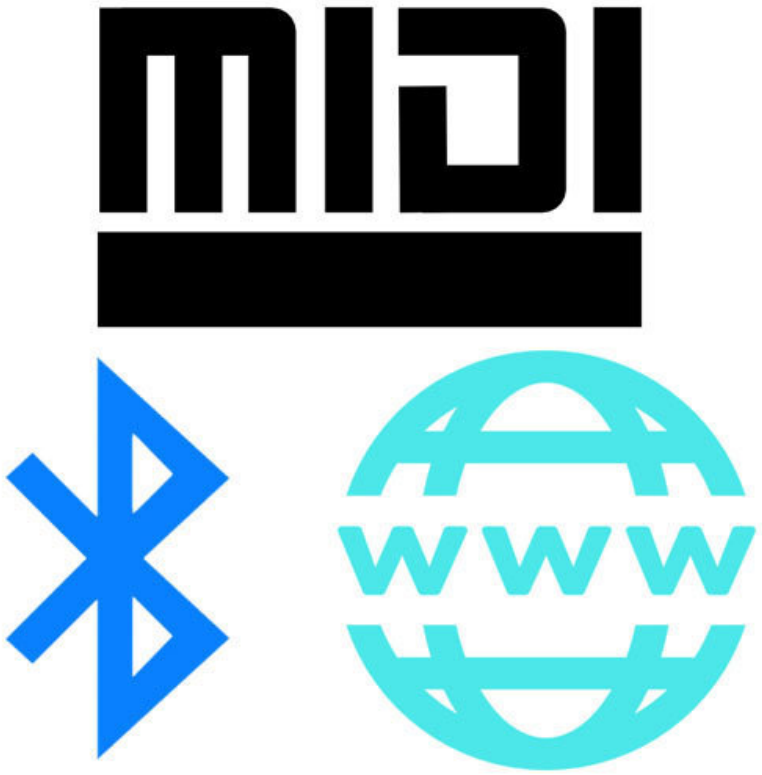
What is Web MIDI & BLE MIDI?

Created by Collin Cunningham



https://learn.adafruit.com/web-ble-midi

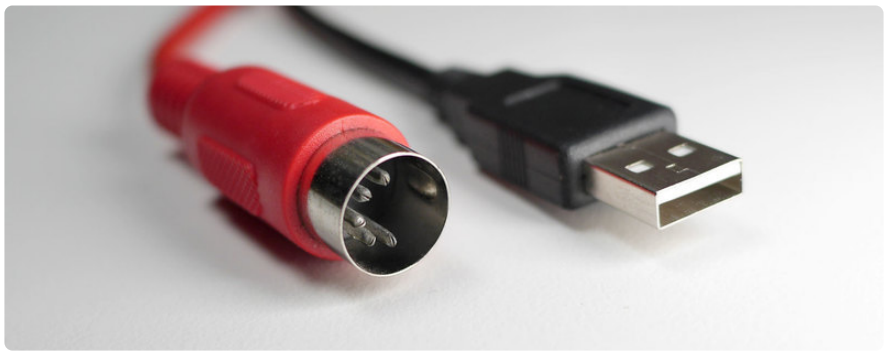
Last updated on 2024-06-03 02:34:06 PM EDT

Table of Contents

Overview

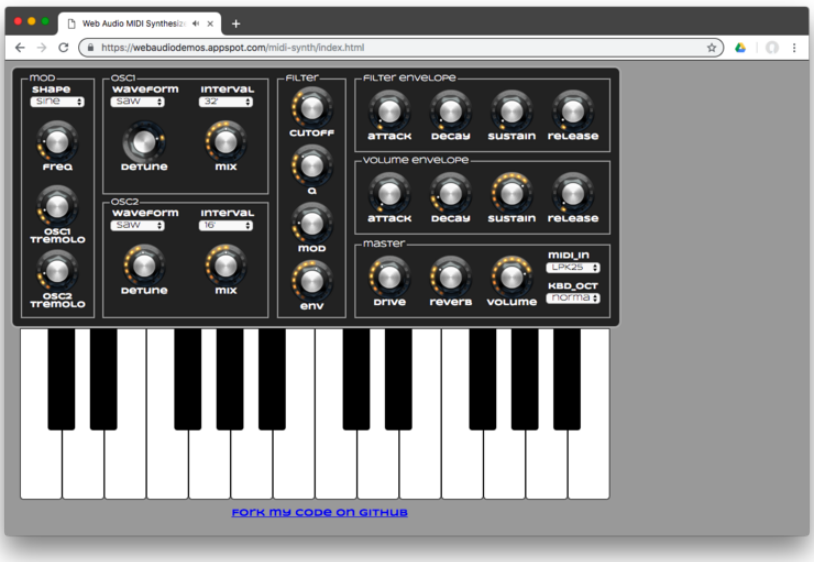
Musical Instrument Digital Interface

The MIDI protocol was created way back in 1983 as a way for musical instruments to communicate digitally. Still alive and well today, MIDI has been adapted to work with new hardware over the years, but the core language of MIDI remains unchanged - note on/off messages, controller values, etc.



For years, the venerable **5-pin DIN connector** was pretty much the only way to send MIDI messages between hardware devices, but that began to change when **USB-MIDI** was introduced in 1999. Today, USB-MIDI is by far the most common way to send MIDI messages (though DIN connectors have had a boutique resurgence of late). Beyond USB, MIDI has been adapted to other means of transport as well. Let's take a look at two of them ...

Web MIDI



The Web MIDI API (https://adafru.it/DcI) was created to allow web applications to respond to MIDI controller inputs. In practical terms, this means a user with a MIDI keyboard can create music without specialized synthesizer or recording software installed on their computer. And it's a two-way street - MIDI messages can also be sent from the web browser to MIDI-capable applications or devices attached to a user's computer. Currently, Web MIDI is supported (https://adafru.it/DcJ) by Chrome (https://adafru.it/B-S), Opera (https://adafru.it/DcK), and Android web browsers.

With so many MIDI-capable hardware devices out there, Web MIDI further blurs the line between local & web applications. Moving beyond standard keyboard/mouse/touchscreen input, web authors can create experimental and accessible content that employs familiar musical interfaces.

```
function onMIDIMessage (message) {
    var frequency = midiNoteToFrequency(message.data[1]);

    if (message.data[0] === 144 &amp;&amp; message.data[2] &gt; 0) {
        playNote(frequency);
    }

    if (message.data[0] === 128 || message.data[2] === 0) {
        stopNote(frequency);
    }
}
```

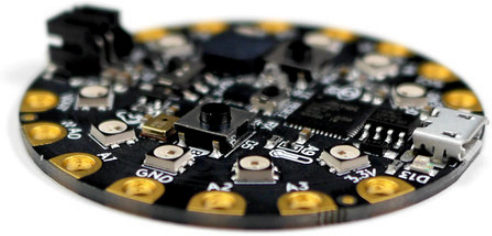
Coding a **javascript**-based web app with Web MIDI support is similar to standard implementations in non-web environments. Check out this tutorial (https://adafru.it/DcL) to get started.

Links

- Chrome Music Lab: Song Maker (https://adafru.it/DcM)
- Web Audio MIDI Synthesizer (https://adafru.it/DcN)
- Web Audio Drum Machine (https://adafru.it/DcO)
- Blokdust (https://adafru.it/DcP)
- drum-machine (with source code) (https://adafru.it/DcQ)

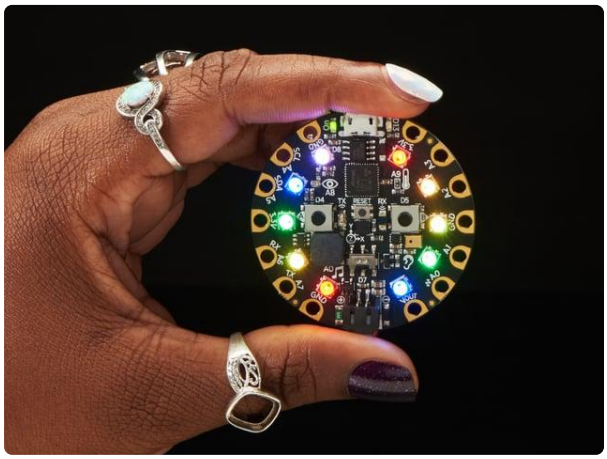
Next, we'll look at how you can program a **Circuit Playground Express** to act a as a MIDI controller for **Web MIDI apps**.

Simple MIDI Controller



You can interact with any Web MIDI app using a **Circuit Playground Express** programmed to function as a **basic MIDI controller**. This project uses the Circuit Playground Express's built-in **buttons** and **light sensor** to trigger **notes** and **modulation control**.

What you'll need

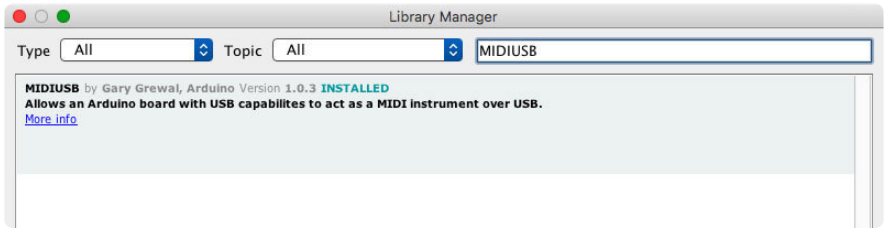


Circuit Playground Express
Circuit Playground Express is the next step towards a perfect introduction to electronics and programming. We've taken the original Circuit Playground Classic and...
https://www.adafruit.com/product/3333

USB cable - USB A to Micro-B
This here is your standard A to micro-B USB cable, for USB 1.1 or 2.0. Perfect for connecting a PC to your Metro, Feather, Raspberry Pi or other dev-board or...
https://www.adafruit.com/product/592

Arduino IDE setup

Follow the steps on this page (https://adafru.it/jDQ) to download and **install** the **Arduino IDE & support for Adafruit boards** on your computer. Then install support for the Circuit Playground Express by following the steps here (https://adafru.it/DcR).



Once installed, **open** Arduino and choose **Tools** -> **Sketch** -> **Manage Libraries** from the top menu. In the new window that appears, type **MIDIUSB** in the search field. Select the **MIDIUSB library** from the results and **install** the latest version.

Code

Create a **new sketch** in **Arduino** and delete the starter code that appears inside of it. **Copy the code seen below, paste** it into that new blank sketch and **save** it.

```
#include &lt;Adafruit_CircuitPlayground.h&gt;
#include "MIDIUSB.h"

bool leftButtonPressed;
bool rightButtonPressed;
bool noteOneOn;
bool noteTwoOn;

void setup() {
  CircuitPlayground.begin();
}

void loop() {

  leftButtonPressed = CircuitPlayground.leftButton();
  rightButtonPressed = CircuitPlayground.rightButton();

  //Control note one based on left button
  if (leftButtonPressed &amp;&amp; !noteOneOn) {
    noteOn(0, 60, 100);
    noteOneOn = true;
  }
  else if (!leftButtonPressed &amp;&amp; noteOneOn) {
    noteOff(0, 60, 100);
    noteOneOn = false;
  }

  //Control note two based on right button
  if (rightButtonPressed &amp;&amp; !noteTwoOn) {
    noteOn(0, 64, 100);
    noteTwoOn = true;
  }
  else if (!rightButtonPressed &amp;&amp; noteTwoOn){
```

```
    noteOff(0, 64, 100);
    noteTwoOn = false;
  }

  //Use light sensor as a modulation control
  int value = CircuitPlayground.lightSensor();  //value from 0-1024
  value = value/8;  //scale to 0-127 for MIDI CC
  controlChange(0, 1, value); //send as MIDI modulation control

}

void noteOn(byte channel, byte pitch, byte velocity) {
  midiEventPacket_t noteOn = {0x09, 0x90 | channel, pitch, velocity};
  MidiUSB.sendMIDI(noteOn);
}

void noteOff(byte channel, byte pitch, byte velocity) {
  midiEventPacket_t noteOff = {0x08, 0x80 | channel, pitch, velocity};
  MidiUSB.sendMIDI(noteOff);
}

void controlChange(byte channel, byte control, byte value) {
  midiEventPacket_t event = {0x0B, 0xB0 | channel, control, value};
  MidiUSB.sendMIDI(event);
}
```

Connect **Circuit Playground Express** to your **computer** using a **micro USB cable**.

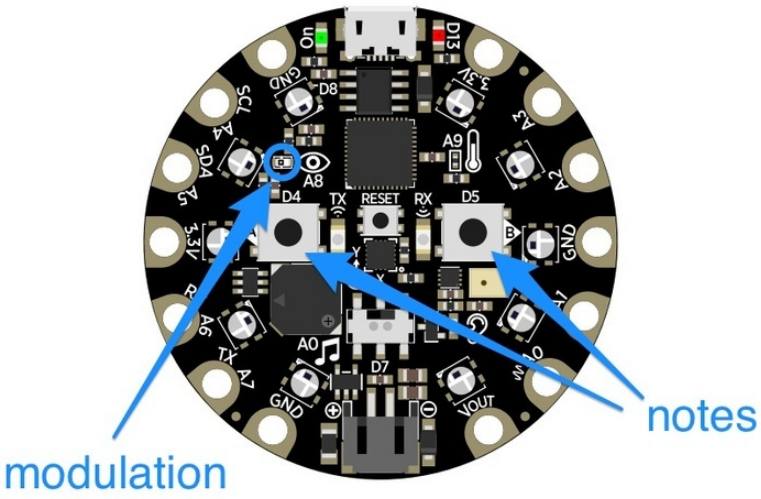
In Arduino's top menu, go to **Tools -> Board**, and choose **Adafruit Circuit Playground Express** from the list. Then go to **Tools -> Port** and choose the port which includes **(Adafruit Circuit Playground Express)** in the name.

Upload the code to your board by going to **Sketch -> Upload**, or pressing the **Upload button** in the sketch window.

Play

Install Google Chrome (https://adafru.it/DcS) on your computer, if you don't have it already. **Launch Chrome** and open the **Web Audio Synthesizer** by going to the following address:

https://webaudiodemos.appspot.com/midi-synth/index.html (https://adafru.it/DcN)



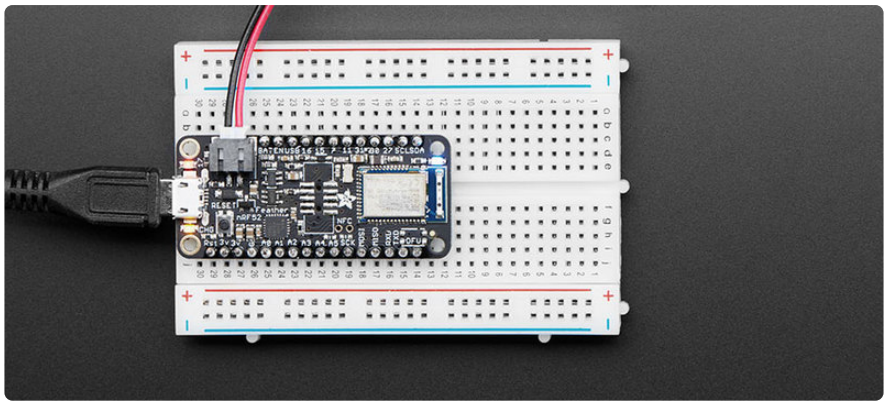
Each of Circuit Playground Express's two **main buttons** will trigger a **note to play** on the synthesizer and the **light sensor** is used as a **modulation controller** - wave your hand over the light sensor to modulate a note's sound.

BLE MIDI

BLE MIDI (https://adafru.it/DcT) makes MIDI wireless, by sending MIDI messages over a Bluetooth Low Energy connection - which makes it a great solution for art & performance applications. The latest versions of Windows and MacOS, & iOS support the BLE MIDI standard.

Mobile Devices

BLE MIDI is a good fit for **smartphones & tablets** which are usually lacking in the hardware connectivity department. Many mobile music apps such as Apple's Garageband (https://adafru.it/C-7) include support for BLE MIDI devices, and Android users can add support using third-party apps (https://adafru.it/DcU).



Latency

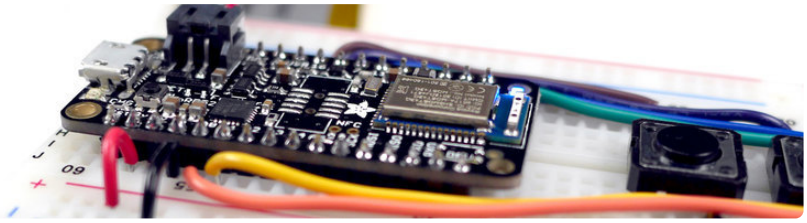
Because of Bluetooth Low Energy's limitations, BLE MIDI messages will take longer to arrive at their destination compared to a wired MIDI connection. The time will vary based on circumstances such as signal strength, proximity, etc., but you can expect a BLE MIDI message to have about **10-20 milliseconds (ms)** of latency between the time a message is sent and when it is received. This is significant when compared to USB MIDI's **~3ms** latency.

Links

- Bluefruit nRF52 - BLEMidi (https://adafru.it/DcV)
- Bluetooth LE MIDI Drumpad (https://adafru.it/DcW)
- Wireless UNTZtrument (https://adafru.it/DcX)

Next, we'll look at how to program a **Feather Bluefruit nRF52** board for use as a **BLE MIDI controller**.

BLE MIDI Controller



You can create a BLE MIDI controller using a **Feather Bluefruit nRF52** and some **basic components** on a **breadboard**. This is a great way to experiment with BLE MIDI control on a mobile device such as an **iOS** or **Android** phone.

What you'll need

Breadboard trim potentiometer - 10K

2 x Trim Pots — https://www.adafruit.com/product/356

Breadboard trim potentiometer - 10K

---

1 x Feather Bluefruit nRF52 — https://www.adafruit.com/product/3406

Adafruit Feather nRF52 Bluefruit LE - nRF52832

---

1 x Breadboard — https://www.adafruit.com/product/239

Full sized breadboard

---

1 x Hook-up Wire — https://www.adafruit.com/product/1311

Hook-up Wire Spool Set - 22AWG Solid Core - 6 x 25 ft

---

1 x Wire Strippers — https://www.adafruit.com/product/527

Hakko Professional Quality 20-30 AWG Wire Strippers - CSP-30-1

---

4 x Buttons — https://www.adafruit.com/product/1119

Tactile Switch Buttons (12mm square, 6mm tall) x 10 pack

---

1 x USB cable — https://www.adafruit.com/product/592

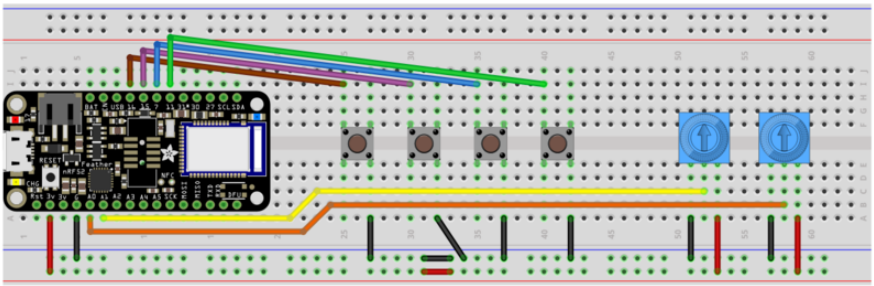
USB cable - USB A to Micro-B - 3 foot long

---

1 x LiPo Battery — https://www.adafruit.com/product/1578

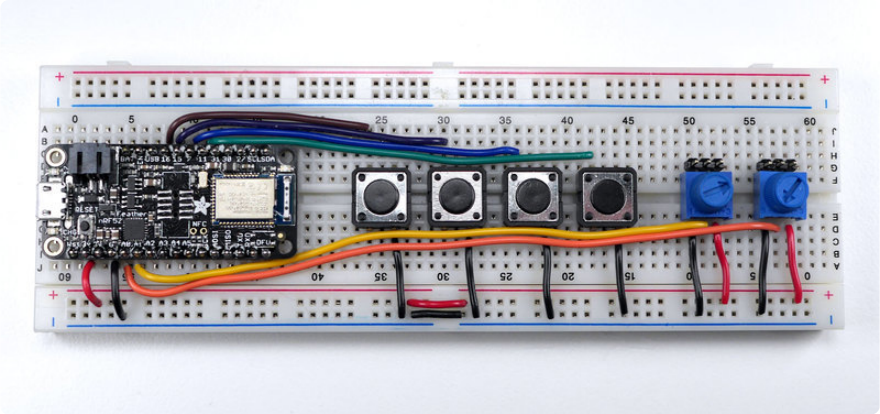
Lithium Ion Polymer Battery - 3.7v 500mAh

Wiring



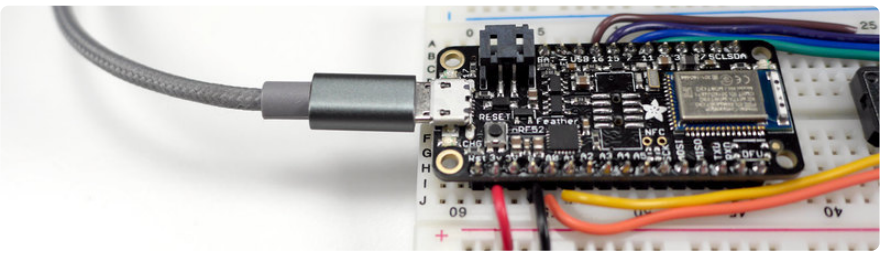
Mount the **Feather**, **buttons**, and **trim pots** on a breadboard and use **jumper wire** to make the connections shown in the diagram above.

After translating that diagram into reality, you should have something like this ...



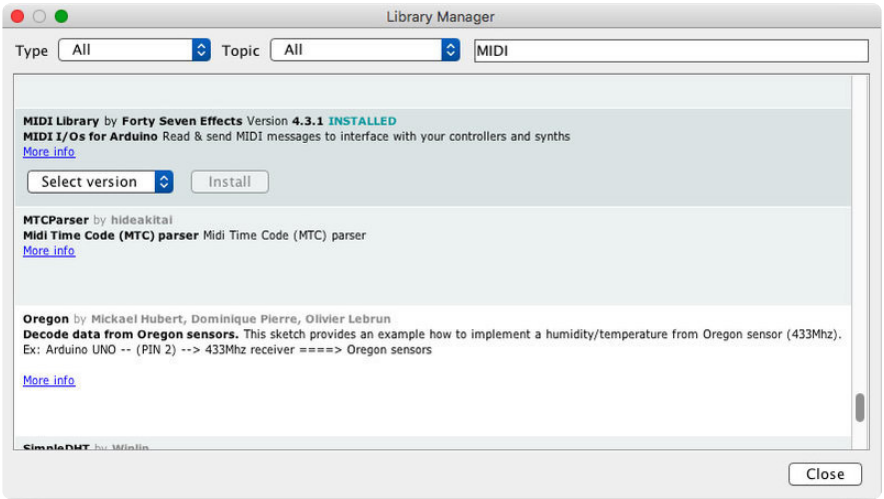
The 3-pin strips of male headers seen above the trim pots are in no way necessary and were simply added for mechanical stability.

Once everything's wired up, **connect the Feather to your computer** using the **micro USB cable**.



Arduino IDE setup

Follow the steps on this page (https://adafru.it/jDQ) to download and **install** the **Arduino IDE** on your computer. Then install support for the Feather Bluefruit nRF52 by following the steps here (https://adafru.it/vnF).



Once installed, **open** Arduino and choose **Tools** -> **Sketch** -> **Manage Libraries** from the top menu. In the new window that appears, type **MIDI** in the search field. Scroll

down in the results to **MIDI Library by Forty Seven Effects,** select it and **install** the latest version.

Code

Create a **new sketch** in **Arduino** and delete the starter code that appears inside of it. **Copy the code seen below, paste** it into that new blank sketch and **save** it.

```
/*********************************************************************
 * Simple MIDI controller for the Feather Bluefruit NRF52
  uses potentiometers connected to pins 2(A0) &amp;amp; 3(A1)
  + four momentary pushbuttons connected to pins 16, 15, 7, &amp;amp; 11

  Adafruit invests time and resources providing this open source code,
  please support Adafruit and open-source hardware by purchasing
  products from Adafruit!

  MIT license
*********************************************************************/

/* For BLE MIDI Setup
   https://learn.adafruit.com/wireless-untztrument-using-ble-midi/overview
*/

#include &amp;lt;bluefruit.h&amp;gt;
#include &amp;lt;MIDI.h&amp;gt;

BLEDis bledis;
BLEMidi blemidi;

// Create a new instance of the Arduino MIDI Library, and attach BluefruitLE MIDI
as the transport.
MIDI_CREATE_BLE_INSTANCE(blemidi);

int buttons[4] = {16, 15, 7, 11}; //pin numbers for each attached button
int notes[4] = {57, 62, 66, 69};  //note each button will play
bool noteStates[4] = {false};     //keep track of the play state of each note

int modPot = 2;  //analog pin A0
int pitchPot = 3;  //analog pin A1
int lastModVal;
int lastPitchVal;

void setup(){

  Serial.begin(115200);
  while ( !Serial ) delay(10);   // for nrf52840 with native usb

  //set input modes for buttons
  for (int i = 0; i &amp;lt; 4; i++) {
    pinMode(buttons[i], INPUT_PULLUP);
  }

  Serial.println("Adafruit Bluefruit52 MIDI over Bluetooth LE Example");

  // Config the peripheral connection with maximum bandwidth
  Bluefruit.configPrphBandwidth(BANDWIDTH_MAX);

  Bluefruit.begin();
  Bluefruit.setName("Bluefruit52 MIDI");
  Bluefruit.setTxPower(4);

  // Setup the on board blue LED to be enabled on CONNECT
```

```
  Bluefruit.autoConnLed(true);

  // Configure and Start Device Information Service
  bledis.setManufacturer("Adafruit Industries");
  bledis.setModel("Bluefruit Feather52");
  bledis.begin();

  // Initialize MIDI, and listen to all MIDI channels, will also call blemidi
service's begin()
  MIDI.begin(MIDI_CHANNEL_OMNI);

  // Attach the handleNoteOn function to the MIDI Library. It will
  // be called whenever the Bluefruit receives MIDI Note On messages.
  MIDI.setHandleNoteOn(handleNoteOn);

  // Do the same for MIDI Note Off messages.
  MIDI.setHandleNoteOff(handleNoteOff);

  // Set up and start advertising
  startAdv();

  // Start MIDI read loop
  Scheduler.startLoop(midiRead);
}

void startAdv(void){

  // Set General Discoverable Mode flag
  Bluefruit.Advertising.addFlags(BLE_GAP_ADV_FLAGS_LE_ONLY_GENERAL_DISC_MODE);

  // Advertise TX Power
  Bluefruit.Advertising.addTxPower();

  // Advertise BLE MIDI Service
  Bluefruit.Advertising.addService(blemidi);

  // Secondary Scan Response packet (optional)
  Bluefruit.ScanResponse.addName();

  //Start Advertising
  Bluefruit.Advertising.restartOnDisconnect(true);
  Bluefruit.Advertising.setInterval(32, 244);    // in unit of 0.625 ms
  Bluefruit.Advertising.setFastTimeout(30);      // number of seconds in fast mode
  Bluefruit.Advertising.start(0);                // 0 = Don't stop advertising
after n seconds
}

void handleNoteOn(byte channel, byte pitch, byte velocity){

  // Log when a note is pressed.
  Serial.printf("Note on: channel = %d, pitch = %d, velocity - %d", channel, pitch,
velocity);
  Serial.println();
}

void handleNoteOff(byte channel, byte pitch, byte velocity){

  // Log when a note is released.
  Serial.printf("Note off: channel = %d, pitch = %d, velocity - %d", channel,
pitch, velocity);
  Serial.println();
}

void loop(){

  // Don't continue if we aren't connected.
  if (! Bluefruit.connected()) {
    return;
  }
```

```
  // Don't continue if the connected device isn't ready to receive messages.
  if (! blemidi.notifyEnabled()) {
    return;
  }

  //check pot values
  int modVal = analogRead(modPot);
  int pitchVal = analogRead(pitchPot);
  pitchVal = map(pitchVal, 0, 1023, -8000, 8000);
  modVal = modVal / 8;

  //send new mod value if it has changed
  if (lastModVal != modVal) {
    Serial.print("modWheel = ");
    Serial.println(modVal);
    MIDI.sendControlChange(1, modVal, 1);
    lastModVal = modVal;
  }

  //send new pitch value if it has changed
  if (lastPitchVal != pitchVal) {
    Serial.print("pitchBend = ");
    Serial.println(pitchVal);
    MIDI.sendPitchBend(pitchVal, 1); //pot value sent as pitch bend
    lastPitchVal = pitchVal;
  }

  //check all buttons
  for (int i = 0; i &lt; 4; i++) {

    bool buttonPressed = !digitalRead(buttons[i]);

    //send note on if button pressed and note is off
    if (buttonPressed &amp;&amp; !noteStates[i]) {
      Serial.print("Button pressed: ");
      Serial.println(i);
      MIDI.sendNoteOn(notes[i], 100, 1);
      noteStates[i] = true;
    }
    //send note off if button released and note is on
    else if (!buttonPressed &amp;&amp; noteStates[i]) {
      Serial.print("Button released: ");
      Serial.println(i);
      MIDI.sendNoteOff(notes[i], 100, 1);
      noteStates[i] = false;
    }
  }

  delay(100);
}

void midiRead(){

  // Don't continue if we aren't connected.
  if (! Bluefruit.connected()) {
    return;
  }

  // Don't continue if the connected device isn't ready to receive messages.
  if (! blemidi.notifyEnabled()) {
    return;
  }

  // read any new MIDI messages
  MIDI.read();
}
```

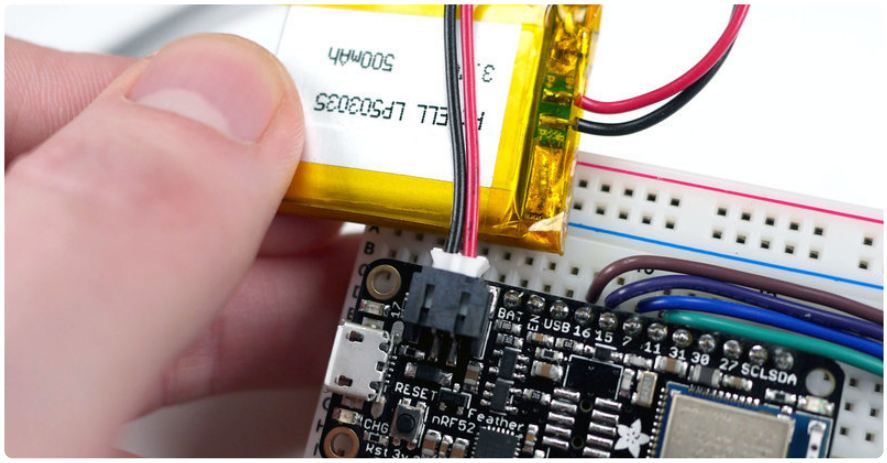
Upload code

Connect your Feather to your computer using a **micro USB cable**. From Arduino's top menu, go to **Tools** -> **Board** and choose **Adafruit Bluefruit nRF52832 Feather** from the list that appears**.**

Next, go to **Tools -> Port** and choose the **SLAB_USBtoUART** port.

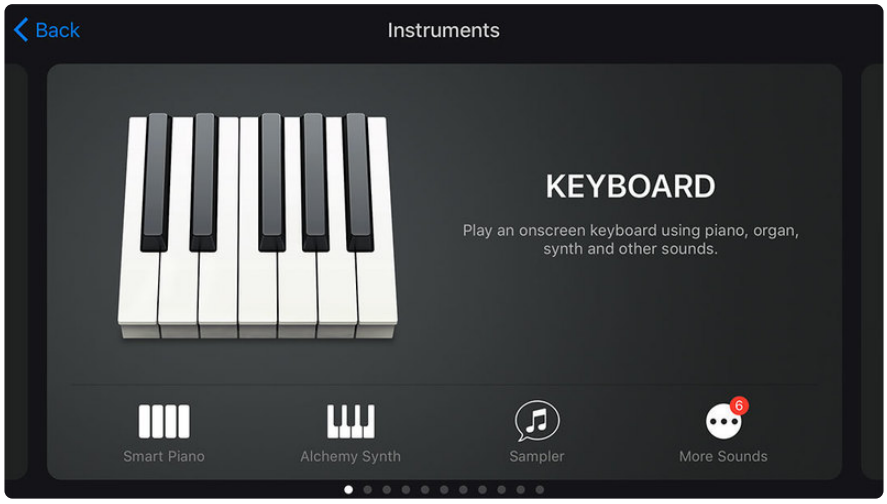
Finally, go **Sketch** -> **Upload** or click the **right-facing arrow button** to upload the sketch to your **Feather**.

Go Wireless

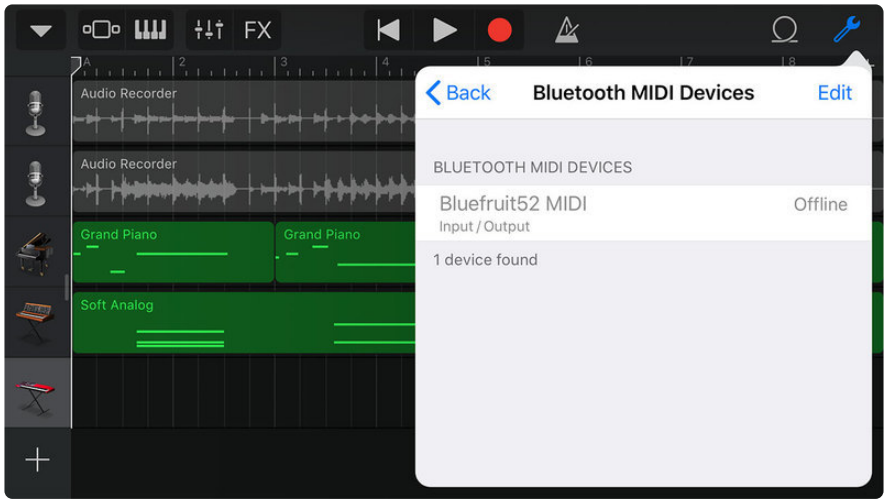


In order to make this a truly wireless test - **disconnect the USB cable** from the Feather and **attach the LiPo battery** to Feather's JST port. You'll know Feather is up and running if you see the **flashing blue LED**.

iOS



Using your iPad or iPhone, download & install Garageband from the App Store (https://adafru.it/DcY). Launch the app and create a new project. When prompted, choose the **piano keyboard** as the first track instrument.

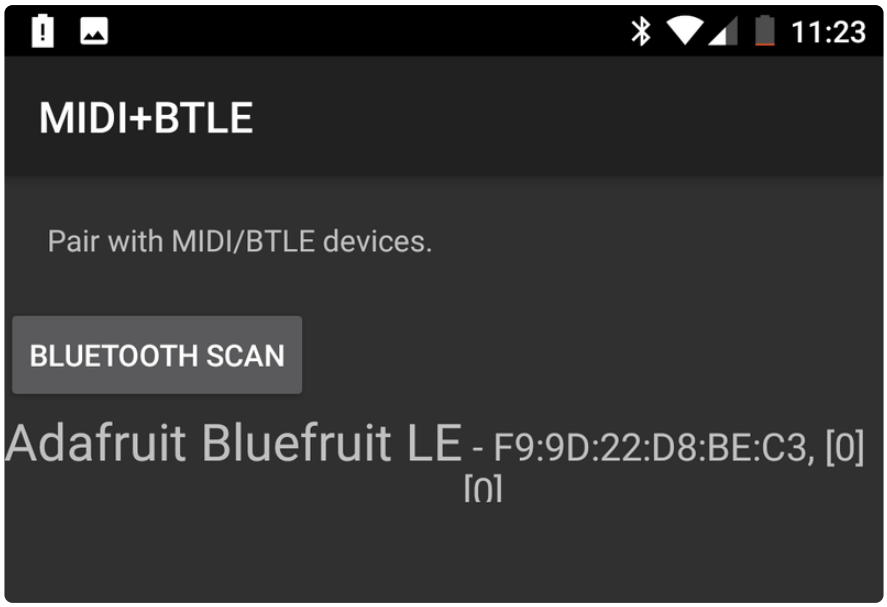


In the main track arrangement view, tap the **wrench icon** in the upper right corner of the screen. From the **Settings** menu that appears, choose **Advanced**, then **Bluetooth MIDI Devices**. The app will scan for available Bluetooth MIDI devices - choose **Bluefruit52 MIDI** from the results list.

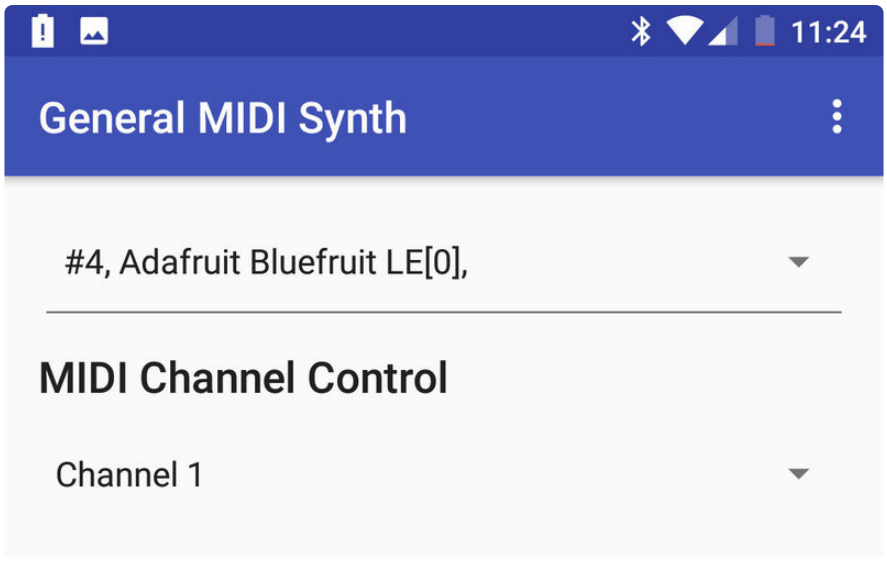
Once the device is paired, you should be able to control Garageband's piano from your breadboard MIDI controller.

Android

On your Android device, open the Play Store and download both MIDI BLE Connect (https://adafru.it/DcZ)& General MIDI Synth. (https://adafru.it/Dc-)

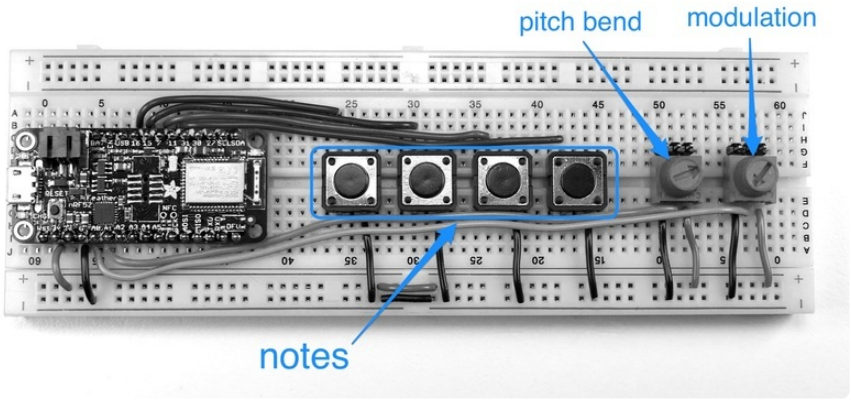


Open MIDI BLE Connect & tap the **BLUETOOTH SCAN** button. Choose **Bluefruit52 MIDI** from the results list.



Next, open the **General MIDI** app, tap the **small triangle icon** at the top, and choose **Bluefruit52 MIDI** from the list. Once the device is paired, you should be able to control **General MIDI's** piano synth from your breadboard BLE MIDI controller.

Play



Each **pushbutton** will **trigger a note**, and the **potentiometers** control **pitch bend** and **modulation**. You can easily expand this project by adding more buttons and soldering the components to a protoboard (http://adafru.it/1606) for more durability.