What is Colourise?

Colourise is an easy colour theme tool for your Unity project. It's a small tool composed of four Unity C# script files. It came about when I was creating my first puzzle game and I had to change the colour theme of the UI.

The earliest version of the script had an array of colour values, a list to load game objects and a script that iterates through the game objects to change their colour.

That proved to be inefficient so I refactored and eventually used Unity Events.

There was also the challenge of previewing the theme without running the project.

All these led to the current Colourise scripts.

I'm now happy to share with you the latest Colourise scripts for free! I hope the scripts help ease colour management in your Unity projects. Since it's free, you can extend it whichever way you want as long as you keep the original code intact. NOTE: You can change the maximum colours per palette and default colour to your liking.

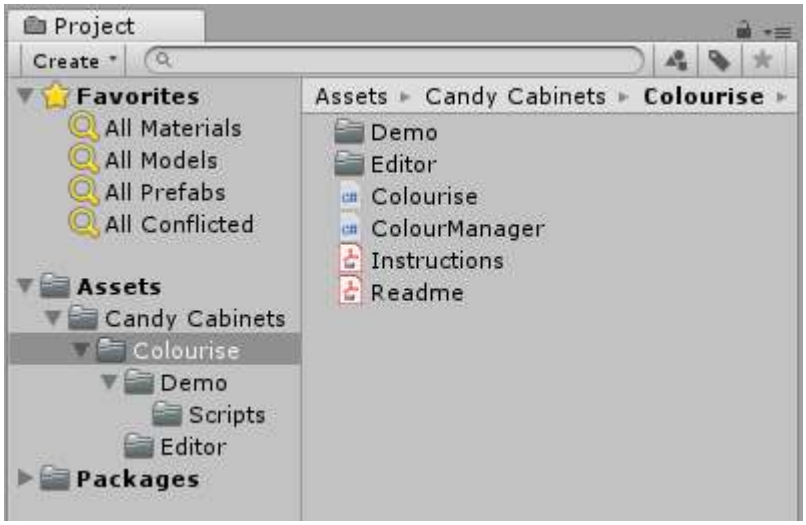
To help you even more, I took some time off my current project to write this mini guide on how-to use Colourise. Although Colourise is really easy to use, it wouldn't be complete without proper documentation ☺

If you have time, please leave a comment or suggestion to help me improve Colourise.

For bugs please email me at james@candycabinets.com

Component Scripts

Component scripts are located in Assets > Candy Cabinets > Colourise folder.



The **Colour Manager** script manages the colour palette of the theme. It has a singleton structure with persistent option. It contains the ColouriseComponent and Palette classes and the Unity Event listener that fires up the ReColour method; the method that colourises the objects during run-time. I also added a couple of useful public methods to the script:

- `void SelectPalette(string paletteName)` – searches and selects a palette. Just pass on the name of the palette, and the script will do the rest. Useful when selecting a palette at run-time. Please run the demo to see this method in action!

- `int SelectedPaletteIndex()` – returns the index of the currently selected palette.

- `string SelectedPaletteName()` – returns the name of the currently selected palette.

- `Palette SelectedPalette()` – returns the currently selected Palette class.

The **Colourise** script connects the color property of game objects to the Colour Manager. It contains the ReColour method that colourises the object, a component scanning method and it initializes the Unity Event listeners. Methods of interests are:

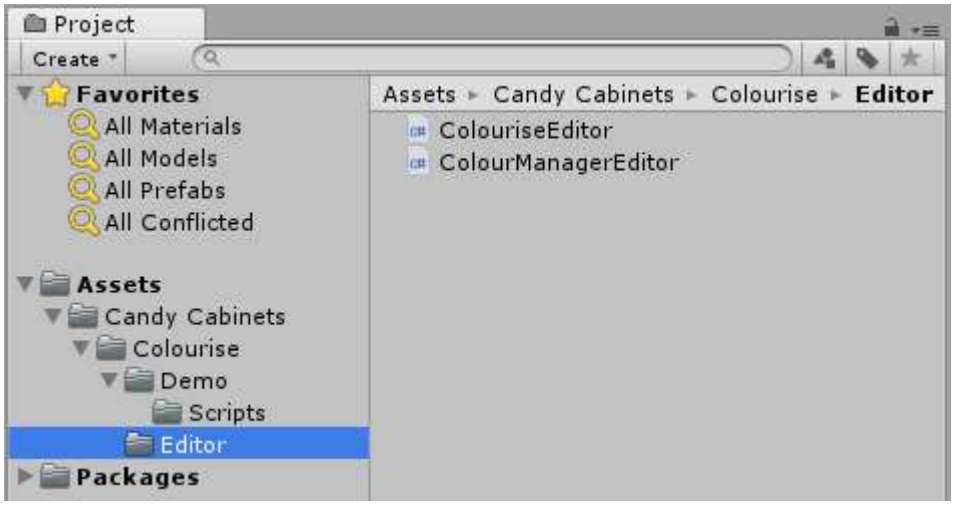
- `void ReColour()` – method that checks the current selected palette and changes the colour of the component if the palette changes.

- `List<Component> ScanForComponents()` – returns a list of components attached to the game object.

- `void SaveComponents(List<Components> comps)` – saves components attached to game object having "color" property.

Editor Scripts

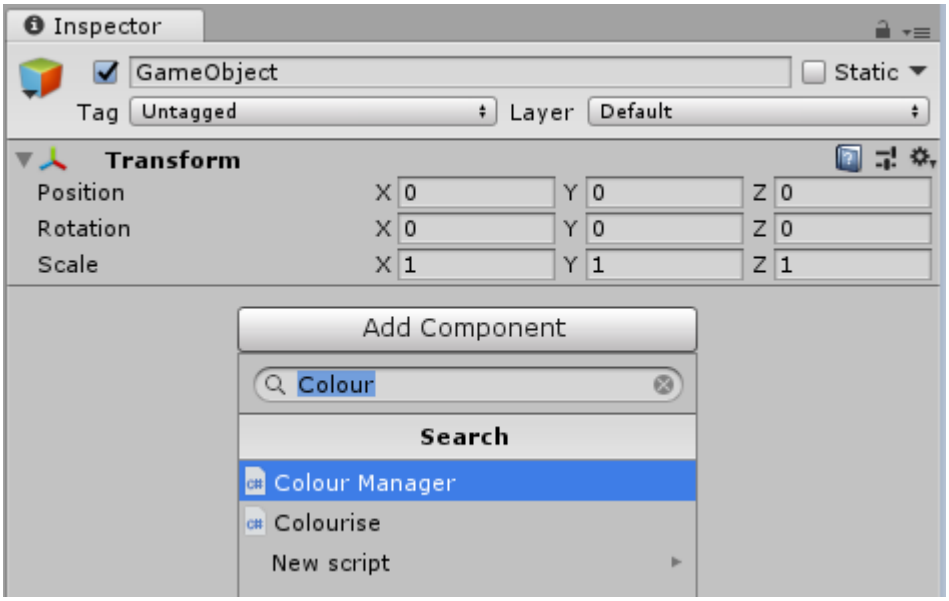
To complete Colourise, we have the editor scripts located in Assets > Candy Cabinets > Colourise > Editor folder.



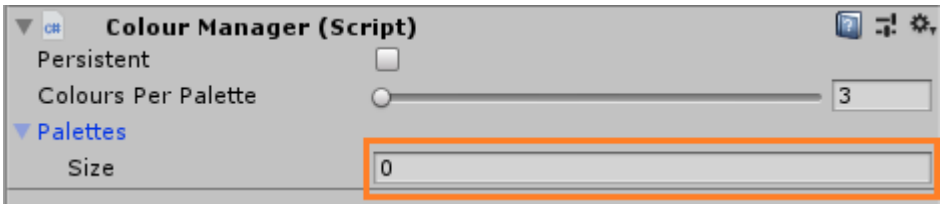
The editor scripts provide; a simple palette list organization when viewing the Colour Manager in the inspector, a colour dropdown selection when viewing Colourise in the inspector and editor preview of the colours when selecting a palette.

How-to use Colourise

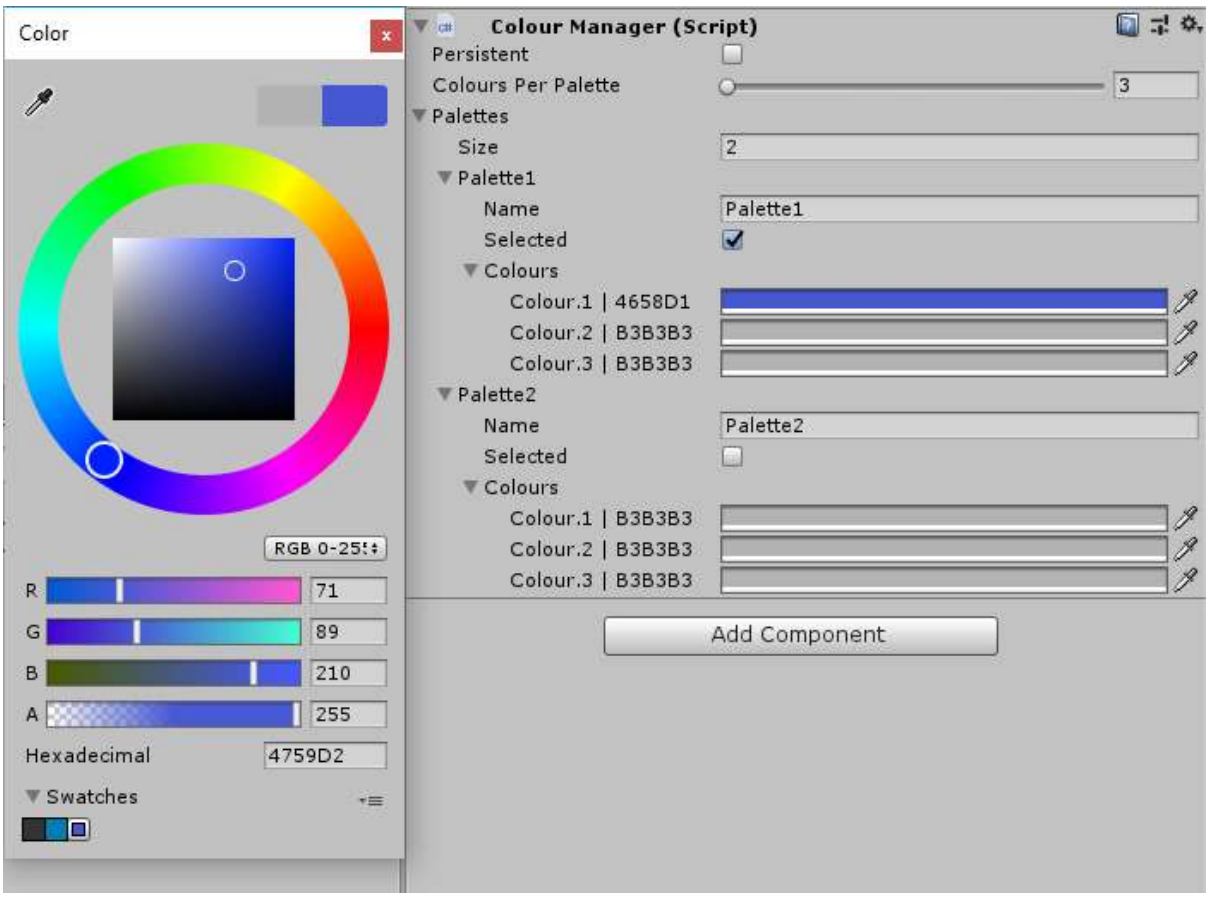
1. Add the **Colour Manager** script to a game object (empty or otherwise).



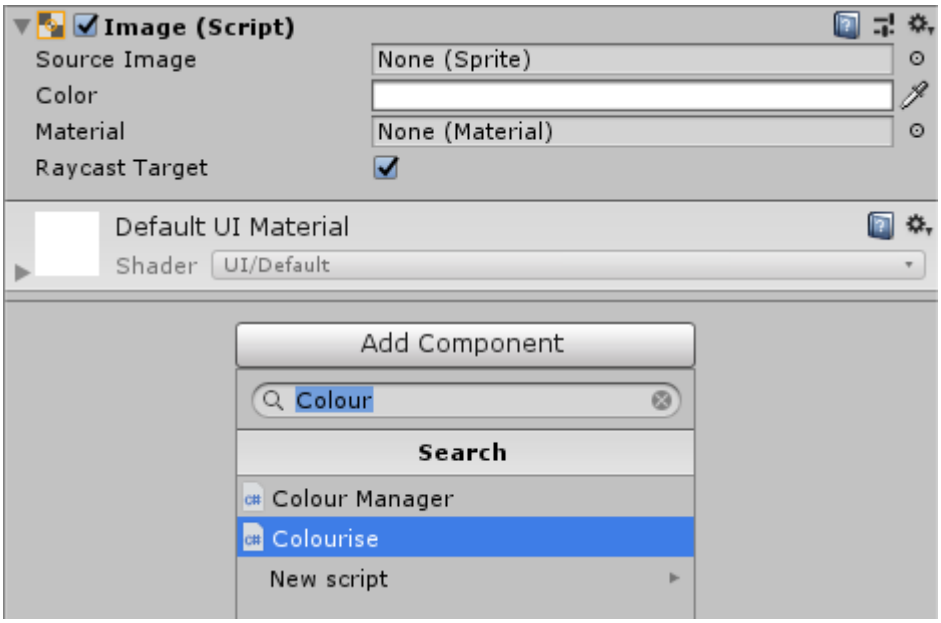
2. Change the **Size** of the Palette list to the number of palettes you want to work with.



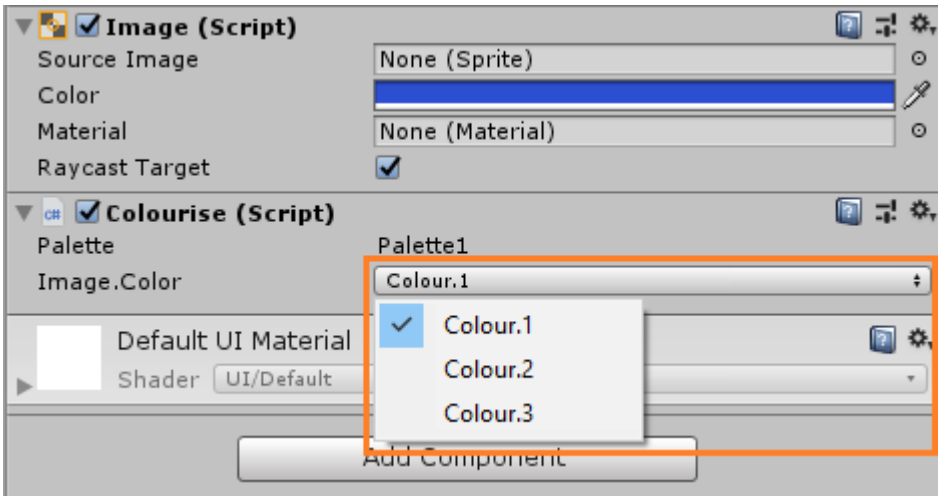
3. Change the **Colour** in your palette.



4. Add the **Colourise** script to a game object with a component having "color" property e.g. Image

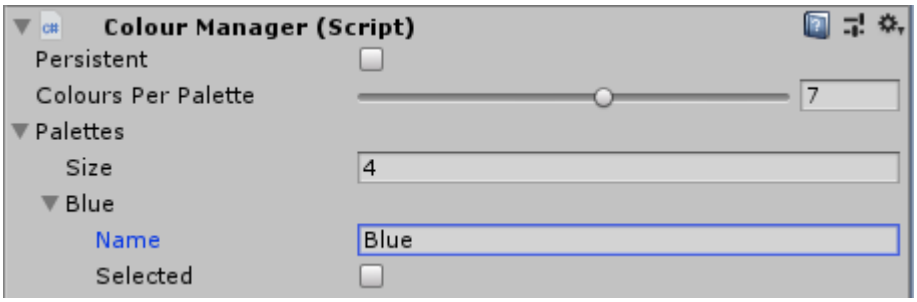


5. Select the Colour you want to use for this component.



Managing Palettes and Colours

1. Naming Palettes – just add the name of your palette in the Name field.



NOTE: Avoid duplicate names. If you have duplicate palette names, only the first palette will be recognized.

2. Adding Colours per Palette – use the slider or type the number of colours per palette. It automatically add/remove the colours per palette.

NOTE: Each palette will always have the same number of colours.

Things to watch out..

Watch out for object renderers that doesn't change colour when exporting in WebGL platform. Like what I've experienced with SpriteRenderer, the script did not find any "color" property so I had to add additional code to capture and change the colour of SpriteRenderer.

Thank you for using Colourise!