Soft Mask

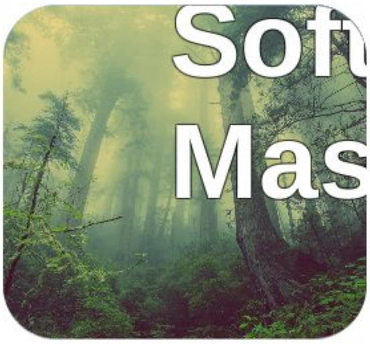
Documentation for the version 1.0

What is Soft Mask

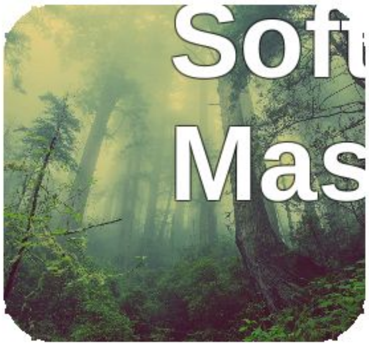
Soft Mask is a component for creating a smooth mask for UI elements in Unity.

When adding to a Game Object, Soft Mask “masks” its child elements. Soft Mask gives a quite similar effect as standard Mask and Mask Rect 2D but has some advantages over them.

- Unlike Rect Mask 2D, Soft Mask supports the rotating of a mask and renders a mask with anti-aliasing.
- Unlike Mask, Soft Mask supports transparency, which allows you to create smooth transitions from visible to invisible parts of the image, as well as use inclined or rounded edges of UI elements.



Soft Mask



Standard Unity’s Mask (notice visual artefacts on the rounded corners caused by cutoff)

Features

- *Easy to use*: Soft Mask works in the same way as the standard Mask. There’s no need to assign special materials or modify the masked elements in any other way.
- *Support for Image, RawImage, Sprite or Texture*: Soft Mask supports the using of Image and RawImage components as a mask. Also, you can explicitly specify the Sprite or Texture.
- *Raycast filtering*: you can set up rectangular or per-pixel test for event raycasting.
- *Flexible adjustment of mask color channels*: you can use black and white images, images with transparency or set a weight value for each color channel.
- *Custom shader support*: Soft Mask can be easily supported by custom shaders, examples are described in program code (see section Support by custom shaders).
- *Real-time update*s: moving, scaling and rotating a mask, including in playtime or animation mode.

Getting Started

Soft Mask requires no additional setup. Just import the package, and you are ready to go.

The package includes example scenes so that you can see Soft Mask in action.

If you use UnityScript instead of C# in your project, you can move the imported `SoftMask` folder to the Plugins folder. It will allow you to use Soft Mask from UnityScript code.

The `Samples` folder isn't necessary for the Soft Mask work. You can remove it completely from your project.

The Use of Soft Mask

Apply Soft Mask

To apply a soft mask, add the UI/Soft Mask component to a Game Object the same way as you apply a standard Mask.

If you want to display the mask image in the scene, ensure that the Game Object has an Image or RawImage component. The soft mask will use the same image that is rendered by an Image.

If you want to use the mask image as a mask only and do not display it in the scene, you have two options. First, if you already have an Image or RawImage on the Game Object, you can just disable it. Soft Mask will continue masking. It has the same effect as disabling *Show Mask Graphic* option of a standard Mask. Second, you can select Sprite or Texture in the Source drop-down list and set a sprite or texture for the mask. In this case, the Game Object doesn't have to have a Graphic component at all.

Change a standard mask to a soft mask

Soft Mask can replace a standard Mask that uses an Image or RawImage component only. For details, see section Limitations.

To replace a standard Mask by a Soft Mask, follow the steps below.

1. Add the UI/Soft Mask component to the Game Object, which already has a standard Mask.
2. Disable the standard mask and check the soft mask works properly.
3. If the *Show Mask Graphic* option has been disabled for the standard mask, disable the Graphic component of the Game Object. Unlike standard Mask, Soft Mask continues to work in such a case.
4. Delete the standard mask from the Game Object.

Disable soft mask

If you want to turn off the soft mask, disable Soft Mask component of the Game Object. When a Graphic component of the Game Object disabling, Soft Mask still keeps masking.

Support by custom shaders

Soft Mask is able to mask all Graphic components, which use the standard Unity UI shader. If you want to use custom shaders for rendering UI elements (for example, your own solutions or another asset from Asset Store), you can add the support of Soft Mask to them.

To add the support of Soft Mask to the custom shader, you have to insert some declarations and instructions in the shader code. For more details, see the *04-CustomShaders* example. The custom shader used by this example can be found at `/Assets/SoftMask/Samples/Materials/CustomWithSoftMask.shader`. All instructions needed for Soft Mask support have the comment “Soft Mask Support”.

Soft Mask instantiation from code

Soft Mask replaces the standard Unity UI shader during the rendering of child objects. When you add the Soft Mask component in the editor, it gets a reference to the replaced shader and works “out of the box”. But if you add Soft Mask from code by using `AddComponent()`, then you have to specify that shader manually. To do this, set `defaultShader` property:

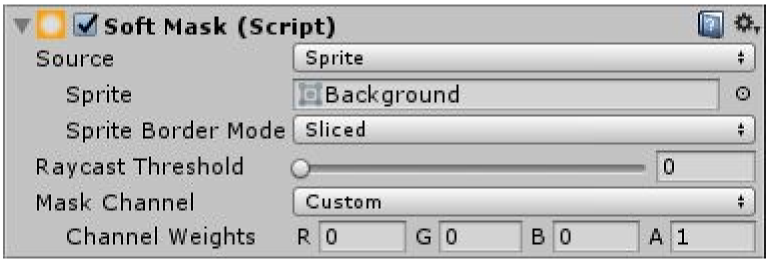
```
public class InstantiateSoftMaskSample : MonoBehaviour {
    public Shader defaultMaskShader;

    void Start() {
        var mask = gameObject.AddComponent<SoftMask>();
        mask.defaultShader = defaultMaskShader;
    }
}
```

If you’re going to publish the project on the mobile platforms and use ETC1 compressed textures in UI, you should also set the `defaultETC1Shader` property.

If all masked objects use a custom shader with Soft Mask support, you don’t have to set the `defaultShader` property, because in such case no replacements will be done.

Properties



☑ Soft Mask (Script)

Source: Sprite

Sprite: Background

Sprite Border Mode: Sliced

Raycast Threshold: 0

Mask Channel: Custom

Channel Weights: R 0 G 0 B 0 A 1

Soft Mask Inspector

Property:	**Function:**
Source	Defines way to configure a mask image. Possible values: **Graphic** The mask image is taken from the Graphic component of the Game Object. Soft Mask supports Image and RawImage components only. If there is no appropriate Graphic on the GameObject, a solid rectangle of the RectTransform dimensions will be used. **Sprite** Mask image is taken from an explicitly specified Sprite. When this mode is used, *Sprite Border Mode* can also be set to determine how to process Sprite's borders. If the sprite isn't set, a solid rectangle of the RectTransform dimensions will be used. This mode is analogous to using an Image with according *Sprite* and *Type* set. **Texture** The mask image should be taken from an explicitly specified Texture2D. When this mode is used, *Texture UV Rect* can also be set to determine what part of the texture should be used. If the texture isn't set, a solid rectangle of the RectTransform dimensions will be used. This mode is analogous to using a RawImage with according *Texture* and *UV Rect* set.
Raycast Threshold	Specifies the minimum mask value that the point should have for an input event to pass. Accepts values in range [0..1]. If the value sampled from the mask is greater or equal this value, the input event is considered 'hit'. The default value is 0, which means that any pixel belonging to RectTransform is considered in input events. If you specify the value greater than 0, the mask's texture should be readable.
Mask Channel	Defines what color channels of the image are used as a mask. Possible values: **Alpha** (by default) Use the alpha channel (transparency) of the image as a mask. Coincides with *Custom Channel Weights* = (0, 0, 0, 1). **Red** Use the red channel of the image as a mask. Coincides with *Custom Channel Weights* = (1, 0, 0, 0). **Green** Use the green channel of the image as a mask. Coincides with *Custom Channel Weights* = (0, 1, 0, 0).

	Blue Use the blue channel of the image as a mask. Coincides with *Custom Channel Weights* = (0, 0, 1, 0). **Gray** Use the average value of the red, green and blue channels of the image as a mask. Coincides with *Custom Channel Weights* = (0.33, 0.33, 0.33, 0). **Custom** Allows to set the specific weight for each color channel of the image.
Channel Weights	Displayed if *Mask Channel* is **Custom**. You can set the weight from 0 to 1 for each color channel. Given the value sampled from the mask texture is *mask* and *weights* are the specified property value, the resulted mask value is calculated as: $\sum_{i \in \{r,g,b,a\}} mask_i \cdot weights_i$

Additional properties for Sprite-based masks:

Sprite	A sprite which is used as a mask.
Sprite Border Mode	How to render the sprite borders. Possible values: **Simple**, **Sliced**, **Tiled**. They all works the same way as according values of Type property of Image.

Additional properties for Texture-based masks:

Texture	A texture which is used as a mask.
Texture UV Rect	Defines coordinates and size of texture fragment to display the image, given in normalized coordinates (range 0.0 to 1.0). This property works the same way as UV Rect property of RawImage.

Limitations

- You can use only Image or RawImage of Graphic components as a soft mask. Text and any other custom Graphics aren’t supported as a soft mask.
- Soft Mask doesn’t support Filled type and Fill Center option of Image.
- When using a Graphic component, its color doesn’t affect the mask. Soft Mask uses the Graphic texture only.
- Soft Mask is intended to work with Unity UI and doesn’t support any other 2D or 3D graphics “out of the box”.

- Soft Mask doesn’t support nested masks. To get past this limitation, use a standard Mask or Rect Mask 2D in combination with Soft Mask.
- Sprites packed in Tight Packing Mode aren’t supported. Disable packing for a mask sprite or use Rectangle Packing Mode.
- Sprites with an alpha split texture aren’t supported. Disable compression for mask textures or use another compression type.

Unlike Unity’s standard Mask, Soft Mask doesn’t render Graphic component. This is the cause of many limitations listen. When you set *Source* to *Graphic*, Soft Mask just gets some properties (texture and border mode or UV rect) from Graphic component, but doesn’t really use Graphic itself.

Performance

- Soft Mask uses 3 Shader Keywords, while Unity 5 provides 128. Before using Soft Mask, make sure that you have enough free slots in the project. For details about Shader Keywords, see Multiple Program Variants section in the Unity Documentation.
- Soft Mask support adds to shaders from 7 to 10 constants and 1 sampler, depending on Soft Mask settings.
- All children elements of Soft Mask are masked by using the special shader. This shader is more complex and a bit slower than the standard shader. Execution speed of the shader depends on the *Border Mode* value. The fastest mode is Simple. It’s recommended to use Soft Mask only when you lack standard Mask features.

Compatibility

Soft Mask 1.0 is compatible with Unity versions 5.3-5.5.

Tested platforms:

- standalone (Windows/Mac OS),
- WebGL,
- Android,
- Windows Phone.

Support and feedback

If you need support for this product or wish to provide suggestions, feel free to email me at knyazev751@gmail.com.