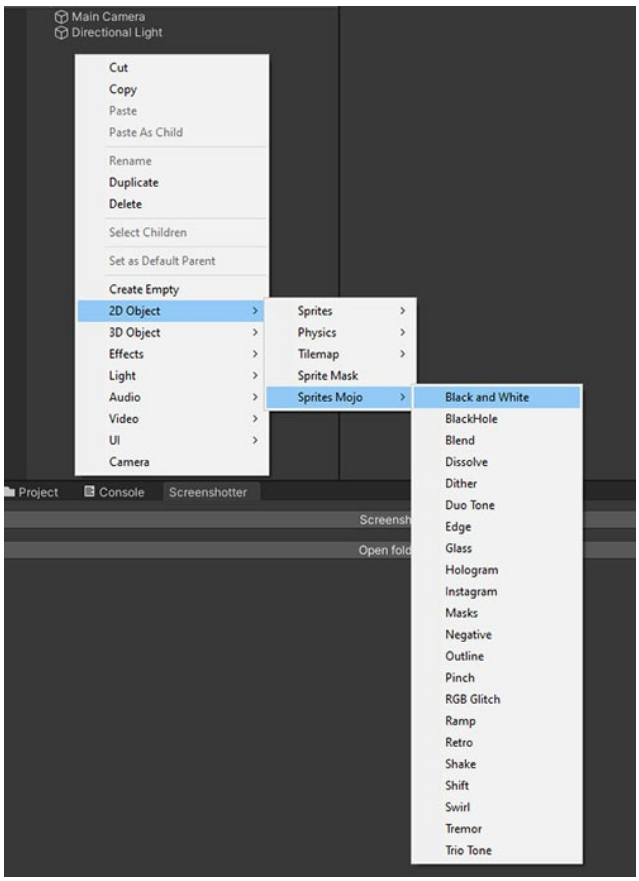
A collection of 2D effects to add more juice to your games

check the online documentation!

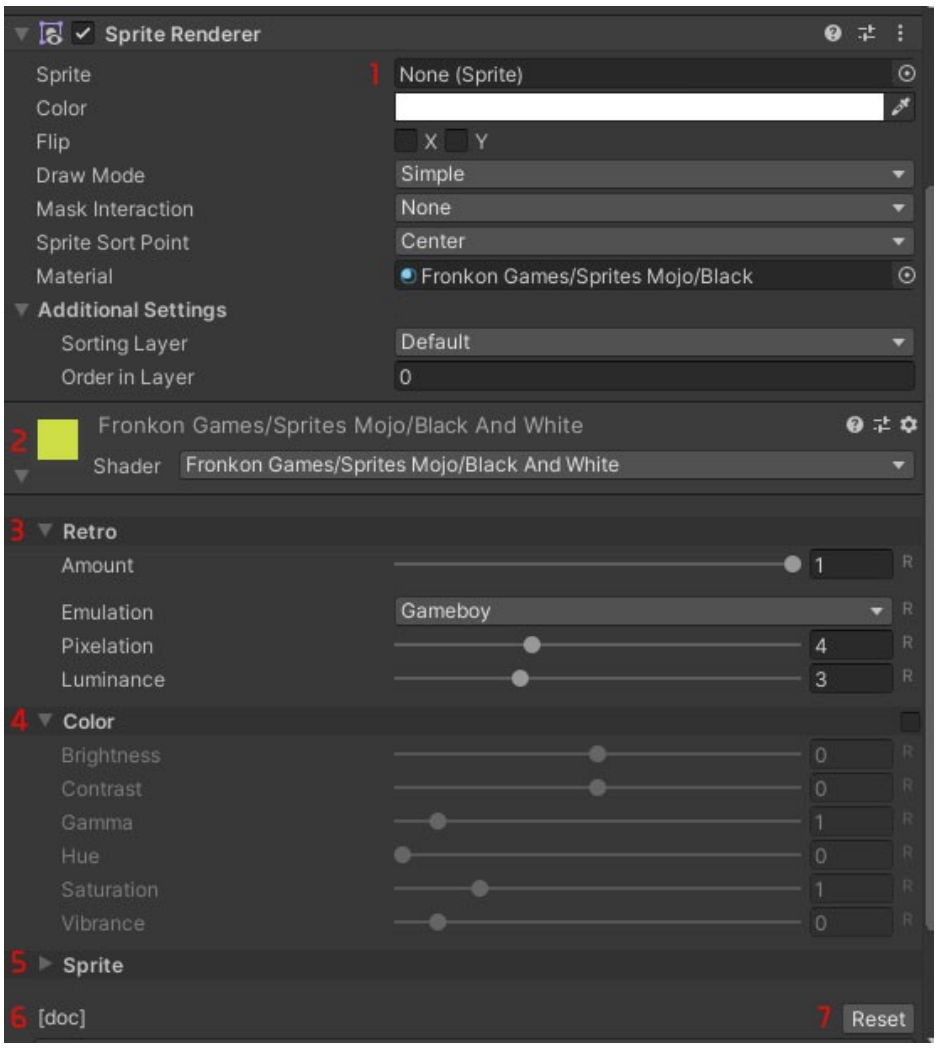
Editor

From the editor you can create ‘Sprites Mojo’ effects in two different ways: creating sprites and creating materials.

To create a sprite that uses one of the ‘Sprites Mojo’ effects, right click on the ‘Hierarchy’ window and select an effect from the list that will appear under ‘2D Object > Sprites Mojo’.



This will create an empty sprite with a ‘Sprites Mojo’ material. When you assign a sprite (1) you will see the effect. ‘Sprites Mojo’ is material based, so you will have to display its material (2) to see its parameters.



All effects have a layout similar to the example above. A first part (3) with the effect parameters. Another part that you can activate to adjust the color (4) and another part where you can adjust sprite values (5).

Finally, by clicking on ‘[doc]’ (6) you can access the online documentation and by clicking on the ‘Reset’ button (7) you will set all the parameters to their original values.

This is the easiest way to create sprites using ‘Sprites Mojo’, but you must keep in mind that it will create a new material for each sprite you create. If your sprites do not share textures between them, no problem. If your sprites share the same texture, the second way is more optimal: creating materials.

To create a material that uses a ‘Sprites Mojo’ effect, just right click on the ‘Project’ window and select one of the ‘Create > Sprites Mojo’ effects. Once the material is created, you only have to assign it to the sprites you want to use it.

Code

All the code is inside the namespace ‘FronkonGames.SpritesMojo’. The first thing to do is to import the namespace:

```
using FronkonGames.SpritesMojo;
```

You can create sprites that use the effects of ‘Sprites Mojo’ as follows (in this example ‘Retro’):

```
GameObject gameObject = Retro.CreateSprite();
```

Remember that each sprite will have a new material. If you want several sprites to use the same material with an effect, you can create it that way and then assign it to the sprites you want:

```
Material material = Retro.CreateMaterial();
```

Each effect has those two functions (CreateSprite() and CreateMaterial()) to create its own sprites and materials. You can find them all in the ‘Fronkon Games/SpritesMojo/Runtime’ folder.

Continuing with the previous example, if you wanted to modify or query any parameter of the ‘Retro’ effect, you can do it like this:

```
SpriteRenderer sprite = gameObject.GetComponent<SpriteRenderer>();
// Changes the sprite emulation mode.
Retro.Mode.Set(sprite, Retro.Emulations.Gameboy);
// Check pixelation.
int pixelation = Retro.Pixelation.Get(sprite);

// You can also do it with the material.
Retro.Mode.Set(material, Retro.Emulations.Gameboy);
float luminance = Retro.Luminance.Get(material);
```

In addition, all effects have some common parameters that you can access through the ‘SpritesMojo’ class that you can find in ‘Fronkon Games/Sprites Mojo/Runtime’. For example, let’s enable ‘ColorAdjust’ if it is disabled and modify some of its parameters:

```
if (SpriteMojo.ColorAdjust.Get(sprite) == false)
{
  SpriteMojo.ColorAdjust.Set(sprite, true);

  SpriteMojo.Gamma.Set(sprite, 1.2f);
  SpriteMojo.Hue.Set(sprite, 0.25f);
}
```

All effect variables of type IntVariable, FloatVariable, ColorVariable and VectorVariable can be set to a value and you can specify how long (in seconds) you want the transition to last from the current value to the one you have set. For example, if you want to transition to a pixel size of 8 in 2 seconds:

```
Retro.Pixelation.Set(sprite, 8, 2.0f);
```

Check the class comments for more information.