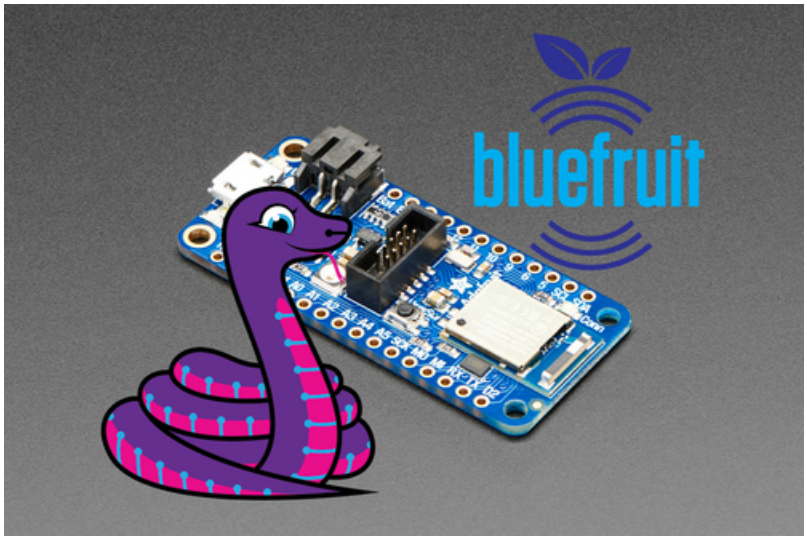
adafruit learning system

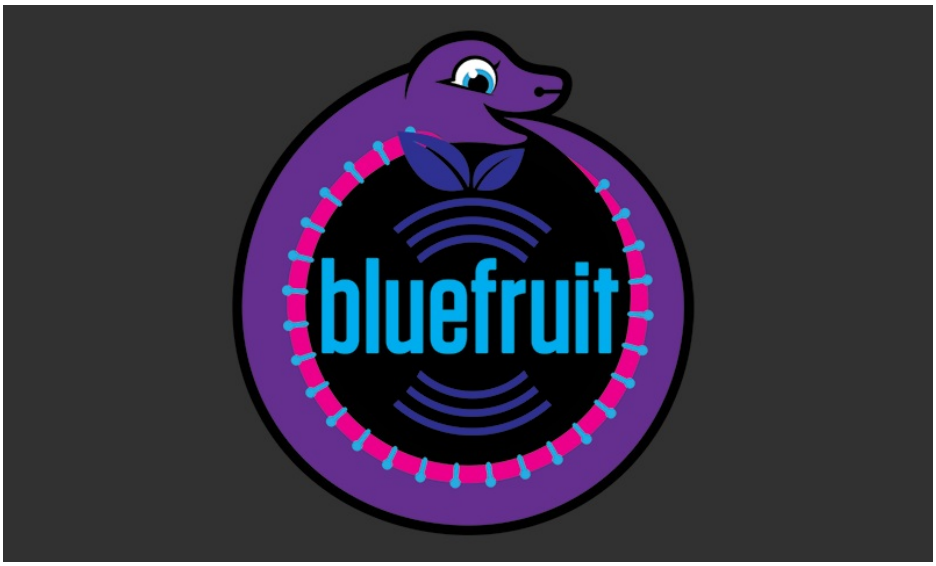
Getting Started with CircuitPython and Bluetooth Low Energy

Created by Kattni Rembor



Last updated on 2020-09-11 04:42:53 PM EDT

Overview



The Adafruit nRF52840 has joined the growing list of CircuitPython compatible boards. It features a Cortex M4 chip with Bluetooth Low Energy. We've added BLE support to CircuitPython which makes it super simple to get started!

BLE support in CircuitPython is currently in development. Be aware that you might find bugs! If you do run into a problem, please let us know! You can contact us on Discord or post an issue here: https://github.com/adafruit/circuitpython/issues

CircuitPython is Python that runs on microcontrollers. It's designed to lower the barrier for entry to learning programming and electronics. If you're new to CircuitPython, please check out the Welcome to CircuitPython guide (https://adafru.it/cpy-welcome) to find out how to get started with CircuitPython.

CircuitPython, BLE and Bluefruit LE Connect

This guide is designed to help you get started with CircuitPython, the Adafruit nRF52840 and the Bluefruit LE Connect app.

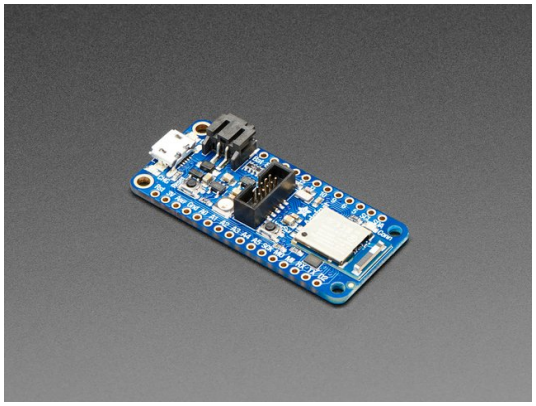
First you'll learn some Bluetooth Low Energy Basics to help you understand what your CircuitPython code is doing.

There is a quick intro to the Adafruit Bluefruit LE Connect app with links to install it, and how to get to the controller interface.

Then you'll be provided a series of quick demos using CircuitPython, the Adafruit nRF52840, and the Bluefruit LE Connect app for mobile devices. You'll learn things like how to read a button press from the app's control pad interface, how to change the color of a NeoPixel, or how to read acceleration data from your mobile device.

Let's get started!

Parts



Adafruit Feather nRF52840 Express

$24.95
IN STOCK

Add To Cart

USB cable - USB A to Micro-B

$2.95
IN STOCK

Add To Cart

Bluetooth Low Energy Basics

The nRF52840 uses Bluetooth Low Energy, or BLE. BLE is a wireless communication protocol used by many devices, including mobile devices. You'll be able to communicate with your nRF52840 board using your mobile phone!

There's a few terms and concepts commonly used in BLE with which you may want to familiarise yourself. This will help you understand what your code is doing when you're using CircuitPython and BLE.

The major concepts can be broken down into two categories: connection set up and communication. The first deals with setting up connections between devices, such as between your mobile phone and the nRF52840 board. The second deals with communication between the devices once they are connected.

Bluetooth Terms

- **Central** - The host computer. This is often a mobile device such as a phone or tablet, or it could be a desktop or laptop.
- **Peripheral** - The connected device. Examples of peripherals are: heart rate monitor, smart watch, or fitness tracker. The CircuitPython code we have so far is designed to make the Adafruit nRF52840 devices work as peripherals.
- **Advertising** - Information sent by the peripheral during connection set up. When a device advertises, it is transmitting the name of the device and describing its capabilities. The central looks for an advertising peripheral to connect to, and uses that information to determine what the peripheral is capable of.
- **Service** - A function the peripheral provides. The peripheral advertises its services. A really common service that we use is the UART service, which acts like a hardware UART and is a way of bidirectionally sending information to and from devices.
- **Packet** - Data transmitted by a device. BLE devices and host computers transmit and receive data in small bursts called packets.

Making a Bluetooth Connection

To use these terms in the context of connecting to your Adafruit nRF52840:

- You run CircuitPython code that makes your board act as a peripheral by advertising its name and the services it's capable of.
- You start up Adafruit's **Bluefruit LE Connect app** on an Android or iOS device in central mode, that device becomes the central, and begins listening for the peripheral.
- You set up the connection between the nRF52840 peripheral and the Bluefruit LE Connect app, and the app discovers the details about the services that the peripheral is capable of.
- Once this connection is made, you can use CircuitPython code to read packets sent from the Bluefruit LE Connect app to your nRF52840 board. For example, you can receive data describing screen button presses or RGB color values.

Now that you have a general idea of basic BLE terms and concepts, it's time to install the Bluefruit LE Connect

application, and run some CircuitPython demos!

Bluefruit LE Connect Basics

Getting the free Adafruit Bluefruit LE Connect App

To interact with your board from your phone or tablet, you'll use the Adafruit Bluefruit LE Connect App (https://adafru.it/iCi). Install it from the Apple App Store (https://adafru.it/BYj) or Google Play App Store (https://adafru.it/f4G).

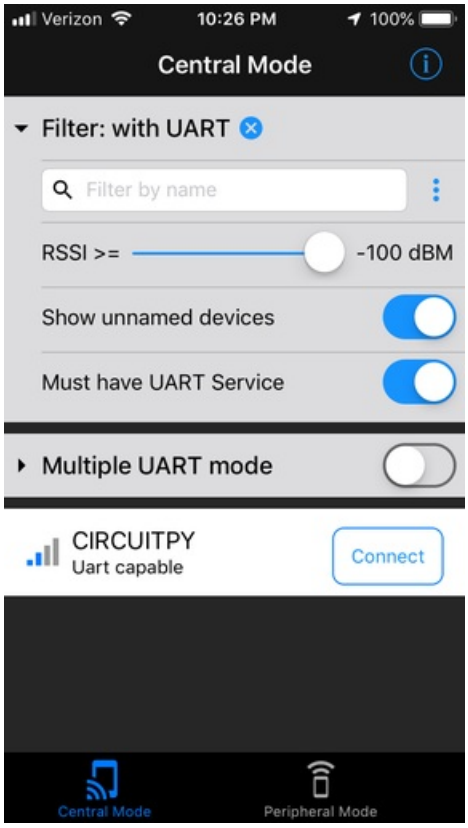
Connect to nRF52840

Start your Adafruit nRF52840 board with code running from **code.py**. You must have code running for this to work! The examples on the following pages have ready to go CircuitPython **code.py** scripts to try out.

Next, start up the Bluefruit LE Connect app, and make sure it's in Central Mode (left button on the bottom). When you start the app, you should see a device named **CIRCU** or **CIRCUITPY**. If there's a long list of devices, you can shorten it by turning on the "**Must have UART Service**" switch.

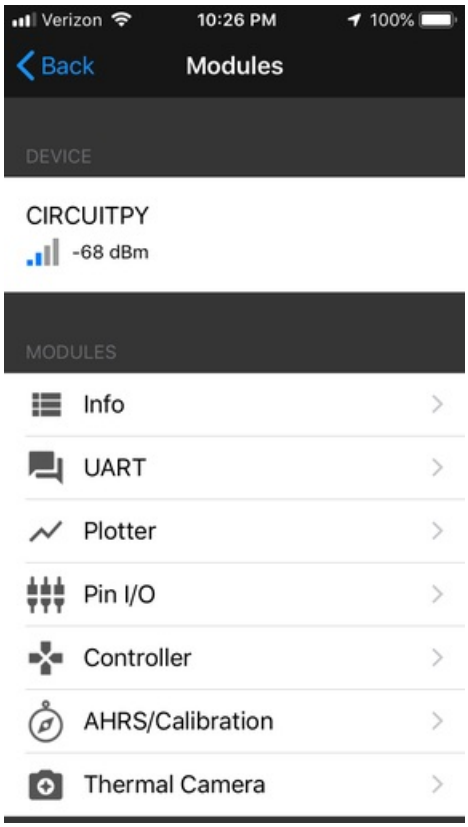
To connect to the Adafruit nRF52840, touch the **Connect** button. You should see "**Connecting**" and then "**Discovering Services**".

If you don't see **CIRCU** or **CIRCUITPY** right away, try pulling down to refresh. If that doesn't work, try turning Bluetooth off and back on on your phone or tablet and restarting the app.



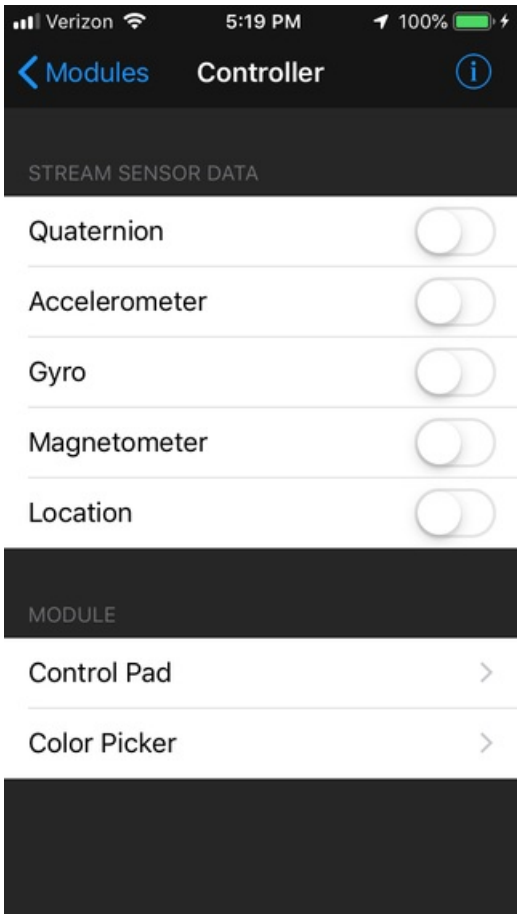
Device Menu

After you connect, you'll see a menu of how you can interact with the device. Choose **Controller**.



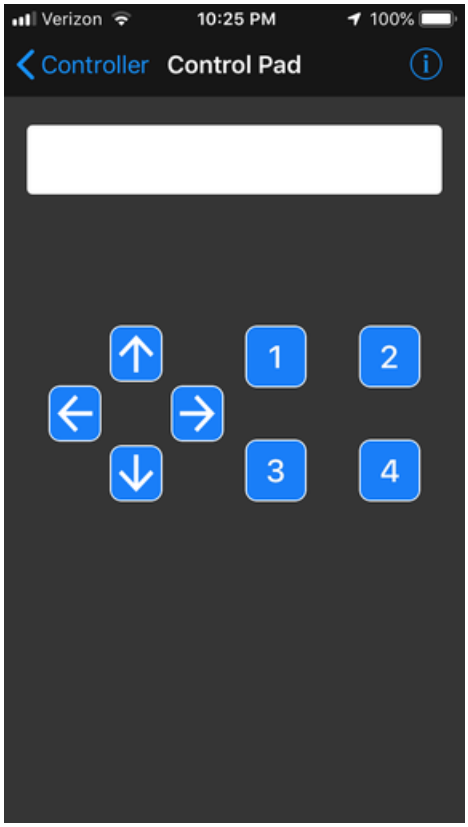
Controller Menu

After you choose Controller, you'll see another screen with more choices.



Control Pad

Here's the **Control Pad** screen. You can use the buttons as inputs in CircuitPython. For example, you could have code that moved a servo arm up and down when the up or down button is pressed.



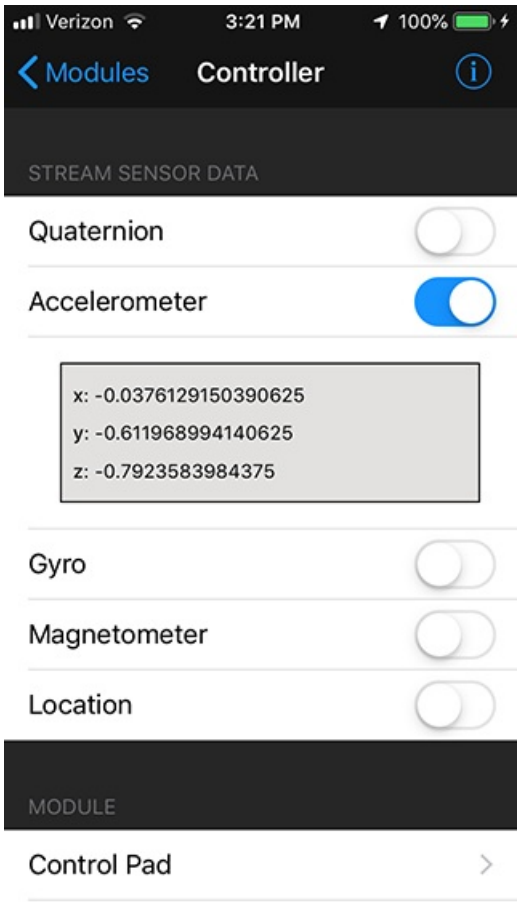
Color Picker

To change the color of the NeoPixels, use the **Color Picker** screen. Choose a color and press **Send Selected Color**.

Stream Sensor Data

There are multiple options under Stream Sensor Data including **Accelerometer**. You enable these options to stream the associated data from your mobile device to your Adafruit nRF52840 board, and use CircuitPython to read that data.

For example, if you have code that reads the acceleration data from your mobile device, you'd want to enable **Accelerometer** once you connected to the app.



Check out this detailed guide (https://adafru.it/DNc) for more information about the Adafruit Bluefruit LE Connect application.

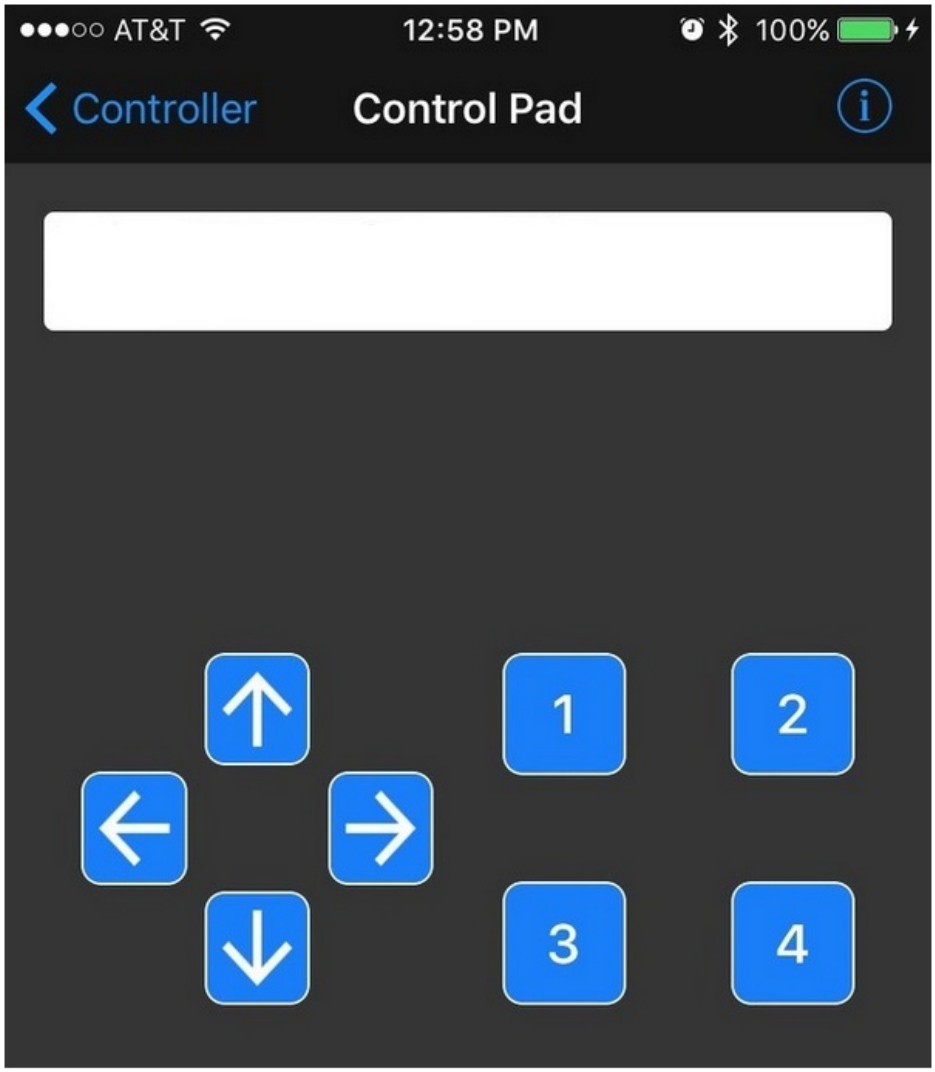
Button Press

The Bluefruit LE Connect app has a control pad with 8 buttons: UP, DOWN, LEFT, RIGHT, 1, 2, 3 and 4. This demo will show you how to access those buttons, and use them to print a message to the serial console.

Save the following as **code.py** on your CIRCUITPY drive and then connect to the board using the Bluefruit LE Connect app. Then, connect to the serial console.

```
from adafruit_ble import BLERadio
from adafruit_ble.advertising.standard import ProvideServicesAdvertisement
from adafruit_ble.services.nordic import UARTService

from adafruit_bluefruit_connect.packet import Packet
from adafruit_bluefruit_connect.button_packet import ButtonPacket

ble = BLERadio()
uart = UARTService()
advertisement = ProvideServicesAdvertisement(uart)

while True:
    ble.start_advertising(advertisement)
    while not ble.connected:
        pass

    # Now we're connected

    while ble.connected:
        if uart.in_waiting:
            packet = Packet.from_stream(uart)
            if isinstance(packet, ButtonPacket):
                if packet.pressed:
                    if packet.button == ButtonPacket.BUTTON_1:
                        # The 1 button was pressed.
                        print("1 button pressed!")
                    elif packet.button == ButtonPacket.UP:
                        # The UP button was pressed.
                        print("UP button pressed!")

    # If we got here, we lost the connection. Go up to the top and start
    # advertising again and waiting for a connection.
```

Connect to your board using the Bluefruit LE Connect application and navigate to the Control Pad page.



Let's take a look at the code.

First we import the necessary libraries and instantiate the UART server.

Inside the loop, we first begin advertising.

Once connected, the board begins listening for packets. In the event that a button packet is received, it checks to see if the button was pressed. Then it checks to see whether it was button 1 or the UP button, and prints a message to the serial console.

The buttons are available as:

- 1 = `ButtonPacket.BUTTON_1`
- 2 = `ButtonPacket.BUTTON_2`
- 3 = `ButtonPacket.BUTTON_3`
- 4 = `ButtonPacket.BUTTON_4`
- UP = `ButtonPacket.DOWN`
- DOWN = `ButtonPacket.UP`
- LEFT = `ButtonPacket.LEFT`
- RIGHT = `ButtonPacket.RIGHT`

If the board loses connection, the loop will begin again and the board will resume advertising until a new connection is made.

Now you can press the up button or the 1 button, and see the resulting print statement in the serial console!

That's all there is to reading button presses from the Bluefruit LE Connect app with CircuitPython! Now you can use those buttons do all sorts of things like control a servo, light up an LED, or control your robotic creation. Give it a try!

NeoPixel Color

The Bluefruit LE Connect app has a Color Picker that allows you to easily set the color of NeoPixels connected to your Adafruit nRF52840. For this demo, we'll change the color of the on-board NeoPIxel using CircuitPython and the Bluefruit LE Connect app.

Save the following as **code.py** on your CIRCUITPY drive and then connect to the board using the Bluefruit LE Connect app.

This example expects the latest CircuitPython and the latest version of the Adafruit CircuitPython BLE library! Click below to download the latest CircuitPython and library bundle including the BLE library.

https://adafru.it/Em8

https://adafru.it/Em8

https://adafru.it/ENC

https://adafru.it/ENC

```
Temporarily unable to load content:
```

Connect to your board using the Bluefruit LE Connect application and navigate to the Color Picker page.

Let's take a look at the code.

First we import the necessary libraries and instantiate the UART server.

Inside the loop, we first begin advertising.

Once connected, the board begins listening for packets. In the event that a color packet is received, it changes the color of the NeoPixel to the chosen color.

Now you can use the color picker to change the color and send that color to the NeoPixel.

That's all there is to changing the color of NeoPixels with the Bluefruit LE Connect app and CircuitPython! You can easily attach external NeoPixels and change the color with the app as well. Check it out!

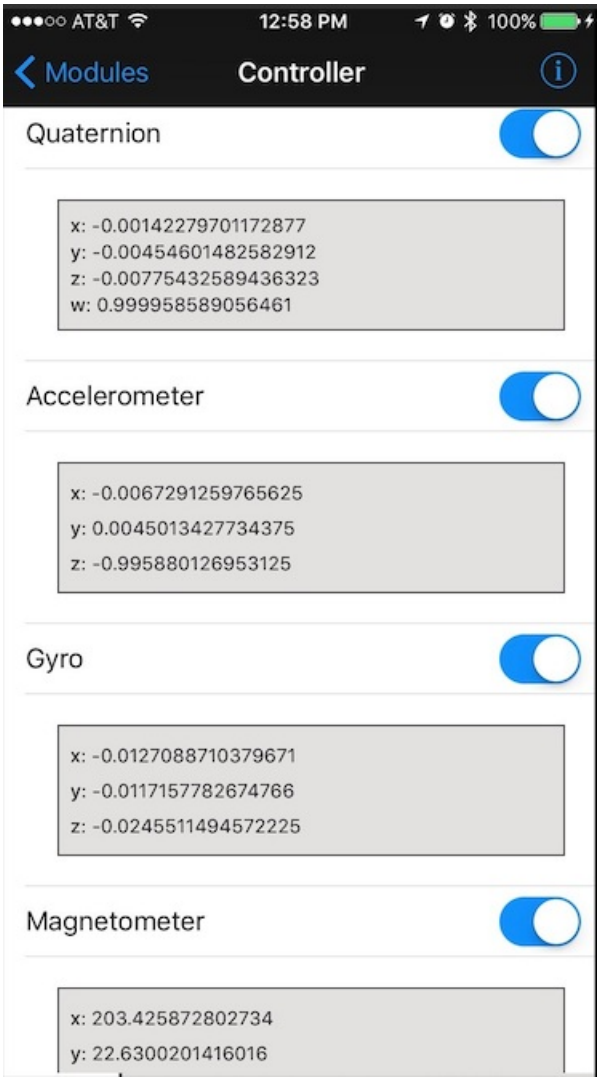
Mobile Movement Data

The Bluefruit Connect app allows you to access the accelerometer, magnetometer, gyroscopic and quaternion movement information from your mobile device. This demo will show you how to access that data and print it to the serial console.

Save the following as **code.py** on your `CIRCUITPY` drive and then connect to the board using the Bluefruit LE Connect app. Then, connect to the serial console.

```
from adafruit_ble import BLERadio
from adafruit_ble.advertising.standard import ProvideServicesAdvertisement
from adafruit_ble.services.nordic import UARTService

from adafruit_bluefruit_connect.packet import Packet
from adafruit_bluefruit_connect.accelerometer_packet import AccelerometerPacket
from adafruit_bluefruit_connect.magnetometer_packet import MagnetometerPacket
from adafruit_bluefruit_connect.gyro_packet import GyroPacket
from adafruit_bluefruit_connect.quaternion_packet import QuaternionPacket

ble = BLERadio()
uart = UARTService()
advertisement = ProvideServicesAdvertisement(uart)

while True:
    ble.start_advertising(advertisement)
    while not ble.connected:
        pass

    # Now we're connected

    while ble.connected:
        if uart.in_waiting:
            packet = Packet.from_stream(uart)
            if isinstance(packet, AccelerometerPacket):
                print("Acceleration:", packet.x, packet.y, packet.z)
            if isinstance(packet, MagnetometerPacket):
                print("Magnetometer:", packet.x, packet.y, packet.z)
            if isinstance(packet, GyroPacket):
                print("Gyro:", packet.x, packet.y, packet.z)
            if isinstance(packet, QuaternionPacket):
                print("Quaternion:", packet.x, packet.y, packet.z)

    # If we got here, we lost the connection. Go up to the top and start
    # advertising again and waiting for a connection.
```

Connect to your board using the Bluefruit LE Connect application and navigate to the controller page. Enable the following streams: Quaternion, Accelerometer, Gyro, Magnetometer.



Let's take a look at the code.

First we import the necessary libraries and instantiate the UART server.

Inside the loop, we first begin advertising.

Once connected, the board begins listening for packets. In the event that an accelerometer, magnetometer, gyro or quaternion packet is received, it prints a message to the serial console.

That's all there is to accessing acceleration, magnetometer, gyro and quaternion (https://adafru.it/fEd) data from your mobile using the Bluefruit LE Connect app and CircuitPython!

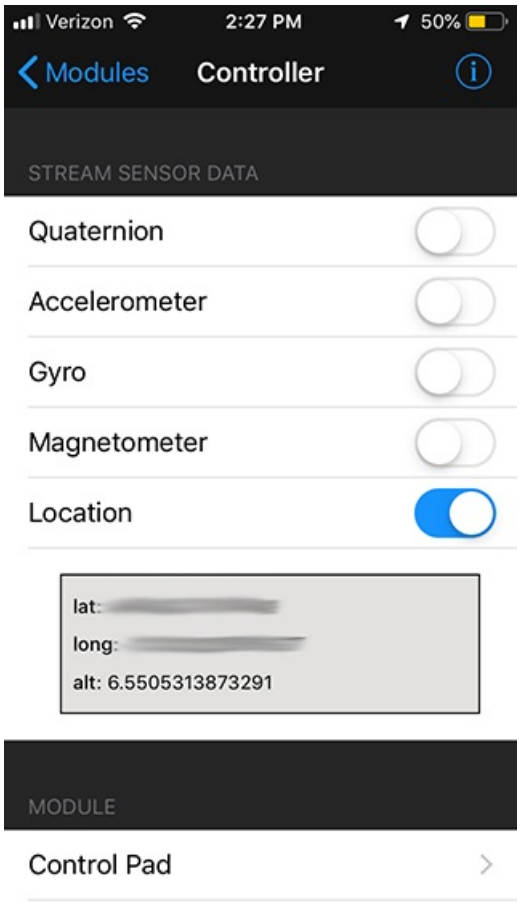
Location

The Bluefruit LE Connect application allows you to access the location data from your phone. This demo will show you how to access that data and print it to the serial console.

Save the following as **code.py** on your `CIRCUITPY` drive and then connect to the board using the Bluefruit LE Connect app. Then, connect to the serial console.

```
from adafruit_ble import BLERadio
from adafruit_ble.advertising.standard import ProvideServicesAdvertisement
from adafruit_ble.services.nordic import UARTService

from adafruit_bluefruit_connect.packet import Packet
from adafruit_bluefruit_connect.location_packet import LocationPacket

ble = BLERadio()
uart = UARTService()
advertisement = ProvideServicesAdvertisement(uart)

while True:
    ble.start_advertising(advertisement)
    while not ble.connected:
        pass

    # Now we're connected

    while ble.connected:
        if uart.in_waiting:
            packet = Packet.from_stream(uart)
            if isinstance(packet, LocationPacket):
                print("Latitude:", packet.latitude)
                print("Longitude", packet.longitude)
                print("Altitude:", packet.altitude)

    # If we got here, we lost the connection. Go up to the top and start
    # advertising again and waiting for a connection..
```

Connect to your board using the Bluefruit LE Connect application and navigate to the Controller page. Enable the Location data stream.



Let's take a look at the code.

First we import the necessary libraries and instantiate the UART server.

Inside the loop, we first begin advertising.

Once connected, the board begins listening for packets. In the event that a location packet is received, it prints a message to the serial console.

That's all there is to accessing location data from your mobile using the Bluefruit LE Connect app and CircuitPython!

BLE Technical Details

BLE Technical Details (https://adafru.it/DN8)