Character Sprites and Portraits Editing

Author: madsiur
Last modified: 08/05/2017
Credits: FF6 T-Edition Sprite Tutorial by tsushiy, part of the FF6-T Translation by Kain Stryder.

1. Introduction

The character sprites in FF6 can be modified in various ways based on exactly what you want to do. Edits like "*I want to change Terra's casting pose*" or "*I want to create an alternate shadow character sprite*" are possible, however there are elements to consider regarding colors, graphic tiles and animations.

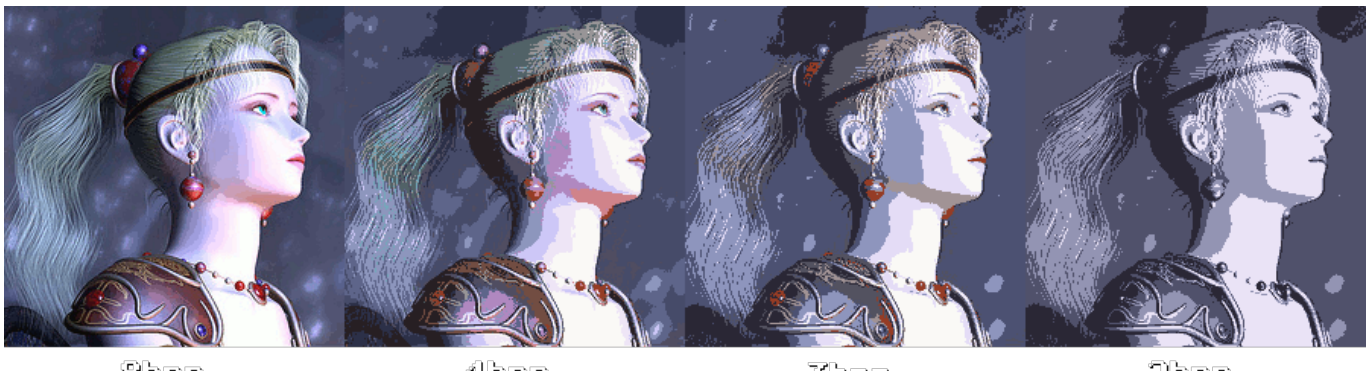
Spriting with FF6 limitations is harder than simply editing a sprite image but not so far from it when you know and get used to the limitations and specifications. There are three utilities that are made for FF6 sprite work and at least one general GFX utility that can be used to edit sprites and portraits. They will not be covered in-depth here as they are relatively straightforward when you know the basics to the exception maybe of YY-CHR that will be covered in depth in a future tutorial.

You'll notice I mainly refer to the generic *FF6* name. *FF6* shall be understand as SNES FF3us 1.0 or 1.1 but a lot of element discussed here can be applied to the GBA versions. However not all differences will be covered or even mentioned. Utilities discussed apply only to FF3us unless mentioned otherwise. Offsets given all all Hi-ROM offsets.

2. Understanding Sprites

Skip this section if you don't want to read about the technical side of sprites. However references to these explanation in simpler ways will be made in the tutorial.

SNES sprites are the combination of two elements: graphic data (GFX) and a palette. First the GFX can have a different maximum number of colors, which is called the graphic format. Most SNES graphics are 4bpp (4 bits per pixel), meaning they can have a maximum of 16 colors (transparency color + 15 colors). There is only 16 number possibilities with 4 bits (0 to 15) and they are in binary 0000, 0001, 0010, 0011, 0100, 0101, 0110, 0111, 1000, 1001, 1010, 1011, 1100, 1101, 1110 and 1111. However there are game GFX that can be 3bpp (8 colors maximum) like some monsters or 2bpp (4 colors maximum) like the fonts.



To easily picture a GFX, imagine in the case of a 8×8 4bpp graphic a 8×8 array or table where each pixel is represented by a number from 0 to 15. Since we are in 4bpp, one byte (8 bits) can hold 2 x 4 bits, so 2 pixels per byte. Instead on an array of 64 bytes (where GFX format would be one color per byte), you GFX is 32 bytes since 2 pixels fits on one bytes. As an example a byte such as 00000010 is the combination of 0000 (0) and 0010 (2). This byte contains therefore a pixel of color 0 and a pixel of color 2. Don't worry if you struggle with this, this is the detailed technical side and mastering it is not required at all to sprite. Same goes for the coming palette description.

Palettes are a sequence of color data that is “applied” to the GFX. Color 2 of the palette is “applied” to all the pixels of color 2 of the GFX. This method allow to reuse GFX with different palettes. Palettes are usually 2, 4, 8, 16 or 256 colors (Mode 7) but there is no data in the palette saying its length. It is specified in the game code how many palette bytes anf GFX bytes to read to display correctly a certain graphic on-screen.

A SNES Color is in 15 bits RGB format. In short, this mean each color can stand on 2 bytes (16 bits). Each channel value (R, G or B) is 5 bits and the first bit is always 0. The specific format on 2 bytes is **0BBBBBGG GGGRRRRR**. A 5 bits number can be between 0 and 31 meaning there are 32 767 possible colors on SNES (32 x 32 x 32). More modern color formats like 24 bits RGB and above (1 byte per channel) have a 0-255 range for each color channel. Translated in 24 bits RGB, a SNES color will always end by 0 or 8 and 1 on a 1-31 scale is 8 on a 0-255 scale. If we take the hex color notation, a valid SNES color could be 18,20,18 (sort of black) but the 24 bits RGB hex color 18,19,18 has no SNES equivalent (0x19 cannot be set on a 0-31 scale because 0x19 divided by 8 equals either 0x18 + 0x01 or 0x20 - 0x07). To be the closest possible, this color should be set as 18,18,18 so in other word 3,3,3 in SNES palette editing programs. Most if not all spriting utilities does the color *rounding* to take in account free edits in Gimp and such, so you don't need to worry about the color *rounding* at all.

So we know a SNES color fit on 2 bytes, therefore a 16 colors palette is 32 bytes in the ROM. This is valid for all SNES games. As quickly mentioned previously, transparency is a color, there is no alpha channel in the 15 bits RGB format. The game determine which color is transparency. In the case of FF6, it is often color 0. So this color on the palette will have a RGB value but it will be *masked* by the game to show transparency instead.

This cover basically the technical side of colors, palettes and GFX. Little of what explained is mandatory as an example for editing a sprite in FF3usME but I think it is good to understand the byte level working of these elements. If you are using an utility that let you import images (.png, .bmp, .gif, .jpg, etc.) you should work in your image editor such as Gimp in format indexed 16 for 4bpp graphic.

Indexed mean the image data is an array of color ID like SNES GFX and *16* is maximum number of colors. Depending of what you use, the all purpose image editor maybe have *Indexed* format only and you set the maximum number of colors yourself. This is always a safer approach than editing an image in 32bpp ARGB format and adding too much colors or even transparency.

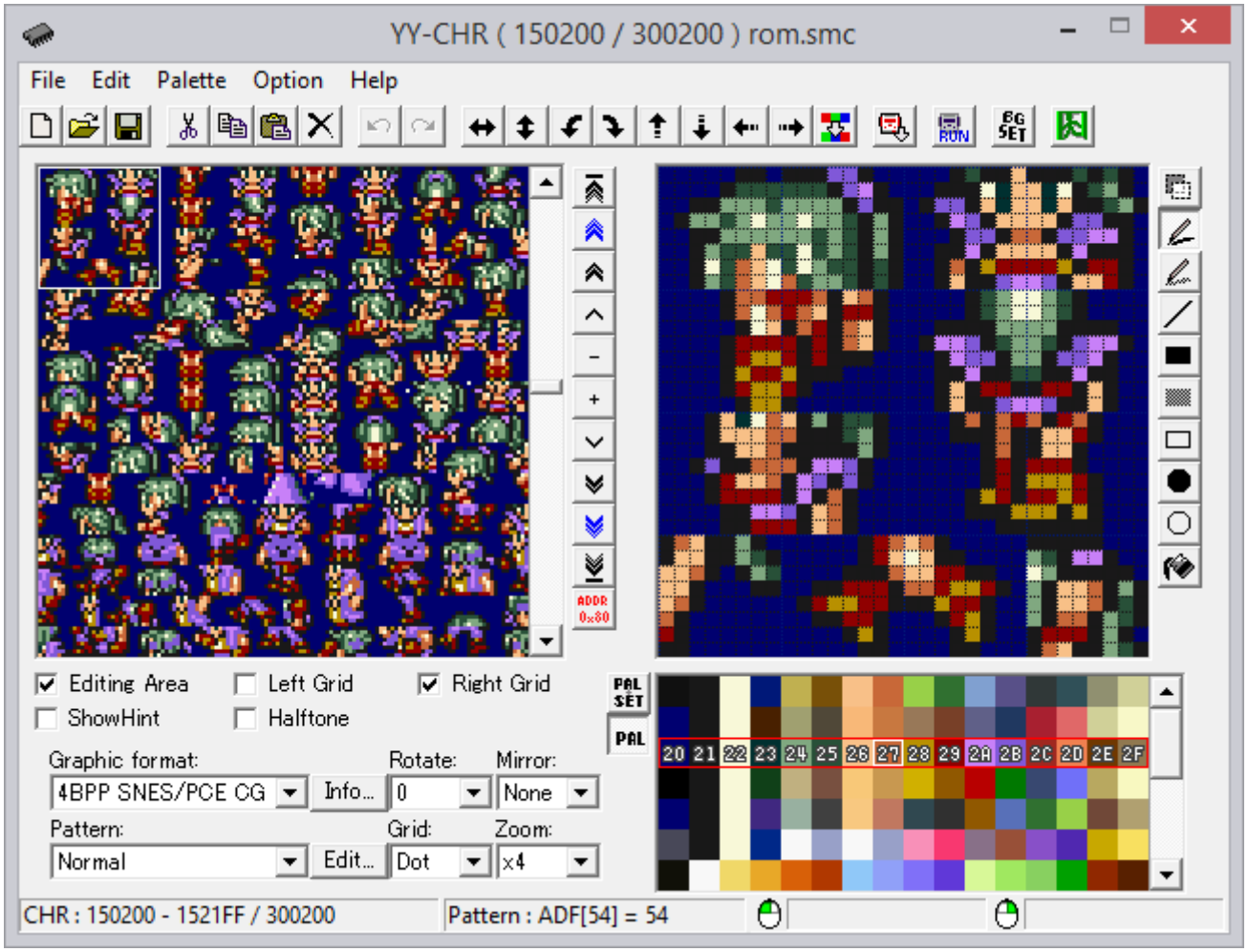
3. Editing Tiles (YY-CHR)

If you wish to do simple tile edits, YY-CHR can be a solution. There is a .NET version and a C++ version. I personally go with the C++ one, it is older but has more features that were not all ported (yet) in the .NET version. The image shown here is from the C++ version.

Character sprite are located at $D50000 ($D50200 with header). You can either enter that address (as $150000 or $150200 if header present) or scroll until you see the GFX tiles. Note that you need to set to 4bpp SNES the graphic format.

If you want the right colors, *i.e. the right palette*, you can import the FF6 NPC Palettes that you can import in YY-CHR by going to *Palette → Open Palette (*.pal)*. This make editing a lot easier but is not mandatory, as the palette shown in YY-CHR is not saved in game and the GFX is not modified by this step since it use a palette (see section 2).

There different tile arrangements available. Since we are dealing with 16×24 sprites poses for characters, a more convenient way to view the tiles would be 16x24. If you had a 32×32 NPC sprite to edit, choosing 32×32 as tile arrangement would help editing. As previously stated, other utilities are more useful than YY-CHR. FF3usME and FF3SpriteEd both have a tile editor (see here and here). FF3SE on the other side only allow only sprite sheet editing but follow the same tile logic as for saving the sprite in the ROM.



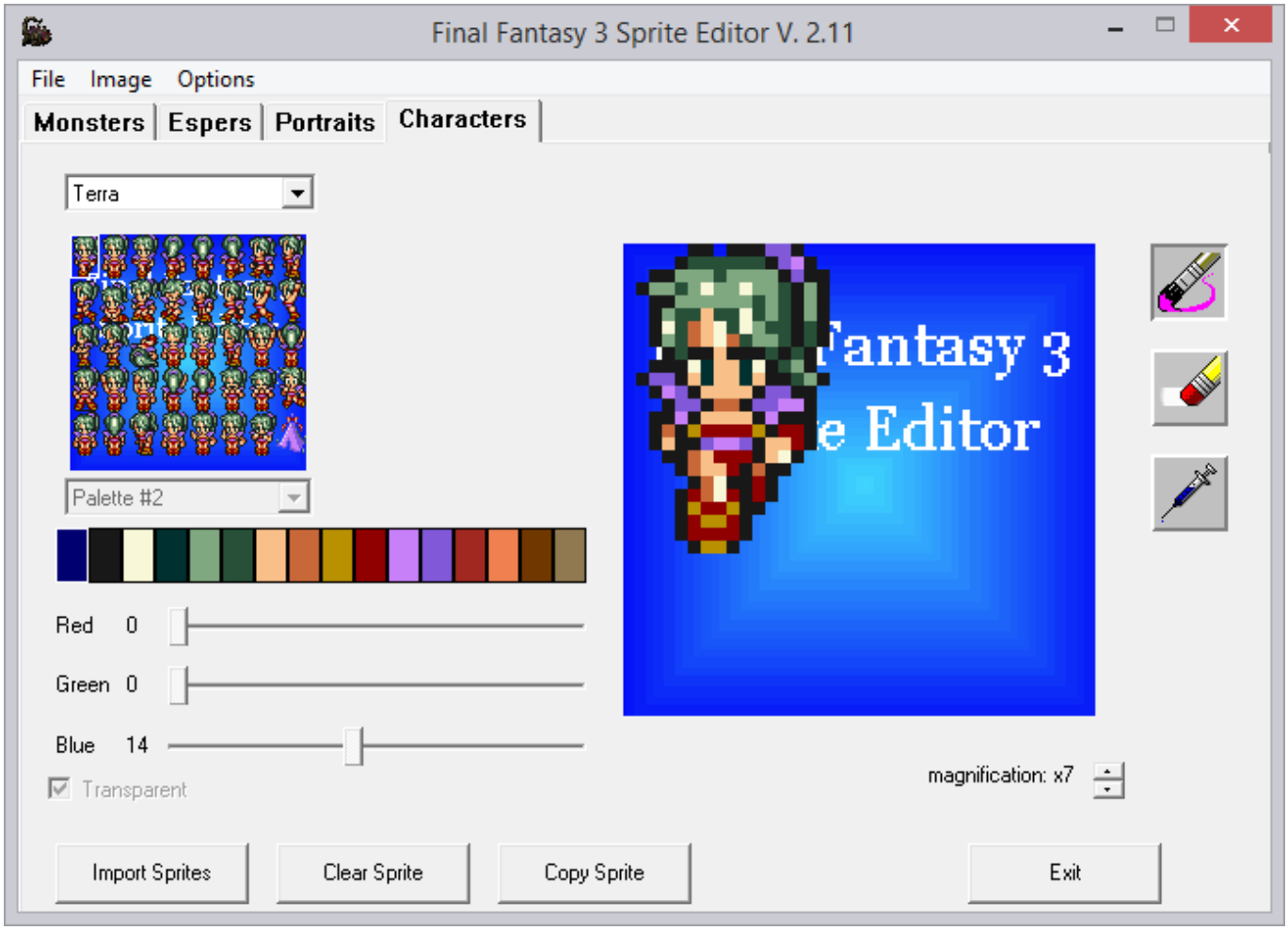
To summarize this section, YY-CHR can do some basic job but for complex sprite editing, FF3usME, FF3SpriteEd or FF3SE are a lot more convivial and allow a faster working pace.

4. Editing Sprites

This section will cover the FF6 spriting utilities features and way to edit sprites using FF3usME, FF3SpriteEd or FF3SE.

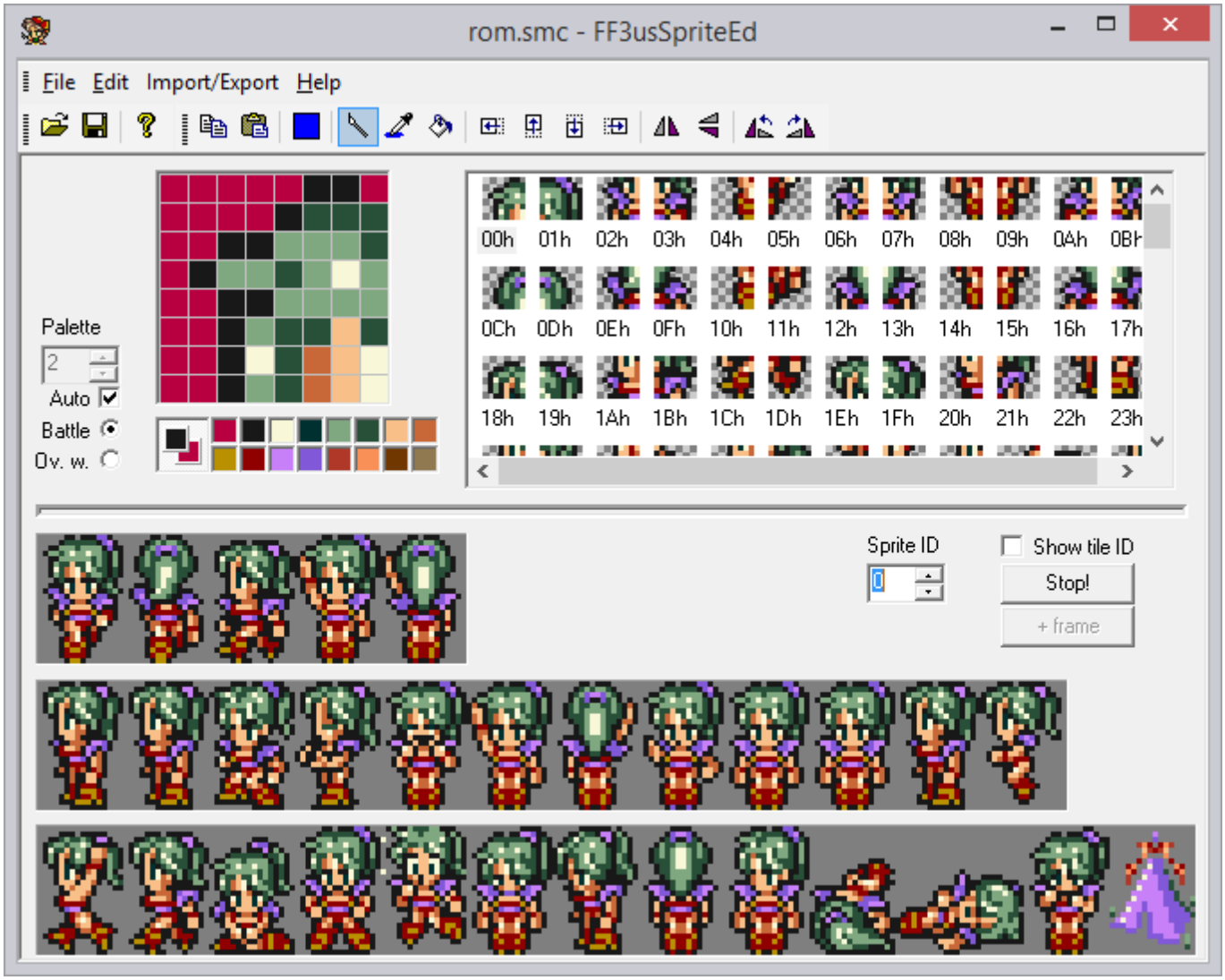
A. FF3SE (no longer developed)

First FF3SE has no tile editor but allow sprite sheet editing. Its features are more basic than FF3usME or FF3SpriteEd. This can be more difficult to deal with if you are a novice and don't know which tiles repeat. Also there is no way to animate the sprite to see if your work is correct or need adjustments. Note that there is a “trick” to correctly import a sprite with good palette order every time. You simply have to draw your palette on first pixel row like shown here. This “trick” can be used in FF3usME, FF3SpriteEd or any editor that get the palette in a *top-left to bottom-right* way.



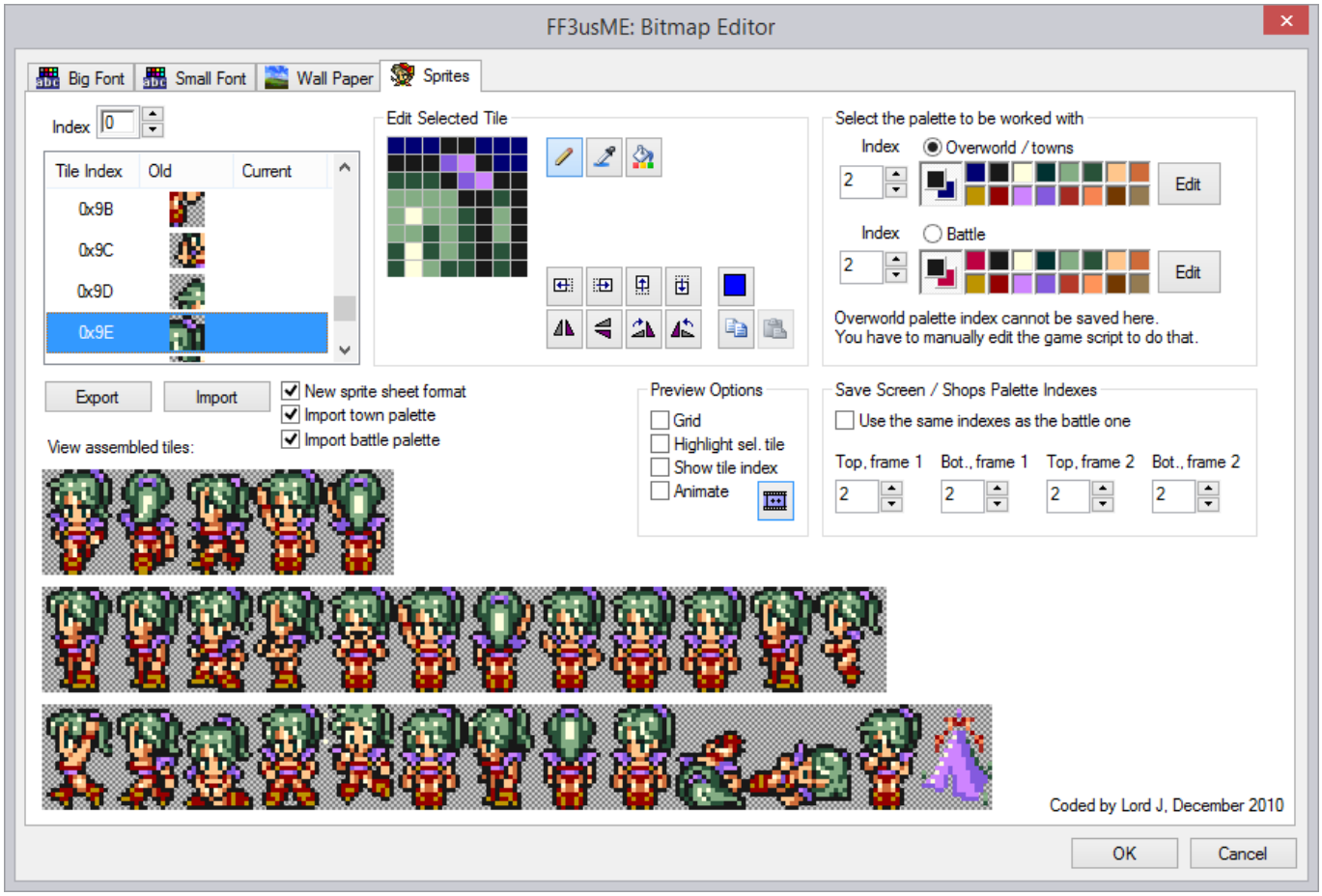
B. FF3SpriteEd (merged with FF3usME)

Second come FF3SpriteEd, the predecessor of the character sprite editor of FF3usME. They are somewhat close to each other. Without detailing everything, one of the major difference is the binary import / export for the tileset and palette import / export (.pal) that can be done separately. The advantage of these types of import is you can apply a new palette to the tileset in a single click. However the tileset nor the palette can be imported elsewhere, making FF3usME PNG import / export more convenient.



C. FF3usME (still being developed)

Finally there is FF3usME that is now at version 6.80. The character sprite editor is a bit more advanced than FF3SpriteEd and the PNG import / export and FF3SE import compatibility makes it a candidate of choice. All editors are pretty straightforward so little explanation is required.



To conclude this section, I'll talk about the spritesheet formats. FF3SE export the sprite sheet in the *old* format, meaning they have no riding and dead pose included. It is also the only format it can import. FF3usME use a close but newer format that include the dead and riding poses. Lord J's editor can import / export in the *old* or *new* format making FF3usME more versatile than FF3SE or FF3SpriteEd that only has binary import / export.

5. Sprite Animations and Poses

Each character sprite in FF3usME consist of 17 sprites animations and 13 static poses, at total of 46 static poses (~92 if we count flipped poses) and a maximum of 181 different tiles making those poses. Each pose consist of 6 tiles. Many tiles repeat in different poses. This animation count is only a reference for sprite editing in FF3usME, it does not represent all the possibilities of the game. However the poses and tiles counts are exact as per what the game has.

FF3usME has 2 kind of animation: *three poses animations* following a 1-2-1-3 pattern and *two poses animations* following a 1-2 pattern. There are 5 *three poses animations* and 12 *two poses animations*. A FF3usME animations table and poses table with corresponding tiles IDs (on images) are available:

FF3usME animations table
FF3usME static poses table

For a short description of each pose ID, refer to the movement action codes used in the action queue of characters in the event code.

The possible animations and pose patterns are used mainly in the event code (as well as battle module and world map module). For events, you can call a pose in an action queue of a character or NPC (event commands $00-$35). Walking combinations have their own set of movement actions ($80-$AB). You can call a single pose in an event queue with graphical action $00-$7F. Actions $00-$3F are can be view in FF3usME for the most part and $40-$7F is the horizontal flip of $00-$3F.

If we come back to spriting, is it important to make sure all the edited poses work correctly with the sprites animations shown in FF3usME. Editing a tile can sometime have an effect on a pose we did not suspected at first glance. Get used to the tile sharing of the sprite poses, that is the same if we look at all sprite sheets of the same size.

6. Editing Palettes

One of the biggest puzzle for those who have sprite with different palette is making sure everything fit together. Why? Because sprite share palettes! As an example, if you change Celes hair color to blue, Sabin, Edgar and Leo's hair will be blue.

Also there are two 16 colors palettes assigned for each character, one palette for battle and one for maps and overworld. Most of the time they match but they are exceptions. There are 8 battle palettes (located at $ED6300) and 32 overworld palettes (located at $E68000). Here are the first 7 of each

type:

They are both assigned in a different way. The battle palette is assigned via a table of 26 one byte entries for the palette ID (14 first entries being the main cast in regular order). This table is located at $C2CE2B. As for the overworld palette it is assigned by event usually when recruiting the character. *Map characters* are reused, as an example character $07 is first a moogle, then a ghost to finally become Strago. There is many palette assignations needed in a case like this.

This palette and sprite changes document made by runelancer is quite handy to find which event code need to be changed. Check also Lockirby's Event Document for the use of command $43 that change the palette. As an example if you want to set Locke to palette 05, the command would be 43 01 05, because Locke is actor 01.

While there are many overworld palettes all regular characters and *character NPCs* use a palette ID between 0 and 5, with exception of Esper Terra that use overworld palette 8 and battle palette 6. The reason for you being unable to use 6 and 7 is due to those palettes not being able compatible with save/store/character selection screens. For NPCs, the palette ID is located in the NPC data, it's something that can be edited easily with FF6LE. Here are two tables of the palettes used by the main sprites and main NPCs. Some NPCs can be used with different palettes such as most generic NPCs, however they only appear in the table with their default palette.

Overworld palette

Palette ID	Sprites using the palette
00	Edgar, Sabin, Celes, Imp, Leo, Ghost, generic elder, generic man, Maria, Rachel
01	Locke, Imperial, Merchant, Scholar, Returner, Clyde, generic woman, generic boy, generic girl, Narshe guard, sailor
02	Terra, waitress, Figaro Sergent, Figaro Guard, Katarin, Darryl
03	Strago, Relm, Gau, Gogo, Banon, Kefka, Gestahl, Gau (suit), Cid, generic thief
04	Cyan, Shadow, Setzer, Interceptor, Draco, Arvis, Matron, pigeon, maestro, Maduin, Vargas
05	Mog, Umaro, Ultros, Chupon
08	Esper Terra

Battle Palette

Palette ID	Sprites using the palette
00	Edgar, Sabin, Celes, Imp, Leo, Ghost, generic elder, generic man, Interceptor
01	Locke, Imperial, Merchant
02	Terra
03	Strago, Relm, Gau, Gogo, Banon, Kefka, Gestahl
04	Cyan, Shadow, Setzer
05	Mog, Umaro
06	Esper Terra

Finally when you edit your sprite and colors, there is an order to respect. One of the reason is explained in next section and also why you can only use the first 12 colors for a battle sprite:

Color ID	Description
00	transparent
01	outline (black)
02	eyes
03	pupils
04	hair (lighter shade)
05	hair (darker shade)
06	skin (lighter tone)
07	skin (darker tone)
08	outfit 1 (lighter shade)
09	outfit 1 (darker shade)
10	outfit 2 (lighter shade)
11	outfit 2 (darker shade)

7. Other Specifications

A) You can set different sprite for battle and overworld

While the vanilla game has same sprite for battle and NPC or overworld PC, you could set a different sprite for battle. The game get a sprite GFX index for each battle sprite in a pointer table at $C2CE43. Each entry is 3 bytes and there are 24 entries. As for NPCs, this is also in the NPC data which can be edited with FF6LE. Finally playable characters get their GFX set with an event command in the same way the palette is assigned, except it is with event command $37.

B) Color changes in battle

One thing you must keep in mind and be made aware of is battle sprites have color changes during battle. To be more specific, the outer line of characters glows from effects such as Haste, Protect, etc, and their skin tone changes due to Poison, Zombie, etc. These change the colors of the specified number on the palette:

#1 is what's used for the outline of magical effects around the character, and #2 and #3 are what's used for skin color changes like Poison, Zombie, etc. Please note if you use the skin color for clothing or draw black outlines around your eyes, this will make them change color with the above. Personally, I don't think it's super noticeable, so please don't worry too much.

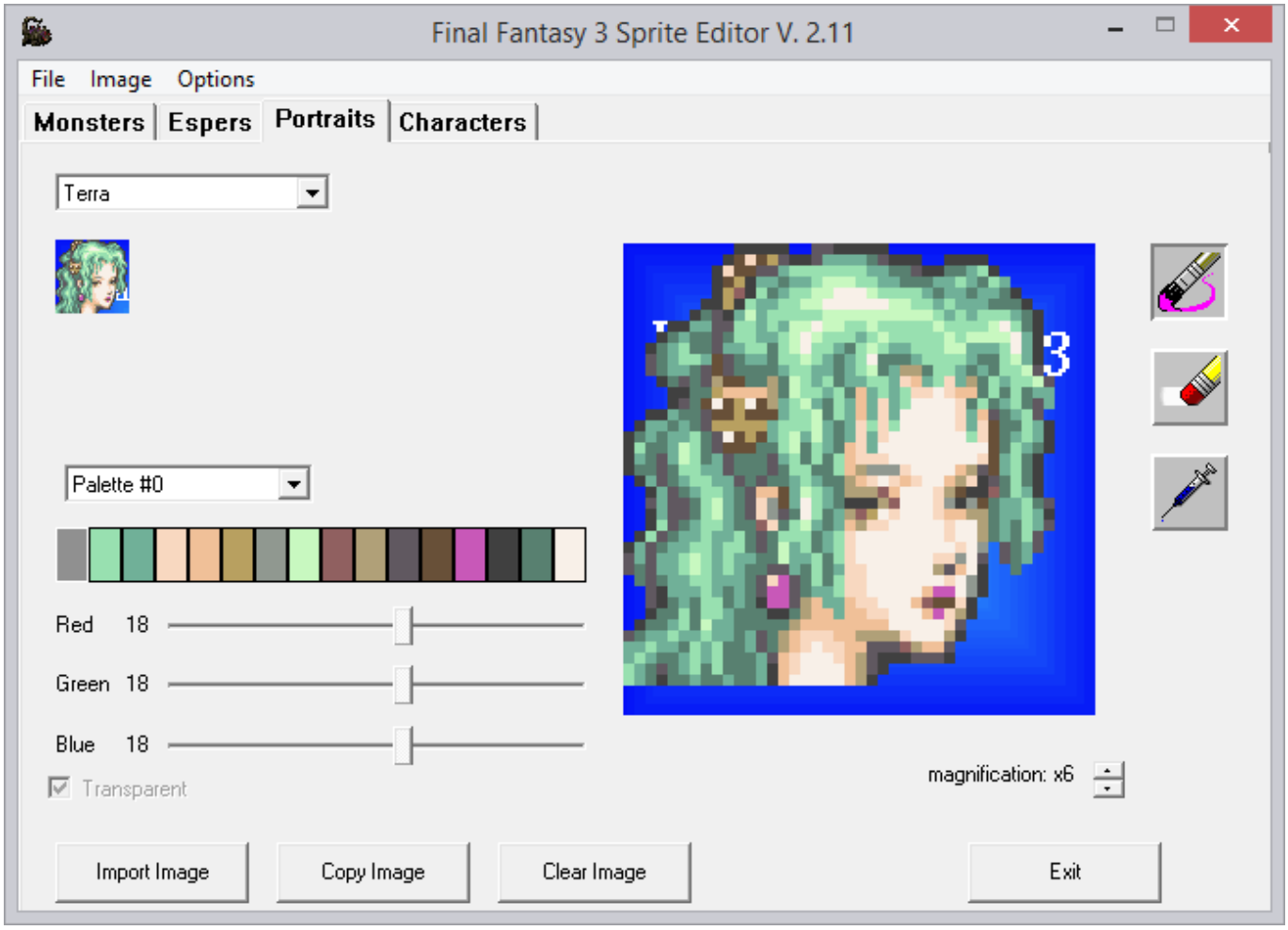
C) Color changes in battle

The total number of colors usable for battle sprites isn't 16 actually, but 12. Out of the 16 colors, 4 are masked via the system colors used (finger cursor, damage font color, etc). There is however a Expanded palette Hack that can make use of those colors and allow you to edit them.

However using this hack will make characters on palette 4 and 5 display the last 4 colors save/store/character select screens incorrectly. This is due to the last 4 colors of palette 4 being temporarily overwritten by the color of the status ailment on the menu screen and the last 4 colors of palette 5 being used for the color of the cursor.

8. Portraits

Portraits are 40×40 with a palette of 16 colors. Each portrait has its own palette. The easiest way to edit portraits and import custom ones is with FF3SE. Note that the "trick" to have correct import by drawing your palette on the first pixel row also work here. However it is not mandatory but a workaround if you get a bad import.



If you ever edit directly in the ROM with as an example YY-CHR, the tile order is as follow:

9. Conclusion

Becoming a good FF6 spriter require to take a lot in consideration. However many started from nothing and became good with time. It's more a matter of practice and knowing the basics, specifications and exceptions before starting for real. This is what I tried to do with this tutorial. I might revise it a bit but I covered most of the content I wanted to cover.

Tools used

FF3usME 6.80
FF3SE 2.11
FF3usSpriteEd 1.1
YY-CHR (C++)