# Sprite Maker

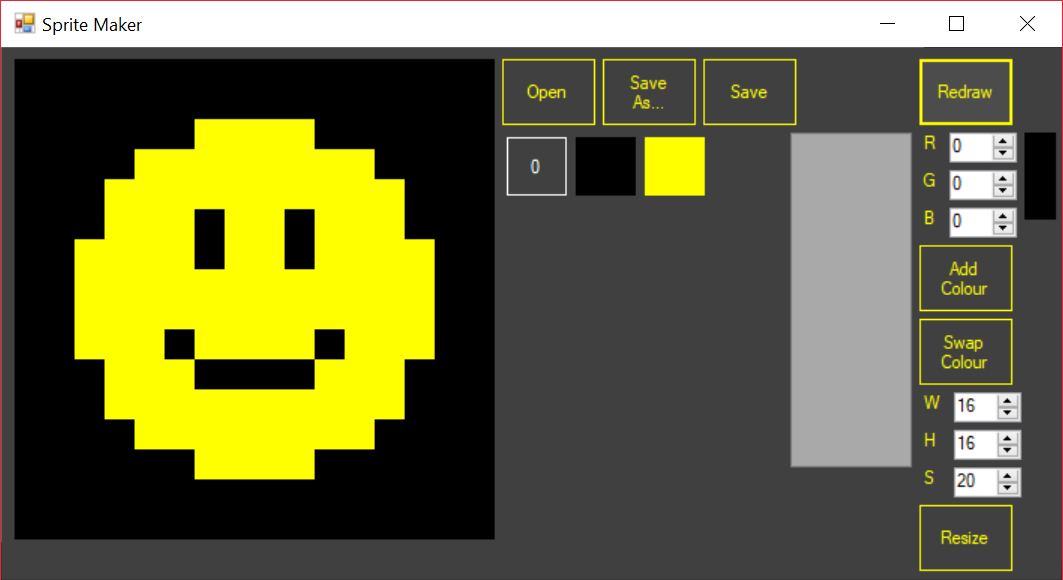
## Files (.sprt)

If the Platformer Engine is to use sprites, then I need to have a way to make them. I want to encode the sprites in a text file like the following:

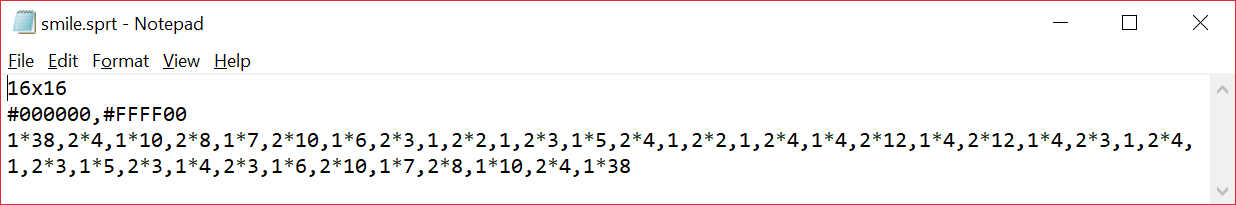
[Width]x[Height] (e.g. 16x16)

Colour Hex Codes (e.g. #ff00e5)

Colour indices for each pixel



1.1 Example Sprite

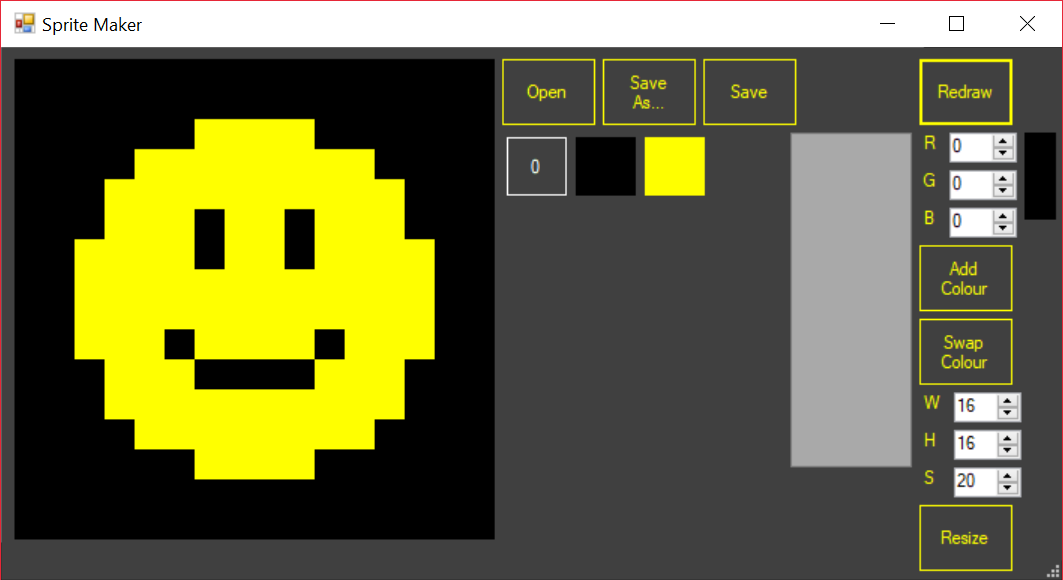


1.2 .sprt file for above sprite

Each colour index refers to a colour, either 0 for transparent or above that for colours defined in the second line of the file. In this example index 1 refers to #000000 (black) and index 2 refers to #FFFF00 (yellow).

After some indices there is a multiplication sign (\*) and another number. This is a form of compression I tried to use to reduce file size, by making repeats of the same colour be abbreviated. So instead of the same number repeating (e.g. 2,2,2,2) it displays how many times it gets repeated (e.g. 2\*4).

## User Interface



4

5

1

2

3

1.3 Layout of sprite maker

The interface of the sprite maker is split into a few groups.

1. Draw Area
2. Save/Load
3. Colour Palette
4. New Colour Maker
5. Resizer

### Draw Area

This shows the sprite currently being worked on. The user draws on it by selecting a colour with the colour palette and then clicking or dragging on the draw area.

Because the draw area is rendered using the VB.NET System.Drawing namespace the draw area may suddenly not show the sprite any more. I do not know how to prevent this, but it can be caused by a few things, for example dragging part of the form offscreen. To remedy this, I added a button labelled “Redraw”. If the drawing is cleared the user can click this button to have the program render the sprite again.

### Save/Load

This area has the buttons for saving and loading sprites. The “Open” button asks the user to select a file, which is then loaded.

The “Save As…” button asks the user to choose a folder and filename for the sprite to be saved.

The “Save” button is like the previous two buttons but will save to the location where the file is already loaded from/saved to.

### Colour Palette

The colour palette is used by the user to select which colour they would like to use. It begins with only transparent (index 0) as the only option available, but new colour can be added using the New Colour Maker section of the form.

Each colour on the palette is represented by a button of the same colour. When the user clicks a colour, the index of the colour being used is updated so when the user next draws the selected colour will be used.

Each time a colour is added a new button must be added, so I’ve had to put an artificial limit on how many colours can be on the palette (currently 20 max).

### New Colour Maker

This area gives the user to use any colour they want in the sprite. The user can create a colour using red (R), green (G) and blue (B) colour channels. These values can vary from 0 to 255 and are inputted by the user in the numerical up/down boxes next to the R, G and B labels.

Next to the RGB values is a panel, currently black, which displays a preview of the colour.

The “Add Colour” button will add the current colour to the colour palette, if the exact colour doesn’t already appear in the palette.

The “Swap Colour” button will swap the current colour with the currently selected colour on the colour palette.

### Resizer

The controls here are used to resize the draw area. W represents Width, H is for Height and S means Scale. The scale is how many pixels wide and tall each square on the draw area is. Values are entered in the same way as the RGB values for new colours are.

When the user presses the “Resize” button a warning is displayed, if the user proceeds then the current work is lost and the grid is resized.

## Issues

1. No file validation
2. Limited number of usable colours
3. Can’t resize sprite and keep any of the previously loaded sprite, must reset
4. When using load sprite user may draw a square because the mouse is down over the draw area when the load dialog closes

# Level/Game Creator

# Panel Render Engine 2

# Game Executor