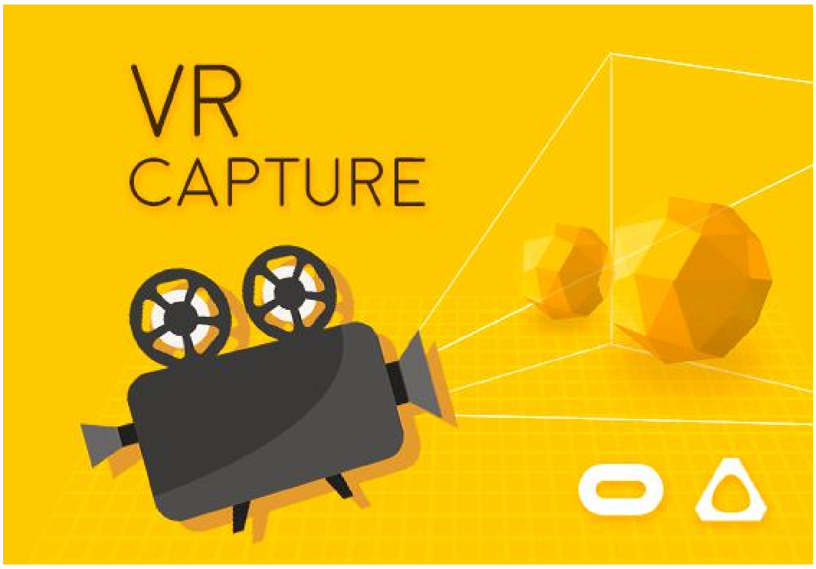


VR Capture Plugin

Rift Component

Version 1.1

Created by RockVR

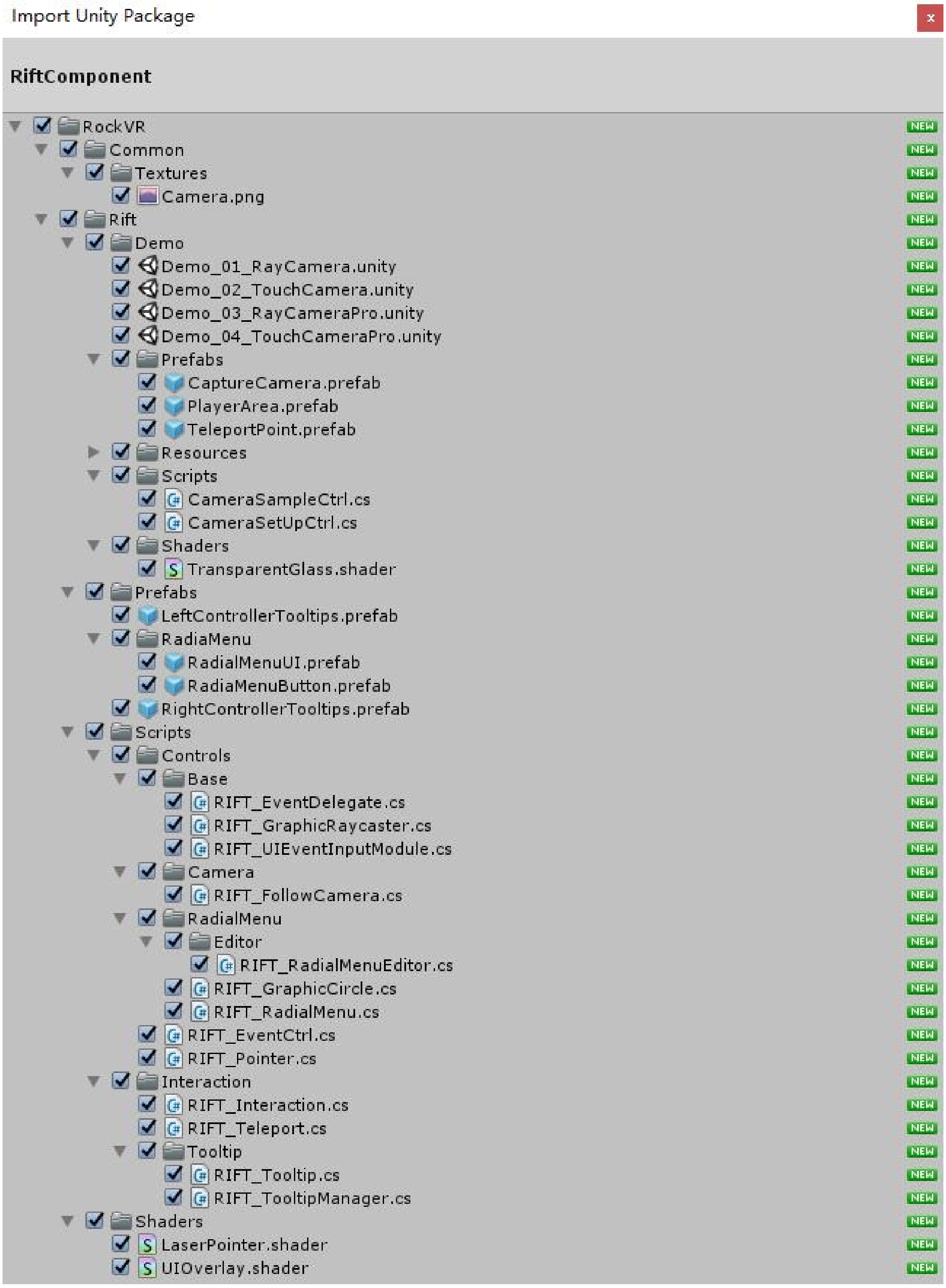
http://www.rockvr.com/vrcapture

Contact: dev@rockvr.com

1. Introduction and Overview

Rift Component is a unity package can help you to quickly understand how video capture work in VR scenes with Rift device.

When you import *Rift Component* into your Unity project, the following files will be added:



RiftComponent

```
▼ ☑ RockVR                                      NEW
    ▼ ☑ Common                                  NEW
        ▼ ☑ Textures                            NEW
              ☑ Camera.png                      NEW
    ▼ ☑ Rift                                    NEW
        ▼ ☑ Demo                                NEW
              ☑ Demo_01_RayCamera.unity         NEW
              ☑ Demo_02_TouchCamera.unity       NEW
              ☑ Demo_03_RayCameraPro.unity      NEW
              ☑ Demo_04_TouchCameraPro.unity    NEW
            ▼ ☑ Prefabs                         NEW
                  ☑ CaptureCamera.prefab        NEW
                  ☑ PlayerArea.prefab           NEW
                  ☑ TeleportPoint.prefab        NEW
            ▶ ☑ Resources                       NEW
            ▼ ☑ Scripts                         NEW
                  ☑ CameraSampleCtrl.cs         NEW
                  ☑ CameraSetUpCtrl.cs          NEW
            ▼ ☑ Shaders                         NEW
                  ☑ TransparentGlass.shader     NEW
        ▼ ☑ Prefabs                             NEW
              ☑ LeftControllerTooltips.prefab   NEW
            ▼ ☑ RadiaMenu                       NEW
                  ☑ RadialMenuUI.prefab         NEW
                  ☑ RadiaMenuButton.prefab      NEW
              ☑ RightControllerTooltips.prefab  NEW
        ▼ ☑ Scripts                             NEW
            ▼ ☑ Controls                        NEW
                ▼ ☑ Base                        NEW
                      ☑ RIFT_EventDelegate.cs   NEW
                      ☑ RIFT_GraphicRaycaster.cs  NEW
                      ☑ RIFT_UIEventInputModule.cs  NEW
                ▼ ☑ Camera                      NEW
                      ☑ RIFT_FollowCamera.cs    NEW
                ▼ ☑ RadialMenu                  NEW
                    ▼ ☑ Editor                  NEW
                          ☑ RIFT_RadialMenuEditor.cs  NEW
                      ☑ RIFT_GraphicCircle.cs   NEW
                      ☑ RIFT_RadialMenu.cs      NEW
                  ☑ RIFT_EventCtrl.cs           NEW
                  ☑ RIFT_Pointer.cs             NEW
            ▼ ☑ Interaction                     NEW
                  ☑ RIFT_Interaction.cs         NEW
                  ☑ RIFT_Teleport.cs            NEW
                ▼ ☑ Tooltip                     NEW
                      ☑ RIFT_Tooltip.cs         NEW
                      ☑ RIFT_TooltipManager.cs  NEW
        ▼ ☑ Shaders                             NEW
              ☑ LaserPointer.shader             NEW
              ☑ UIOverlay.shader                NEW
```

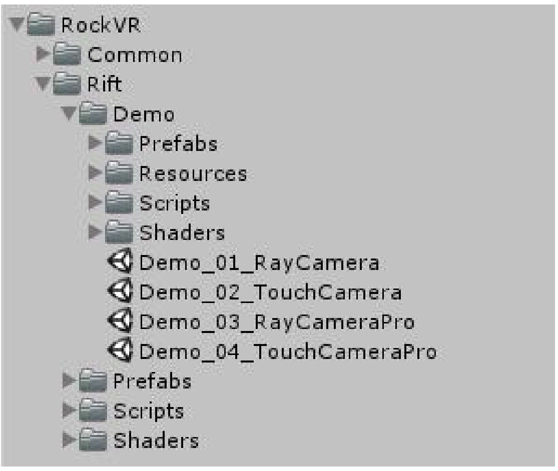
RockVR/Rift/Demo	Contains the scene file and all other assets for a fully functional demonstration of Video Capture with Rift interaction.
RockVR/Rift/ Prefabs	Contains useful prefabs can be dragged and dropped in to your scene.
RockVR/Rift/ Scripts	Contains the core Rift interaction logic scripts.
RockVR/Rift/ Shaders	Shaders work with materials for Rift interaction.

This guide covers integrating VR Capture to your own Unity project with Oculus Rift device, and provides a detailed explanation on how the package works under the hood.

If you have any questions, feedback or having issues, please contact us directly at dev@rockvr.com. We will respond to you as quickly as possible.

2. Demo Quick Start

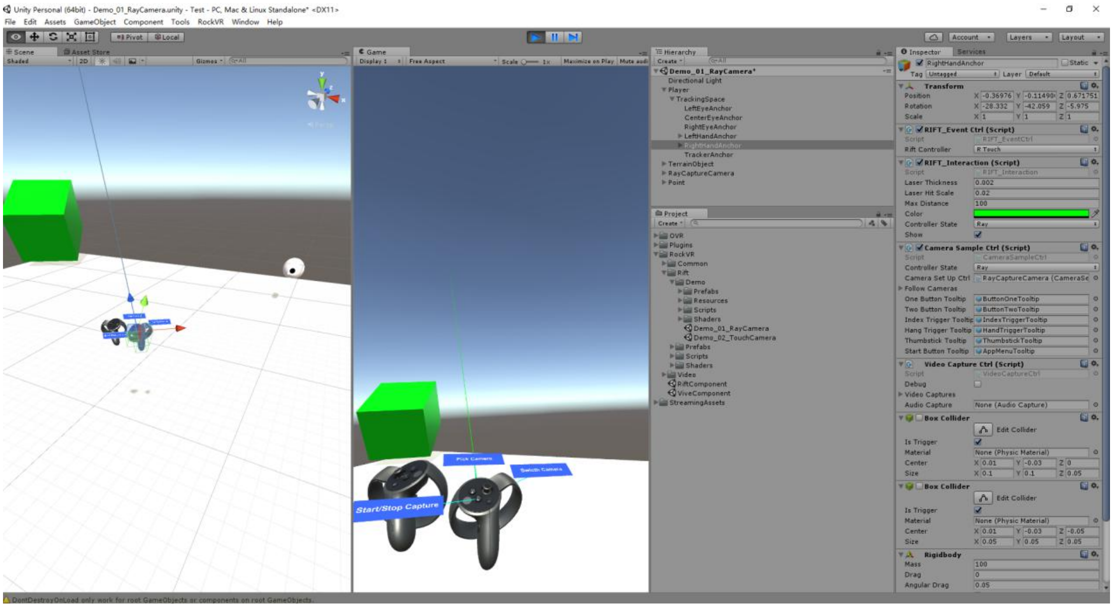
VR Capture *RiftComponent* come with several demos to help you understand functionality quickly. Start by importing Oculus plugin then importing all *RiftComponent* package assets included demo scenes files.



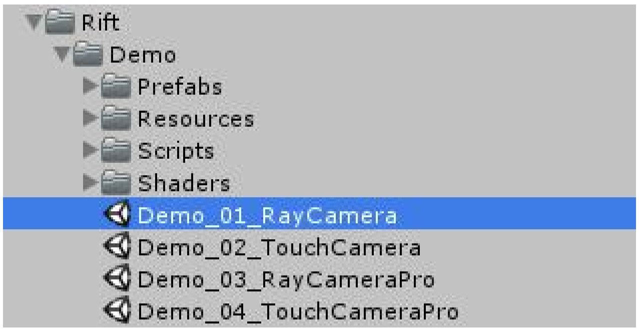
- RockVR
  - Common
  - Rift
    - Demo
      - Prefabs
      - Resources
      - Scripts
      - Shaders
      - Demo_01_RayCamera
      - Demo_02_TouchCamera
      - Demo_03_RayCameraPro
      - Demo_04_TouchCameraPro
    - Prefabs
    - Scripts
    - Shaders

We provide several ways of interaction with recording camera, including touch, laser ray controller, etc.

2.1 Laser Ray Controller Demo

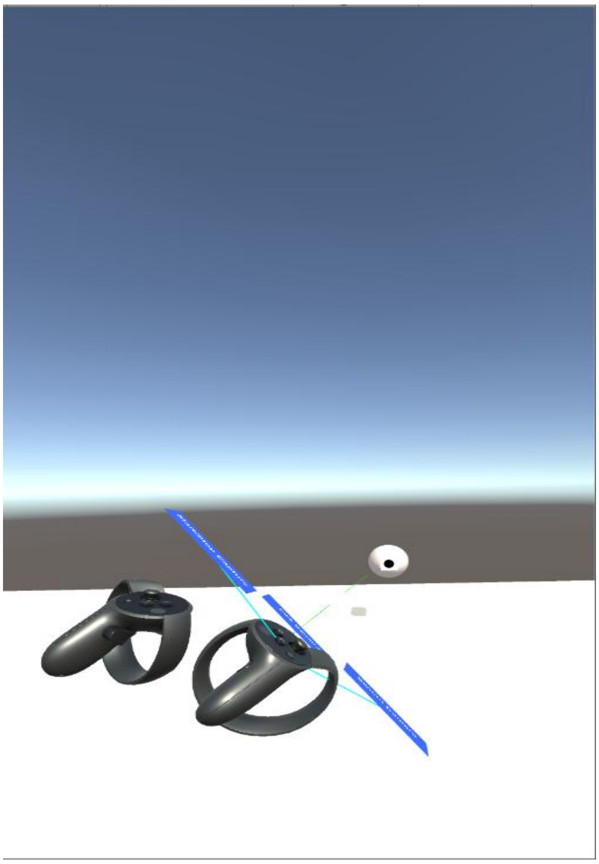

Unity Personal (64bit) - Demo_01_RayCamera.unity - Test - PC, Mac & Linux Standalone* <DX11>
File Edit Assets GameObject Component Tools RockVR Window Help
Scene
Asset Store
Game
Hierarchy
Project
Inspector
Services
Start/Stop Capture
Switch Camera
Transform
RIFT_Event Ctrl (Script)
RIFT_Interaction (Script)
Camera Sample Ctrl (Script)
Video Capture Ctrl (Script)
Box Collider
Rigidbody
DontDestroyOnLoad only work for root GameObjects or components on root GameObjects.

Step 1: Open the demo scene located in */RockVR/Rift/Demo/Demo_01_RayCamera* :

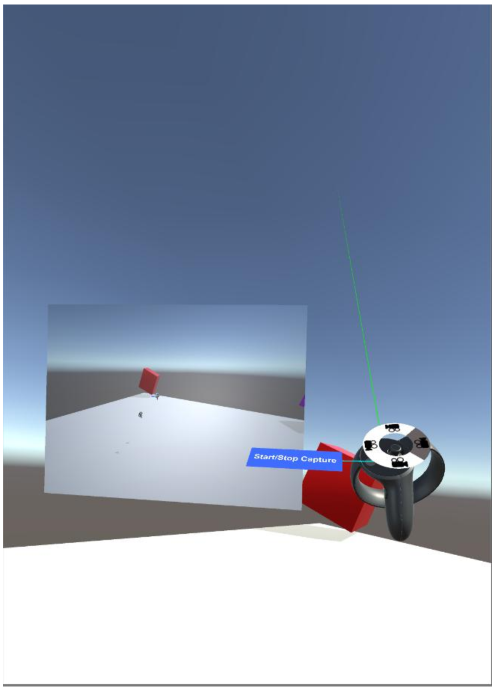


- Rift
  - Demo
    - Prefabs
    - Resources
    - Scripts
    - Shaders
    - Demo_01_RayCamera
    - Demo_02_TouchCamera
    - Demo_03_RayCameraPro
    - Demo_04_TouchCameraPro

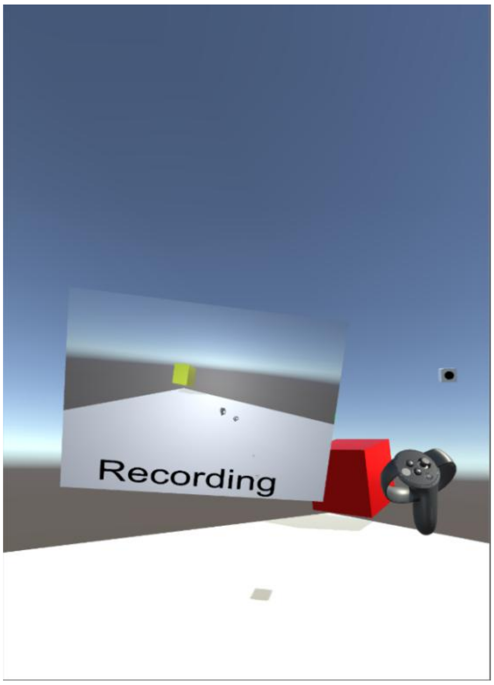
Step 2: Interact with recording camera. You need using laser ray to select the camera:



Step 3: Once you grab the camera, you can use radial menu to choose the shooting position with pre-set value:

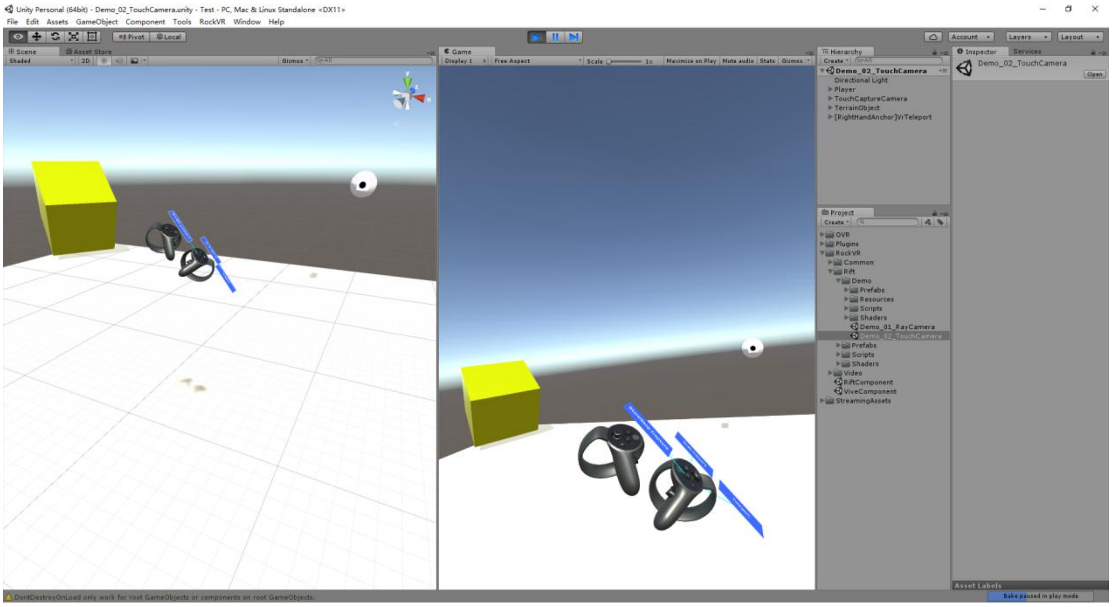


Step 4: After your perfect camera position set, press trigger to start video recording session:

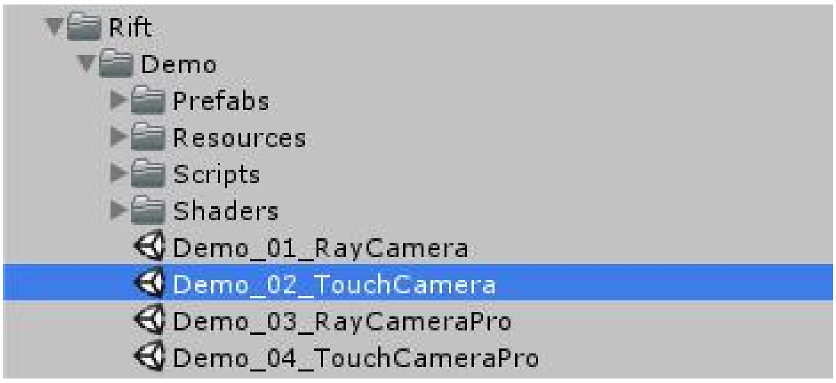


There are a few more demo you can try, *Demo_03_RayCameraPro* and *Demo_04_TouchCameraPro* both of them is recording by VideoCapturePro. The setup process should be same.

2.1 Touch Controller Demo

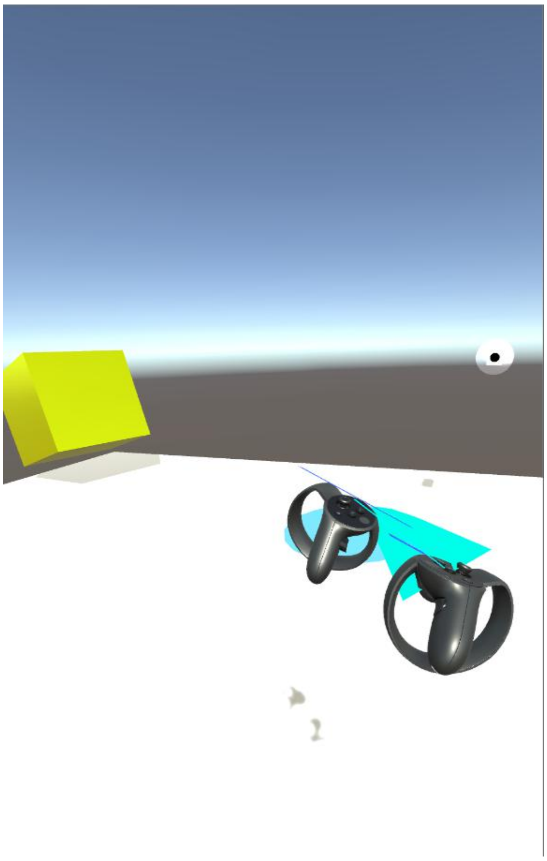


Step 1: Open the demo scene located in */RockVR/Rift/Demo/Demo_02_TouchCamera :*

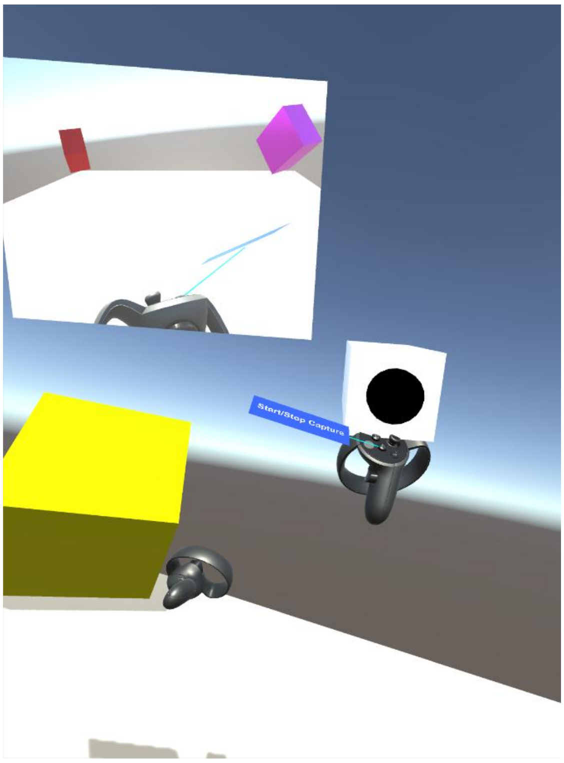


- Rift
  - Demo
    - Prefabs
    - Resources
    - Scripts
    - Shaders
    - Demo_01_RayCamera
    - Demo_02_TouchCamera
    - Demo_03_RayCameraPro
    - Demo_04_TouchCameraPro

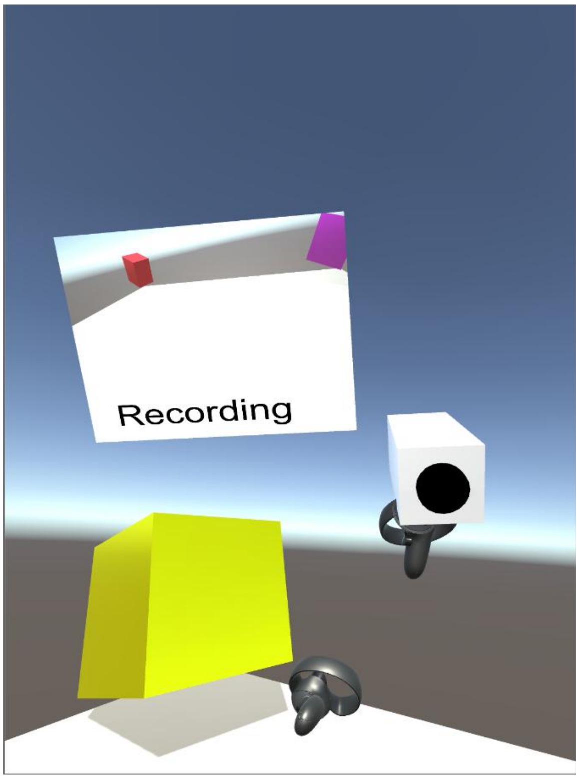
Step 2: Teleport to camera object by interaction with touchpad:



Step 3: Grab the camera and adjust to desired position and angle:



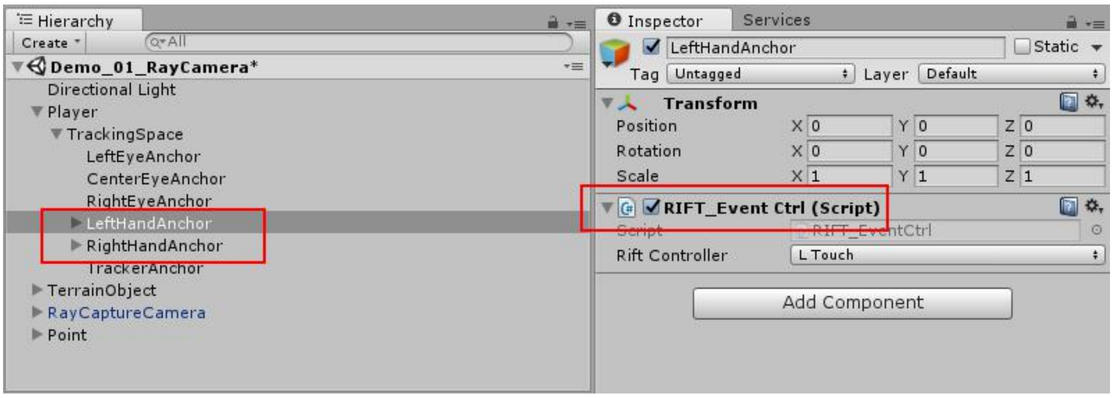
Step 4: Press the trigger to start video recording session, also you can grab camera and move it around while recording:



3. Integration Guide

RIFT_Interation module implemented the core function of VR interaction control.

Step 1: Attach *RIFT_EventCtrl* script to the rift devices controller in the scene you want to control.



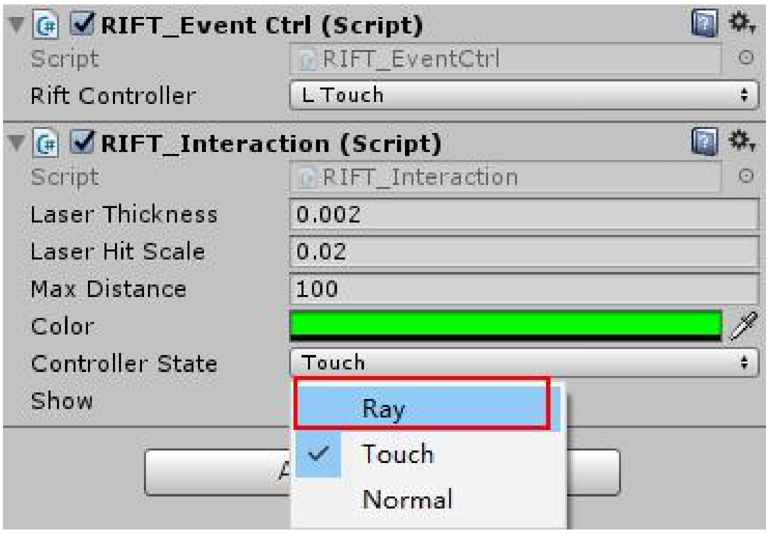
```
▼ Demo_01_RayCamera*
    Directional Light
  ▼ Player
    ▼ TrackingSpace
        LeftEyeAnchor
        CenterEyeAnchor
        RightEyeAnchor
      ▶ LeftHandAnchor
      ▶ RightHandAnchor
        TrackerAnchor
  ▶ TerrainObject
  ▶ RayCaptureCamera
  ▶ Point
```

```
☑ LeftHandAnchor                      □ Static
Tag  Untagged          Layer  Default
▼ Transform
Position     X 0     Y 0     Z 0
Rotation     X 0     Y 0     Z 0
Scale        X 1     Y 1     Z 1
▼ ☑ RIFT_Event Ctrl (Script)
Script            RIFT_EventCtrl
Rift Controller   L Touch
```

Note: *RIFT_EventCtrl* is based on the *RIFT_EventDelegate* and Oculus plugin, it is an event script to control the rift handle devices.

Step 2: Adding different interactive features of the script to the rift devices after the first step.

RIFT_Interaction - Set the interaction patterns, choose Ray or Touch interaction type.



☑ RIFT_Event Ctrl (Script)

Script: RIFT_EventCtrl

Rift Controller: L Touch

☑ RIFT_Interaction (Script)

Script: RIFT_Interaction

Laser Thickness: 0.002

Laser Hit Scale: 0.02

Max Distance: 100

Color: 🟩

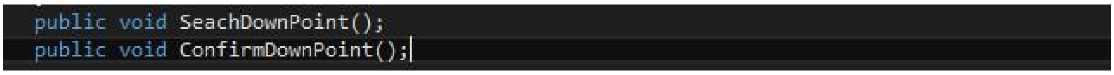
Controller State: Touch

Show

RIFT_Pointer - This script encapsulates system event to take response for user interaction.

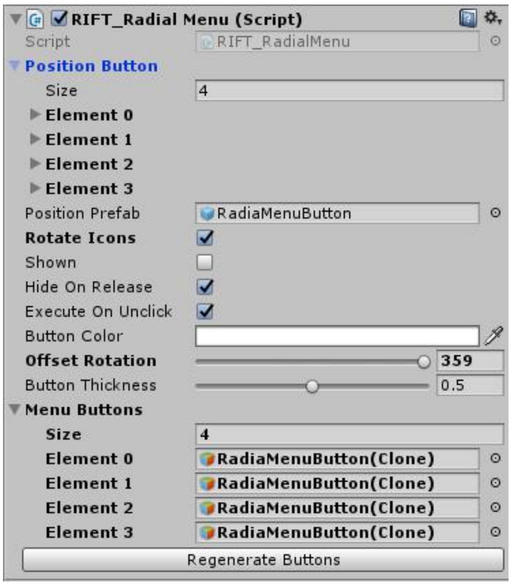
RIFT_UIEventInputModule is designed to work as you would expect how a Rift controller input to work. Including button presses, dragging, and touch events.

RIFT_Teleport - Implemented functionality of teleport in VR scene. Enable *SeachDownPoint* and *ConfirmDownPoint* function to implement teleport.



```
public void SeachDownPoint();
public void ConfirmDownPoint();
```

RIFT_RadialMenu - Set radial menu attribute, and binding keys to event listener.



☑ RIFT_Radial Menu (Script)

Script: RIFT_RadialMenu

Position Button

Size: 4

Element 0

Element 1

Element 2

Element 3

Position Prefab: RadiaMenuButton

Rotate Icons: ☑

Shown: □

Hide On Release: ☑

Execute On Unclick: ☑

Button Color:

Offset Rotation: 359

Button Thickness: 0.5

Menu Buttons

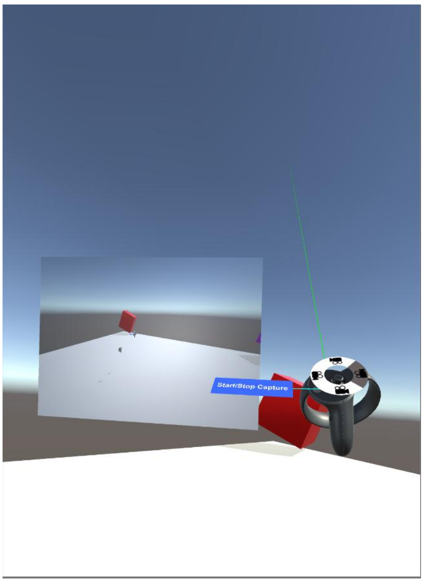
Size: 4

Element 0: RadiaMenuButton(Clone)

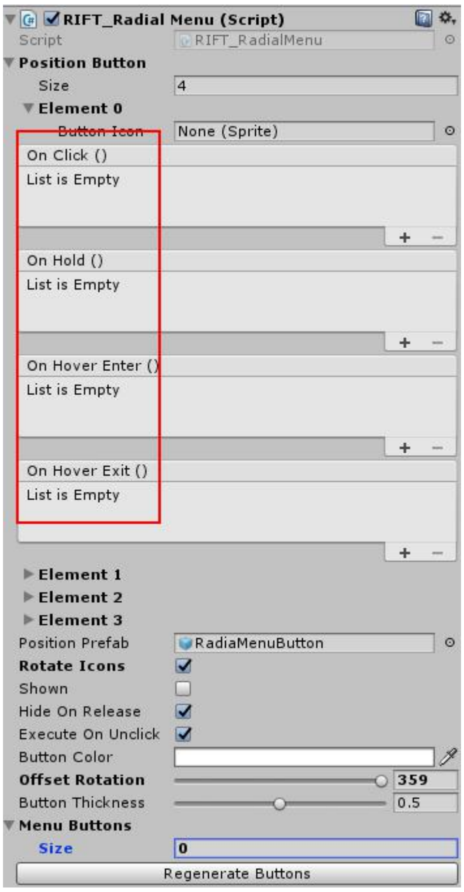
Element 1: RadiaMenuButton(Clone)

Element 2: RadiaMenuButton(Clone)

Element 3: RadiaMenuButton(Clone)



Add different events to objects corresponding to *RadialMenuButtons* state.



☑ RIFT_Radial Menu (Script)

Script: RIFT_RadialMenu

Position Button

Size: 4

Element 0

Button Icon: None (Sprite)

On Click ()

List is Empty

On Hold ()

List is Empty

On Hover Enter ()

List is Empty

On Hover Exit ()

List is Empty

Element 1

Element 2

Element 3

Position Prefab: RadiaMenuButton

Rotate Icons: ☑

Shown: □

Hide On Release: ☑

Execute On Unclick: ☑

Button Color:

Offset Rotation: 359

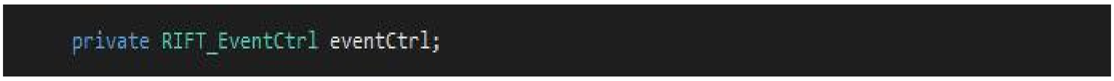
Button Thickness: 0.5

Menu Buttons

Size: 0

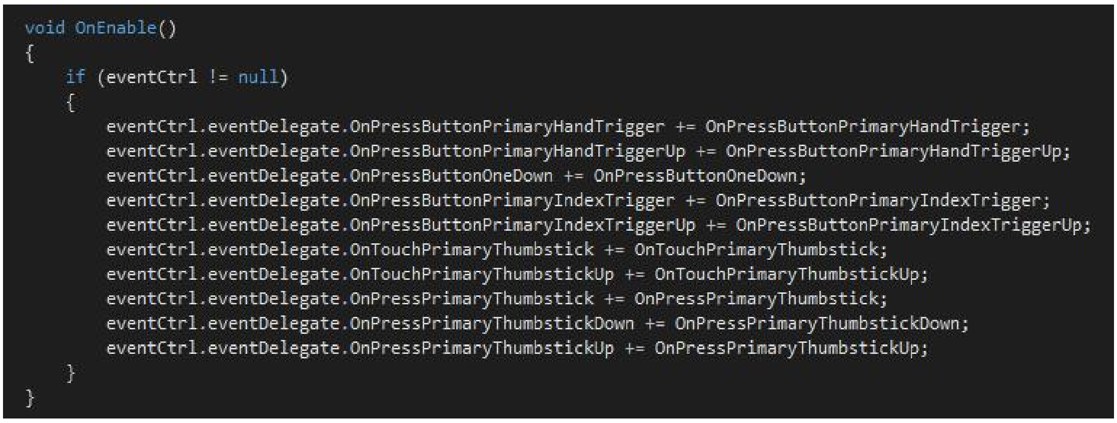
Step 3: Create a control management script to manage the handle events.

Create *RIFT_EventCtrl* object.



```
    private RIFT_EventCtrl eventCtrl;
```

Register to create a delegated events.



```
void OnEnable()
{
    if (eventCtrl != null)
    {
        eventCtrl.eventDelegate.OnPressButtonPrimaryHandTrigger += OnPressButtonPrimaryHandTrigger;
        eventCtrl.eventDelegate.OnPressButtonPrimaryHandTriggerUp += OnPressButtonPrimaryHandTriggerUp;
        eventCtrl.eventDelegate.OnPressButtonOneDown += OnPressButtonOneDown;
        eventCtrl.eventDelegate.OnPressButtonPrimaryIndexTrigger += OnPressButtonPrimaryIndexTrigger;
        eventCtrl.eventDelegate.OnPressButtonPrimaryIndexTriggerUp += OnPressButtonPrimaryIndexTriggerUp;
        eventCtrl.eventDelegate.OnTouchPrimaryThumbstick += OnTouchPrimaryThumbstick;
        eventCtrl.eventDelegate.OnTouchPrimaryThumbstickUp += OnTouchPrimaryThumbstickUp;
        eventCtrl.eventDelegate.OnPressPrimaryThumbstick += OnPressPrimaryThumbstick;
        eventCtrl.eventDelegate.OnPressPrimaryThumbstickDown += OnPressPrimaryThumbstickDown;
        eventCtrl.eventDelegate.OnPressPrimaryThumbstickUp += OnPressPrimaryThumbstickUp;
    }
}
```

4. Feedback

If you have any feedback to *VR Capture* plugin, please email us directly, your suggestion will be very valuable to us. If you plan integrate a plugin into your game, please contact us by *dev@rockvr.com* and we will provide more help to let you share your awesome game more efficient.