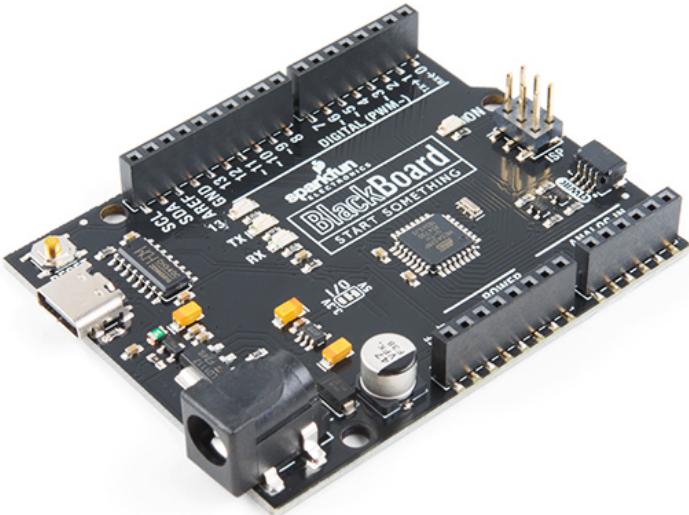


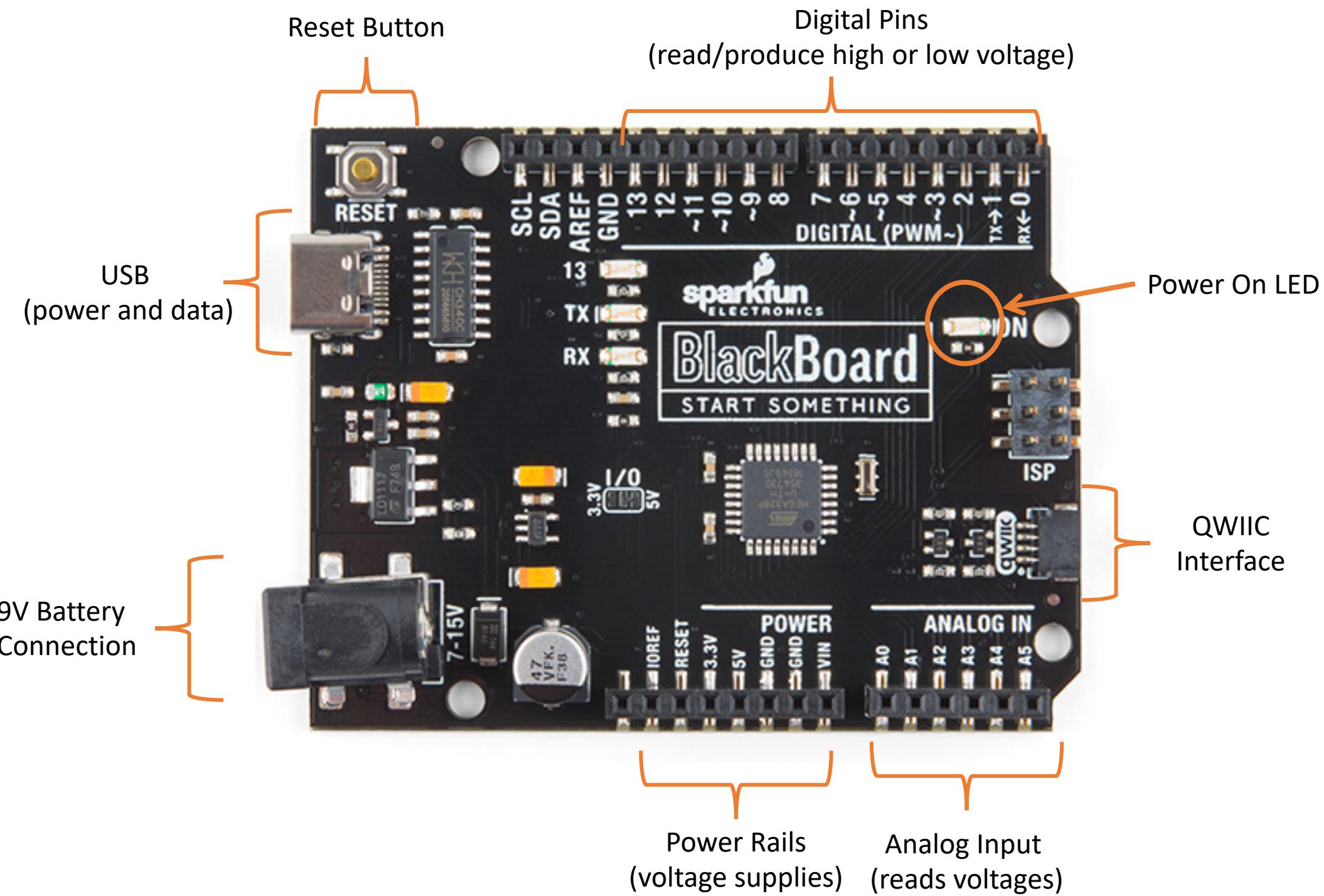
Getting Started with Arduino

Running Programs and Collecting Data

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.

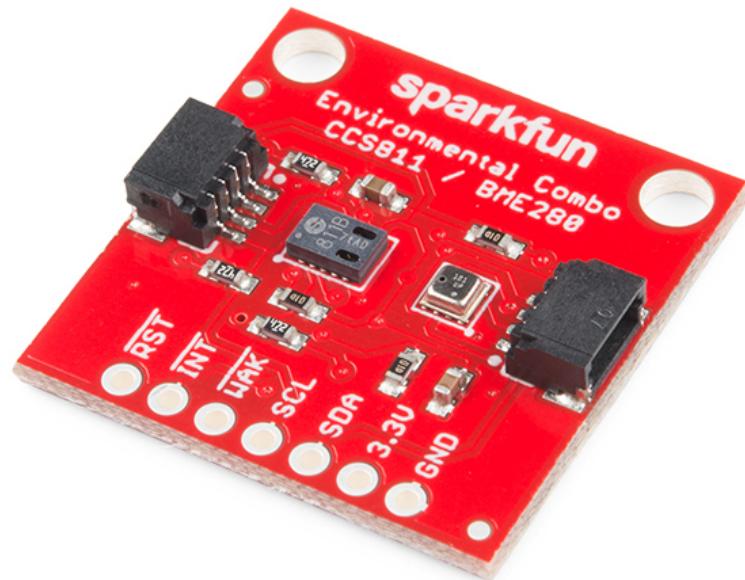


Glossary of Terms

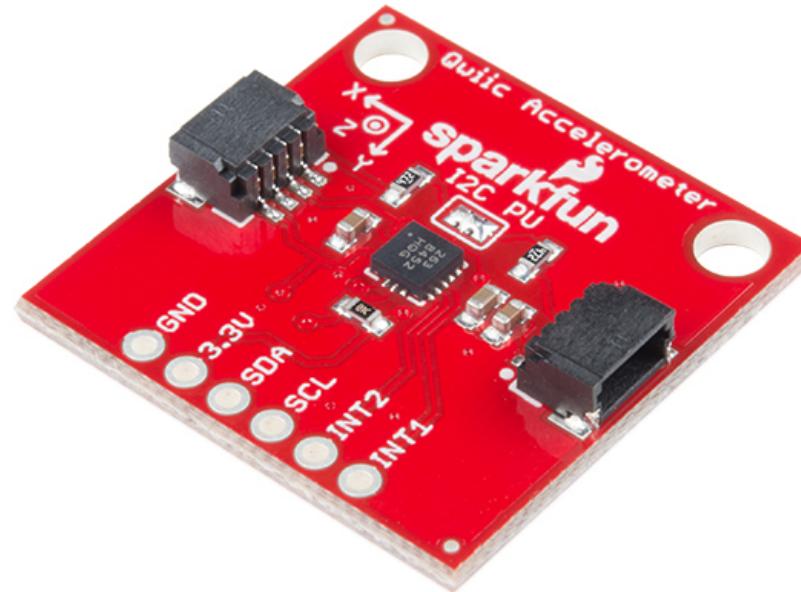
- **Arduino/Blackboard:** for the purpose of this class these words are interchangeable. Technically a Blackboard is an Arduino-compatible clone produced by SparkFun.
- **Digital:** signals or information that exist in discrete states (ON/OFF, HIGH/LOW, 5 volts / 0 volts).
- **Analog:** signals or information 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



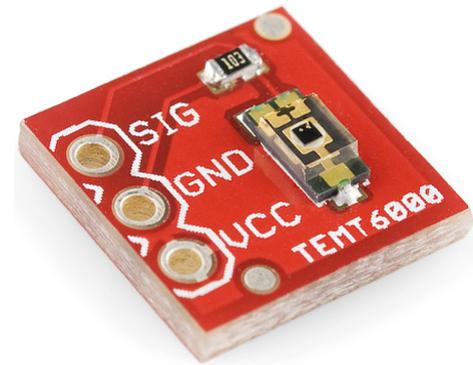
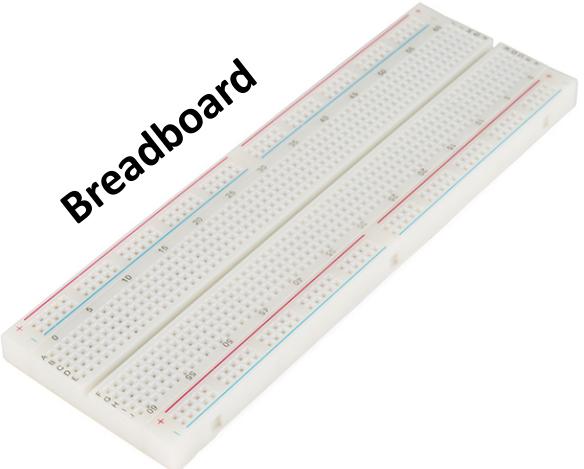
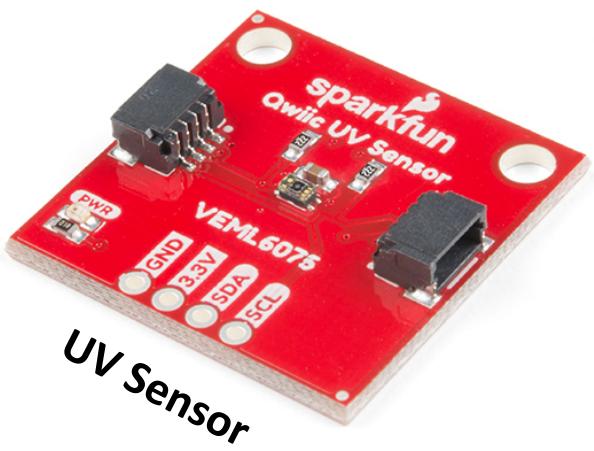
Environmental Combo board housing two sensors:
CCS811 (CO₂ and TVOC) and **BME280** (humidity,
pressure, altitude, and temperature)



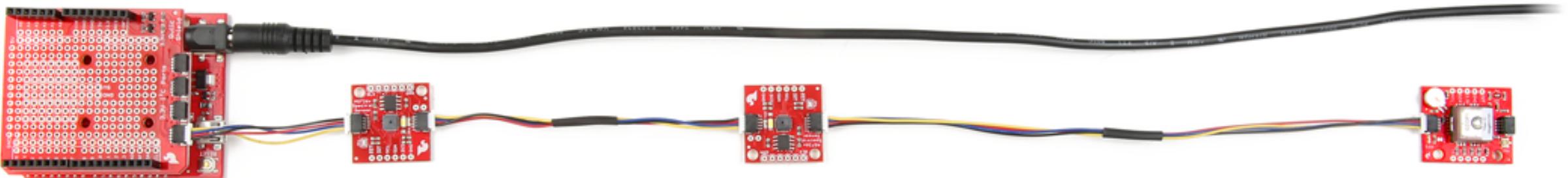
Board contains **MMA8452** Accelerometer that
provides acceleration in X, Y, and Z axis. Axis are
printed on the board silkscreen.

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

Additional Parts Available as Necessary...



Connecting QWIIC Sensors

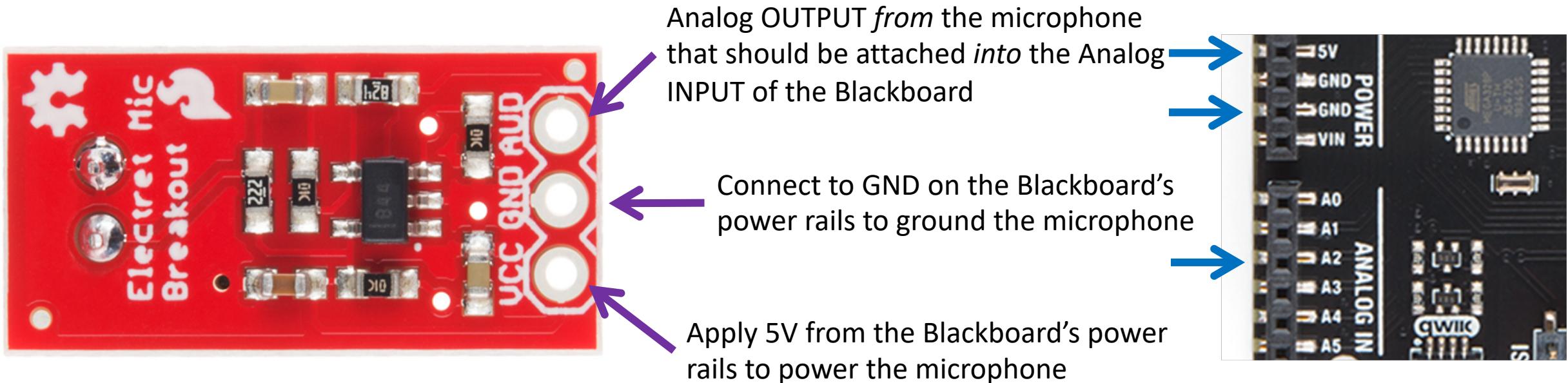


Devices use the “digital” I2C interface to communicate. Simply chain/connect QWIIC compatible sensors with a QWIIC cable.

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.

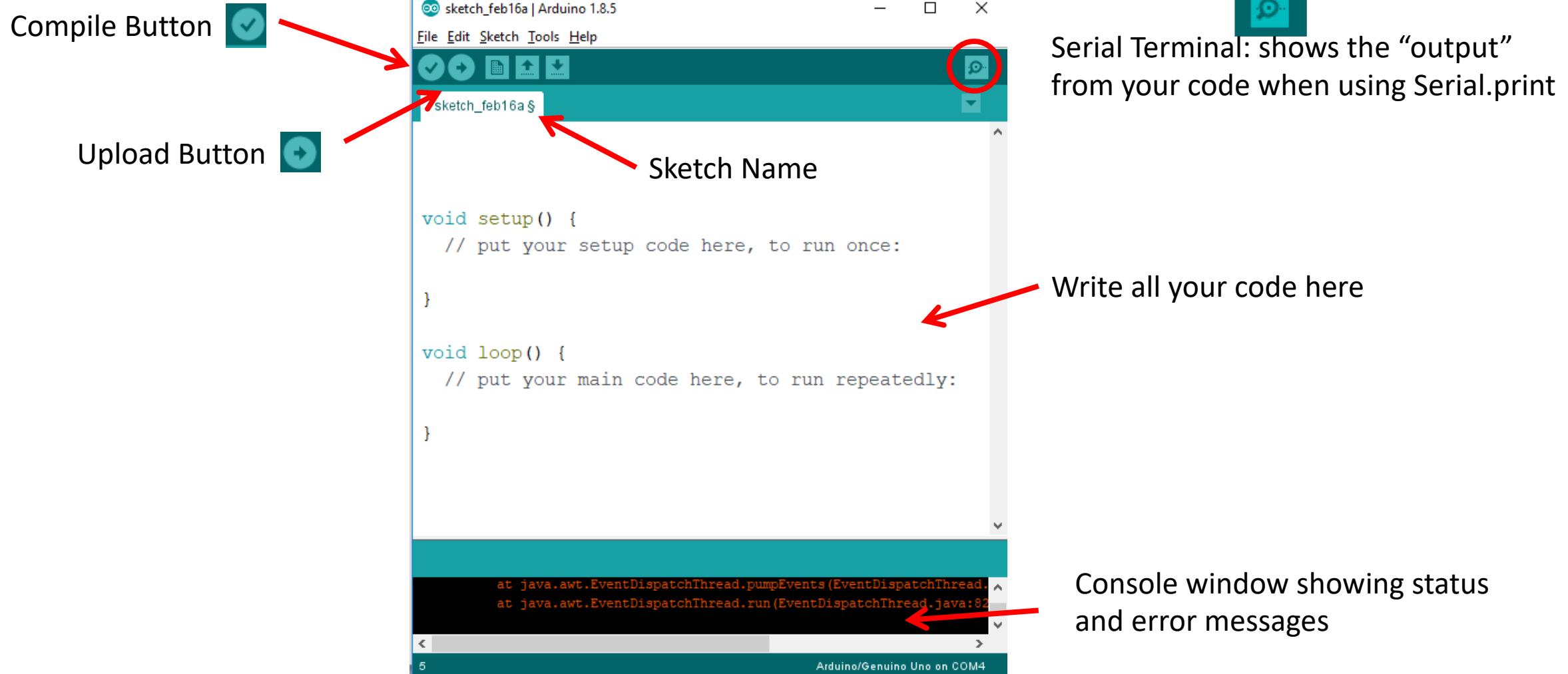


Warnings on Analog Sensors

- 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



Anatomy of an Arduino Program



The screenshot shows the Arduino IDE interface. The title bar reads "sketch_feb16a | Arduino 1.8.5". The menu bar includes "File", "Edit", "Sketch", "Tools", and "Help". Below the menu is a toolbar with icons for save, upload, and other functions. The main window displays the code for "sketch_feb16a". The code consists of two functions: `void setup()` and `void loop()`.

```
void setup() {
  // put your setup code here, to run once:
}

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

- 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);
}
```

Turn the LED ON by making
a pin HIGH voltage

```
// 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 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.getAcceleration();  
  
float yAccel = accel.getAcceleration();  
  
float zAccel = accel.getAcceleration();  
  
  
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.

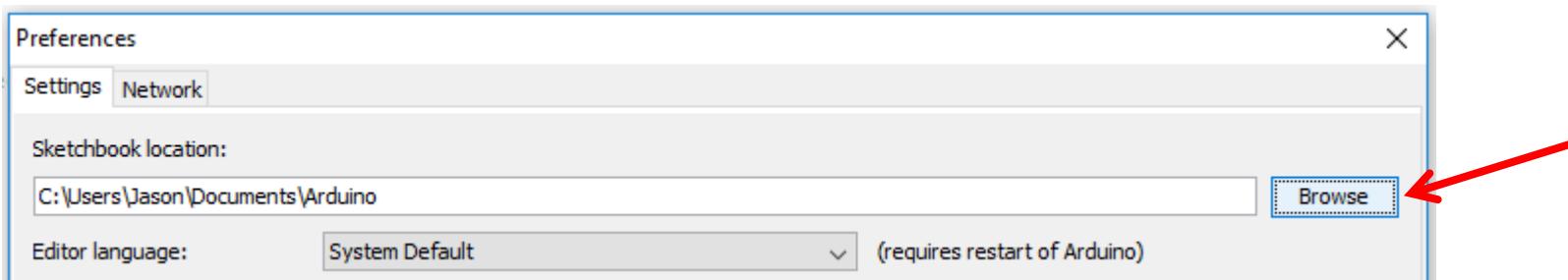
Wait, is this a programming class?!

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.

Repo for code and datasheets for JMU's ENGR 112 course

20 commits 1 branch 0 releases 2 contributors

Branch: master ▾ New pull request Find file Clone or download ▾

 YorkCpE Reduce print out

 datasheets Add CCS datasheet

 guides Add intro arduino guide

 samples Sample data showing increase of humidity via breathing on it

 sketchbook Reduce print out

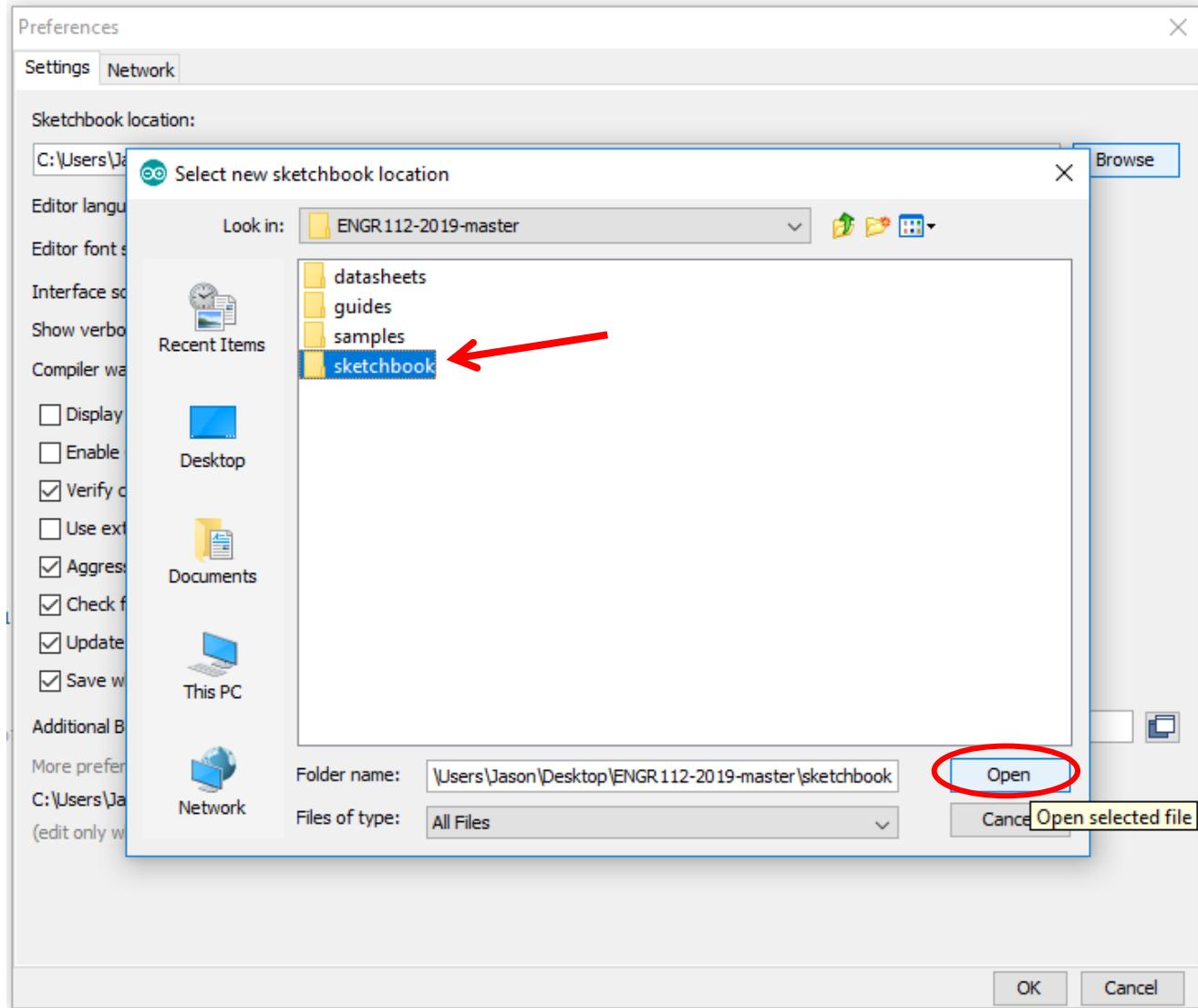
Clone with HTTPS ⓘ
Use Git or checkout with SVN using the web URL.
<https://github.com/jforsyth/ENGR112-2019> 

[Open in Desktop](#) [Download ZIP](#)

3 hours ago



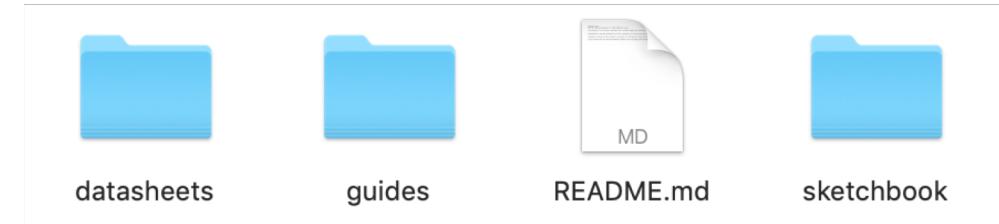
Setting Your Sketchbook



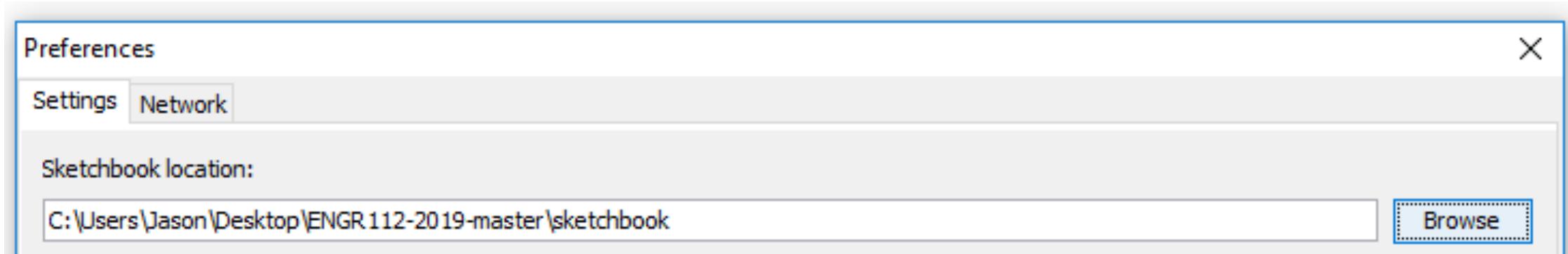
Unzip the “ENGR112-2019-master” file.
Mac/OSX may do this for you.

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

Make sure you download the file to a
known location.

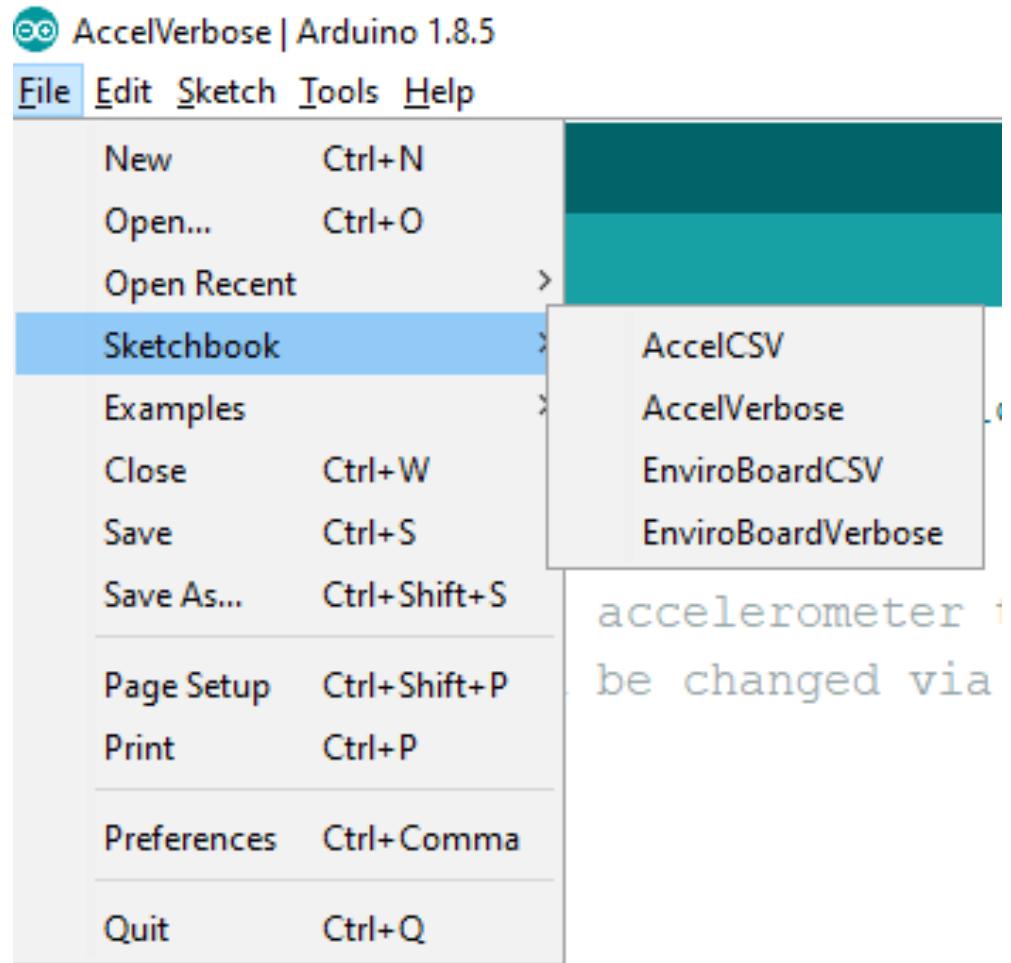


Properly Configured Sketchbook



- 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

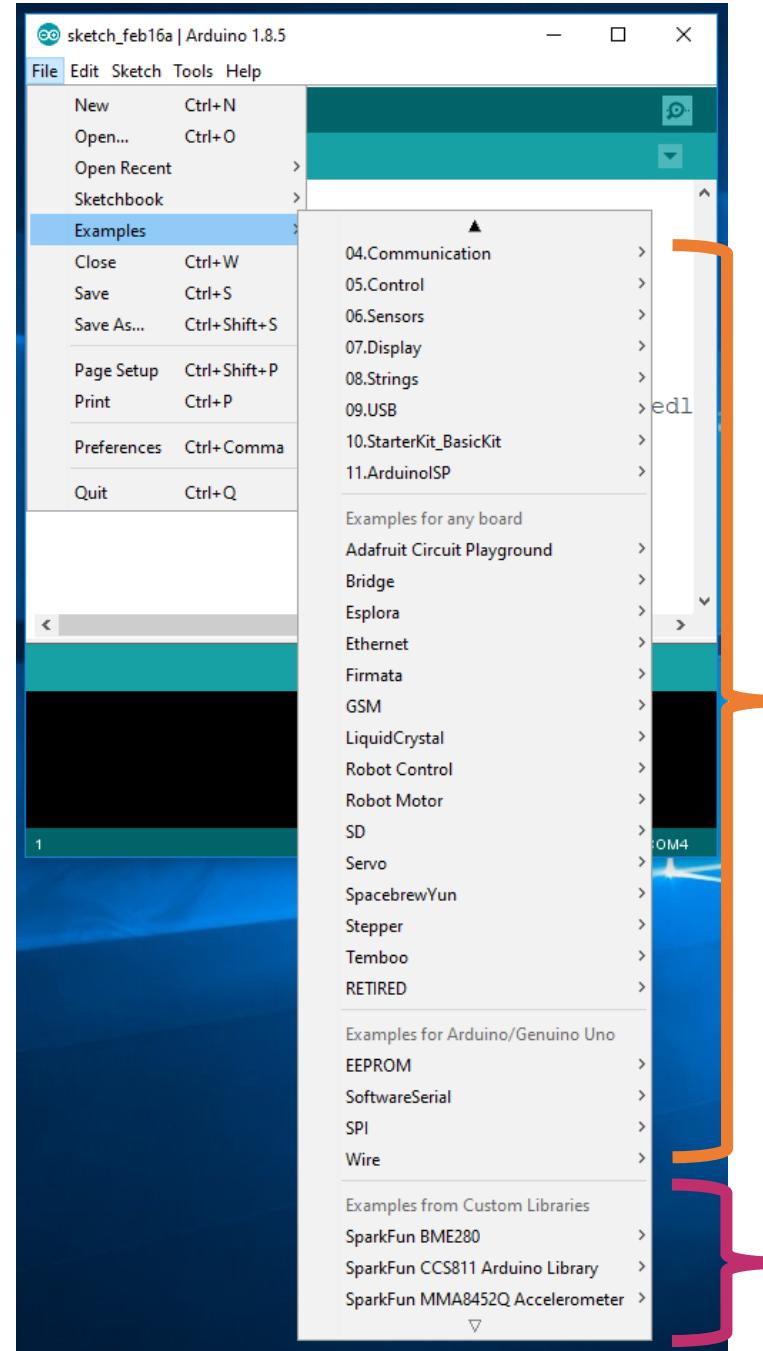


Once configured you will now see instructor provided programs.

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**

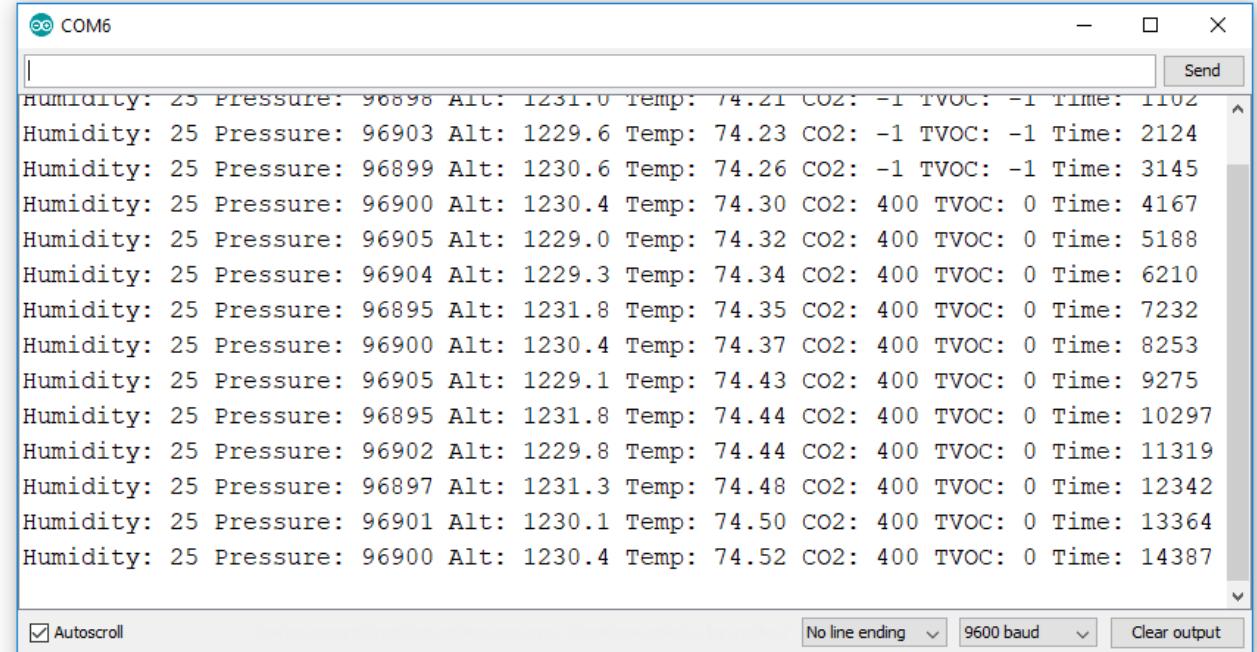


**Arduino provided
Examples**

**Sparkfun Examples for
sensors**

Step 2: Uploading Program and Collecting Data

```
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 ///////////////////
```



The screenshot shows the Arduino Serial Monitor window titled "COM6". The window displays a series of sensor readings in a text-based log. Each entry consists of four pieces of data: Humidity, Pressure, Altitude, and Temperature. The readings are as follows:

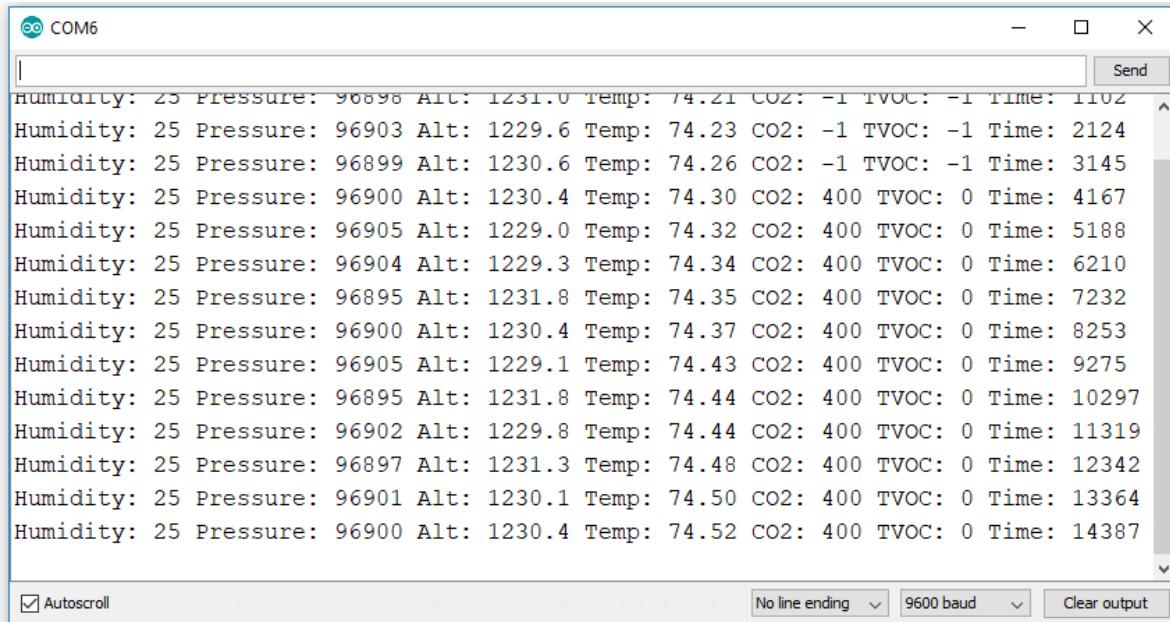
Humidity	Pressure	Altitude	Temperature
25	96898	1231.0	74.21
25	96903	1229.6	74.23
25	96899	1230.6	74.26
25	96900	1230.4	74.30
25	96905	1229.0	74.32
25	96904	1229.3	74.34
25	96895	1231.8	74.35
25	96900	1230.4	74.37
25	96905	1229.1	74.43
25	96895	1231.8	74.44
25	96902	1229.8	74.44
25	96897	1231.3	74.48
25	96901	1230.1	74.50
25	96900	1230.4	74.52

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

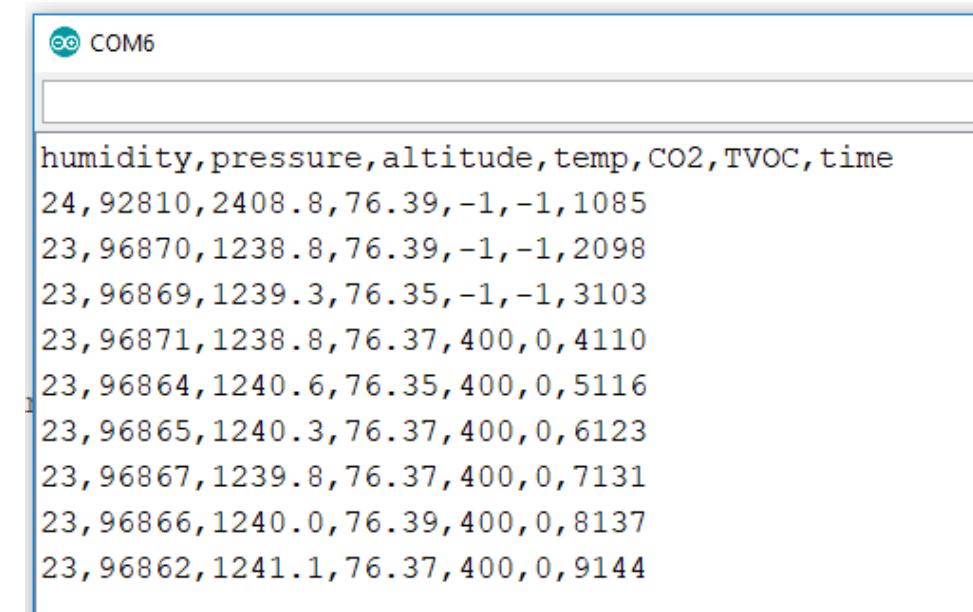
- There are two versions of each program included: Verbose and CSV
- Verbose programs have more “printed” information and are for you (the human) to read.
- CSV files produce a comma-separated value file that has only data printed. These are meant for importing into MATLAB and Excel.

Comparing Verbose and CSV Output



A screenshot of a Windows-style serial terminal window titled "COM6". The window displays a series of sensor readings in a verbose, human-readable format. Each line of output contains seven pieces of data: Humidity, Pressure, Altitude, Temperature, CO2 concentration, TVOC concentration, and timestamp. The data is repeated multiple times. At the bottom of the window, there are several control buttons: "Send", "No line ending", "9600 baud", and "Clear output". A checkbox labeled "Autoscroll" is checked.

```
Humidity: 25 Pressure: 96896 Alt: 1231.0 Temp: 74.21 CO2: -1 TVOC: -1 Time: 1102
Humidity: 25 Pressure: 96903 Alt: 1229.6 Temp: 74.23 CO2: -1 TVOC: -1 Time: 2124
Humidity: 25 Pressure: 96899 Alt: 1230.6 Temp: 74.26 CO2: -1 TVOC: -1 Time: 3145
Humidity: 25 Pressure: 96900 Alt: 1230.4 Temp: 74.30 CO2: 400 TVOC: 0 Time: 4167
Humidity: 25 Pressure: 96905 Alt: 1229.0 Temp: 74.32 CO2: 400 TVOC: 0 Time: 5188
Humidity: 25 Pressure: 96904 Alt: 1229.3 Temp: 74.34 CO2: 400 TVOC: 0 Time: 6210
Humidity: 25 Pressure: 96895 Alt: 1231.8 Temp: 74.35 CO2: 400 TVOC: 0 Time: 7232
Humidity: 25 Pressure: 96900 Alt: 1230.4 Temp: 74.37 CO2: 400 TVOC: 0 Time: 8253
Humidity: 25 Pressure: 96905 Alt: 1229.1 Temp: 74.43 CO2: 400 TVOC: 0 Time: 9275
Humidity: 25 Pressure: 96895 Alt: 1231.8 Temp: 74.44 CO2: 400 TVOC: 0 Time: 10297
Humidity: 25 Pressure: 96902 Alt: 1229.8 Temp: 74.44 CO2: 400 TVOC: 0 Time: 11319
Humidity: 25 Pressure: 96897 Alt: 1231.3 Temp: 74.48 CO2: 400 TVOC: 0 Time: 12342
Humidity: 25 Pressure: 96901 Alt: 1230.1 Temp: 74.50 CO2: 400 TVOC: 0 Time: 13364
Humidity: 25 Pressure: 96900 Alt: 1230.4 Temp: 74.52 CO2: 400 TVOC: 0 Time: 14387
```



A screenshot of a Windows-style serial terminal window titled "COM6". The window displays the same sensor data as the first window, but in a CSV (Comma-Separated Values) format. Each line of output consists of a single row of data with commas separating the individual measurements. The data is repeated multiple times. At the bottom of the window, there are several control buttons: "Send", "No line ending", "9600 baud", and "Clear output".

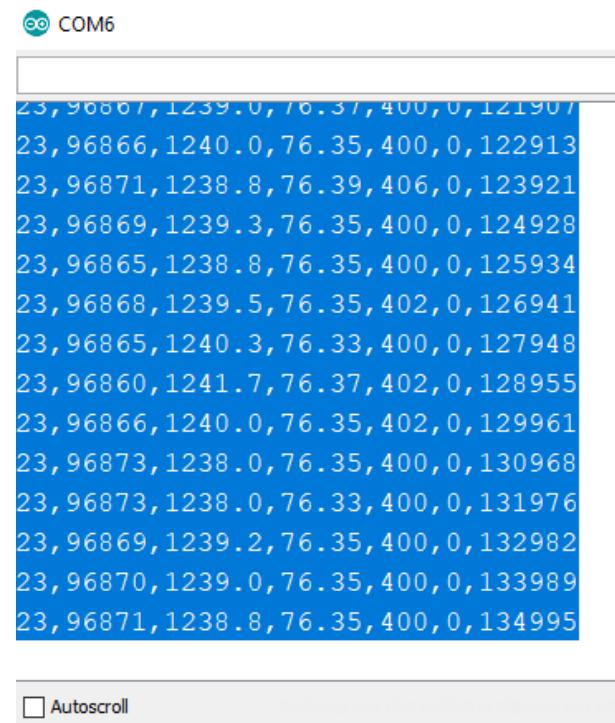
```
humidity,pressure,altitude,temp,CO2,TVOC,time
24,92810,2408.8,76.39,-1,-1,1085
23,96870,1238.8,76.39,-1,-1,2098
23,96869,1239.3,76.35,-1,-1,3103
23,96871,1238.8,76.37,400,0,4110
23,96864,1240.6,76.35,400,0,5116
23,96865,1240.3,76.37,400,0,6123
23,96867,1239.8,76.37,400,0,7131
23,96866,1240.0,76.39,400,0,8137
23,96862,1241.1,76.37,400,0,9144
```

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 “autoscroll” 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

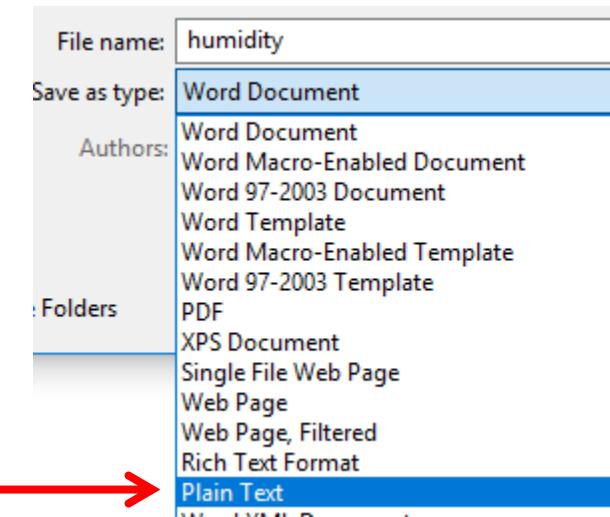
- Copy your data/text into Text Editor (WORD, Notepad...etc.).
- Make sure the top line is the “header” information of *humidity, temp, pressure...* This should be the only line containing words and not data
- Then save the file as a **TEXT FILE** with a **.txt** extension. (Not Word Doc, XML, ODT...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

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

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

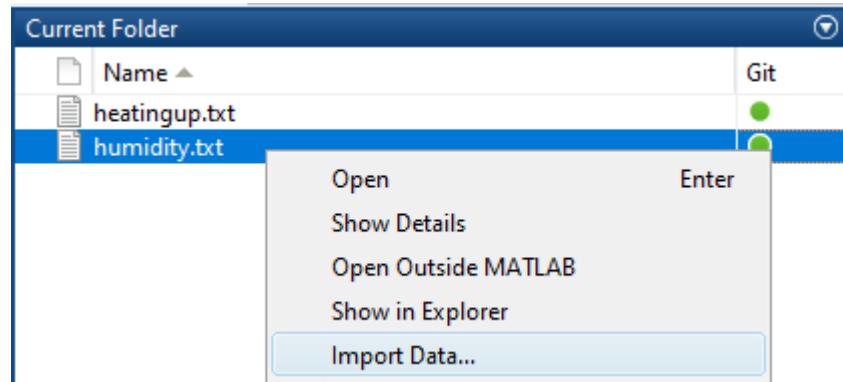
Data → 23,96870,1238.8,76.39,-1,-1,2098

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



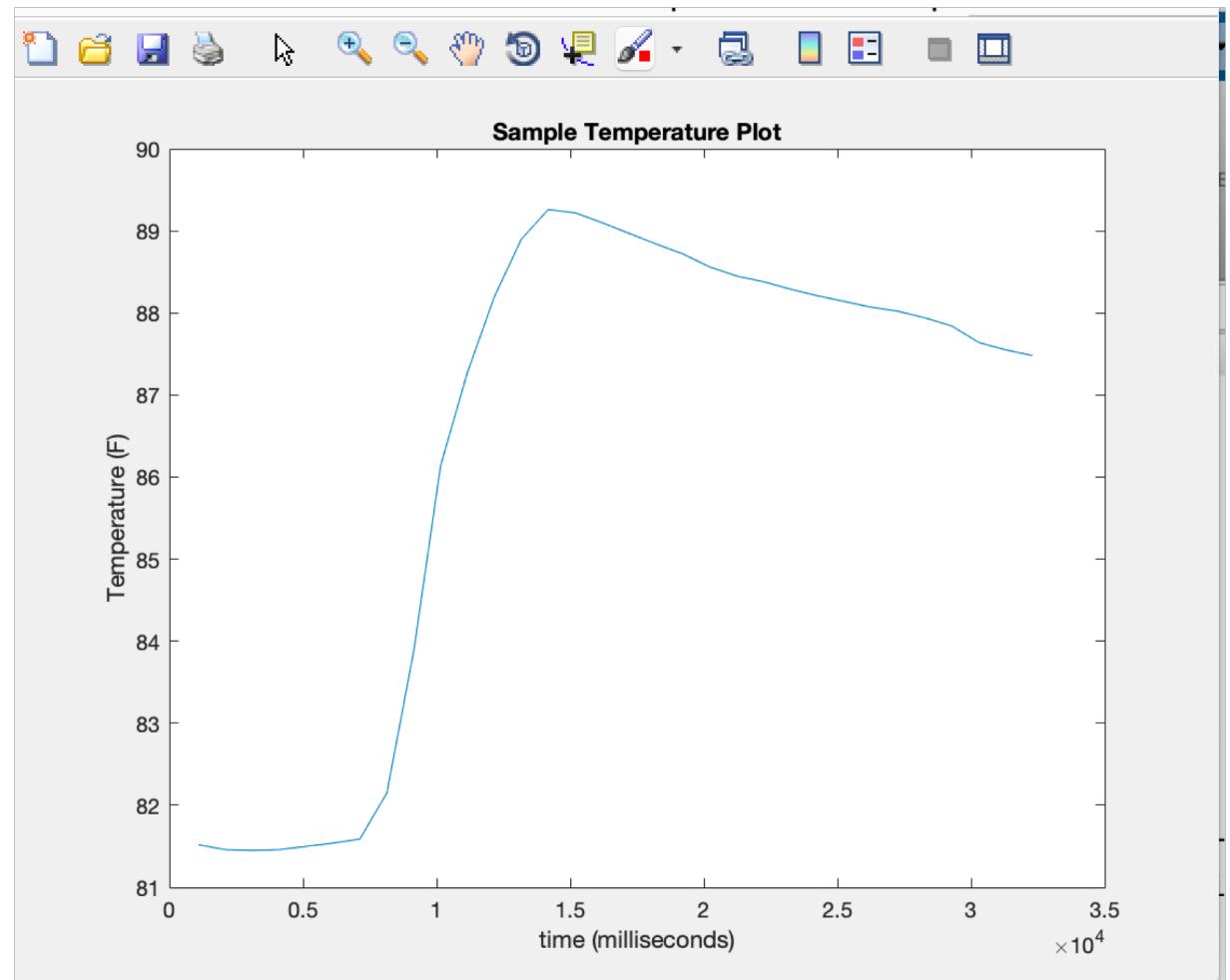
A screenshot of the MATLAB Import Data dialog box for the file 'humidity.txt'. The 'Column vectors' option under 'Output Type' is selected, indicated by a red arrow. The dialog shows the file path 'Import - C:\Users\Jason\Documents\GitHub\ENGR112-2019\samples\humidity.txt', the delimiter 'Comma', and the variable names row '1'. The imported data table has columns labeled 'humidity', 'pressure', 'altitude', 'temp', 'CO2', 'TVOC', and 'time'.

	A	B	C	D	E	F	G
	humidity	pressure	altitude	temp	CO2	TVOC	time
Number	Number	Number	Number	Number	Number	Number	Number
1	humidity	pressure	altitude	temp	CO2	TVOC	time
2	24	91798	2707.0	80.37	-1	-1	1086
3	23	96185	1433.5	80.28	-1	-1	2099
4	23	96184	1433.7	80.24	-1	-1	3104
5	23	96181	1434.6	80.22	400	0	4111
6	23	96179	1435.1	80.22	400	0	5117
7	23	96178	1435.6	80.22	487	13	6125

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

Workspace	
Name	Value
altitude	26x1 double
CO2	26x1 double
humidity	26x1 double
pressure	26x1 double
temp	26x1 double
time	26x1 double
TVOC	26x1 double



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.