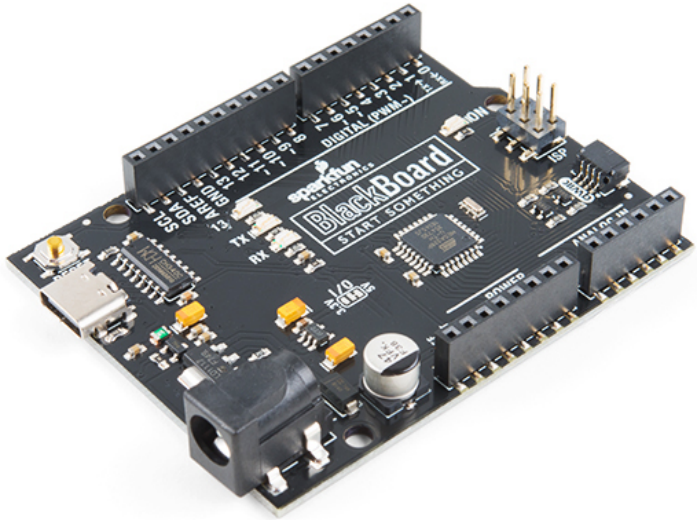
Getting Started with Arduino

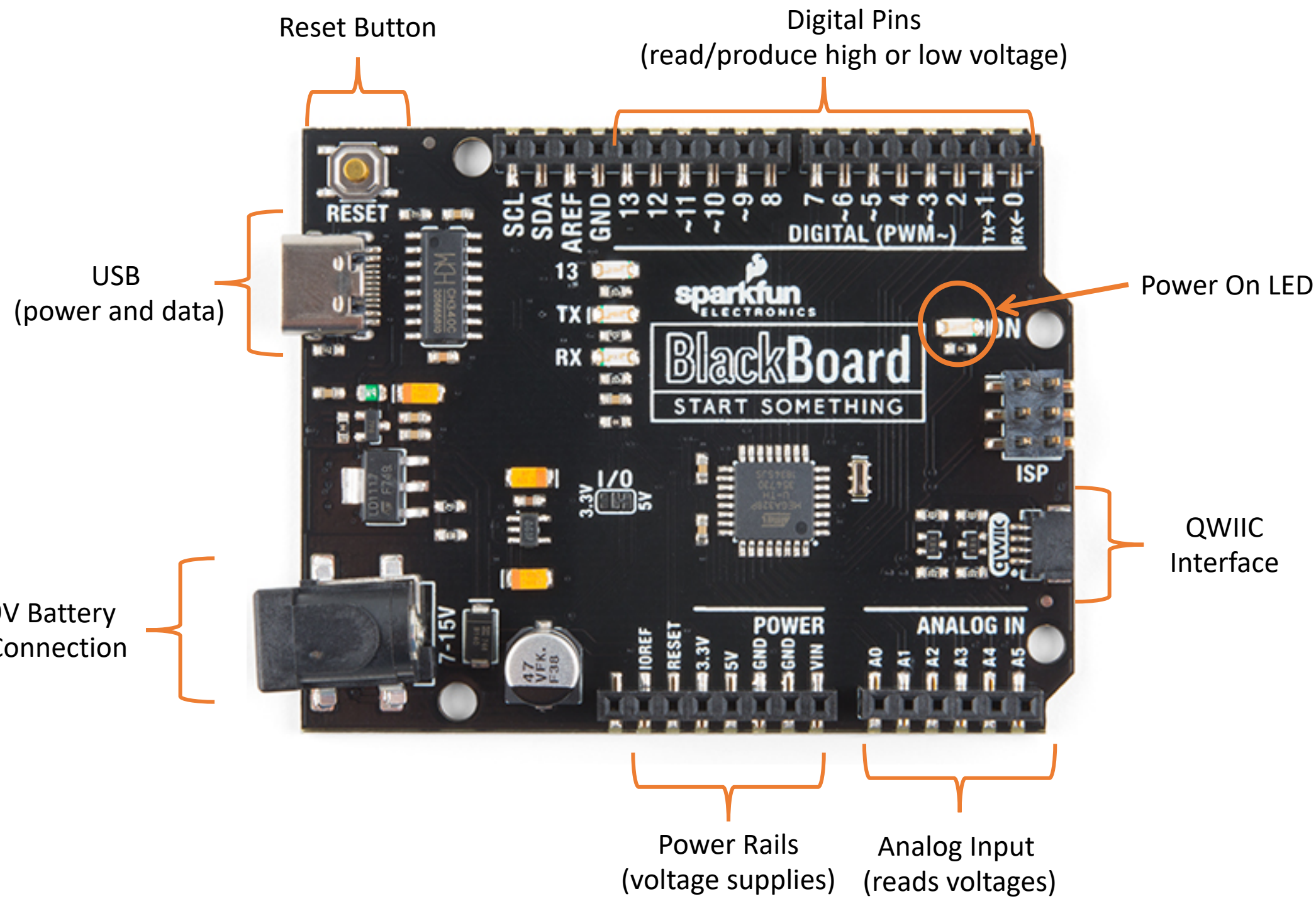
Running Programs and Collecting Data

Product images © SparkFun with Creative Commons 3.0

Sparkfun Blackboard



- Sparkfun Blackboard is a clone of the popular Arduino microprocessor
- Powered via USB connection and programmed with free / open-source Arduino IDE software.
- Will connect sensors to board via the QWIIC and Analog interfaces.

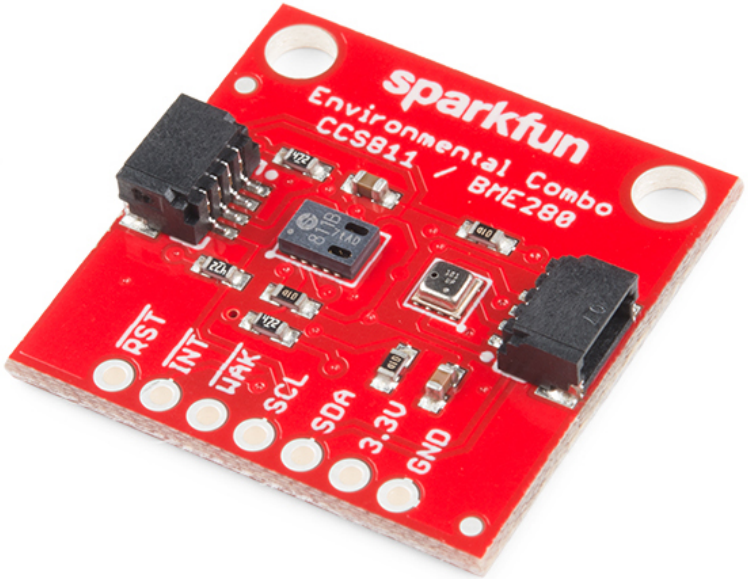

Reset Button
Digital Pins
(read/produce high or low voltage)
USB
(power and data)
Power On LED
QWIIC
Interface
9V Battery
Connection
Power Rails
(voltage supplies)
Analog Input
(reads voltages)
RESET
SCL
SDA
AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX→1
RX←0
DIGITAL (PWM~)
13
TX
RX
sparkfun
ELECTRONICS
BlackBoard
START SOMETHING
ON
ISP
I/O
3.3V
5V
7-15V
POWER
IOREF
RESET
3.3V
5V
GND
GND
VIN
ANALOG IN
A0
A1
A2
A3
A4
A5

Glossary of Terms

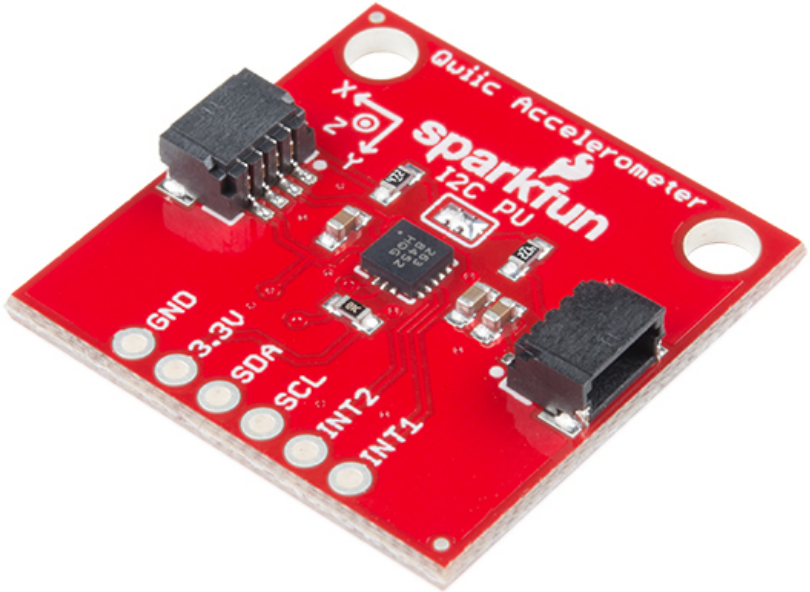
- **Arduino/Blackboard**: for the purpose of this class these words are interchangeable. Technically a Blackboard is an Arduino-compatible clone produced by SparkFun.
- **Digital**: signals or information that exist in discrete states (ON/OFF, HIGH/LOW, 5 volts / 0 volts).
- **Analog**: signals or information that can take on a variety of values (e.g. anything between 0 volts and 5 volts.)
- **QWIIC**: a proprietary connector that makes attaching sensors to the Blackboard easier.
- **Compile**: use a software tool to turn your code into machine language
- **Upload**: use a software tool transmit your compiled program to your Blackboard.
- **GitHub**: a popular online platform for sharing/editing software projects. We will use this to host the course software and informational materials.

Your Hardware Kit

Starting Arduino Kit



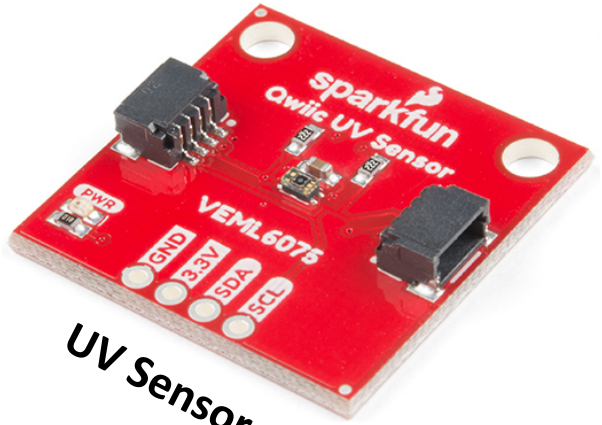
Environmental Combo board housing two sensors: **CCS811** (CO_2 and TVOC) and **BME280** (humidity, pressure, altitude, and temperature)



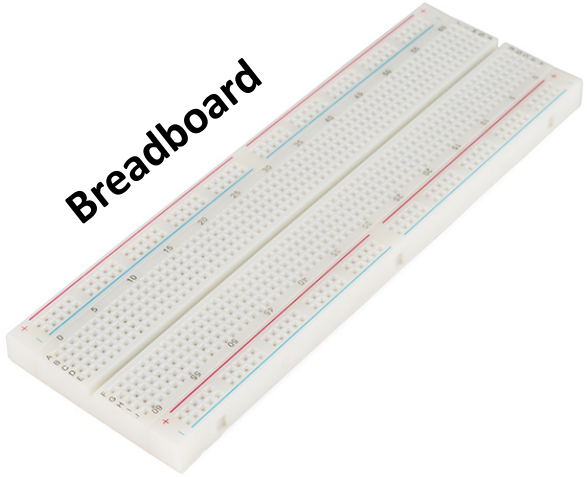
Board contains **MMA8452** Accelerometer that provides acceleration in X, Y, and Z axis. Axis are printed on the board silkscreen.

Sensors are accessed via the QWIIC interface and will be demo'd with provided software

Additional Parts Available as Necessary…



UV Sensor

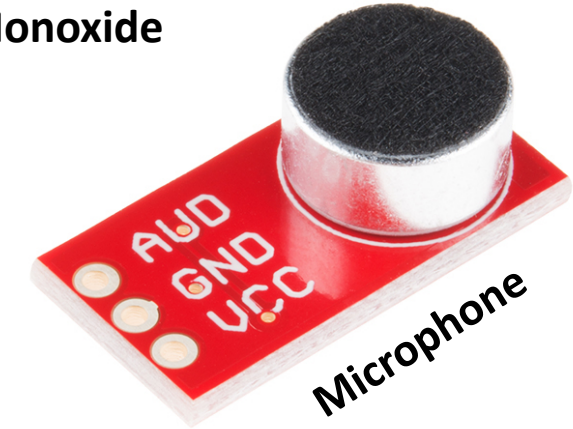


Breadboard

Carbon Monoxide

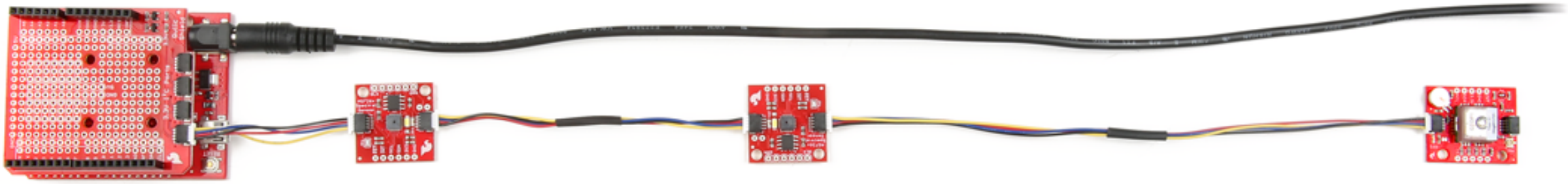
Ambient Light

Radio



Microphone

Connecting QWIIC Sensors

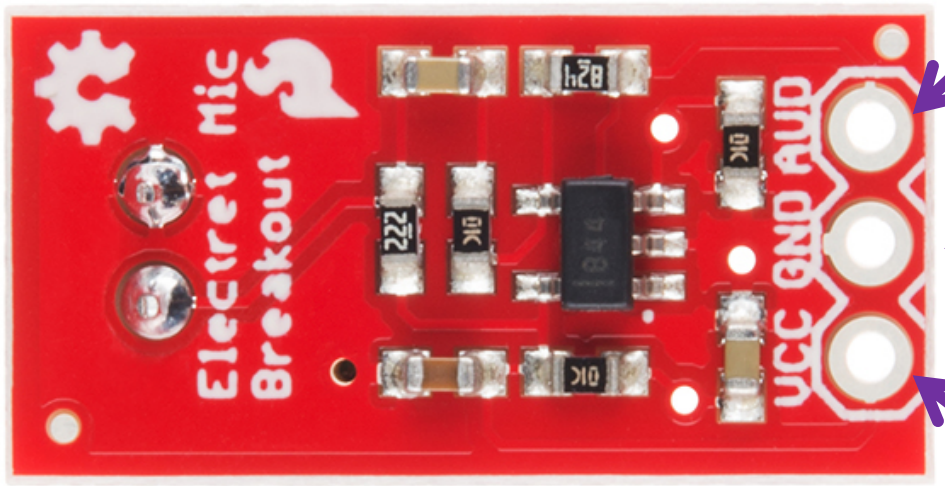


Devices use the “digital” I2C interface to communicate. Simply chain/connect QWIIC compatible sensors with a QWIIC cable.

Order of components does not matter. Connector will only fit one orientation.

Connecting Analog Sensors

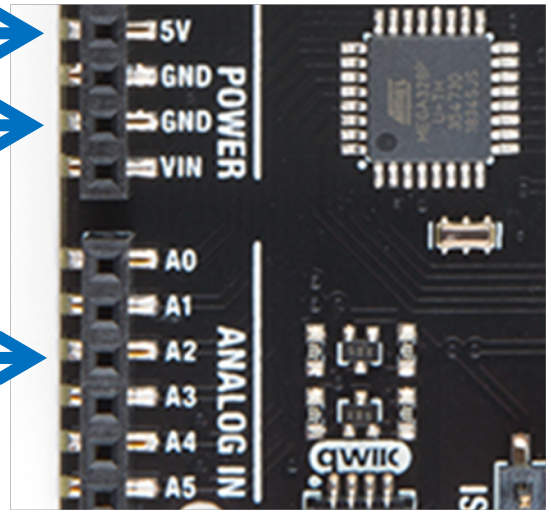
- Can be more challenging. Generally need to: (1) apply power (5V or 3.3V) and ground to the device with breadboard and wires and (2) read in a signal via Analog Input.



Analog OUTPUT *from* the microphone that should be attached *into* the Analog INPUT of the Blackboard

Connect to GND on the Blackboard's power rails to ground the microphone

Apply 5V from the Blackboard's power rails to power the microphone



Warnings on Analog Sensors

- You can easily damage the sensor and the Blackboard if components are connected incorrectly. You must get instructor approval before “powering” these devices.

- Always check the SparkFun website for Hook Up and Connection guides. They may have videos showing exactly what to do.

- Regardless of your section, visit with Dr. Forsyth to discuss your wiring and connections.

The Arduino Interface

Using the Arduino IDE to “talk” with your board

Compile Button

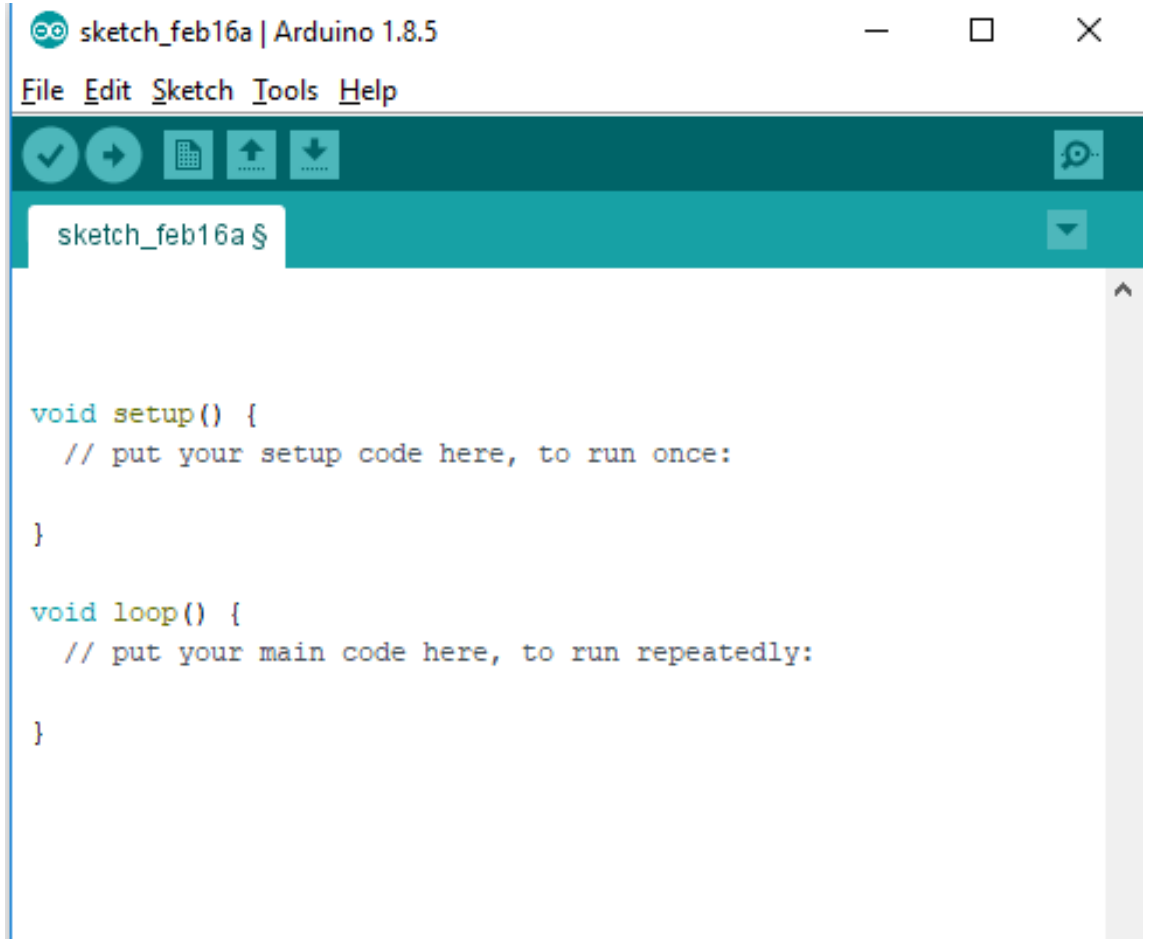
Upload Button

Serial Terminal: shows the “output” from your code when using Serial.print

Write all your code here

Console window showing status and error messages

Anatomy of an Arduino Program



- Has two functions that govern behavior: setup() and loop()
- Items placed in setup() will execute once when the device is powered on or when the reset button is pressed
- Code in loop() will execute continuously. Once the program reaches the bottom of loop it starts back over at the top.
- Code placed outside of loop() and setup() can be seen by “anyone”. Variables inside each function are only seen by those functions.

Making a light blink….

Make the LED pin an output. Do this once.

```
// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);                       // wait for a second
  digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);                       // wait for a second
}
```

Turn the LED ON by making a pin HIGH voltage

Turn the LED OFF by making a pin LOW voltage

These four lines of code execute forever because they are in loop()

How to get data off the Arduino…

```
//Request the x-acceleration. Note: this calls the read() function implicitly
//instead of explicitly like the SparkFun example. All accelerations are in g's.
float xAccel = accel.getXAcceleration();

float yAccel = accel.getYAcceleration();

float zAccel = accel.getZAcceleration();

Serial.print("X: ");
Serial.print(xAccel,3); //print to three decimal places

Serial.print(" Y: ");
Serial.print(yAccel,3); //print to three decimal places

Serial.print(" Z: ");
Serial.println(zAccel,3); //print to three decimal places
```

- Easiest method is to “print” messages from the Arduino to your computer via the USB/Serial interface.
- Many other methods: add an LED display, transmit via radio, store on external memory…etc.
- *print()* displays some variable or string. *println()* adds a “new line” character after printing.

Wait, is this a programming class?!

No, but you need to understand some of what the program does. You are also free to write your own code. The instructors will provide basic software to get started.

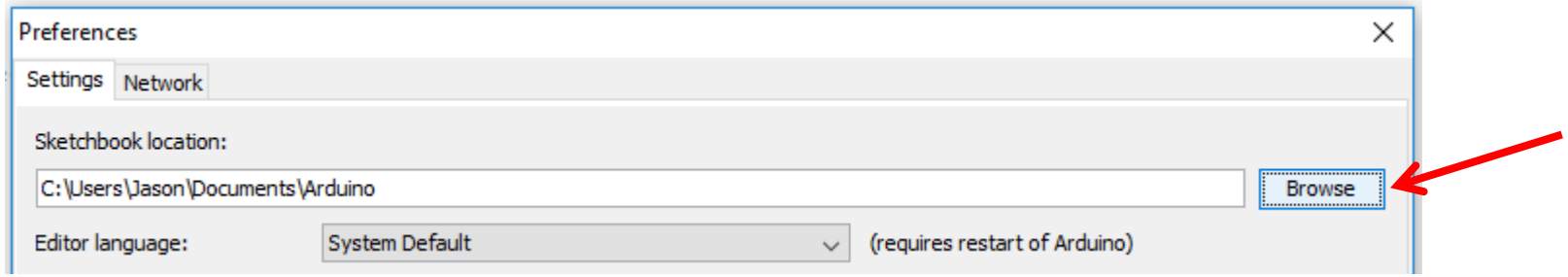
Step 1: Setup the Arduino Software

Configuring your Arduino IDE

- All Arduino “programs” live in a “sketchbook” on your computer. We will provide a sketchbook with all required software.

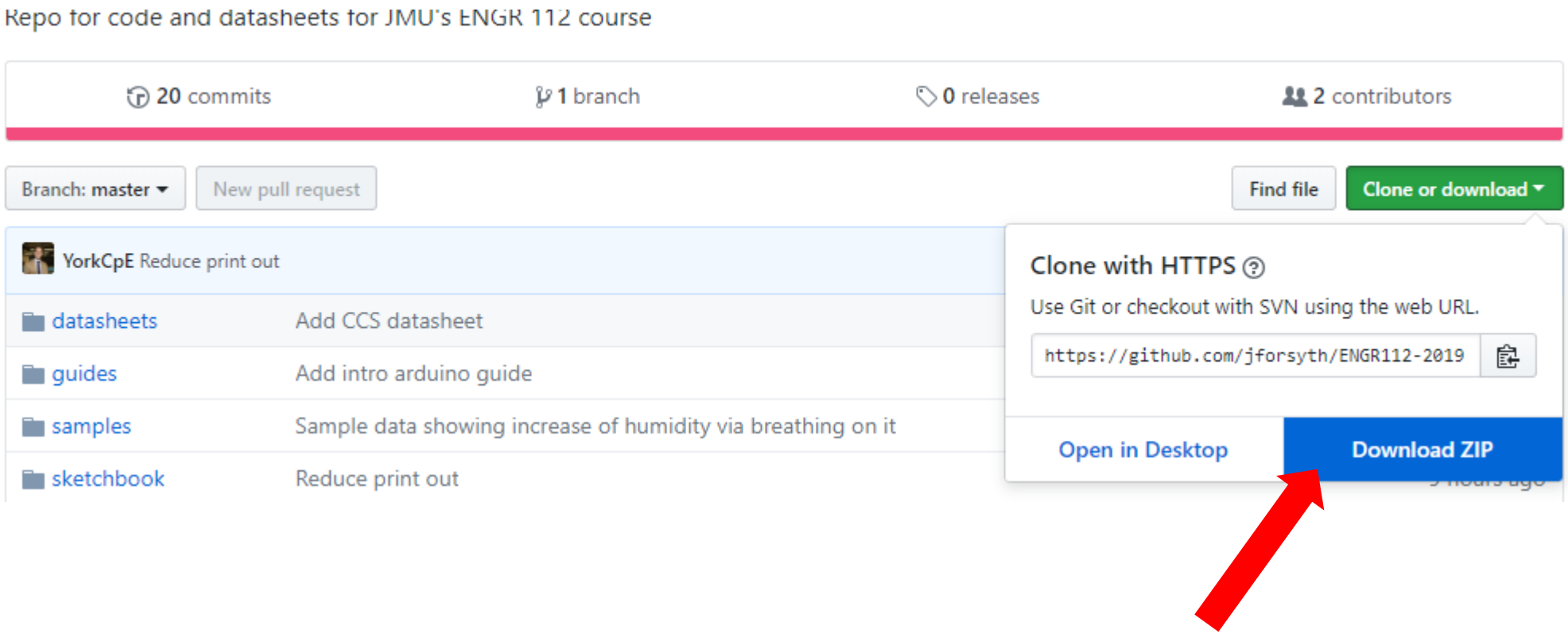
- You must point your Arduino IDE towards the correct sketchbook for all items to work.

- Change to a new sketchbook inside “Preferences” in the IDE

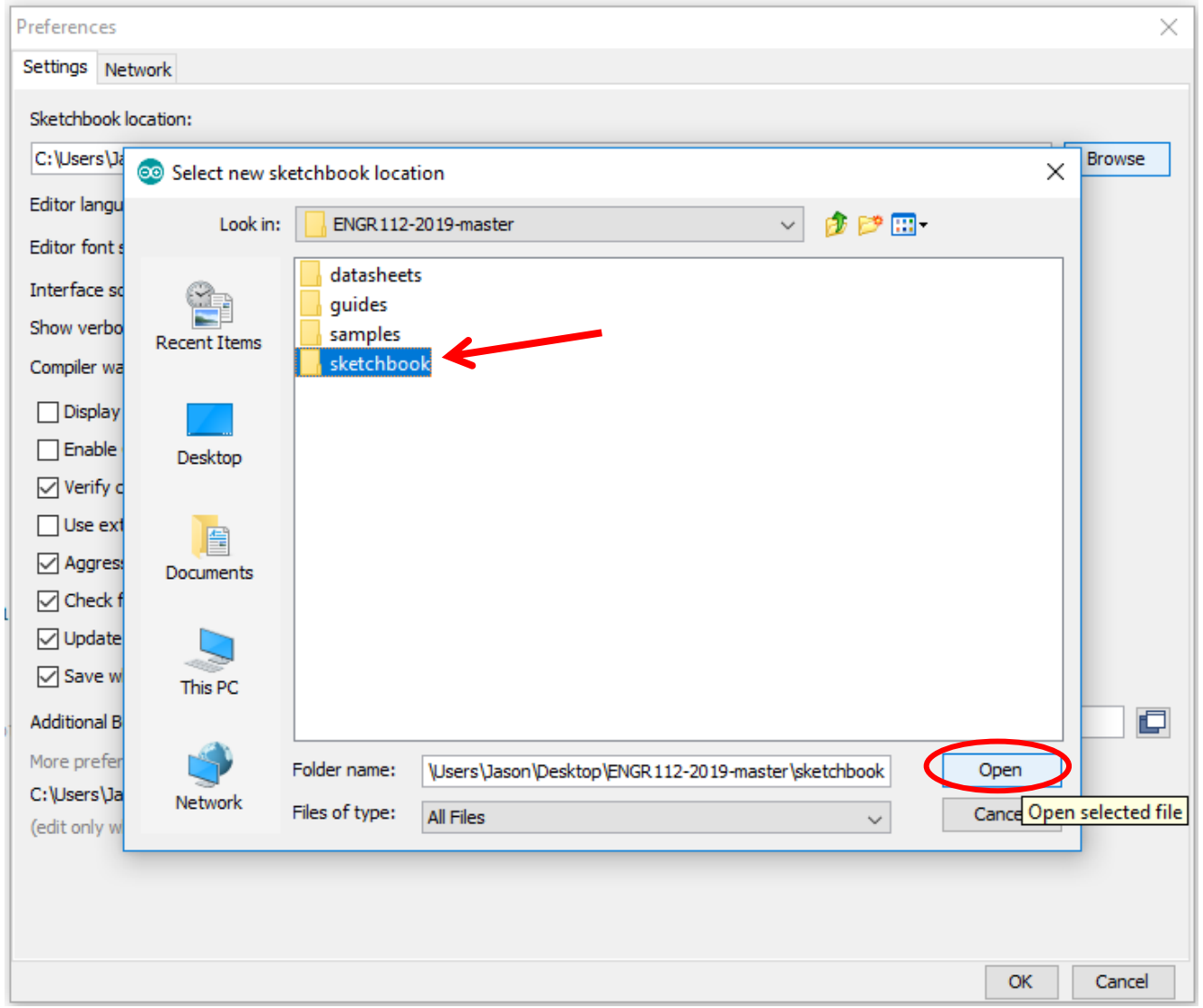


Downloading the Course Sketchbook

- Go to: https://github.com/jforsyth/ENGR112-2019 and click “Clone or Download” to Download as Zip.



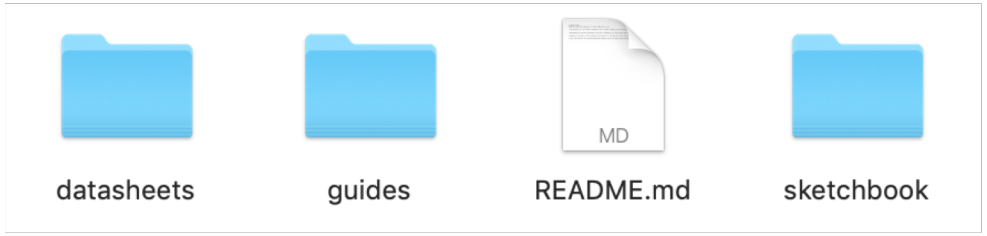
Setting Your Sketchbook



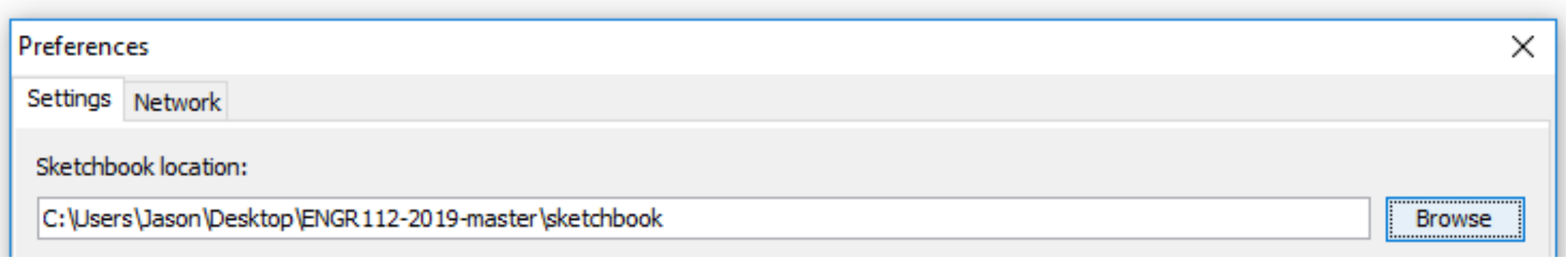
Unzip the “ENGR112-2019-master” file. Mac/OSX may do this for your.

Inside Arduino IDE preferences find your sketchbook, click on it, and then select OPEN.

Make sure you download the file to a known location.



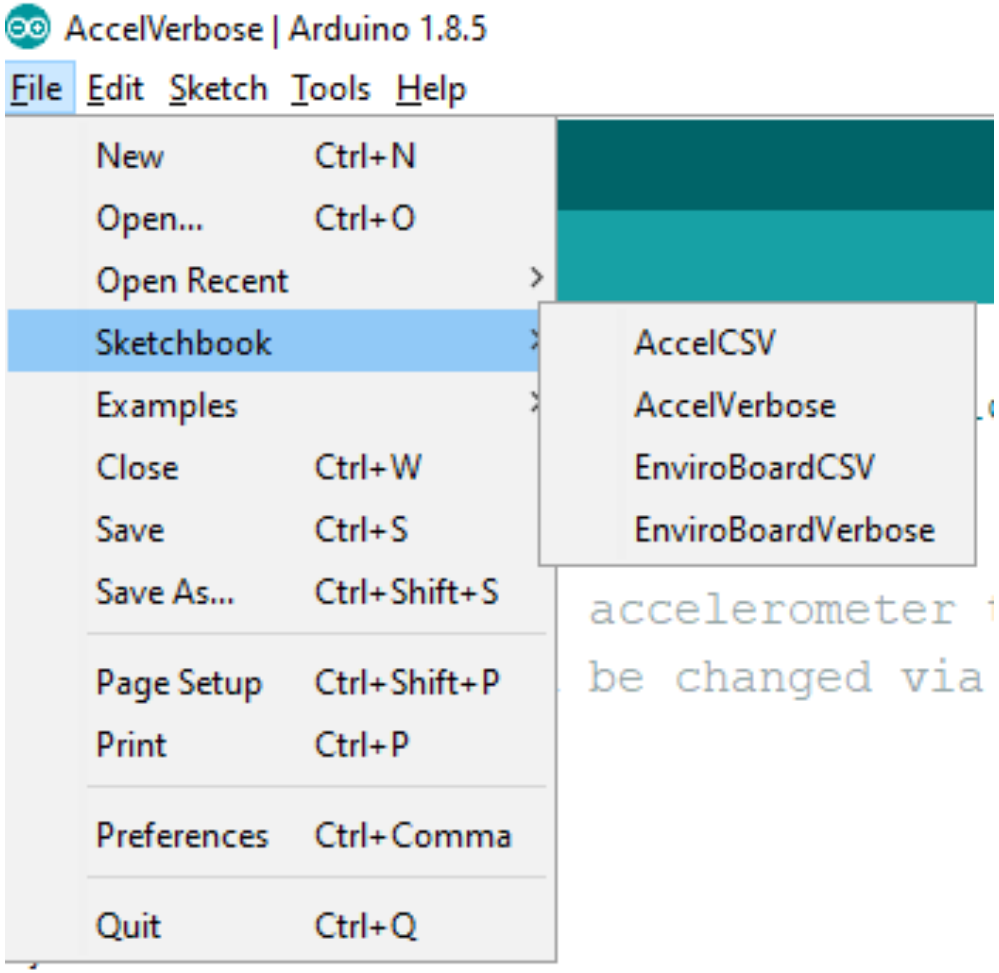
Properly Configured Sketchbook



- The preferences window should now point to your sketchbook location. Make sure it points to a FOLDER and not a ZIP file.

- If it appears as a ZIP, go back to where you downloaded the file, and UNZIP. Mac/OSX may automatically perform this operation.

Sketchbook with Instructor Code

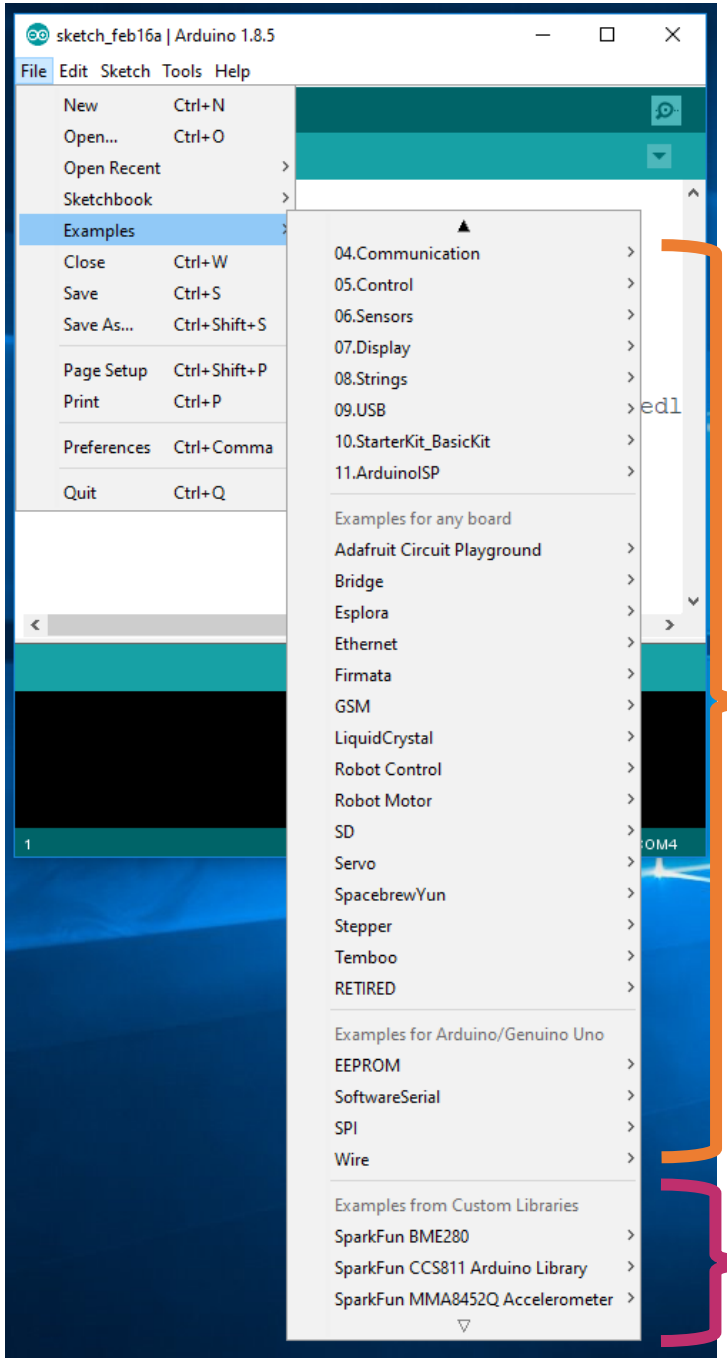


Once configured you will now see instructor provided programs.

Any additional programs you create and save will automatically go into your sketchbook.

Instructor code will be hosted on GitHub and you can copy/paste new programs and/or correct any errors.

Still have access to “Examples” provided by Arduino and Sparkfun

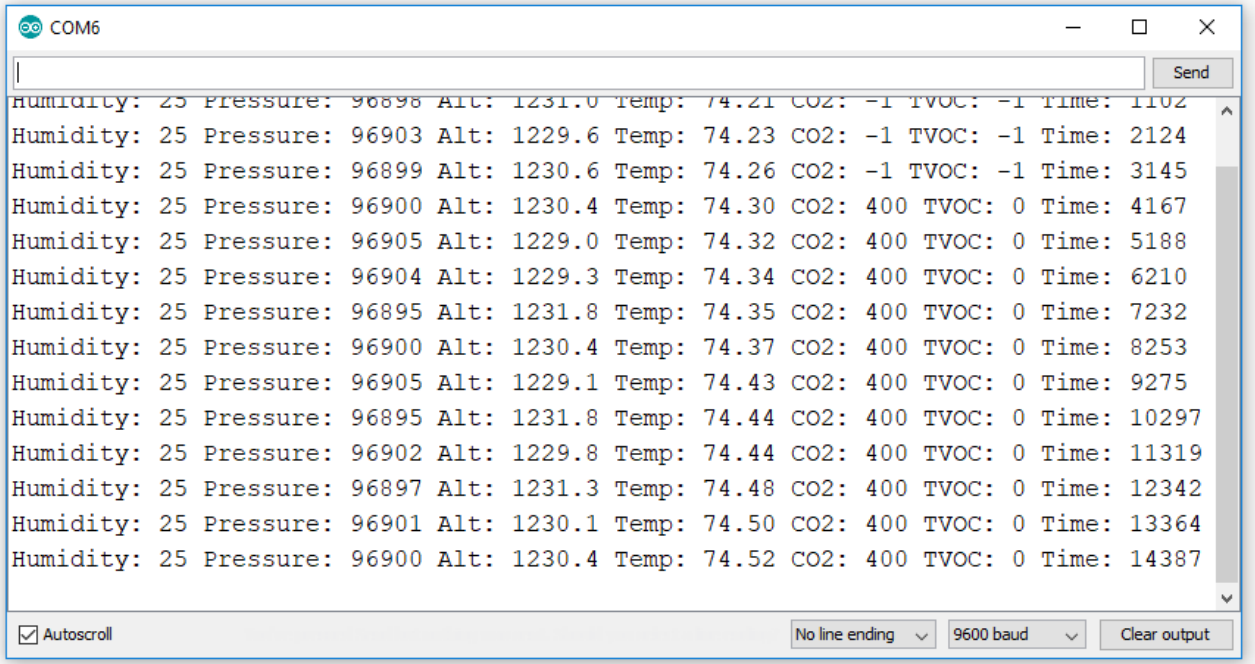


Arduino provided Examples

Sparkfun Examples for sensors

Step 2: Uploading Program and Collecting Data

```ino
void loop() {
  // put your main code here, to run repeatedly:

  ////////////////////// Get Data from the BME Board //////////////////////
  float humidity = bmeSensor.readFloatHumidity();
  float pressure = bmeSensor.readFloatPressure();
  float altitude = bmeSensor.readFloatAltitudeFeet();
  float temp = bmeSensor.readTempF();

  Serial.print("Humidity: ");
  Serial.print(humidity, 0); //print with no decimal places

  Serial.print(" Pressure: ");
  Serial.print(pressure, 0); //print with no decimal places

  Serial.print(" Alt: ");
  Serial.print(altitude, 1); //print with one decimal place

  Serial.print(" Temp: ");
  Serial.print(temp, 2); // print with two decimal places

  ////////////////////// Get Data from the CCS Board //////////////////////
```

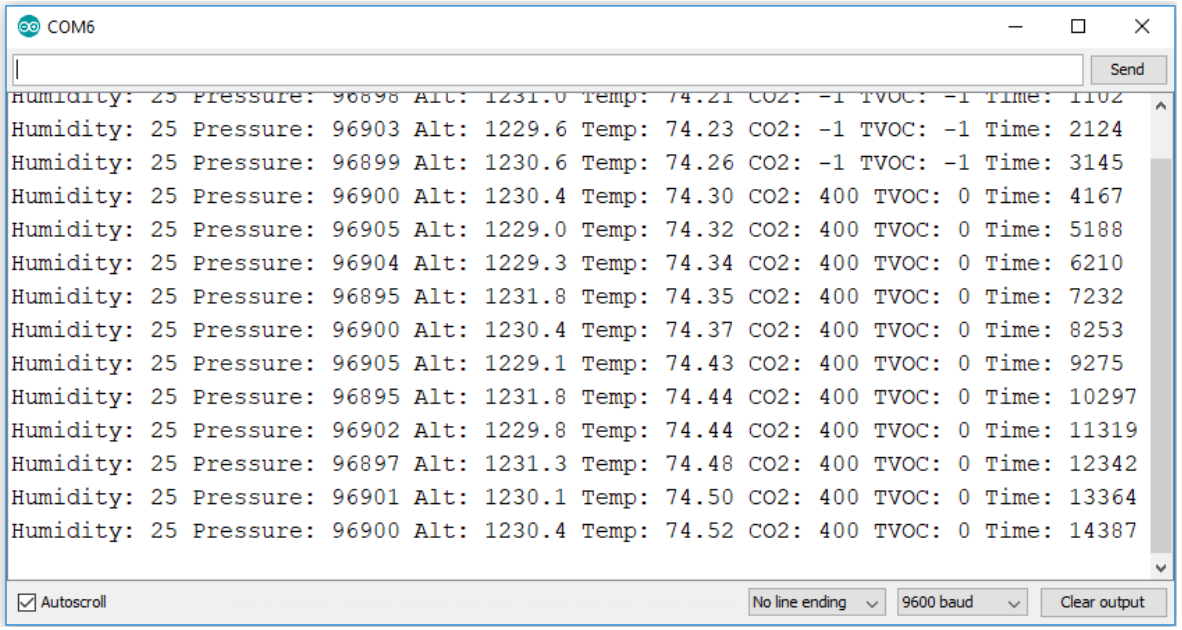


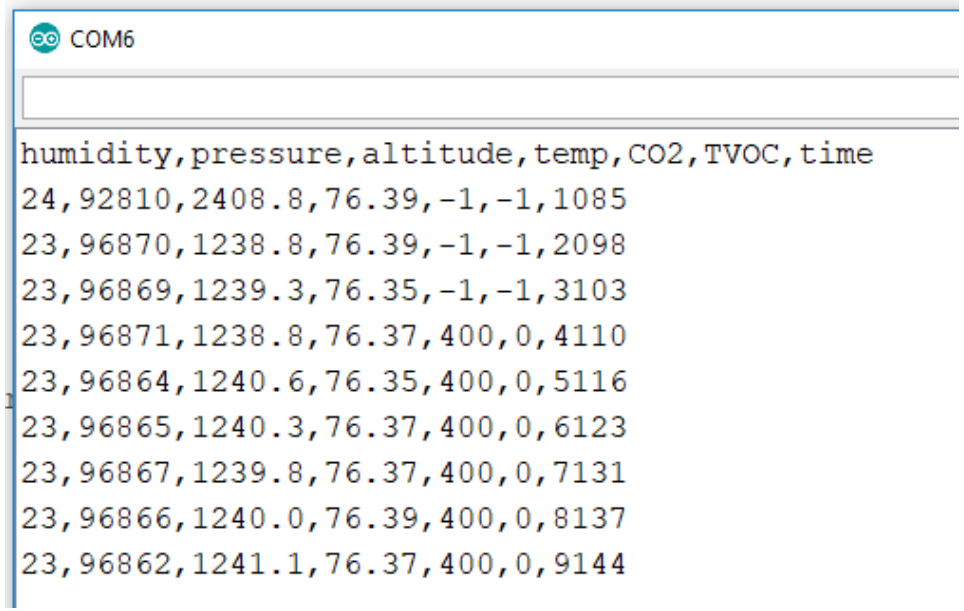
1. Select a program from the Sketchbook.
2. Use the Upload button  to compile and send the program to your Arduino
3. Use the Serial Monitor  to view the output

Verbose and CSV Programs

- There are two versions of each program included: Verbose and CSV
- Verbose program have more “printed” information and are for you (the human) to read.
- CSV files produce a comma-separated value file that has only data printed. These are meant for importing into MATLAB and Excel.

Comparing Verbose and CSV Output



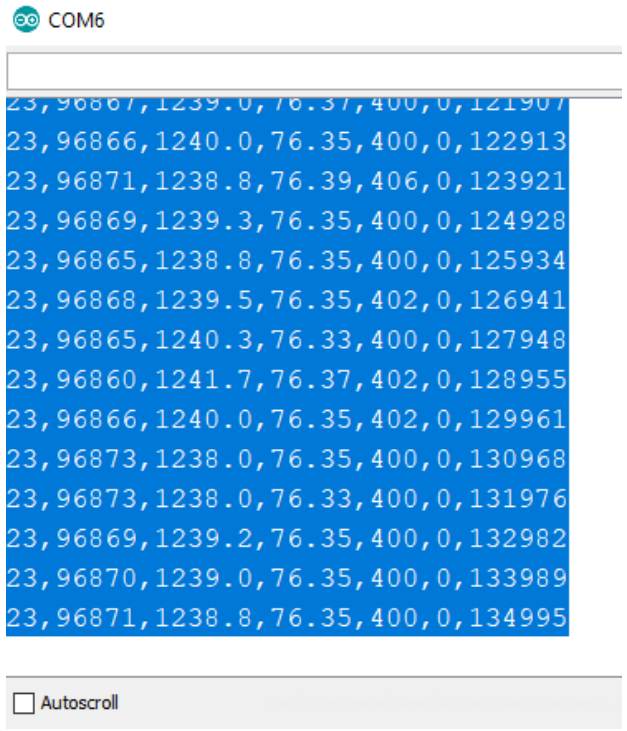


Both provide the same information just in different formats.

Step 3: Importing and Displaying Data in MATLAB

Use the CSV Format When Importing

- Un-check “autoscoll” in the serial monitor and then copy and paste data from the window into a text file. Use keyboard shortcuts: CTRL-A (select all), CTRL-C (copy), CTRL-V (paste)



Copy Into a Text Editor

- Copy your data/text into Text Editor (WORD, Notepad…etc.).
- Make sure the top line is the “header” information of *humidity, temp, pressure…* This should be the only line containing words and not data
- Then save the file as a **TEXT FILE** with a **.txt** extension. (Not Word Doc, XML, ODT… just a simple text file)

Header

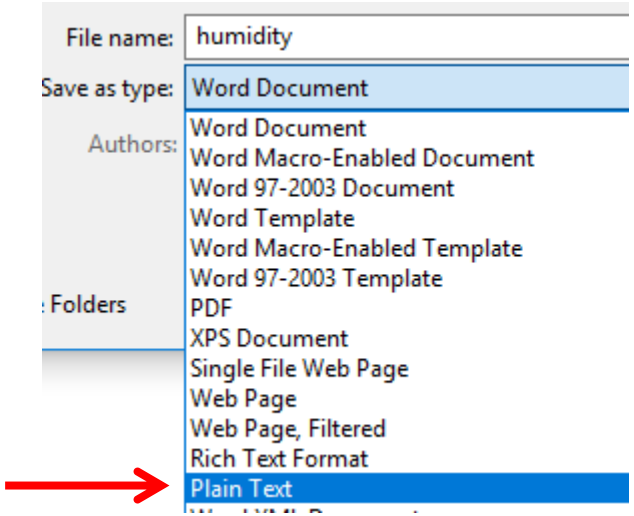
humidity,pressure,altitude,temp,CO2,TVOC,time

24,92810,2408.8,76.39,-1,-1,1085

Data

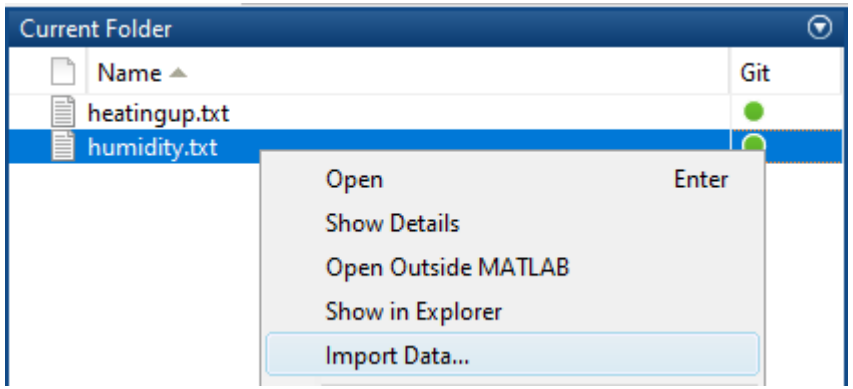
23,96870,1238.8,76.39,-1,-1,2098

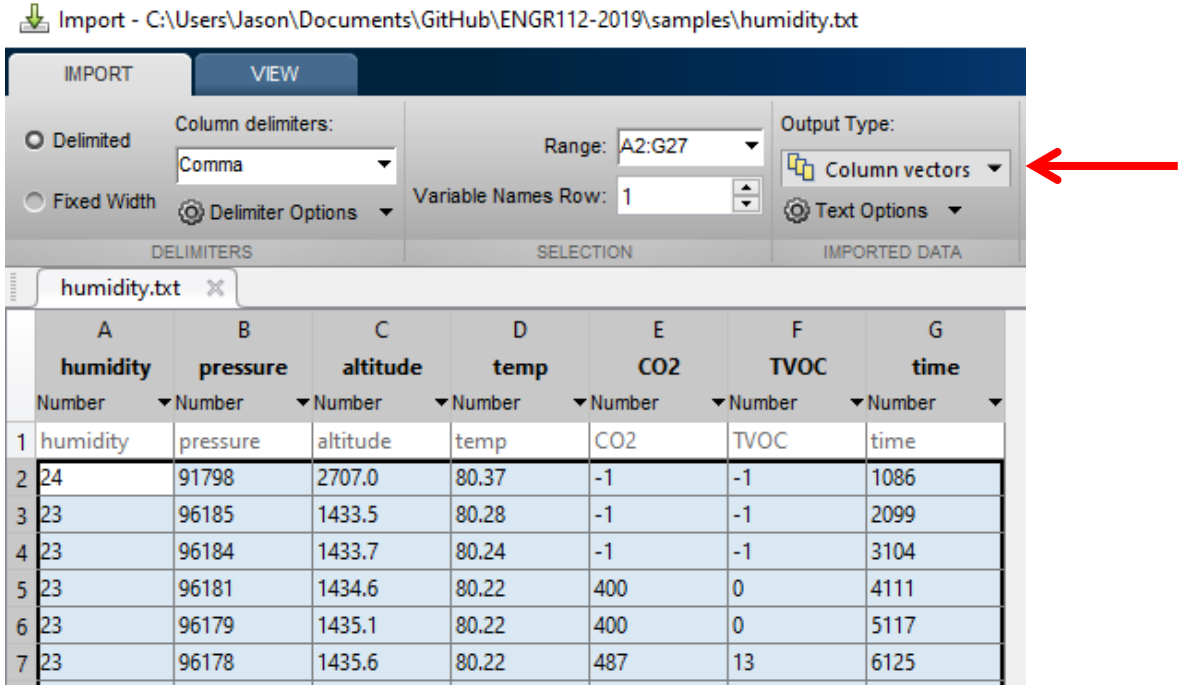
23,96869,1239.3,76.35,-1,-1,3103



Import into MATLAB

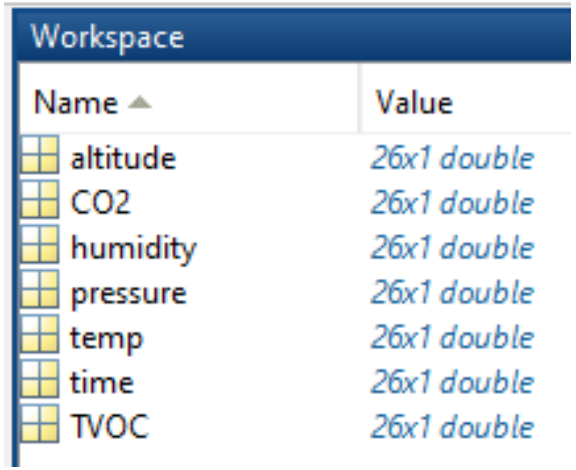
- Launch MATLAB and navigate to where your file is located.
- Right click on the file and select “Import Data”.
- Then select each column and import as column vectors

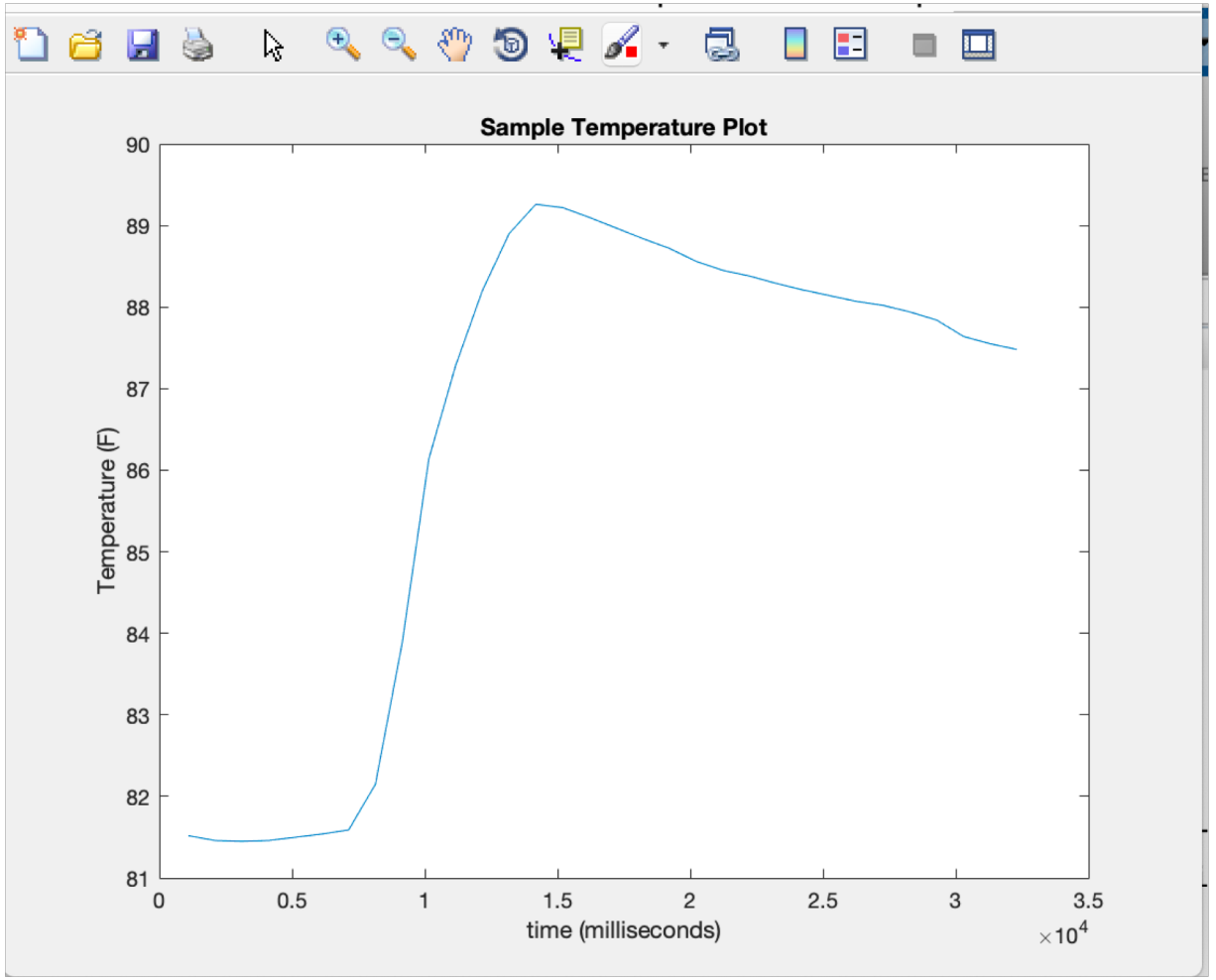




Your Data in MATLAB

- Now your data is available in the MATLAB workspace with each variables properly named ☺
- You can easily create plots of your data and begin analysis





Sample plot of me holding then releasing the temp sensor over ~3.5 seconds

Activities

- Connect the Environment and Accelerometer boards. Run the sample programs to ensure your setup is operational.

- For each sensor type consider what tests you can run to make sure the data is “valid”.
  - If you turn the accel on its side what acceleration should you expect? What happens if you bounce it?
  - What is an acceptable altitude for Harrisonburg? How close are you?
  - How could you test values for TVOC? Is there a location on campus that may have more?

- Practice some of these tests with imported data. Predict what you should expect and see if that matches up.