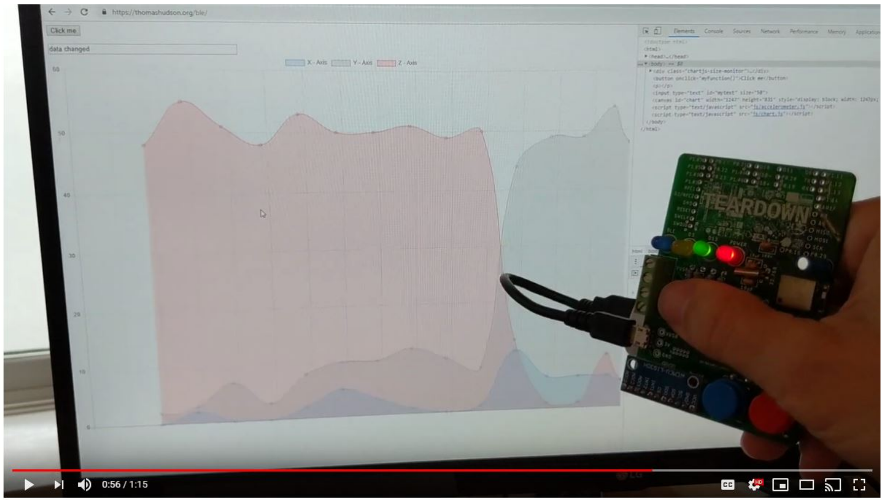
Teardown 2019 - Soldering and Web BLE tutorial

This is a soldering and programming class in Circuitpython that includes a Web Ble demo



github.com/hydronics2/Teardown-2019

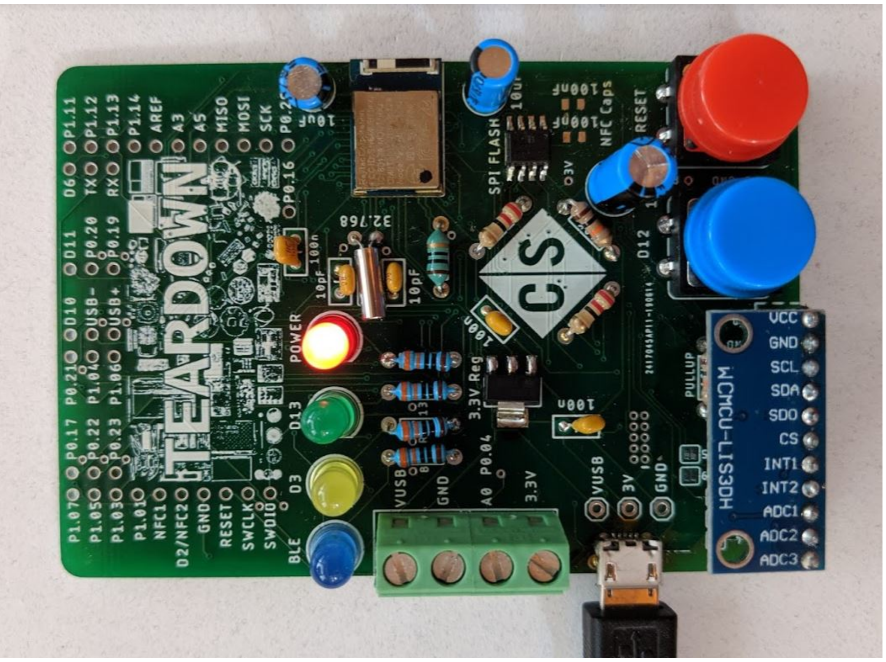
This class occurs in two parts -- Soldering and Programming.

We'll be soldering parts and headers onto a custom printed circuit board(PCB).
This board is a copy of the Adafruit Feather nrf52840 express.

When you connect from USB to your computer it shows up as an external hard drive with pre-loaded example scripts.

Soldering Tutorial

Programming Tutorial



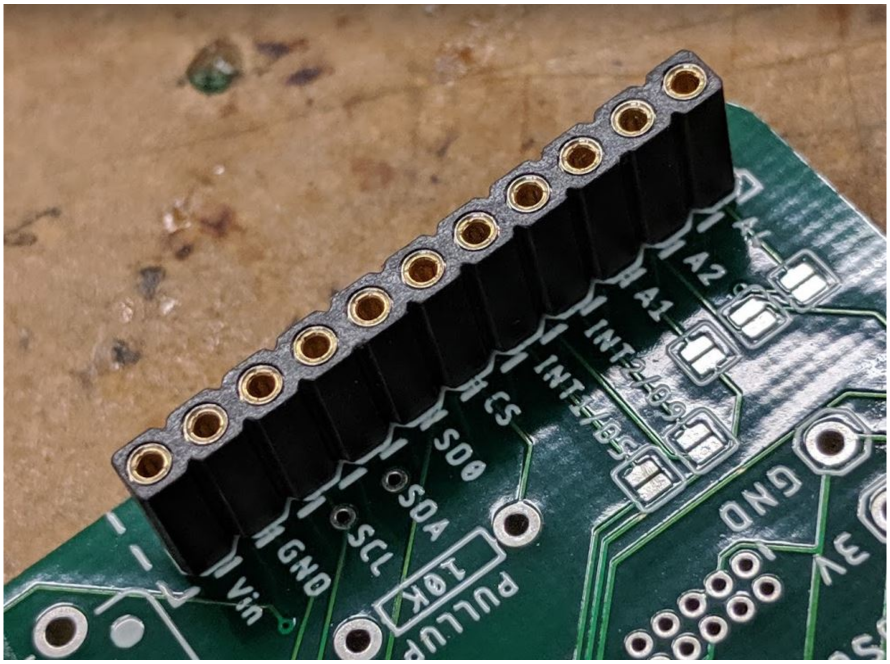
Soldering

Parts list

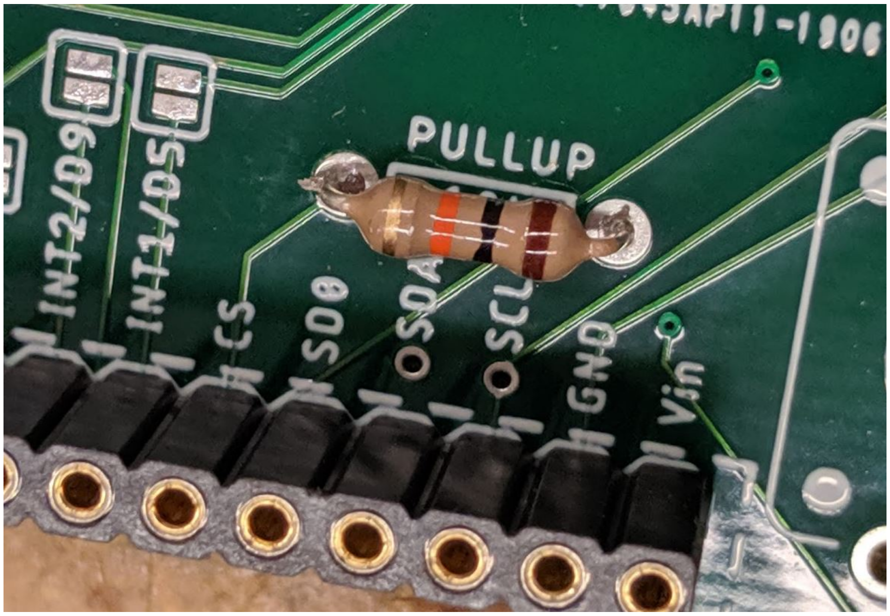
- Nordic Semiconductor nrf52840 bluetooth module
- LIS3DH Accelerometer
- SPI QUAD FLASH 2MB Memory

Full parts list on mouser.

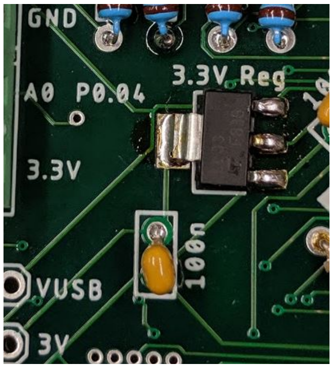
Link to schematic



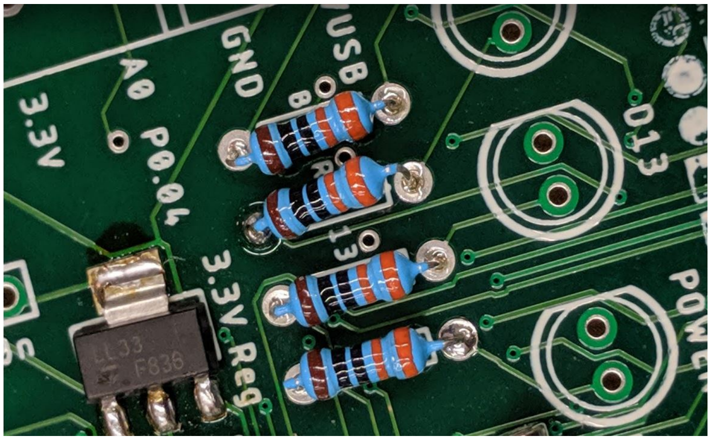
11 pin header for accelerometer.



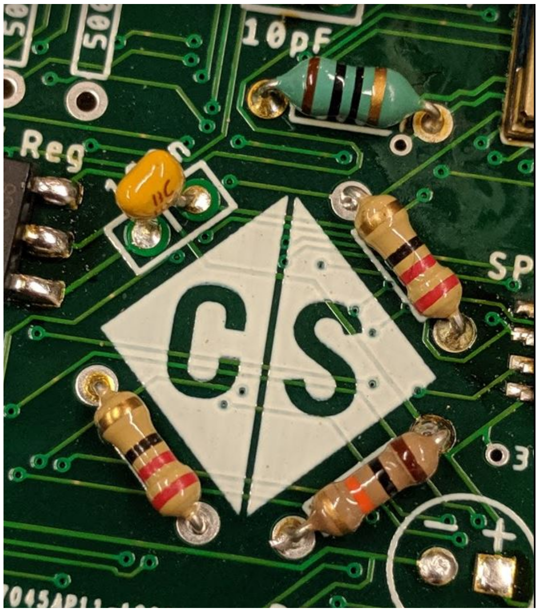
10K pullup resistor for the CS pin to initiate i2c on the accelerometer.



3.3volt regulator and 100n(0.1uF) capacitor



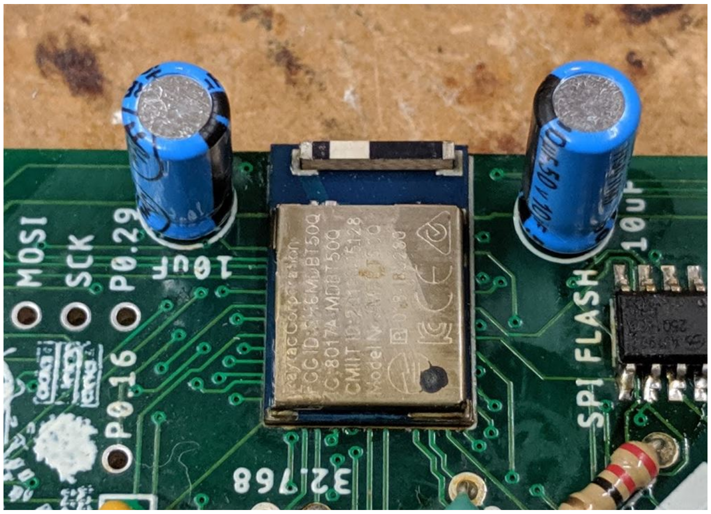
4 x Current limiting resistors for the LEDs. ~330ohms but anywhere from 220 to 510 is fine.



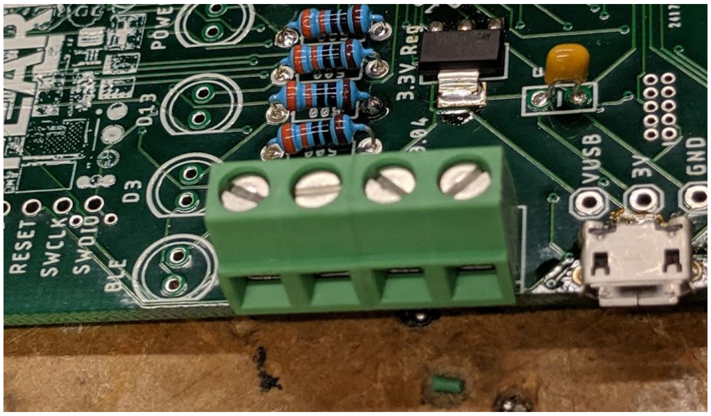
2 x 22ohm resistors connect the USB lines to the uController

1 x 10K pullup for the reset signal.

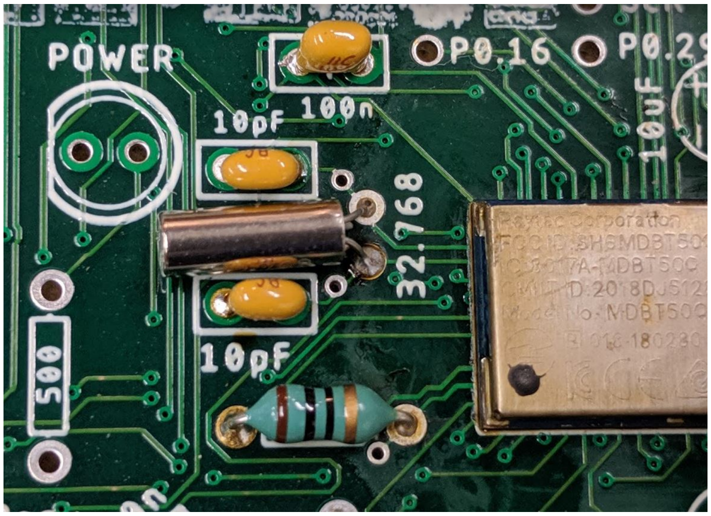
1 x 10uH ferrite inductor that filters high frequency noise per the mfg recommended layout



10uF caps along side the bluetooth module per the MFG recomended board layout.



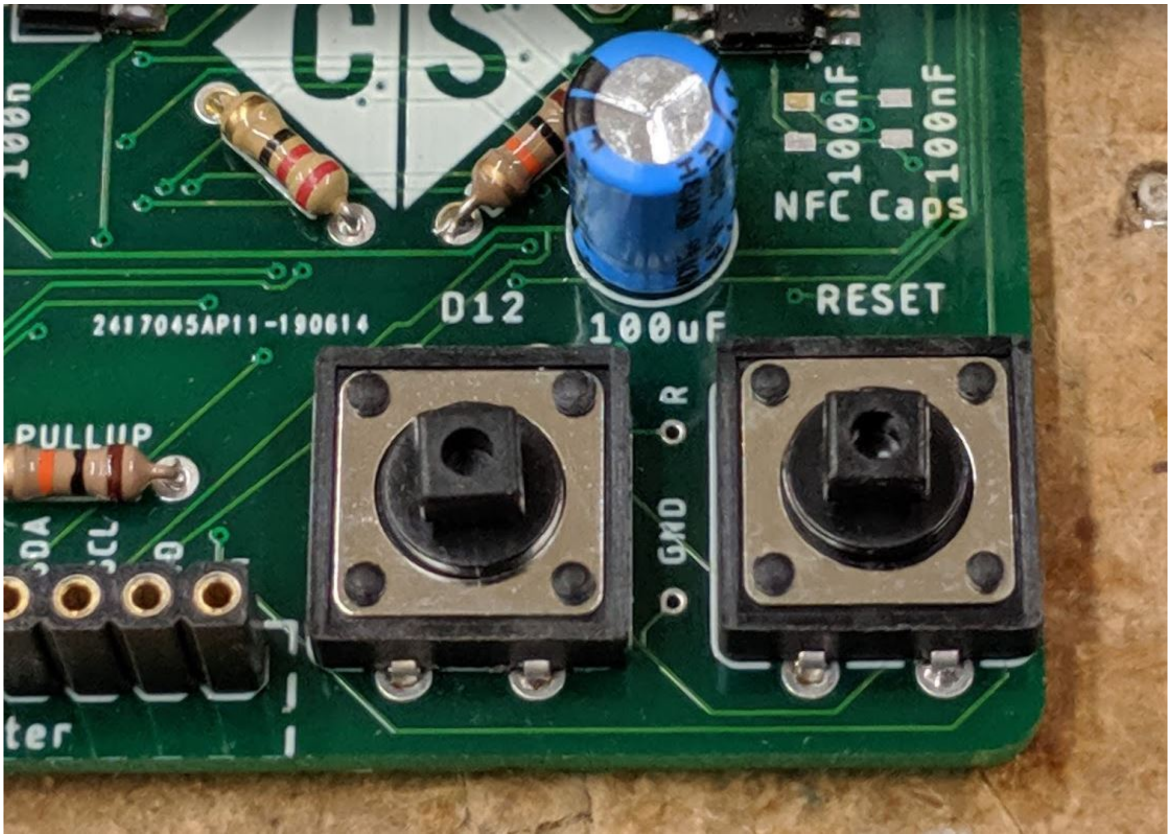
5mm pitch screw headers that we'll use for connecting wires and potentiometers.



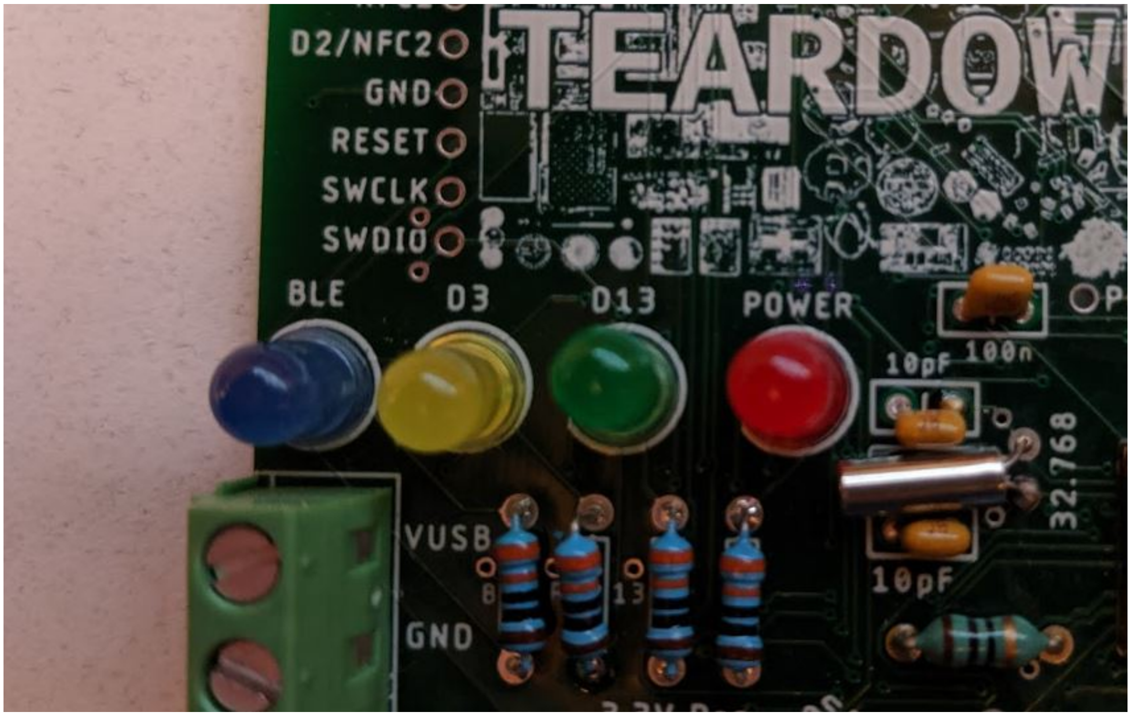
1 x 32.768khz crystal

2 x caps along side the crystal. these are 20pF!!!! not 10 as show on the silkscreen.

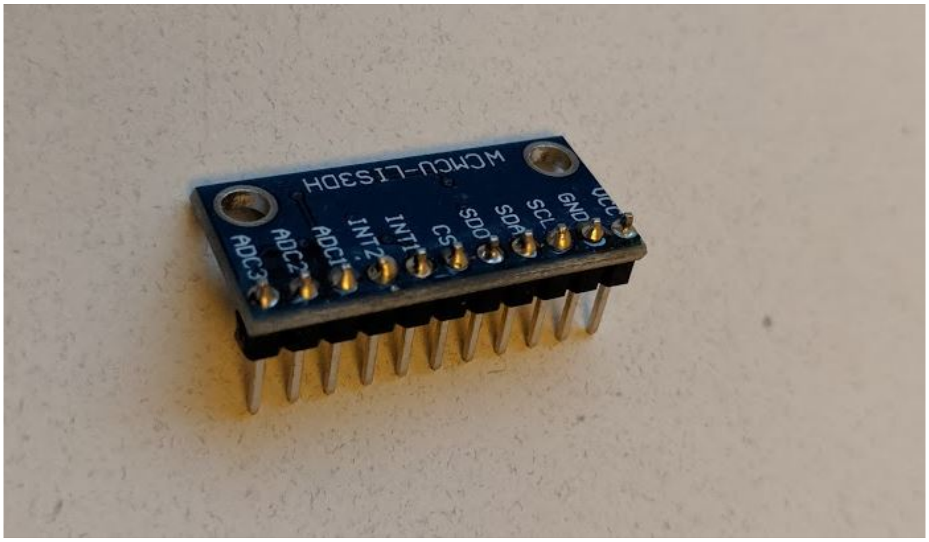
1 x 100nF(0.1uF) bypass capacitor that serves the aref signal per the mfg layout.



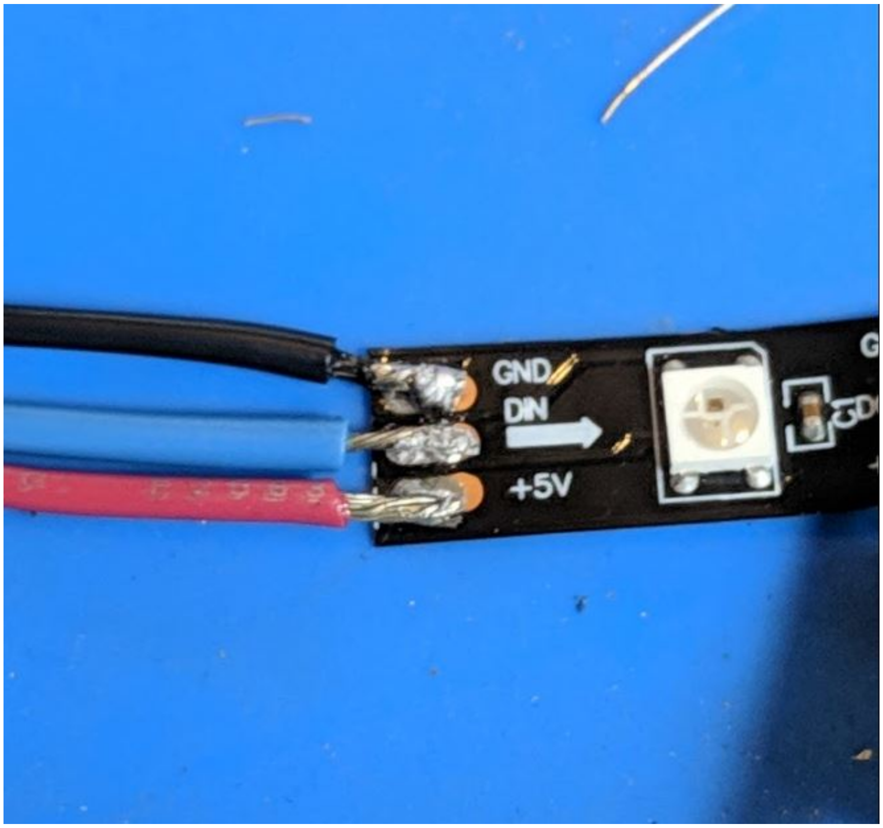
1 x 100uF capacitor serving the 3.3v regulator per the mfg layout.



LEDs!! pic whatever color you want. They are directional! Ask your neighbor if you don't know.



solder the male headers onto the little acclerometer board.



Solder 3 lines onto some WS2812 LEDs.

That's it. We're all finished! We're leaving a few footprints unpopulated for this class.

CircuitPython

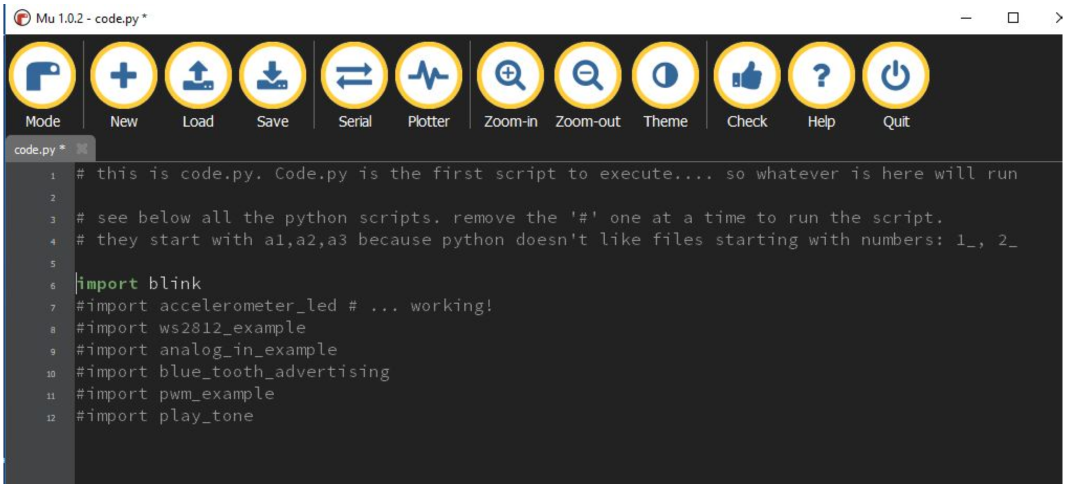
If you need to flash a bare board... here are some steps..

I uploaded the python scripts onto the 2MB SPI Flash chip before the class. When you plug the unit into your board it appears as a flash drive.

Download and install the MU editor if you have not done so already.

code.py

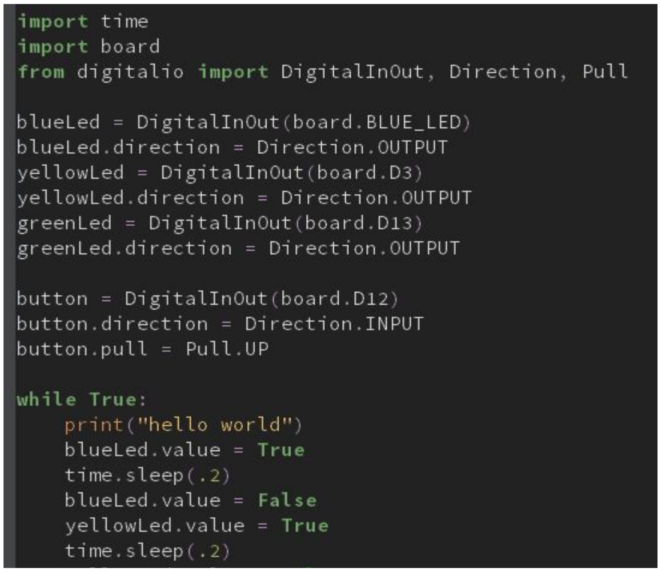
LOAD the code.py script. Code.py determines which scripts run. Uncomment the files you would like to run.



```
# this is code.py. Code.py is the first script to execute.... so whatever is here will run

# see below all the python scripts. remove the '#' one at a time to run the script.
# they start with a1,a2,a3 because python doesn't like files starting with numbers: 1_, 2_

import blink
#import accelerometer_led # ... working!
#import ws2812_example
#import analog_in_example
#import blue_tooth_advertising
#import pwm_example
#import play_tone
```

blink

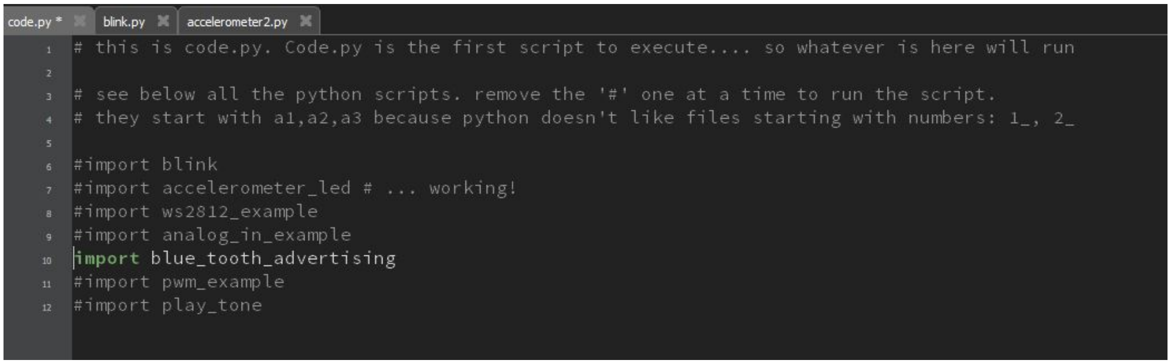
LOAD the "blink.py" script that's currently running. It blinks the LEDs on pin 13, pin 3, and BLUE_LED pin. Try adjusting the time.sleep(x) to speed up the blinking.



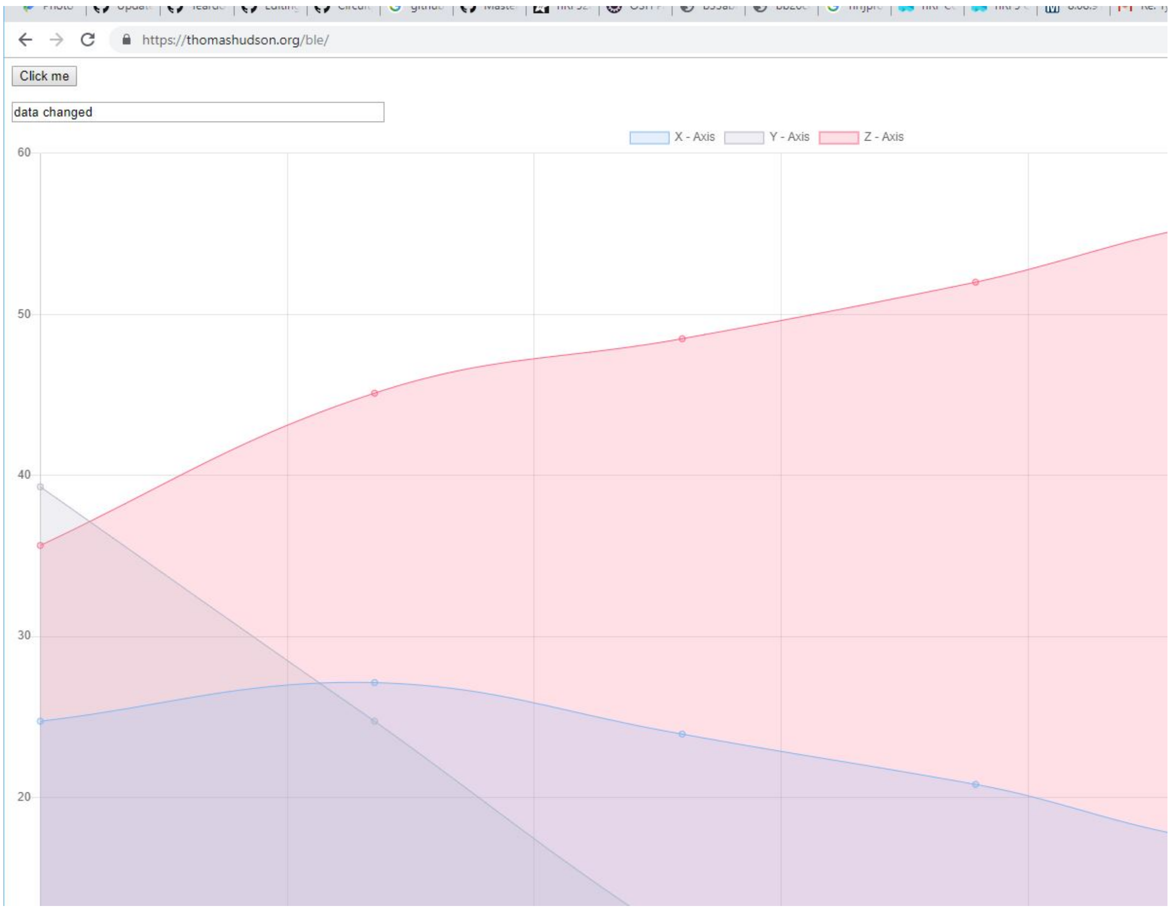
```
import time
import board
from digitalio import DigitalInOut, Direction, Pull

blueLed = DigitalInOut(board.BLUE_LED)
blueLed.direction = Direction.OUTPUT
yellowLed = DigitalInOut(board.D3)
yellowLed.direction = Direction.OUTPUT
greenLed = DigitalInOut(board.D13)
greenLed.direction = Direction.OUTPUT

button = DigitalInOut(board.D12)
button.direction = Direction.INPUT
button.pull = Pull.UP

while True:
    print("hello world")
    blueLed.value = True
    time.sleep(.2)
    blueLed.value = False
    yellowLed.value = True
    time.sleep(.2)
```

Web Ble

LOAD the blue_tooth_advertising.py script. Next open a chrome browser and connect through the browser using this address: https://thomashudson.org/ble. You have to use the https!



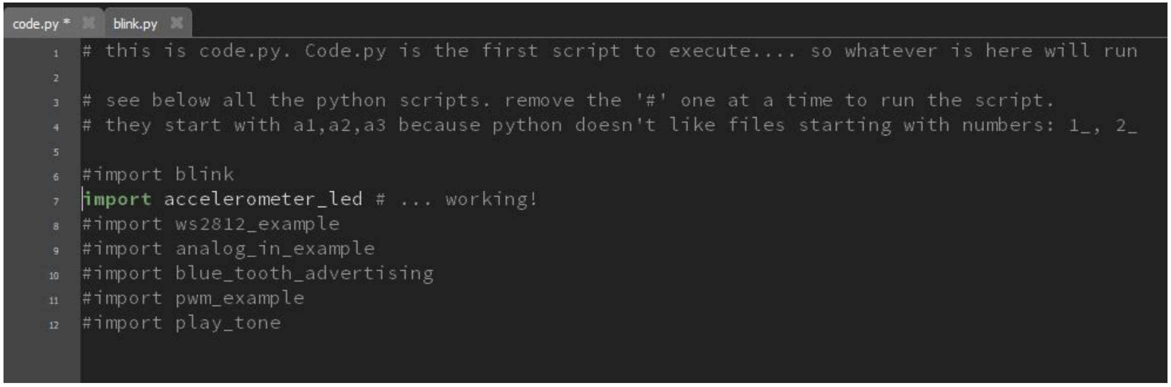
```
# this is code.py. Code.py is the first script to execute.... so whatever is here will run

# see below all the python scripts. remove the '#' one at a time to run the script.
# they start with a1,a2,a3 because python doesn't like files starting with numbers: 1_, 2_

#import blink
#import accelerometer_led # ... working!
#import ws2812_example
#import analog_in_example
import blue_tooth_advertising
#import pwm_example
#import play_tone
```



data changed

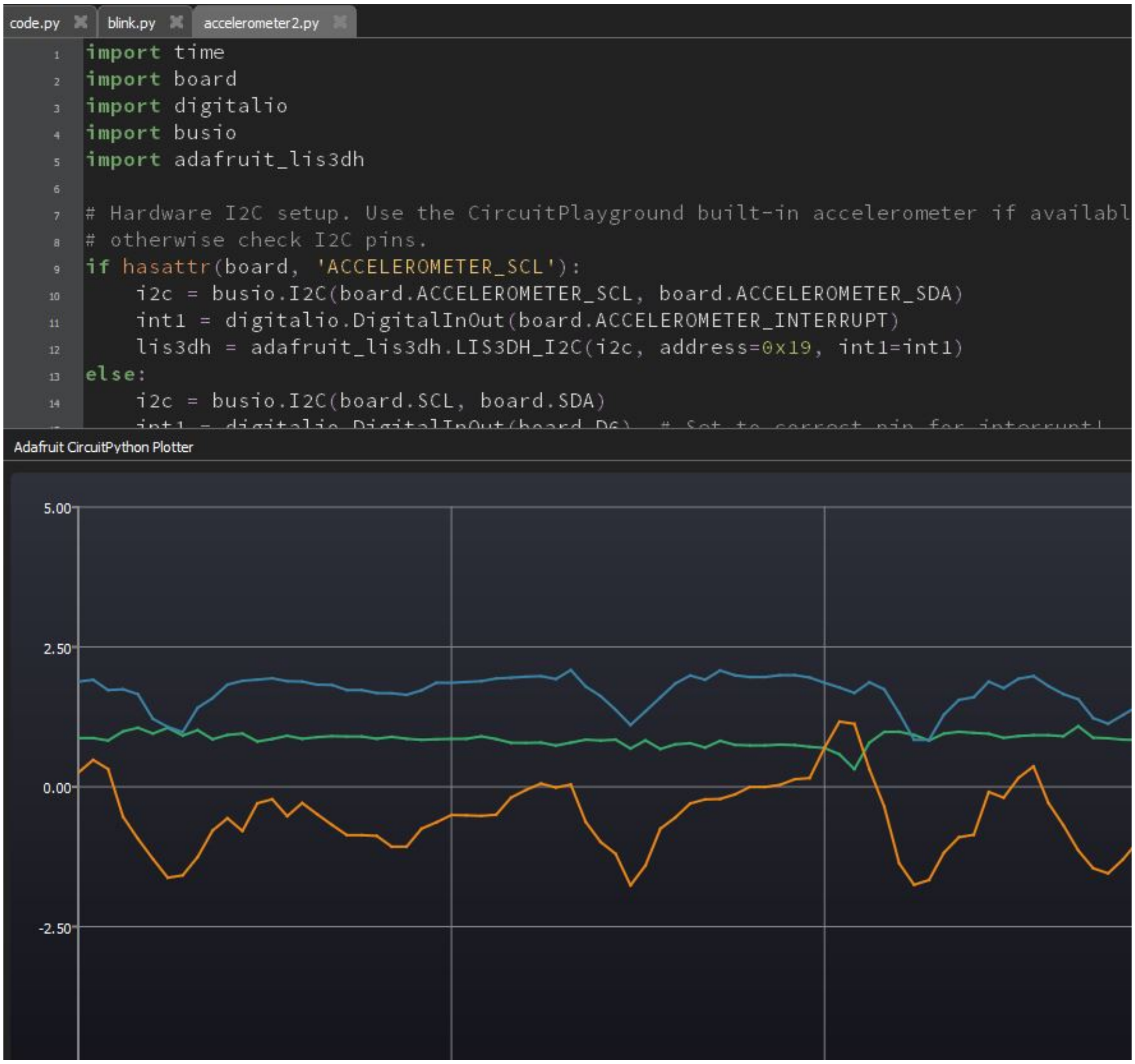
accelerometer

Modify the code.py script and uncomment accelerometer_led.



```
# this is code.py. Code.py is the first script to execute.... so whatever is here will run

# see below all the python scripts. remove the '#' one at a time to run the script.
# they start with a1,a2,a3 because python doesn't like files starting with numbers: 1_, 2_

#import blink
import accelerometer_led # ... working!
#import ws2812_example
#import analog_in_example
#import blue_tooth_advertising
#import pwm_example
#import play_tone
```

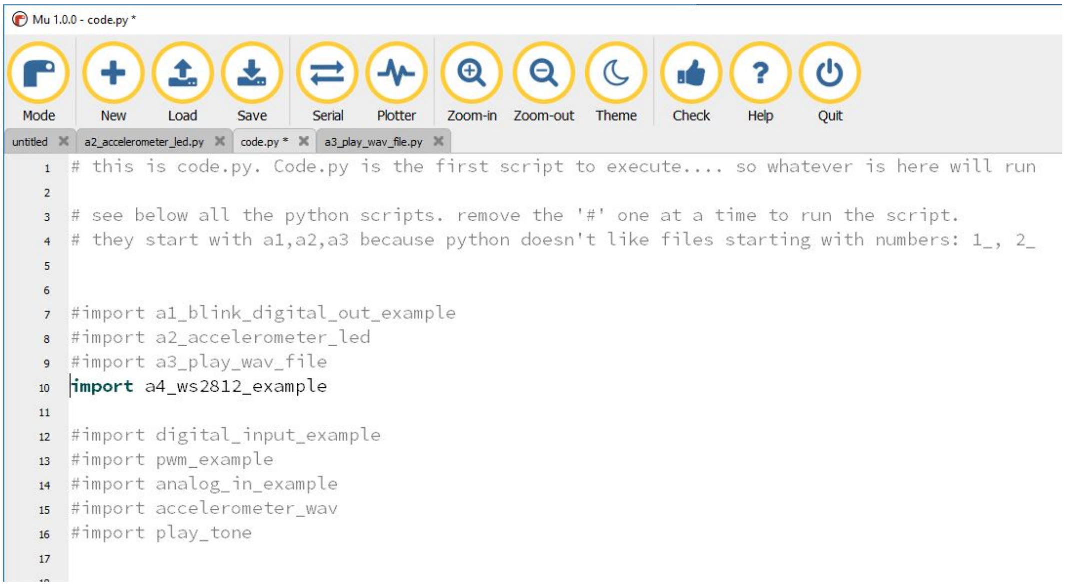
The script lights up an LED with increasing tilt. When you use a print statement with double perenthesis it allows the data to the plotter; such as, print((x,y,z)).



```
import time
import board
import digitalio
import busio
import adafruit_lis3dh

# Hardware I2C setup. Use the CircuitPlayground built-in accelerometer if availabl
# otherwise check I2C pins.
if hasattr(board, 'ACCELEROMETER_SCL'):
    i2c = busio.I2C(board.ACCELEROMETER_SCL, board.ACCELEROMETER_SDA)
    int1 = digitalio.DigitalInOut(board.ACCELEROMETER_INTERRUPT)
    lis3dh = adafruit_lis3dh.LIS3DH_I2C(i2c, address=0x19, int1=int1)
else:
    i2c = busio.I2C(board.SCL, board.SDA)
    int1 = digitalio.DigitalInOut(board.D6)  # Set to correct pin for interrupt!
```

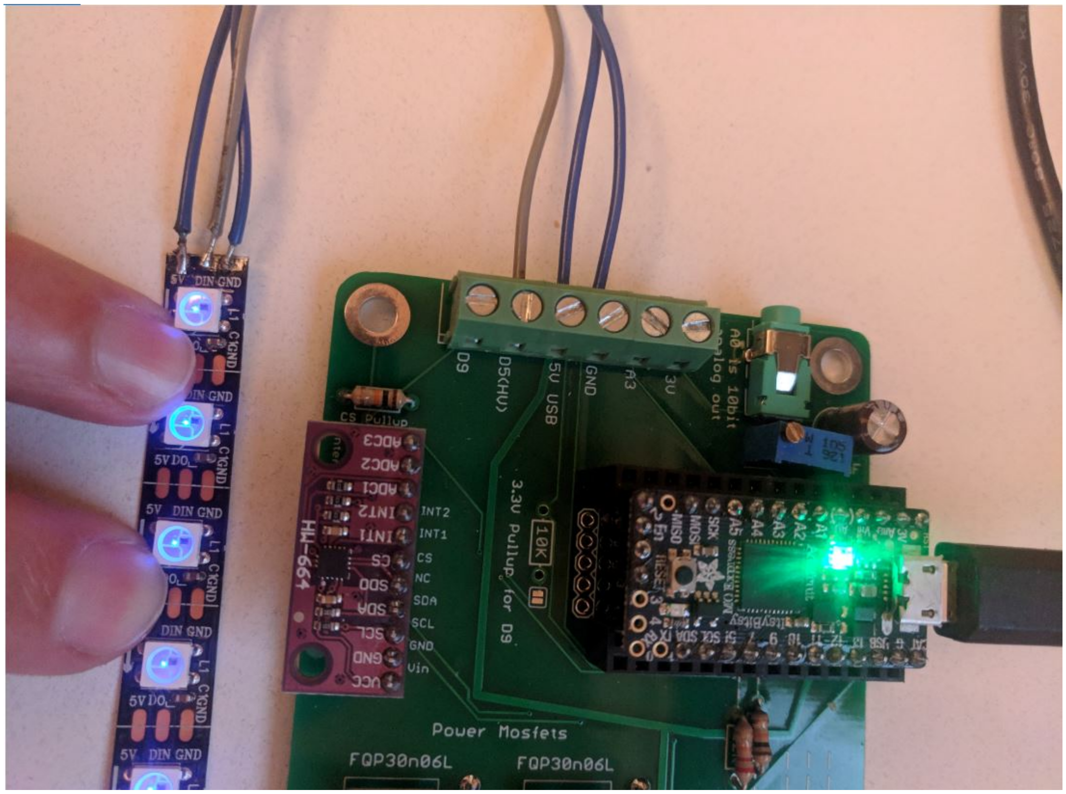
ws2812_example

Modify the code.py script and unomment a4_ws2812_example



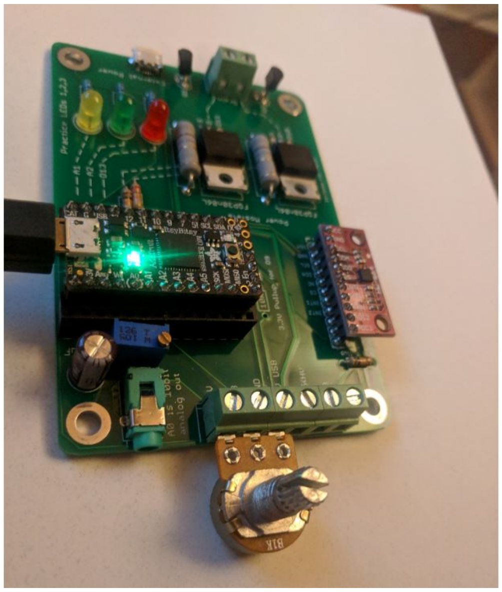
```
# this is code.py. Code.py is the first script to execute.... so whatever is here will run

# see below all the python scripts. remove the '#' one at a time to run the script.
# they start with a1,a2,a3 because python doesn't like files starting with numbers: 1_, 2_


#import a1_blink_digital_out_example
#import a2_accelerometer_led
#import a3_play_wav_file
import a4_ws2812_example

#import digital_input_example
#import pwm_example
#import analog_in_example
#import accelerometer_wav
#import play_tone

```

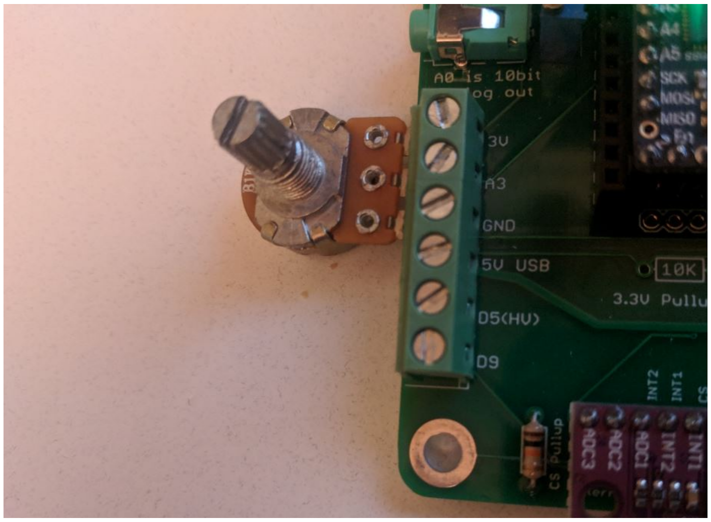
solder three leads onto your LED strip on the correct side (start of the arrow). Insert the LEDs into the screw headers. LOAD the sketch and modify the "num_leds" variable to the number of LEDs in your strip. The red, green, and blue LEDs can have values from 0 to 255.



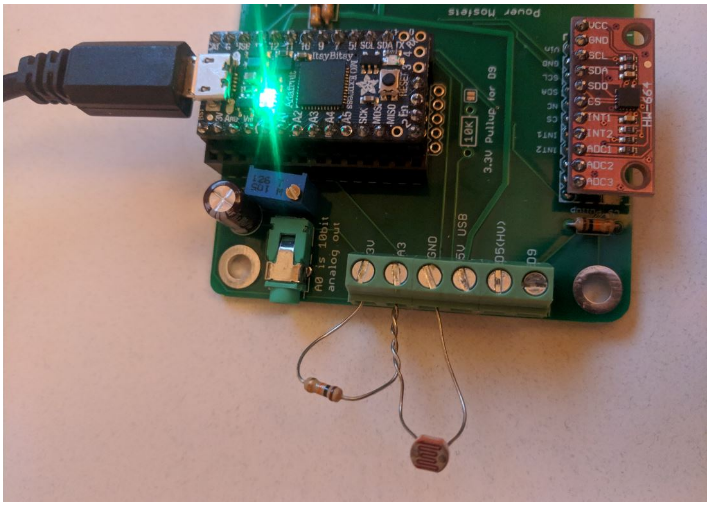
analog_in_example

Modify the code.py script and unomment analog_in_example. Try plugging in both the potentiometer and photocell to see how the analog-to-digital converter(ADC) on pin A0 interprets the signals.





Make the photocell voltage divder using a photocell and a 10k resistor.



You can see the data on the plotter by using the print statement with extra perenthesis and a comma, "print((analogValue,))"