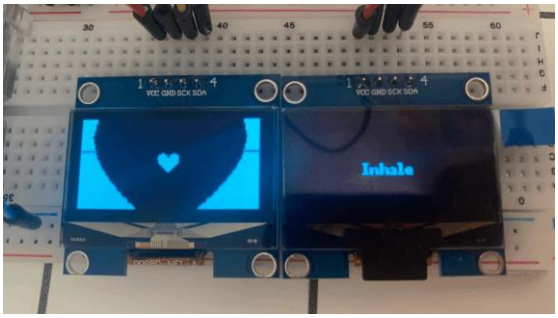
Overview of the Arduino Breath Visualiser

The goal of making this simple breath visualiser is to help you stay (or return to being) grounded, calm and present in times of panic, anxiety or stress.

Admittedly, it is probably easier to find and use one of these online, but I figured I'd challenge myself to make something physical and learn a bit about embedded systems while I'm at it.



Maybe this can help someone 😊

Equipment I used

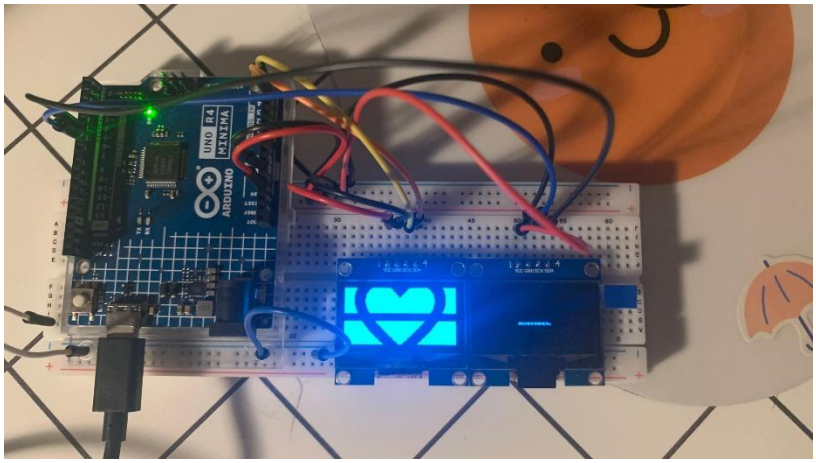
- Arduino Uno R4 Minima
- Jumper wires (male-male)
- x2 SH1106 OLED screens
- Breadboard
- heart GIF

Process

1. **<u>Pin wiring for 2 OLED screens</u>**

OLED #1	VCC	3.3V
	GND	GND
	SCL/SCK (clock)	A5
	SDA	A4

OLED #2	VCC	3.3V
	GND	GND
	SCL/SCK (clock)	Digital 6
	SDA	Digital 5



2. **Optimise GIF**

Originally, the expanding and contracting heart GIF was 96x96 pixels.
I cropped it (using this site: https://ezgif.com/) to 128x64 so it would be able to fit the size of my OLED screen. Although the full image couldn't be in it, it proved to be an easier approach when you convert each frame to byte arrays since everything needs to be very specific and to scale, if not, you'll end up with some "corrupted looking" pixels in the animation.

Then, I split the frames of the GIF (also using https://ezgif.com/). I got around ~17 frames for the entire animation. Save it as PNG.

(Note: when it comes to the software, you can get away with just downloading and converting the first ~6-ish frames or half of your total frames and then have the loop run the frames backwards for the contraction of the heart.)

Using this site: https://javl.github.io/image2cpp/ you can convert PNGs to byte arrays and vice versa. I converted most of my frames of the animation into byte arrays.
Depending on how the images show up on the OLED, you may have to select the "flip image: horizontal" option in image settings on the site.
For the output, you can select "Arduino code" or "plain bytes".
The "draw mode" should be set to "Horizontal – 1 bit per pixel".

Reference the Arduino sketch and copy in your byte arrays for each frame and place them in the “heart_bits[][]” array.

3. **Coding the software**
   I installed the u8g2 library since it’s the one that works for my SH1106 OLED screens. There should be exactly 1024 bytes in each byte array since 128×64=8192/8=1024. If you have less, e.g 1008 bytes when you copied the byte arrays from the website, you can add in 16 (or however many you need) “0x00”’s to the end of each array to get it to 1024. I have also declared in the heart_bits[] array to have [1040] bytes since I got an error when I declared it as [1024]…but it works! So I won’t be touching it at the moment… 🥶

   The drawAnimation() function and heart GIF was taken from https://github.com/tigrisli/oled-animation/.

4. Compile (and pray that you get no errors…) and then upload onto the Arduino !

Thanks for reading 💗 - A.S

// note to self: write out line by line code explanation for textAnimation(void);