SteamVRInput - Class FAQ/Documentation

FAQ
QuickStart
Documentation

#This requires the SteamVR package from the asset store in order to function

The SteamVRInput Class is designed to ease coding and accelerate prototyping inside SteamVR. This allows you to put input code anywhere in your project, rather than attaching it to the Vive Controllers. This script further allows you to expose the buttons to test different control schemes quickly, without significant new edits to your own code. This class functions similarly to the input classes for Oculus and Unity. It also provides controls for haptics and D-pad functionality, as well as calls for the vibration motor.

This package also works with Windows Mixed Reality when run in SteamVR mode.

Features:

- Script allows you to call input from any class in your project.
- Detect multi-state inputs from controllers - left, right, or both.
- Generate haptics - adjust power and duration on either/both of the controllers.
- Detect directional clicks on the Vive controllers' touchpads.

If you have any problems or questions please contact at: https://discord.gg/W3CVAPu

Please Visit 3lbgames.com for more Information on the other assets we provide.

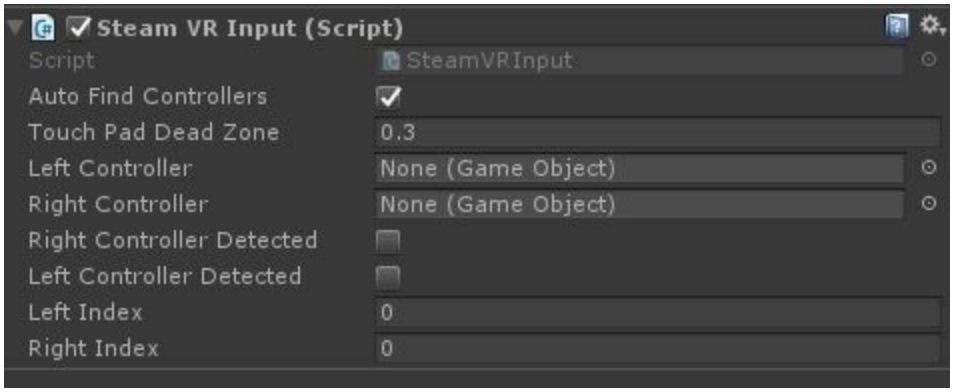
FAQ:

Q: Does this work with Oculus Touch?
A: No. This is designed to work only with the HTC Vive. However, it will work for someone using Oculus Touch in SteamVR.

Q: Do I need the SteamVRInput prefab in every scene?
A: Yes, you will call SteamVRInput in every scene.

SteamVRInput QuickStart

1) Drag the SteamVRInput prefab into your scene.
2) Check the autoFindControllers checkbox.
3) Code your inputs.



For example, to get the status of the Grip Button on the Left Controller, you call this line:

SteamVRInput.Get(SteamVRInput.Button.Grip,SteamVRInput.Controller.Left);

Other examples:

SteamVRInput.GetDown(SteamVRInput.Button.Grip,SteamVRInput.Controller.Left);
SteamVRInput.GetUp(SteamVRInput.Button.Trigger,SteamVRInput.Controller.Left);
*SteamVRInput.Get(SteamVRInput.Button.*TouchPadLeft*,SteamVRInput.Controller.Left);*

Getting axes:

Returns Vector2 from Trigger on Left Controller;
SteamVRInput.GetTriggerAxis(Controller.Left);

Returns Vector2 from TouchPad on Right Controller;
SteamVRInput.GetTouchPadAxis(Controller.Right);

SteamVRInput also can play a haptic pulse:

*PlayHapticPulse(**CONTROLLER**, **STRENGTH**);*

- Play a haptic pulse for 1 Frame at 50% power on Left Controller
  SteamVRInput.PlayHapticPulse(Controller.Left, 2000);

- Play a haptic pulse for 1 Frame at 50% power on Both Controllers
  SteamVRInput.PlayHapticPulse(Controller.Any, 2000);

Vive Input Documentation

Vive Input is designed to mimic the OVRInput by Oculus Rift. This allows you to easily acquire inputs for the left and the right controllers without having to guess which one you are using. SteamVRInput is a static class that can be called from any script, but requires the prefab in every single scene you are using SteamVR.

Steps to Get Started:

1) Drag the SteamVRInput prefab into your scene.
2) Check the autoFindControllers checkbox.
3) Code your inputs.

SteamVRInput Prefab Inspector Info

autoFindControllers:
Will grab left and right from the SteamVR_ControllerManager.

LeftController
RightController
Manually drag the controllers if you are not using autoFindControllers.

touchPadDeadZone
This is the deadzone for using Touchpad buttons to help with accidental taps.

RightControllerDetected
LeftControllerDetected
Visuals to ensure that your controllers are being detected by SteamVR

Vive Input follows a set pattern and exposed enums for controllers and buttons.

Examples:

- This will return true if the grip button is pressed down on the right controller.
  SteamVRInput.Get(SteamVRInput.Button.Trigger,SteamVRInput.Controller.Right);

- This will return true if the trigger button is pressed down on the right controller;
  SteamVRInput.Get(SteamVRInput.Button.Trigger,SteamVRInput.Controller.Right);

- This will return true if the trigger button is pressed down on either controller.
  SteamVRInput.Get(SteamVRInput.Button.Trigger,SteamVRInput.Controller.Any);

Available Static Functions

Button Functions:

SteamVRInput.Get(Button,Controller)
SteamVRInput.GetDown(Button,Controller)
SteamVRInput.GetUp(Button,Controller)

SteamVRInput.GetTouch(Button,Controller)
SteamVRInput.GetTouchDown(Button,Controller)
SteamVRInput.GetTouchUp(Button,Controller)

Axis Functions:

SteamVRInput.GetTriggerAxis(Controller,Clamped = True)
SteamVRInput.GetTouchPadAxis(Controller,Clamped = True)

Haptics Function:

SteamVRInput.HapticPulse(Controller,Strength);

Button List

SteamVRInput.Button.None
SteamVRInput.Button.Trigger
SteamVRInput.Button.Grip
SteamVRInput.Button.TouchPad
SteamVRInput.Button.ApplicationsMenu

SteamVRInput.Button.TouchPadUp
SteamVRInput.Button.TouchPadDown
SteamVRInput.Button.TouchPadLeft
SteamVRInput.Button.TouchPadRight

Vive Input also includes D-Pad emulation for the touch pad. You call these like you would any other button.

Note that GetTouchUp does NOT work with the TouchPad. GetUp has a known inconsistency due to the way the TouchPad returns zeros if you are not touching the pad.

Controller List

SteamVRInput.Controller.Left
SteamVRInput.Controller.Right
SteamVRInput.Controller.Any

Function Explanations

Gets and get touches work by returning a boolean, just like the Unity Input Class using SteamVRInput.Controller. *Any* will return true if the event happens on either controller.

SteamVRInput.Get(SteamVRInput.Button.Trigger,Controller.Right)
SteamVRInput.GetDown(SteamVRInput.Button.Trigger,Controller.Right)
SteamVRInput.GetUp(SteamVRInput.Button.Trigger,Controller.Right)

GetTouch only works with the Touch Pad

SteamVRInput.GetTouch(SteamVRInput.Button.TouchPad,Controller.Right)
SteamVRInput.GetTouchDown(SteamVRInput.Button.TouchPad,Controller.Right)
SteamVRInput.GetTouchUp(SteamVRInput.Button.TouchPad,Controller.Right)

GetTriggerAxis

Clamped Example:
This function will return a Vector2 containing the amount the trigger has been pressed.

```
SteamVRInput.GetTriggerAxis(Controller.Right);
```

Unclamped Example:
The optional argument clamped is used to determine wheter two buttons are pressed at once. It must be used with Controller.Any. If both triggers are down, it will give back values above 1.7F. If it is clamped it will only return up to 1.

```
SteamVRInput.GetTriggerAxis(Controller.Any,false);
```

GetTriggerAxis

Clamped Example:
This function will return a Vector2 containing the X/Y information of the finger on the touch pad.

SteamVRInput.GetTriggerAxis(Controller.Right);

Unclamped Example:
The optional argument clamped is used to combine the touch of both controllers together. It must be used with Controller.Any.

```
SteamVRInput.GetTriggerAxis(Controller.Any,false);
```

HapticPulse:

Haptic Pulse will use the vibration motor on whichever controller you set it to. The strength has a value that ranges from 0-3999. Any values outside that range will be clamped.

Single Controller Example:

```
SteamVRInput.HapticPulse(SteamVRInput.Controller.Right,2000);
```

Both Controller Example:

```
SteamVRInput.HapticPulse(SteamVRInput.Controller.Any,2000);
```