Particle

PARTICLE 101 - A LAP AROUND THE PARTICLE ECOSYSTEM

WHY PARTICLE

THE PARTICLE CLOUD & FRIENDS

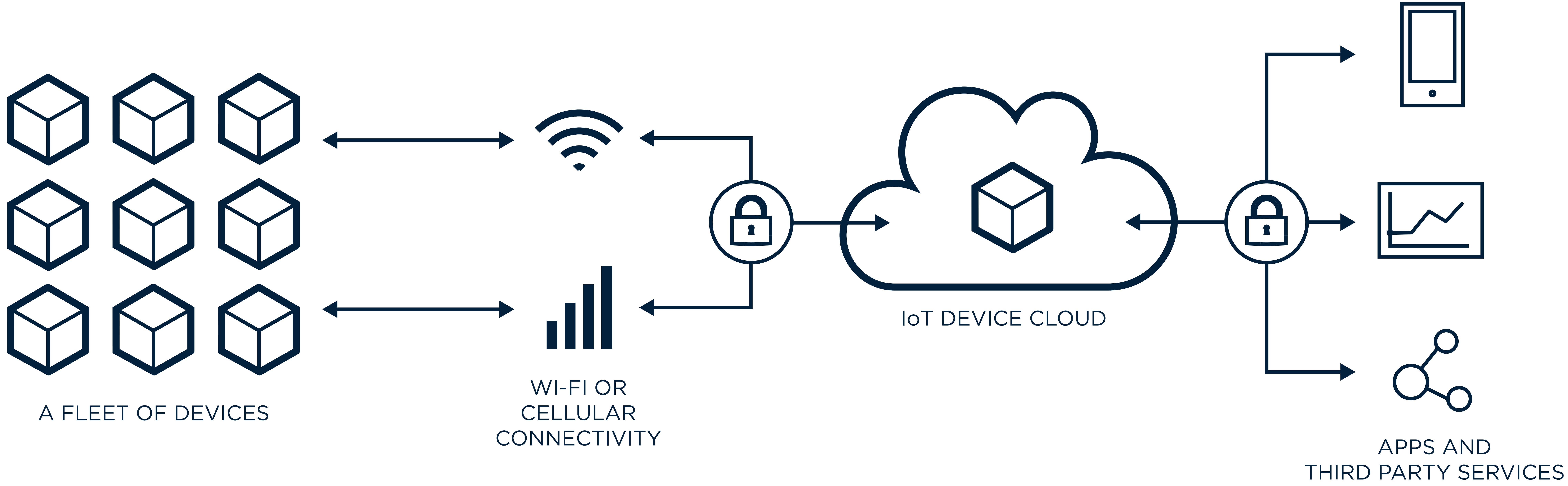
CLAIMING YOUR FIRST DEVICE

WHY PARTICLE

THE PARTICLE CLOUD & FRIENDS

CLAIMING YOUR FIRST DEVICE

PARTICLE IS A FULL-STACK IOT DEVICE PLATFORM



A FLEET OF DEVICES

WI-FI OR CELLULAR CONNECTIVITY

IoT DEVICE CLOUD

APPS AND THIRD PARTY SERVICES

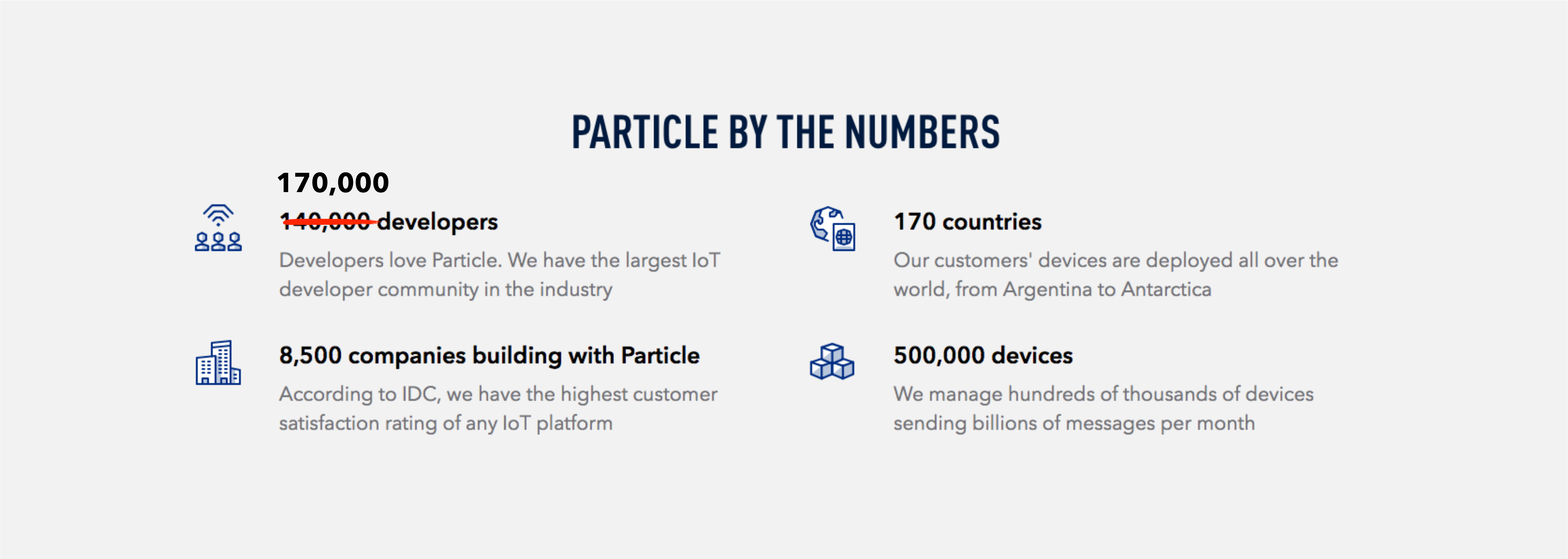
IOT DEVICE HARDWARE AND FIRMWARE

WI-FI AND CELLULAR MVNO

IOT DEVICE CLOUD

WEB/MOBILE APP SDKS AND INTEGRATIONS WITH THIRD-PARTY SERVICES

WE HAVE GROWN THE WORLD'S LARGEST IOT DEVELOPER COMMUNITY



PARTICLE BY THE NUMBERS

170,000

~~140,000~~ developers

Developers love Particle. We have the largest IoT developer community in the industry

170 countries

Our customers' devices are deployed all over the world, from Argentina to Antarctica

8,500 companies building with Particle

According to IDC, we have the highest customer satisfaction rating of any IoT platform

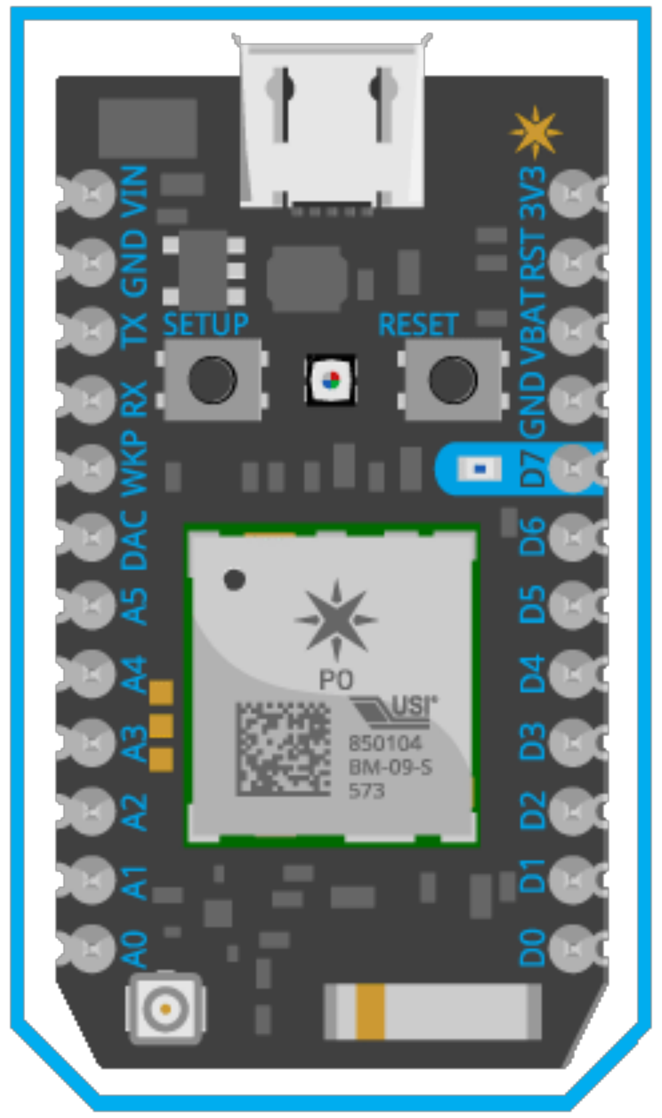
500,000 devices

We manage hundreds of thousands of devices sending billions of messages per month

WE FOCUS ON SOLVING REAL PROBLEMS FOR REAL CUSTOMERS

WI-FI FOR PROTOTYPING AND PRODUCTION

WITH THE SAME DEVICE OS ON ALL MODULES, YOU CAN PROTOTYPE ON A PHOTON AND SCALE UP TO A P0 OR P1 WITH NO FIRMWARE CHANGES

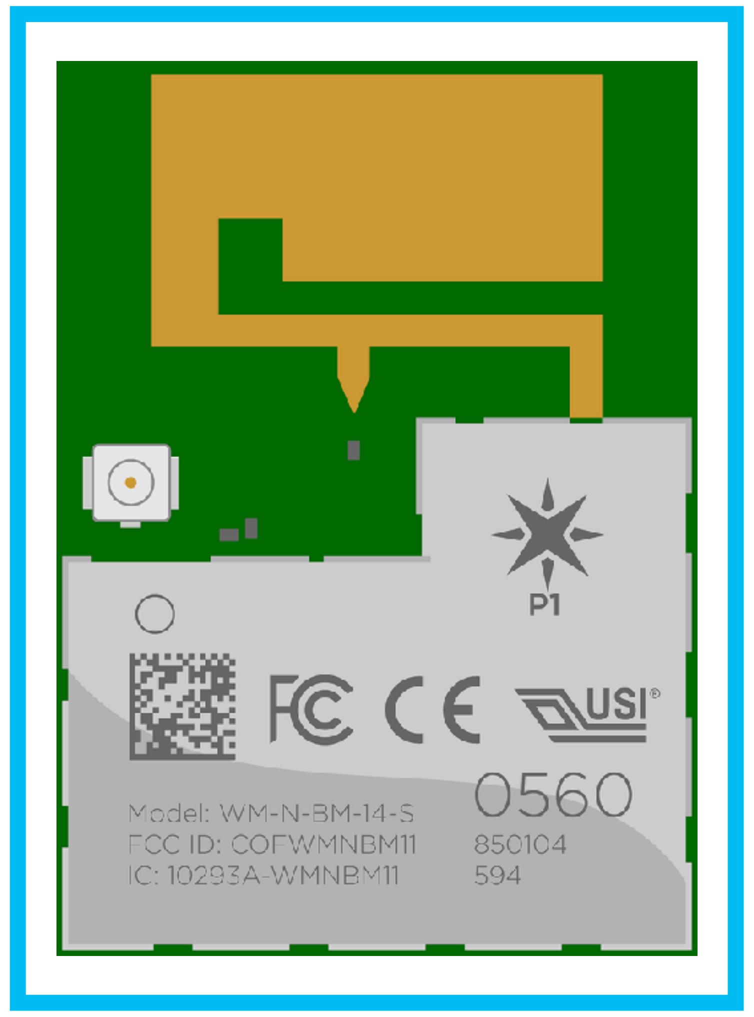


Photon Development Kit

- Breadboard-friendly Dev Kit
- Available with or without headers

P0

- Small, mass-production form-factor
- Available in mfg-ready reels

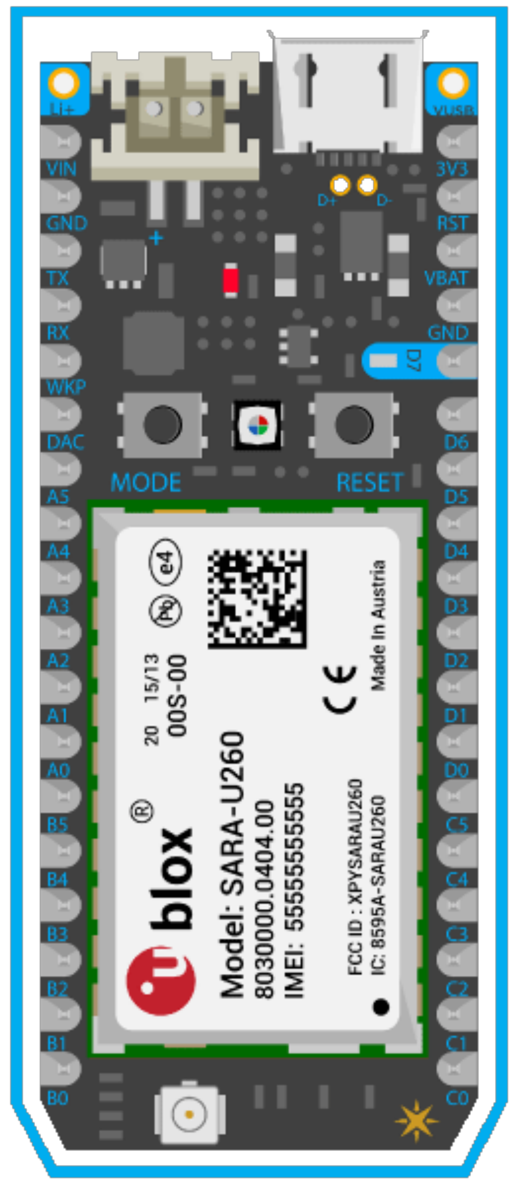


P1

- Built-in PCB antenna and external antenna connector
- Available in mfg-ready reels

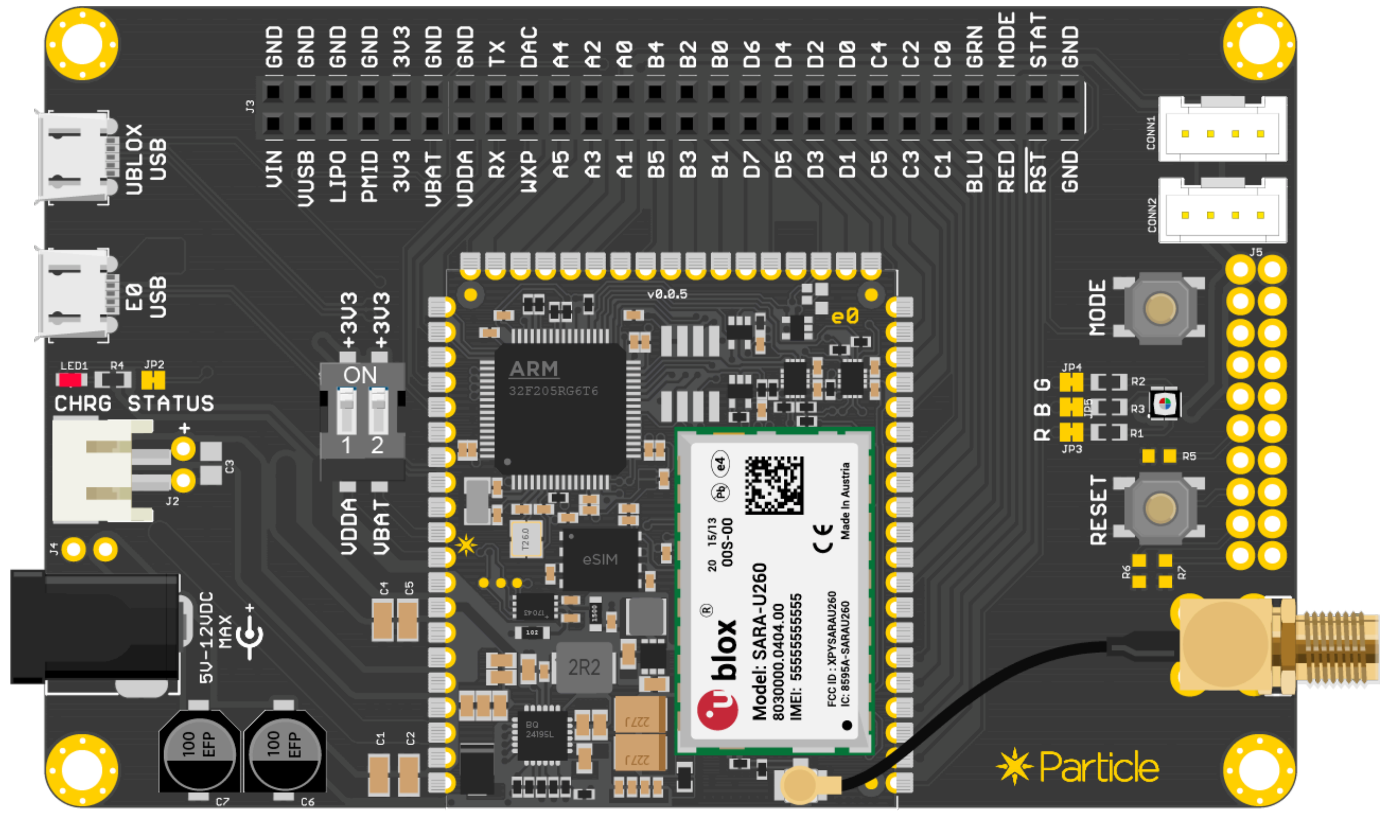
CELLULAR FOR PROTOTYPING AND PRODUCTION

WITH THE SAME DEVICE OS ON ALL MODULES, YOU CAN PROTOTYPE ON AN ELECTRON AND SCALE UP TO AN E-SERIES WITH MINIMAL FIRMWARE CHANGES



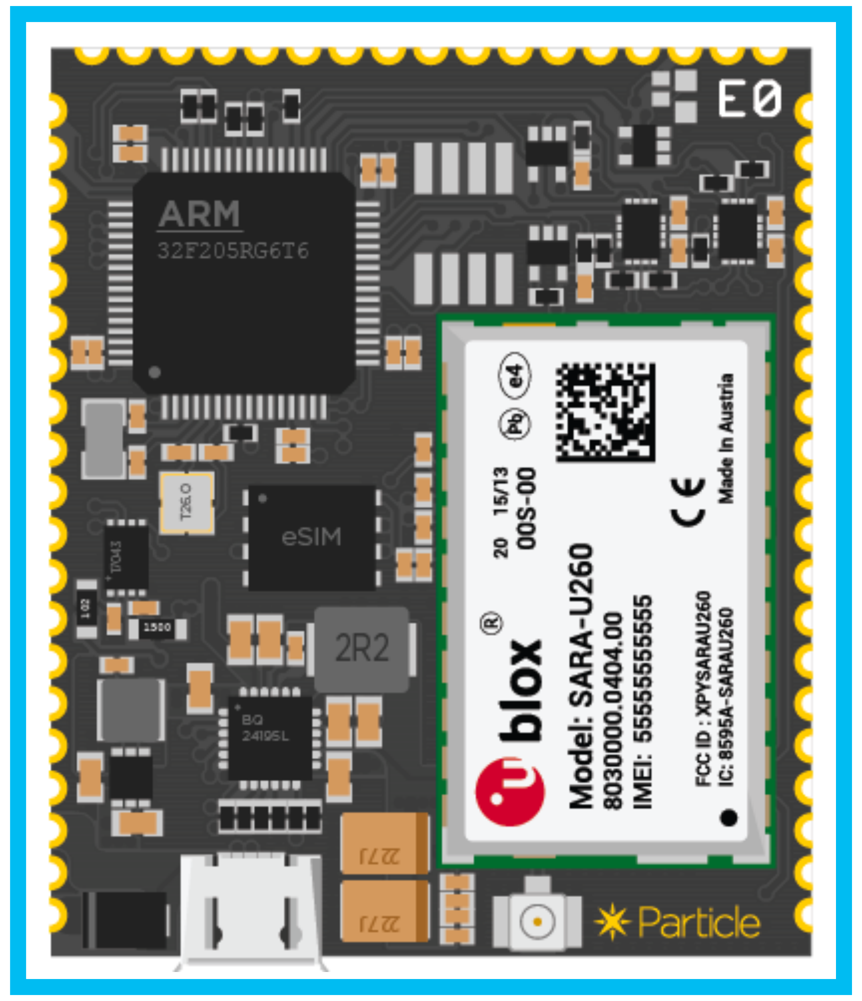
Electron Development Kit

- Breadboard-friendly Dev Kit
- Available only with headers



E-Series Eval Board

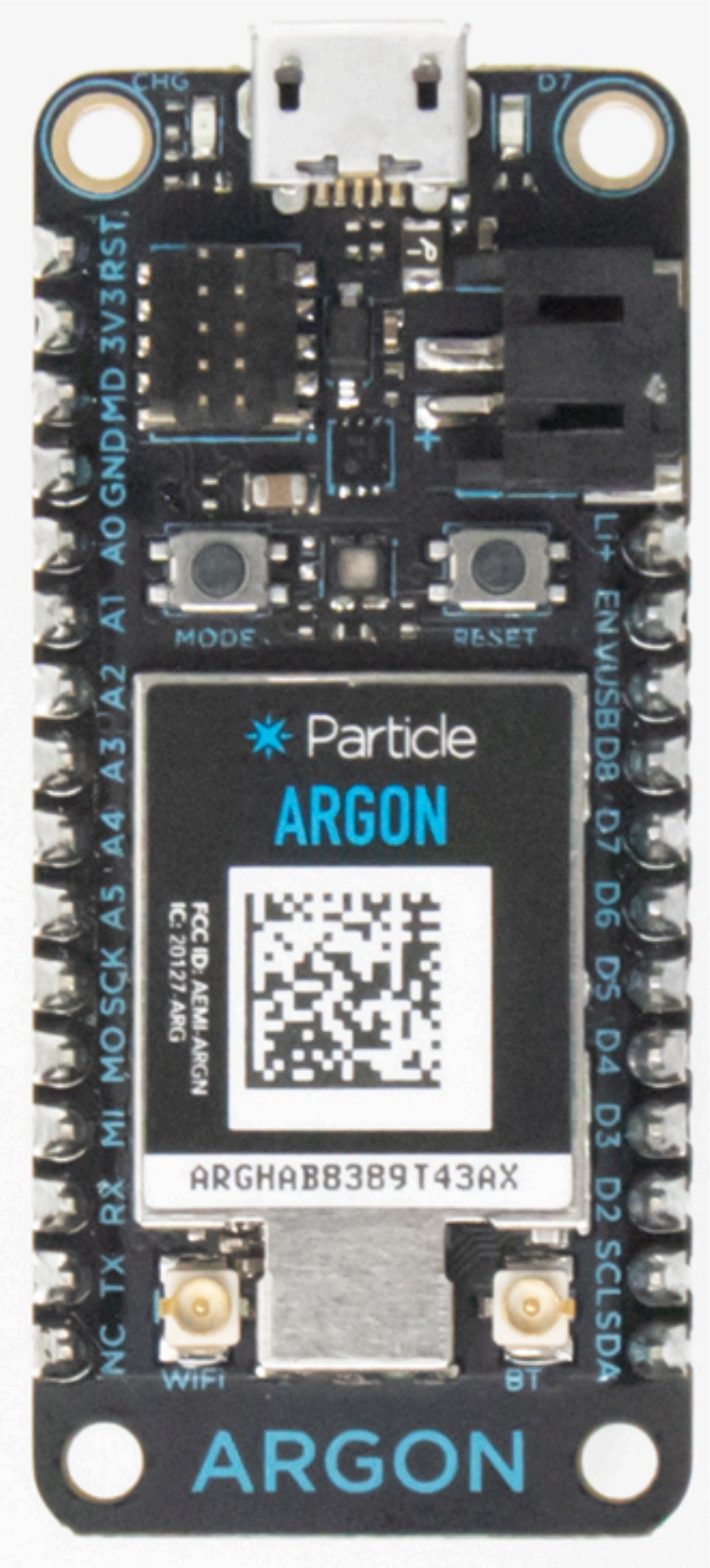
- Simple breakout board for evaluating the E-Series module
- Pins accessible via easy-to-use headers



E-Series Modules

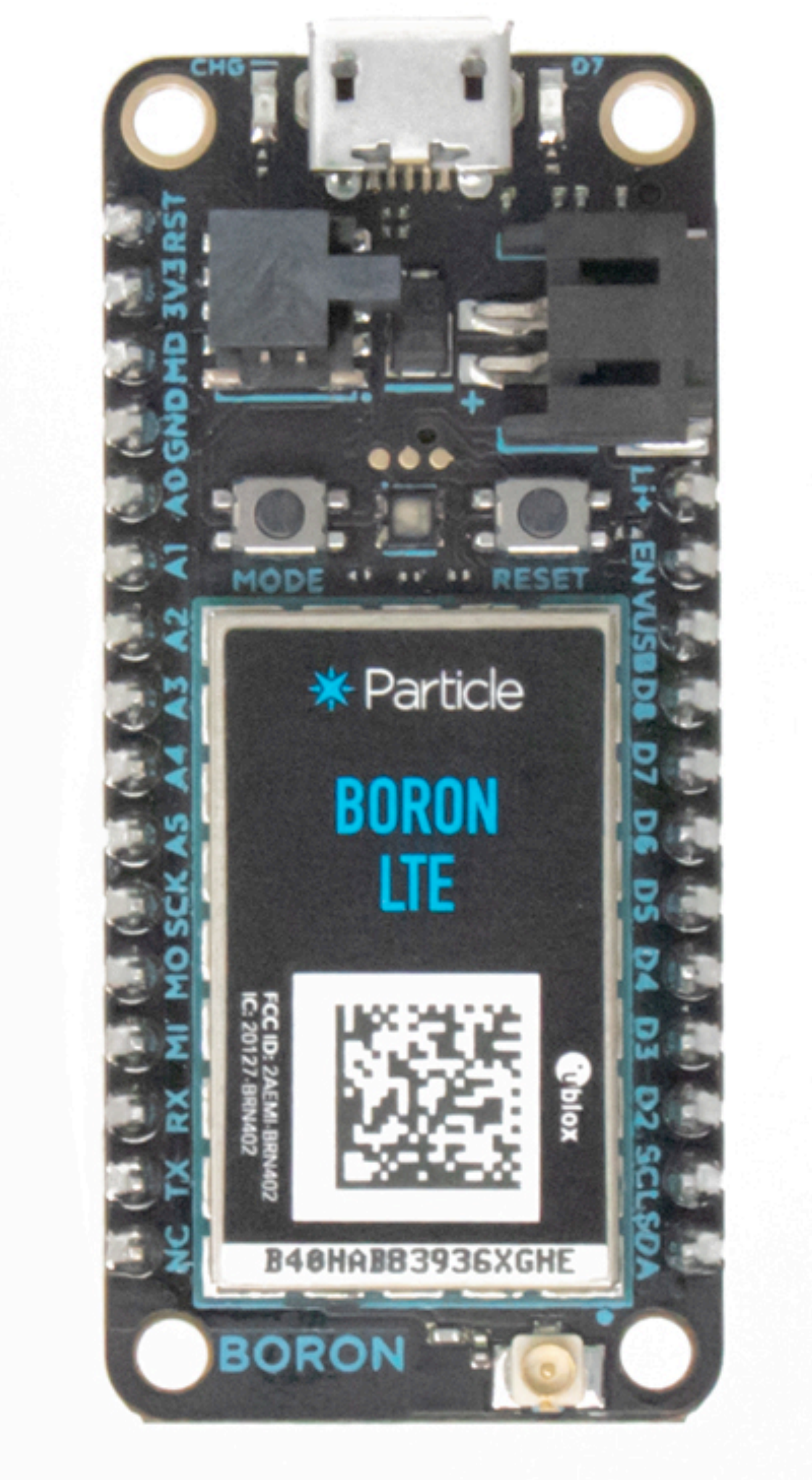
- Castellated edges for easy PCB inclusion
- Available in mfg-ready trays

PARTICLE 3RD GEN: MODERN DEVICES WITH BLE & MESH NETWORKING BAKED-IN



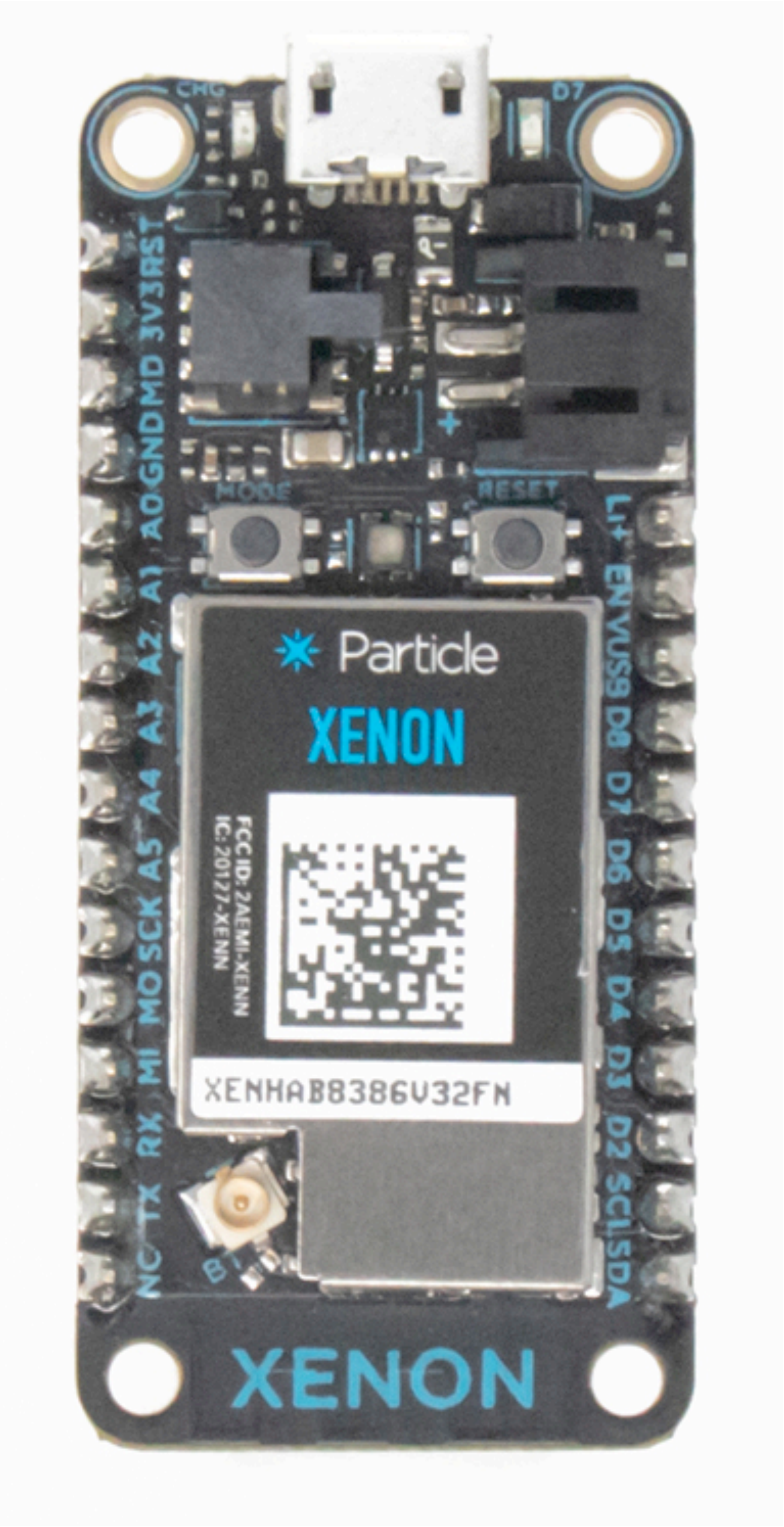
Argon

Wi-Fi, BLE & Mesh
Can function as a gateway, repeater and/or endpoint



Boron

LTE CAT-M1, BLE & Mesh
Can function as a gateway, repeater and/or endpoint



Xenon

BLE & Mesh
Can function as a repeater and/or endpoint

WHY PARTICLE

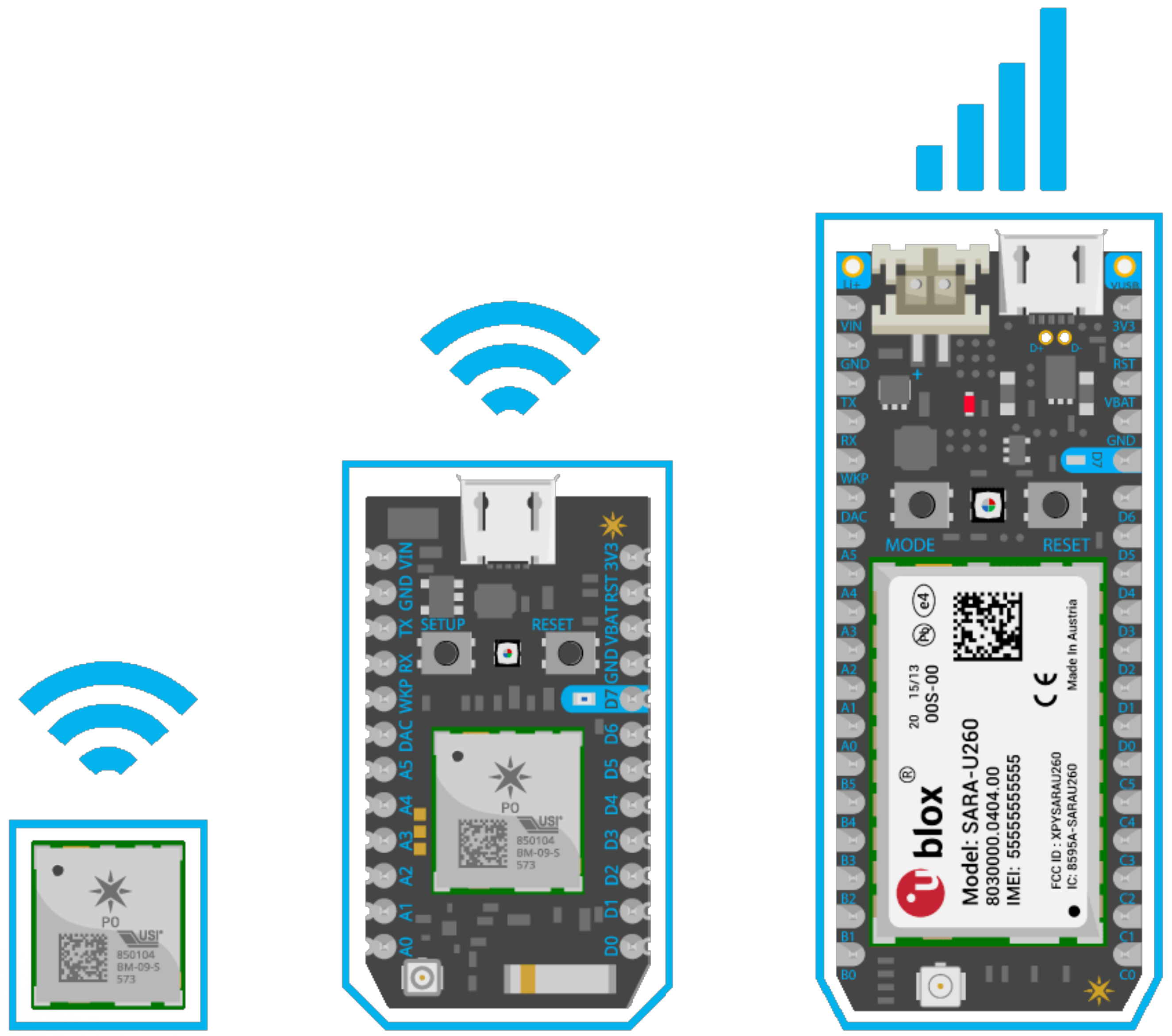
THE PARTICLE CLOUD & FRIENDS

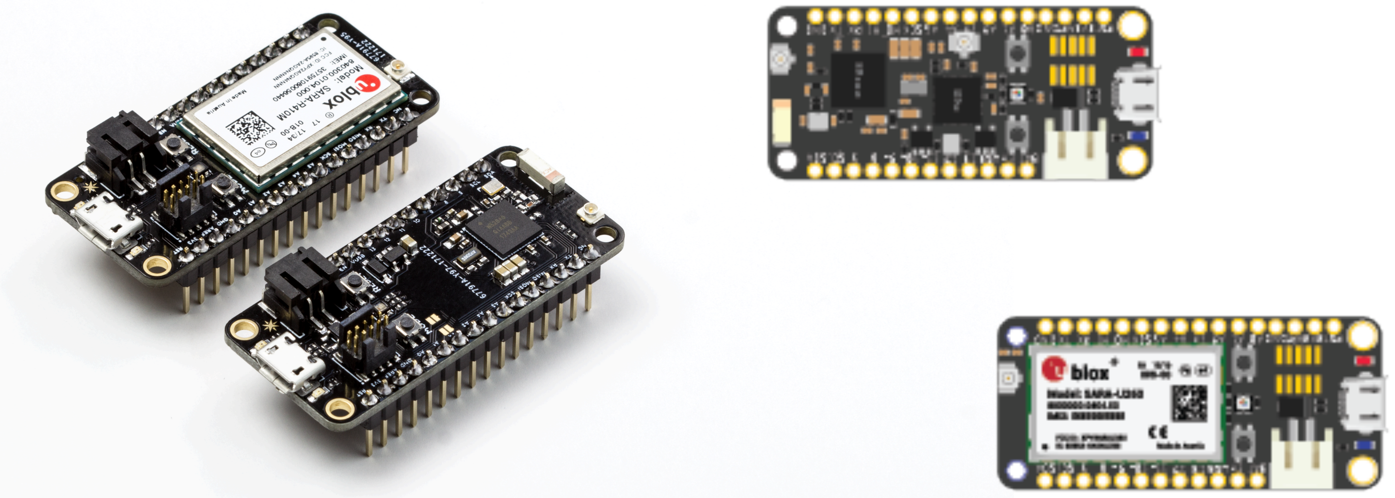
CLAIMING YOUR FIRST DEVICE

THE PARTICLE ECOSYSTEM: HARDWARE, FIRMWARE, & SOFTWARE

THE PARTICLE ECOSYSTEM: HARDWARE, FIRMWARE, & SOFTWARE

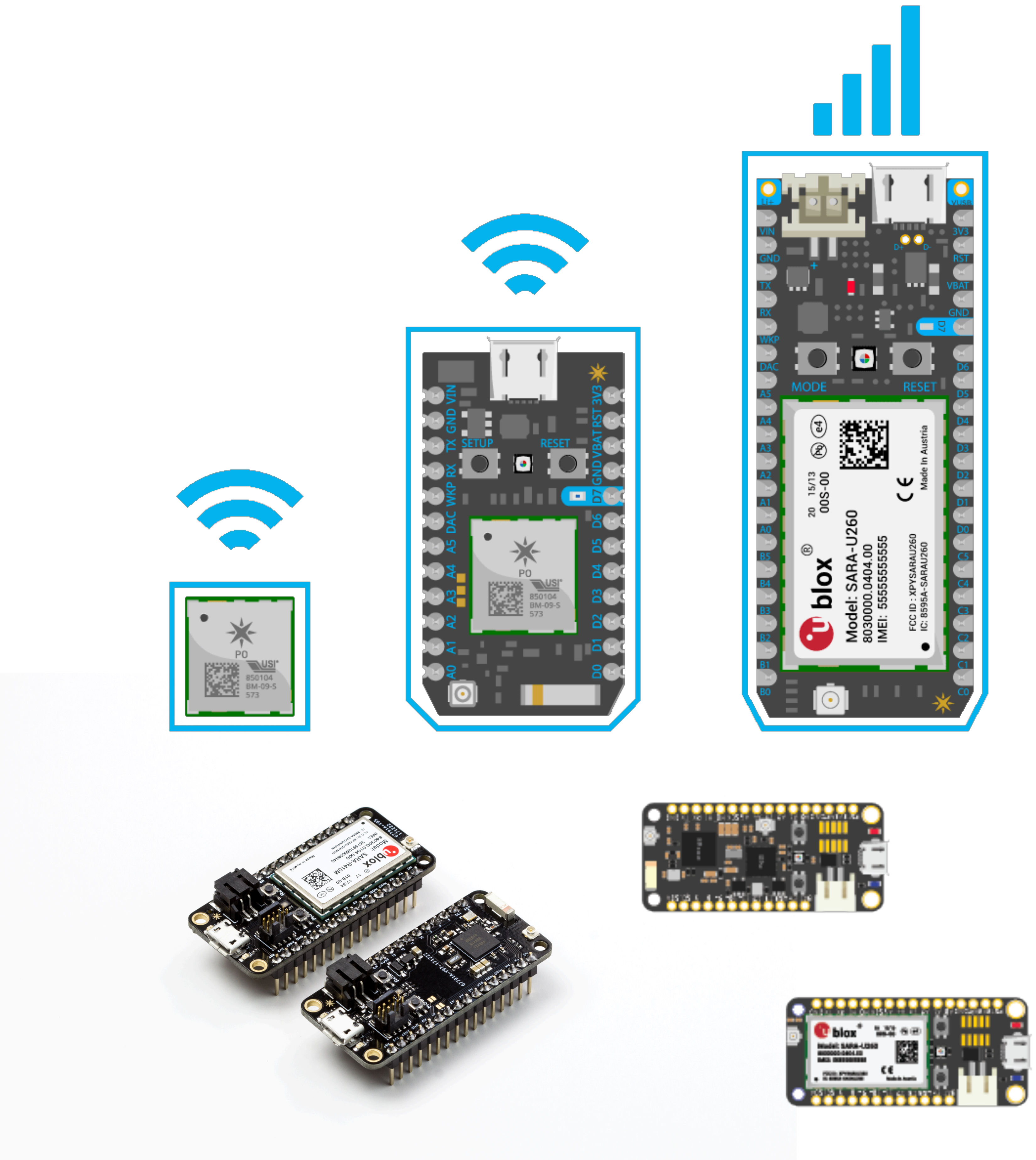
DEVICE HARDWARE FOR PROTOTYPING & PRODUCTION





THE PARTICLE ECOSYSTEM: HARDWARE, FIRMWARE, & SOFTWARE

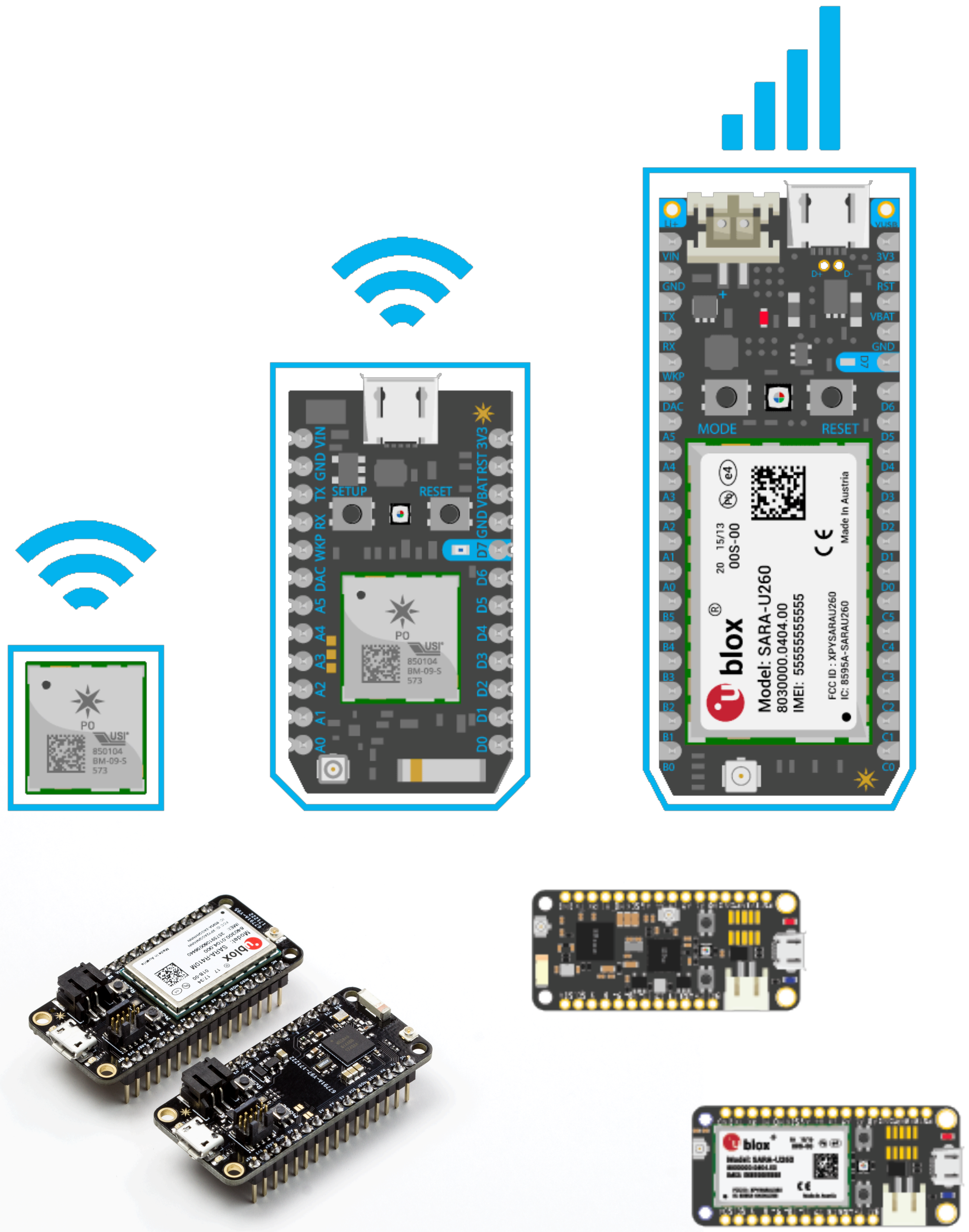
DEVICE HARDWARE FOR PROTOTYPING & PRODUCTION



DEVICE OS FIRMWARE & LIBRARIES

THE PARTICLE ECOSYSTEM: HARDWARE, FIRMWARE, & SOFTWARE

DEVICE HARDWARE FOR PROTOTYPING & PRODUCTION



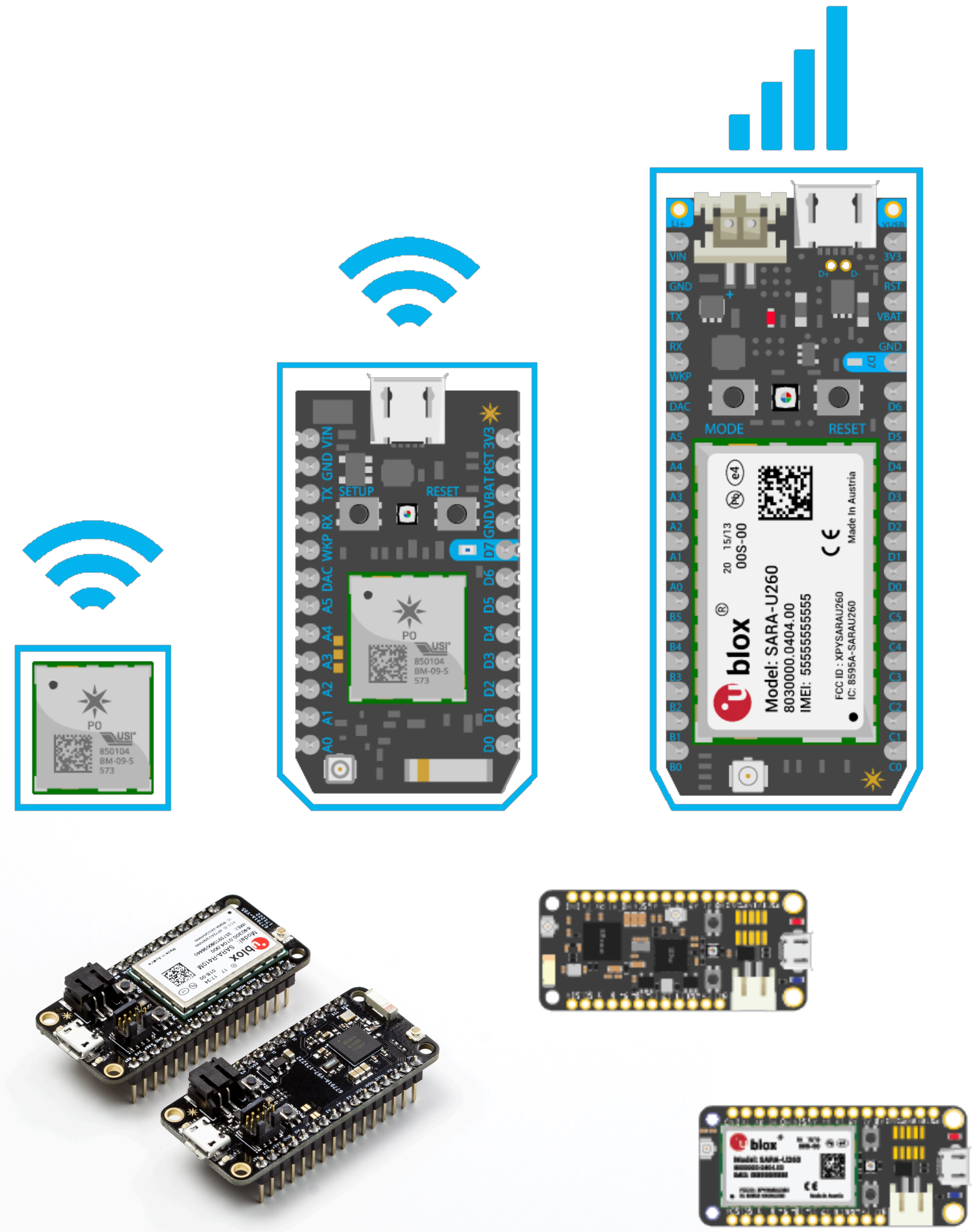
DEVICE OS FIRMWARE & LIBRARIES

DEVICE CLOUD & SOFTWARE TOOLS

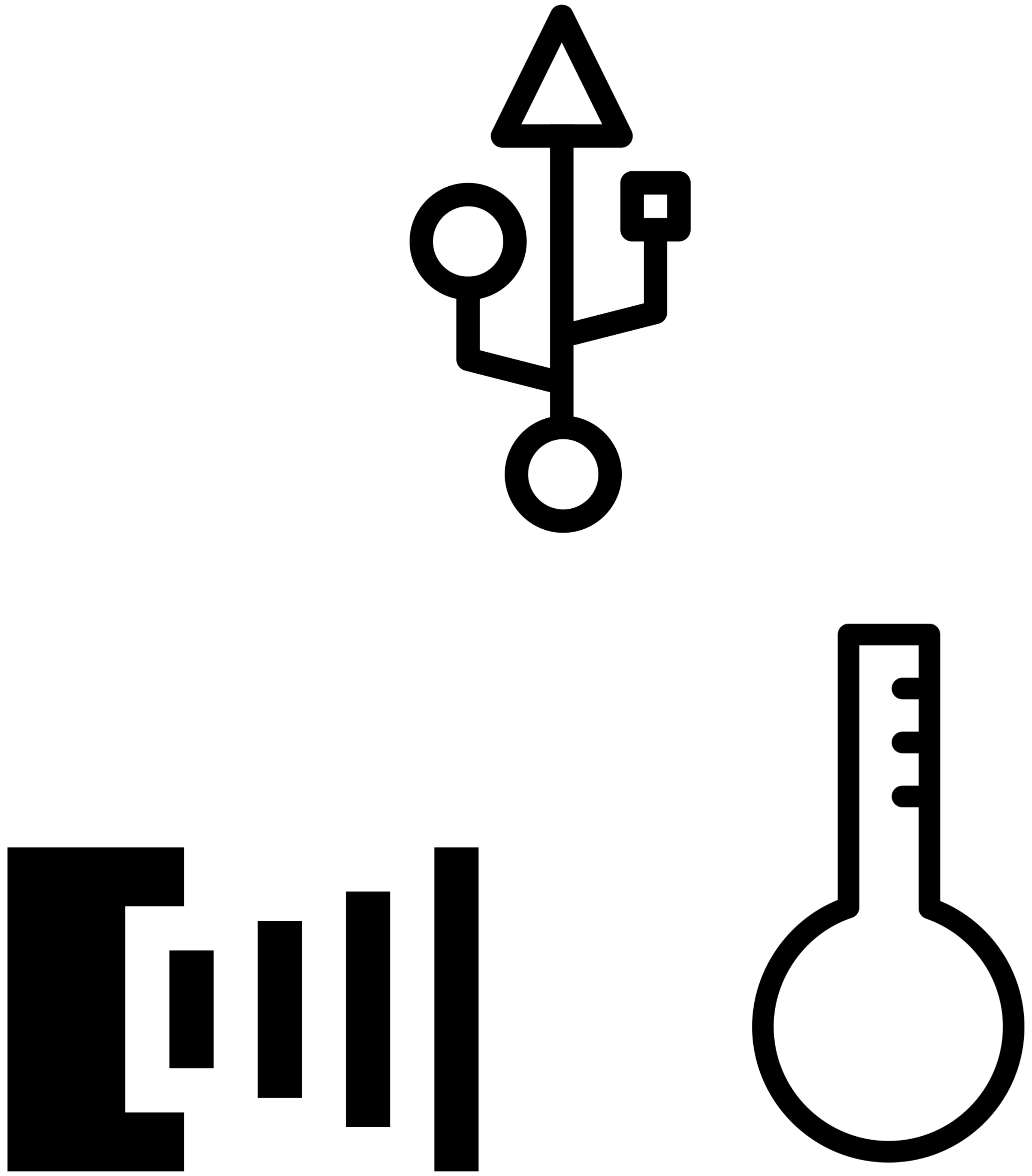
JS

THE PARTICLE ECOSYSTEM: HARDWARE, FIRMWARE, & SOFTWARE

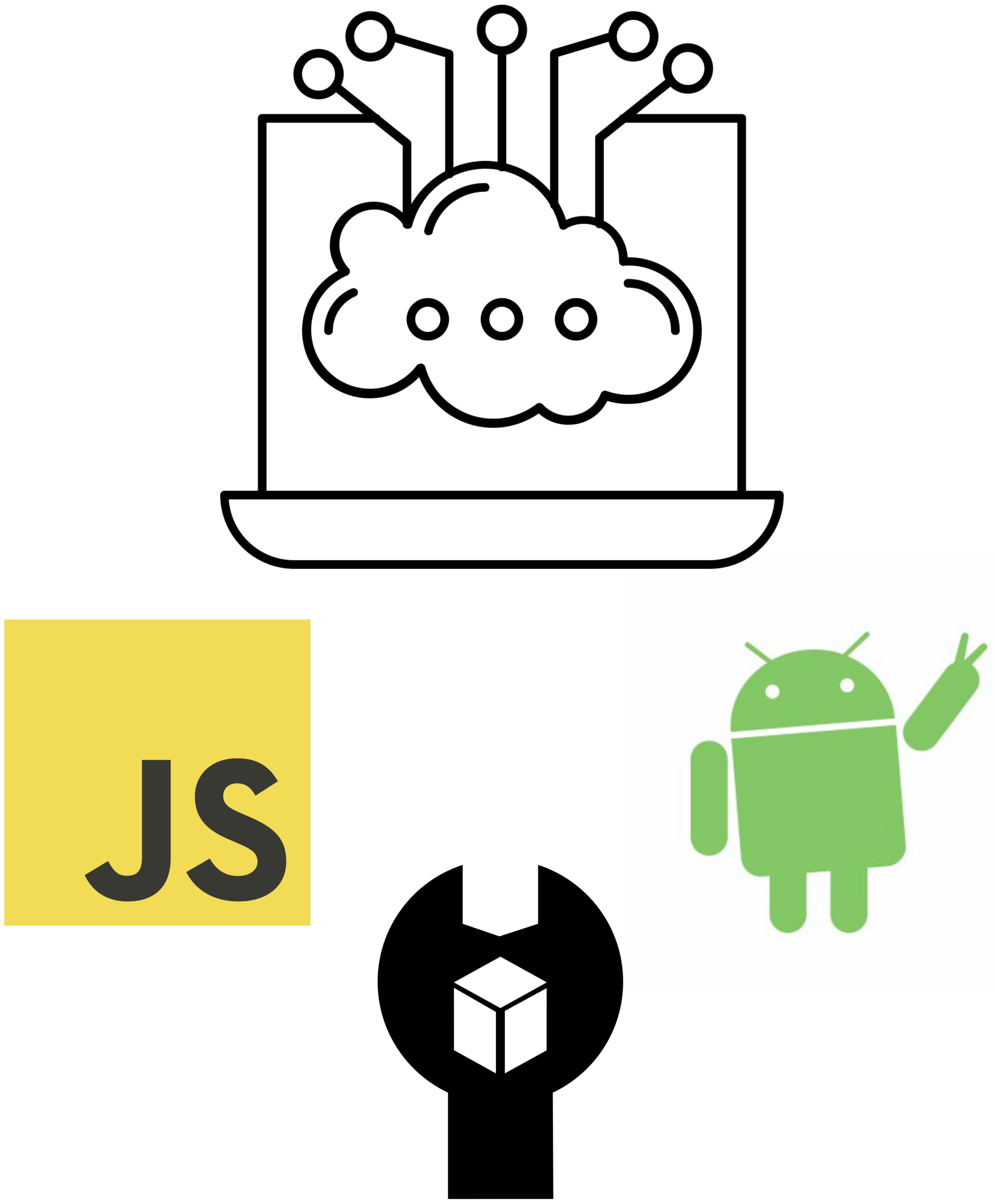
DEVICE HARDWARE FOR PROTOTYPING & PRODUCTION



DEVICE OS FIRMWARE & LIBRARIES

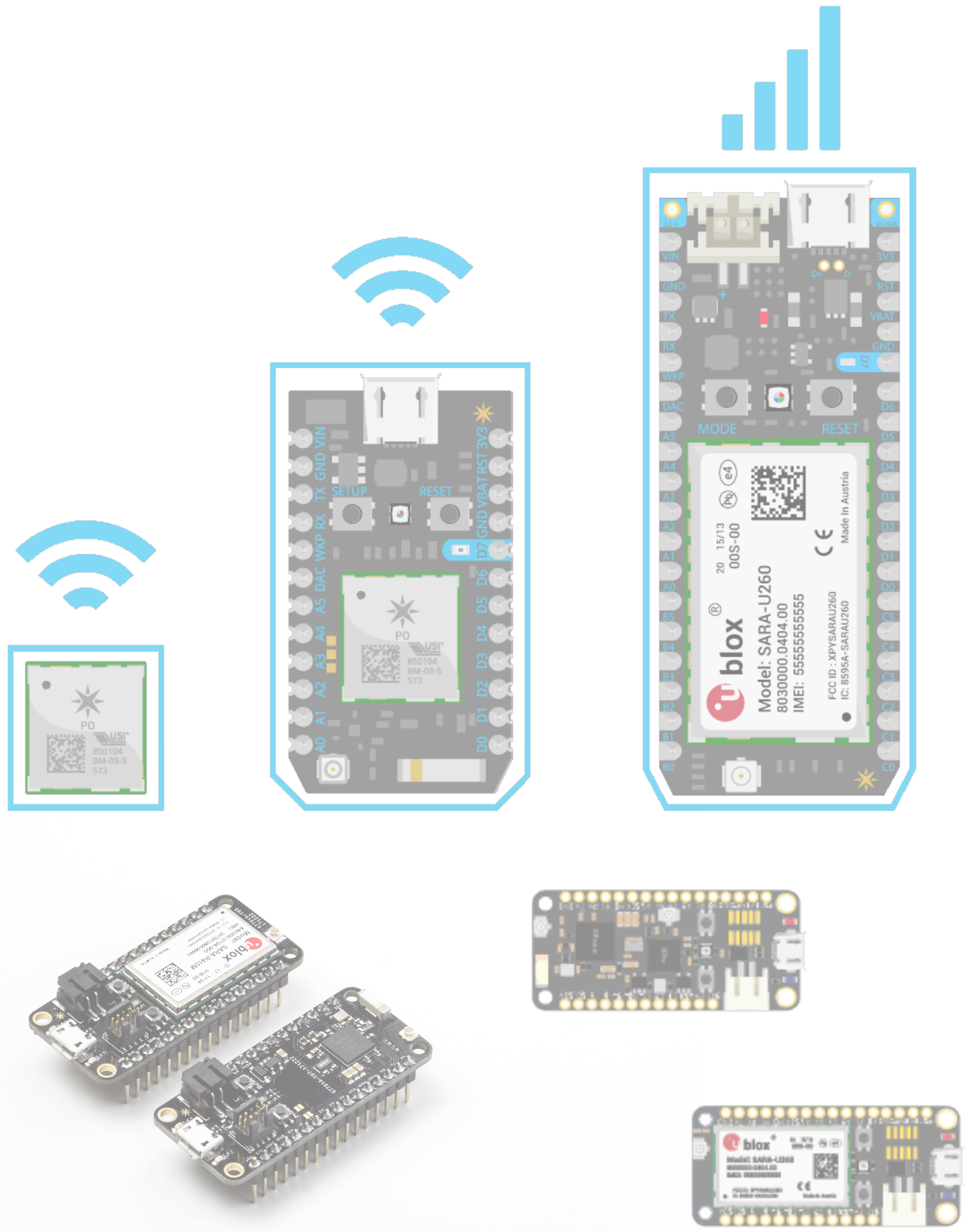


DEVICE CLOUD & SOFTWARE TOOLS



THE PARTICLE ECOSYSTEM: HARDWARE, FIRMWARE, & SOFTWARE

DEVICE HARDWARE FOR PROTOTYPING & PRODUCTION



DEVICE OS FIRMWARE & LIBRARIES

DEVICE CLOUD & SOFTWARE TOOLS

PARTICLE DEVICE OS – POWERFUL FIRMWARE WITH TONS OF FEATURES

Call a function, remotely
Particle.function()

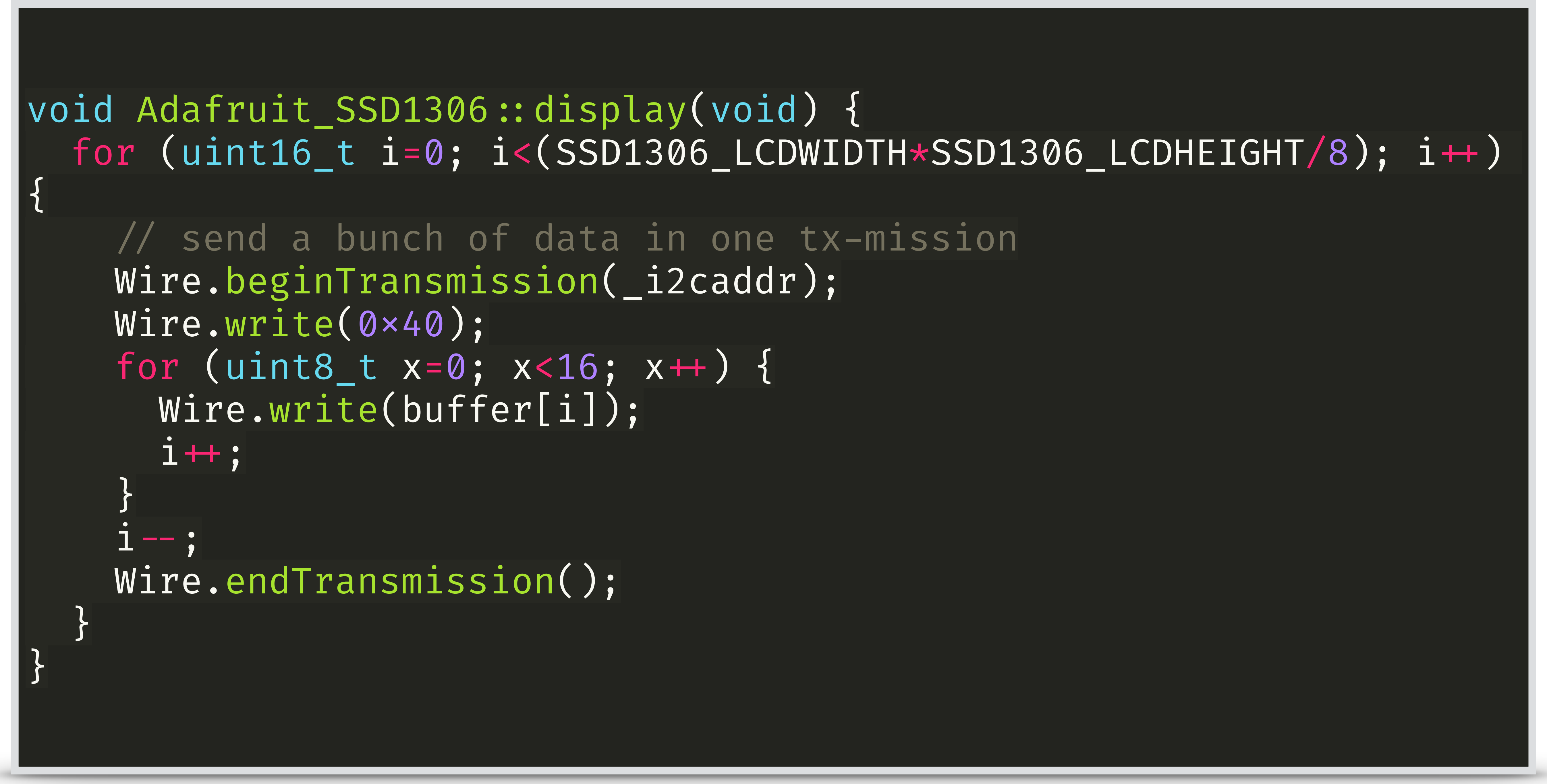
Fetch a variable, remotely
Particle.variable()

Send an event to the cloud
Particle.publish()

Listen for events
Particle.subscribe()

ADVANTAGES OF USING FIRMWARE LIBRARIES

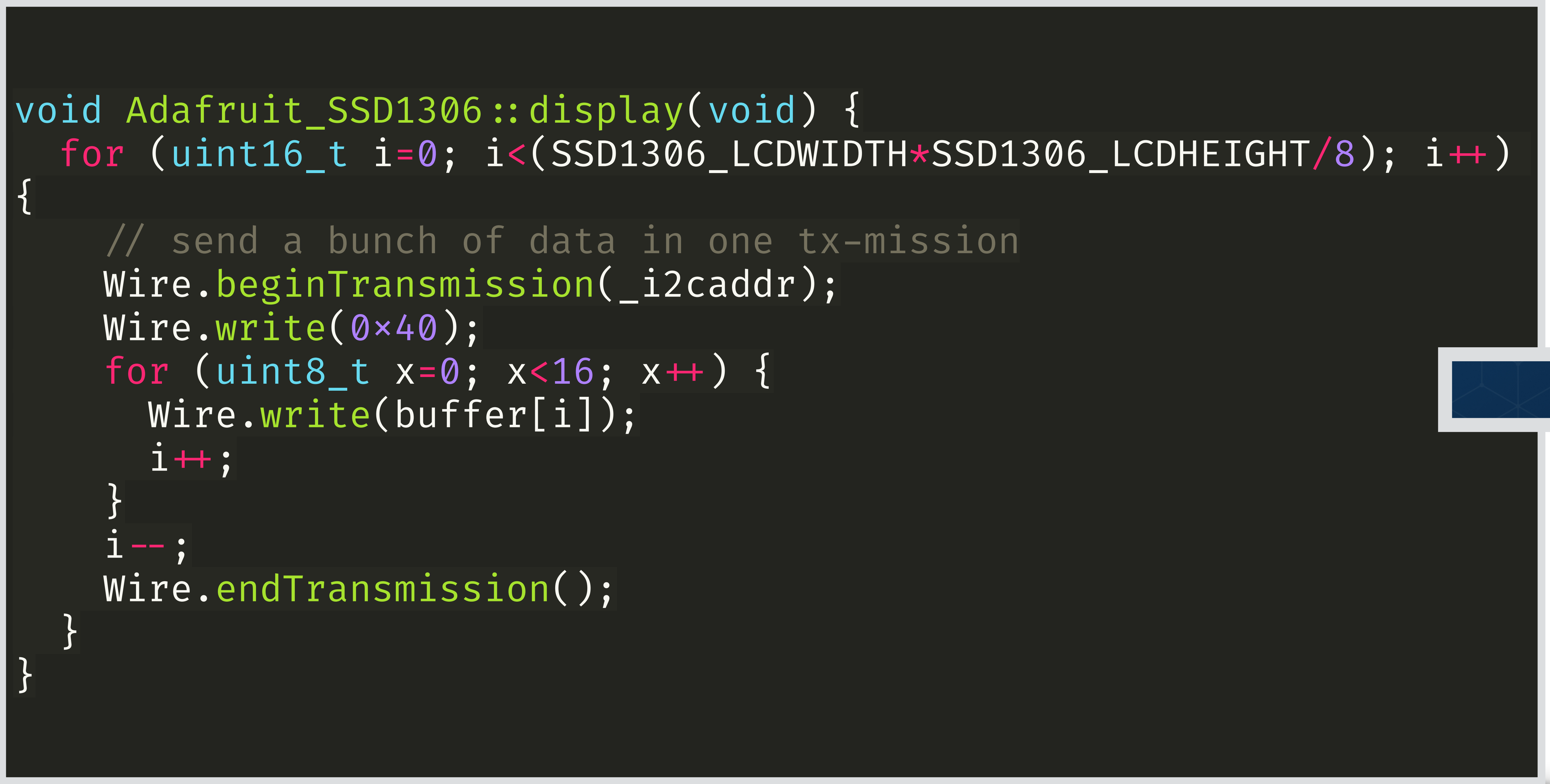
- Simplify interactions with sensors and actuators
- Reuse code across projects
- Leverage contributions from Particle’s community of 165k developers



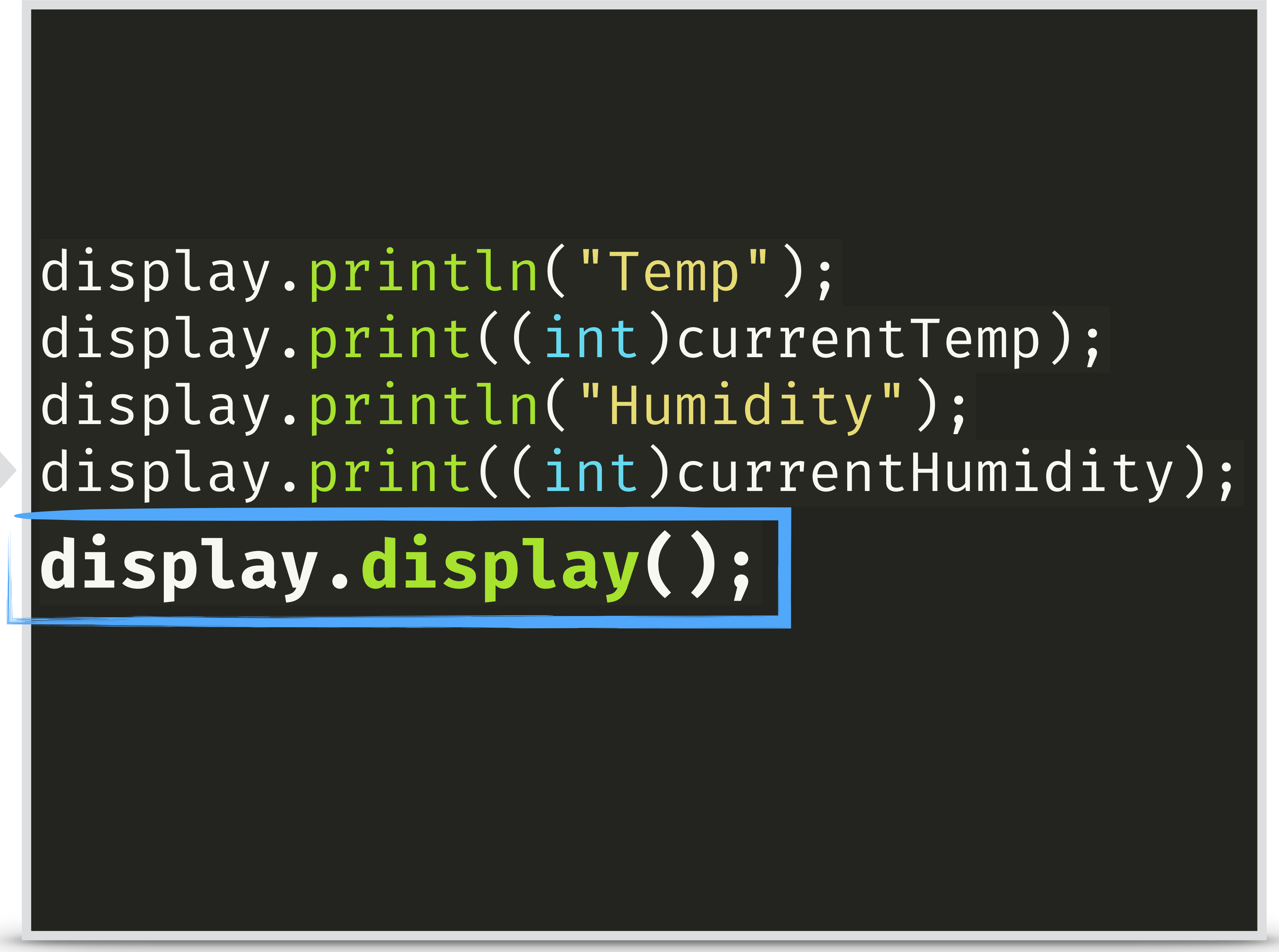
```
void Adafruit_SSD1306::display(void) {
  for (uint16_t i=0; i<(SSD1306_LCDWIDTH*SSD1306_LCDHEIGHT/8); i++)
{
    // send a bunch of data in one tx-mission
    Wire.beginTransmission(_i2caddr);
    Wire.write(0x40);
    for (uint8_t x=0; x<16; x++) {
      Wire.write(buffer[i]);
      i++;
    }
    i--;
    Wire.endTransmission();
  }
}
```

ADVANTAGES OF USING FIRMWARE LIBRARIES

- Simplify interactions with sensors and actuators
- Reuse code across projects
- Leverage contributions from Particle’s community of 165k developers



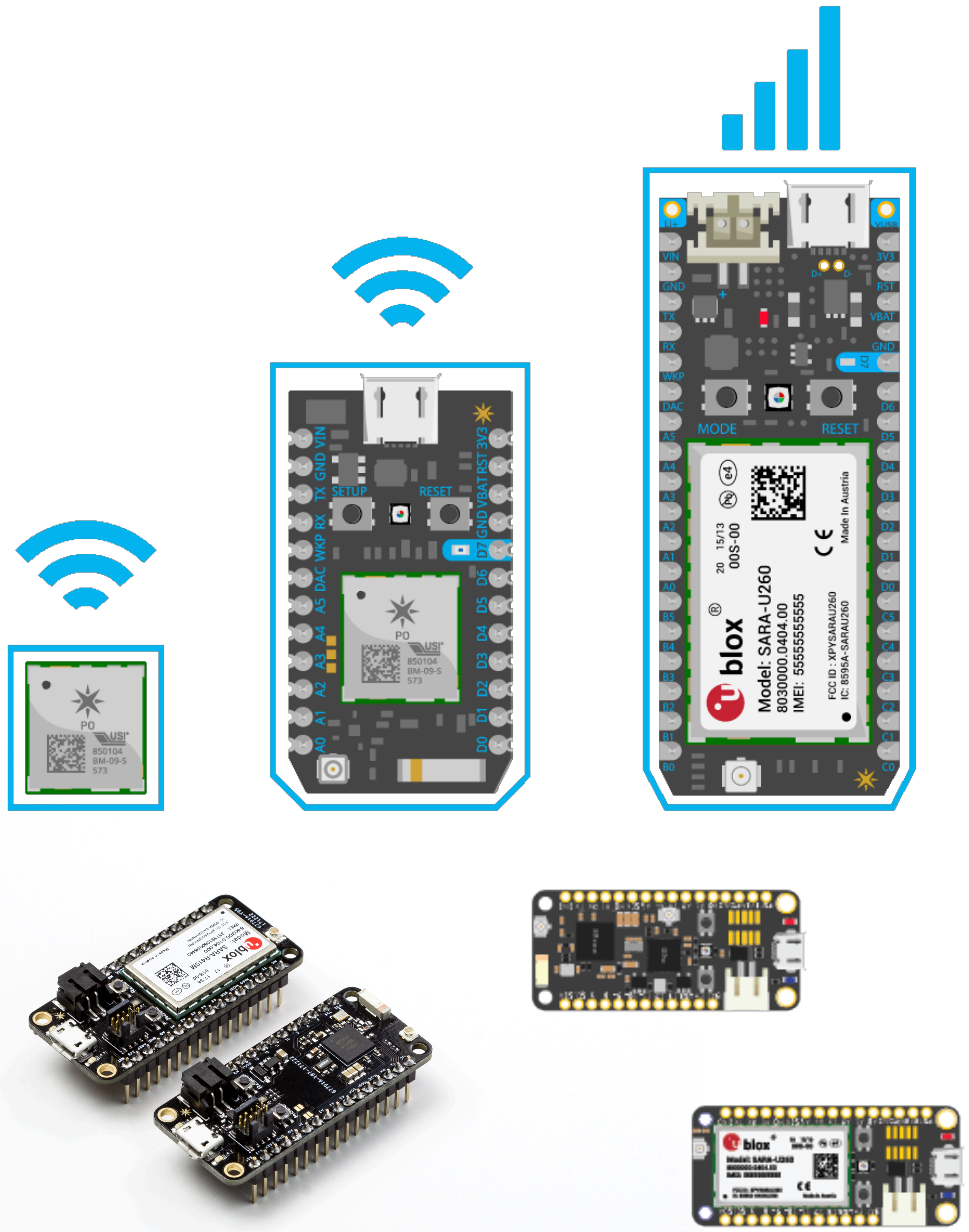
```
void Adafruit_SSD1306::display(void) {
  for (uint16_t i=0; i<(SSD1306_LCDWIDTH*SSD1306_LCDHEIGHT/8); i++)
{
    // send a bunch of data in one tx-mission
    Wire.beginTransmission(_i2caddr);
    Wire.write(0x40);
    for (uint8_t x=0; x<16; x++) {
      Wire.write(buffer[i]);
      i++;
    }
    i--;
    Wire.endTransmission();
  }
}
```



```
display.println("Temp");
display.print((int)currentTemp);
display.println("Humidity");
display.print((int)currentHumidity);
display.display();
```

THE PARTICLE ECOSYSTEM: HARDWARE, FIRMWARE, & SOFTWARE

DEVICE HARDWARE FOR PROTOTYPING & PRODUCTION



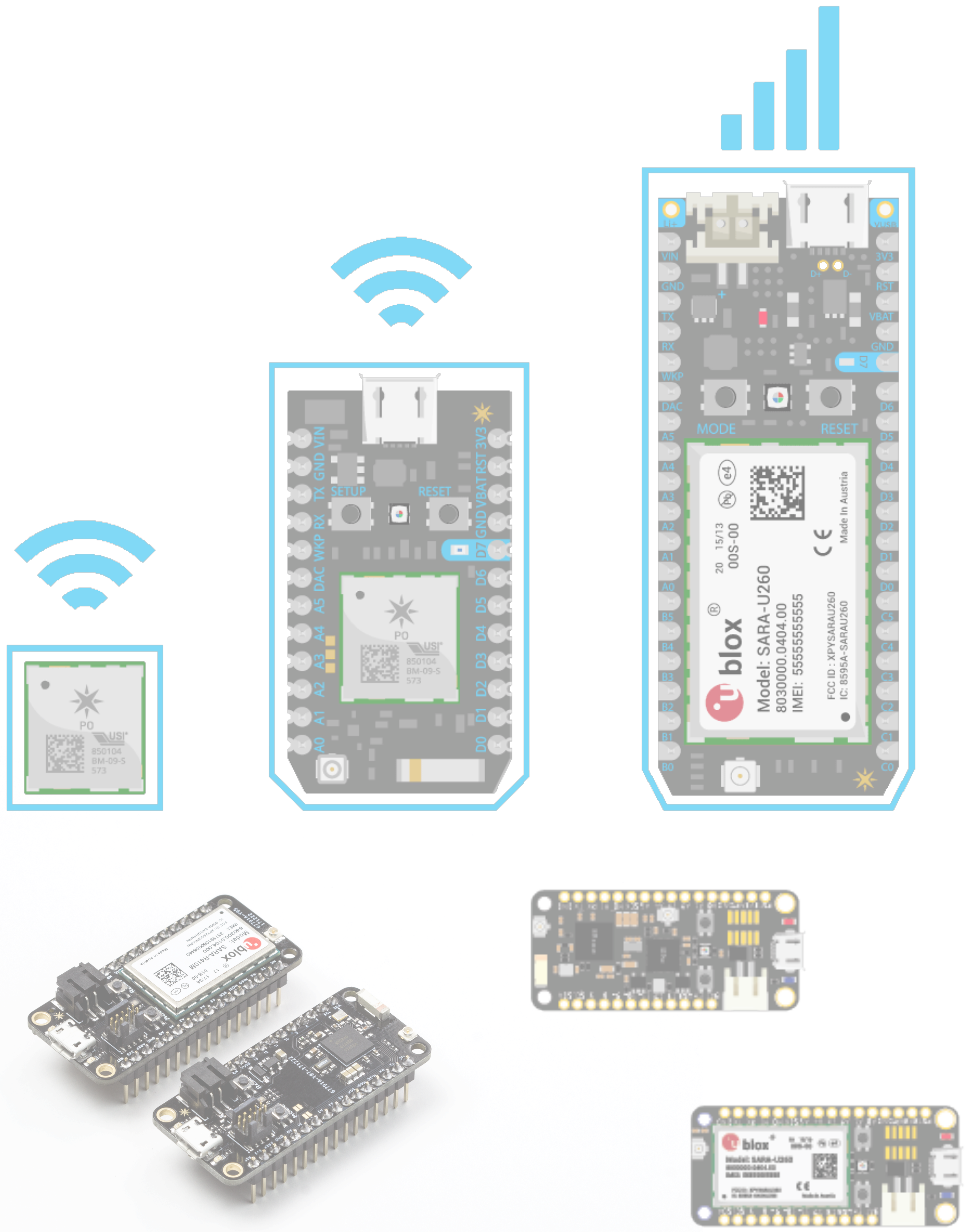
DEVICE OS FIRMWARE & LIBRARIES

DEVICE CLOUD & SOFTWARE TOOLS

JS

THE PARTICLE ECOSYSTEM: HARDWARE, FIRMWARE, & SOFTWARE

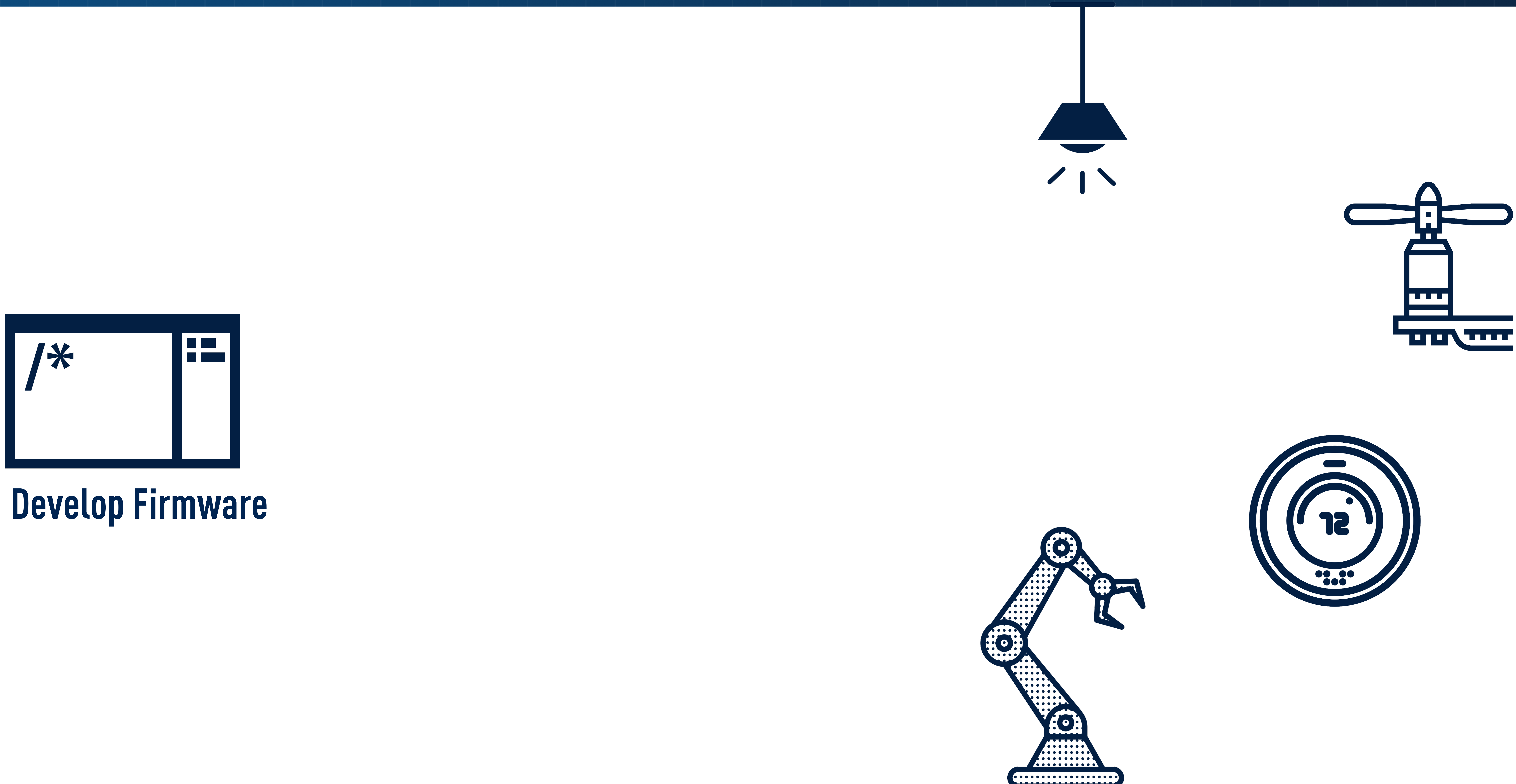
DEVICE HARDWARE FOR PROTOTYPING & PRODUCTION



DEVICE OS FIRMWARE & LIBRARIES

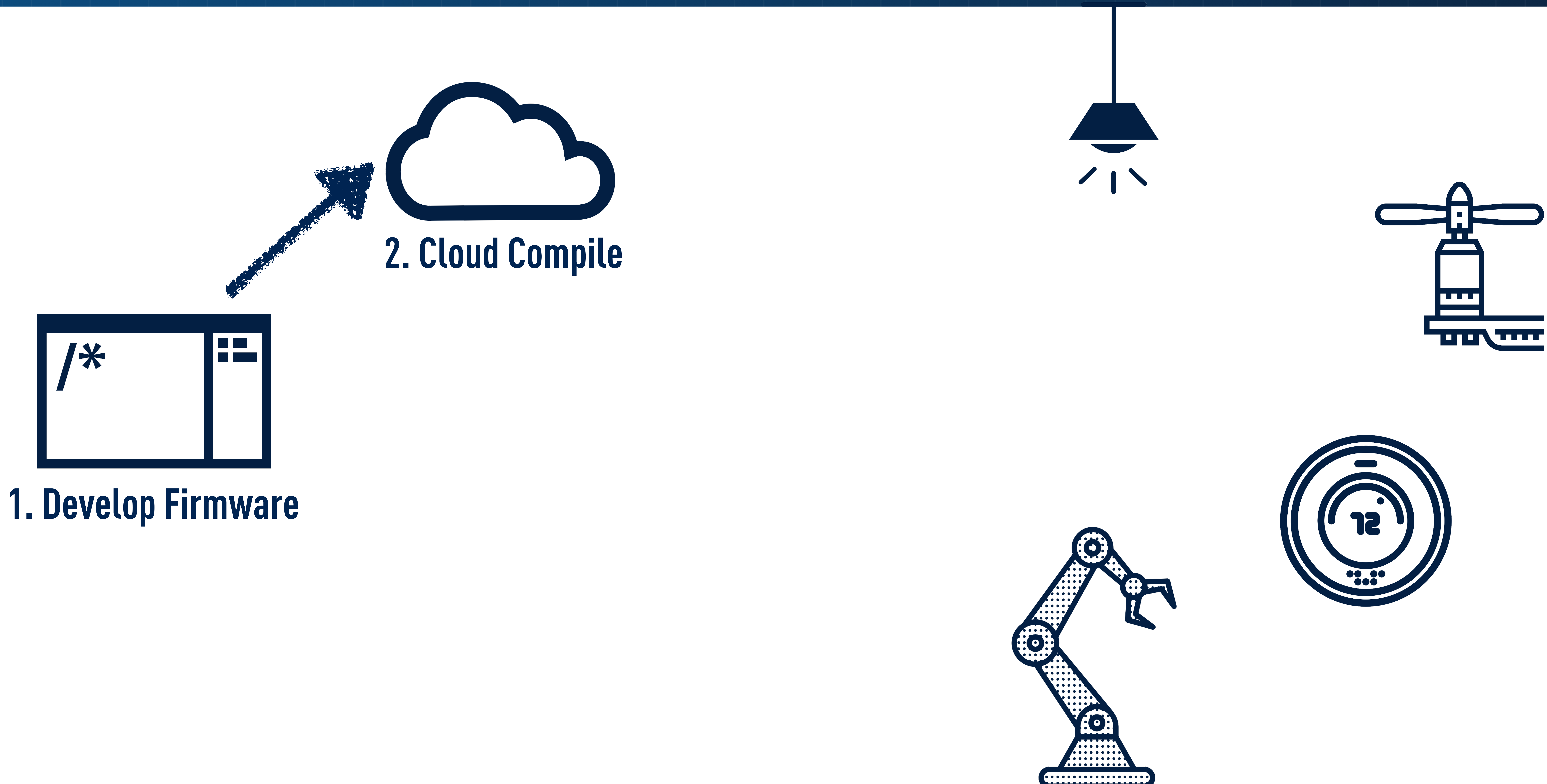
DEVICE CLOUD & SOFTWARE TOOLS

OVER THE AIR (OTA) DEVICE UPDATES



1. Develop Firmware

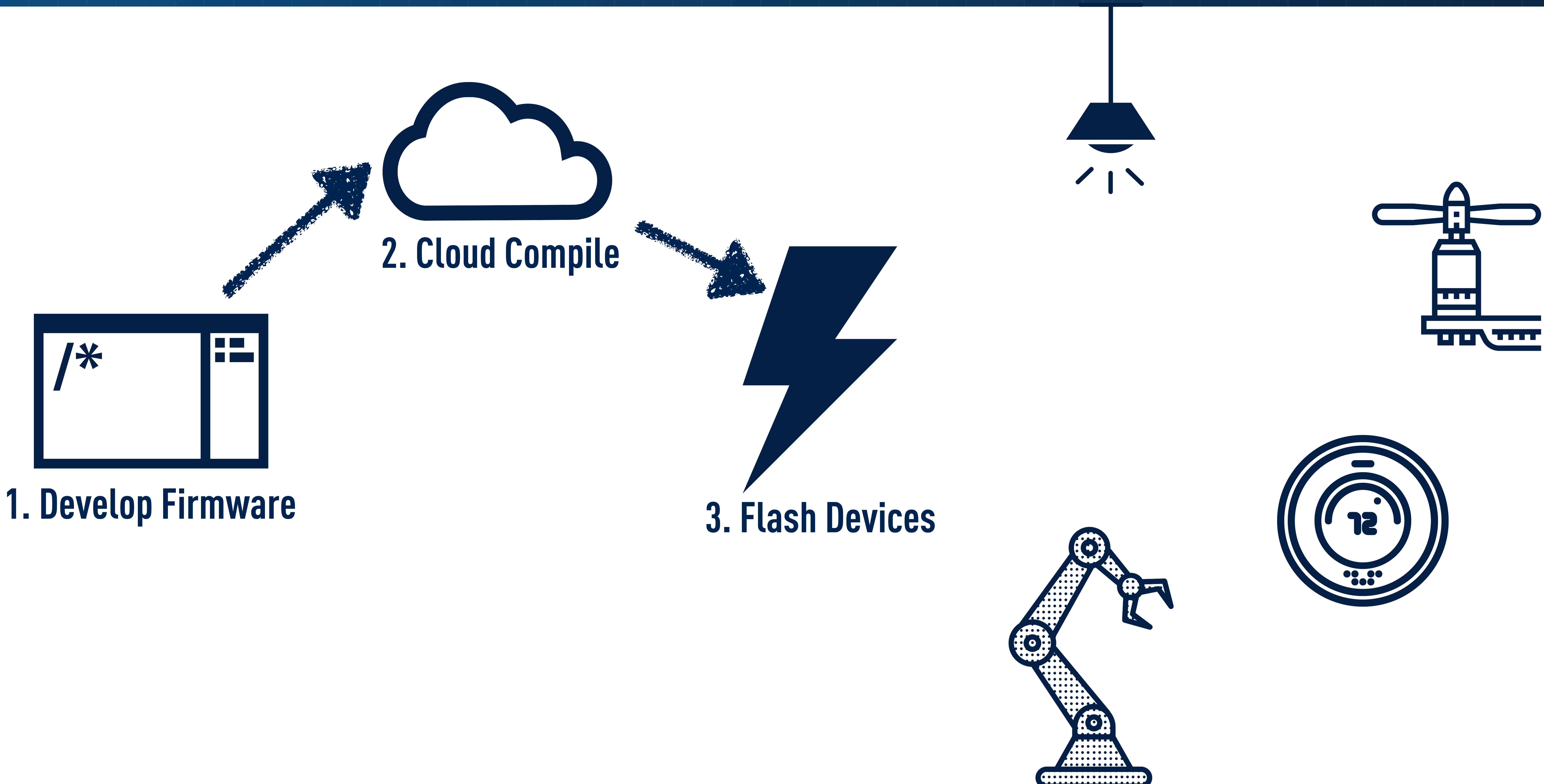
OVER THE AIR (OTA) DEVICE UPDATES



1. Develop Firmware

2. Cloud Compile

OVER THE AIR (OTA) DEVICE UPDATES

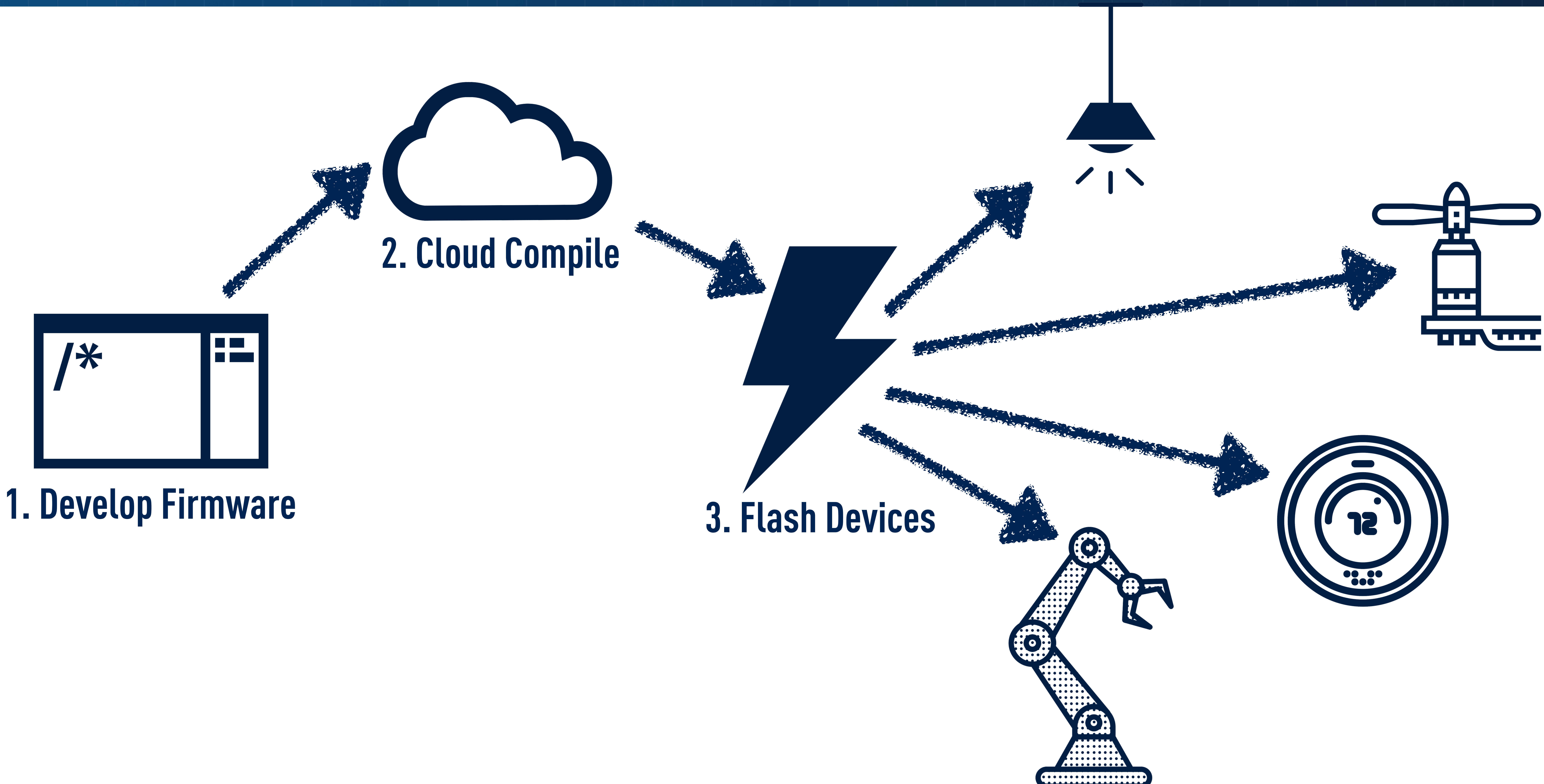


1. Develop Firmware

2. Cloud Compile

3. Flash Devices

OVER THE AIR (OTA) DEVICE UPDATES



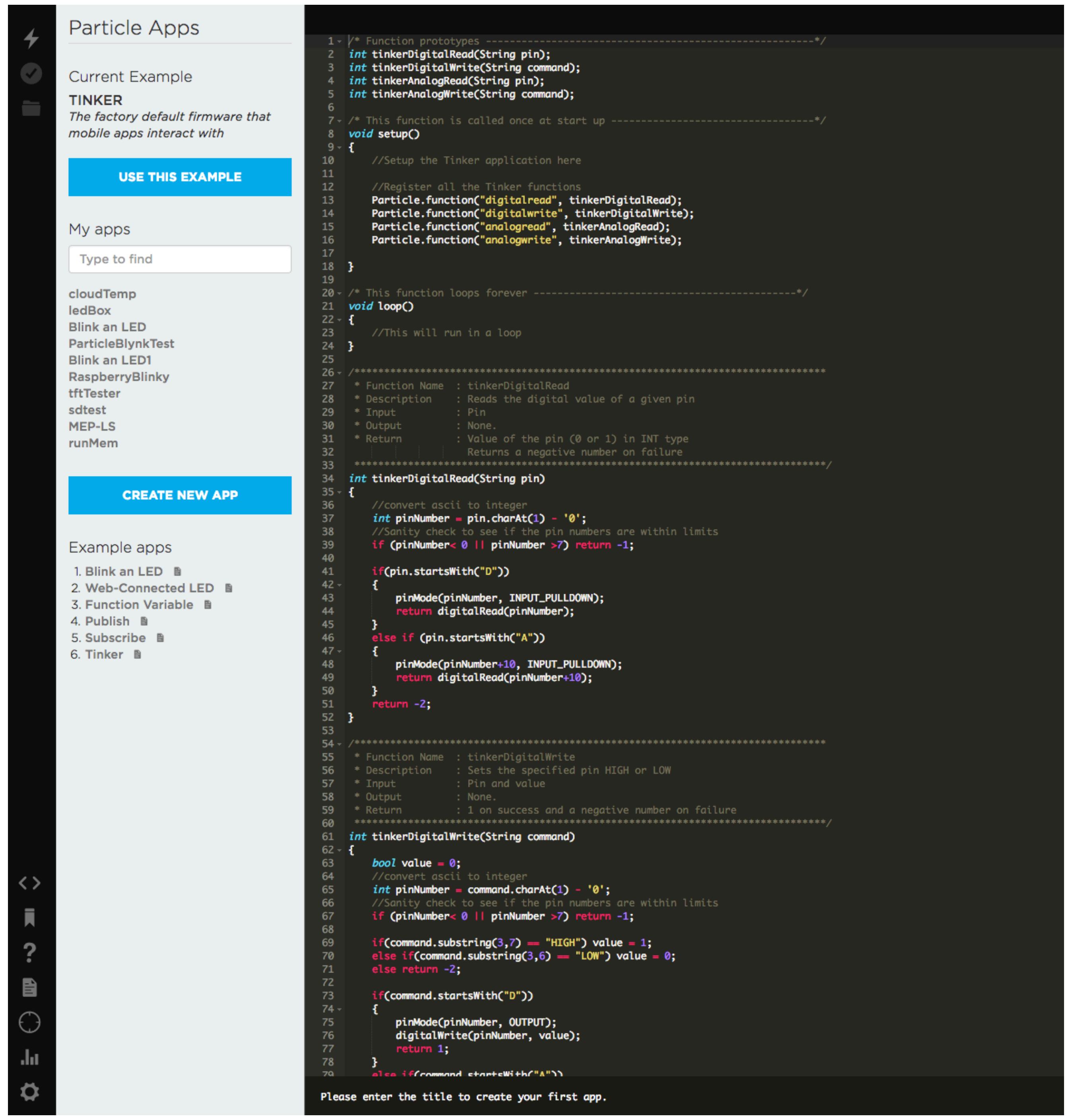
1. Develop Firmware

2. Cloud Compile

3. Flash Devices

IDES AND DEVELOPER TOOLING

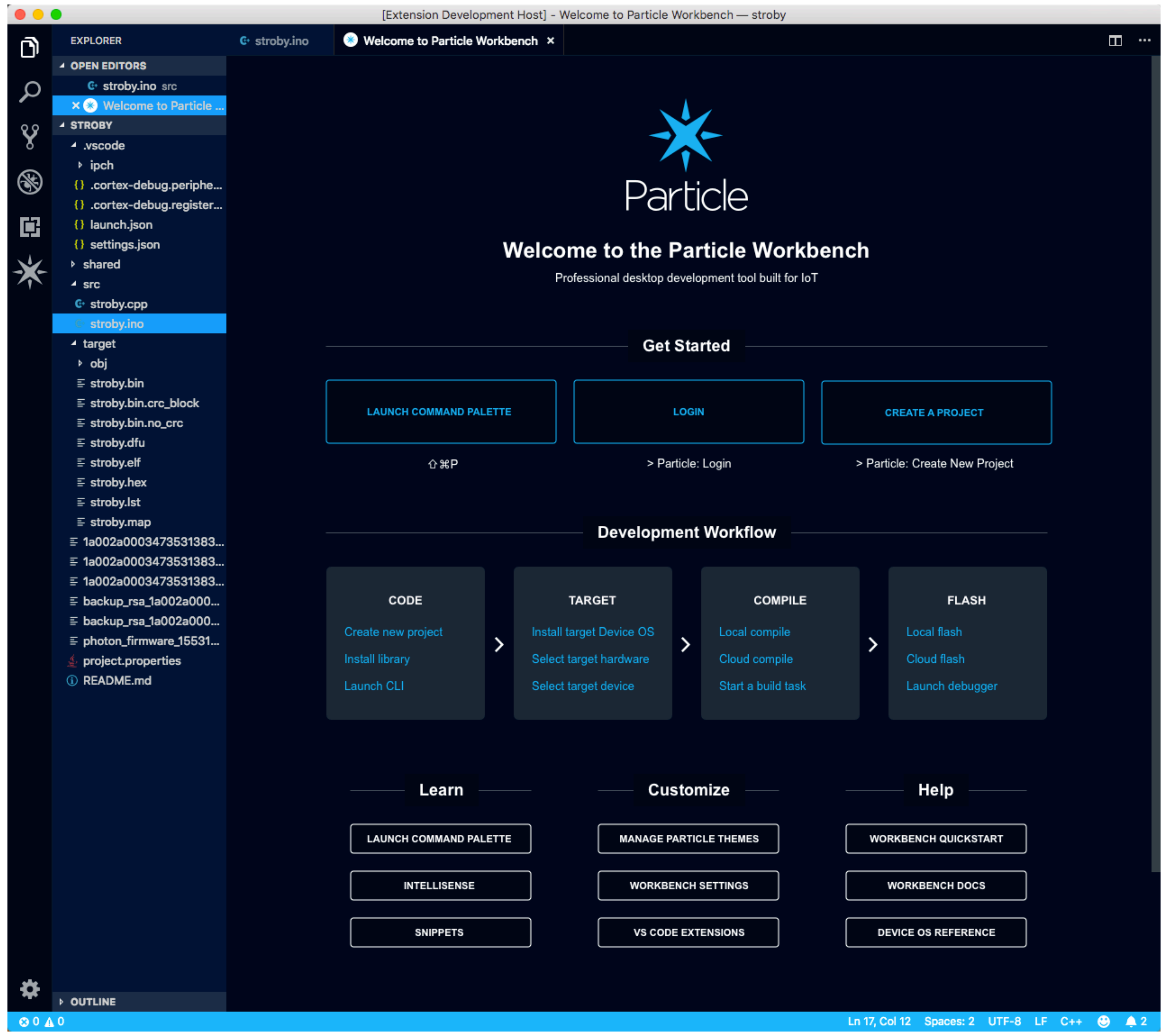
PARTICLE BUILD (WEB)



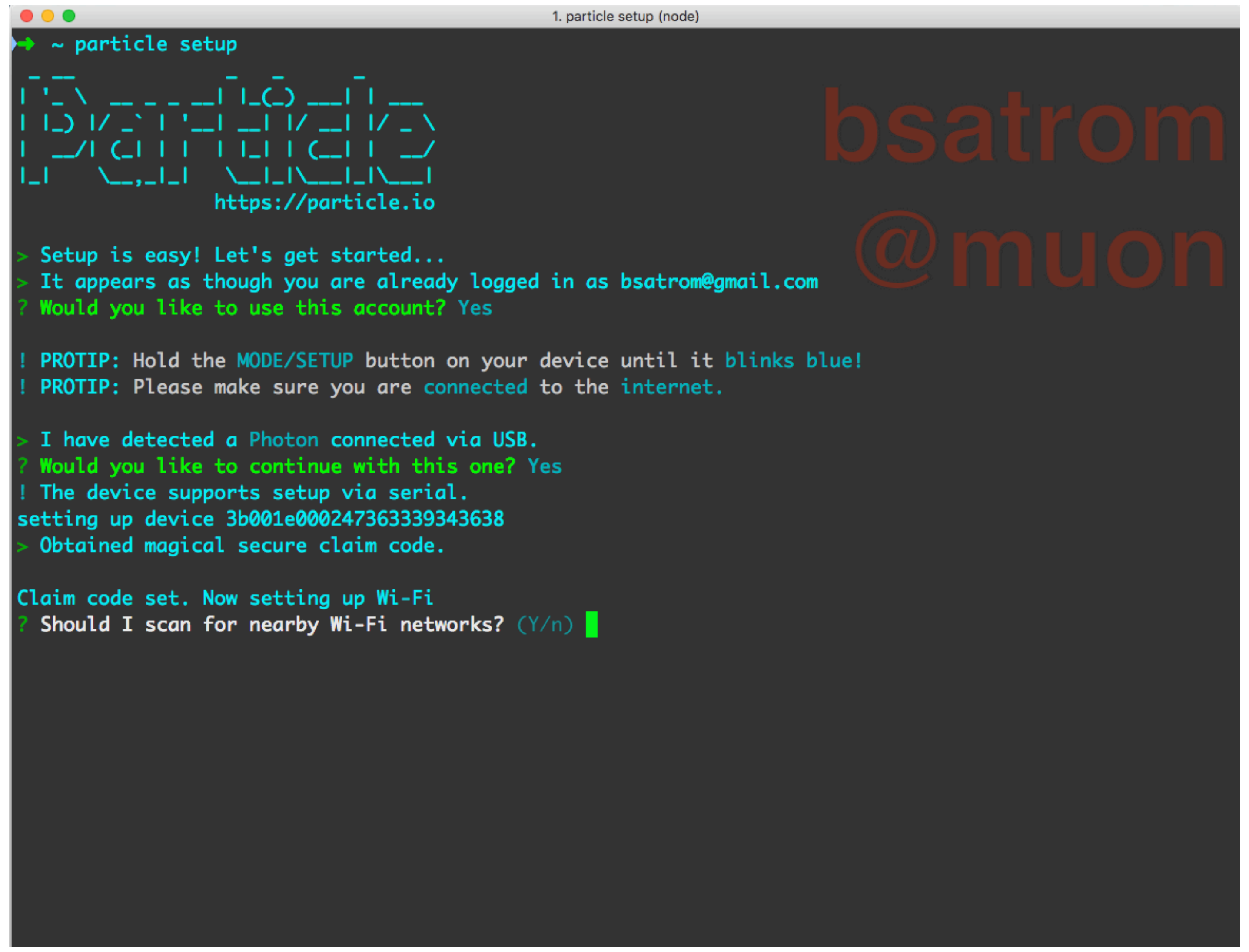
```
/* Function prototypes ------------------------------------------------------*/
int tinkerDigitalRead(String pin);
int tinkerDigitalWrite(String command);
int tinkerAnalogRead(String pin);
int tinkerAnalogWrite(String command);

/* This function is called once at start up ----------------------------------*/
void setup()
{
    //Setup the Tinker application here

    //Register all the Tinker functions
    Particle.function("digitalread", tinkerDigitalRead);
    Particle.function("digitalwrite", tinkerDigitalWrite);
    Particle.function("analogread", tinkerAnalogRead);
    Particle.function("analogwrite", tinkerAnalogWrite);

}

/* This function loops forever --------------------------------------------*/
void loop()
{
    //This will run in a loop
}

/*******************************************************************************
 * Function Name  : tinkerDigitalRead
 * Description    : Reads the digital value of a given pin
 * Input          : Pin
 * Output         : None.
 * Return         : Value of the pin (0 or 1) in INT type
                    Returns a negative number on failure
 *******************************************************************************/
int tinkerDigitalRead(String pin)
{
    //convert ascii to integer
    int pinNumber = pin.charAt(1) - '0';
    //Sanity check to see if the pin numbers are within limits
    if (pinNumber< 0 || pinNumber >7) return -1;

    if(pin.startsWith("D"))
    {
        pinMode(pinNumber, INPUT_PULLDOWN);
        return digitalRead(pinNumber);
    }
    else if (pin.startsWith("A"))
    {
        pinMode(pinNumber+10, INPUT_PULLDOWN);
        return digitalRead(pinNumber+10);
    }
    return -2;
}

/*******************************************************************************
 * Function Name  : tinkerDigitalWrite
 * Description    : Sets the specified pin HIGH or LOW
 * Input          : Pin and value
 * Output         : None.
 * Return         : 1 on success and a negative number on failure
 *******************************************************************************/
int tinkerDigitalWrite(String command)
{
    bool value = 0;
    //convert ascii to integer
    int pinNumber = command.charAt(1) - '0';
    //Sanity check to see if the pin numbers are within limits
    if (pinNumber< 0 || pinNumber >7) return -1;

    if(command.substring(3,7) == "HIGH") value = 1;
    else if(command.substring(3,6) == "LOW") value = 0;
    else return -2;

    if(command.startsWith("D"))
    {
        pinMode(pinNumber, OUTPUT);
        digitalWrite(pinNumber, value);
        return 1;
    }
    else if(command.startsWith("A"))
```

Please enter the title to create your first app.

PARTICLE WORKBENCH

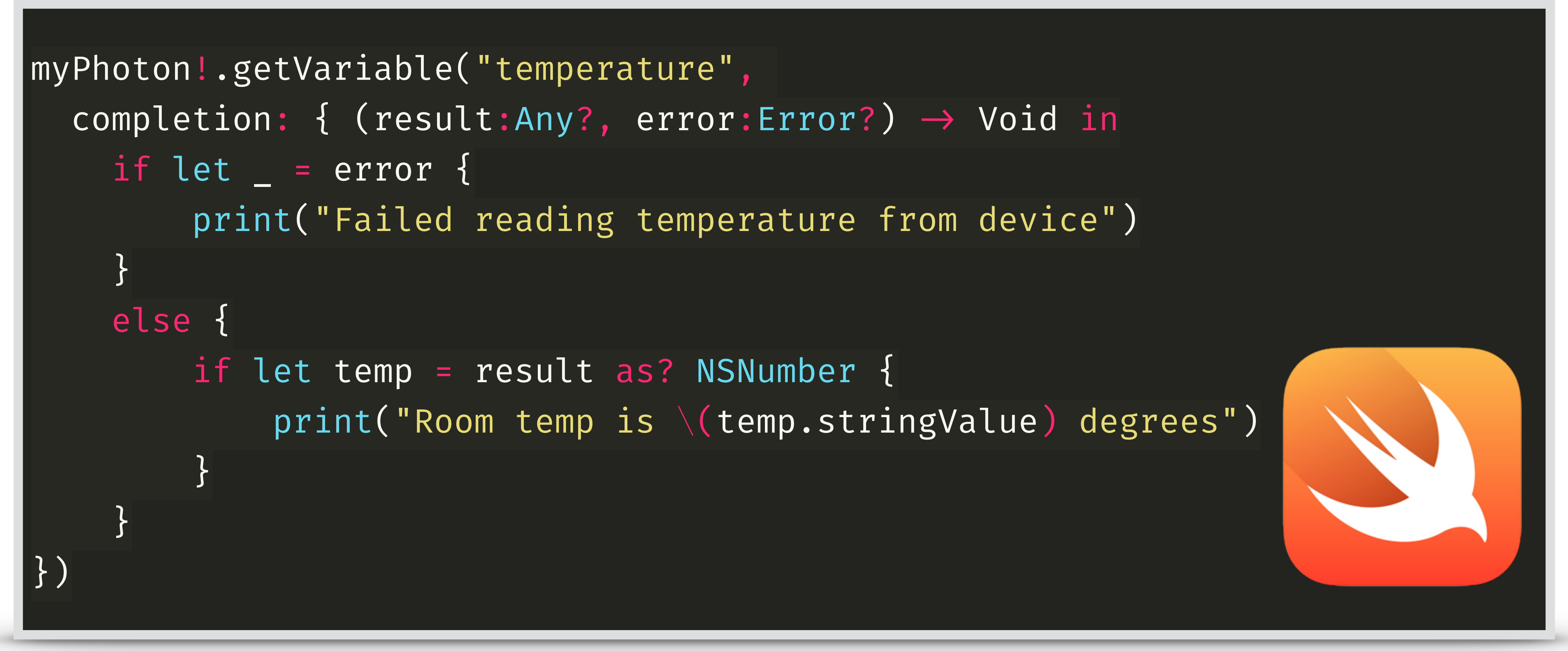


PARTICLE CLI

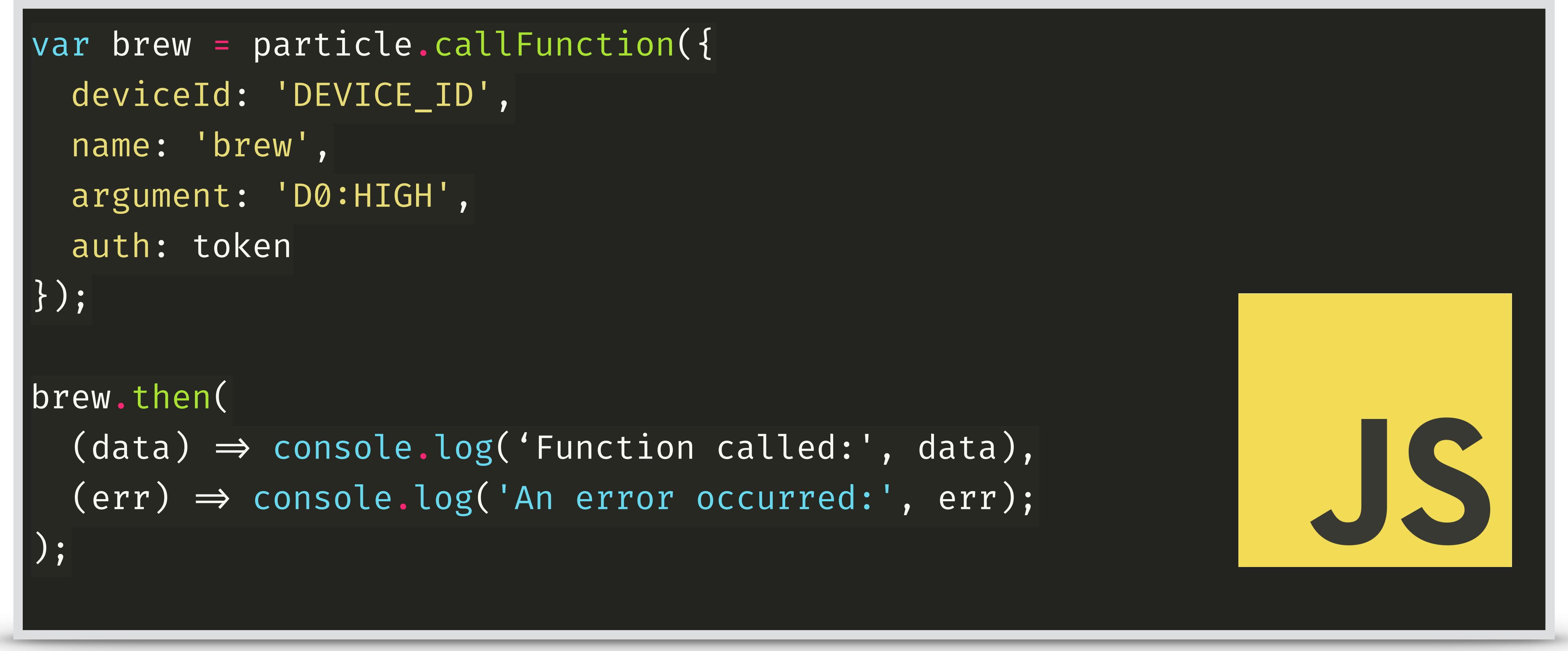


```
➜ ~ particle setup
 _ __                  _   _      _
| '_ \  __ _ _ __| |_(_) ___| | ___
| |_) |/ _` | '__| __| |/ __| |/ _ \
|  __/| (_| | |  | |_| | (__| |  __/
|_|    \__,_|_|   \__|_|\___|_|\___|
                  https://particle.io

> Setup is easy! Let's get started...
> It appears as though you are already logged in as bsatrom@gmail.com
? Would you like to use this account? Yes

! PROTIP: Hold the MODE/SETUP button on your device until it blinks blue!
! PROTIP: Please make sure you are connected to the internet.

> I have detected a Photon connected via USB.
? Would you like to continue with this one? Yes
! The device supports setup via serial.
setting up device 3b001e000247363339343638
> Obtained magical secure claim code.

Claim code set. Now setting up Wi-Fi
? Should I scan for nearby Wi-Fi networks? (Y/n)
```

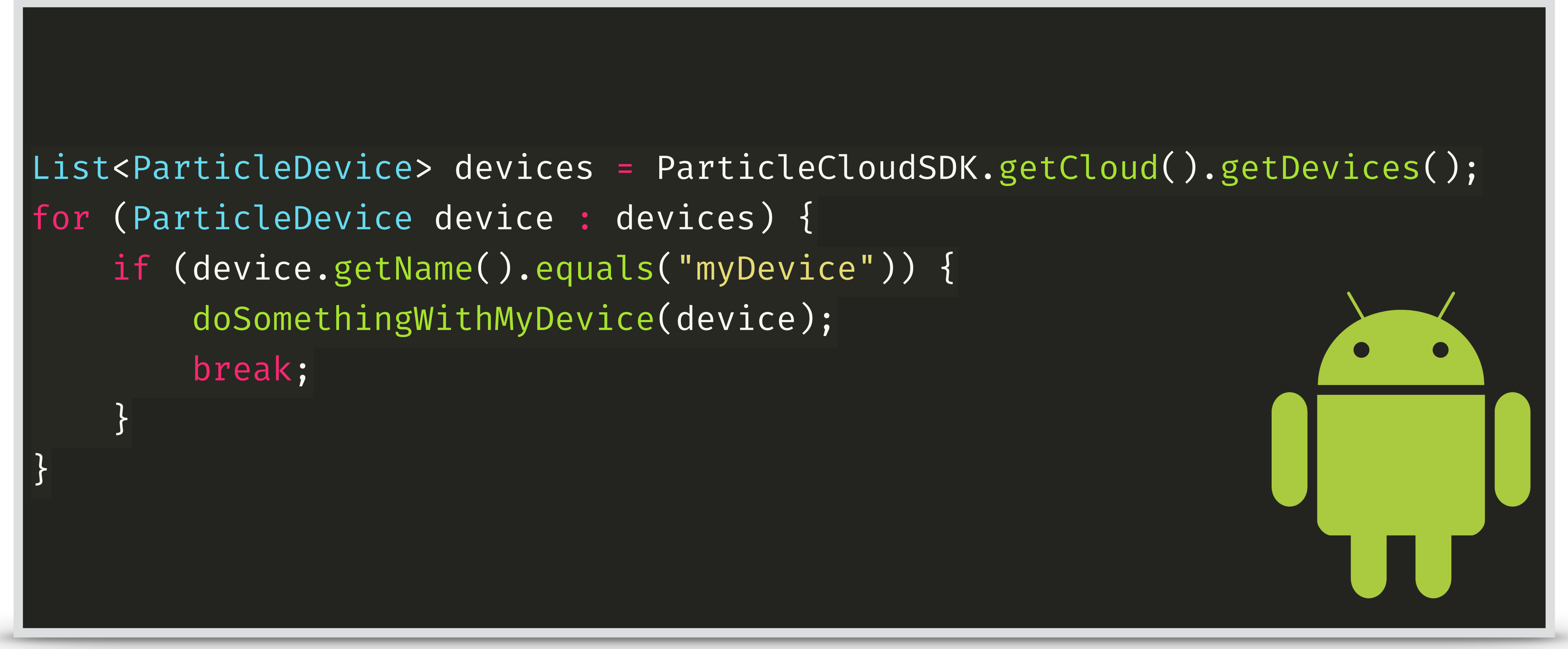
SDKS FOR MOBILE AND WEB DEVELOPMENT



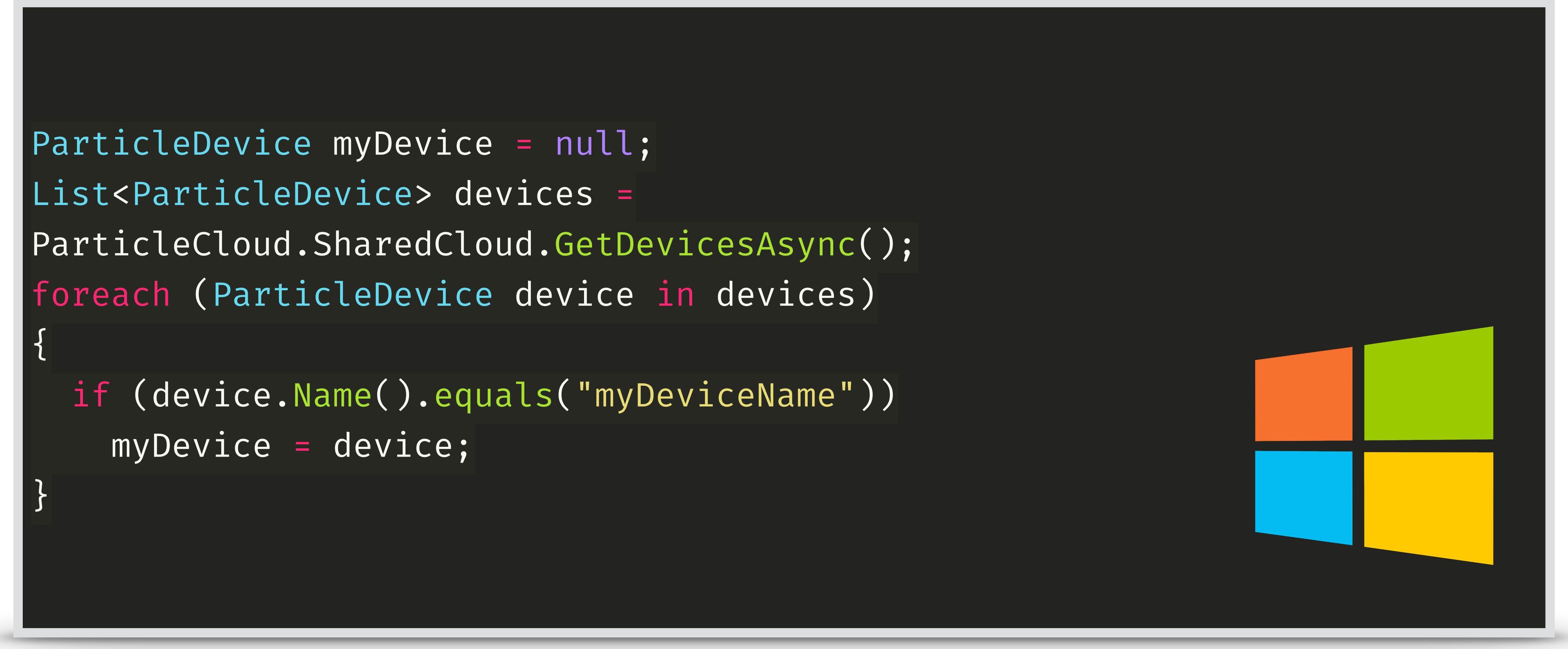
```swift
myPhoton!.getVariable("temperature",
  completion: { (result:Any?, error:Error?) → Void in
    if let _ = error {
        print("Failed reading temperature from device")
    }
    else {
        if let temp = result as? NSNumber {
            print("Room temp is \(temp.stringValue) degrees")
        }
    }
})
```



```javascript
var brew = particle.callFunction({
  deviceId: 'DEVICE_ID',
  name: 'brew',
  argument: 'D0:HIGH',
  auth: token
});

brew.then(
  (data) ⇒ console.log(‘Function called:', data),
  (err) ⇒ console.log('An error occurred:', err);
);
```

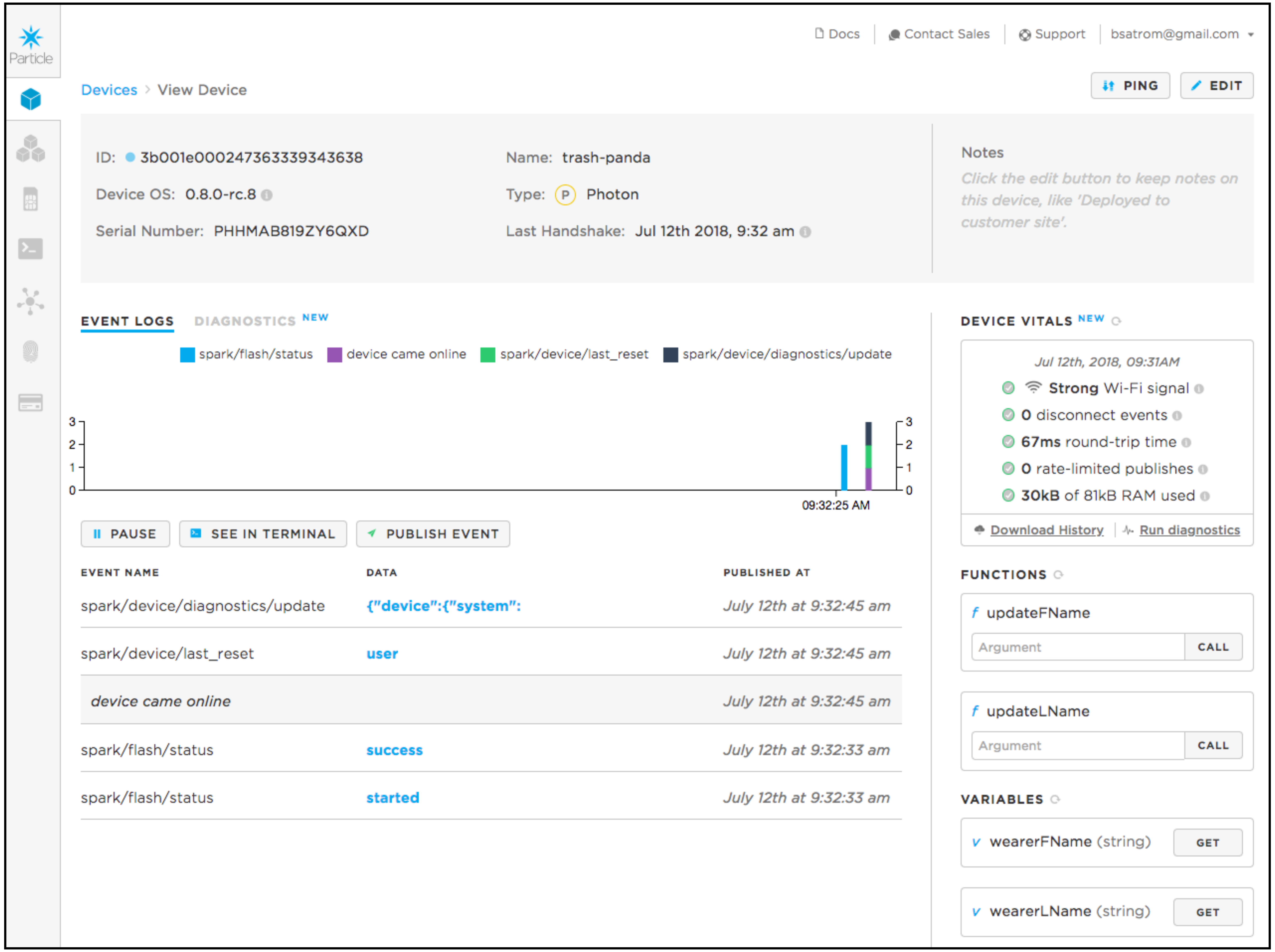


```java
List<ParticleDevice> devices = ParticleCloudSDK.getCloud().getDevices();
for (ParticleDevice device : devices) {
    if (device.getName().equals("myDevice")) {
        doSomethingWithMyDevice(device);
        break;
    }
}
```



```csharp
ParticleDevice myDevice = null;
List<ParticleDevice> devices =
ParticleCloud.SharedCloud.GetDevicesAsync();
foreach (ParticleDevice device in devices)
{
  if (device.Name().equals("myDeviceName"))
    myDevice = device;
}
```

PARTICLE DEVICE CLOUD & CONSOLE



Devices > View Device

ID: 3b001e000247363339343638

Name: trash-panda

Device OS: 0.8.0-rc.8

Type: P Photon

Serial Number: PHHMAB819ZY6QXD

Last Handshake: Jul 12th 2018, 9:32 am

Notes

Click the edit button to keep notes on this device, like 'Deployed to customer site'.

🟦 spark/flash/status 🟪 device came online 🟩 spark/device/last_reset ⬛ spark/device/diagnostics/update

EVENT NAME	DATA	PUBLISHED AT
spark/device/diagnostics/update	{"device":{"system":	*July 12th at 9:32:45 am*
spark/device/last_reset	user	*July 12th at 9:32:45 am*
device came online		*July 12th at 9:32:45 am*
spark/flash/status	success	*July 12th at 9:32:33 am*
spark/flash/status	started	*July 12th at 9:32:33 am*

DEVICE VITALS NEW

Jul 12th, 2018, 09:31AM

- ✓ **Strong** Wi-Fi signal
- ✓ **0** disconnect events
- ✓ **67ms** round-trip time
- ✓ **0** rate-limited publishes
- ✓ **30kB** of 81kB RAM used

Download History | Run diagnostics

FUNCTIONS

f updateFName

Argument CALL

f updateLName

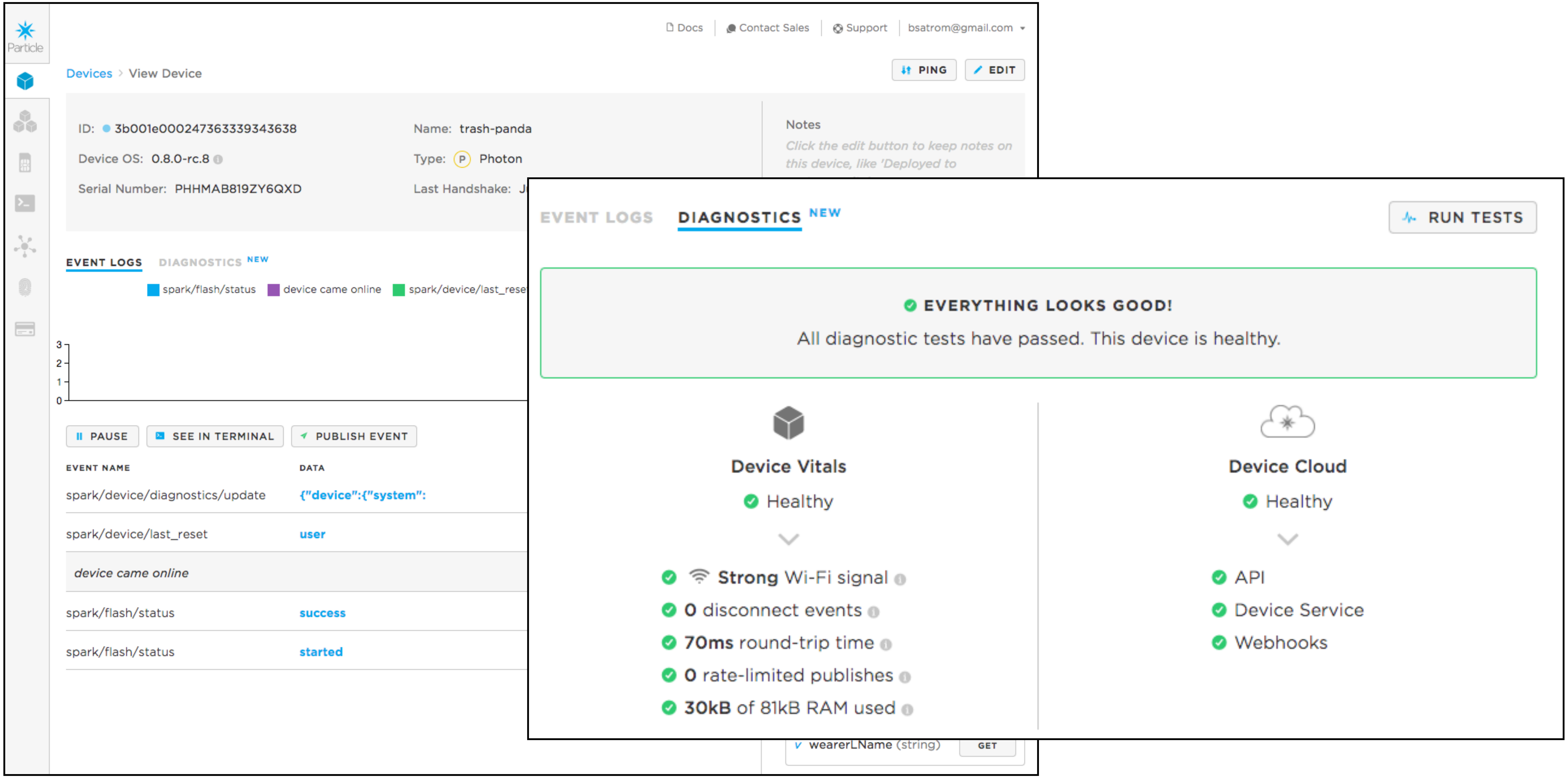
Argument CALL

VARIABLES

v wearerFName (string) GET

v wearerLName (string) GET

PARTICLE DEVICE CLOUD & CONSOLE



Devices › View Device

ID: 3b001e000247363339343638

Name: trash-panda

Device OS: 0.8.0-rc.8

Type: Ⓟ Photon

Serial Number: PHHMAB819ZY6QXD

Last Handshake: J

Notes

Click the edit button to keep notes on this device, like 'Deployed to

EVENT NAME	DATA
spark/device/diagnostics/update	{"device":{"system":
spark/device/last_reset	user
device came online	
spark/flash/status	success
spark/flash/status	started

✓ EVERYTHING LOOKS GOOD!

All diagnostic tests have passed. This device is healthy.

Device Vitals

✓ Healthy

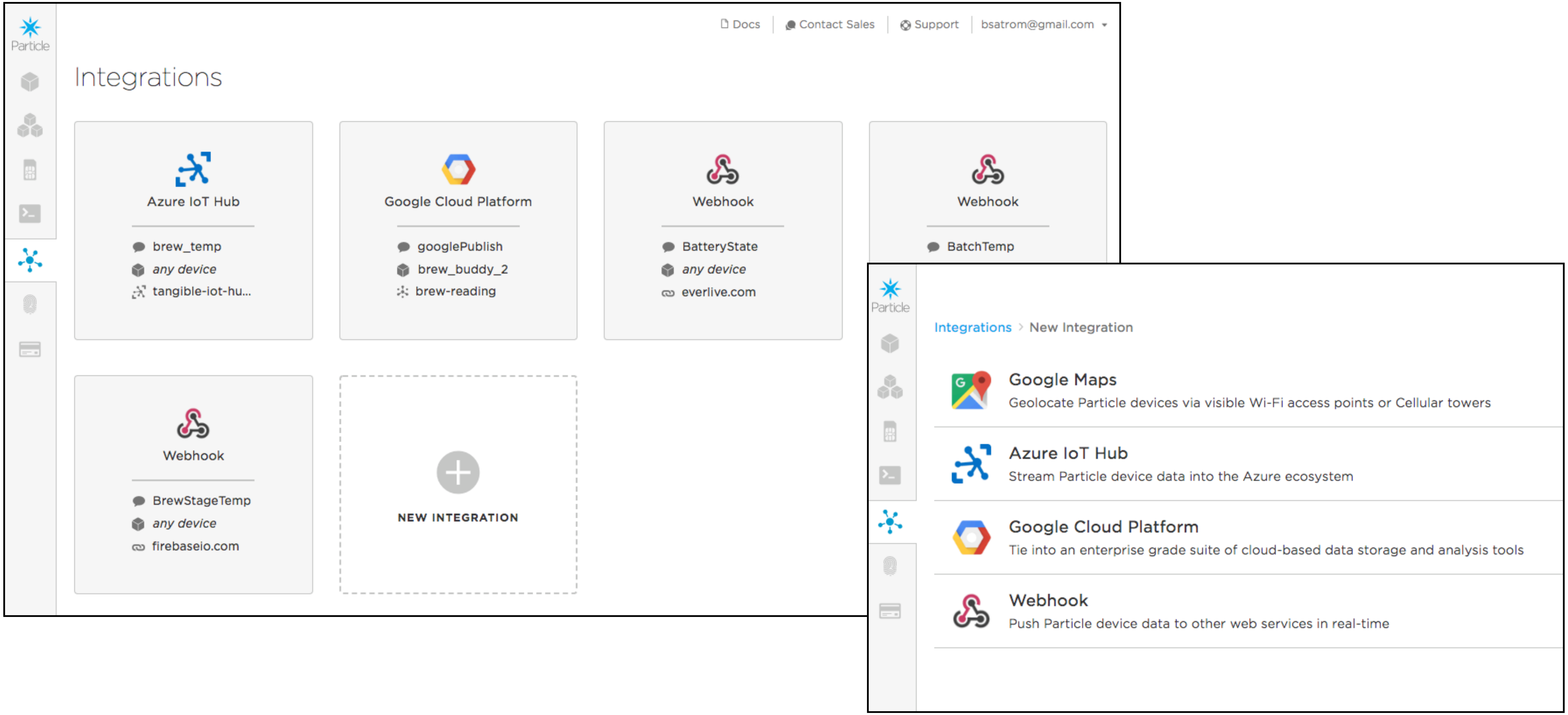
- ✓ **Strong** Wi-Fi signal
- ✓ **0** disconnect events
- ✓ **70ms** round-trip time
- ✓ **0** rate-limited publishes
- ✓ **30kB** of 81kB RAM used

Device Cloud

✓ Healthy

- ✓ API
- ✓ Device Service
- ✓ Webhooks

INTEGRATIONS FOR EXTENDING YOUR IOT SOLUTIONS TO OTHER CLOUDS



Integrations

Azure IoT Hub
- brew_temp
- *any device*
- tangible-iot-hu...

Google Cloud Platform
- googlePublish
- brew_buddy_2
- brew-reading

Webhook
- BatteryState
- *any device*
- everlive.com

Webhook
- BatchTemp

Webhook
- BrewStageTemp
- *any device*
- firebaseio.com

NEW INTEGRATION

Integrations › New Integration

Google Maps
Geolocate Particle devices via visible Wi-Fi access points or Cellular towers

Azure IoT Hub
Stream Particle device data into the Azure ecosystem

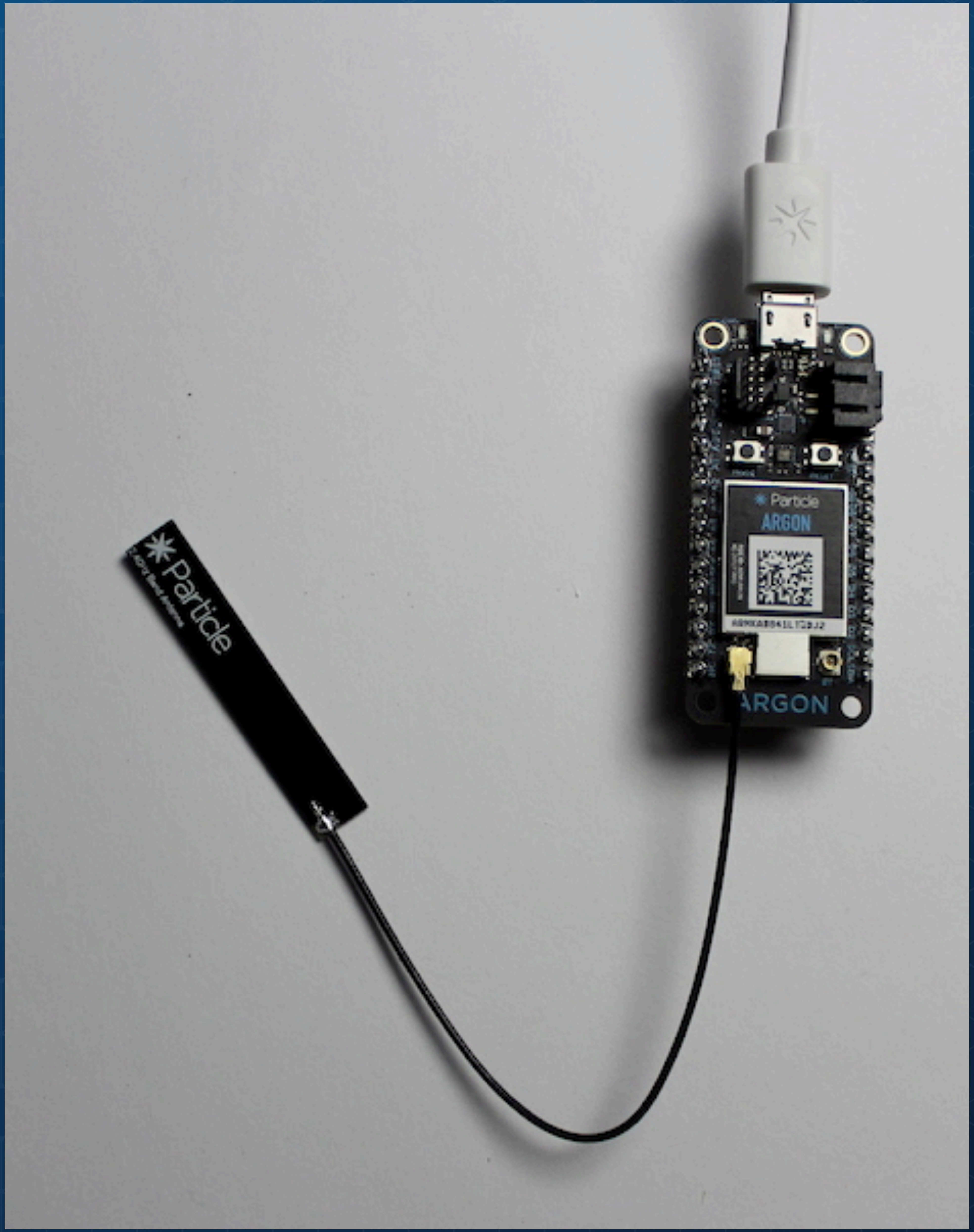
Google Cloud Platform
Tie into an enterprise grade suite of cloud-based data storage and analysis tools

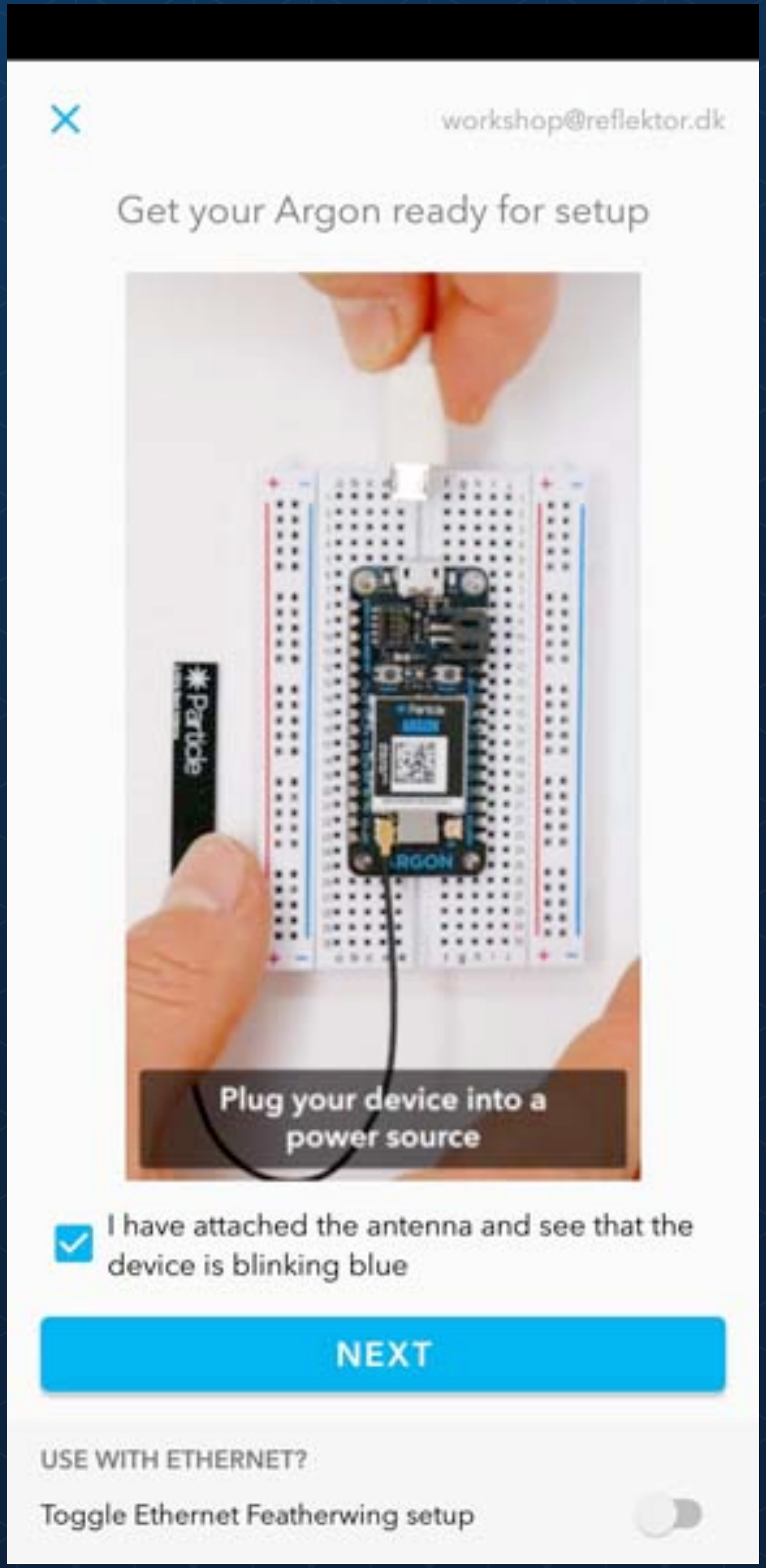
Webhook
Push Particle device data to other web services in real-time

WHY PARTICLE

THE PARTICLE CLOUD & FRIENDS

CLAIMING YOUR FIRST DEVICE





workshop@reflektor.dk

Get your Argon ready for setup

Plug your device into a power source

☑ I have attached the antenna and see that the device is blinking blue

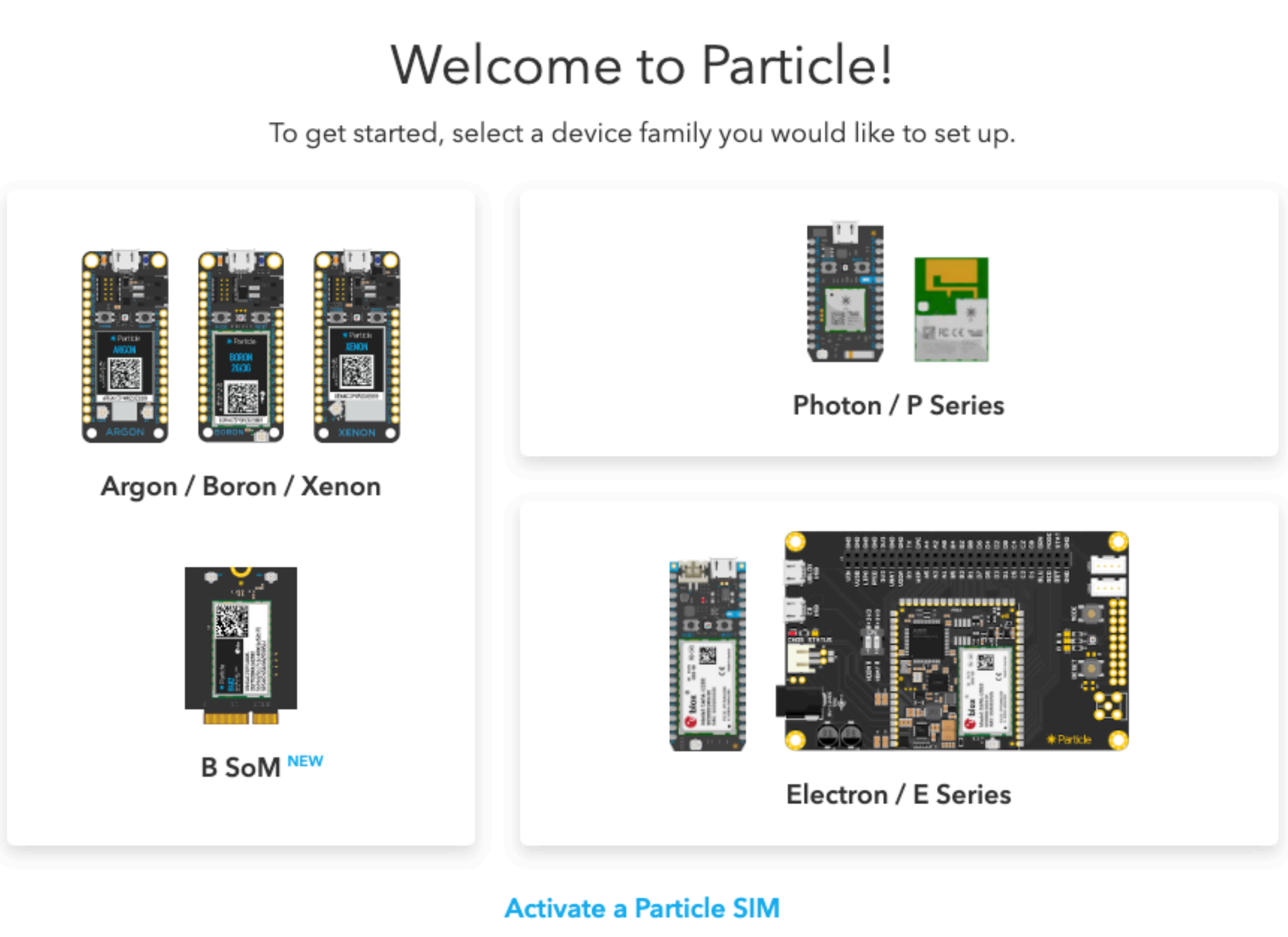
NEXT

USE WITH ETHERNET?

Toggle Ethernet Featherwing setup

WAYS TO CLAIM A NEW PARTICLE DEVICE

WEB



Welcome to Particle!

To get started, select a device family you would like to set up.

Argon / Boron / Xenon

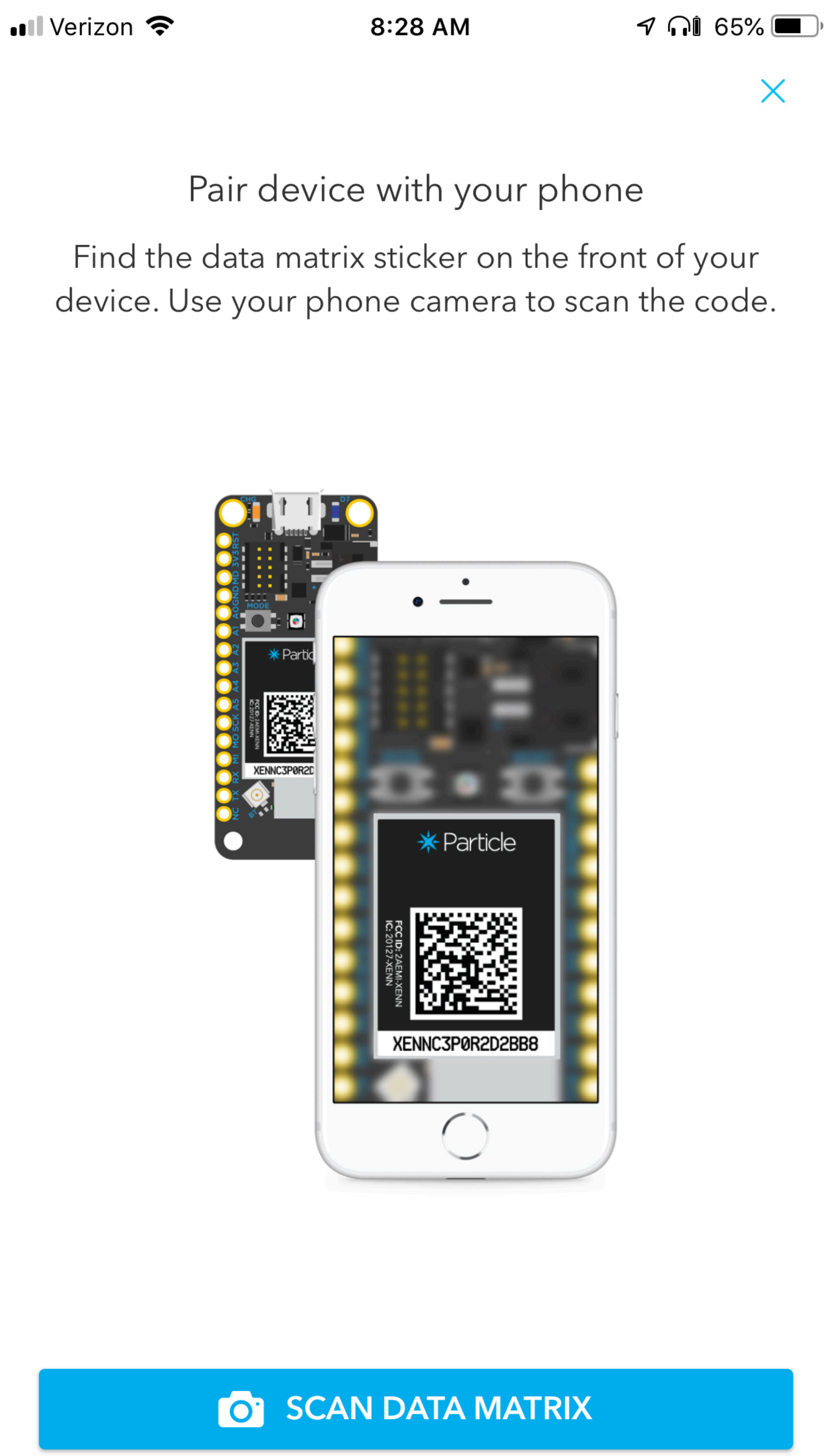
B SoM NEW

Photon / P Series

Electron / E Series

Activate a Particle SIM

MOBILE

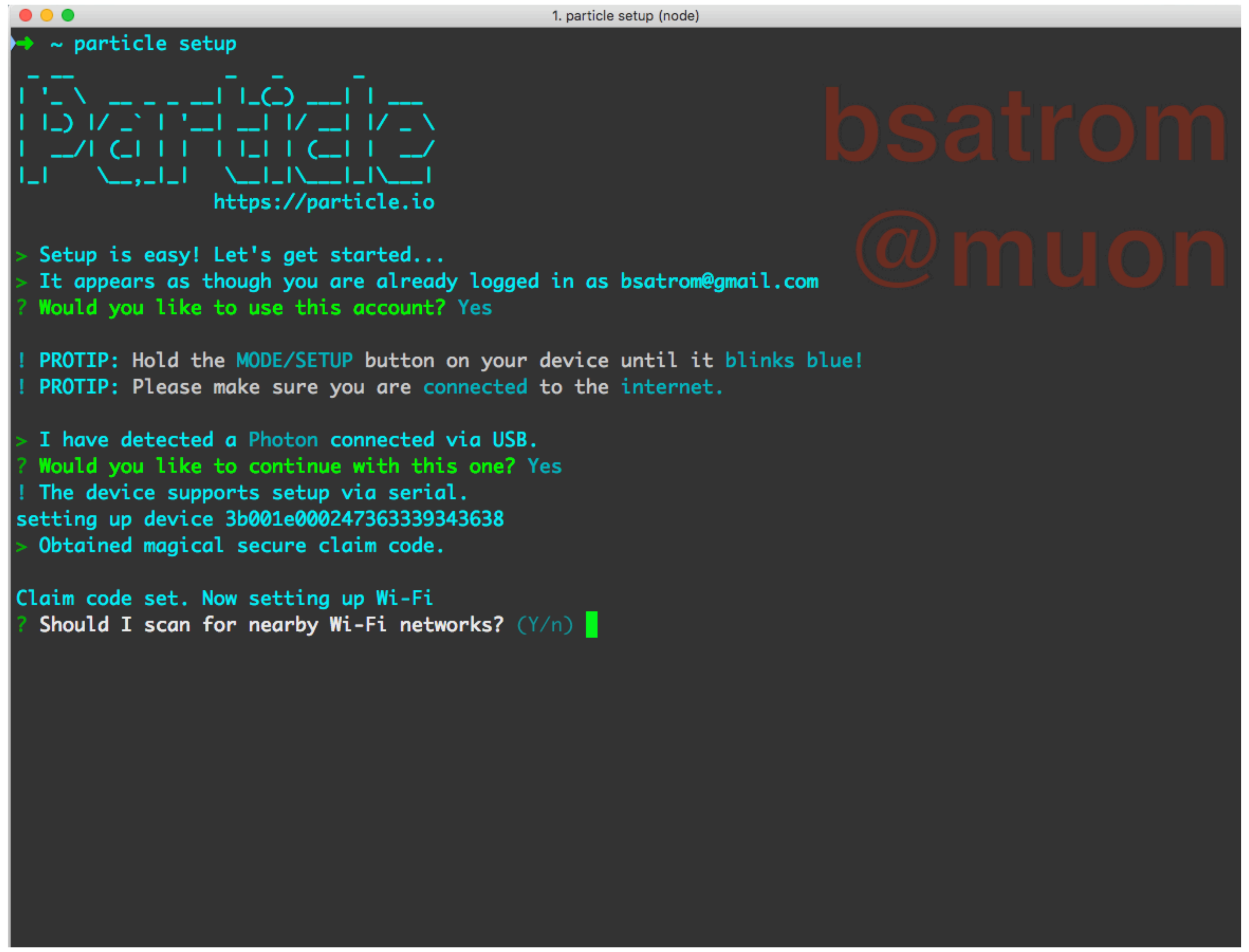


Pair device with your phone

Find the data matrix sticker on the front of your device. Use your phone camera to scan the code.

SCAN DATA MATRIX

PARTICLE CLI



```
➜ ~ particle setup
 _ __             _   _      _
| '_ \  __ _ _ __| |_(_) ___| | ___
| |_) |/ _` | '__| __| |/ __| |/ _ \
|  __/| (_| | |  | |_| | (__| |  __/
|_|    \__,_|_|   \__|_|\___|_|\___|
                  https://particle.io

> Setup is easy! Let's get started...
> It appears as though you are already logged in as bsatrom@gmail.com
? Would you like to use this account? Yes

! PROTIP: Hold the MODE/SETUP button on your device until it blinks blue!
! PROTIP: Please make sure you are connected to the internet.

> I have detected a Photon connected via USB.
? Would you like to continue with this one? Yes
! The device supports setup via serial.
setting up device 3b001e000247363339343638
> Obtained magical secure claim code.

Claim code set. Now setting up Wi-Fi
? Should I scan for nearby Wi-Fi networks? (Y/n)
```

LAB PREREQUISITES

https://part.cl/accelerate

- Create a new Particle account at login.particle.io
- Install Particle Workbench (or the Workbench VS Code Extension)
- Install the Particle iOS or Android App
- Install the Particle CLI