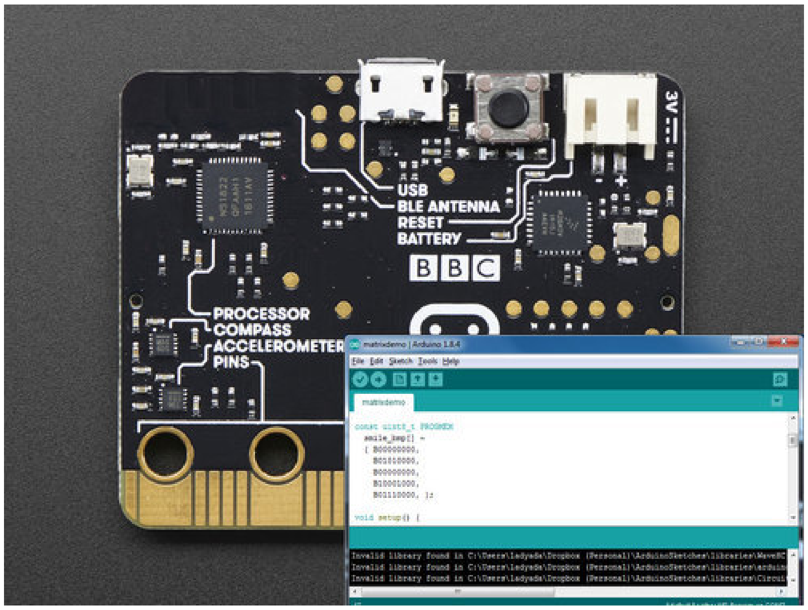
adafruit learning system

Micro:bit with Arduino

Created by lady ada

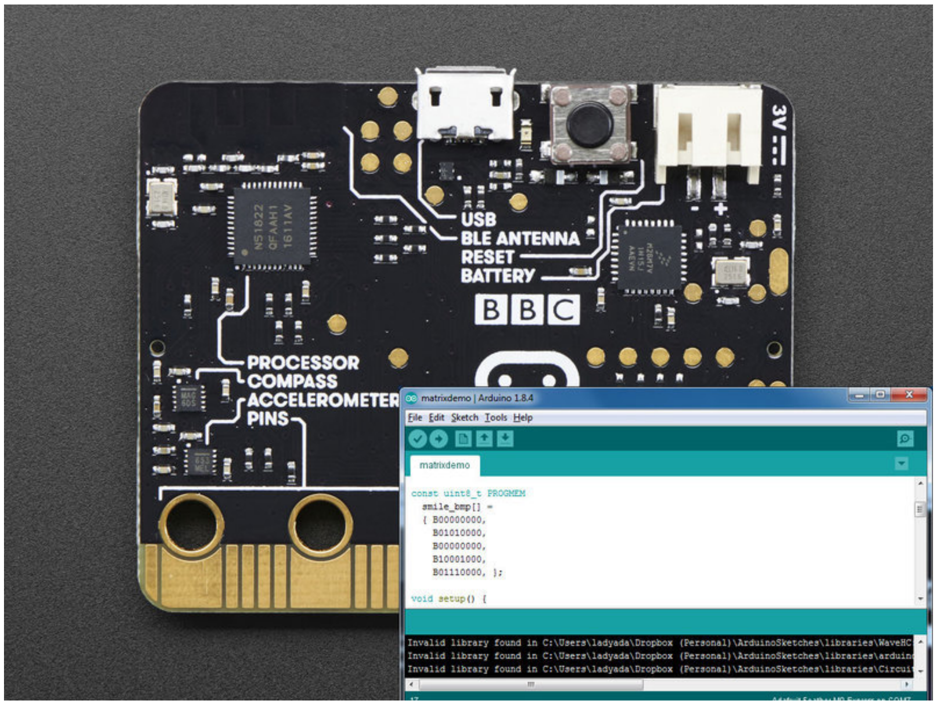


```
const uint8_t PROGMEM
  smile_bmp[] =
  { B00000000,
    B01010000,
    B00000000,
    B10001000,
    B01110000, };

void setup() {
```

```
Invalid library found in C:\Users\ladyada\Dropbox (Personal)\ArduinoSketches\libraries\WaveHC
Invalid library found in C:\Users\ladyada\Dropbox (Personal)\ArduinoSketches\libraries\arduino
Invalid library found in C:\Users\ladyada\Dropbox (Personal)\ArduinoSketches\libraries\Circuit
```

Last updated on 2020-05-18 09:21:46 PM EDT

Overview



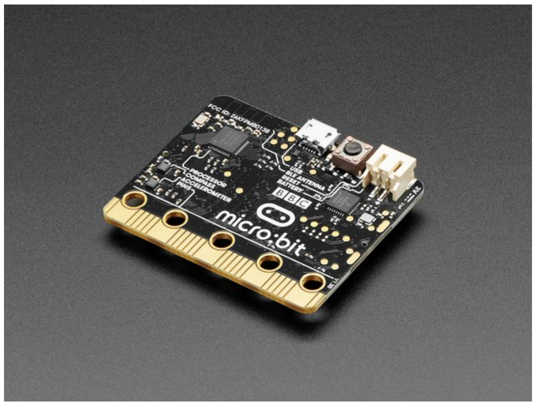
```
const uint8_t PROGMEM
  smile_bmp[] =
  { B00000000,
    B01010000,
    B00000000,
    B10001000,
    B01110000, };

void setup() {
```

```
Invalid library found in C:\Users\ladyada\Dropbox (Personal)\ArduinoSketches\libraries\WaveHC
Invalid library found in C:\Users\ladyada\Dropbox (Personal)\ArduinoSketches\libraries\arduin
Invalid library found in C:\Users\ladyada\Dropbox (Personal)\ArduinoSketches\libraries\Circui
```

Did you know that the Arduino IDE can be used to program the micro:bit? Now you have yet another way to use this cool board! Learn how to set up Arduino to program your micro:bit, blink some LEDs, read the internal temperature sensor, send and receive data over Bluetooth - even log data to Adafruit.IO!

The micro:bit is a small nRF51-powered learning platform for kids - you can use it with Microsoft MakeCode (https://adafru.it/zrb) (drag-n-drop block programming or Javascript), micropython (https://adafru.it/zrc), or mbed. But we really like using the Arduino IDE, especially since there's thousands of existing projects you can use and adapt. Also, you get to have much more advanced projects since you won't run out of memory (as you would with micropython) and you can write just about any code you want (with MakeCode you are more constrained to what has already been provided for you, a trade-off for ease-of-use).

Pick up a microbit and follow along on how you can do some pretty advanced things with your 'bit!



BBC micro:bit

$14.95
IN STOCK
Add To Cart

Your browser does not support the video tag.

BBC micro:bit Go Bundle

$17.50
IN STOCK
Add To Cart

Install board and blink!

Install Windows 7 Driver

If you are running Windows 7 you need to install this driver.

If you are on Mac, Win 10+ or Linux it is not required! Skip this step

https://adafru.it/pNa

https://adafru.it/pNa

Download Arduino IDE

You will need to use the Desktop IDE. Make sure you are running the latest version.

https://adafru.it/fvm

https://adafru.it/fvm

Install SoftDevice onto MicroBit

Arduino assumes there's a 'softdevice' radio already installed. If you used MicroPython with your microbit, that softdevice was erased.

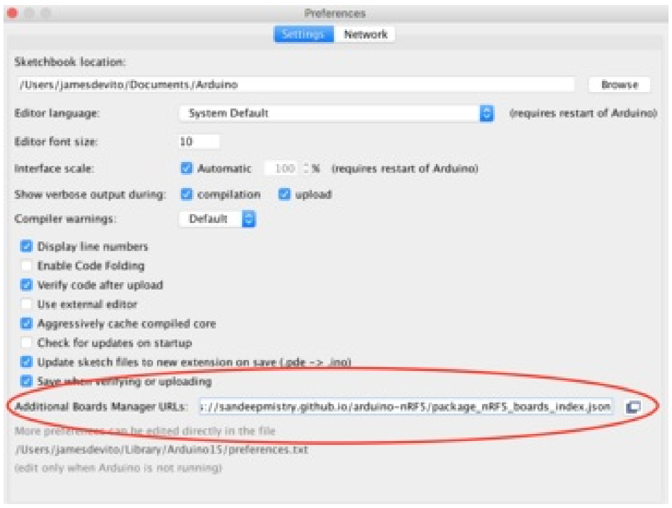
Reinstalling it is easy, download this hex and 'drag' it to your MICROBIT drive to install a MakeCode bluetooth advertising example. That, for sure, will force you to have a softdevice on :)

https://adafru.it/zwf

https://adafru.it/zwf

Add NRF5x Board Support

The microbit uses the nRF51 which is not 'natively' supported. But its easy to add support!


Preferences

Settings | Network

Sketchbook location: /Users/jamesdevito/Documents/Arduino

Editor language: System Default (requires restart of Arduino)

Editor font size: 10

Interface scale: ☑ Automatic 100 % (requires restart of Arduino)

Show verbose output during: ☑ compilation ☑ upload

Compiler warnings: Default

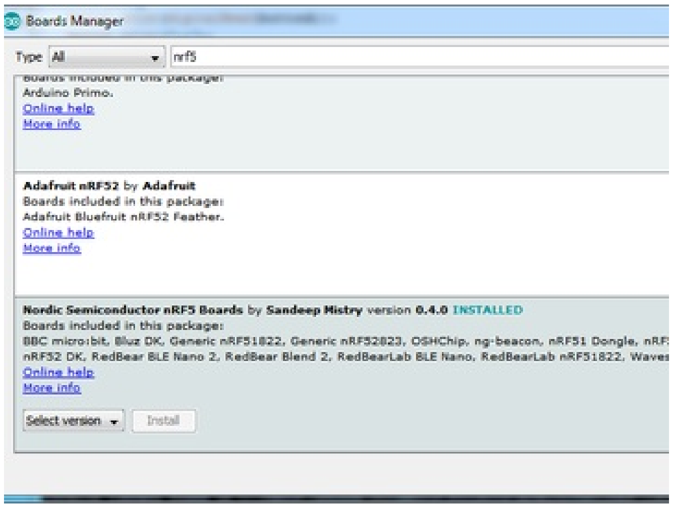
- ☑ Display line numbers
- ☐ Enable Code Folding
- ☑ Verify code after upload
- ☐ Use external editor
- ☑ Aggressively cache compiled core
- ☐ Check for updates on startup
- ☑ Update sketch files to new extension on save (.pde -> .ino)
- ☑ Save when verifying or uploading

Additional Boards Manager URLs: ://sandeepmistry.github.io/arduino-nRF5/package_nRF5_boards_index.json

More preferences can be edited directly in the file
/Users/jamesdevito/Library/Arduino15/preferences.txt
(edit only when Arduino is not running)

In Arduino, go to Preferences and add *https://sandeepmistry.github.io/arduino-nRF5/package_nRF5_boards_index.json* into the Additional Board Manager URL text box.

If this is not your first, make sure to separate URLs with a comma.


Boards Manager

Type: All | nrf5

Boards included in this package:
Arduino Primo.
Online help
More info

Adafruit nRF52 by **Adafruit**
Boards included in this package:
Adafruit Bluefruit nRF52 Feather.
Online help
More info

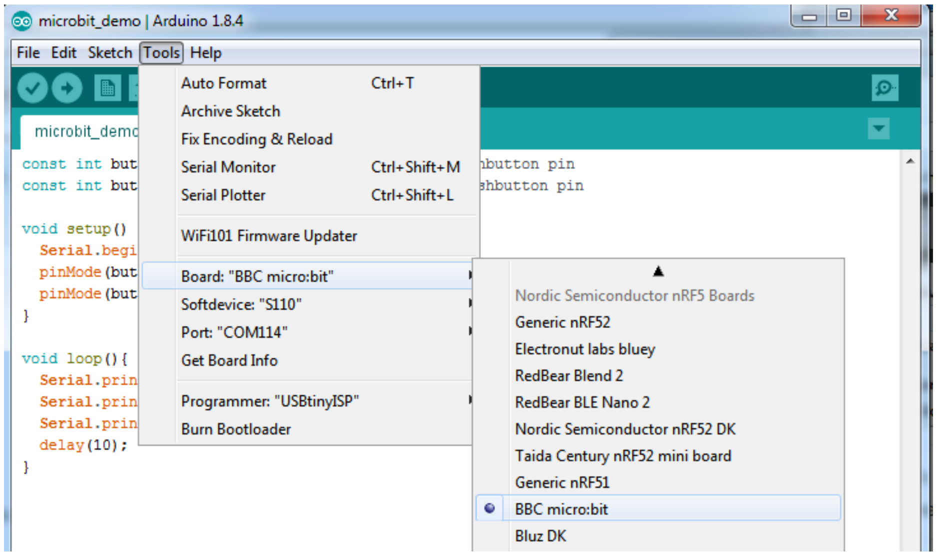
Nordic Semiconductor nRF5 Boards by **Sandeep Mistry** version **0.4.0** INSTALLED
Boards included in this package:
BBC micro:bit, Bluz DK, Generic nRF51822, Generic nRF52823, OSHChip, ng-beacon, nRF51 Dongle, nRF
nRF52 DK, RedBear BLE Nano 2, RedBear Blend 2, RedBearLab BLE Nano, RedBearLab nRF51822, Waves
Online help
More info

Select version | Install

Open **Tools>Board>Boards Manager** from the menu bar, search for **nRF5** and install "**Nordic Semiconductor nRF5 Boards**" by Sandeep Mistry

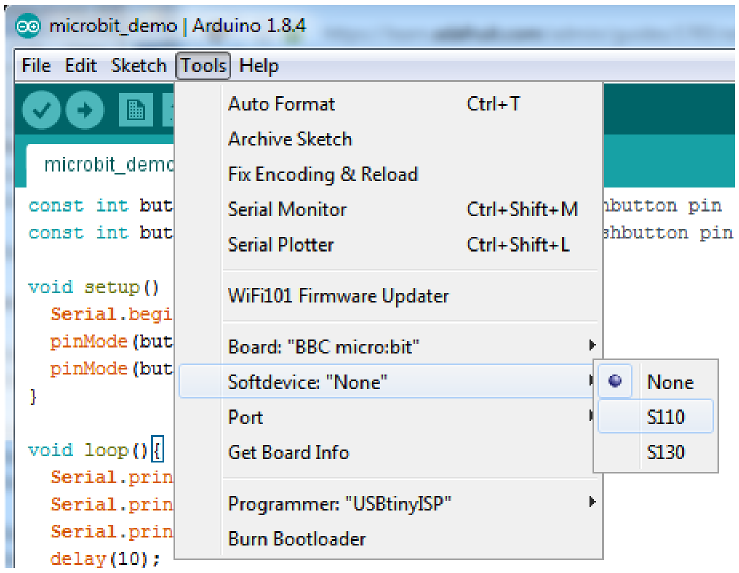
Select Board and Upload

Select BBC micro:bit from the Boards menu



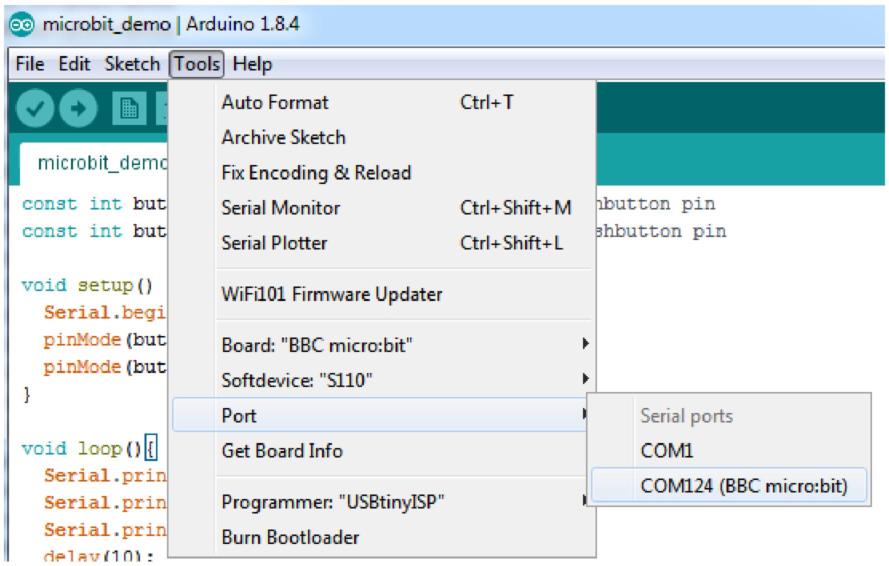
```
const int but                                              hbutton pin
const int but                                              ehbutton pin

void setup()
  Serial.begi
  pinMode(but
  pinMode(but
}

void loop(){
  Serial.prin
  Serial.prin
  Serial.prin
  delay(10);
}
```

Set SoftDevice to S110



```
const int but                                  nbutton pin
const int but                                  ehbutton pin

void setup()
  Serial.begi
  pinMode(but
  pinMode(but
}

void loop(){
  Serial.prin
  Serial.prin
  Serial.prin
  delay(10);
```

And set the Port to the microbit



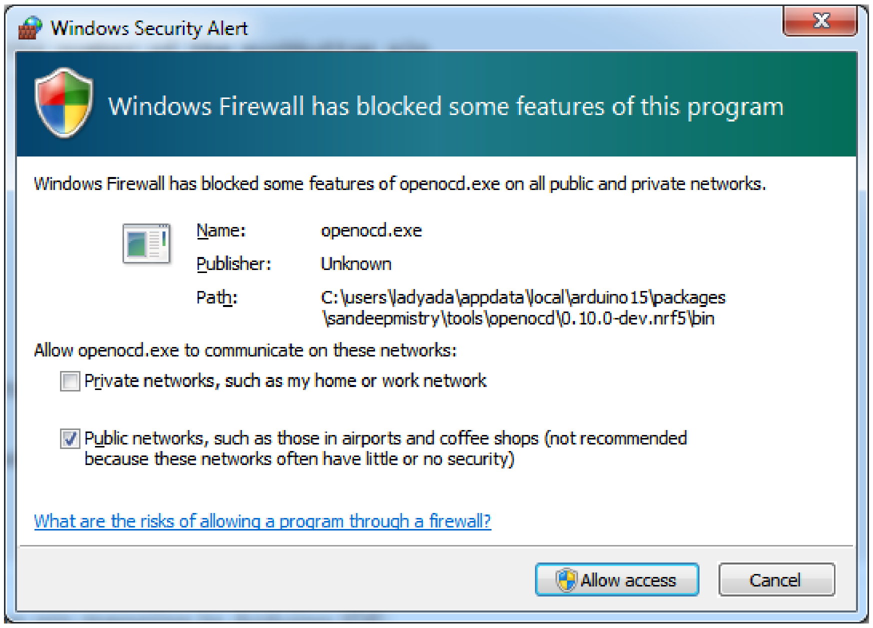
```
const int but                                  nbutton pin
const int but                                  ehbutton pin

void setup()
  Serial.begi
  pinMode(but
  pinMode(but
}

void loop(){
  Serial.prin
  Serial.prin
  Serial.prin
  delay(10);
```

And create a new sketch with this blink demo

```
const int COL1 = 3;     // Column #1 control
const int LED = 26;     // 'row 1' led

void setup() {
  Serial.begin(9600);

  Serial.println("microbit is ready!");

  // because the LEDs are multiplexed, we must ground the opposite side of the LED
  pinMode(COL1, OUTPUT);
  digitalWrite(COL1, LOW);

  pinMode(LED, OUTPUT);
}

void loop(){
  Serial.println("blink!");

  digitalWrite(LED, HIGH);
  delay(500);
  digitalWrite(LED, LOW);
  delay(500);
}
```

Click **Upload!**

If you get a warning about **openocd** - approve access so it can upload the code


Windows Security Alert

Windows Firewall has blocked some features of this program

Windows Firewall has blocked some features of openocd.exe on all public and private networks.

Name: openocd.exe
Publisher: Unknown
Path: C:\users\ladyada\appdata\local\arduino15\packages\sandeepmistry\tools\openocd\0.10.0-dev.nrf5\bin

Allow openocd.exe to communicate on these networks:

[ ] Private networks, such as my home or work network

[✓] Public networks, such as those in airports and coffee shops (not recommended because these networks often have little or no security)

What are the risks of allowing a program through a firewall?

You should see the top left LED blinking!

Linux Specific

The openocd build in the sandeepmistry package appears to be 32 bit. If you are running on a 64 bit linux install, you may get this error when trying to upload sketches:

> {arduino folder}/packages/sandeepmistry/tools/openocd/0.10.0-dev.nrf5/bin/openocd: error while loading shared libraries: libudev.so.1: cannot open shared object file: No such file or directory

To get around this, install the 32 bit version of the libudev library with:

```
sudo apt-get install libudev1:i386
```

You may also run into a permissions issue when trying to upload the sketch to the micro:bit. If you get an error message that ends with lines like this:

> Error: unable to open CMSIS-DAP device 0xd28:0x204
> Error: No Valid JTAG Interface Configured.
> Error: No Valid JTAG Interface Configured.

Then you'll need to add a udev rule. Place the contents below in a file named **/etc/udev/rules.d/99-microbit.rules**

```
ATTRS{idVendor}=="0d28", ATTRS{idProduct}=="0204", MODE="664", GROUP="plugdev"
```

And then plug the micro:bit back in for the settings to take effect.

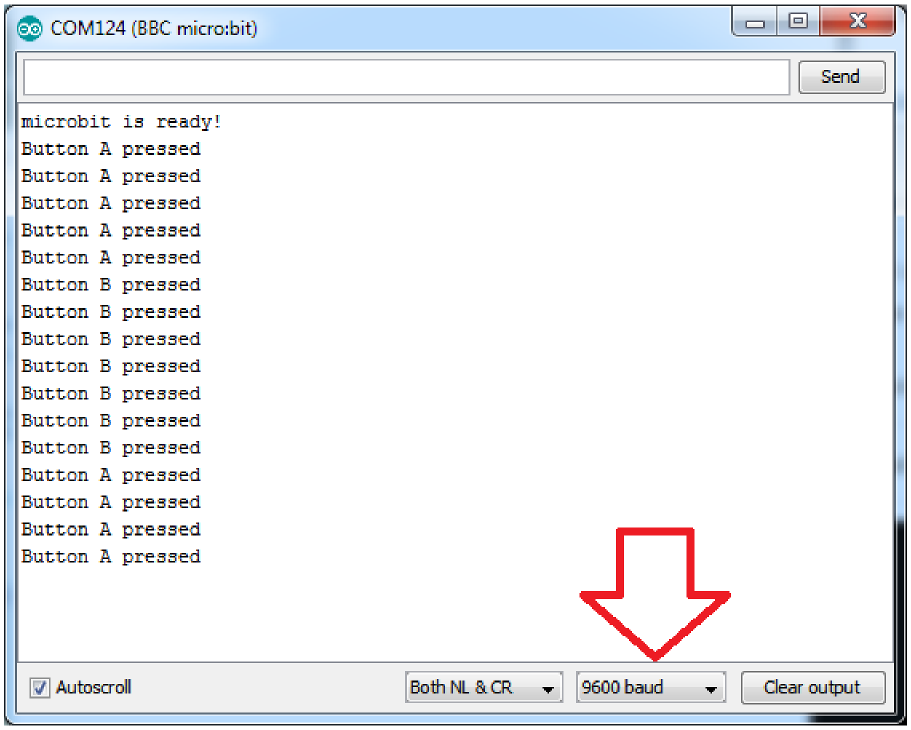
Buttons

Create a new sketch and upload the following code to read the button presses in the Serial Console

```
const int buttonA = 5;     // the number of the pushbutton pin
const int buttonB = 11;     // the number of the pushbutton pin

void setup() {
  Serial.begin(9600);

  Serial.println("microbit is ready!");

  pinMode(buttonA, INPUT);
  pinMode(buttonB, INPUT);
}

void loop(){
  if (! digitalRead(buttonA)) {
    Serial.println("Button A pressed");
  }
  if (! digitalRead(buttonB)) {
    Serial.println("Button B pressed");
  }
  delay(10);
}
```

Open up the Serial console at **9600 baud** and **press the reset button on the back of the microbit** to restart the software

You will see the welcome message and then try pressing some buttons!



```
microbit is ready!
Button A pressed
Button A pressed
Button A pressed
Button A pressed
Button A pressed
Button B pressed
Button B pressed
Button B pressed
Button B pressed
Button B pressed
Button B pressed
Button B pressed
Button A pressed
Button A pressed
Button A pressed
Button A pressed
```

Both NL & CR
9600 baud

Other GPIO

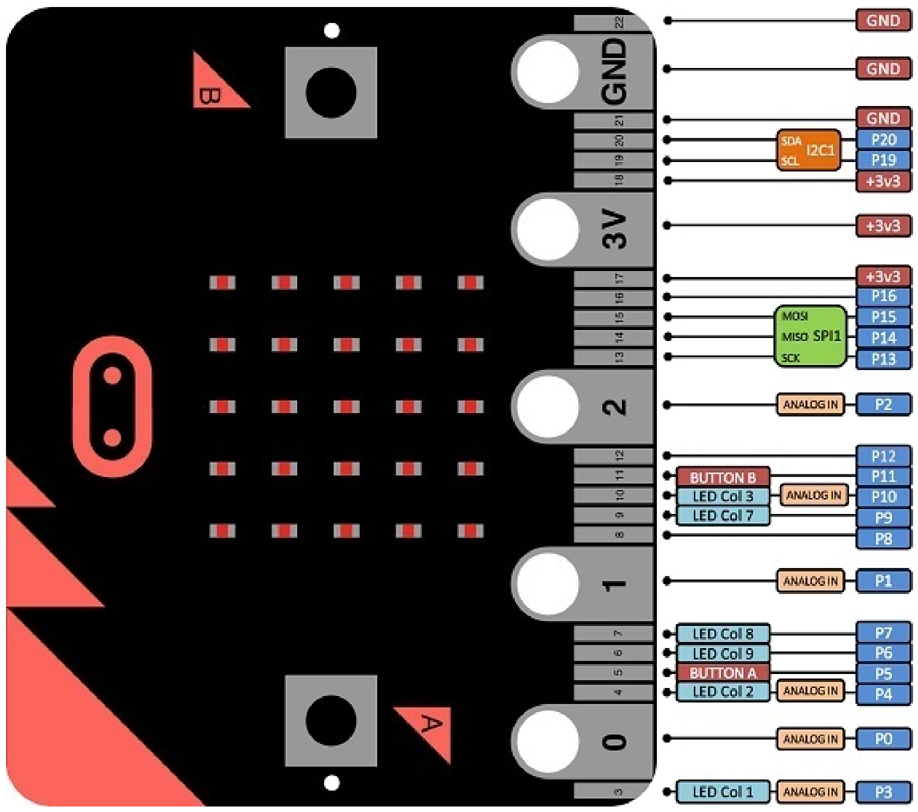
You can easily get to GPIO #0 #1 and #2 using large alligator clips. With a microbit breakout, you can access the other 16 or so.

Note that many of these pins are *multiplexed* with the LED matrix so if you decide to use that, you can't use them for analog inputs or digital I/O

- Pin #0 - large pad - analog in
- Pin #1 - large pad - analog in
- Pin #2 - large pad - analog in
- Pin #3 - analog in, also used for LED matrix
- Pin #4 - analog in, also used for LED matrix
- Pin #5 - also used for Button A
- Pin #6 - also used for LED matrix
- Pin #7 - also used for LED matrix
- Pin #8
- Pin #9 - also used for LED matrix
- Pin #10 - analog in, also used for LED matrix
- Pin #11 - also used for button B
- Pin #12
- Pin #13 - also available as SPI clock
- Pin #14 - also available as SPI MISO
- Pin #15 - also available as SPI MOSI
- Pin #16
- Pin #19 - also available as I2C clock
- Pin #20 - also available as I2C data

So really, if you're using the buttons and LEDs, you can use pins: #0, #1, #2, #8, #12, #13, #14, #15, #16, #19 and #20. Which is still a good amount!

blynk.cc (https://adafru.it/yLE) has a great graphic of the pin mapping to Arduino IDE:

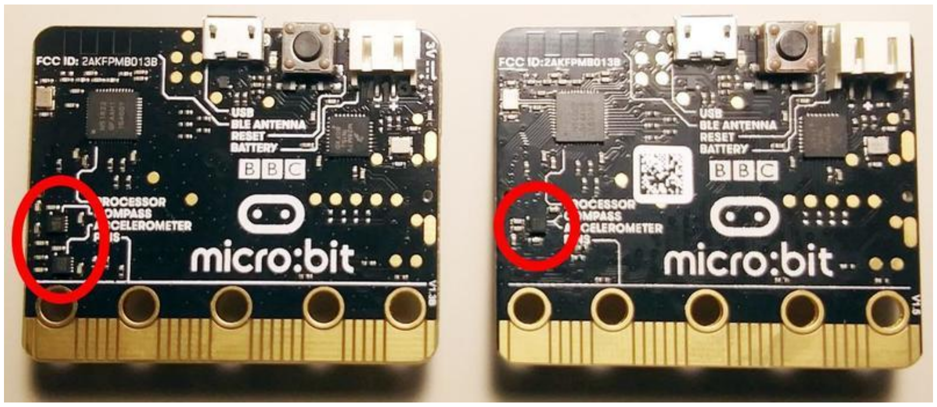


GND
GND
GND
P20
P19
+3v3
+3v3
+3v3
P16
P15
P14
P13
P2
P12
P11
P10
P9
P8
P1
P7
P6
P5
P4
P0
P3
SDA
SCL
I2C1
MOSI
MISO
SCK
SPI1
ANALOG IN
BUTTON B
LED Col 3
LED Col 7
LED Col 8
LED Col 9
BUTTON A
LED Col 2
LED Col 1
GND
3V
2
1
0
A
B

Accelerometer and Magnetometer

Check Board Version

The original version of the micro:bit has two separate sensors for the accelerometer and the magnetometer. Newer versions have a single sensor. To check which version you have, look on the board in the area shown below.



If you have **two sensors** then use the following libraries:

- Magnetometer = MAG3110
- Accelerometer = MMA8653

If you have a **single sensor**, then use the following library:

- Magno/Accelo = LSM303AGR

Magnetometer (MAG3110)

Lets start with the magnetometer chip, the MAG3110

https://adafru.it/z4B

https://adafru.it/z4B

We can talk to the chip using an Arduino library

You can download Sparkfun's library by clicking the button below! (https://adafru.it/z4C)

https://adafru.it/z4D

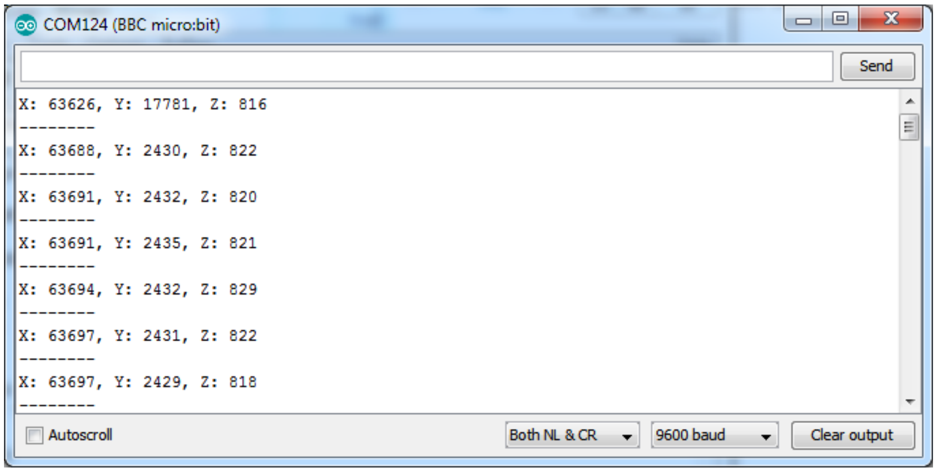
https://adafru.it/z4D

And read our guide on how to install libraries (https://adafru.it/nEO)

Restart the IDE. Now you can upload some examples (https://adafru.it/z4E). I suggest starting with the **Basic example** (https://adafru.it/z4F) which is replicated below

```
/*  ********************************************
 *  SparkFun_MAG3110_Basic
 *  Triple Axis Magnetometer Breakout - MAG3110
 *  Hook Up Guide Example
 *
 *  Utilizing Sparkfun's MAG3110 Library
 *  A basic sketch that reads x y and z readings
 *  from the MAG3110 sensor
 *
 *  George B. on behalf of SparkFun Electronics
 *  Created: Sep 22, 2016
 *  Updated: n/a
 *
 *  Development Environment Specifics:
 *  Arduino 1.6.7
 *
 *  Hardware Specifications:
 *  SparkFun MAG3110
 *  Bi-directional Logic Level Converter
 *  Arduino Micro
 *
 *  This code is beerware; if you see me (or any other SparkFun employee) at the
 *  local, and you've found our code helpful, please buy us a round!
 *  Distributed as-is; no warranty is given.
 *  ********************************************/

#include <SparkFun_MAG3110.h>

MAG3110 mag = MAG3110(); //Instantiate MAG3110

void setup() {
  Serial.begin(9600);

  mag.initialize(); //Initializes the mag sensor
  mag.start();      //Puts the sensor in active mode
}

void loop() {

  int x, y, z;
  //Only read data when it's ready
  if(mag.dataReady()) {
    //Read the data
    mag.readMag(&x, &y, &z);

    Serial.print("X: ");
    Serial.print(x);
    Serial.print(", Y: ");
    Serial.print(y);
    Serial.print(", Z: ");
    Serial.println(z);

    Serial.println("--------");
  }
}
```

Upload this to the microbit to see the following raw data:



```
X: 63626, Y: 17781, Z: 816
--------
X: 63688, Y: 2430, Z: 822
--------
X: 63691, Y: 2432, Z: 820
--------
X: 63691, Y: 2435, Z: 821
--------
X: 63694, Y: 2432, Z: 829
--------
X: 63697, Y: 2431, Z: 822
--------
X: 63697, Y: 2429, Z: 818
--------
```

Note that the magnetometer is not calibrated, so you'll get different numbers on XYZ *but* when you twist and rotate the mirobit the numbers should move up and down a bit! (This is why magnetometers must be calibrated)

Accelerometer (MMA8653)

The microbit has an onboard 3-axis accelerometer as well!

You can use this **akafugu MMA8653** to communicate with it:

https://adafru.it/z5a

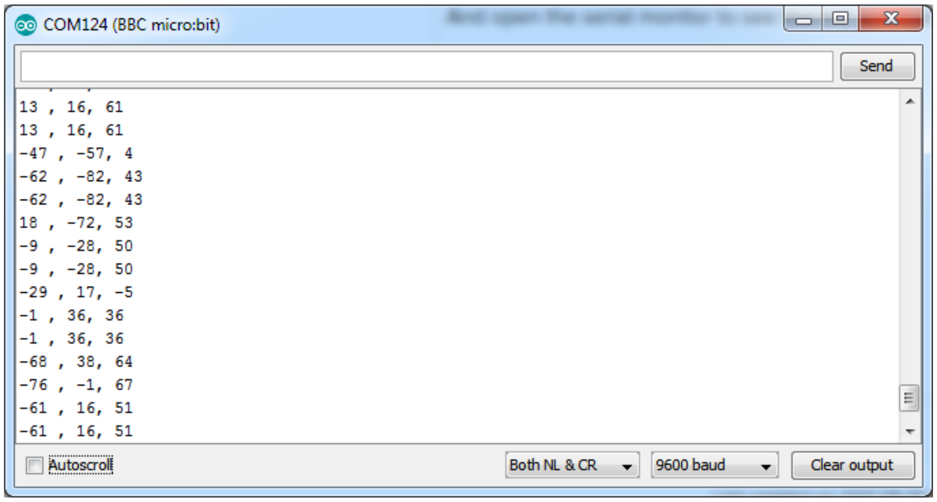
https://adafru.it/z5a

Install like other libraries!

Next up, run this example code:

```
/*
 * MMA845XQ test code
 * (C) 2012 Akafugu Corporation
 *
 * This program is free software; you can redistribute it and/or modify it under the
 * terms of the GNU General Public License as published by the Free Software
 * Foundation; either version 2 of the License, or (at your option) any later
 * version.
 *
 * This program is distributed in the hope that it will be useful, but WITHOUT ANY
 * WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A
 * PARTICULAR PURPOSE.  See the GNU General Public License for more details.
 *
 */


#include "Wire.h"
#include "MMA8653.h"

MMA8653 accel;

void setup() {
  Serial.begin(9600);

  Serial.println("microbit accel test");

  accel.begin(false, 2); // 8-bit mode, 2g range
}


void loop() {
  accel.update();
  Serial.print(accel.getX());    Serial.print(" , ");
  Serial.print(accel.getY());    Serial.print(", ");
  Serial.println(accel.getZ());

  delay(100);
}
```

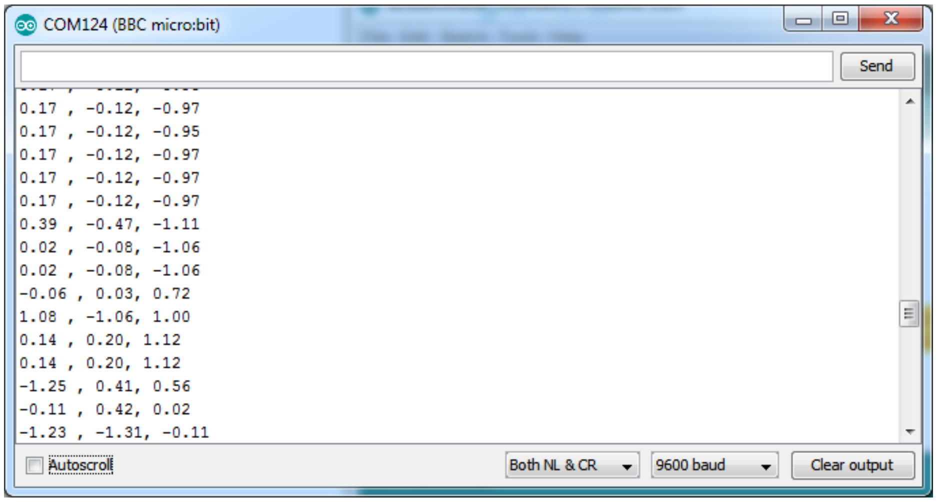
And open the serial monitor to see the X Y and Z acceleration data points!



```
13 , 16, 61
13 , 16, 61
-47 , -57, 4
-62 , -82, 43
-62 , -82, 43
18 , -72, 53
-9 , -28, 50
-9 , -28, 50
-29 , 17, -5
-1 , 36, 36
-1 , 36, 36
-68 , 38, 64
-76 , -1, 67
-61 , 16, 51
-61 , 16, 51
```

This library is pretty old and incomplete so at this time you can only use it in 8-bit mode. If you want to get the data in **g**'s use this for the loop:

```
void loop() {
  accel.update();
  Serial.print((float)accel.getX() * 0.0156);    Serial.print(" , ");
  Serial.print((float)accel.getY() * 0.0156);    Serial.print(", ");
  Serial.println((float)accel.getZ() * 0.0156);

  delay(100);
}
```



```
0.17 , -0.12, -0.97
0.17 , -0.12, -0.95
0.17 , -0.12, -0.97
0.17 , -0.12, -0.97
0.17 , -0.12, -0.97
0.39 , -0.47, -1.11
0.02 , -0.08, -1.06
0.02 , -0.08, -1.06
-0.06 , 0.03, 0.72
1.08 , -1.06, 1.00
0.14 , 0.20, 1.12
0.14 , 0.20, 1.12
-1.25 , 0.41, 0.56
-0.11 , 0.42, 0.02
-1.23 , -1.31, -0.11
```

Combined Accelerometer / Magnetometer (LSM303AGR)

Newer versions of the micro:bit have a single **LSM303AGR** sensor which has both an accelerometer and a magnetometer:

https://adafru.it/Fik

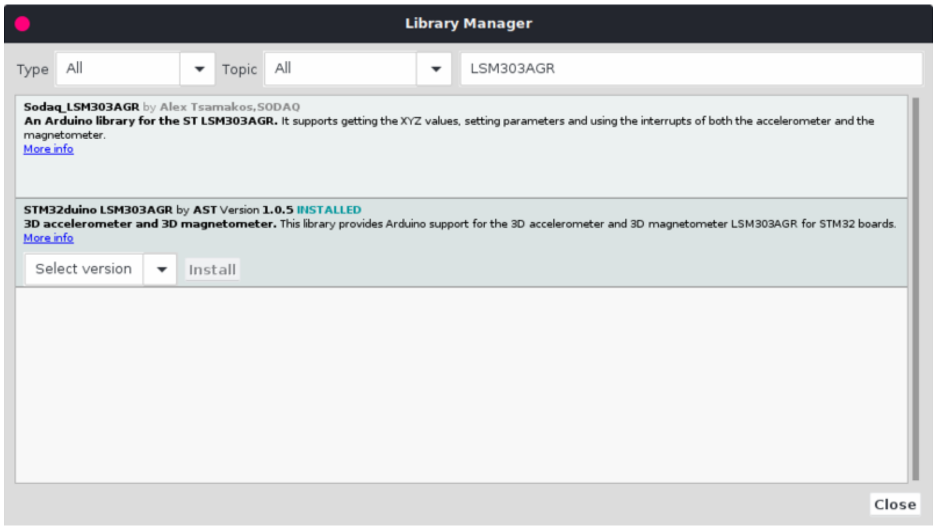
https://adafru.it/Fik

To use this version, you can use the LSM303AGR library from here:

https://adafru.it/Fil

https://adafru.it/Fil

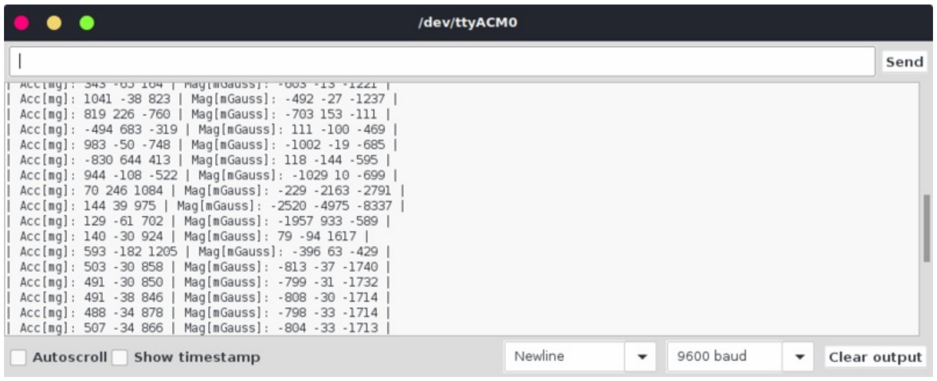
You can install it via the Library Manager. Just search for **LSM303AGR** and pick the one from **STM32duino**:



Sodaq_LSM303AGR by Alex Tsamakos,SODAQ
An Arduino library for the ST LSM303AGR. It supports getting the XYZ values, setting parameters and using the interrupts of both the accelerometer and the magnetometer.
More info

STM32duino LSM303AGR by **AST** Version **1.0.5** INSTALLED
3D accelerometer and 3D magnetometer. This library provides Arduino support for the 3D accelerometer and 3D magnetometer LSM303AGR for STM32 boards.
More info

The library comes with one example (https://adafru.it/Fin). Running it should give you the following output:



```
| Acc[mg]: 343 -63 164 | Mag[mGauss]: -603 -13 -1221 |
| Acc[mg]: 1041 -38 823 | Mag[mGauss]: -492 -27 -1237 |
| Acc[mg]: 819 226 -760 | Mag[mGauss]: -703 153 -111 |
| Acc[mg]: -494 683 -319 | Mag[mGauss]: 111 -100 -469 |
| Acc[mg]: 983 -50 -748 | Mag[mGauss]: -1002 -19 -685 |
| Acc[mg]: -830 644 413 | Mag[mGauss]: 118 -144 -595 |
| Acc[mg]: 944 -108 -522 | Mag[mGauss]: -1029 10 -699 |
| Acc[mg]: 70 246 1084 | Mag[mGauss]: -229 -2163 -2791 |
| Acc[mg]: 144 39 975 | Mag[mGauss]: -2520 -4975 -8337 |
| Acc[mg]: 129 -61 702 | Mag[mGauss]: -1957 933 -589 |
| Acc[mg]: 140 -30 924 | Mag[mGauss]: 79 -94 1617 |
| Acc[mg]: 593 -182 1205 | Mag[mGauss]: -396 63 -429 |
| Acc[mg]: 503 -30 858 | Mag[mGauss]: -813 -37 -1740 |
| Acc[mg]: 491 -30 850 | Mag[mGauss]: -799 -31 -1732 |
| Acc[mg]: 491 -38 846 | Mag[mGauss]: -808 -30 -1714 |
| Acc[mg]: 488 -34 878 | Mag[mGauss]: -798 -33 -1714 |
| Acc[mg]: 507 -34 866 | Mag[mGauss]: -804 -33 -1713 |
```

Autoscroll: ☐ Show timestamp: ☐ Newline 9600 baud

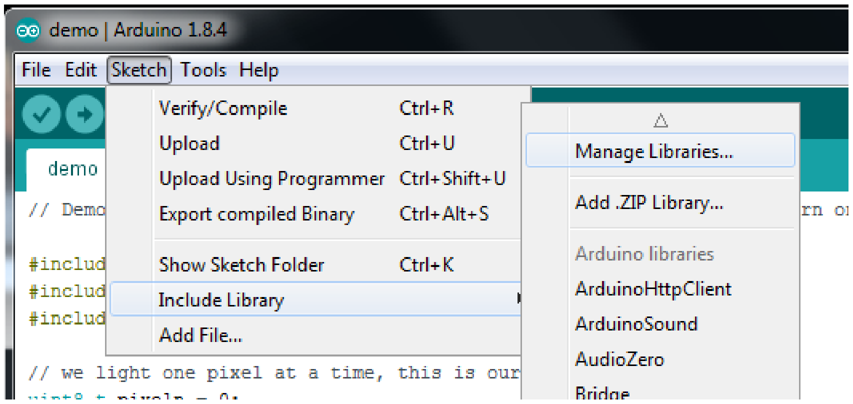
Adafruit Libraries

Once you want to get any more complex stuff going, you'll need a helper library to manage stuff like the internal temperature sensor, LED matrix, or Bluetooth connection.

To make your life easier, we've written up a wrapper library that manages all this stuff for you.

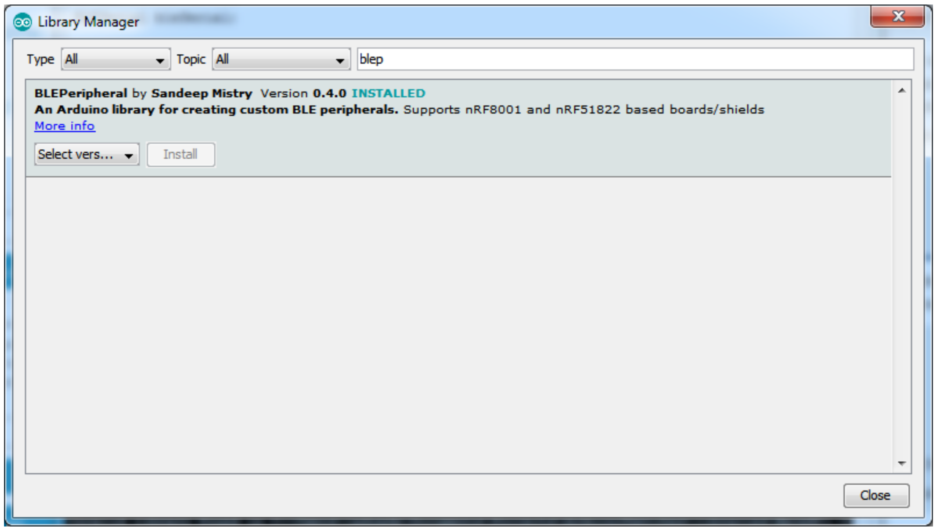
You'll also need to install some helpers.

Open up the Arduino library manager:



```
// Demo
#includ
#includ
#includ
// we light one pixel at a time, this is our
```

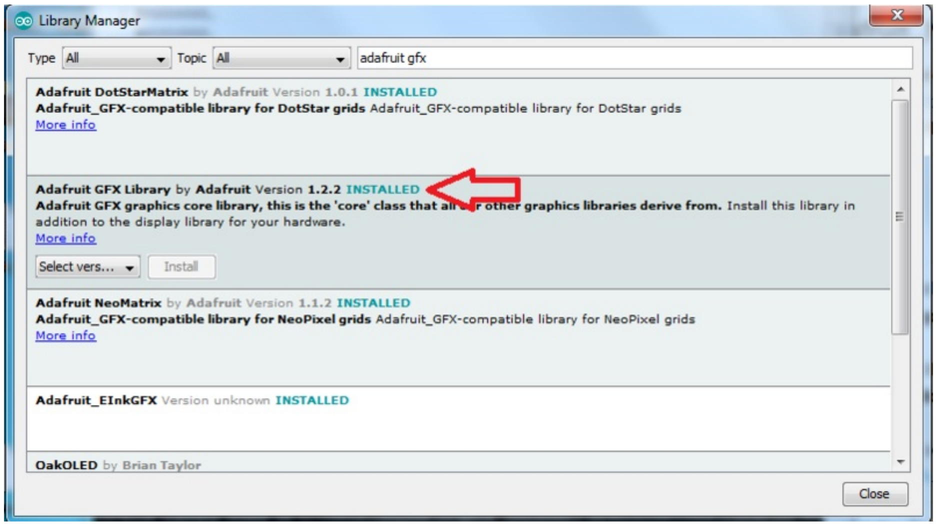
Search for the **BLEPeripheral** library and install it



Type All Topic All blep

BLEPeripheral by **Sandeep Mistry** Version **0.4.0** INSTALLED
An Arduino library for creating custom BLE peripherals. Supports nRF8001 and nRF51822 based boards/shields
More info

Search for the **Adafruit GFX** library and install it



Adafruit DotStarMatrix by Adafruit Version 1.0.1 INSTALLED
Adafruit_GFX-compatible library for DotStar grids Adafruit_GFX-compatible library for DotStar grids
More info

Adafruit GFX Library by Adafruit Version 1.2.2 INSTALLED
Adafruit GFX graphics core library, this is the 'core' class that all our other graphics libraries derive from. Install this library in addition to the display library for your hardware.
More info

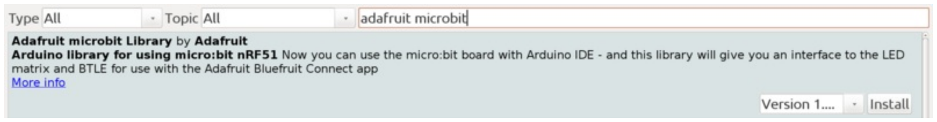
Adafruit NeoMatrix by Adafruit Version 1.1.2 INSTALLED
Adafruit_GFX-compatible library for NeoPixel grids Adafruit_GFX-compatible library for NeoPixel grids
More info

Adafruit_EInkGFX Version unknown INSTALLED

OakOLED by Brian Taylor

If using an earlier version of the Arduino IDE (prior to 1.8.10), also locate and install **Adafruit_BusIO** (newer versions will install this dependency automatically).

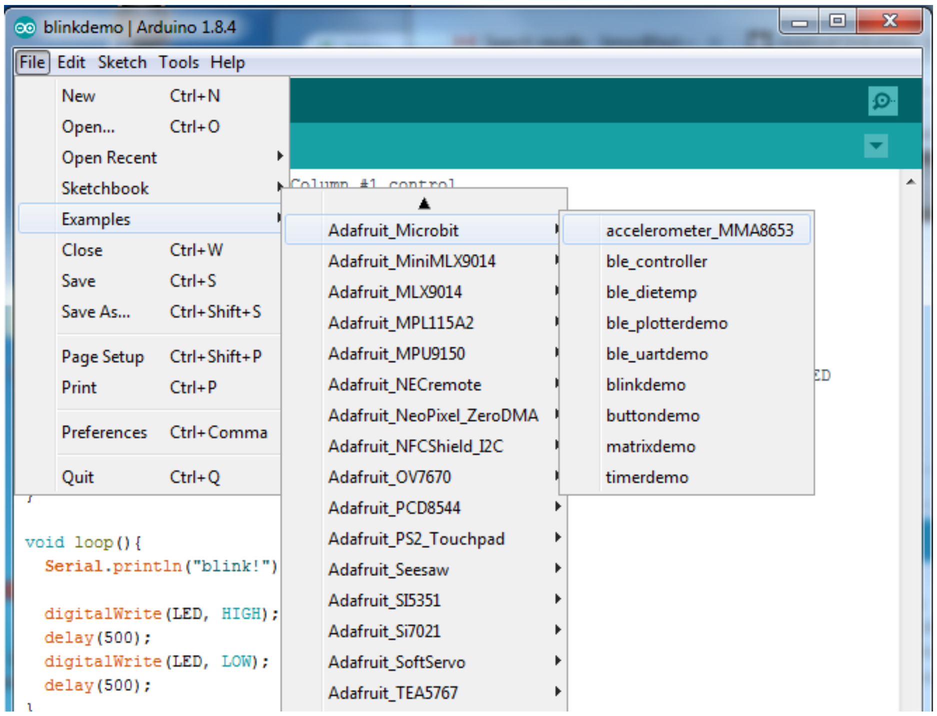
Search for the **Adafruit Microbit** library and install it



Adafruit microbit Library by Adafruit
Arduino library for using micro:bit nRF51 Now you can use the micro:bit board with Arduino IDE - and this library will give you an interface to the LED matrix and BTLE for use with the Adafruit Bluefruit Connect app
More info

We also have a great tutorial on Arduino library installation at:
http://learn.adafruit.com/adafruit-all-about-arduino-libraries-install-use (https://adafru.it/aYM)

Once you've re-started the Arduino IDE you should see the library examples appear in the **File->Examples->Adafruit_Microbit** menu



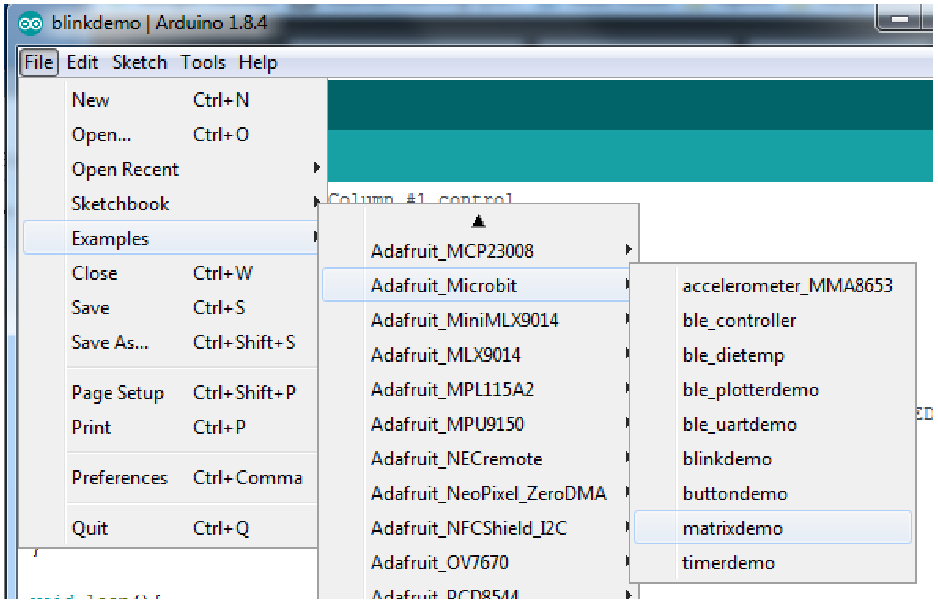
```
Column #1 control
}

void loop(){
  Serial.println("blink!")

  digitalWrite(LED, HIGH);
  delay(500);
  digitalWrite(LED, LOW);
  delay(500);
}
```

LED Matrix

the LED matrix is a 25-LED multiplexed array. You can't just set the LED's on or off, they must be scanned through continuously to keep them lit. To make this easy for you we've added support to the **Adafruit_Microbit** library - it even uses a timer (Timer 2) to manage all the multiplexing for you.

First up, install the Adafruit helper libraries (https://adafru.it/zqF)

Open up the **matrixdemo** sketch and upload it to your bit



The library manages all the refreshing, the drawing routines are all handled by **Adafruit_GFX** which we have documented in detail here https://learn.adafruit.com/adafruit-gfx-graphics-library/ (https://adafru.it/zra)

Note that the screen is small so even though the GFX library can do a lot, there's not a ton of space for complex shapes. We do have a small and simple font you can use that will fit in the 5-pixel-tall display, called **TomThumb**.h

We've overloaded the print function so you can use that to print scrolling text:

```
// scroll some text the 'easy' way
microbit.print("HELLO WORLD");
```

or even numbers and floating points

```
// count up!
for (int i=0; i<10; i++) {
  microbit.print(i);
  delay(500);
}
microbit.print(3.1415, 4);  // pi time, 4 digits of precision!!
```

Note that for single-digits or single-char `print()` 's you wont get scrolling text.

For bitmaps, the screen is 5x5 but to make the math easier on everyone, you can store bitmaps in 8-bit-wide structures and just ignore the left-most 3 bits in each 8-bit word:

```
const uint8_t
  smile_bmp[] =
  { B00000,
    B01010,
    B00000,
    B10001,
    B01110, };
```

We also have a range of built in images you can use here at the bottom (https://adafru.it/CgH).

You can draw bitmaps with `show()` - pass in either one of the built in `microbit.BITMAP` 's or your own custom bitmap

```
// draw a heart
microbit.show(microbit.HEART);
delay(1000);

// draw a no cross
microbit.show(microbit.NO);
delay(1000);

// draw a yes check
microbit.show(microbit.YES);
delay(1000);

// draw a custom made bitmap face
microbit.show(smile_bmp);
delay(1000);
```

Bluetooth UART

The main chip has a bluetooth LE radio built in, which is how you can make cool wireless projects!

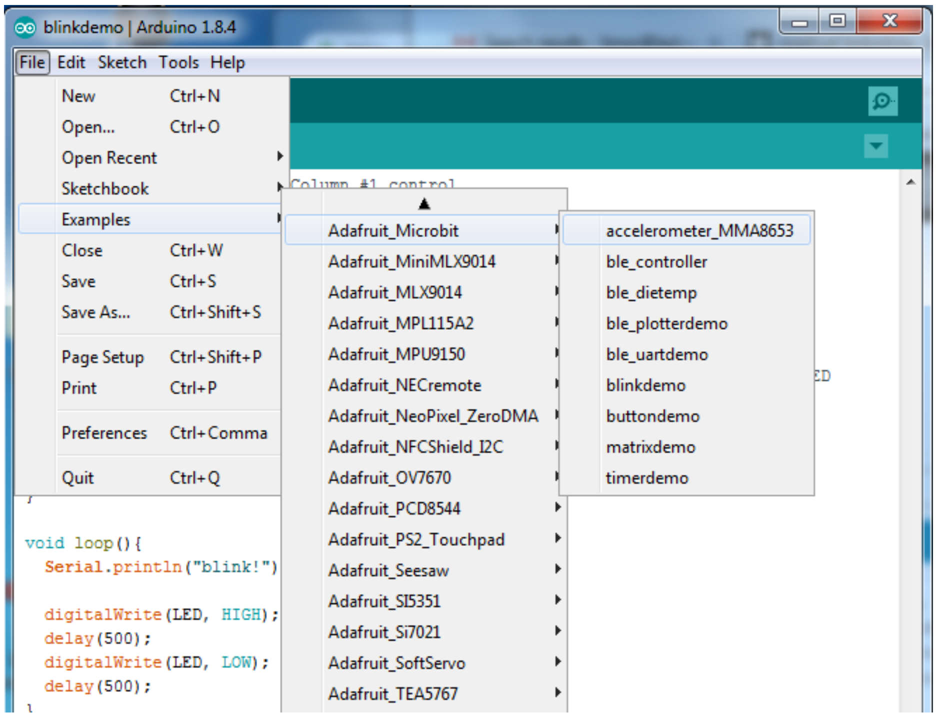
You can use the radio with our Adafruit Bluefruit Connect app without too much difficulty! You can download **Bluefruit Connect** in both the iOS App store (https://adafru.it/ddu) and Android app stores (https://adafru.it/f4G)

Learn more about our app over at the Connect guide, (https://adafru.it/wsD)we'll assume you've read thru it so you have a rough idea how it works

Install Library & Example Code

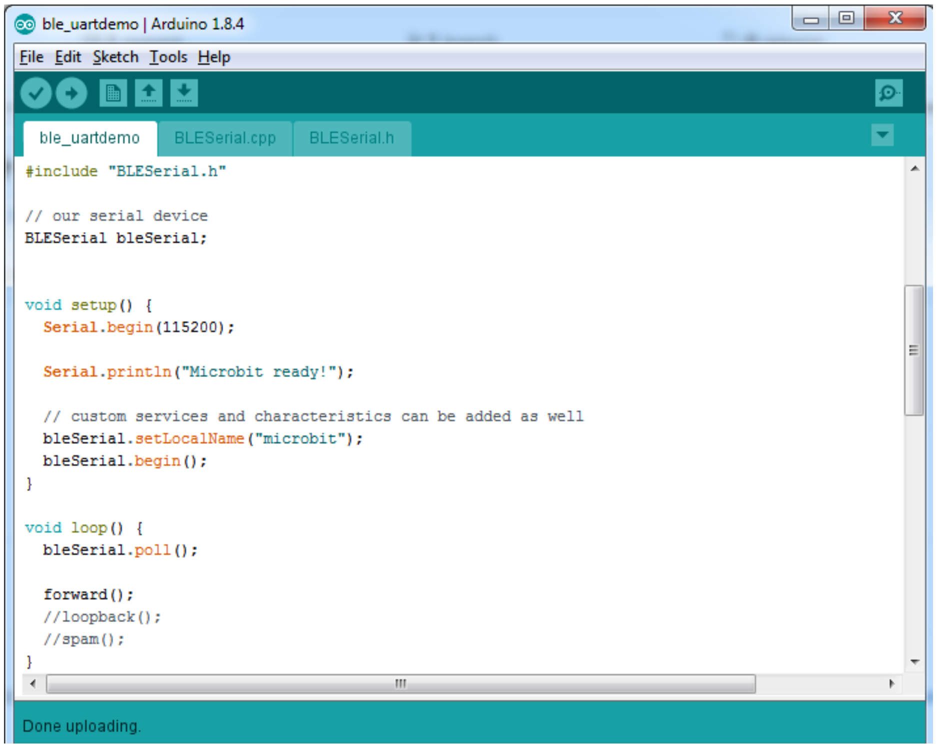
First up, install the Adafruit helper library (https://adafru.it/zqF) and friends

You can find our BLE demos in the examples menu:



```
void loop(){
  Serial.println("blink!")

  digitalWrite(LED, HIGH);
  delay(500);
  digitalWrite(LED, LOW);
  delay(500);
```

Load up the BLE UART demo to start



```
#include "BLESerial.h"

// our serial device
BLESerial bleSerial;


void setup() {
  Serial.begin(115200);

  Serial.println("Microbit ready!");

  // custom services and characteristics can be added as well
  bleSerial.setLocalName("microbit");
  bleSerial.begin();
}

void loop() {
  bleSerial.poll();

  forward();
  //loopback();
  //spam();
}
```

Done uploading.

Find these three lines:

```
  forward();
  //loopback();
  //spam();
```

and change them to:

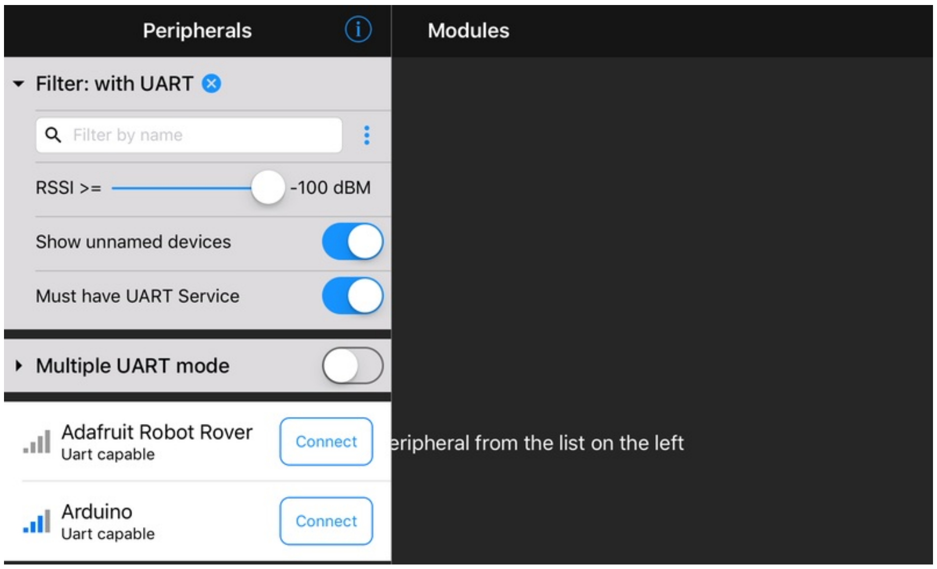
```
//forward();
loopback();
spam();
```

This will turn on auto-transmitting data once a second which will make testing easier. Then upload the sketch

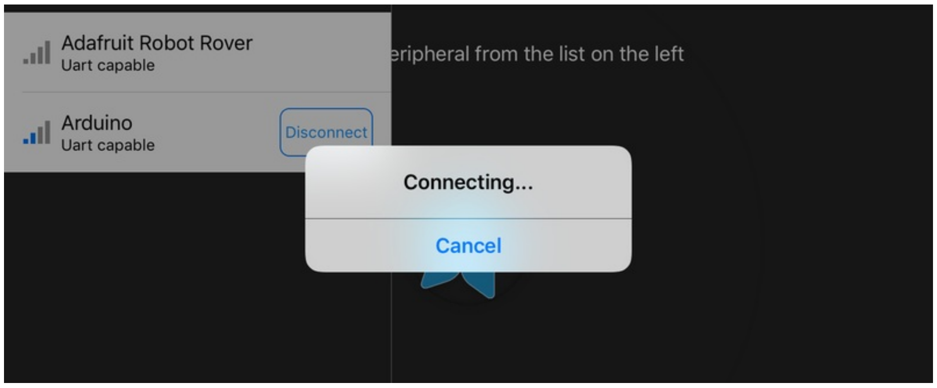
Bluetooth Connection

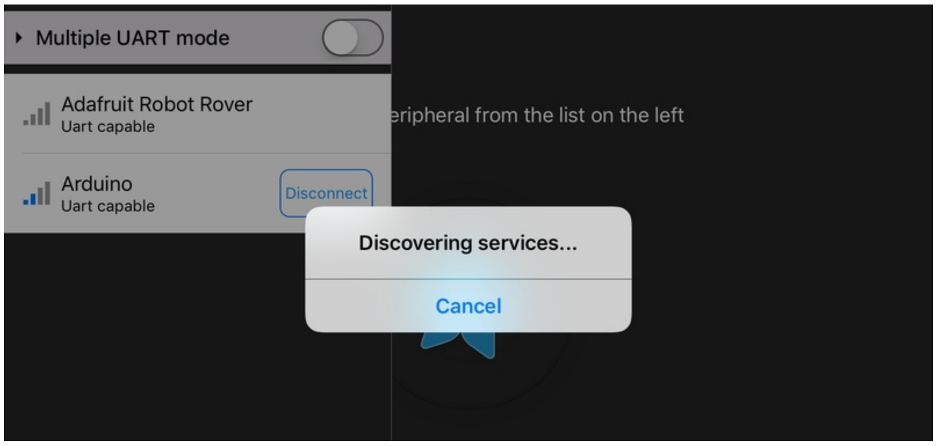
Once you have the sketch on the microbit, open up the Adafruit Bluefruit Connect app.

On the left there's a menu you can open. Select the microbit, it might be named **UART** or **Arduino**

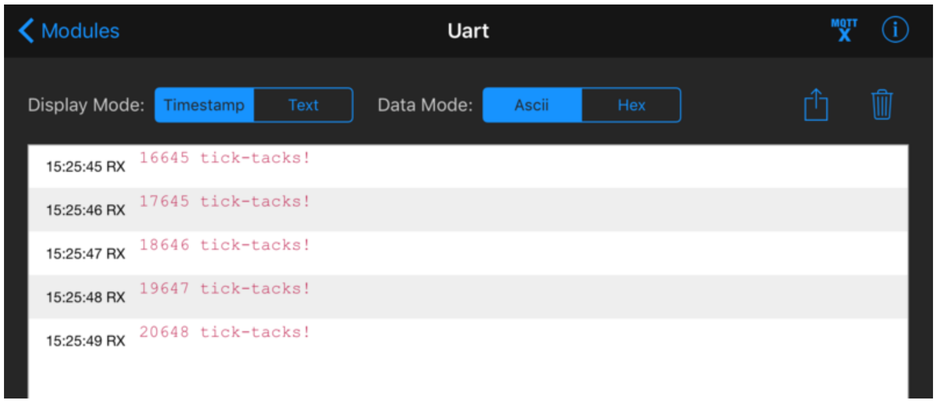


Press **Connect**





Then select **UART** from the list of Modules. Go into **Timestamp** mode and you should see messages once a second:



```
15:25:45 RX 16645 tick-tacks!
15:25:46 RX 17645 tick-tacks!
15:25:47 RX 18646 tick-tacks!
15:25:48 RX 19647 tick-tacks!
15:25:49 RX 20648 tick-tacks!
```

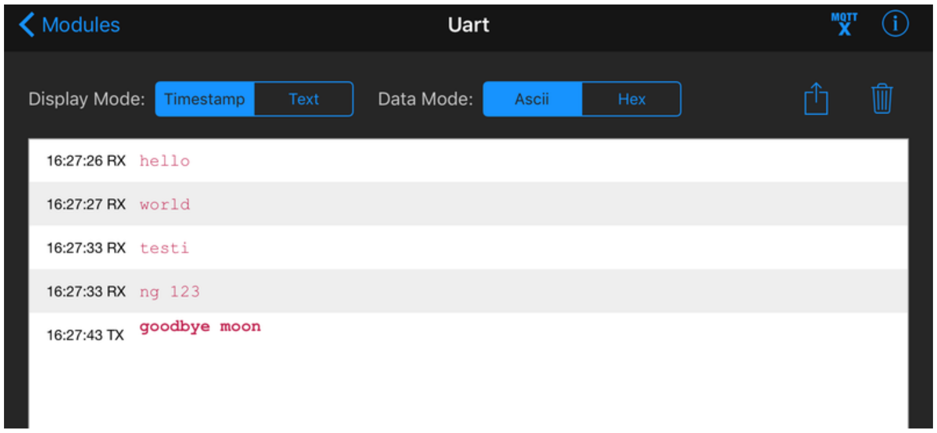
Go back to the sketch and change it back to:

```
forward();
//loopback();
//spam();
```

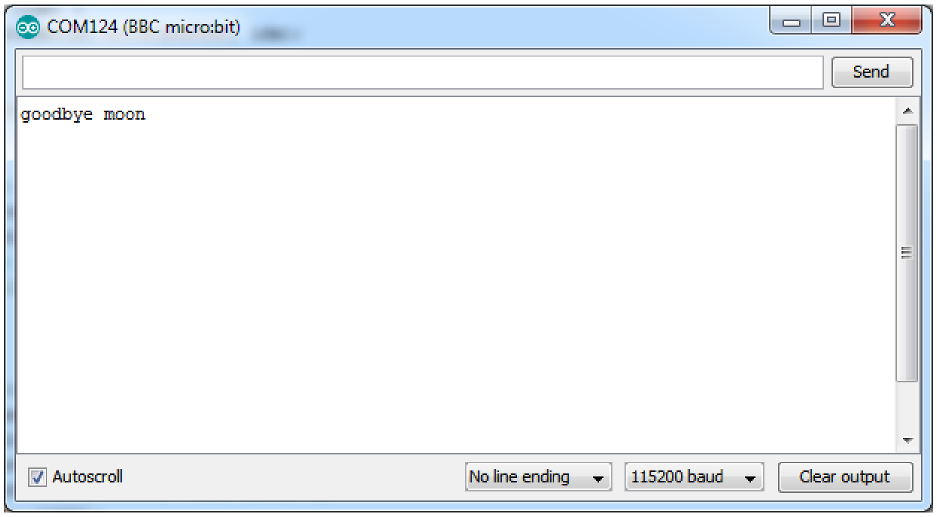
Re-upload. The app will complain you disconnected, just go back and disconnect from the peripheral-list menu.

Open the serial console at **115200 baud**

Then when you go back to UART mode, you can send data from the tablet to the bit and back. Note that the microbit's UART is a little odd - don't send more than 10 or so characters 'at a time' through the serial monitor or it may hang.



```
16:27:26 RX hello
16:27:27 RX world
16:27:33 RX testi
16:27:33 RX ng 123
16:27:43 TX goodbye moon
```



```
goodbye moon
```

Autoscroll ✓ | No line ending | 115200 baud

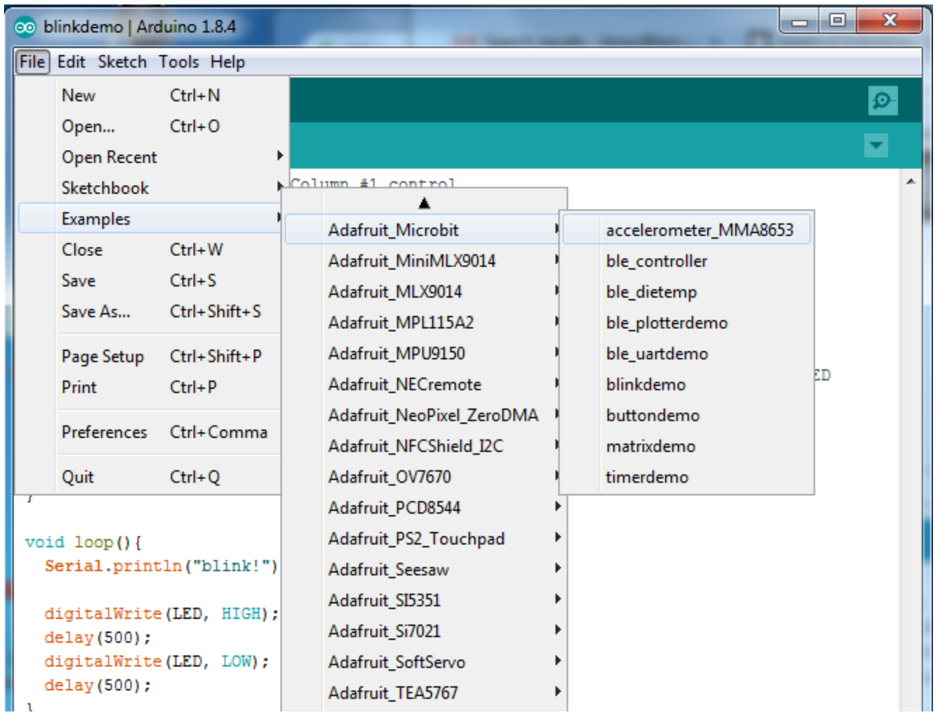
Once you've got all that working, you can try our controller sketch!

The Bluefruit App has a built in plotter that will let you easily visualize data from your microbit! Be sure you got the UART examples working from earlier.

Install Library & Example Code

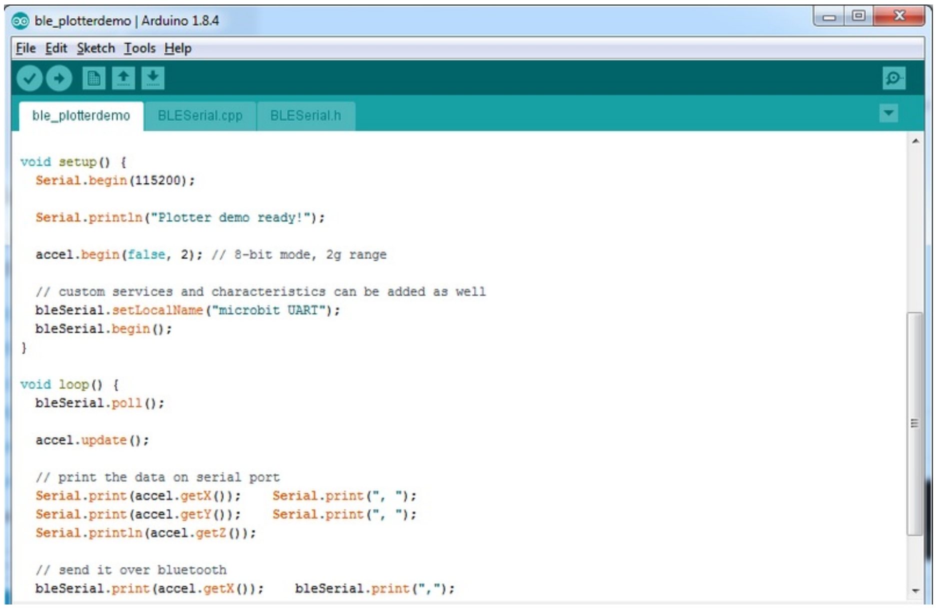
First up, install the Adafruit helper library (https://adafru.it/zqF) and friends

You can find our BLE demos in the examples menu:



```
void loop(){
  Serial.println("blink!")

  digitalWrite(LED, HIGH);
  delay(500);
  digitalWrite(LED, LOW);
  delay(500);
```

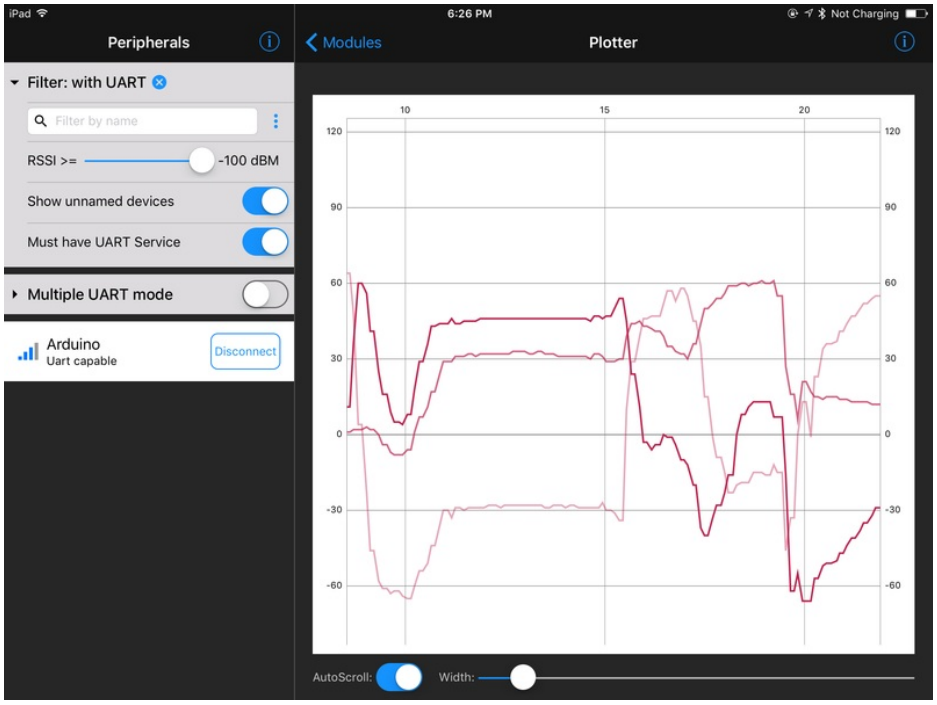
Load up the BLE Plotter demo



```
void setup() {
  Serial.begin(115200);

  Serial.println("Plotter demo ready!");

  accel.begin(false, 2); // 8-bit mode, 2g range

  // custom services and characteristics can be added as well
  bleSerial.setLocalName("microbit UART");
  bleSerial.begin();
}

void loop() {
  bleSerial.poll();

  accel.update();

  // print the data on serial port
  Serial.print(accel.getX());    Serial.print(", ");
  Serial.print(accel.getY());    Serial.print(", ");
  Serial.println(accel.getZ());

  // send it over bluetooth
  bleSerial.print(accel.getX());    bleSerial.print(",");
```

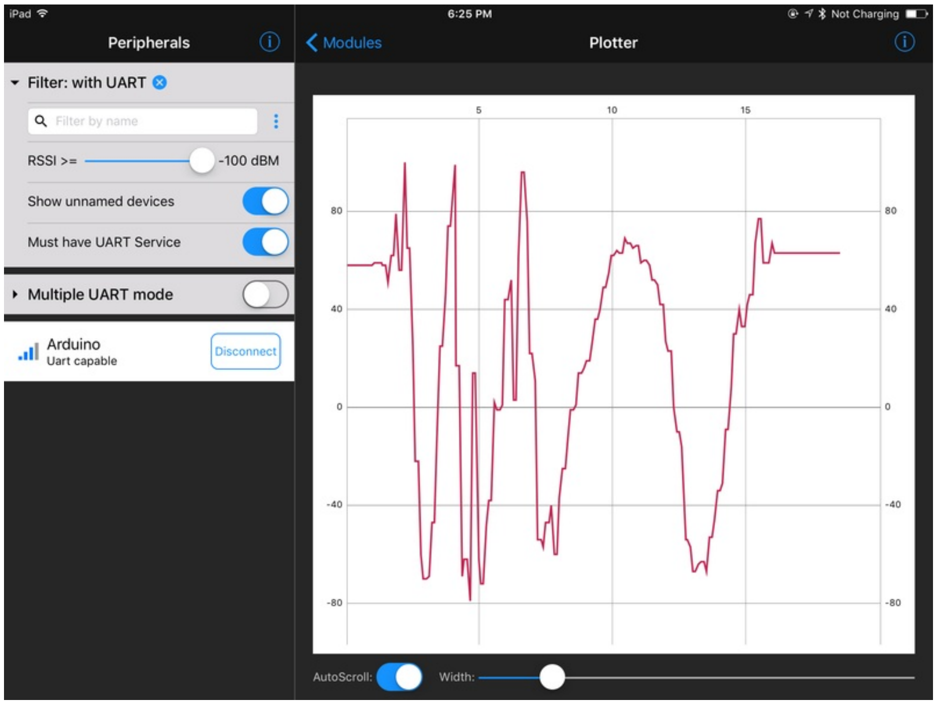
This time, in the App, select the **Plotter** module. You will be able to see the X, Y and Z data appear and scroll down!

You can plot anything you like, just use **bleSerial.print()** and print out your data with commas in between. At the end of a data set have a **bleSerial.println()** and it will plot each comma-separated-element as a unique graph



So if you want to just graph the total acceleration vector `sqrt(x^2 + y^2 + z^2)`, use this code snippet:

```
void loop() {
  bleSerial.poll();

  accel.update();

  // print the data on serial port
  Serial.print(accel.getX());    Serial.print(", ");
  Serial.print(accel.getY());    Serial.print(", ");
  Serial.println(accel.getZ());

  float vector = (accel.getX() * accel.getX()) + (accel.getY() * accel.getY()) +  (accel.getZ() *
accel.getZ());
  vector = sqrt(vector);

  // send it over bluetooth
  bleSerial.println(vector);

  delay(100);
}
```


iPad
6:25 PM
Not Charging
Peripherals
Modules
Plotter
Filter: with UART
Filter by name
RSSI >=
-100 dBM
Show unnamed devices
Must have UART Service
Multiple UART mode
Arduino
Uart capable
Disconnect
AutoScroll:
Width:

Bluetooth Controller

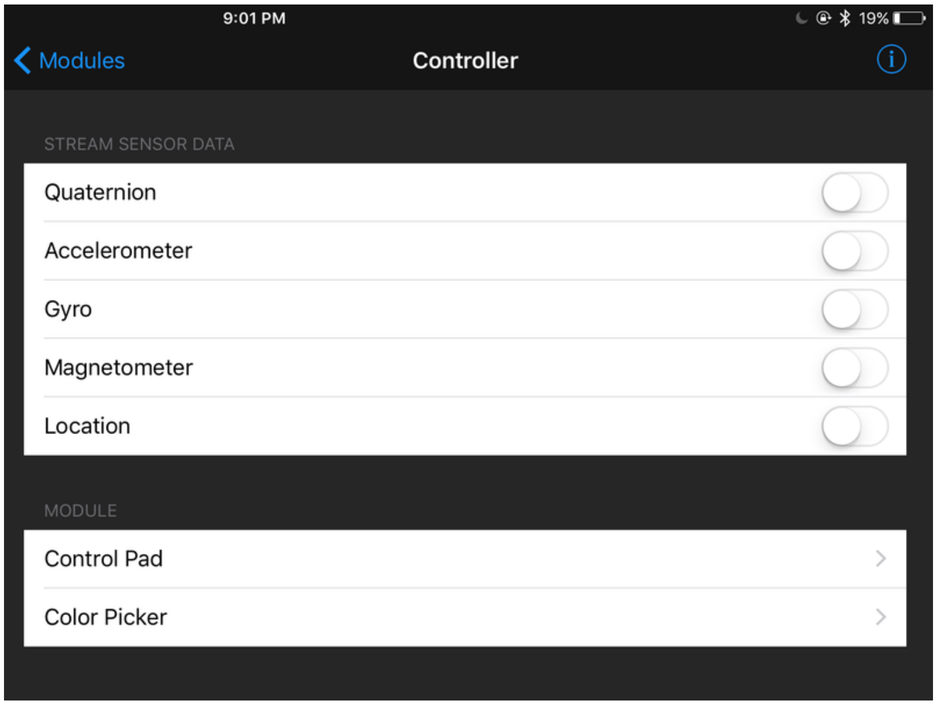
For controlling projects, you may want to use our **Controller** module. It has a bunch of useful interface features that will let you make your next LED or robotics project super easy

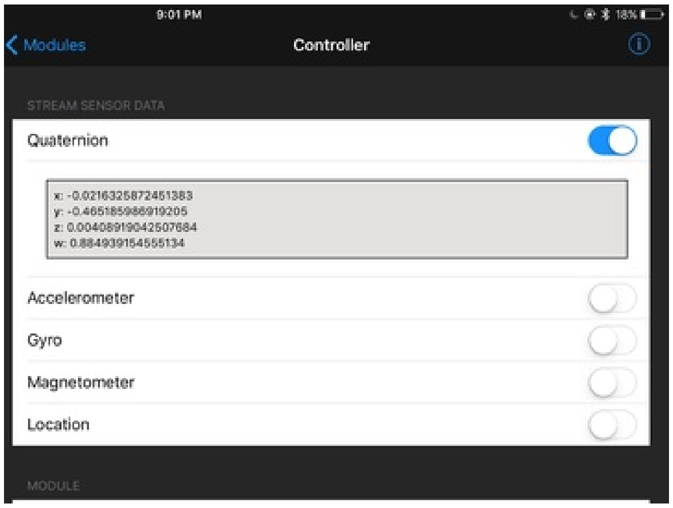
Install Library & Example Code

Install the Adafruit helper library (https://adafru.it/zqF) and friends

Open up the BLE Controller demo

Load that into your microbit, and connect using BLE connect





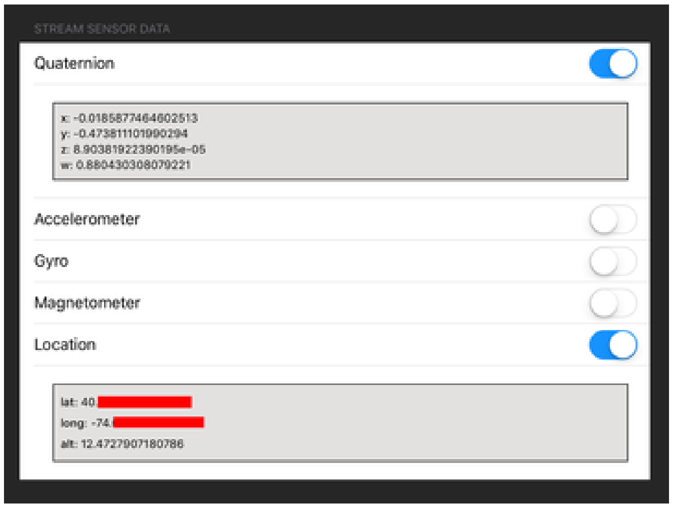
STREAM SENSOR DATA

Quaternion: on

```
x: -0.0216325872451383
y: -0.465185986919205
z: 0.00408919042507684
w: 0.884939154555134
```

Accelerometer: off
Gyro: off
Magnetometer: off
Location: off

MODULE



STREAM SENSOR DATA

Quaternion: on

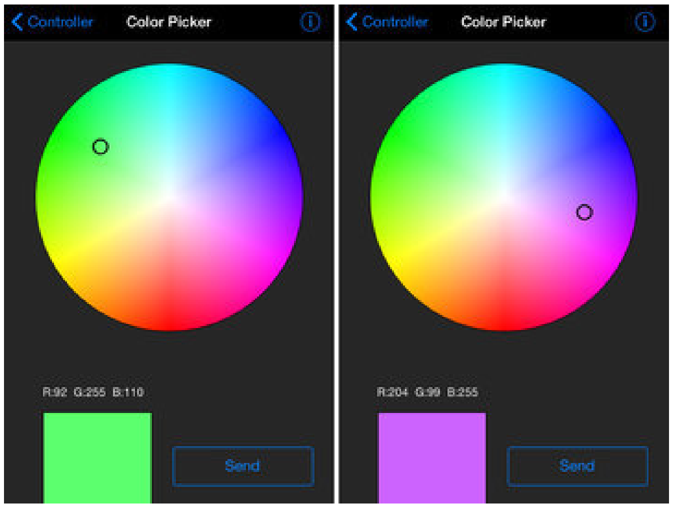
```
x: -0.0185877464602513
y: -0.473811101990294
z: 8.90381922390195e-05
w: 0.880430308079221
```

Accelerometer: off
Gyro: off
Magnetometer: off
Location: on

```
lat: 40.[redacted]
long: -74.[redacted]
alt: 12.4727907180786
```

The top 5 selectors allow you to stream data from your tablet or phone to the 'bit. So for example you can send tablet orientation (Quaternion) or GPS location to the 'bit. Turn on one, all five or none.

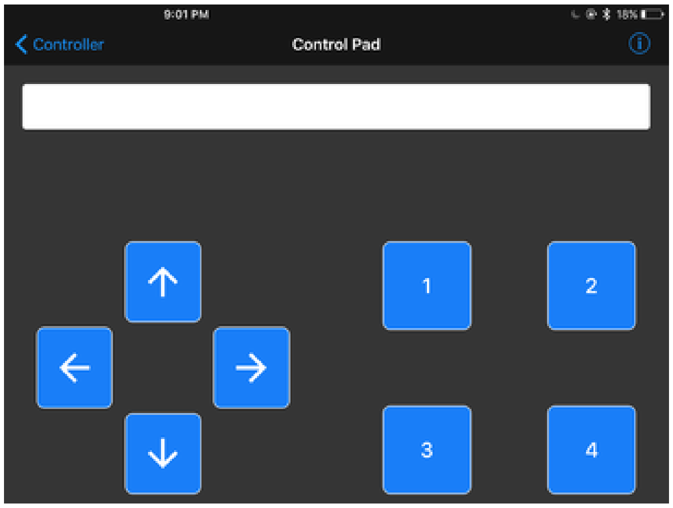
The two bottom modules can be run whenever you like, click to open up the interfaces:



R:92 G:255 B:110

R:204 G:99 B:255

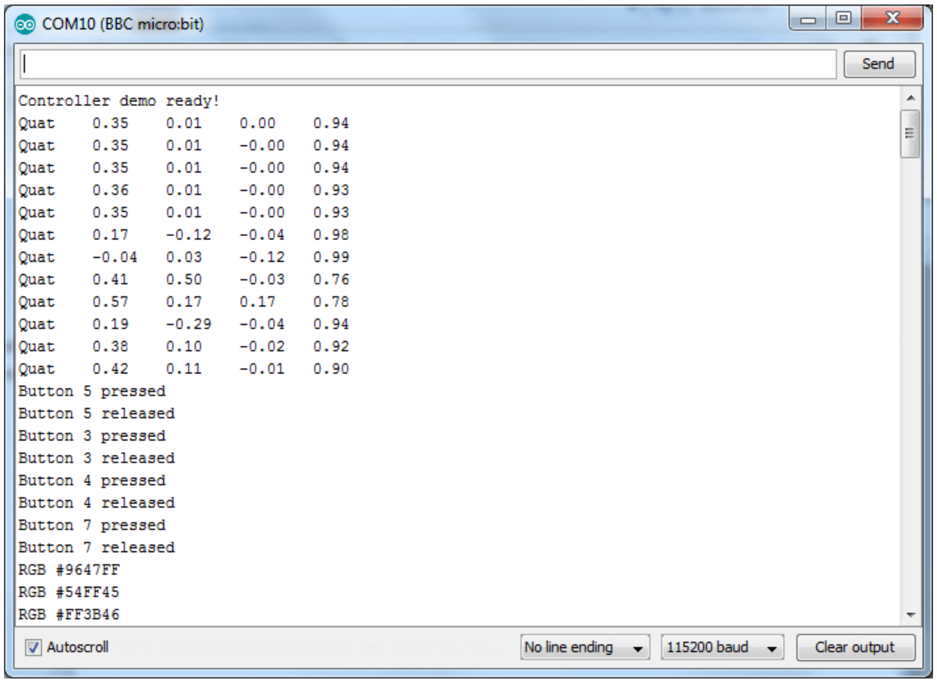
The color picker will let you choose from a color wheel and send the 24-bit color over BLE to the microbit



The control pad interface gives you 8 buttons that you can press - each press and release will send a signal to the microbit.

If the microbit sends any data *back* to the device, it will appear in the text bar above.

You can look at the serial monitor to see the messages as they are received.



```
Controller demo ready!
Quat    0.35    0.01    0.00    0.94
Quat    0.35    0.01    -0.00   0.94
Quat    0.35    0.01    -0.00   0.94
Quat    0.36    0.01    -0.00   0.93
Quat    0.35    0.01    -0.00   0.93
Quat    0.17    -0.12   -0.04   0.98
Quat    -0.04   0.03    -0.12   0.99
Quat    0.41    0.50    -0.03   0.76
Quat    0.57    0.17    0.17    0.78
Quat    0.19    -0.29   -0.04   0.94
Quat    0.38    0.10    -0.02   0.92
Quat    0.42    0.11    -0.01   0.90
Button 5 pressed
Button 5 released
Button 3 pressed
Button 3 released
Button 4 pressed
Button 4 released
Button 7 pressed
Button 7 released
RGB #9647FF
RGB #54FF45
RGB #FF3B46
```

Logging Temperature to Adafruit IO

All this Bluetooth data stuff is good if you want to plot the data or add control from your phone. But what if you want to store the data long term, or add remote control from around the world?

It's not too hard! We can use Adafruit IO to create graphs and dashboards. And, best of all, its free just like the Bluefruit app!

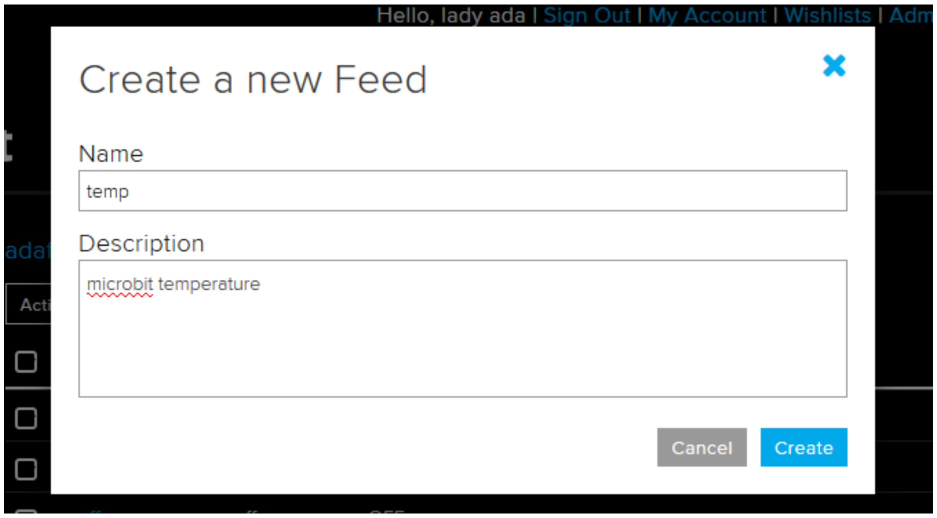
You can read more about Adafruit IO in this guide (https://adafru.it/nEP)

Before continuing, set yourself up with an Adafruit IO account (https://adafru.it/fsU)

We won't cover all the details of Adafruit IO here, so check out the guides we have already written for that good stuff!

Create a Microbit Temperature Feed

We'll want a 'place' for our temperature, so create a new feed called **temp**



Name: temp

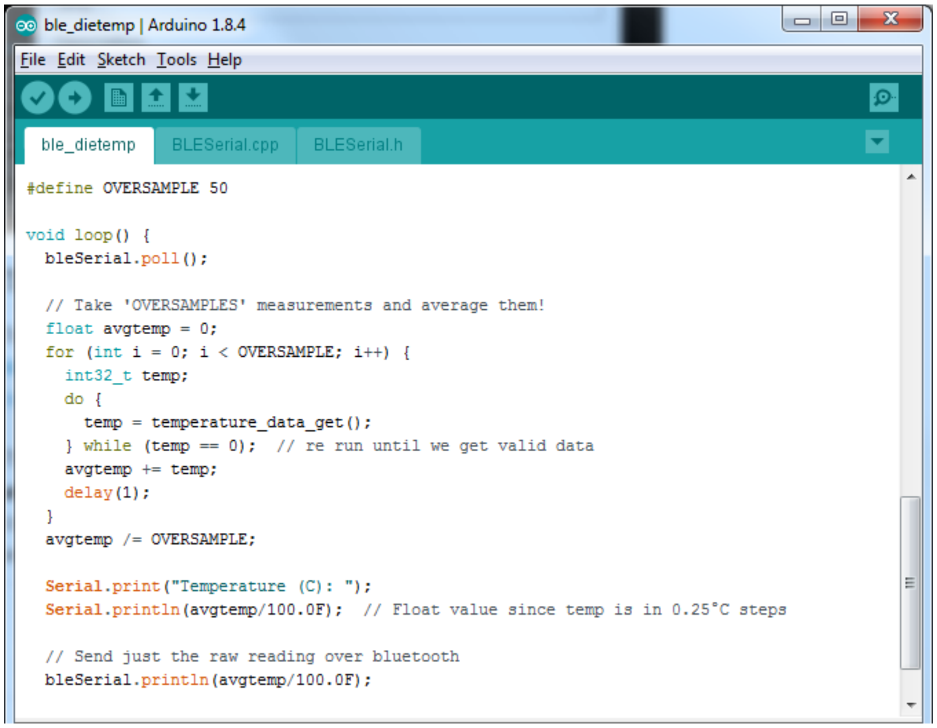
Description: microbit temperature

Temperature Logger Sketch

Install the Adafruit helper library (https://adafru.it/zqF) and friends

Open up the BLE die temp demo

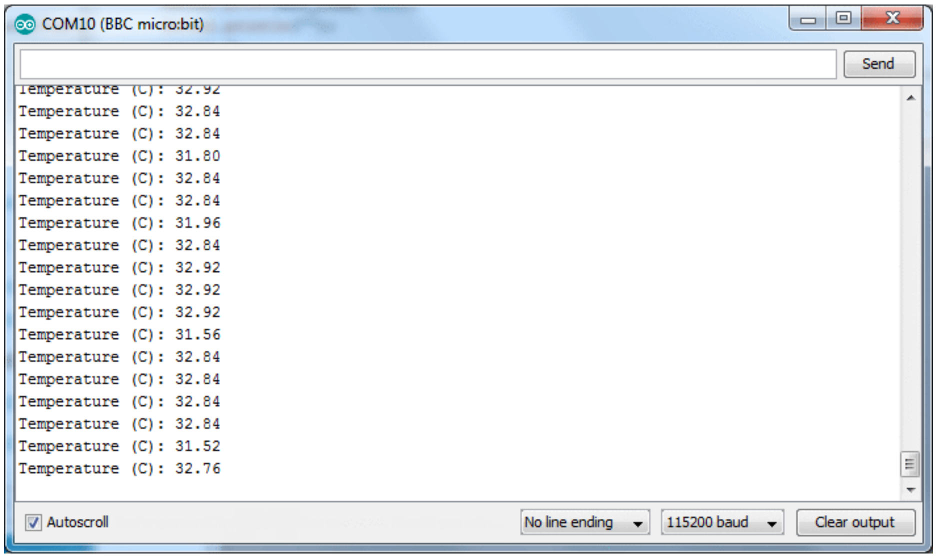
This will read the temperature on the chip itself. **It's not precise at all** but it does go up when it gets hotter and down when it gets cooler, so its a good place to start and you don't need any additional hardware



```
#define OVERSAMPLE 50

void loop() {
  bleSerial.poll();

  // Take 'OVERSAMPLES' measurements and average them!
  float avgtemp = 0;
  for (int i = 0; i < OVERSAMPLE; i++) {
    int32_t temp;
    do {
      temp = temperature_data_get();
    } while (temp == 0);  // re run until we get valid data
    avgtemp += temp;
    delay(1);
  }
  avgtemp /= OVERSAMPLE;

  Serial.print("Temperature (C): ");
  Serial.println(avgtemp/100.0F);  // Float value since temp is in 0.25°C steps

  // Send just the raw reading over bluetooth
  bleSerial.println(avgtemp/100.0F);
```

Note that this sketch takes 50 readings and averages it, then waits 5000 ms (5 seconds) between data reports. That's because Adafruit IO is limited in how much data you can upload and store, so we will take it a little slowly.

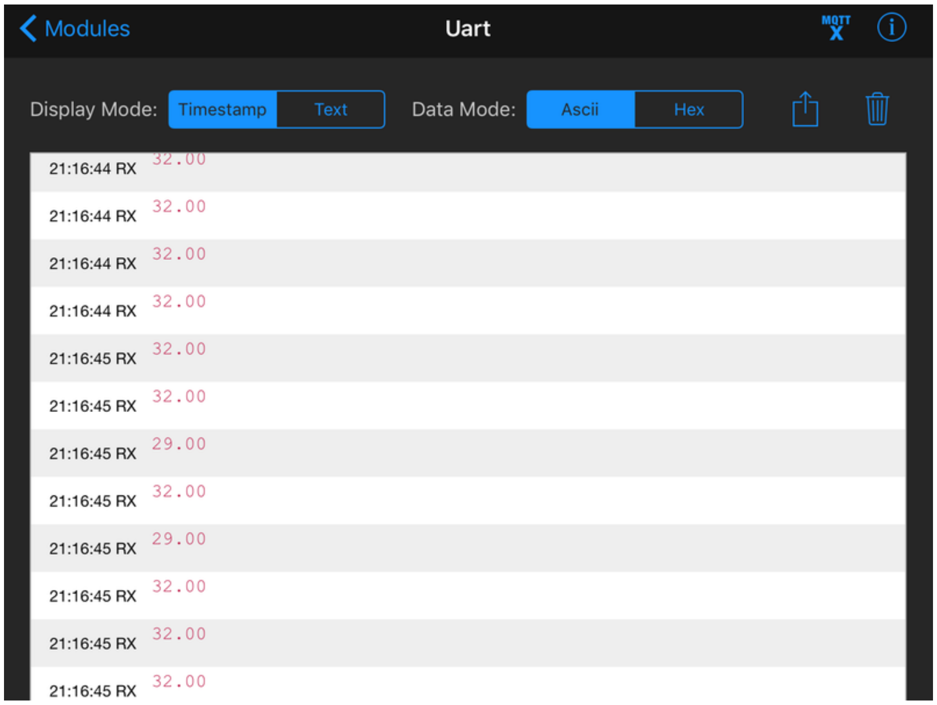
Upload the sketch and open the serial monitor so you can verify the temperature data there:



```
Temperature (C): 32.92
Temperature (C): 32.84
Temperature (C): 32.84
Temperature (C): 31.80
Temperature (C): 32.84
Temperature (C): 32.84
Temperature (C): 31.96
Temperature (C): 32.84
Temperature (C): 32.92
Temperature (C): 32.92
Temperature (C): 32.92
Temperature (C): 31.56
Temperature (C): 32.84
Temperature (C): 32.84
Temperature (C): 32.84
Temperature (C): 32.84
Temperature (C): 31.52
Temperature (C): 32.76
```

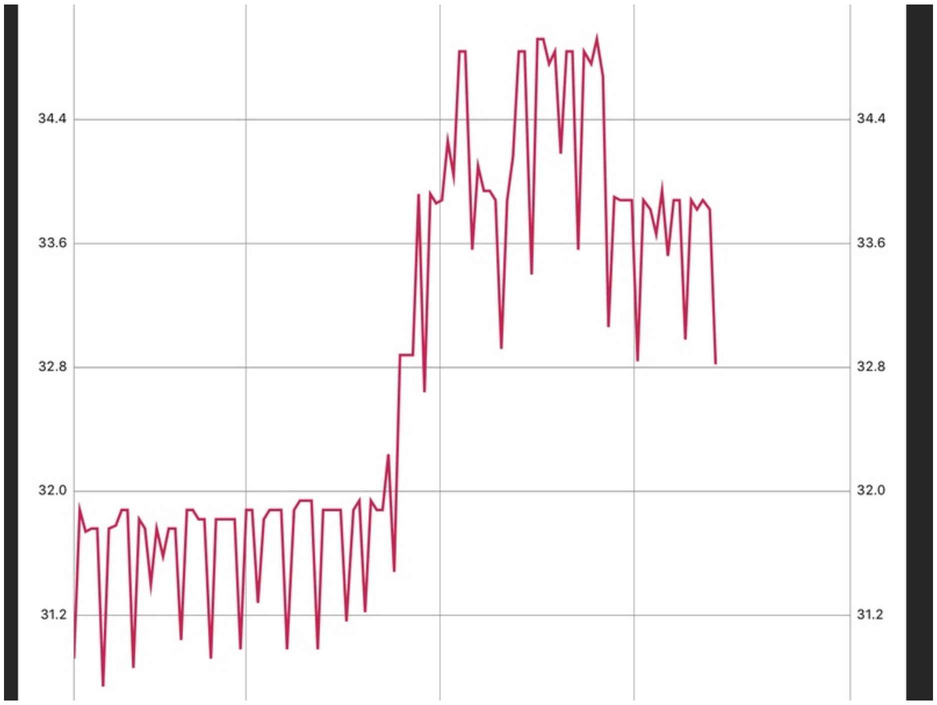
Test UART Mode

Connect to the microbit over your device using Adafruit Bluefruit Connect as covered in the previous projects, and select **UART** mode. You should see data slowly coming in

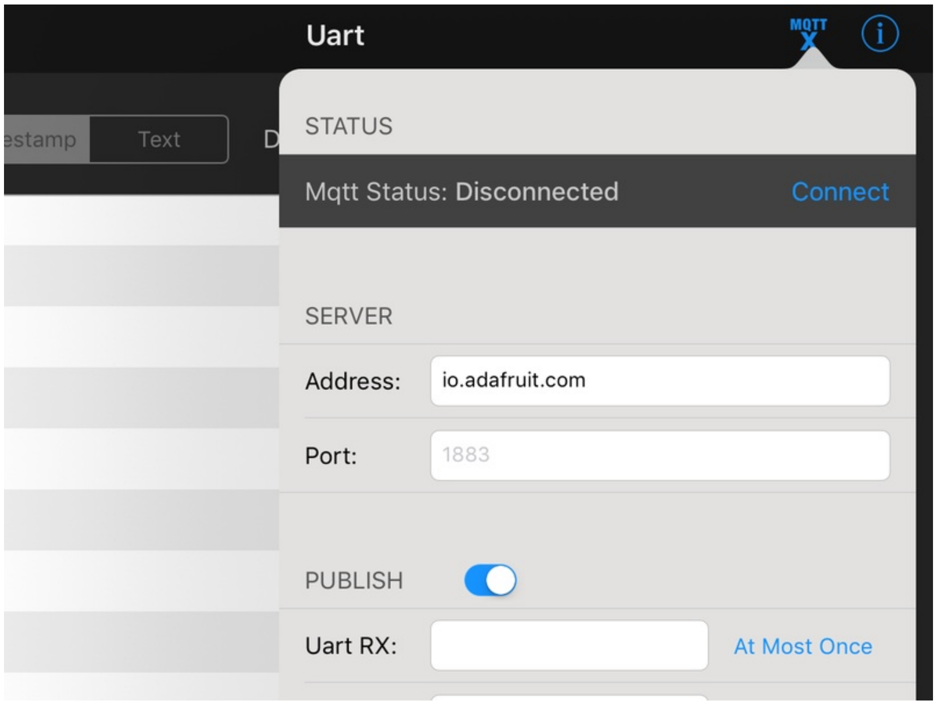


```
21:16:44 RX 32.00
21:16:44 RX 32.00
21:16:44 RX 32.00
21:16:44 RX 32.00
21:16:45 RX 32.00
21:16:45 RX 32.00
21:16:45 RX 29.00
21:16:45 RX 32.00
21:16:45 RX 29.00
21:16:45 RX 32.00
21:16:45 RX 32.00
21:16:45 RX 32.00
```

You can also plot the data. Note that the data is really not very precise or accurate. But if you heat up the nRF51 with a lamp, the temperature will slowly rise up:



Now go back to UART mode and click the **MQTT** button in the top right:



Mqtt Status: Disconnected

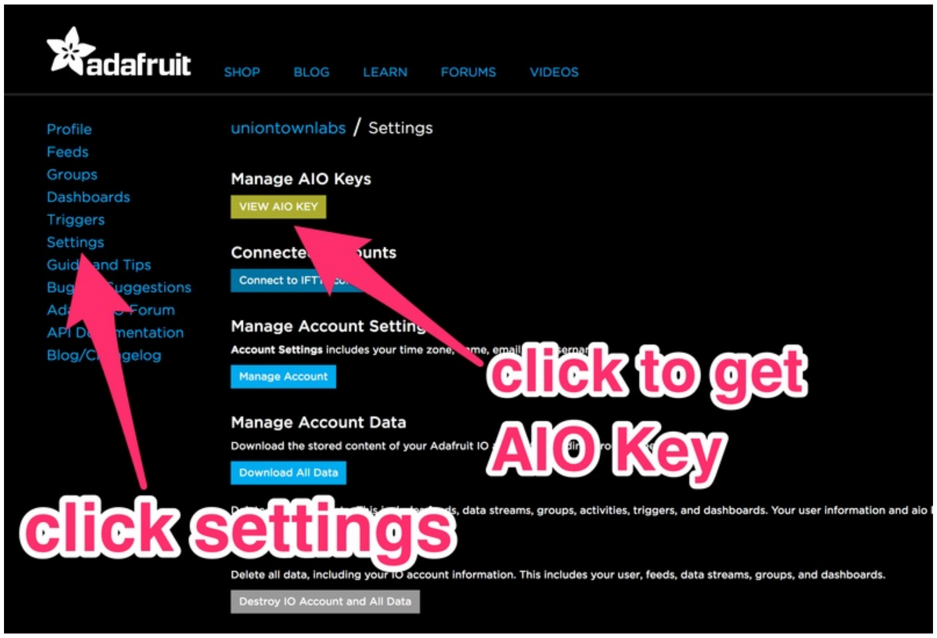
Address: io.adafruit.com

Port: 1883

Uart RX:

Note that the MQTT server and port will be prefilled for Adafruit IO.

Skip down and enter in your Adafruit IO **Username** and **API Key** (even though it says Password, use the long alphanumeric API key)



uniontownlabs / Settings

Manage AIO Keys

Connecte unts

Manage Account Settin

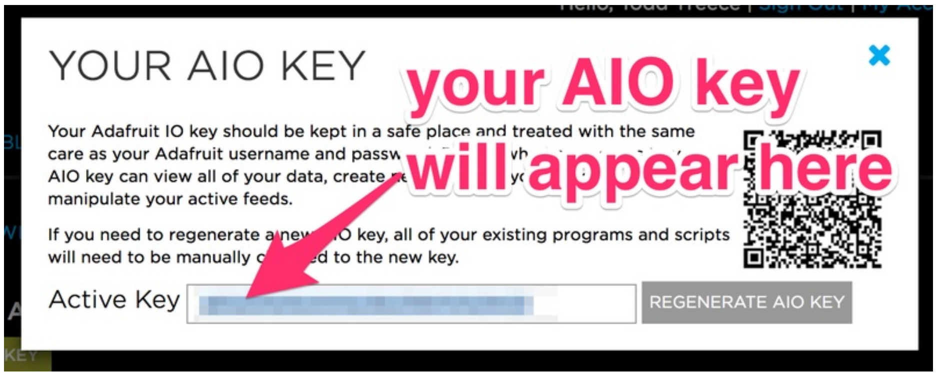
Account Settings includes your time zone, me, email ernar

Manage Account Data

Download the stored content of your Adafruit IO

ds, data streams, groups, activities, triggers, and dashboards. Your user information and aio

Delete all data, including your IO account information. This includes your user, feeds, data streams, groups, and dashboards.

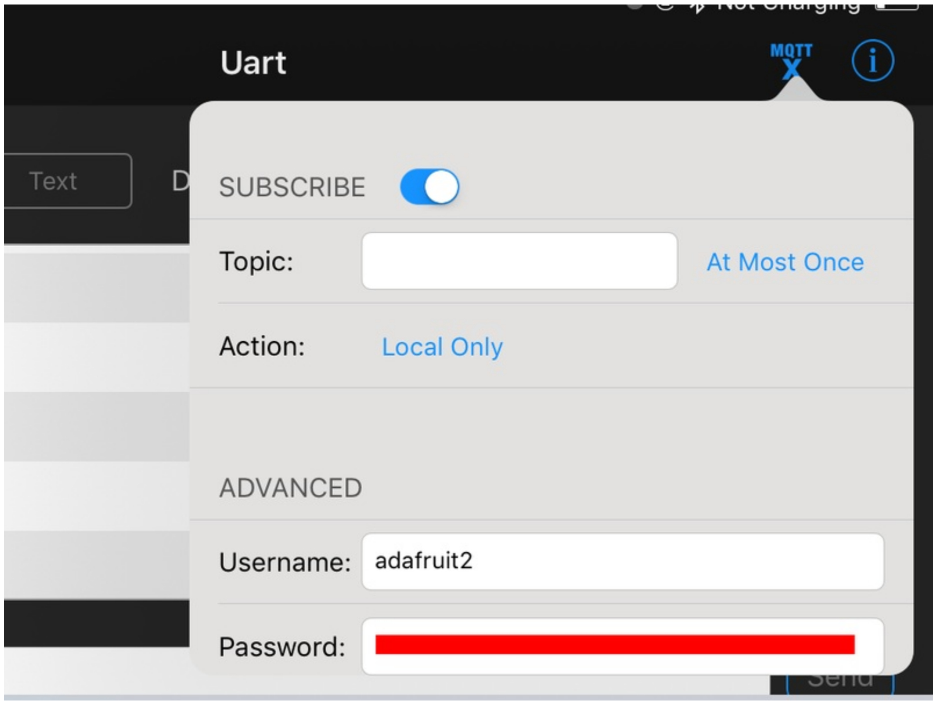

YOUR AIO KEY

Your Adafruit IO key should be kept in a safe place and treated with the same care as your Adafruit username and passw… AIO key can view all of your data, create … manipulate your active feeds.

If you need to regenerate a new … O key, all of your existing programs and scripts will need to be manually … ed to the new key.

Active Key [redacted]

Then take that and put it here like so:



SUBSCRIBE

Topic: At Most Once

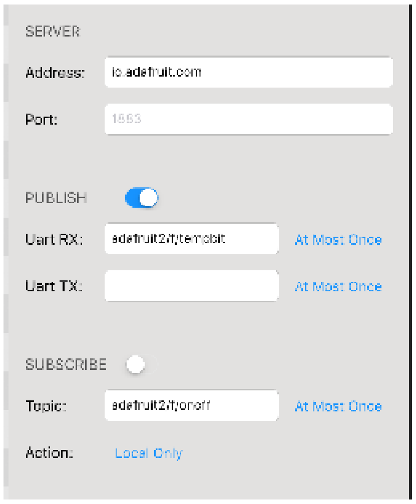
Action: Local Only

ADVANCED

Username: adafruit2

Password: [redacted]

Finally enter in the feed name which is ***username*/f/tempbit** into the UART RX Publish entry. (There's currently a bug where you have to have something in the Subscribe so we put the same feed in there):


SERVER

Address: io.adafruit.com

Port: 1883

PUBLISH

Uart RX: adafruit2/f/tempbit At Most Once

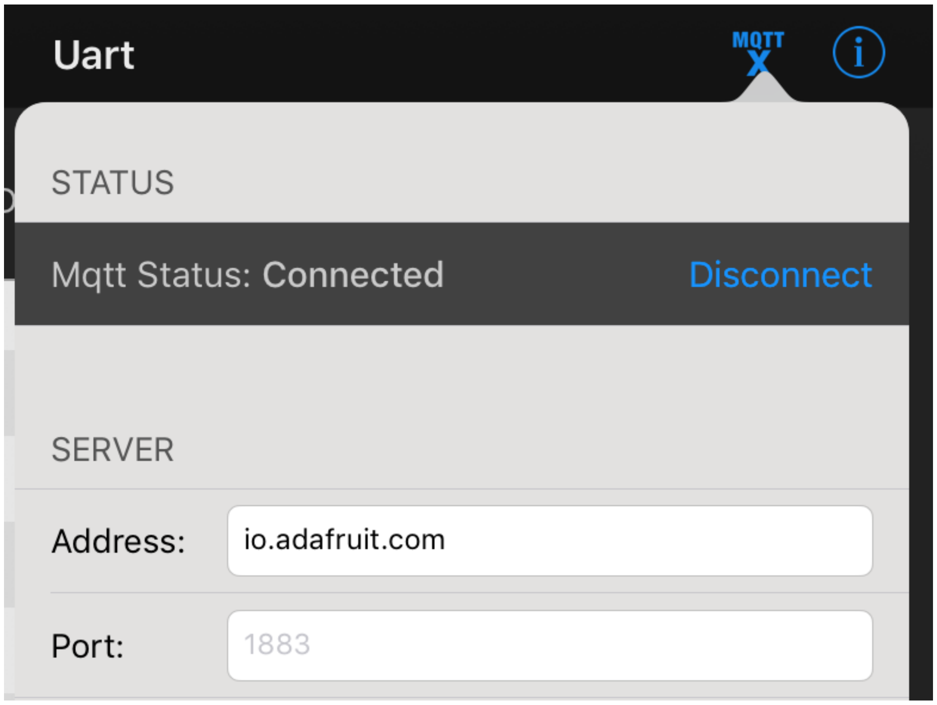
Uart TX: At Most Once

SUBSCRIBE

Topic: adafruit2/f/onoff At Most Once

Action: Local Only

Then click **Connect** at the top:

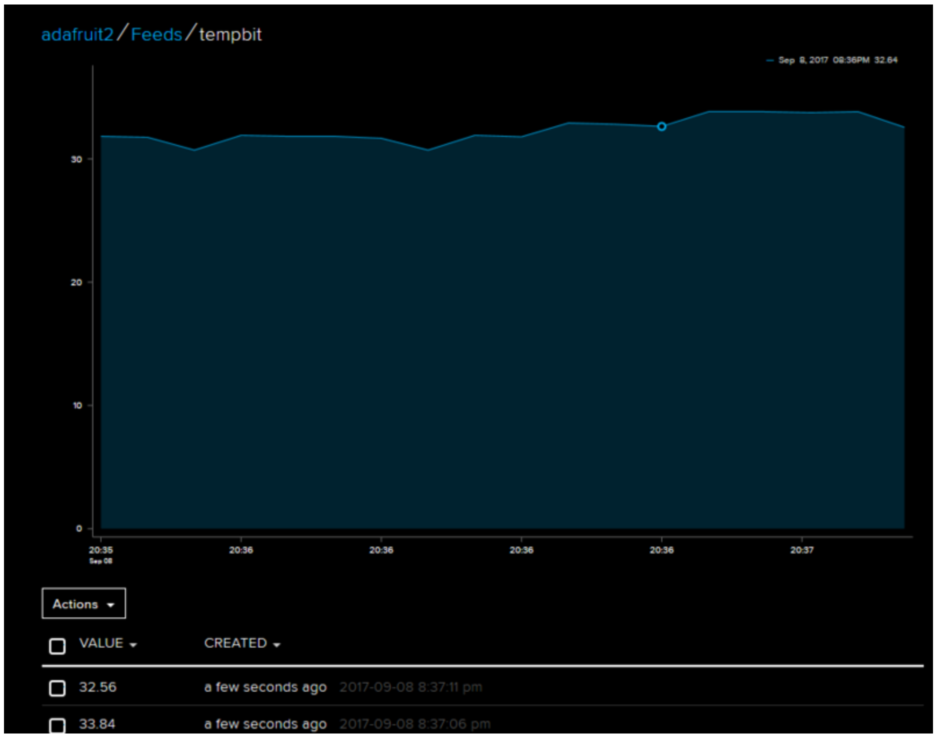


Mqtt Status: Connected

Address: io.adafruit.com

Port: 1883

Wait a few minutes and go visit your Adafruit IO Feeds page, you should see the data start to stream in!



VALUE	CREATED
32.56	a few seconds ago 2017-09-08 8:37:11 pm
33.84	a few seconds ago 2017-09-08 8:37:06 pm

Huzzah! You can now create a public dashboard if you like, to share it with others(https://adafru.it/z8e)

HALP!!!

If you're having issues, you may want to check Sandeep's installation guide for the nRF5x package which may have more details (in case there are updates) (https://adafru.it/zwe)

Some people reported that their microbit did not have a softdevice on it already (which seems odd but is possible!) You can try installing this hex file which will use MakeCode to install a softdevice. Just drag it onto the MICROBIT disk drive

https://adafru.it/zwf

https://adafru.it/zwf

There's also instructions here on how to manually install a softdevice or in the off chance you want softdevice 130 instead of the 'standard' s110 (see Flashing SoftDevice) (https://adafru.it/zwe)