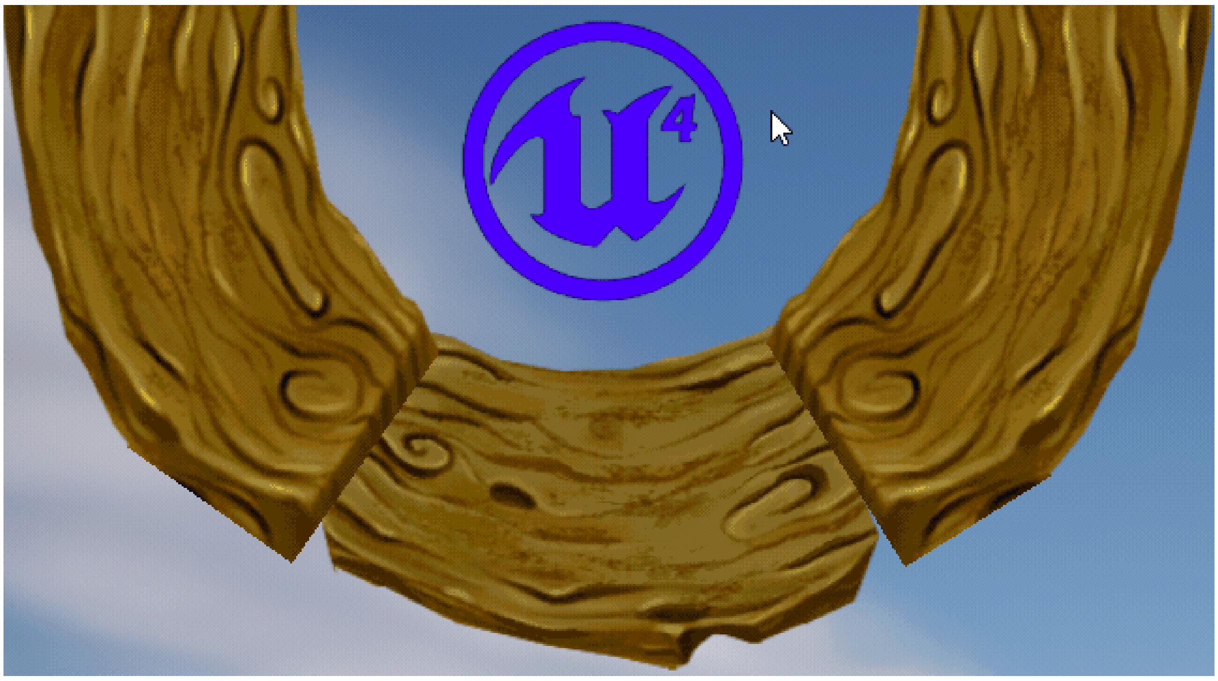
Custom Shape Button documentation

GitHub page

Introduction

The Custom Shape Button plugin revolutionizes the way buttons are designed in Unreal Engine. Traditionally, the Unreal Engine restricts you to creating buttons with only rectangular forms. This limitation, however, is eliminated by our plugin.
With the Custom Shape Button plugin, you can now create buttons of any shape or form you envision. Whether you want a circular button, a star-shaped one, or one in the shape of a custom image, this plugin allows you to bring that vision to life. Additionally, the plugin ensures the hover and press behavior works flawlessly with the custom shapes, ensuring a seamless user experience.



Getting Started

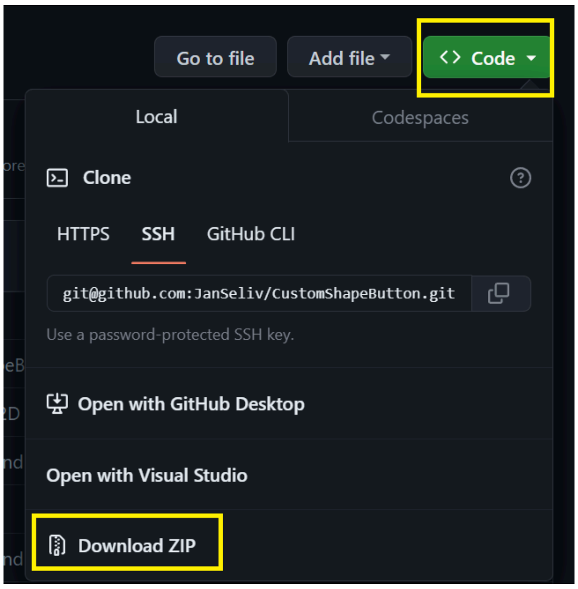
The Custom Shape Button is extremely easy to setup and doesn't require any specific steps or custom logic. Just use the `Custom Shape Button` widget and assign your texture with an alpha channel or a material with an Opacity mask.

Check out great video guide from @UntitledProjectX:
https://youtu.be/Db8prQdXWvo?si=M3hT_EdF8ccn-nbO

Or follow next steps:

Install the plugin in 4 steps

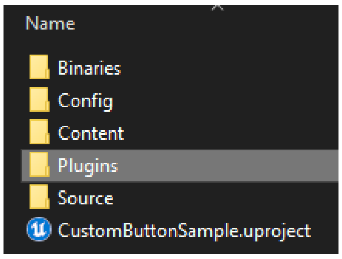
Step 1.
Download the plugin: https://github.com/JanSeliv/CustomShapeButton



```
git@github.com:JanSeliv/CustomShapeButton.git
```

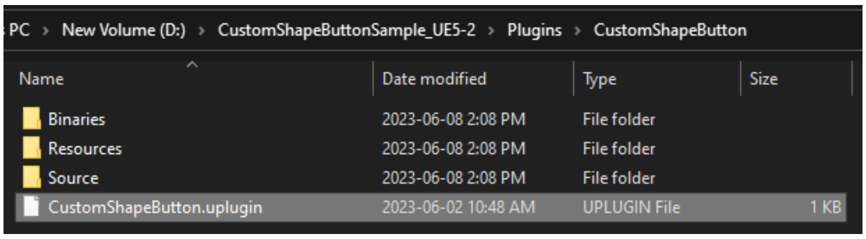
Use a password-protected SSH key.

Step 2.
Create in your game project *'Plugins'* folder and inside *'CustomShapeButton*' one



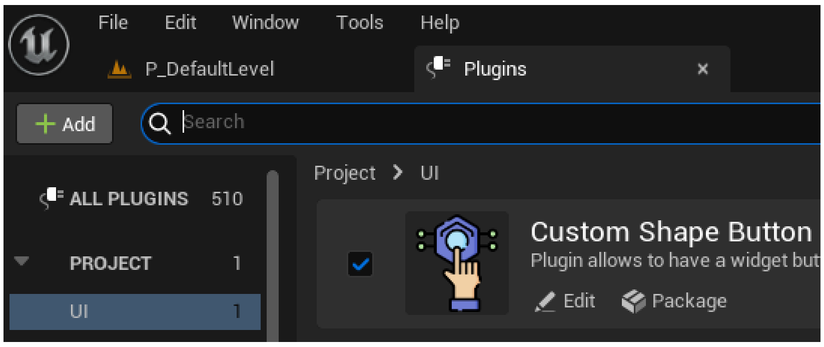
Name
Binaries
Config
Content
Plugins
Source
CustomButtonSample.uproject

Step 3.
Extract downloaded files into *Plugins -> CustomShapeButton*: `*CustomShapeButton.uplugin*` has to be located under next path:
YourGame\Plugins\CustomShapeButton\CustomShapeButton.uplugin


PC › New Volume (D:) › CustomShapeButtonSample_UE5-2 › Plugins › CustomShapeButton

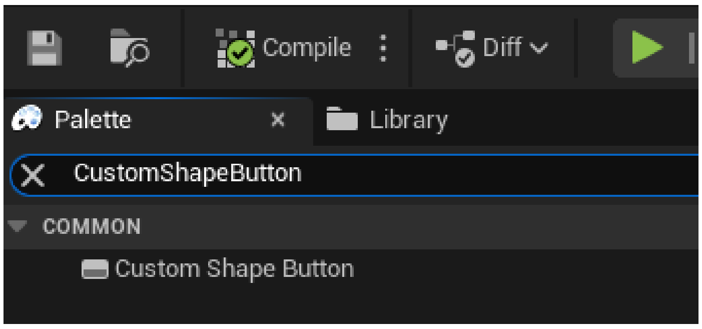
Name	Date modified	Type	Size
Binaries	2023-06-08 2:08 PM	File folder	
Resources	2023-06-08 2:08 PM	File folder	
Source	2023-06-08 2:08 PM	File folder	
CustomShapeButton.uplugin	2023-06-02 10:48 AM	UPLUGIN File	1 KB

Step 4.
Open your project,'Edit' -> 'Plugins' window to make sure the *'Custom Shape Button*' plugin is enabled for your project:



Setup the button in 4 steps

Step 1.
Add '*Custom Shape Button'* to any widget blueprint you need:

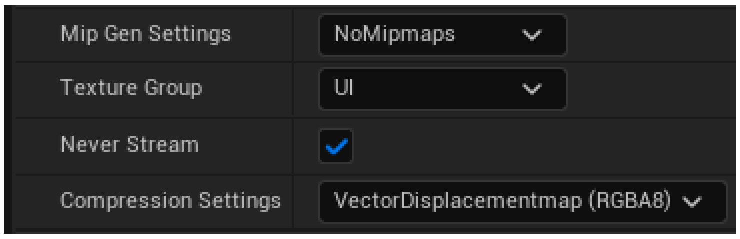


Option A: prepare texture with Alpha channel

You only need your texture to contain alpha pixels, no materials are needed:

Step 2A.
Import to the Engine the button texture that contains alpha pixels.

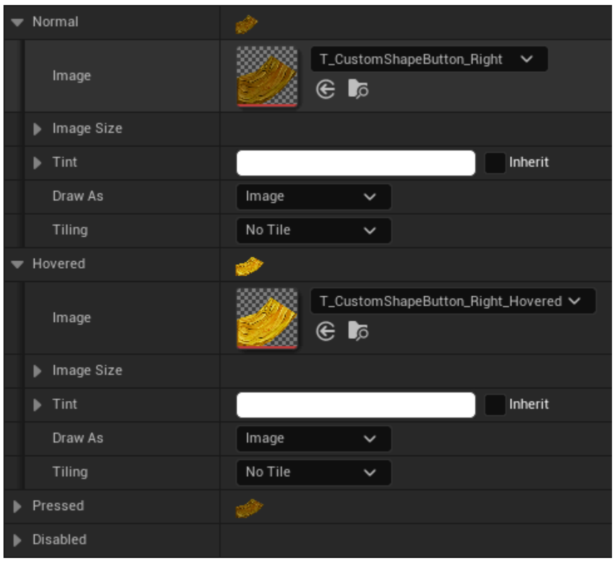
Step 3A.
Open imported texture and set next properties:



Mip Gen Settings	NoMipmaps
Texture Group	UI
Never Stream	✓
Compression Settings	VectorDisplacementmap (RGBA8)

- **Mip Gen Settings**: NoMipmaps
- **Texture Group**: UI
- **Never Stream**: true
- **Compression Settings**: VectorDisplacementmap (RGBA8)

Step 4A.
Under the 'Details' window of your button, find 'Button Style' and select your textures:

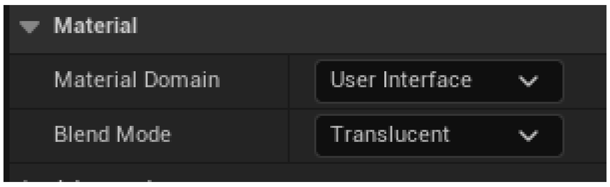


Normal	
Image	T_CustomShapeButton_Right
Image Size	
Tint	Inherit
Draw As	Image
Tiling	No Tile
Hovered	
Image	T_CustomShapeButton_Right_Hovered
Image Size	
Tint	Inherit
Draw As	Image
Tiling	No Tile
Pressed	
Disabled	

Option B [Alternative]: or prepare material with Opacity mask

If for any reason you prefer materials over textures, materials with Opacity mask are also supported:

Step 2B.
Create material with next settings in the Details window:

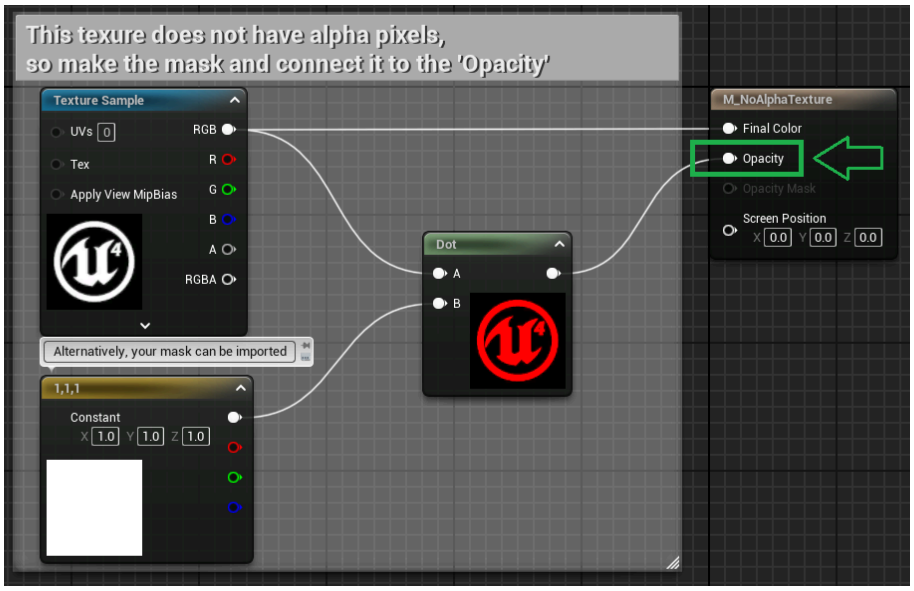


Material

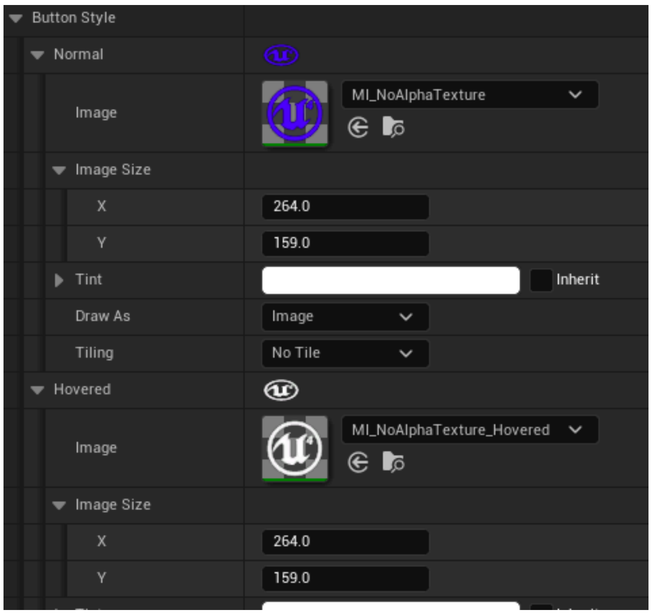
Material Domain	User Interface
Blend Mode	Translucent

- **Material Domain**: User Interface
- **Blend Mode**: Translucent

Step 3B.
Setup your mask and connect it to the `Opacity`:



Step 4B.
Under the 'Details' window of your button, find 'Button Style' and select your Materials or Material Instances:



Button Style
Normal
Image: MI_NoAlphaTexture
Image Size
X: 264.0
Y: 159.0
Tint: Inherit
Draw As: Image
Tiling: No Tile
Hovered
Image: MI_NoAlphaTexture_Hovered
Image Size
X: 264.0
Y: 159.0

Notes:
'**Image Size'** from the image is important:

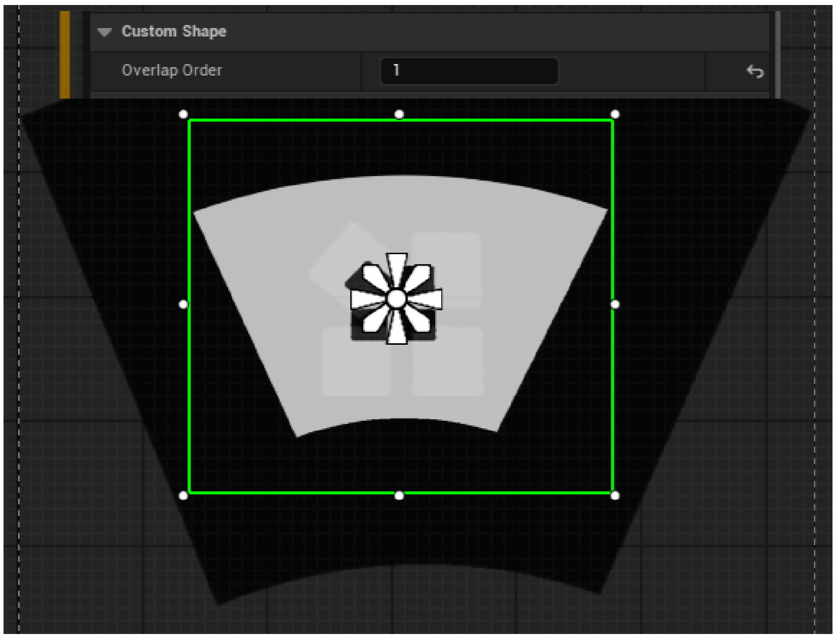
- it should match with original texture size.
- Normal and Hovered Image Sizes should be the same.

Overlap Order setting

This is an optional setting intended for advanced setups where custom shape buttons fully overlap each other. The default value is 0. The higher the Overlap Order, the higher the button's priority for receiving hover and click events.

For example, in the image below:

- The button on top has Overlap Order = 1
- The button behind it has Overlap Order = 0



Custom Shape
Overlap Order 1

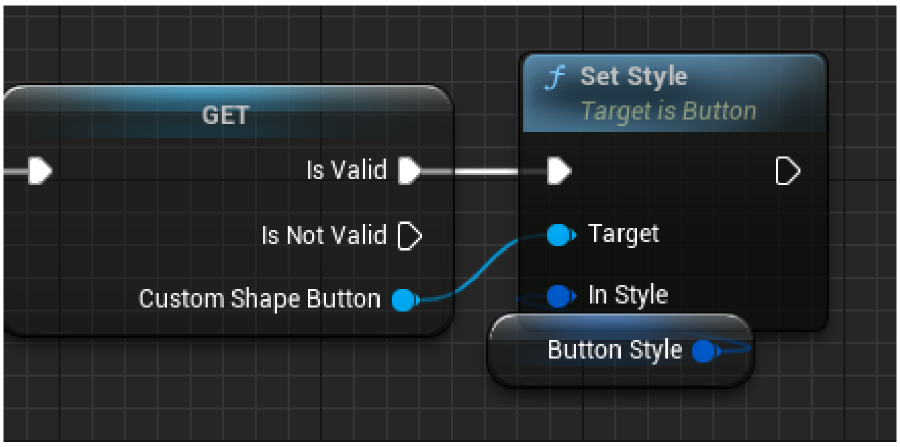
Another example if you have multiple custom shape buttons on the screen which might overlap each other both in 3D and 2D space.

This ensures that when you hover or click on the overlapping area, only the top button will react. Without setting Overlap Order, both buttons might compete for input or none would respond correctly.

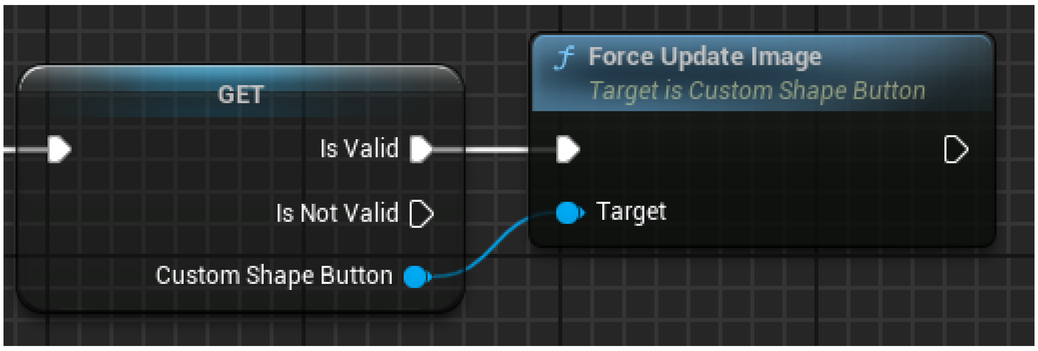
You only need to set this in complex UI with actual visible overlap. In radial menus, for instance, this is often unnecessary - if buttons don't truly cover each other (only their transparent areas do), interaction will still behave correctly without setting Overlap Order.

Public functions

Set Style:
Alternatively to steps 4A\4B, where image is set from Details panel of the button, it also can can be set right in blueprints. It might be useful if Style is contained somewhere else (like in another widget):



Force Update Image:
Forces to update the pixels data about current image. Is not recommended to use at all since often updates are expensive. However, can be useful if button changes in runtime (new texture set or material is changing dynamically). By default, image is cached only once at the beginning.



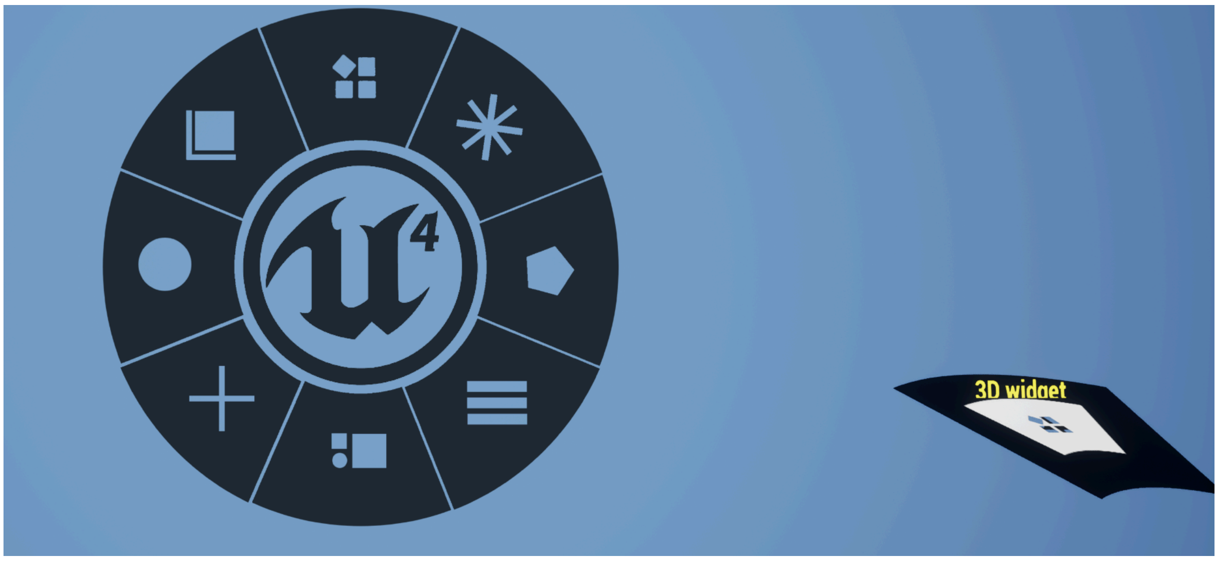
Sample Project

Please download the sample project that demonstrates how it works:
https://github.com/JanSeliv/CustomShapeButton/releases

This project provides a practical example of creating not just 2D widget buttons but also 3D widget buttons.

UE logo is material with alpha mask applied inside while all other buttons are simple textures. Both materials and textures work fine.

It's a great starting point for understanding and utilizing the plugin to its full potential.



2D UI

- WBP_2D: main user widget, contains all the buttons on 2D UI.
- T_RadialMenu_X: are regular textures with alpha channel that used for radial menu.
- M_AlphaMask: is material that is used for UE logo Center button. It serves an example of materials usage. Its texure does not have alpha pixels by itself, but we generate the mask that is connected to the 'Opacity'.

3D UI

- WBP_3D: another user widget to be displayed in world.
- BP_Widget3D: actor blueprint with proper setup the 3D widget.