Step 1: Visit => https://javl.github.io/image2cpp/
Select the image for which you want the byte format to use in Arduino code.

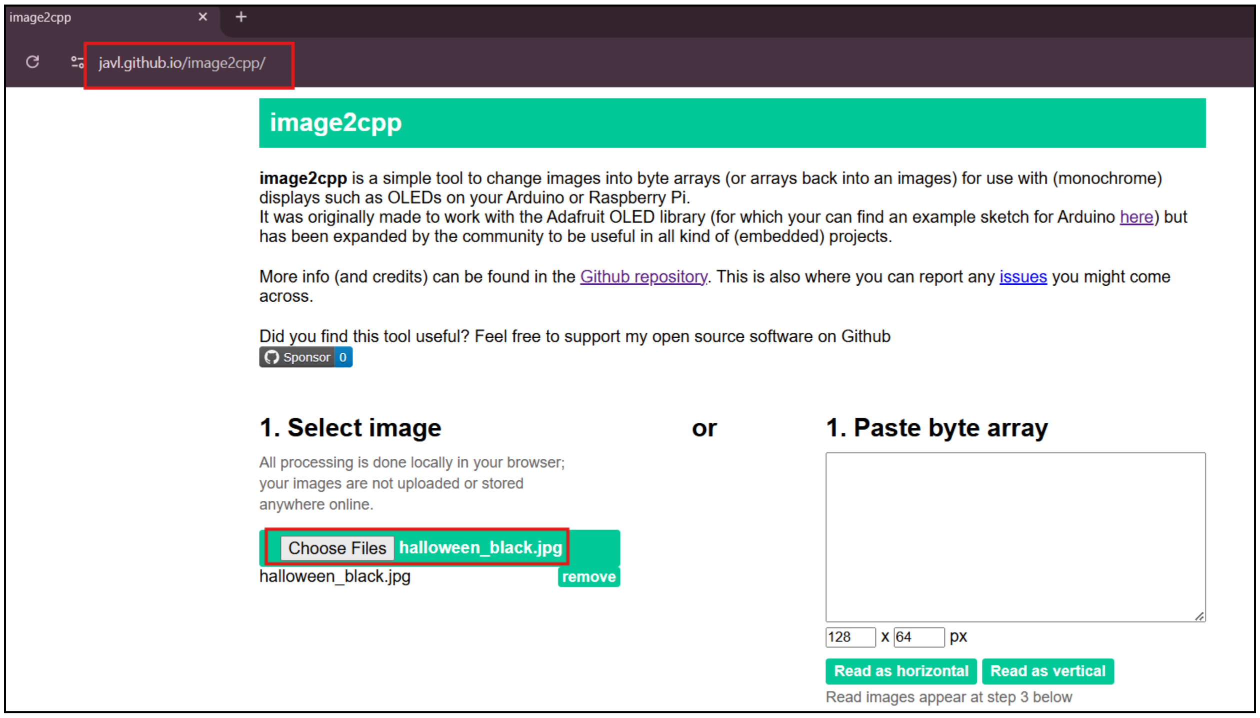


image2cpp

javl.github.io/image2cpp/

image2cpp

image2cpp is a simple tool to change images into byte arrays (or arrays back into an images) for use with (monochrome) displays such as OLEDs on your Arduino or Raspberry Pi.
It was originally made to work with the Adafruit OLED library (for which your can find an example sketch for Arduino here) but has been expanded by the community to be useful in all kind of (embedded) projects.

More info (and credits) can be found in the Github repository. This is also where you can report any issues you might come across.

Did you find this tool useful? Feel free to support my open source software on Github

Sponsor 0

1. Select image or **1. Paste byte array**

All processing is done locally in your browser; your images are not uploaded or stored anywhere online.

Choose Files halloween_black.jpg

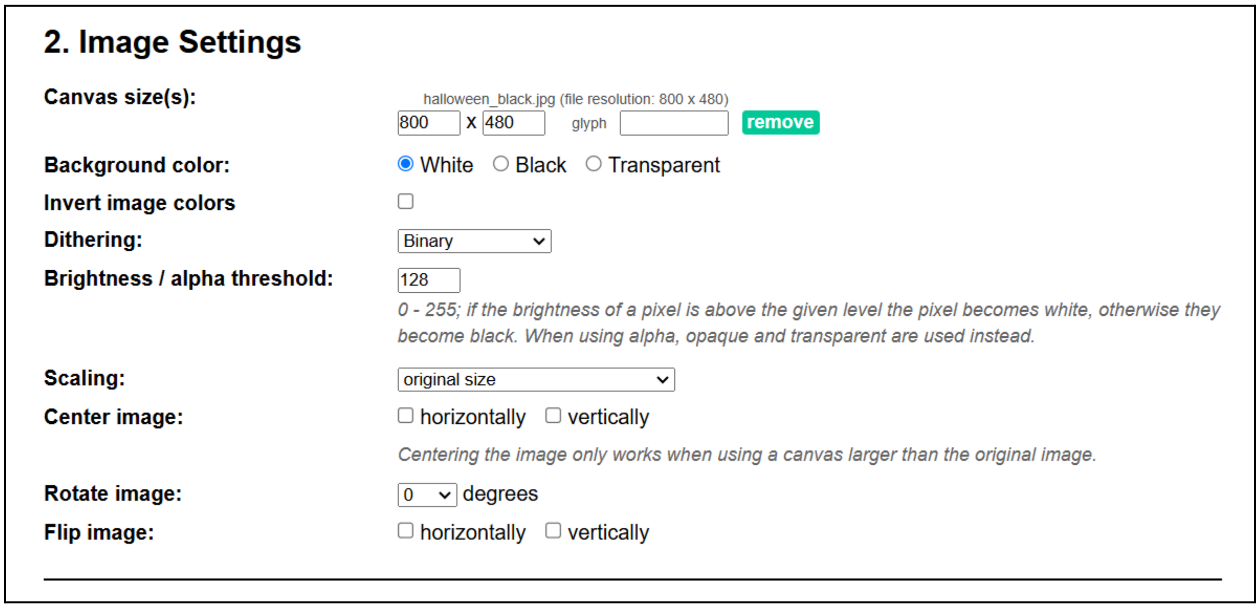
halloween_black.jpg remove

128 x 64 px

Read as horizontal | Read as vertical

Read images appear at step 3 below

Step 2: Perform image settings as per your display size
For example, suppose EnkFi 7.5 is used and its resolution is 800 x 480,
You can adjust other parameters too for better visuals, to verify proper setting checkout preview.



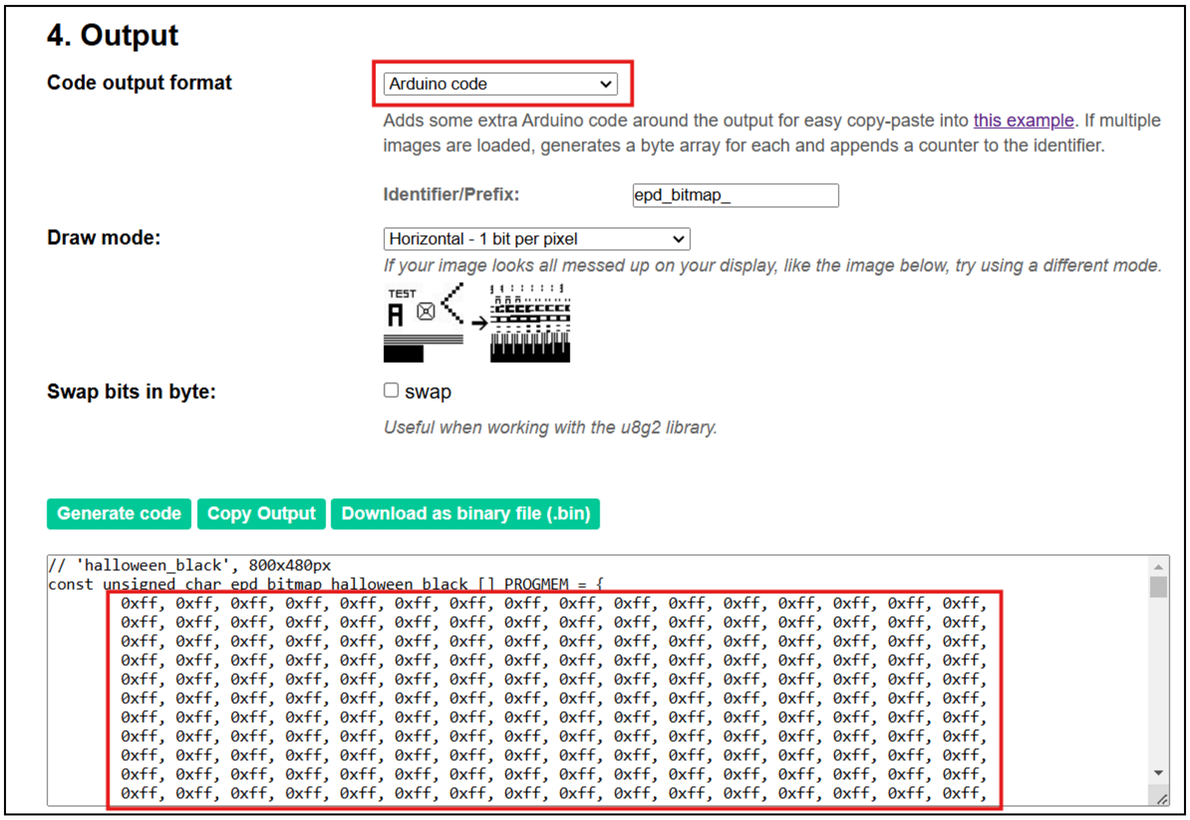
2. Image Settings

Canvas size(s):	halloween_black.jpg (file resolution: 800 x 480) 800 x 480 glyph remove
Background color:	[✓ White] Black Transparent
Invert image colors	☐
Dithering:	Binary
Brightness / alpha threshold:	128 *0 - 255; if the brightness of a pixel is above the given level the pixel becomes white, otherwise they become black. When using alpha, opaque and transparent are used instead.*
Scaling:	original size
Center image:	☐ horizontally ☐ vertically *Centering the image only works when using a canvas larger than the original image.*
Rotate image:	0 degrees
Flip image:	☐ horizontally ☐ vertically

3. Preview

HALLOWEEN
OFFERS
COMING SOON

Step 3: Generate byte code using either Arduino option or plain byte, copy complete bytes into image array.



4. Output

Code output format Arduino code

Adds some extra Arduino code around the output for easy copy-paste into this example. If multiple images are loaded, generates a byte array for each and appends a counter to the identifier.

Identifier/Prefix: epd_bitmap_

Draw mode: Horizontal - 1 bit per pixel

If your image looks all messed up on your display, like the image below, try using a different mode.

Swap bits in byte: ☐ swap

Useful when working with the u8g2 library.

```
// 'halloween_black', 800x480px
const unsigned char epd_bitmap_halloween_black [] PROGMEM = {
	0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,
	0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,
	0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,
	0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,
	0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,
	0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,
	0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,
	0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,
	0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,
	0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,
	0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,
```