

About Palette Swapper

Some time ago, I was looking for a way to change the colors of the sprites in Godot, because modulate and self modulate, included in the engine, weren't what I needed for the project I'm doing... I found several methods , all too complicated for me at the time.

Until I found a shader that looked promising: it was small enough to read understanding what was happening (or a good part of what was happening) and functional... unfortunately I don't remember the author or the name given to the shader, but I thank him (or she) for the basis provided to me.

After much trial and error, I managed to reduce the shader to what I consider to be the bare minimum needed for color swapping using color palettes in a single file. Not going to lie, the process is a little laborious, but it pays off.

This **Palette Swapper** performs the swapping of sprite colors efficiently with color palettes with up to 256 shades of gray/red and many patterns to use. The explanation is also found inside the shader file, but if you don't want to download it, but still want to know how it works, here you go:

Requirements for the sprite

- Can be any image, either grayscale/red or color
- Can be spritesheet or isolated sprite
- The sprite or spritesheet can have any dimension

Requirements for the palette

- Height to powers of 2. See the FAQ for more information.
- Can be any width (shader automatically detects to generate palette indices)
- May have transparency (total or partial)

The shader checks the red channel of the sprite's color to determine which color will be accessed in the palette.

Table for the color palette

IMPORTANT

The shader ALWAYS reads the color palette from top to bottom, no matter how many color columns it has. See before the FAQ of this PDF or in the example project how the palette is structured.

Just for guidance, I made this table to show the color ranges and to avoid unnecessary calculations:

Notes:


- a) By Colors, understand "Number of colors used in the sprite"
- b) By Tones, understand "How many shades of gray/red I separate each color for the palette"

Colors	Shades	Usage example in known games
2	128	Gato Roboto (Switch - 2 colors)
4	64	Any game of Game Boy (Game Boy - 4 colors)
8	32	Super Mario World (SNES - 8 colors per sprite)
16	16	Chrono Trigger (SNES - 15 colors per spritesheet)
32	8	Metal Slug (Arcade - 30 colors per Spritesheet)
64	4	Donkey Kong Country (SNES - 64 colors)
128	2	Ragnarök Online (PC - 98 colors for more complex spritesheets)
256	1	Killer Instinct (Arcade - 150+ colors per fighter)

Searching the internet, in 99% of the cases, the sprites don't come with the exact colors both compared to the original and to the shader (I myself had this problem, with the sprites from Mega Man, to set up the example. I found 6 spritesheets with 6 different shades of blue for his helmet). So as the colors are usually not 100% correct, we need to adjust them.

Any image editor can perform this task, from LibreSprite to Photoshop, going through the Gimp and the like (I'm not considering Paint because despite being able to change what we need, it doesn't work with transparencies).

"Okay, but what do we need to adjust?", you might be wondering... simple: let's change the original color, where the red channel is "wrong" to a value within the scale above. I know it sounds complicated, but it's relatively simple...

 Let's assume your sprite uses 2 colors, black (RGB (0, 0, 0) or #000000) and gray (RGB (120, 120, 120) or #787878).

According to the table below:

Colors	Shades	Usage example in known games
2	128	Gato Roboto (Switch - 2 colors)


I need each color in my sprite to be in a range of 128 tones in the red channel to separate them from each other (well, from 0-127 is one color and from 128-255 is another or in HEX: from 00-7F is one color and 80-FF, another), remembering that if the sprite has more colors, the values change (I'll leave a table with an example using 7 colors, before the FAQ).

Color organization table

Note: By "Range", understand: "range where the color is in the red channel".

Colors in the sprite	Range (HEX)
#000000 ☺	00 - 7F
#787878 ☺	80 - FF

Colors in the sprite	Range (RGB)
RGB (0, 0, 0) ☺	0 - 127
RGB (120, 120, 120) ☺	128 - 255

 If you use the shader without changing the sprite's colors, all the colors will be black because currently the red channel reads 120 (#78....) and that value is within the range that corresponds to black, which the red channel accesses from 0 to 127 (00 - 7F).

So it's enough that, with an image editor, I replace the RGB color (120,120,120) with a close color like RGB (130,120,120), for example. The visual difference for us humans will be minimal, pretty much the same color, but for the shader it's a completely different color. And since the red channel is now at 130, the shader can recognize that color as a second color.

Colors in the sprite	New sprite colors	Range (HEX)
#000000 ☺	#000000 ☺	00 - 7F
#787878 ☺	#827878 ☺	80 - FF

Colors in the sprite	New Sprite colors	Range (RGB)
RGB (0, 0, 0) ☺	RGB (0, 0, 0) ☺	0 - 127
RGB (120, 120, 120) ☺	RGB (130, 120, 120) ☺	128 - 255

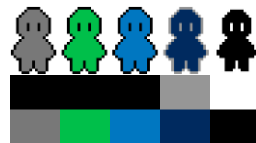


With the sprite with the color adjusted, let's go to the palette: in the example, we now have a sprite with the colors black (RGB (0, 0, 0) or #000000) and gray (RGB (130, 120, 120) or # 827878).

Colors in the sprite	New sprite colors	Palette Cores	Range (HEX)
#000000 🟤	#000000 🟤	#000000 🟤	00 - 7F
#787878 🟤	#827878 🟤	#787878 🟤	80 - FF

Colors in the sprite	New sprite colors	Palette Cores	Range (RGB)
RGB (0, 0, 0) 🟤	RGB (0, 0, 0) 🟤	RGB (0, 0, 0) 🟤	0 - 127
RGB (120, 120, 120) 🟤	RGB (130, 120, 120) 🟤	RGB (120, 120, 120) 🟤	128 - 255

The palette will be an image that is only 2px high and 1px wide for each sprite color variation (remember to put the original colors in the palette, too, so you can switch back to them at any time). So if I plan on using a sprite with 5 color variations (counting the original), my palette would be 5x2px in dimensions.



Enlarged for viewing only

Example table with sprite using 7 colors in HEX:

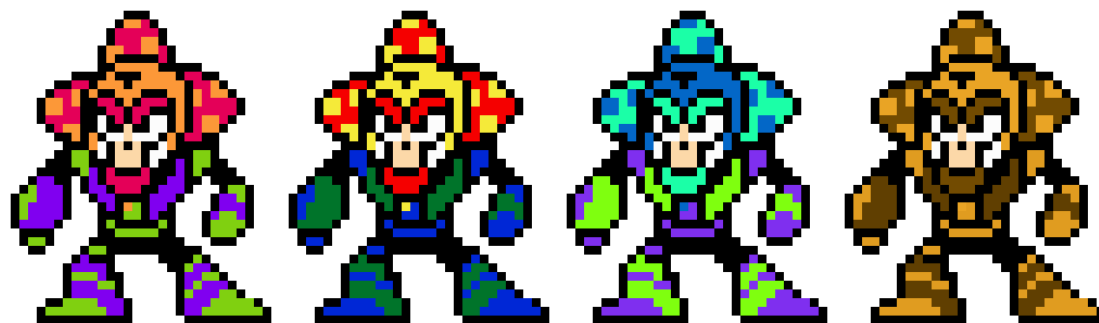
Colors in the sprite	New sprite colors	Palette Cores	Range (HEX)
#000000 🟤	#000000 🟤	#000000 🟤	00-1F
#FCFCFC 🟤	#FCFCFC 🟤		20-3F
#FC9838 🟤	#4C9838 🟤	#FC9838 🟤	40-5F
#FCD8A8 🟤	#6CD8A8 🟤	#FCD8A8 🟤	60-7F
#80D010 🟤	#80D010 🟤	#80D010 🟤	80-9F
#8000F0 🟤	#A000F0 🟤	#8000F0 🟤	A0-BF
#E40058 🟤	#CF0058 🟤	#E40058 🟤	C0-DF
		#FCFCFC 🟤	E0-FF

If you notice, white went to the last position in the table, that's because if a color is detected that has a red that varies from #E0.... to #FF.... it will be replaced by the color #FCFCFC. "But what about that blank space after #000000? Stays like this?" Yes... as there is no color within that range. Notice in the "New sprite colors" column: there is no color that has a red that is between 20 and 3F.

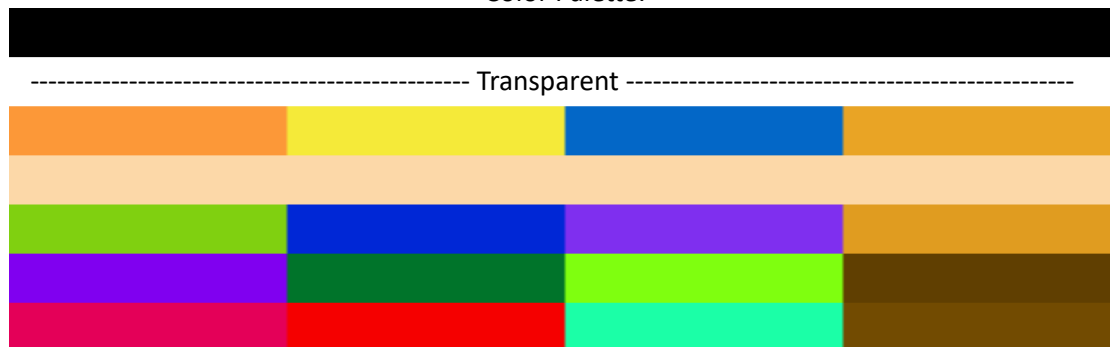
Example table with sprite using 7 colors in RGB:

Colors in the sprite	New sprite colors	Palette Cores	Range (RGB)
RGB (0, 0, 0) 🟤	RGB (0, 0, 0) 🟤	RGB (0, 0, 0) 🟤	0-31
RGB (252, 252, 252) 🟤	RGB (252, 252, 252) 🟤		32-63
RGB (252, 152, 56) 🟤	RGB (76, 152, 56) 🟤	RGB (252, 152, 56) 🟤	64-95
RGB (252, 216, 168) 🟤	RGB (108, 216, 168) 🟤	RGB (252, 216, 168) 🟤	96-127
RGB (128, 208, 16) 🟤	RGB (128, 208, 16) 🟤	RGB (128, 208, 16) 🟤	128-159
RGB (128, 0, 240) 🟤	RGB (160, 0, 240) 🟤	RGB (128, 0, 240) 🟤	160-191
RGB (228, 0, 88) 🟤	RGB (207, 0, 88) 🟤	RGB (228, 0, 88) 🟤	192-223
		RGB (252, 252, 252) 🟤	224-255

If you notice, white went to the last position in the table, because if a color is detected that has a red that varies from 224 to 255, it will be replaced by the RGB color (252, 252, 252). "But what about that empty space after the RGB(0, 0, 0)? Stays like this?" Yes... as there is no color within that range. Notice in the "New sprite colors" column: there is no color that has a red that is between 32 and 63.



Color Palette:



Enlarged for viewing only

As you may have noticed, this PDF also comes with an executable to see how the shader acts and the editable project for you to grab whatever you want from it. I hope you enjoyed reading it, and remember to read the below FAQ as well.

FAQ

Q: What are these channels you talk about so much?

A: Forcing the idea, it is a term used to indicate changes in a part of the color. Changing the red channel means I'm only going to change the amount of red in the color and leave the green and blue as they are.

Q: You said earlier to use images with heights as powers of 2. What do you mean?

A: I said so not to write multiple times that the heights must be 2, 4, 8, 16, 32, 64, 128 or 256 pixels. This is also due to the fact that the calculation divides the 256 colors by the height of the palette, **which always generates an integer number** as seen in the table above. It seems limited, but it gives a wide range of color patterns to work with, given that the width is free.

Q: My palette is not the height indicated. What do I do?

A: **If the height is not a power of 2, use the next value (or the previous value) according to the table presented** (whatever is not used may be transparent or have an irrelevant color). For example, if my sprite has 3 colors, I'll use a palette with 4, which is the next power of 2 and the 4th color can be made transparent, just by not having any channel associated with it. If it has 17 colors, either I remove a color, reducing it to 16 colors, or I use a 32-color palette, if I want to use all the colors in the sprite.

Q: My sprite has few colors. Does it cause any problems to use the exact number of colors in the palette?

A: It depends. The shader is configured to be as light as possible to use. So if your sprite has 3 colors, the calculation for the index is imprecise, generating two or more indices in the same color. Now with 2 or 4 colors this problem does not occur because they are powers of 2.

Q: My sprite has messed up colors, this shader sucks!! Is it not possible to fix this?

A: Calm down, brother! Check if the colors are organized in the palette and if your sprite has the colors obeying the standards I informed in the document above... Remember that the shader follows a logical order to do its work, just follow the rules.

Q: I want to make a simple hologram, can I make the sprite half transparent with this shader?

A: Yes, just put the half-transparent color on the palette (I recommend Libresprite, Krita, Gimp or Photoshop for this task).

Q: I want to make the player partially disappear behind a wall. Does this shader do that?

A: Unfortunately, no. It is for color changes only, with or without transparencies.

Q: My game has a part where everything goes dark, is there a way to leave only the white outline on the sprite?

A: Yes. See an example of this above when I explain how to organize the colors of the sprite and the palette: one of the sprites has a white outline and black color (it could be transparent too).

Q: I don't know anything about shaders. Can I use it anyway if I need to?

A: Of course! All you will need to know is:

- How to insert a “material shader” in Godot (to indicate which palette you are going to use and which column you are going to use to color your sprite)
- A bit of GDScript (to change the palette during gameplay)
- How to create and edit images (your preferred editor)
- And, of course, the information in this pdf (to clear doubts or remember how the shader works)