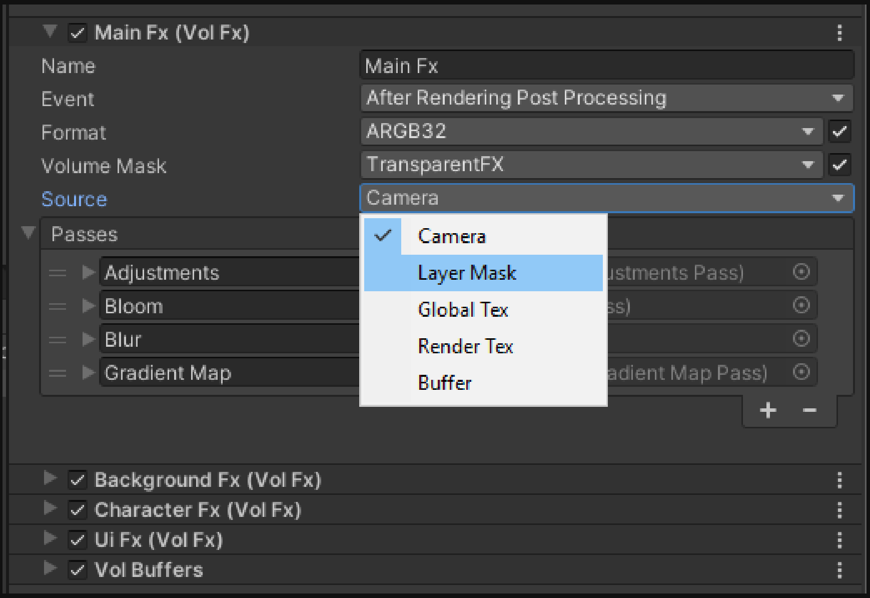
VolFx

Quick Guide

VolFx is customizable selective post-processing vis buffer system by NullTale ⁺₊
that also allows to build a custom scene architexture for visual effects creation

* To use this Asset you need basic understanding of what Urp is and how to configure it for your project, it can be found in the official documentation. Debugging and building PostProcessing Chain cannot be done blindly, it is highly recommended to know what Frame Debugger is, how to use it as a basic understanding of scene rendering process (information can be found in the official Unity documentation)

Common Post Processing usually applied to the Camera content VolFx is consists from modules (**RenderFeatures**) that can process different sources, such as **Camera** content, **GlobalTexture** or a object collected by **LayerMask**



Main Fx (Vol Fx)

Name	Main Fx
Event	After Rendering Post Processing
Format	ARGB32 ✓
Volume Mask	TransparentFX ✓
Source	Camera

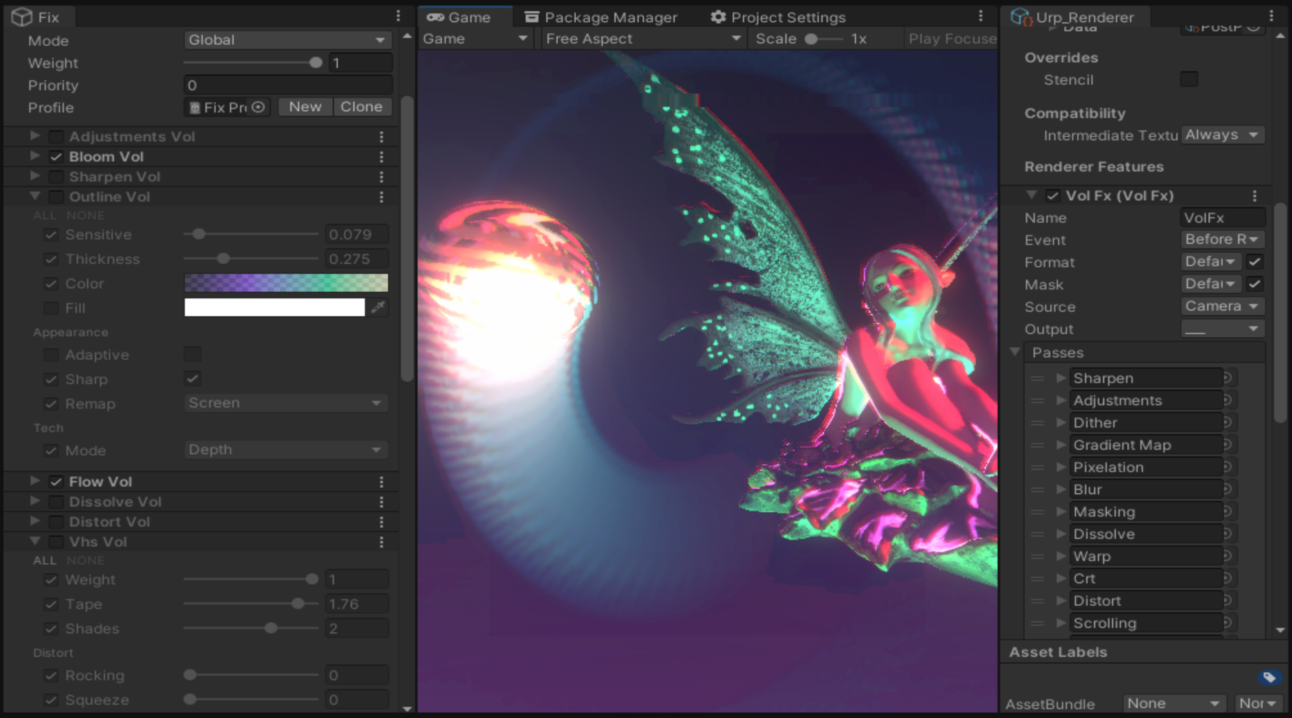
Passes

- Adjustments (…ustments Pass)
- Bloom (…ss)
- Blur
- Gradient Map (…adient Map Pass)

- ✓ Background Fx (Vol Fx)
- ✓ Character Fx (Vol Fx)
- ✓ Ui Fx (Vol Fx)
- ✓ Vol Buffers

It can be used to control the scene and the display of groups of objects on it or to process textures for effects(like light maps, pattern animations, height etc)

To create a processing module you need to add **VolFxRenderFeature** to **UrpRenderer**, configure pass sequence and specify on which source thay will be processed.
It can be a group of objects rendered by **LayerMask**, **GlobalTexture** or **Camera** content.



Fix

Mode	Global
Weight	1
Priority	0
Profile	Fix Pr… New Clone

- Adjustments Vol
- ✓ Bloom Vol
- Sharpen Vol
- Outline Vol

ALL NONE

✓ Sensitive	0.079
✓ Thickness	0.275
✓ Color	
Fill	

Appearance

Adaptive	
✓ Sharp	✓
✓ Remap	Screen

Tech

✓ Mode	Depth

- ✓ Flow Vol
- Dissolve Vol
- Distort Vol
- Vhs Vol

ALL NONE

✓ Weight	1
✓ Tape	1.76
✓ Shades	2

Distort

✓ Rocking	0
✓ Squeeze	0

Urp_Renderer

Overrides

Stencil	

Compatibility

Intermediate Textu	Always

Renderer Features

✓ Vol Fx (Vol Fx)

Name	VolFx
Event	Before R…
Format	Defa… ✓
Mask	Defa… ✓
Source	Camera
Output	___

Passes

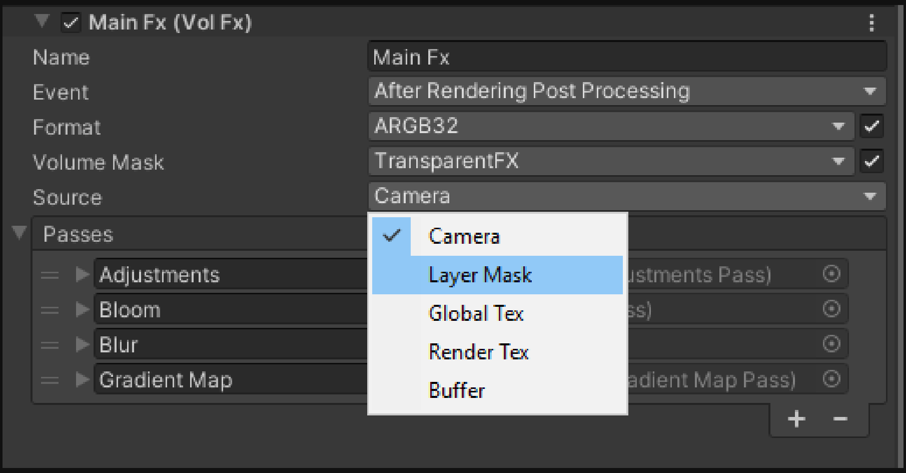
- Sharpen
- Adjustments
- Dither
- Gradient Map
- Pixelation
- Blur
- Masking
- Dissolve
- Warp
- Crt
- Distort
- Scrolling

Asset Labels

AssetBundle	None	Nor

Then add **Passes** that will be applied to the source, in the order in which they are arranged in the queue and use it via **Volume**

* by right clicking on passes heder you can select the menu add all effects using «Add all unique»

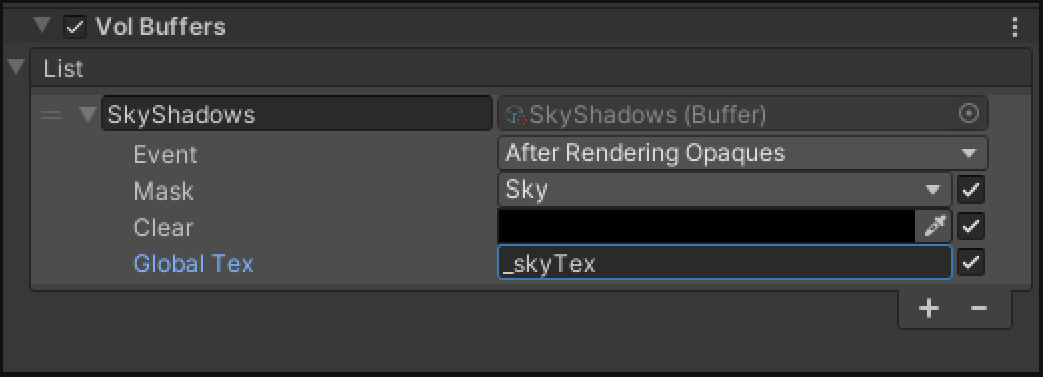

Main Fx (Vol Fx)

Name: Main Fx
Event: After Rendering Post Processing
Format: ARGB32 ✓
Volume Mask: TransparentFX ✓
Source: Camera

Passes
- Adjustments (...ustments Pass)
- Bloom (...ss)
- Blur
- Gradient Map (...adient Map Pass)

* Screenshot demonstates configured **VolFx** Module with 4 custom passes controlled via **Volume Profile** with **TransparentFx LayerMask** that applied to **Camera content,** after unity post processing whill be rendered

For some effects it may be useful to draw objects into a pull in order to later apply its texture through a shader. (It can be light, fog of war or just a mask for some effect)

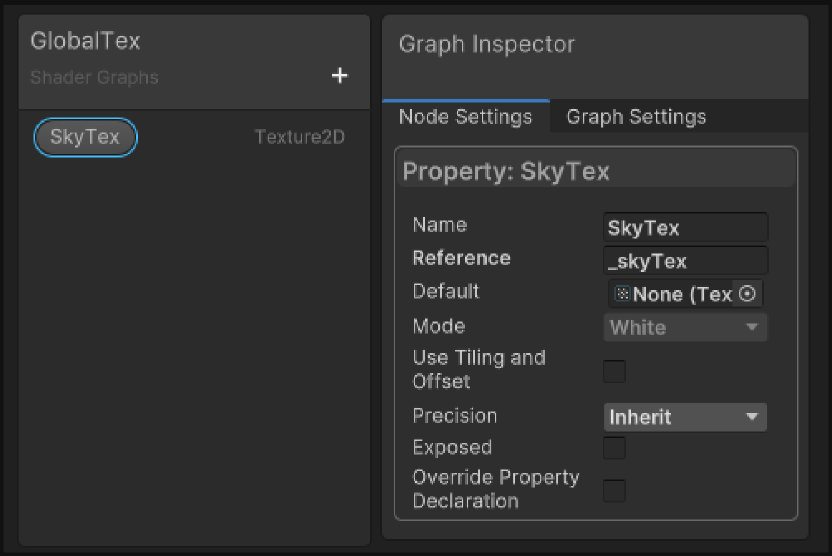

Vol Buffers

List
- SkyShadows: SkyShadows (Buffer)
  - Event: After Rendering Opaques
  - Mask: Sky ✓
  - Clear: ✓
  - Global Tex: _skyTex ✓

* Objects are collected by **LayerMask** and rendered into a separate texture

GlobalTexture output can be processed with realtime and used throug a shader

* screenshot of global texture imported used in shader graph Exposed box is unchecked

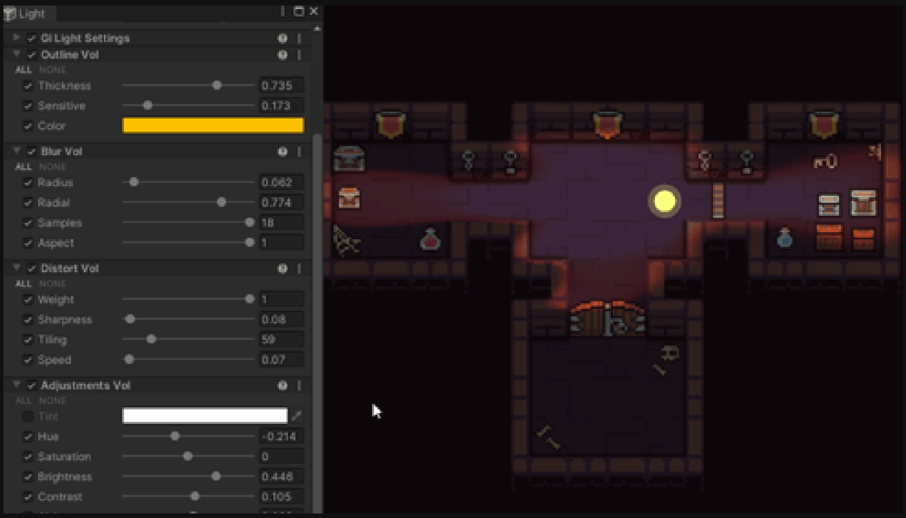

GlobalTex
Shader Graphs
SkyTex Texture2D

Graph Inspector
Node Settings | Graph Settings

Property: SkyTex

Name	SkyTex
Reference	_skyTex
Default	None (Tex
Mode	White
Use Tiling and Offset	☐
Precision	Inherit
Exposed	☐
Override Property Declaration	☐

* **VolumeMask** must be specified in order to select what settings to use for processing, example of post processing Light Texture



Light

✓ Gi Light Settings

✓ Outline Vol

ALL NONE

✓ Thickness	0.735
✓ Sensitive	0.173
✓ Color	

✓ Blur Vol

ALL NONE

✓ Radius	0.062
✓ Radial	0.774
✓ Samples	18
✓ Aspect	1

✓ Distort Vol

ALL NONE

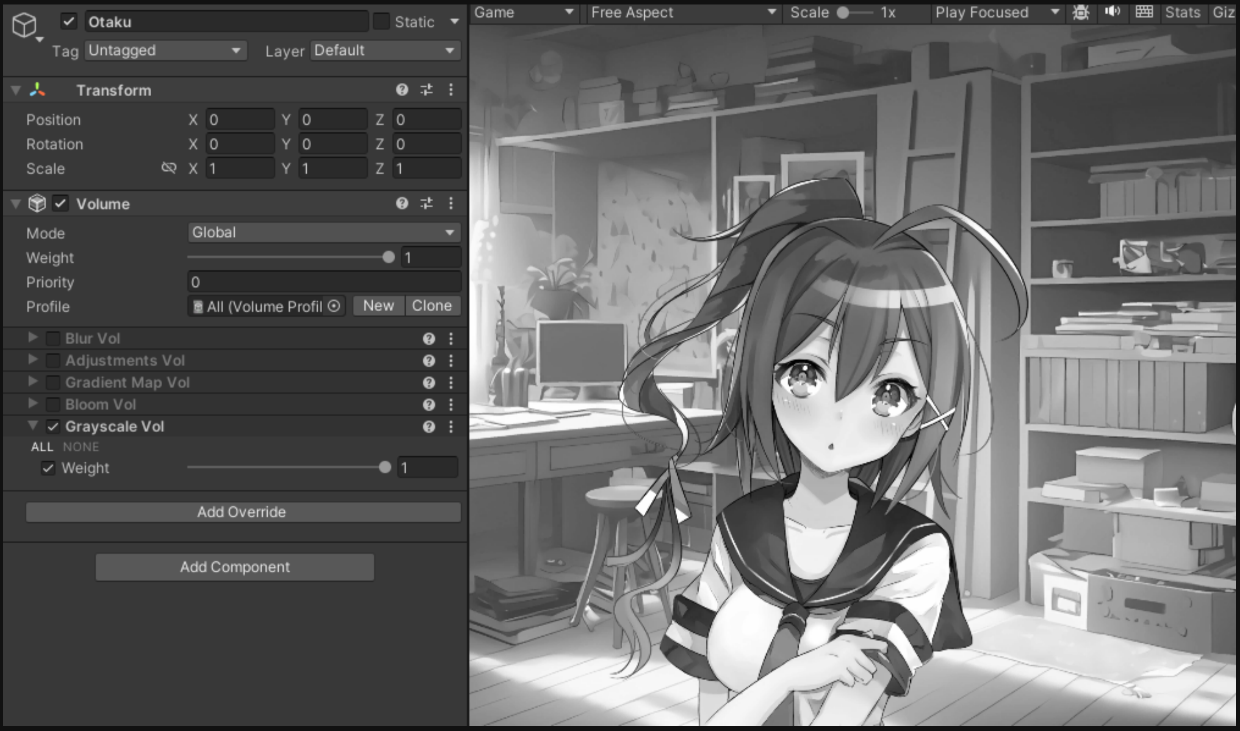
✓ Weight	1
✓ Sharpness	0.08
✓ Tiling	59
✓ Speed	0.07

✓ Adjustments Vol

ALL NONE

Tint	
✓ Hue	-0.214
✓ Saturation	0
✓ Brightness	0.446
✓ Contrast	0.105

VolFx also allows you to create **CustomPasses**
* Example-template of a simple **GrayscalePass** can be found in ProjectSamples and used as a template, basically you need a shader override pass and control them via **VolumeSettings**



✓ Otaku Static

Tag Untagged Layer Default

Transform

	X	Y	Z
Position	0	0	0
Rotation	0	0	0
Scale	1	1	1

✓ Volume

Mode	Global
Weight	1
Priority	0
Profile	All (Volume Profil

Blur Vol
Adjustments Vol
Gradient Map Vol
Bloom Vol
✓ Grayscale Vol
ALL NONE
✓ Weight 1

Add Override

Add Component

To create a CustomPass, it must be inherited from **VolFx.Pass** and then it will appear in the list to be added.

```
[ShaderName("Hidden/VolFx/Grayscale")] // shader name for pass material
public class GrayscalePass : VolFx.Pass
{
    public override string ShaderName => string.Empty;

    // =======================================================================
    public override bool Validate(Material mat)
    {
        // use stack from feature settings, feature use custom VolumeStack with its own
LayerMask
         var settings = Stack.GetComponent<GrayscaleVol>();

         // return false if we don't want to execute pass, standart check
         if (settings.IsActive() == false)
             return false;

         // setup material before drawing
         mat.SetFloat("_Weight", settings.m_Weight.value);
         return true;
     }
}
```

By default material is created automatically using path and **ShaderNameAttribute** and is updated every time before processing is called.
But you can also overiider low-level to access additional functionality.

```
// called to perform rendering
public virtual void Invoke(CommandBuffer cmd, RTHandle source, RTHandle dest,
                             ScriptableRenderContext context,
                              ref RenderingData renderingData)
{
    Utils.Blit(cmd, source, dest, _material, 0, Invert);
}
```

In this way you can expand the engine and create dynamic effects controlled via VolumeProfile and scenes that have their own processing pipelines.

Known Issues – Problem solving

• In order to prevent bugs with effects compatibility recomended install effect over the VolFx Package

* unity has a complex system of compilation and adding types from version to version may cause problems with adding new VolFx Render Feature
* the problem occurs, **VolFxPass** and **VolFxApi** files can be manually removed from effect scripts so that for some reason Unity does not identify them as an added Render Feature.

• For selective compilation custom define VOL_FX is used if there are any errors in compatibility and compilation of scripts the project state can be returned manually by removing this flag

* basically this only applies if the effect script was added with contained errors, missing plugins required for operation (urp) and was not compiled to create detection services.