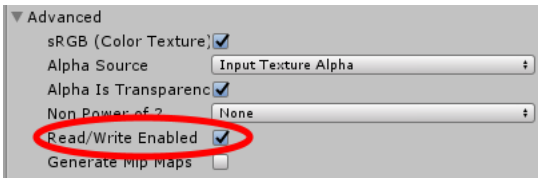
FreeDraw for Unity

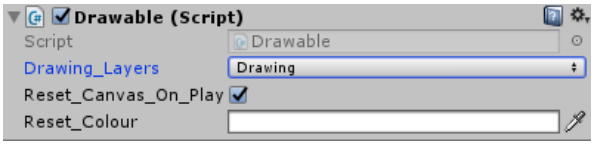
By Michael Long (Foolish Mortals), michael@foolish-mortals.net

The best way to learn is to open the 'ExampleDrawingScene'

1.1 Setup

1. Select the Sprite you wish to draw on, and in the Import Settings check 'Read/Write Enabled'.
   THIS IS NECESSARY FOR DRAWING TO WORK
2. Drag the read/write enabled sprite onto the scene, and attach a 2D collider of some sort to it (probably a BoxCollider2D)
3. Attach the `Drawable.cs` script to your Sprite GameObject
4. Set the layer of your Sprite GameObject to something unique, and then set the 'Drawing_Layers' property of the Drawable script to match that layer
5. **Optional**: use the helper methods in `DrawingSettings.cs` to change both the colour and width of the drawing pen





If you want a larger/smaller canvas: Create your own transparent PNG images with the right dimensions to suit your own needs, and use that.

IF YOU WANT TO BE ABLE TO USE TRANSPARENCY/THE ERASER: Start with a completely transparent PNG image. JPEGs cannot be transparent, and if your starting image is not transparent, you cannot apply the alpha properly. I may look more into this when I have the time.

1.2 Code

You must reference the `namespace FreeDraw` to reference any of the scripts.

1.3 Static Properties

`Drawing.Pen_Colour`: Changes the drawing colour

`Drawing.Pen_Width`: Changes the width of the current drawing pen. An integer given in pixels, it is actually the pen radius.

1.4 Known Issues

Changing the scale or rotation of your Sprite object will result in errors. I'll fix it when I have the time.

To fix the error to the right, try changing the 'Compression' in the Sprite import settings to 'None'

1.5 License

1.6 IMPLEMENTING YOUR OWN BRUSHES

FreeDraw only comes with the default 'pen' brush. To implement your own brush, open `Drawable.cs`, then:

- Carefully read the comments in the BrushTemplate()
- Copy and paste the BrushTemplate() function to create your own brush function
- Delegates are used to call your function, therefore your new function must also only take in a Vector2 as a parameter, and return nothing
- To change what brush you're using, create a helper method such as `SetPenBrush()` in `Drawable.cs`, then either call that function from a button, or create a new function in `DrawingSettings` to a hook a button to it
- The initial brush type is set in `Drawable.cs`'s `Awake()` function