HurricaneVR

Overview

HurricaneVR is a Physics-Based Interaction and Hand Posing toolkit built completely with Non-Kinematic Physics in mind. Enhance the immersion of your games with grab point pose editing right in the scene window, or use the Half-Life: Alyx style dynamic pose solver as a fallback. Easily create hand poses in debug mode using Oculus Quest hand tracking or the dynamic pose solver.

Feature Highlights

- Grabbing any object requires only one component on that object.
- Physics Hands and Physics-Based Grabbing.
- Player Controller
  - Smooth Locomotion
  - Smooth Turning
  - Snap Turning
  - Physics-Checked Teleporting
- Easy Posable Grab Point Editing with per finger blendable animations
  - Edit hand poses right in the scene window for object.
  - Joint offset per grab point to adjust angles for guns / swords etc.
- Half-Life: Alyx Style Dynamic Hand Pose Solver
- Posable Grab Point Recording while debugging in the editor
  - Works well with dynamic pose solver
  - Supports Oculus Hand tracking
- Physics-Based Force Grabber
  - Configurable to behave like either Half-Life: Alyx or Walking Dead S&S style
  - Grabbable rotates to designated Posable Grab Point
- Socketable Grabbers
  - Filterable by string, gameobject, enum, or enum flags
  - Set grabbables to start in specified sockets
  - Permanently link grabbables to a desired socket; dropping returns to the socket.
- Samples included:
  - Interactables: doors, drawers, dials, buttons, and levers
  - Over the shoulder backpack inventory

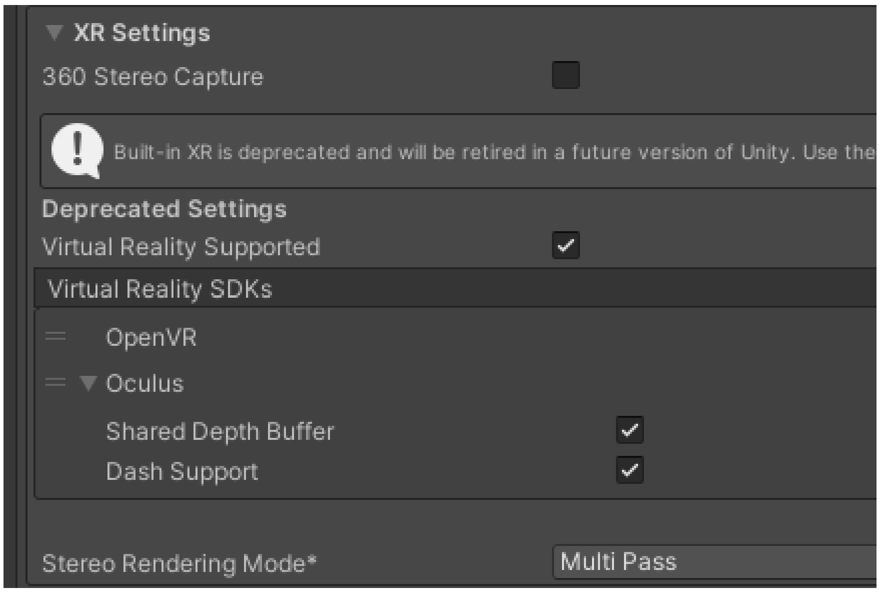
- Pistol with destructible objects
- Auto collecting chest inventory

Setup

If you have already setup your Unity VR environment you can skip this section. Choose between Legacy VR (only available in versions prior to 2020) or XR Plugin Management (2019+).

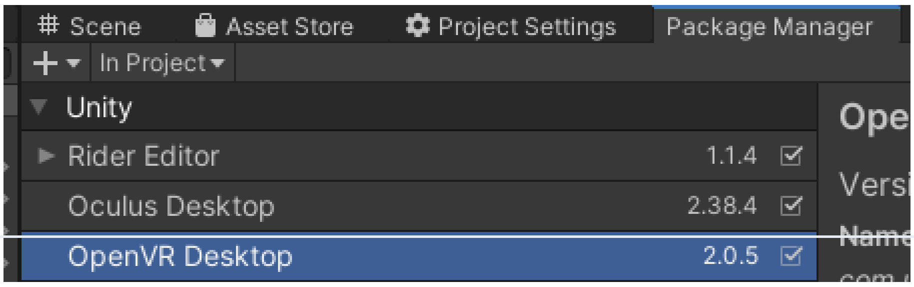
Legacy VR

Legacy VR is recommended as it supports the most headsets and is stable compared to the new XR Plugin system. Open your Project Settings and navigate to the Player Tab and check Virtual Reality Supported as seen below. Adding the Oculus and / or OpenVR packages will install the corresponding package in the package manager.



XR Settings
360 Stereo Capture ☐
Built-in XR is deprecated and will be retired in a future version of Unity. Use the
Deprecated Settings
Virtual Reality Supported ✓
Virtual Reality SDKs
OpenVR
Oculus
Shared Depth Buffer ✓
Dash Support ✓
Stereo Rendering Mode* Multi Pass

Ensure Oculus Desktop and OpenVR Desktop packages are installed in your project from the Package Manager.

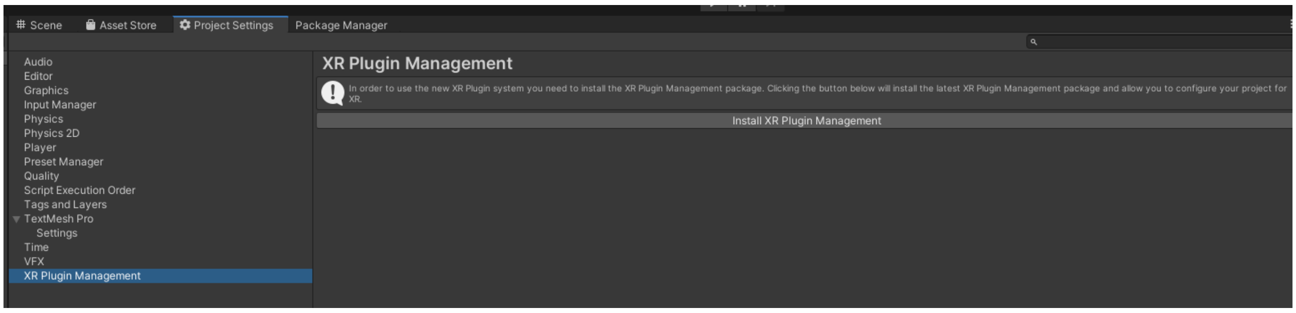


Unity		
Rider Editor	1.1.4	✓
Oculus Desktop	2.38.4	✓
OpenVR Desktop	2.0.5	✓

XR Plugin Management

XR Plugin Management is available in 2019, and becomes the only option in 2020 and beyond. Not compatible with Legacy VR; ensure sure Legacy VR packages are disabled. To set up, follow the directions below:

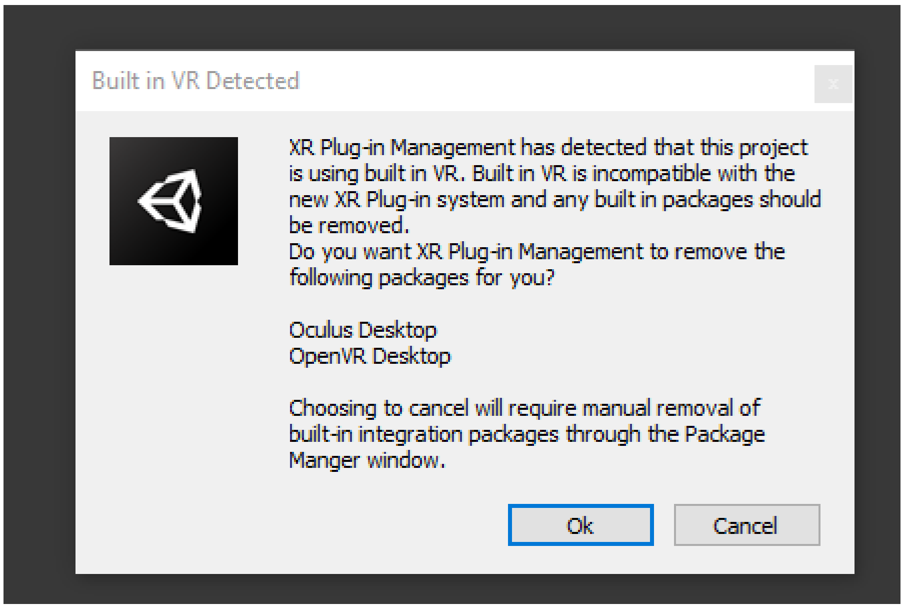
Navigate to the XR Plugin Management tab in your Project Settings and press Install XR Plugin Management.



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.

If you happen to have had Legacy VR installed already, you will be prompted with the message shown below. You must press 'Ok,' otherwise you will have to remove the packages manually.



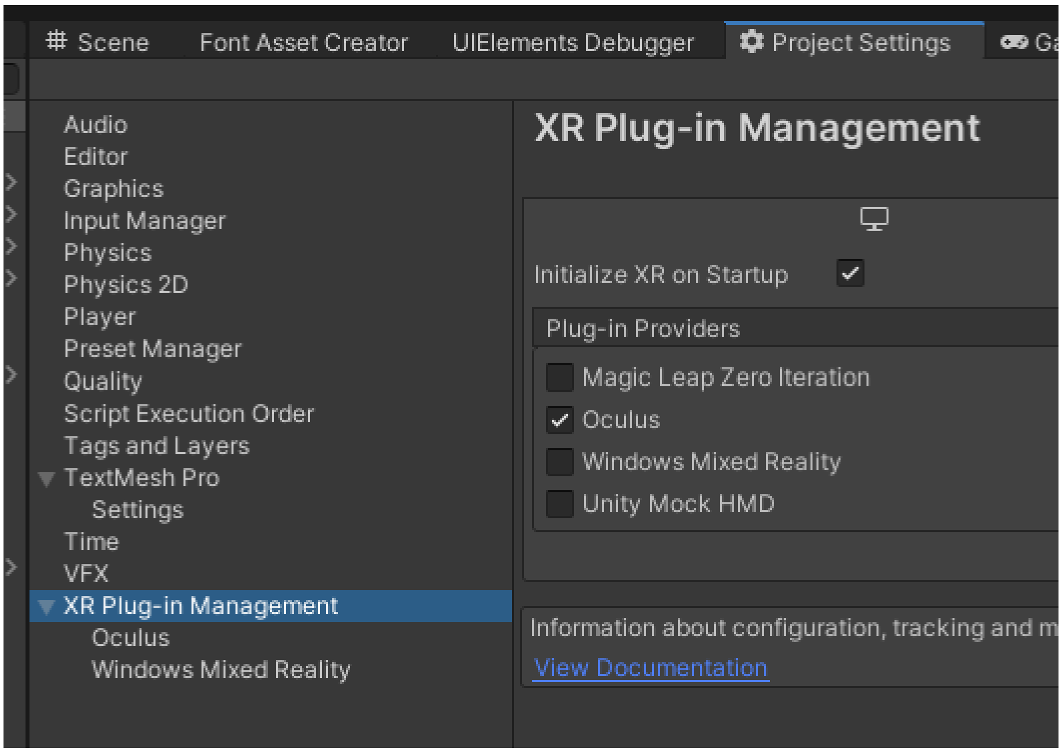
Built in VR Detected

XR Plug-in Management has detected that this project is using built in VR. Built in VR is incompatible with the new XR Plug-in system and any built in packages should be removed.
Do you want XR Plug-in Management to remove the following packages for you?

Oculus Desktop
OpenVR Desktop

Choosing to cancel will require manual removal of built-in integration packages through the Package Manger window.

Select the Plugin-in Providers you wish to support. This will trigger those plugins to install.



Scene | Font Asset Creator | UIElements Debugger | Project Settings | Ga

- Audio
- Editor
- Graphics
- Input Manager
- Physics
- Physics 2D
- Player
- Preset Manager
- Quality
- Script Execution Order
- Tags and Layers
- TextMesh Pro
  - Settings
- Time
- VFX
- XR Plug-in Management
  - Oculus
  - Windows Mixed Reality

XR Plug-in Management

Initialize XR on Startup [✓]

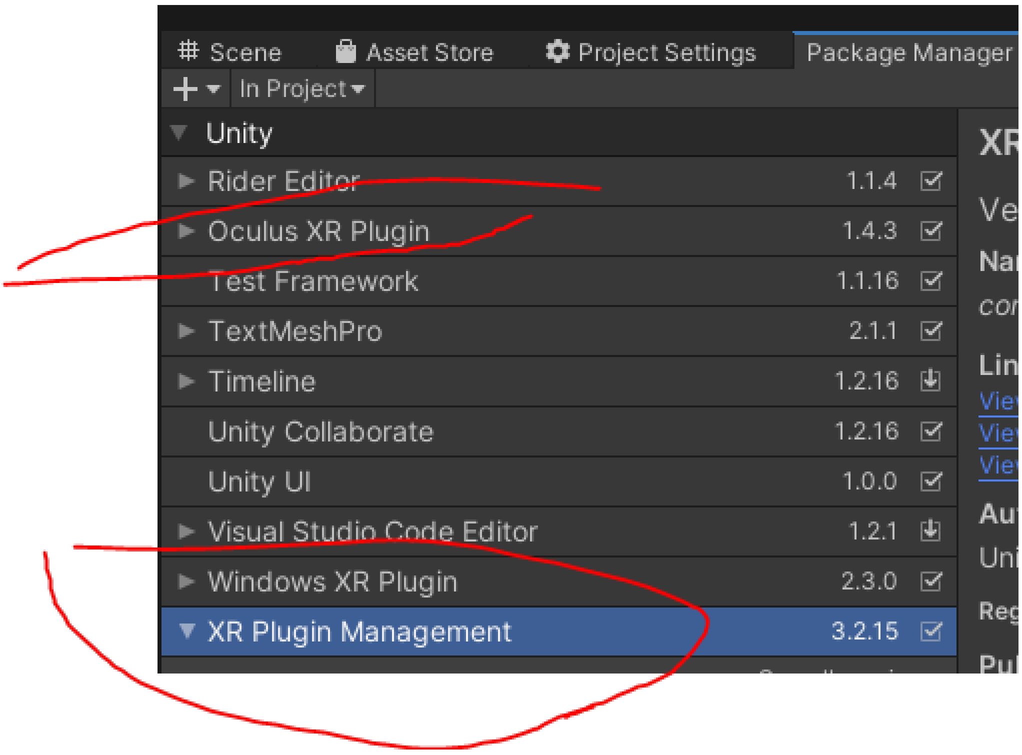
Plug-in Providers

- [ ] Magic Leap Zero Iteration
- [✓] Oculus
- [ ] Windows Mixed Reality
- [ ] Unity Mock HMD

Information about configuration, tracking and m

View Documentation

Verify your XR Plugins were installed in the package manager.



Scene | Asset Store | Project Settings | Package Manager

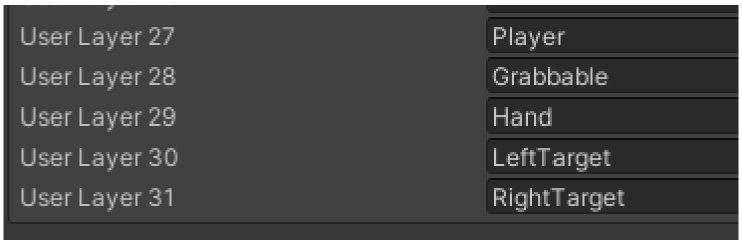
+ | In Project

Unity

Rider Editor	1.1.4	✓
Oculus XR Plugin	1.4.3	✓
Test Framework	1.1.16	✓
TextMeshPro	2.1.1	✓
Timeline	1.2.16	⤓
Unity Collaborate	1.2.16	✓
Unity UI	1.0.0	✓
Visual Studio Code Editor	1.2.1	⤓
Windows XR Plugin	2.3.0	✓
XR Plugin Management	3.2.15	✓

Physics Settings

Add the following layers to your project settings: Tags and Layers

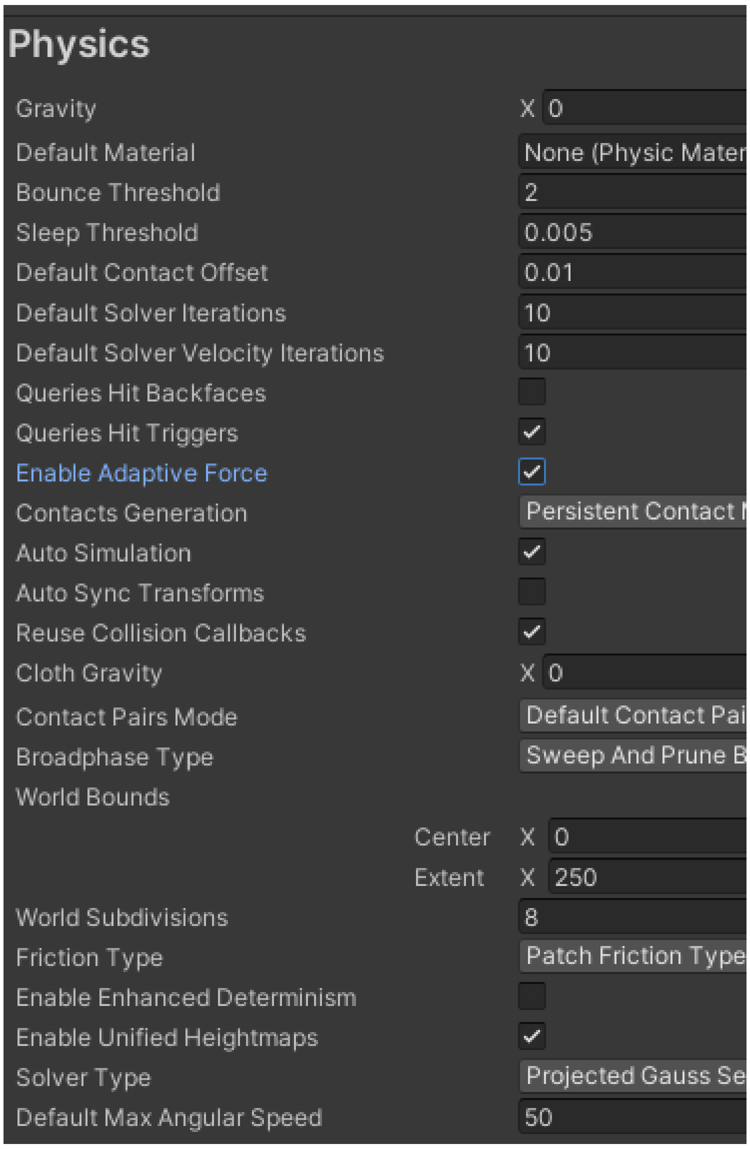


User Layer 27	Player
User Layer 28	Grabbable
User Layer 29	Hand
User Layer 30	LeftTarget
User Layer 31	RightTarget

Assign Layers

1. Set your PlayerController object layer to Player
2. Set your LeftHand and RightHand prefabs to Hand
3. Optionally assign Grabbable to your grabbable objects. If you do not, by default their layers will be assigned on Start

Set your physics settings to the following. Default solver settings can be increased if you experience unstable physics at the expense of performance. Default Max Angular speed needs to be increased for the physics hand tracking to work correctly.

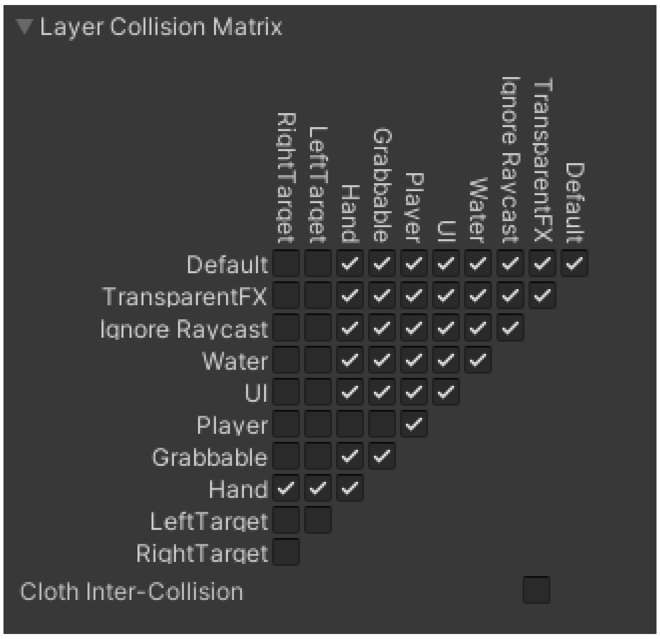


Physics

Gravity	X 0
Default Material	None (Physic Mater
Bounce Threshold	2
Sleep Threshold	0.005
Default Contact Offset	0.01
Default Solver Iterations	10
Default Solver Velocity Iterations	10
Queries Hit Backfaces	☐
Queries Hit Triggers	✓
Enable Adaptive Force	✓
Contacts Generation	Persistent Contact
Auto Simulation	✓
Auto Sync Transforms	☐
Reuse Collision Callbacks	✓
Cloth Gravity	X 0
Contact Pairs Mode	Default Contact Pai
Broadphase Type	Sweep And Prune B
World Bounds	
Center	X 0
Extent	X 250
World Subdivisions	8
Friction Type	Patch Friction Type
Enable Enhanced Determinism	☐
Enable Unified Heightmaps	✓
Solver Type	Projected Gauss Se
Default Max Angular Speed	50

Layer Collision Matrix

Setup the layer collision matrix for the newly added layers like below.



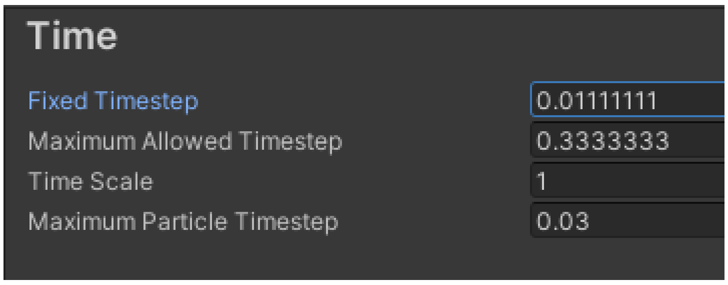
Layer Collision Matrix

	RightTarget	LeftTarget	Hand	Grabbable	Player	UI	Water	Ignore Raycast	TransparentFX	Default
Default	☐	☐	✓	✓	✓	✓	✓	✓	✓	✓
TransparentFX	☐	☐	✓	✓	✓	✓	✓	✓	✓	
Ignore Raycast	☐	☐	✓	✓	✓	✓	✓	✓		
Water	☐	☐	✓	✓	✓	✓	✓			
UI	☐	☐	✓	✓	✓	✓				
Player	☐	☐	☐	☐	✓					
Grabbable	☐	☐	✓	✓						
Hand	✓	✓	✓							
LeftTarget	☐	☐								
RightTarget	☐									

Cloth Inter-Collision ☐

Time:

Fixed Timestep. Most headsets support 90hz. In order to provide a fluid experience for your user, you should set this to 90 frames per second. For Oculus Quest you can set this to 1/72.



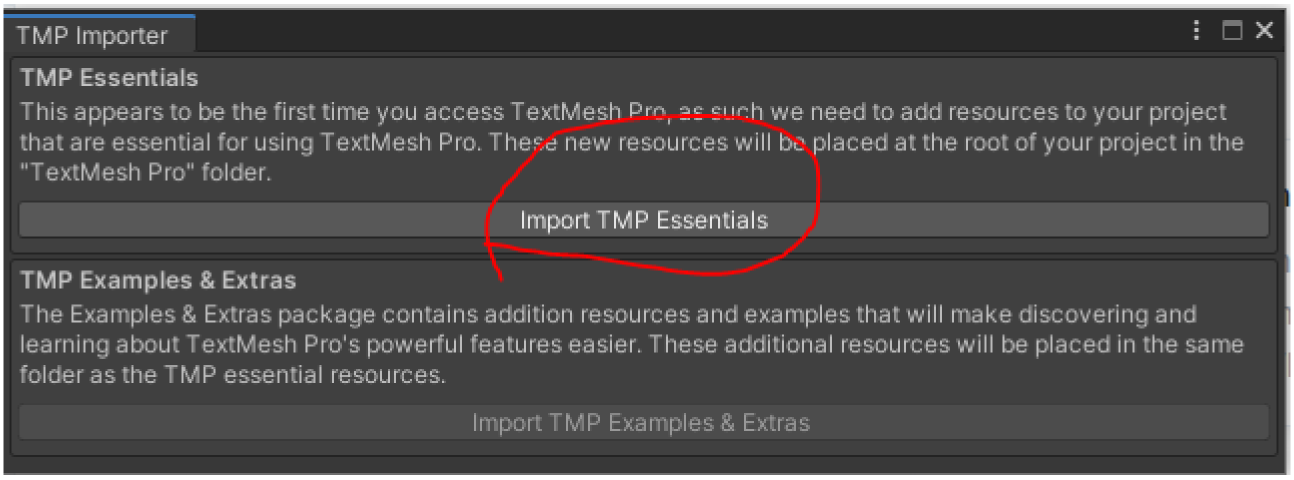
Time

Fixed Timestep	0.01111111
Maximum Allowed Timestep	0.3333333
Time Scale	1
Maximum Particle Timestep	0.03

Dependencies

TextMesh Pro

The samples provided make use of TextMesh Pro for high quality text in VR. The font sizes may bug if you import the TMP package after importing HurricaneVR. If that is the case, close the example scene and then reopen it.



TMP Importer

TMP Essentials

This appears to be the first time you access TextMesh Pro, as such we need to add resources to your project that are essential for using TextMesh Pro. These new resources will be placed at the root of your project in the "TextMesh Pro" folder.

Import TMP Essentials

TMP Examples & Extras

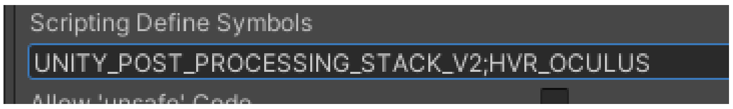
The Examples & Extras package contains addition resources and examples that will make discovering and learning about TextMesh Pro's powerful features easier. These additional resources will be placed in the same folder as the TMP essential resources.

Import TMP Examples & Extras

Oculus Integration

If you wish to use the Oculus Integration package, please download from https://assetstore.unity.com/packages/tools/integration/oculus-integration-82022.

Add the HVR_OCULUS preprocessor directive under Project Settings / Player / Other Settings



Scripting Define Symbols

UNITY_POST_PROCESSING_STACK_V2;HVR_OCULUS

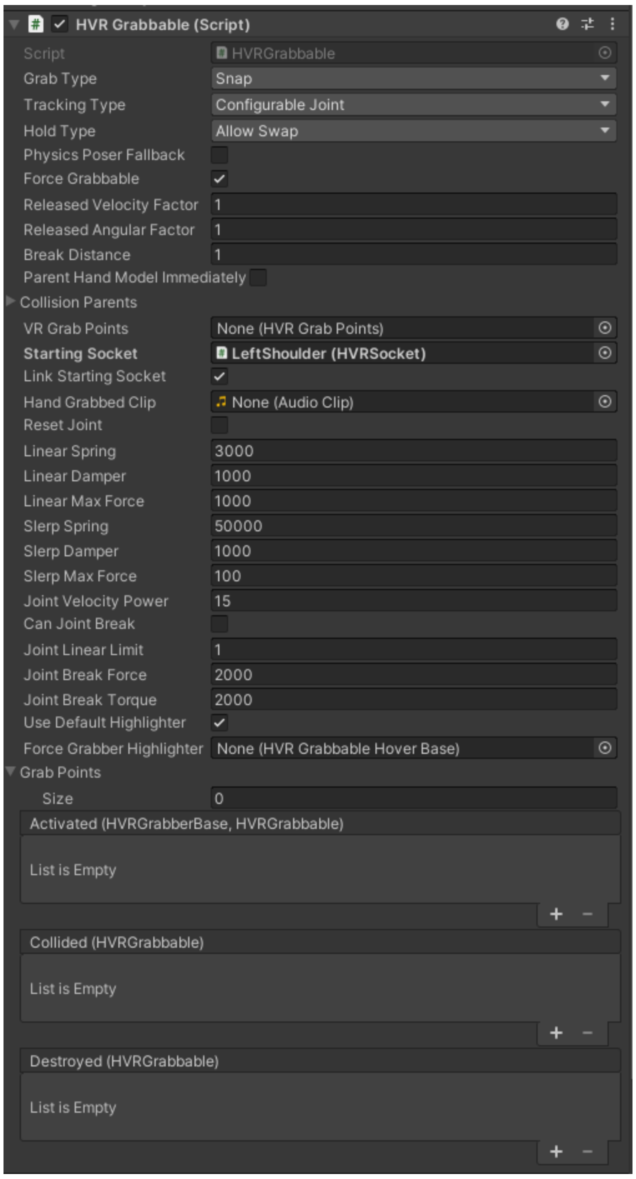
Interactions

HVRGrabbable

Allows the game object to be grabbed by appropriate grabbers. To setup a Physics-Based Grabbable, add a non-kinematic RigidBody component as well as an HVRGrabbable component.

1. **GrabType**
    1. Snap: The hand model will snap to the closest grab point based on distance and rotation deltas
    2. Physics Poser: The hand will open and close around the objects colliders after it hits the palm of the hand against the closest collider
2. **Physics Poser Fallback**
    1. If there are no valid snap points, then the component will try a physics-based grab as a fallback
3. **TrackingType**
    1. Configurable Joint
        1. Joint Settings (Spring, Damper, MaxForce) can be modified to create fake intertia
        2. Joint Velocity Power: lower values dampen the spring earlier, higher values dampen when grabbable is closer to the hand.
    2. Fixed Joint
    3. None: Used when you do not want physics interactions on the held object.
4. **HoldType**
    1. One Hand: One hand may hold the grabbable and swapping is not possible
    2. Allow Swap: One hand may hold the grabbable, and the other hand can take the grabbable
    3. Two Hands: The grabbable may be held by two hands
    4. Many Hands: In case you have multiplayer and desire to use 3 or more hands
5. **Throwing Modifiers**
    1. Released Velocity Factor: Multiplied against the final velocity calculation when releasing.
    2. Released Angular Factor: Multiplied against the final angular velocity calculation when releasing.
6. **Break Distance**

    1. Set a value that determines at what point the grabbable will be dropped if the distance between the grabber and the grabbable exceeds it.
7. **Parent Hand Model Immediately**
    1. If true, the hand will not wait for the grabbable to rotate into position.
8. **Starting Socket**
    1. Determines the socket that this grabbable will start out snapped to
9. **Link Starting Socket**
    1. If true, then the socket and grabbable are permanently linked. Dropping the object will return it to the socket and no other grabbable is valid in that socket.
10. **Reset Joint**
    1. Re-creates the joint once the grabbable rotates into position, this is only needed if you expect external forces to be applied to your grabbable while holding it.
11. **Can Joint Break**
    1. If you want your object to break away under load, set the Tracking Type to Fixed Joint and check this box, modifying the Joint Break Force and Joint Break Torque.
12. **Use Default Highlighter**
    1. By default the grabbable will create a ring based highlighter to let you know it can be remote grabbed. If this is not desired, uncheck this field. Then, provide your own highlighter in the ForceGrabberHighlighter field.
13. **GrabPoints**
    1. Collection of grab points to snap to. If this is empty on startup the grab points will be automatically filled by checking children for HVRGrabPoints component, or "Grab



HVR Grabbable (Script)

Script	HVRGrabbable
Grab Type	Snap
Tracking Type	Configurable Joint
Hold Type	Allow Swap
Physics Poser Fallback	☐
Force Grabbable	✓
Released Velocity Factor	1
Released Angular Factor	1
Break Distance	1
Parent Hand Model Immediately	☐
Collision Parents	
VR Grab Points	None (HVR Grab Points)
Starting Socket	LeftShoulder (HVRSocket)
Link Starting Socket	✓
Hand Grabbed Clip	None (Audio Clip)
Reset Joint	☐
Linear Spring	3000
Linear Damper	1000
Linear Max Force	1000
Slerp Spring	50000
Slerp Damper	1000
Slerp Max Force	100
Joint Velocity Power	15
Can Joint Break	☐
Joint Linear Limit	1
Joint Break Force	2000
Joint Break Torque	2000
Use Default Highlighter	✓
Force Grabber Highlighter	None (HVR Grabbable Hover Base)
Grab Points	
Size	0

Activated (HVRGrabberBase, HVRGrabbable)

List is Empty

Collided (HVRGrabbable)

List is Empty

Destroyed (HVRGrabbable)

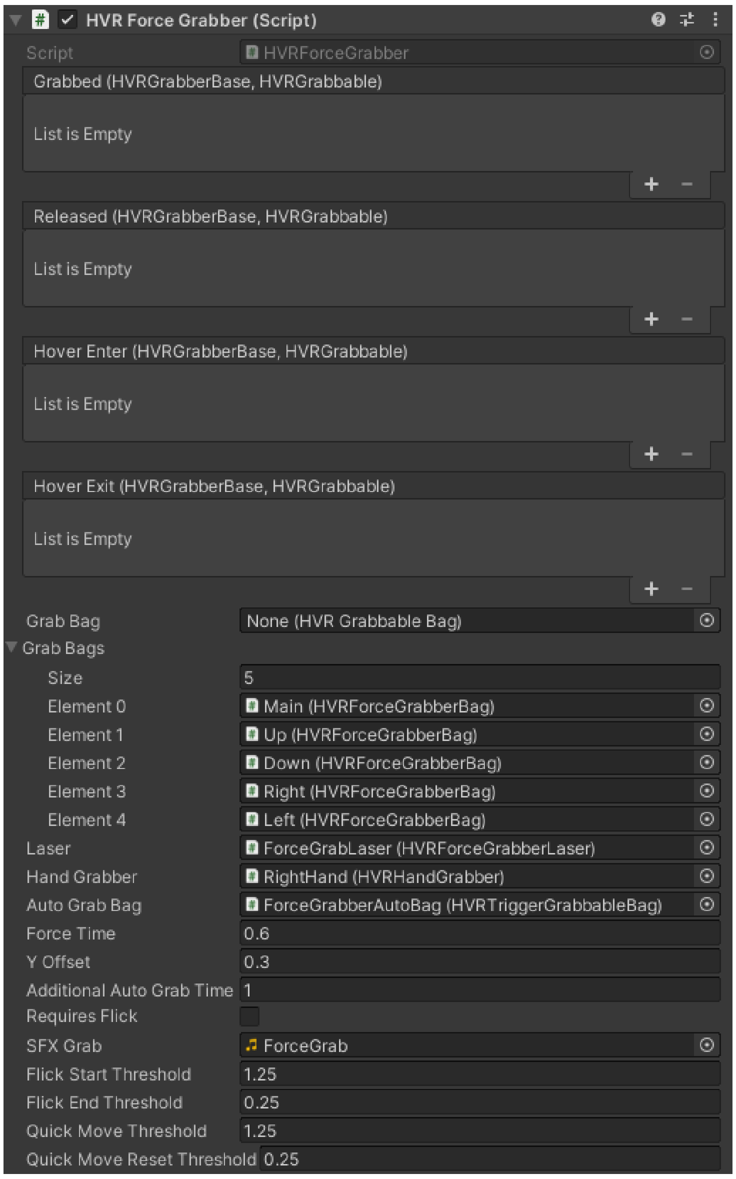
List is Empty

HVRForceGrabber

The force grabber component lets you grab objects from a distance. It is configurable to require a quick movement of the hand. If Require Flick is checked, the flick (angular velocity) or quick move (velocity) threshold must be met while holding down the grab button to intiate the grab. Otherwise, simply holding down the grab button will remotely grab the item.

1. **Grab Bags**
   1. Priority based grab bags, 0 being the highest priority, to prioritize which grabbable to select if multiple end up in your trigger zone.

2. **Laser**
    1. The line renderer that will draw a laser when requiring flick movements.
3. **Force Time**
    1. Defines (in seconds) how long it takes for the item to reach your hand no matter the distance.
4. **Y Offset**
    1. The grabbable will arc in a parabola peaking this high above your hand.
5. **Additional Auto Grab Time**
    1. The additional time after the Force Time you have to automatically grab the object if you are holding down the grab button.
6. **SFX Grab**
    1. The sound effect to play when succesfully force grabbing.
7. **Auto Grab Bag**
    1. Trigger bag that will automatically grab the object if holding down the grab button.


HVR Force Grabber (Script)

Script	HVRForceGrabber

Grabbed (HVRGrabberBase, HVRGrabbable)

List is Empty

Released (HVRGrabberBase, HVRGrabbable)

List is Empty

Hover Enter (HVRGrabberBase, HVRGrabbable)

List is Empty

Hover Exit (HVRGrabberBase, HVRGrabbable)

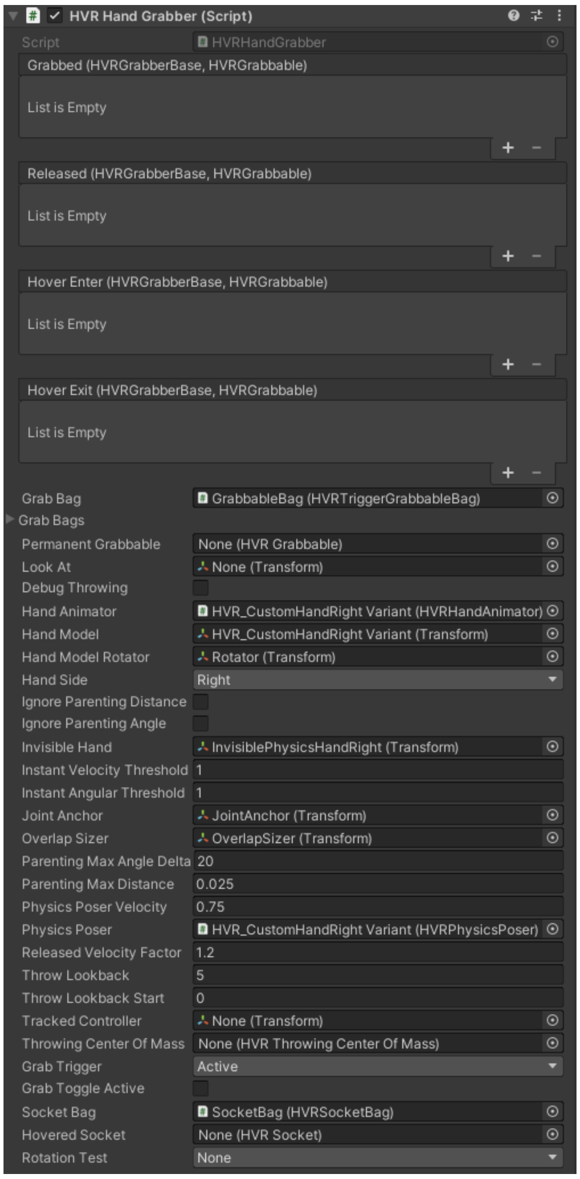
List is Empty

Grab Bag	None (HVR Grabbable Bag)
Grab Bags	
Size	5
Element 0	Main (HVRForceGrabberBag)
Element 1	Up (HVRForceGrabberBag)
Element 2	Down (HVRForceGrabberBag)
Element 3	Right (HVRForceGrabberBag)
Element 4	Left (HVRForceGrabberBag)
Laser	ForceGrabLaser (HVRForceGrabberLaser)
Hand Grabber	RightHand (HVRHandGrabber)
Auto Grab Bag	ForceGrabberAutoBag (HVRTriggerGrabbableBag)
Force Time	0.6
Y Offset	0.3
Additional Auto Grab Time	1
Requires Flick	☐
SFX Grab	ForceGrab
Flick Start Threshold	1.25
Flick End Threshold	0.25
Quick Move Threshold	1.25
Quick Move Reset Threshold	0.25

HVRHandGrabber

1. **Grab Bag**
    1. The bag that defines what grabbables can be picked up.
2. **Grab Bags**
    1. Additional priority based bags for fine tune grabbing of close objects.
3. **Hand Model**
    1. The hand model transform
4. **Hand Model Rotator**
    1. An empty transform for the framework to easily calculate pose deltas when grabbing.

5. **HandSide**
    1. Determines whether this is the left or right hand
6. **Ignore Parenting Distance**
    1. Ignores the distance check when snapping to a grab point.
7. **Ignore Parenting Angle**
    1. Ignores the rotation delta check when snapping to a grab point.
8. **Invisible Hand**
    1. Hand model with physics to prevent your hand from going through objects while you're grabbing onto something
9. **Instant Velocity / Angular Threshold**
    1. If the grabbable exceeds either threshold while physics grabbing, the hand will immediately snap to the object.
10. **Physics Poser Velocity**
    1. The speed that the hand moves to the grabbable when doing a physics based grab.
11. **Released Velocity Factor**
    1. Multiplied by the final throwing velocity.
12. **Throw Lookback**
    1. The number of frames to look back for throwing velocity.
13. **Throw Lookback Start**
    1. How many frames to skip before looking back for throwing velocity.
14. **Tracked Controller**
    1. The controller transform that is tracking the device.
15. **Throwing Center of Mass**
    1. Automatically chosen based on the device this hand is tracking
    2. HVRThrowingCenterOfMass component: Holds transforms per device that you can define on the hand that you think best represent the phsyics controllers center of mass. This greatly improves the accuracy of the throwing computation.
    3. If the transform for that device is not set, the hand's RigidBody world center of mass will be used.
16. **Grab Trigger:**
    1. Active / Toggle (Active requires that the button be held down to hold)
17. **Socket Bag**
    1. Used to detect which socket this hand is hovering over.



HVR Hand Grabber (Script)

Script	HVRHandGrabber

Grabbed (HVRGrabberBase, HVRGrabbable)

List is Empty

Released (HVRGrabberBase, HVRGrabbable)

List is Empty

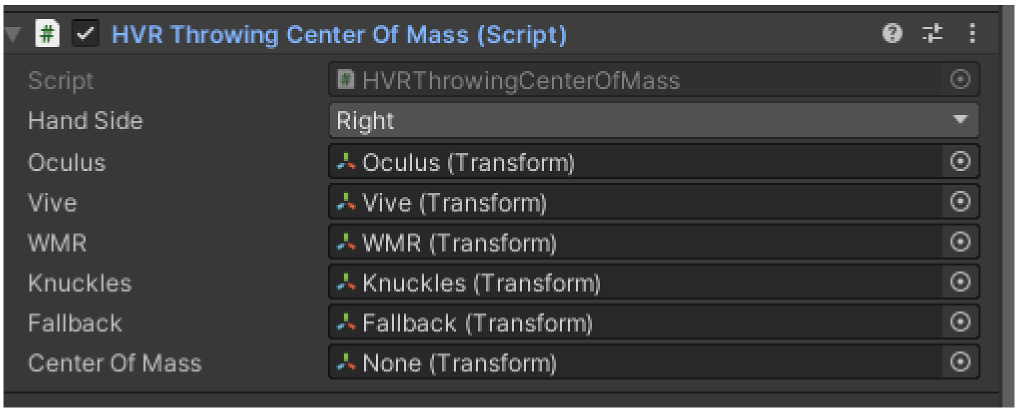
Hover Enter (HVRGrabberBase, HVRGrabbable)

List is Empty

Hover Exit (HVRGrabberBase, HVRGrabbable)

List is Empty

Grab Bag	GrabbableBag (HVRTriggerGrabbableBag)
Grab Bags	
Permanent Grabbable	None (HVR Grabbable)
Look At	None (Transform)
Debug Throwing	☐
Hand Animator	HVR_CustomHandRight Variant (HVRHandAnimator)
Hand Model	HVR_CustomHandRight Variant (Transform)
Hand Model Rotator	Rotator (Transform)
Hand Side	Right
Ignore Parenting Distance	☐
Ignore Parenting Angle	☐
Invisible Hand	InvisiblePhysicsHandRight (Transform)
Instant Velocity Threshold	1
Instant Angular Threshold	1
Joint Anchor	JointAnchor (Transform)
Overlap Sizer	OverlapSizer (Transform)
Parenting Max Angle Delta	20
Parenting Max Distance	0.025
Physics Poser Velocity	0.75
Physics Poser	HVR_CustomHandRight Variant (HVRPhysicsPoser)
Released Velocity Factor	1.2
Throw Lookback	5
Throw Lookback Start	0
Tracked Controller	None (Transform)
Throwing Center Of Mass	None (HVR Throwing Center Of Mass)
Grab Trigger	Active
Grab Toggle Active	☐
Socket Bag	SocketBag (HVRSocketBag)
Hovered Socket	None (HVR Socket)
Rotation Test	None

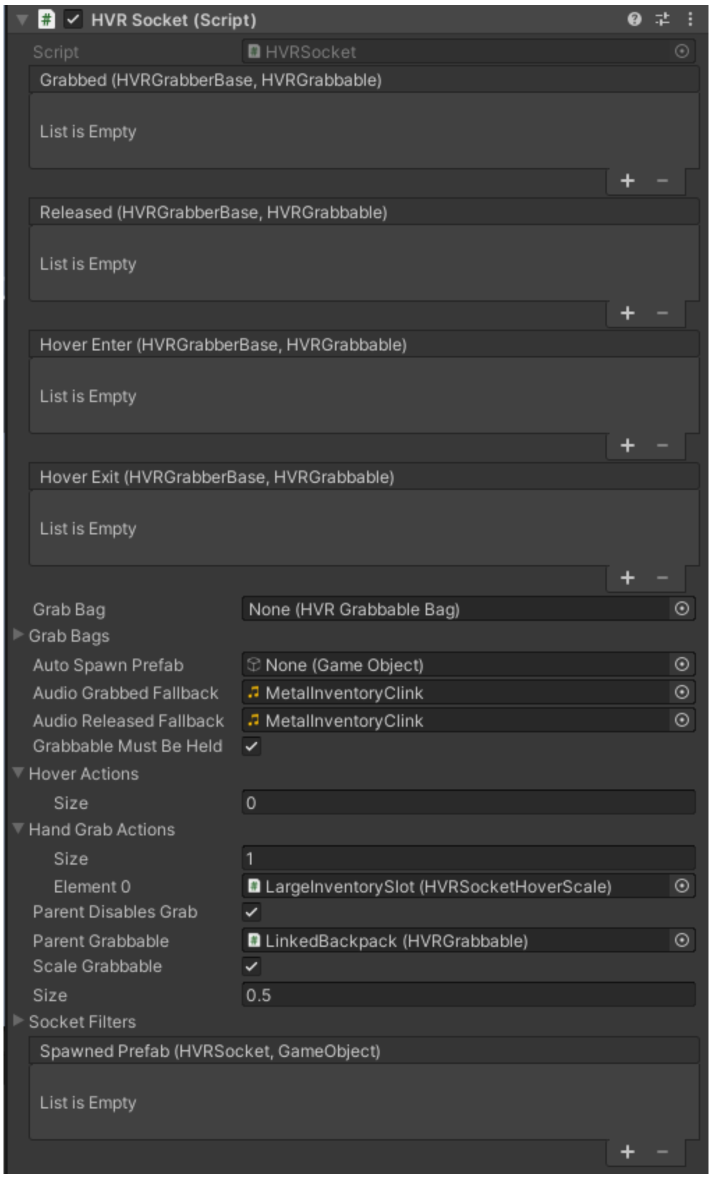


HVR Throwing Center Of Mass (Script)

Script	HVRThrowingCenterOfMass
Hand Side	Right
Oculus	Oculus (Transform)
Vive	Vive (Transform)
WMR	WMR (Transform)
Knuckles	Knuckles (Transform)
Fallback	Fallback (Transform)
Center Of Mass	None (Transform)

Sockets

1. **Grab Bag**: If none provided, one will be created using the colliders on this object.
2. **Auto Spawn Prefab:** Determines if this socket should automatically spawn an infinite supply of something
3. **Audio Fallbacks:** If the socketable doesn't have grabbed / released SFX then this will be used when grabbing or releasing a grabbable from the socket
4. **Grabbable Must Be Held:** If true, you must use HVRHandGrabber to place the grabbable into the socket. Else if false, then it can grab things out of the air.
5. **Hover Actions:** HVRSocketHoverAction components that will respond to hovering and unhovering grabbables in the grab bag.
6. **Hand Grab Actions:** HVRSocketHoverAction components that will respond when hovering and unhovering to the HVRHandGrabber
7. **Parent Disables Grab:** If a ParentGrabbable is provided and this is true, then you cannot remove items from this socket if that ParentGrabbable is also socketed. Useful for something like a backpack.
8. **Size:** Determines the amount to scale the socketed grabbables by if Scale Grabbable is checked.
9. **Socket Filters:** HVRSocketFilter components that will determine if a grabbable can be socketed.


HVR Socket (Script)

Script	HVRSocket

Grabbed (HVRGrabberBase, HVRGrabbable)

List is Empty

Released (HVRGrabberBase, HVRGrabbable)

List is Empty

Hover Enter (HVRGrabberBase, HVRGrabbable)

List is Empty

Hover Exit (HVRGrabberBase, HVRGrabbable)

List is Empty

Grab Bag	None (HVR Grabbable Bag)
Grab Bags	
Auto Spawn Prefab	None (Game Object)
Audio Grabbed Fallback	MetalInventoryClink
Audio Released Fallback	MetalInventoryClink
Grabbable Must Be Held	✓
Hover Actions	
Size	0
Hand Grab Actions	
Size	1
Element 0	LargeInventorySlot (HVRSocketHoverScale)
Parent Disables Grab	✓
Parent Grabbable	LinkedBackpack (HVRGrabbable)
Scale Grabbable	✓
Size	0.5
Socket Filters	

Spawned Prefab (HVRSocket, GameObject)

List is Empty

Socket Filtering:

1. HVRStringSocketFilter: Filter grabbables by string
2. HVREnumSocketFilter: Filter grabbables by custom define enums
    1. Subclass this using a generic of your enum type
3. HVREnumFlagsSocketFilter: Filter grabbables by custom defined enum flags
    1. Subclass this using a generic of your enum type
    2. Refer to the backpack example
4. HVRGrabbableSocketFilter: Filter by specified grabbable objects

Socketable Grabbables

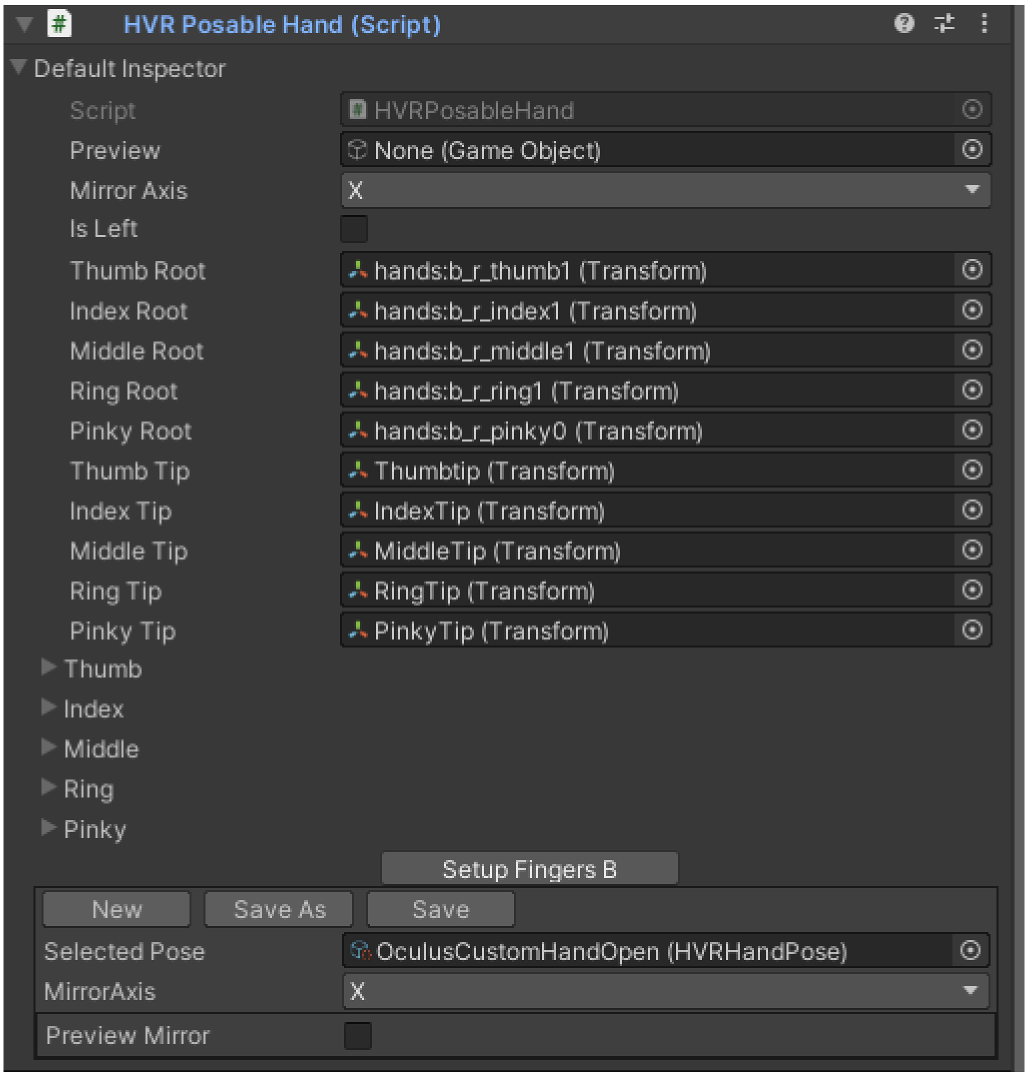
Add this component or subclasses to a HVRGrabbable to allow this grabbable to be socketed and filtered appropriately by the matching socket filter.

1. HVRStringSocketable: Matches up with HVRStringSocketFilter
2. HVREnumSocketable: Matches up with HVREnumSocketFilter
3. HVREnumFlagsSocketable: Matches up with HVREnumFlagsSocketFilter

Hand Posing

Posable Hands

The HVRPosableHand component holds bone information of your hand model. You can use this component to preview, modify, and update your hand poses. If you wish to use your own hand model with this framework simply add a "Tip" game object to the last bone of each finger with it's position right at the tip of the finger. Supply the root and tip transforms as below and press Setup Fingers to store the information of each bone.



HVR Posable Hand (Script)

Default Inspector

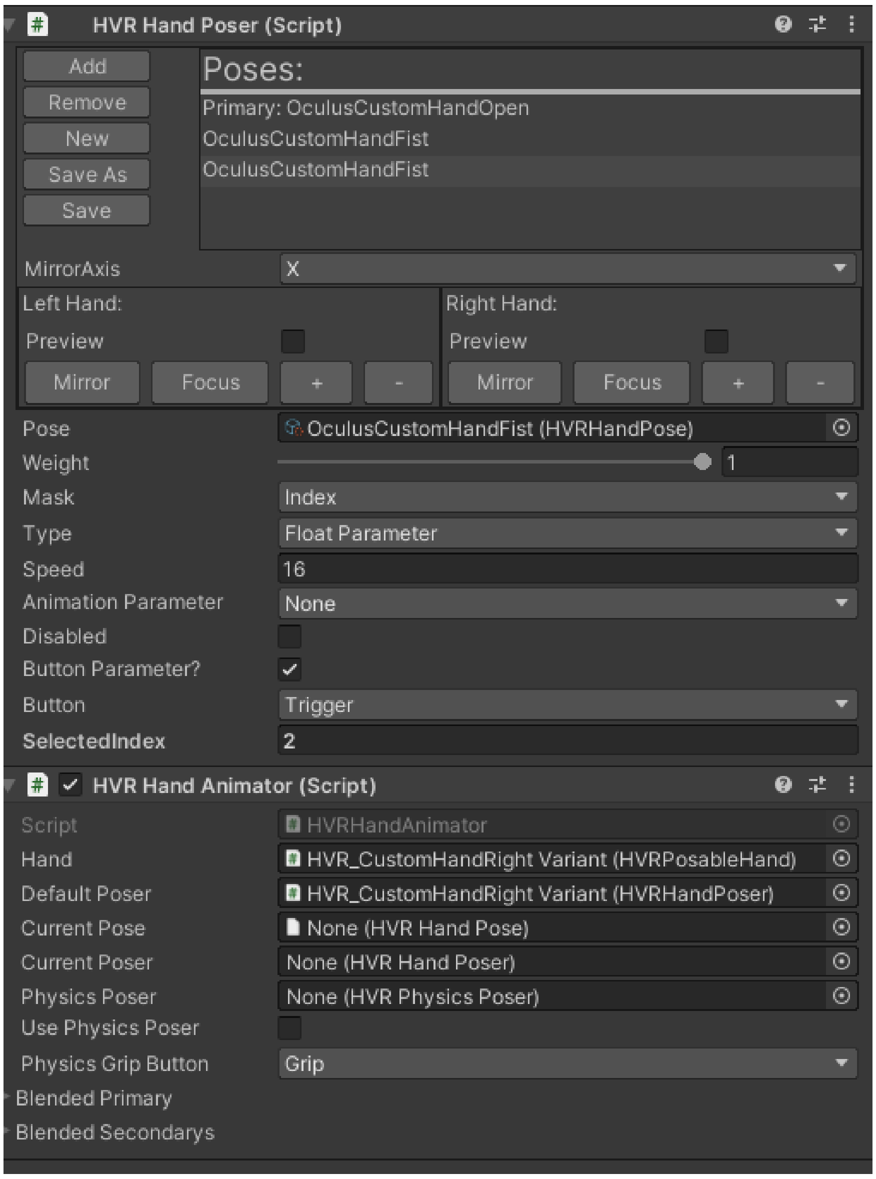
Script	HVRPosableHand
Preview	None (Game Object)
Mirror Axis	X
Is Left	
Thumb Root	hands:b_r_thumb1 (Transform)
Index Root	hands:b_r_index1 (Transform)
Middle Root	hands:b_r_middle1 (Transform)
Ring Root	hands:b_r_ring1 (Transform)
Pinky Root	hands:b_r_pinky0 (Transform)
Thumb Tip	Thumbtip (Transform)
Index Tip	IndexTip (Transform)
Middle Tip	MiddleTip (Transform)
Ring Tip	RingTip (Transform)
Pinky Tip	PinkyTip (Transform)

Selected Pose	OculusCustomHandOpen (HVRHandPose)
MirrorAxis	X
Preview Mirror	

Default Hand Pose

The hand model has a HVRHandPoser component to manage the pose for your hand when it is not holding anything.

The HVRHandAnimator component is responsible for taking the currently assigned pose and animating it based on the pararameters you have set.


HVR Hand Poser (Script)

Poses:
Primary: OculusCustomHandOpen
OculusCustomHandFist
OculusCustomHandFist

MirrorAxis	X

Left Hand:
Preview ☐

Right Hand:
Preview ☐

Pose	OculusCustomHandFist (HVRHandPose)
Weight	1
Mask	Index
Type	Float Parameter
Speed	16
Animation Parameter	None
Disabled	☐
Button Parameter?	✓
Button	Trigger
SelectedIndex	2

HVR Hand Animator (Script)

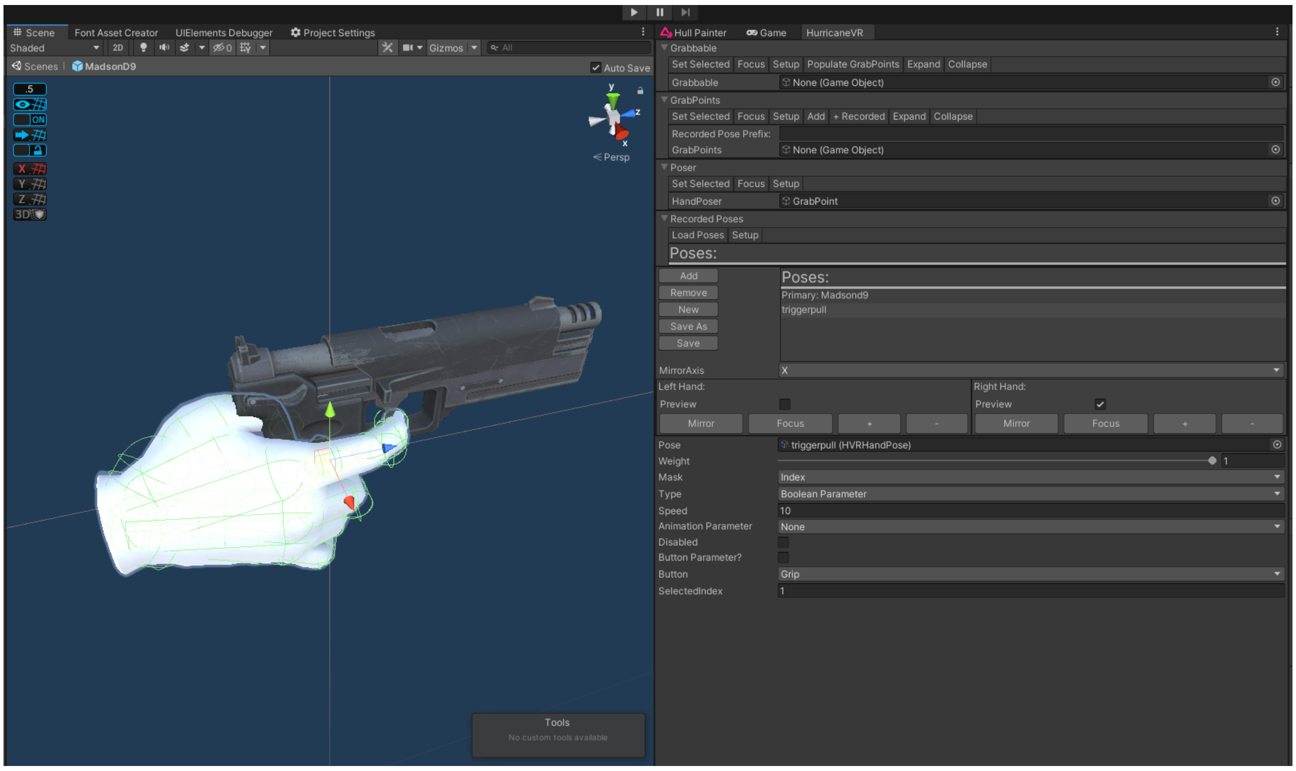
Script	HVRHandAnimator
Hand	HVR_CustomHandRight Variant (HVRPosableHand)
Default Poser	HVR_CustomHandRight Variant (HVRHandPoser)
Current Pose	None (HVR Hand Pose)
Current Poser	None (HVR Hand Poser)
Physics Poser	None (HVR Physics Poser)
Use Physics Poser	☐
Physics Grip Button	Grip

Blended Primary
Blended Secondarys

GrabPoint Posing

Grab Points have the HVRPosableGrabPoint component which requires a HVRHandPoser component for posing.

Hand poses for grab points can be modified directly in the editor. Each grab point poser has a primary pose with additional "blendable poses" in the event you want to animate the hand. These per finger animations can be activated by button press or any custom animation parameter that you define and set in your game.



Grabbable
Grabbable: None (Game Object)
GrabPoints
Recorded Pose Prefix:
GrabPoints: None (Game Object)
Poser
HandPoser: GrabPoint
Recorded Poses
Poses:
Poses:
Primary: Madsond9
triggerpull

MirrorAxis	X
Left Hand: Preview	☐
Right Hand: Preview	✓
Pose	triggerpull (HVRHandPose)
Weight	1
Mask	Index
Type	Boolean Parameter
Speed	10
Animation Parameter	None
Disabled	☐
Button Parameter?	☐
Button	Grip
SelectedIndex	1

The HVRHandPoser component can be managed in the HurricaneVR management window from Tools -> HurricaneVR or edited in the inspector of the component.

Input Handling

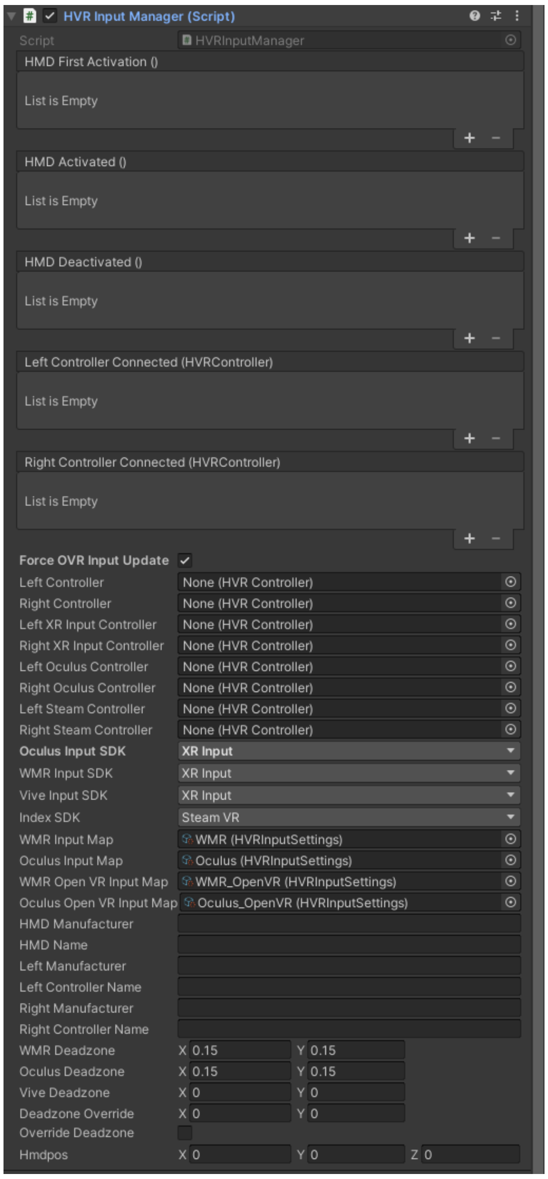
HVRInputManager

Inputs from various devices are encapsulated in HVRController components. The HVRInputManager manages these components for you based on which device is connected.

Each device can have separate Input SDK's defined. For example, if desired to use OVRInput for Oculus devices then switch the Oculus Input SDK to Oculus. If you are not using OVRManager in your game, make sure ForceOVRInput date is checked when using OVRInput.

Default dead-zones can be set at your discretion. If you would like the user to override this in a menu or configuration you provide, simply set Override Deadzone to true and set the Deadzone Override field in your code.

SteamVR is not currently supported but will be soon!



HVR Input Manager (Script)

Script	HVRInputManager

HMD First Activation ()

List is Empty

HMD Activated ()

List is Empty

HMD Deactivated ()

List is Empty

Left Controller Connected (HVRController)

List is Empty

Right Controller Connected (HVRController)

List is Empty

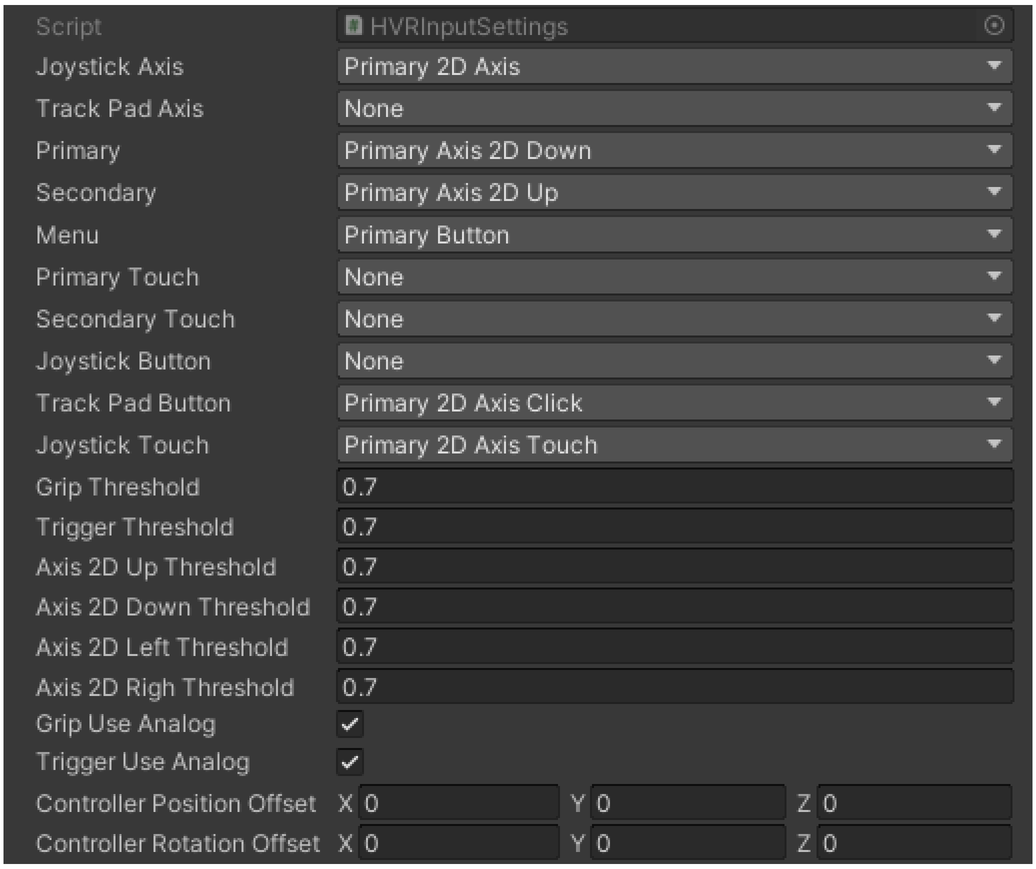
Force OVR Input Update	✓
Left Controller	None (HVR Controller)
Right Controller	None (HVR Controller)
Left XR Input Controller	None (HVR Controller)
Right XR Input Controller	None (HVR Controller)
Left Oculus Controller	None (HVR Controller)
Right Oculus Controller	None (HVR Controller)
Left Steam Controller	None (HVR Controller)
Right Steam Controller	None (HVR Controller)
Oculus Input SDK	XR Input
WMR Input SDK	XR Input
Vive Input SDK	XR Input
Index SDK	Steam VR
WMR Input Map	WMR (HVRInputSettings)
Oculus Input Map	Oculus (HVRInputSettings)
WMR Open VR Input Map	WMR_OpenVR (HVRInputSettings)
Oculus Open VR Input Map	Oculus_OpenVR (HVRInputSettings)
HMD Manufacturer	
HMD Name	
Left Manufacturer	
Left Controller Name	
Right Manufacturer	
Right Controller Name	
WMR Deadzone	X 0.15 Y 0.15
Oculus Deadzone	X 0.15 Y 0.15
Vive Deadzone	X 0 Y 0
Deadzone Override	X 0 Y 0
Override Deadzone	
Hmdpos	X 0 Y 0 Z 0

HVRInputSettings

You can override which XRInput feature is mapped to HVRButton types per device type. Desired settings may vary for each user, so it is configurable per your needs.

Each controller reports their initial position and rotation offsets differently. You can modify these here to move your hand into a more natural position per device.

https://docs.unity3d.com/Manual/xr_input.html



Script	HVRInputSettings
Joystick Axis	Primary 2D Axis
Track Pad Axis	None
Primary	Primary Axis 2D Down
Secondary	Primary Axis 2D Up
Menu	Primary Button
Primary Touch	None
Secondary Touch	None
Joystick Button	None
Track Pad Button	Primary 2D Axis Click
Joystick Touch	Primary 2D Axis Touch
Grip Threshold	0.7
Trigger Threshold	0.7
Axis 2D Up Threshold	0.7
Axis 2D Down Threshold	0.7
Axis 2D Left Threshold	0.7
Axis 2D Righ Threshold	0.7
Grip Use Analog	✓
Trigger Use Analog	✓
Controller Position Offset	X 0 Y 0 Z 0
Controller Rotation Offset	X 0 Y 0 Z 0