UI Toolkit Custom Shader Image URP

Table of contents

Requirements & Setup

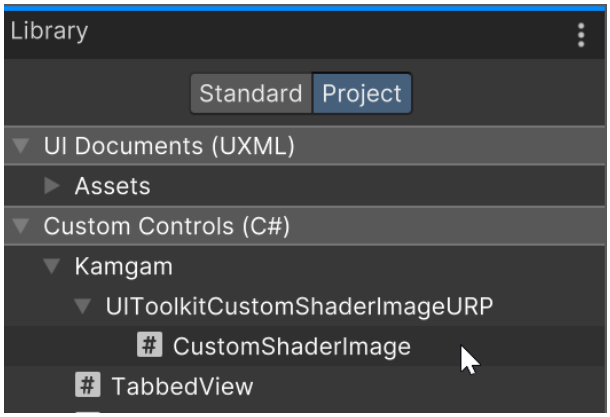
Requirements

Unity 2022.3 or higher is required since that is the min version Unity allows for new assets in the store. However it may work just fine in older versions of Unity with minor changes.

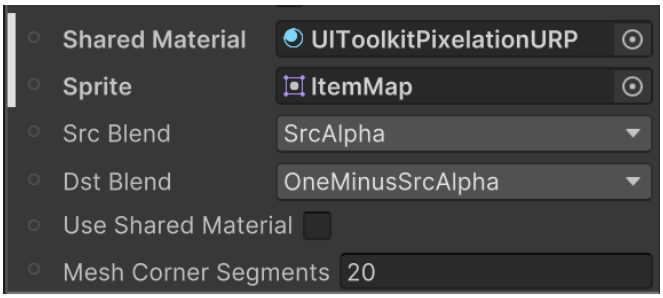
Universal Render Pipeline (URP) needs to be installed and activated. The included example shaders have only been tested in URP. Some of them may work in HDRP and Built-In too but there are no guarantees.

Usage

You can find the image element under "Custom Controls" in the library of the UI Builder.



Attributes



Shared Material

This is the base material. It is similar to the sharedMaterial of a Renderer meaning it can be shared between many images (if „UseSharedMaterial" is enabled).

Use Shared Material

Usually each image creates its own copy of the SharedMaterial upong instantiation but if this is enabled then they will share the same material. May be handy for batching though I would recommend using it only if the profiler tells you it's necessary. Also if you use it then changes to the material at runtime will be persisted in the material (just like with MeshRenderer and the „sharedMaterial" property).

Sprite

It's the image (_MainTex) of the shader. Drop in your sprite here.

SrcBlend & DstBlend

Blend Mode of the material. Notice this should not be changed dynamically. It may trigger a shader variant compilation.

Mesh Corner Segments

How many segments the mesh of the image should use for rounded corners. Usually a value of 10-20 is fine.

Using Custom Shaders

There are some limitations on what you can do with custom shaders in UI Toolkit. Please read this section if you plan on adding your own custom shaders.

Frame Buffer Access

As of now (Unity 6.x and before) in UI Toolkit your shader does NOT have access to the frame buffer. Thus no grab pass will work. This means you can not take previously rendered content into account (except via blend modes).

Masking

The masking in UI Toolkit uses a custom stencil buffer setup which is not exposed nor documented and, according to Unity, subject to change. Thus only rect masks (overflow:hidden) are supported.

If you want to know more then check out this thread with Unity's official answer: https://discussions.unity.com/t/ui-toolkit-default-shader-source/1647930/4

Batching

Introducing an new shader (next to the single hidden Uber-Shader) will break batching in UI Toolkit. Therefore you should use these custom shader images sparingly. Don't plaster your whole UI with them.

Special Shader Variables:

If you want to write your own shader then please take a look at the example shaders. There are some predefined shader properties the image uses. Here are some:

_MainTex ("Texture", 2D)
That's the „Sprite" you set on the image.

_Opacity ("Opacity", Float)
Opacity is taken control of by the image. It forwards the „opacity" style value to the shader.

[HideInInspector]_UVMinMax

If you implement this like in the example shaders then you will get tiling support for sliced sprites (very handy). x/y = UVMin slice coordinates, z/w = UVMax slice coordinates

_SrcBlend ("SrcBlend", Int) = 5 // SrcAlpha
Blend Mode of the material. Notice this can not be changed dynamically. It may trigger a shader variant compilation.

_DstBlend ("DstBlend", Int) = 10 // OneMinusSourceAlpha
Blend Mode of the material. Notice this can not be changed dynamically. It may trigger a shader variant compilation.

_Color ("Tint", Color)
A spearate color to control only the image part (useful for opacity changes that should only be applied to the image and not the borders).

Frequently Asked Questions

The image looks darker in the UI Builder than in the GameView

Yes, sadly this is a current limitation of the UI Builder (it does not support sRGB textures). When in doubt always check the game view.

Quote from Unity: „This is expected, but the ui builder does not currently support displaying a linear UI."

Source: https://discussions.unity.com/t/ui-builder-doesnt-support-linear-color-space-and-ui-toolkit-too/870864/2

My custom shader does not affect the background or borders, why?

Sadly shaders in UI Toolkit do not have access to the frame buffer and thus no content other than the image itself can be affected by them. This includes background and borders of the image.

Changing the border at runtime does not immediately update the image area.

Yes, changing the border size or radius does not automatically update the image area inside because it does not trigger a geometry changed event (sadly). Styles are checked for changes every 50 MS by the image itself but you may call CheckStyles() yourself after doing the changes to get an immediate repaint of the image.

I can not mask the image (except for rect masks). Why?

The masking in UI Toolkit uses a custom stencil buffer setup which is not exposed nor documented and, according to Unity, subject to change. Thus only rect masks (overflow:hidden) is supported.

If you want to know more then check out this thread with Unity's official answer: https://discussions.unity.com/t/ui-toolkit-default-shader-source/1647930/4

Can I set the material and image via style sheets?

Not yet, if there is enough demand for it then I might add it. Please let me know. The question is always how to pass in an image. Is it from a Resource, Streaming Assets an Asset Bundle? I wanted to keep it simple for now (code API that takes a Sprite).

Is there a Scripting API

Yes, just check out the public properties of the CustomShaderImage. It's just 4,5 attributes which are pretty straight forward. If you have any questions please don't hesitate to ask under office@kamgam.com.