



---

# SPRITE EDITOR JR

---

F256 JR



<https://github.com/econtrerasd/F256-spreditjr>

## Table of content

REQUIREMENTS.....	3
DESCRIPTION .....	3
FUNCTION ICONS .....	5
 SAVE .....	5
 LOAD .....	6
 RESIZE SPRITE .....	6
 ADD SPRITE .....	7
 DELETE SPRITE.....	7
 MOVE TO NEXT SPRITE .....	7
 MOVE TO PREVIOUS SPRITE.....	8
COLOR PALETTE.....	9
Picking a Color from the Palette .....	9
RGB Value.....	9
Picking a Color directly from the GRID .....	9
Editing New colors.....	10
Default Palette Color Credits.....	10
SPRITE EDITOR GRID AND SPRITE REPRESENTATION.....	11
DRAWING TOOLS ICONS .....	12
 BRUSH SIZE.....	12
 COPY.....	13
 PASTE .....	13
 CLEAR .....	14
 GRIDLINES.....	14
 MIRROR X.....	14



MIRROR Y ..... 14

SPRITE FILE STRUCTURE ..... 15

LOADING A SPRITE FROM BASIC ..... 16

LIMITATIONS, KNOWN ISSUES ..... 17

CONTACT INFORMATION ..... 17

## REQUIREMENTS

The Sprite Editor requires the following to operate:

1. Foenix F256 Jr (any model)
2. Joystick in Port 1 -or-
3. PS/2 Mouse\*
4. SD card or IEC Drive if you want to save your creations.



\* A Perixx PS/2 Mouse is highly recommended since its resolution defaults at 400 dpi, higher resolutions can "overwhelm" the built in mouse driver in the F256jr

you can get the files from:

<https://github.com/econtrerasd/F256-spreditjr>

1. SPREDIT.BAS
2. SPR2BAS.BAS

While most of the program is in BASIC (SuperBASIC) a few Machine Language Routines are used for speed, like drawing the Sprite representation on the bitmap screen, mirroring sprites, moving sprites, etc.

SuperBASIC allows the program to compile its own Machine Language routines.

## DESCRIPTION

SPREDIT is a simple sprite editor written in BASIC for the F256 Jr. series of Computers created by Stefany Allaire, check her website for more information

<https://c256foenix.com/>

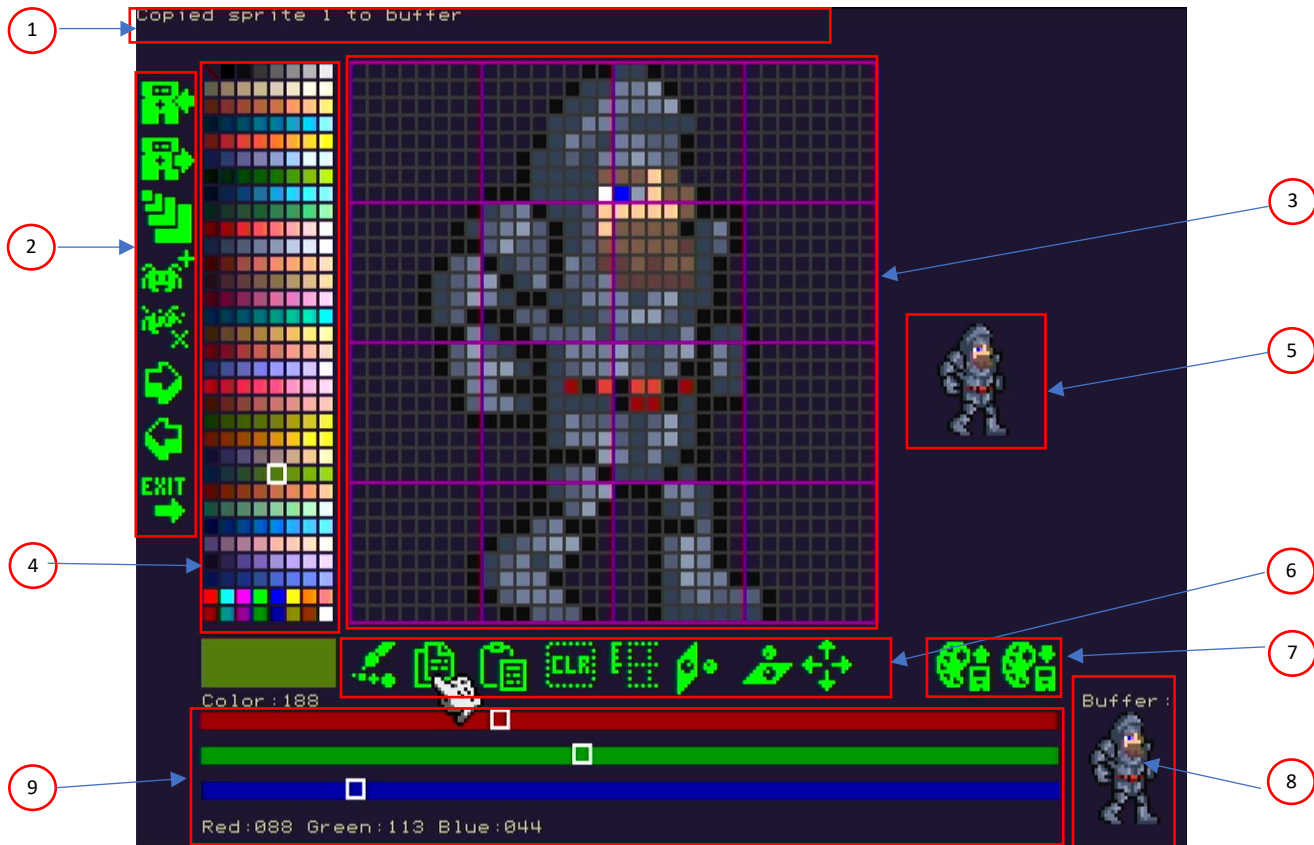
The Motivations to create this program were:

1. Provide a real example of using SuperBASIC and the inline assembler in the F256jr
2. Explore the Hardware capabilities of the F256Jr computer.

3. Create native tools in the F256 computer to remove dependency from external tools.

The Program is either mouse or joystick driven and operates on the 320x240 graphical screen mode; this resolution allows the users to see a reasonable sized representation of the current sprite you are working on.

The editor divides the screen in various sections:











- |                                   |                                  |
|-----------------------------------|----------------------------------|
| 1. Message Area                   | 6. Drawing Tools Icons           |
| 2. Main Menu Icons                | 7. Palette Save/Load Icons       |
| 3. Sprite Editor Grid             | 8. Sprite in Memory Buffer       |
| 4. Customizable 256 Color Palette | 9. Palette Color Editor Controls |
| 5. Actual Sprite                  |                                  |

The program operation will be discussed as we review each one of the sections of the screen.

## FUNCTION ICONS

The Function Icons allow you to handle the main operations needed to administer your Sprites:

Default Menu icons description

Icon	Function	ICON DESCRIPTION
	SAVE	5.25 Disc with incoming arrow
	LOAD	5.25 Disc with outgoing arrow
	SPRITE RESIZE	4 different sized squares
	ADD SPRITE	Space Invader with '+' sign
	DELETE SPRITE	Space Invader with 'x' sign
	MOVE TO NEXT SPRITE	-> arrow
	MOVE TO PREVIOUS SPRITE	<- arrow
	EXIT PROGRAM	Exit Sign



### SAVE

Clicking this ICON allows you to save all your sprites, a prompt appears at the top of the screen asking for a filename\*. Consider that a default extension (.SPR) is added to your file.

*The filename must be a name of up to 8 characters, extra characters will be truncated.*



**!** *If you type a filename already in use the program will overwrite the file without any warning!*



## LOAD

Clicking this ICON allows you to load all sprites from an existing sprite file (.SPR extension). A prompt is shown to request the file to load, type a filename (without the extension) and press ENTER.

If the file is located the file is loaded and the first sprite is shown



**!** *If the file name doesn't exist a "File doesn't exist" message is displayed. Press the Load icon again to retry loading with a different name*



## RESIZE SPRITE

Clicking this ICON will show a prompt to select the new size of the sprite, you can press the following keys to resize accordingly:

Key	Resulting sprite size
0	8x8 Sprite
1	16x16 Sprite
2	24x24 Sprite
3	32x32 Sprite

Please note that you can only resize the last sprite of the set (otherwise the program would need some complex memory management to move all sprites in memory! And rebuild the sprite index on the fly)



**!** *Resizing a sprite will clear the current image*



## ADD SPRITE

When you first enter the Sprite editor there is only one sprite to edit, if you press the MOVE TO NEXT SPRITE or MOVE TO PREVIOUS SPRITE a message will let you know that you are at either the *Last sprite* or *First sprite* already.

To Add an additional sprite, click the ADD SPRITE ICON, which will cause a new sprite to be **appended after the last sprite** and moving to the last sprite.

The new sprite will be created using the same size of the last sprite in the set.



## DELETE SPRITE

When you Click this Icon **the last sprite of the set is deleted**, and you are positioned at the new *Last Sprite*.

If you only have one sprite a message will remind you that you can't delete your *Last Sprite*.



## MOVE TO NEXT SPRITE

When you have many sprites, this ICON allows you to navigate to the next sprite, the grid automatically changes and reflects the sprite definition of your next sprite.

If you click continuously, the sprites change fast enough to do some rudimentary animation tests, -this effect is negated when you move between different sized sprites since the grid needs to be redrawn-





### **MOVE TO PREVIOUS SPRITE**

When you have many sprites, this ICON allows you to navigate to the previous sprite, the grid automatically changes and reflects the sprite definition of your next sprite.

If you click continuously, the sprites change fast enough to do some rudimentary animation tests.



### **EXIT**

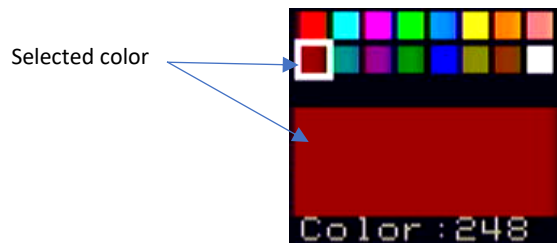
Click to exit the Sprite Editor, please remember to save your work before exiting the program, if you exit and enter the program again all sprites will be cleared as part of initializing the Editor.

## COLOR PALETTE

The color palette includes a pre-selection of color Gradients, this color palette is currently hardcoded in the Editor.

### Picking a Color from the Palette

Left Clicking on any color on the palette will change the brush color to the selected color, a rectangle will show the currently selected color on the palette, along with the current color index number for easy identification.



### RGB Value

Red:128 Green:000 Blue:000

Below the color mixing controls the red, green, and blue components of the selected color are shown, (just in case you are curious about this RGB values).

### Transparent Color

The top left color of the Grid is the transparent color, or better said the background color, this is denoted by an \ symbol over this color.

### Picking a Color directly from the GRID

When you are editing a Sprite and need to switch colors it's easy to forget what color you were using on another section of the image. -Yes, even with a limited palette of 256 colors things can get confusing!- and hunting down the right color on the palette consumes precious time.

To improve the color selection workflow, a color pickup function was added to the right mouse button, (or the spacebar in the keyboard), when you click it, you'll pick up the color under the mouse\*, this helps a lot in getting a previous selected color faster!

*\*Some validations are made to prevent you from picking the grid color, they are not perfect but reduce the times you pick the grid color*

## Editing New colors

A new section at the bottom of the screen shows the color components (RGB) of the current selected color with handles on the corresponding bar relating to the value of each component (RGB).

Moving the handles with the joystick/mouse modifies the current color value and change the color in real time.

As an additional help, the red, green, and blue components values change as you edit the colors.



## Loading / Saving Palette Colors

The Icons with a color palette and a disk Icon allow you to either load a palette from disk or save a palette to disk.

## Filesystem considerations

An extension (.PAL) will be appended automatically to the name you provided.

## Default Palette Color Credits

- The initial color palette was borrowed from the following page:

<https://lospec.com/palette-list/duel>

Although it has evolved since version 1, with a major revamp to make smother gradients in some cases and add true black, true white and 16 primary and secondary colors at the end.

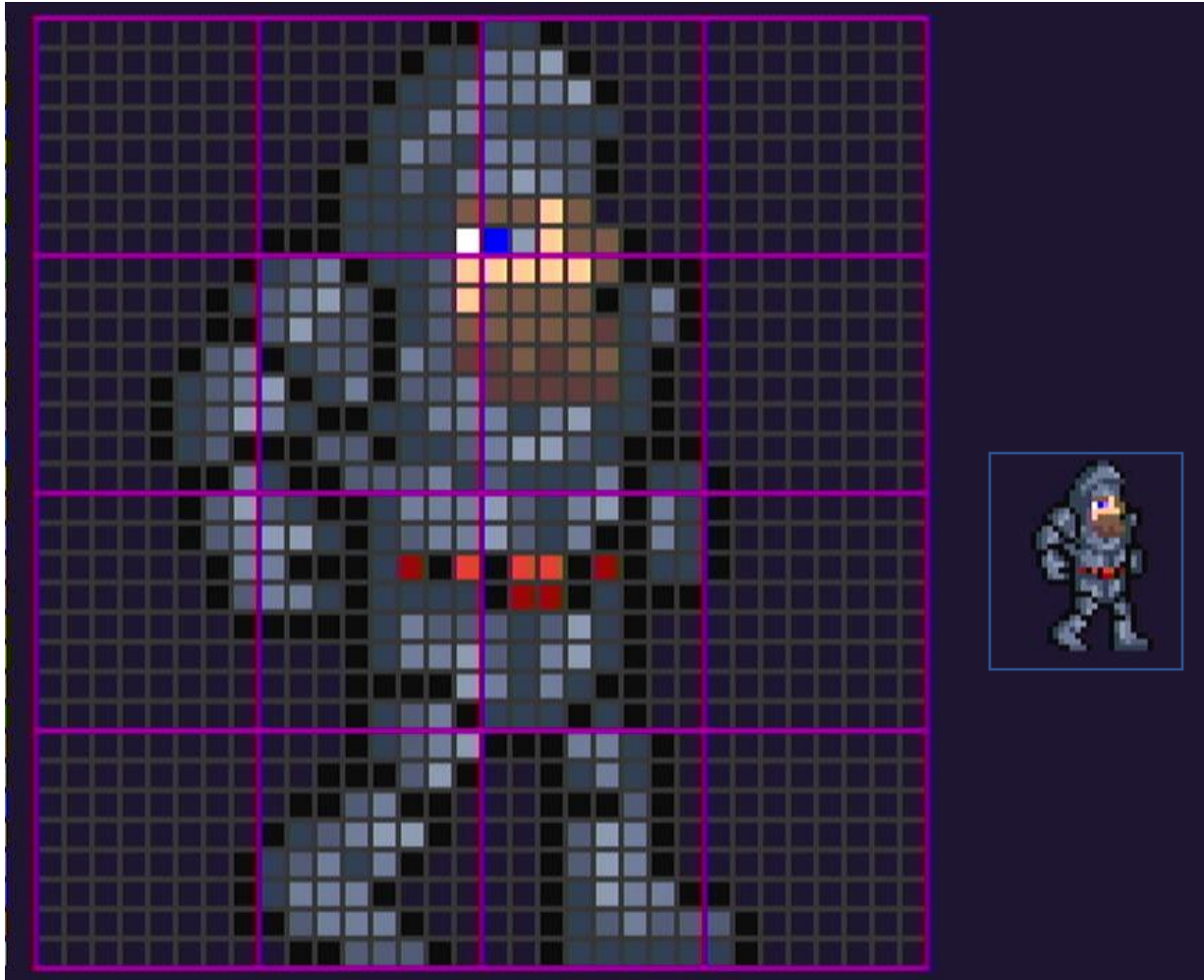
- Credits to the original creation of this palette are due to ARILYN@ARILYNART

<https://t.co/3Kxr6ZPzmu>

## SPRITE EDITOR GRID AND SPRITE REPRESENTATION









Once you have selected a color, just left click on any square on the grid to assign the current color to the pixel, you can hold the mouse button and drag the mouse around to draw faster.

As you Edit the sprite in the grid an actual sprite on the right side of the grid is updated in real time, so yes, no guessing how it will look!



## DRAWING TOOLS ICONS

The Drawing tools contains tools that help you in drawing your sprites (description of the icon in parenthesis):

	Tool	Icon Description
	BRUSH SIZE	A Brush with increasing point size
	COPY	Copy Icon
	PASTE	Board with document ready to be tagged
	CLEAR	Empty space with CLR letters
	GRIDLINES	Ruler with guidelines
	MIRROR X	Ball reflected in Standing Mirror
	MIRROR Y	Ball reflected in Flat Mirror
	SCROLL	Arrows Left/Right/Up/Down



### BRUSH SIZE

Using the BRUSH SIZE Tool will change how many pixels are drawn on the SPRITE EDITOR GRID.

The default brush size is 1 pixel, but available options are 2x2 / 3x3 / 4x4 pixels\*.

Every time you click in the BRUSH SIZE icon the BRUSH SIZE toggles to the next option and the size is shown by a digit (0-4) next to the Icon.

If you click enough times the option wraps around to go back to the default pixel size (1).

*\*Please note that when the BRUSH SIZE is 4x4 it lays down more pixels, but the responsiveness of the program is somewhat reduced*



**COPY**

When you are editing sprites, chances are that you are animating something and having the option to copy an already completed sprite into a new frame of animation is essential.

### **What did I just Copy?**

When you click the COPY Icon a representation of the current sprite is copied into the buffer memory, you might continue to edit your current sprite and choose to Paste the untouched copy latter.

To make it easier to remember what you copied into memory the sprite you copied appears in the bottom right corner of the screen under the "Buffer" section, so you can see what is in the buffer.



**PASTE**

When you click PASTE the current sprite in the buffer replaces whatever is in your current sprite.



*Currently you are only allowed to paste sprites into another sprite from the same size as the source image.*



## CLEAR

When you click this icon *-not surprisingly-* the current sprite is wiped clean, yes sometimes is better starting from scratch. (And yes... this was added since erasing a 32x32 sprite pixel by pixel takes a while)



## GRIDLINES

Clicking this ICON will show a color grid on your sprite editing area, the gridlines toggle it's spacing with each click between 4x4, 8x8, 16x16 and removing them completely.

This function is designed to help aligning or centering sprites, calculate height, or plan displacement in animation frames.

When you move to a different sized sprite the grid resets and disappears, you need to select it again manually.



## MIRROR X

Clicking this ICONS flips your sprite from left to right.



## MIRROR Y

Clicking this ICONS flips your sprite from top to bottom.



## SCROLL

Clicking this ICONS enables the scroll sprite function, moving the joystick will move the sprite image around the editing area, if you move past one of the sprite boundaries the sprite image will wrap around on the other side. This functionality is useful to align and recenter sprites.

## SPRITE FILE STRUCTURE in SPREDITjr

The Save file structure is very simple:

1. First byte - number of sprites in the file
2. 2048 bytes of Index data for all (potential) sprites in the set. Each index entry uses 3 bytes organized as follows:

Byte #	Description	bits (7-4)	bits (3-2)	bits(1-0)
Low Byte	Lowest byte of the sprite data address and sprite size	Address - you should and this byte against 240 to get the right address	unused	sprite size 0-8x8 1-16x16 2-24x24 3-32x32
Mid Byte	Mid byte of sprite data address			
High Byte	High byte of sprite data address			

3. *Sprite Data for all sprites in the file*



## CONVERTING A SPRITE TO SUPERBASIC STRUCTURE

While the Sprite editor uses 2kb in memory for indexing sprites and its properties SUPERBASIC uses a proprietary structure to keep track of the sprites in memory. The format is not directly compatible, but the following BASIC program can be used to convert a saved file to SUPERBASIC format:

```

5      sprites on :cls
10     input "sprite file to convert to BASIC format:";a$
15     bload a$+".spr", $FFFF
20     print "Loading "+a$+".spr sprites in $FFFF"
30     xpeekval=alloc(3)
40     xpeek($FFFF):n=peek(xpeekval)
50     dest=$30000:index=$10000
60     for c=0 to n
65     meta=dest
70     if c=0 then xpoke(dest,$11):dest=dest+1
80     xpeekl(index):sz=peekl(xpeekval)&03: size=sz*8+8:
src=peekl(xpeekval)-sz
85     src=src-$20000:rem "adjust address in index"
90     xpoke(dest,sz):dest=dest+1
100    xpoke(dest,0):dest=dest+1:orig=dest
110    memcpy src,size*size to dest:dest=dest+(size*size)
120    print ".";
130    index=index+3
140    next
145    xpoke(dest,$80)
150    sprite 0 to 100,100 image 0
160    print :print str$(n+1)+" Sprites converted"
170    print "saving "+a$+".bspr in BASIC sprite format"
190    bsave a$+".bspr", $30000, dest-$30000
200    end
1000   proc xpoke(address,value)
1010   memcpy address,1 poke value
1020   endproc
1030   proc xpeekl(address)
1040   memcpy address,3 to xpeekval
1050   endproc
1060   proc xpeek(address)
1065   pokel xpeekval,0
1070   memcpy address,1 to xpeekval
1080   endproc

```

*This program resides in the [gitHub](#) as `spr2basic.bas`*

## USING SPRITES IN SUPERBASIC

Once you have used the previous program on a sprite file you end up with a different file with the same name but with a **.bspr** extension.

This file can be loaded directly from SUPERBASIC using BLOAD into the default memory for sprites (\$30000).

Note: make sure you initialize the sprites before loading the file (sprites on), since initialization clears any previous sprite definitions in memory

The following example code loads a converted sprite file in memory and displays the first sprite of the file:

```
10 CLS: SPRITES ON
20 BLOAD "SPRITES.BSPR", $30000
30 sprite 0 to 100,100 Image 0
```

## LIMITATIONS, KNOWN ISSUES

- The sprite file can only store 255 sprites (not that I have tried to define that many sprites... yet)

## CONTACT INFORMATION

Hopefully you'll enjoy using this program as much as I enjoyed creating it, if you have any comment or suggestion feel free to contact me: Ernesto Contreras

Email	eacontrerasd@gmail.com
Discord Server	<a href="https://discord.com/channels/691915291721990194/1008139105386889346">https://discord.com/channels/691915291721990194/1008139105386889346</a>