Assignment 5: Android and Arduino Interaction

The purpose of this assignment is to help you get started with Arduino and Android Interaction. See the Appendix in this document for getting started.

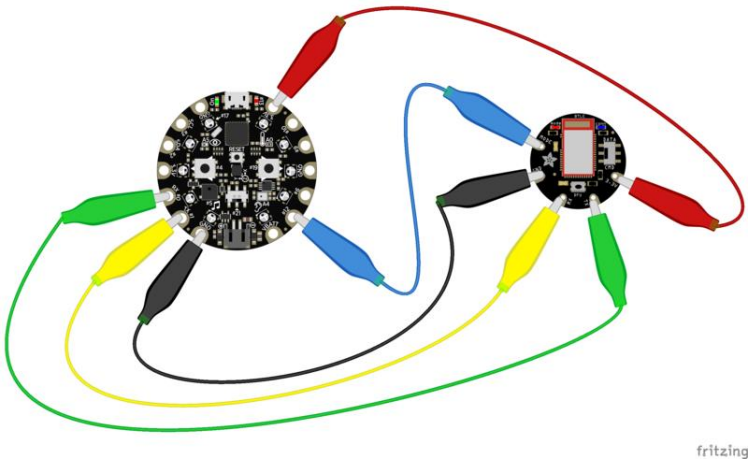
Submission

Submit your assignment using Canvas, under the Assignment section. The due date is shown in Canvas. You must submit a **.zip** file, containing the following items:

- Arduino Project folder
- README.txt file (see instructions below)
- video file with a demo.

The file named README.txt should contain your name and email address and a very brief description of it, along with any special instructions that the user might need to know in order to use it properly (if there are any). For example:

```
Laura Arjona <arjonal@uw.edu>
Arduino Counter – I used the left button instead of the right just to confuse the instructor.
```



Configure your timer (up to 100 points)

In this assignment you are going to develop an Android app that configures the timer that you developed in the previous assignment with the Arduino board.

1. The user should be able to configure the following parameters using an Android App.
   - Timer duration (you can set some upper and lower bound if you want)
   - LED colors for the timer.
   - Make buzz or not at the end.
   - Read temperature and display value on the phone.
   - Add a widget in the Android app (button for example) to start the timer.

2. Additionally, once configured, the timer should be started both by pressing a button in the Android App, and by pressing a button in the Arduino board.

3. It is not required to res-start the timer WHILE is counting.

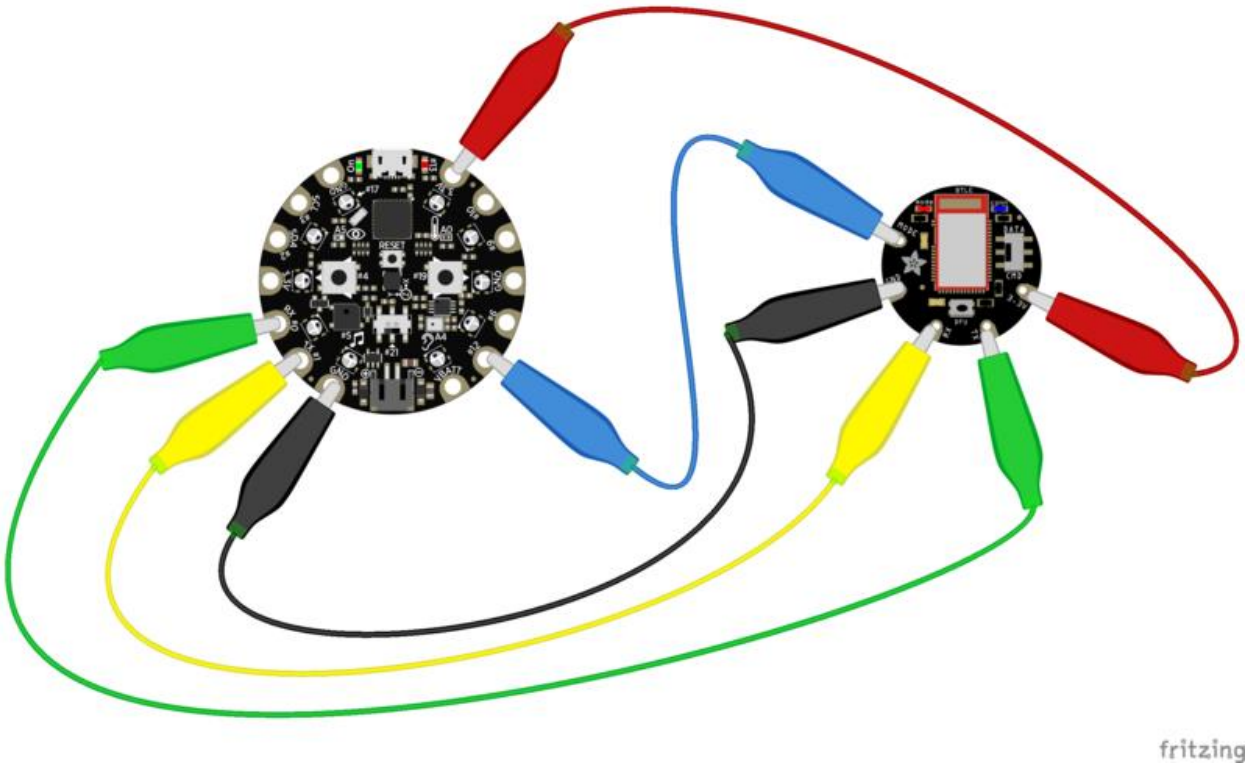
4. You can set some default configuration, in case the user wants to start the counter without configuring it.

Bonus: Pull-up counter (up to 120 points)

- Create a different project that uses Arduino to count the number of pull-ups that the user achieves. You can use the accelerometer sensor from the Arduino board.
- The user will start the counting process by pressing a start button in the Android App. Then, it will communicate with Arduino to indicate the beginning of the accelerometer data capture.
- Arduino will start recording the accelerometer data and will keep track of the number of pull-ups detected.
- For each pull-up detected, Arduino will blink one LED or will use the speaker to make a buzz (or other indication that you like).
- The user can then click a button in the Android App to display the number of pull-ups detected by Arduino
- You can assume that the pull-up detection is equivalent as a step detection
- A very basic algorithm is provided to help you get started. It uses the peak-finding based on a threshold. However, it uses a fixed window size. You could implement a sliding window to capture the data. You can also change the size of the moving average filter (set to 2 in the example).
- The validation for this project will be made by moving Arduino up and down in a slightly rapid movement.

Appendix

Connect the Arduino board to the BLE shield



- Circuit Playground 3.3V to Flora Bluefruit LE 3.3V (red wire).
- Circuit Playground GND to Flora Bluefruit GND (black wire).
- Circuit Playground serial TX to Flora Bluefruit serial RX (yellow wire). Double check you connect TX to RX and not TX to TX!
- Circuit Playground serial RX to Flora Bluefruit serial TX (green wire). Again double check you connect RX to TX and not RX to RX!
- Circuit Playground #12 to Flora Bluefruit MODE (blue wire). You can actually use any of the other numbered pins on Circuit Playground for this mode switch connection, however you'll need to modify the examples to use the pin number. For simplicity stick with pin 12 so you don't need to modify the code in the rest of the guide.

One last very important step is to flip the mode switch on the Flora Bluefruit LE module (or Bluefruit LE UART Friend) into the DATA position. If the switch is in the CMD position the software won't work correctly so be careful to double check the switch is in DATA mode.

Don't forget, set the Bluefruit LE module switch to the DATA position!

Possible implementation of a Sliding window of size 5

```
internal var slidingWindow = DoubleArray(5)

fun addToWindow(x: Double) {
    // Shift everything one to the left
    for (i in 1 until slidingWindow.size) {
        slidingWindow[i - 1] = slidingWindow[i]
    }

    // Add the new data point
    slidingWindow[slidingWindow.size - 1] = x

}
```

Concept of moving average with Sliding window