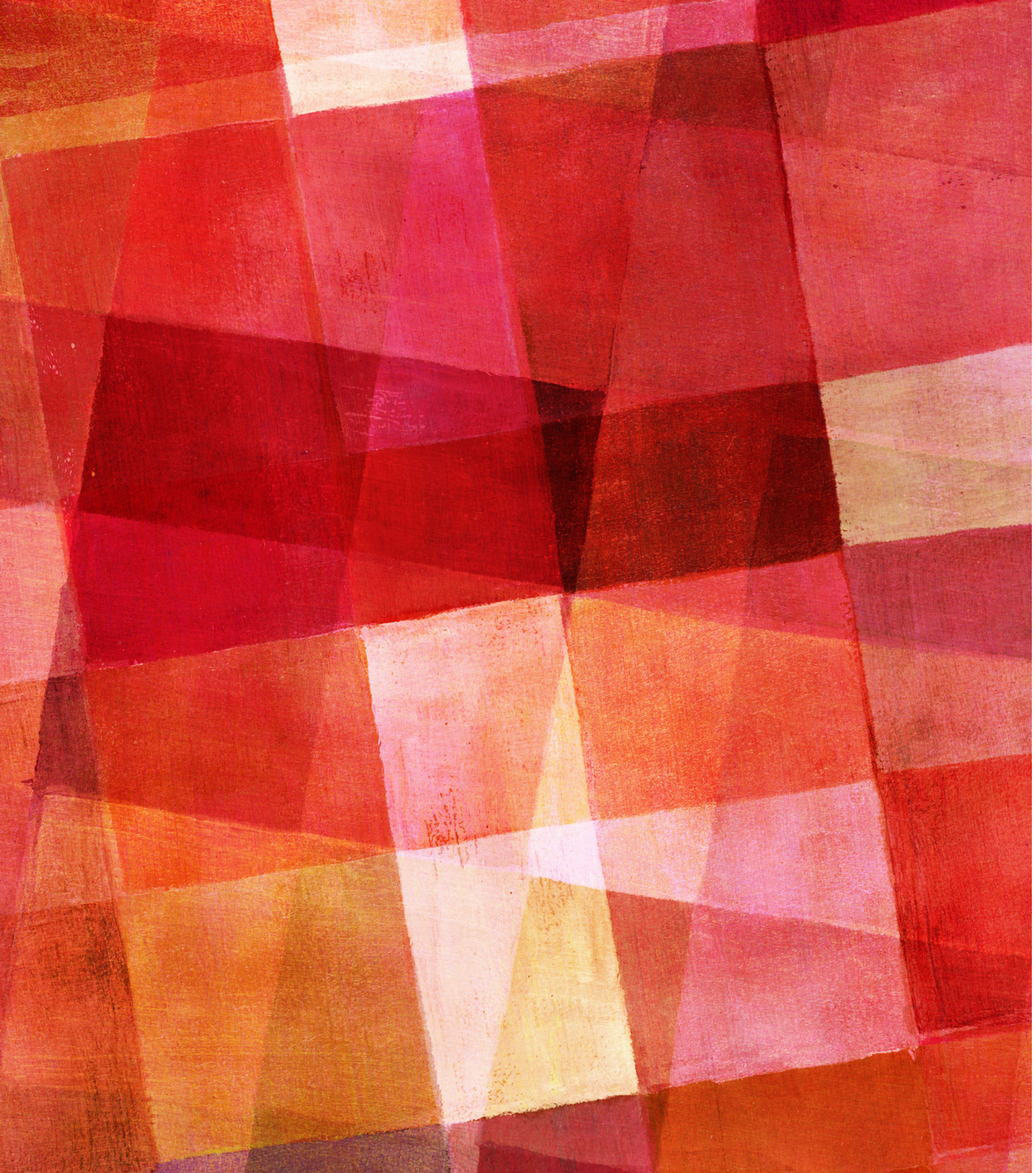
RUNNING PYTHON ON HARDWARE WITH CIRCUITPYTHON

Tammy Cravit

itsme@tammymakesthings.com
*github.com/**TammyMakesThings***
*twitch.tv/**TammyMakesThings***

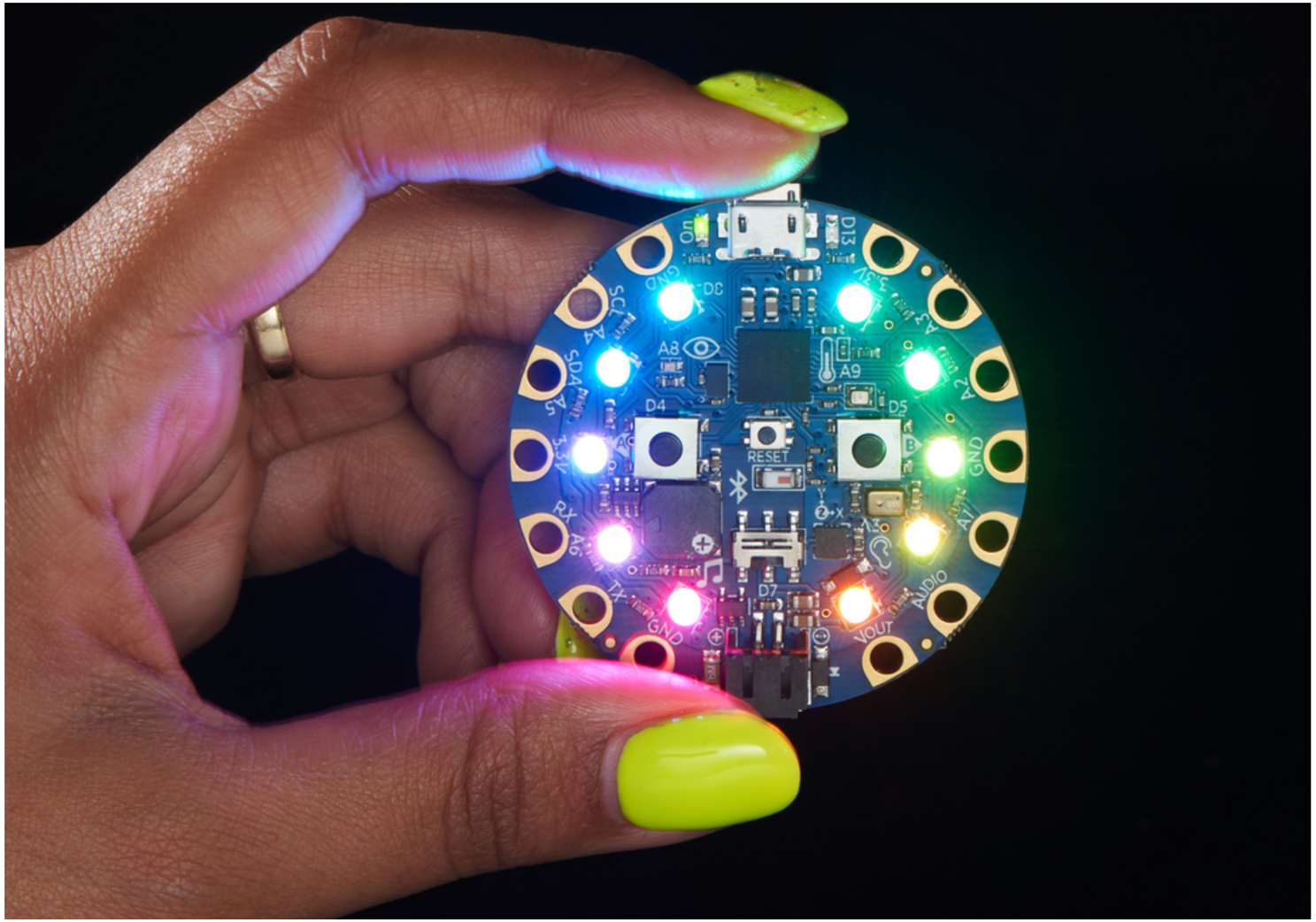


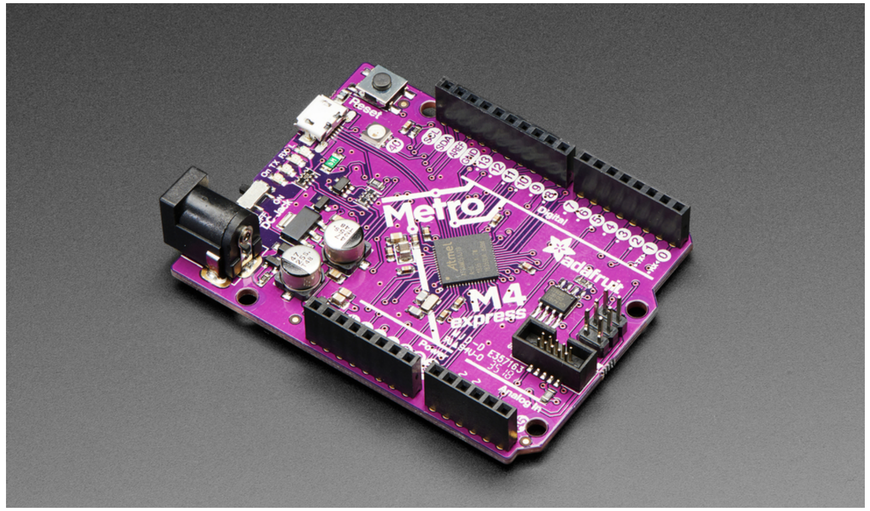
WHAT ARE MICROCONTROLLERS?

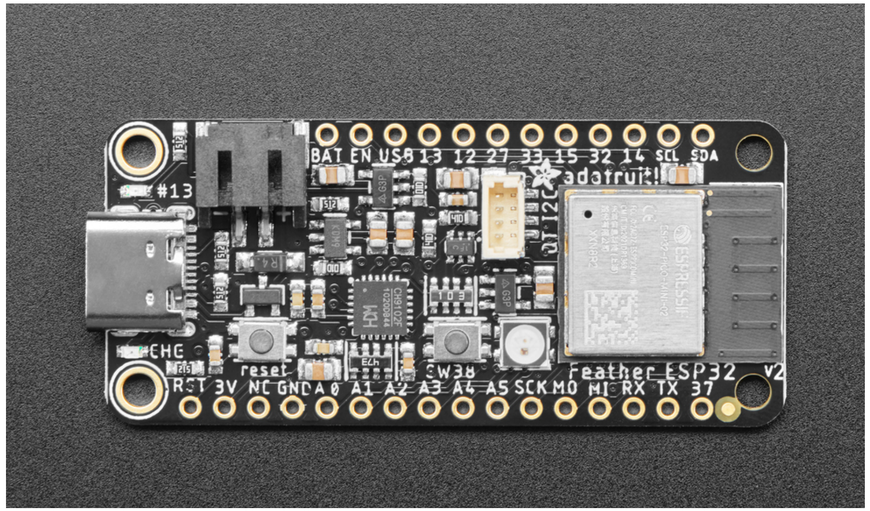
- Wikipedia says:

”A microcontroller is a small computer on a single…integrated circuit. Microcontrollers ate designed for embedded applications, in contrast to the microprocessors used in personal computers consisting of various… chips.”

- Microcontrollers typically include a variety of digital and analog I/O pins for interfacing with external hardware.

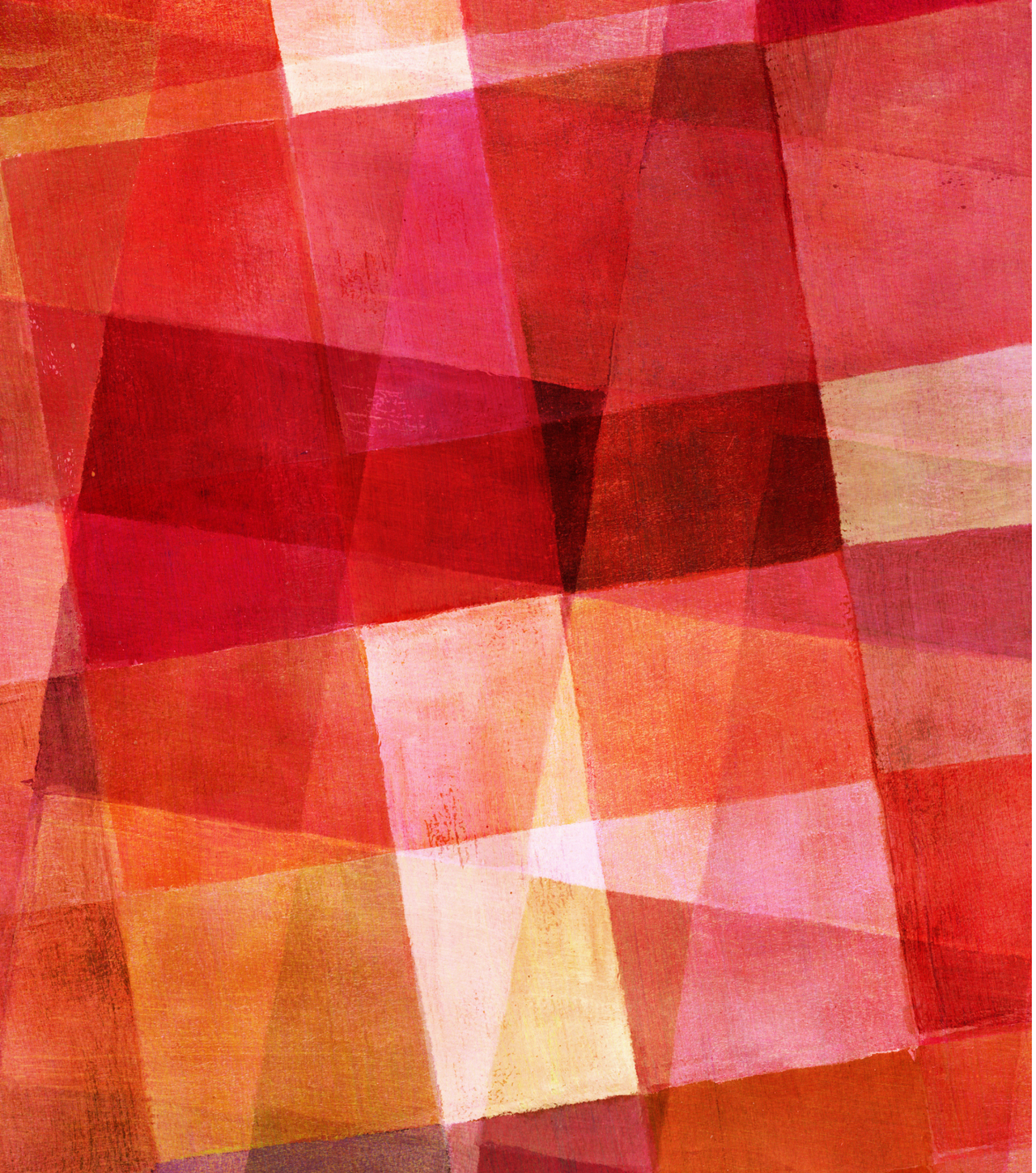

RESET
D4
D5
D7
A8
A9
A1
A2
A3
A4
A5
A6
GND
3.3V
SCL
SDA
RX
TX
AUDIO
VOUT


Metro
M4
express
adafruit


BAT EN USB 13 12 27 33 15 32 14 SCL SDA
adafruit!
#13
reset
Feather ESP32 V2
RST 3V NC GND A0 A1 A2 A3 A4 A5 SCK MO MI RX TX 37

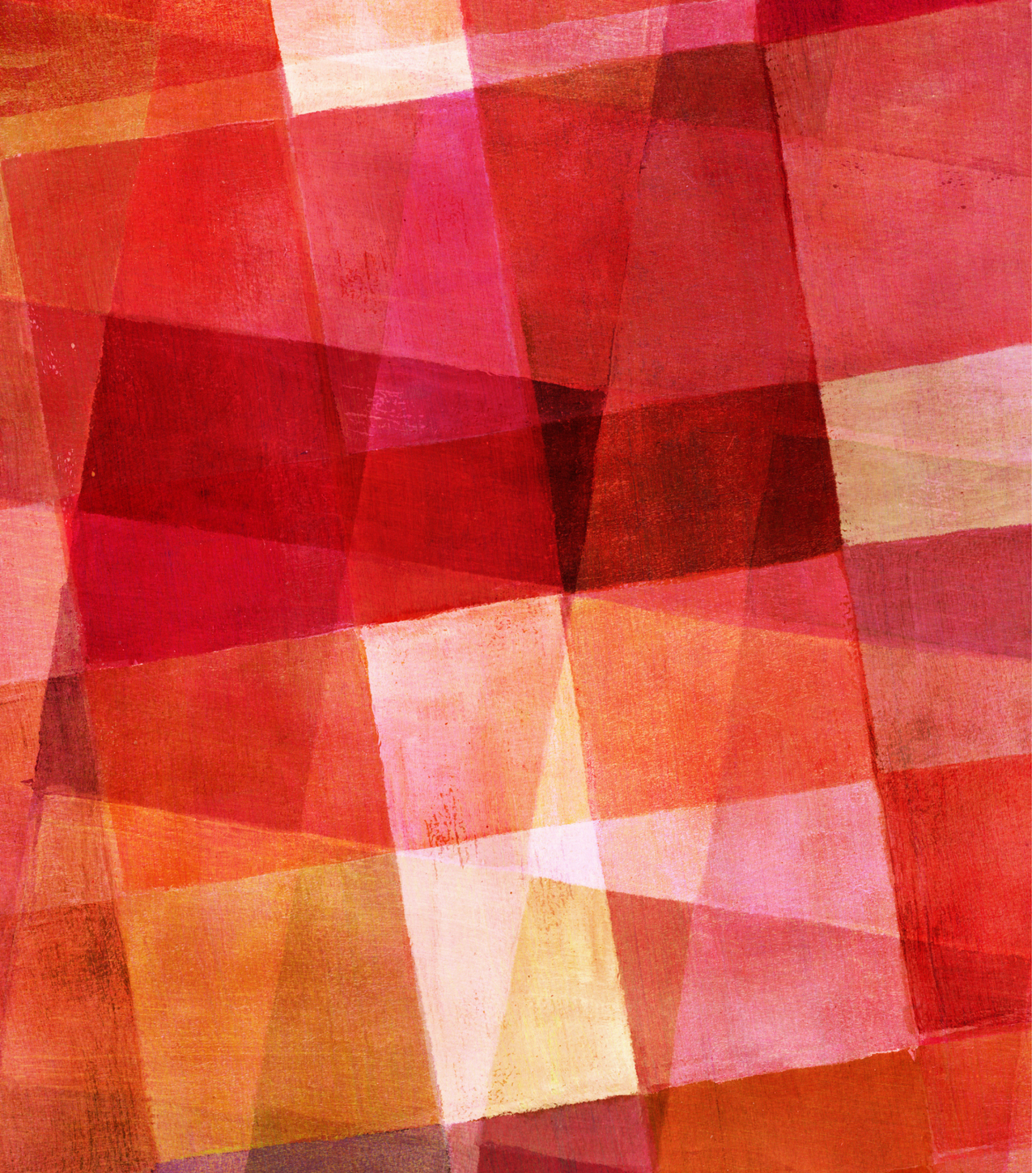
WHAT IS CIRCUITPYTHON?

- A version of Python that runs on more than 300 microcontroller boards
- CircuitPython also runs on Raspberry Pi and other single-board computers (SBC)
- CircuitPython is a fork of the MicroPython project
- Sponsored by Adafruit, an NYC-based open source hardware manufacturer



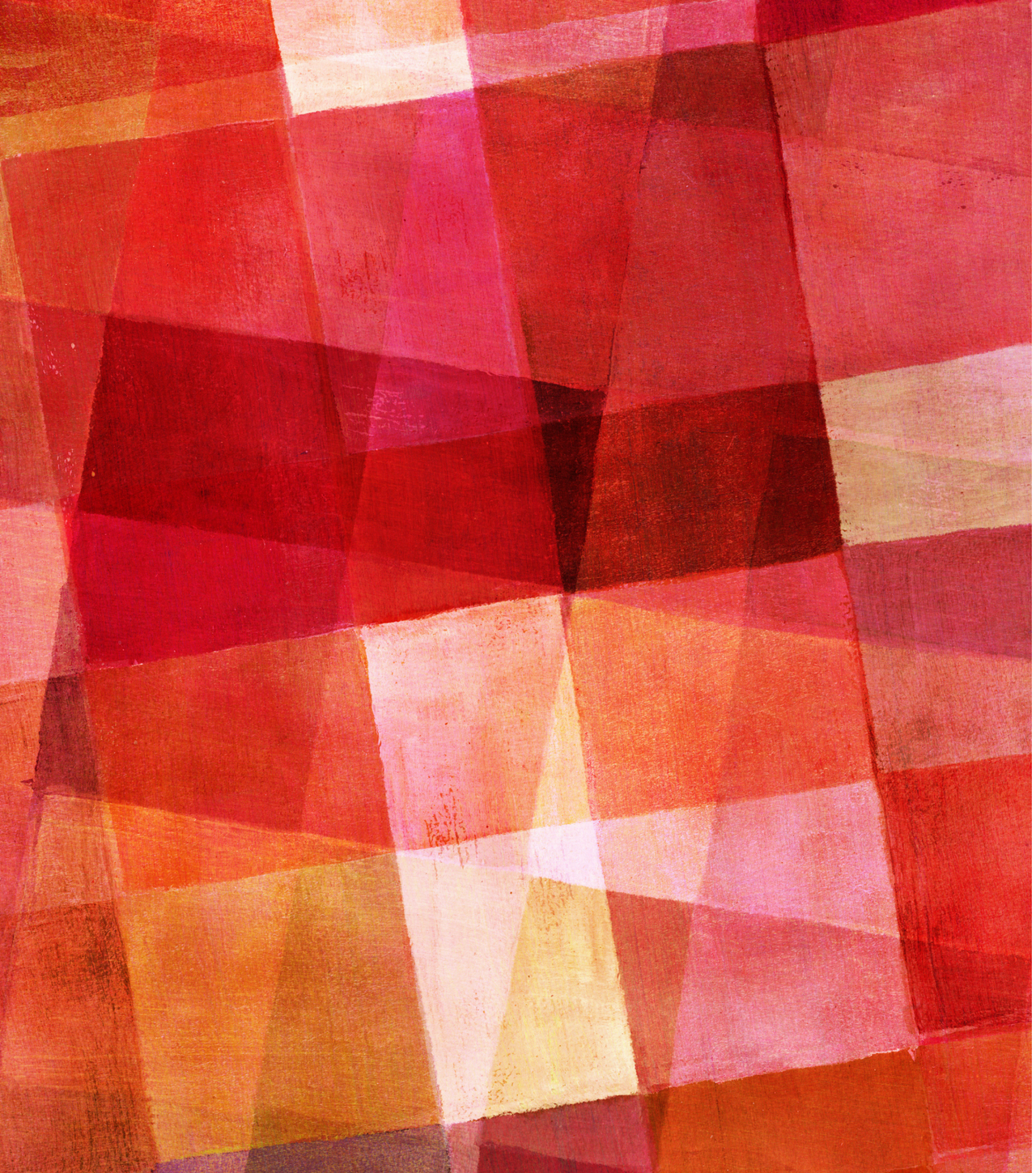
WHY CIRCUITPYTHON?

- Microcontrollers were traditionally programmed in C or C++
- Changing your code typically required recompiling and then re-flashing the microcontroller
- Interfacing with hardware traditionally required direct manipulation of the hardware's I/O signals
- All of these things added up to significant **barriers to entry and use** for non-expert users of this hardware.



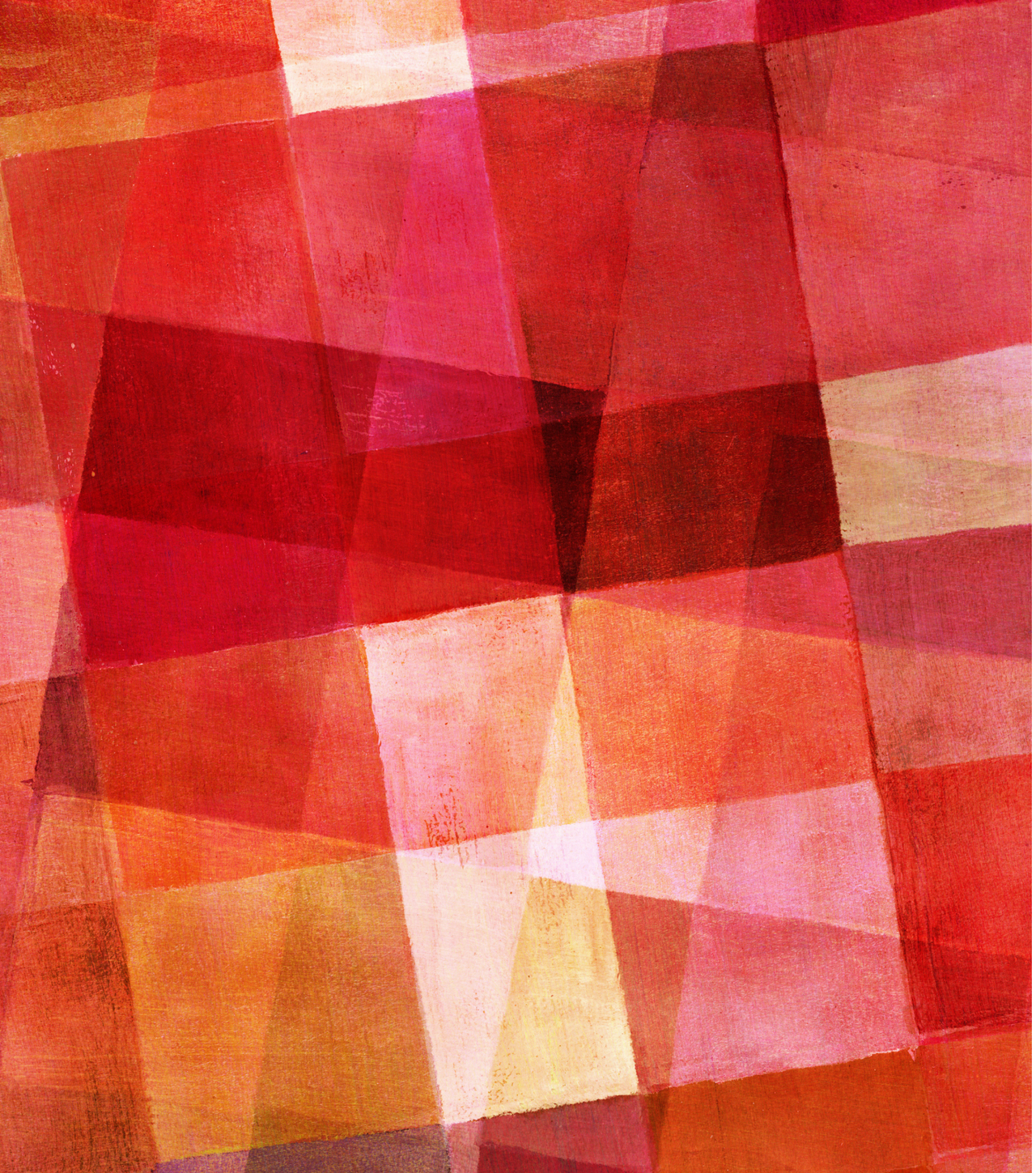
HOW DOES IT WORK?

- Plug a CircuitPython microcontroller into your computer's USB port. It appears on your computer as a USB drive.
- Download your code (and hardware support libraries) to the drive. CircuitPython automatically starts running your code.
- Connect to the USB serial port to see debugging output and access a Python REPL.
- When you change your code, the device automatically restarts.
- It's that easy!



DEVELOPMENT TOOLS

- You can use any Python development tools you want:
  - Visual Studio Code
  - PyCharm
  - TextMate/Sublime Text/etc.
- The easiest way to install libraries is with **circup** (pip install circup)
- The **mu editor** (codewith.mu) is handy for beginners and also provides an integrated connection to the serial port and REPL.



SO WHAT CAN WE DO WITH THIS?

- CircuitPython projects can interact with:
  - Sensors
  - Lights
  - Motors
  - WiFi / Bluetooth / LoRA
  - And much more!
- Some CircuitPython devices can behave like USB HID devices - so you can build your own smart keyboard.
- **The limit is really your imagination!**

DEMO 1

The obligatory blinking light

THE BLINKING NEOPIXEL

```
import time
import board
import neopixel

pixel = neopixel.NeoPixel(board.NEOPIXEL, 1)
while True:
    pixel.fill((140, 25, 255))
    time.sleep(1.0)
    pixel.fill((0, 0, 0))
    time.sleep(1.0)
```

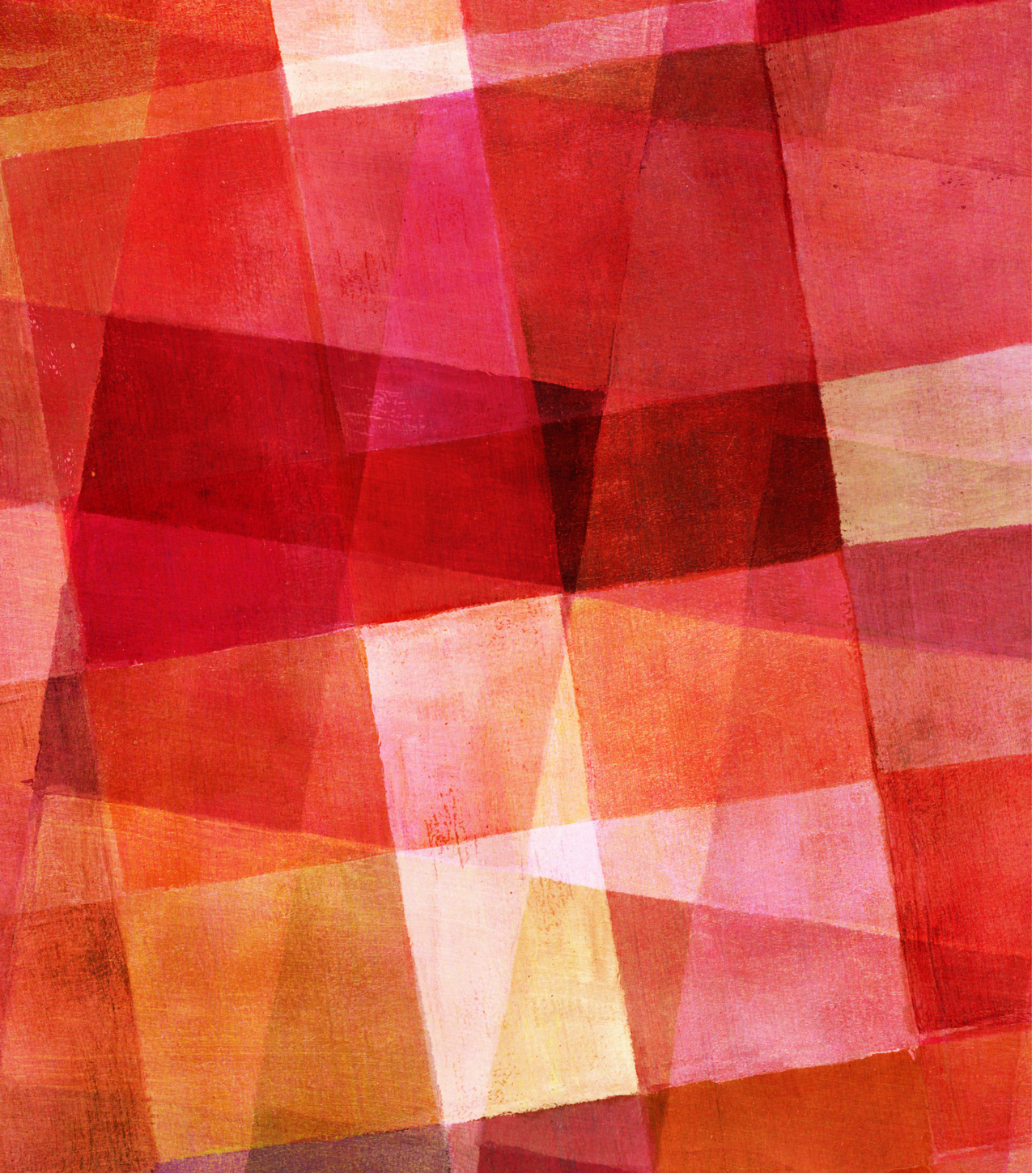
DEMO 2

Fun with NeoPixels

RAINBOW NEOPIXELS

```
import time
import board
from rainbowio import colorwheel
import neopixel

pixels = neopixel.NeoPixel(board.NEOPIXEL, 1, auto_write=False)
while True:
    for i in range(255):
        pixels.fill(colorwheel(i % 255))
        pixels.show()
        time.sleep(0.1)
```



THAT'S COOL - WHAT ELSE CAN IT DO?

- Let's try some code that talks to the environment.
  - BMP-280 temperature/barometric pressure sensor
- We'll connect these to our microcontroller with STEMMA QT, standard connectors for I2C sensors which makes the hardware super easy.
- If your microcontroller or sensor don't have STEMMA QT ports, you can connect them directly. (See the documentation for which pins to use.)

DEMO 3

Let's interact with the environment

READING THE SENSOR VALUES

```
import board
import adafruit_bmp280

i2c = board.I2C()
sensor = adafruit_bmp280.Adafruit_BMP280_I2C(i2c)

# Calibrate the sensor
sensor.sea_level_pressure = 1011.85

while True:
    print(f"Temperature: {((sensor.temperature*1.8)+32):.2f} F")
    print(f"Pressure   : {(sensor.pressure / 33.86):.2f} in Hg")
    print(f"Altitude   : {(sensor.altitude * 3.3):.1f} ft MSL\n")
    time.sleep(5)
```

DEMO 4

DisplayIO and Drawing on LED displays

SIMPLE DISPLAY EXAMPLE IN DISPLAYIO

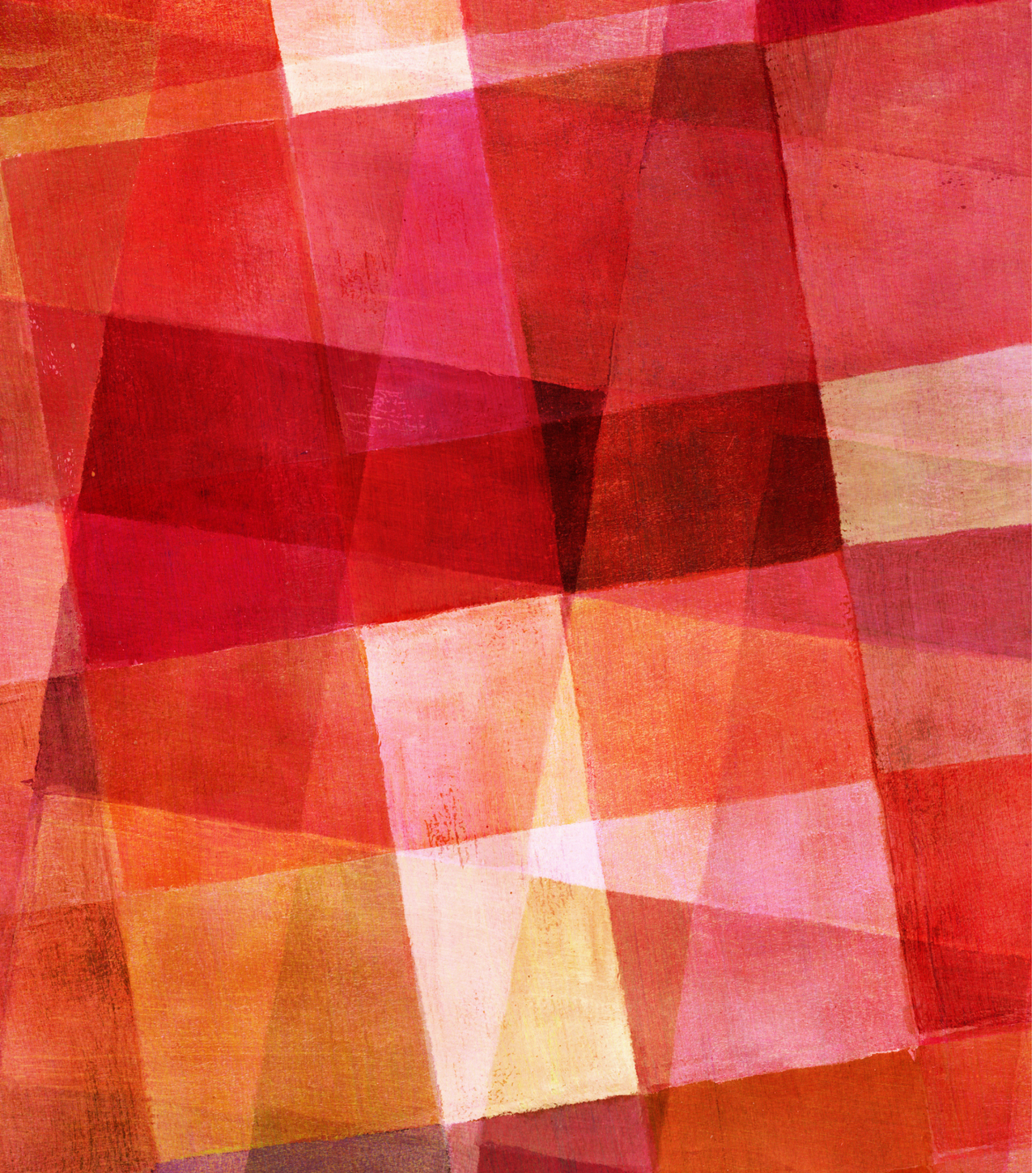
```
import time
from random import randint
import board
import terminalio
from adafruit_display_text import label

display = board.DISPLAY

text = "HELLO WORLD"
font = terminalio.FONT
color = 0x0000FF

text_area = label.Label(font, text=text, color=color)
text_area.x = 100
text_area.y = 80
display.show(text_area)

while True:
    time.sleep(0.5)
    text_area.x = random.randint(1, 120)
    text_area.y = random.randint(1, 120)
```



GETTING INVOLVED

- Buy a CircuitPython device and try it out!
  - www.adafruit.com
  - www.circuitpython.org
  - Use the discount code **DESERTPY** on Adafruit to save on your hardware!
- Join the Adafruit Discord (adafru.it/discord) in the #help-with-circuitpython and #circuitpython-dev channels
- Check out hundreds of project guides at learn.adafruit.com

QUESTIONS?

twitch.tv/**tammymakesthings**

itsme@tammymakesthings.com
github.com/**tammymakesthings**