Sending Live Sensor Data to iOS with BLE

Created by Todd Treece

https://learn.adafruit.com/bno055-ble-ios-app

Last updated on 2023-08-29 02:57:36 PM EDT

Table of Contents

Overview

If you have ever wanted to create an iOS application that interacts with an Arduino project, but didn't want to take the time to learn Objective-C or Swift, then this project is for you. We will be using Apache Cordova () to create an iOS application using HTML and Javascript. If you missed our previous guide about creating a simple iOS weather application, then you may want to check out that guide () before continuing with this more advanced Cordova project.

The example application that we will be creating will give visual and audible feedback about the orientation of a BNO055 9-DOF sensor. The sensor data will be sent using a Bluefruit LE UART Friend from a Pro Trinket.

Installing Dependencies

You will need a Mac running OS X 10.10 to run the latest version of XCode, which is required for uploading to iOS devices. Thanks to changes in XCode 7, you no longer need an Apple Developer account to test applications on iOS devices. You can also use XCode 6 for this tutorial, but you will need an Apple Developer account to upload to your iOS device. XCode 7 is still in beta and not available in the App Store yet, so you will need to download it from Apple's Developer site and install.

XCode 7 Download

You will also need the latest version of node.js () installed on your computer. Download and install the latest version using the link below.

Node.js Download

You will also need the latest version of the Arduino IDE (v1.6.4+) for this project. Download and install the latest version using the link below.

Arduino IDE Download

Git Repository

Now that you have all of the dependencies installed, you will need to download or clone a copy of the git repository ().

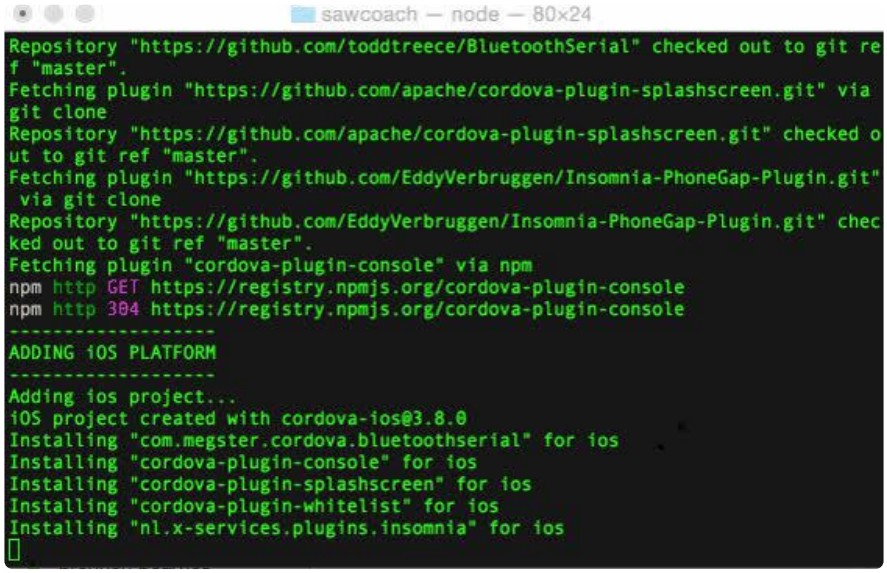
```
git clone https://github.com/adafruit/BNO055-Cordova-Example.git
```

ZIP Archive Download

Next, you will need to navigate to the extracted or cloned repo in the terminal and run the install script using the following command.

```
bash install.sh
```

You will see output like the image below as the script installs the necessary Cordova and Arduino library dependencies.

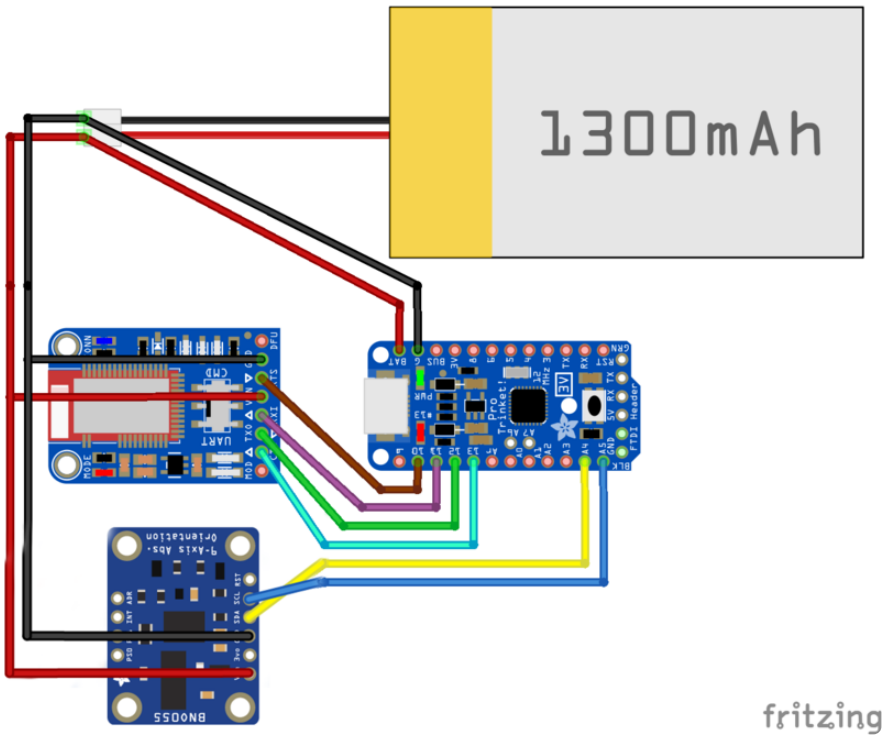


Now that we have installed all of the software dependencies, we are ready to wire up the hardware.

Wiring

First, you will need to wire the BNO055 to the Pro Trinket and battery.

- VIN to battery +
- GND to battery GND
- SDA to Pro Trinket A4
- SCL to Pro Trinket A5

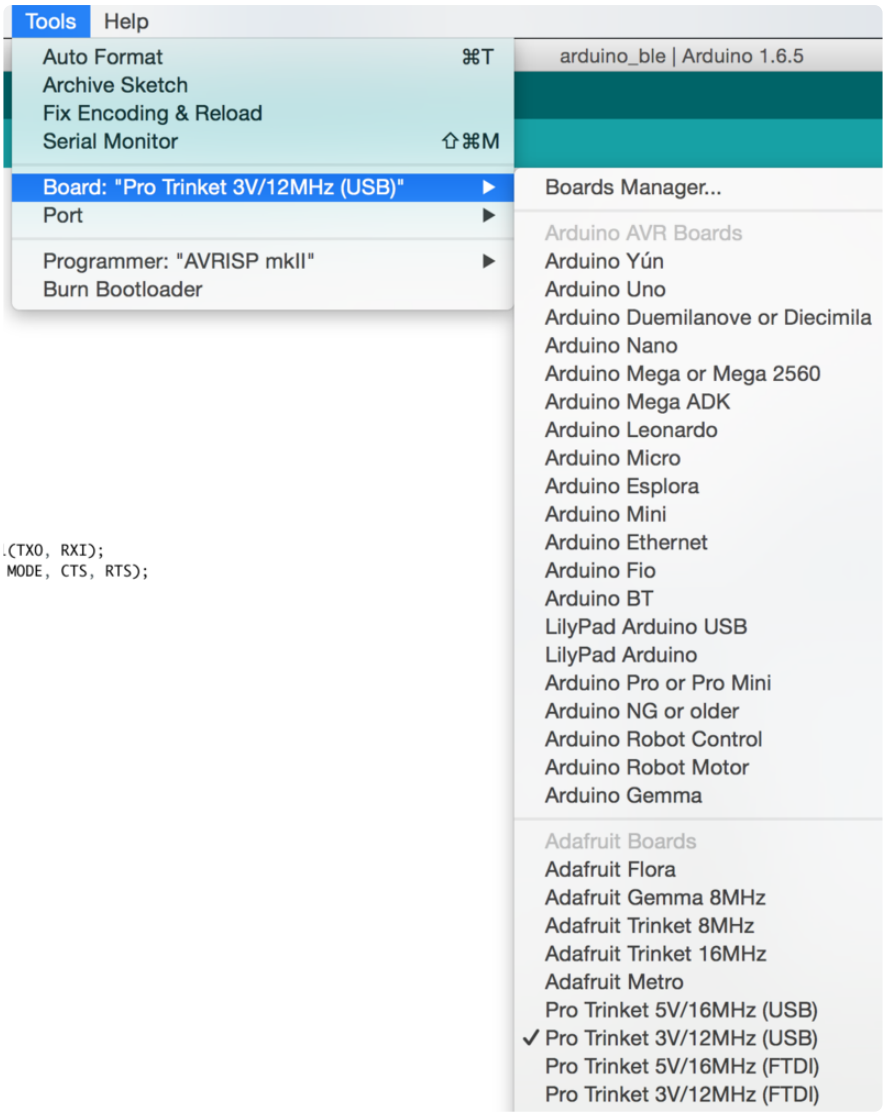


Next, wire the Bluefruit to the Pro Trinket and battery.

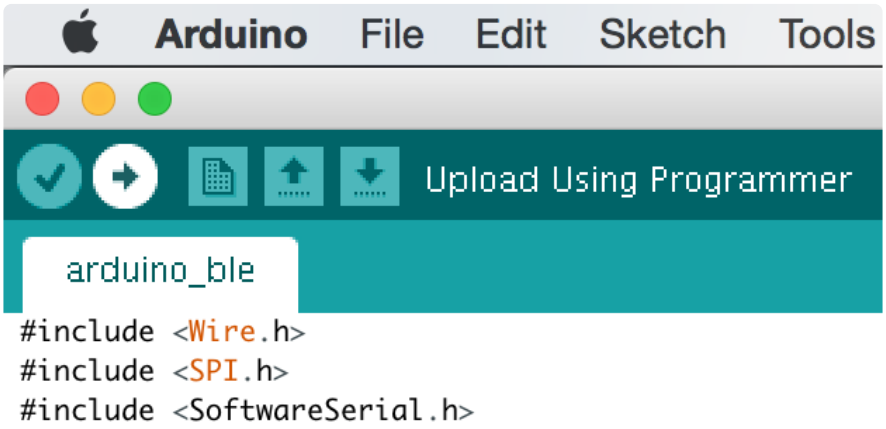
- VIN to battery +
- GND to battery GND
- RTS to Pro Trinket 10
- RXI to Pro Trinket 11
- TXO to Pro Trinket 12
- CTS to Pro Trinket 13

Uploading & Testing

Open the arduino_ble.ino example sketch in the Arduino IDE, and attach the Pro Trinket 3V to your computer using a USB cable. Then, select Pro Trinket 3V/12MHz (USB) from the Tools->Board menu.



Then, use the upload button at the top of the IDE to upload the sketch to the Pro Trinket.



Installing the iOS App

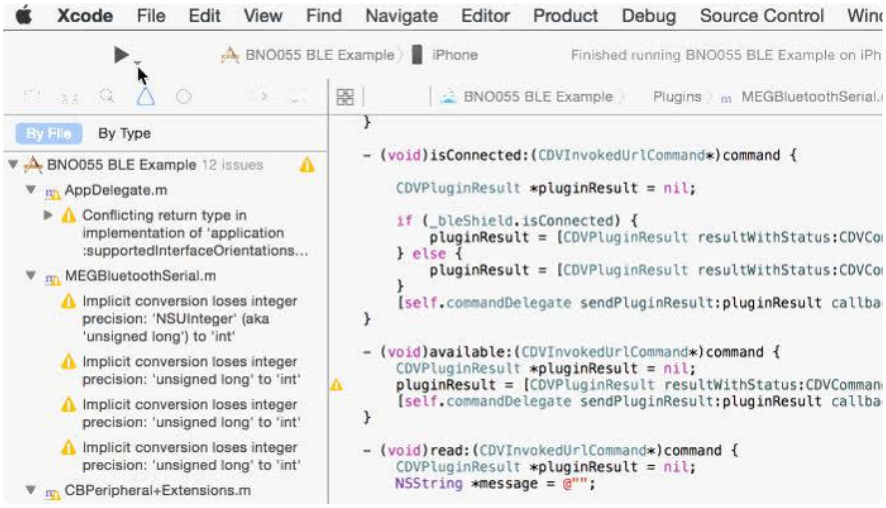
Next, navigate to the repo in the terminal and run the following command to tell Cordova to build the iOS app.

```
cordova build
```

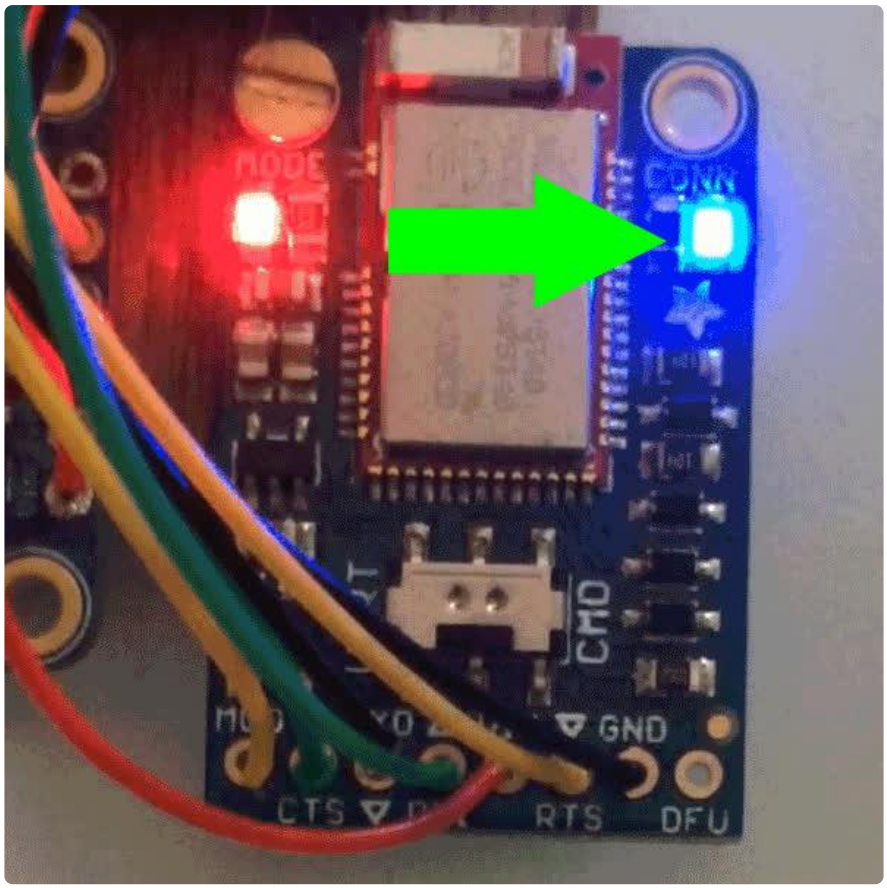
Once you have built the app, you can open up the XCode project located at the following path inside the repo folder:

```
platforms/ios/BNO055 BLE Example.xcodeproj
```

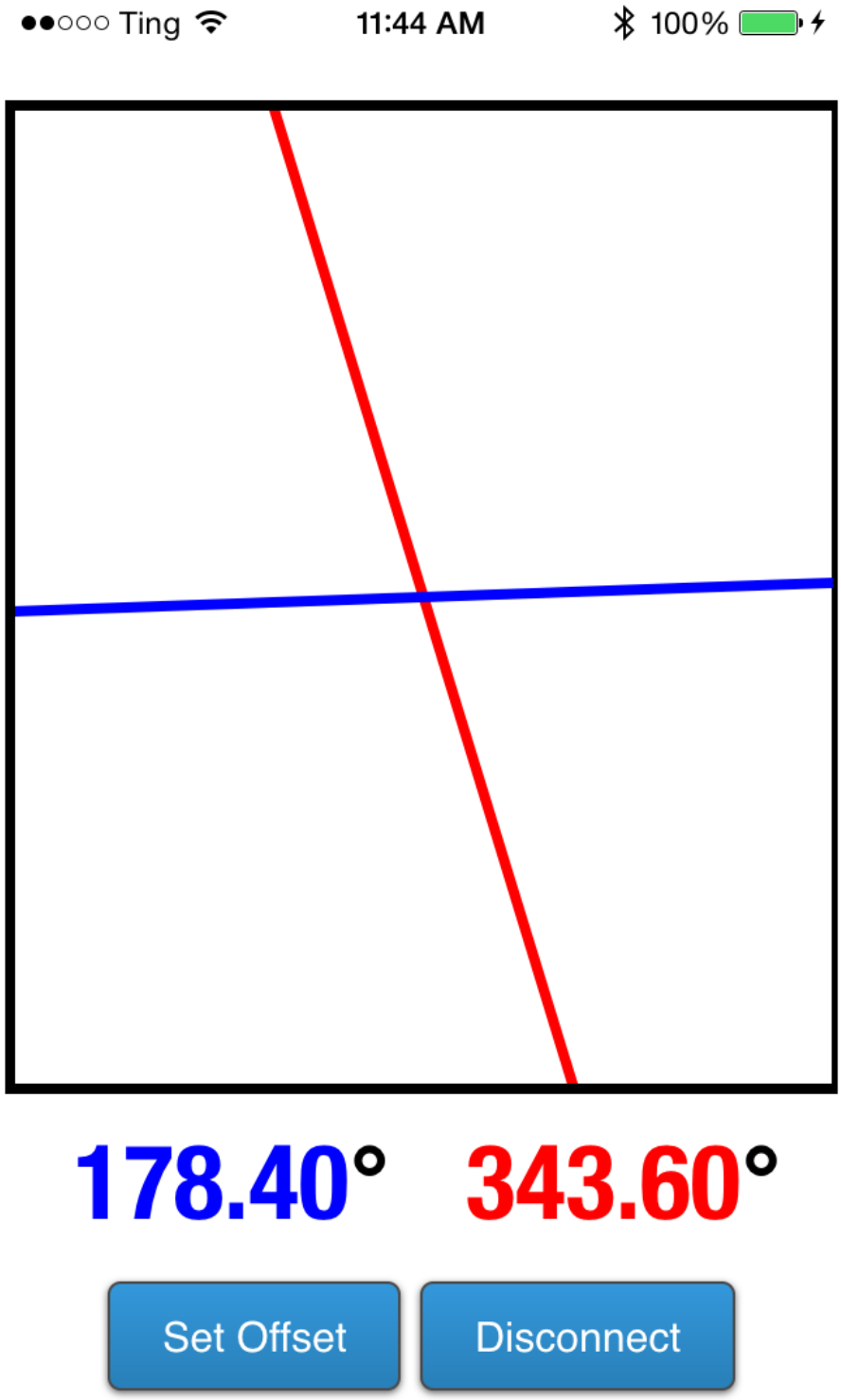
Plug in your iOS device, and click the play button at the top of the XCode project to upload it to your device.



Plug in the battery to the Pro Trinket, BNO055, and Bluefruit. You should then see the blue connection light change on the Bluefruit when the iOS app connects.



The iOS app will then display the current orientation info for roll and yaw.



You can use the Set Offset button to zero the graph at the current orientation. All readings will then be relative to that offset.

Going Further

The iOS app is split into separate Javascript modules for charting, audio, bluetooth, and sensor readings. Some of these might not be useful for your specific needs, but they are a good starting point for hardware interaction over BLE.

```
├── README.md
├── arduino_ble
│   └── arduino_ble.ino
├── config.xml
├── hooks
│   └── README.md
├── install.sh
└── www
    ├── css
    │   └── index.css
    ├── index.html
    └── js
        ├── audio.js
        ├── chart.js
        ├── d3.js
        ├── index.js
        └── reading.js

5 directories, 12 files
```

If you need to modify any of the files in the www folder, you will need to run cordova build after every change, and re-upload the app to your device with XCode.

The main BLE connection and data parsing portion of the code can be found in www/js/index.js.

```
  proto.processData = function(data) {

    past_low = this.low_battery;

    data = data.split(',');

    this.yaw.update(data[0]);
    this.roll.update(parseFloat(data[2]) + 180);
    this.low_battery = parseInt(data[3]) &amp;lt; 3300 ? true : false;

    if(this.low_battery &amp;amp;&amp;amp; !past_low)
      navigator.notification.alert('The Bluefruit\'s battery is low', null,
'Warning');

    this.update();

  };
```

You can see that the data parsed out here is sent in the CSV format from the main loop in the arduino_ble.ino sketch.

```
  ble.print("AT+BLEUARTTX=");
  ble.print(event.orientation.x, 1);
  ble.print(",");
```

```
  ble.print(event.orientation.y, 1);
  ble.print(",");
  ble.print(event.orientation.z, 1);
  ble.print(",");
  ble.print(battery, DEC);
  ble.println("|");
```

You can modify this portion of the sketch to send data from any sensor to iOS.

Resources

You may find the following resources helpful when developing or modifying a BLE Cordova Application.

- Cordova Documentation ()
- Bluetooth Serial Plugin ()
- Phonegap Documentation () - Phonegap is based on Cordova, so a large portion of the documenatation, tutorials and plugins apply to both Cordova and Phonegap.
- Debugging in Phonegap ()
- BNO055 Guide ()
- Bluefruit LE UART Friend Guide ()