UGUI Blurred Background

Table of contents

Requirements **2**
Manual Installation 2
Creating a blurred background image **4**
Settings **5**
Strength 6
Quality 7
Iterations 8
Resolution 9
Color 10
Use Custom Mesh 11
How to Blurr UI that is behind a blurred image **12**
Camera and World space limitations **13**
Distortions (in world space images) 14
Foreground content in blurred image (in world psace images) 17
Frequently Asked Questions **18**
All the blurred UIs have the same blur settings applied. 18
The blur does not update or align properly in the scene view. 18
If „Blur Strength" or „Blur Iterations" is set to 0 then the background image vanishes. 18
How do I make a white tint? 19
Blurred UI over other UI does not show the UI behind. 21
It works in the editor but not in build 21

When I change the color space from „linear“ to „gamma“ the blurred image is too bright or very dark / inverted or not blurred....22
The blurred image is upside down if multiple cameras are used....23
There are some objects created in my scene („UGUI Blur Manager.“, „UGUI BlurredBackground Custom Pass..“)....23
Transparent objects are NOT visible in „Camera“ or „World“ space canvases....24
Is the URP 2D Renderer Supported?....25
The blur image is empty (nothing is blurred). Why?....26
In my HDRP / URP project the blurred image is just a white square?....27
My UI is flickering wildly in HDRP?!?....28
The blur is not working in Unity 6 (using render graph)....28

Requirements

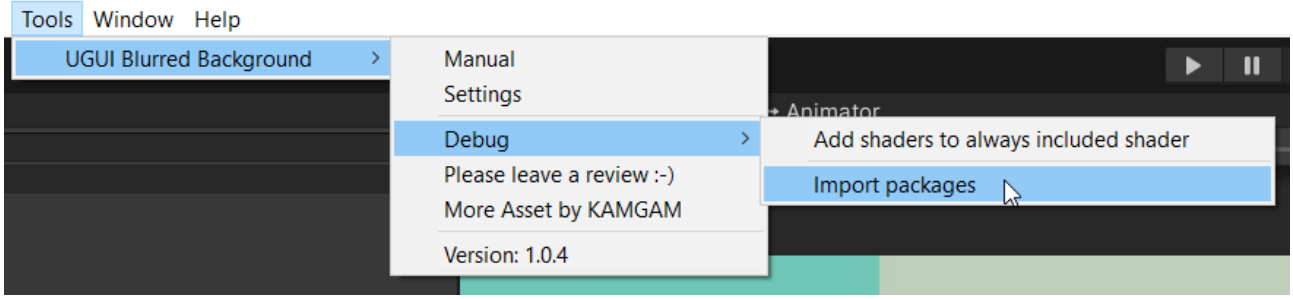
Unity 2021.2 or higher is required. Though it may work in earlier versions too (they have not been tested).

You are using a canvas (UGUI). If you need a blurred background for UI Toolkit then please check out the dedicated asset „UI Toolkit Blurred Background“ for that.

Manual Installation

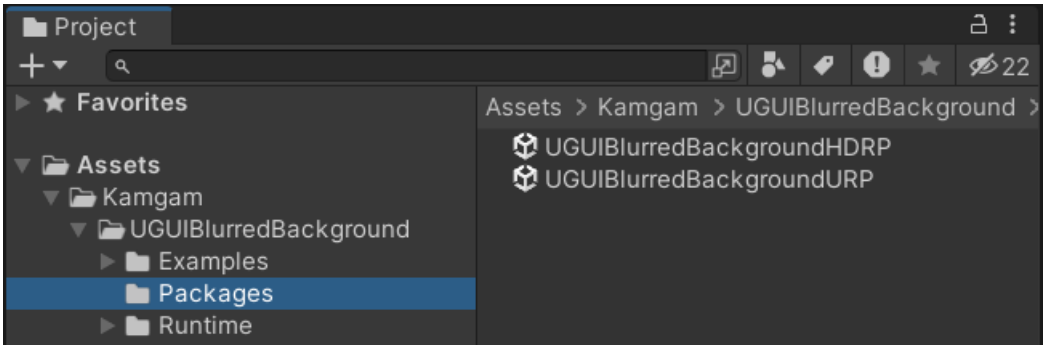
Usually this is not necessary because there is an automated post-install process. However I have received reports from users that this sometimes does not work so here is a manual guide on how to do the post-install steps:

1) Trigger the import of the package

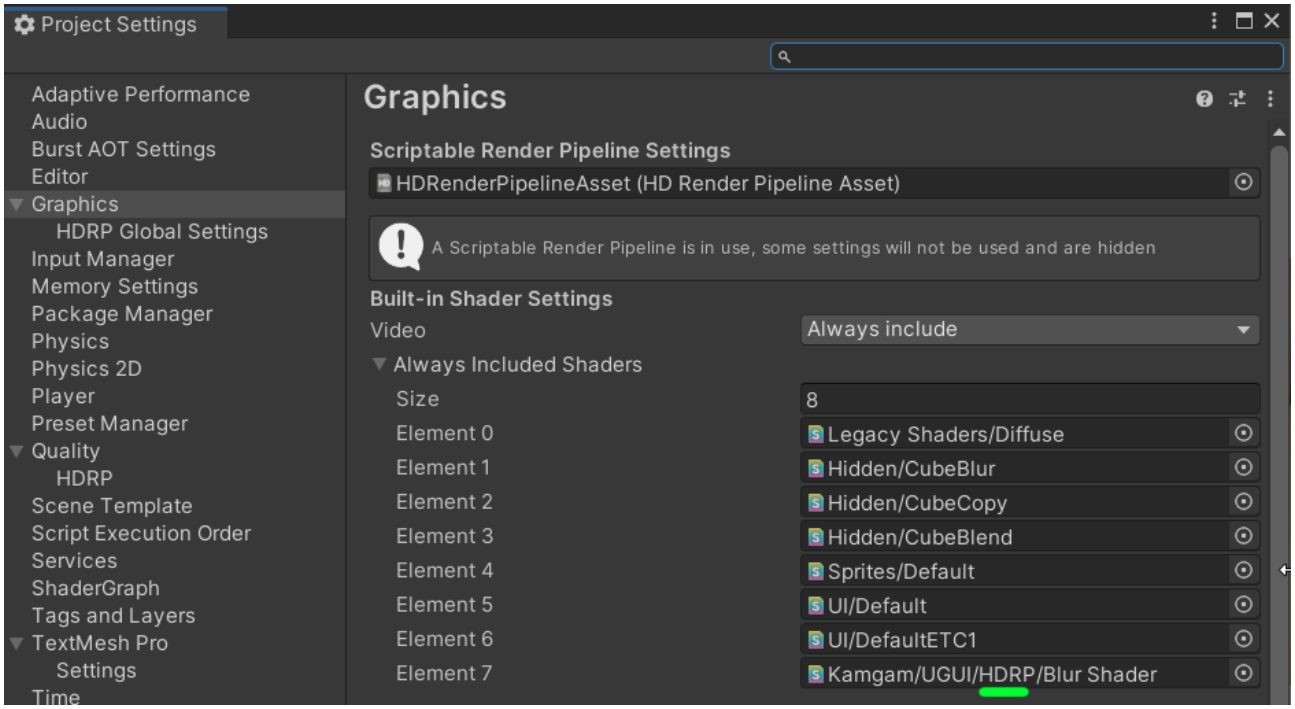


This option detects what render pipleline you are using and imports the appropriate package.

[OPTIONAL] If you want to be extra sure you can also import the package manually. They are located under „Assets/Kamgam/UGUIBlurredBackground/Packages“:



2) After the import you should be able to replace the „BuiltIn“ version of the shader with the correct one (HDRP in this example):

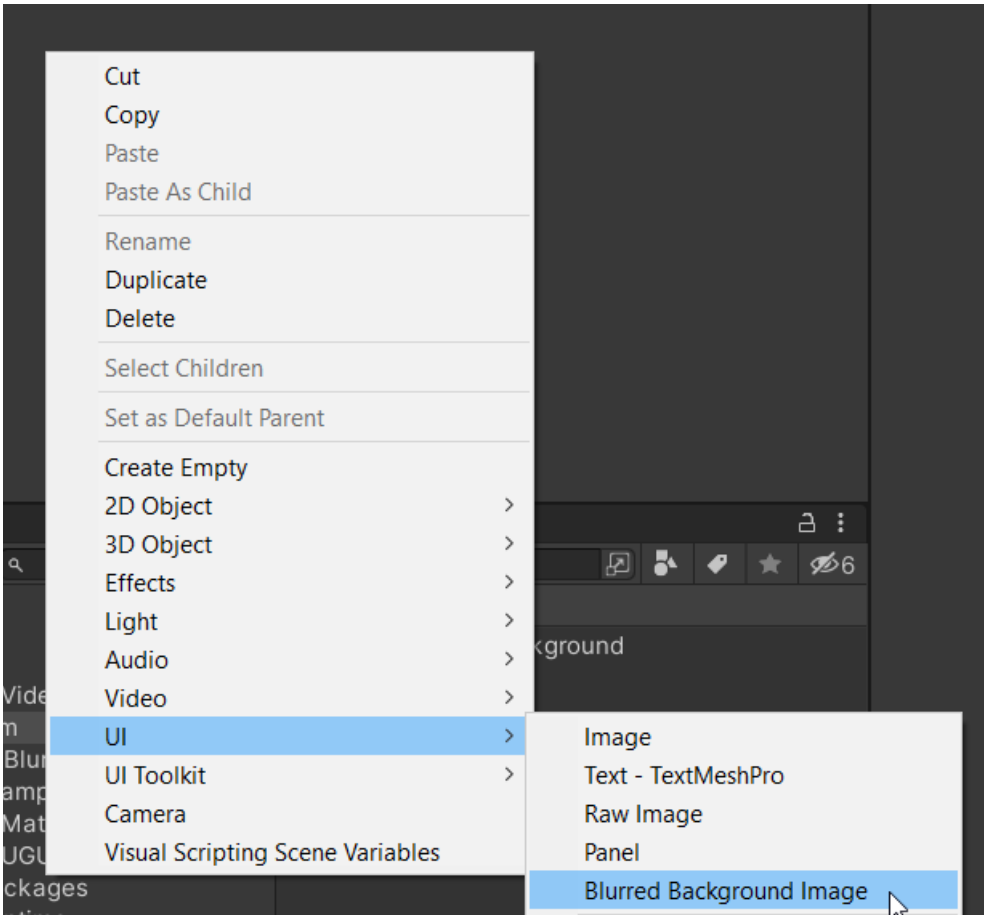


And that's it.

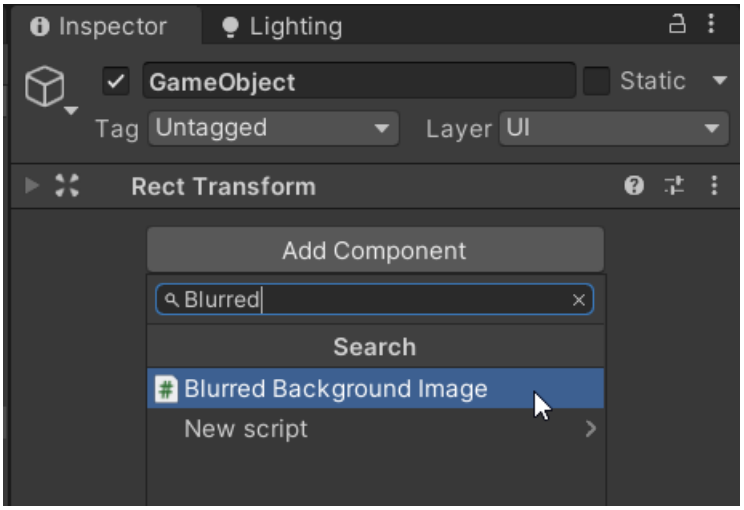
If you are having troubles with the setup then please don't hesitate to write to office@kamgam.com

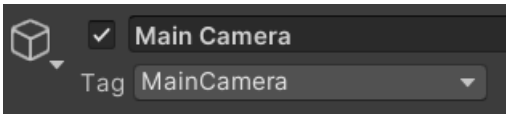
Creating a blurred background image

In the Hierarchy window **Right-Click > UI > Blurred Background Image**. It will add a new blurred image to the first canvas it finds in the scene.



You can also add it to an existing UI element via „Add Component“:





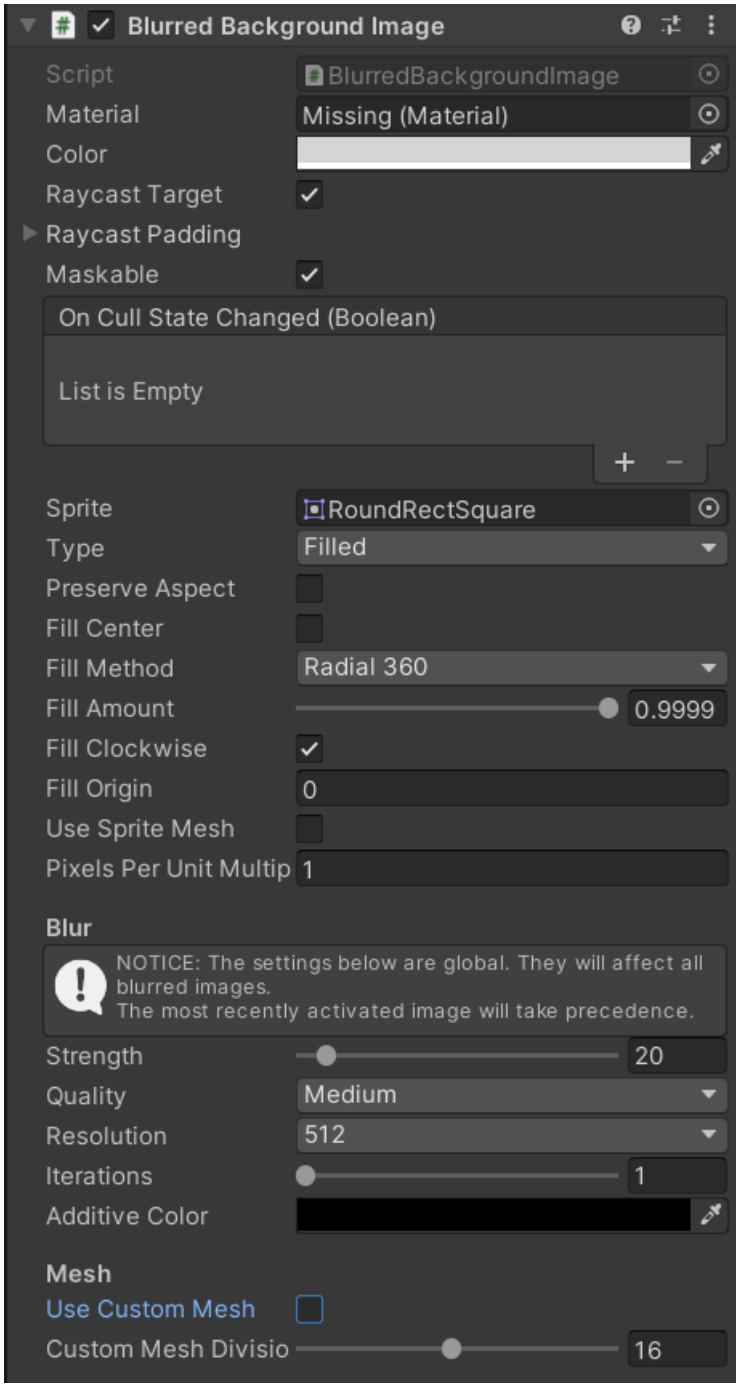
NOTICE: The blur renderer always uses the camera with the „MainCamera“ tag. Please make sure you have your camera tagged accordingly.

Settings

The blurred images can be controlled via the inspector.

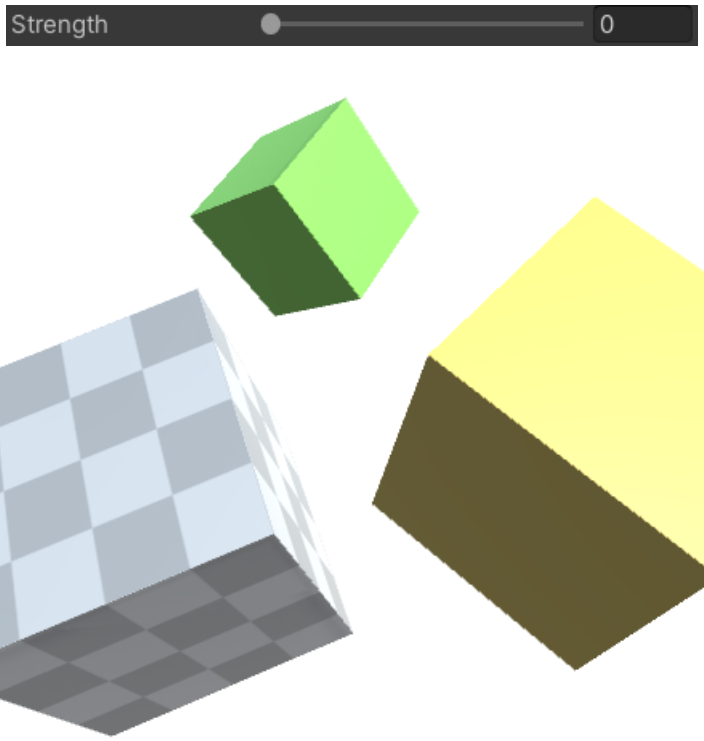
⚠ At the moment there is only **one global blur setting**. This means that all blurred images will use the same blur settings. - If you need more please let me know. If demand is high enough it will be added.

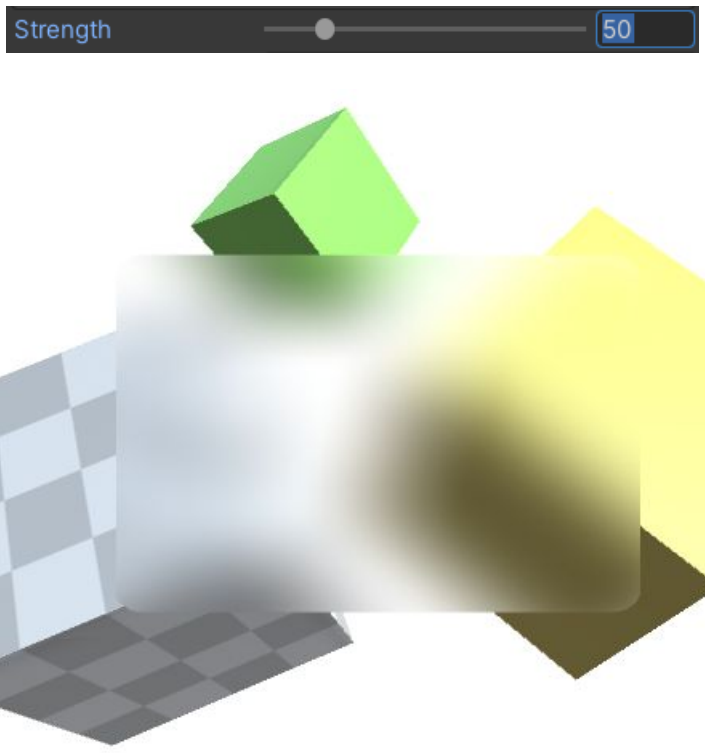
Most of the options are similar to the normal image (it extends the normal image).



Strength

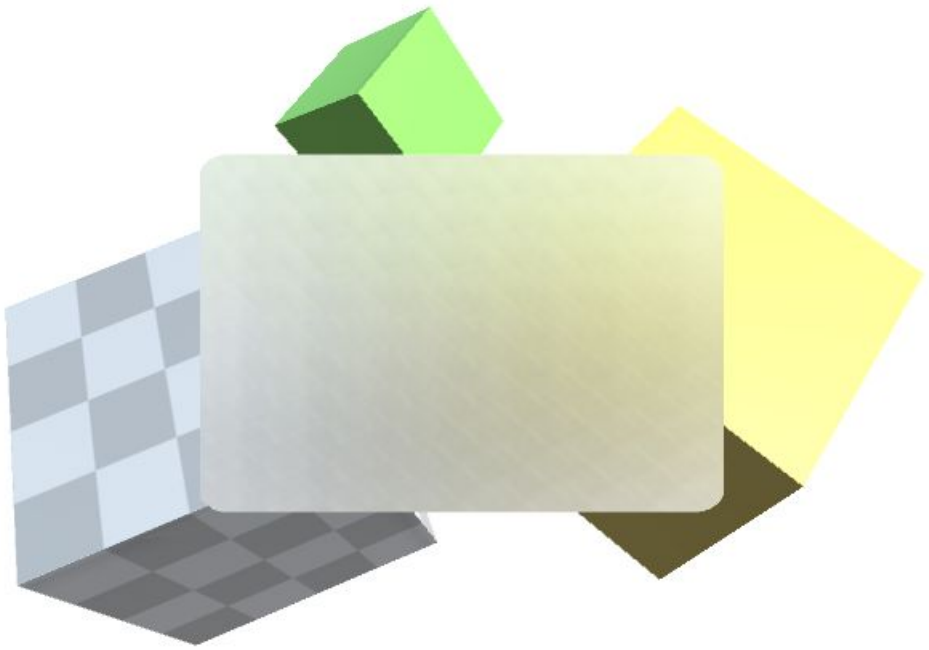
Defines how strong the blur effect will be. Notice that the quality may degrade more the higher the strength is (more on that below).

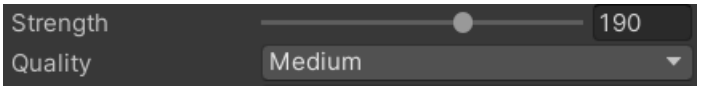


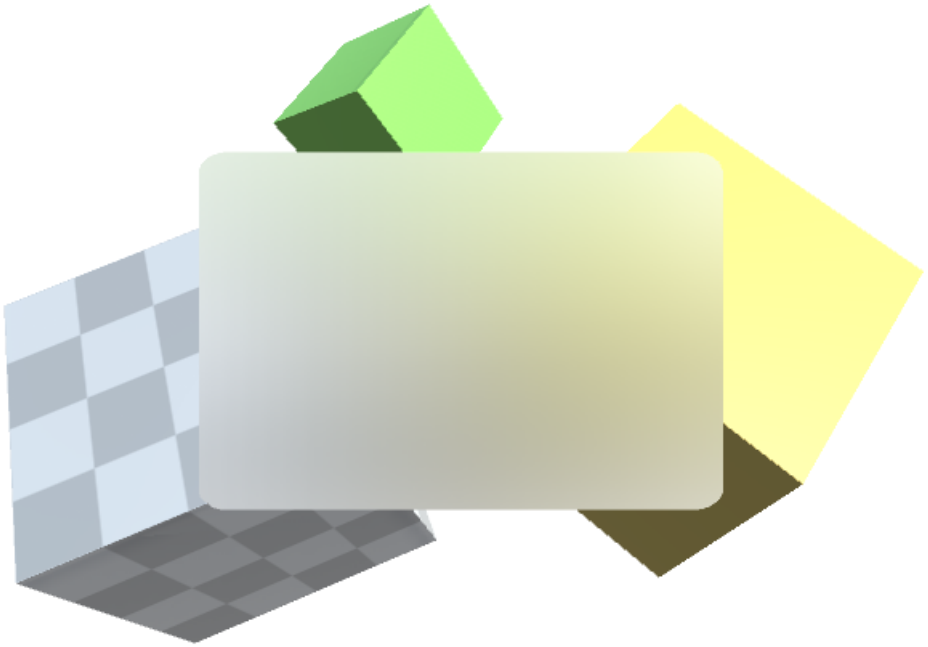


Quality

If high blur strengths are used then you may notice visible artefacts. To avoid these increase the quality. NOTICE: The higher the quality the more performance it will cost. As a rule of thumb (low = a cost of 1x, medium = a cost of 3x, high = a cost of 10x).

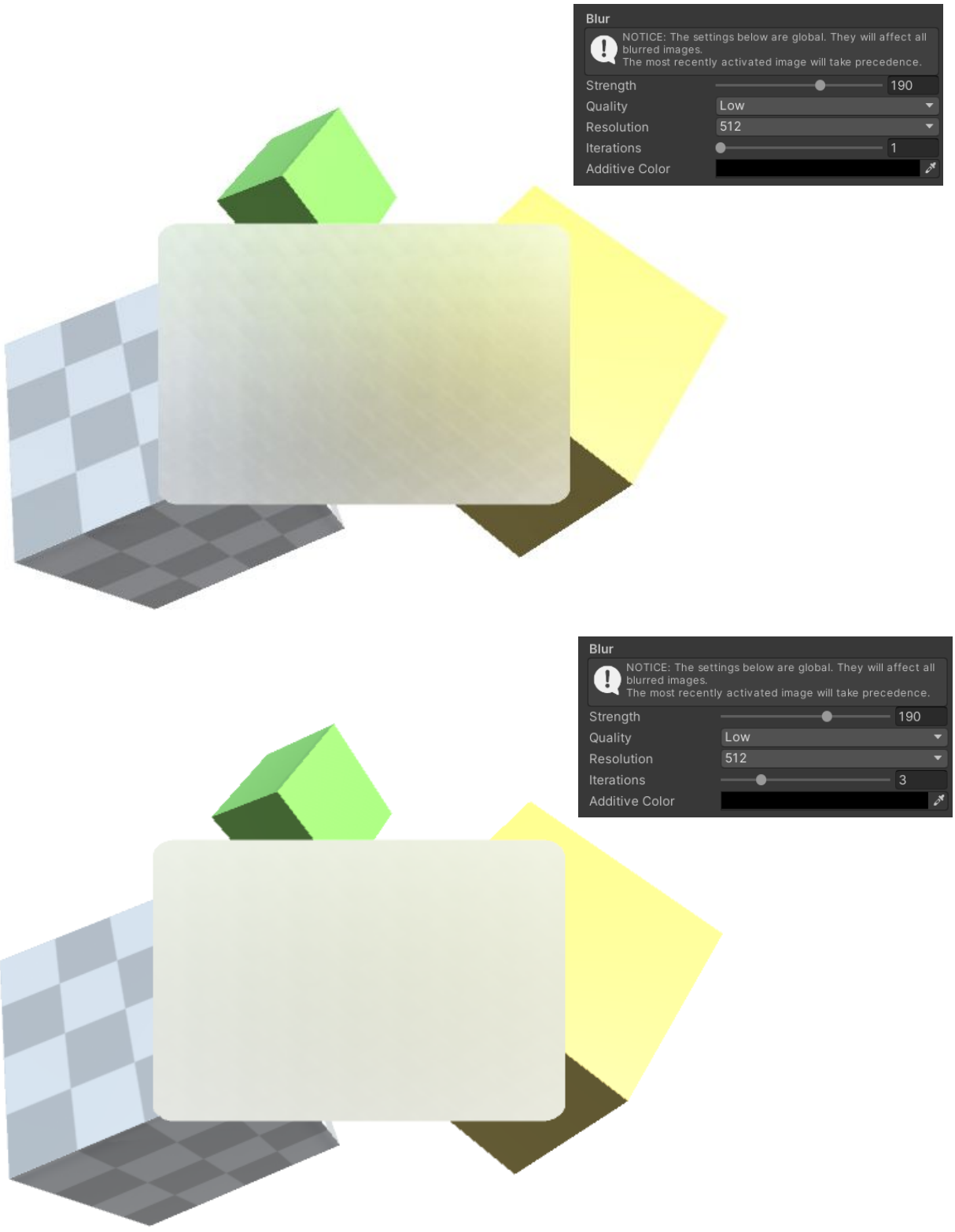






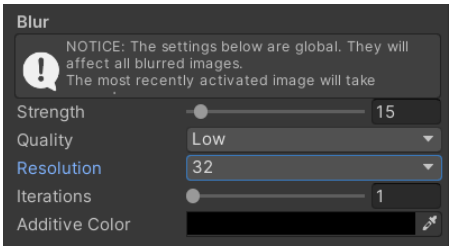
Iterations

Blur iterations should be kept at 1. This defines how often the blur filter will be applied. In terms of performance this the most expensive setting you can increase. Use with care (avoid if you can).



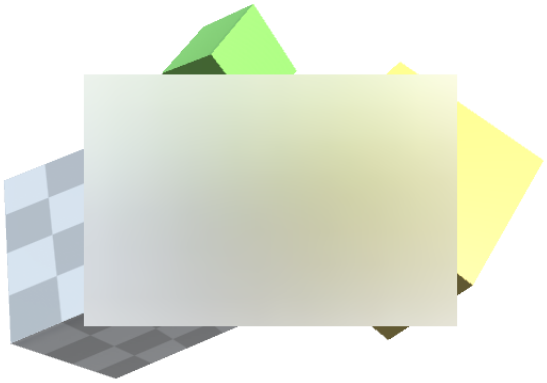
Resolution

Reducing the resolution is a great way to increase the blurryness of your image while also saving a LOT of performance. Halfing the resolution usually makes the blur 4 times faster.

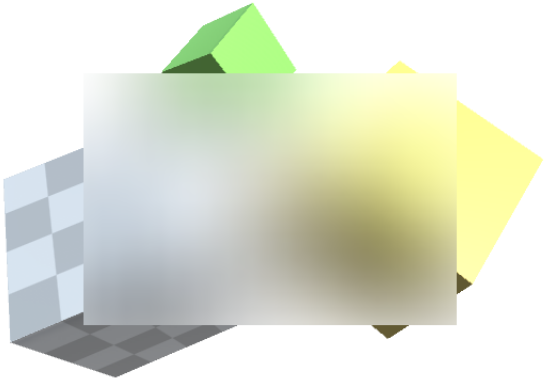


Using a low resolution can also allow you to get away with the quality set to low.

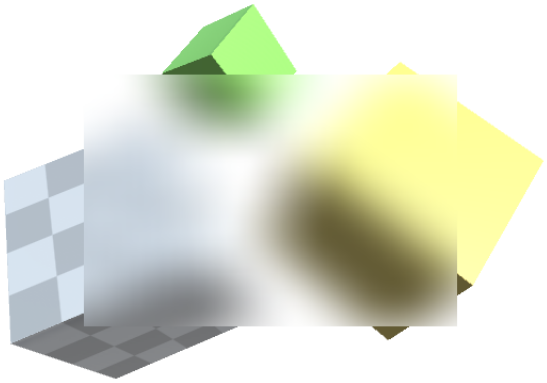
32 x 32



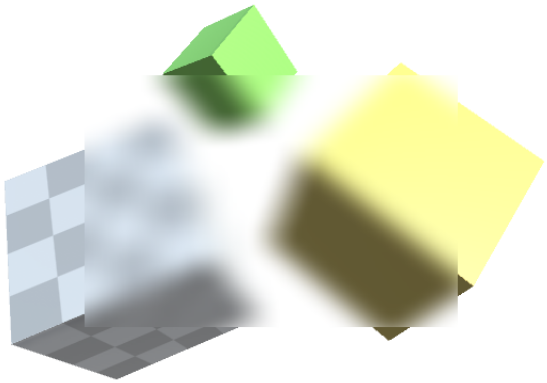
64 x 64



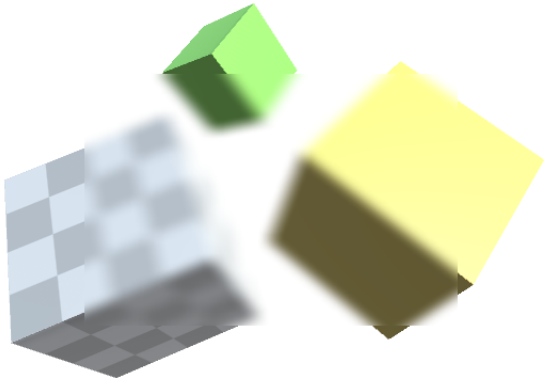
128 x 128



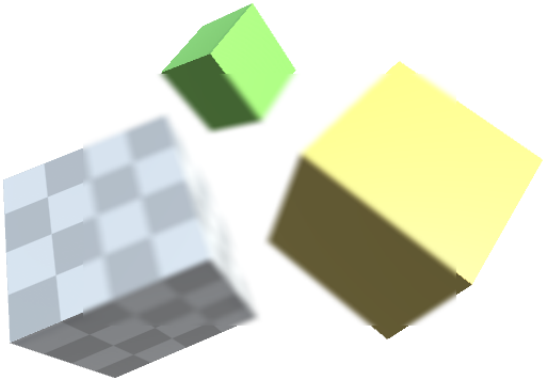
256 x 256



512 x 512

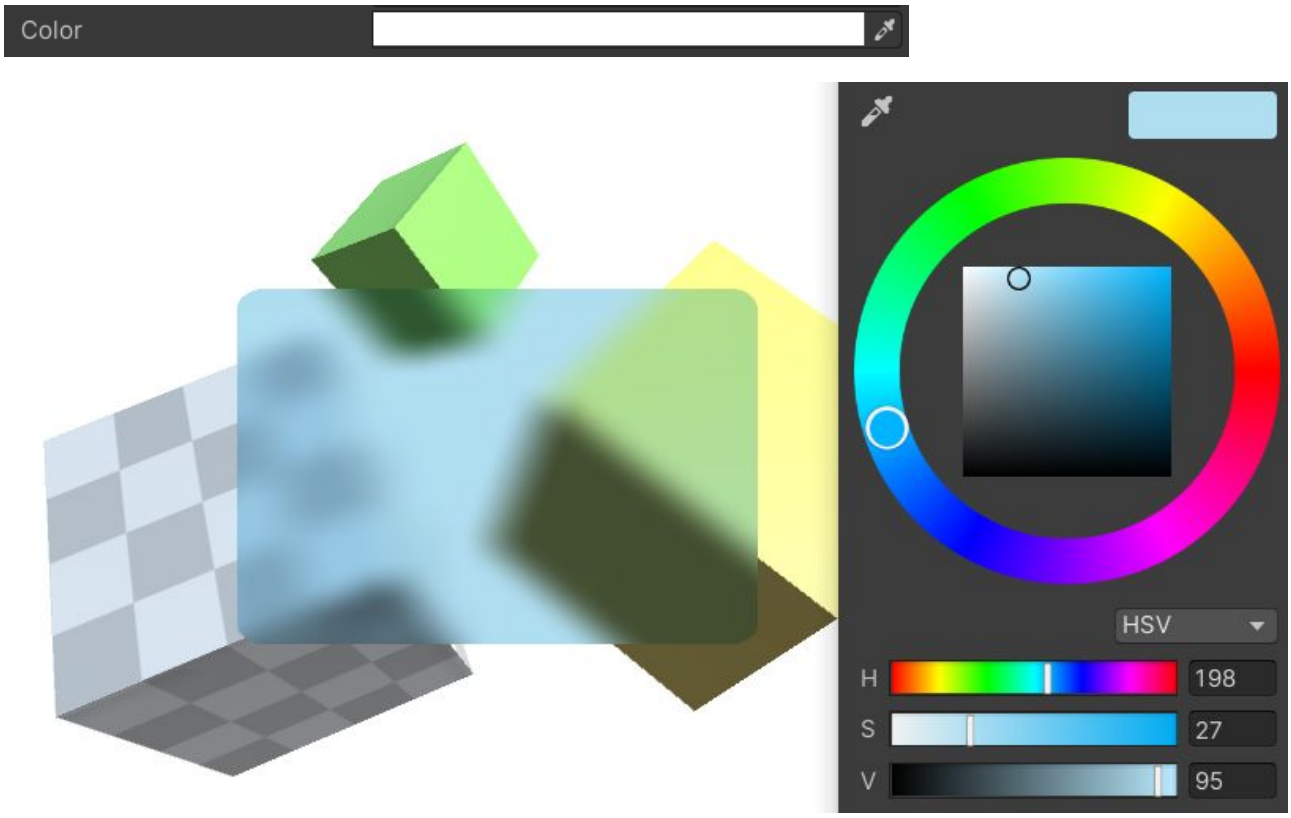


1024 x 1024



Color

This work just like a normal image color does. It tints the image.



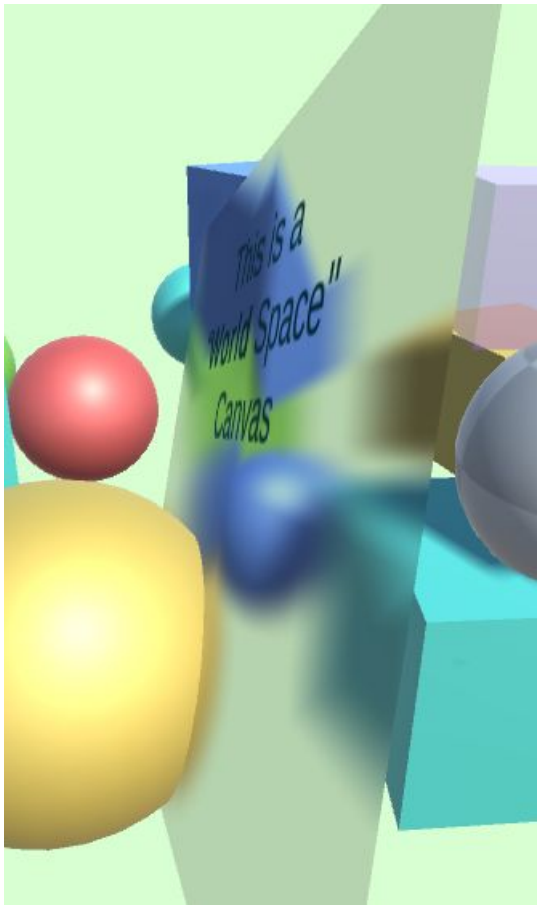
You may have noticed that with this you can tint and darken the image but not brighten it.

If you want to achieve a milky, glass like overlay please refer to the „How do I make a white tint?“ in the FAQs below.

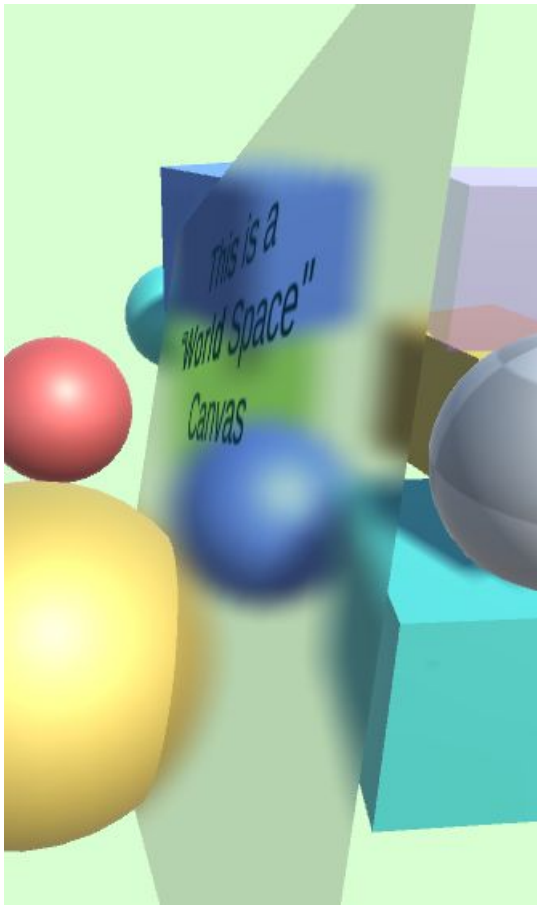
Use Custom Mesh

If you enable this you will not be able to use any of the regular image options as then the image mesh will always be a quad with N subdivisions. This exists to work around some of the limitiations of „world“ and „camera“ space canvases (see section „Camera and World space limitations“ below).

A highly rotated world space canvas (notice the distortions).



A custom mesh world space canvas (distortions minimized).



How to Blurr UI that is behind a blurred image

Please notice that for UIs that only use Screen-Space-Overlay canvases this is not really possible.

What you can do is make all the blurred images in a single canvas include all the UI behind (aka modal dialogs are okay). So basically you have to split your UI into two canvases:

BACK: Stuff that's behind the blurred image (your normal UI).

FRONT: Stuff that's in front of your blurred image (your modal dialog for example).

To achieve that you have to use a special setup that fullfills certain conditions:

1. BACK and FRONT have to be on separate canvases
2. BACK has to be either a WORLD-Space or a CAMERA-Space canvas. It must not be a SCREEN-Space canvas.
3. The rendering camera of the WORLD/CAMERA-Space canvas has to be the camera that is tagged with CAMERA_MAIN.

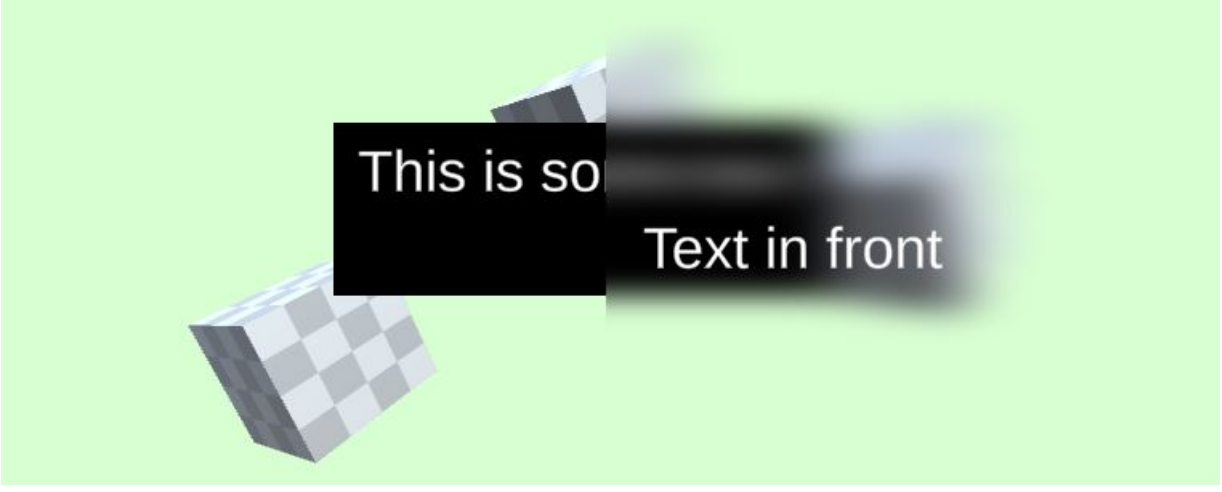
There are some demo scenes under Assets/Kamgam/UIBlurredBackground/Examples/:
* UGUIBlurredBackgroundUIOverUI-Demo
* UGUIBlurredBackgroundUIOverUI-Layers-Demo (Built-In only: URP and HDRP require a different setup, see camera stacking)

Both of them use one Screen-Space-Overlay (BACK) and one Screen-Space-Camera (FRONT) canvas to achieve the UI over UI blur effect.

NOTICE: Using a CAMERA-Space canvas has some drawbacks:

- Your UI is affected by the post processing effects of the rendering camera. You can work around that by using layers (see „UGUIBlurredBackgroundUIOverUI-Layers-Demo" example). HINT: In URP you can disable post processing on the camera.
- Your world may clip into your UI.

NOTICE the use of the „Camera Override" in the layers BiRP example.

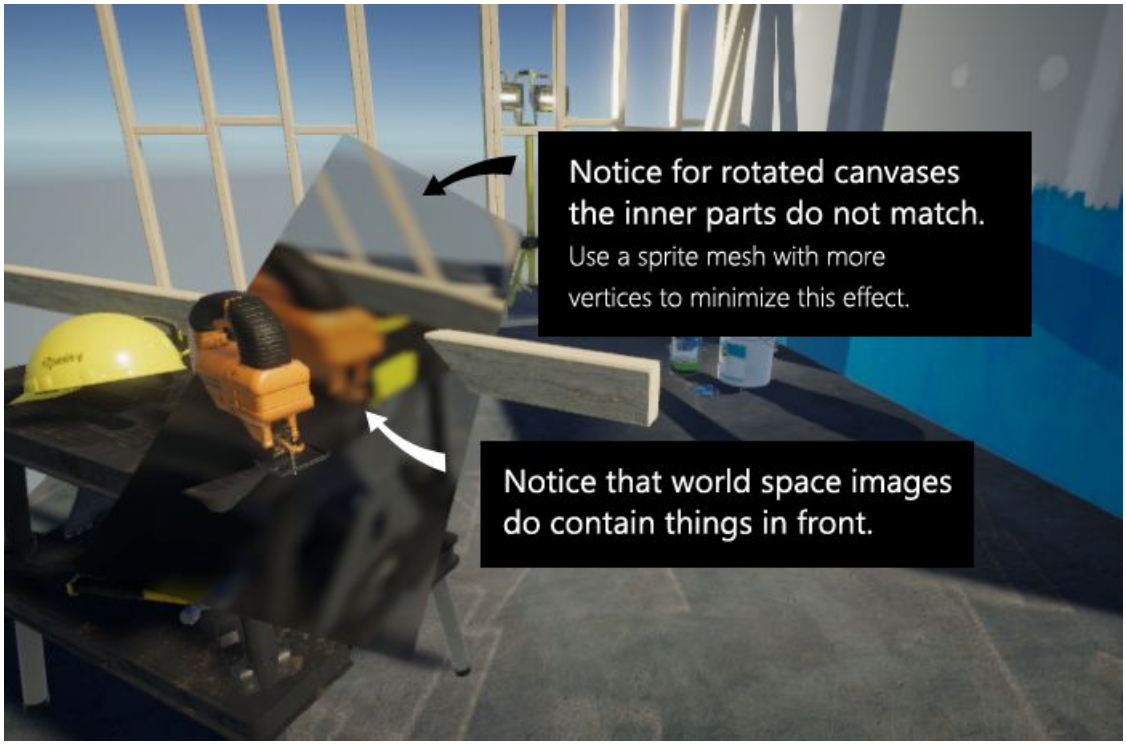


Camera and World space limitations

There are some limitations when using the blurred image on camera and/or world space canvases.

A) Distortions

B) Foreground content in blurred image.



The sections below will explain these limitations in detail and how to work around them if possible.

Distortions (in world space images)

The interpolation on the pixels inside vertices is done by the default UI shader. That shader assumes that it needs to do perspective distortion on the texture.

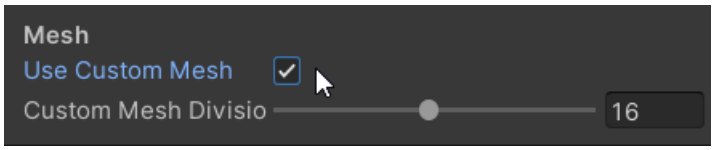
However, the texture (blurred screen) we are using was already rendered in perspective and thus it will be distorted. This becomes especially visible if the image is rotated a lot in the z direction.

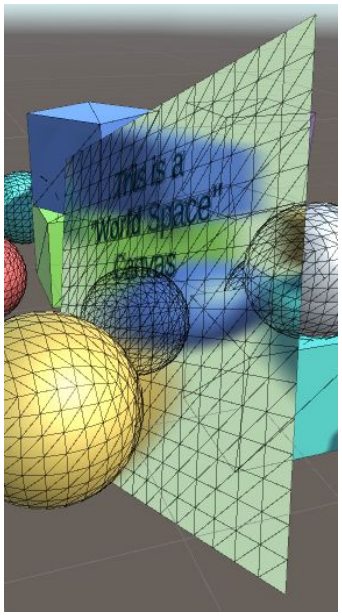
The solution to this would be to make a separate shader for the in-world canvases or patch the texture (rotate) to match the world object leading to one texture per object (expensive). I opted not to do any of those.

Instead the fastest solution in terms of performance is to add additional vertices to the mesh to reduce the distortion effect as much as possible.

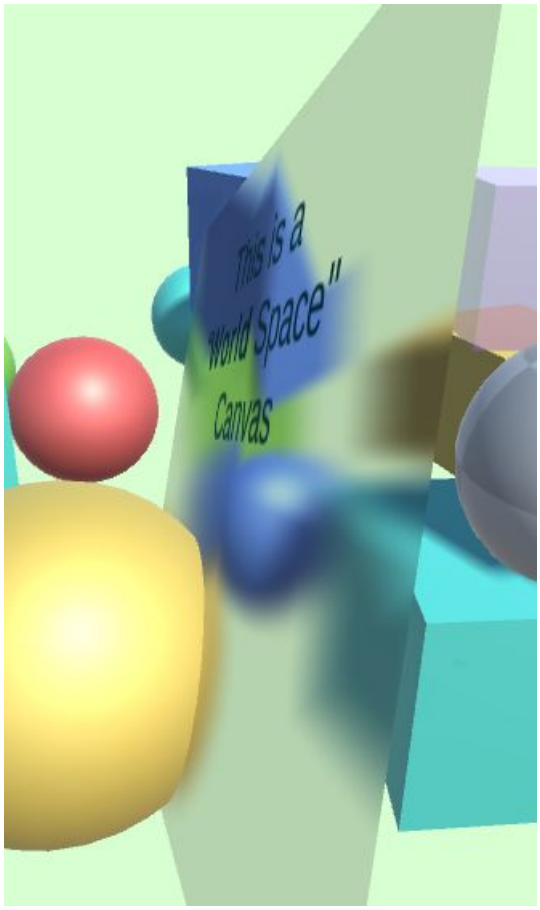
One workaround is to enable the „Custom Mesh“ option.

If you enable this you will not be able to use any of the regular image options as then the image mesh will always be a quad with N subdivisions.

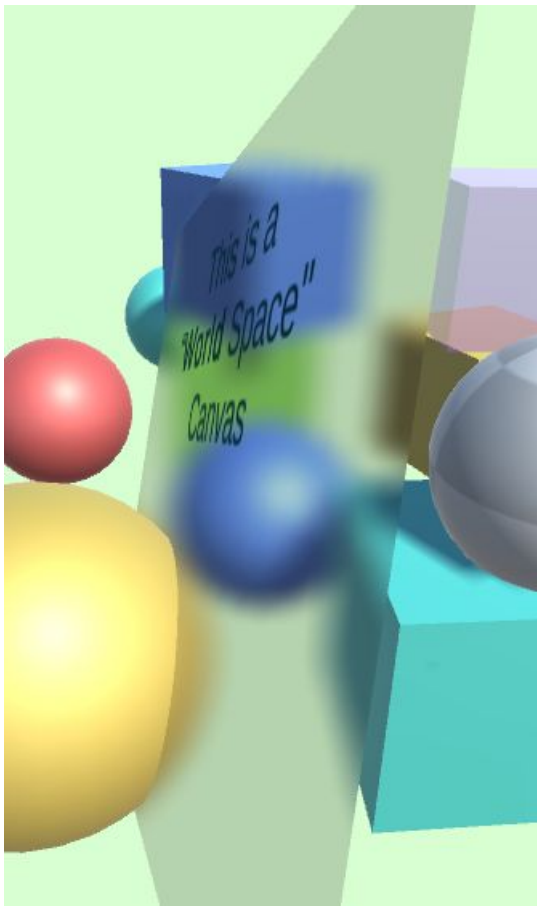




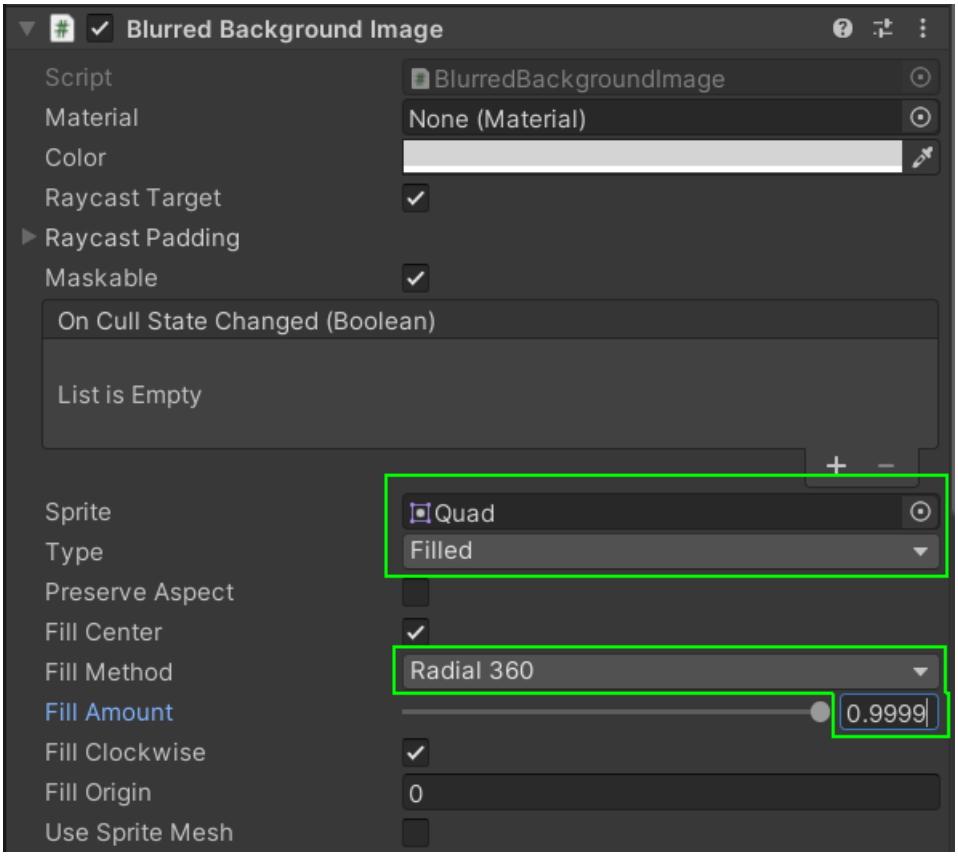
A highly rotated world space canvas (notice the distortions).



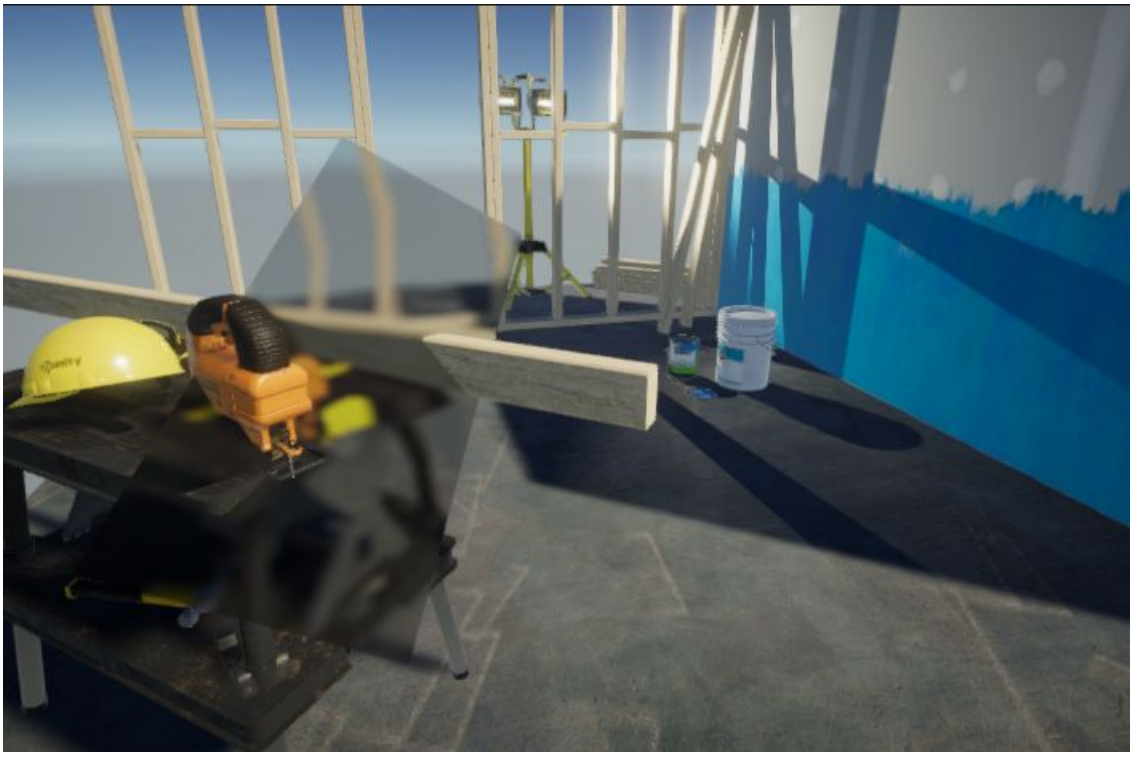
A custom mesh world space canvas (distortions minimized).



Another quick workaround that works for canvases without custom sprite shapes is to introduce a new vertex in the center. You can do this by using „fill“ with 0.999



This introduces a new vertex in the center and keeps the gap invisible. With the additional vertex the distorition is much less visible.

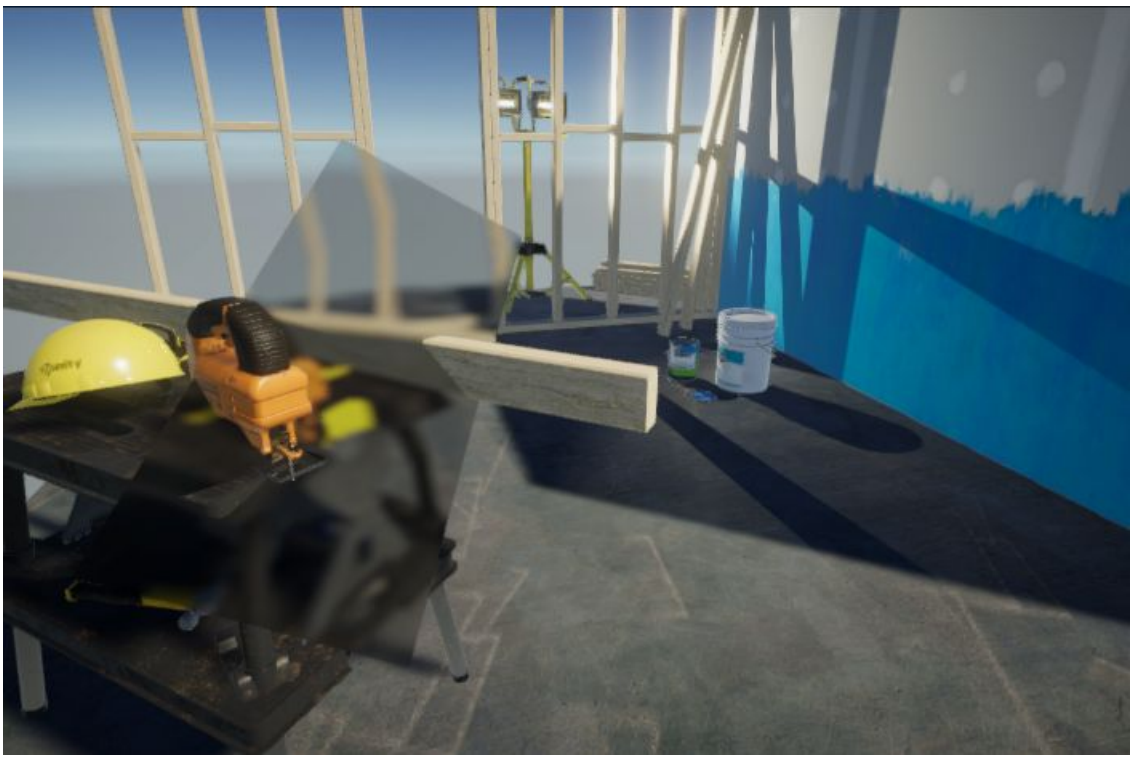


Foreground content in blurred image (in world psace images)

This is because the blurred image is taken AFTER rendering the whole scene. In order for front objects to be hidden in the blurred image we would have to introduce a separate step for each world space image (render the scene without the foreground objects, that's very slow) or add a custom UI shader that blurs all the background. I have not yet written such a custom shader. If you really need it please let me know. If demand is high enough I may add it.

For now all blurred images contain all the scene objects.

Usually this should not be very noticable if the distortions are kept in check.



Frequently Asked Questions

Here are some common issues that have been reported.

If you can, please upgrade to the highest LTS version of Unity. The newer the version the better.

All the blurred UIs have the same blur settings applied.

AKA: When I activate a new blurred image the other images will suddenly switch to the same blur settings?!?

Yes, that is done on purpose to save performance. The blur effect is generated by extracting the rendered image and blurring it. To support multiple blur factors at once it would have to be done multiple times, which means double or tripple the cost (basically one for each settings combination). This escalates quickly (exponential growth) and reduces the performance quite a bit. Thus it is not (yet) supported.

If you need multiple different blurs within one UI then please contact support. It will be added if demand is high enough.

The blur does not update or align properly in the scene view.

The blur is always rendered from the perspective of the camera in the game view. Don't worry it will be okay in play mode and in builds.

If „Blur Strength" or „Blur Iterations" is set to 0 then the background image vanishes.

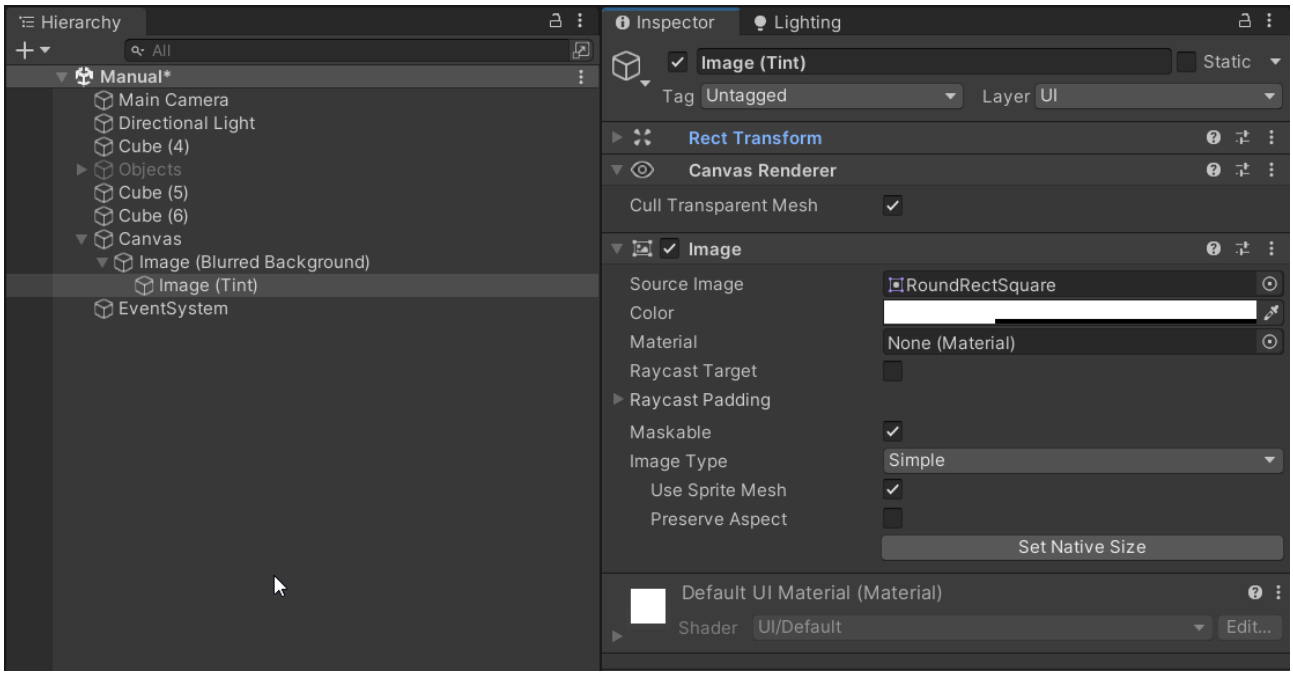
Having either of these set to 0 would result in no blur at all and thus we do not even generate the mesh for the background. That is done to save performance.

How do I make a white tint?

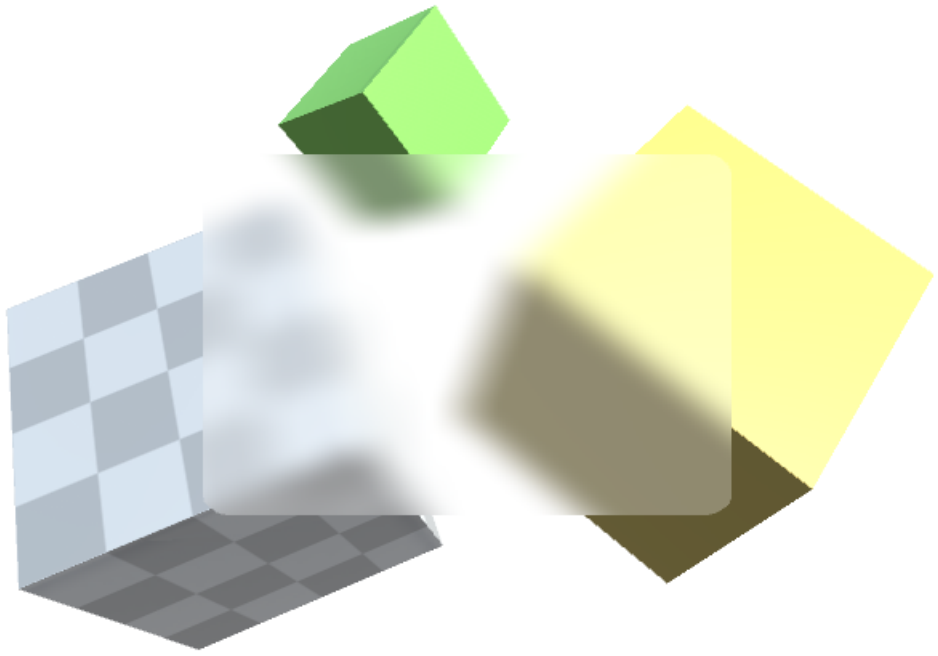
You will have two options:
Option A (add a second image) will work for each image individually.
Option B (using AdditiveColor) will add the color to ALL blurred images.

Option A:

Add another image inside the blurred image it with reduced alpha (30%).

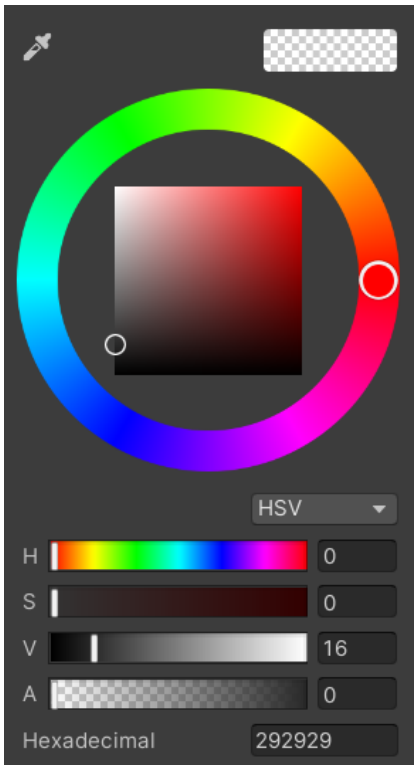


This is how it will look like:

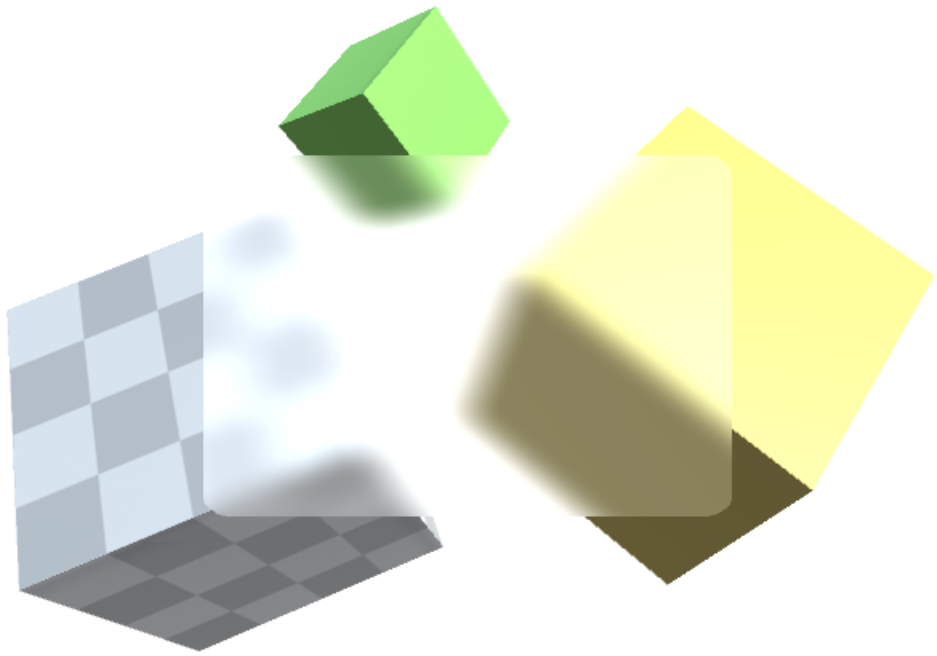


Option B:

You can use the Additive Color with a very dark grey to brighten the image. Remember this is equal to an ADDITIVE effect (the color you choose is added to the image) and it will affect ALL images.



This is how it will look like:



Blurred UI over other UI does not show the UI behind.

Yes, by default that's on purpose. Screen Space Overlay canvases are always rendered last and thus will render the whole scene (including Camera-Space or World-Space canvases).

NOTICE: World and Camera space canvases will only render the opaque parts of the scene as they themselves are part of the transparent queue.

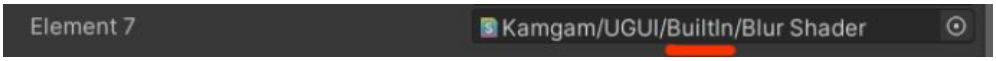
Please refer to the „How to Blurr UI that is behind a blurred image" section above for more details. It can be done if you mix Camera-Space and Screen-Space canvases.

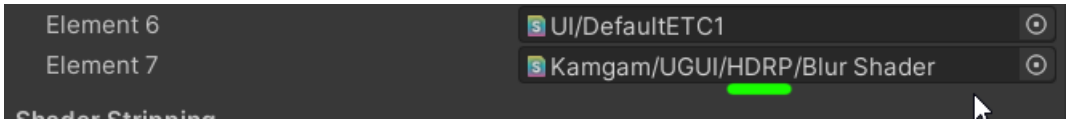
It works in the editor but not in build

Maybe you have reset the „Always included shaders". Since the blur effect is a graphics effect it is run on the graphics card and requires a shader.

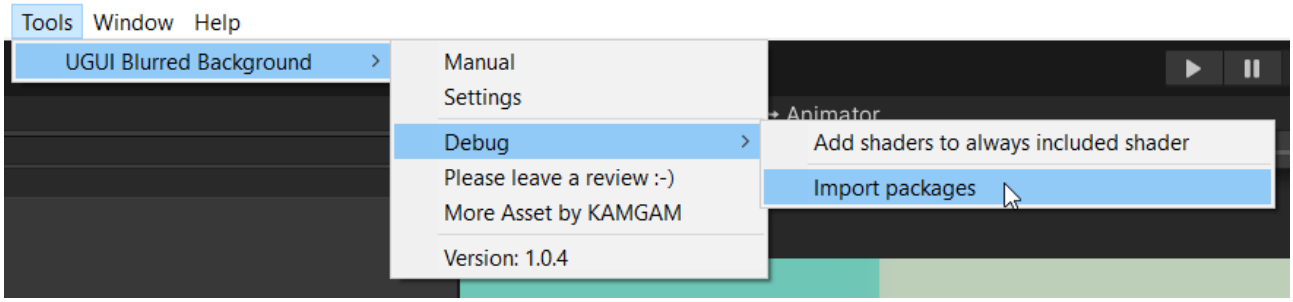
Usually the shaders are added to the „Always included" list at the start. Please check if they are added and if they are using the correct render pipeline (BuiltIn, URP or HDRP).

For example if your are using HDRP then make sure the HDRP shader is in the list. Here the Built-In shader is in the list and needs to be replace wit the HDRP shader:



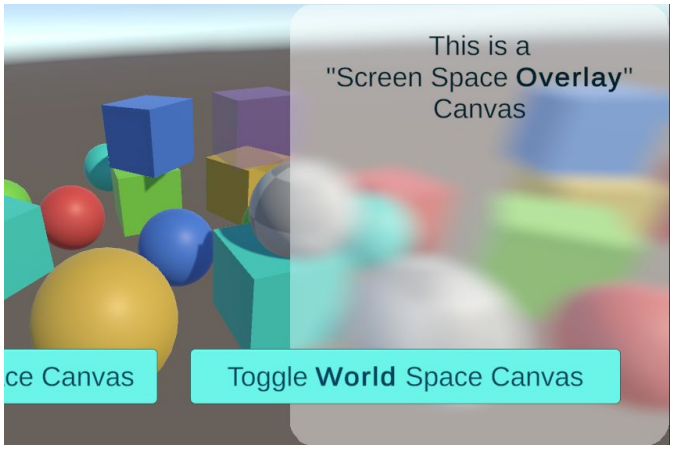


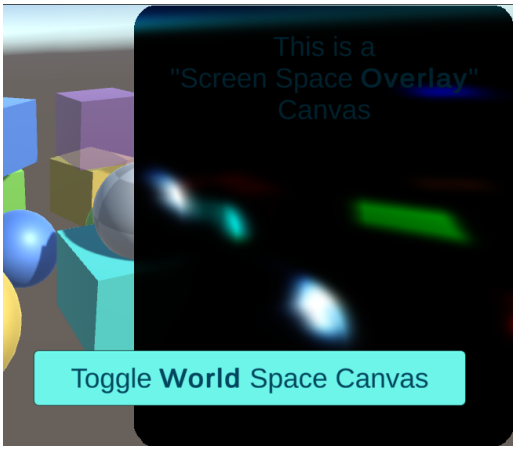
If the shader you need is not available then you may have to re-import the files like this:



When I change the color space from „linear“ to „gamma“ the blurred image is too bright or very dark / inverted or not blurred.

Please press the play-mode button once to reinitialize all render caches of the blurred image. It should workd fine afterwards. You only have to do this once.

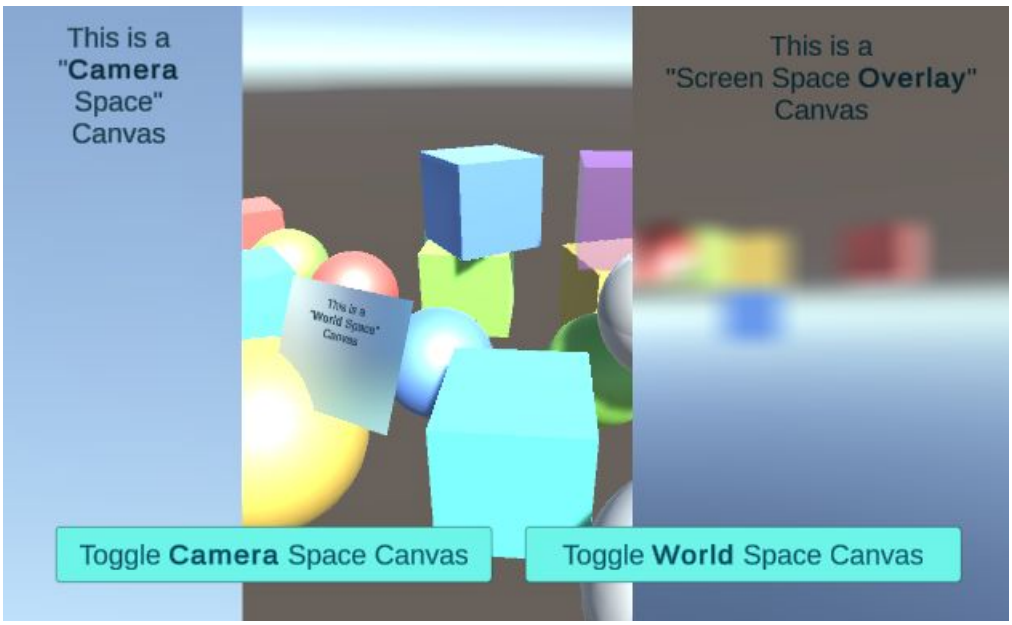




If you have „Domain Reload“ turned off then you may have to restart Unity (or turn it on and hit play once).

The blurred image is upside down if multiple cameras are used.

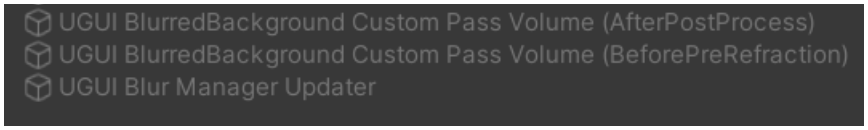
This happens if you have multiple cameras with the „MainCamera" tag active. In this case the blur renderer does not know what camera to use and gets confused. You should only ever have one camera with the „MainCamera" tag active at the same time. If you only have one camera active then the blur will return to rendering that one camera.



There are some objects created in my scene („UGUI Blur Manager..", „UGUI BlurredBackground Custom Pass..")

These are some game objects necessary for rendering. Please just ignore them. They are created automatically and they are NOT saved in your scene (that's why you can not edit them).

Do not delete them manually. If you did on acctident then hit PLAY once and they will be recreated.

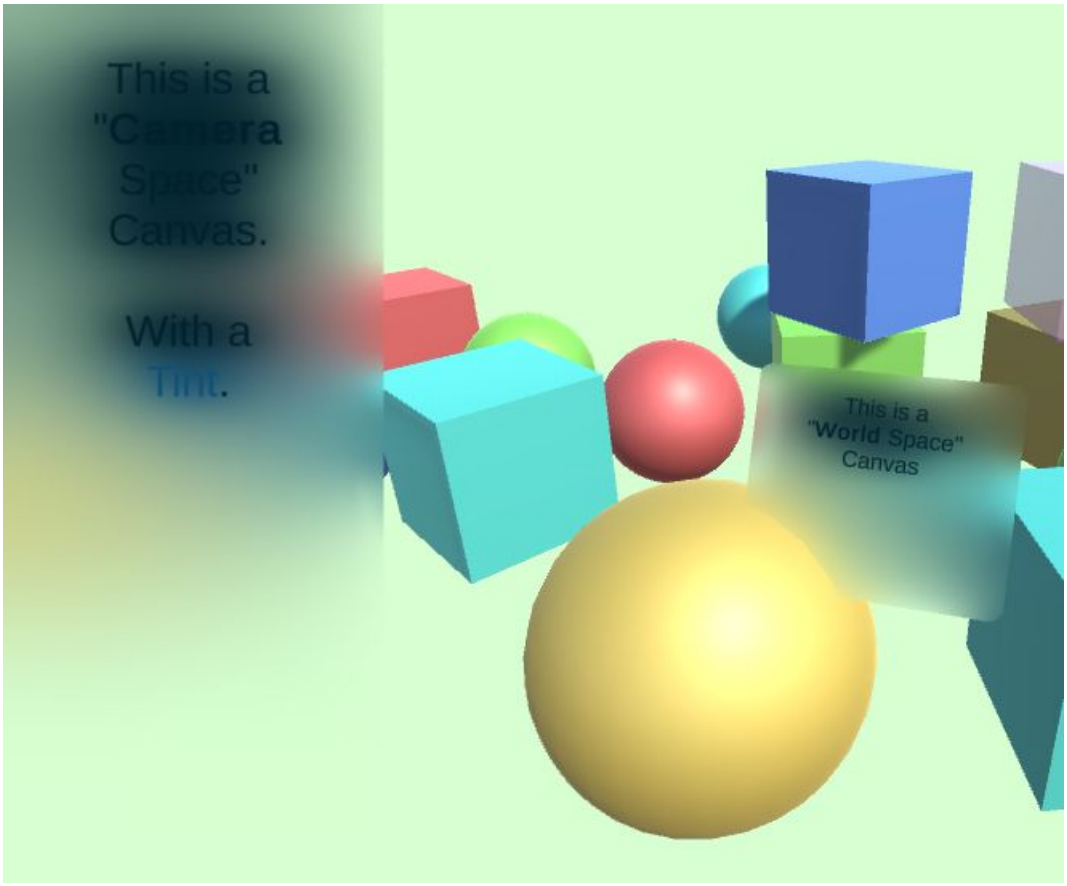


Transparent objects are NOT visible in „Camera“ or „World“ space canvases.

Yes, this is by design.

Sadly this is necessary since the UI itself is considered a „transparent“ object in the render pipeline if it is in WORLD or CAMERA space.

This means if we were to render transparent objects too then the blurred UI itself would be rendered too and then blurred (again) and then rendered again and blurred again, … . It would look like the image below (I have tried it ;-)



That’s why the blur source image for WORLD and CAMERA canvases excludes all transparent objects. If you look in the demo scene then you can see this on the purple cube. It is a transparent object.

In Screen Space OVERLAY canvases transparent objects are shown as that is rendered much later.

Is the URP 2D Renderer Supported?

Yes, but also no. The thing is in the 2D renderer there is no transparent queue and thus WORLD and CAMERA space canvases are out of a question with a single camera.

Screen Space OVERLAY canvases work, though there is a major caveat. There seems to be no way to read the image after post-processing in URP 2D (source) and thus it uses the image from before post pro, which may look weird.

Like this (notice the sky):

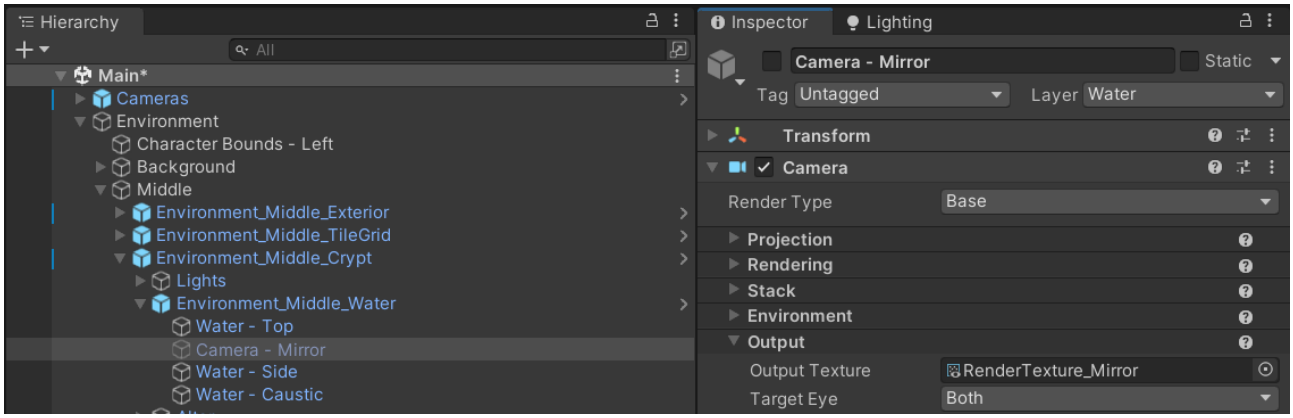
You can see the difference if you turn of PostPro Volumes (then it matches).

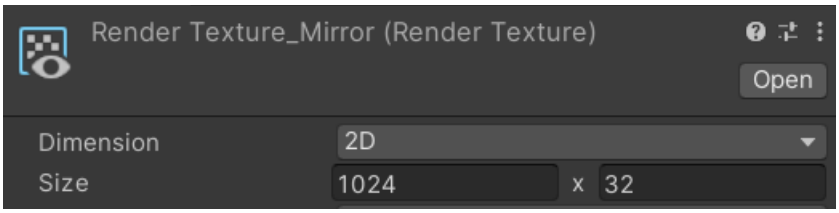
As soon as fetching the PostPro result is supported in URP 2D it can be added. Until then you will have to use the compromise of having no post-pro effects on your blurred image in overlay canvases.

World and Camera space canvases can only be supported if a transparent queue is introduced, which is unlikely but I'll keep an eye on that. If you stumble upon any news on this feel free to send me a message :-)

HINT:

The official Unity way to do some effects in the world is to add another camera and set a render texture for it and then use that render texture. If you download their „Lost Crypt" Demo then in there you will find the setup used for this.

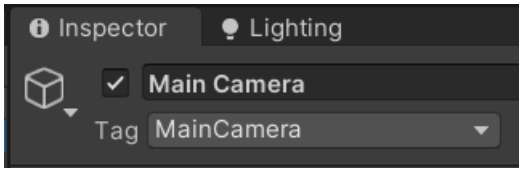




The blur image is empty (nothing is blurred). Why?

This may happen if the blur renderer does not find the camera.

The blur renderer always uses the camera with the „MainCamera" tag. Please make sure you have your camera tagged accordingly.

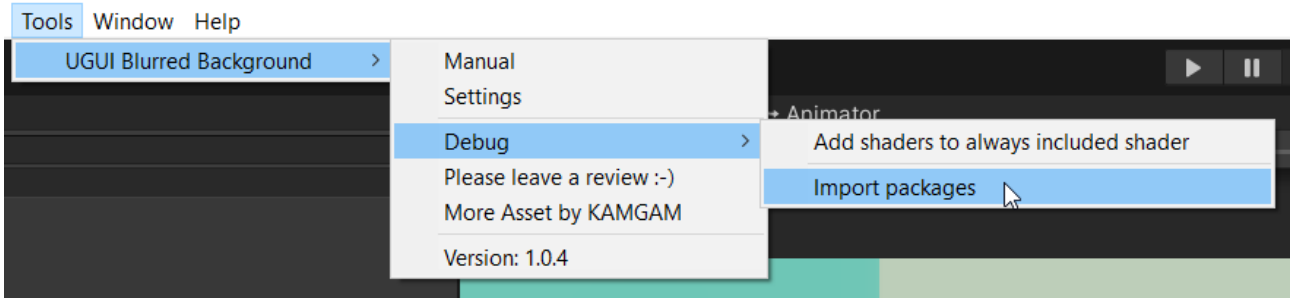


In my HDRP / URP project the blurred image is just a white square?

This may happen if the automated patching after installation did not work. To resolve this please manually import the HDRP files and add the HDRP shader to the list of always included shader.

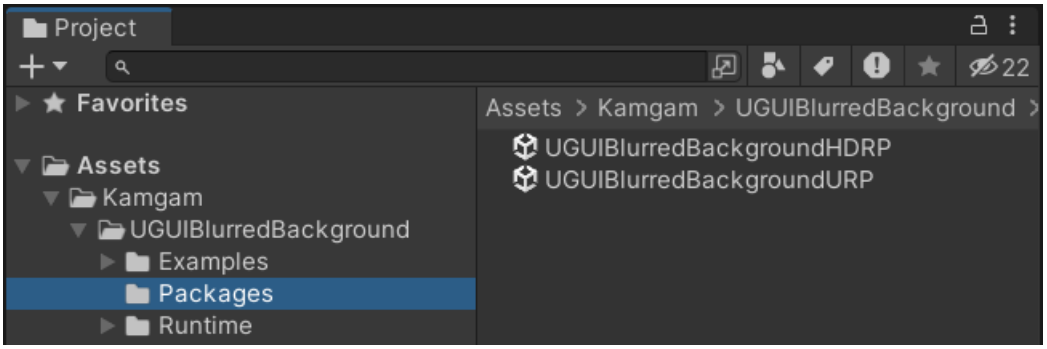
Here is a two-step guide:

1) Trigger the import of the package

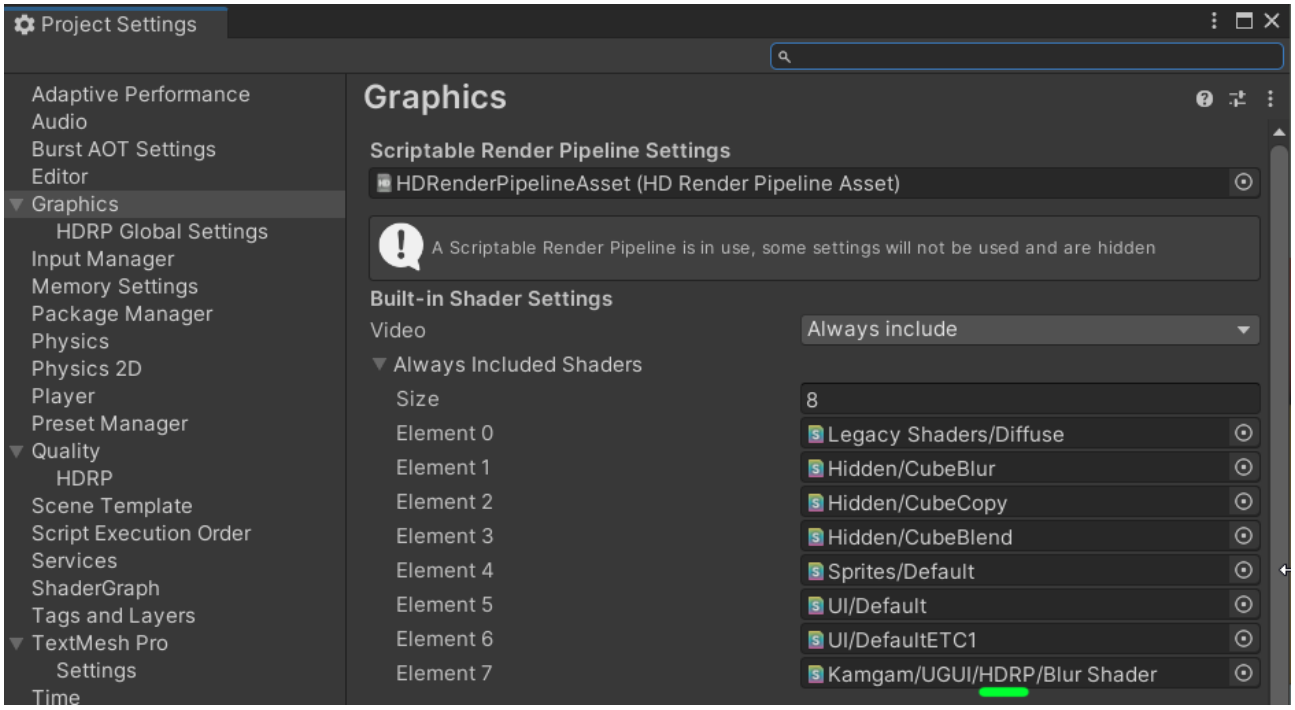


This option detects what render pipleline you are using and imports the appropriate package file.

[OPTIONAL] If you want to be extra sure you can also import the package manually. They are located under „Assets/Kamgam/UGUIBlurredBackground/Packages“:



2) After the import you should be able to replace the „BuiltIn“ version of the shader with the correct one (HDRP in this example):



My UI is flickering wildly in HDRP?!?

This seems to be an error introduced in Unity 2022.3.18f1 and Unity 2023 beta:

https://forum.unity.com/threads/strange-canvas-size-behavior-in-2022-3-18-hdrp.1539974/

As of now I do not know when/if Unity will fix it.

The blur is not working in Unity 6 (using render graph)

Please upgrade to the latest version (1.1.0+). That one supports render graph.

If you can not upgrade then please enable the compatibility mode under ProjectSettings > Graphics > Render Graph: