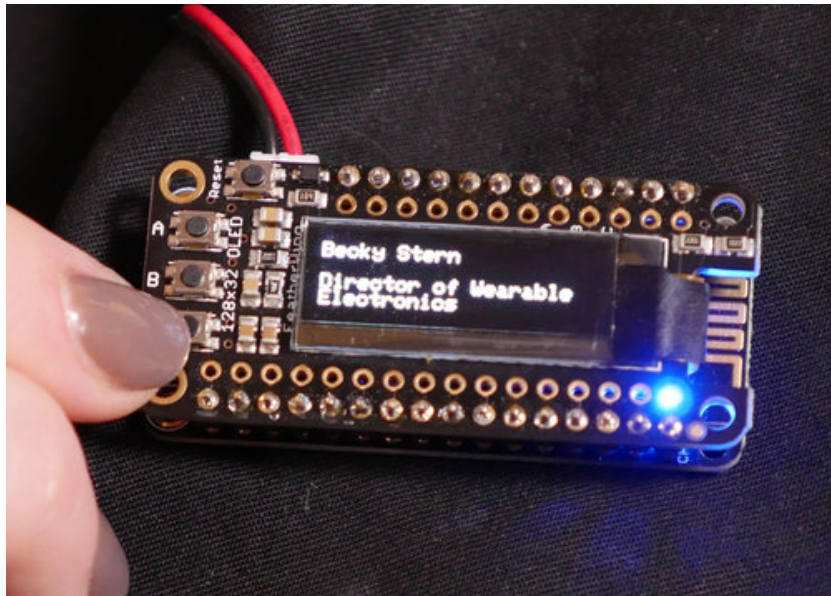




## WiFi OLED Display Badge

Created by Becky Stern



Last updated on 2016-01-20 11:00:14 AM EST

## Guide Contents

Guide Contents	2
Overview	3
Assemble Circuit	5
Adafruit IO and IFTTT Setup	9
Code	12
Wear it!	19

# Overview

---

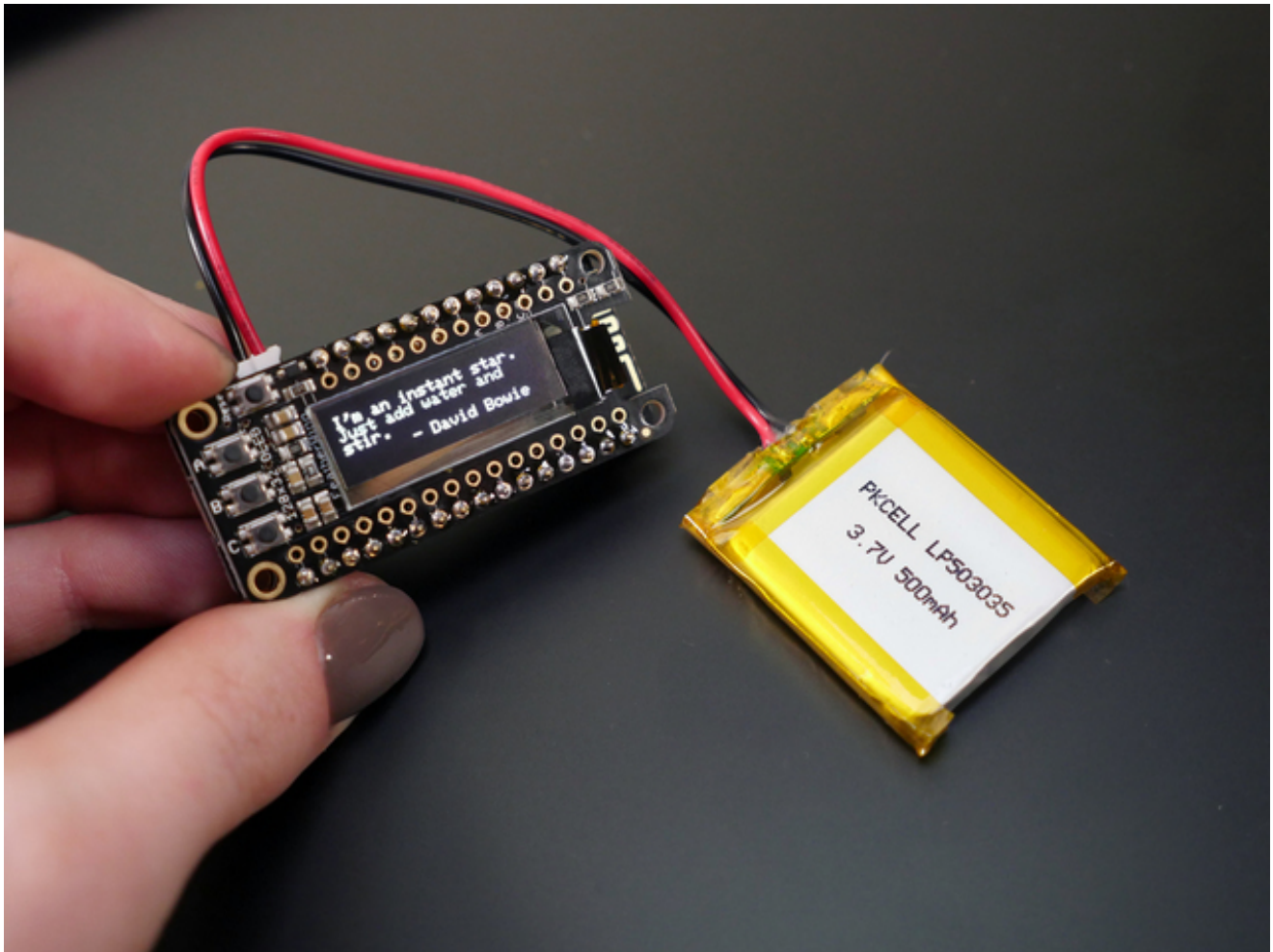
Create an internet connected display brooch and wear your data on your sleeve (or jacket, or backpack...)! This simple Adafruit IO project pulls forecast data from IFTTT and displays it on a FeatherWing OLED display, made wearable with a magnetic pin back. It also toggles to display a favorite quote, or serve as a nametag!

For this project you will need:

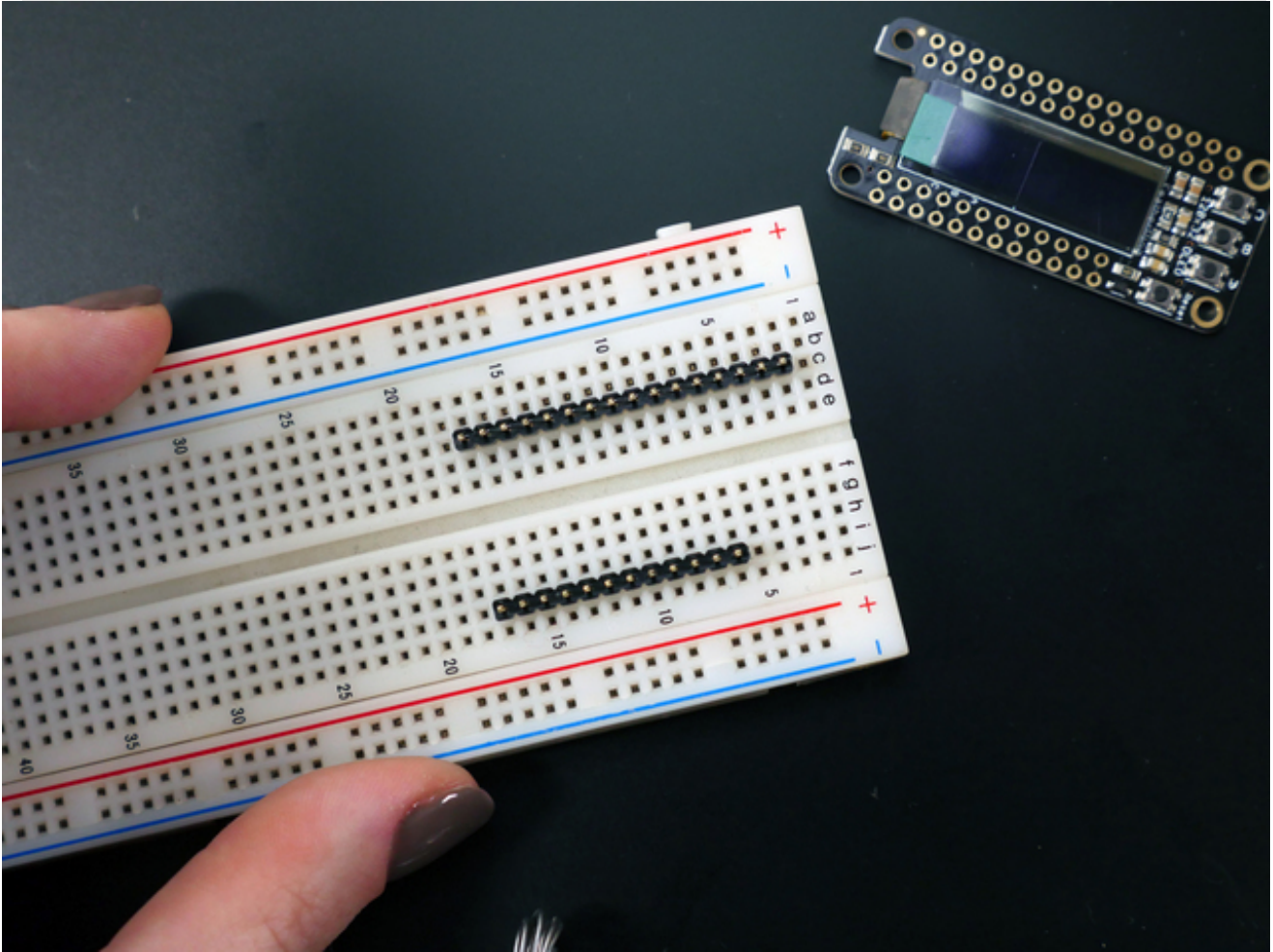
- [Adafruit Feather HUZZAH ESP8266 WiFi board \(http://adafru.it/2821\)](http://adafru.it/2821)
- [FeatherWing OLED display \(http://adafru.it/2900\)](http://adafru.it/2900)
- [Magnetic pin back \(http://adafru.it/1170\)](http://adafru.it/1170)
- Lipoly battery (350mAh (<http://adafru.it/2750>) or 500mAh (<http://adafru.it/1578>) recommended)
- [Soldering tools and supplies \(http://adafru.it/drl\)](http://adafru.it/drl)
- Accounts on [Adafruit IO \(http://adafru.it/fsU\)](http://adafru.it/fsU) and [IFTTT \(http://adafru.it/iOe\)](http://adafru.it/iOe)

Before you begin, please review and understand the following prerequisite guides:

- [Adafruit Feather HUZZAH ESP8266 \(http://adafru.it/kD3\)](http://adafru.it/kD3)
- [Monochrome OLED breakouts \(http://adafru.it/dAZ\)](http://adafru.it/dAZ)
- [Adafruit IO \(http://adafru.it/kSb\)](http://adafru.it/kSb)
- [MQTT, Adafruit.IO & you! \(http://adafru.it/kSc\)](http://adafru.it/kSc)
- [Adafruit guide to excellent soldering \(http://adafru.it/drl\)](http://adafru.it/drl)

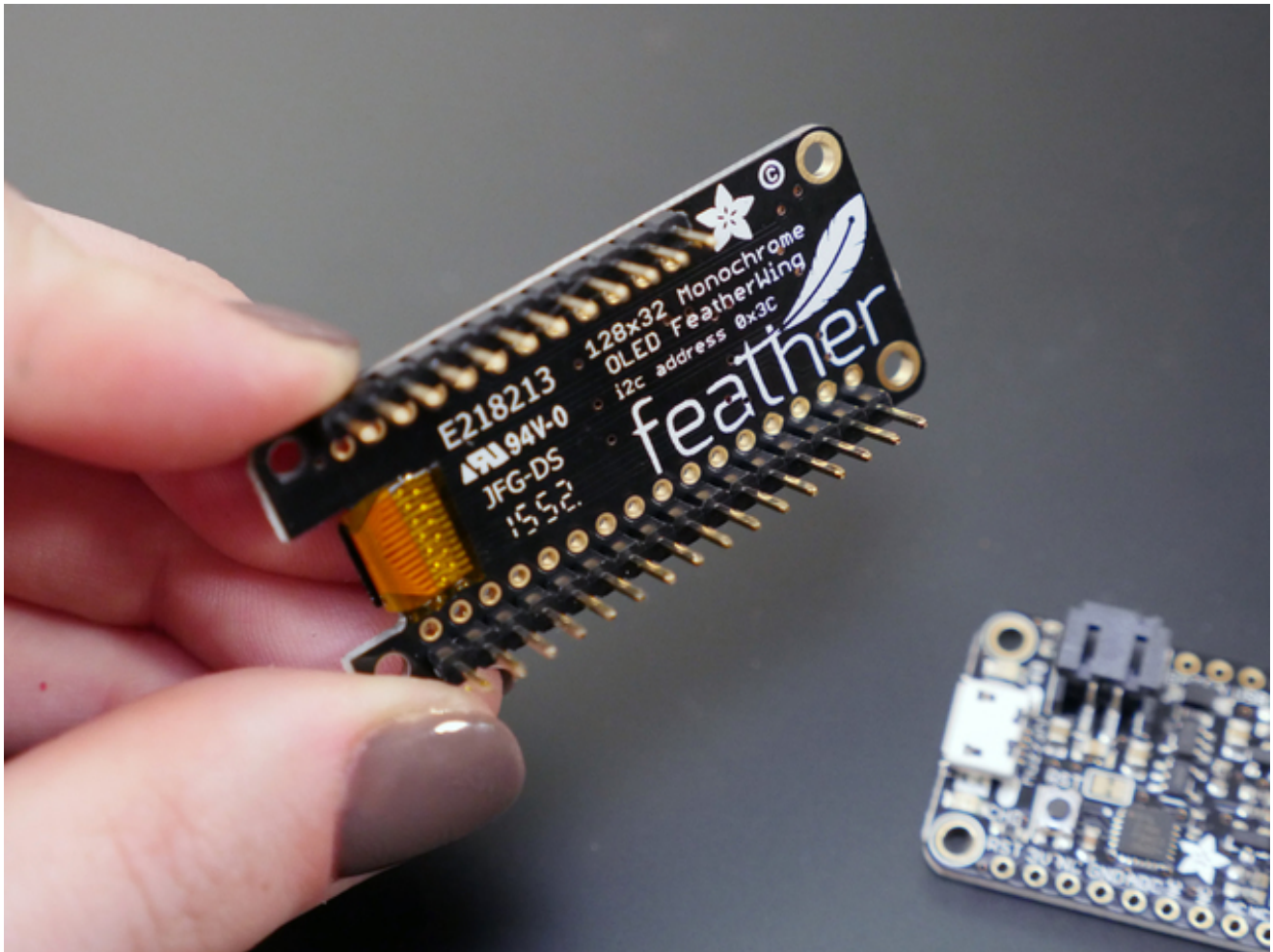


## Assemble Circuit

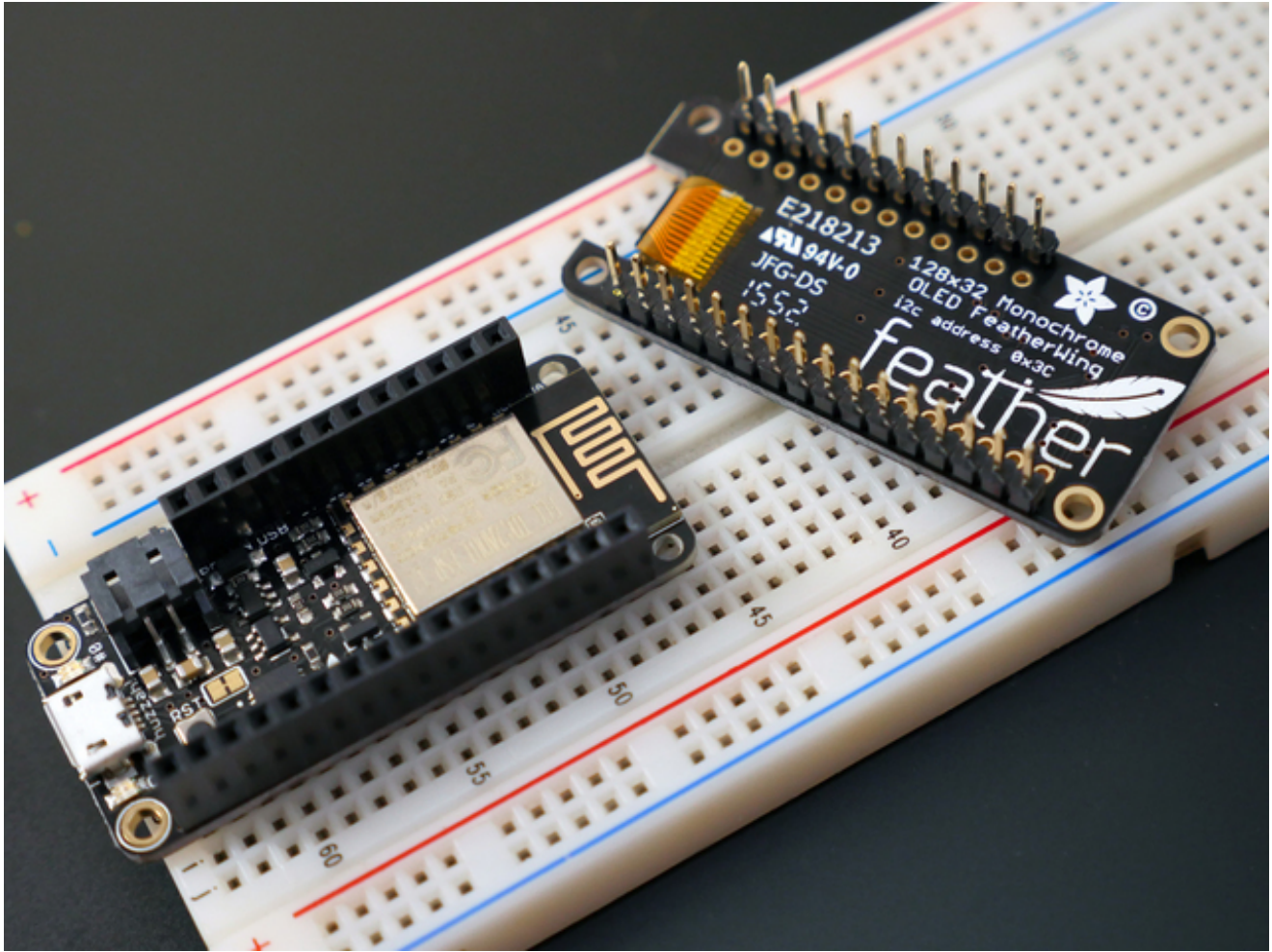


First up, solder headers onto the FeatherWing OLED. The easiest way to get the headers installed with proper alignment is to do this in a solderless breadboard. Trim one of the sections to fit the shorter row of holes on the FeatherWing and press into the breadboard. Slide the FeatherWing OLED onto the headers and solder in place.

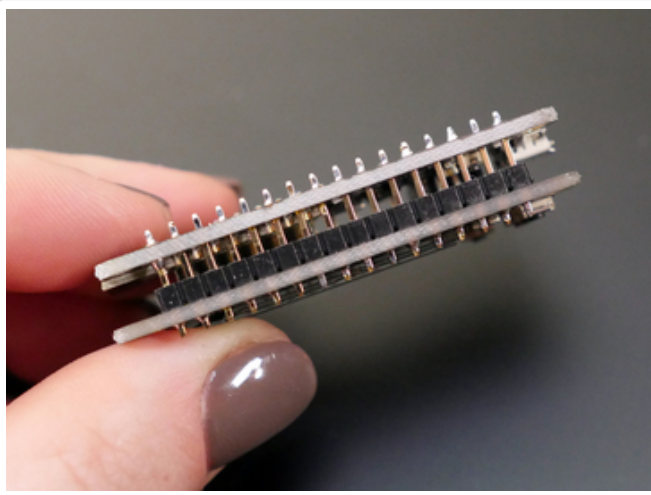
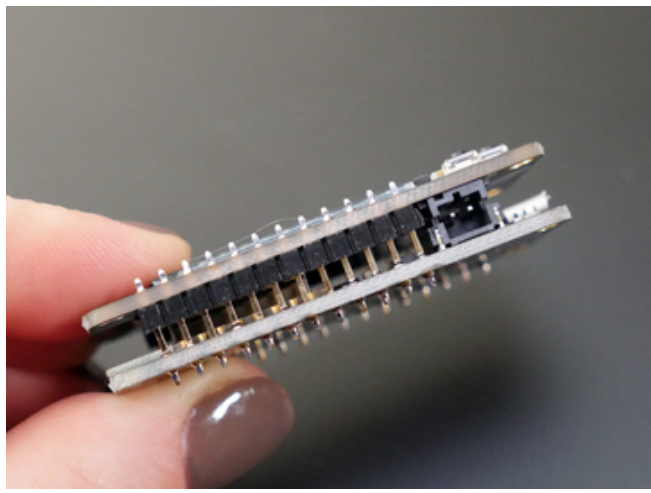




If you have an extra feather to spare, setting one up on a breadboard with [stacking headers](#) (<http://adafruit.it/2830>) is a great way to prototype with FeatherWings. Though not strictly necessary, this lets you try out different FeatherWings and assess any faulty solder connections without committing to permanently connecting the two.



Next, solder your FeatherWing OLED to your Feather main board-- but don't let it be crooked! It's easiest to solder the pin closest to the JST battery connector first, then level the board as that solder cools. Then the rest of the pins will be oriented nicely for soldering up a perfectly parallel board sandwich.





# Adafruit IO and IFTTT Setup

The screenshot shows the Adafruit IO website interface. A modal window titled "CREATE A NEW FEED" is open, allowing a user to create a new data feed. The "NAME" field contains the text "hightemp", and the "DESCRIPTION" field is empty. Below the form are "CANCEL" and "CREATE FEED" buttons. The background shows the user's account page, including a sidebar menu with links like "Active", "Your Feeds", "Your Triggers", "Settings", "Guide and Tutorials", "Bugs or Suggestions", and "Adafruit IO Forum". Below the sidebar, there is a section titled "Your Feeds" with a search bar and a "CREATE FEED" button. A table of existing feeds is visible at the bottom, with columns for ID, NAME, KEY, and LAST VALUE.

ID	NAME	KEY	LAST VALUE
1224	Welcome Feed	welcome-feed	68

First, create a feed to store your data on Adafruit IO called "hightemp"-- this will catch the value from IFTTT (<http://adafru.it/iOe>) and your Feather will check this feed for new values.



Every day at 07:00 AM

Send data to hightemp feed

### Recipe Title

If every day at 7 AM, send data to hightemp feed

use '#' to add tags



Receive notifications when this Recipe runs



Turn off

Publish

Check now

Log

Delete

created 1 day ago  
last run about 1 hour ago  
run 2 times

## Trigger

### Today's weather report

This Trigger retrieves today's current weather report at the time you specify. NOTE: Pollen count available only in the USA.



### Time of day

07 AM



:

00



## Action

### Send data to Adafruit IO

This Action will send data to a feed in your Adafruit IO account.



### Feed name

hightemp



The name of the feed to save data to.



### Data to save

HighTempFahrenheit

The data to be saved to your feed.

Next, create a recipe on IFTTT using the weather channel, and set it to trigger at the time of your choice. Select the Adafruit channel as the output and configure it to send today's forecasted high temperature value to the "hightemp" feed you already created.

You can also adopt the pre-made recipe we published on IFTTT! (<http://adafru.it/kSd>)

Now each day the recipe will update the feed with today's projected high temperature! Substitute this recipe with any value, weather related or not, to customize the project.

You can manually add a value to the feed to test the project, since the IFTTT recipe will only update the feed once per day.

# Code

Load up the following code in your Arduino IDE. You must first have followed instructions for installing the driver and board type for the ESP8266, available in the [Feather HUZZAH guide \(http://adafru.it/kD3\)](http://adafru.it/kD3).

The following Arduino libraries are required (install manually or through the library manager under Sketch-> Include Library-> Manage Libraries...):

- [Adafruit MQTT \(http://adafru.it/fp6\)](http://adafru.it/fp6)
- [Adafruit SSD1306 \(http://adafru.it/aHq\)](http://adafru.it/aHq)

Modify the variables near the top of the sketch to match your WiFi network's SSID, password, Adafruit IO username, and Adafruit IO key before programming your Feather with the sketch.

```
/*
 * WiFi OLED Display Badge
 * guide: https://learn.adafruit.com/digital-display-badge/
 * based on Home Automation sketch by M. Schwartz
 * with contributions from Becky Stern and Tony Dicola
 */

#include <ESP8266WiFi.h>
#include "Adafruit_MQTT.h"
#include "Adafruit_MQTT_Client.h"

const char* ssid    = "SSID";
const char* password = "password";

// Adafruit IO
#define AIO_SERVER    "io.adafruit.com"
#define AIO_SERVERPORT 1883
#define AIO_USERNAME  "your IO username"
#define AIO_KEY       "your IO key"

// Functions
void connect();

// Create an ESP8266 WiFiClient class to connect to the MQTT server.
WiFiClient client;

// Store the MQTT server, client ID, username, and password in flash memory.
// This is required for using the Adafruit MQTT library.
const char MQTT_SERVER[] PROGMEM = AIO_SERVER;
// Set a unique MQTT client ID using the AIO key + the date and time the sketch
```



```

// was compiled (so this should be unique across multiple devices for a user,
// alternatively you can manually set this to a GUID or other random value).
const char MQTT_CLIENTID[] PROGMEM = AIO_KEY __DATE__ __TIME__;
const char MQTT_USERNAME[] PROGMEM = AIO_USERNAME;
const char MQTT_PASSWORD[] PROGMEM = AIO_KEY;

// Setup the MQTT client class by passing in the WiFi client and MQTT server and login details.
Adafruit_MQTT_Client mqtt(&client, MQTT_SERVER, AIO_SERVERPORT, MQTT_CLIENTID, MQTT_USERNAME, MQTT_PASSWORD);

/***** Feeds *****/

// Setup a feed called 'hightemp' for subscribing to changes.
// Notice MQTT paths for AIO follow the form: <username>/feeds/<feedname>
const char hightemp_FEED[] PROGMEM = AIO_USERNAME "/feeds/hightemp";
Adafruit_MQTT_Subscribe hightemp = Adafruit_MQTT_Subscribe(&mqtt, hightemp_FEED);

#include <SPI.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

#define OLED_RESET 3
Adafruit_SSD1306 display(OLED_RESET);

#define LOGO16_GLCD_HEIGHT 16
#define LOGO16_GLCD_WIDTH 16
static const unsigned char PROGMEM logo16_glcd_bmp[] =
{ B00000000, B11000000,
  B00000001, B11000000,
  B00000001, B11000000,
  B00000011, B11100000,
  B11110011, B11100000,
  B11111110, B11111000,
  B01111110, B11111111,
  B00110011, B10011111,
  B00011111, B11111100,
  B00001101, B01110000,
  B00011011, B10100000,
  B00111111, B11100000,
  B00111111, B11110000,
  B01111100, B11110000,
  B01110000, B01110000,
  B00000000, B00110000 };

```

```

#if (SSD1306_LCDHEIGHT != 32)
#error("Height incorrect, please fix Adafruit_SSD1306.h!");
#endif

// this constant won't change:
const int buttonPin = 2; // the pin that the pushbutton is attached to

// Variables will change:
int buttonPushCounter = 0; // counter for the number of button presses
int buttonState = 0;       // current state of the button
int lastButtonState = 0;   // previous state of the button

void setup() {
  Serial.begin(115200);
  delay(100);

  // initialize the button pin as a input:
  pinMode(buttonPin, INPUT_PULLUP);

  // by default, we'll generate the high voltage from the 3.3v line internally! (neat!)
  display.begin(SSD1306_SWITCHCAPVCC, 0x3C); // initialize with the I2C addr 0x3C (for the 128x32)
  // init done

  // Show image buffer on the display hardware.
  // Since the buffer is initialized with an Adafruit splashscreen
  // internally, this will display the splashscreen.
  display.display();
  delay(2000);

  // Clear the buffer.
  display.clearDisplay();

  // We start by connecting to a WiFi network
  Serial.println();
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);

  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
}

```

```

Serial.println("");
Serial.println("WiFi connected");
Serial.println("IP address: ");
Serial.println(WiFi.localIP());

display.setTextSize(1);
display.setTextColor(WHITE);
display.setCursor(0,0);
display.print("Connected to ");
display.println(ssid);
display.setCursor(0,16);
display.print("IP address: ");
display.println(WiFi.localIP());
display.display();
delay(2000);

// listen for events on the weather feed
mqtt.subscribe(&hightemp);

// connect to adafruit io
connect();
}

int temp = 151;
int whichbutton = 0;

void loop() {

  Adafruit_MQTT_Subscribe *subscription;

  // ping adafruit io a few times to make sure we remain connected
  if(! mqtt.ping(3)) {
    // reconnect to adafruit io
    if(! mqtt.connected())
      connect();
  }

  // read the pushbutton input pin:
  buttonState = digitalRead(buttonPin);

  // compare the buttonState to its previous state
  if (buttonState != lastButtonState) {
    // if the state has changed, increment the counter
    if (buttonState == LOW) {
      // if the current state is LOW, then pushbutton has been pressed.
      // increment counter
      whichbutton++;
    }
  }
}

```

```

// if the current state is LOW then the button was pressed
buttonPushCounter++;
Serial.print("number of button pushes: ");
Serial.println(buttonPushCounter);
}

}

// this is our 'wait for incoming subscription packets' busy subloop
if (subscription = mqtt.readSubscription(1000)) {

    // we only care about the weather events
    if (subscription == &hightemp) {

        temp = atoi((char*)hightemp.lastread);
        Serial.print(F("Received: "));
        Serial.println(temp);

    }

}

//A button position - display today's high temperature
if (buttonPushCounter % 3 == 0) {
    if(temp == 151){
        display.clearDisplay();
        display.setCursor(0,16);
        display.println("Waiting for temp data");
        display.display();
        Serial.println(F("Printed: Waiting for temp data"));
    }else{
        display.clearDisplay();
        display.setCursor(0,16);
        display.print("Today's high: ");
        display.print(temp);
        display.println(" F");
        display.display();
        Serial.print(F("Printed: "));
        Serial.println(temp);
    }
}

//B button position - David Bowie quote
if (buttonPushCounter % 3 == 1) {

```



```

display.clearDisplay();
display.setCursor(0,0);
display.println("I'm an instant star. Just add water and ");
display.println("stir.");
display.println("      - David Bowie");
display.display();
}

```

```

//C button position - nametag
if (buttonPushCounter % 3 == 2) {
  display.clearDisplay();
  display.setCursor(0,0);
  display.println("Your Name");
  display.println();
  display.println("Your Title");
  display.display();
}

```

```

// save the current state as the last state,
//for next time through the loop
lastButtonState = buttonState;

}

```

```

// connect to adafruit io via MQTT
void connect() {

```

```

  Serial.print(F("Connecting to Adafruit IO... "));

```

```

  int8_t ret;

```

```

  while ((ret = mqtt.connect()) != 0) {

```

```

    switch (ret) {
      case 1: Serial.println(F("Wrong protocol")); break;
      case 2: Serial.println(F("ID rejected")); break;
      case 3: Serial.println(F("Server unavail")); break;
      case 4: Serial.println(F("Bad user/pass")); break;
      case 5: Serial.println(F("Not authed")); break;
      case 6: Serial.println(F("Failed to subscribe")); break;
      default: Serial.println(F("Connection failed")); break;
    }

```

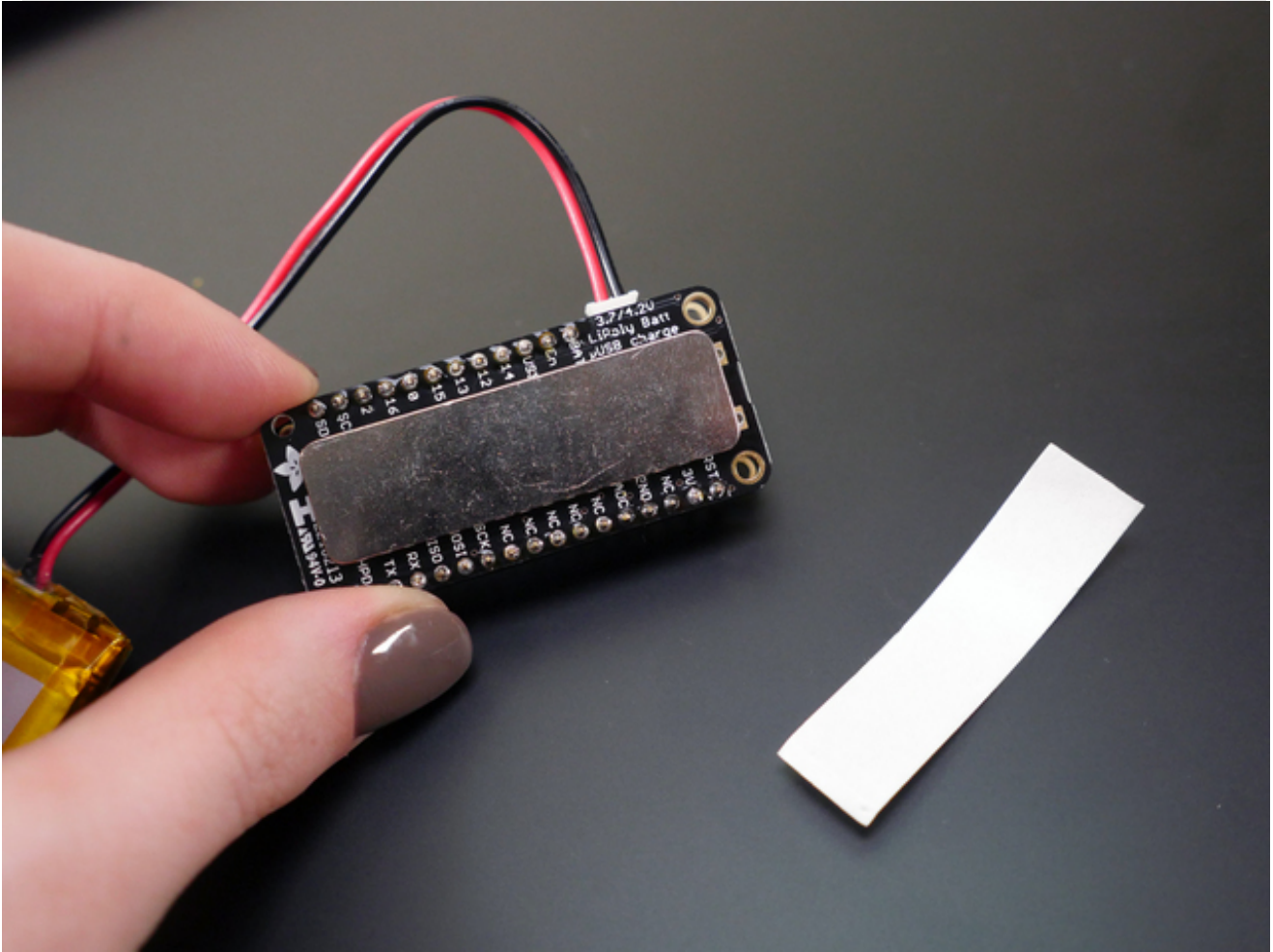
```

    if(ret >= 0)
      mqtt.disconnect();

```

```
Serial.println(F("Retrying connection..."));  
delay(5000);  
  
}  
  
Serial.println(F("Adafruit IO Connected!"));  
  
}
```

## Wear it!



When your circuit is working properly, attach the magnetic pin back by peeling the paper from the adhesive backing on the metal plate. Stick it to the back of the Feather, and use the magnetic plate to attach it to your jacket, backpack, and more!

The C button on the FeatherWing OLED serves to toggle between display modes. If you wish, you may tuck your battery inside a [3D printed pocket](http://adafru.it/kSa) (<http://adafru.it/kSa>)!

