



Particle

PARTICLE 101 - A LAP AROUND THE PARTICLE ECOSYSTEM

WHY PARTICLE

THE PARTICLE CLOUD & FRIENDS

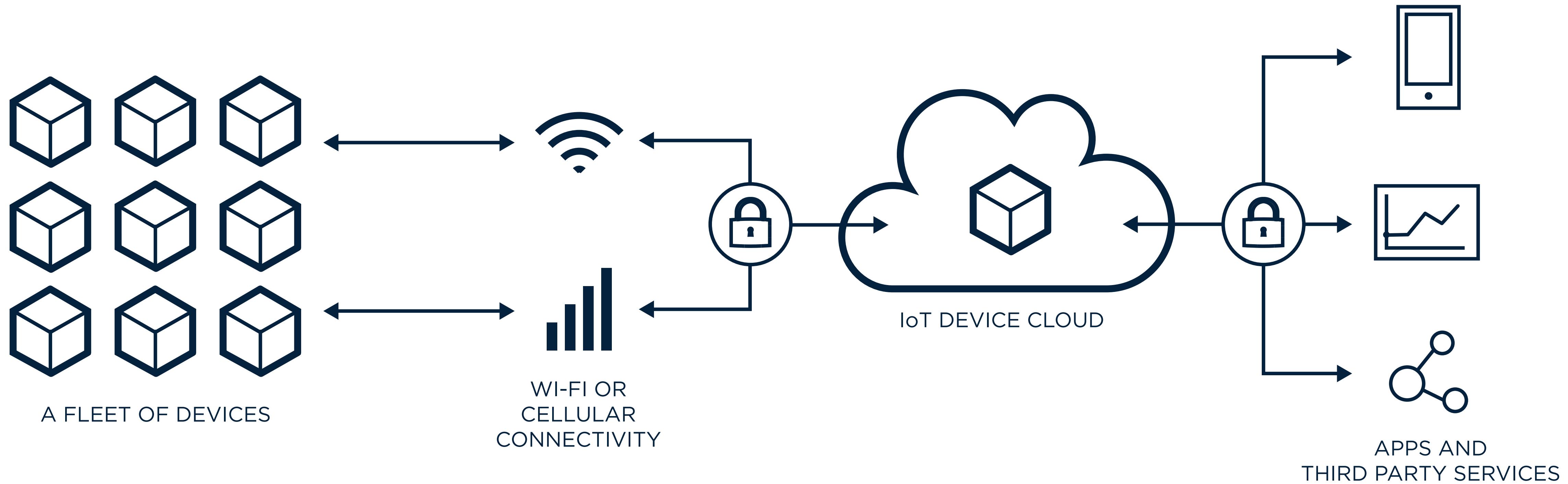
CLAIMING YOUR FIRST DEVICE

WHY PARTICLE

THE PARTICLE CLOUD & FRIENDS

CLAIMING YOUR FIRST DEVICE

PARTICLE IS A FULL-STACK IOT DEVICE PLATFORM



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

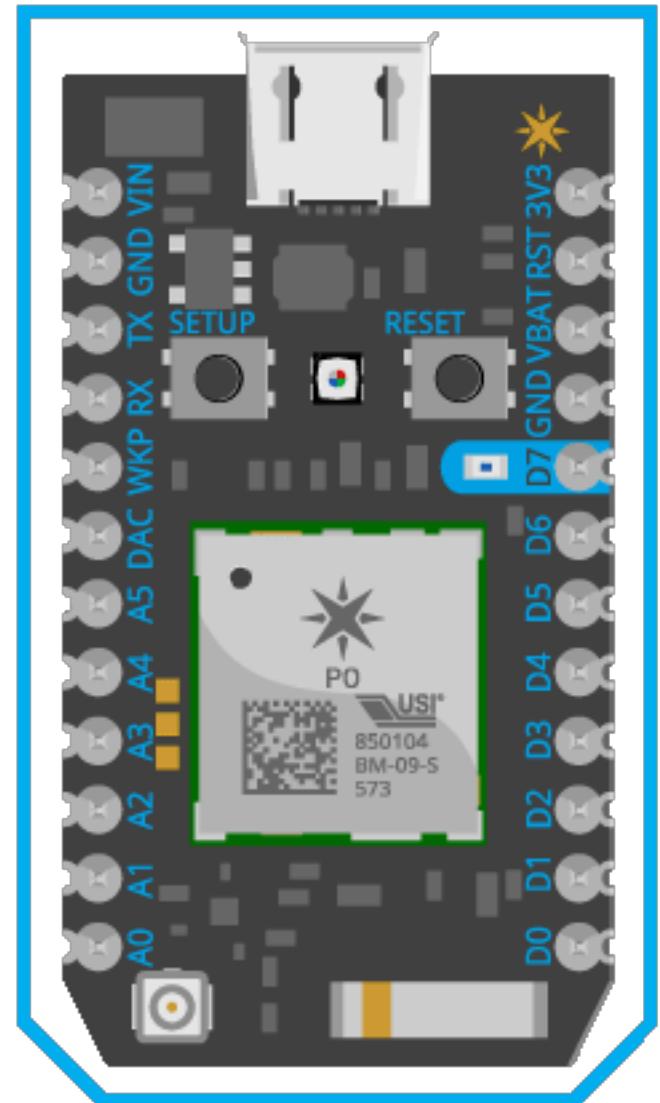


500,000 devices

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



Photon Development Kit

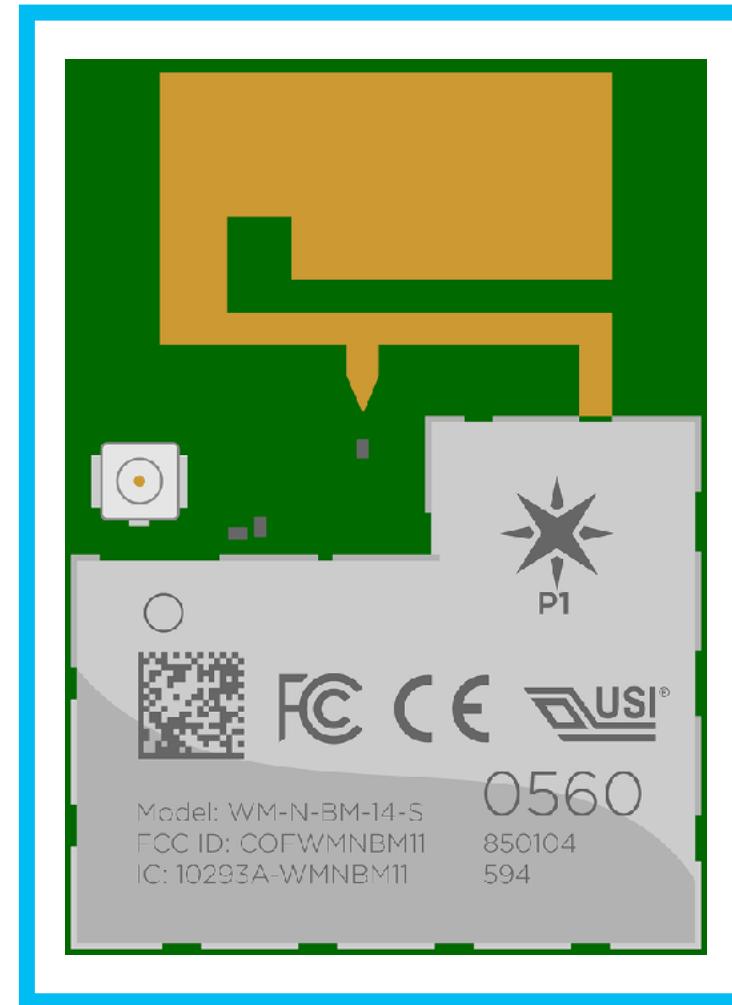
- Breadboard-friendly Dev Kit
- Available with or without headers

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



P0

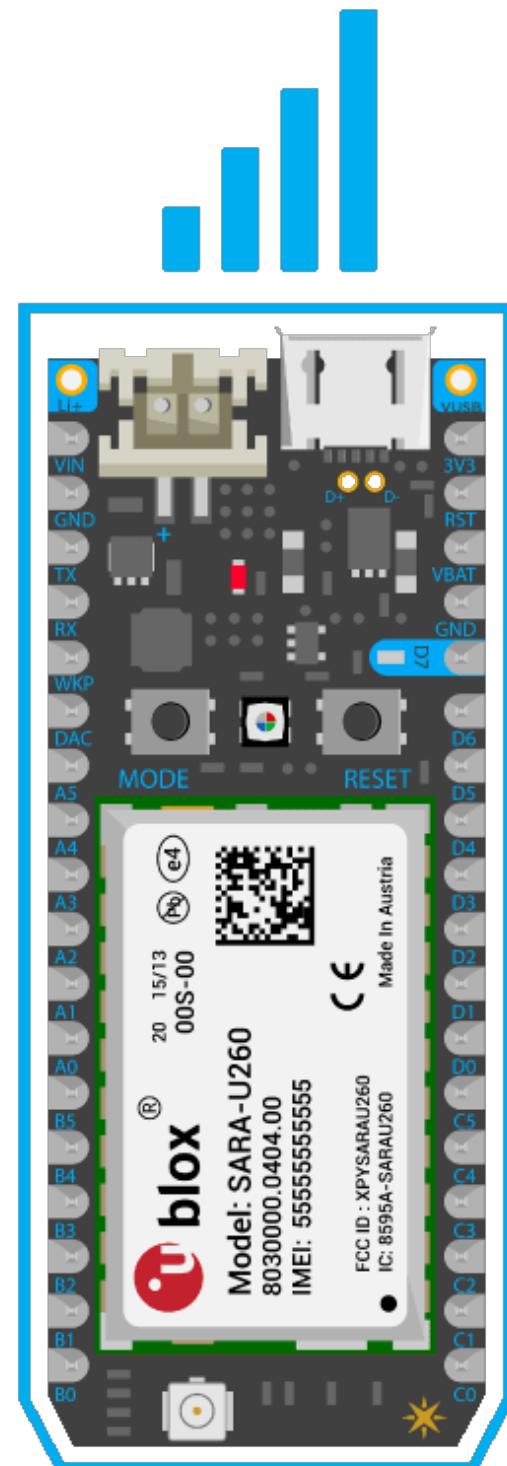
- Small, mass-production form-factor
- Available in mfg-ready reels



P1

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

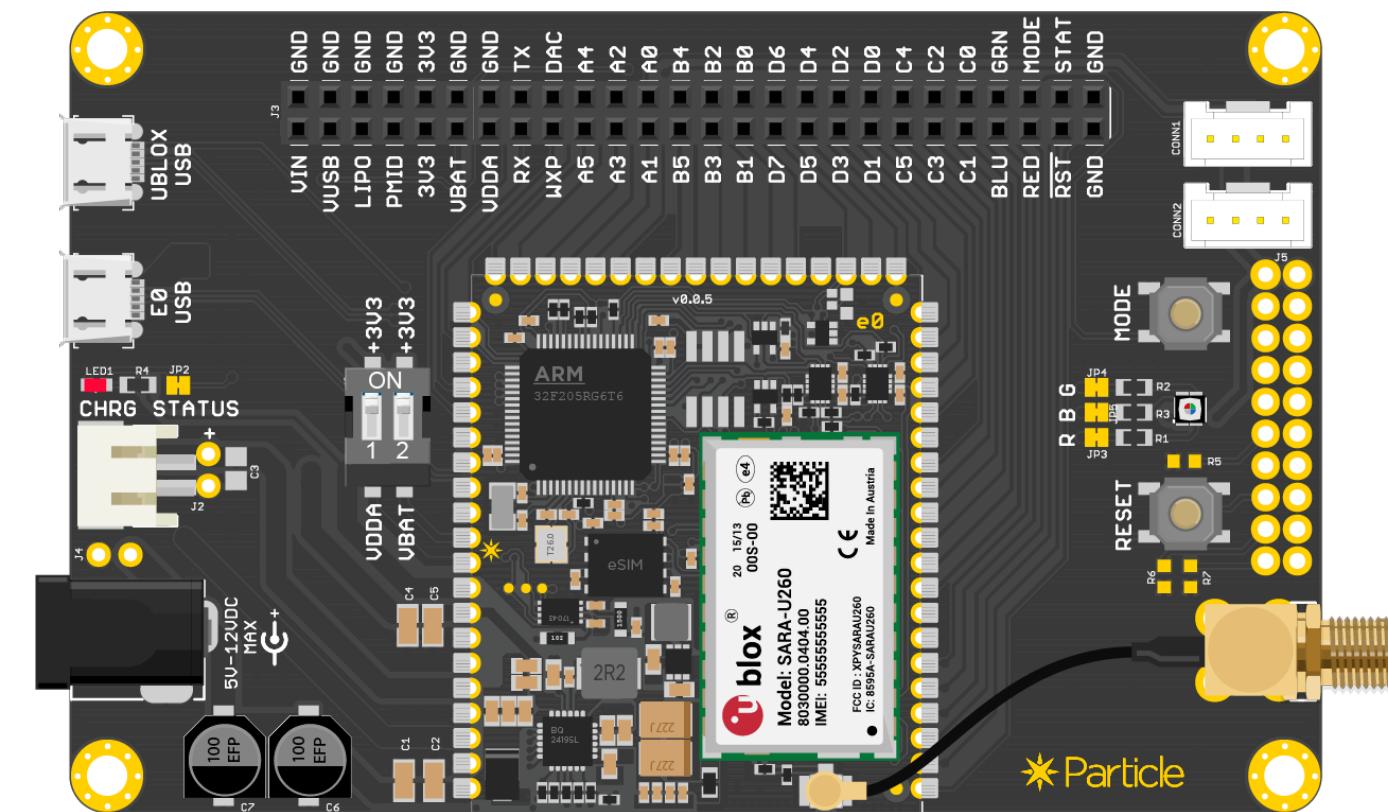
CELLULAR FOR PROTOTYPING AND PRODUCTION



Electron Development Kit

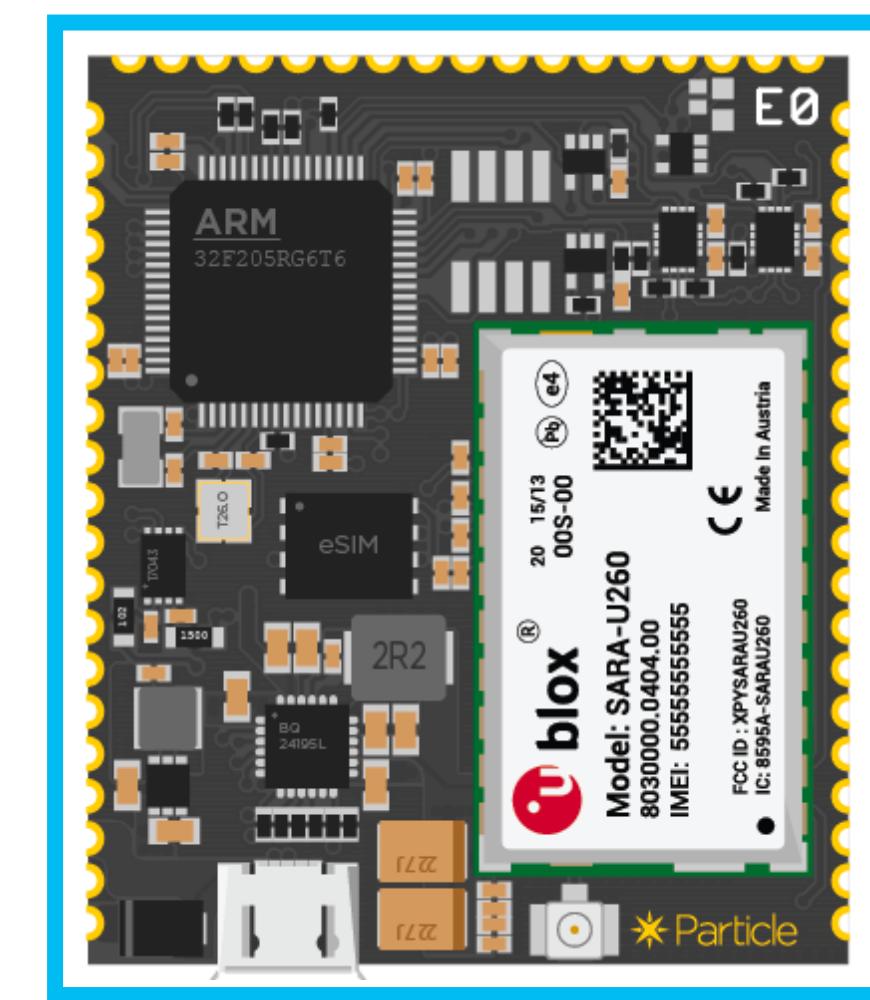
- Breadboard-friendly Dev Kit
- Available only with headers

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



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

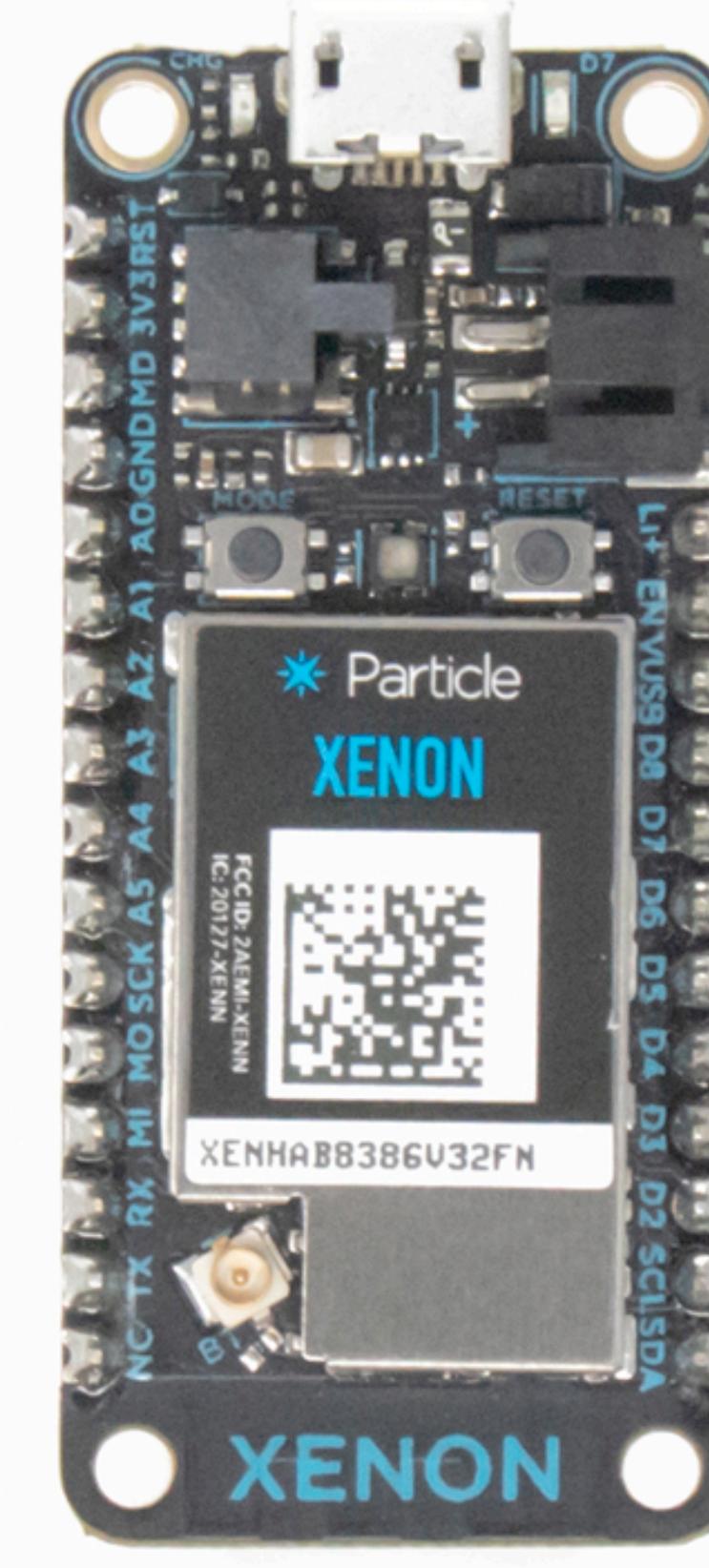
Can function as a gateway,
repeater and/or endpoint



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

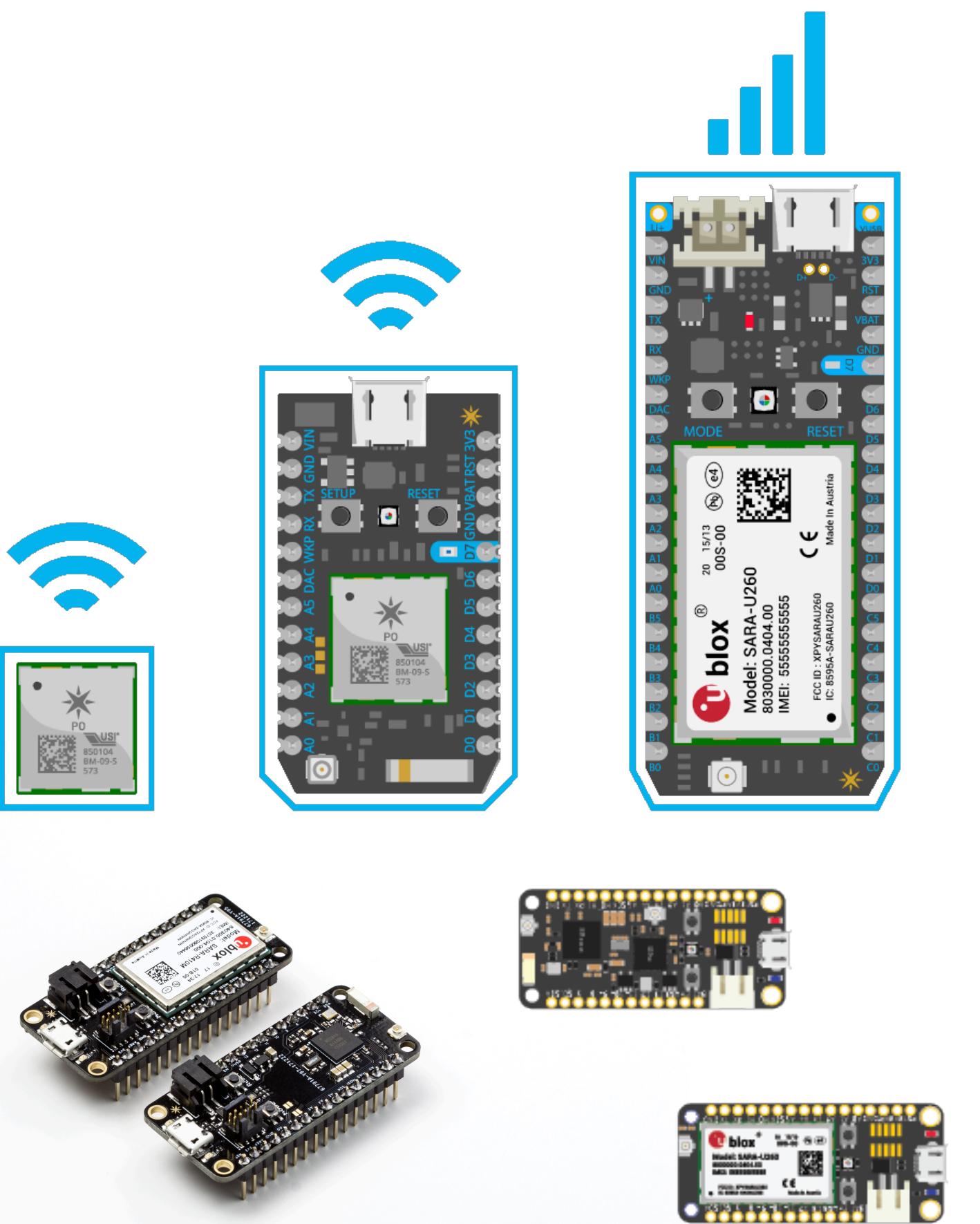
THE PARTICLE CLOUD & FRIENDS

CLAIMING YOUR FIRST DEVICE

THE PARTICLE ECOSYSTEM: HARDWARE, FIRMWARE, & SOFTWARE

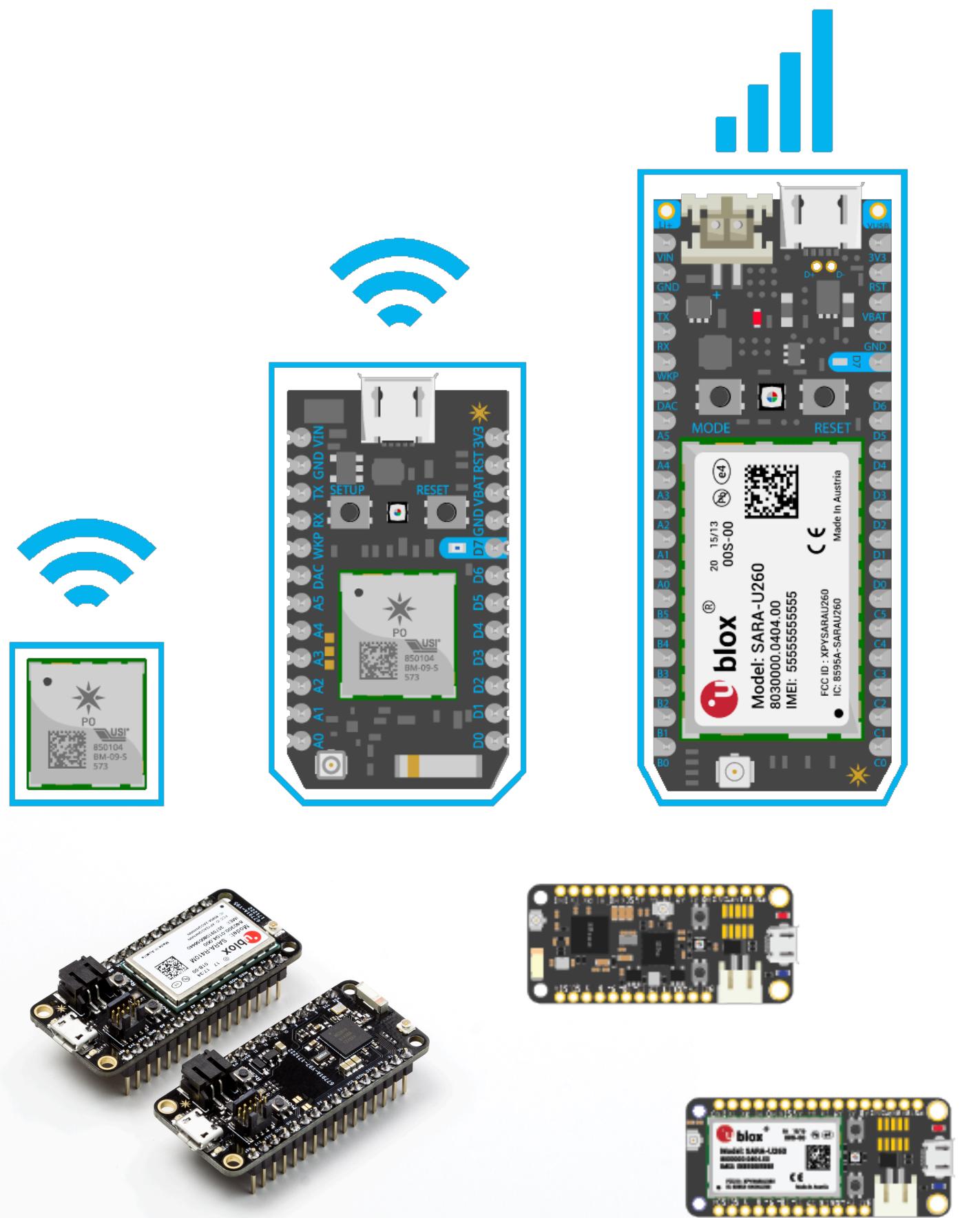
THE PARTICLE ECOSYSTEM: HARDWARE, FIRMWARE, & SOFTWARE

DEVICE HARDWARE FOR PROTOTYPING & PRODUCTION

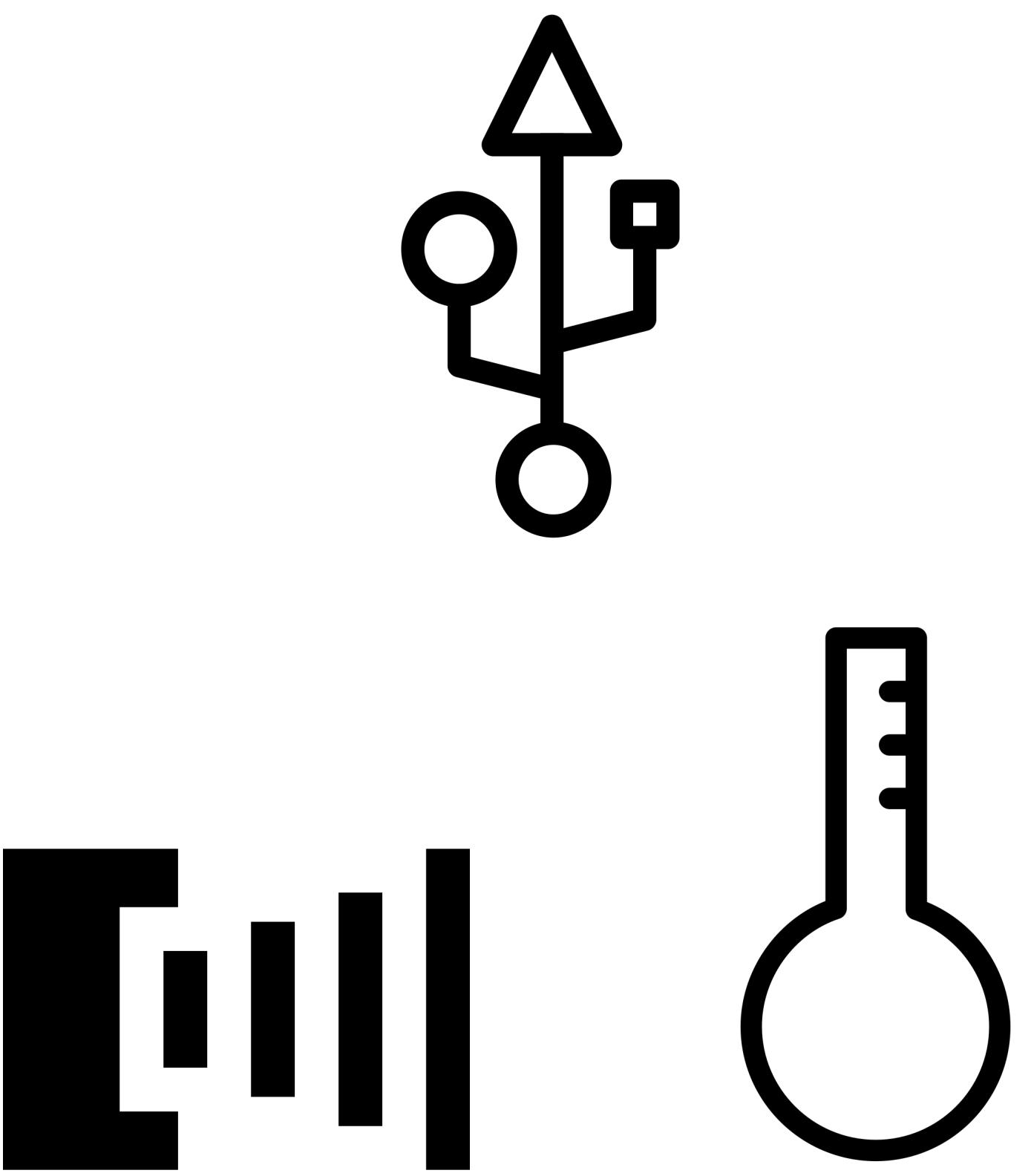


THE PARTICLE ECOSYSTEM: HARDWARE, FIRMWARE, & SOFTWARE

DEVICE HARDWARE FOR PROTOTYPING
& PRODUCTION

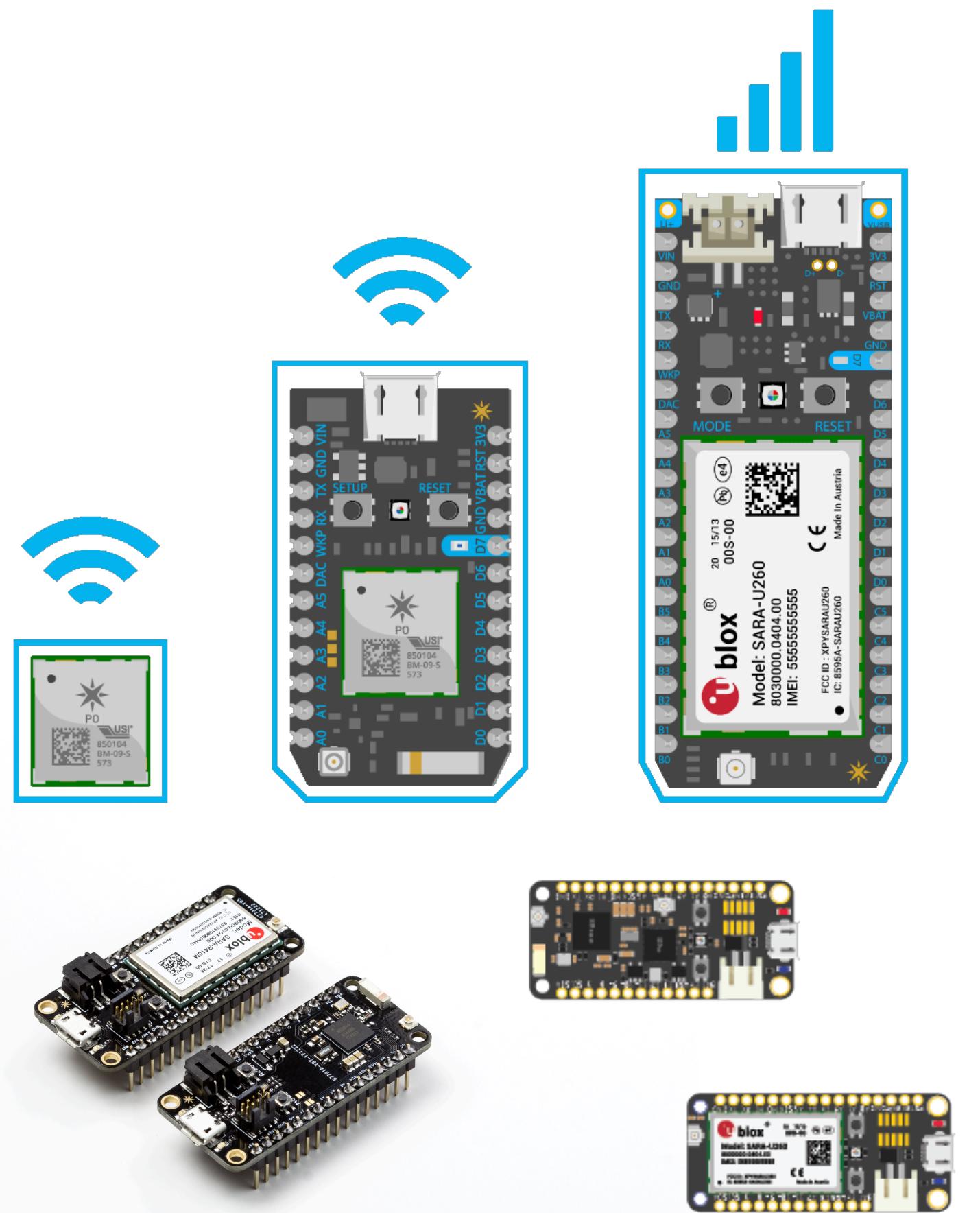


DEVICE OS FIRMWARE & LIBRARIES

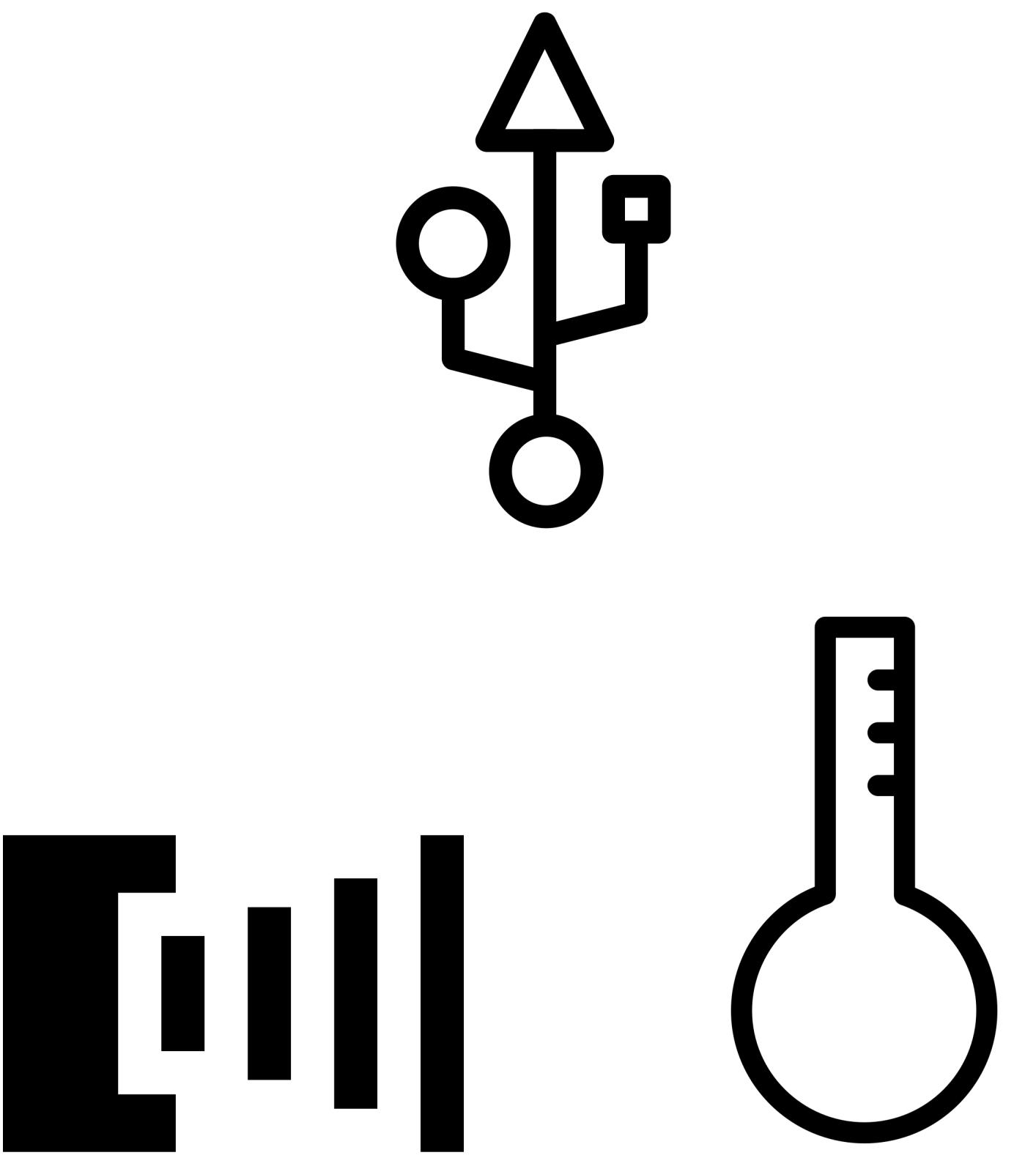


THE PARTICLE ECOSYSTEM: HARDWARE, FIRMWARE, & SOFTWARE

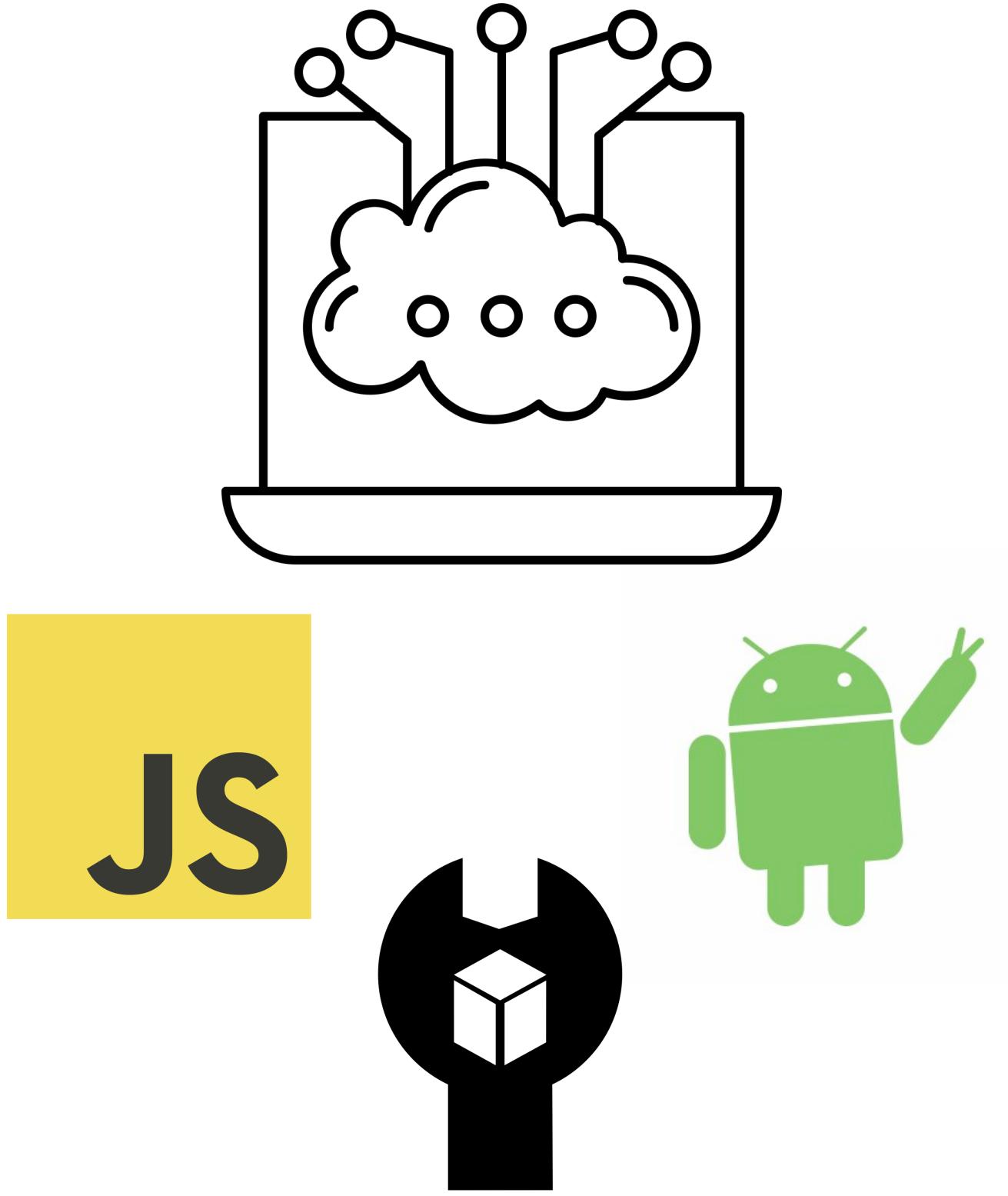
DEVICE HARDWARE FOR PROTOTYPING
& PRODUCTION



DEVICE OS FIRMWARE & LIBRARIES

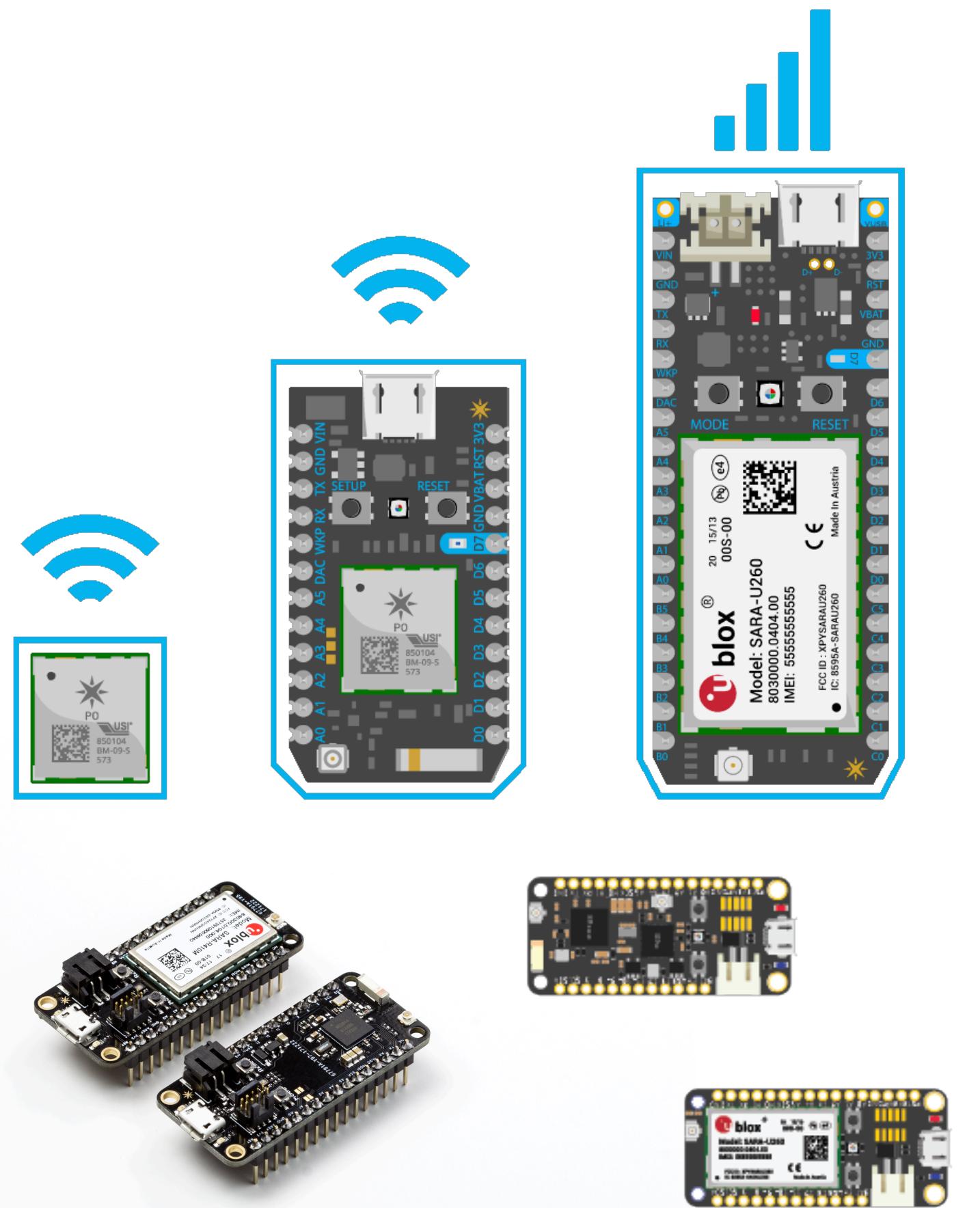


DEVICE CLOUD & SOFTWARE TOOLS

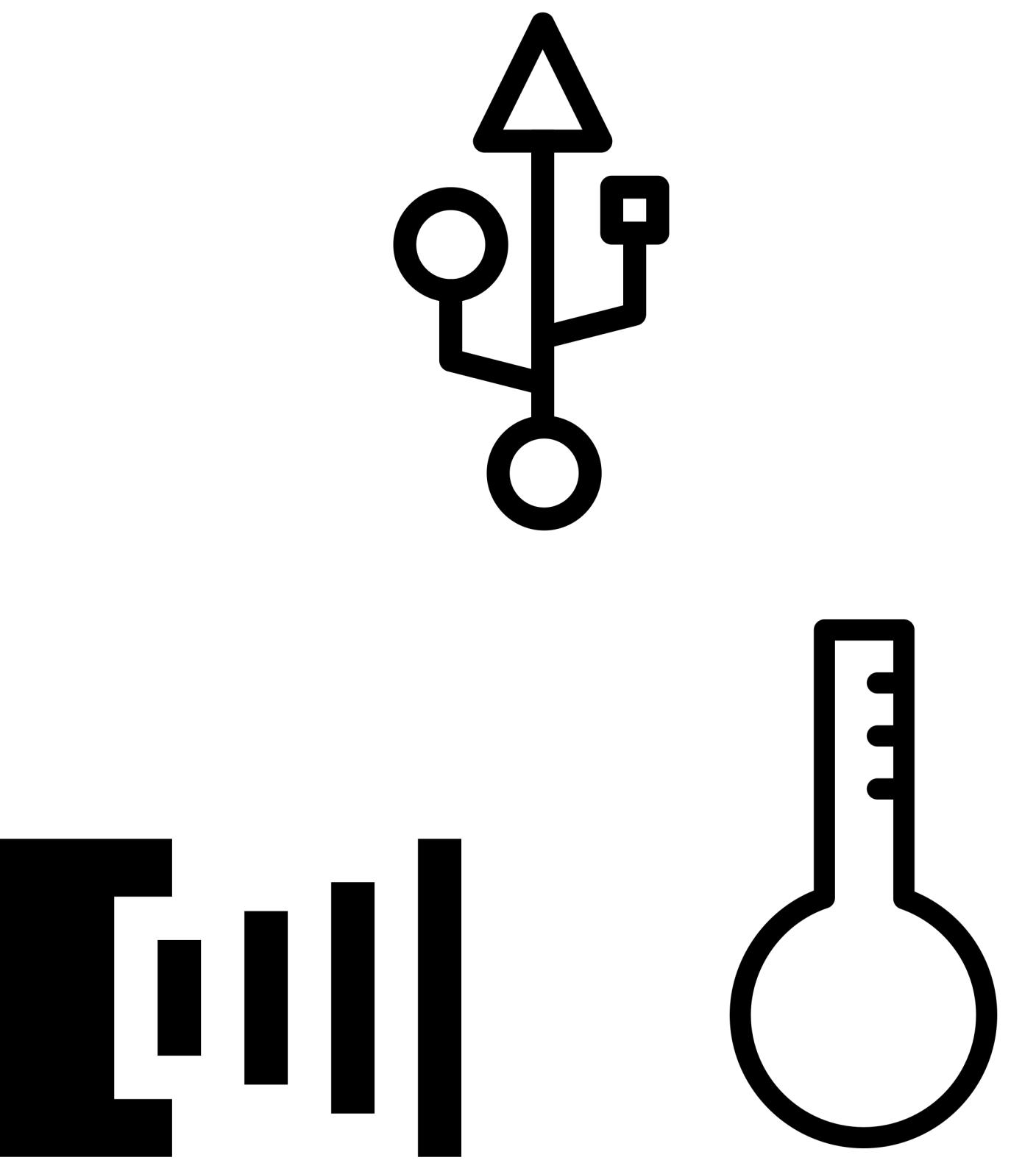


THE PARTICLE ECOSYSTEM: HARDWARE, FIRMWARE, & SOFTWARE

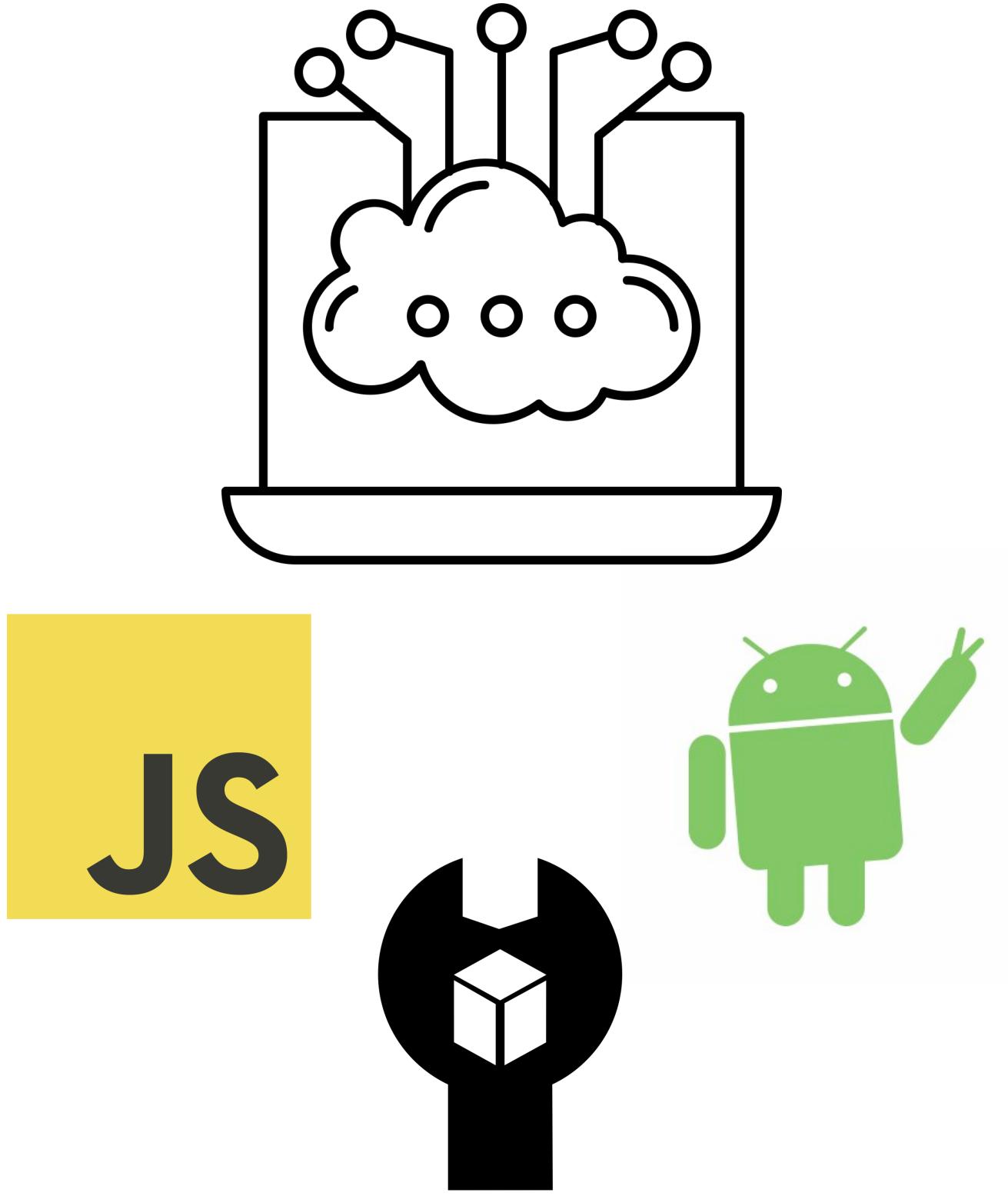
DEVICE HARDWARE FOR PROTOTYPING
& PRODUCTION



DEVICE OS FIRMWARE & LIBRARIES

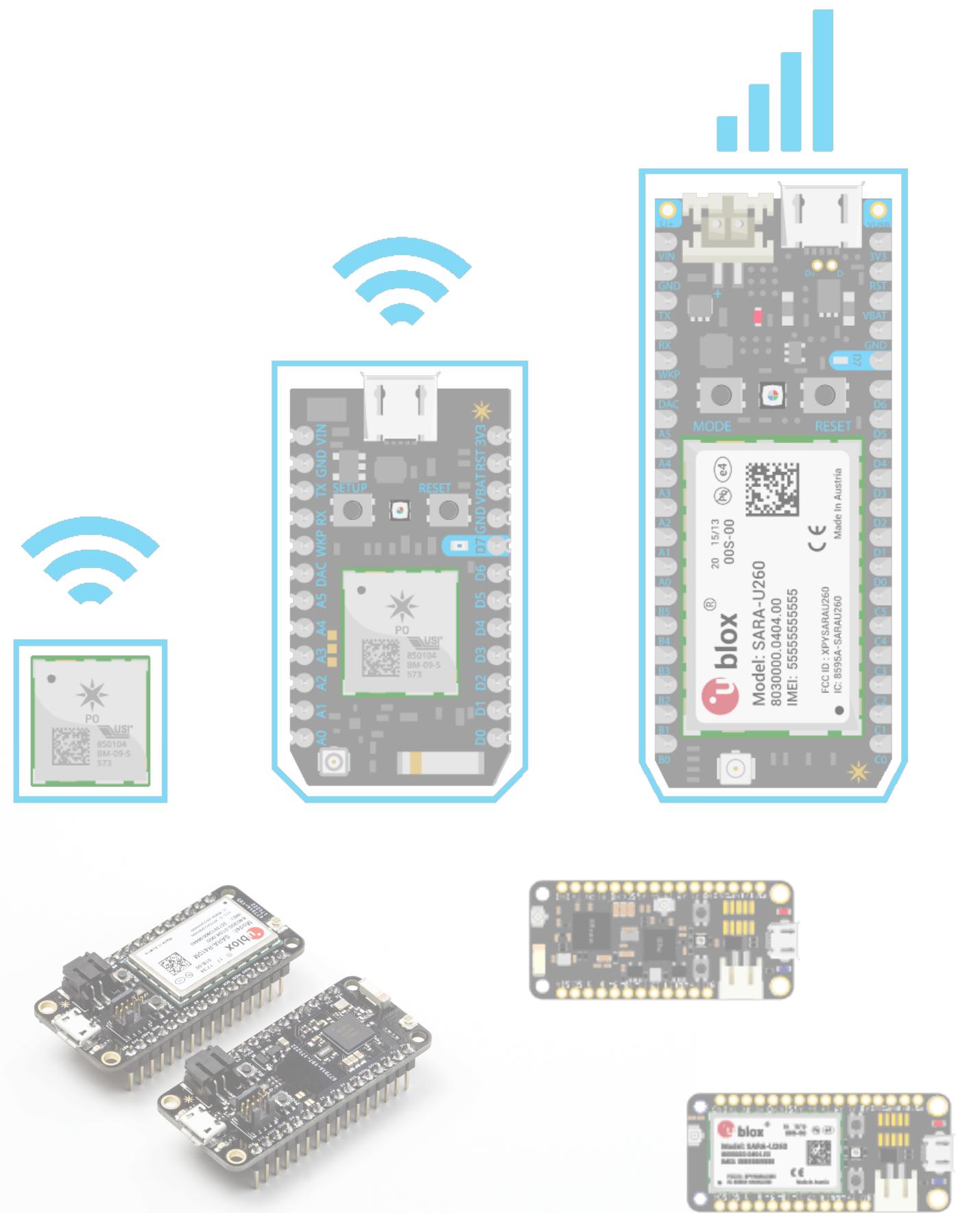


DEVICE CLOUD & SOFTWARE TOOLS

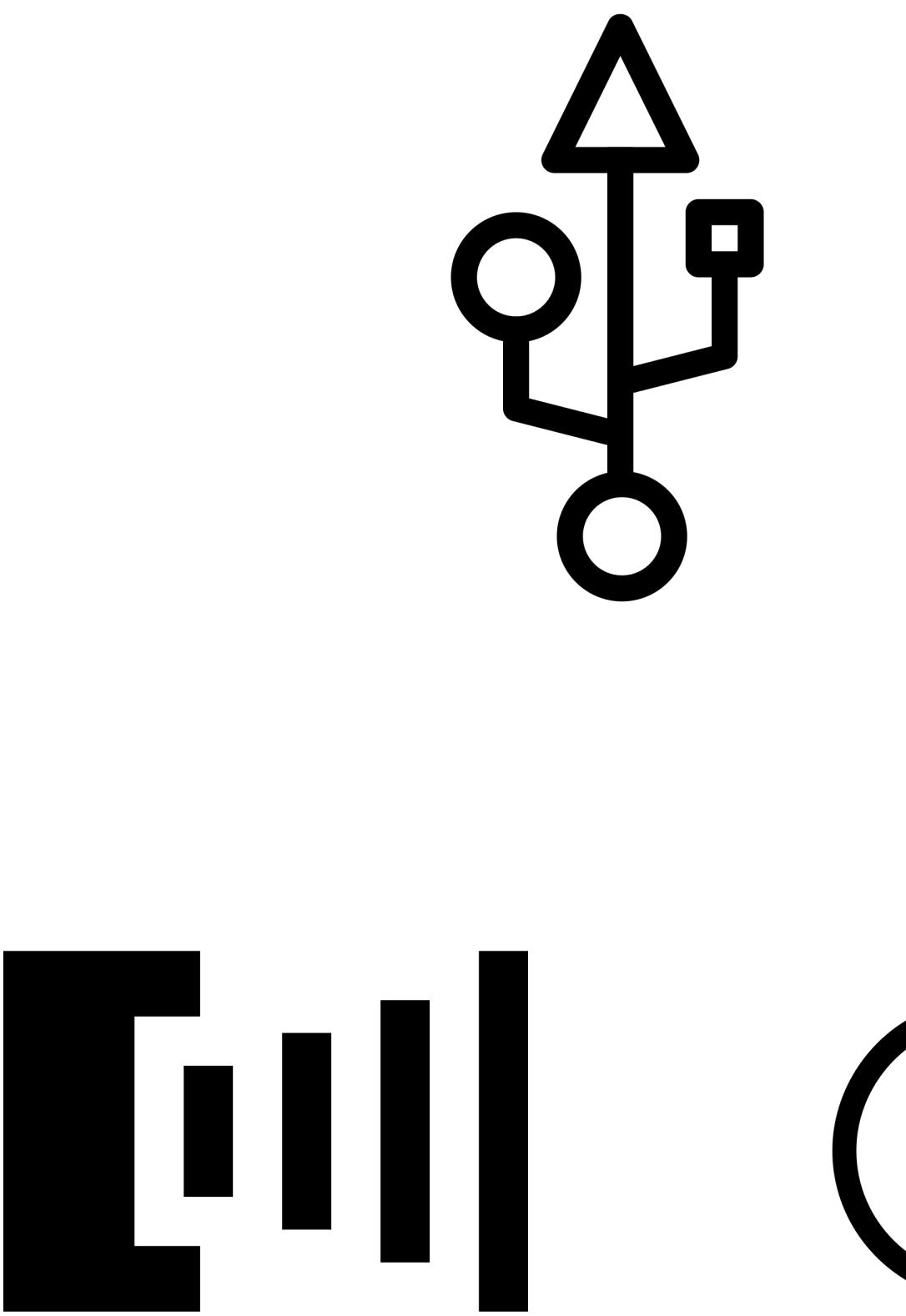


THE PARTICLE ECOSYSTEM: HARDWARE, FIRMWARE, & SOFTWARE

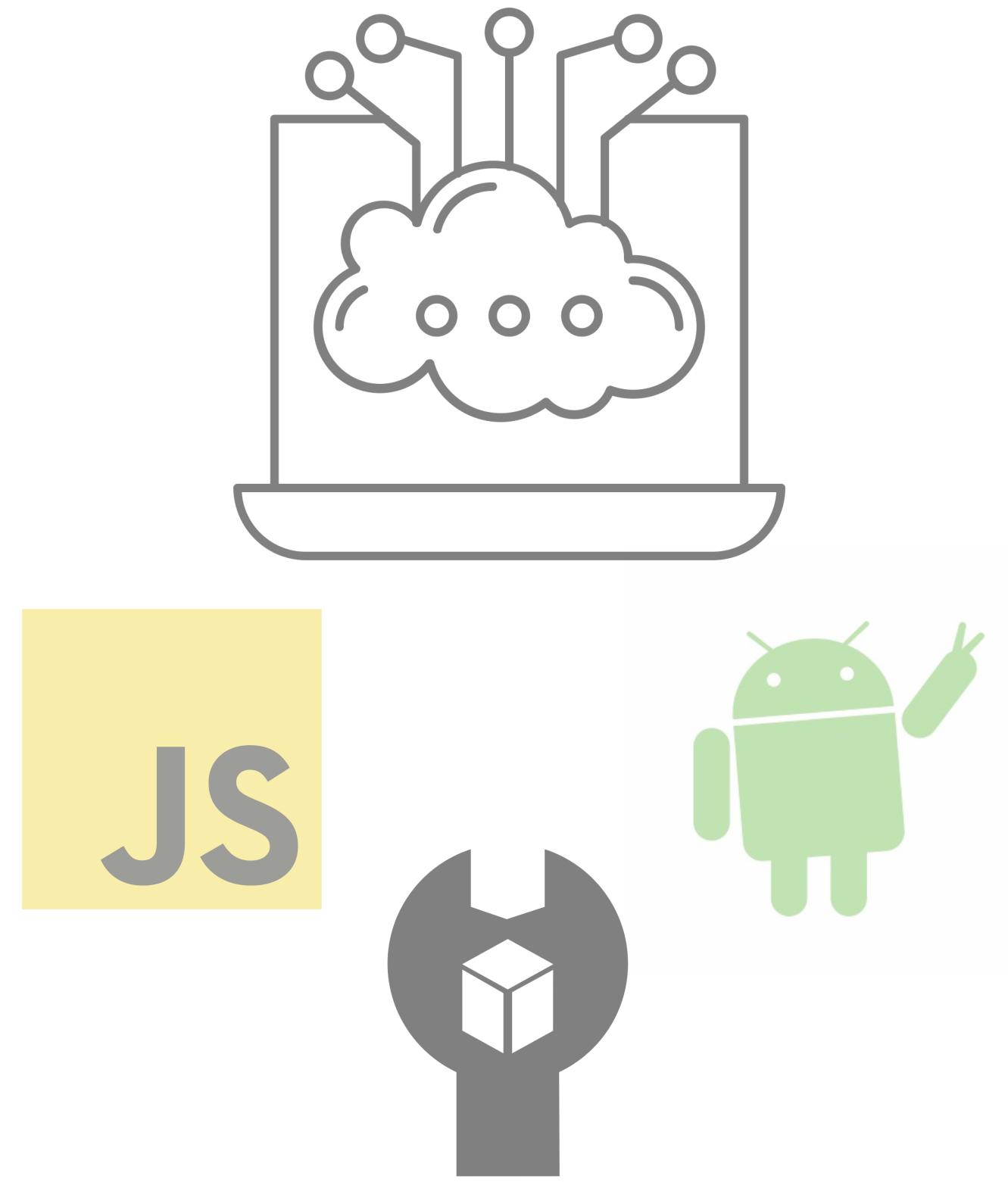
DEVICE HARDWARE FOR PROTOTYPING
& PRODUCTION



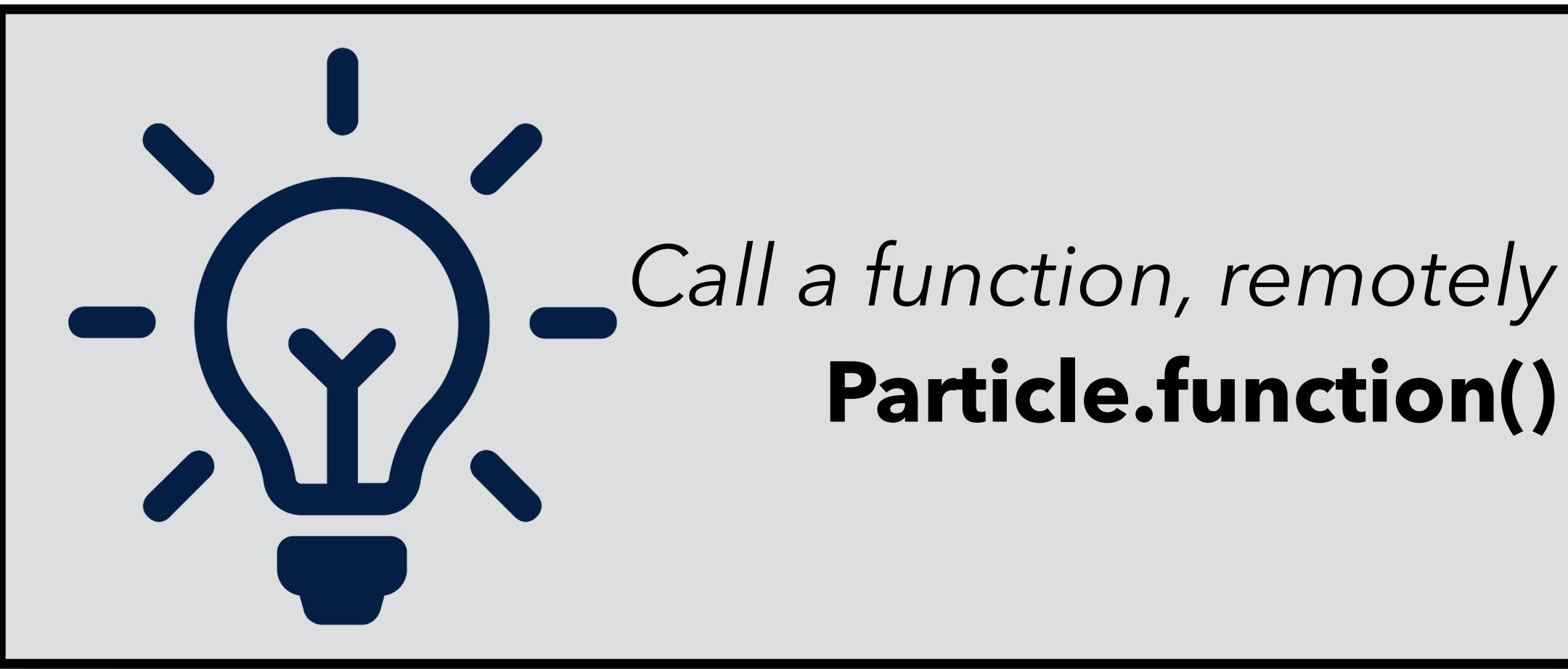
DEVICE OS FIRMWARE & LIBRARIES



DEVICE CLOUD & SOFTWARE TOOLS



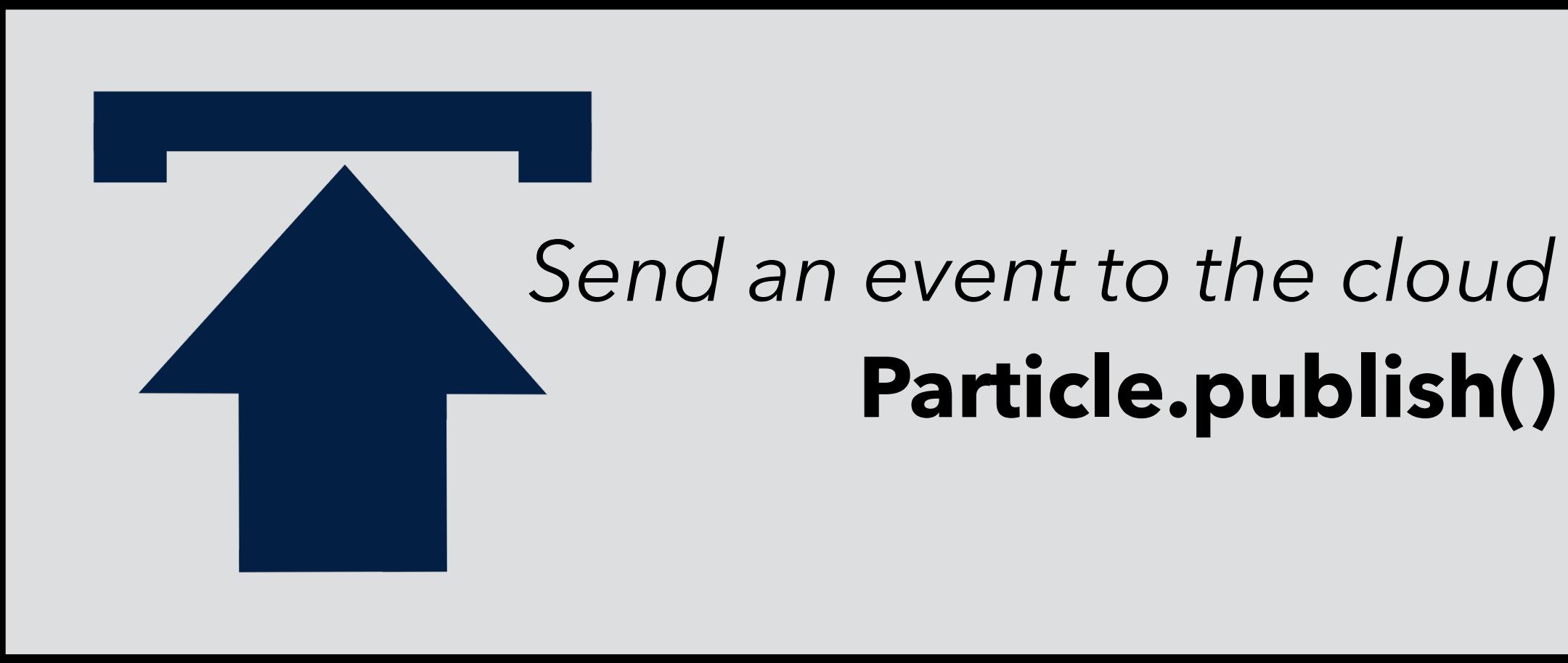
PARTICLE DEVICE OS – POWERFUL FIRMWARE WITH TONS OF FEATURES



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

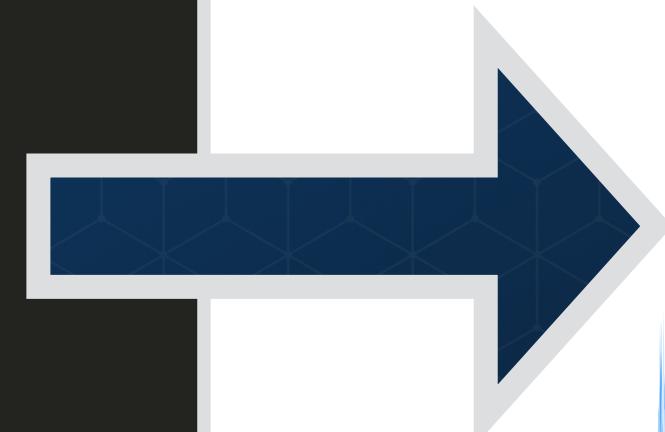
- * Simplify interactions with sensors and actuators
- * Reuse code across projects
- * Leverage contributions from Particle's community of 165k developers

```
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

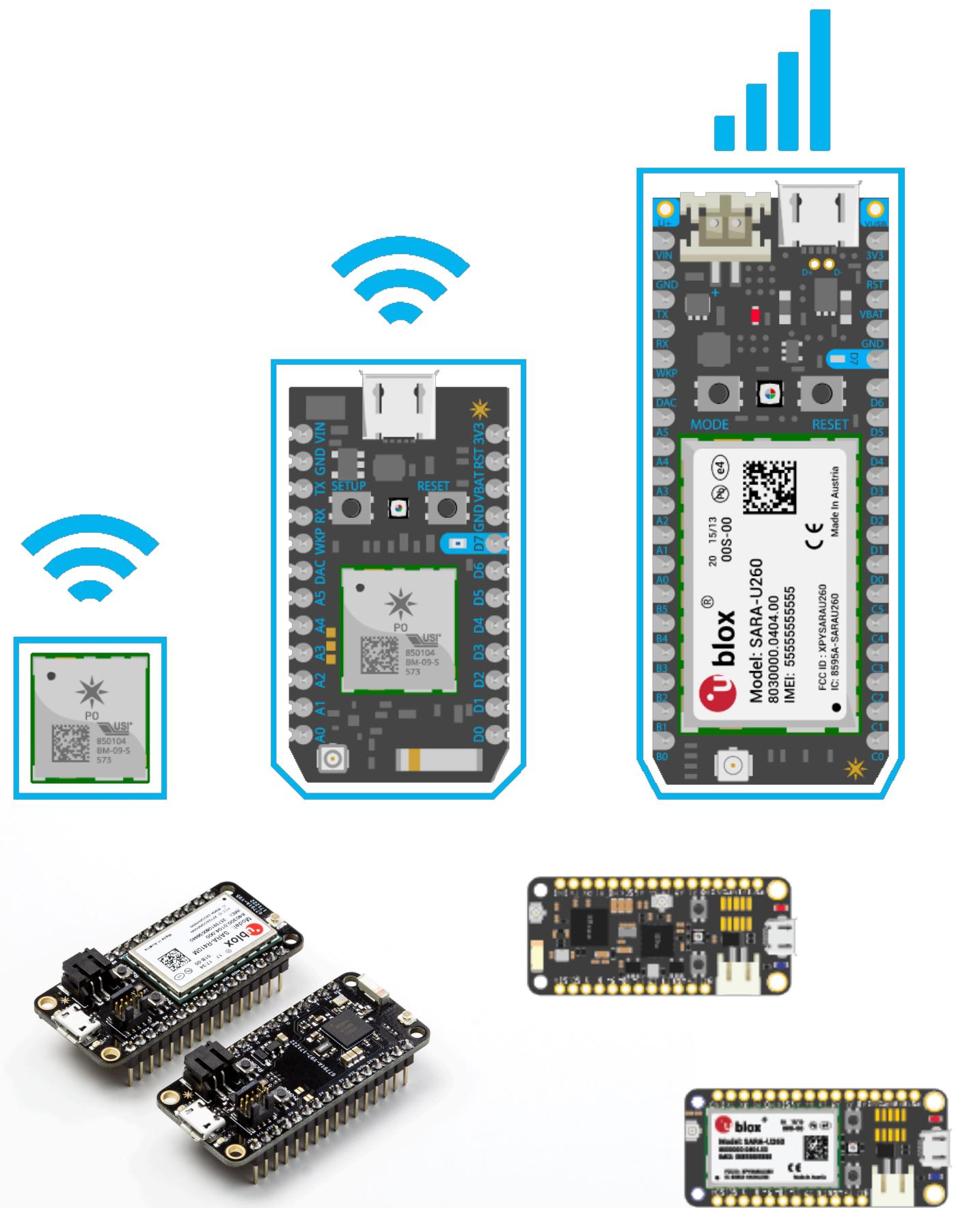
```
void Adafruit_SSD1306 ::display(void) {  
    for (uint16_t i=0; i<(SSD1306_LCDWIDTH*SSD1306_LCDHEIGHT/8); i++) {  
        // send a bunch of data in one tx-mission  
        Wire.beginTransmission(_i2caddr);  
        Wire.write(0x40);  
        for (uint8_t x=0; x<16; x++) {  
            Wire.write(buffer[i]);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



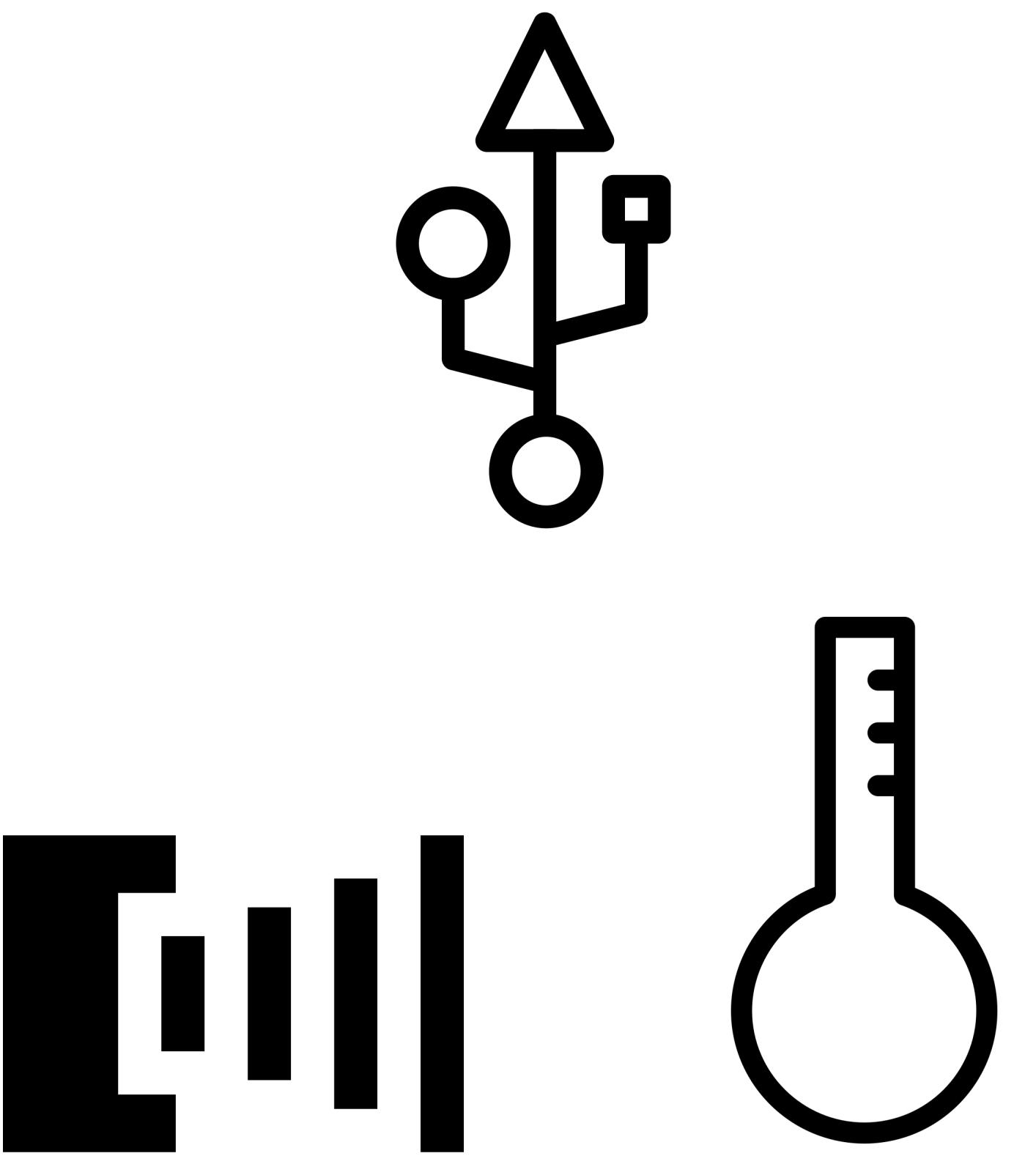
```
display.println("Temp");  
display.print((int)currentTemp);  
display.println("Humidity");  
display.print((int)currentHumidity);  
display.display();
```

THE PARTICLE ECOSYSTEM: HARDWARE, FIRMWARE, & SOFTWARE

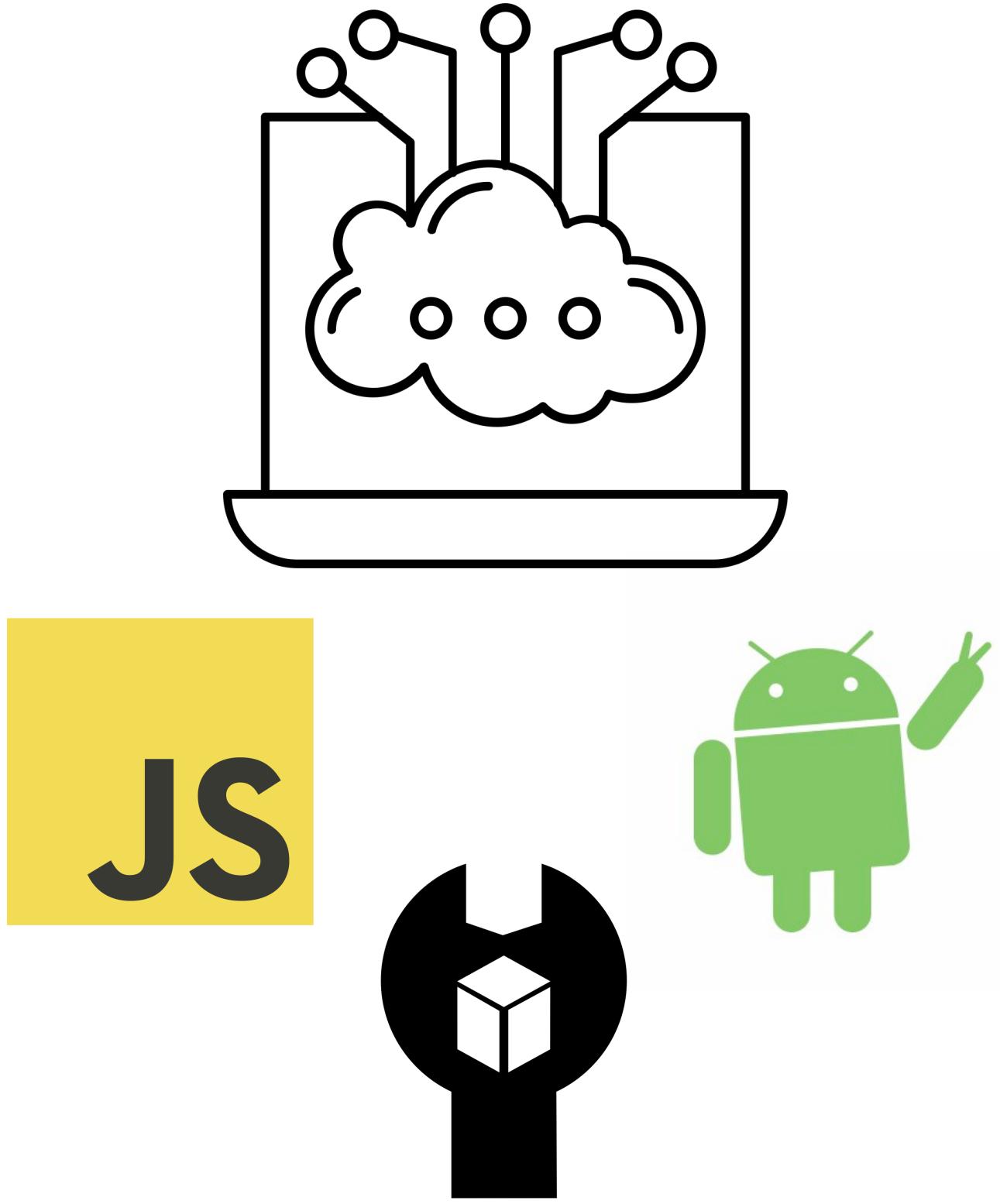
DEVICE HARDWARE FOR PROTOTYPING
& PRODUCTION



DEVICE OS FIRMWARE & LIBRARIES

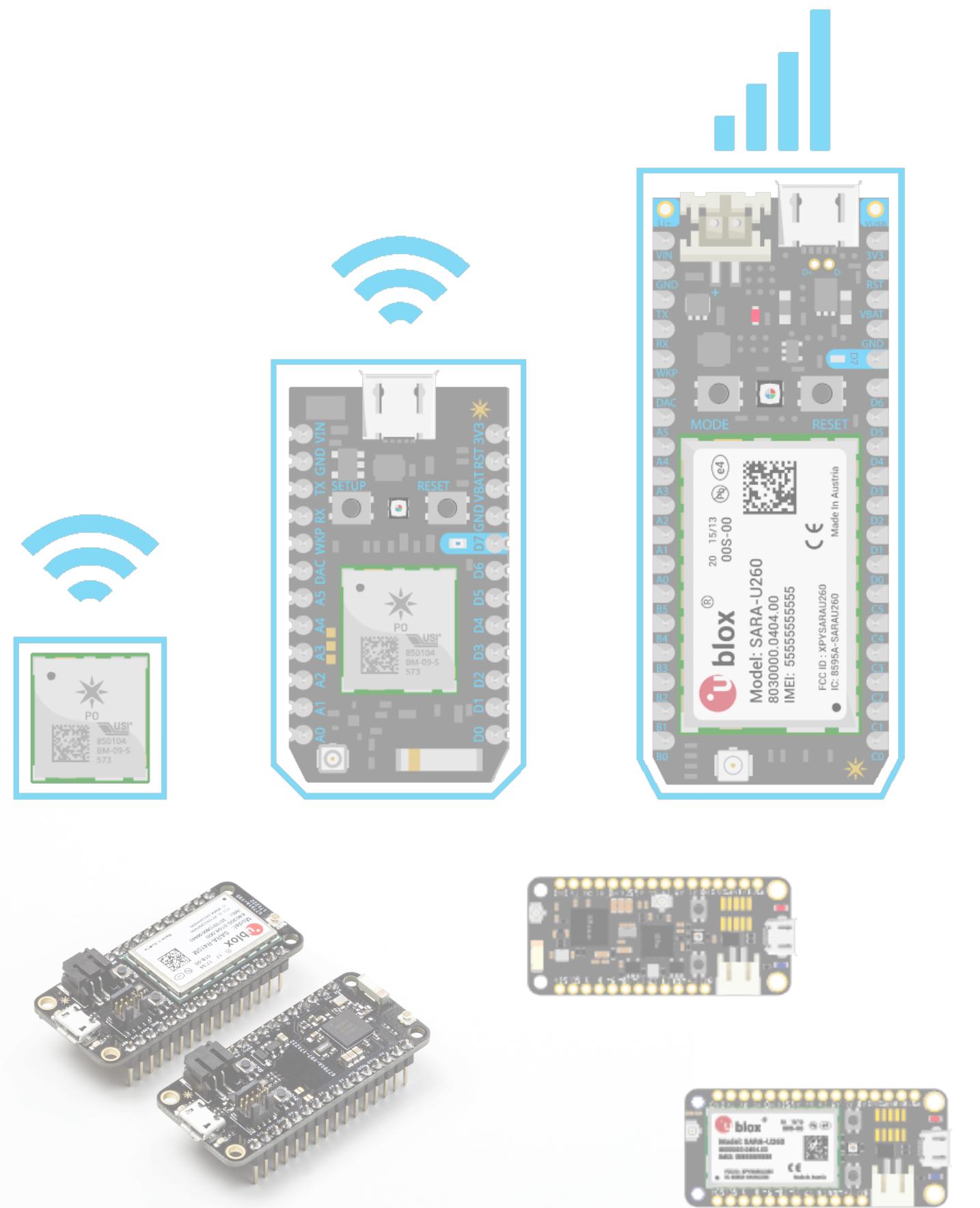


DEVICE CLOUD & SOFTWARE TOOLS

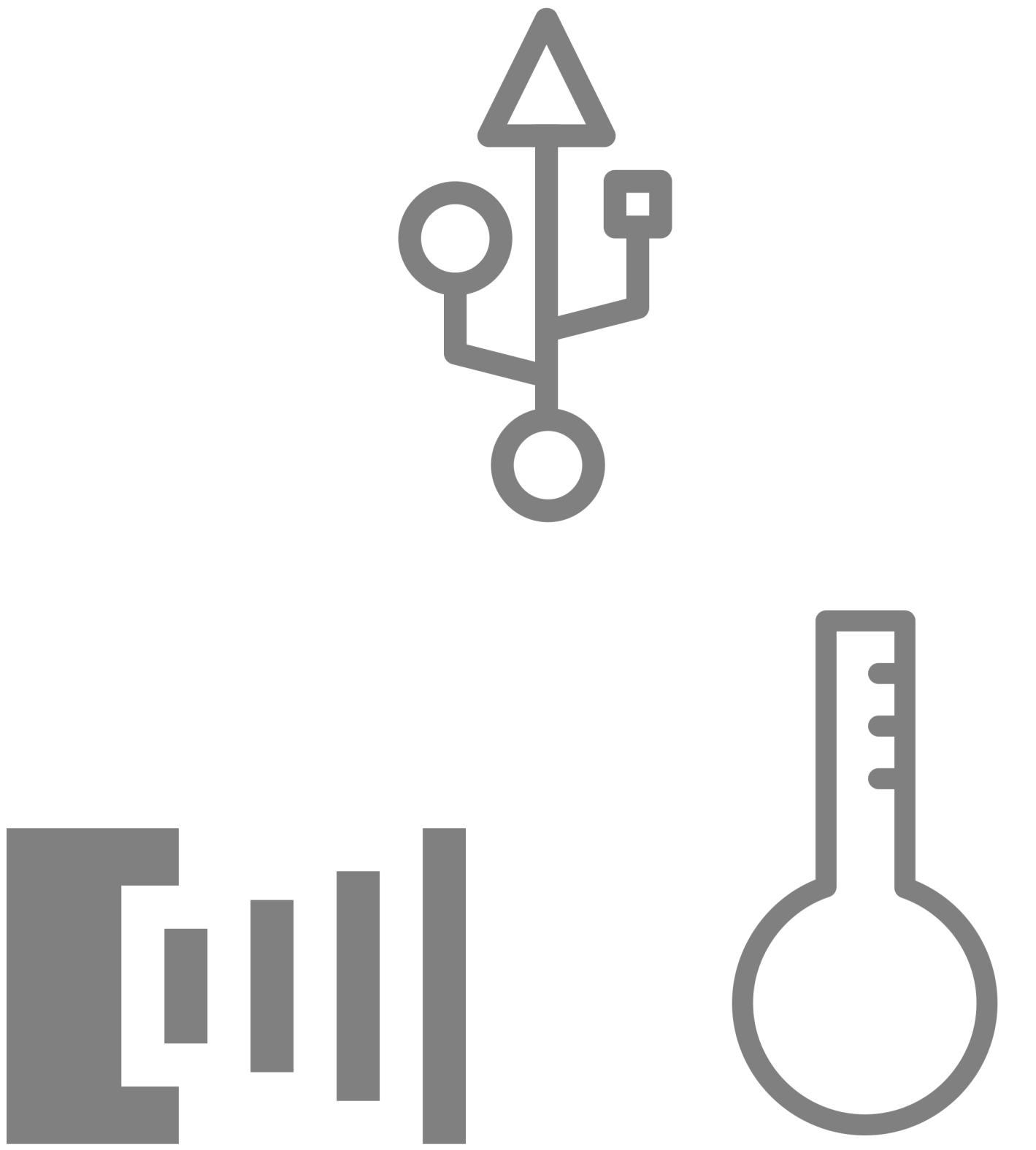


THE PARTICLE ECOSYSTEM: HARDWARE, FIRMWARE, & SOFTWARE

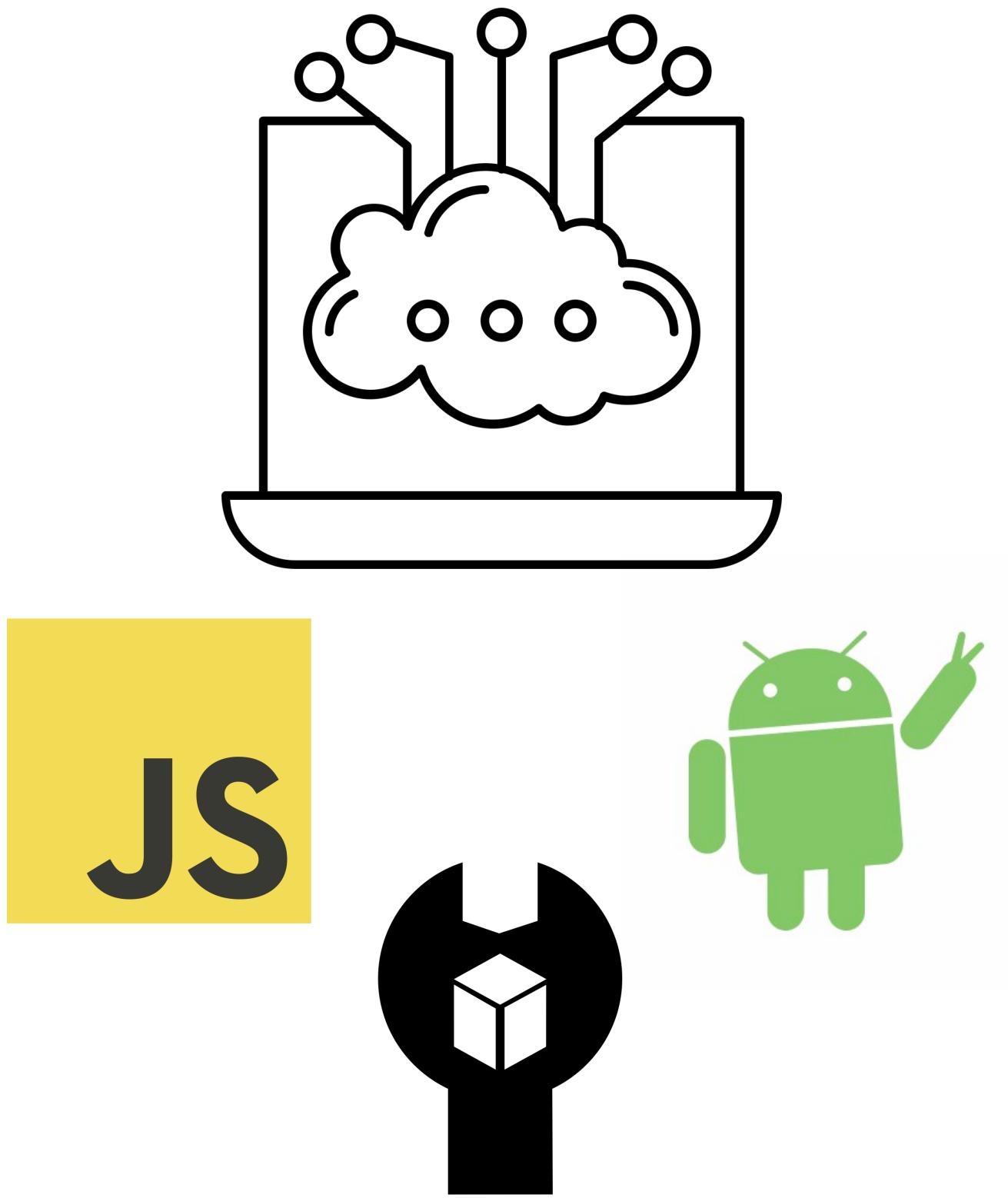
DEVICE HARDWARE FOR PROTOTYPING & PRODUCTION



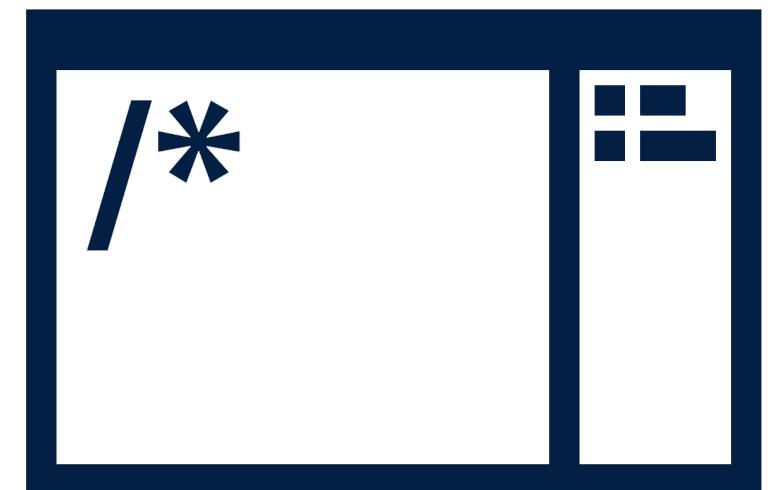
DEVICE OS FIRMWARE & LIBRARIES



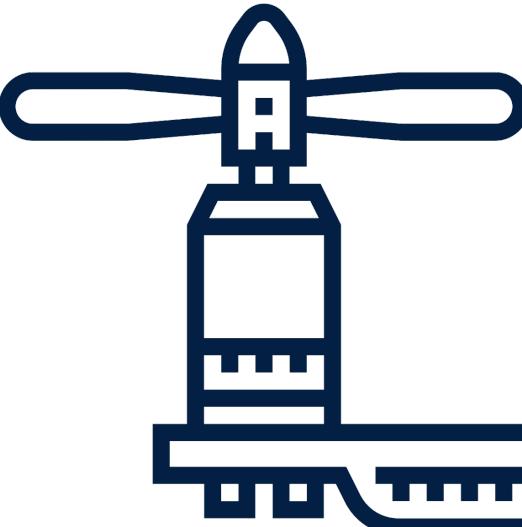
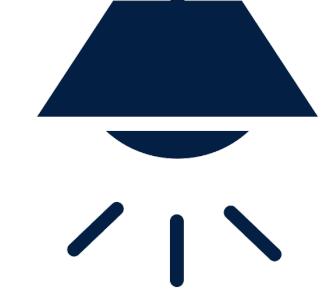
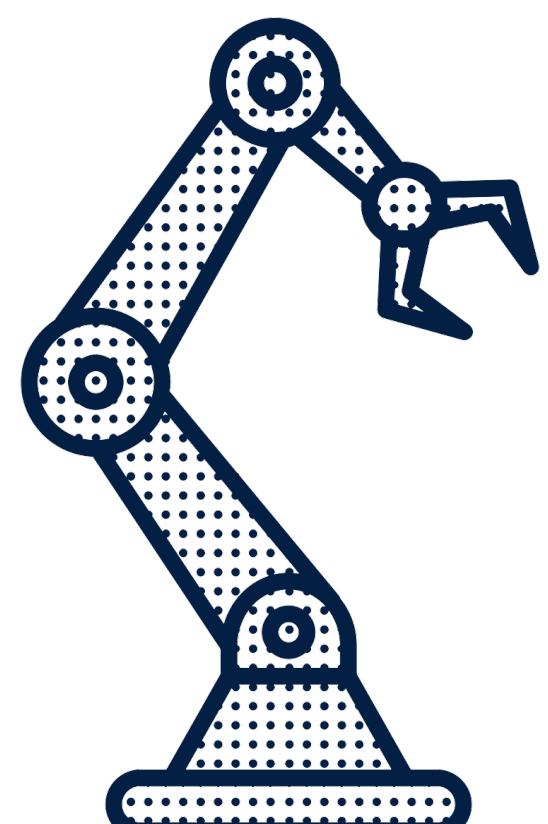
DEVICE CLOUD & SOFTWARE TOOLS



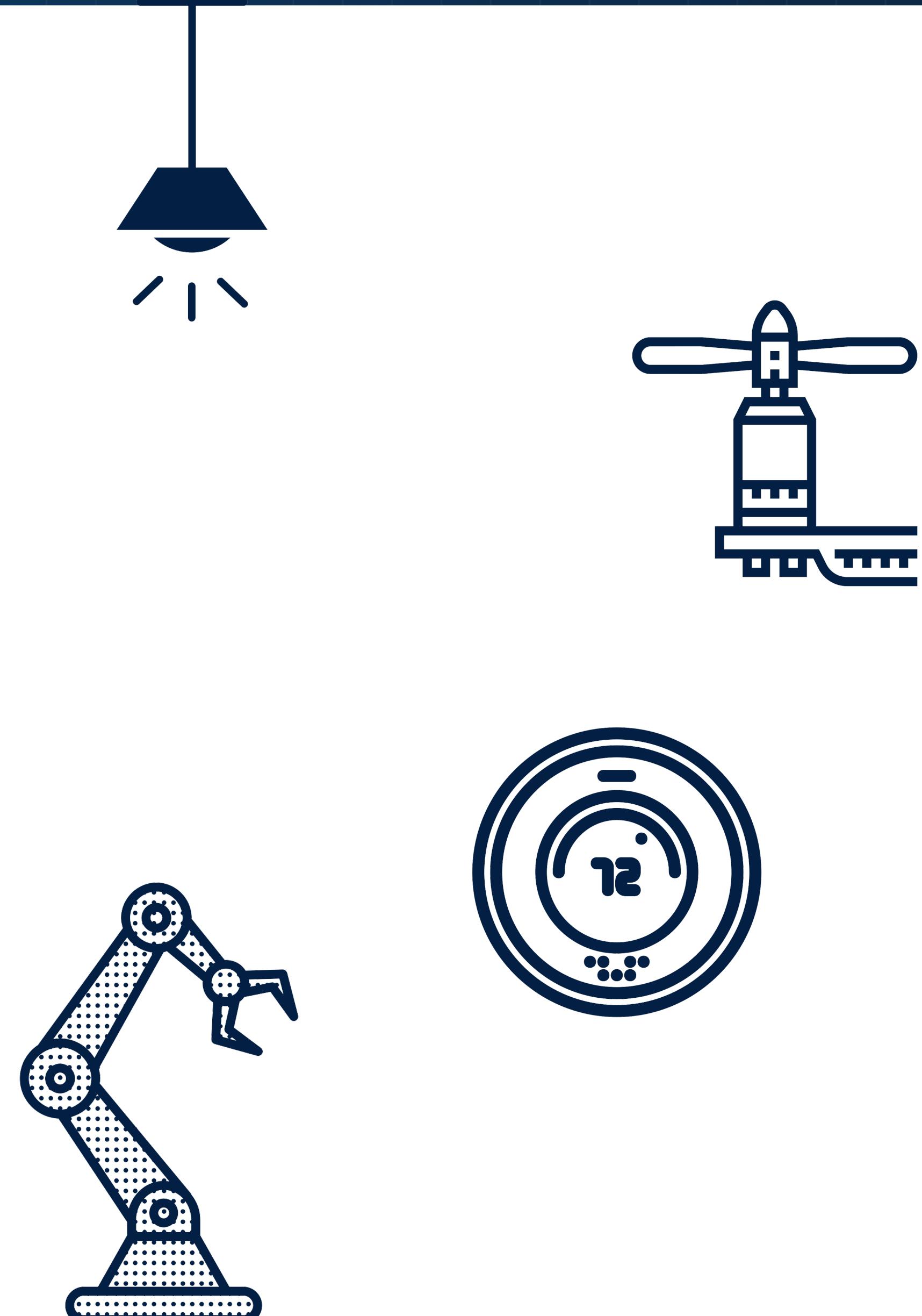
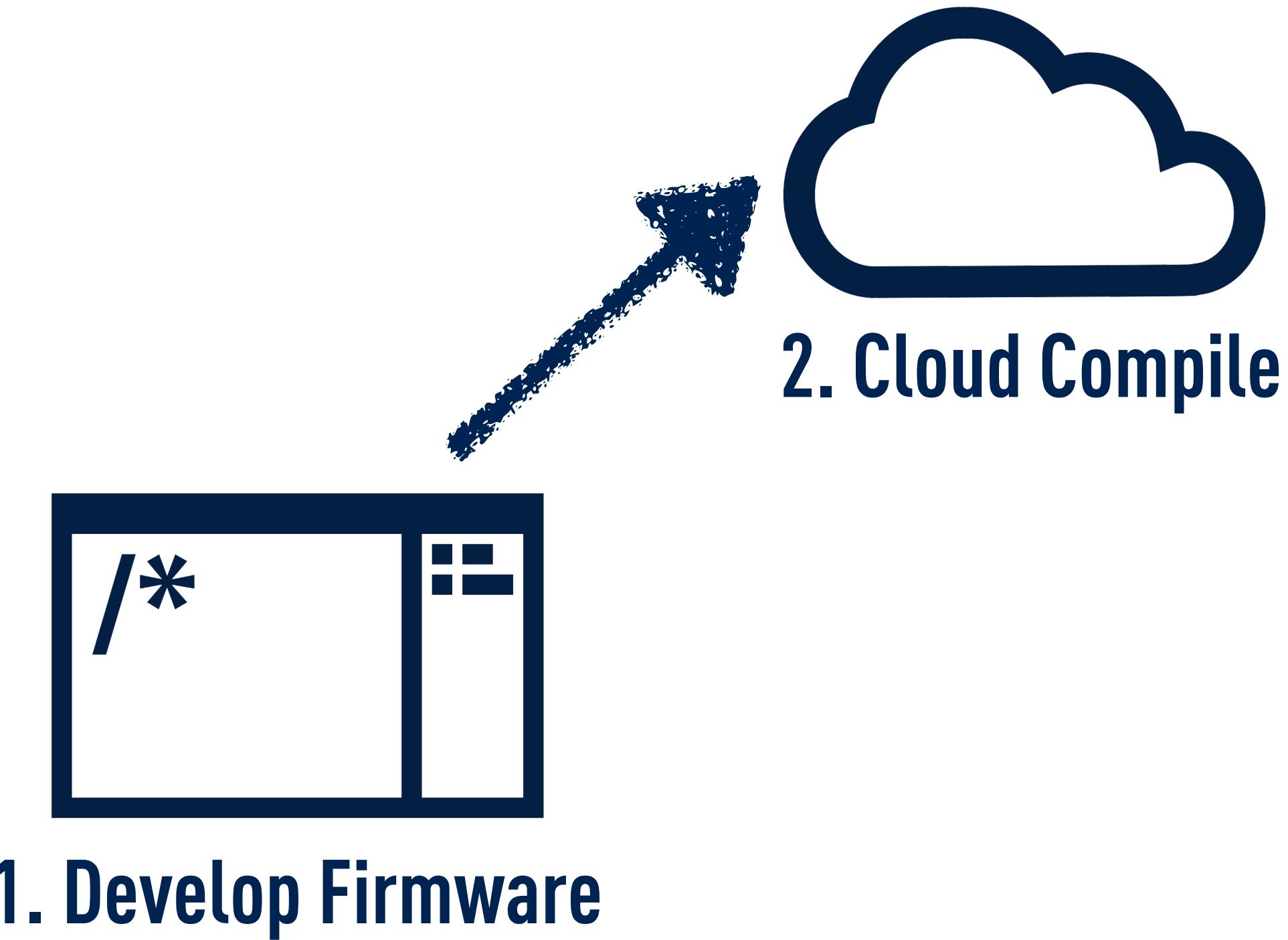
OVER THE AIR (OTA) DEVICE UPDATES



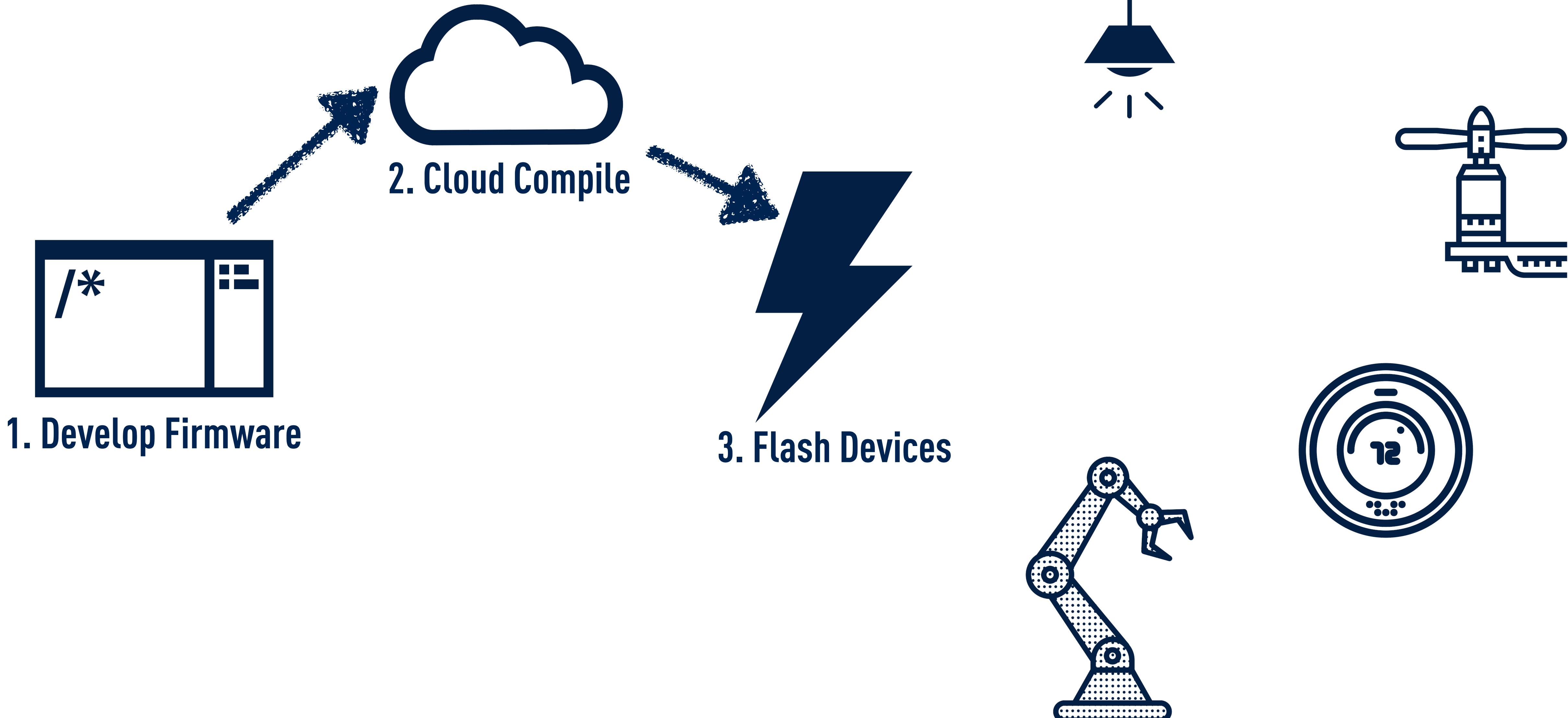
1. Develop Firmware



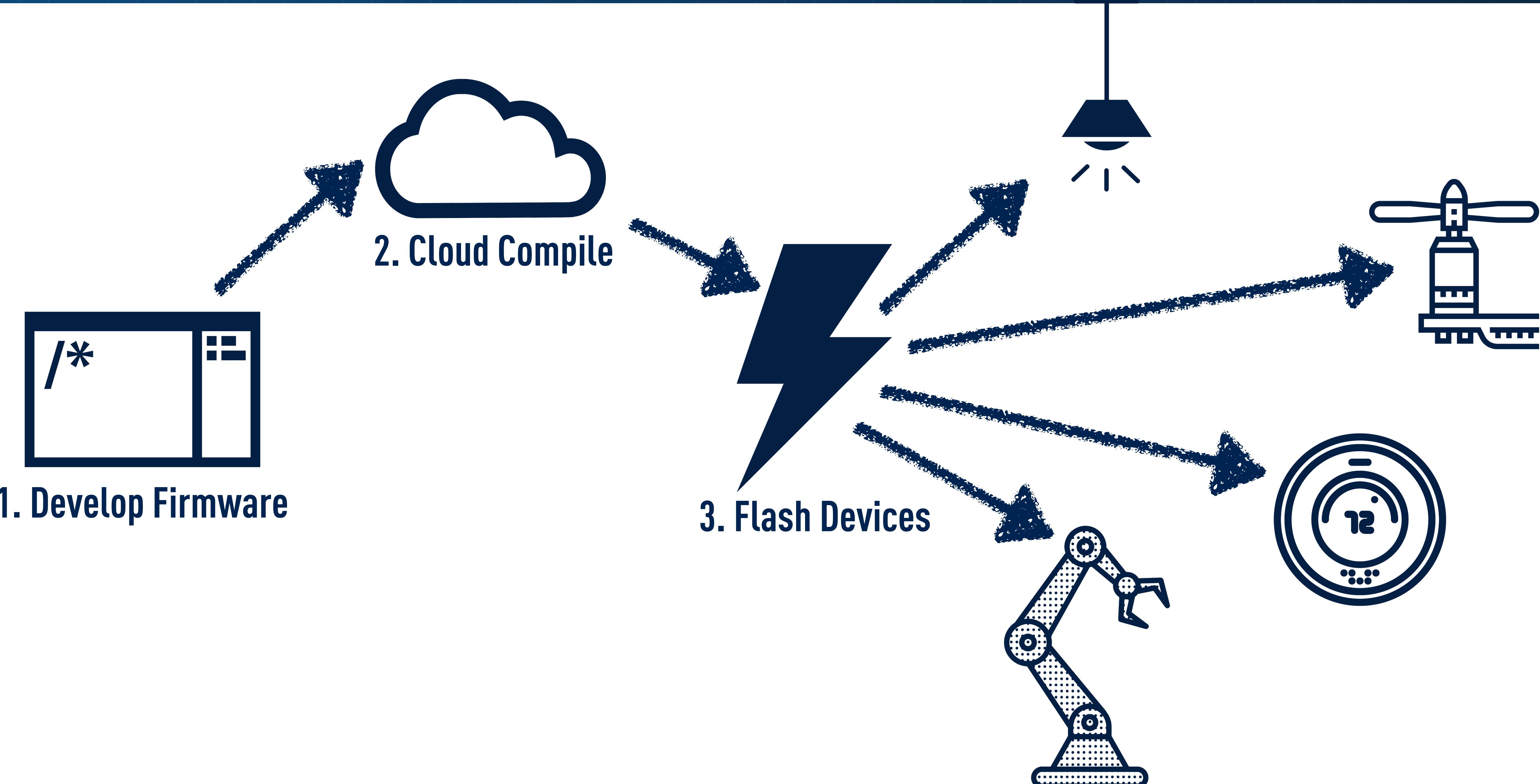
OVER THE AIR (OTA) DEVICE UPDATES



OVER THE AIR (OTA) DEVICE UPDATES



OVER THE AIR (OTA) DEVICE UPDATES



IDES AND DEVELOPER TOOLING

PARTICLE BUILD (WEB)

The Particle Build (Web) interface is a web-based development environment. It features a sidebar on the left with sections for "Particle Apps" (Current Example: TINKER), "My apps" (a search bar and a list of projects like cloudTemp, ledBox, Blink an LED, ParticleBlynkTest, Blink an LED1, RaspberryBlynk, tftTester, sdtest, MEP-LS, runMem), and "Example apps" (1. Blink an LED, 2. Web-Connected LED, 3. Function Variable, 4. Publish, 5. Subscribe, 6. Tinker). The main area is a code editor with syntax highlighting for C++ code related to the TINKER example. The code includes functions for digitalRead and digitalWrite, and a loop function that reads a pin and prints its value.

```
1 // Function prototypes
2 int tinkerDigitalRead(String pin);
3 int tinkerDigitalWrite(String command);
4 int tinkerAnalogRead(String pin);
5 int tinkerAnalogWrite(String command);
6
7 /* This function is called once at start up */
8 void setup()
9 {
10     //Setup the Tinker application here
11
12     //Register all the Tinker functions
13     Particle.function("digitalRead", tinkerDigitalRead);
14     Particle.function("digitalWrite", tinkerDigitalWrite);
15     Particle.function("analogRead", tinkerAnalogRead);
16     Particle.function("analogWrite", tinkerAnalogWrite);
17 }
18
19 /* This Function loops forever */
20 void loop()
21 {
22     //This will run in a loop
23
24     /* Function Name : tinkerDigitalRead
25      * Description : Reads the digital value of a given pin
26      * Input : Pin
27      * Output : None.
28      * Return : Value of the pin (0 or 1) in INT type
29      * Returns a negative number on failure
30 */
31
32     /* Function Name : tinkerDigitalWrite
33      * Description : Sets the specified pin HIGH or LOW
34      * Input : Pin and value
35      * Output : None.
36      * Returns 1 on success and a negative number on failure
37 */
38
39     /* Function Name : tinkerAnalogWrite
40      * Description : Sets the specified pin HIGH or LOW
41      * Input : Pin and value
42      * Output : None.
43      * Returns 1 on success and a negative number on failure
44 */
45
46     /* Function Name : tinkerAnalogRead
47      * Description : Reads the analog value of a given pin
48      * Input : Pin
49      * Output : Value from 0 to 1023
50      * Returns a negative number on failure
51 */
52 }
53
54 /* Function Name : tinkerDigitalWrite
55 * Description : Sets the specified pin HIGH or LOW
56 * Input : Pin and value
57 * Output : None.
58 * Returns 1 on success and a negative number on failure
59 */
60
61 int tinkerDigitalWrite(String command)
62 {
63     bool value = 0;
64     String commandString = command;
65     int pinNumber = command.charAt(0) - '0';
66     //Safety check to see if the pin numbers are within limits
67     if(pinNumber < 0 || pinNumber > 7) return -1;
68
69     if(command.substring(1,2) == "HIGH") value = 1;
70     else if(command.substring(1,2) == "LOW") value = 0;
71     else return -2;
72
73     if(command.startsWith("D"))
74     {
75         pinMode(pinNumber, OUTPUT);
76         digitalWrite(pinNumber, value);
77     }
78     else if(command.startsWith("A"))
79     {
80         analogWrite(pinNumber, value);
81     }
82 }
```

Please enter the title to create your first app.

PARTICLE WORKBENCH

The Particle Workbench is a desktop application. The interface includes a sidebar with "OPEN EDITORS" (strobby.ino), "STROBY" (vscode, ipch, cortex-debug.periph..., launch.json, settings.json, shared, src, strobby.cpp, strobby.ino, target, obj, strobby.bin, strobby.bin.crc_block, strobby.bin_no_crc, strobby.dfu, strobby.elf, strobby.hex, strobby.lst, strobby.map, 1a02ea0003473531383..., 1a02ea0003473531383..., 1a02ea0003473531383..., backup_rsa_1a02ea000..., photon_firmware_15531..., project.properties, README.md), and "OUTLINE". The main area displays the "Welcome to Particle Workbench" screen with a "Get Started" button and a "Development Workflow" diagram. The workflow consists of four steps: CODE (Create new project, Install library, Launch CLI), TARGET (Install target Device OS, Select target hardware, Select target device), COMPILE (Local compile, Cloud compile, Start a build task), and FLASH (Local flash, Cloud flash, Launch debugger).

PARTICLE CLI

The Particle CLI terminal window shows the setup process for a new project. It starts with a "particle setup" command, followed by a series of prompts asking if the user wants to use their account, if they have a Photon connected via USB, and if they want to continue with serial setup. It then detects a Photon connected via USB and asks if the user wants to continue. Finally, it prompts the user to scan for nearby Wi-Fi networks.

```
1. particle setup
bsatrom@muon: ~
> 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) [Y]
```

SDKS FOR MOBILE AND WEB DEVELOPMENT

```
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")
            }
        }
    }
)
```

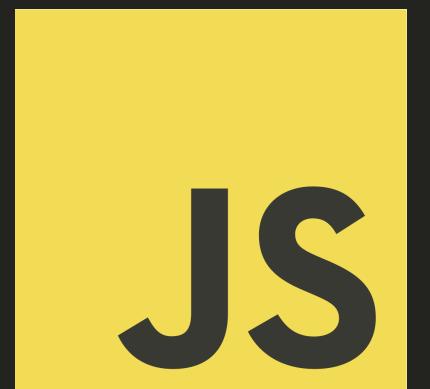


```
List<ParticleDevice> devices = ParticleCloudSDK.getCloud().getDevices();
for (ParticleDevice device : devices) {
    if (device.getName().equals("myDevice")) {
        doSomethingWithMyDevice(device);
        break;
    }
}
```

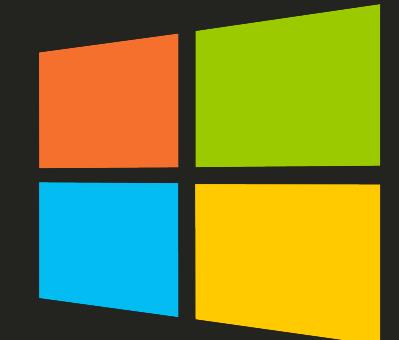


```
var brew = particle.callFunction({
    deviceId: 'DEVICE_ID',
    name: 'brew',
    argument: 'D0:HIGH',
    auth: token
});

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



```
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

Particle

Devices > View Device

ID: 3b001e000247363339343638 Name: trash-panda

Device OS: 0.8.0-rc.8 Type: Photon

Serial Number: PHHMAB819ZY6QXD Last Handshake: Jul 12th 2018, 9:32 am

EVENT LOGS **DIAGNOSTICS NEW**

spark/flash/status device came online spark/device/last_reset spark/device/diagnostics/update

PAUSE SEE IN TERMINAL PUBLISH EVENT

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

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

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

Particle

Devices > View Device

ID: 3b001e000247363339343638 Name: trash-panda

Device OS: 0.8.0-rc.8 Type: Photon

Serial Number: PHHMAB819ZY6QXD Last Handshake: J

EVENT LOGS DIAGNOSTICS NEW

spark/flash/status device came online spark/device/last_rese

PAUSE SEE IN TERMINAL PUBLISH EVENT

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

Docs | Contact Sales | Support | bsatrom@gmail.com | PING | EDIT

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

EVENT LOGS DIAGNOSTICS NEW

spark/flash/status device came online spark/device/last_rese

PAUSE SEE IN TERMINAL PUBLISH EVENT

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

RUN TESTS

INTEGRATIONS FOR EXTENDING YOUR IOT SOLUTIONS TO OTHER CLOUDS

The screenshot shows the Particle Integrations interface. On the left, a sidebar contains icons for Device Management, Project Management, and Analytics. The main area displays several integration cards:

- Azure IoT Hub**: Icons for brew_temp, any device, and tangible-iot-hu...
- Google Cloud Platform**: Icons for googlePublish, brew_buddy_2, and brew-reading
- Webhook**: Icons for BatteryState, any device, and everlive.com
- Webhook**: Icon for BatchTemp
- Webhook**: Icons for BrewStageTemp, any device, and firebaseio.com

In the center, a dashed box highlights a card with a plus sign and the text "NEW INTEGRATION".

At the top right, there are navigation links: Docs, Contact Sales, Support, and an email link (bsatrom@gmail.com). On the far right, a sidebar lists the current integrations: Google Maps, Azure IoT Hub, Google Cloud Platform, and Webhook.

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

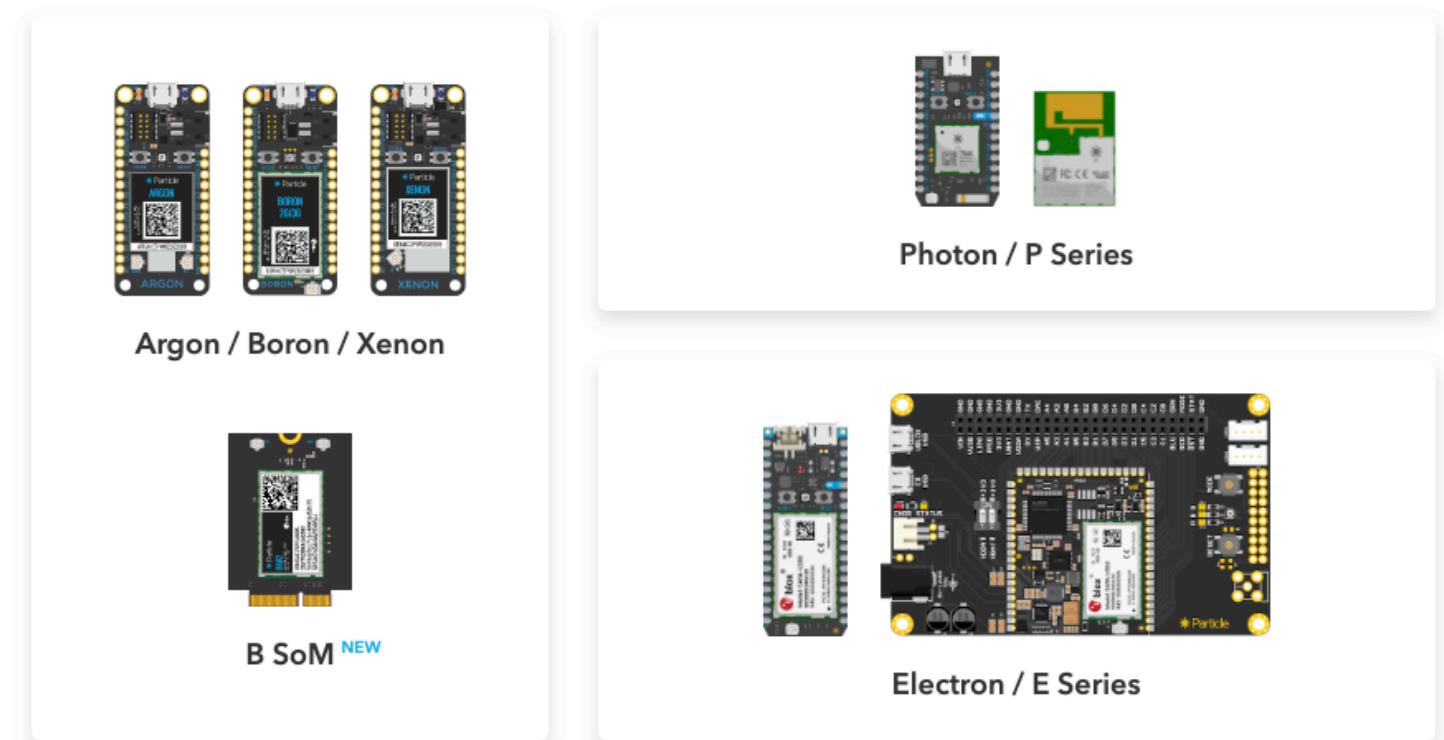
USE WITH ETHERNET?

Toggle Ethernet Featherwing setup

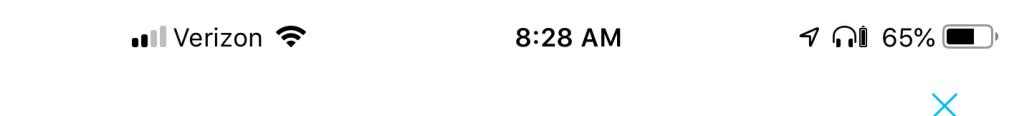
A screenshot of a mobile application interface for setting up a Particle Argon device. The top right shows an email address: workshop@reflektor.dk. Below it, the text "Get your Argon ready for setup" is displayed. In the center, there is a small image showing a hand plugging a black power cable into a Particle Argon board which is mounted on a breadboard. A callout box points to the power cable with the text "Plug your device into a power source". Below this image, there is a checkbox with the text "I have attached the antenna and see that the device is blinking blue". At the bottom, a large blue button labeled "NEXT" is visible. At the very bottom, there is a link "USE WITH ETHERNET?" followed by a "Toggle Ethernet Featherwing setup" button with a gear icon.

WAYS TO CLAIM A NEW PARTICLE DEVICE

WEB



MOBILE



SCAN DATA MATRIX

PARTICLE CLI

```
1. particle setup (node)
~ $ 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 3b001e00024736339343638
> Obtained magical secure claim code.

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

bsatrom
@muon

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