



DTT Procedural UI

Documentation | V1.0.0 | 03-06-21

SINCE 2010





Table of Contents

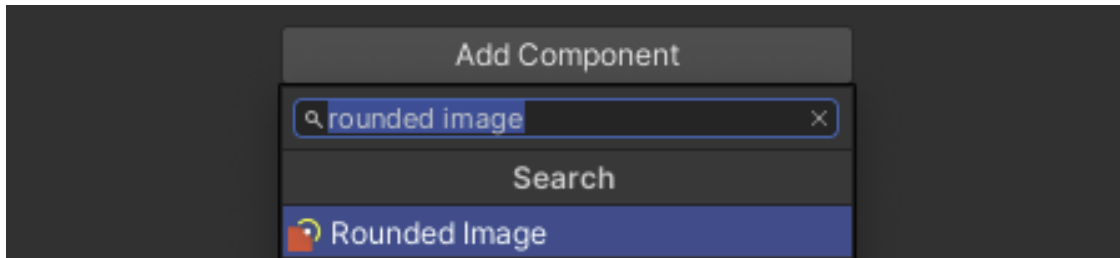
1. Get started quickly	3
2. Introduction	5
3. Changelog	6
4. Set-Up	7
5. Editor	8
5.a Overview	8
5.b Image Options	9
5.c Unit settings	10
5.d Corner target mode	11
5.e Fill mode	12
5.f Hitbox settings	13
6. Known Limitations	14
7. Support and feedback	15



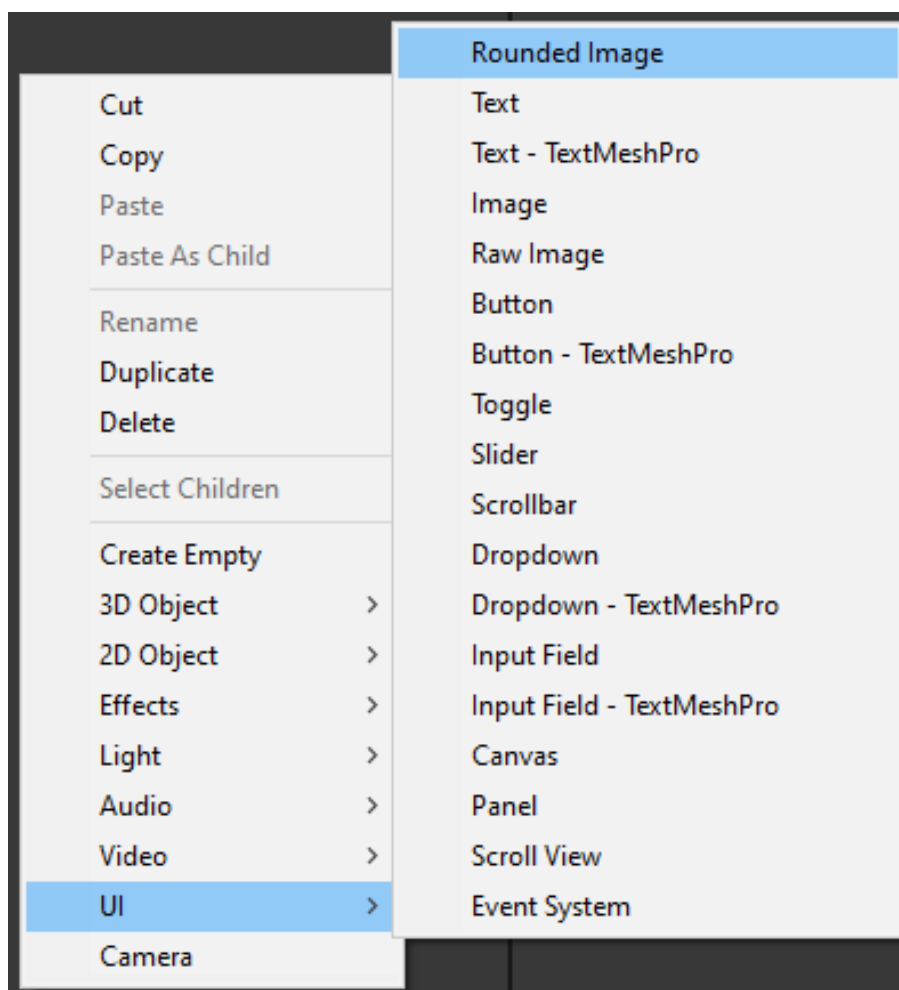
1. Get started quickly

This is a summary to help you get started quickly. More details are provided below.

To get started, use the inspector to add the 'Rounded Image' component to any UI element:



Or use the scene context menu to create a new Rounded Image UI object:





Then, use the **inspector** to adjust the UI element to your desired outcome using:

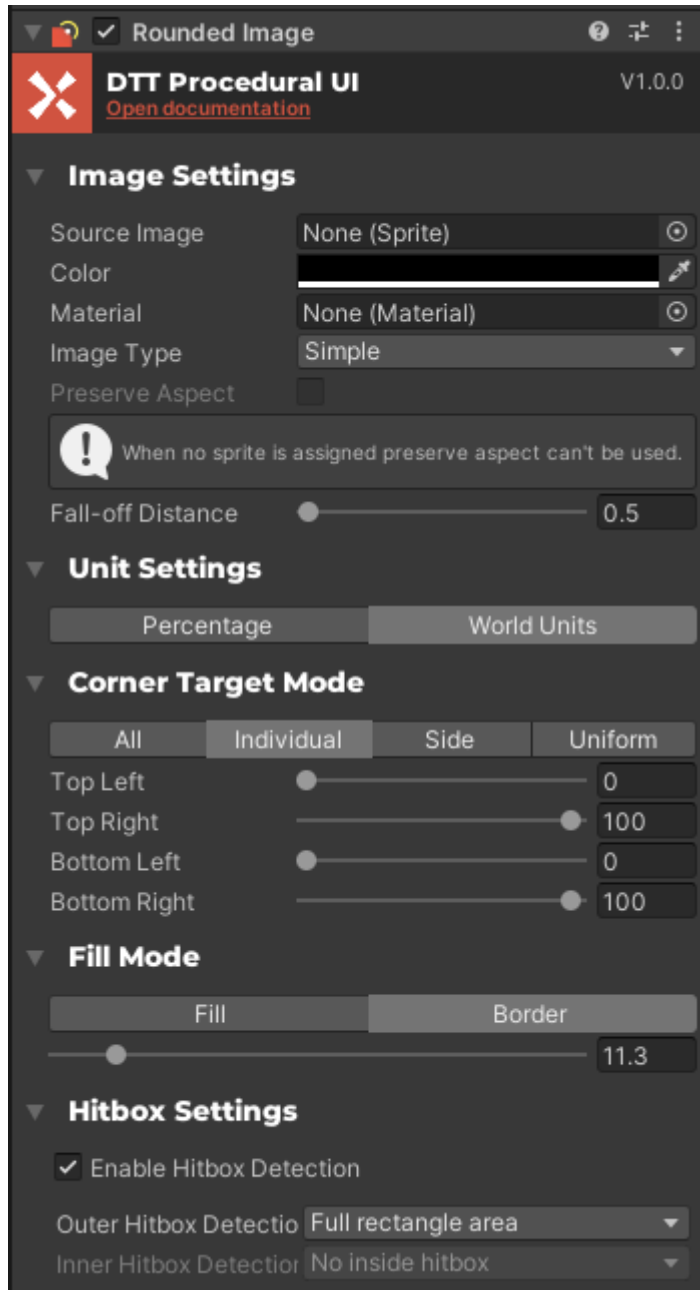


Image Settings

Procedural image is an extension of Image; all Image functionality are compatible, including support for 'sprite mode: multiple' and atlases.

Unit Settings

Allows you to switch the method of rounding.

Corner Target Mode

Choose how to adjust corners. Use the slider to set the amount of rounding.

Fill Mode

Allows you to set a border, including its width.

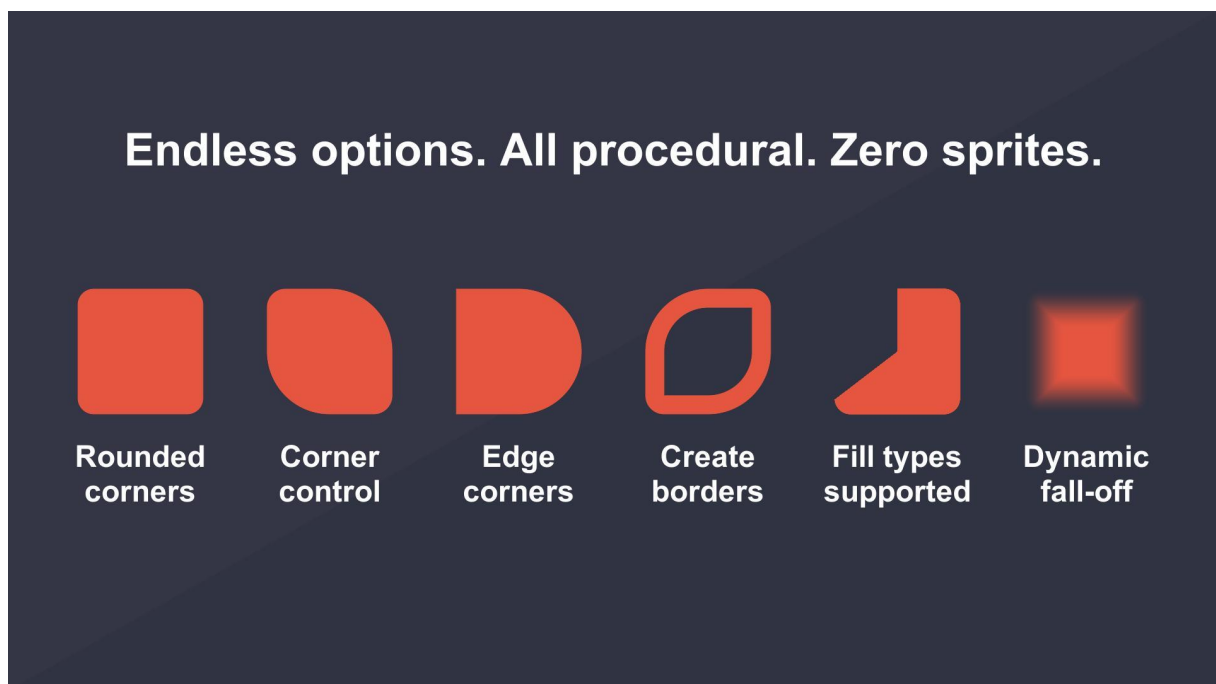
Hitbox Settings

Enable hit detection and indicate if you want the hit area to take rounded corners into account.



2. Introduction

DTT Procedural UI functions as an extension of the Unity UI Image Component. You can use it both with and without a sprite, which it renders in real time using shaders. It allows you a greater amount of freedom in creating a flat styled UI without having to create all individual sprites in external editors. This can save you large amounts of disk space, making for a more compact game build. This, combined with our extremely efficient shaders, makes it perfect for mobile development while still being able to achieve stunning effects even on high resolutions. You can also adapt the values in real time during gameplay, allowing you to achieve dynamic effects.





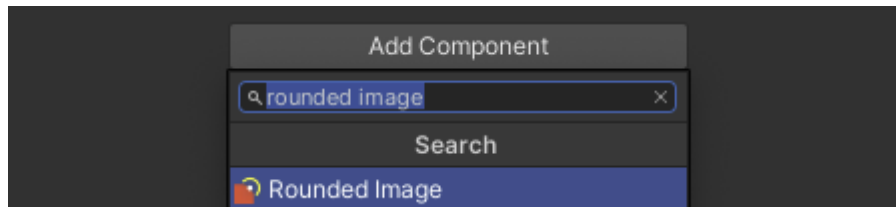
3. Changelog

1. version 1.0.0 – initial release

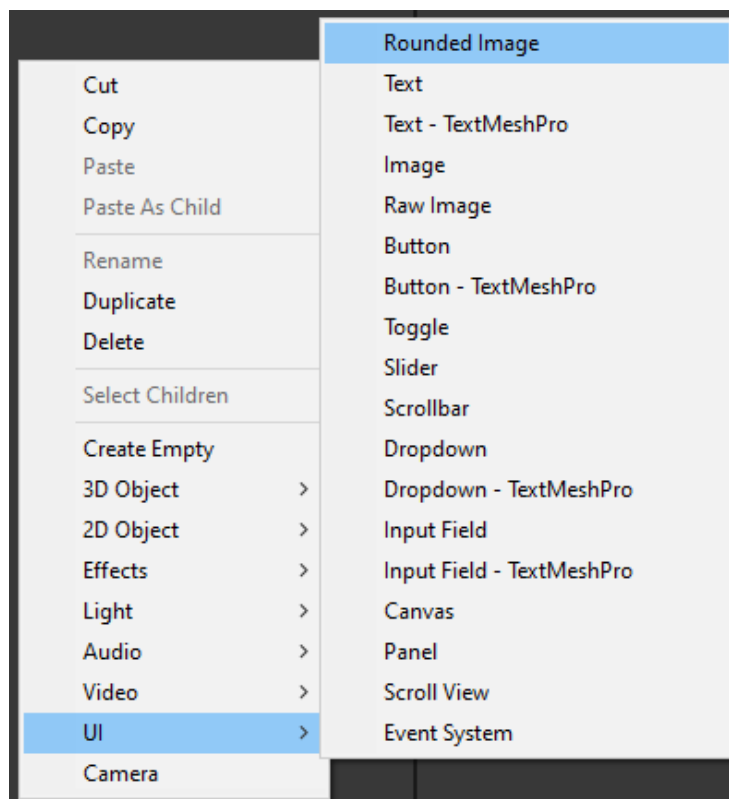


4. Set-Up

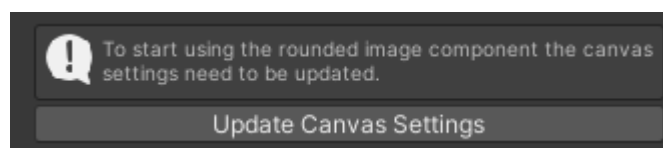
To get started, use the inspector to add the 'Rounded Image' component to any UI element after importing the package:



Or use the scene context menu to create a new Rounded Image UI object:



Depending on your canvas settings, you might see the following warning in your inspector after selecting the object:



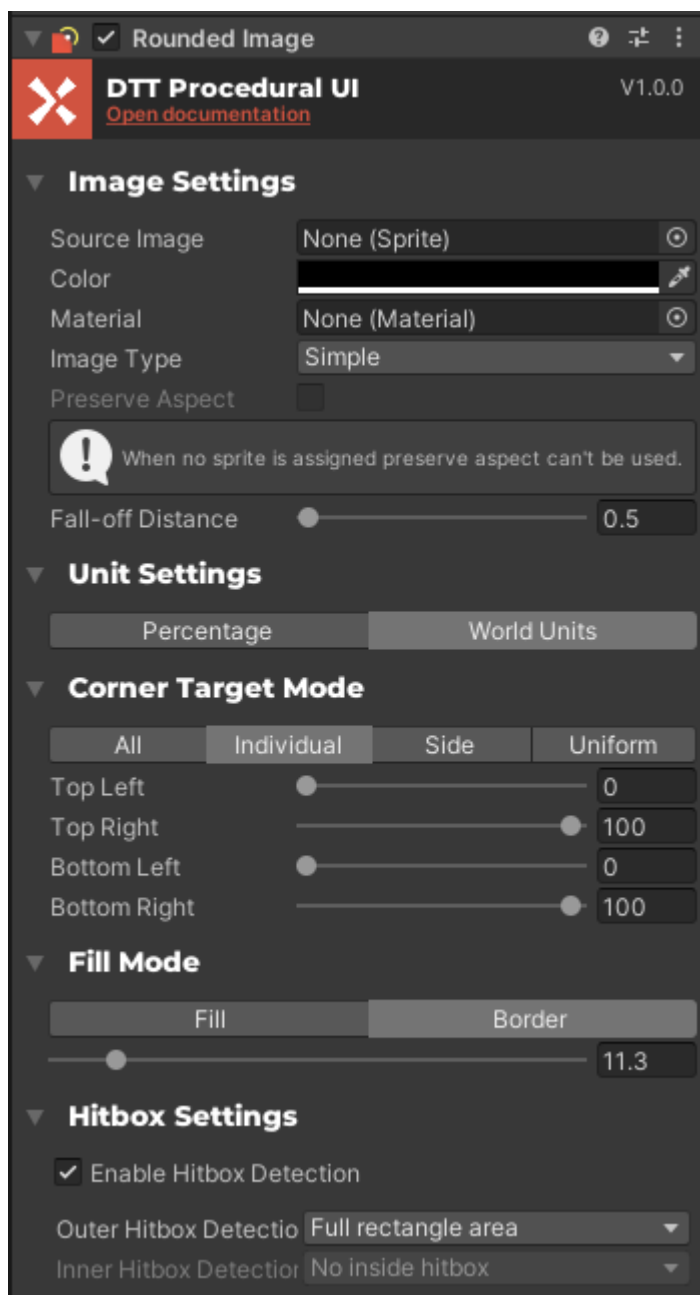
This most commonly means that your canvas does not have the correct shader channels enabled. Simply click the button and this will be resolved. You're good to go!



5. Editor

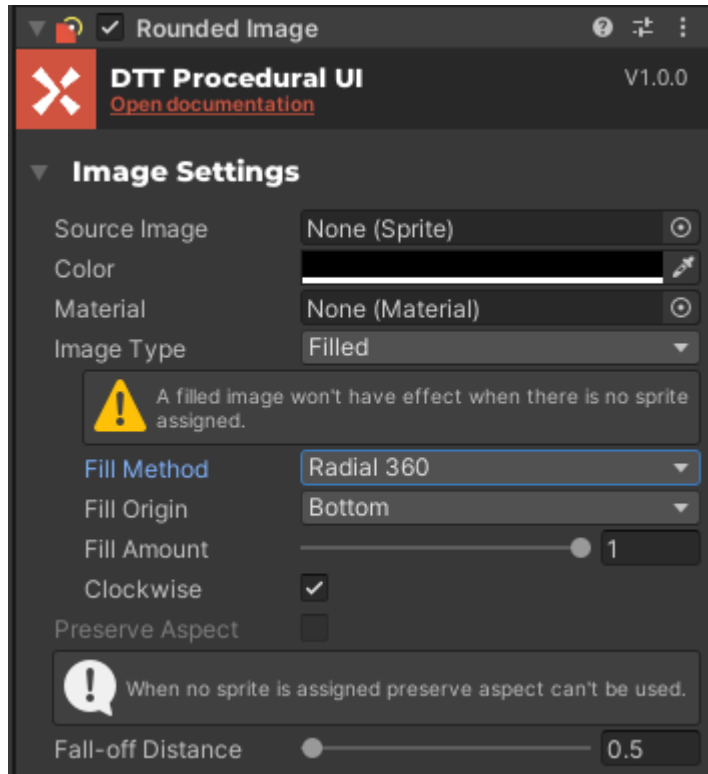
5.a Overview

This is the DTT Procedural UI Editor. It is a powerful tool with which you can adjust your images without using pre-prepared sprites. In this chapter we will break down the functionality of each section of the editor in detail.





5.b Image Options



As DTT Procedural Image is an extension of the Unity Image Component, you will find that the features you are familiar with are still available to you. This includes selecting sprites to adapt using the editor, changing the colour, filled image type, and sprite mode: multiple and atlases¹, and more.

One new feature you will notice in the bottom of this section is Falloff distance. Using this slider, you can adjust how the edge of your image is rendered. You can use this to create **shadows**.

¹ sprite mode: multiple and atlases are not supported in versions of Unity older than Unity 2020.2.



5.c Unit settings



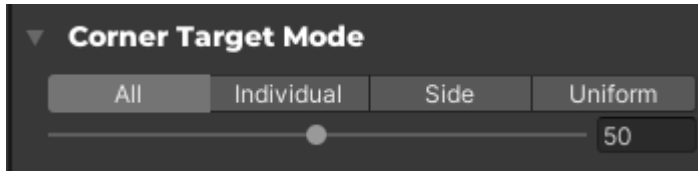
The Unit Settings section lets you choose how the amount of rounding, or the size of your border, is calculated:

- **Percentage:** You can use a percentage value on your corner and borders. By using this, it will remain visually similar regardless of the size of the object.
- **World Units:** This is a hard value that will persist even if the object is scaled either larger or smaller. It will, however, lock when the maximum amount of rounding has been achieved.

These options give you the flexibility to decide what fits your project and use-case best.

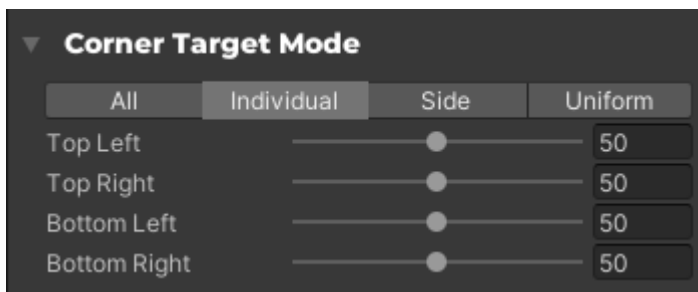


5.d Corner target mode

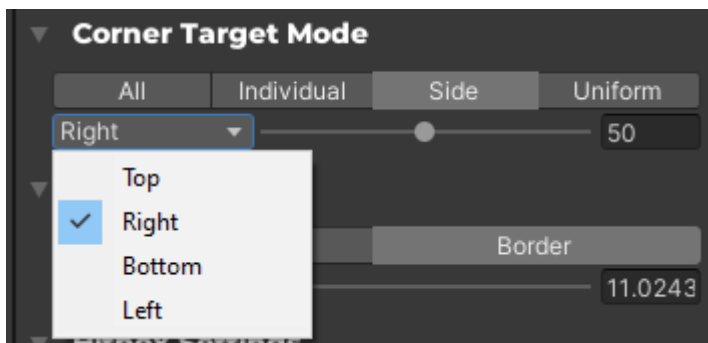


Using the corner target mode section, you can adapt the rounding effect on your image using either percentages or world

units. Here you see **All Mode**. This mode rounds all four corners by the given amount. You can change this by using the slider or typing in the value.



When selecting **Individual Mode** you can adjust every corner in minute detail using the four individual sliders.



Using **Side Mode** the system will automatically take the two relevant corners and round them together. This allows you to quickly give your UI an edge without spending too much time on fine tuning.



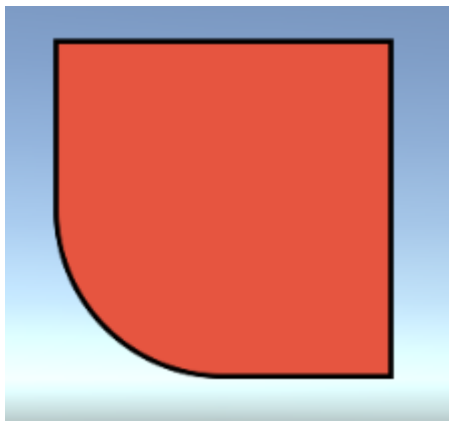
Uniform Mode automatically rounds all sides to the maximum value with a single click, turning squares into circles. This very easily gives your UI that slight polished edge it might be lacking.



5.e Fill mode



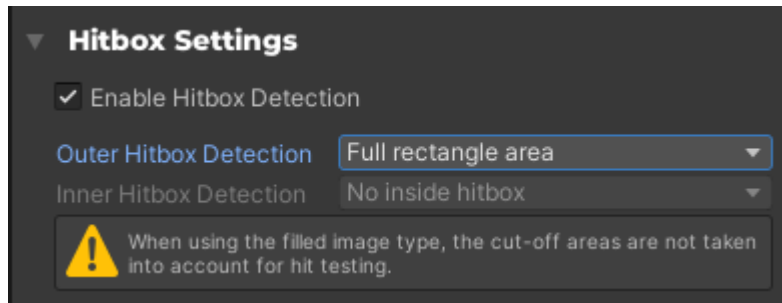
Fill Mode allows you to use the image as a border by opening a hole in its center. Using the slider, you can determine the size of your border in either percentages or world units. This can be a very powerful tool to create versatile effects.



Here we used 2 objects to create this button. A black image with border mode turned on and set to 3%, and a separate image to serve as the clickable area.



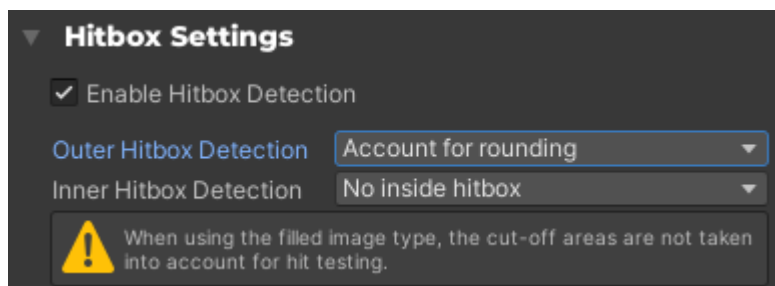
5.f Hitbox settings



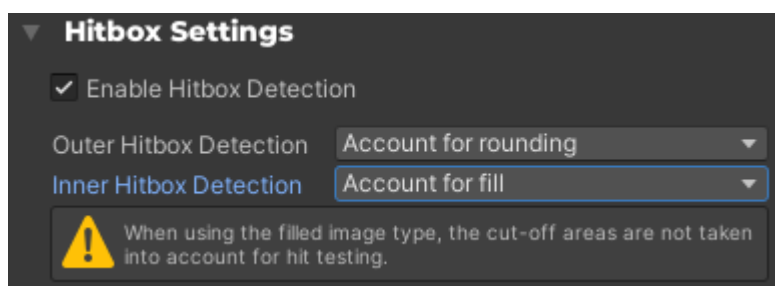
DTT Procedural UI comes with a custom hitbox solution. This allows you to dynamically adapt the interactive areas of your buttons to fit the new visuals, or ignore the new

visuals if you so desire. If you set the values as shown in the above image, you can expect identical results as with using Unity's hit detection.

Also note that it is showing a warning related to 'filled image type'. The editor will always inform you if you are at risk of relying on an unsupported feature.



With 'Account for rounding' turned on, any effects applied to the outside of your image will be taken into account when checking for clicks or taps.



With '**Account for fill**' turned on, any effects applied to the inside of the image, such as **fill mode**, will be taken into account¹.

Together these options give you the ability to choose where and when the user can click successfully.

¹ Note that this references our fill mode option you can use to create borders. The standard Unity fill mode image type is currently not supported by this feature.



6. Known Limitations

- **Unity Fill Image:** Unity Fill image type effects are not taken into consideration with our custom hit detection (much like with the standard unity implementation). Research into this is ongoing.
- **The Readme:** If you close the Unity Editor while the readme panel is open it might prevent you from re-opening it using the editor button.
 - You can resolve this by resetting your layout (windows/layouts).
- **Sprite mode: Multiple:** Due to limitations within the engine, Sprite mode: multiple is not supported in versions older than Unity 2020.2.
- **Atlases:** Due to limitations within the engine, atlasas are not supported in versions older than Unity 2020.2.
- **Grey Buttons:** If you are working within a canvas group and your game object has an inactive button component, setting the canvas group's 'interactable' value to False can cause the image to render in gray.
 - Turning the button component on and off again will resolve this.
 - If you have removed the button component, re-adding it will resolve this.



7. Support and feedback

If you have any questions regarding the use of this asset, we are happy to help you out.

Always feel free to contact us at:

unity-support@d-tt.nl

(We typically respond within 1-2 business days)

We are actively developing this asset, with many future updates and extensions already planned. We are eager to include feedback from our users in future updates, be they 'quality of life' improvements, new features, bug fixes or anything else that can help you improve your experience with this asset. You can reach us at the email above.

Reviews and ratings are very much appreciated as they help us raise awareness and to improve our assets.

DTT stands for Doing Things Together

DTT is an app, web and game development agency based in the centre of Amsterdam. Established in 2010, DTT has over a decade of experience in mobile, game, and web based technology.

Our game department primarily works in Unity where we put significant emphasis on the development of internal packages, allowing us to efficiently reuse code between projects. To support the Unity community, we are publishing a selection of our internal packages on the Asset Store, including this one.

More information about DTT (including our clients, projects and vacancies) can be found here:

<https://www.d-tt.nl/en/>