



Volumetric Fog & Lighting for URP

User Manual

v 1.1.7

Table of Contents

Introduction	3
What is it for?	3
What is it NOT for?	3
Features	3
Requirements	4
Known restrictions	4
How to use	5
Setup	5
Renderer feature settings	10
Haze Settings Overrides	13
Global fog	14
Local fog volumes	18
Additional light contribution	20
Multiple scattering	21
Transparency support	23
Performance considerations	25
FAQ	25

Introduction

HAZE is a volumetric fog and lighting system, designed to help you enhance the visuals of your project. It is based on a froxel-based approach, making it quite fast to compute while offering a large degree of flexibility to balance visual quality and performance.

What is it for?

HAZE is useful for unifying the lighting of a scene, by applying a soft layer of atmospheric fog. It's a great way to introduce more atmosphere into your levels without too much overwhelming visual bloat. It's made with art direction in mind, rather than physical accuracy.

What is it NOT for?

This is not a fully fledged system for dense volumetric effects like clouds and heavy mist. Because of the way froxel-based fog works, HAZE is not suited for very well-defined volumetric forms and shapes, especially in long distances. A raymarched approach would be more suited for effects like that, and HAZE currently doesn't support that.

Features

- Lightweight froxel-based implementation
- Global fog controls
- Local fog volumes for more granular control
- Integration with Unity's volume system for
 - Global fog visuals
 - Overriding renderer feature settings
- Integrated screen-space multiple scattering effect (SSMS) to blur the scene in the distance based on the density of the volumetric fog
- Support for main light shadows
- Support for secondary lights and shadows (**currently only in Forward+/Deferred+ pipelines**)
- Light cookie support
- Support for height-based cutoff
- Support for both perspective and orthographic camera projection
- Supports illumination contribution from adaptive probe volumes (APV)

Requirements

- HAZE is made for Unity 6+ versions
- HAZE is currently only supported in the **Universal Render Pipeline (URP)**
- HAZE is based on the Unity RenderGraph API, so it **currently doesn't support compatibility mode**.

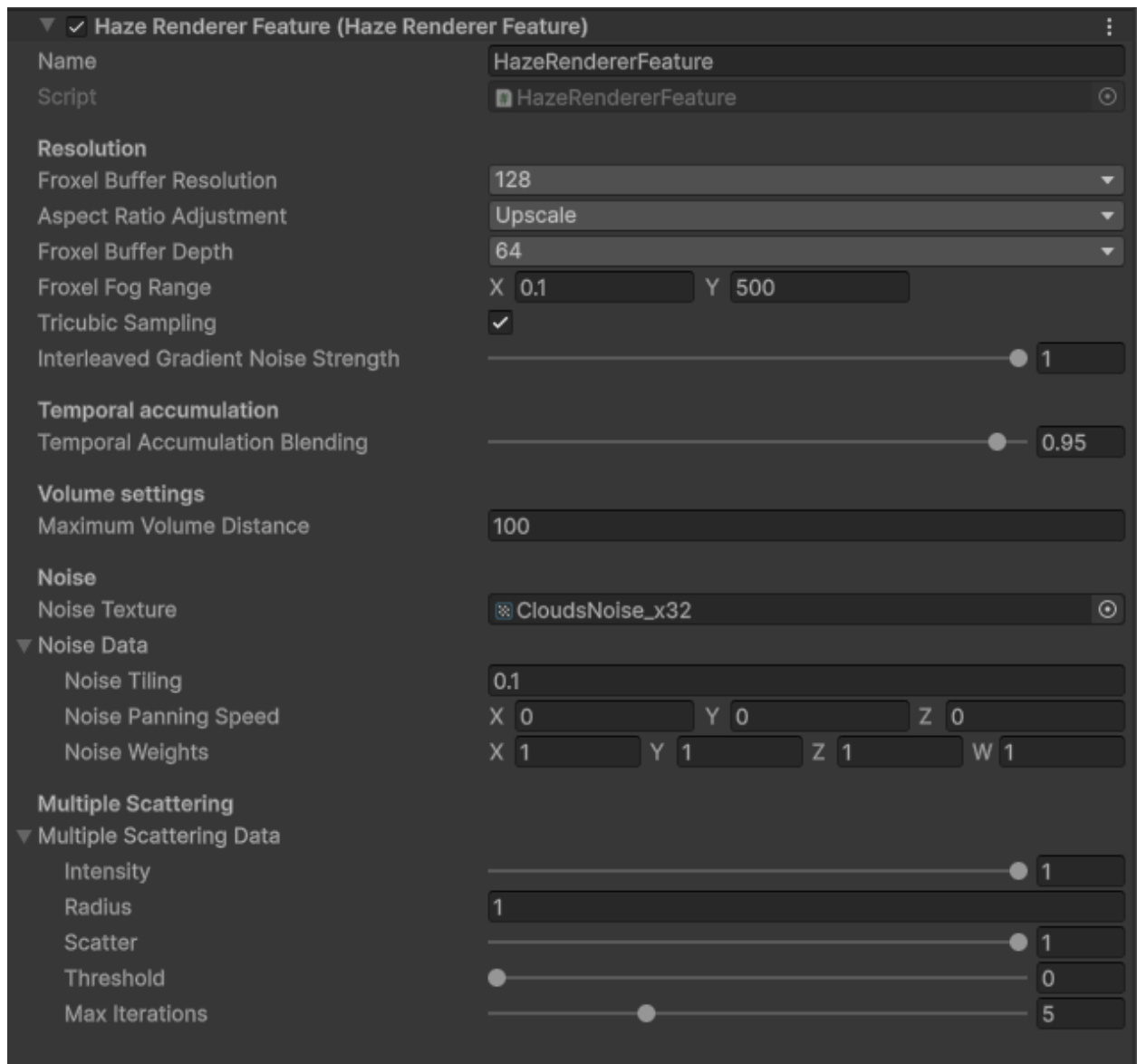
Known restrictions

- HAZE uses temporal reprojection in order to reduce noise and artifacts from lower-resolution buffers and perform better. Because of the way the history buffer is set up for temporal reprojection, **HAZE doesn't currently support multiple cameras**.
- Because of the way secondary light data is stored by Unity, **HAZE only supports punctual light contribution to the fog in the Forward+ and Deferred+ pipelines**. This can be updated if there's enough request for it, but it might come at a performance overhead.
- At this moment, all fog (global and local) share the same 3D noise texture which is assigned via **HAZE's** renderer feature. Support for per-volume noise can also be added if there's enough demand for that.

How to use

Setup

The main body of settings for **HAZE** come from the **Haze Renderer Feature** that needs to be added to your active renderer:

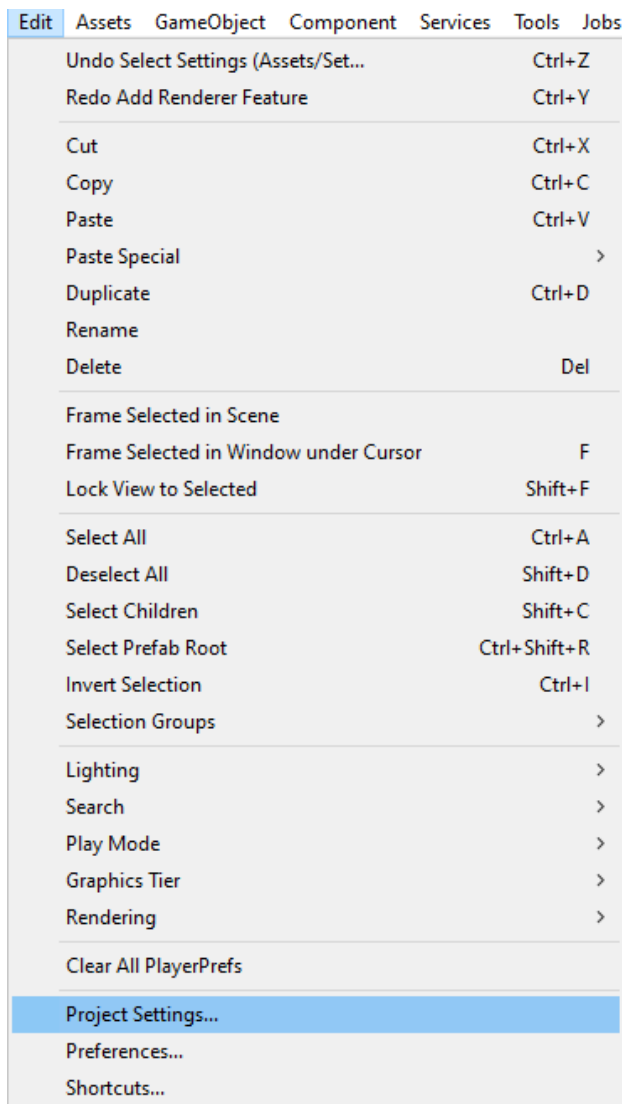


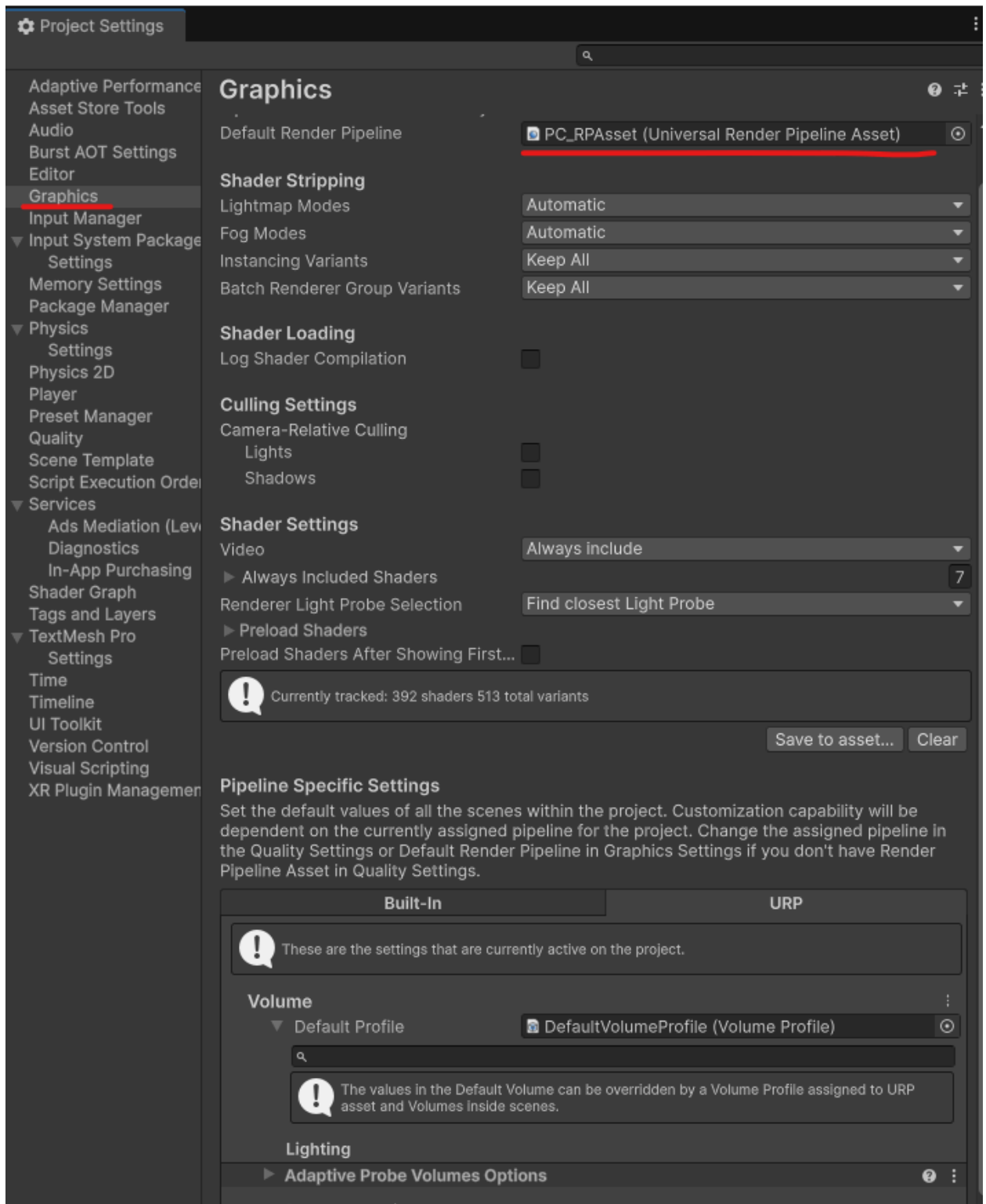
The image shows a settings panel for the 'Haze Renderer Feature'. The panel is dark-themed and contains various configuration options organized into sections. At the top, there is a dropdown menu for 'Name' set to 'HazeRendererFeature' and a 'Script' dropdown set to 'HazeRendererFeature'. Below these are sections for 'Resolution' (Froxel Buffer Resolution: 128, Aspect Ratio Adjustment: Upscale, Froxel Buffer Depth: 64), 'Froxel Fog Range' (X: 0.1, Y: 500), 'Tricubic Sampling' (checked), and 'Interleaved Gradient Noise Strength' (slider at 1). The 'Temporal accumulation' section has 'Temporal Accumulation Blending' set to 0.95. 'Volume settings' include 'Maximum Volume Distance' set to 100. The 'Noise' section has 'Noise Texture' set to 'CloudsNoise_x32'. Under 'Noise Data', 'Noise Tiling' is 0.1, 'Noise Panning Speed' has X, Y, and Z all set to 0, and 'Noise Weights' has X, Y, Z, and W all set to 1. The 'Multiple Scattering' section has 'Multiple Scattering Data' with 'Intensity' at 1, 'Radius' at 1, 'Scatter' at 1, 'Threshold' at 0, and 'Max Iterations' at 5.

Section	Property	Value
General	Name	HazeRendererFeature
	Script	HazeRendererFeature
Resolution	Froxel Buffer Resolution	128
	Aspect Ratio Adjustment	Upscale
	Froxel Buffer Depth	64
	Froxel Fog Range	X: 0.1, Y: 500
Sampling	Tricubic Sampling	✓
	Interleaved Gradient Noise Strength	1
Temporal accumulation	Temporal Accumulation Blending	0.95
	Volume settings	Maximum Volume Distance: 100
Noise	Noise Texture	CloudsNoise_x32
	Noise Data	Noise Tiling: 0.1
Noise Data	Noise Panning Speed	X: 0, Y: 0, Z: 0
	Noise Weights	X: 1, Y: 1, Z: 1, W: 1
Multiple Scattering	Multiple Scattering Data	Intensity: 1
	Radius	1
	Scatter	1
	Threshold	0
	Max Iterations	5

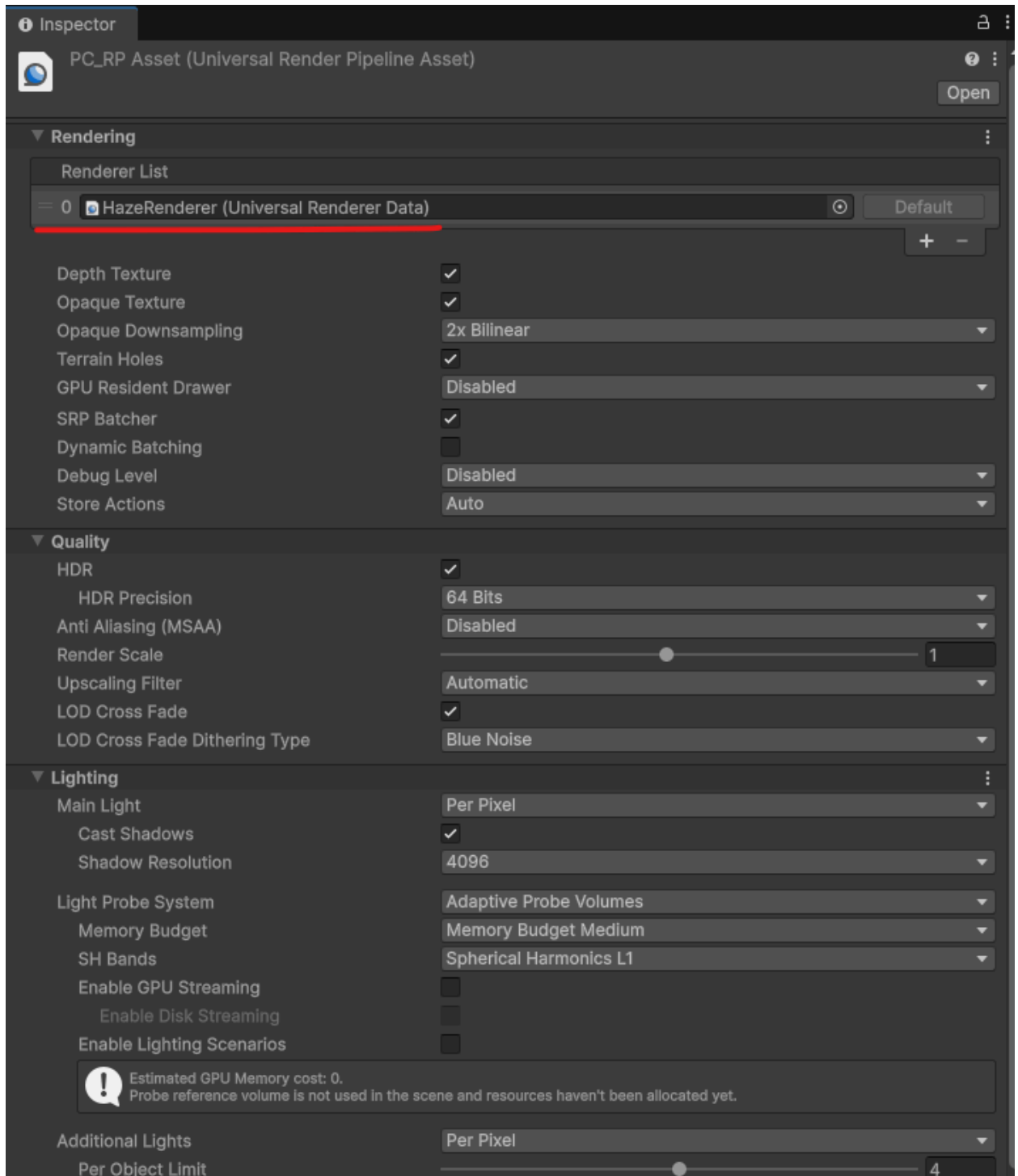
In order to add the renderer feature to your renderer, you'll have to follow these steps:

1. Find your **URP asset**. A way to do that is by going into the project settings (and finding the **Default Render Pipeline** in the **Graphics** section.

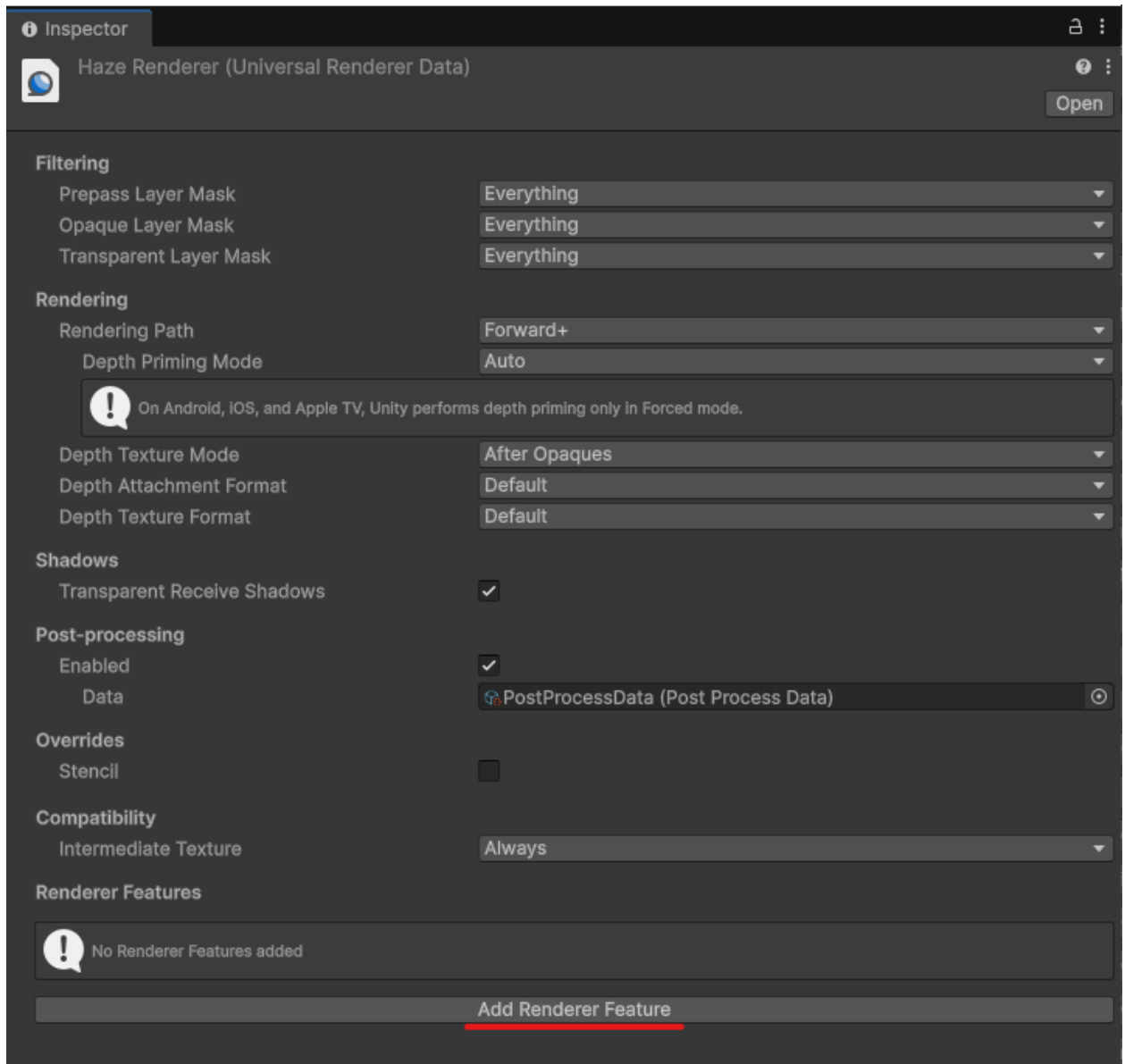




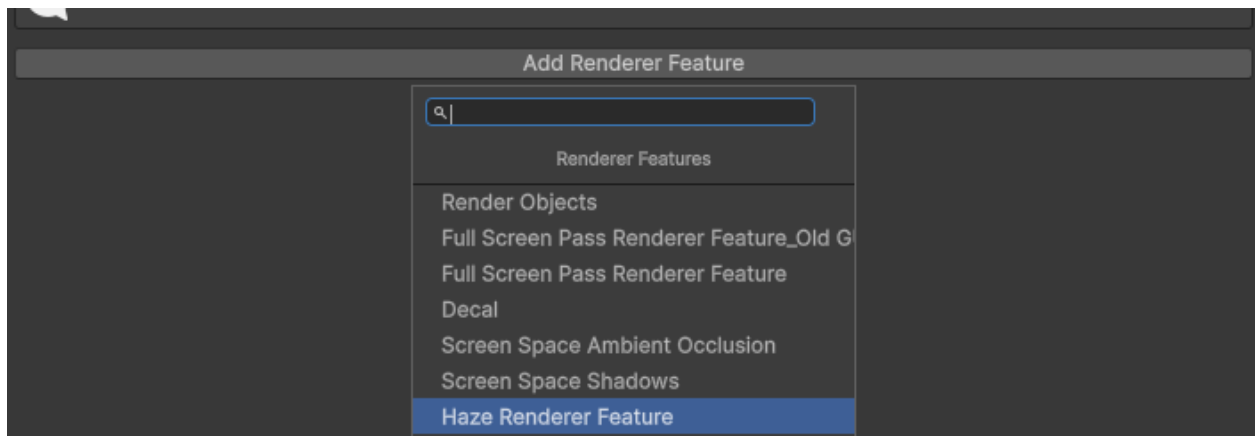
2. Inside the **URP Asset** you'll see a list of renderers holding the **Universal Renderer Data**:



3. Selecting the default **Universal Renderer Data** object will allow you to **Add Renderer Features**:



4. There you will be able to add the **Haze Renderer Feature** in your renderer asset:

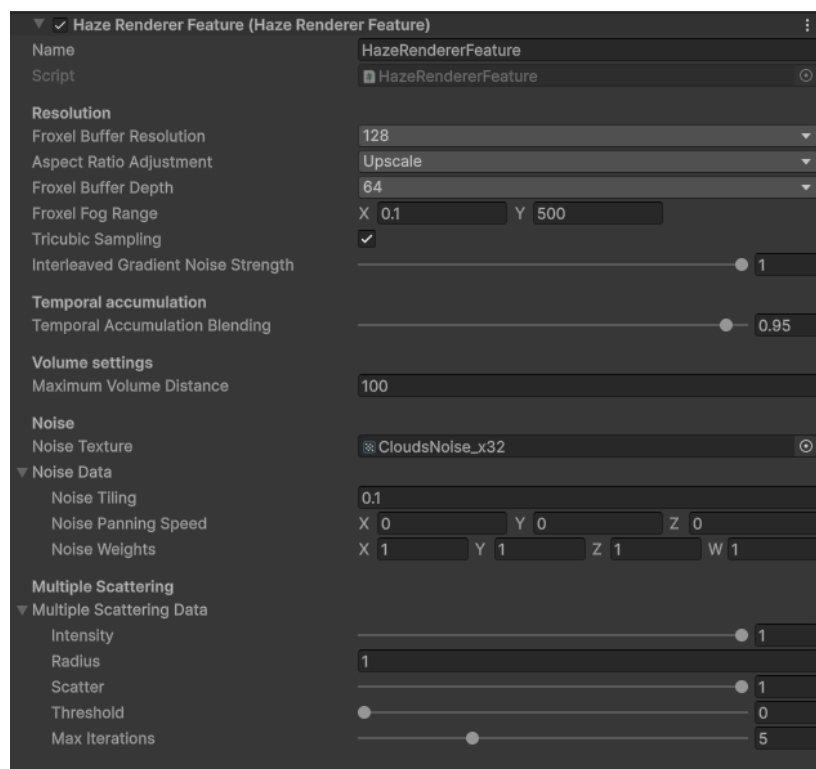


After you've added the **Haze Renderer Feature** to your renderer asset, the demo scenes under the “**Demo**” folder should work as intended.

Renderer feature settings

The **Haze Renderer Feature** holds many settings that are mostly related to the quality and performance of the fog.

When you add the renderer feature, the default values should look like this:



Here's a rundown of the **Haze Renderer Feature** settings:

Parameter	Description
Froxel Buffer Resolution	The width and height resolution of the 3D buffer used to store the fog data. Larger values allow for better detail, but increased performance overhead.
Aspect Ratio Adjustment	As of HAZE 1.1.2 there are options to scale the froxel buffer up or down to match the aspect ratio of the main camera. This should help reduce flickering artifacts even more.
Froxel Buffer Depth	The depth resolution of the 3D buffer used to store the fog data. As before, increasing the resolution will result in better quality in some cases, but will also increase performance overhead.
Froxel Fog Range	The minimum and maximum distance at which the fog will be visible, relatively to the camera position.
Tricubic Sampling	Greatly reduces low-resolution artifacts in the fog but comes at a slightly increased performance overhead.
Interleaved Gradient Noise Strength	Adjusts the strength of the interleaved gradient noise (IGN) applied upon compositing. When using TAA, IGN is useful to reduce artifacts, but it can otherwise be noticeable and noisy.
Temporal Accumulation Blending	As of HAZE 1.1.6 , the temporal accumulation blending value is exposed, allowing to reduce the blending amount in order to address trailing issues with moving lights. Set to 0 to completely disable temporal accumulation.
Maximum Volume Distance	The maximum distance at which a density volume will be considered visible.
Noise Texture	The 3D noise texture used in global fog and local density volumes to introduce cloud-like shapes.

The **Haze Renderer Feature** also hosts settings for the 3D noise as well as for the **Screen-Space Multiple Scattering (SSMS)** effect, as they apply to both global and local fog.

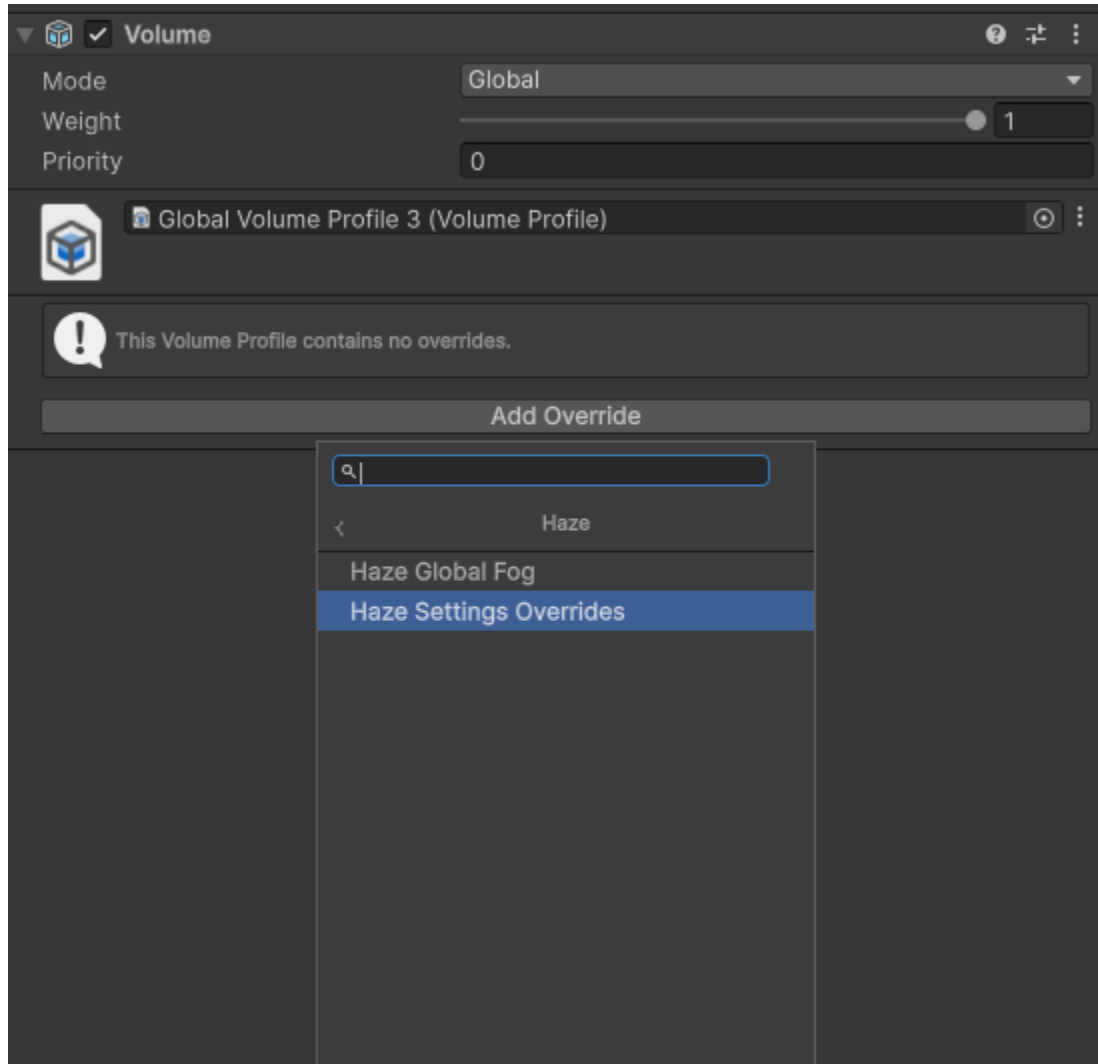
Noise Data	
Noise Tiling	The tiling of the 3D noise texture in world-space. Increase to make the noise detail frequency higher.
Noise Panning Speed	The panning speed of the 3D noise in each axis. Used to make the cloud-like shapes of the fog move in the desired direction.
Noise Weights	The weights for the noise texture channels. Each component is multiplied by the corresponding channel (RGBA) of the 3D noise texture. The included noise texture stores 3D worley noise in different frequencies with the red channel hosting the lowest frequency and the alpha channel hosting the highest. Therefore, increasing the value of the X component of the noise weights will result in bigger denser shapes.

Multiple Scattering Data	
Intensity	Determines the blend between regular volumetric fog and fog with SSMS enabled. Set to 0 to completely disable SSMS , which should also increase performance.
Radius	Determines the brightness of the blurred image that gets composited with the fog. Lower values make the result brighter.
Scatter	Determines the blur distance on the SSMS buffer. Values closer to 1 make the composited image more blurry.
Threshold	The brightness threshold for the pre-filtering of the SSMS . A value of 0 doesn't perform any filtering and the whole image is blurred. Larger values will cause only brighter parts of the image to be blurred.
Max Iterations	The maximum number of blur iterations performed for SSMS . Larger values will increase the blur distance, but will increase performance overhead.

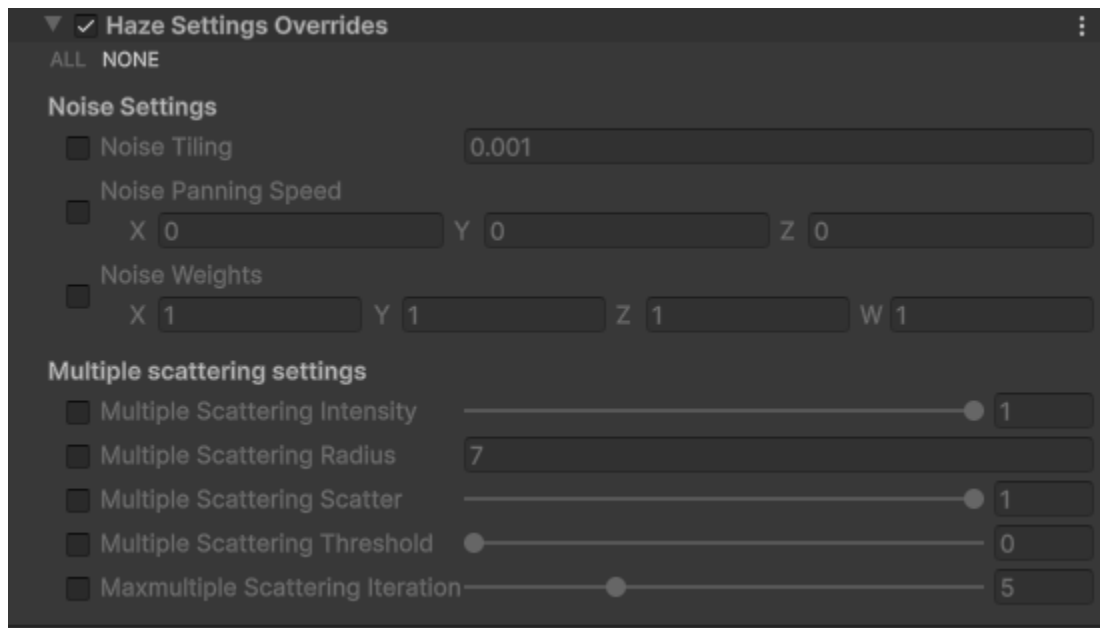
Haze Settings Overrides

While the noise and SSMS settings can also affect performance, it's handy to be able to override some settings at least on a per-level basis.

HAZE supports optionally overriding these settings through a volume component called **Haze Settings Overrides**. You can add it to a Unity volume through the **Add Override** option:



This override will expose overrides for the noise and **SSMS** settings described above:

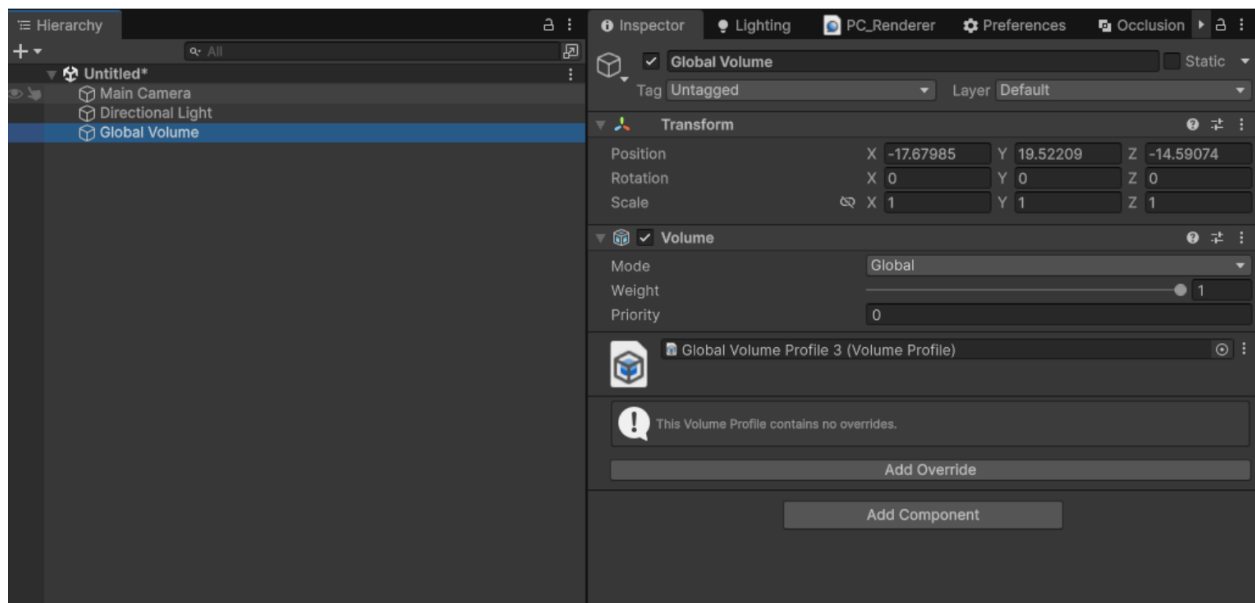


That way, noise tiling can differ in different scenes or areas, without modifying other parameters of the noise or **SSMS**.

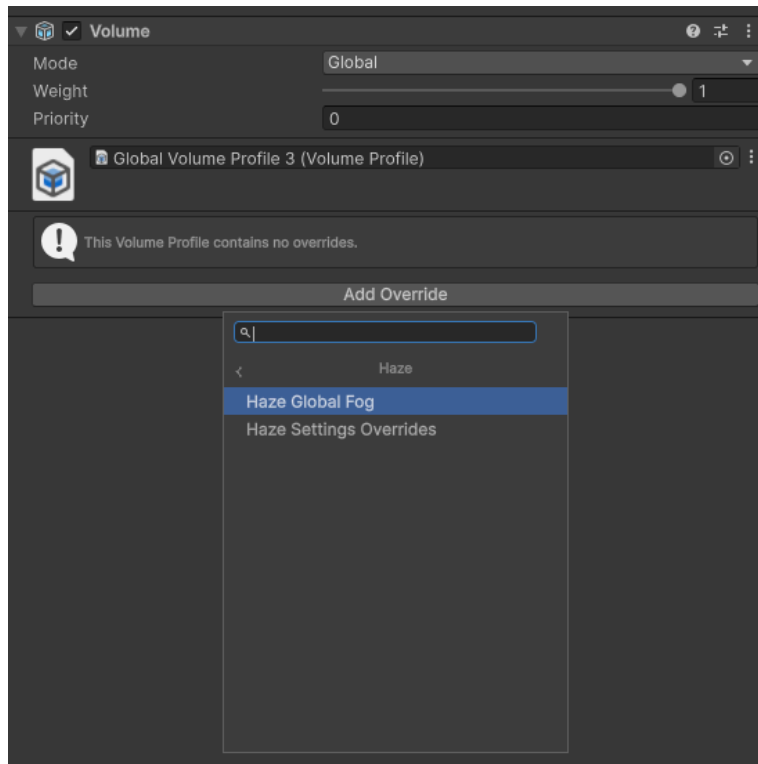
Global fog

HAZE provides the option to use settings for a global fog that affects all areas within the visible range of the effect.

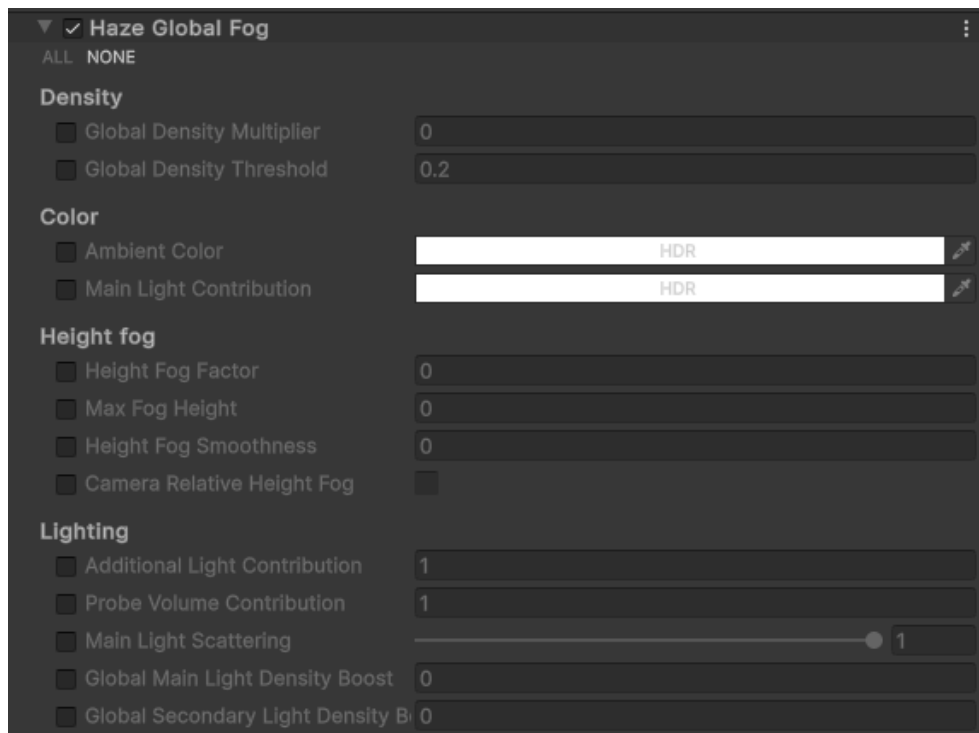
In order to use **Haze Global Fog** in your scene you'll need a Unity volume:



From there, you can add the **Haze > Haze Global Fog** override to your volume:



This override will present you with these options for the **Haze Global Fog**:



Provided that you have included the **Haze Renderer Feature** in your renderer and you have post-processing enabled, enabling and changing these overrides should immediately affect your scene view.

Check out the **GlobalFog** demo scene included in the **Demo** folder for an example as to how to set up the override values.

Here's a rundown of what the **Haze Global Fog** override parameters do:

Parameter	Description
Global Density Multiplier	Determines the density of the global fog. Increase for a thicker fog effect.
Global Density Threshold	The noise threshold that will cut away from the density of the fog. Increase to reveal the shapes from the 3D noise texture.
Ambient Color	The main color of the global fog.
Main Light Contribution	The additional color that gets multiplied by the main light color. Increase HDR intensity for more intense sun rays.
Height Fog Factor	The factor by which the fog density is reduced based on world-space height. Set to 0 if you don't want any height fog effects.
Max Fog Height	The maximum height of the fog, assuming the height fog factor is larger than 1.
Height Fog Smoothness	The smoothness of the height fog threshold. Increasing the value makes the transition around the maximum height smoother. Negative values will invert the height fog effect (essentially removing density underneath the maximum height).
Camera Relative Height Fog	Makes the maximum fog height relative to the world-space camera position.
Additional Light Contribution	Only available in Forward+/Deferred+; determines how much additional lights contribute to the final color of the global fog.
Probe Volume Contribution	Determines how much adaptive probe volume illumination contributes to the final color of the global fog.

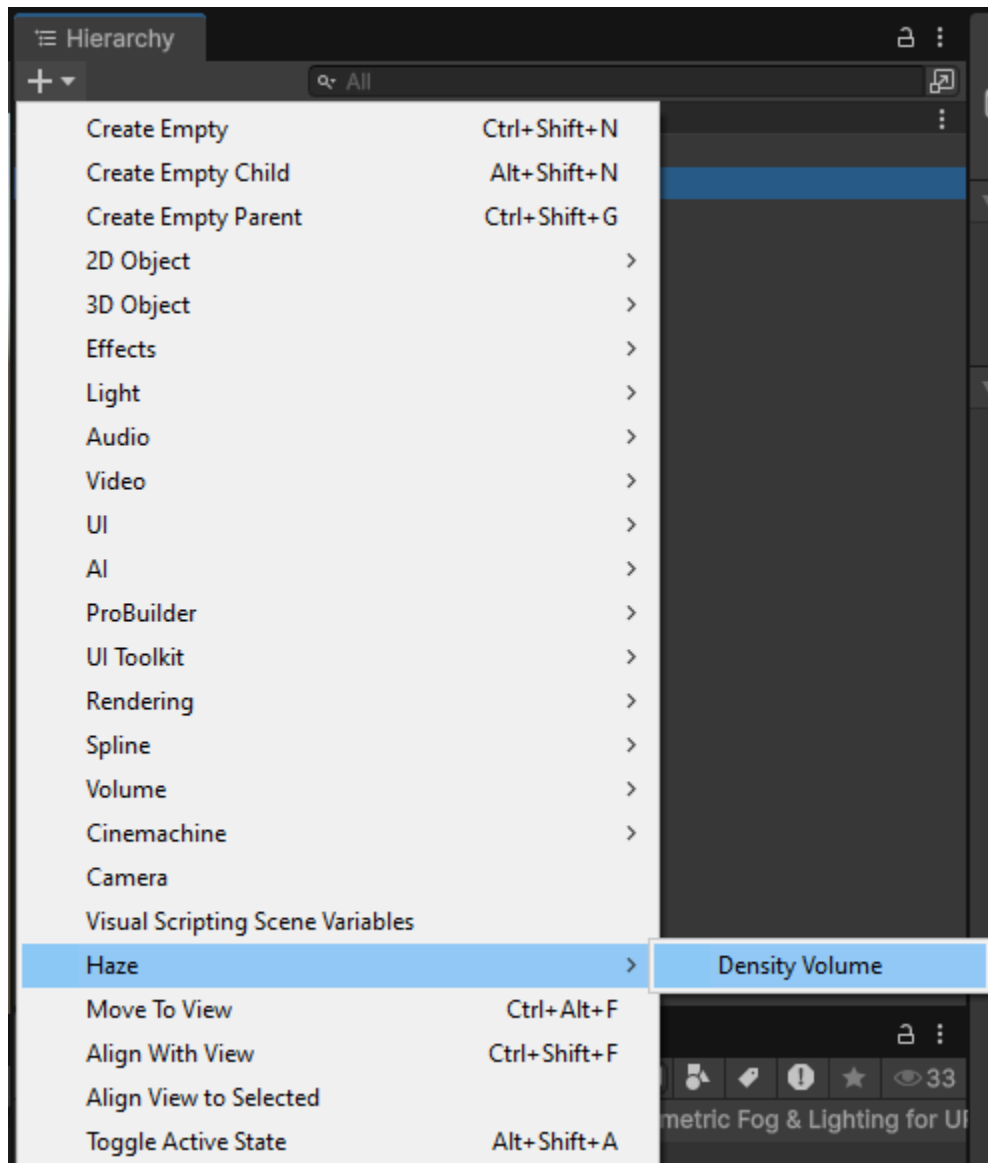
Main Light Scattering	Determines the scattering phase of the main light. Values closer to 1 make the main light scatter more into the global fog.
Global Main Light Density Boost	Additional density boost for areas that are not in the shadow of the main light. Used to enhance the look of sun rays.
Global Secondary Light Density Boost	Only available in Forward+/Deferred+; adds fog density based on the volume that secondary lights occupy. Can be used to add volumetric lights in your scene without having fog occupy the whole space.

Note: Both **main light density boost** and **secondary light density boost** (in both global and local fog) get affected by the noise threshold and the height fog.

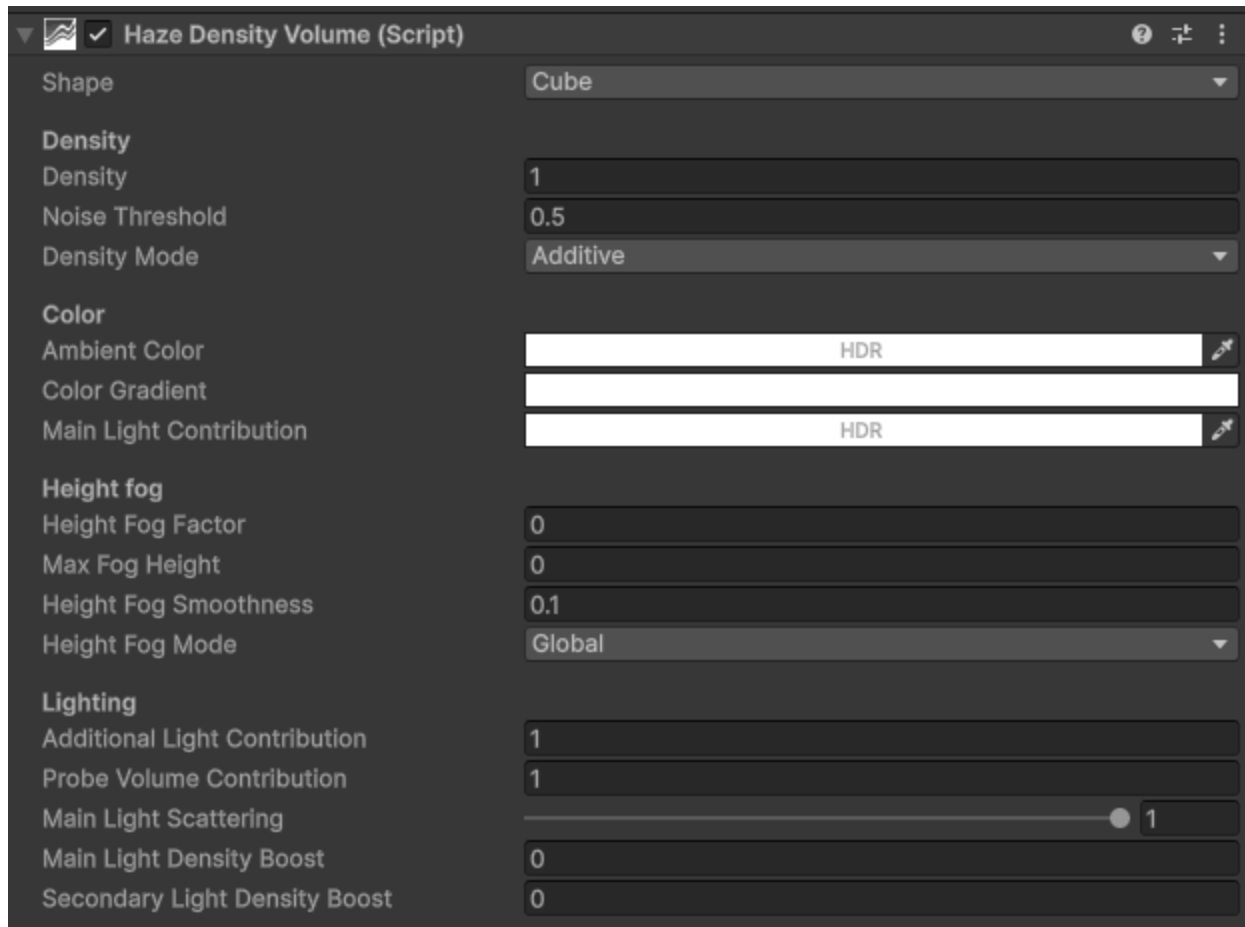
Local fog volumes

For more spatial and art direction control, **HAZE** supports using local density volumes that locally add fog in your scene featuring the same controls as the **global fog**.

Adding a local fog volume is possible through the **Game Object > Haze > Density Volume** menu option:



This option will create a **Haze Density Volume** in your scene that has a set of properties for you to adjust:



The volumes come in two shapes: **Cube** and **Sphere**. They can be placed anywhere in your scene and can be scaled and rotated in any axis you desire.

The fog inside those volumes utilizes the corresponding parameters in the same manner as the global fog.

That being said, **it's entirely possible to emulate the global fog effect using a large enough volume that follows the camera.**

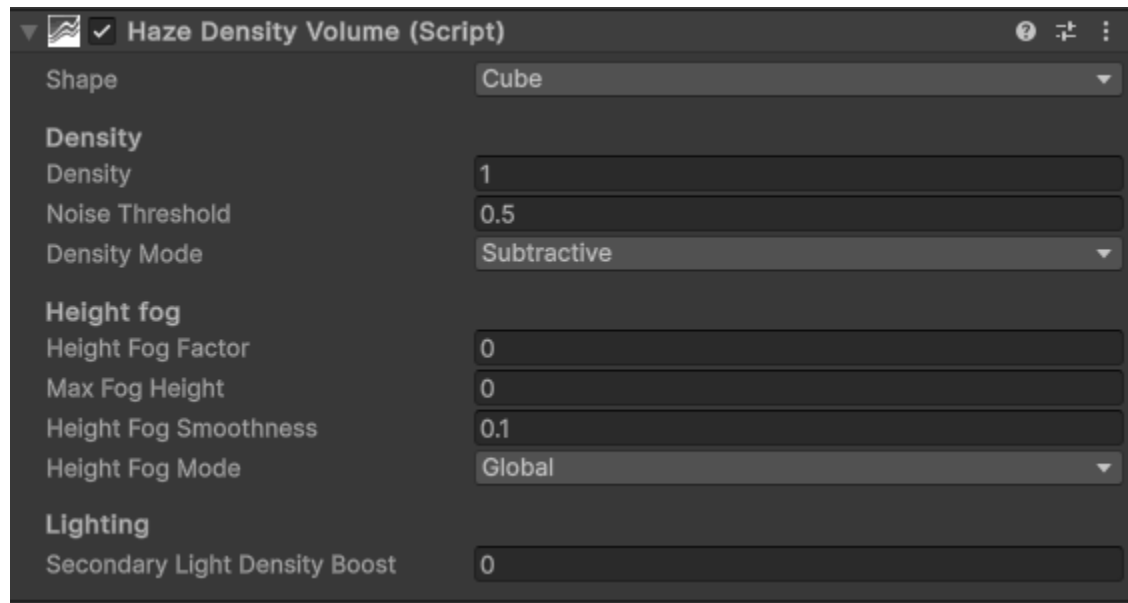
Make sure to check out the **HazeDensityVolumes** demo scene in the **Demo** folder for examples as to how **Haze Density Volumes** can be set up.

As of version 1.1.2 of **HAZE**, **Haze Density Volumes** can be either **Additive** or **Subtractive**:

- **Additive:** The default behaviour of the density volumes, adding density and color to the overall fog.
- **Subtractive:** A new mode of density volumes that **subtracts** density from the overall fog. The volumes still calculate their density like before (including height fog and secondary light density boost) but the final density of the volume is **subtracted** from

the fog. This can be used for removing fog from an area (like interiors) or to do more dynamic effects like point lights dissolving the fog of an area.

Since subtractive volumes don't utilize color in any way, their interface only contains density-related fields:

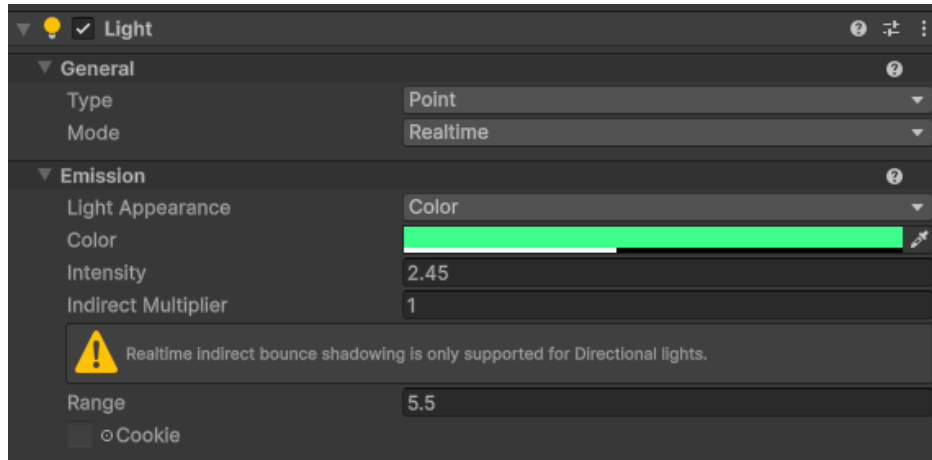


As of version 1.1.5 of **HAZE**, density volumes also have a **Color Gradient** input, which maps a color gradient along the local Z axis of the volume and multiplies its ambient color with the resulting color of the gradient. The alpha value of the gradient can be used to reduce the density of the volume in specific areas, as mapped by the gradient.

Additionally, as of 1.1.5, density volumes are created as **static game objects by default**. Marking a volume as static ensures that it won't update its data every frame, thus saving performance on the CPU side.

Additional light contribution

As of version 1.1.0 of **HAZE**, additional per-light control has been added in order to more easily adjust the contribution of specific lights. No special component or setup is necessary; all you need to do is change the **alpha** of your light's color:



For each light, the alpha of its color will get multiplied by the **Additional Light Contribution** value. For that reason, a suggested workflow would include having a fairly high value for the additional light contribution in your global or local fog, and control how much each light contributes to the final color of the fog through its color alpha value.

As of **HAZE 1.1.1**, the alpha value also affects the **Secondary Light Density Boost** of both global and local fog. That way you can adjust how each secondary light contributes to **both** the final color and the additional density.

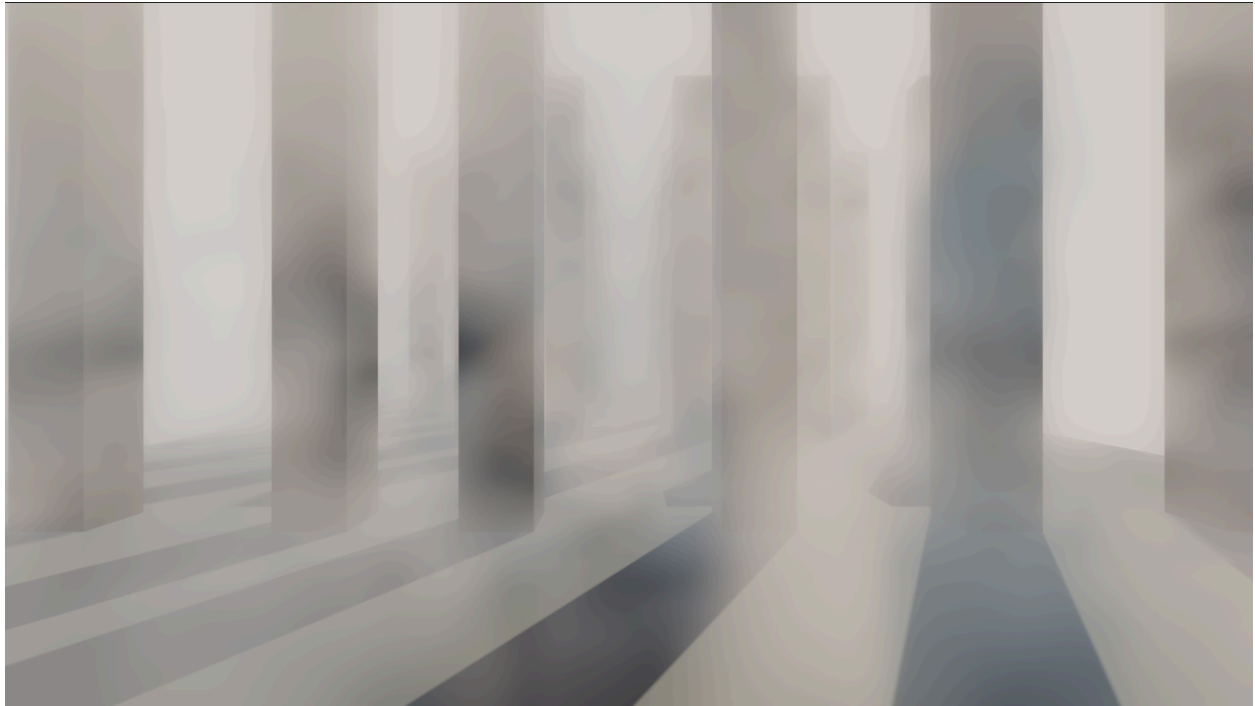
Multiple scattering

As mentioned above, settings for **Screen-Space Multiple Scattering (SSMS)** can be modified either via the **Haze Renderer Feature** or through the **Haze Settings Overrides** inside a Unity volume.

You can find a good example of the effect in the **MultipleScattering** demo scene inside the **Demo** folder.

The effect is essentially a separate bloom pass on the background image that also samples the froxel fog in order to adjust the blur scattering and make areas of higher density blurrier.

To better illustrate the effect, here's a screenshot with **SSMS enabled**:



When **SSMS** is **disabled**, the same scene looks like this:



Notice how the fog falls back to its ambient color at higher densities, instead of outputting a blurry version of the image. That's the difference between using **SSMS** and not.

With **SSMS** enabled, but with scattering set to 0, the scene looks like this:



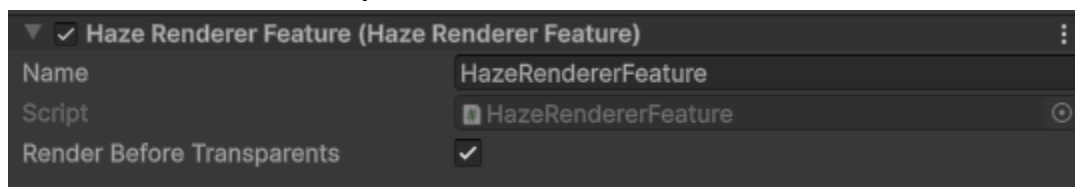
The fog is being composited with the image colors, but they're not blurred at all, so the result looks as if the fog isn't that dense.

Transparency support

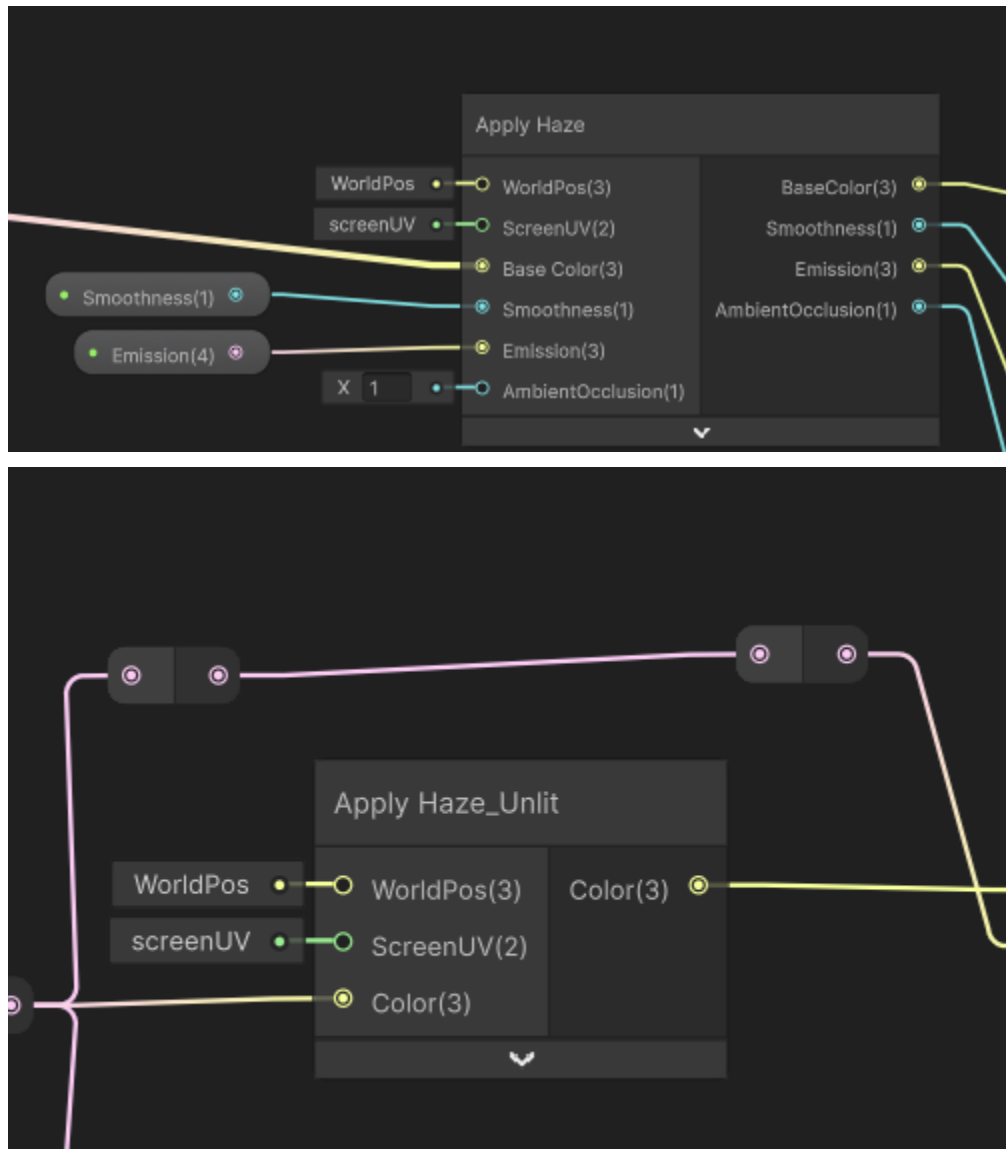
As of **HAZE 1.1.7** there is support for better handling of transparent objects. Due to the nature of the effect being applied as a post-processing effect, ordering transparent objects can be challenging since they don't write to the depth buffer.

Update 1.1.7 introduces the option to render **HAZE before** transparent objects so that the shader on the transparent objects can itself sample **HAZE** and apply it, separately from the post-processing effect. This is a process that requires a few steps:

First, **Render Before Transparents** needs to be checked on the **HAZE** renderer feature:



The transparent objects that we want to sample **HAZE** need to use a material with a shader that supports **sampling HAZE**. The project includes examples of transparent shaders made with Shader Graph that use the **Apply Haze** subgraphs:



There are 4 subgraphs included in the package:

- HAZE / ApplyHaze
- HAZE / ApplyHazeTricubic
- HAZE / ApplyHaze_Unlit
- HAZE / ApplyHazeTricubic_Unlit

The **Unlit** versions are used to apply HAZE just for the output color used in unlit shaders, while the other versions affect multiple PBR properties.

The **Tricubic** versions are used to sample the HAZE volume with tricubic interpolation. This is **significantly more expensive** but it's given as an option in case low-resolution artifacts on transparent objects are too intense.

For users that are more experienced with hand-written shaders, the **HazeFunctions.hlsl** file includes functions to apply haze in your custom HLSL shaders, if need be.

For transparent objects and effects that are not too deep in the fog, applying HAZE in their shader might not be necessary. These subgraphs and functions are included for objects in the far distance (like bodies of water) that stand out from the fog.

NOTE: For lit transparent shaders, make sure to disable “**Preserve Specular Lighting**” as this makes the objects stand out in the fog.

Performance considerations

HAZE utilizes a froxel-based approach that is rather lightweight and offers good results with small performance overhead in general.

However, if you are targeting low-end platforms, there are some performance considerations to keep in mind in order to achieve your framerate goals:

1. **SSMS** is probably the most expensive part of **Haze**. One might think that the volumetric fog is an expensive effect on its own, but the repeated application of blurring on large buffers can get expensive. If you’re considering releasing on low-end platforms, disabling the **SSMS** effects is probably a good idea.
2. Performance for the fog passes is directly linked to the **resolution and depth of the froxel buffers**. Reducing the **Froxel Buffer Resolution** and **Froxel Buffer Depth** should yield better performance overall. For lower depth resolution consider reducing the maximum distance of the froxel fog to make up for some of the lost resolution.
3. **Tricubic sampling** is really useful to reduce aliasing artifacts, but it comes at a slightly more increased cost due to the additional texture samples. Consider disabling it if you’re targeting low-end platforms.
4. The resolution of the **3D Noise Texture** can also affect the performance of the fog. Consider generating a smaller noise texture (or completely removing it) in order to save up more on performance.
5. As of version 1.1.5, density volume game objects marked as **static** do not update their bounds and data on every frame. This can result in improved CPU performance, so consider marking your volumes as static if their properties do not change on runtime.

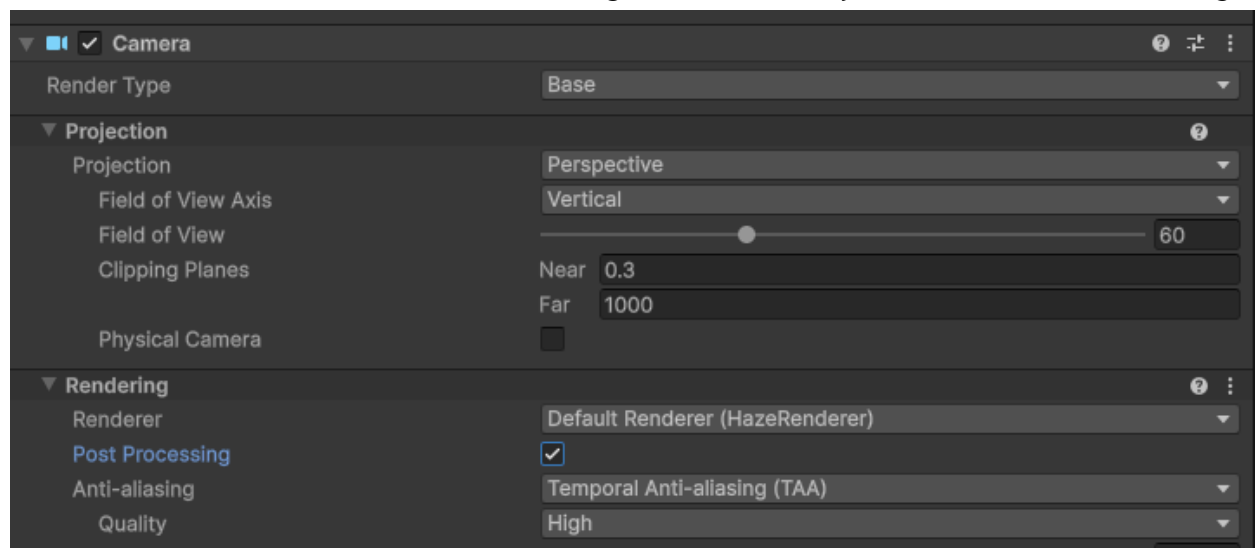
FAQ

Q: I added the **Haze Renderer Feature** in my renderer but I don’t see anything different in my scene. Why?

A: The **Haze Renderer Feature** adjusts the settings of the fog, but in order to see the fog in your scene you’ll have to either set up **Haze Global Fog** via a Unity volume or add a **Haze Density Volume** in your scene.

Q: HAZE is visible in the scene view but not in the main camera. Why?

A: Make sure to have **Post Processing Enabled** in your main camera settings:



Q: My scene view looks great with **HAZE** but the game view is not as high quality and has more flickering. Why?

A: This is due to temporal reprojection not working outside of playmode. In builds and once you enter playmode, the fog should look much cleaner, like in the scene view. This was a conscious decision based on the assumption that it would be more valuable to see a better representation of the final effect while creating a scene.

Q: Some light is shining through thinner objects. Can I do something about that?

A: This is, unfortunately, a known issue that stems from the sample jittering to allow for lower froxel depth resolution. Increasing the resolution of the **Froxel Buffer Depth** and reducing the fog's maximum range should help, but keep in mind that the former option will increase performance overhead.

Q: Can I change the noise properties (tiling/panning speed etc) on a local density volume?

A: As of now that's not possible. This was a conscious decision to save on performance, as modifying noise sampling per-density volume would require re-sampling the noise texture which would come at additional performance cost. If it's a much-requested feature it can potentially be added in future updates.