInfo<bool> { }
```

```
using UnityEngine.InputSystem;

public class AbstractBodyInputUtility
{
    public enum HandType { Left, Right }
    public enum FingerName { Pinky, Ring, Middle, Index, Thumb }
    public enum FingerPart { Start, MiddleStart, MiddleTip, Tip }
    public enum BodyAnchor
    {
        Nose,UpperNeck,LowerNeck,SpineChestUp,SpineNavalMiddle,SpinePelvisDown,LeftEye,
        LeftEar,LeftClavicle,LeftShoulder,LeftElbow,LeftWrist,LeftHand,LeftHandThumb,
        LeftHandTip,LeftHip,LeftKnee,LeftAnkle,LeftFoot,RightEye,RightEar,RightClavicle,
        RightShoulder,RightElbow,RightWrist,RightHand,RightHandThumb,RightHandTip,
        RightHip,RightKnee,RightAnkle,RightFoot
    }
    public static T GetInfoOfBody<T>(in TrackedBodyInfo<T> body
        , in BodyAnchor bodyAnchor)...

    public static void SetInfoOfBody<T>(in TrackedBodyInfo<T> body, in T value, in BodyAnchor bodyAnchor)...
    public static T GetInfoOfHand<T>(in TrackedHandsInfo<T> hand
        , in HandType handType, in FingerName finger, in FingerPart part)...

    public static void SetInfoOfHand<T>(in TrackedHandsInfo<T> hand, in T info
        , in HandType handType, in FingerName finger, in FingerPart part)...
    public static T GetInfoOfHand<T>(in TrackedHandInfo<T> hand
        , in FingerName finger, in FingerPart part)...
    public static void SetInfoOfHand<T>(in TrackedHandInfo<T> hand, in T info
        , in FingerName finger, in FingerPart part)...
    public static Vector3 GetInfoOfBodyPosition(in AbstractBodyInput source, in BodyAnchor anchor)...
    public static Vector3 GetWorldPosition(in AbstractBodyInput source, in HandType handType, in FingerName finger, in FingerPart part)...
    public static void AddListenerToUpdateCallback(Action<InputAction.CallbackContext> callback, in AbstractBodyInput source, in HandType handType, in FingerName finger, in FingerPart part)...
    public static void AddListenerToStartCallback(Action<InputAction.CallbackContext> callback, in AbstractBodyInput source, in HandType handType, in FingerName finger, in FingerPart part)...
    public static void SetWorldPosition(in Vector3 position, in AbstractBodyInput source, in HandType handType, in FingerName finger, in FingerPart part)
    {

        throw new System.NotImplementedException();


    }
}
```

```cs
using UnityEngine;
using UnityEngine.InputSystem;

public class SetPositionOfAbstractBodyInput : MonoBehaviour
{

    public AbstractBodyInput m_abstractInputSource;
    public AbstractBodyInputUtility.HandType m_handType;
    public AbstractBodyInputUtility.FingerName m_fingerName;
    public AbstractBodyInputUtility.FingerPart m_fingerAnchor;
    public Transform m_targetToAffect;

    private void Start()
    {
        m_abstractInputSource = new AbstractBodyInput();
        m_abstractInputSource.Enable();
        AbstractBodyInputUtility.AddListenerToStartCallback(DoOnChange, in m_abstractInputSource, in m_handType, in m_fingerName, in m_fingerAnchor);
        AbstractBodyInputUtility.AddListenerToUpdateCallback(DoOnChange, in m_abstractInputSource, in m_handType, in m_fingerName, in m_fingerAnchor);
    }
    private void OnDisable()
    {
        m_abstractInputSource.Disable();
        m_abstractInputSource.Dispose();
    }
    private void DoOnChange(InputAction.CallbackContext obj)
    {
        Vector3 position = obj.ReadValue<Vector3>();
        bool isDefined = position != Vector3.zero;
        if (isDefined && m_targetToAffect)
            m_targetToAffect.position = position;
        if(m_targetToAffect && m_targetToAffect.gameObject)
        m_targetToAffect.gameObject.SetActive(isDefined);
    }

    public void Reset()
    {
        m_targetToAffect = GetComponent<Transform>();
    }
}
```


My project - Attempt Abstract body - Universal Windows Platform - Unity 2020.3.27f1 Personal <DX11>
File Edit Assets GameObject Component Tools Window Help
Pivot Local
Account Layers Layout
Hierarchy
Attempt Abstract body
Fake Humanoide to Hand And body input
Push Fake Body Input
Push Fake Hand Input By Singleton
Breakdance Freeze Var 1
Beta_Joints
Beta_Surface
mixamorig:Hips
mixamorig:LeftUpLeg
mixamorig:RightUpLeg
mixamorig:RightLeg
mixamorig:Spine
mixamorig:Spine1
mixamorig:Spine2
mixamorig:LeftShoulder
mixamorig:LeftArm
mixamorig:Neck
mixamorig:Head
mixamorig:RightShoulder
mixamorig:RightArm
GameObject
Directional Light
XR Origin
XR Body Draw
XR Follow
Scene
Shaded
2D
Gizmos
Persp
Inspector
Breakdance Freeze Var 1
Static
Tag Untagged
Layer Default
Transform
Position X -0.5592766 Y 0.160965 Z 3.488993
Rotation X 0 Y -179.993 Z 0
Scale X 1 Y 1 Z 1
Animator
Controller BreakDanceAnimator
Avatar None (Avatar)
Apply Root Motion
Update Mode Normal
Culling Mode Always Animate
Animator is visible
Clip Count: 3
Curves Pos: 3 Quat: 156 Euler: 0 Scale: 0 Muscles: 0 Generic: 0 PPtr: 0
Curves Count: 633 Constant: 40 (6.3%) Dense: 0 (0.0%) Stream: 593 (93.7%)
Add Component
Project
Favorites
All Materials
All Models
All Prefabs
Assets
OpenXRHands
Dancer FBX
Scene
Script
Body
Device
Get
Push
Hand
Attempt
Device
Get
Push
Utility
Samples
XR Interaction Toolkit
2.0.2
Starter Assets
Scenes
XR
XRI
Packages
Assets > OpenXRHands > Script > Body
Device
Get
Push
Console
(*) AbstractBodyInput (Inp...
AbstractBodyTracking
All Devices
Save Asset
Auto-Save
Action Maps
RightHandWorldSpace
LeftHandWorldSpace
BodyPointWorldSpace
Actions
EyeRight
leftEye [Fake Body Tracked]
EyeLeft
rightEye [Fake Body Tracked]
EarRight
rightEar [Fake Body Tracked]
EarLeft
Nose
UpperNeck
LowerNeck
ClavicleRight
ClavicleLeft
ShoulderRight
rightShoulder [Fake Body Tracked]
ShoulderLeft
ElbowRight
ElbowLeft
WristRight
rightHand [Fake Body Tracked]
WristLeft
leftHand [Fake Body Tracked]
<No Binding>
Game
Display 1
Free Aspect
Scale 1x
Maximize On Play

mixamo Characters Animations

Log in Sign up

Search

48 Per page

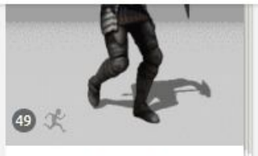
DEFAULT CHARACTER

DOWNLOAD

SEND TO AERO

UPLOAD CHARACTER

FIND ANIMATIONS



Sword And Shield Pack

Capoeira Pack

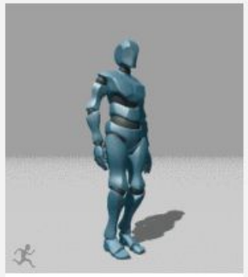
Praying

Crouch To Stand

Hip Hop Dancing

Silly Dancing

Rumba Dancing



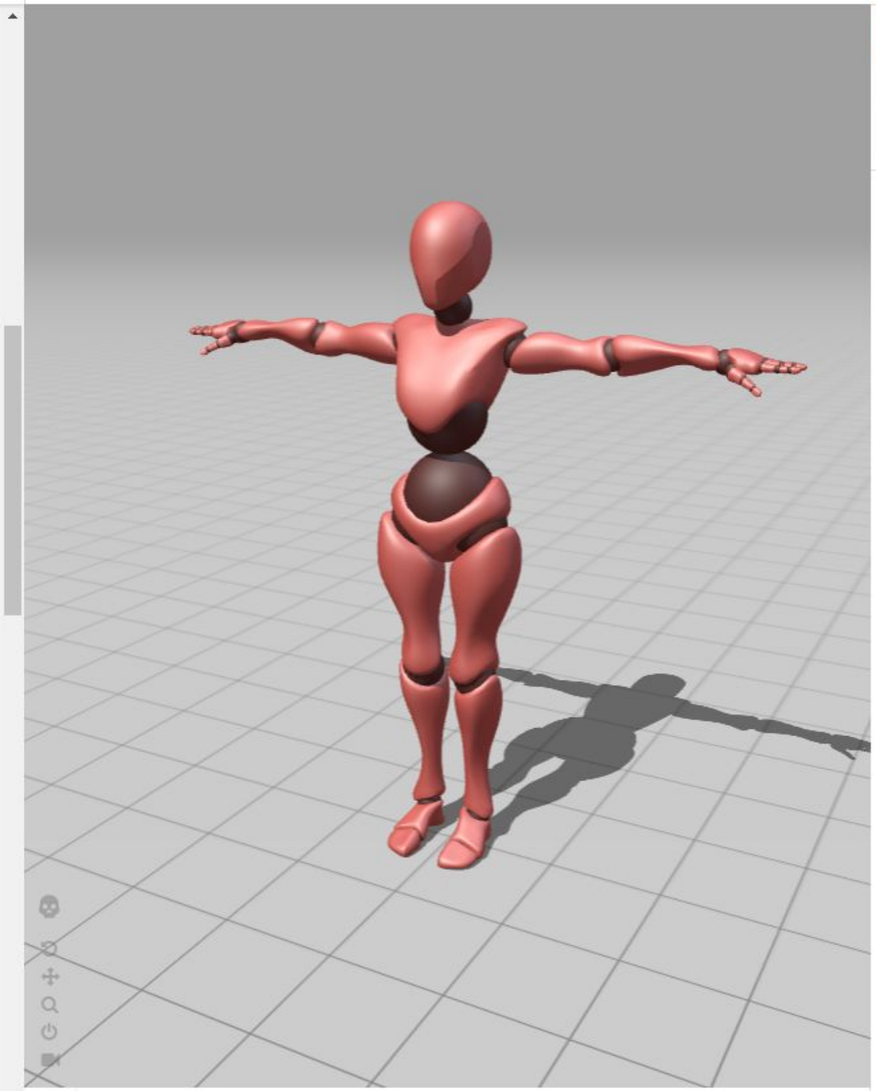
Joyful Jump

Zombie Idle

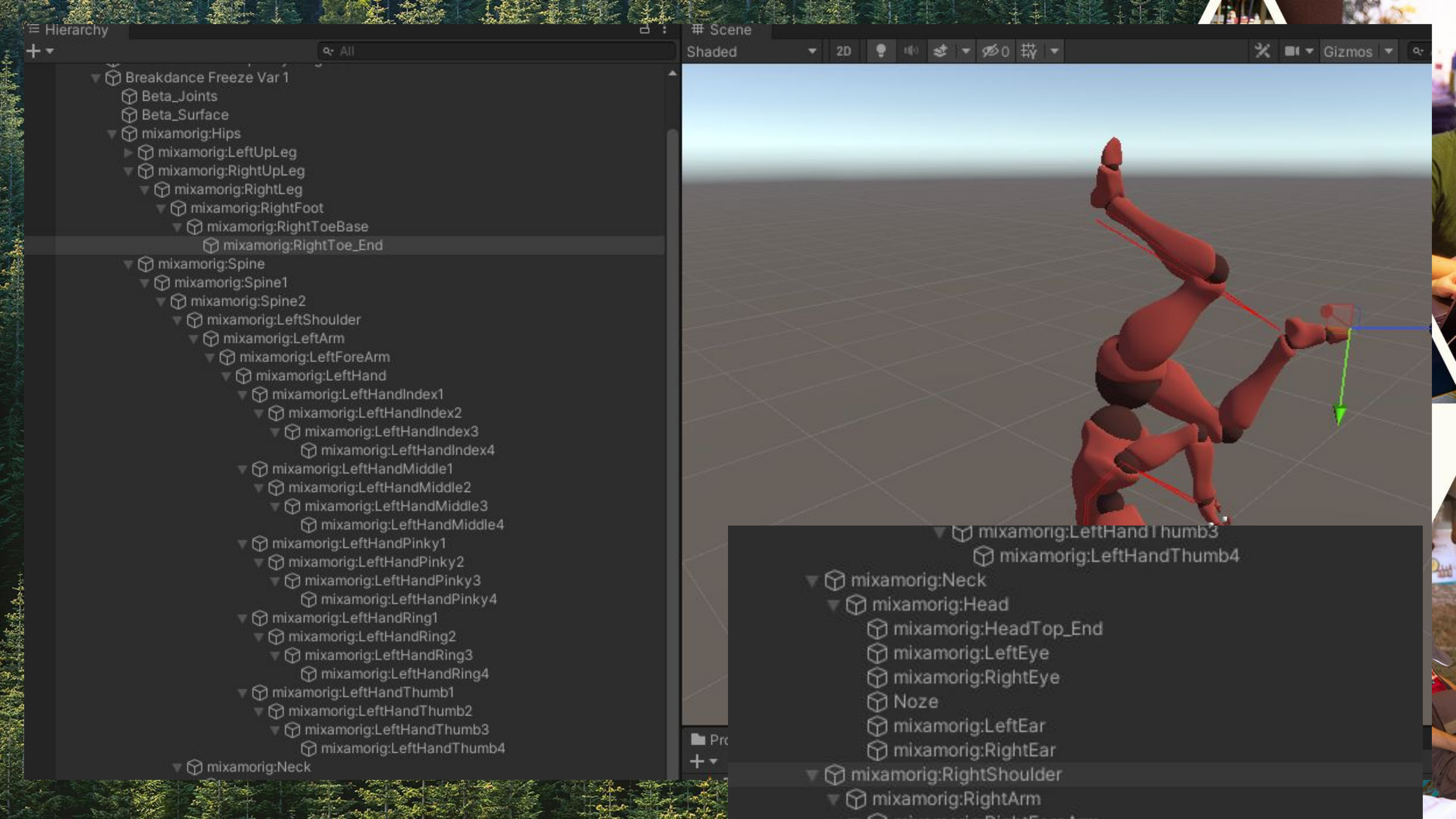
Pro Melee Axe Pack

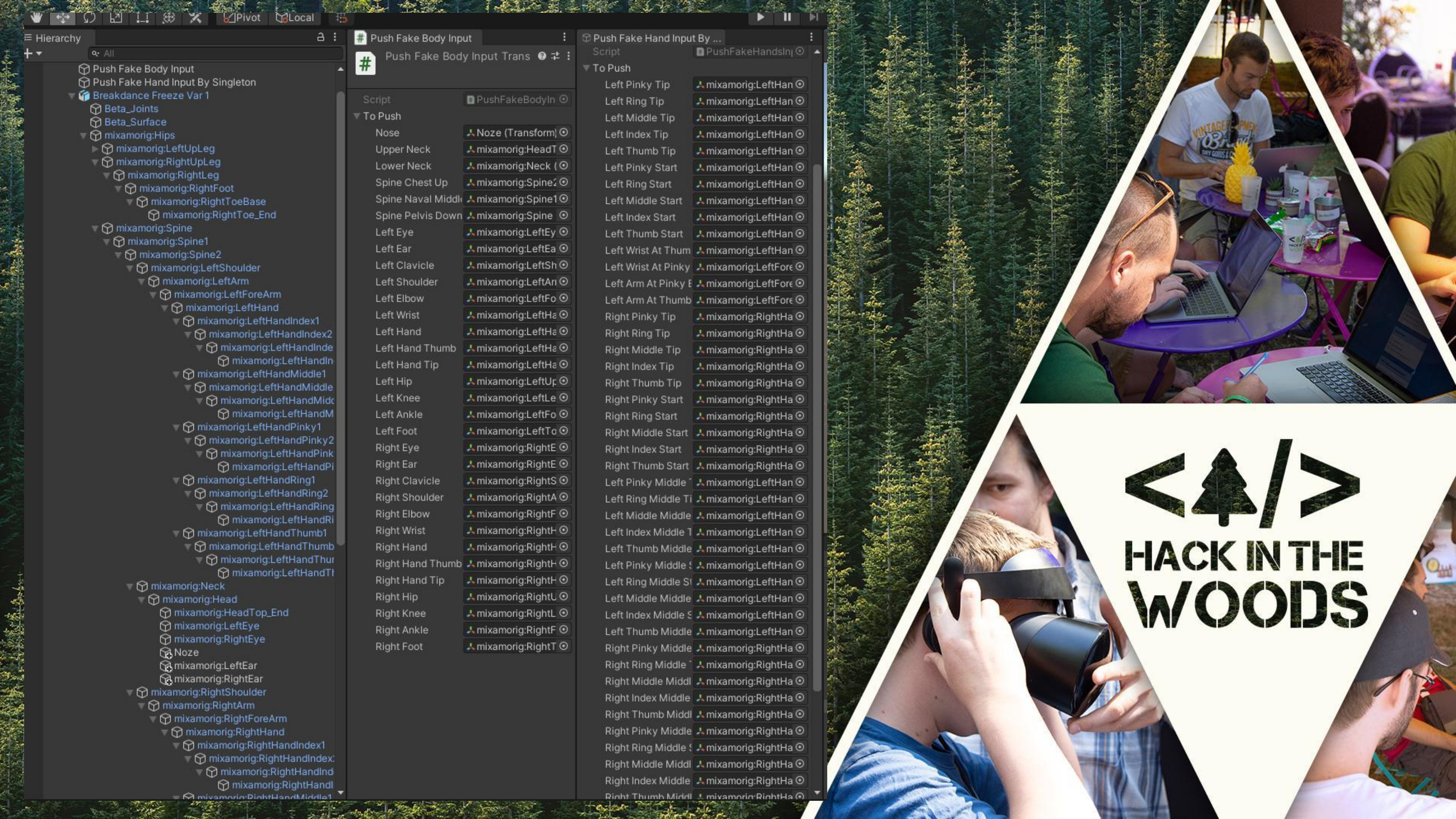
Basic Shooter Pack

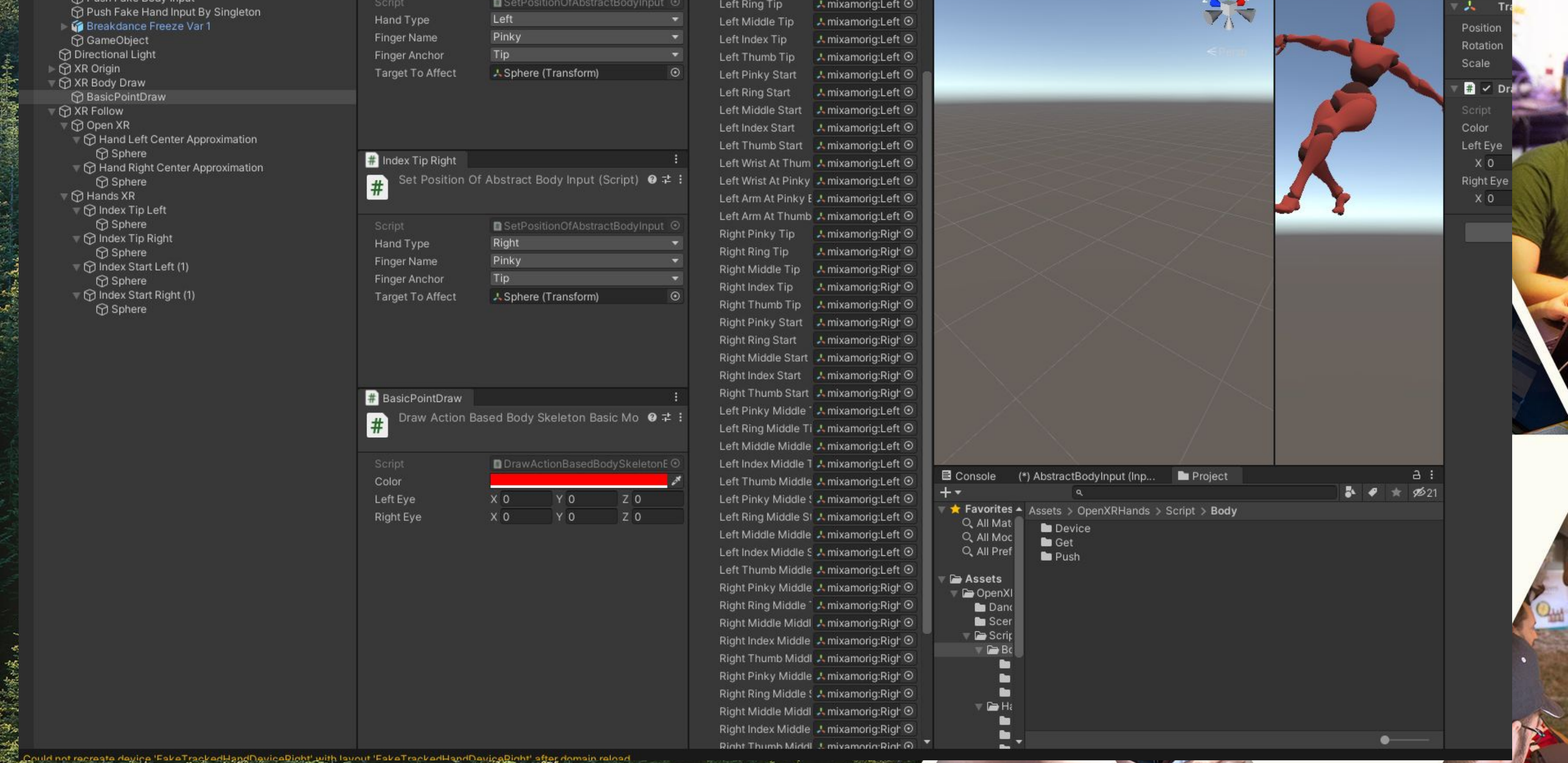
Standing Torch Light Torch



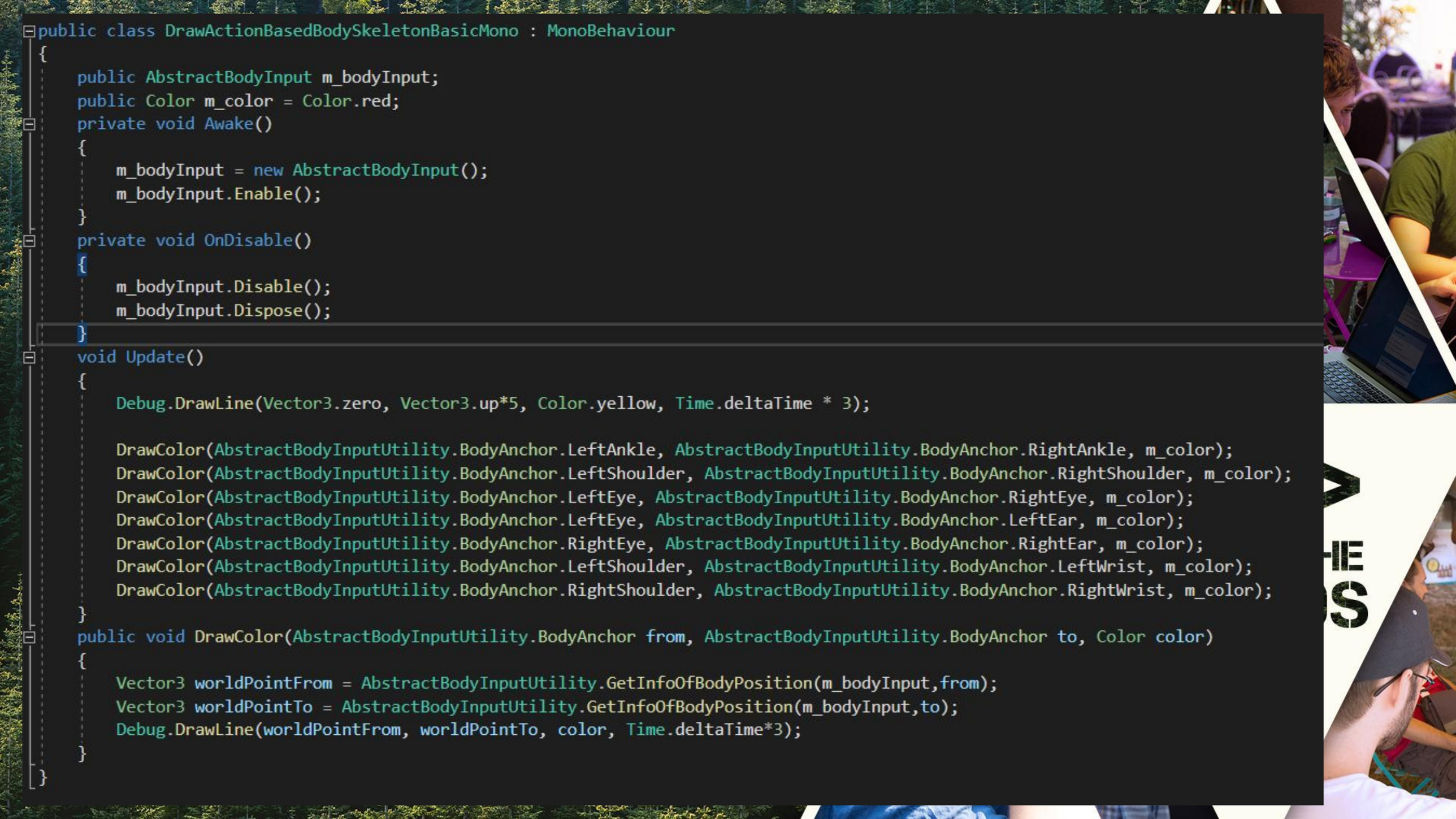
0 / 0


Hierarchy
All
Breakdance Freeze Var 1
Beta_Joints
Beta_Surface
mixamorig:Hips
mixamorig:LeftUpLeg
mixamorig:RightUpLeg
mixamorig:RightLeg
mixamorig:RightFoot
mixamorig:RightToeBase
mixamorig:RightToe_End
mixamorig:Spine
mixamorig:Spine1
mixamorig:Spine2
mixamorig:LeftShoulder
mixamorig:LeftArm
mixamorig:LeftForeArm
mixamorig:LeftHand
mixamorig:LeftHandIndex1
mixamorig:LeftHandIndex2
mixamorig:LeftHandIndex3
mixamorig:LeftHandIndex4
mixamorig:LeftHandMiddle1
mixamorig:LeftHandMiddle2
mixamorig:LeftHandMiddle3
mixamorig:LeftHandMiddle4
mixamorig:LeftHandPinky1
mixamorig:LeftHandPinky2
mixamorig:LeftHandPinky3
mixamorig:LeftHandPinky4
mixamorig:LeftHandRing1
mixamorig:LeftHandRing2
mixamorig:LeftHandRing3
mixamorig:LeftHandRing4
mixamorig:LeftHandThumb1
mixamorig:LeftHandThumb2
mixamorig:LeftHandThumb3
mixamorig:LeftHandThumb4
mixamorig:Neck
Scene
Shaded
2D
Gizmos
mixamorig:LeftHandThumb3
mixamorig:LeftHandThumb4
mixamorig:Neck
mixamorig:Head
mixamorig:HeadTop_End
mixamorig:LeftEye
mixamorig:RightEye
Noze
mixamorig:LeftEar
mixamorig:RightEar
mixamorig:RightShoulder
mixamorig:RightArm

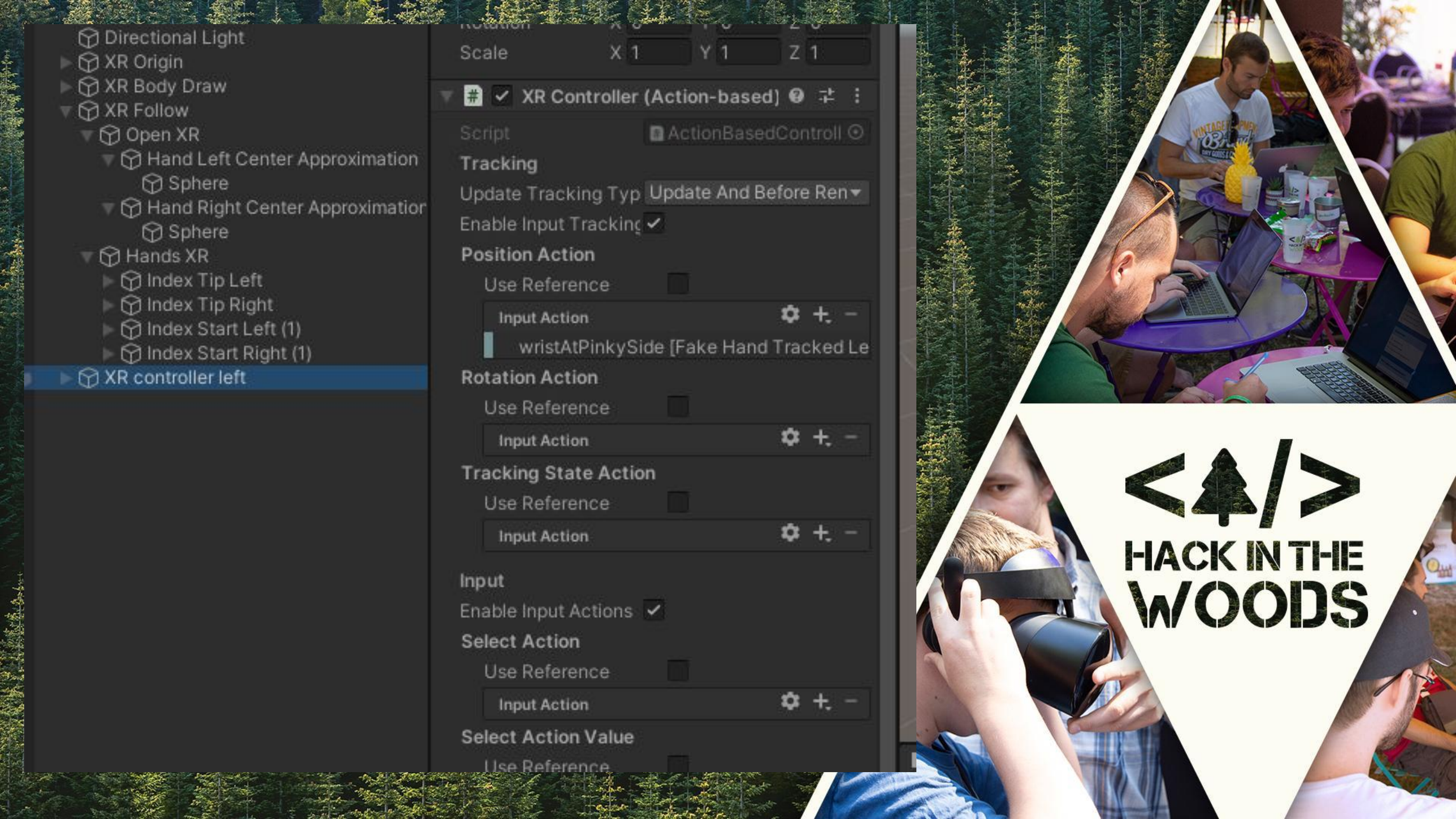

Pivot
Local
Hierarchy
All
Push Fake Body Input
Push Fake Hand Input By Singleton
Breakdance Freeze Var 1
Beta_Joints
Beta_Surface
mixamorig:Hips
mixamorig:LeftUpLeg
mixamorig:RightUpLeg
mixamorig:RightLeg
mixamorig:RightFoot
mixamorig:RightToeBase
mixamorig:RightToe_End
mixamorig:Spine
mixamorig:Spine1
mixamorig:Spine2
mixamorig:LeftShoulder
mixamorig:LeftArm
mixamorig:LeftForeArm
mixamorig:LeftHand
mixamorig:LeftHandIndex1
mixamorig:LeftHandIndex2
mixamorig:LeftHandMiddle1
mixamorig:LeftHandPinky1
mixamorig:LeftHandPinky2
mixamorig:LeftHandRing1
mixamorig:LeftHandRing2
mixamorig:LeftHandThumb1
mixamorig:Neck
mixamorig:Head
mixamorig:HeadTop_End
mixamorig:LeftEye
mixamorig:RightEye
Noze
mixamorig:LeftEar
mixamorig:RightEar
mixamorig:RightShoulder
mixamorig:RightArm
mixamorig:RightForeArm
mixamorig:RightHand
mixamorig:RightHandIndex1
Push Fake Body Input
Push Fake Body Input Trans
Script
PushFakeBodyIn
To Push
Nose
Noze (Transform)
Upper Neck
mixamorig:HeadT
Lower Neck
mixamorig:Neck (
Spine Chest Up
mixamorig:Spine2
Spine Naval Middl
mixamorig:Spine1
Spine Pelvis Down
mixamorig:Spine
Left Eye
mixamorig:LeftEy
Left Ear
mixamorig:LeftEa
Left Clavicle
mixamorig:LeftSh
Left Shoulder
mixamorig:LeftAr
Left Elbow
mixamorig:LeftFo
Left Wrist
mixamorig:LeftHa
Left Hand
mixamorig:LeftHa
Left Hand Thumb
mixamorig:LeftHa
Left Hand Tip
mixamorig:LeftHa
Left Hip
mixamorig:LeftUp
Left Knee
mixamorig:LeftLe
Left Ankle
mixamorig:LeftFo
Left Foot
mixamorig:LeftTo
Right Eye
mixamorig:RightE
Right Ear
mixamorig:RightE
Right Clavicle
mixamorig:RightS
Right Shoulder
mixamorig:RightA
Right Elbow
mixamorig:RightF
Right Wrist
mixamorig:RightH
Right Hand
mixamorig:RightH
Right Hand Thumb
mixamorig:RightH
Right Hand Tip
mixamorig:RightH
Right Hip
mixamorig:RightU
Right Knee
mixamorig:RightL
Right Ankle
mixamorig:RightF
Right Foot
mixamorig:RightT
Push Fake Hand Input By ...
Script
PushFakeHandsInp
To Push
Left Pinky Tip
Left Ring Tip
Left Middle Tip
Left Index Tip
Left Thumb Tip
Left Pinky Start
Left Ring Start
Left Middle Start
Left Index Start
Left Thumb Start
Left Wrist At Thum
Left Wrist At Pinky
Left Arm At Pinky E
Left Arm At Thumb
Right Pinky Tip
Right Ring Tip
Right Middle Tip
Right Index Tip
Right Thumb Tip
Right Pinky Start
Right Ring Start
Right Middle Start
Right Index Start
Right Thumb Start
Left Pinky Middle
Left Ring Middle Ti
Left Middle Middle
Left Index Middle T
Left Thumb Middle
Left Pinky Middle S
Left Ring Middle St
Left Middle Middle
Left Index Middle S
Left Thumb Middle
Right Pinky Middle
Right Ring Middle
Right Middle Middl
Right Index Middle
Right Thumb Middl
Right Pinky Middle
Right Ring Middle S
Right Middle Middl
Right Index Middle
Right Thumb Middl
mixamorig:LeftHan
mixamorig:LeftFore
mixamorig:RightHa
HACK IN THE
WOODS



https://github.com/EloiStree/2022_06_21_AbstractBodyInputActionBased.git

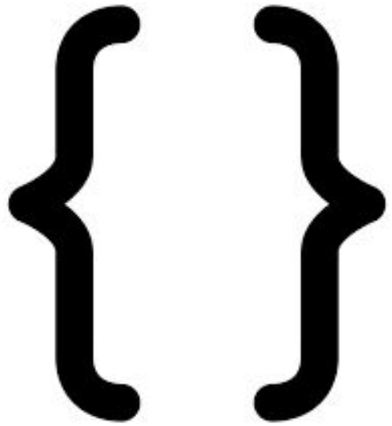


```
public class DrawActionBasedBodySkeletonBasicMono : MonoBehaviour
{
    public AbstractBodyInput m_bodyInput;
    public Color m_color = Color.red;
    private void Awake()
    {
        m_bodyInput = new AbstractBodyInput();
        m_bodyInput.Enable();
    }
    private void OnDisable()
    {
        m_bodyInput.Disable();
        m_bodyInput.Dispose();
    }
    void Update()
    {
        Debug.DrawLine(Vector3.zero, Vector3.up*5, Color.yellow, Time.deltaTime * 3);

        DrawColor(AbstractBodyInputUtility.BodyAnchor.LeftAnkle, AbstractBodyInputUtility.BodyAnchor.RightAnkle, m_color);
        DrawColor(AbstractBodyInputUtility.BodyAnchor.LeftShoulder, AbstractBodyInputUtility.BodyAnchor.RightShoulder, m_color);
        DrawColor(AbstractBodyInputUtility.BodyAnchor.LeftEye, AbstractBodyInputUtility.BodyAnchor.RightEye, m_color);
        DrawColor(AbstractBodyInputUtility.BodyAnchor.LeftEye, AbstractBodyInputUtility.BodyAnchor.LeftEar, m_color);
        DrawColor(AbstractBodyInputUtility.BodyAnchor.RightEye, AbstractBodyInputUtility.BodyAnchor.RightEar, m_color);
        DrawColor(AbstractBodyInputUtility.BodyAnchor.LeftShoulder, AbstractBodyInputUtility.BodyAnchor.LeftWrist, m_color);
        DrawColor(AbstractBodyInputUtility.BodyAnchor.RightShoulder, AbstractBodyInputUtility.BodyAnchor.RightWrist, m_color);
    }
    public void DrawColor(AbstractBodyInputUtility.BodyAnchor from, AbstractBodyInputUtility.BodyAnchor to, Color color)
    {
        Vector3 worldPointFrom = AbstractBodyInputUtility.GetInfoOfBodyPosition(m_bodyInput,from);
        Vector3 worldPointTo = AbstractBodyInputUtility.GetInfoOfBodyPosition(m_bodyInput,to);
        Debug.DrawLine(worldPointFrom, worldPointTo, color, Time.deltaTime*3);
    }
}
```


Directional Light
XR Origin
XR Body Draw
XR Follow
Open XR
Hand Left Center Approximation
Sphere
Hand Right Center Approximation
Sphere
Hands XR
Index Tip Left
Index Tip Right
Index Start Left (1)
Index Start Right (1)
XR controller left
Scale
X 1
Y 1
Z 1
XR Controller (Action-based)
Script
ActionBasedControll
Tracking
Update Tracking Typ
Update And Before Ren
Enable Input Tracking
Position Action
Use Reference
Input Action
wristAtPinkySide [Fake Hand Tracked Le
Rotation Action
Use Reference
Input Action
Tracking State Action
Use Reference
Input Action
Input
Enable Input Actions
Select Action
Use Reference
Input Action
Select Action Value
Use Reference
HACK IN THE
WOODS

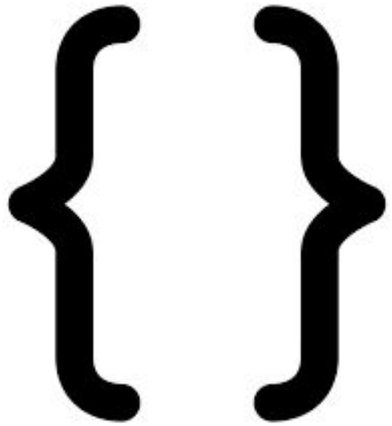
HACK IN THE WOODS

```
public class PinchDetector : MonoBehaviour
{
    public Transform m_fingerTipA;
    public Transform m_fingerTipB;
    public float m_distanceToPinch=0.05f;

    public PinchingChangeEvent m_onPinchChange;
    [Header("Debug")]
    public bool m_isPinching;

    private bool m_previousPinching;

    void Update()
    {
        if (m_previousPinching != m_isPinching) {
            m_onPinchChange.Invoke(m_isPinching);
            m_previousPinching = m_isPinching;
        }
        m_isPinching = Vector3.Distance(m_fingerTipA.position, m_fingerTipB.position) < m_distanceToPinch;
    }

    public Quaternion GetPinchRotation()
    {
        return Quaternion.Slerp(m_fingerTipA.rotation, m_fingerTipB.rotation, 0.5f);
    }

    public Vector3 GetPinchPosition()
    {
        return (m_fingerTipA.position + m_fingerTipB.position) / 2f;
    }

    [System.Serializable]
    public class PinchingChangeEvent : UnityEvent<bool> { }

}
```

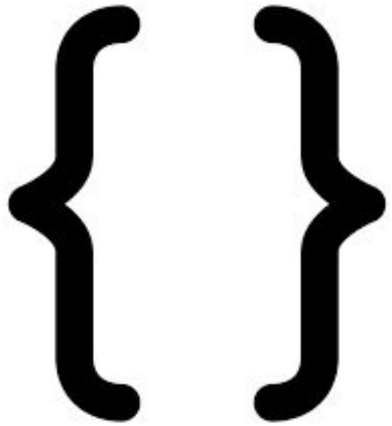


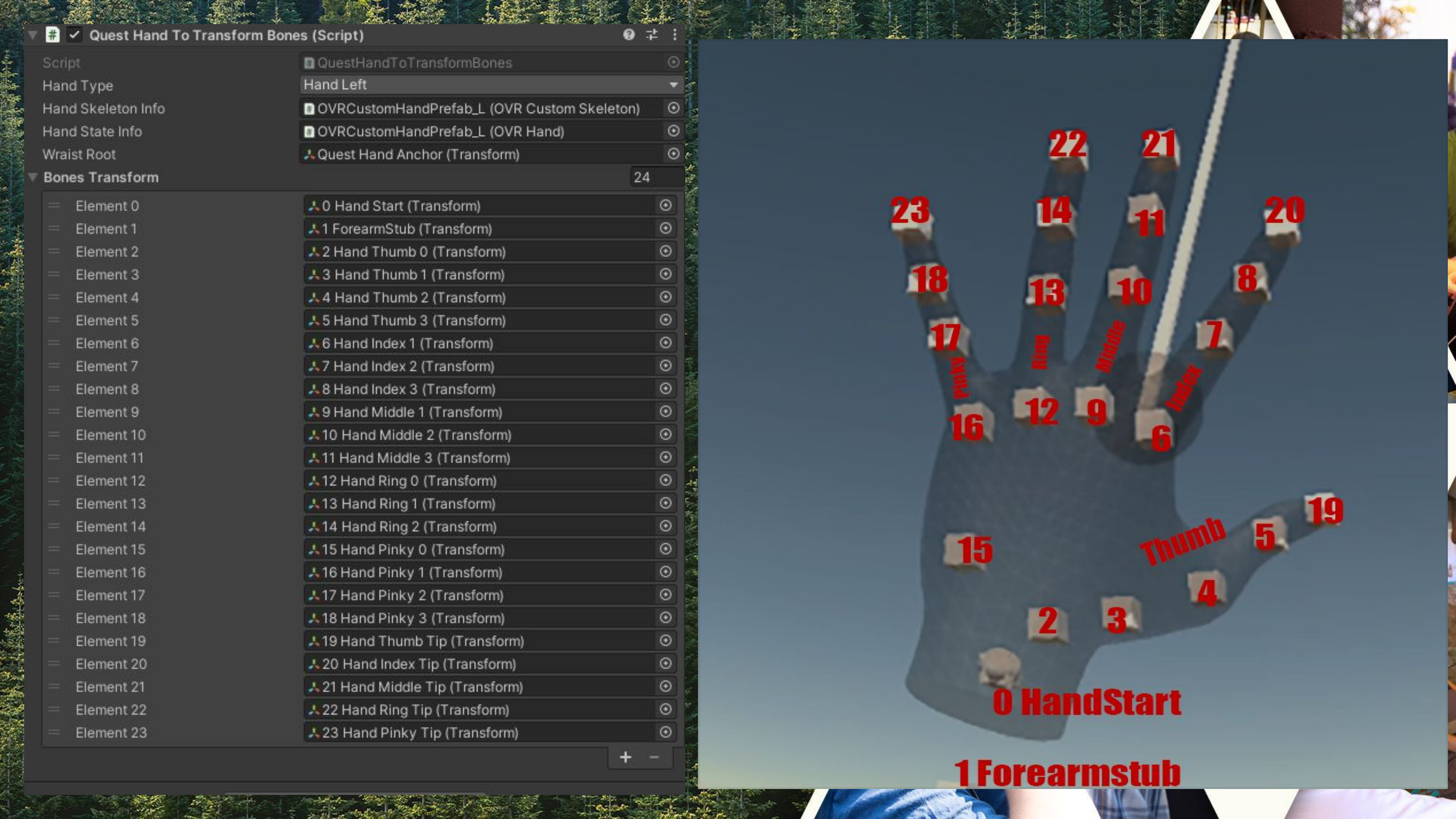
HACK IN THE WOODS

```
public class MovePinchableObjectWithTransform : MonoBehaviour
{
    public PinchDetector m_pinchDetector;
    public Transform m_oriantation;
    public LayerMask m_affectedLayers = -1;
    [Header("Debug")]
    public Quaternion m_rotationDifferenceAtPinch;
    public PinchableObject m_objectsToMove;
    public void Start()
    {
        m_pinchDetector.m_onPinchChange.AddListener(Pinching);
    }
    public void Update()
    {
        if (m_objectsToMove) {

            m_objectsToMove.transform.position = m_pinchDetector.GetPinchPosition();
            m_objectsToMove.transform.rotation = m_oriantation.rotation * m_rotationDifferenceAtPinch  ;
        }
    }
    public void Pinching(bool state)
    {
        if (state)
            StartPinching();
        else StopPinching();
    }
```

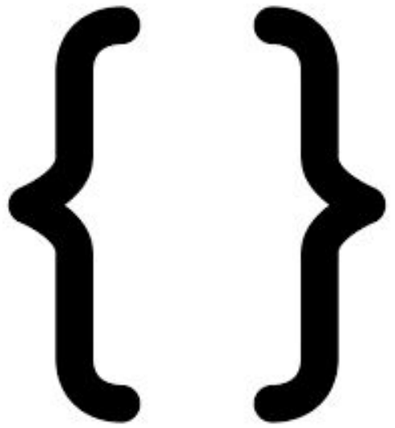


```
void StartPinching()
{
    m_objectsToMove = GetPinchableObjectInRegion();
    if (m_objectsToMove != null) {
        m_objectsToMove.NotifyAsStartPinching();
        m_rotationDifferenceAtPinch = Quaternion.Inverse(m_oriantation.rotation)* m_objectsToMove.transform.rotation
    }
    else m_rotationDifferenceAtPinch = Quaternion.identity;
}
private PinchableObject GetPinchableObjectInRegion()
{
    Collider[] touched = Physics.OverlapSphere(m_pinchDetector.GetPinchPosition(), 0.01f, m_affectedLayers);
    for (int i = 0; i < touched.Length; i++) {
        PinchableObject pinchable = touched[i].GetComponent<PinchableObject>();

        if (pinchable != null)
            return pinchable;
    }
    return null;
}
void StopPinching()
{

    if (m_objectsToMove != null) {

        m_objectsToMove.ResetVelocity();
        m_objectsToMove.NotifyAsReleasing();
        m_objectsToMove = null;
    }


}
```

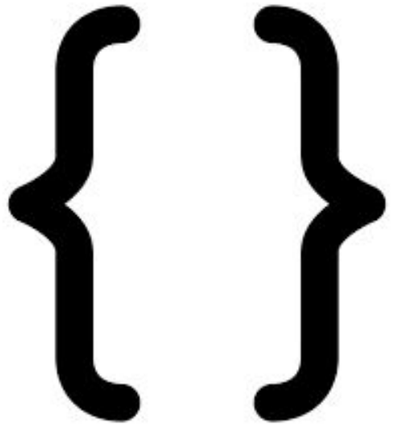


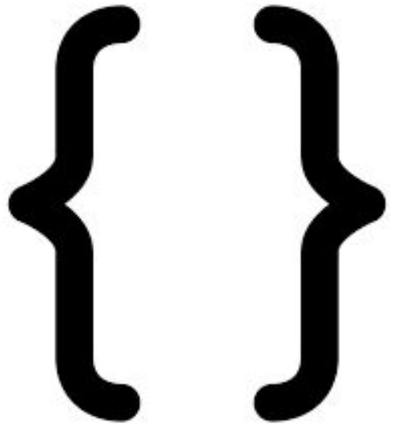

Quest Hand To Transform Bones (Script)
Script
QuestHandToTransformBones
Hand Type
Hand Left
Hand Skeleton Info
OVRCustomHandPrefab_L (OVR Custom Skeleton)
Hand State Info
OVRCustomHandPrefab_L (OVR Hand)
Wraist Root
Quest Hand Anchor (Transform)
Bones Transform
24
Element 0
0 Hand Start (Transform)
Element 1
1 ForearmStub (Transform)
Element 2
2 Hand Thumb 0 (Transform)
Element 3
3 Hand Thumb 1 (Transform)
Element 4
4 Hand Thumb 2 (Transform)
Element 5
5 Hand Thumb 3 (Transform)
Element 6
6 Hand Index 1 (Transform)
Element 7
7 Hand Index 2 (Transform)
Element 8
8 Hand Index 3 (Transform)
Element 9
9 Hand Middle 1 (Transform)
Element 10
10 Hand Middle 2 (Transform)
Element 11
11 Hand Middle 3 (Transform)
Element 12
12 Hand Ring 0 (Transform)
Element 13
13 Hand Ring 1 (Transform)
Element 14
14 Hand Ring 2 (Transform)
Element 15
15 Hand Pinky 0 (Transform)
Element 16
16 Hand Pinky 1 (Transform)
Element 17
17 Hand Pinky 2 (Transform)
Element 18
18 Hand Pinky 3 (Transform)
Element 19
19 Hand Thumb Tip (Transform)
Element 20
20 Hand Index Tip (Transform)
Element 21
21 Hand Middle Tip (Transform)
Element 22
22 Hand Ring Tip (Transform)
Element 23
23 Hand Pinky Tip (Transform)
22
21
23
14
11
20
18
13
10
8
17
7
Pinky
Ring
Middle
Index
16
12
9
6
19
15
Thumb
5
4
2
3
0 HandStart
1 Forearmstub

```
using System;
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using static OVRSkeleton;

public class QuestHandToTransformBones : MonoBehaviour
{
    public SkeletonType m_handType;
    public OVRSkeleton m_handSkeletonInfo;
    public OVRHand m_handStateInfo;
    public Transform m_wraistRoot;
    public Transform[] m_bonesTransform;
    private Vector3 m_leftAdjustementRotation = new Vector3(0,90,180);
    private Vector3 m_rightAdjustmentRotation = new Vector3(0, -90, 0);
    public void Update()
    {
        if (!m_handStateInfo.IsTracked)
        {SetAsNotTracked();}
        else {SetPositionOfTrackedHandWithBones();}
    }
```



```
private void SetPositionOfTrackedHandWithBones()
{
    int boneId = -1;
    Transform selectedBone = null;
    for (int i = 0; i < m_handSkeletonInfo.Bones.Count; i++)
    {
        boneId = (int)m_handSkeletonInfo.Bones[i].Id;
        if (boneId >= 0 && boneId < m_bonesTransform.Length)
        {
            selectedBone = m_bonesTransform[boneId];
            if (boneId == 0)
            {
                if (m_wraistRoot != null && selectedBone != null)
                {
                    m_wraistRoot.position = selectedBone.position;
                    m_wraistRoot.rotation = selectedBone.rotation;
                }
            }
            m_bonesTransform[i].gameObject.SetActive(true);
            selectedBone.position = m_handSkeletonInfo.Bones[i].Transform.position;
            ApplyRotationToPutStandardToTheDirectionOfHandBones(selectedBone, i);
        }
    }
}
```





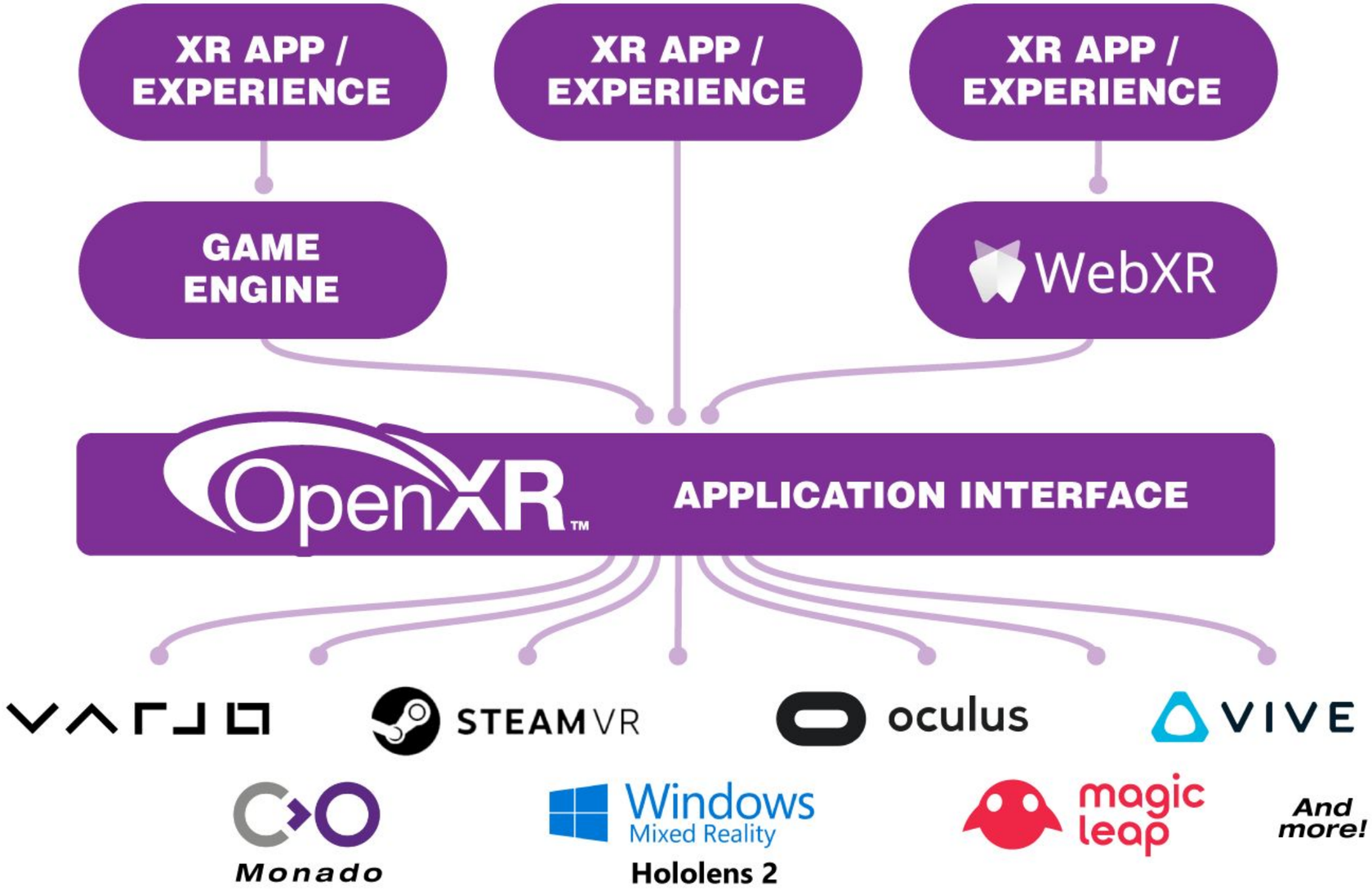
```
private void ApplyRotationToPutStandardToTheDirectionOfHandBones(Transform selectedBone, int i)
{
    Quaternion rotationAdjustement = Quaternion.identity;
    if (m_handType == SkeletonType.HandLeft)
        rotationAdjustement = Quaternion.Euler(m_leftAdjustementRotation)*Quaternion.Euler(0,180,0);
    if (m_handType == SkeletonType.HandRight)
        rotationAdjustement = Quaternion.Euler(m_rightAdjustmentRotation) * Quaternion.Euler(0, 180, 0);

    selectedBone.rotation = m_handSkeletonInfo.Bones[i].Transform.rotation * rotationAdjustement;
}
```

Open XR

KHRONOS GROUP

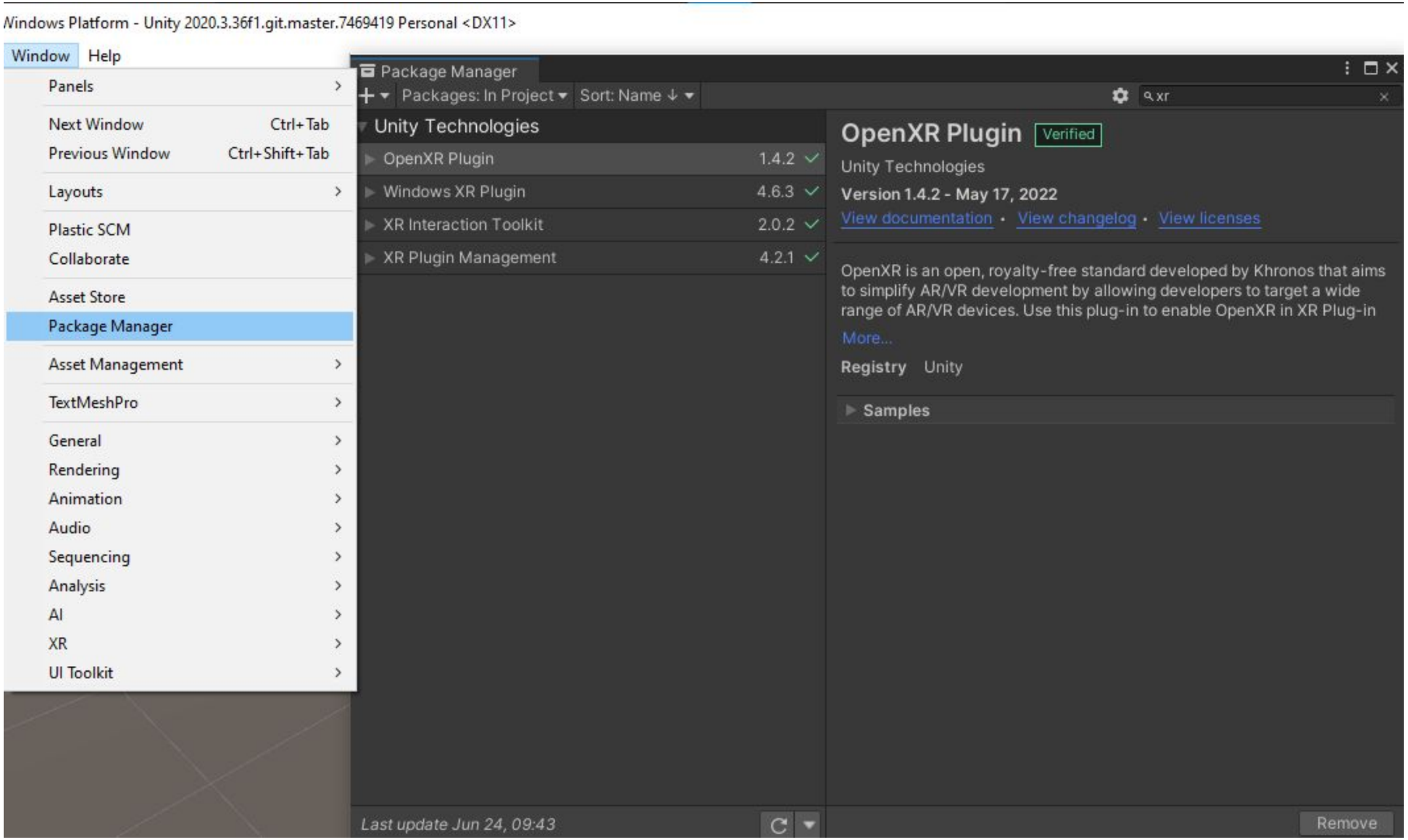
CONNECTING SOFTWARE TO SILICON

Developers Conformance Membership News & Events About

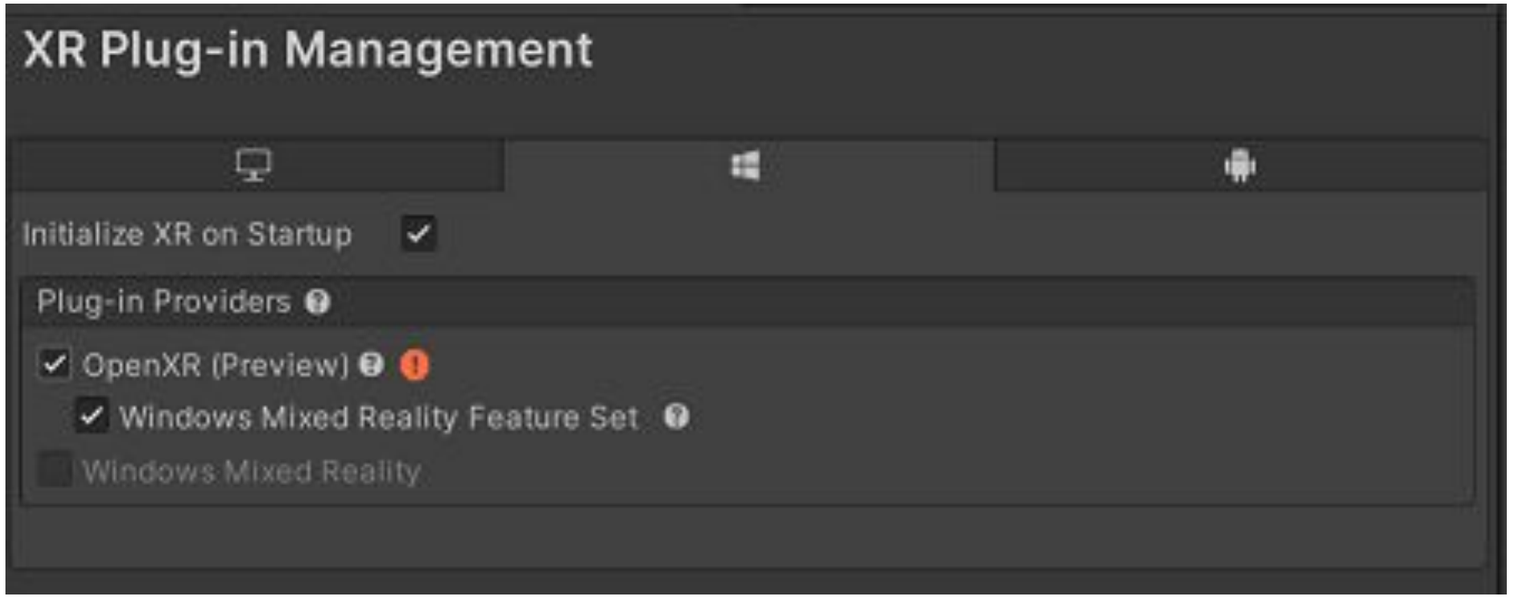


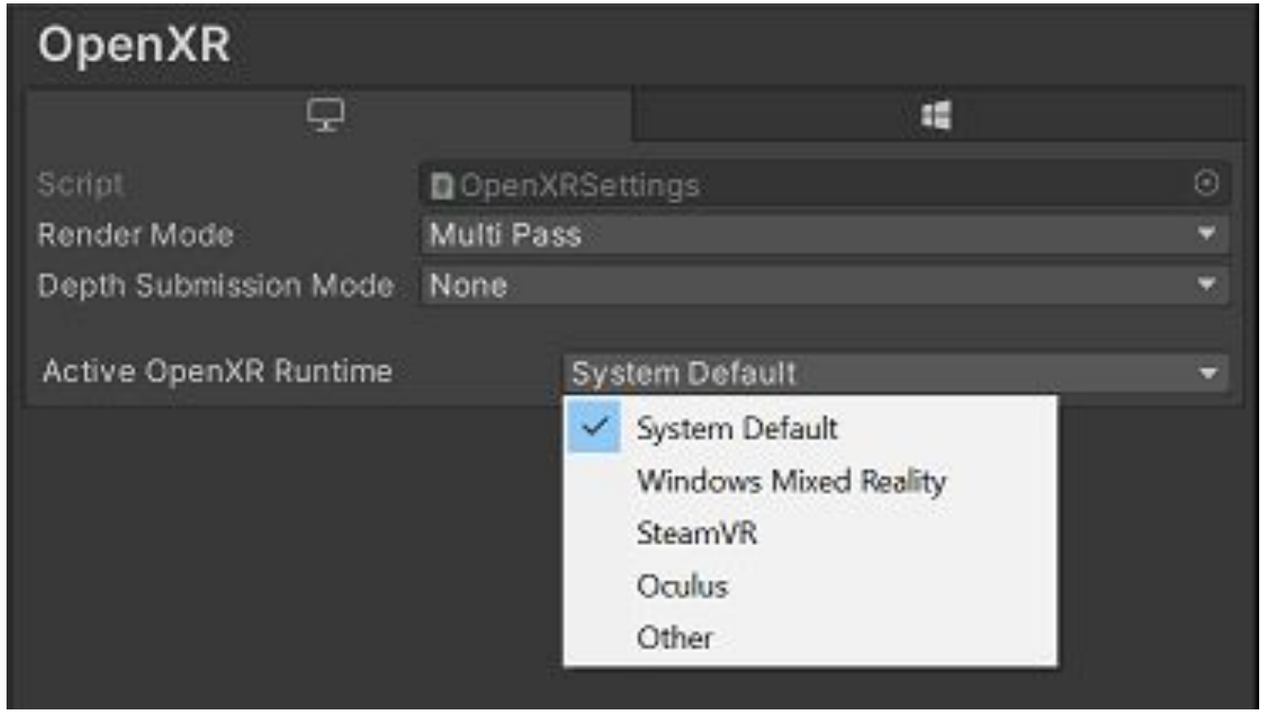
OpenXR provides a single cross-platform, high-performance API between applications and all conformant devices.


AMD
antilatency
AREA
arm
ATLATL
AUTODESK
blender
DIMENCO
DisplayLink XR
EPIC GAMES
ERICSSON
Google
HAPTICS INDUSTRY FORUM
HOLOCHIP
htc
HUAWEI
Imagination
INFINADECK
intel
LG
immersion
JUICE
MTA SZTAKI
LVV
logitech
LUNARG
magic leap
MEDIATEK
Meta
Microsoft
mozilla
National Institute of Standards and Technology
NOKIA
NVIDIA
oppo
Pico
pluto
Qualcomm
RAZER
Rokid
SAMSUNG
SILICONARTS
SONY
tobii
ultraleap
Unity
UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN
UX3D
VALVE
VARJO
VeriSilicon
兆芯
XEED
zSpace
HACK IN THE WOODS

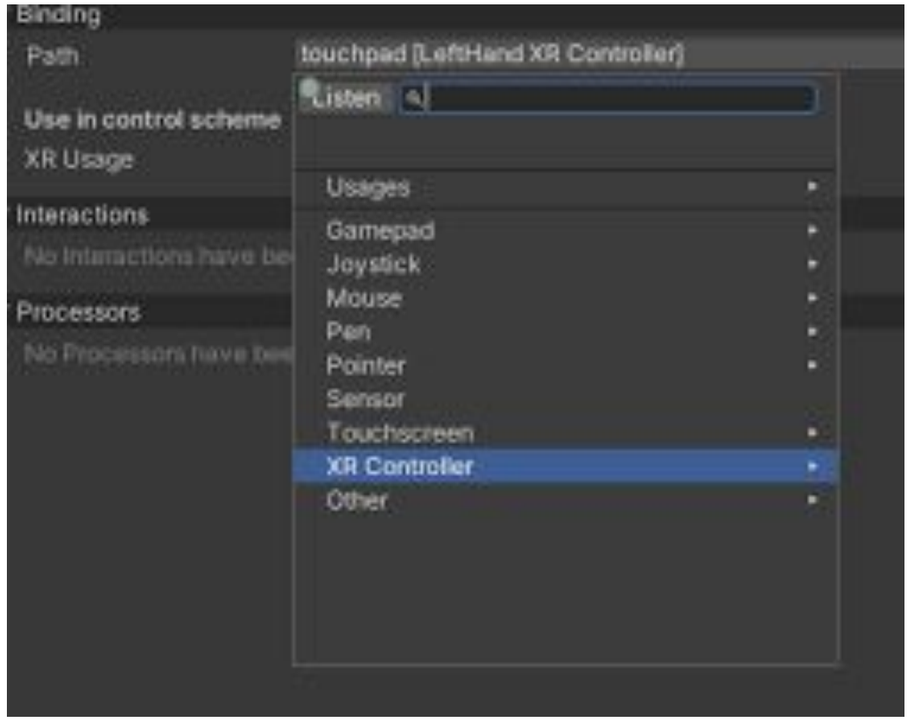


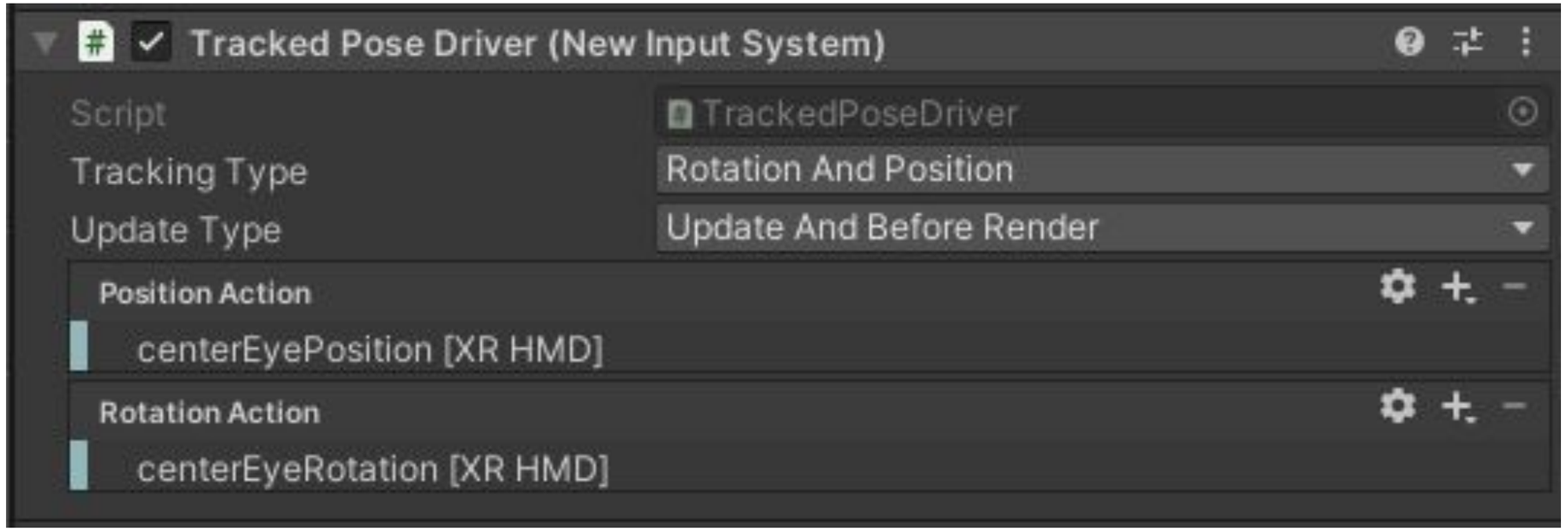
HACK IN THE WOODS





HACK IN THE WOODS

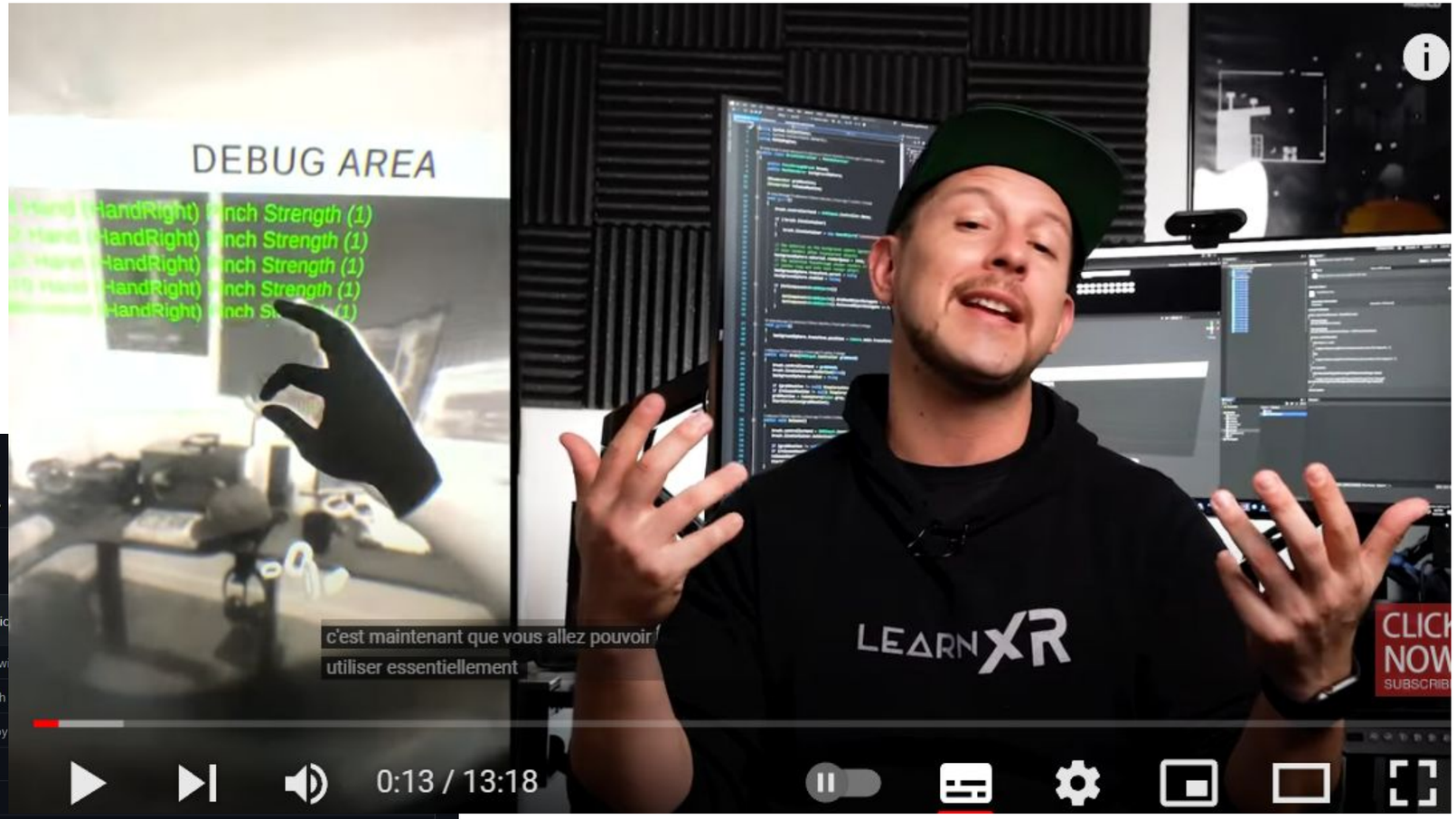

Binding
Path
touchpad [LeftHand XR Controller]
Use in control scheme
XR Usage
Interactions
Processors
Listen
Usages
Gamepad
Joystick
Mouse
Pen
Pointer
Sensor
Touchscreen
XR Controller
Other

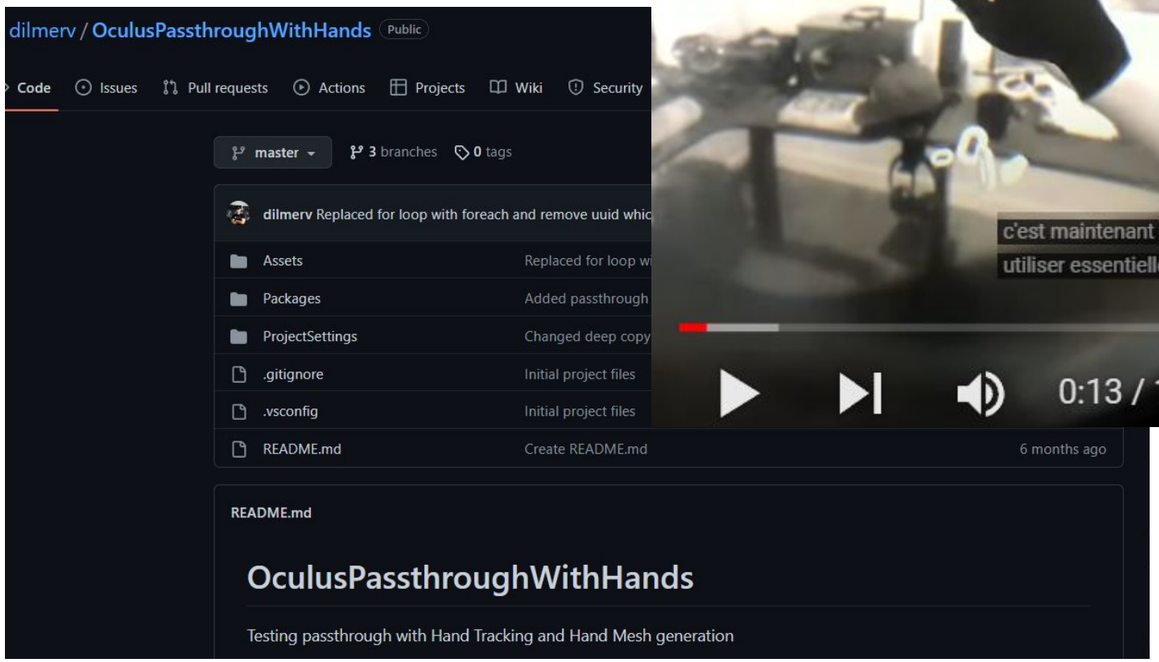

Tracked Pose Driver (New Input System)
Script
TrackedPoseDriver
Tracking Type
Rotation And Position
Update Type
Update And Before Render
Position Action
centerEyePosition [XR HMD]
Rotation Action
centerEyeRotation [XR HMD]

HACK IN THE
WOODS

Setup Oculus Passthrough with hands

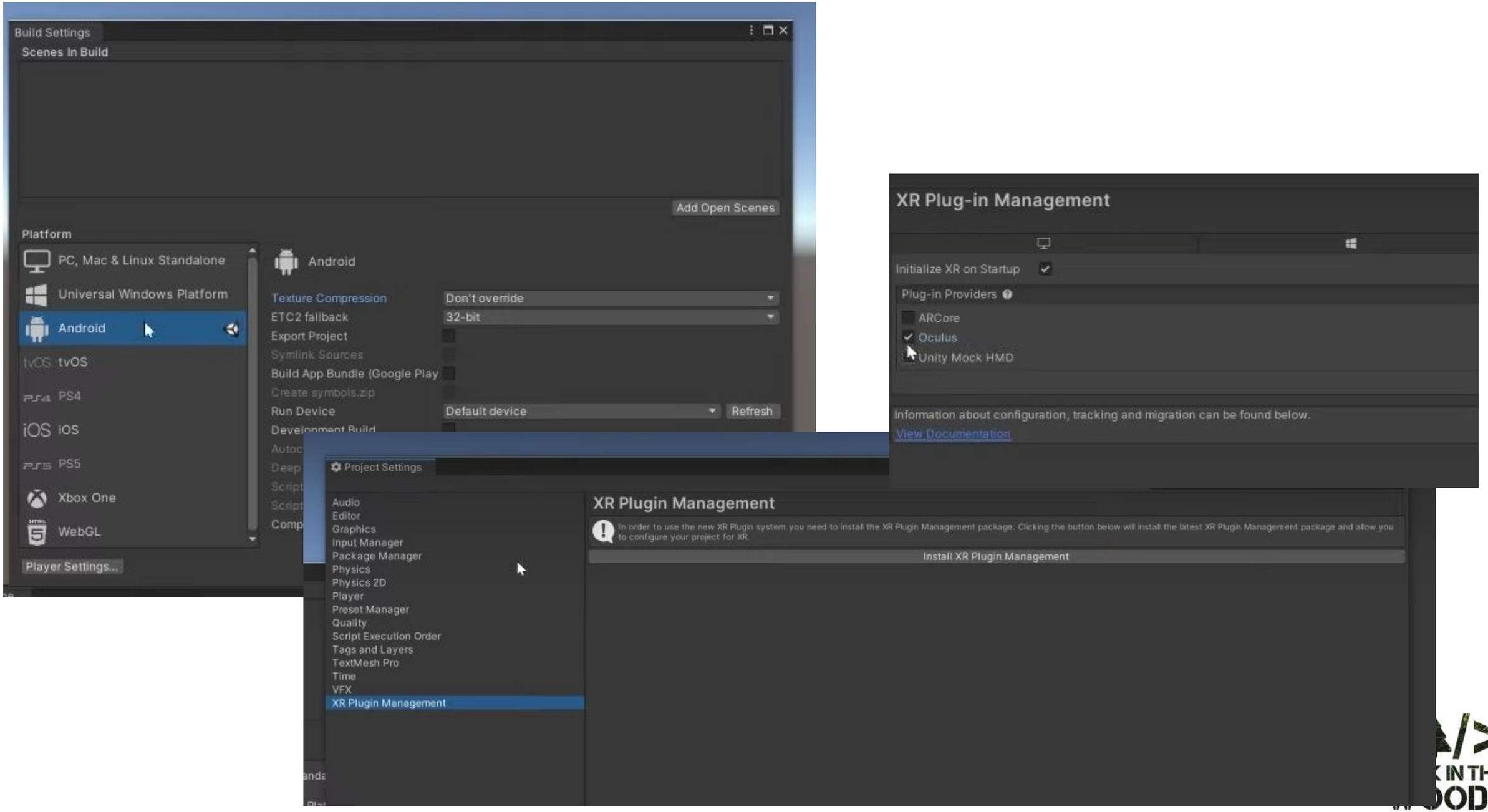
HACK IN THE WOODS

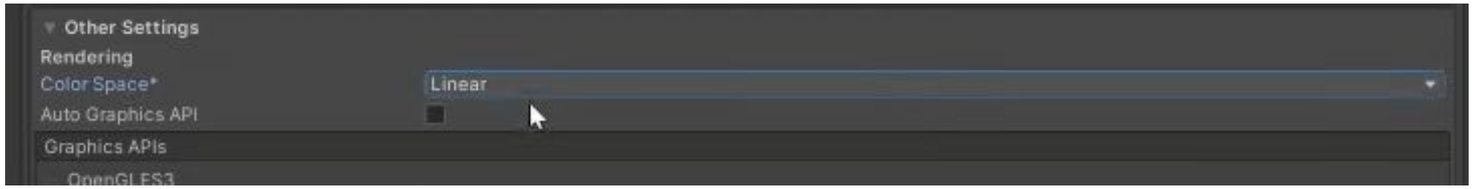


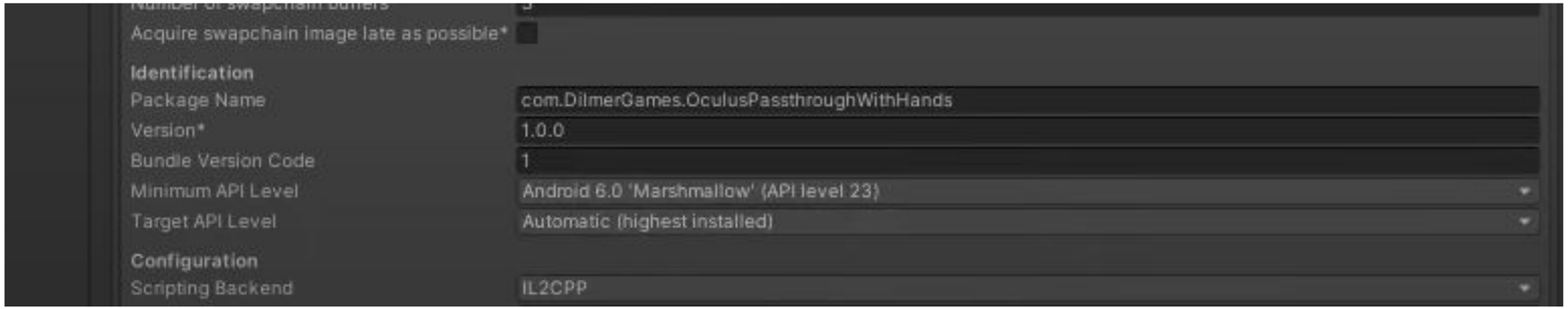


https://www.youtube.com/watch?v=g2PnINKleTM
https://github.com/dilmerv/OculusPassthroughWithHands

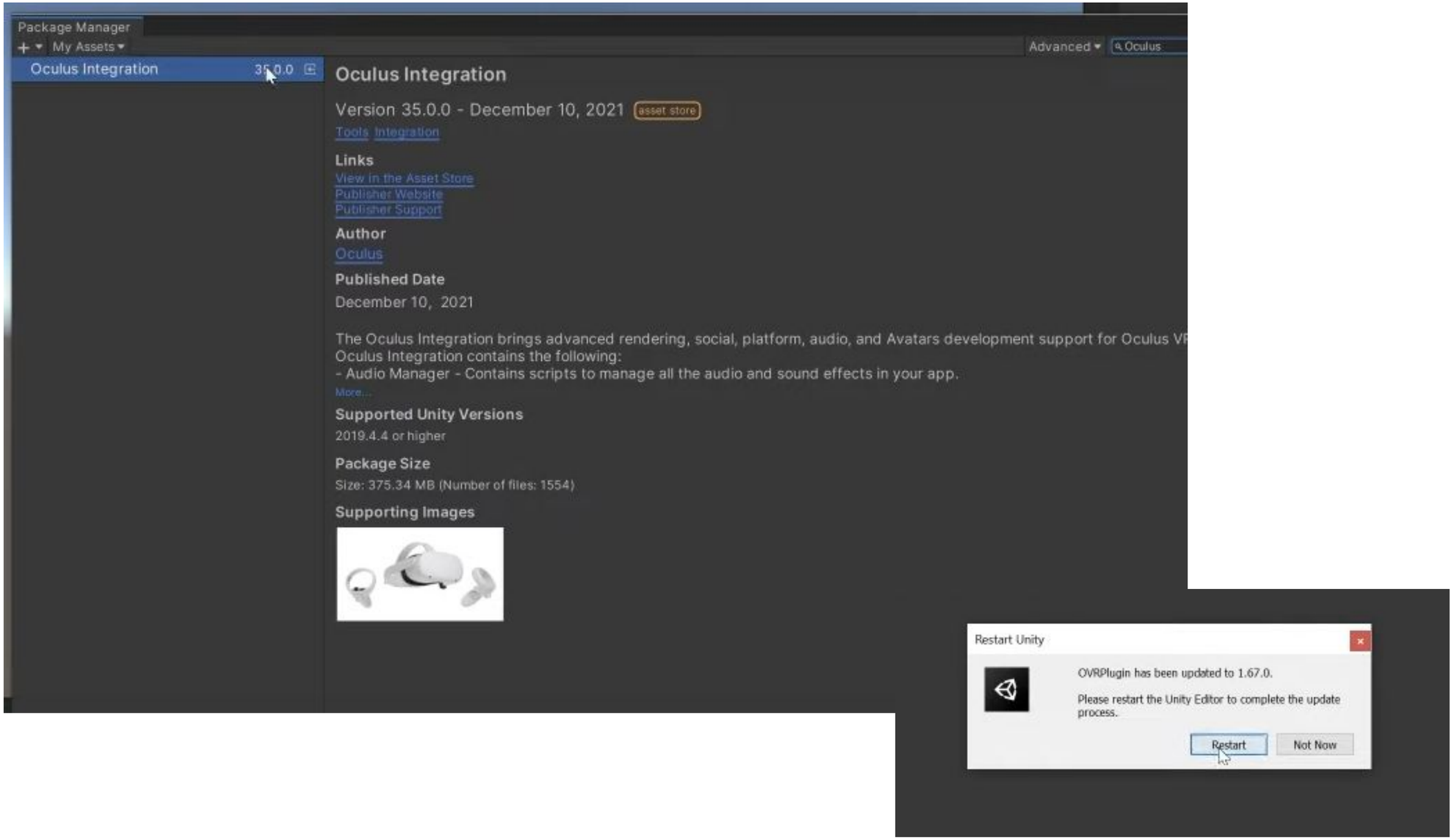
HACK IN THE WOODS


Build Settings
Scenes In Build
Add Open Scenes
Platform
PC, Mac & Linux Standalone
Universal Windows Platform
Android
tvOS
PS4
iOS
PS5
Xbox One
WebGL
Player Settings...
Android
Texture Compression
Don't override
ETC2 fallback
32-bit
Export Project
Symlink Sources
Build App Bundle (Google Play
Create symbols.zip
Run Device
Default device
Refresh
Development Build
Project Settings
Audio
Editor
Graphics
Input Manager
Package Manager
Physics
Physics 2D
Player
Preset Manager
Quality
Script Execution Order
Tags and Layers
TextMesh Pro
Time
VFX
XR Plugin Management
XR Plugin Management
In order to use the new XR Plugin system you need to install the XR Plugin Management package. Clicking the button below will install the latest XR Plugin Management package and allow you to configure your project for XR.
Install XR Plugin Management
XR Plug-in Management
Initialize XR on Startup
Plug-in Providers
ARCore
Oculus
Unity Mock HMD
Information about configuration, tracking and migration can be found below.
View Documentation

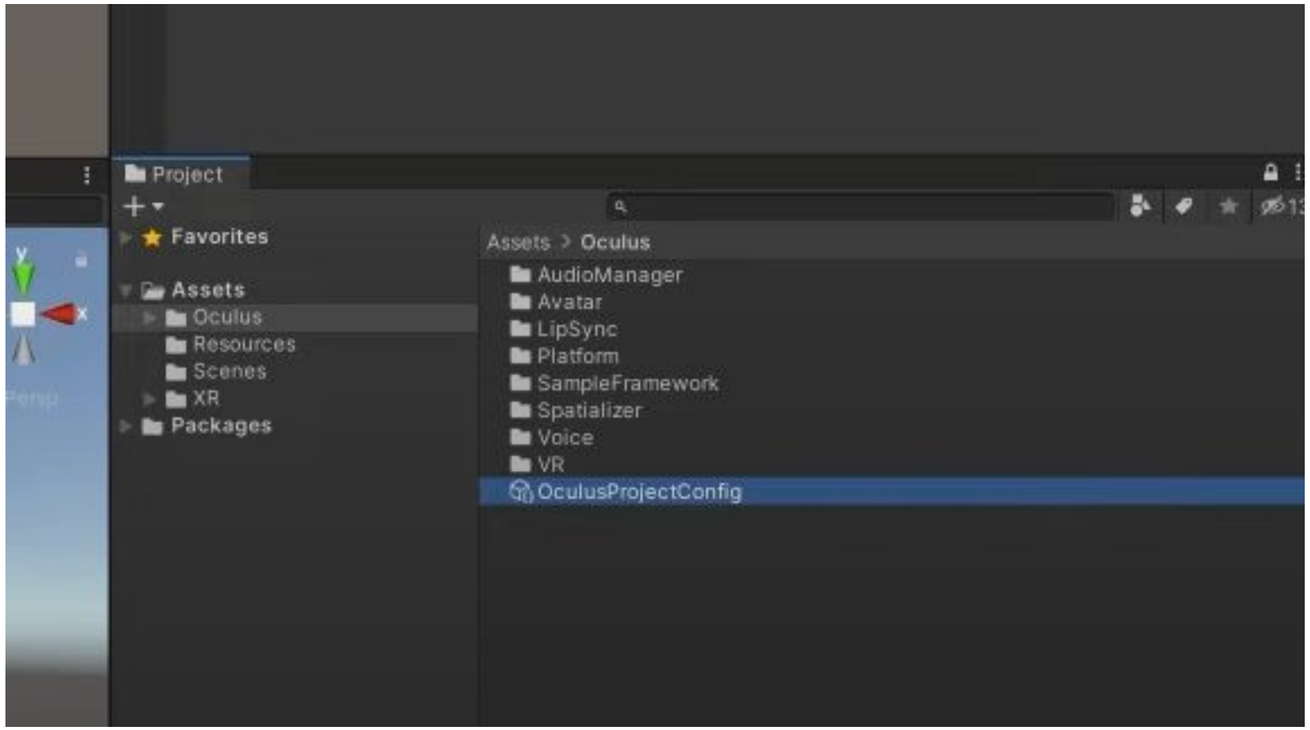


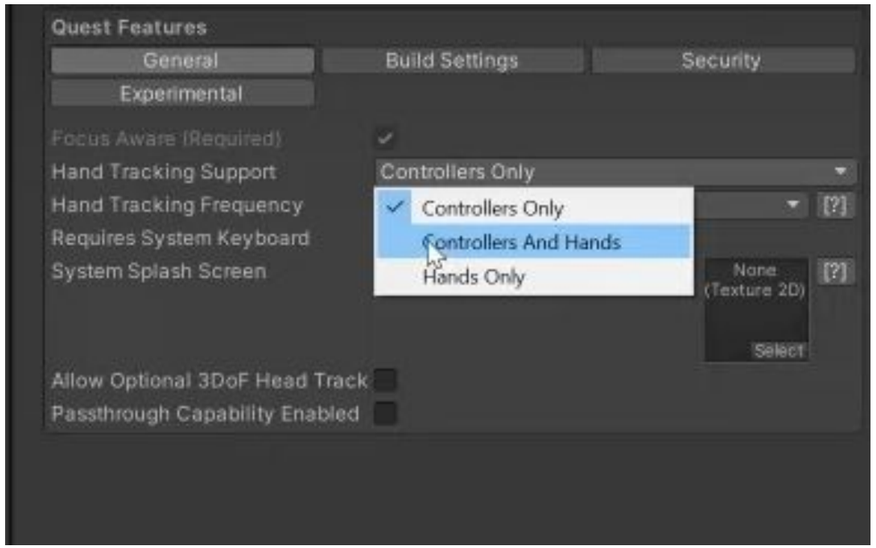


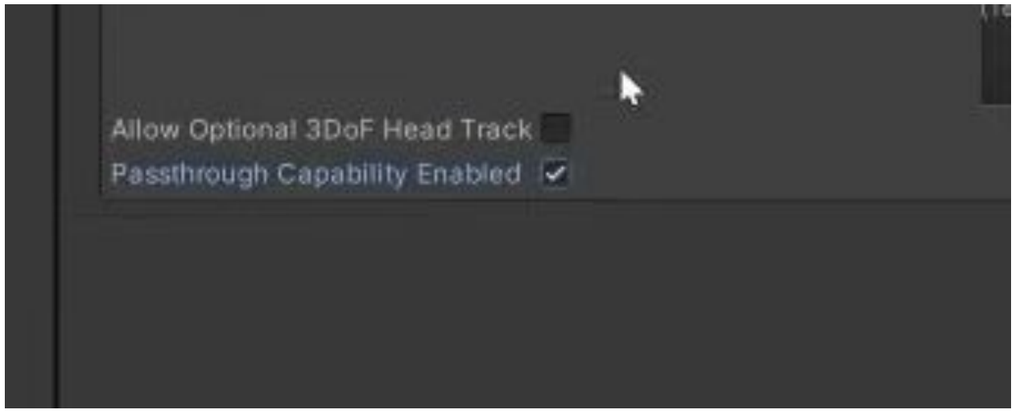
HACK IN THE WOODS



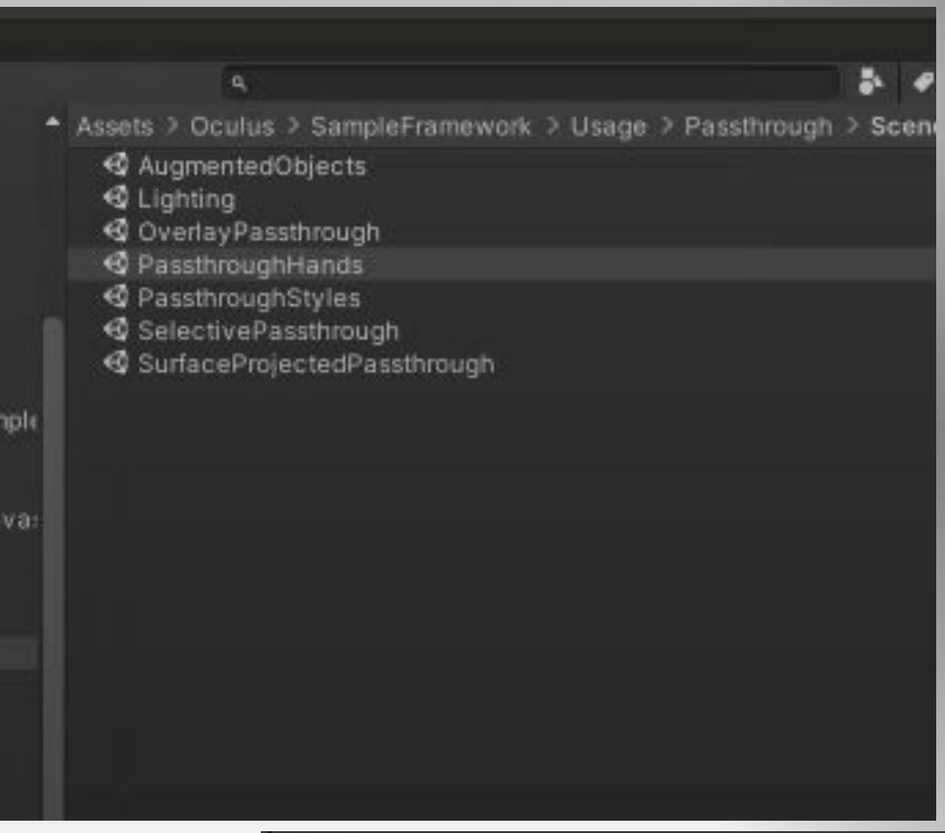
HACK IN THE WOODS

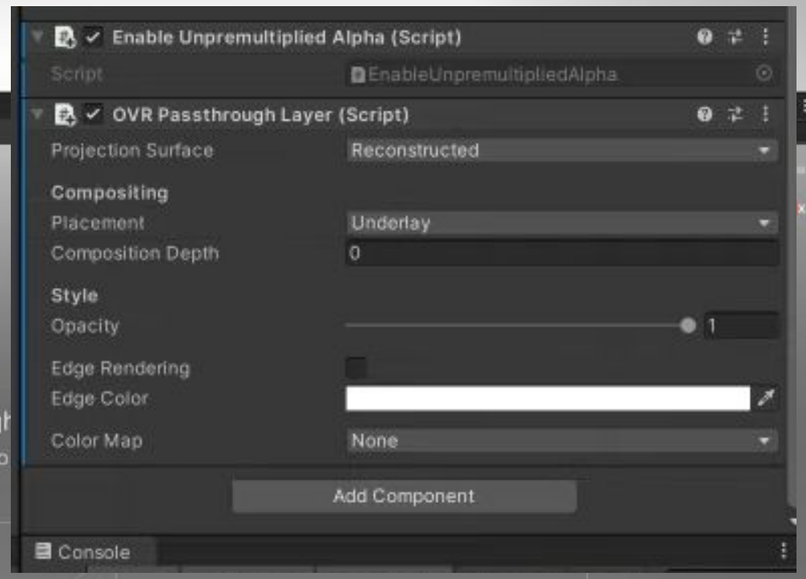

Project
Favorites
Assets
Oculus
Resources
Scenes
XR
Packages
Assets > Oculus
AudioManager
Avatar
LipSync
Platform
SampleFramework
Spatializer
Voice
VR
OculusProjectConfig


Quest Features
General
Build Settings
Security
Experimental
Focus Aware (Required)
Hand Tracking Support
Controllers Only
Hand Tracking Frequency
Requires System Keyboard
System Splash Screen
Controllers Only
Controllers And Hands
Hands Only
None
(Texture 2D)
Select
Allow Optional 3DoF Head Track
Passthrough Capability Enabled

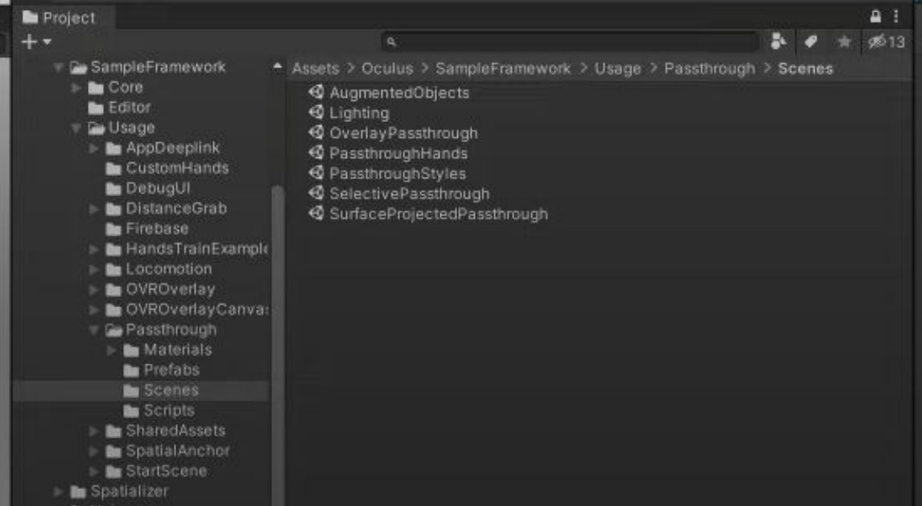

Allow Optional 3DoF Head Track
Passthrough Capability Enabled

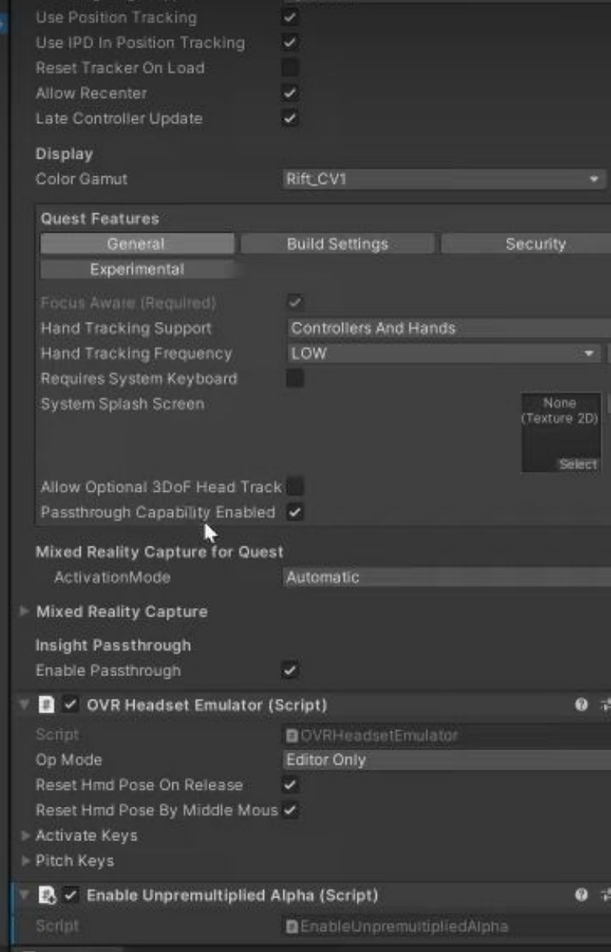
HACK IN THE
WOODS

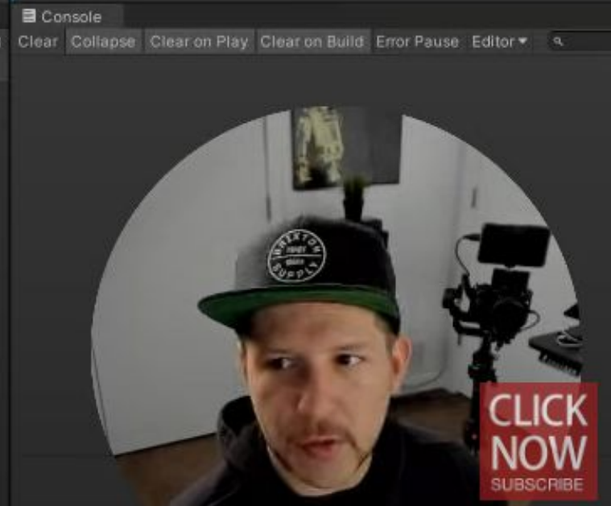

Assets > Oculus > SampleFramework > Usage > Passthrough > Scene
AugmentedObjects
Lighting
OverlayPassthrough
PassthroughHands
PassthroughStyles
SelectivePassthrough
SurfaceProjectedPassthrough


Enable Unpremultiplied Alpha (Script)
Script
EnableUnpremultipliedAlpha
OVR Passthrough Layer (Script)
Projection Surface
Reconstructed
Compositing
Placement
Underlay
Composition Depth
0
Style
Opacity
1
Edge Rendering
Edge Color
Color Map
None
Add Component
Console

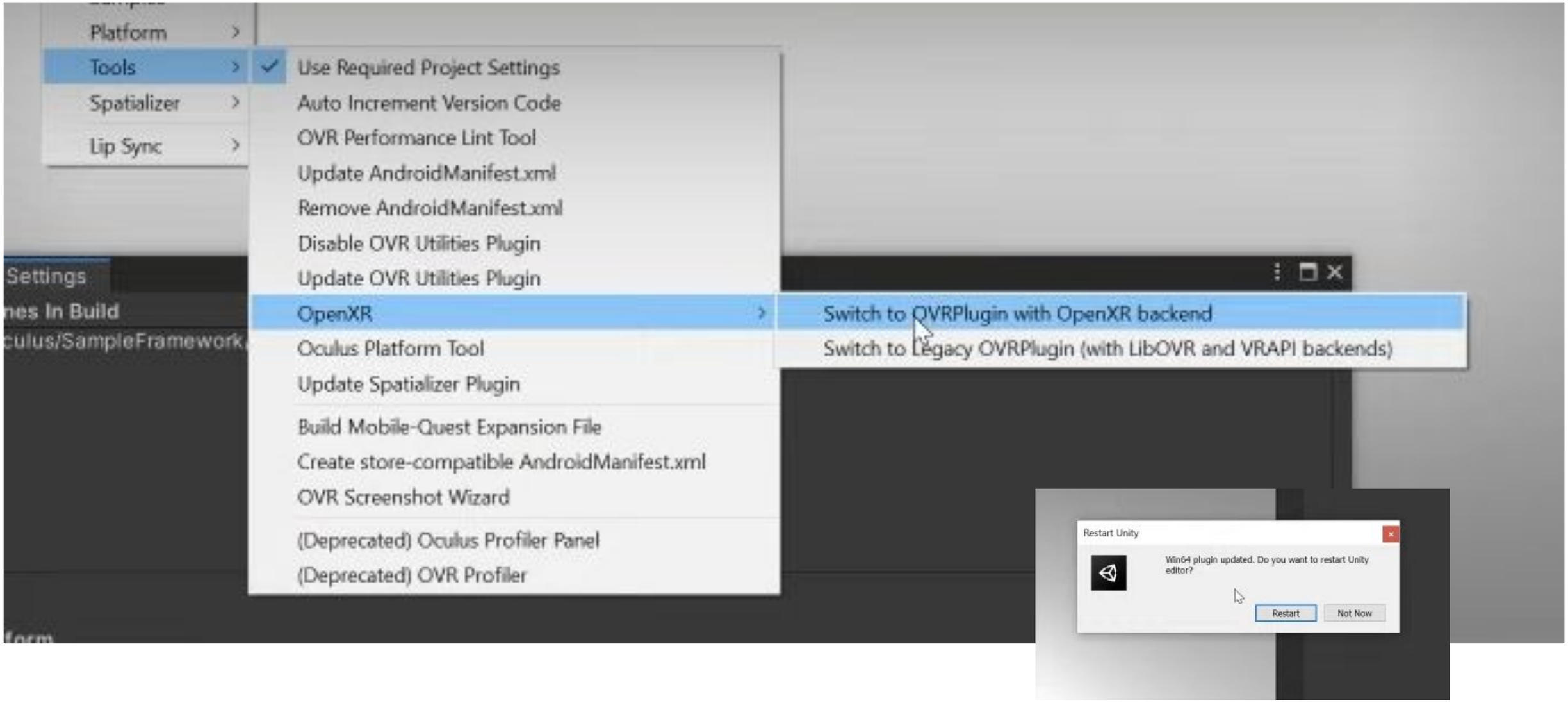

Directional Light
OVRCameraRig
Experience
RightMaskHandMesh
LeftMaskHandMesh
HandMeshUI


Project
SampleFramework
Core
Editor
Usage
AppDeeplink
CustomHands
DebugUI
DistanceGrab
Firebase
Locomotion
OVROverlay
Passthrough
Materials
Prefabs
Scenes
Scripts
SharedAssets
SpatialAnchor
StartScene
Spatializer
Assets > Oculus > SampleFramework > Usage > Passthrough > Scenes
AugmentedObjects
Lighting
OverlayPassthrough
PassthroughHands
PassthroughStyles
SelectivePassthrough
SurfaceProjectedPassthrough


Use Position Tracking
Use IPD In Position Tracking
Reset Tracker On Load
Allow Recenter
Late Controller Update
Display
Color Gamut
Rift_CV1
Quest Features
General
Build Settings
Security
Experimental
Focus Aware (Required)
Hand Tracking Support
Controllers And Hands
Hand Tracking Frequency
LOW
Requires System Keyboard
System Splash Screen
None (Texture 2D)
Select
Allow Optional 3DoF Head Track
Passthrough Capability Enabled
Mixed Reality Capture for Quest
ActivationMode
Automatic
Mixed Reality Capture
Insight Passthrough
Enable Passthrough
OVR Headset Emulator (Script)
Script
OVRHeadsetEmulator
Op Mode
Editor Only
Reset Hmd Pose On Release
Reset Hmd Pose By Middle Mous
Activate Keys
Pitch Keys
Enable Unpremultiplied Alpha (Script)
Script
EnableUnpremultipliedAlpha


Console
Clear
Collapse
Clear on Play
Clear on Build
Error Pause
Editor

CLICK
NOW
SUBSCRIBE



HACK IN THE WOODS

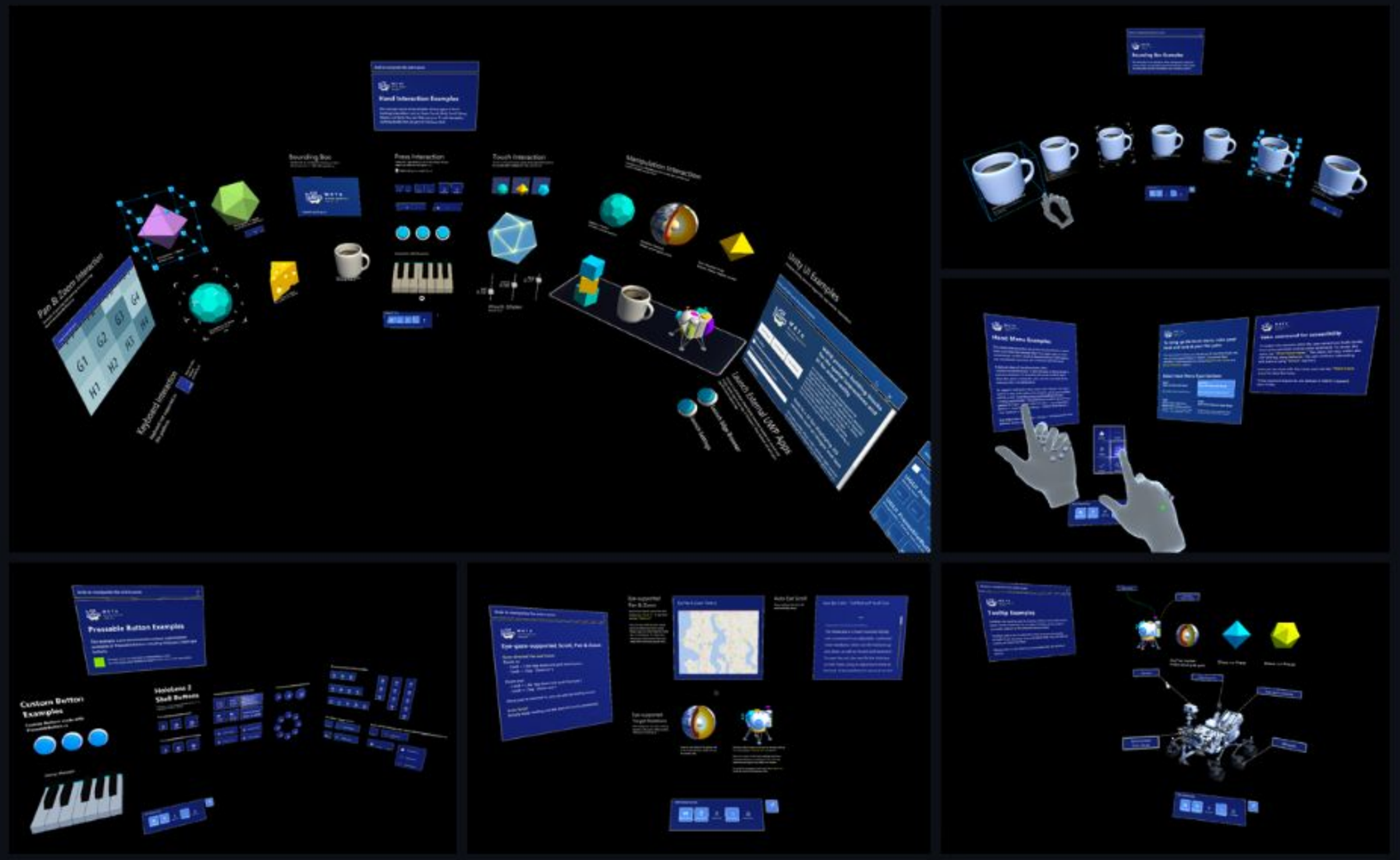
```
Unity Script (2 asset references) | 0 references | Dilmer Valecillos, 4 hours ago | 1 author, 1 change
public class HandGestures : MonoBehaviour
{
    [SerializeField]
    private OVRHand ovrHand;

    [SerializeField]
    private OVRHand.Hand HandType = OVRHand.Hand.None;

    Unity Message | 0 references | Dilmer Valecillos, 4 hours ago | 1 author, 1 change
    private void OnEnable()
    {
        if(ovrHand == null)
        {
            Logger.Instance.LogError("ovrHand must be set in the inspector...");
        }
        else
        {
            Logger.Instance.LogInfo("ovrHand was set correctly in the inspector...");
        }
    }
    Unity Message | 0 references | Dilmer Valecillos, 4 hours ago | 1 author, 1 change
    void Update()
    {
        if(ovrHand.GetFingerIsPinching(OVRHand.HandFinger.Index))
        {
            Logger.Instance.LogInfo($"Hand ({HandType}) Pinch Strength ({ovrHand.GetFingerPinchStrength(OVRHand.HandFinger.Index)})");
        }
    }
}
```

GitHub

https://github.com/dilmerv/OculusPassthroughWithHands

Toolkit(s)

Explore MRTK's various types of interactions and UI controls through the example scenes. You can find example scenes under **Assets/MRTK/Examples/Demos** folder.



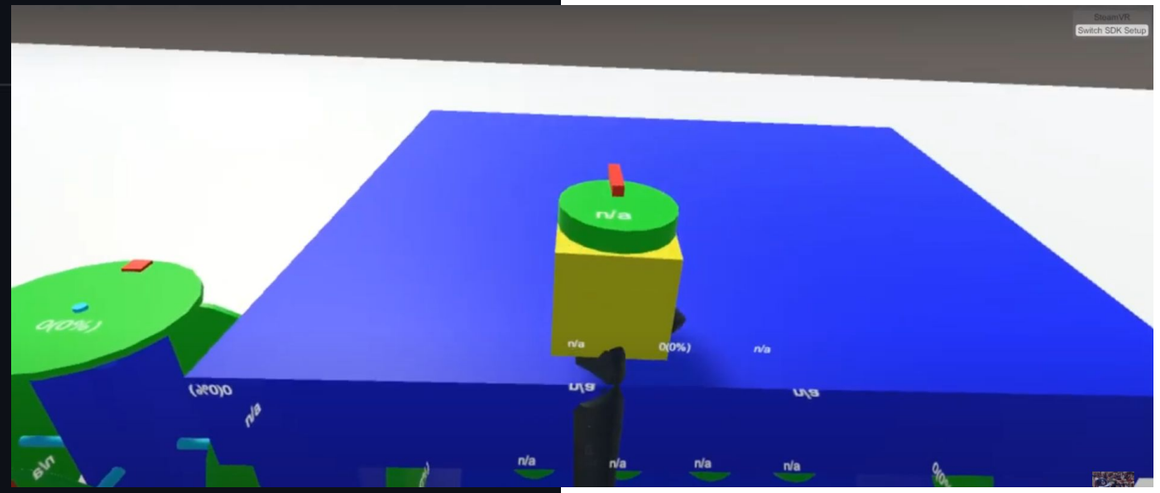
MRTK examples hub

HACK IN THE WOODS

README.md

VRTK

virtual reality toolkit



VRTK Farm Yard Example - Virtual Reality Toolkit

A Farm Yard example scene of how to use VRTK v4 for rapidly building spatial computing solutions in the Unity software.

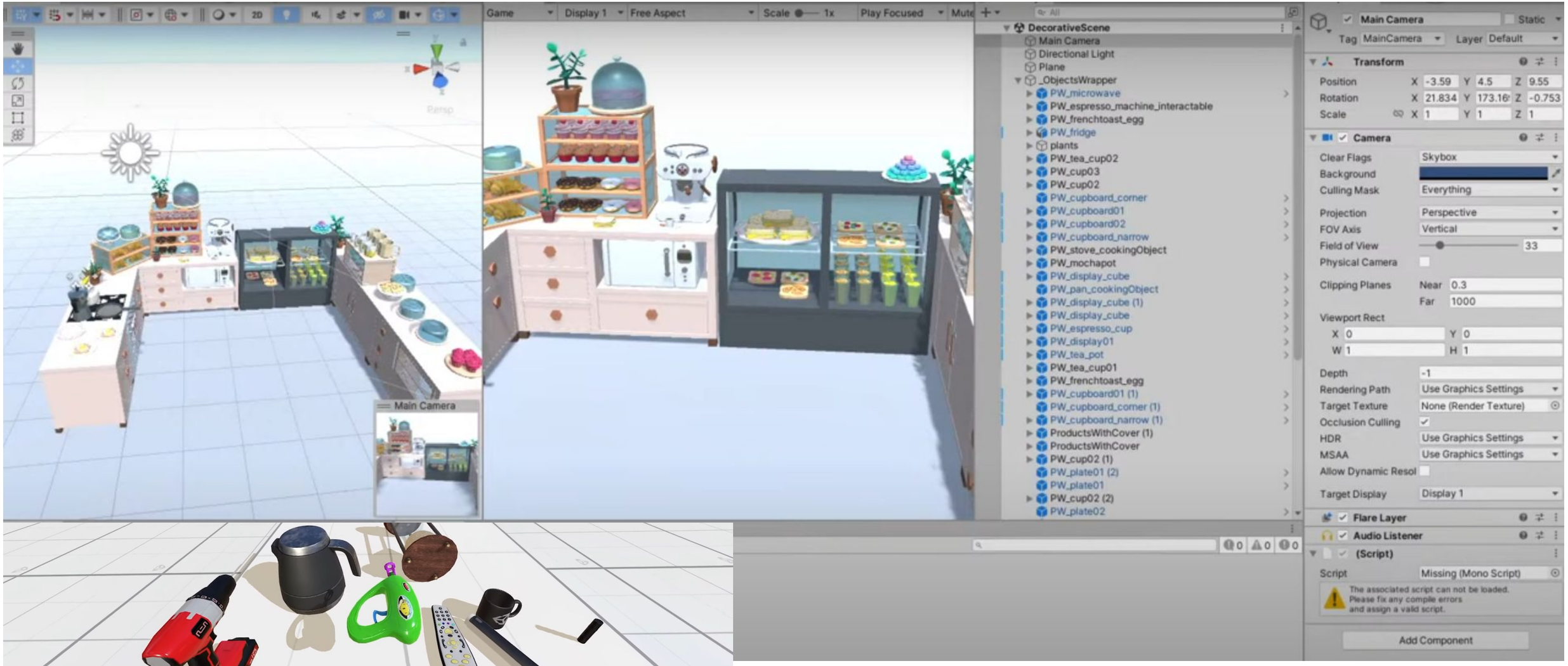
Requires the Unity software version 2020.3.24f1.

Beta Disclaimer

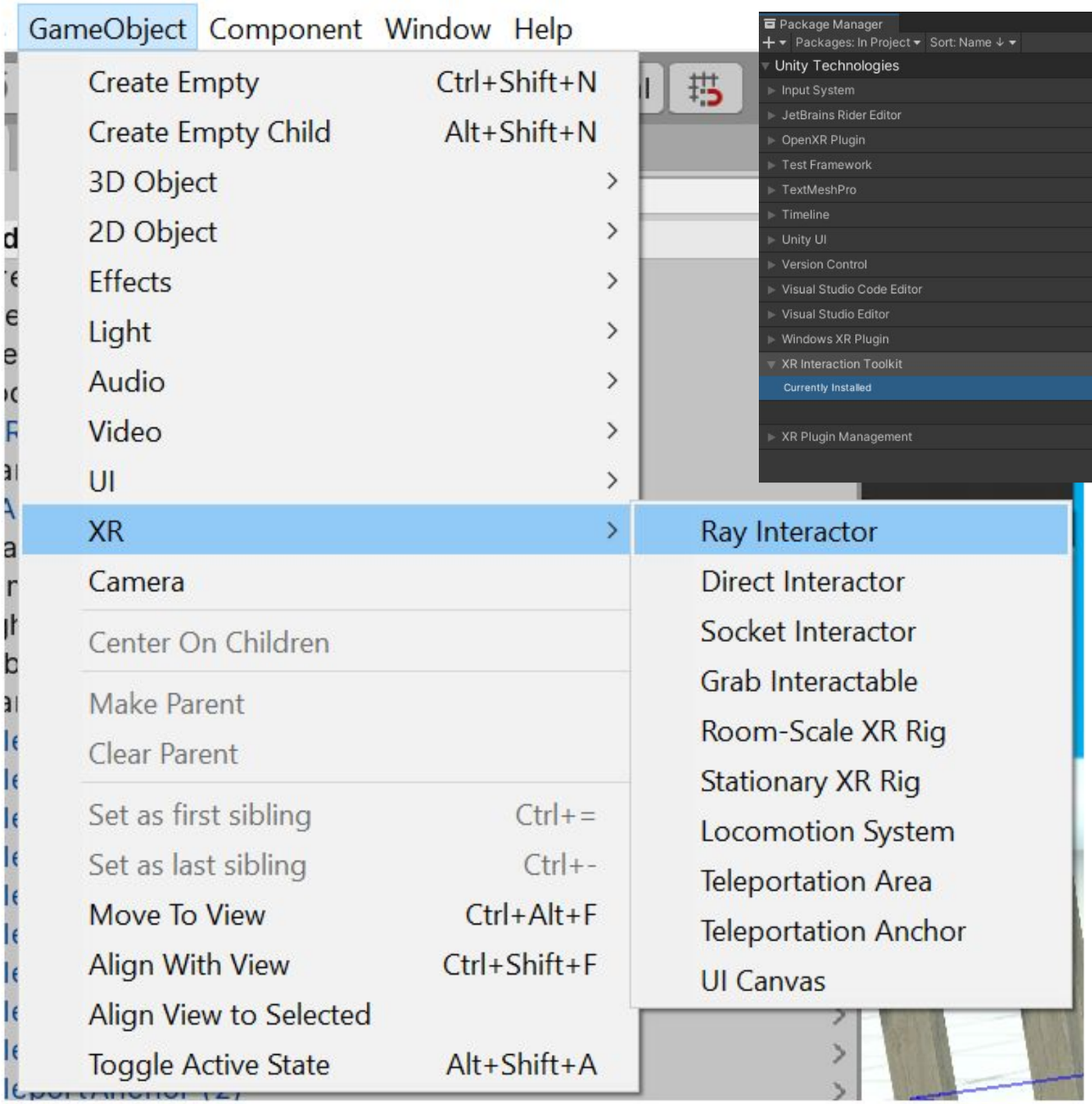
This project was built using 2020.3.24f1 and should work as expected on that version. It is feasible to downgrade this project to a previous version of the Unity software but it may cause issues in doing so.

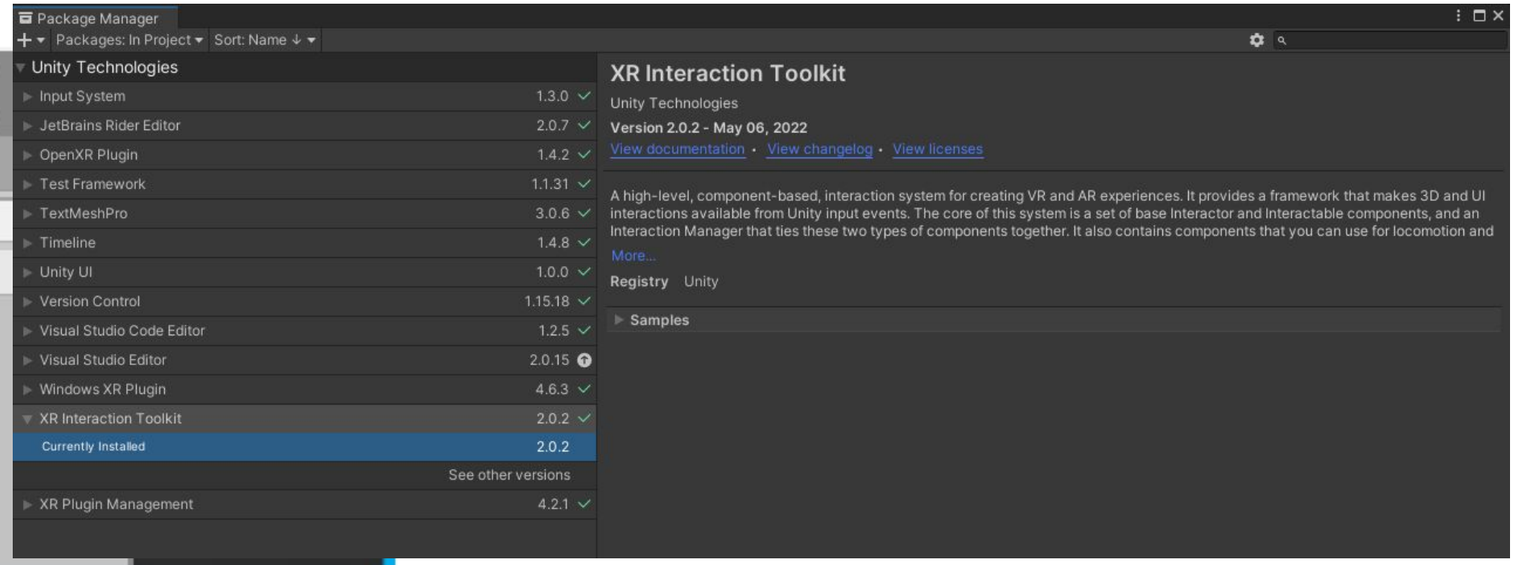
This project uses the newer Unity software XR management system and the newer Unity Input system.

This VRTK v4 Farm Yard example project has been updated to use the latest Tilia packages but is still in development and is missing a number of features from the previous release that used the deprecated VRTK.Prefabs package.

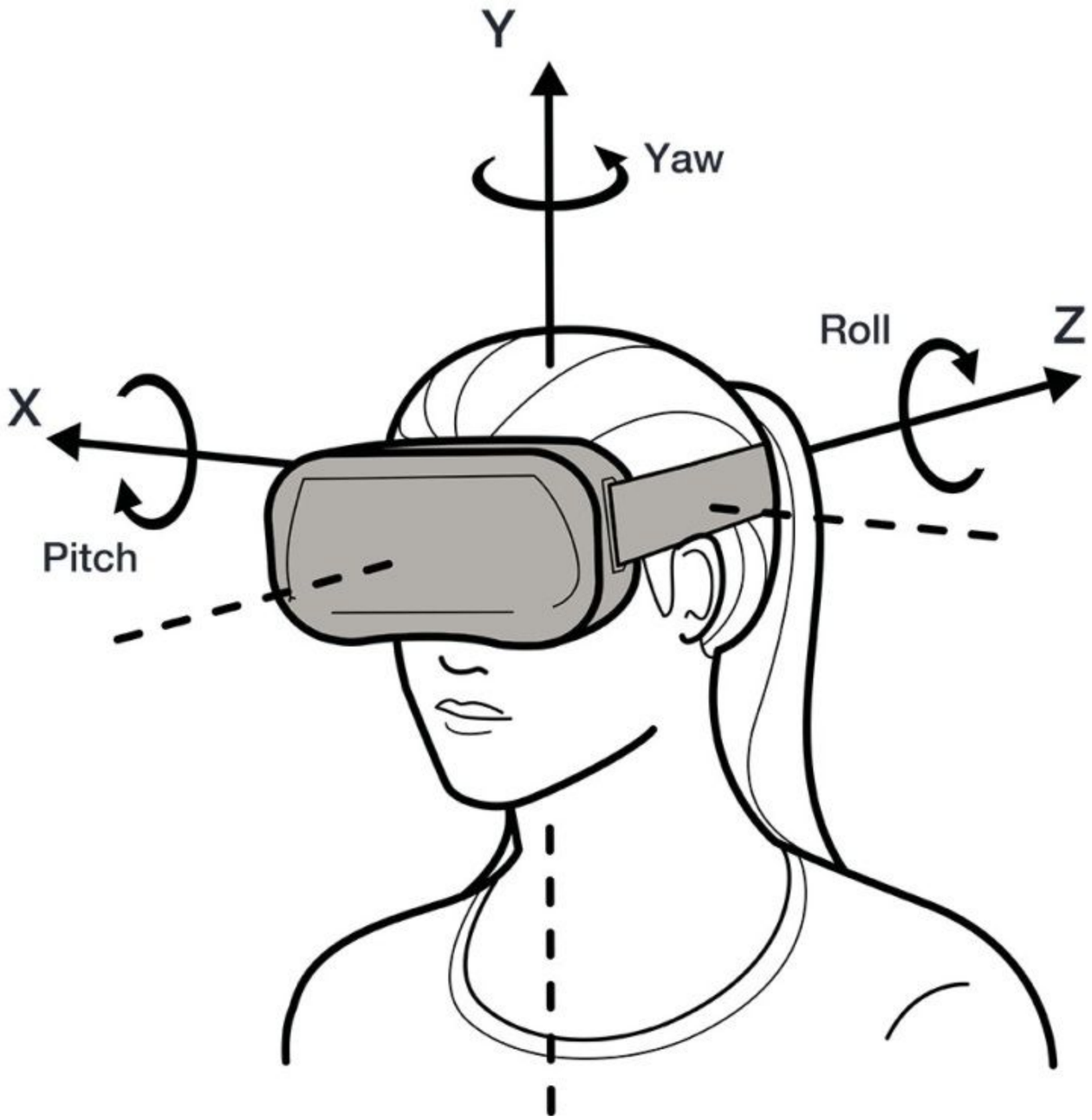


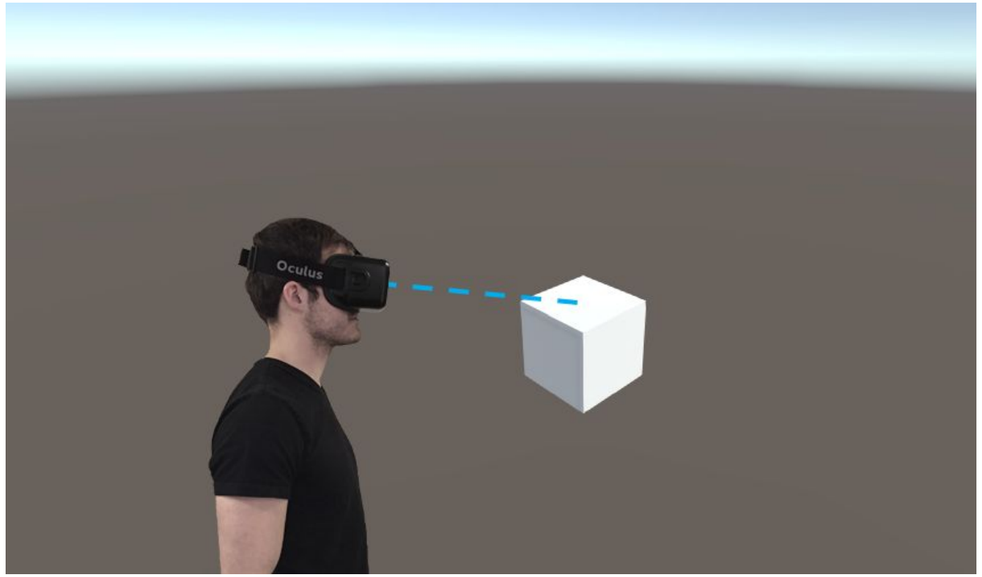
HACK IN THE WOODS





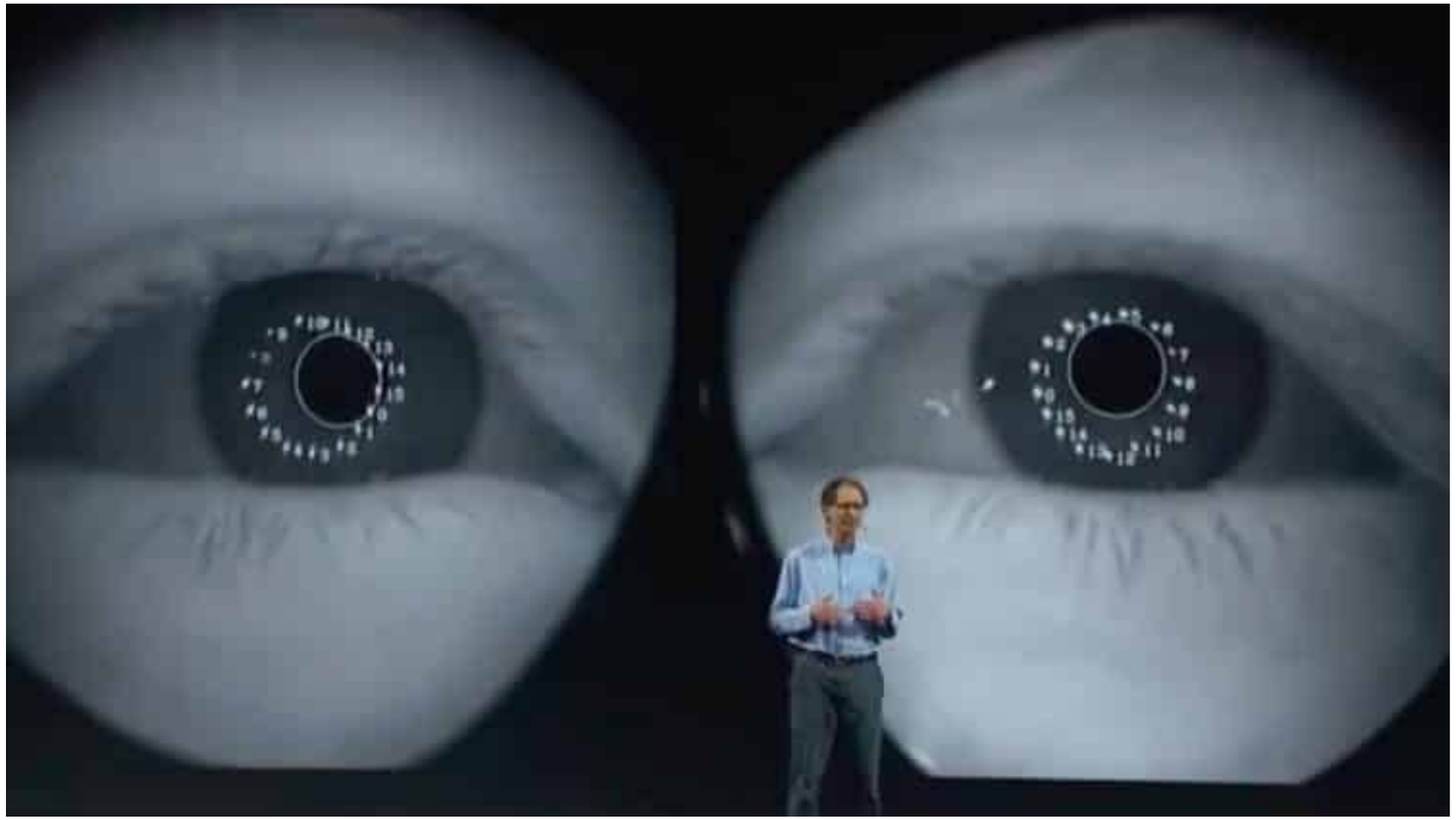
What could be abstract through time ?

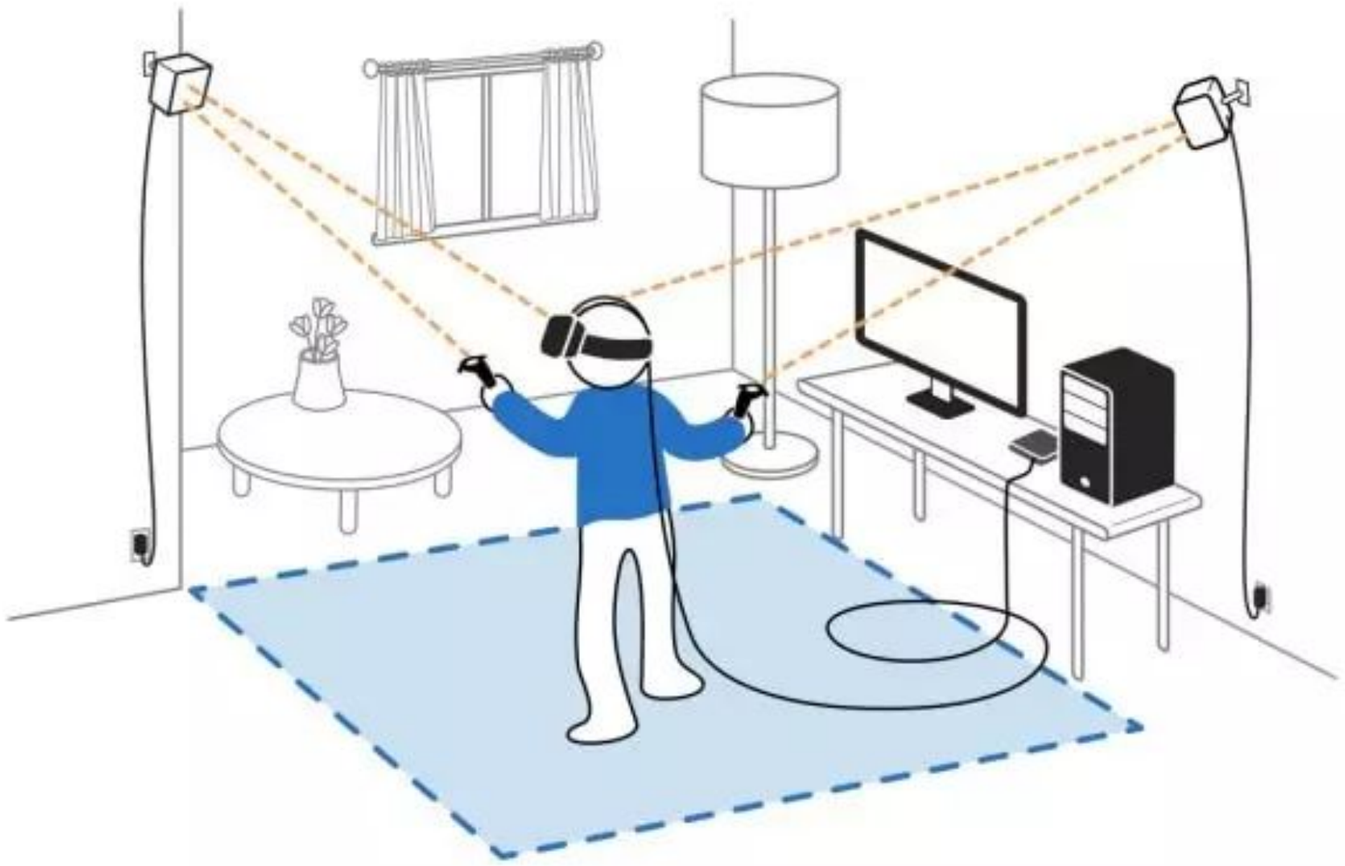


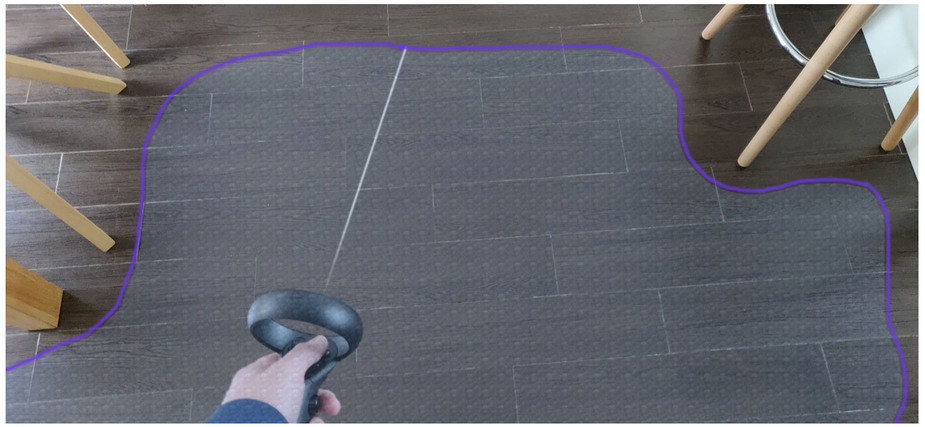


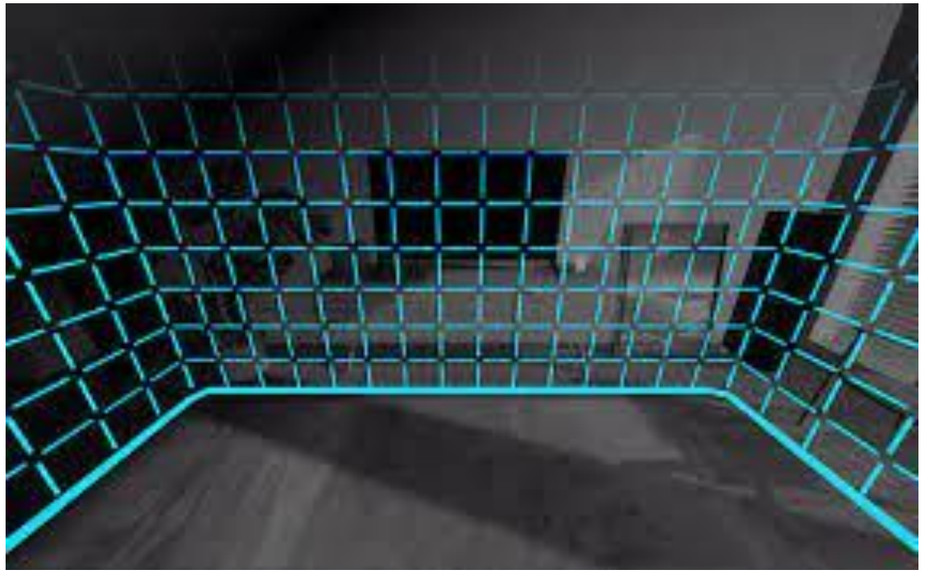
HACK IN THE WOODS

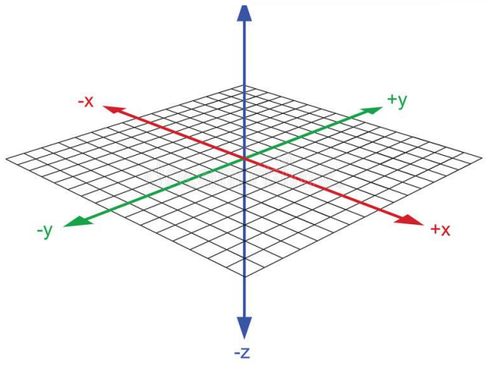

```
private void ApplyRotationToPutStandardToTheDirectionOfHandBones(Transform selectedBone, int i)
{
    Quaternion rotationAdjustement = Quaternion.identity;
    if (m_handType == SkeletonType.HandLeft)
        rotationAdjustement = Quaternion.Euler(m_leftAdjustementRotation)*Quaternion.Euler(0,180,0);
    if (m_handType == SkeletonType.HandRight)
        rotationAdjustement = Quaternion.Euler(m_rightAdjustmentRotation) * Quaternion.Euler(0, 180, 0);

    selectedBone.rotation = m_handSkeletonInfo.Bones[i].Transform.rotation * rotationAdjustement;
}
```



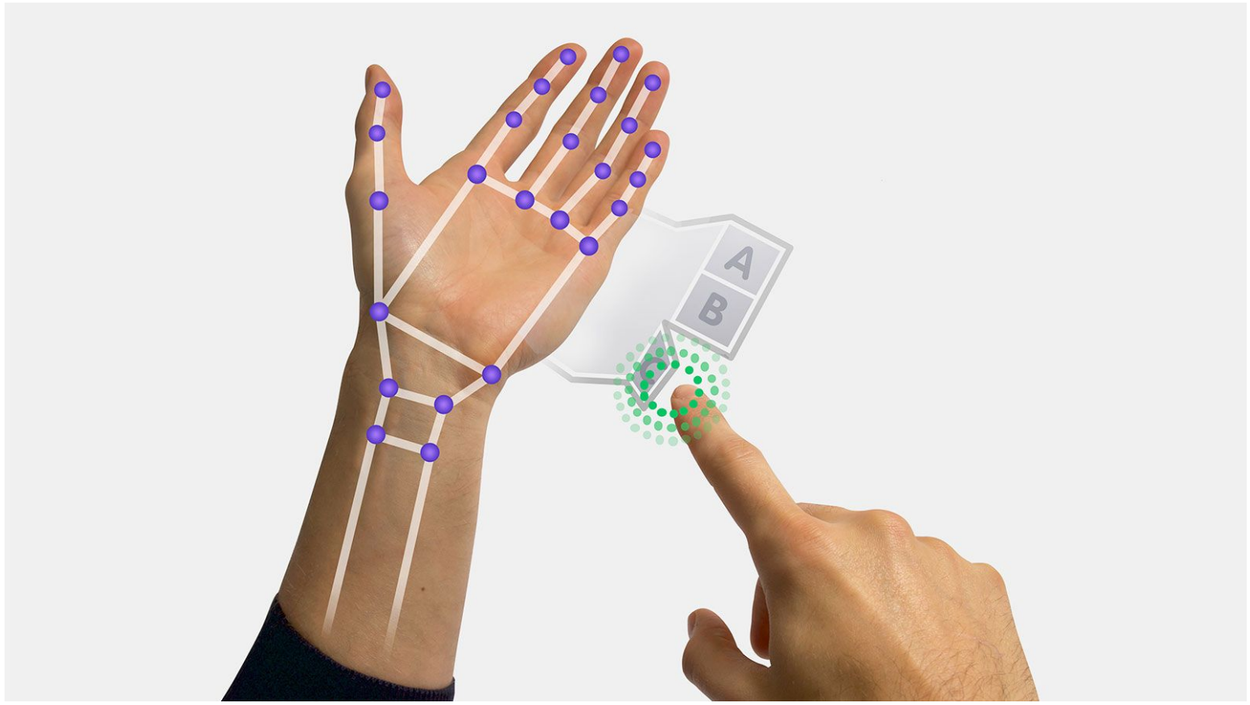


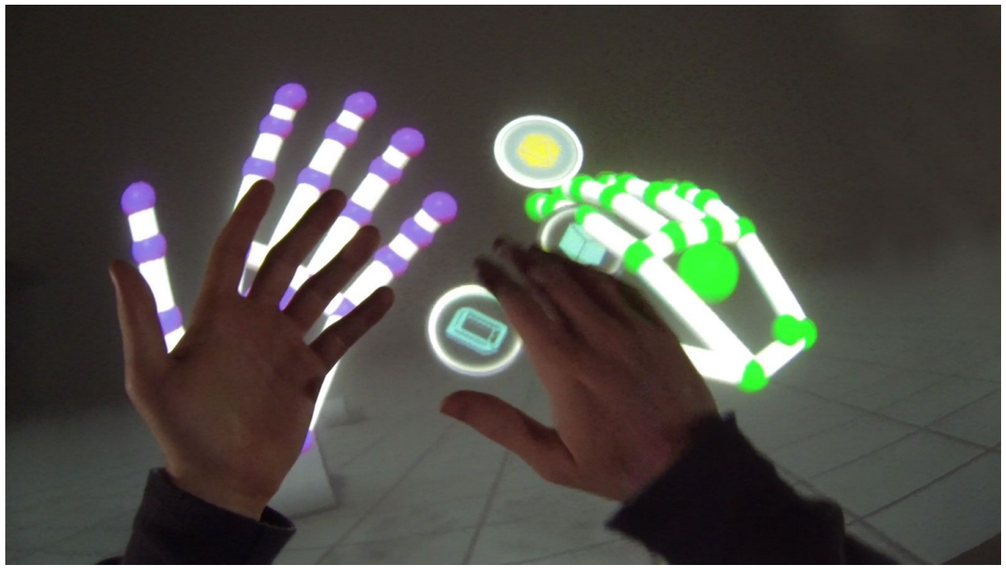


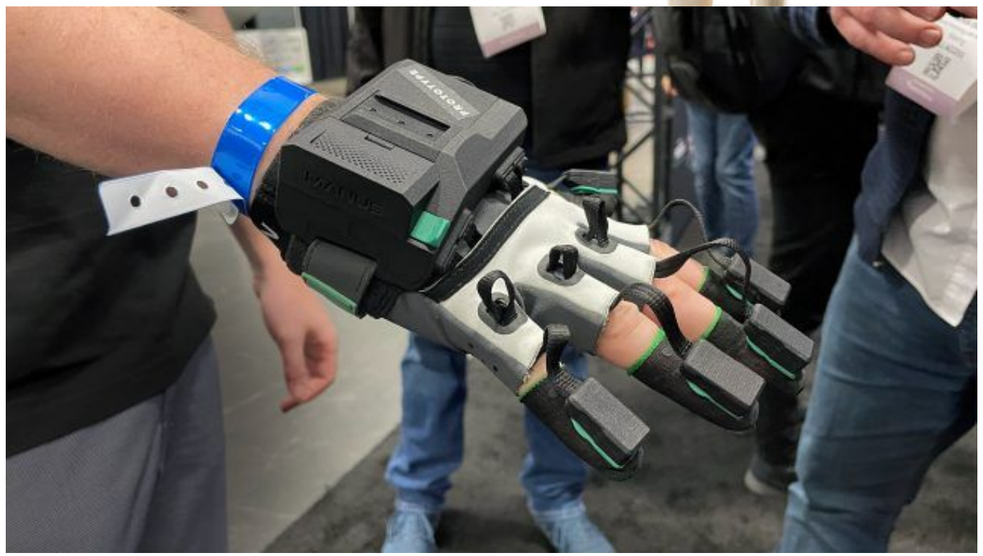




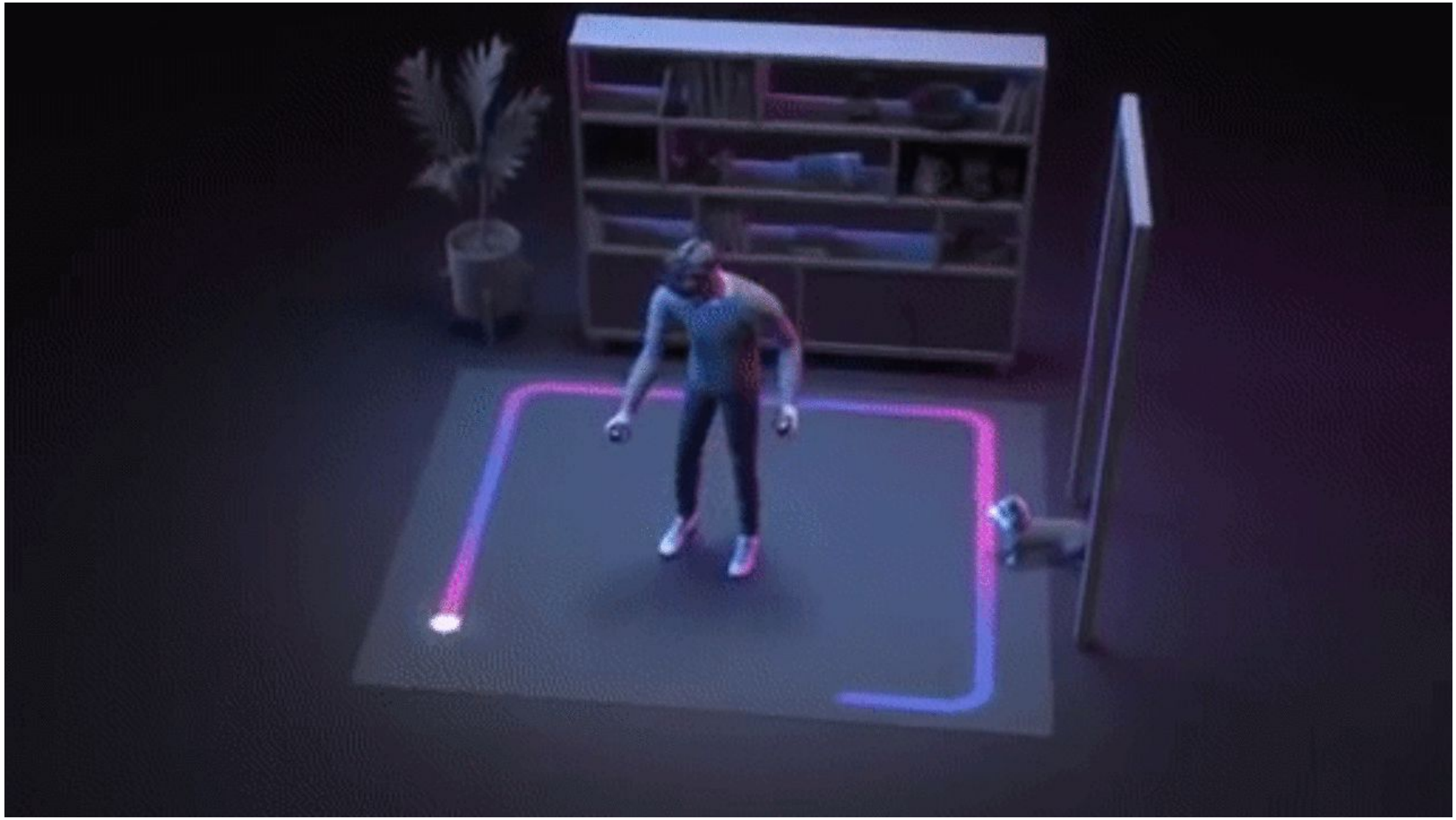
HACK IN THE WOODS



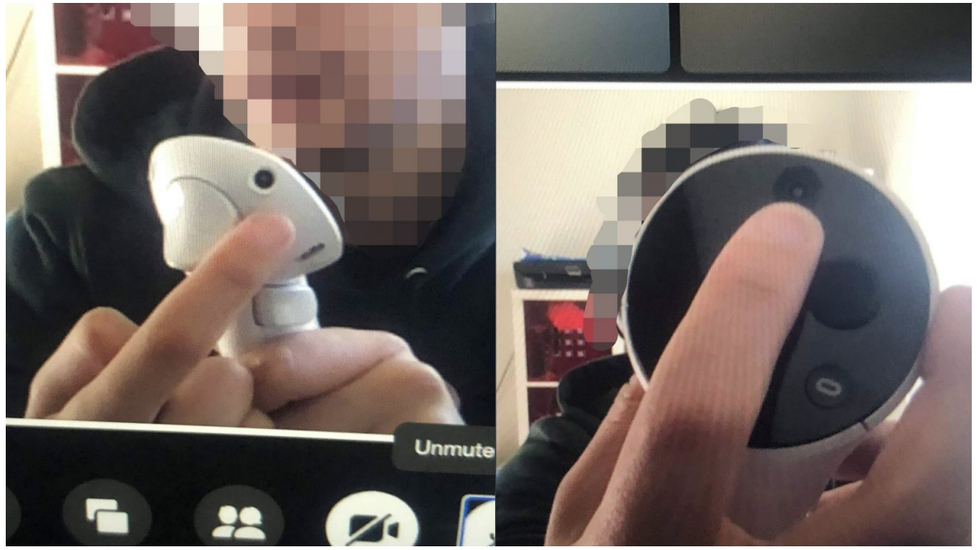


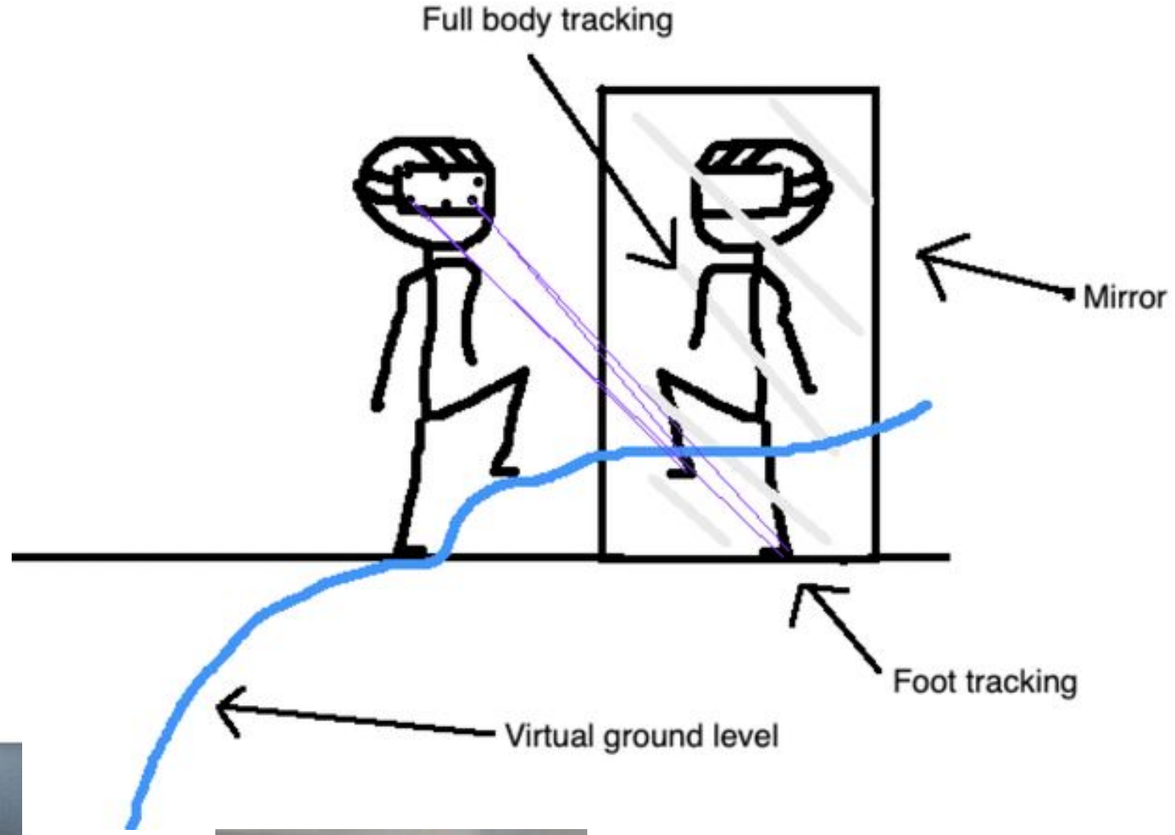


HACK IN THE WOODS



HACK IN THE WOODS

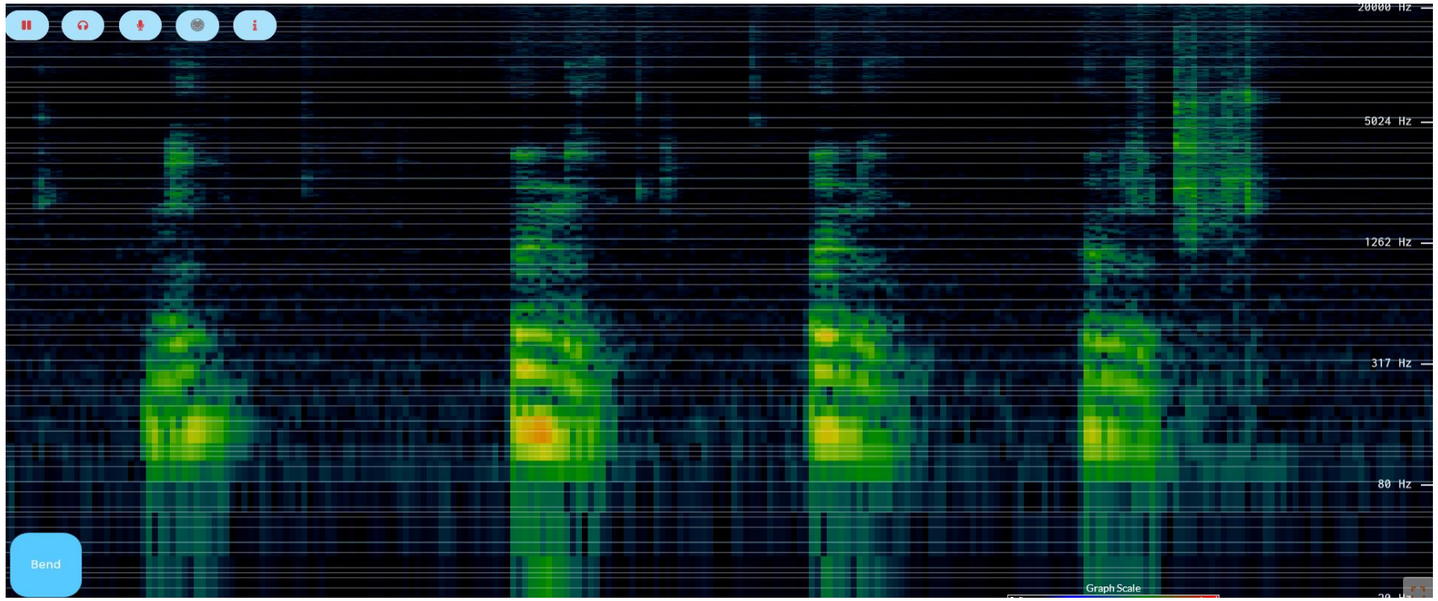




HACK IN THE WOODS

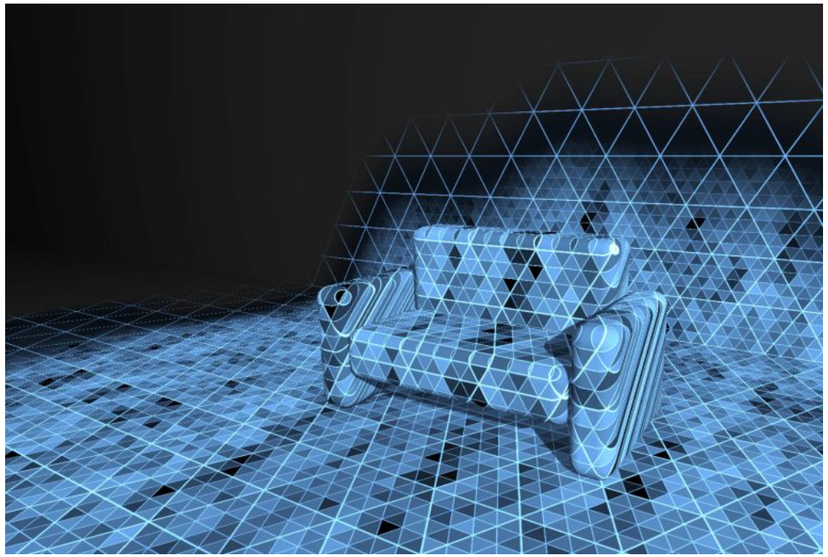
The BEST VR Full Body Tracking is also Affordable!! [Tundra Tracker]

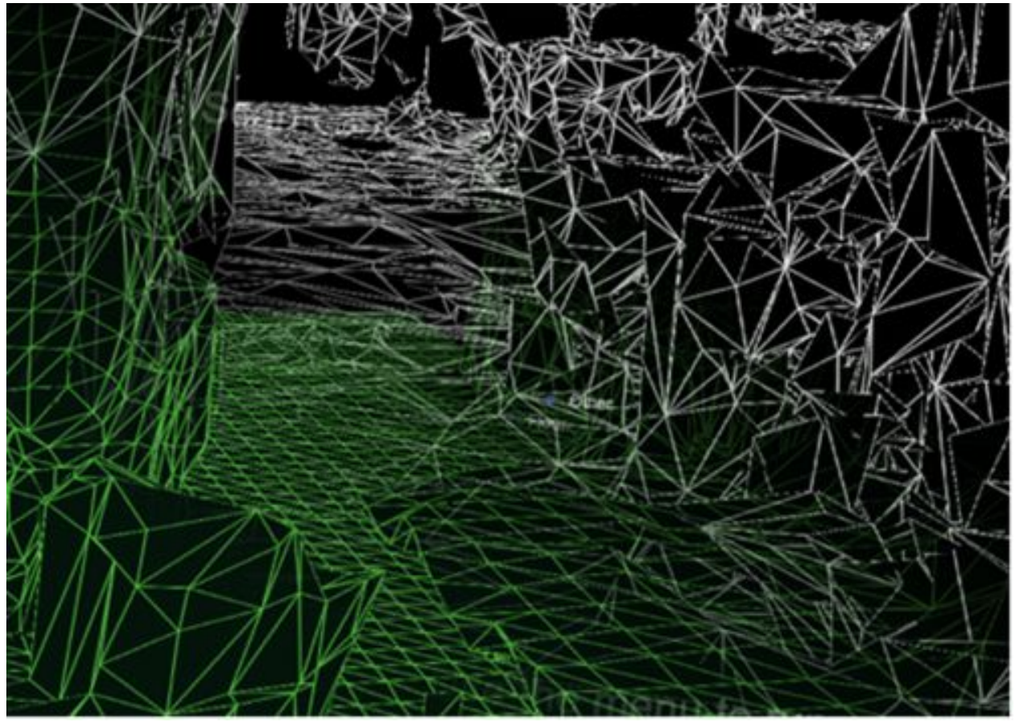
HACK IN THE WOODS

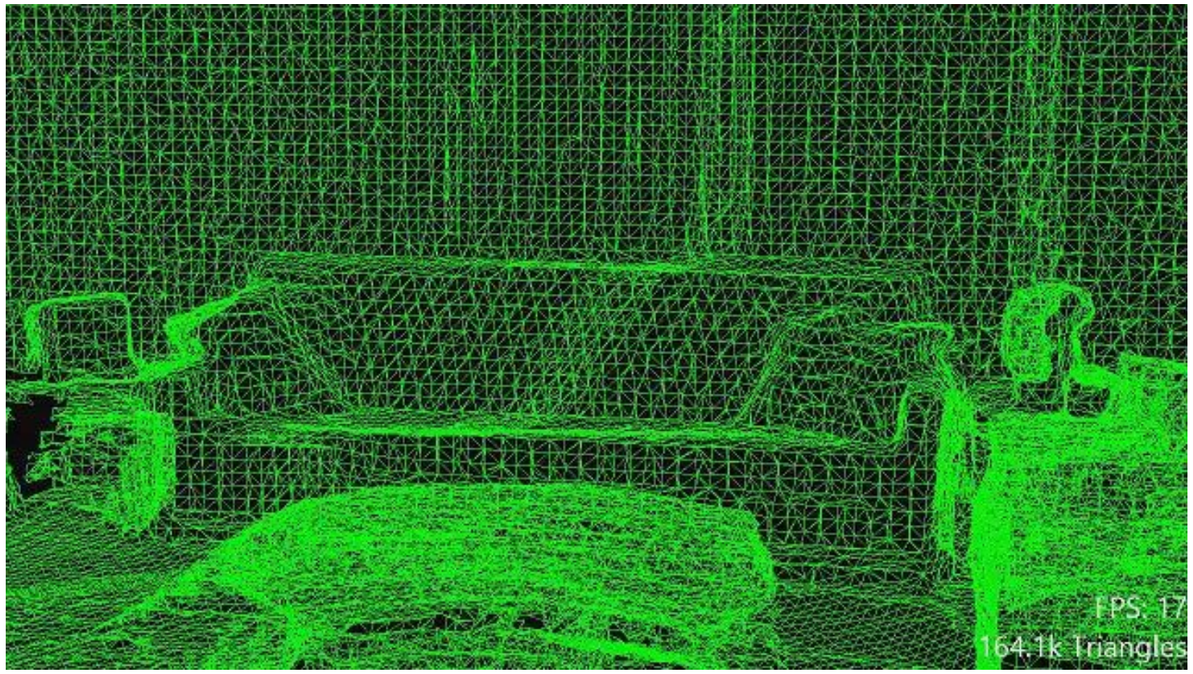


HACK IN THE WOODS

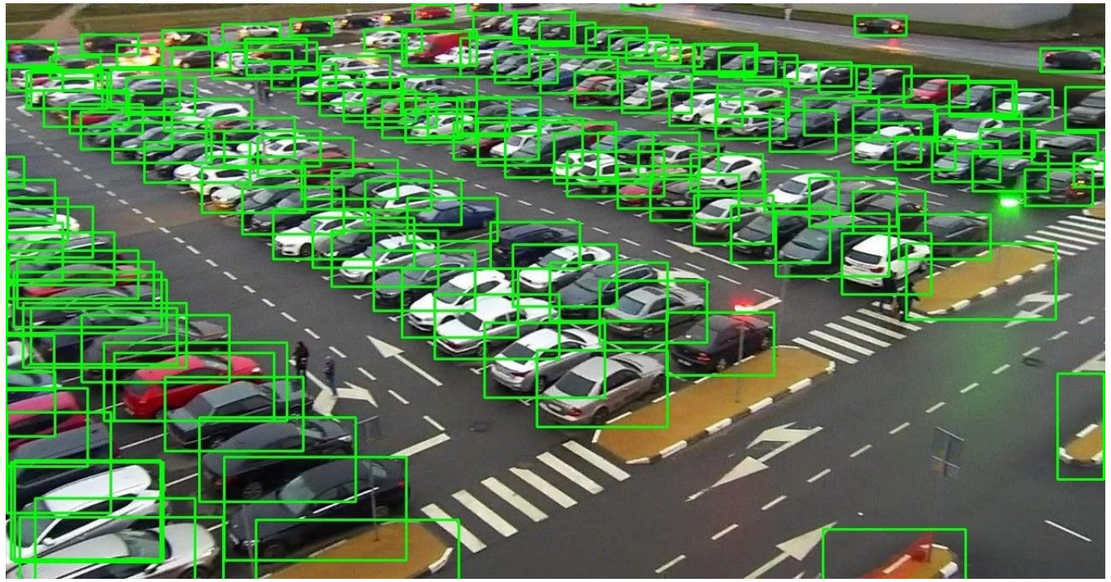
HACK IN THE WOODS

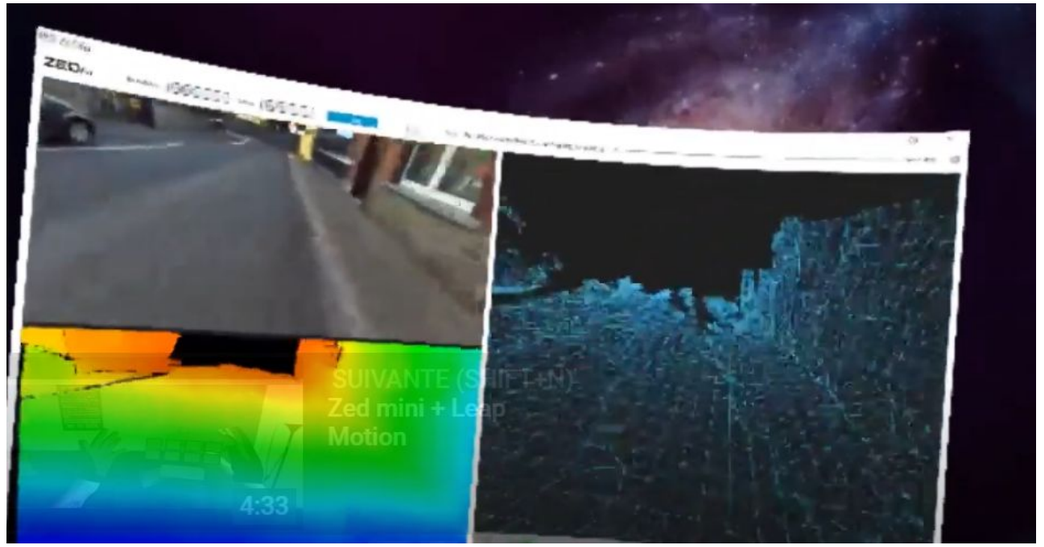






HACK IN THE WOODS





HACK IN THE WOODS


GitHub
SCAN ME
Find all the link, code and, source here:
https://github.com/EloiStree/2022_06_18_HackInTheWood_Topic_LynxAndOpenXR
HACK IN THE WOODS


SDR
(Simulated)
HDR
(HLG/PQ 2020)
2:38 / 4:08
ELECTRONIC VARIFOCAL MODULE
LIQUID CRYSTAL LENSES
1:58 / 4:08
Focal Depth
E
F P
T O Z
L P E D
P E C F D
E D F C Z P
F E L O P E D
D E F P O T E C
20/200
20/100
20/70
20/50
20/40
20/30
20/25
20/20
0:42 / 4:08
BUTTERSCOTCH

Zuckerberg Reveals Meta's Latest Prototype VR Headsets

HACK IN THE WOODS

```
// Copyright Microsoft Corporation. All rights reserved.
// Licensed under the MIT License. See LICENSE in the project root for license information.

using HoloToolkit.Unity;
using System;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.Events;
using UnityEngine.XR.WSA.Input;



//Source modified: https://github.com/Microsoft/GalaxyExplorer/blob/master/Assets/Scripts/Input/HandInput.cs
public class HandsInput : SingleInstance<HandsInput>
    {



    [System.Serializable]
        public struct HandStateRaw
    {
        [Tooltip("Does the hand is tracked ?")]
        public bool valid;

        [Tooltip("Each Time different")]
        public uint id;


        [Tooltip("Is tapping ?")]
        public bool isTappingDown;

        [Tooltip("Time Since Level Load")]
        public float tapPressedStartTime;
          [Tooltip("From Hololens camera start point")]
            public Vector3 latestTapStartPosition;
        [Tooltip("From Hololens camera start point")]
        public Vector3 latestObservedPosition;
        [Tooltip("From Hololens camera start point")]
        public Vector3 firstObservedPosition;

        public UnityEvent m_onEnter;
            public UnityEvent m_onDown;
            public UnityEvent m_onUp;
            public UnityEvent m_onExit;


    }
```

https://github.com/EloiStree/2022_06_18_HackInTheWood_Topic_LynxAndOpenXR/blob/main/Note/2022_06_18.md