The Lab Renderer for Unity

Contents

Introduction

The Lab Renderer is what Valve used in all of the Unity-created experiences in The Lab (everything except Robot Repair). We are shipping this for free to Unity developers as an example of how we think current generation VR rendering can provide the highest fidelity experience with the best performance.

This isn't a complete rendering solution that is meant to replace Unity's extensive rendering features, but it does provide all of the features Valve used to ship The Lab. We are providing all source code and shader code so developers can modify it to best fit their needs.

Below is the list of features this renderer provides above the standard Unity renderer:

Single-Pass Forward Rendering and MSAA

Forward rendering is critical for VR, because it enables applications to use MSAA which is the best known anti-aliasing method for VR. Deferred renderers suffer from aliasing issues due to the current state of the art in image-space anti-aliasing algorithms.

Unity's default forward renderer is multi-pass where geometry is rendered an additional time for each runtime spotlight or point light that lights each object. The Lab's renderer supports up to 18 dynamic, shadowing lights in a single forward pass.

Adaptive quality

Valve presented the Adaptive Quality algorithm at GDC 2016 in a presentation by Alex Vlachos, "Advanced VR Rendering Performance". Slides and video of that presentation are available free on the GDC Vault: http://www.gdcvault.com/play/1023522/Advanced-VR-Rendering

Adaptive Quality is a method for dynamically changing rendering fidelity to maintain framerate without having to rely on reprojection methods that can cause judder and other visual artifacts. The parameters to the Adaptive Quality algorithm are fully customizable per map.

Custom Shaders

The plugin requires that all materials use The Lab's shaders. If you have a mixture of Unity shaders and The Lab shaders, Unity will render shadows multiple times, which will cost perf. There are menu options under Valve->ShaderDev to help you convert your existing materials to use The Lab's shaders. We recommend backing up your project before running any of those helper commands.

We are shipping full shader source code, so you can customize the shaders to meet your product's needs.

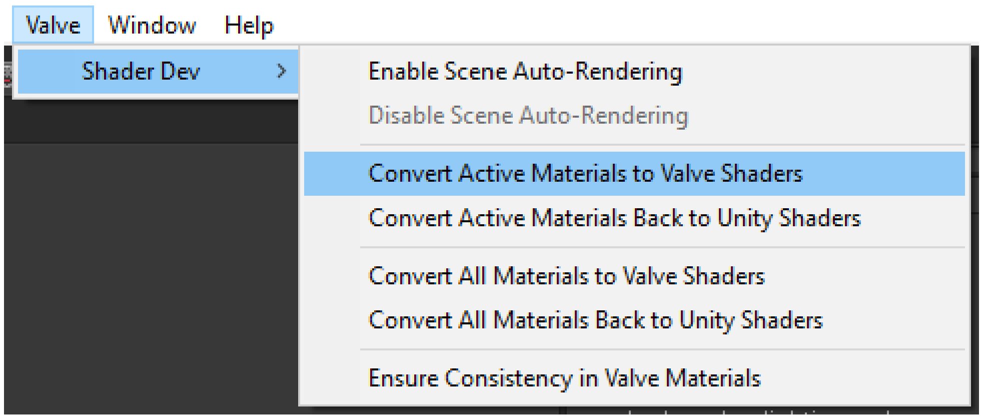
GPU Flushing

One of the least-known performance tricks in VR rendering is to flush the rendering API at a granularity that ensures the GPU is fed often enough to keep busy and avoid bubbles. This plugin calls Unity's GL.Flush() in the main camera's OnPostRender() and after shadow rendering for a total of up to 3 times per frame. This is critical for ensuring the GPU is receiving the draw calls that are being queued in the DirectX runtime in a timely manner.

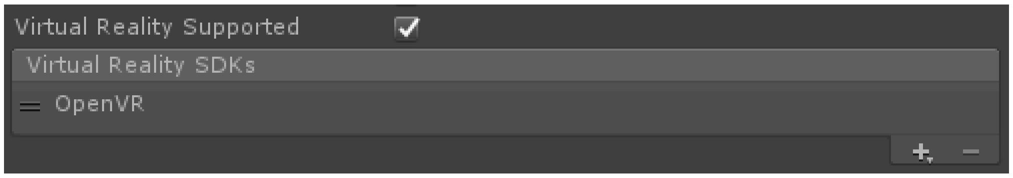
Quick Start Guide

Using The Lab Renderer can be as simple as making these changes to your existing project:

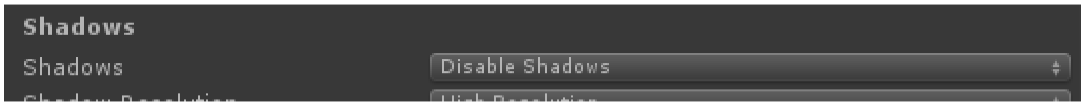
1. Add the ValveCamera.cs script to your scene camera
2. Add the ValveRealtimeLight.cs script to each dynamic, realtime light
3. Convert all materials to use Valve's shaders by executing one of the commands in the menu Valve->ShaderDev like "Convert All Materials to Valve Shaders". This will map all materials to one of the shaders under the Valve subfolder. The primary shader is Valve/vr_standard.


Valve
Window
Help
Shader Dev
Enable Scene Auto-Rendering
Disable Scene Auto-Rendering
Convert Active Materials to Valve Shaders
Convert Active Materials Back to Unity Shaders
Convert All Materials to Valve Shaders
Convert All Materials Back to Unity Shaders
Ensure Consistency in Valve Materials

4. Make OpenVR the first and only Virtual Reality SDK in Player Settings. NOTE: The Adaptive Quality feature requires OpenVR to drive the logic appropriately.


Virtual Reality Supported ✓
Virtual Reality SDKs
OpenVR

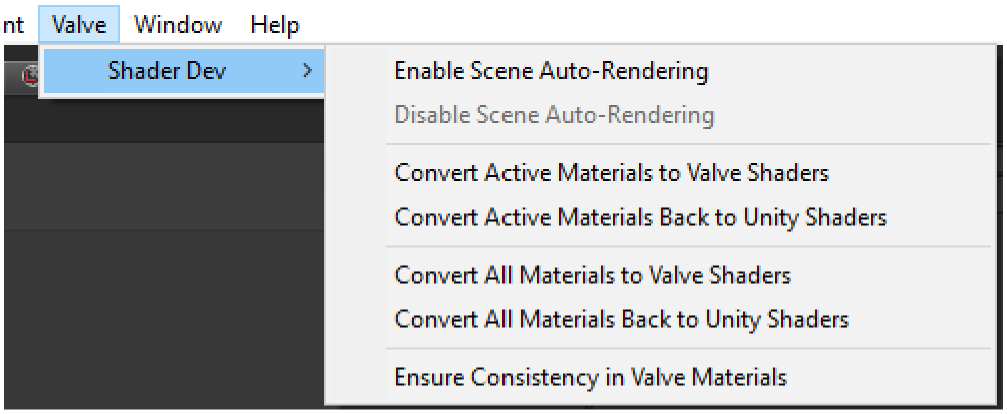
5. Disable Unity's realtime shadows by setting "Shadows" to "Disable Shadows" in the Quality->Shadows settings


Shadows
Shadows: Disable Shadows

For most simple scenes, that is sufficient to make full use of the rendering plugin. The ValveCamera.cs script and the ValveRealtimeLight.cs script both have many custom settings available in the inspector.

Elements

Valve Menu Items


nt
Valve
Window
Help
Shader Dev
Enable Scene Auto-Rendering
Disable Scene Auto-Rendering
Convert Active Materials to Valve Shaders
Convert Active Materials Back to Unity Shaders
Convert All Materials to Valve Shaders
Convert All Materials Back to Unity Shaders
Ensure Consistency in Valve Materials

The Valve->ShaderDev submenu has several useful options.

Enable & Disable Scene Auto-Rendering

This will force Unity's editor scene window to continually render. Normally Unity's scene window only renders when something changes to prevent Unity from using unnecessary GPU cycles, but there are times you want that view to render continuously. One example of when this is useful if you are working on an animated shader effect that uses time as an input. This will allow you to preview that shader code without having to press the play button.

Convert Active Materials to/from Valve or Unity Shaders

This will convert all loaded materials in the current scene to/from Valve shaders.

NOTE: Not all settings will convert perfectly since the shaders in this renderer are different than the standard shaders, but the inputs are nearly identical. You may need to tweak some material settings and some textures relating mostly to specular.

Convert All Materials to/from Valve or Unity Shaders

This will convert all materials in your project to/from Valve shaders.

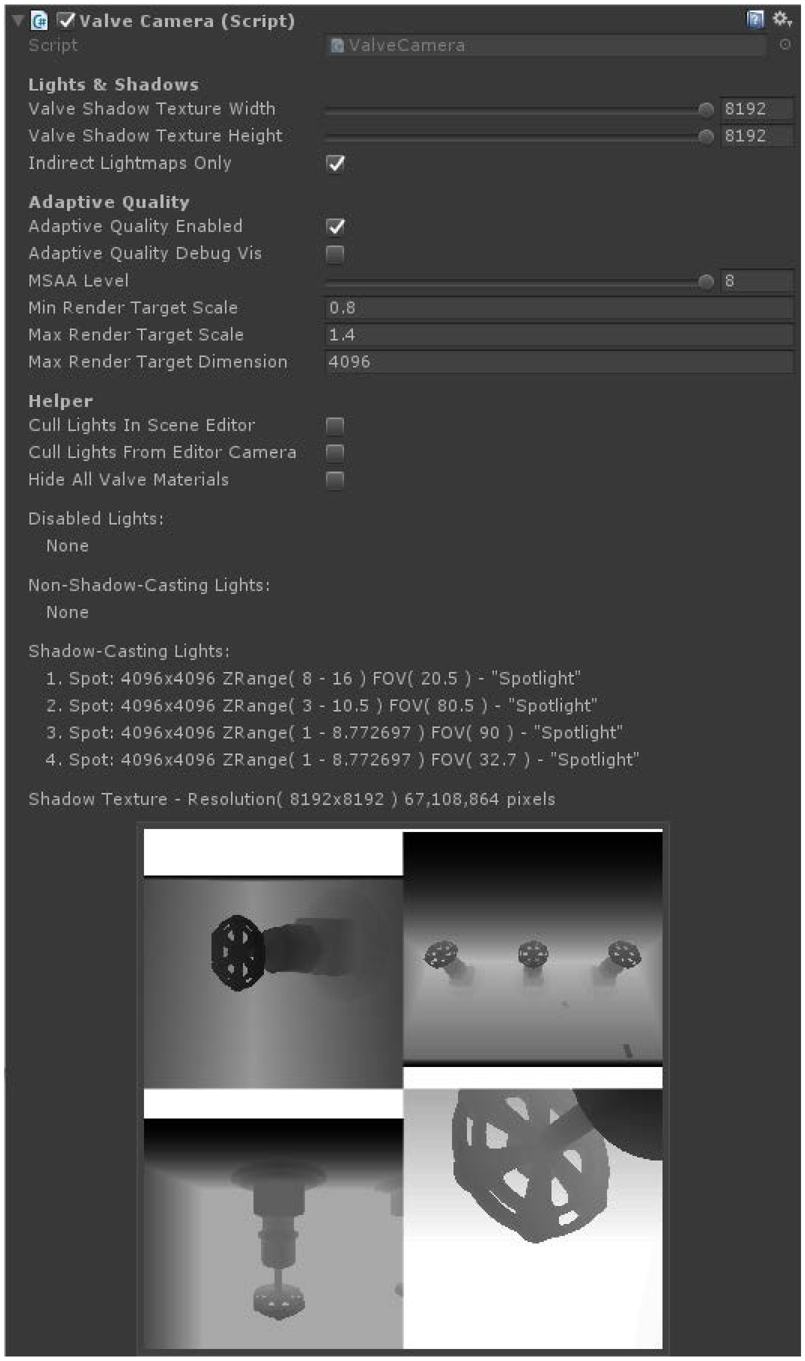
NOTE: Not all settings will convert perfectly since the shaders in this renderer are different than the standard shaders, but the inputs are nearly identical. You may need to tweak some material settings and some textures relating mostly to specular.

NOTE: If you have 3rd party plugins that you don't want modified, don't use this option. Instead, use the "Convert Active Materials" options as described above.

Ensure Consistency in Valve Materials

This is generally useful when downloading an update to this plugin where some material or shader settings may have changed. This can be manually run to ensure all settings on materials using a Valve shader are setup correctly. You shouldn't need to run this if you just converted materials using one of the above methods or created new materials.

Valve Camera Component



Valve Camera (Script)

Script ValveCamera

Lights & Shadows

Valve Shadow Texture Width	8192
Valve Shadow Texture Height	8192
Indirect Lightmaps Only	✓

Adaptive Quality

Adaptive Quality Enabled	✓
Adaptive Quality Debug Vis	☐
MSAA Level	8
Min Render Target Scale	0.8
Max Render Target Scale	1.4
Max Render Target Dimension	4096

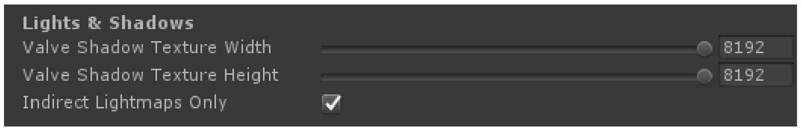
Helper

Cull Lights In Scene Editor	☐
Cull Lights From Editor Camera	☐
Hide All Valve Materials	☐

```
Disabled Lights:
   None

Non-Shadow-Casting Lights:
   None

Shadow-Casting Lights:
   1. Spot: 4096x4096 ZRange( 8 - 16 ) FOV( 20.5 ) - "Spotlight"
   2. Spot: 4096x4096 ZRange( 3 - 10.5 ) FOV( 80.5 ) - "Spotlight"
   3. Spot: 4096x4096 ZRange( 1 - 8.772697 ) FOV( 90 ) - "Spotlight"
   4. Spot: 4096x4096 ZRange( 1 - 8.772697 ) FOV( 32.7 ) - "Spotlight"

Shadow Texture - Resolution( 8192x8192 ) 67,108,864 pixels
```

The Valve Camera component should be added to the main scene camera.

Lights & Shadows

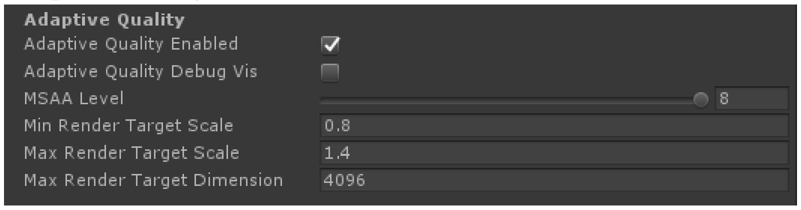


Lights & Shadows

Valve Shadow Texture Width	8192
Valve Shadow Texture Height	8192
Indirect Lightmaps Only	✓

Valve Shadow Texture Width & Height
The size of the scene's global shadow map. This is a shared shadow texture that all lights will render into. We recommend setting this to the largest size you need, because if you are only using a small portion of this shadow texture, you will be paying the cost of the GPU clearing the entire texture every frame.

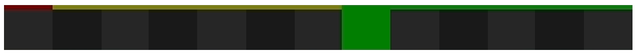
Indirect Lightmaps Only
Only use the indirect portion of baked lightmaps. This allows the Valve Realtime Lights to handle all direct illumination. This options requires that lightmaps be baked in "Directional Specular" mode in Unity's Lighting tab which causes both direct and indirect light to be stored separately in the lightmaps.

Adaptive Quality

Adaptive Quality

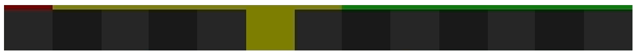
Adaptive Quality Enabled	✓
Adaptive Quality Debug Vis	
MSAA Level	8
Min Render Target Scale	0.8
Max Render Target Scale	1.4
Max Render Target Dimension	4096

Adaptive Quality Enabled
Check this box to enable the Adaptive Quality system. If unchecked, the renderer will render at the recommended resolution provided by OpenVR.

Adaptive Quality Debug Vis
This is for development only. It enables an overlay in VR to show the current quality level. If unchecked, you can enable this dynamically at any point in VR by pressing Shift+F1 in game. The visualization looks like this:



Each square represents a different level on the quality scale. Levels increase from left to right, and the first green box that is lit above represents the recommended render target resolution provided by OpenVR. The yellow boxes represent resolutions below the recommended render target resolution as shown below:



When the red box all the way on the left is lit, the user is likely seeing reprojection in the HMD since the application isn't maintaining framerate.

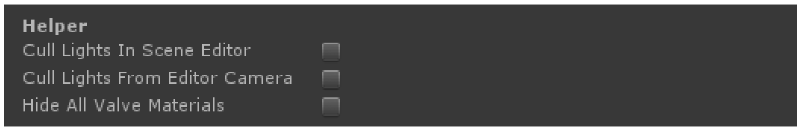
MSAA Level
Valid settings are 0, 2, 4, and 8. We recommend using only 4 or 8 for VR applications.

Min & Max Render Target Scale
Sets a scalar range that is applied to the recommended render target resolution by OpenVR that the Adaptive Quality algorithm scales within.

Max Render Target Dimension

This puts an upper limit on the size of the render target regardless of the max render target scale set above.

Helper



Helper

Cull Lights In Scene Editor ☐

Cull Lights From Editor Camera ☐

Hide All Valve Materials ☐

Cull Lights in Scene Editor

In Unity's editor scene view, this enables a smoother workflow for scenes with a large number of real-time lights. This will cause lights to be culled based on the current view in the scene window.

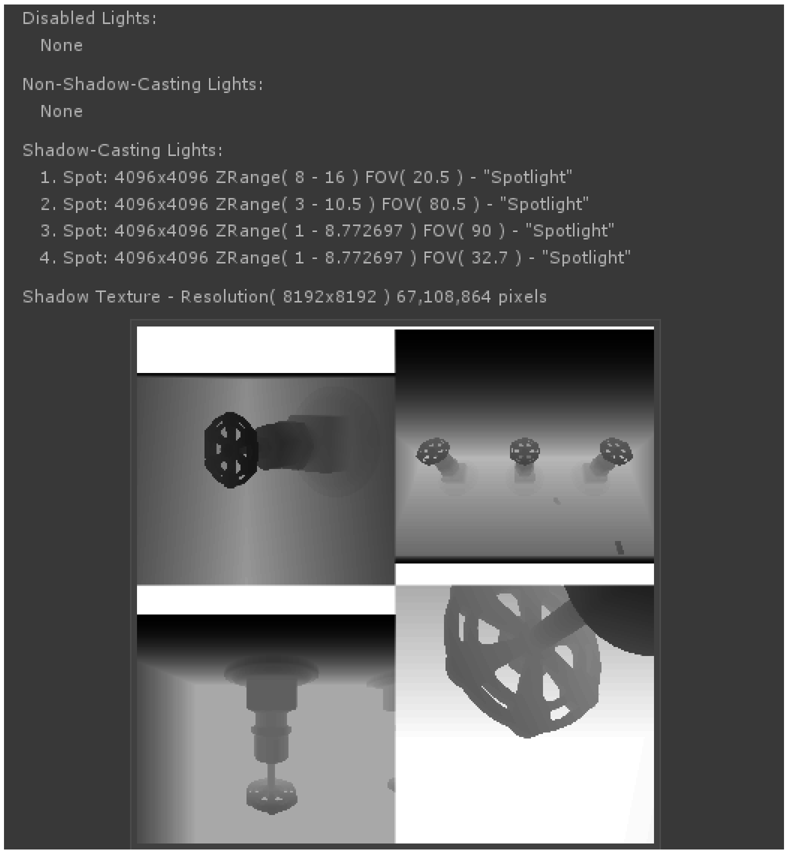
Cull Lights from Editor Camera

In Unity's editor scene view, this enables a smoother workflow for scenes with a large number of real-time lights. This will cause lights to be culled based on the frustum of the camera that has the ValveCamera.cs script attached.

Hide All Valve Materials

Useful for identifying materials that may need to be converted to the Valve/vr_standard shader.

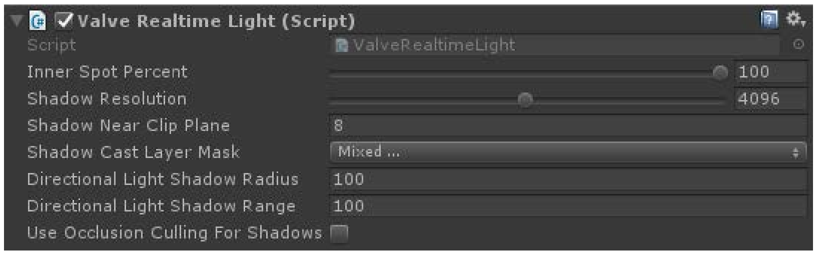
Debug Information



```
Disabled Lights:
   None

Non-Shadow-Casting Lights:
   None

Shadow-Casting Lights:
   1. Spot: 4096x4096 ZRange( 8 - 16 ) FOV( 20.5 ) - "Spotlight"
   2. Spot: 4096x4096 ZRange( 3 - 10.5 ) FOV( 80.5 ) - "Spotlight"
   3. Spot: 4096x4096 ZRange( 1 - 8.772697 ) FOV( 90 ) - "Spotlight"
   4. Spot: 4096x4096 ZRange( 1 - 8.772697 ) FOV( 32.7 ) - "Spotlight"

Shadow Texture - Resolution( 8192x8192 ) 67,108,864 pixels
```

This data updates dynamically to show the number of disabled, non-shadow-casting, and shadow-casting lights every frame. We recommend keeping this panel hidden when viewing your application in VR since it can significantly impact performance, but it can be a very useful debugging tool for understanding the number of lights and shadows being rendered each frame.

Valve Realtime Light Component



Valve Realtime Light (Script)

Script	ValveRealtimeLight
Inner Spot Percent	100
Shadow Resolution	4096
Shadow Near Clip Plane	8
Shadow Cast Layer Mask	Mixed ...
Directional Light Shadow Radius	100
Directional Light Shadow Range	100
Use Occlusion Culling For Shadows	☐

Inner Spot Percent

Allows you to customize the softness of the edge on spotlights.

Shadow Resolution

Squared resolution of the shadow map for this light. The shadow map resolution for the entire scene is specified in the Valve Camera Component. All Valve Realtime Lights' shadow resolutions, combined, must fit in the scene's global shadow map. If they don't, the script will throw an error in the console.

Shadow Near Clip Plane

The distance from the light source that shadows will start to cast. Objects closer than this to the light will not cast shadows.

Shadow Caster Layer Mask

The mechanism by which you tell objects NOT to cast shadows for that light.

Directional Light Shadow Radius

Only used for directional lights. Directional lights can cast shadows from the forward vector of the light with some world-space square radius around that vector. Objects outside this area will be unshadowed by the directional light.

Directional Light Shadow Range

Only used for directional lights. This is effectively the far clip plane for shadow rendering. Objects farther than this from the light source will not cast shadows.

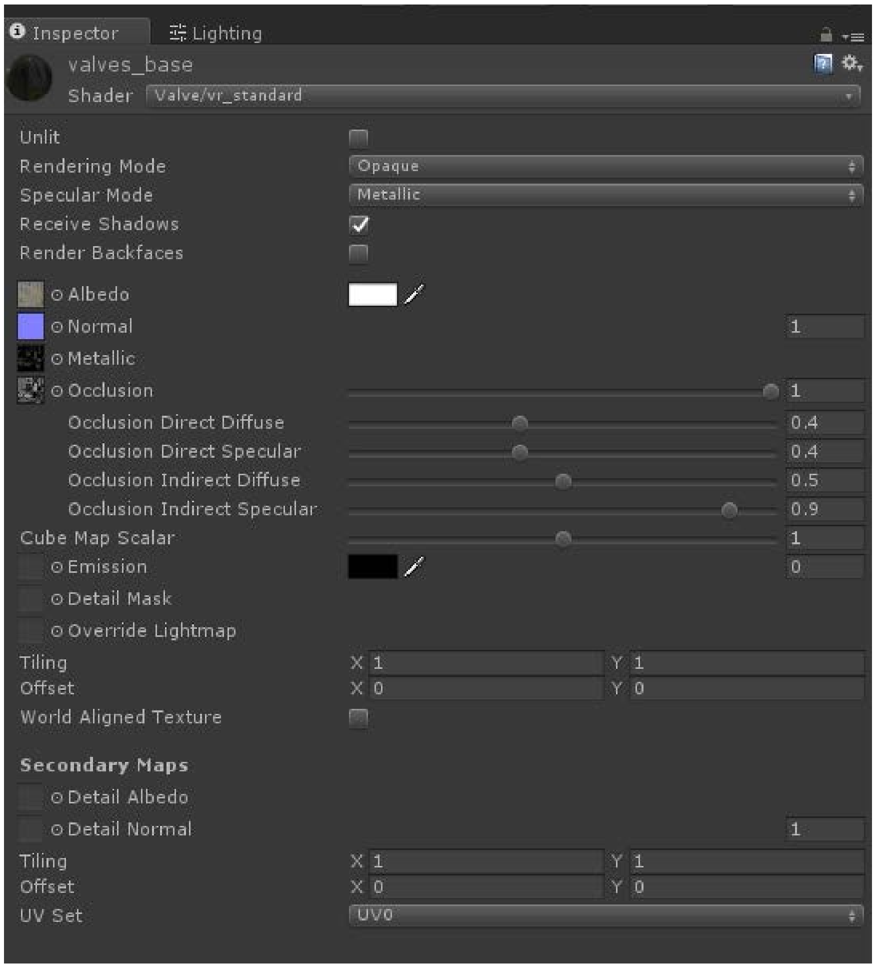
Use Occlusion Culling For Shadows

In scenes with complex baked occlusion, real-time shadows may actually be faster with shadow occlusion culling turned off.

Directional Lights - Known Issue

In the latest Unity 5.4 beta, there is an issue with shadow-casting directional lights where Unity will render its shadows even though there are no materials using their shadow buffer. You can disable shadows in the Quality->Shadows settings to prevent Unity from rendering shadows for directional lights.

Valve VR Standard Shader



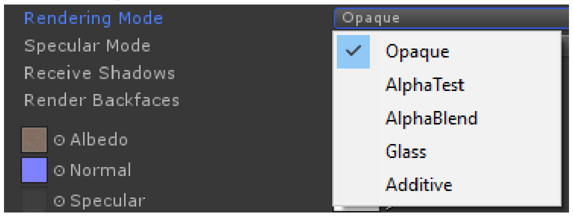
```
Inspector    Lighting
valves_base
Shader  Valve/vr_standard

Unlit                          ☐
Rendering Mode                 Opaque
Specular Mode                  Metallic
Receive Shadows                ✓
Render Backfaces               ☐

Albedo
Normal                                          1
Metallic
Occlusion                                       1
  Occlusion Direct Diffuse                      0.4
  Occlusion Direct Specular                     0.4
  Occlusion Indirect Diffuse                    0.5
  Occlusion Indirect Specular                   0.9
Cube Map Scalar                                 1
Emission                                        0
Detail Mask
Override Lightmap
Tiling                         X 1          Y 1
Offset                         X 0          Y 0
World Aligned Texture          ☐

Secondary Maps
Detail Albedo
Detail Normal                                   1
Tiling                         X 1          Y 1
Offset                         X 0          Y 0
UV Set                         UV0
```

The workflow for the Valve/vr_standard shader is almost identical to that of Unity's default standard shaders. The Valve/vr_standard shader aims to replace this set of built-in shaders:

1. Standard
2. Standard (Specular Setup)
3. Unlit->Color
4. Unlit->Texture
5. Unlit->Transparent
6. Unlit->Transparent Cutout

There are a few extra features worth discussing:

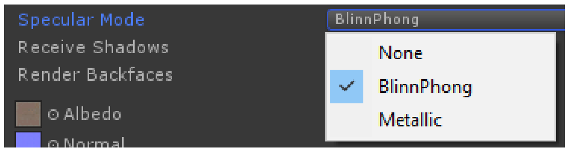
Rendering Mode



```
Rendering Mode        Opaque
Specular Mode           ✓ Opaque
Receive Shadows           AlphaTest
Render Backfaces          AlphaBlend
Albedo                    Glass
Normal                    Additive
Specular
```

The first 4 modes are just renamed modes available in Unity's Standard shaders. The last option, Additive, is an additional blend mode that uses a one/one additive blend.

Specular Mode



Along with the standard Blinn-Phong and Metallic specular modes, you also have the option of omitting specular altogether by selecting None. This can improve performance for surfaces with no visible specular term.

Cube Map Scalar

Per-material control over cube map contribution. This allows artists to tweak the strength of the environment map for a given material.

Override Lightmap

See Valve Override Lightmap Component below.

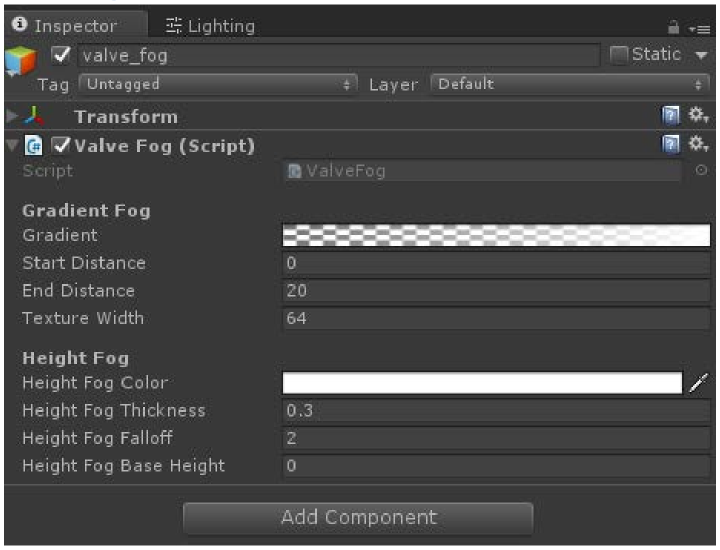
World Aligned Texture

Use this feature to tile a texture across surfaces in world space instead of using the mesh's UVs.

- Size – Size of the texture in meters.
- Normal – Direction perpendicular to the texture plane. Use 0,1,0 to project from above.
- World Position – World space offset for the texture coordinates.

Utility and Helper Elements

Valve Fog Component



valve_fog

Tag: Untagged   Layer: Default

Valve Fog (Script)

Script: ValveFog

Gradient Fog

Gradient

Start Distance: 0

End Distance: 20

Texture Width: 64

Height Fog

Height Fog Color

Height Fog Thickness: 0.3

Height Fog Falloff: 2

Height Fog Base Height: 0

Unity's normal fog won't work with The Lab Renderer, so if you need fog, simply add this component to a single active-and-enabled gameobject in your scene. Both gradient (distance) fog and height fog are supported.

Valve Override Lightmap Component

Materials that use the Valve/vr_standard shader can use texture assets as lightmap overrides. The texture simply needs to be hooked up in the material and the Valve Override Lightmap component needs to be added to the gameobject that holds the Mesh Renderer.

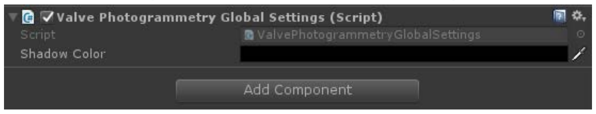
Valve VR Skybox Shader

Use this for your skybox materials in scenes that use The Lab Renderer. The shader's workflow is identical to that of Unity's default skybox shader.

Valve VR Photogrammetry Shader

This shader is optimized for content that is unlit but still needs to receive real-time shadows. The primary use case is photogrammetry scenes where the lighting is baked into the texture maps, but we also want dynamic objects to cast shadows onto that geometry.

Valve Photogrammetry Global Settings Component


Valve Photogrammetry Global Settings (Script)
Script
ValvePhotogrammetryGlobalSettings
Shadow Color
Add Component

On a per-scene basis, allows shadows to draw on materials that use the vr_photogrammetry shader. Shadow color can be customized in the component.

Command Line Arguments

When you make a standalone build, the scripts included in this package will look for the following set of command line args meant for development. These also work in shipped version of The Lab on Steam:

1. "-msaa X" will force the MSAA level to X where X is 0, 2, 4, 8, or any valid value for QualitySettings.antiAliasing
2. "-noaq" disables Adaptive Quality
3. "-aqoverride X" where X is an int, will force the Adaptive Quality system to level X
4. "-vrdebug" will enable the Adaptive Quality debug visualization at launch
5. "-aqminscale X" will override the Adaptive Quality Min Render Target Scale set on the ValveCamera.cs component on your camera. The default is 0.8.
6. "-aqmaxscale X" will override the Adaptive Quality Max Render Target Scale set on the ValveCamera.cs component on your camera. The default is 1.4.
7. "-aqmaxres X" provides a different method for setting the max render target size by specifying a value in pixels. Running with -aqmaxres 2048 will limit the render target to 2048 in both dimensions and will set the Adaptive Quality Max Render Target Scale appropriately.
8. "-noflush" disables calls to GL.Flush() in the rendering scripts. This is used primarily by NVIDIA and AMD for driver testing. NOTE: This can negatively impact performance by causing the GPU to sit idle waiting for rendering work. If you experiment with this command line argument, you should also be looking at GPUView captures of the before and after. Otherwise, buyer beware.

Keyboard Shortcuts

When you are running your application in Editor mode or in a standalone build, the following keyboard shortcuts are available. These also work in shipped version of The Lab on Steam:

1. Shift+F1 toggles the Adaptive Quality debug visualization
2. Shift+F2 toggles the Adaptive Quality manual quality override mode
3. Shift+F3/Shift+F4 increase/decrease the override quality level when enabled