Docs @ https://cloudwalker2020.github.io/HurricaneVR-Docs/manual/intro.html

# Project and Packages Setup

- Import the HurricaneVR Asset from the Unity Package Manager before proceeding.
- Make sure TextMesh Pro is installed and updated to the latest version for the example scenes.
- Setup your Unity VR environment based on the below, then proceed to setup your Project Settings.

# Oculus Store Builds

- Oculus dev article on deprecation of the Unity 2019 Legacy Oculus SDK
- Install XR Plugin Management with either the Oculus XR Plugin or the OpenXR Plugin.
- Legacy SDK is only allowed by Oculus with a waiver according to their article.

# PCVR Builds

Choose between using the SteamVR Plugin or OpenXR Plugin depending on what Unity version you want to use.

Keep in mind Valve Knuckles finger tracking is not supported yet by Unity OpenXR and requires SteamVR plugin to work correctly.

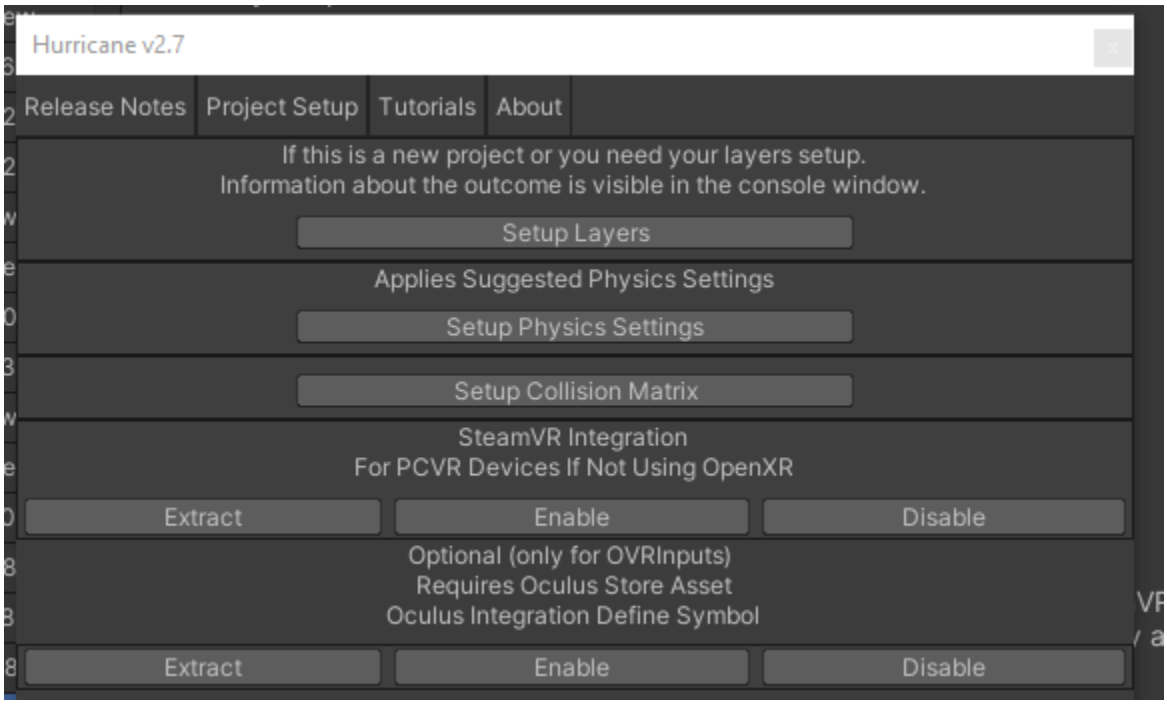
- Unity 2019: Legacy SDK + SteamVR
- Unity 2019 and above: XR Plugin Management + SteamVR
- Unity 2020.3 and above XR Plugin Management + OpenXR 1.3+

# Project Setup

HurricaneVR requires a few project settings to get up and running.

Open the Setup window by navigating to Tools → HurricaneVR → Setup and then click the “Project Setup" button on the toolbar.

- Setup Layers will add layers if necessary and will report the status of the operation in your console.
- Setup Physics settings will set the recommended physics settings for joint and collision stability.
- Setup Collision Matrix will setup the collision layer matrix for you.




# Tags and Layers

## Layer Notes

By default grabbable objects require line of sight from the hand to be picked up.

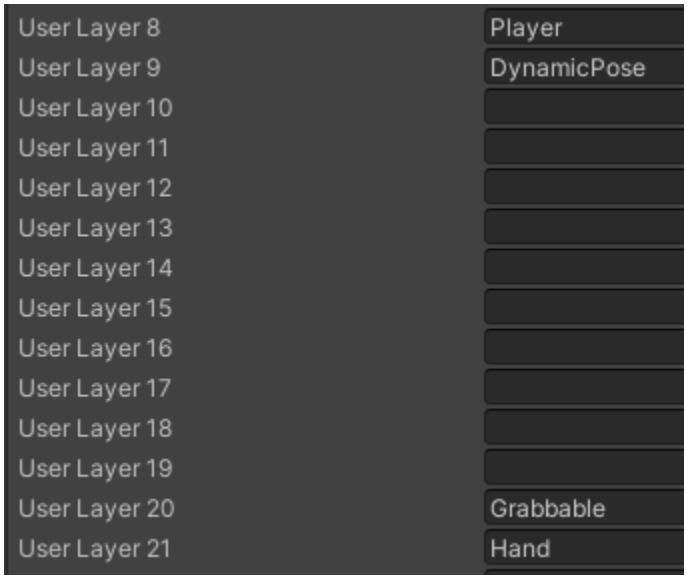
The RaycastLayermask field defines what layers will block the line of sight ray cast. This field will need to be updated whenever you add additional layers for your environment or grabbable objects other than "Grabbable" framework layer.

**NOTE**

Grabbable objects can have their line of sight requirement disabled on their HVRGrabbable component.

## Framework Layers

- Player - used to prevent collision with the player character controller.
    - Assign to the PlayerController object (be careful not to assign to the children)
- Grabbable - used to help prevent collision with the player character controller and is automatically assigned recursively to objects with HVRGrabbable components.
    - Automatic layer assignment can be disabled per grabbable with by setting AutoApplyLayer to false.
    - Automatic layer assignment can be toggled at the project level on the HVRSettings scriptable object.
- Hand - used on the hand collision geometry
    - Automatically applied to the HVRHandGrabber component and children unless disabled on the same.
- DynamicPose - Used for the dynamic posing grab sequence to ensure the fingers only collide with the desired object.
    - Automatically set in code during the auto pose sequence, no need to assign these to any objects.




# Fixed Time Step

For smooth game play in VR, the Fixed Time Step must match the refresh rate of the headset.

The HVRTimeManager component can be added to your scene, and will automatically handle the fixed time step for you.

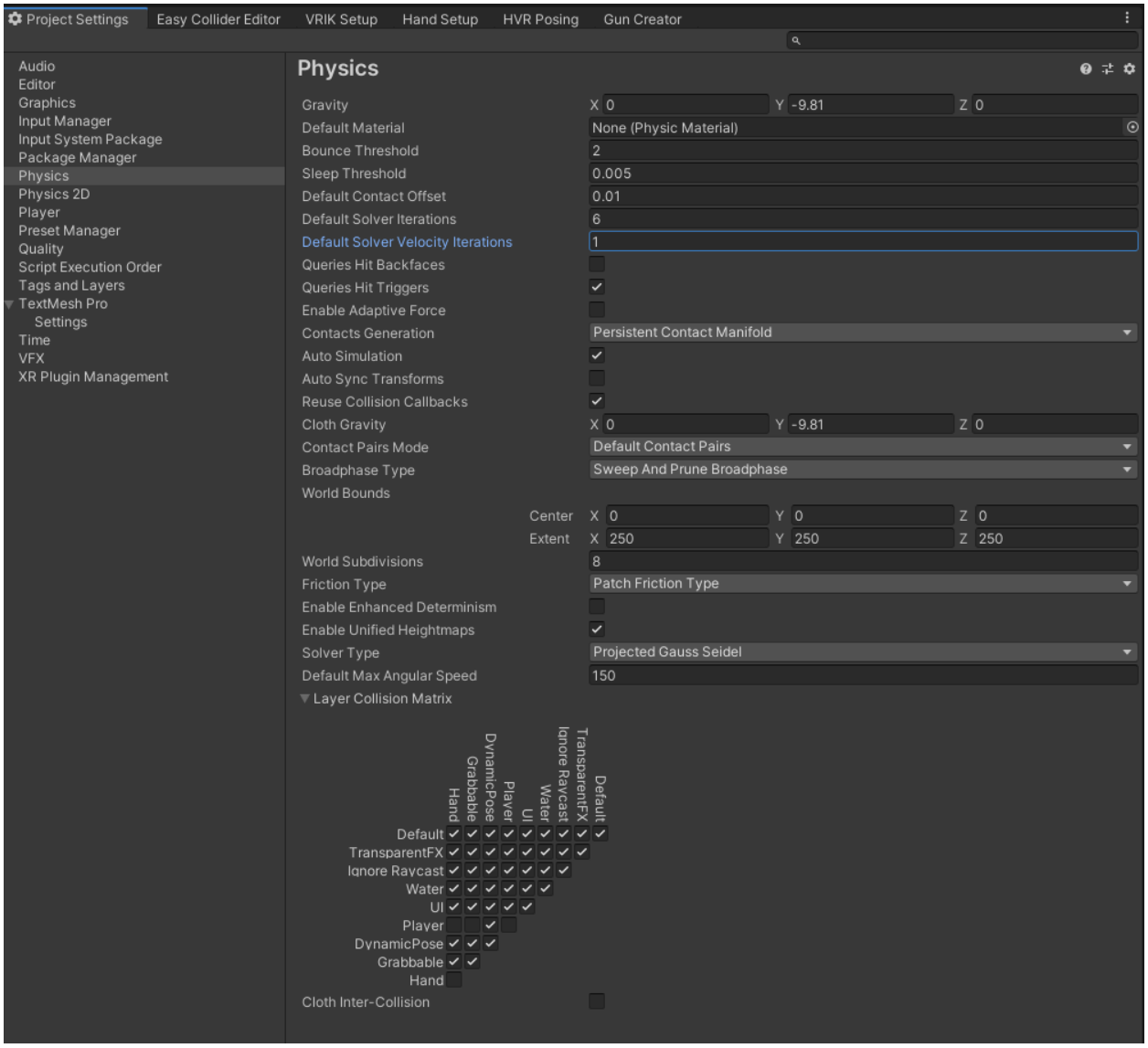
The HVRGlobal prefab which comes with other required components already has this and is in use in the included demo scenes.

# Physics Settings

Edit -> Project Settings -> Physics

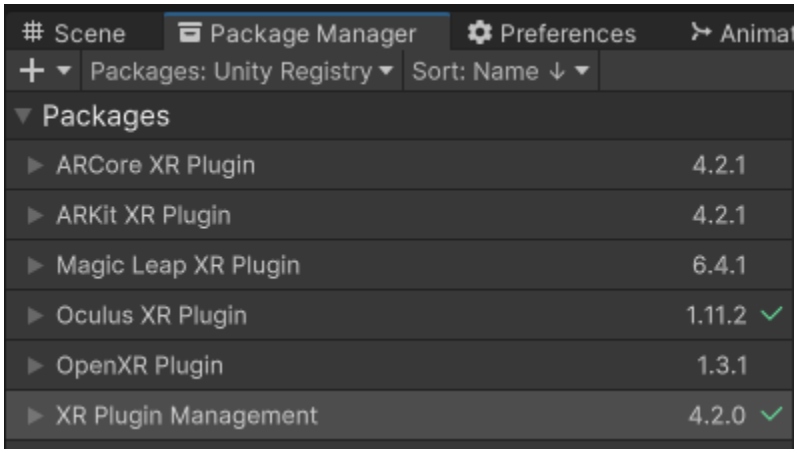
Notable Properties:

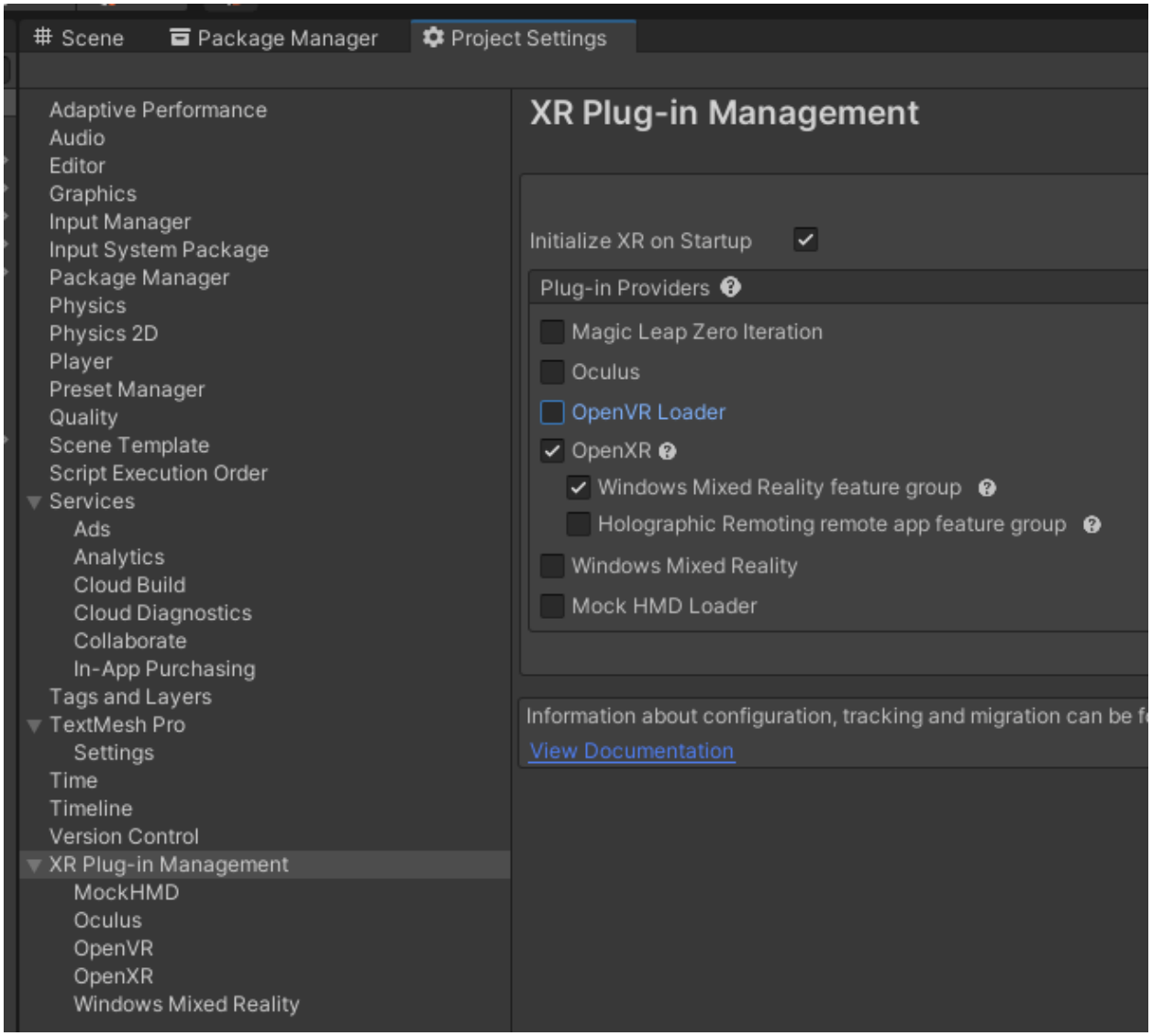
- Default Solver Iterations & Default Solver Velocity Iterations
    - Increasing these can stabilize contact and joint constraints at the cost of CPU budget.
- Default Max Angular Speed
- Layer Collision Matrix
- Solver Type = PGS


Project Settings
Easy Collider Editor
VRIK Setup
Hand Setup
HVR Posing
Gun Creator
Audio
Editor
Graphics
Input Manager
Input System Package
Package Manager
Physics
Physics 2D
Player
Preset Manager
Quality
Script Execution Order
Tags and Layers
TextMesh Pro
Settings
Time
VFX
XR Plugin Management
Physics
Gravity
X 0
Y -9.81
Z 0
Default Material
None (Physic Material)
Bounce Threshold
2
Sleep Threshold
0.005
Default Contact Offset
0.01
Default Solver Iterations
6
Default Solver Velocity Iterations
1
Queries Hit Backfaces
Queries Hit Triggers
Enable Adaptive Force
Contacts Generation
Persistent Contact Manifold
Auto Simulation
Auto Sync Transforms
Reuse Collision Callbacks
Cloth Gravity
X 0
Y -9.81
Z 0
Contact Pairs Mode
Default Contact Pairs
Broadphase Type
Sweep And Prune Broadphase
World Bounds
Center
X 0
Y 0
Z 0
Extent
X 250
Y 250
Z 250
World Subdivisions
8
Friction Type
Patch Friction Type
Enable Enhanced Determinism
Enable Unified Heightmaps
Solver Type
Projected Gauss Seidel
Default Max Angular Speed
150
Layer Collision Matrix
Hand
Grabbable
DynamicPose
Player
UI
Water
Ignore Raycast
TransparentFX
Default
Default
TransparentFX
Ignore Raycast
Water
UI
Player
DynamicPose
Grabbable
Hand
Cloth Inter-Collision

# XR Plugin Management

1. Install the XR Plugin Management package from the Package Manager or the Project Settings window.
2. Install the following packages from the Package Manager depending on your target platforms.
    1. Oculus: Oculus XR Plugin or OpenXR XR Plugin
    2. PCVR: OpenXR XR Plugin 1.3+ or SteamVR Plugin
3. Enable the Plug-in Providers under Edit -> ProjectSettings -> XR Plugin-Management
    1. Oculus and/or OpenVR Loader OR OpenXR






# OpenXR Plugin

Make sure to expand the OpenXR Plugin so that you can find version 1.3 and higher. These are required for haptics to work and your project will not compile without the correct version.

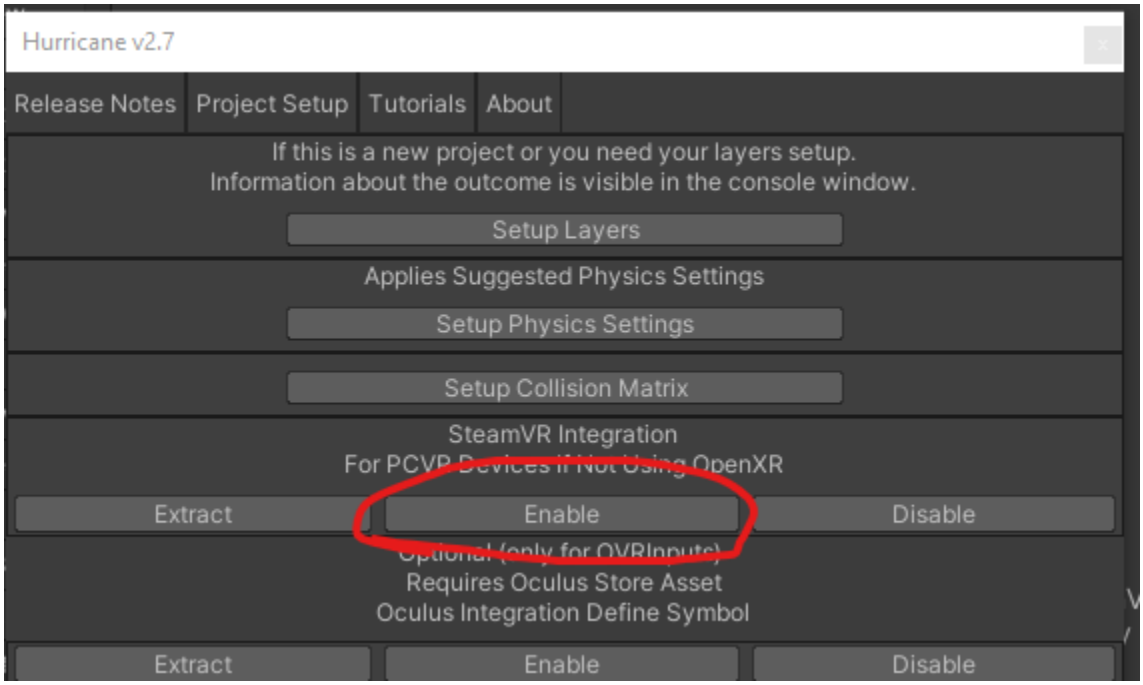

OpenXR Plugin
1.0.3
1.3.1
1.3.0

# SteamVR

Download and import the SteamVR Plugin from the Unity Store.

# Hurricane Integration

1. Extract the SteamVR Integration located at /HurricaneVR/Framework/Integrations.
2. Press “Import” when prompted to import the Partial Input binding for ‘HVR’. If a second option comes up, choose “Replace”, not “Merge”
3. The SteamVR Input window should present itself, if not open this window via your toolbar at : Window → SteamVR Input
4. At the bottom of the SteamVR Input window, locate and press the “Save and generate” button.
5. Add HVR_STEAMVR to your project setting scripting define symbols or by using Tools → HurricaneVR → Setup
    1. Wait a moment as the imported code becomes compiled.

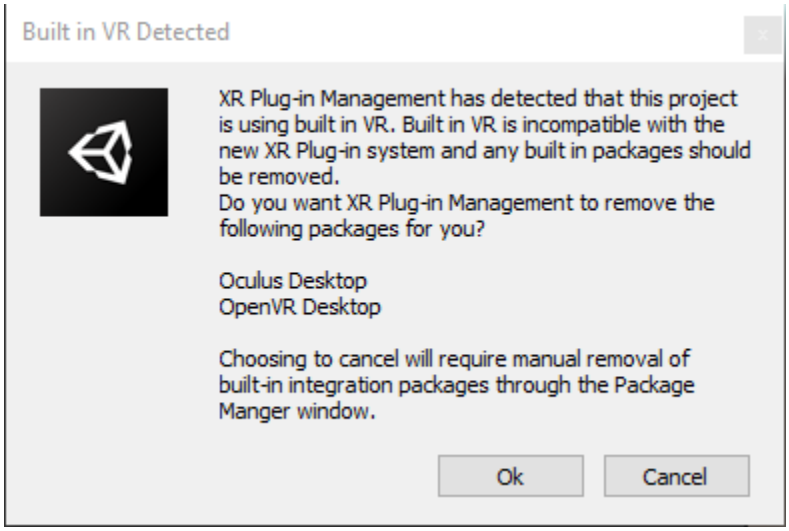



# Unity 2019

Because 2019 has access to Legacy and XR Plugin Management, you may receive this prompt after you import the plugin.

At this point you can decide whether to remain with Legacy OpenVR or update to XR Plugin (OpenVR)

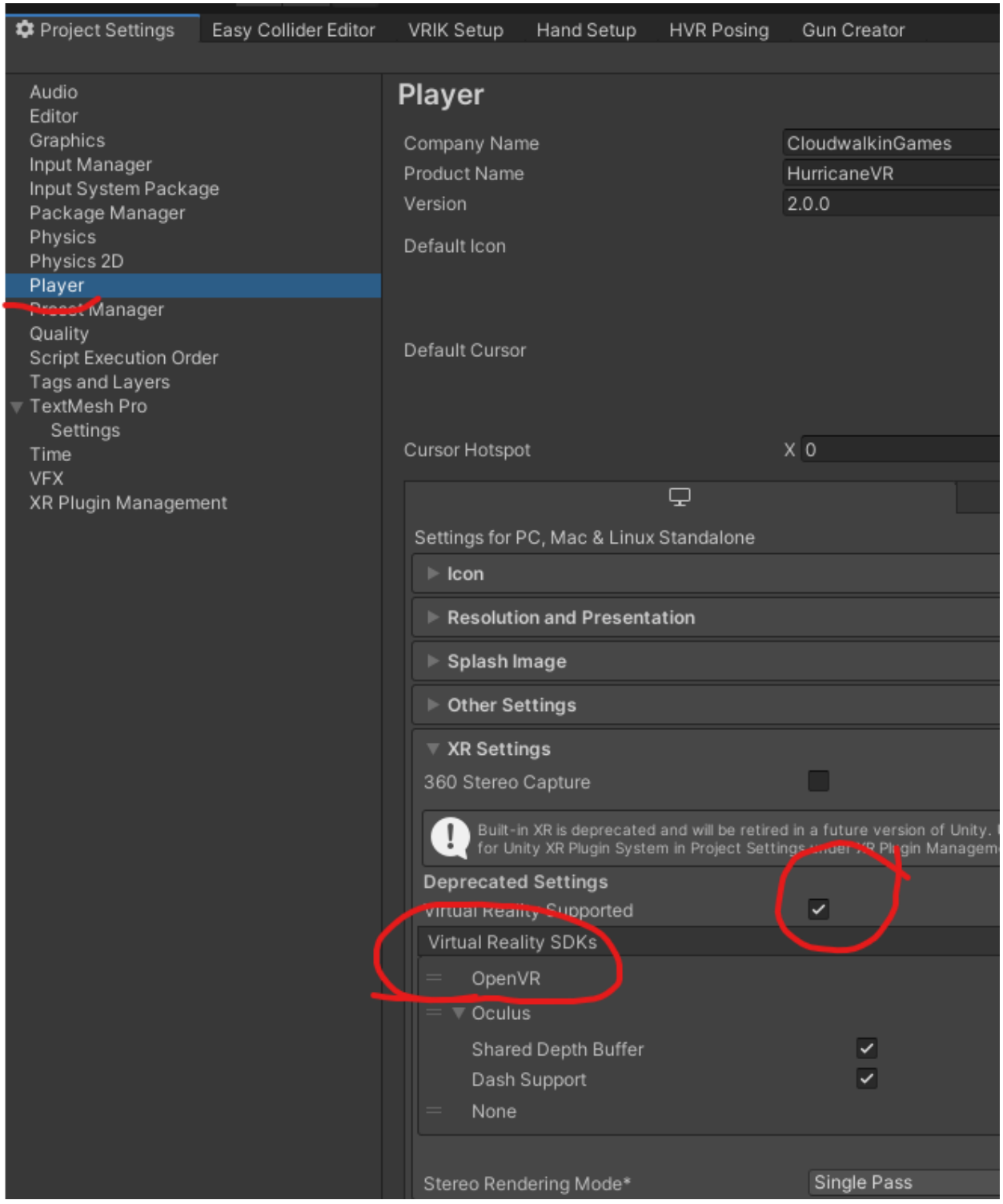

If you decide to convert to XR Plugin and receive this prompt, be sure to press Ok so that it will clean out the Legacy packages for you, if you fail to do so then you must remove the old packages manually.



# Unity 2019 Legacy VR

As mentioned in the Project and Packages Setup, Legacy VR is no longer viable for Oculus store builds. This setup process is only to be used if you are targeting PCVR via SteamVR and using Unity 2019.4 LTS.

1. Install OpenVR Desktop v2.0.5 from the Package Manager
2. Install Package “XR Legacy Input Helpers". This contains the HMD and Controller tracking components.
3. Open Project Settings (Edit → Project Settings → Player).
    - Virtual Reality Supported (checked)
    - OpenVR top of the list


Project Settings
Easy Collider Editor
VRIK Setup
Hand Setup
HVR Posing
Gun Creator
Audio
Editor
Graphics
Input Manager
Input System Package
Package Manager
Physics
Physics 2D
Player
Preset Manager
Quality
Script Execution Order
Tags and Layers
TextMesh Pro
Settings
Time
VFX
XR Plugin Management
Player
Company Name
CloudwalkinGames
Product Name
HurricaneVR
Version
2.0.0
Default Icon
Default Cursor
Cursor Hotspot
X 0
Settings for PC, Mac & Linux Standalone
Icon
Resolution and Presentation
Splash Image
Other Settings
XR Settings
360 Stereo Capture
Built-in XR is deprecated and will be retired in a future version of Unity.
for Unity XR Plugin System in Project Settings
Deprecated Settings
Virtual Reality Supported
Virtual Reality SDKs
OpenVR
Oculus
Shared Depth Buffer
Dash Support
None
Stereo Rendering Mode*
Single Pass

# Scene Setup

The following provides a brief explanation of the required objects / components needed in your scene for the asset to function.

At the minimum only the following is needed to be in the scene to get started:

- HVRGlobal Prefab: /HurricaneVR/Framework/Prefabs
- Rig Prefab

Example scene with the minimum required: /HurricaneVR/TechDemo/Scenes/scenes_barebones

# XR Rigs

Two player rigs are included with the framework already setup for you. Have a read through the Rig Breakdown section if you want to learn more about it's setup.

Rig prefabs are located in /HurricaneVR/TechDemo/Prefabs/

- **TechDemoXRRig**: For use with Oculus / SteamVR plugins.
- **TechDemoXRRigOpenXR**: For use with OpenXR Plugin.
    - The tracked drivers have been replaced with Unity's new input system tracked drivers.
    - Prefab variant of TechDemoXRRig

**TIP**

Create a prefab variant of the Rig Prefab if you decide to customize the rig, as future rig updates will come in the framework prefabs.

# HVRGlobal

This prefab is used to hold framework required components and should be placed into a master scene in your game. It is advised to make a variant of this prefab so that you can override the fields of it's components so that updating the asset in the future painless.

## HVRInputManager

This component is required for the framework to function. It handles device detection and other various input functions. It automatically marks it's object as DontDestroyOnLoad so that it persists between scene changes.

The following fields are scriptable objects that you can clone to customize certain behaviours. Be sure to clone the original if you wish to modify the values and assign it in your own HVRGlobal prefab variant to make updating the asset easier for you.

# Finger Settings

HVRFingerSettings scriptable object that defines how much each capacitive button affects the bending of each finger.

As of this writing, OpenXR via Unity still does not support Knuckles finger tracking out of the box, the KnucklesOverrideGripFingers can be enabled to fallback to standard finger curl behaviour like Oculus controllers.

# Controller Offsets

HVRControllerOffsets scriptable object that defines controller offsets based on the active VR SDK and the device connected.

# Grab Haptics

HVRGrabHaptics defines haptic values used in the following interaction events:

- Hand Grab
- Hand Release
- Hand Hover
- Force Grab
- Force Hover

# Deadzones

Not entirely sure how useful device specific deadzones is, but each device can have a deadzone value applied. 'Override Deadzone' and 'Deadzone Override' can be used to provide a single deadzone value.

The controller component will ignore values less than the provided deadzones when reading thumbstick inputs.