by Daniel Fritz

Simple Scope Shader

Table of contents

Foreword

Thank you for purchasing this asset. This means the world to me.

If you encounter any issues or have questions on how to use the asset for your specific use case, please reach out to me, and I'll get things sorted as quickly as possible.

Additionally, any feature ideas and suggestions are welcome.

Contact: contact.daniel.fritz@gmail.com
Asset link: https://u3d.as/2VWL

Render Pipeline Setup

URP:
URP works out of the box and can directly be used when downloading the asset.

HDRP:

1. You can set up HDRP by importing the "HDRP.unitypackage" in the root folder of the asset.

Built-In:

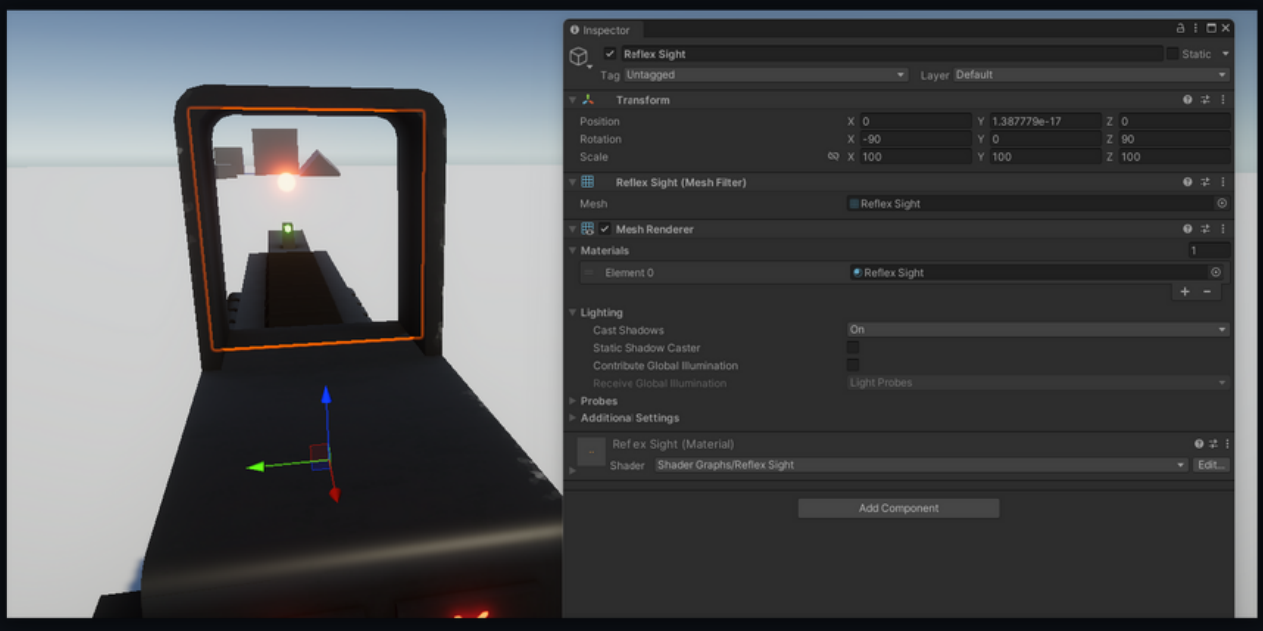
1. You can set up the built-in pipeline by importing the "Built-In.unitypackage" in the root folder of the asset.
2. You need to install the Shader Graph package using the package manager

Reflex sights

Reflex sights

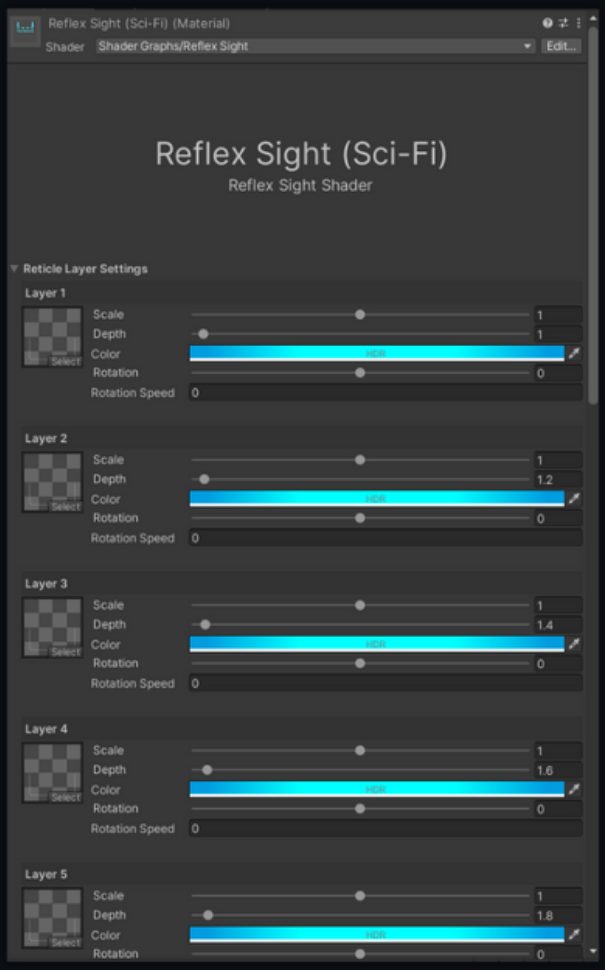
Creating a reflex sight

Creating a reflex sight is easily achieved by adding the "Reflex Sight" material (shader "Shader Graphs/Reflex Sight") to a properly UV mapped mesh. The shader works best with flat surfaces like a plane or a circle.



Reflex sights

Material properties



Reticle:

The reticle texture is used to control the emission of the reticle. The texture should be a white mask.

Depth:

The depth of the reticle. This value determines how much parallax effect is applied. A value of 0 removes the effect completely.

Color:

The emissive color of the reticle.

Scale:

This property controls the size/scale of the reticle texture. You can effectively zoom the texture using this property.

Rotation:

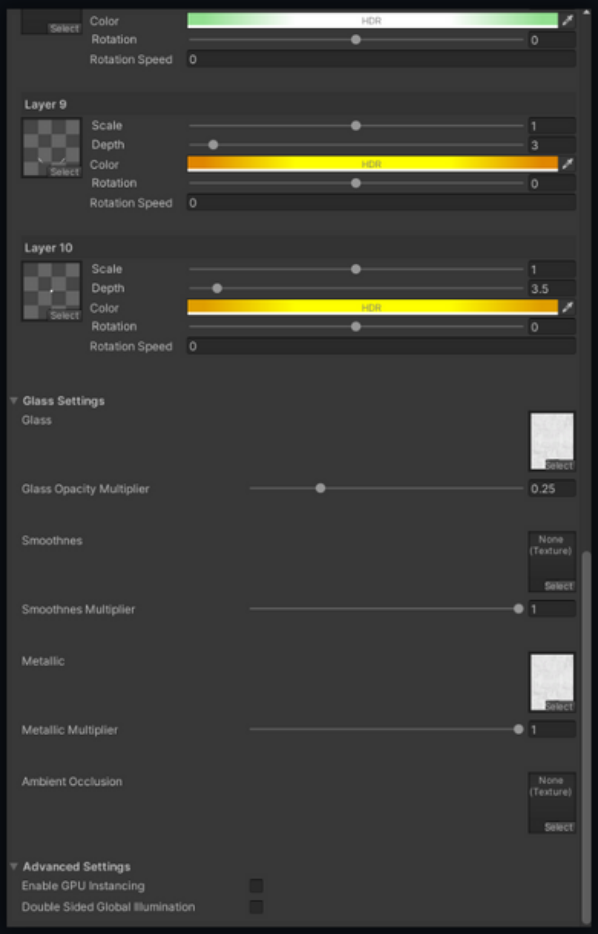
The rotational offset of the reticle texture.

Rotation Speed:

The animation speed of the reticle rotation.

Reflex sights

Material properties



Glass:

The glass texture that is applied to the reflex sight.

Glass Opacity Multiplier

The opacity of the glass texture. This is applied as a multiplier to the glass texture's alpha channel.

Smoothness

The smoothness texture of the glass.

Smoothness Multiplier

The smoothness multiplier of the glass. This is applied to the smoothness texture.

Metallic

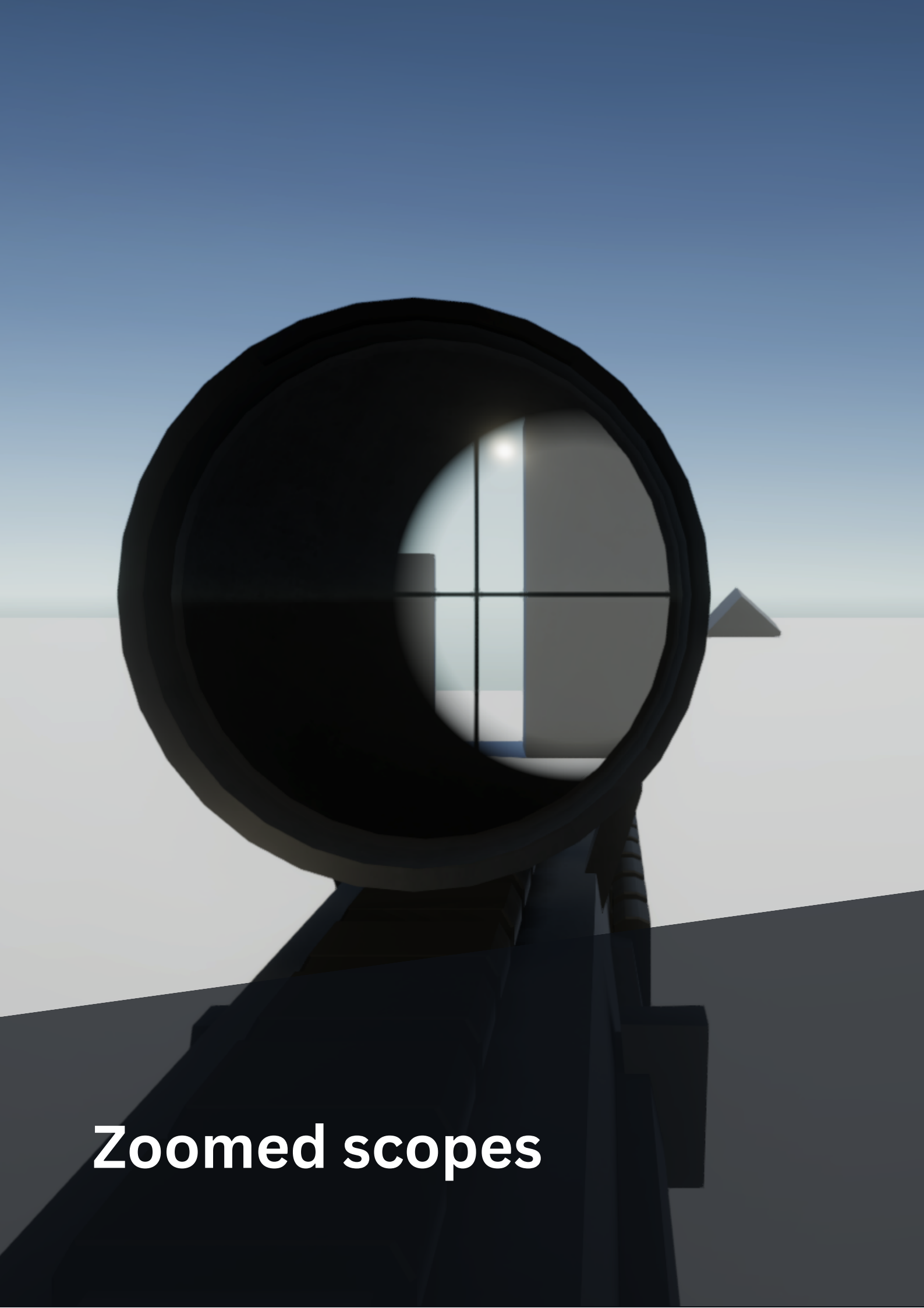
The metallic texture of the glass.

Metallic Multiplier

The metallic multiplier of the glass. This is applied to the metallic texture.

Ambient Occlusion

The ambient occlusion texture applied to the material.



Zoomed scopes

Zoomed scopes

Creating a zoomed scope

1. Pull the Z 400 zoomed scope or the modular zoomed scope lens prefabs into your scene.

2. Add your custom model to the modular zoomed scope lens.

3. Adjust the zoomed scope lenses or replace them with your own.

4. Modify the camera object's position to ensure it does not clip the mesh of your scope.

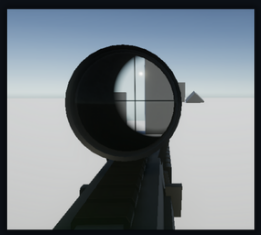
5. Adjust the zoom level on the zoomed scope script to your liking.

Your scope model

+

Modular zoomed scope lens prefab

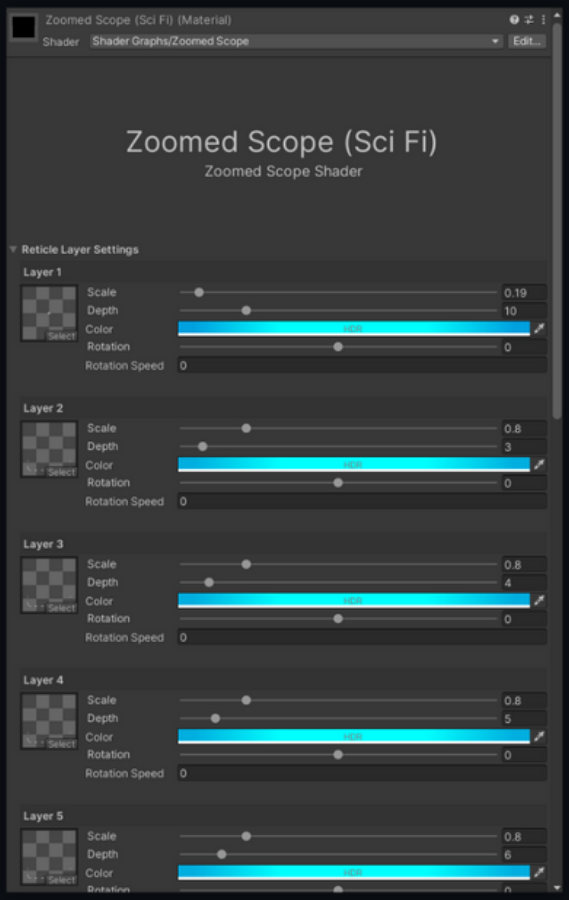
=



Game ready scope

Zoomed scopes

Material properties



Reticle:

The reticle texture is used to control the emission of the reticle. The texture should be a white mask.

Depth:

The depth of the reticle. This value determines how much parallax effect is applied. A value of 0 removes the effect completely.

Color:

The emissive color of the reticle.

Scale:

This property controls the size/scale of the reticle texture. You can effectively zoom the texture using this property.

Rotation:

The rotational offset of the reticle texture.

Rotation Speed:

The animation speed of the reticle rotation.

Zoomed scopes

Material properties

Scene Color Multiplier:
The color multiplier applied to the scene color.

Scene Depth:
The depth effect that is applied to the scene color.

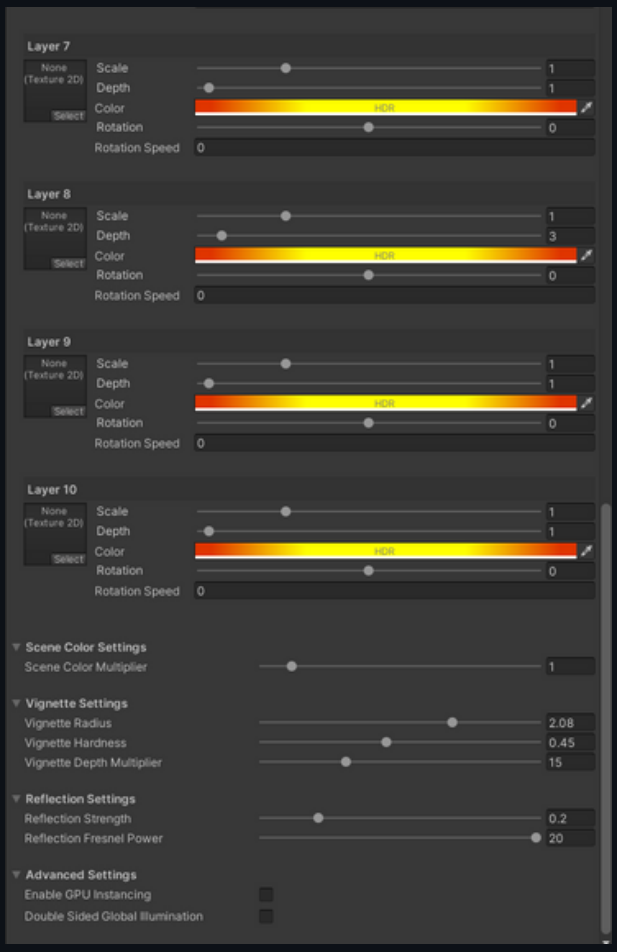
Vignette Radius:
The radius of the procedural vignette.

Vignette Hardness:
The hardness of the procedurally generated vignette.

Vignette Depth Multiplier:
The depth multiplier controls how much the vignette is affected by the parallax effect.

Reflection Strength:
The strength of the applied reflections.

Reflection Fresnel Power:
Controls the power of the fresnel effect used to drive the lens reflections.



Extending the shaders

Adding more reticle layers

You can add more layers to your scope by adding the respective properties to the given shader, sampling the layer in the shader and editing the editor script that handles the inspector.

1. Adding the respective properties to the shader

- Open the shader you want to add more layers to.
- Notice how there is a layer, depth, scale, color, rotation and rotation speed property per reticle layer.
- Create a new set of properties for your layer and name them accordingly. Important here is the reference name of the property. It needs to conform tothe following specifications in order to be picked up by the inspector:

_Reticle_Layer_11
_Reticle_Layer_11_Scale
_Reticle_Layer_11_Depth
_Reticle_Layer_11_Color
_Reticle_Layer_11_Rotation
_Reticle_Layer_11_Rotation_Speed

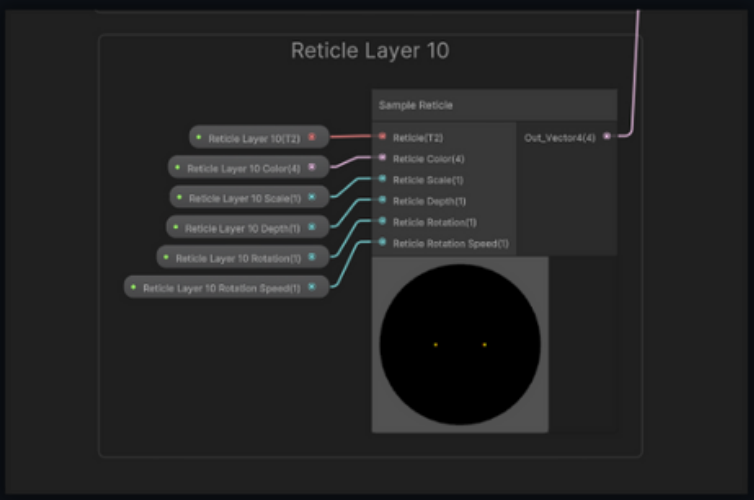
Extending the shaders

Adding more reticle layers

Once added, ensure you increment the green marked number by one. I.e., if you're adding an eleventh layer, ensure the number is 11 and so on.

2. Sampling the layer in the shader

As a second step, you need to sample the new reticle layer. For this, you can use the Sample Reticle node. You can copy this from one of the previous layers. You then need to swap out the properties with the newly created ones from step 1.



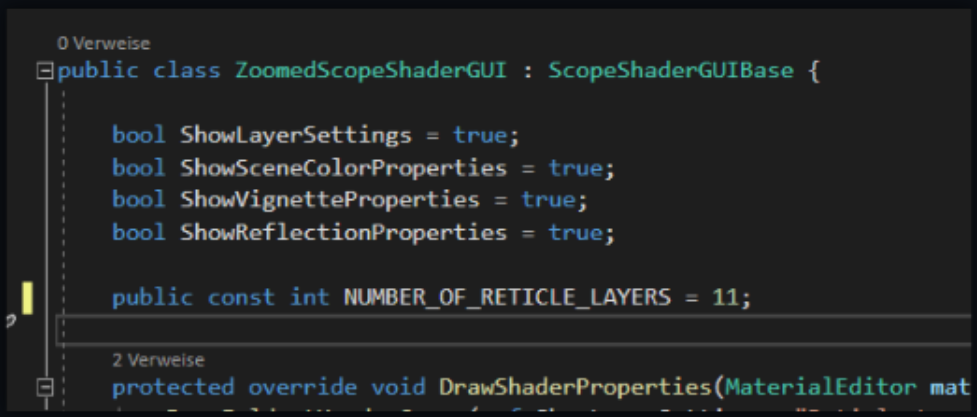
Once done, all you have to do is add the result to the previous layer using the add node.

Extending the shaders

Adding more reticle layers

2. Adding the new layer to the inspector

As a last step, you need to add the newly created layer to the inspector script. This can easily be done by opening either the "ReflexSightShaderGUI" or "ZoomedScopeShaderGUI" scripts, respectively. In there, you need to update the "NUMBER_OF_RETICLE_LAYERS" variable to match the number of overall reticle layers the shader has:



Once complete, the newly added layers should show up in the inspector and be rendered in game.

In case the layers do not show up in the inspector, please make sure the shader properties are correctly named as laid out earlier.