Displaying Data

Introduction

In the “Calibrating a Sensor” activity, you displayed data using the Serial Monitor. In this activity, you will experiment with creating data tables that are easier to read, and use the Serial Plotter to create graphs.

Objectives

- Understand formatting output to the Serial Monitor
- Understand formatting output to the Serial Plotter (computer only)

Materials

SparkFun® RedBoard (or equivalent) with USB cable and power supply
Vernier Analog Protoboard Adapter or Vernier Interface Shield
Vernier Gas Pressure Sensor
Computer or Chromebook™ with Arduino® software

Procedure

Previous activities explored basic information about the Arduino, programming, and connecting Vernier analog sensors. Let’s revisit the architecture of the sketch.

1. Open the program you saved at the end of the previous activity, “Calibrating a Sensor”.
2. Save it with a unique name so that you don’t inadvertently overwrite your previous program.
3. Select the board and COM port from the Tools menu.
4. Upload this program and open the Serial Monitor. You should see the sensor reading changes as you adjust the pressure with the syringe.

A number by itself can inform you about trends, but to make this more valuable, you need to know what you are measuring and what units are being used. The data displayed in the Serial Monitor will need structure.

You may recall that the code that is included in the setup() function runs only one time during the execution of the program. For example, in the “Vernier Sensors with Arduino” activity, you started with an example sketch that included a serial communication rate:

```
 // initialize serial communication at 9600 bits per second:
Serial.begin(9600);
```

Create the following new variables in the setup() function of the program after the "`Serial.begin (9600);`" line of code:

```
char sensor [ ] ="Gas Pressure Sensor";
char measurement [ ] = "Pressure";
char units [ ] =" (kPa)";
```

This creates variables with the values of Gas Pressure Sensor, Pressure, and kPa, respectively. You will use these variables to create a header in the Serial Monitor.

Notice that some of the code changes color as you type. These commands are recognized by the software as having additional associated information. For example, you have experienced their use when you have "declared" a variable.

- char indicates the variable is a 'character' or "string".
- float indicates the variable is a number that has a decimal point.
- int indicates a variable that is an integer number.

You can find additional information about these (and other) types of variables at https://www.arduino.cc/reference/en

Arduino's built-in Serial Monitor has a large family of functions that allow you to format the Serial Monitor output. Refer to https://www.arduino.cc/reference/en/language/functions/communication/serial/ for a complete listing and explanation.

We will focus on the following:

- `Serial.print();      //  prints the item in the parentheses`
- `Serial.println();   // prints the item in the parentheses and a carriage return`
- `Serial.print("\t");  // prints a tab`

Go to https://www.arduino.cc/reference/en/language/functions/communication/serial/print and review how the data in the parentheses are handled. Answer the following questions:

What is the default number of decimal places the serial printer provides? ________________

Do words and letters need to be placed within quotation marks? _________________________

1. Use the string variables you created earlier in the setup () function of your sketch to create a "header" that prints once and includes the following:
    a. Sensor name
    b. On a new line, print the heading "Pressure", followed by " (kPa)".
    c. On the next line (and subsequent lines), the sensor reading should print out, one reading per line.

Compile and upload your sketch to verify it works as expected. Modify the formatting until you are satisfied with its appearance. Save your work.

2. Modify the sketch to print a timestamp for each data point.
   a. In the space immediately above setup() function, insert the following line of code to declare a "global" variable. You will learn more about global variables in later activities:
   ```
   int readingNumber = 1; // global variable to count number of
   samples taken
   ```
   b. In the loop () function insert the following code immediately after the first curly brace: "{":
   ```
   // calculates the time for each reading
     float timeBetweenReadings = 500; // in ms
     float sensorTime;
     sensorTime = timeBetweenReadings/1000 * readingNumber; //
   calculates time
     readingNumber++; // increments readingNumber by 1 for next
   reading
   ```
   c. Modify the Serial.print statements so that the heading of the data includes "time (s)" and the data includes the timestamp on the same line followed by a carriage return for the next reading.
   d. Upload and observe the output from your new sketch in the Serial Monitor.

3. If you are using the Arduino IDE, you also have a Serial Plotter that can be used to quickly graph your data. Close your Serial Monitor and restart your sketch. Once it is compiled and uploaded to your Arduino board, click on Tools and select Serial Plotter. You will see a graph with your data displayed.

Extensions

1. Try out the Serial Plotter again with the "time" code included in the sketch. Is there anything unusual about the graph plotted? Can you fix the problem?
2. Modify the sketch so that it displays and graphs pressure in millimeters of mercury instead of kilopascal.