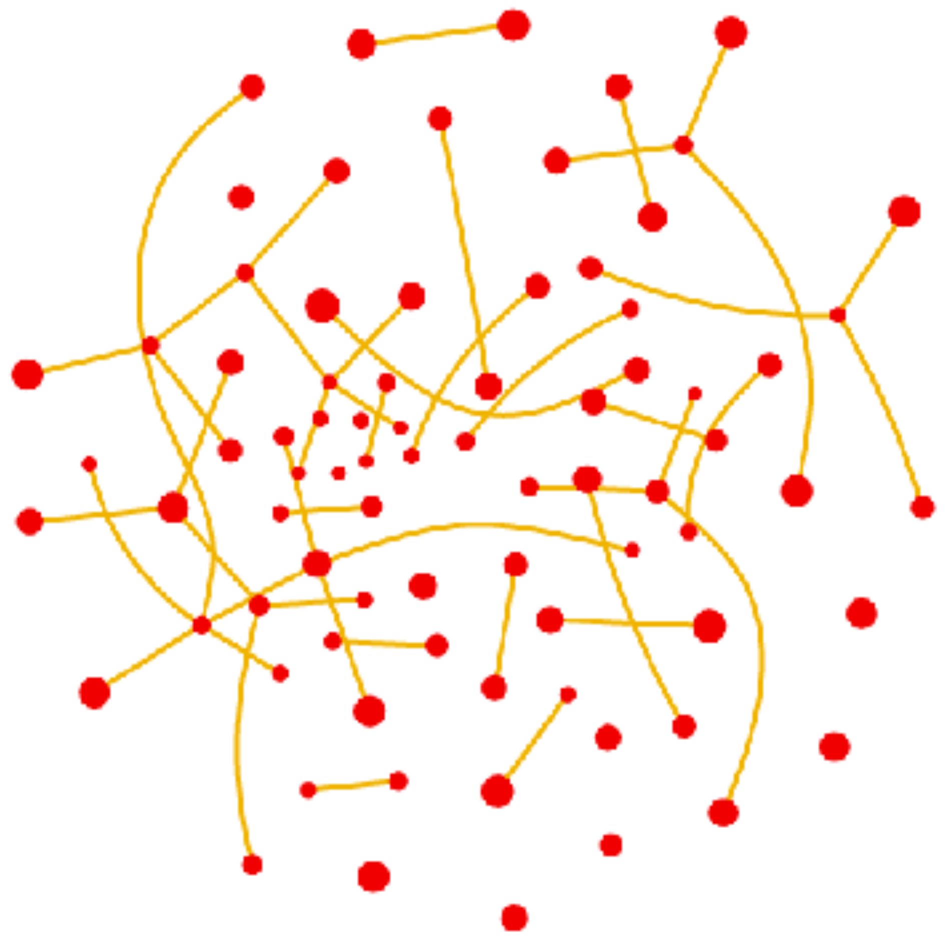


OLED

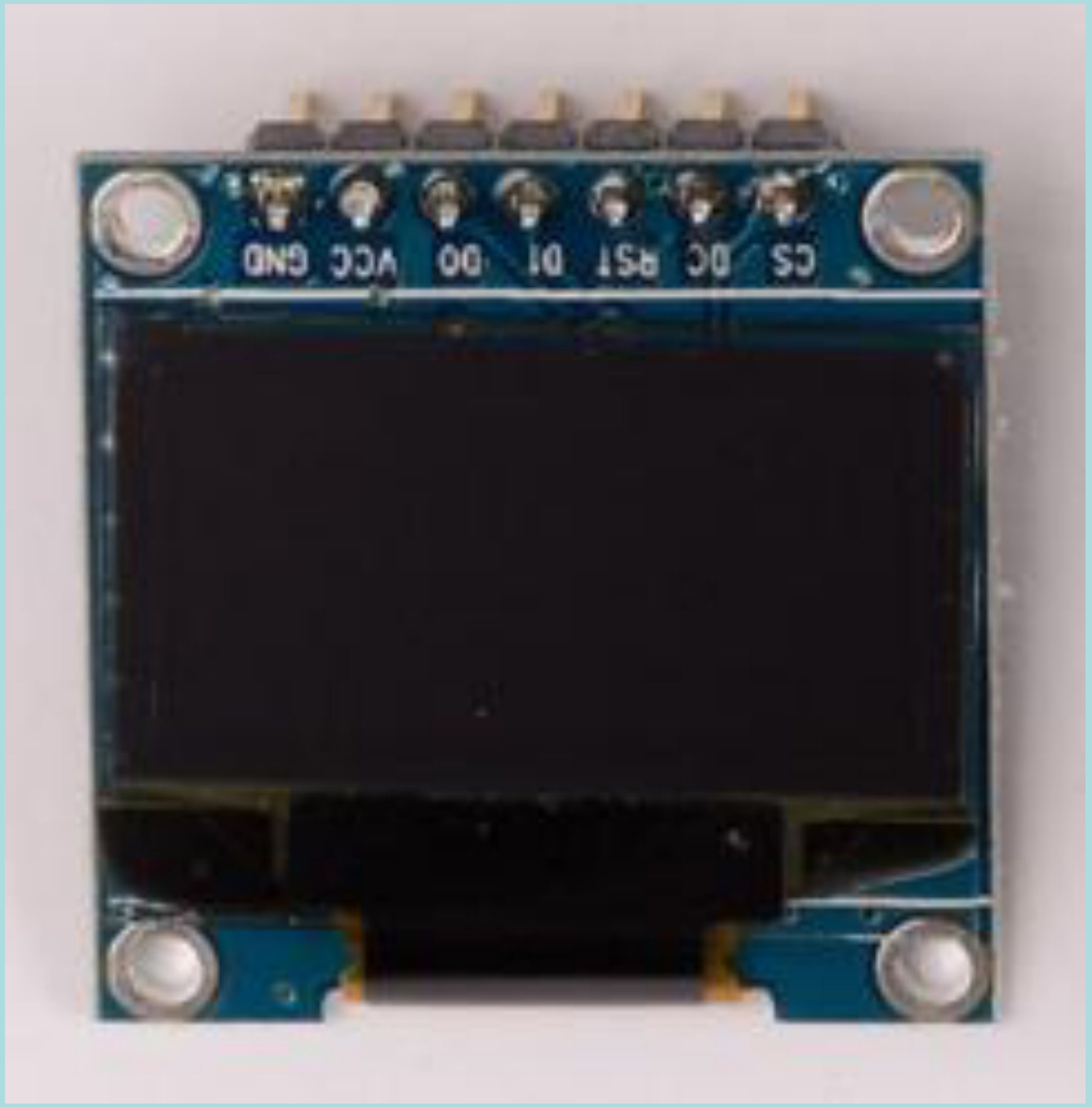
AN OLED

0.96" serial Oled Screen

128x64 pixels

Control: SPI (default) or I2C

Monochrome



AN OLED

Richer output

<u>BUT</u>

They’re much more involved.

& they take up a

bunch of pins

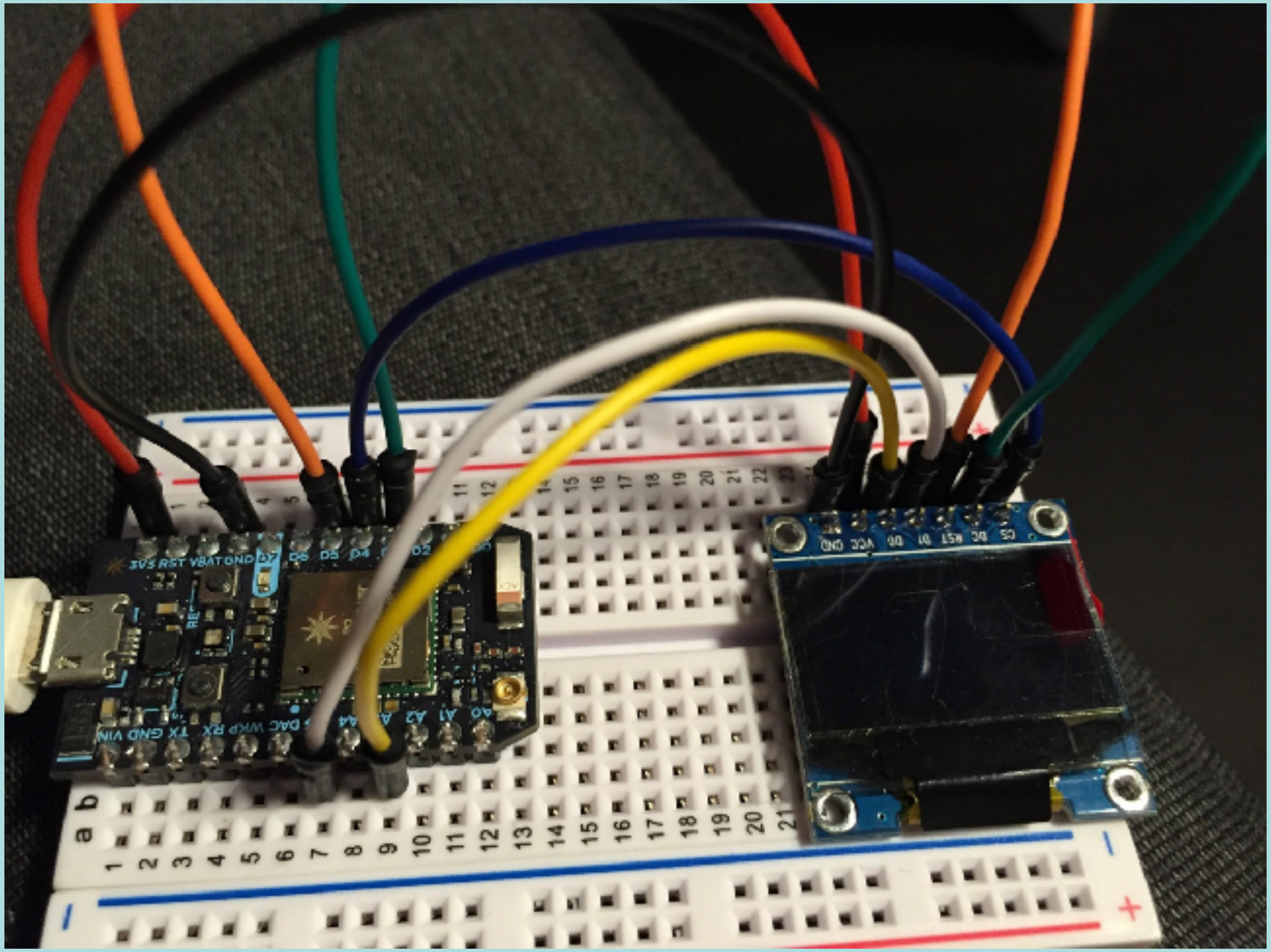
Libraries

Adafruit_SSD1306

Adafruit_GFX

```
#include "Adafruit_SSD1306/Adafruit_GFX.h"
#include "Adafruit_SSD1306/Adafruit_SSD1306.h"
```

Wiring



3V3 --> VCC (Red)

GND --> GND (Black)

D5 --> RST (Orange)

D4 --> CS (Blue)

D3 --> DC (Green)

A5 --> D1 (White)

A3 --> D0 (Yellow)

Library and Commands

Controlling the screen

clears the screen and buffer

```
display.clearDisplay();
```

tell the screen to refresh with new changes

```
display.display();
```

invert the displays
normal display

```
display.invertDisplay(true)
display.invertDisplay(false
```

display is dimmed/backlight
normal display

```
display.dim(true);
display.dim(false);
```

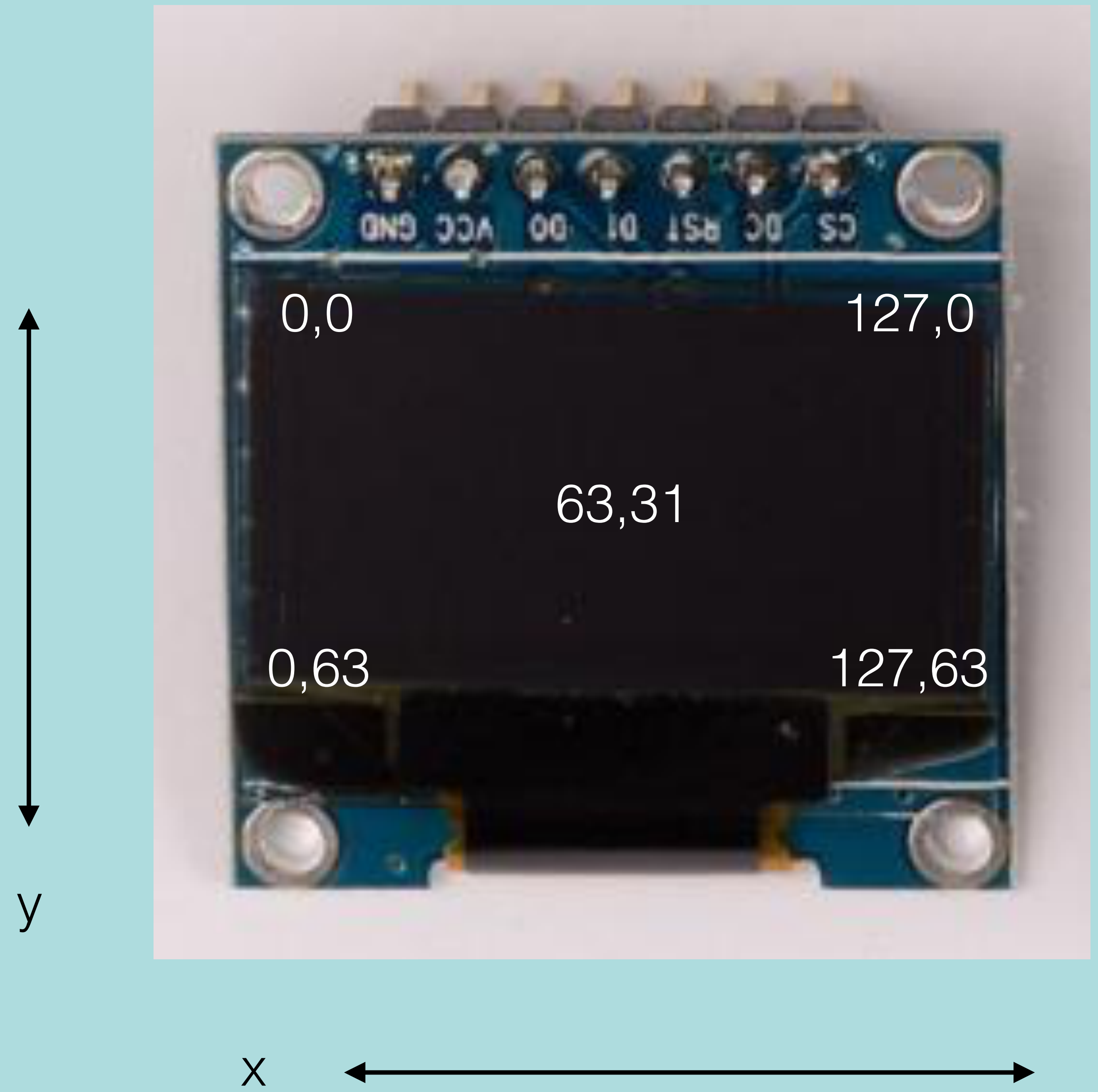
Library and Commands

Add a constructor

```
#include "Adafruit_GFX.h"
#include “Adafruit_SSD1306.h"
// use hardware SPI
#define OLED_DC     D3
#define OLED_CS     D4
#define OLED_RESET  D5

Adafruit_SSD1306
  display(OLED_DC,
      OLED_RESET,
      OLED_CS);

void setup()
{

 display.begin(
    SSD1306_SWITCHCAPVCC);

}
```

Library and Commands

128x64 pixels



Library and Commands

Drawing on screen

two color options

```
WHITE  # maps to 0
BLACK  # maps to 1
```

fill the screen

```
display.fillScreen(WHITE);
```

Library and Commands

Drawing on screen

change a pixel on screen

```
display.drawPixel(
          x, y, color);
          // 1,2,WHITE
```

draw a line

```
display.drawLine(
  x1, y1, x2, y2, color)
```

draw a bordered rectangle

```
display.drawRect(
  x1, y1, x2, y2, color)
```

draw a filled rectangle

```
display.fillRect(
  x1, y1, x2, y2, color)
```

Library and Commands

Drawing on screen

Circles

```
d.drawCircle(x, y, r, c);
d.fillCircle(x, y, r, c);
```

Triangles

```
d.drawTriangle(x0, y0,
 x1, y1, z2, y2, c);
d.fillTriangle(x0, y0,
 x1, y1, z2, y2, c);
```

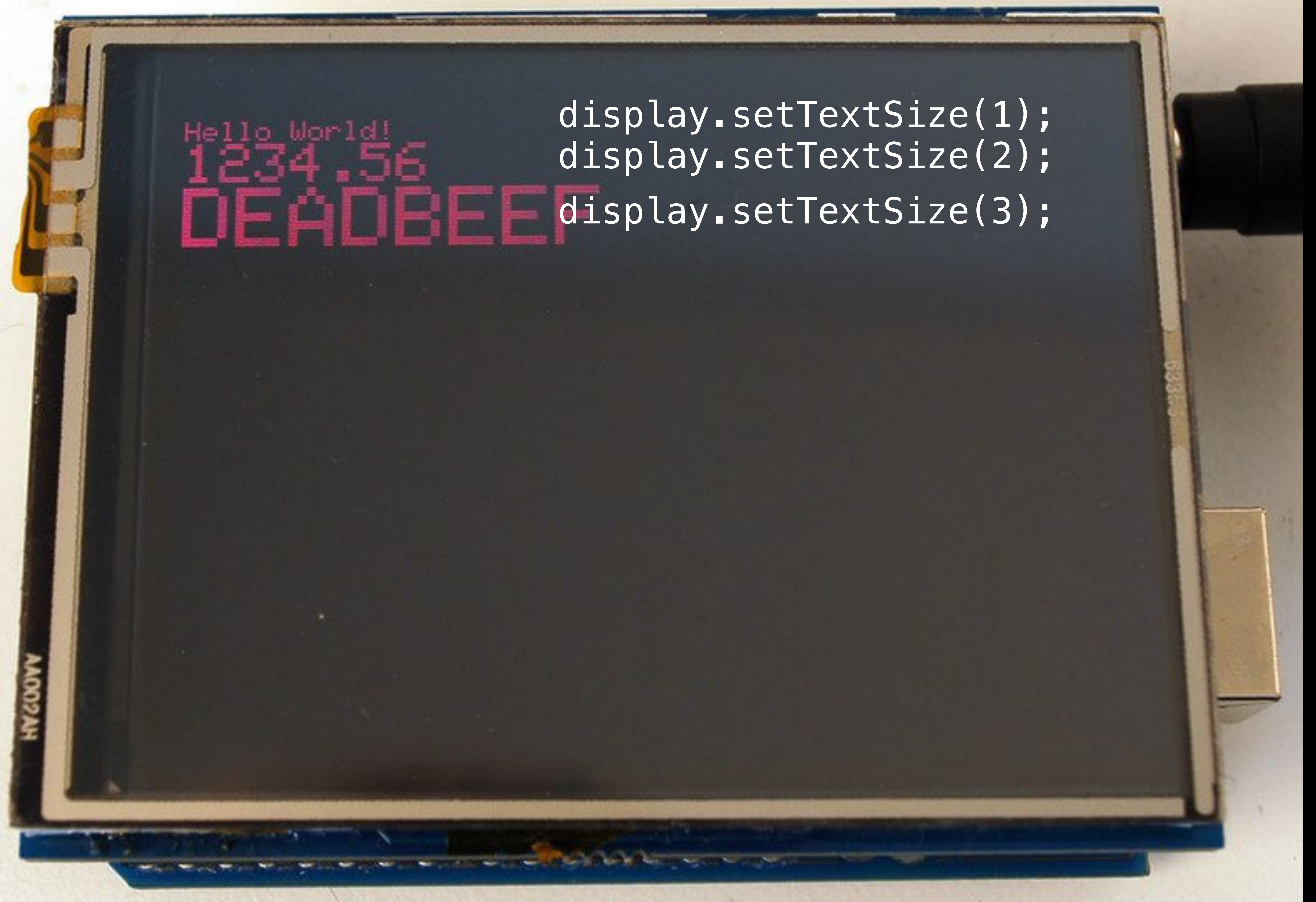
Round rects

```
d.drawRoundRect(x,y,w,h,
 r,c);
d.fillRoundRect(x,y,w,h,
 r,c);
```

Library and Commands

Writing out Text

Set the cursor	`display.setCursor(x, y);`
Set the color	`display.setTextColor(c);`
Set the Size of the font	`display.setTextSize(1);`
Should it wrap to a new line	`display.setTextWrap(true);`
Write text	`display.print(“Hi ”);` `display.println(“there”);`



```
Hello World!
1234.56
DEADBEEF
```

```
display.setTextSize(1);
display.setTextSize(2);
display.setTextSize(3);
```

Library and Commands

Writing out Text

Set the cursor	`display.setCursor(x, y);`
Set the color	`display.setTextColor(c);`
Set the Size of the font	`display.setTextSize(1);`
Should it wrap to a new line	`display.setTextWrap(true);`

Write text	`display.print(“Hi ”);`
	`display.println(“there”);`

tell the screen to refresh with new changes	`display.display();`

Pretty as a Picture

Displaying Bitmaps

Create a monochrome (single color) bitmap in Photoshop and make sure the width and height matches the size and settings for your OLED screen (i.e. 128 x 64)

Save the file as BMP in Windows format with 1 bit depth

If you're on Windows use LCD Assistant. If you're on OSX download bitmapToC (see the releases tab). Then import the Bitmap file and get a lovely character array.

Copy and paste the char array into your main code file.

Use display.drawBitmap to add your bitmap (see code example)