Particle

PARTICLE 201 - PRODUCTS, DIAGNOTICS, FLEET MANAGEMENT AND ON-DEVICE DEBUGGING

GETTING STARTED WITH THE CLI

USING WEBHOOKS AND INTEGRATIONS

FLEET MANAGEMENT & DIAGNOSTICS

ON-DEVICE DEBUGGING

GETTING STARTED WITH THE CLI

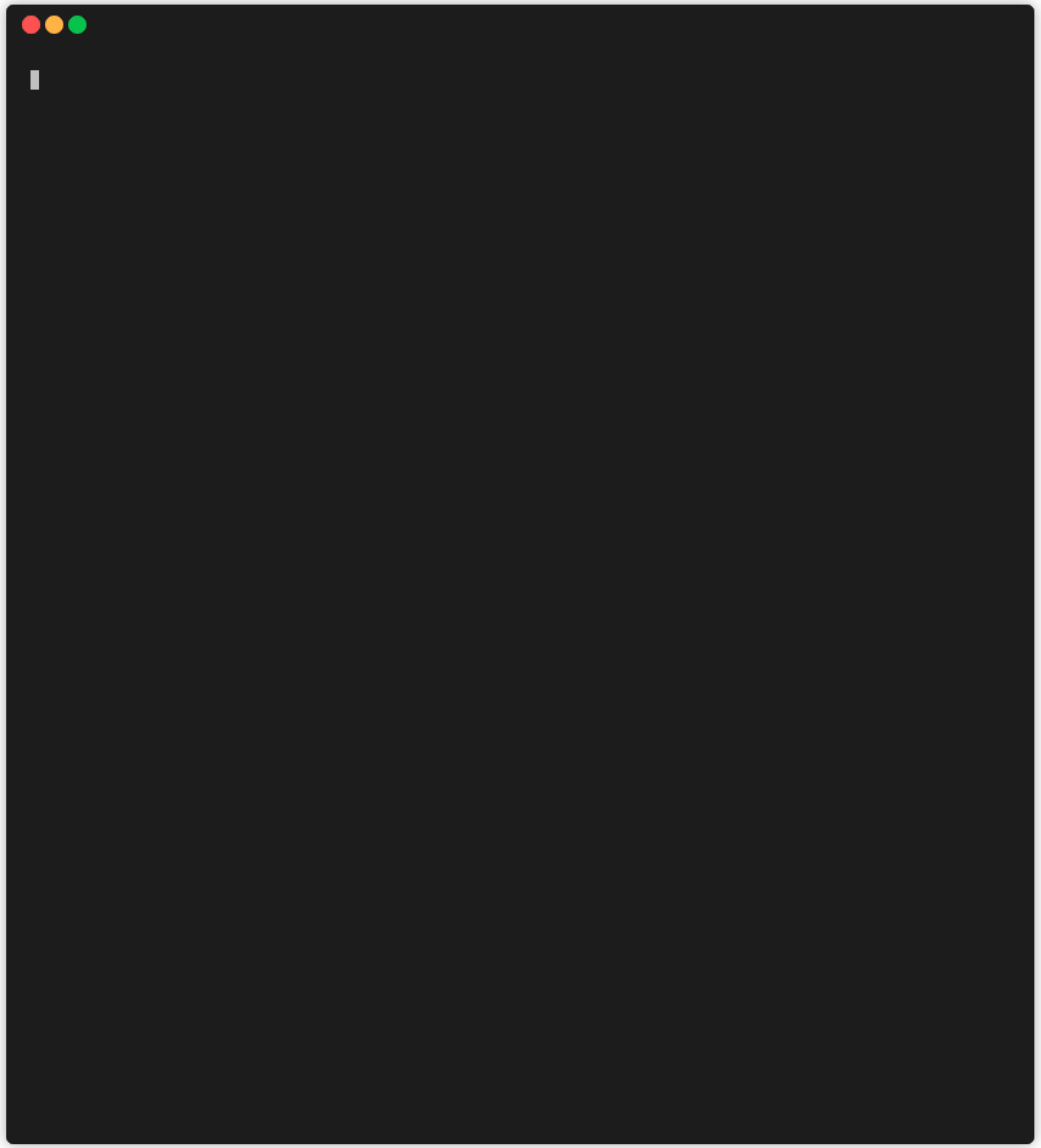
USING WEBHOOKS AND INTEGRATIONS

FLEET MANAGEMENT & DIAGNOSTICS

ON-DEVICE DEBUGGING

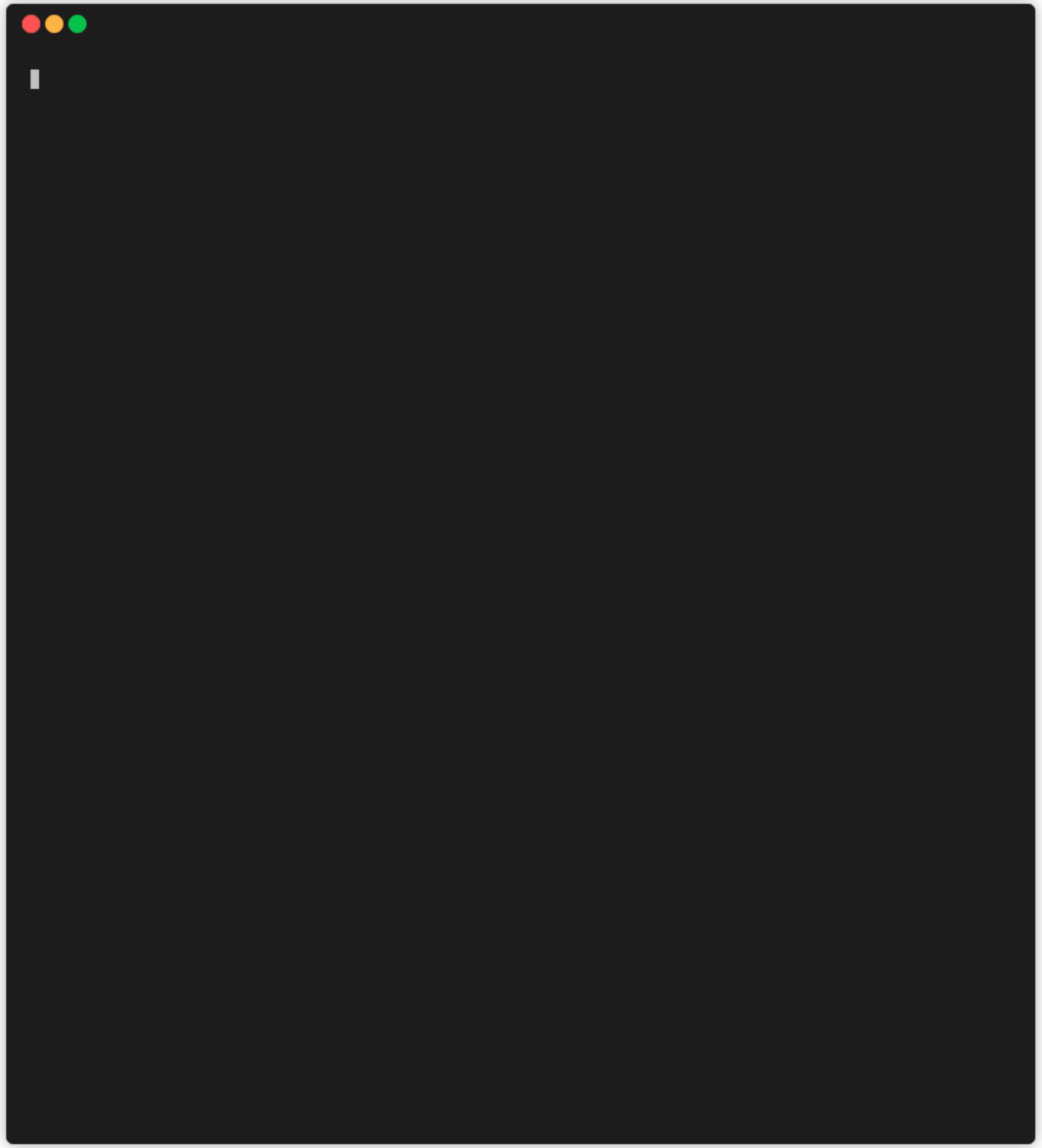
PARTICLE COMMAND-LINE INTERFACE (CLI)

- List all your devices
- Setup a new device
- Call functions and get variables
- Publish and subscribe to events
- Create new projects
- Compile firmware and flash devices
- Search for and install libraries
- Setup webhooks



PARTICLE COMMAND-LINE INTERFACE (CLI)

- List all your devices
- Setup a new device
- Call functions and get variables
- Publish and subscribe to events
- Create new projects
- Compile firmware and flash devices
- Search for and install libraries
- Setup webhooks



PARTICLE DOCTOR FOR DEVICE RECOVERY

- Update Device OS
- Reset device antenna
- Reset IP configuration
- Reset SoftAP hotspot
- Clear EEPROM
- Clear Wi-Fi credentials
- Reset server and device key
- Flash the default Tinker app

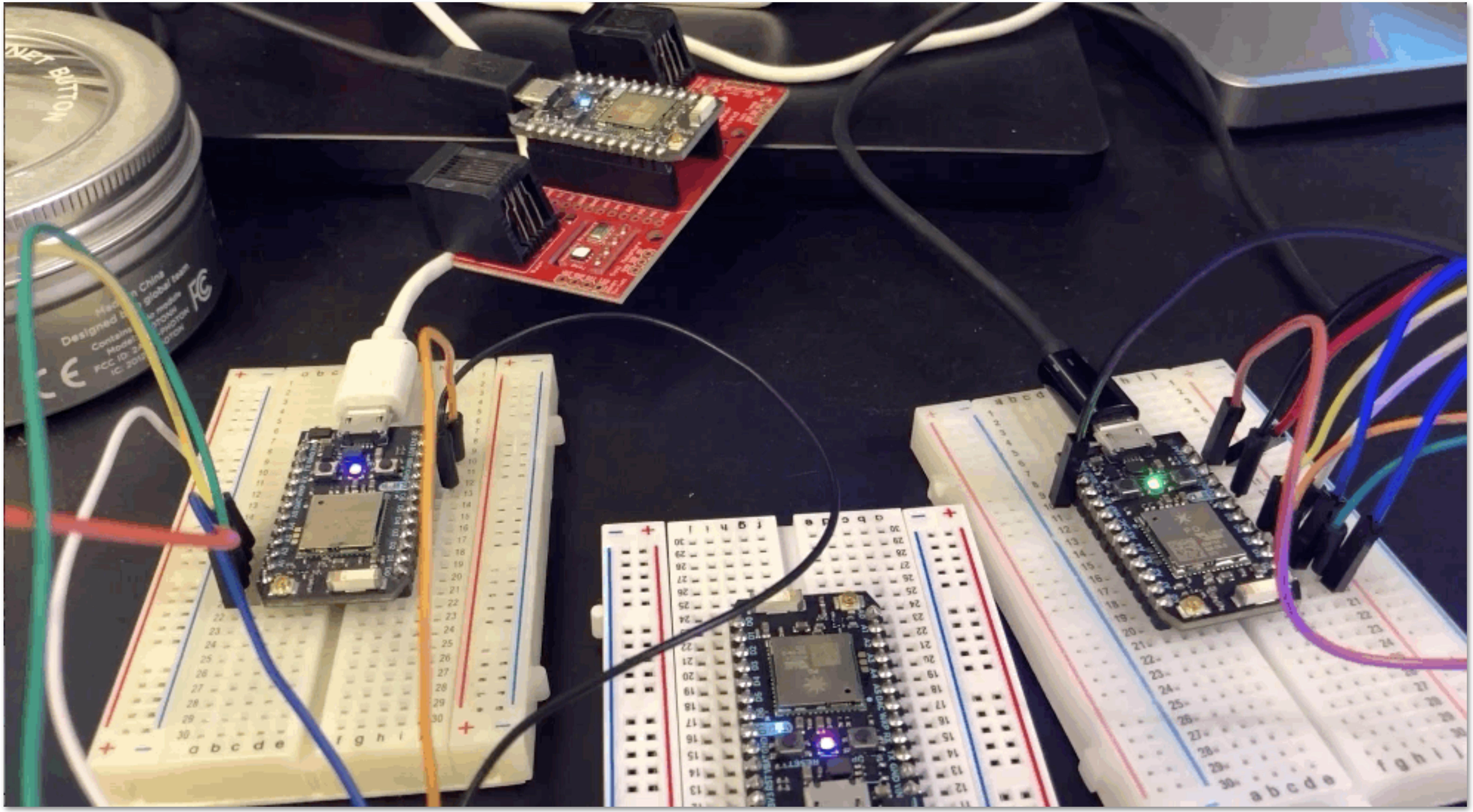
```
muon:~ bsatrom$
```

PARTICLE DOCTOR FOR DEVICE RECOVERY

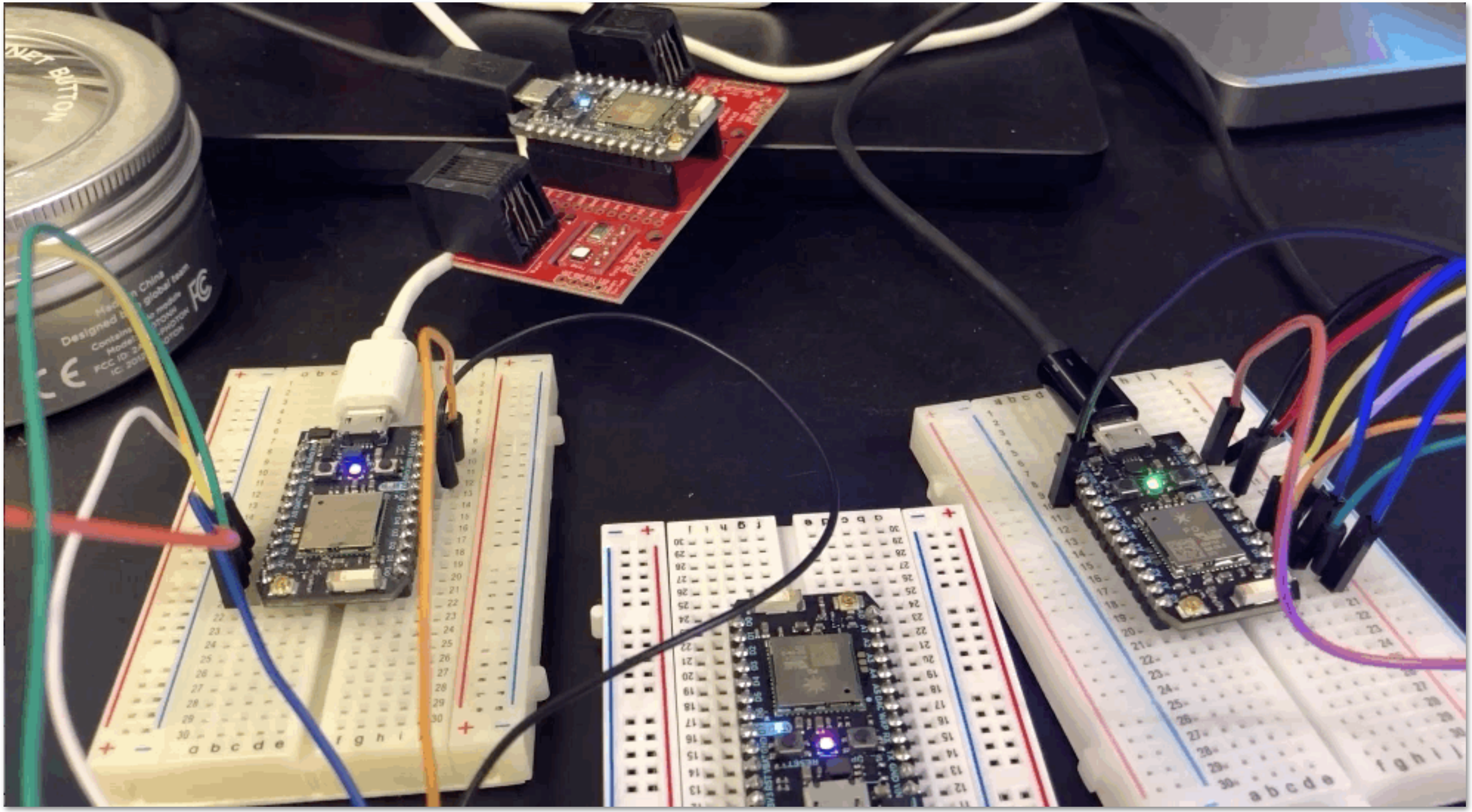
- Update Device OS
- Reset device antenna
- Reset IP configuration
- Reset SoftAP hotspot
- Clear EEPROM
- Clear Wi-Fi credentials
- Reset server and device key
- Flash the default Tinker app

```
muon:~ bsatrom$
```

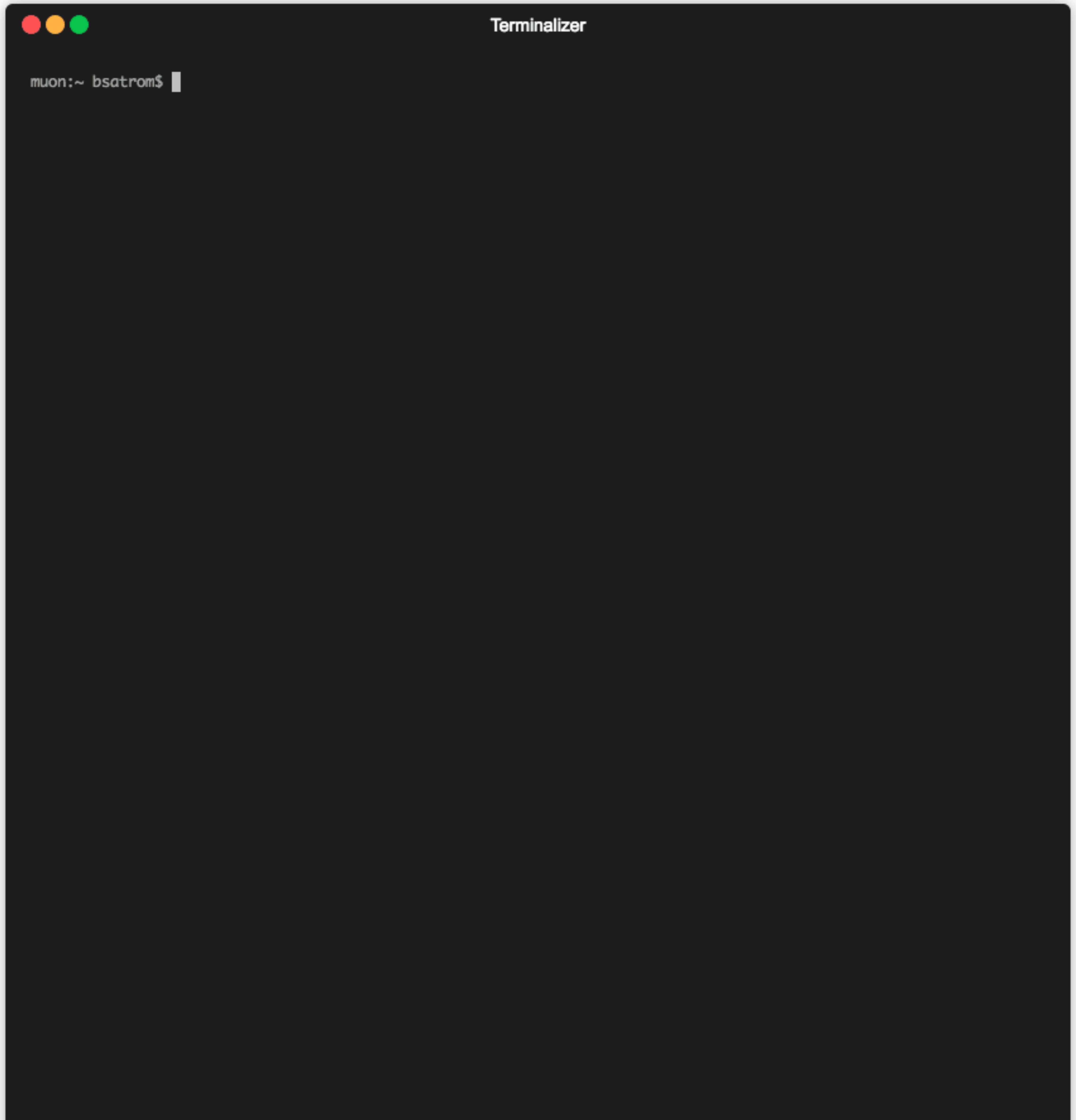
PARTICLE NYAN



PARTICLE NYAN



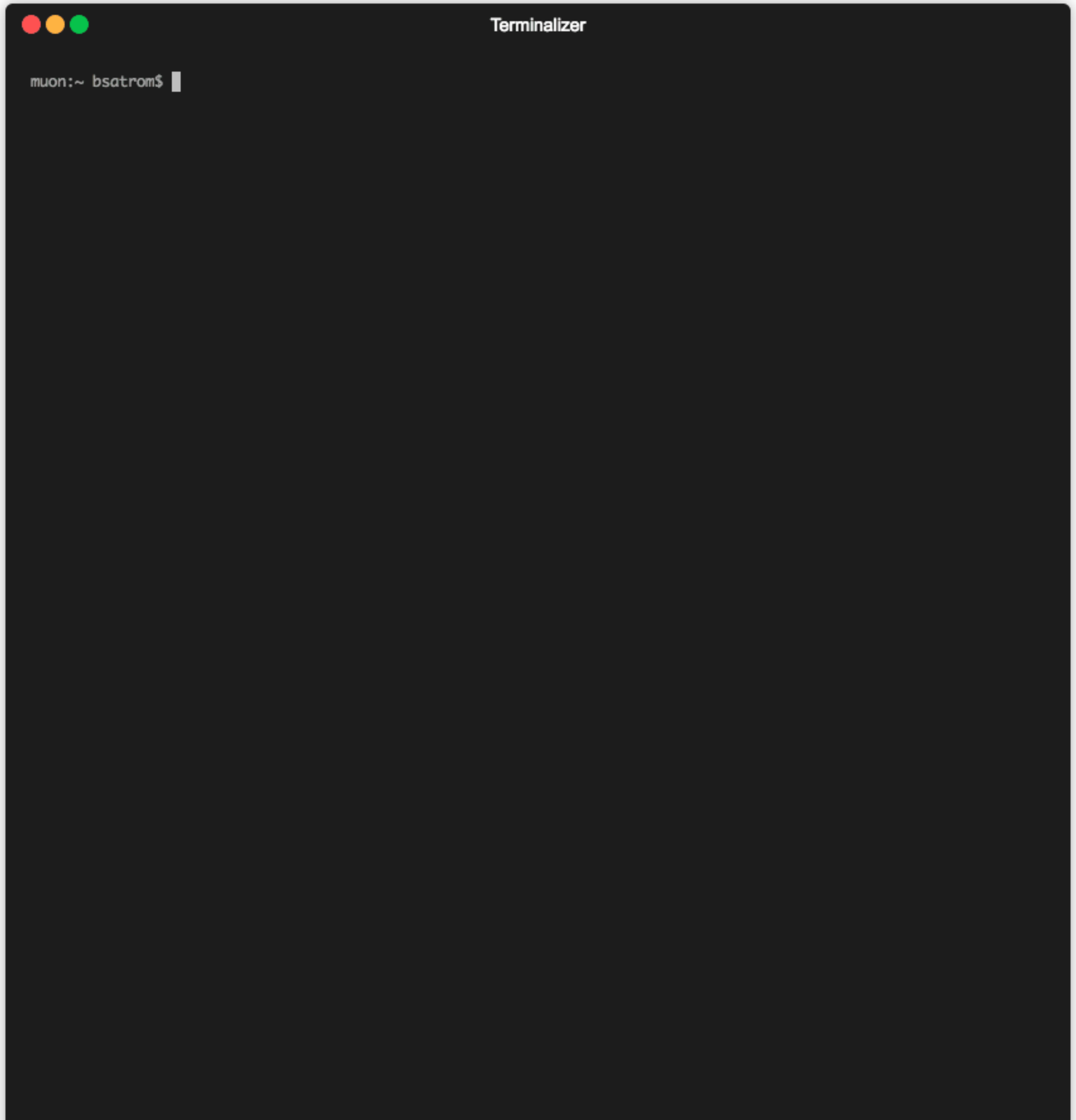
PARTICLE DEVICE CLOUD API



Terminalizer

```
muon:~ bsatrom$ 
```

PARTICLE DEVICE CLOUD API



```
muon:~ bsatrom$
```

GETTING STARTED WITH THE CLI

DEMO

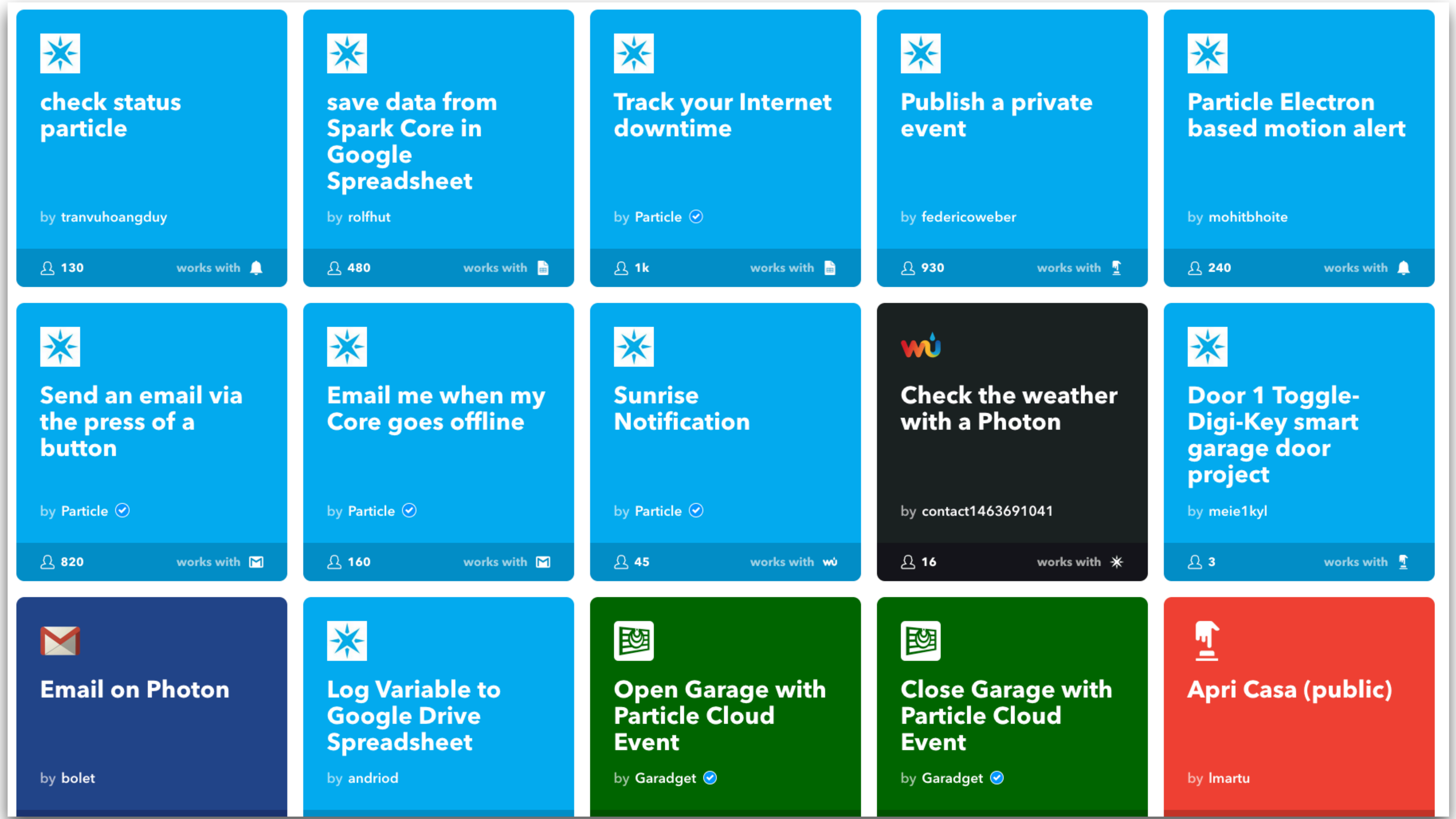
GETTING STARTED WITH THE CLI

USING WEBHOOKS AND INTEGRATIONS

FLEET MANAGEMENT & DIAGNOSTICS

ON-DEVICE DEBUGGING

IFTTT (IF THIS, THEN THAT) + PARTICLE



check status particle
by tranvuhoangduy
130 works with

save data from Spark Core in Google Spreadsheet
by rolfhut
480 works with

Track your Internet downtime
by Particle
1k works with

Publish a private event
by federicoweber
930 works with

Particle Electron based motion alert
by mohitbhoite
240 works with

Send an email via the press of a button
by Particle
820 works with

Email me when my Core goes offline
by Particle
160 works with

Sunrise Notification
by Particle
45 works with

Check the weather with a Photon
by contact1463691041
16 works with

Door 1 Toggle-Digi-Key smart garage door project
by meie1kyl
3 works with

Email on Photon
by bolet

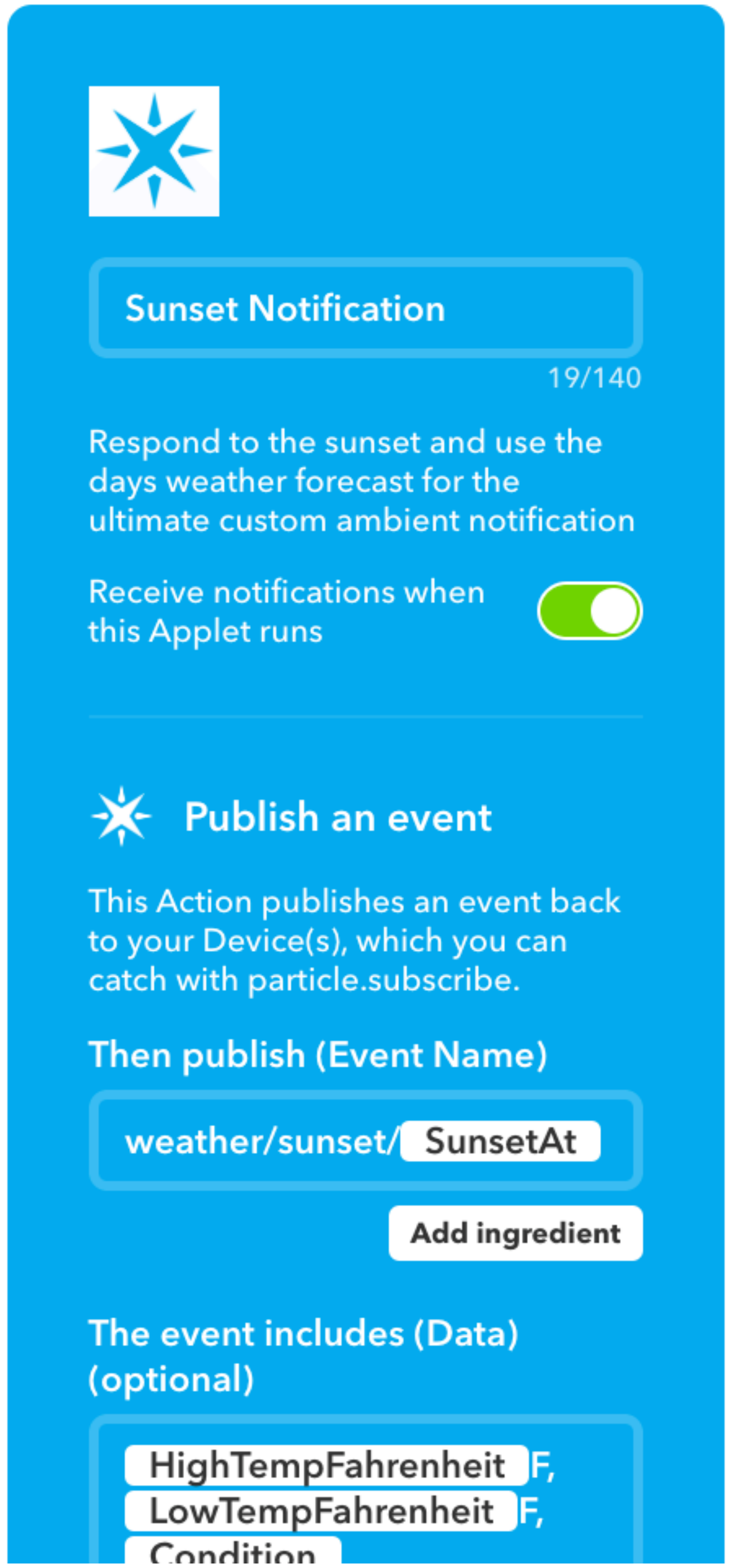
Log Variable to Google Drive Spreadsheet
by andriod

Open Garage with Particle Cloud Event
by Garadget

Close Garage with Particle Cloud Event
by Garadget

Apri Casa (public)
by lmartu

IFTTT (IF THIS, THEN THAT) + PARTICLE



Sunset Notification

19/140

Respond to the sunset and use the days weather forecast for the ultimate custom ambient notification

Receive notifications when this Applet runs

Publish an event

This Action publishes an event back to your Device(s), which you can catch with particle.subscribe.

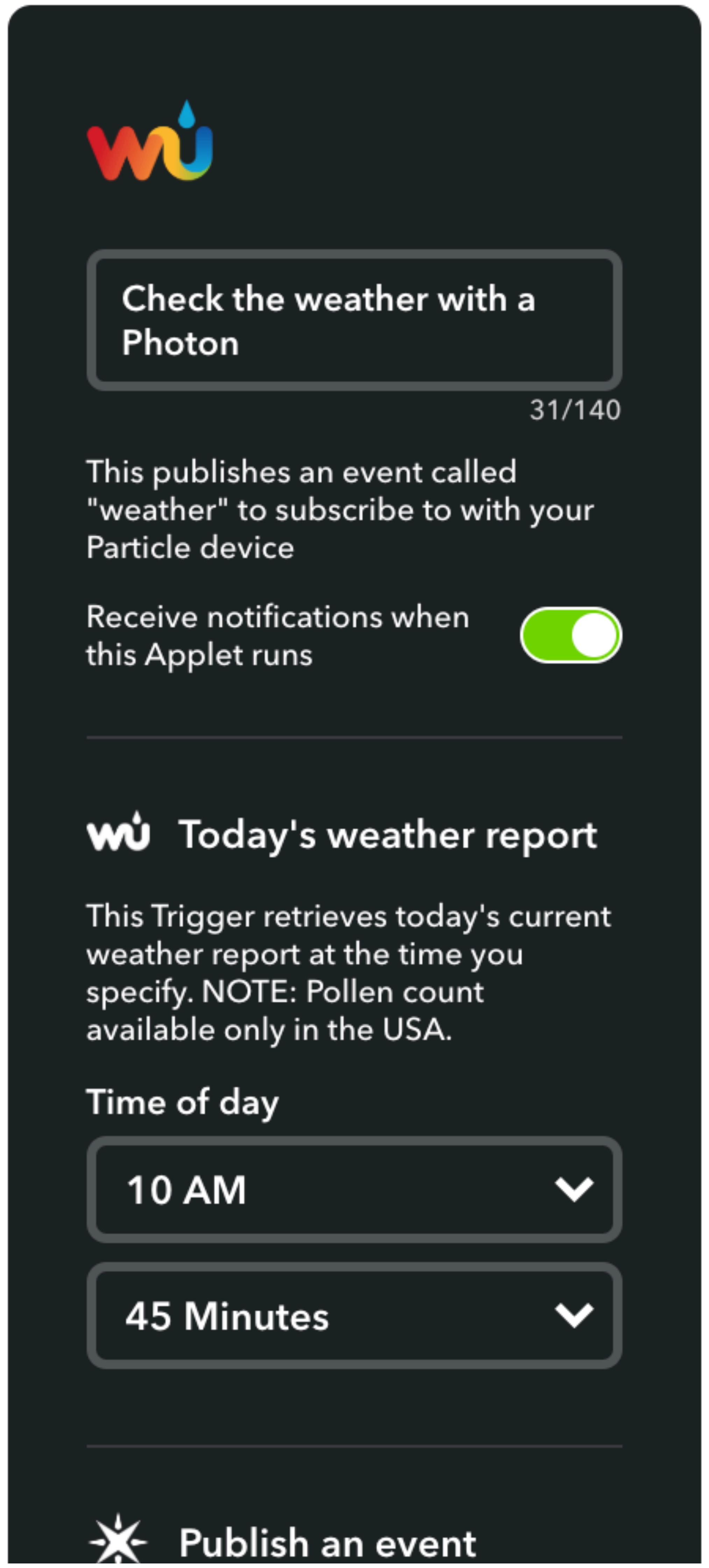
Then publish (Event Name)

weather/sunset/ SunsetAt

Add ingredient

The event includes (Data) (optional)

HighTempFahrenheit F, LowTempFahrenheit F, Condition



Check the weather with a Photon

31/140

This publishes an event called "weather" to subscribe to with your Particle device

Receive notifications when this Applet runs

Today's weather report

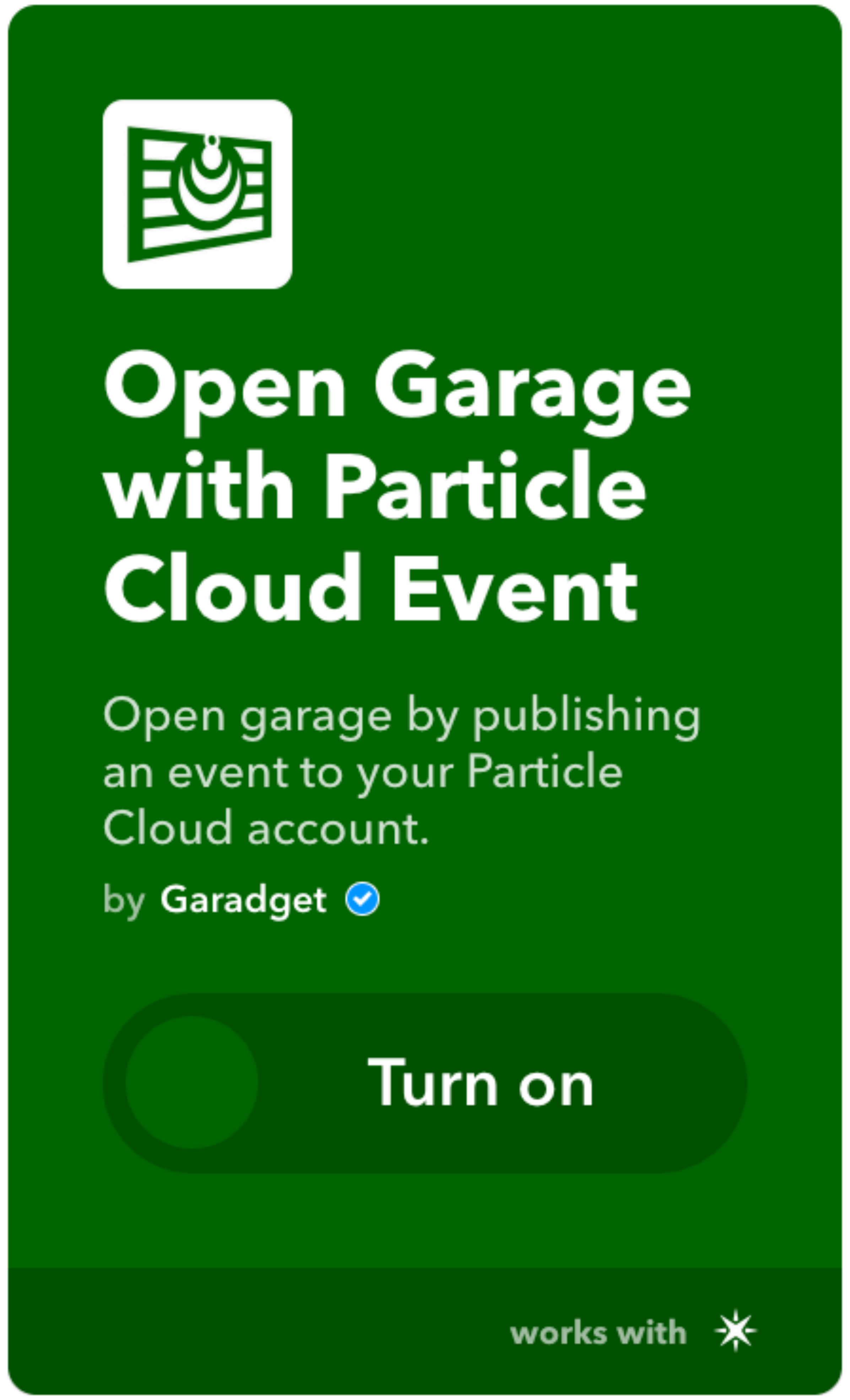
This Trigger retrieves today's current weather report at the time you specify. NOTE: Pollen count available only in the USA.

Time of day

10 AM

45 Minutes

Publish an event



Open Garage with Particle Cloud Event

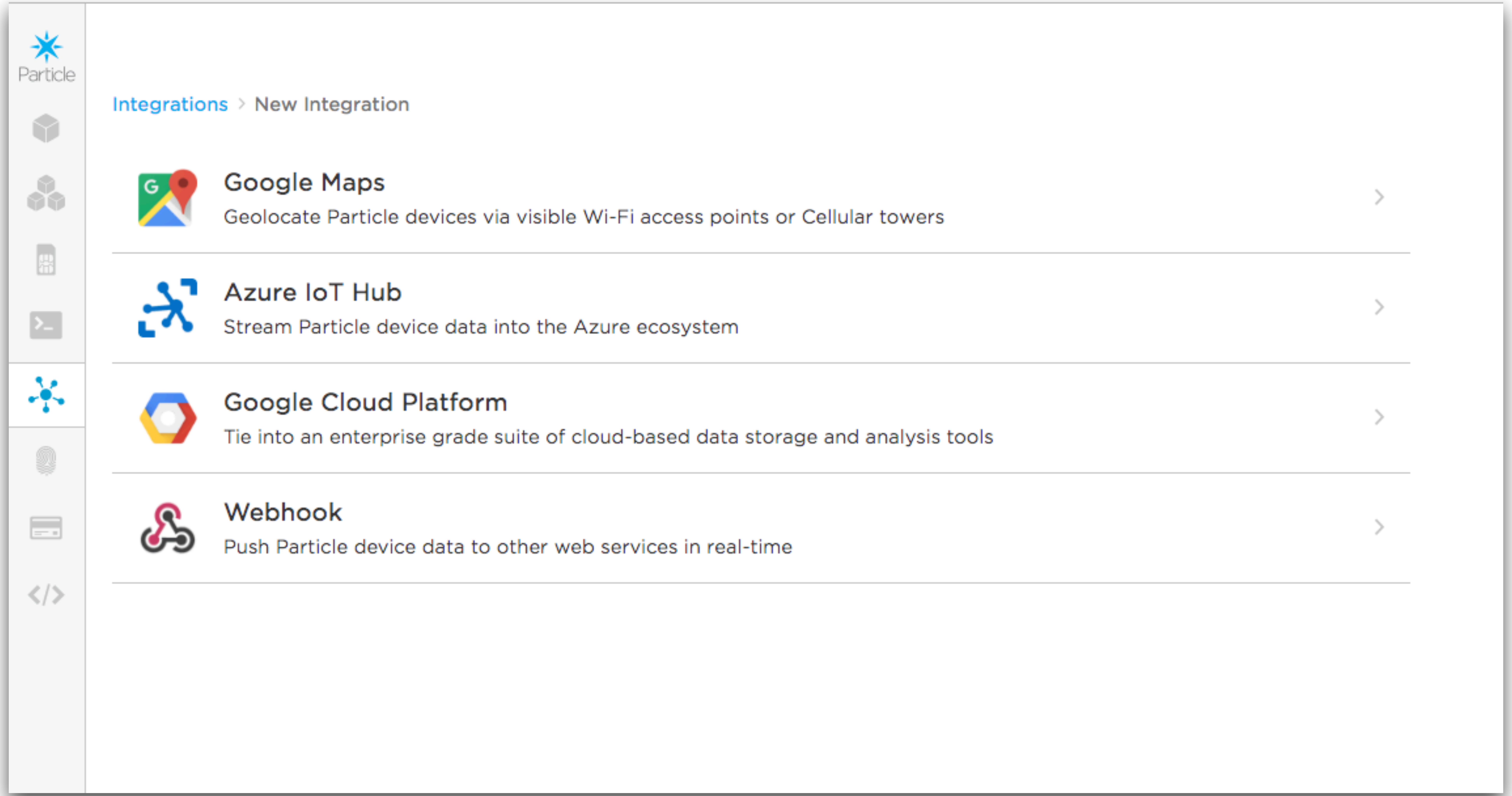
Open garage by publishing an event to your Particle Cloud account.

by Garadget

Turn on

works with

INTEGRATIONS – WEBHOOKS



Integrations > New Integration

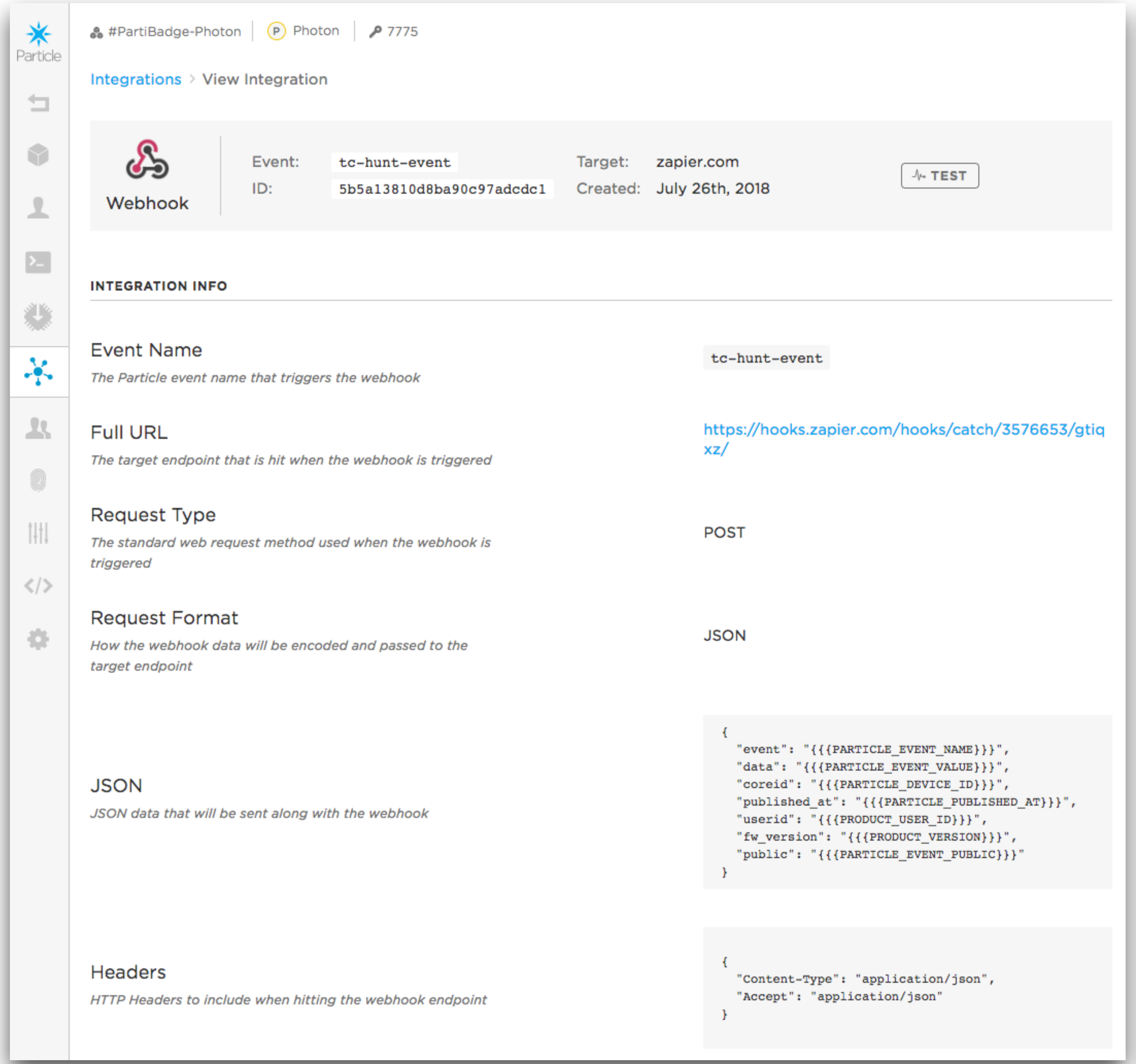
Google Maps
Geolocate Particle devices via visible Wi-Fi access points or Cellular towers

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

INTEGRATIONS – WEBHOOKS



Integrations › View Integration

Webhook

Event: tc-hunt-event
ID: 5b5a13810d8ba90c97adcdc1
Target: zapier.com
Created: July 26th, 2018

INTEGRATION INFO

Event Name
The Particle event name that triggers the webhook
tc-hunt-event

Full URL
The target endpoint that is hit when the webhook is triggered
https://hooks.zapier.com/hooks/catch/3576653/gtiqxz/

Request Type
The standard web request method used when the webhook is triggered
POST

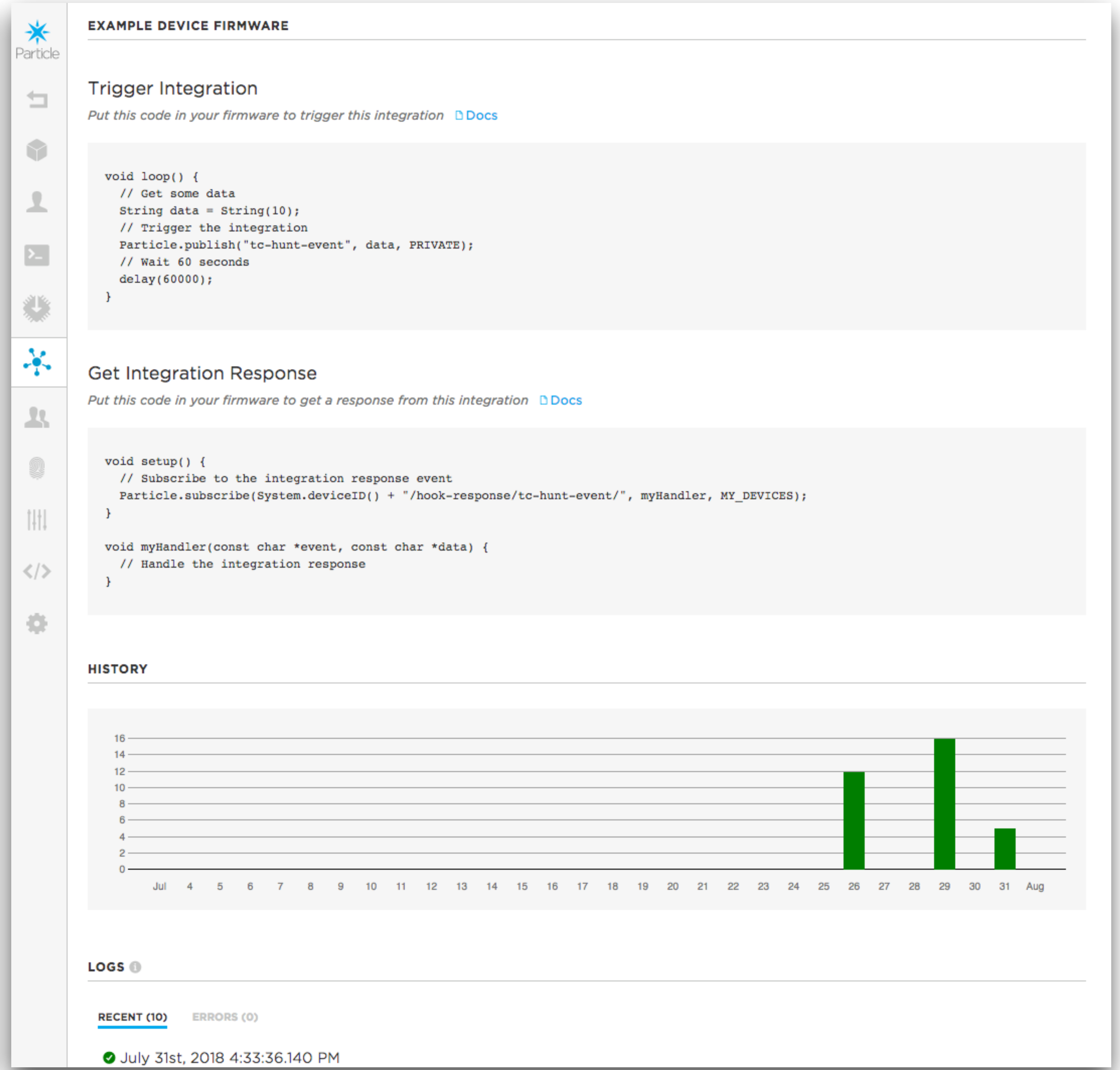
Request Format
How the webhook data will be encoded and passed to the target endpoint
JSON

JSON
JSON data that will be sent along with the webhook

```
{
  "event": "{{{PARTICLE_EVENT_NAME}}}",
  "data": "{{{PARTICLE_EVENT_VALUE}}}",
  "coreid": "{{{PARTICLE_DEVICE_ID}}}",
  "published_at": "{{{PARTICLE_PUBLISHED_AT}}}",
  "userid": "{{{PRODUCT_USER_ID}}}",
  "fw_version": "{{{PRODUCT_VERSION}}}",
  "public": "{{{PARTICLE_EVENT_PUBLIC}}}"
}
```

Headers
HTTP Headers to include when hitting the webhook endpoint

```
{
  "Content-Type": "application/json",
  "Accept": "application/json"
}
```


Particle
16
14
12
10
8
6
4
2
0
Jul 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 Aug

EXAMPLE DEVICE FIRMWARE

Trigger Integration
Put this code in your firmware to trigger this integration Docs

```
void loop() {
  // Get some data
  String data = String(10);
  // Trigger the integration
  Particle.publish("tc-hunt-event", data, PRIVATE);
  // Wait 60 seconds
  delay(60000);
}
```

Get Integration Response
Put this code in your firmware to get a response from this integration Docs

```
void setup() {
  // Subscribe to the integration response event
  Particle.subscribe(System.deviceID() + "/hook-response/tc-hunt-event/", myHandler, MY_DEVICES);
}

void myHandler(const char *event, const char *data) {
  // Handle the integration response
}
```

HISTORY

LOGS

RECENT (10) ERRORS (0)

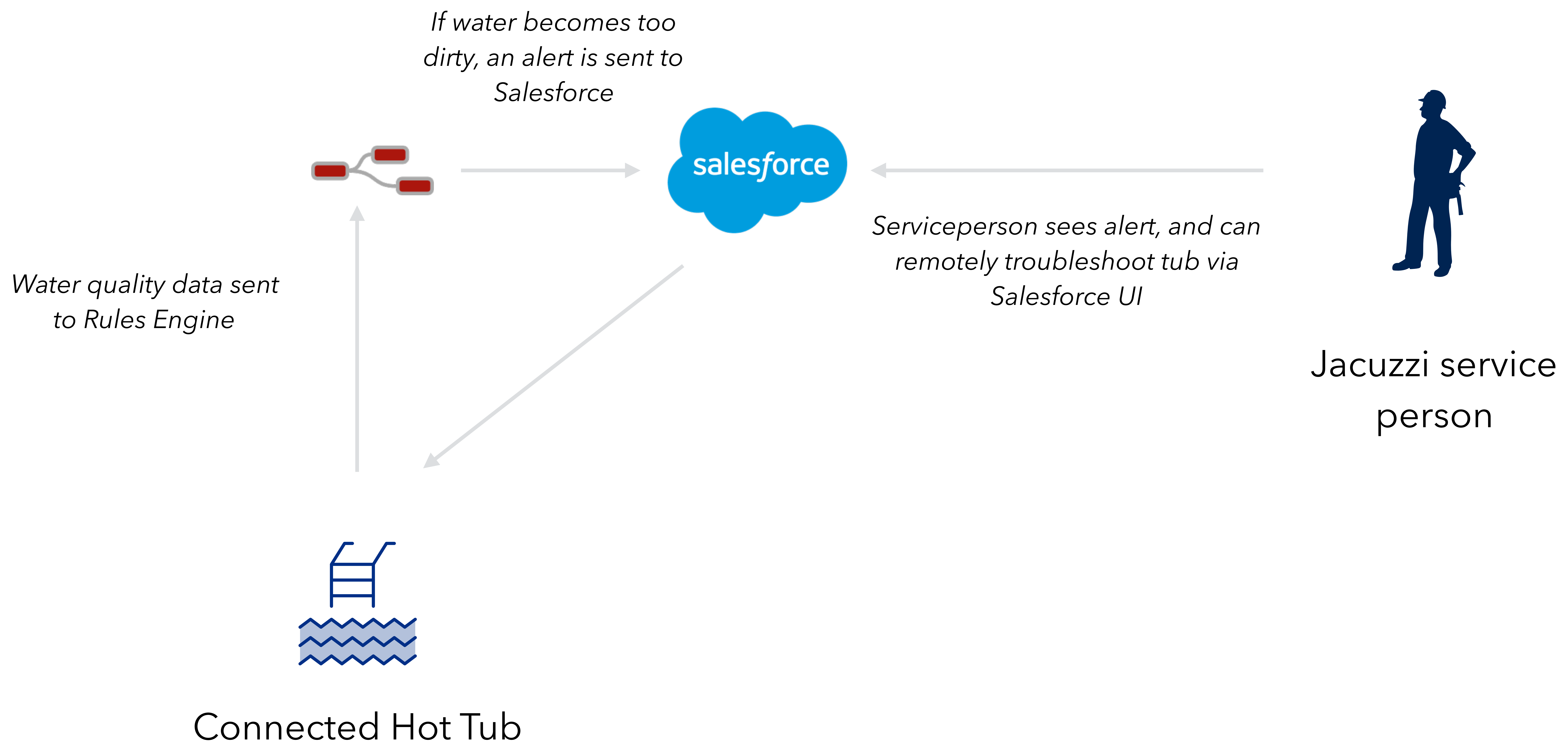
July 31st, 2018 4:33:36.140 PM

USE CASE: INTEGRATING 3RD PARTY TOOLS AND SERVICES

Jacuzzi is a well-known hot tub manufacturer that sells spas to thousands of consumers and hotels each year.

Their Particle-powered connected tub allows service people to remotely monitor and troubleshoot hot tubs using an interface presented to them in Salesforce Service Cloud.

USE CASE: INTEGRATING 3RD PARTY TOOLS AND SERVICES

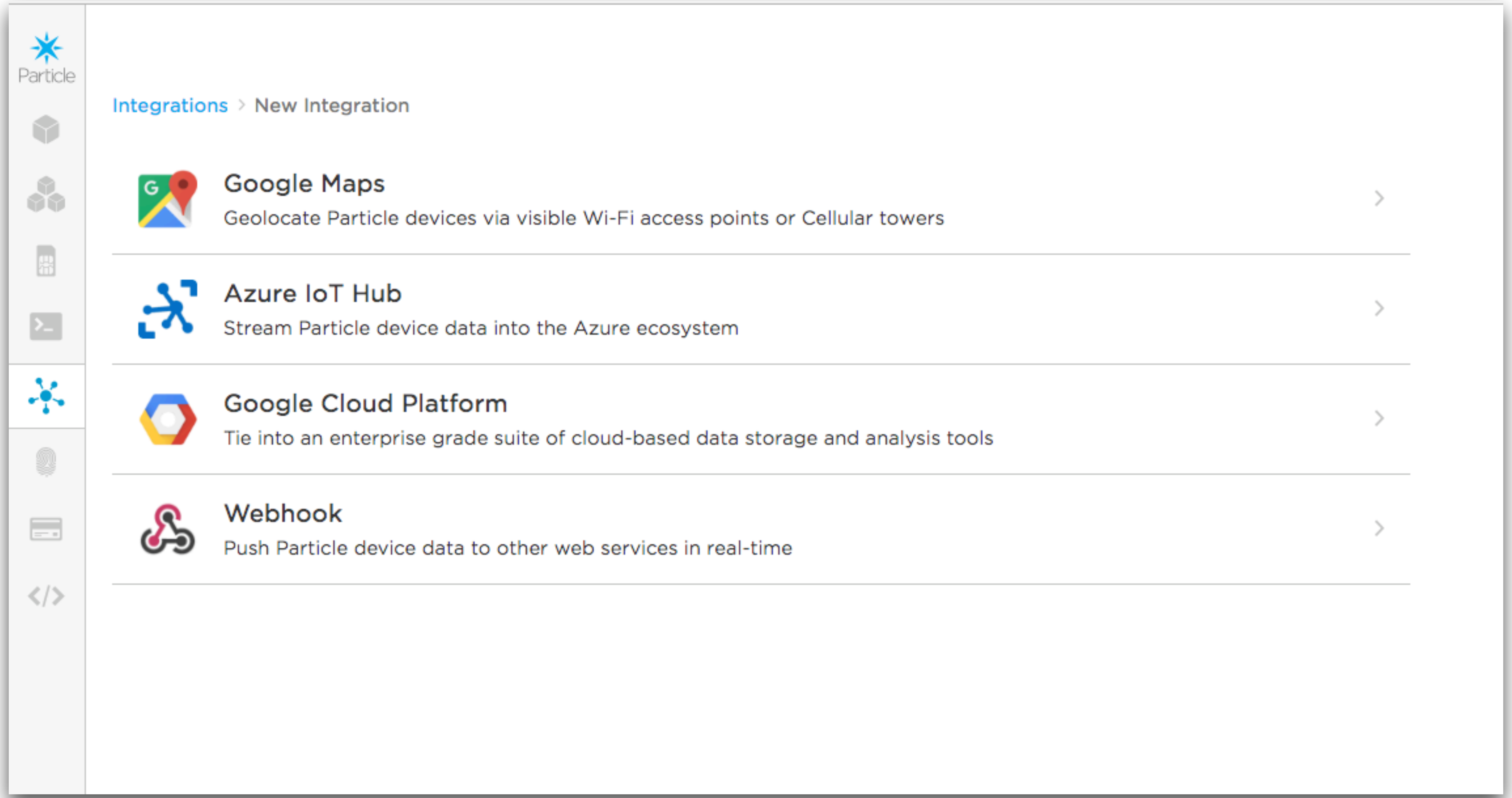


If water becomes too dirty, an alert is sent to Salesforce
salesforce
Serviceperson sees alert, and can remotely troubleshoot tub via Salesforce UI
Jacuzzi service person
Water quality data sent to Rules Engine
Connected Hot Tub

USE CASE: INTEGRATING 3RD PARTY TOOLS AND SERVICES

CRMs	ERPs	Value-Add Services
salesforce	SAP	stripe
ZOHO	ORACLE	IBM Watson
	NETSUITE	Google Maps

INTEGRATIONS – AZURE IOT & GOOGLE CLOUD PLATFORM



Integrations › New Integration

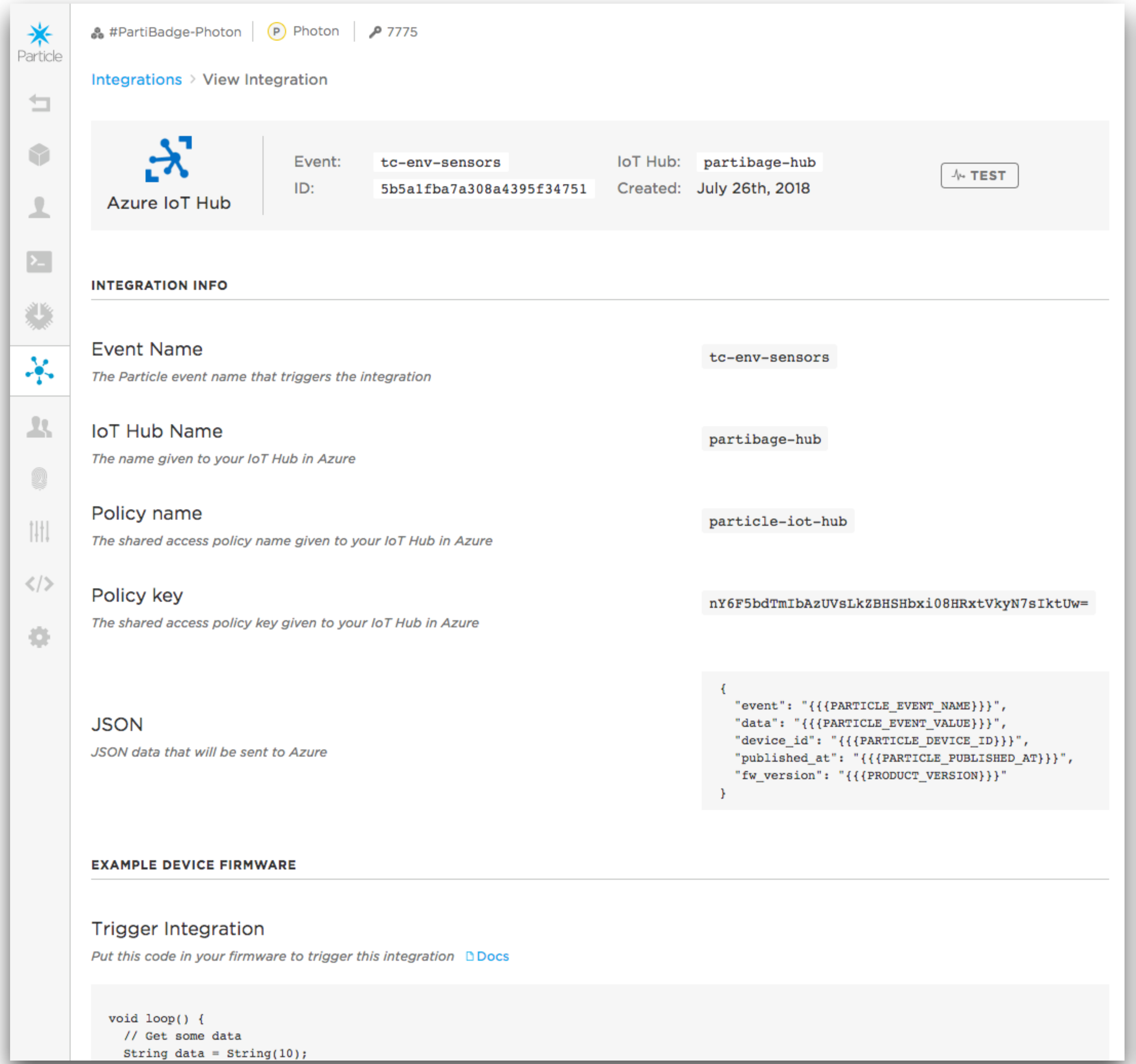
Google Maps
Geolocate Particle devices via visible Wi-Fi access points or Cellular towers

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

INTEGRATIONS – AZURE IOT & GOOGLE CLOUD PLATFORM



Integrations > View Integration

Azure IoT Hub

Event: `tc-env-sensors`
ID: `5b5a1fba7a308a4395f34751`
IoT Hub: `partibage-hub`
Created: July 26th, 2018

INTEGRATION INFO

Event Name *The Particle event name that triggers the integration*	`tc-env-sensors`
IoT Hub Name *The name given to your IoT Hub in Azure*	`partibage-hub`
Policy name *The shared access policy name given to your IoT Hub in Azure*	`particle-iot-hub`
Policy key *The shared access policy key given to your IoT Hub in Azure*	`nY6F5bdTmIbAzUVsLkZBHSHbxi08HRxtVkyN7sIktUw=`

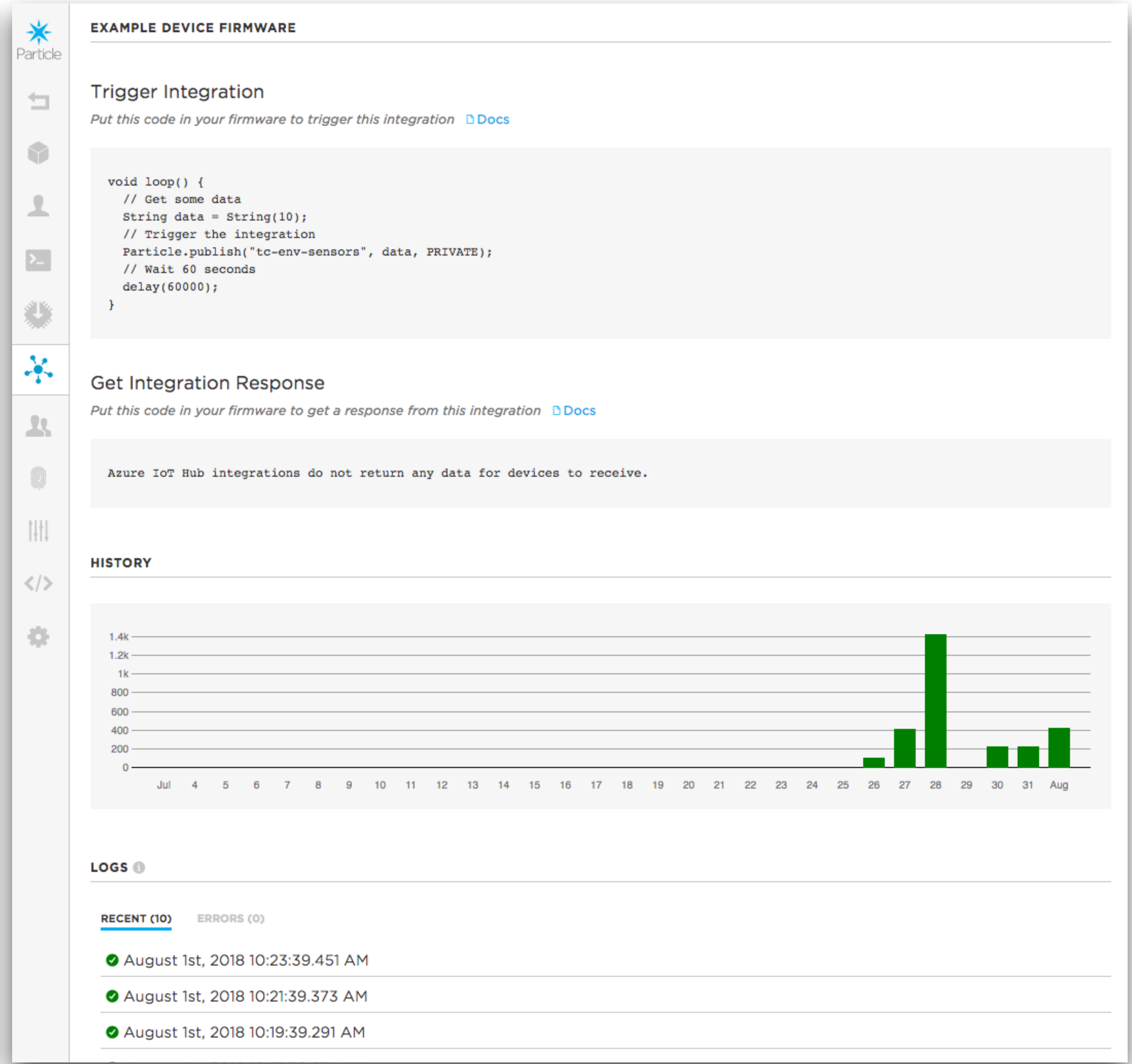
JSON
JSON data that will be sent to Azure

```
{
  "event": "{{{PARTICLE_EVENT_NAME}}}",
  "data": "{{{PARTICLE_EVENT_VALUE}}}",
  "device_id": "{{{PARTICLE_DEVICE_ID}}}",
  "published_at": "{{{PARTICLE_PUBLISHED_AT}}}",
  "fw_version": "{{{PRODUCT_VERSION}}}"
}
```

EXAMPLE DEVICE FIRMWARE

Trigger Integration
Put this code in your firmware to trigger this integration Docs

```
void loop() {
  // Get some data
  String data = String(10);
```



EXAMPLE DEVICE FIRMWARE

Trigger Integration
Put this code in your firmware to trigger this integration Docs

```
void loop() {
  // Get some data
  String data = String(10);
  // Trigger the integration
  Particle.publish("tc-env-sensors", data, PRIVATE);
  // Wait 60 seconds
  delay(60000);
}
```

Get Integration Response
Put this code in your firmware to get a response from this integration Docs

```
Azure IoT Hub integrations do not return any data for devices to receive.
```

HISTORY

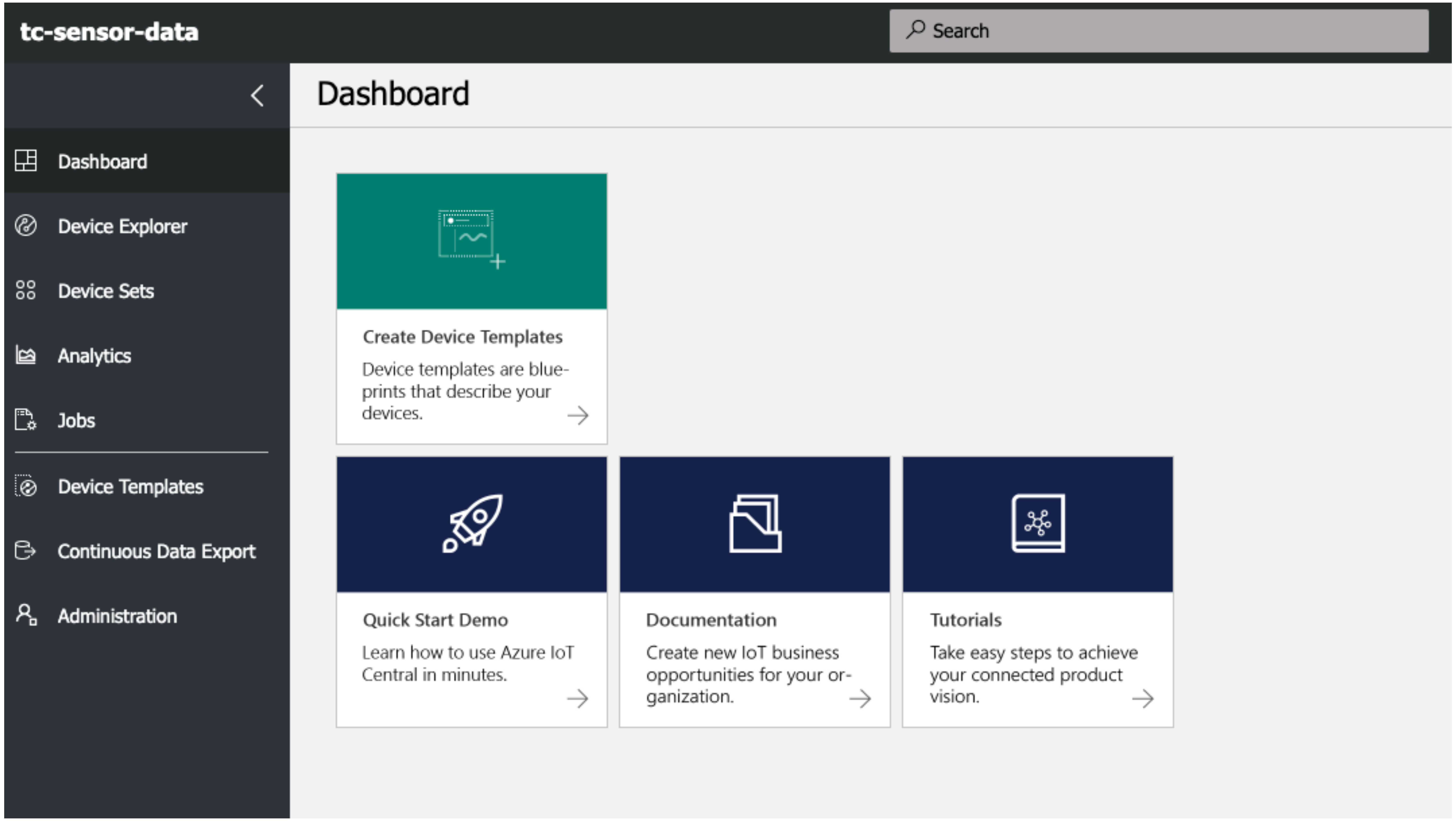
1.4k
1.2k
1k
800
600
400
200
0
Jul 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 Aug

LOGS

RECENT (10) ERRORS (0)

```
✓ August 1st, 2018 10:23:39.451 AM
✓ August 1st, 2018 10:21:39.373 AM
✓ August 1st, 2018 10:19:39.291 AM
```

AZURE IOT CENTRAL – CREATING DEVICE TEMPLATES



Dashboard

Create Device Templates

Device templates are blue-prints that describe your devices.

Quick Start Demo

Learn how to use Azure IoT Central in minutes.

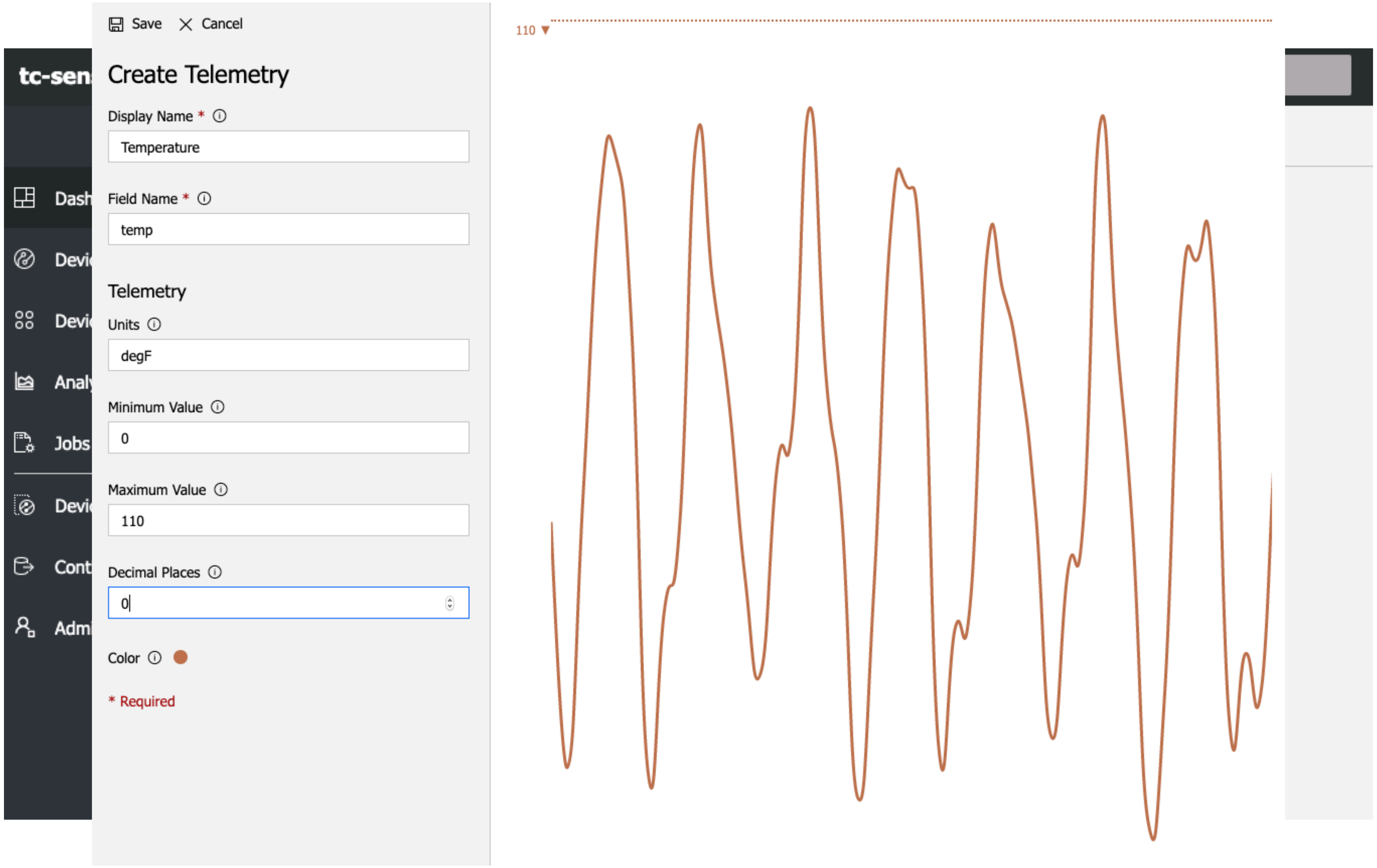
Documentation

Create new IoT business opportunities for your or-ganization.

Tutorials

Take easy steps to achieve your connected product vision.

AZURE IOT CENTRAL – CREATING DEVICE TEMPLATES



Create Telemetry

Display Name * Temperature

Field Name * temp

Telemetry

Units degF

Minimum Value 0

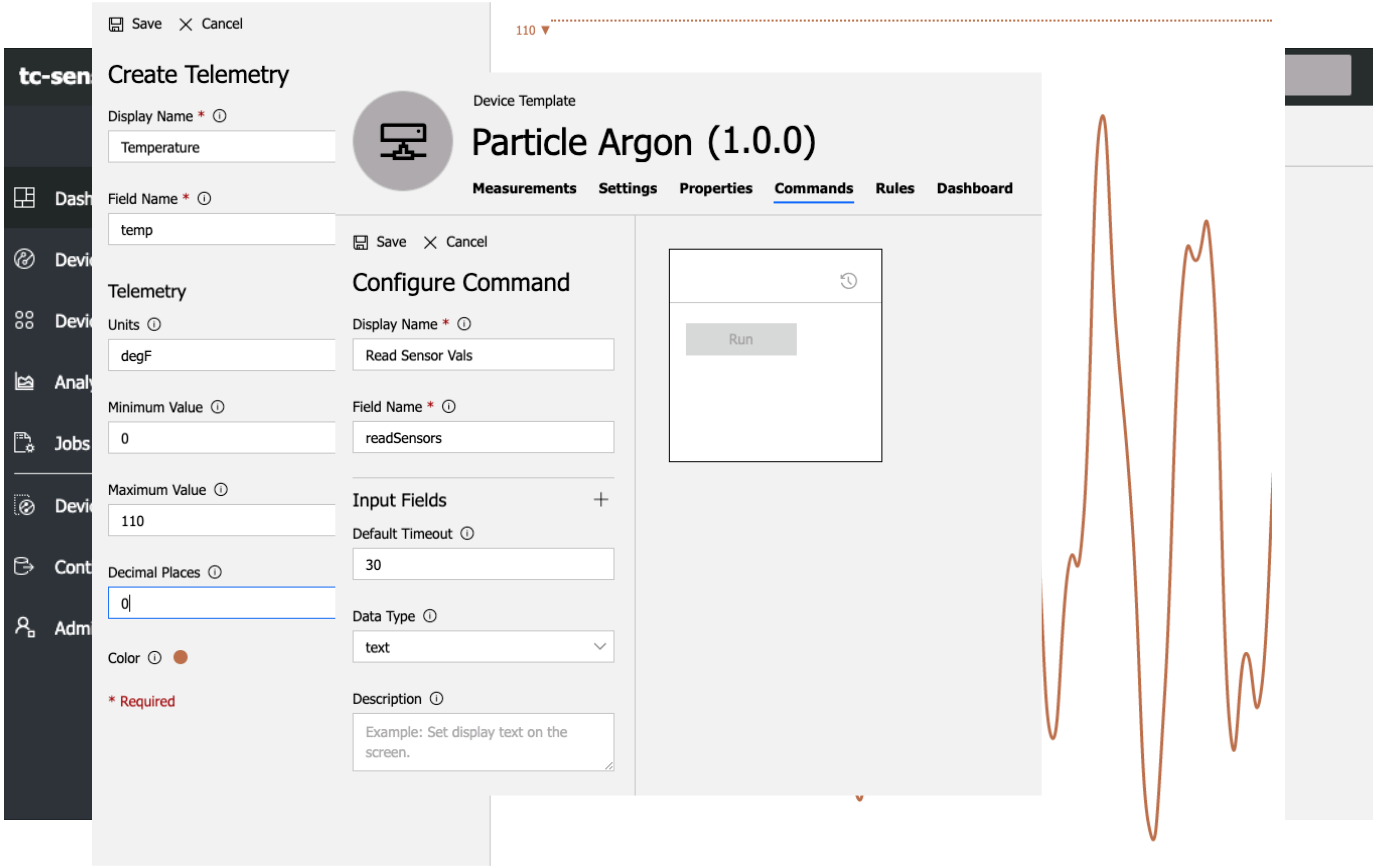
Maximum Value 110

Decimal Places 0

Color ●

* Required

AZURE IOT CENTRAL – CREATING DEVICE TEMPLATES



Save Cancel

Create Telemetry

Display Name * Temperature

Field Name * temp

Telemetry

Units degF

Minimum Value 0

Maximum Value 110

Decimal Places 0

Color ●

* Required

Device Template

Particle Argon (1.0.0)

Measurements Settings Properties Commands Rules Dashboard

Save Cancel

Configure Command

Display Name * Read Sensor Vals

Field Name * readSensors

Input Fields

Default Timeout 30

Data Type text

Description Example: Set display text on the screen.

USING WEBHOOKS AND INTEGRATIONS

DEMO

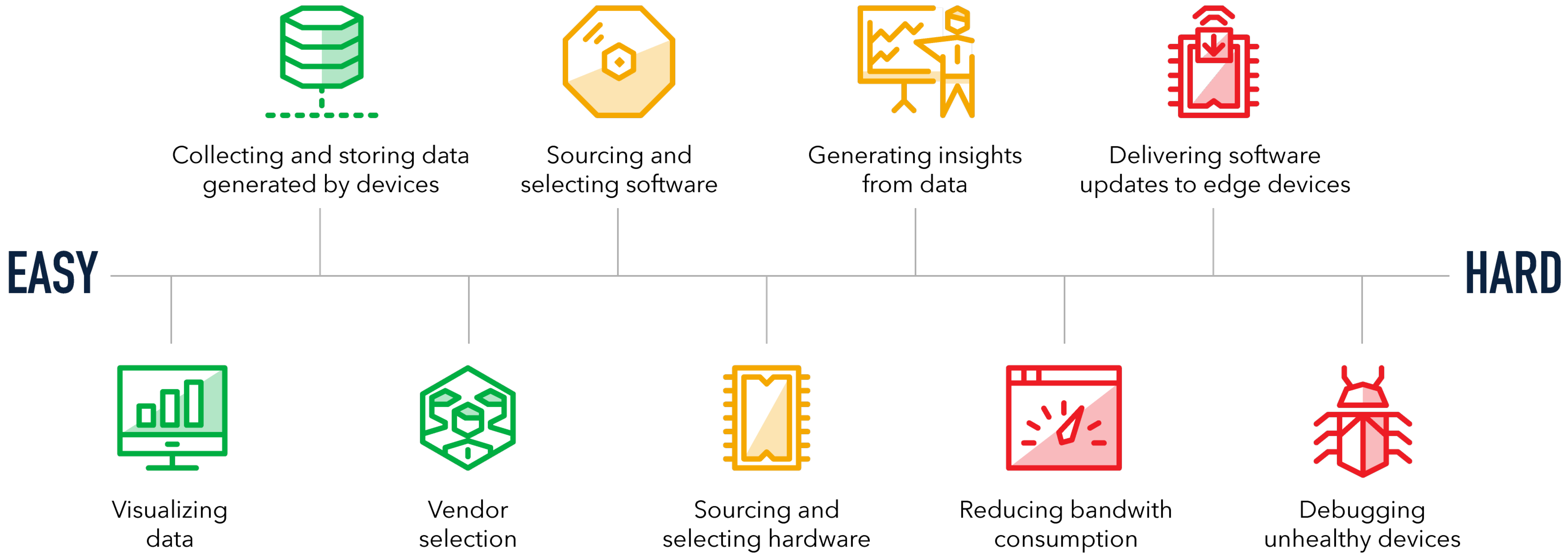
GETTING STARTED WITH THE CLI

USING WEBHOOKS AND INTEGRATIONS

FLEET MANAGEMENT & DIAGNOSTICS

ON-DEVICE DEBUGGING

WHAT IS ACTUALLY HARD ABOUT IOT IS NOT WHAT YOU MIGHT THINK…


EASY
Visualizing data
Collecting and storing data generated by devices
Vendor selection
Sourcing and selecting software
Sourcing and selecting hardware
Generating insights from data
Reducing bandwith consumption
Delivering software updates to edge devices
Debugging unhealthy devices
HARD

Data from 2019 State of IoT report conducted by Particle

THE MOST CHALLENGING REPORTED IOT TASK IS REMOTELY MANAGING DEVICE HEALTH

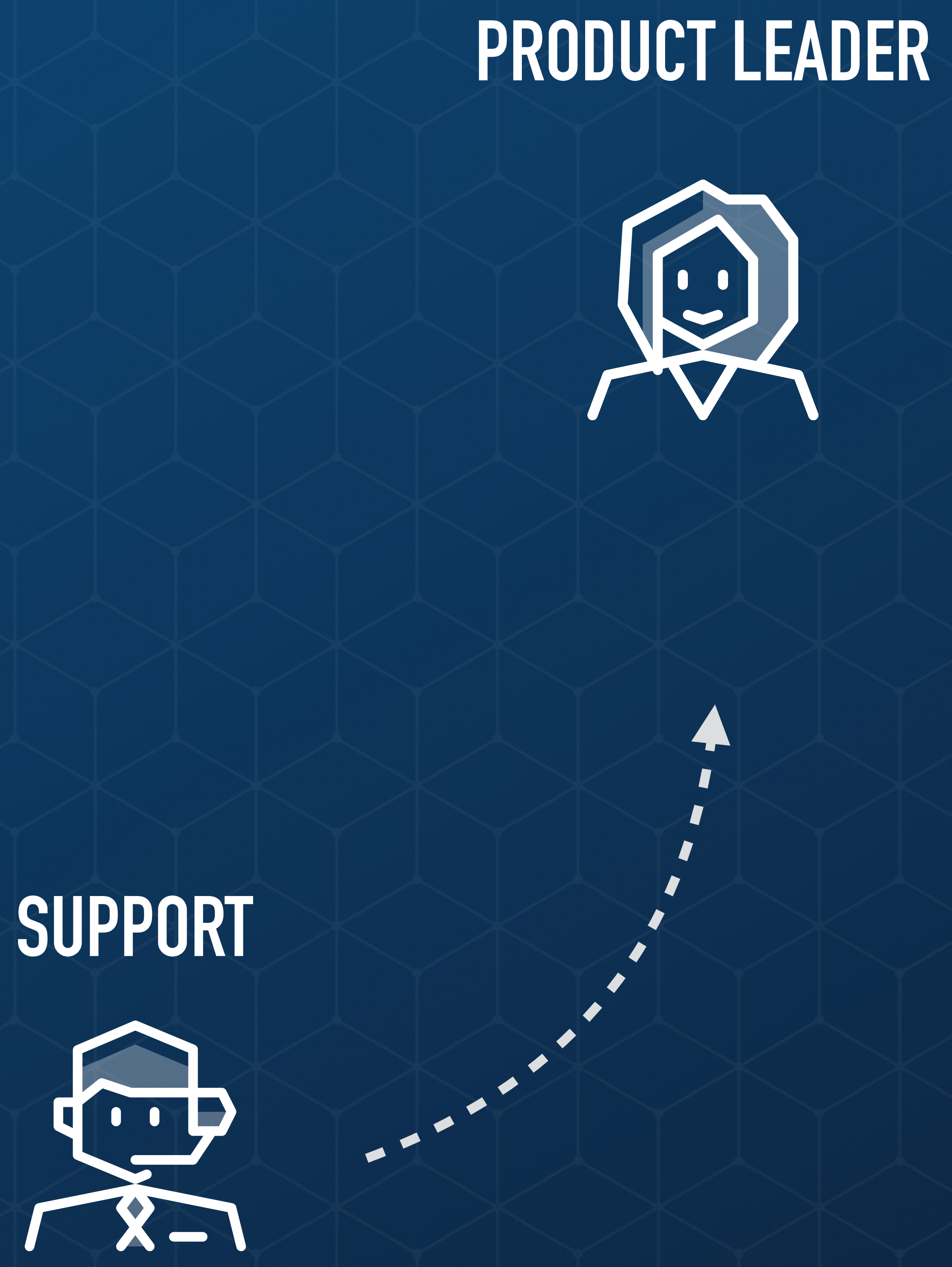
55%

of respondents listed debugging unhealthy devices as difficult.

WHAT HAPPENS TODAY, FOR MOST COMPANIES WITH CONNECTED PRODUCTS

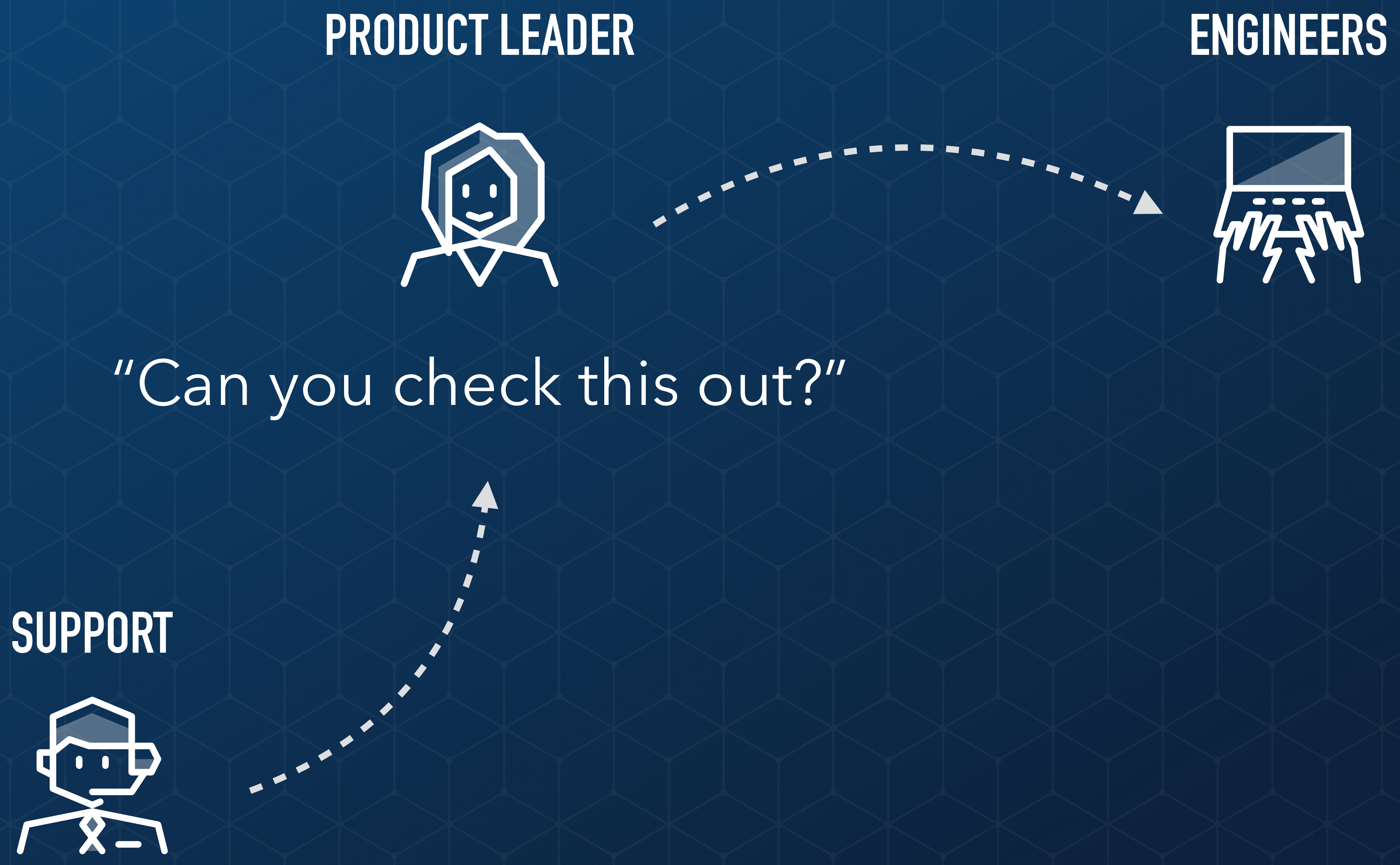
SUPPORT

WHAT HAPPENS TODAY, FOR MOST COMPANIES WITH CONNECTED PRODUCTS



"I think there's a problem"

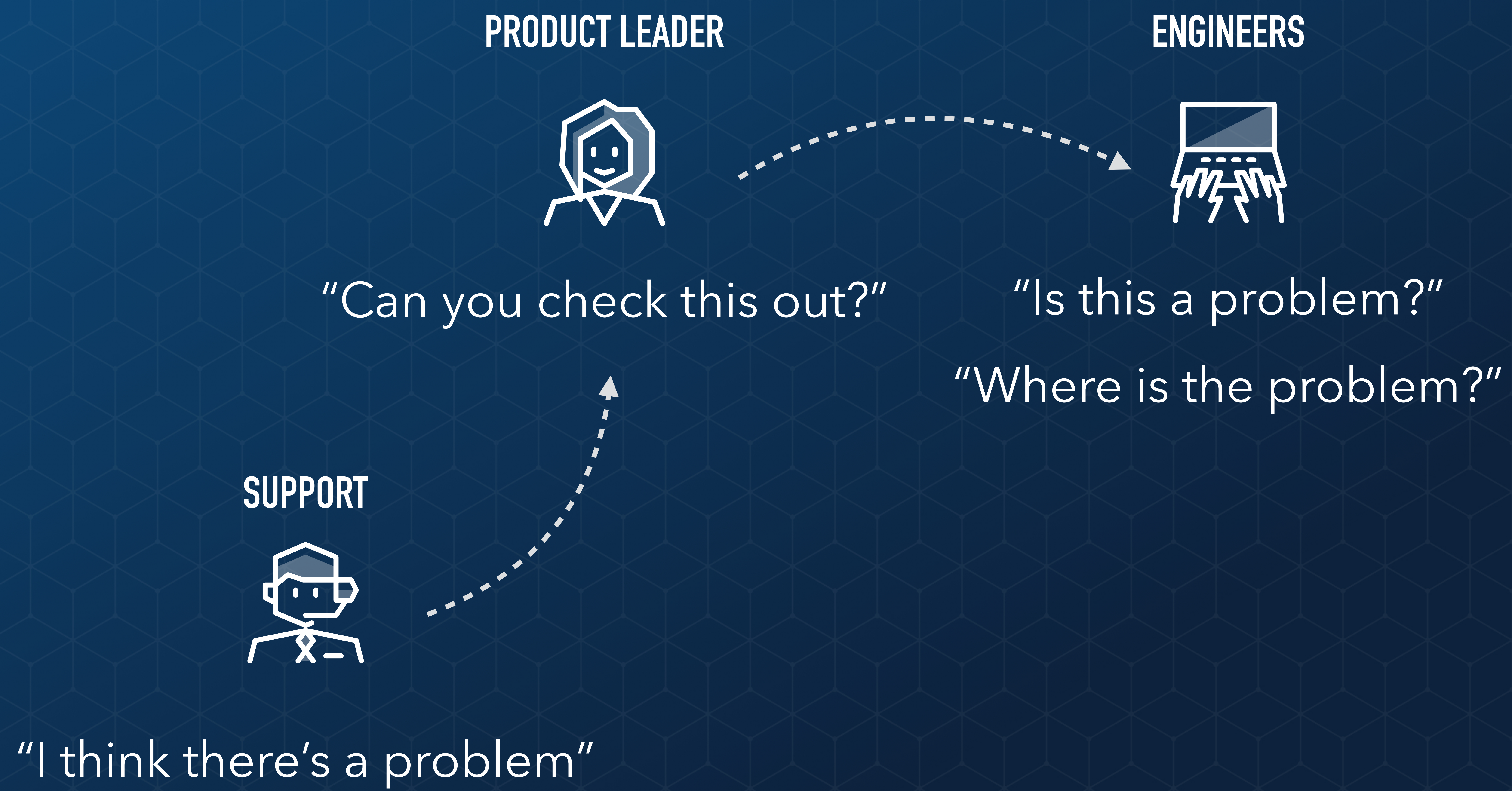
WHAT HAPPENS TODAY, FOR MOST COMPANIES WITH CONNECTED PRODUCTS



"Can you check this out?"

"I think there's a problem"

WHAT HAPPENS TODAY, FOR MOST COMPANIES WITH CONNECTED PRODUCTS



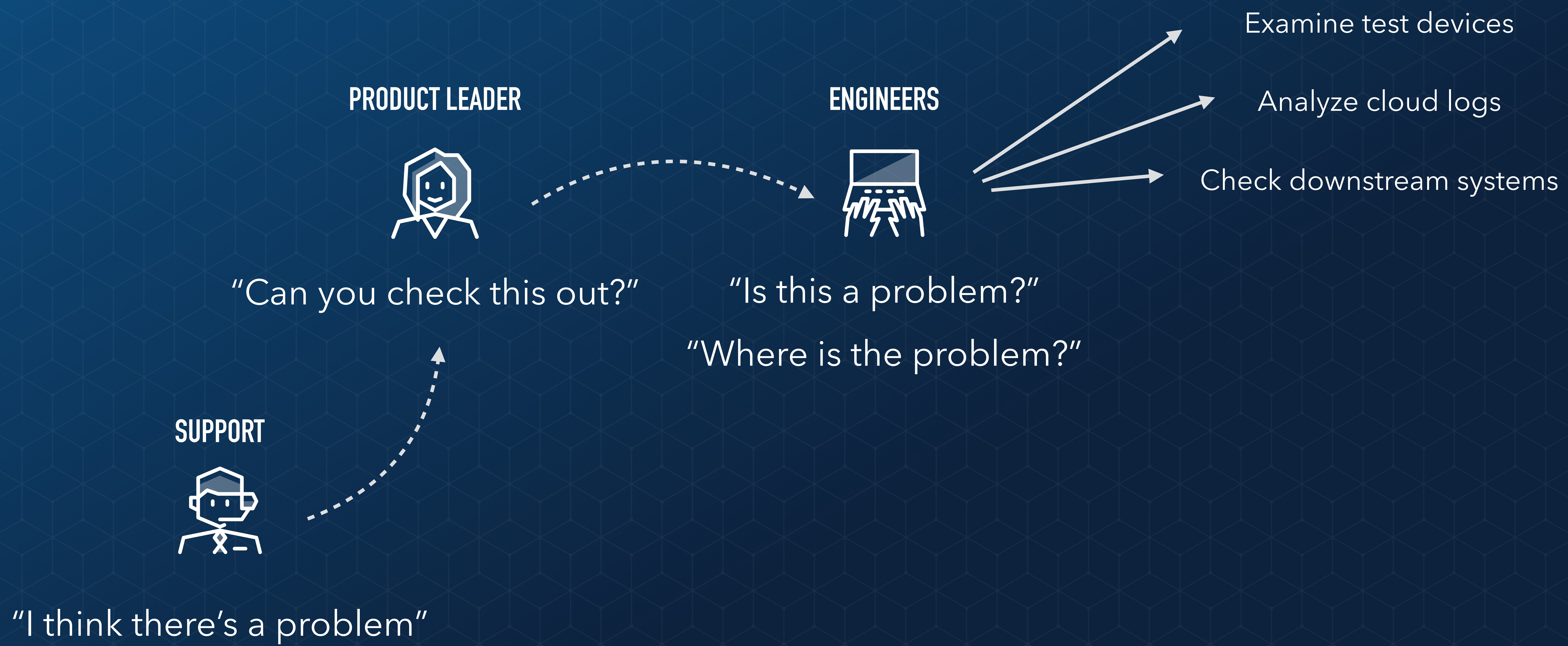
“Can you check this out?”

“Is this a problem?”

“Where is the problem?”

“I think there’s a problem”

WHAT HAPPENS TODAY, FOR MOST COMPANIES WITH CONNECTED PRODUCTS



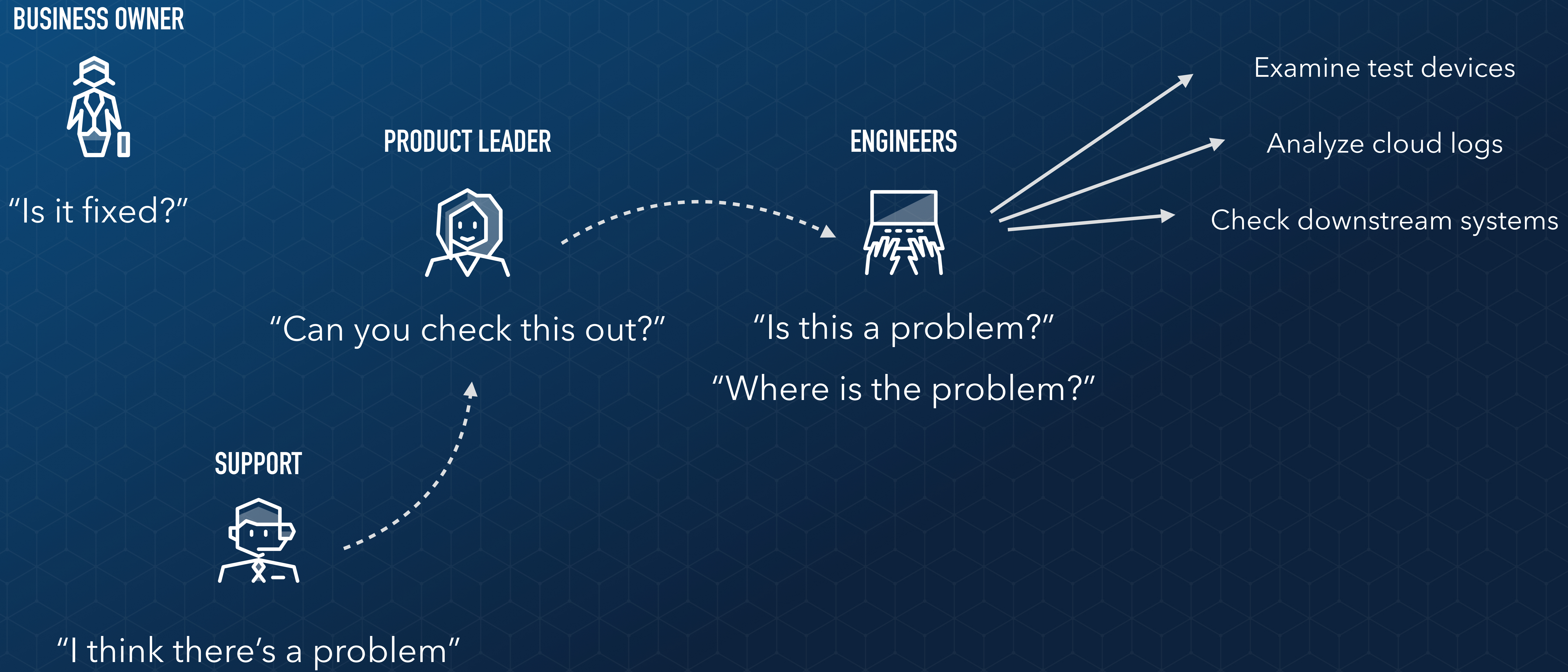
“I think there’s a problem”

“Can you check this out?”

“Is this a problem?”

“Where is the problem?”

WHAT HAPPENS TODAY, FOR MOST COMPANIES WITH CONNECTED PRODUCTS



BUSINESS OWNER

"Is it fixed?"

SUPPORT

"I think there's a problem"

PRODUCT LEADER

"Can you check this out?"

ENGINEERS

"Is this a problem?"

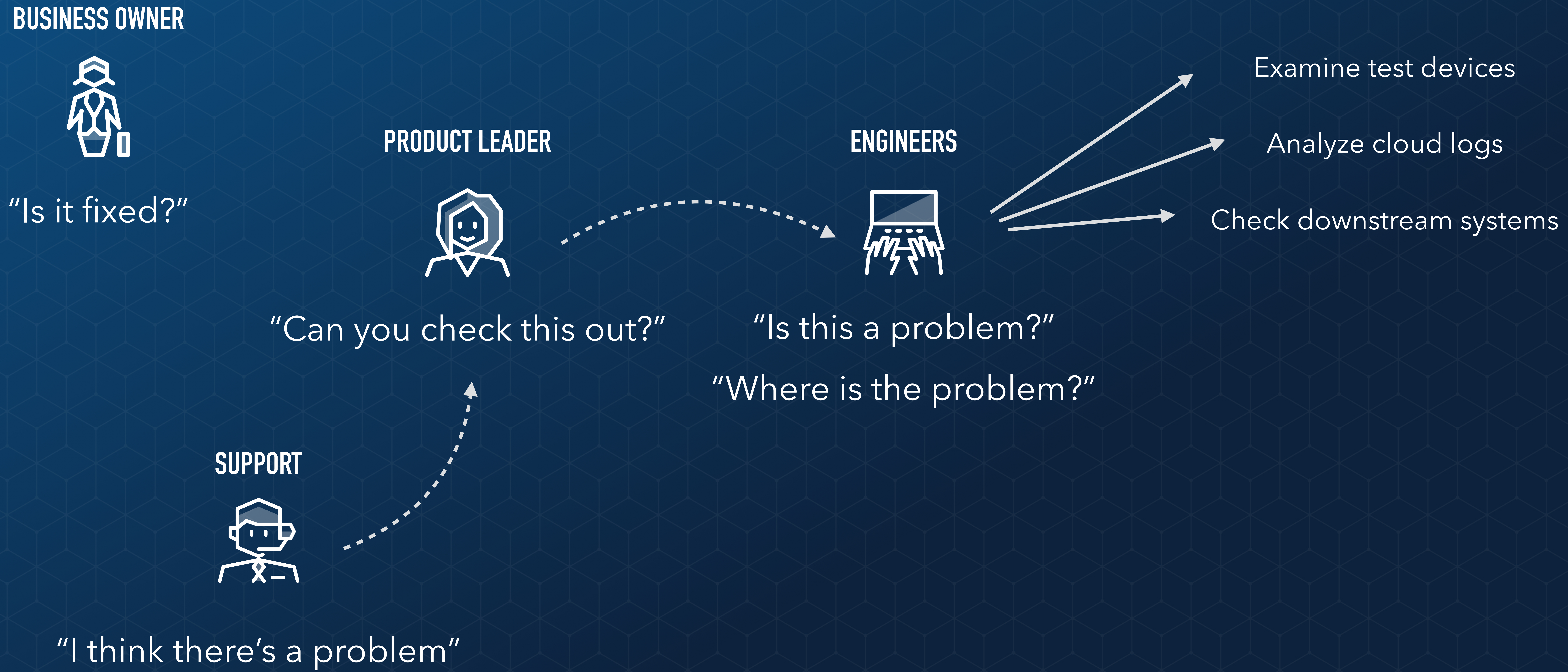
"Where is the problem?"

Examine test devices

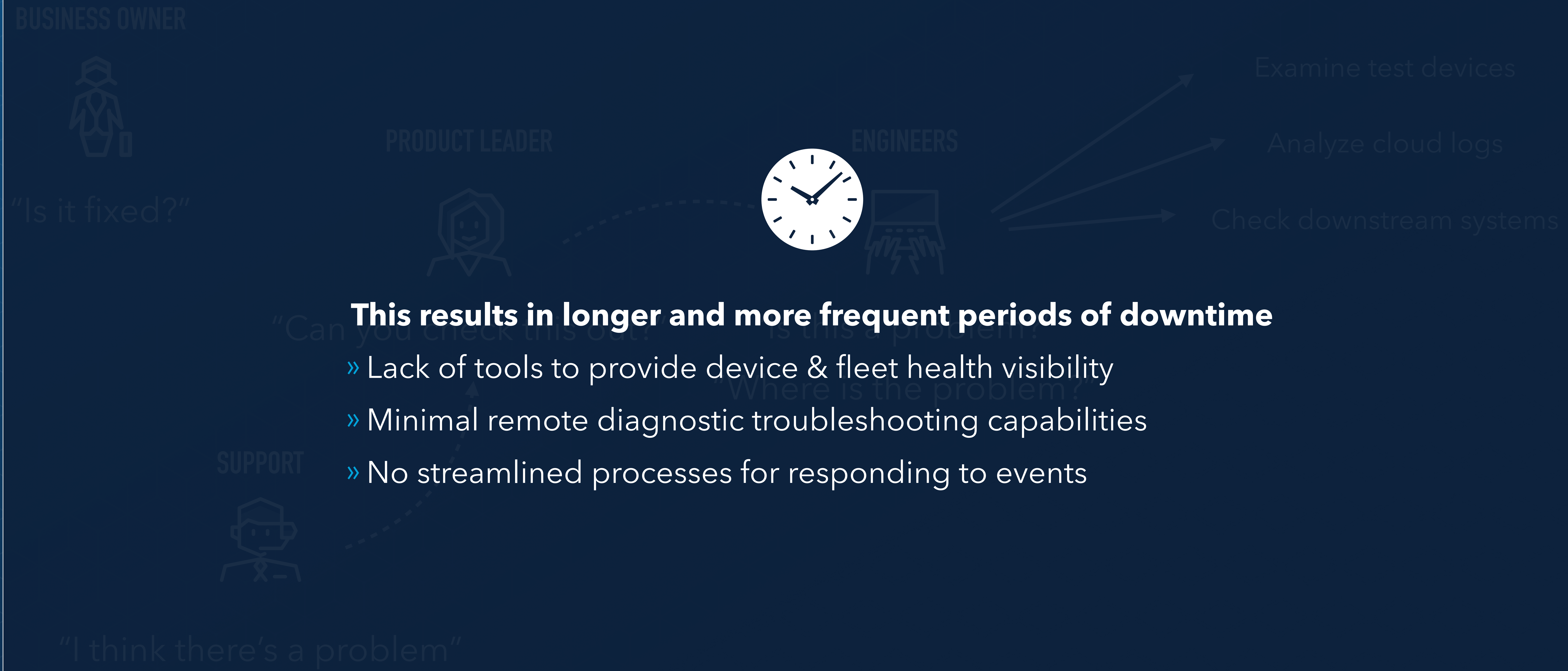
Analyze cloud logs

Check downstream systems

WHAT HAPPENS TODAY, FOR MOST COMPANIES WITH CONNECTED PRODUCTS



WHAT HAPPENS TODAY, FOR MOST COMPANIES WITH CONNECTED PRODUCTS



This results in longer and more frequent periods of downtime

» Lack of tools to provide device & fleet health visibility

» Minimal remote diagnostic troubleshooting capabilities

» No streamlined processes for responding to events

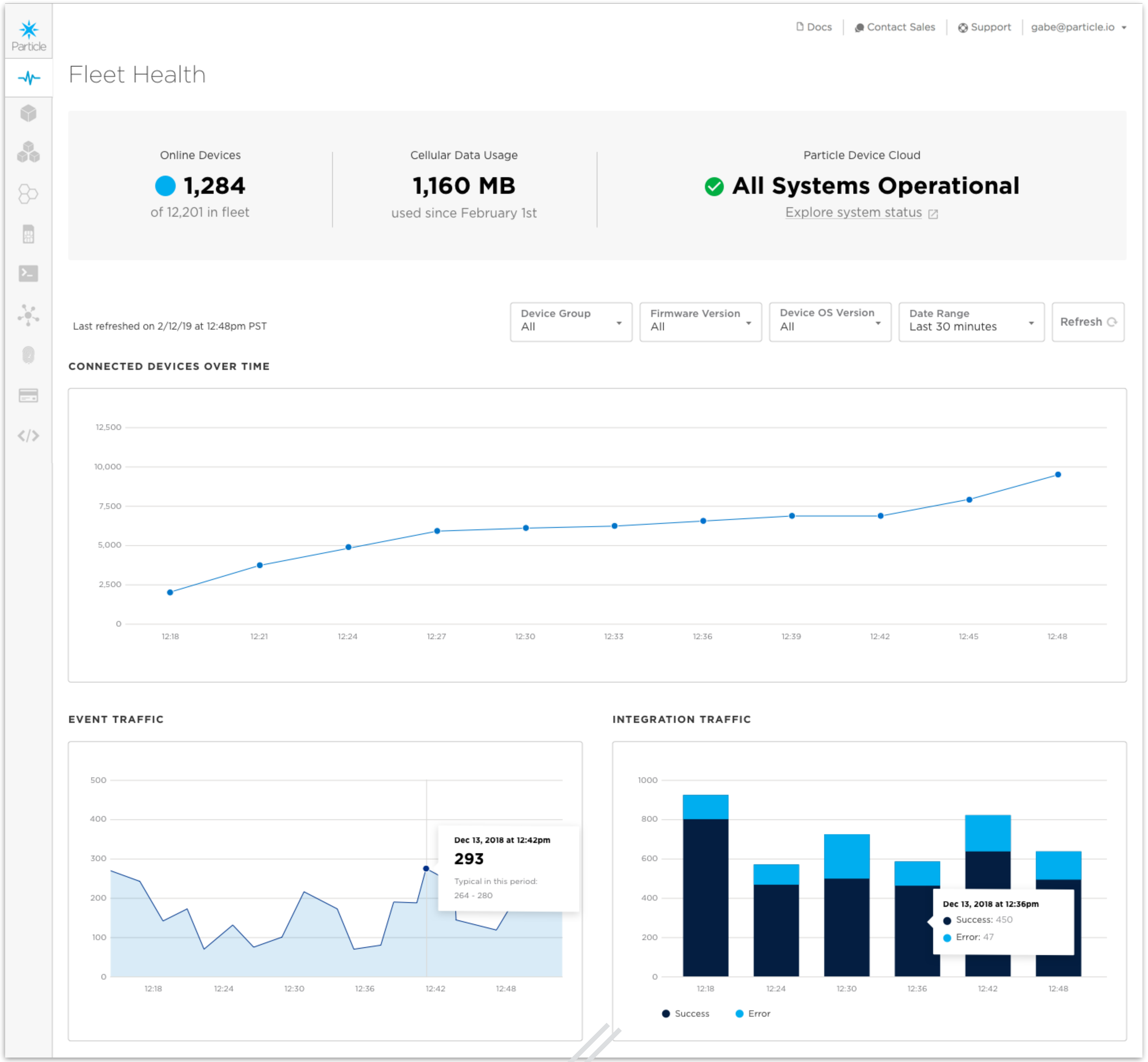
INTRODUCING DIAGNOSTICS FROM PARTICLE

PARTICLE DIAGNOSTICS

Fleet Health to quickly identify and respond to system-wide disruptions

Device Vitals for deep device-level visibility into connectivity health

FLEET HEALTH PROVIDES SYSTEM-WIDE VISIBILITY INTO YOUR IoT DEPLOYMENT



Fleet Health

Online Devices
● 1,284
of 12,201 in fleet

Cellular Data Usage
1,160 MB
used since February 1st

Particle Device Cloud
✓ All Systems Operational
Explore system status

Last refreshed on 2/12/19 at 12:48pm PST

Device Group: All
Firmware Version: All
Device OS Version: All
Date Range: Last 30 minutes

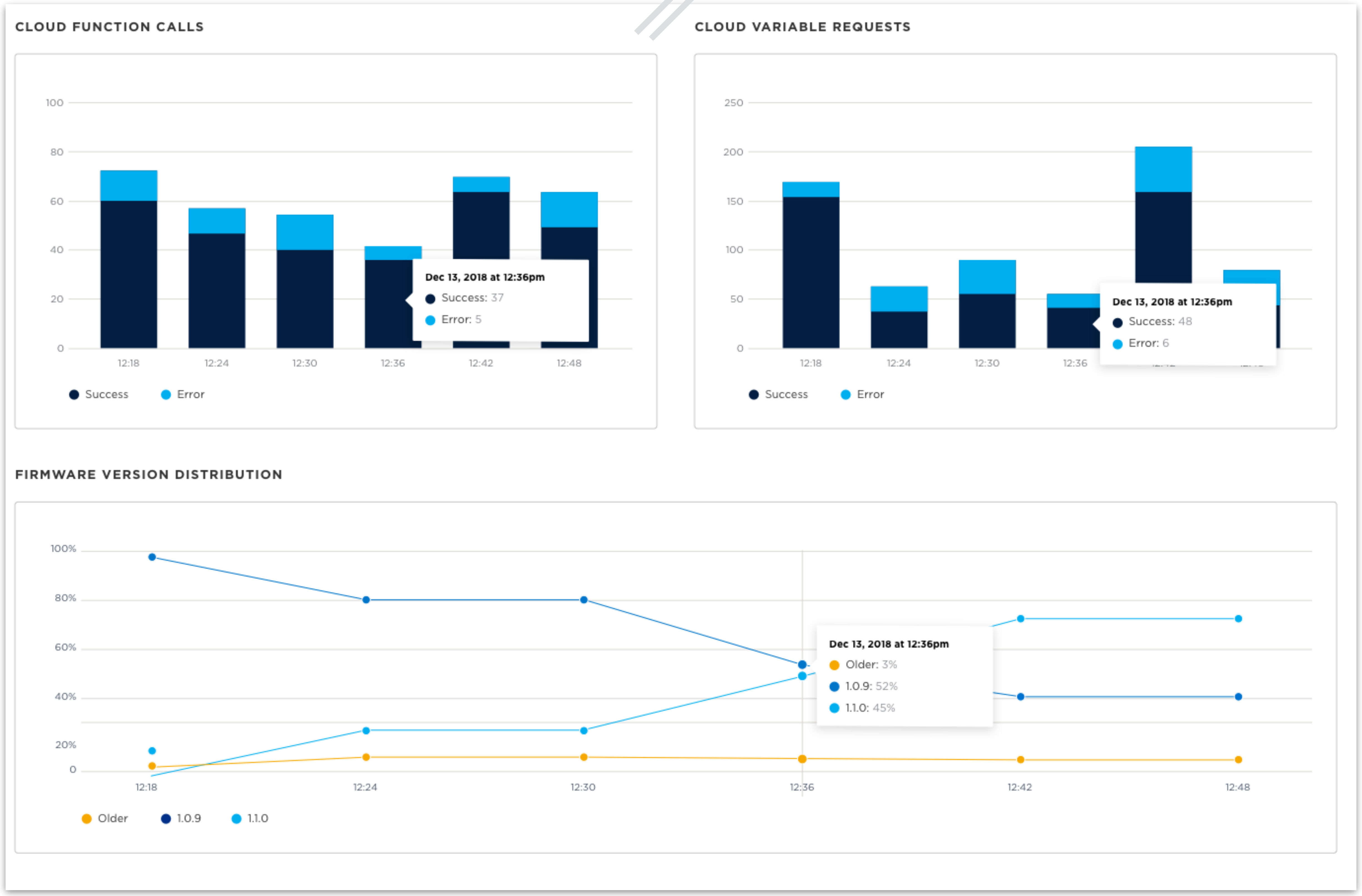
CONNECTED DEVICES OVER TIME

EVENT TRAFFIC

Dec 13, 2018 at 12:42pm
293
Typical in this period:
264 - 280

INTEGRATION TRAFFIC

Dec 13, 2018 at 12:36pm
● Success: 450
● Error: 47



CLOUD FUNCTION CALLS

Dec 13, 2018 at 12:36pm
● Success: 37
● Error: 5

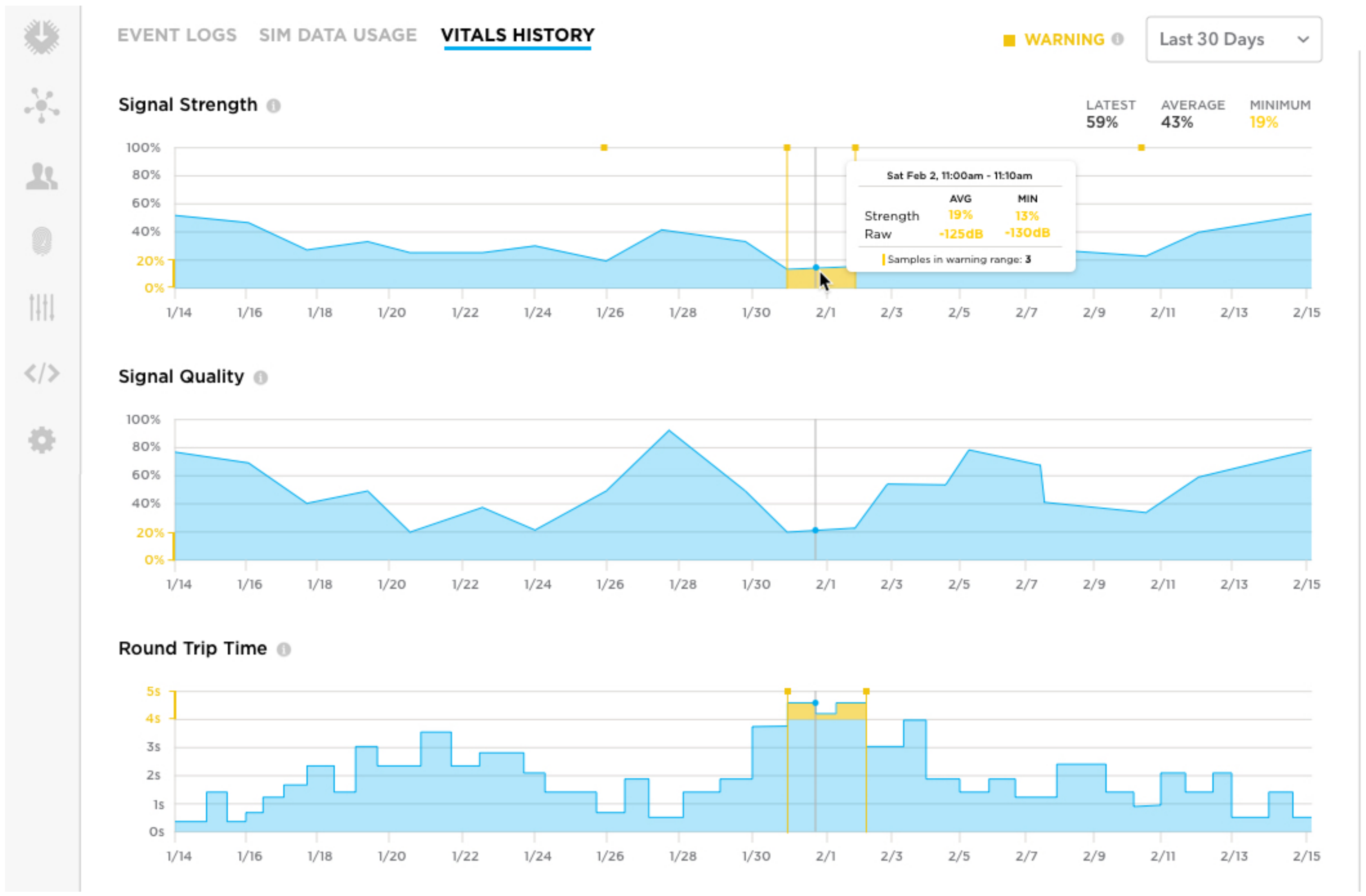
CLOUD VARIABLE REQUESTS

Dec 13, 2018 at 12:36pm
● Success: 48
● Error: 6

FIRMWARE VERSION DISTRIBUTION

Dec 13, 2018 at 12:36pm
● Older: 3%
● 1.0.9: 52%
● 1.1.0: 45%

DEVICE VITALS LET YOU ZOOM IN TO INDIVIDUAL DEVICE'S HEALTH



PARTICLE DIAGNOSTICS

DEMO

GETTING STARTED WITH THE CLI

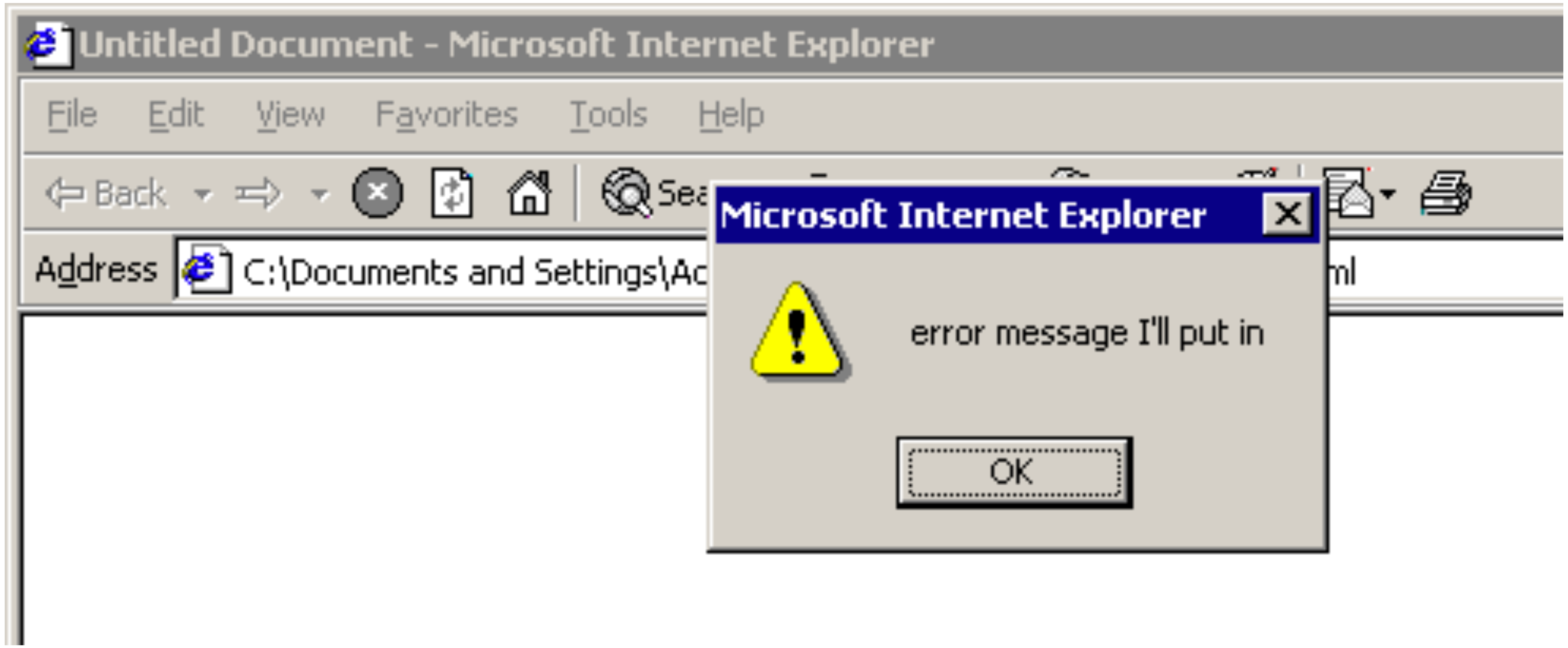
USING WEBHOOKS AND INTEGRATIONS

FLEET MANAGEMENT & DIAGNOSTICS

ON-DEVICE DEBUGGING

DEBUGGING ON THE WEB, CIRCA 1999

DEBUGGING ON THE WEB, CIRCA 1999


Untitled Document - Microsoft Internet Explorer
File
Edit
View
Favorites
Tools
Help
Back
Sea
Address
C:\Documents and Settings\Ac
ml
Microsoft Internet Explorer
OK

error message I'll put in

DEBUGGING ON THE WEB, CIRCA 1999

DEBUGGING ON THE WEB, CIRCA 1999

```
alert(“HERE”);

// Code that’s probably buggy

alert(“HERE 2”);

// Code that’s probably also buggy

alert(“WHAT ARE WEEKENDS ANYWAY? I LIVE IN THIS CUBICLE NOW.”);

// Code that may be buggy, but you don’t know because the app
// never seems to get this far.
```

DEBUGGING IOT APPS, CIRCA 2018

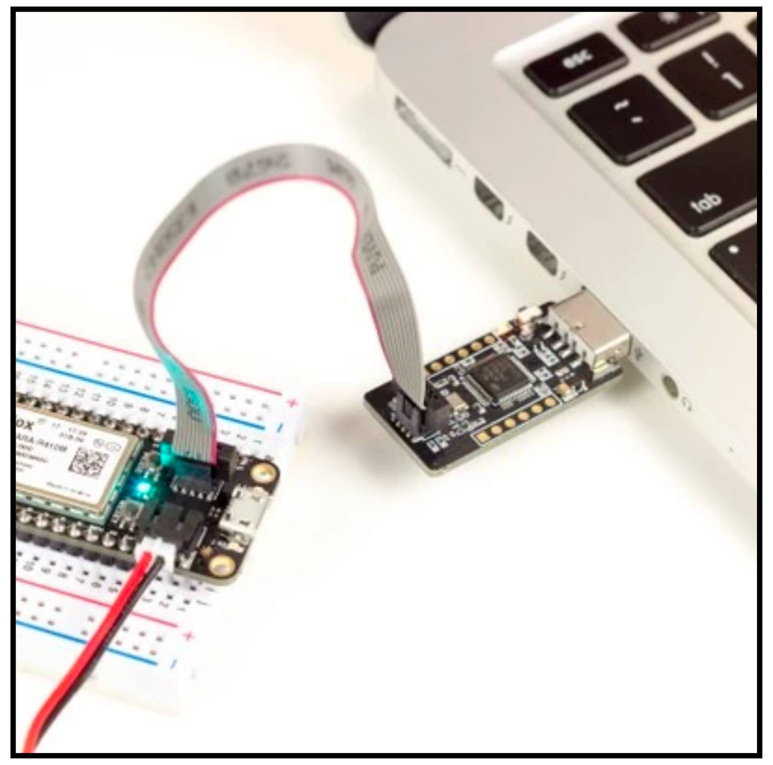
DEBUGGING IOT APPS, CIRCA 2018

```
Serial.println(“HERE”);

// Code that’s probably buggy because you forgot how to write C

uint32_t freemem = System.freeMemory();
Serial.printlnf("current free memory: %d", freemem);

// Code that’s probably also buggy because you forgot to connect
// that sensor to ground

Serial.printlnf(“Maybe I should have become an English
teacher.”);
```

ENTER PARTICLE WORKBENCH

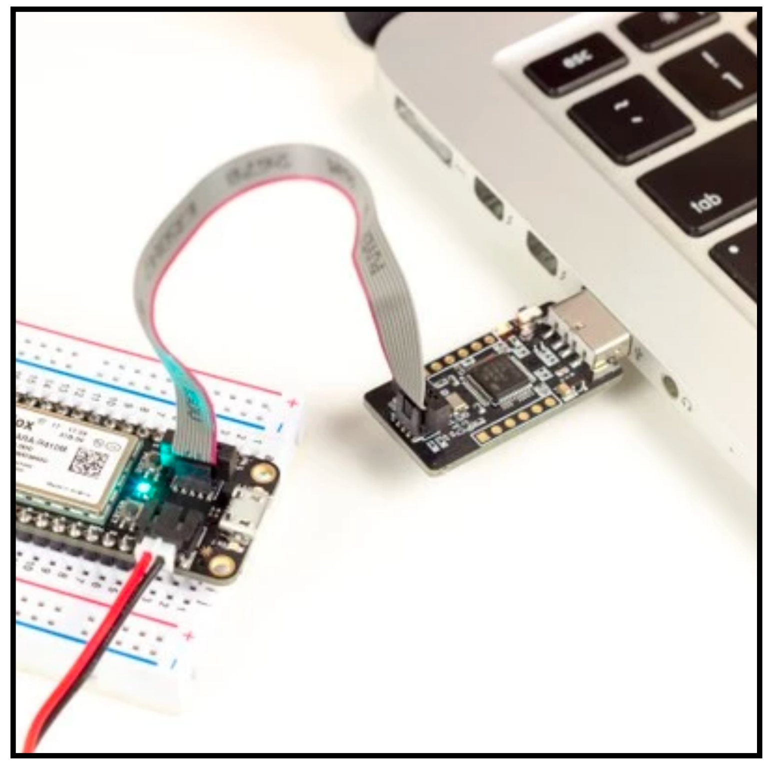
ON-DEVICE DEBUGGING WITH PARTICLE WORKBENCH

ON-DEVICE DEBUGGING WITH PARTICLE WORKBENCH



1. Connect a Particle Debugger

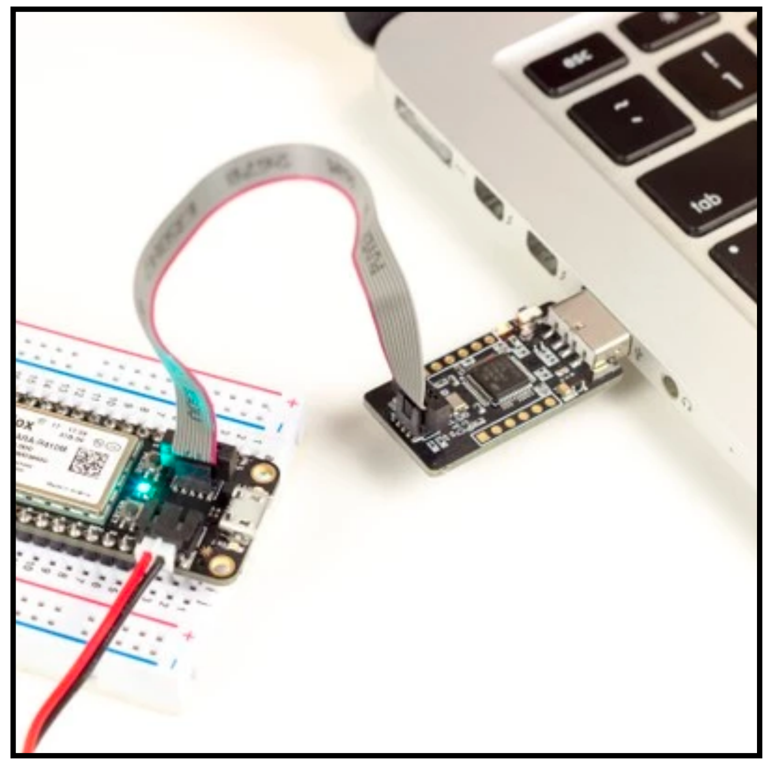
ON-DEVICE DEBUGGING WITH PARTICLE WORKBENCH



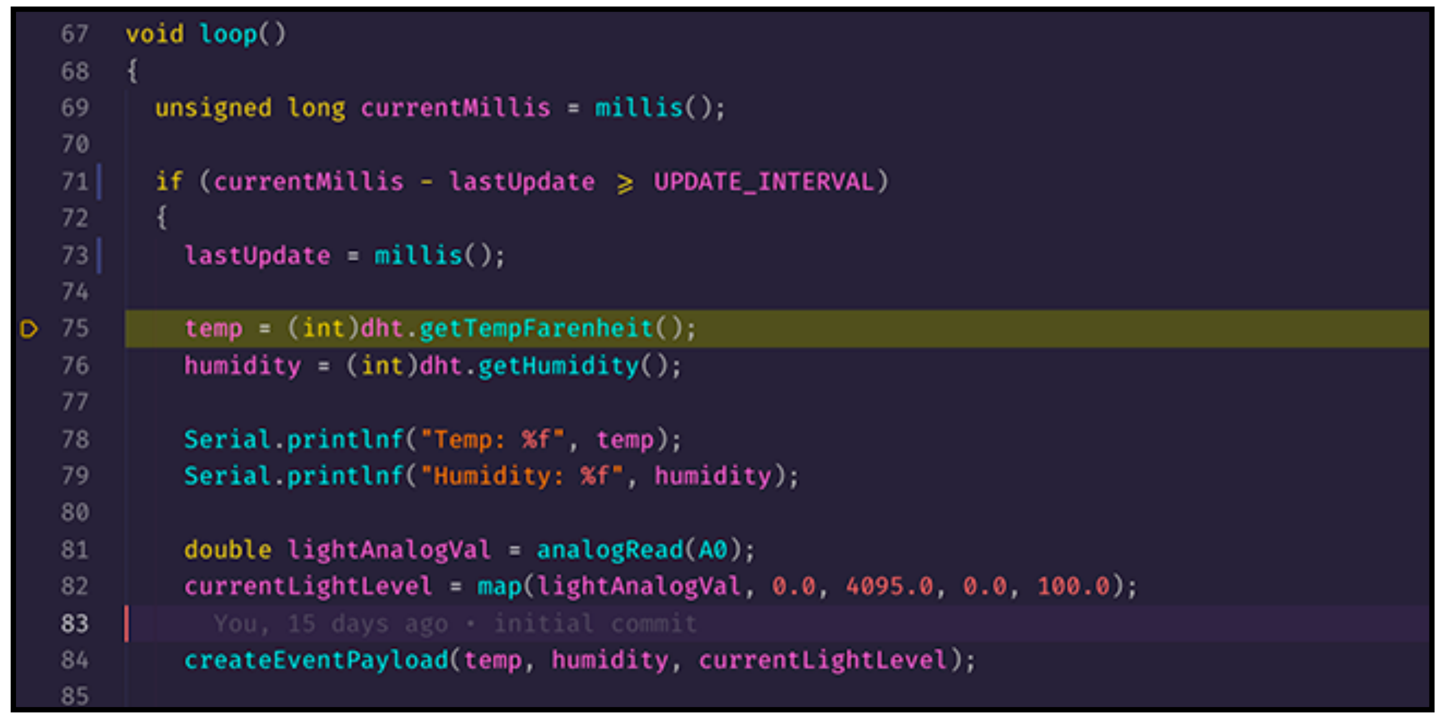
2. Create a Debug Build

1. Connect a Particle Debugger

ON-DEVICE DEBUGGING WITH PARTICLE WORKBENCH


1. Connect a Particle Debugger

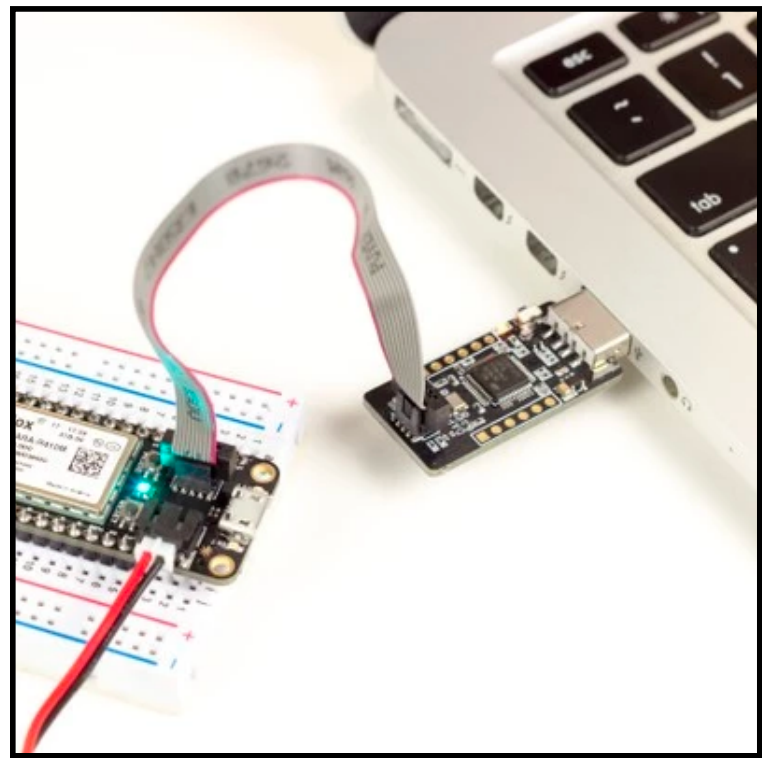
2. Create a Debug Build



```
void loop()
{
  unsigned long currentMillis = millis();

  if (currentMillis - lastUpdate >= UPDATE_INTERVAL)
  {
    lastUpdate = millis();

    temp = (int)dht.getTempFarenheit();
    humidity = (int)dht.getHumidity();

    Serial.printlnf("Temp: %f", temp);
    Serial.printlnf("Humidity: %f", humidity);

    double lightAnalogVal = analogRead(A0);
    currentLightLevel = map(lightAnalogVal, 0.0, 4095.0, 0.0, 100.0);
      You, 15 days ago • initial commit
    createEventPayload(temp, humidity, currentLightLevel);

```

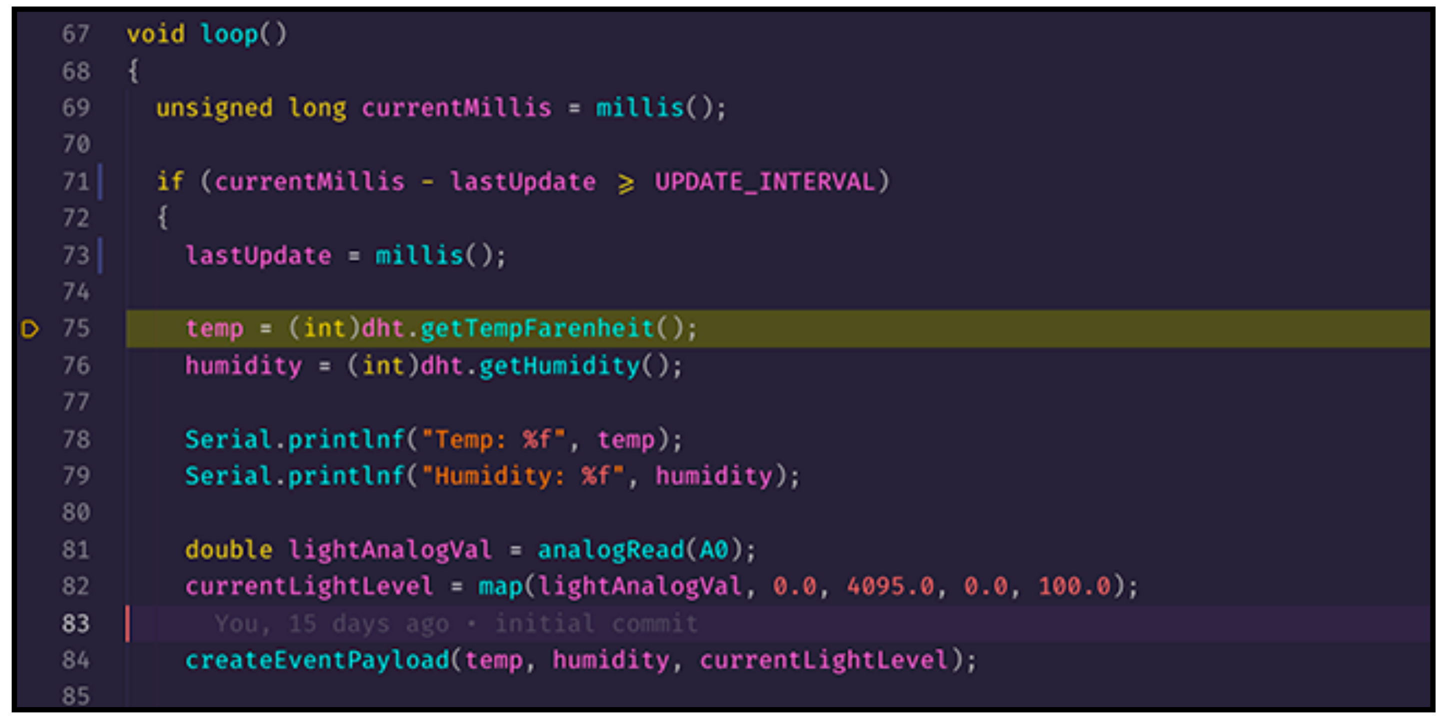
3. Inspect & Debug

ON-DEVICE DEBUGGING WITH PARTICLE WORKBENCH



1. Connect a Particle Debugger

2. Create a Debug Build



```
void loop()
{
  unsigned long currentMillis = millis();

  if (currentMillis - lastUpdate >= UPDATE_INTERVAL)
  {
    lastUpdate = millis();

    temp = (int)dht.getTempFarenheit();
    humidity = (int)dht.getHumidity();

    Serial.printlnf("Temp: %f", temp);
    Serial.printlnf("Humidity: %f", humidity);

    double lightAnalogVal = analogRead(A0);
    currentLightLevel = map(lightAnalogVal, 0.0, 4095.0, 0.0, 100.0);
      You, 15 days ago • initial commit
    createEventPayload(temp, humidity, currentLightLevel);

```

3. Inspect & Debug

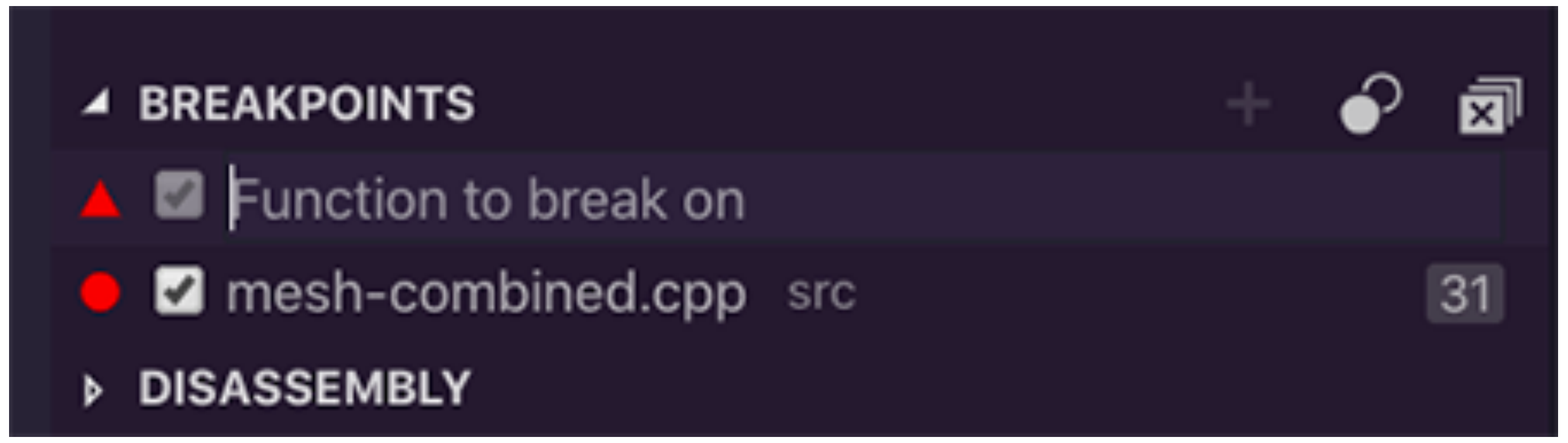
4. Re-flash a modular build to your device

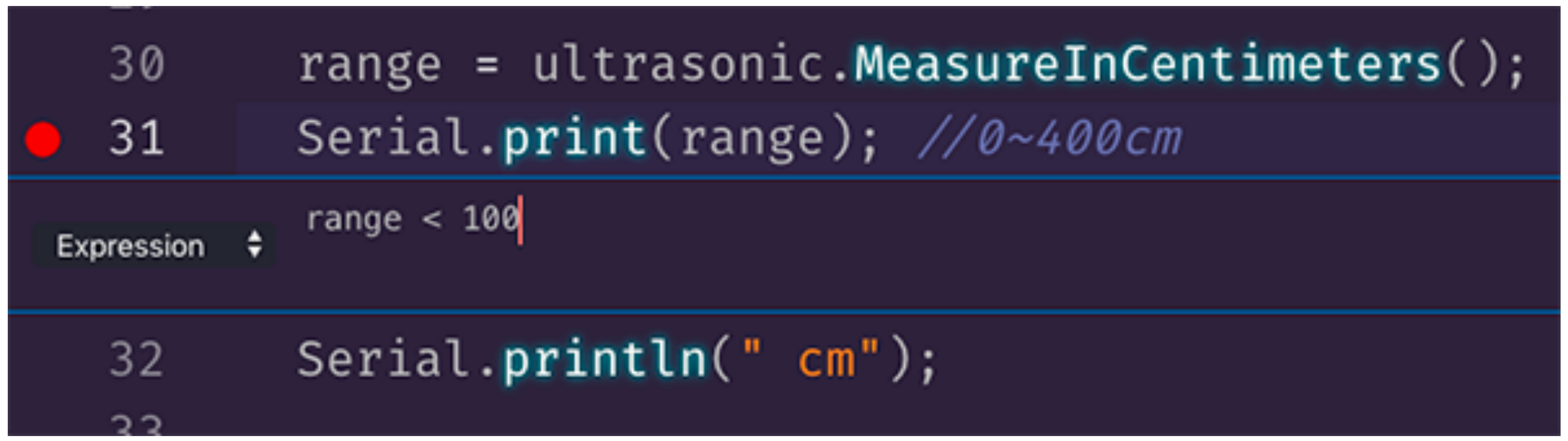
PARTICLE WORKBENCH DEBUGGING CAPABILITIES

- Breakpoints (incl. conditional)
- Step-debugging (into, over, etc.)
- Inspecting local, global and static variables
- Watching values
- Navigating the call stack
- Inspecting Registers and peripherals
- And more!

PARTICLE WORKBENCH DEBUGGING CAPABILITIES

- Breakpoints (incl. conditional)
- Step-debugging (into, over, etc.)
- Inspecting local, global and static variables
- Watching values
- Navigating the call stack
- Inspecting Registers and peripherals
- And more!





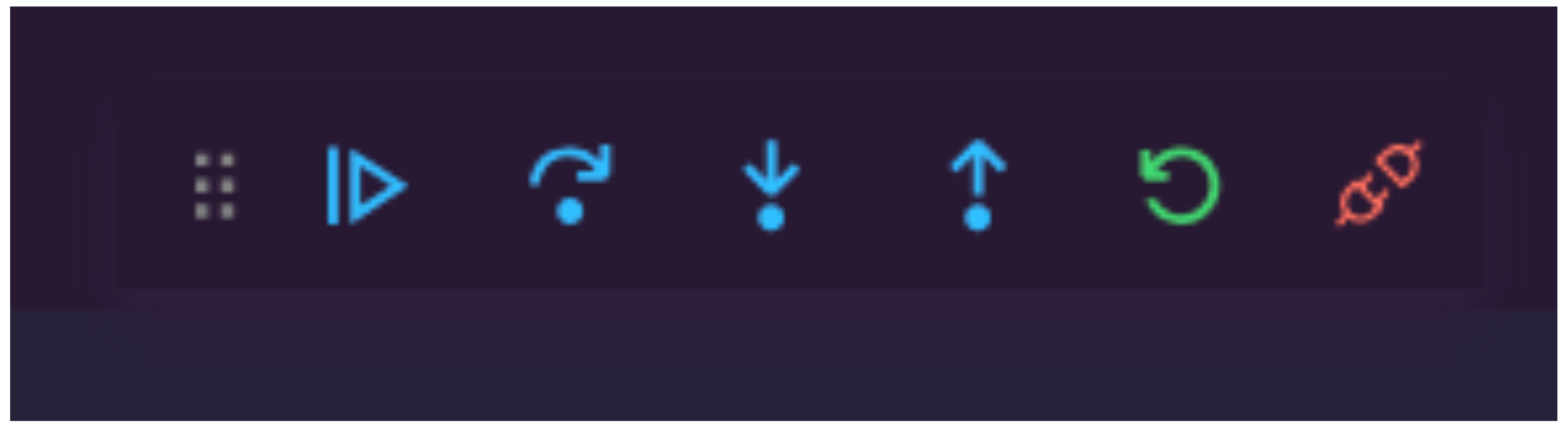
```
range = ultrasonic.MeasureInCentimeters();
Serial.print(range); //0~400cm
Expression    range < 100
Serial.println(" cm");
```

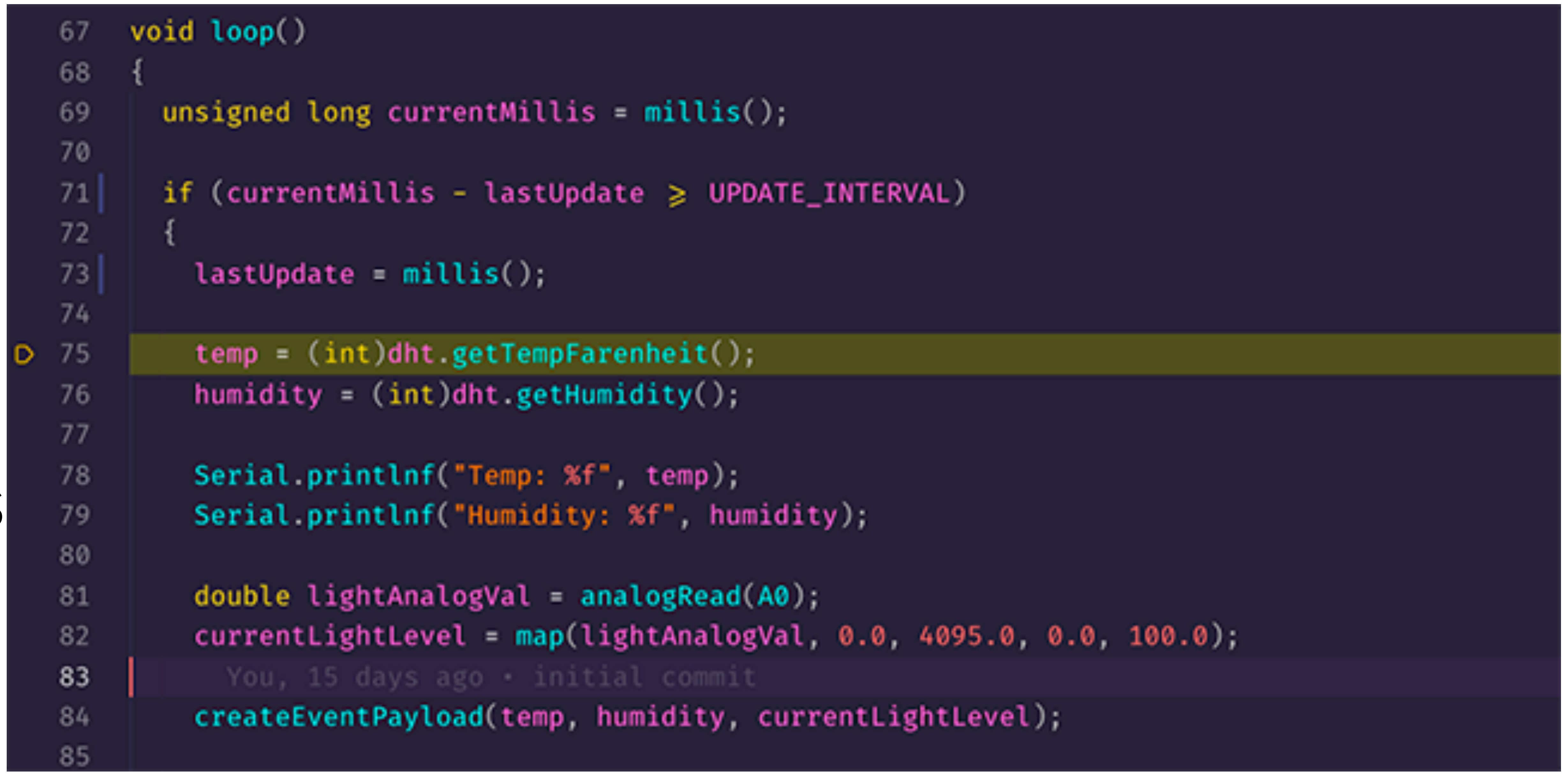
PARTICLE WORKBENCH DEBUGGING CAPABILITIES

- Breakpoints (incl. conditional)
- Step-debugging (into, over, etc.)
- Inspecting local, global and static variables
- Watching values
- Navigating the call stack
- Inspecting Registers and peripherals
- And more!

PARTICLE WORKBENCH DEBUGGING CAPABILITIES

- Breakpoints (incl. conditional)
- Step-debugging (into, over, etc.)
- Inspecting local, global and static variables
- Watching values
- Navigating the call stack
- Inspecting Registers and peripherals
- And more!





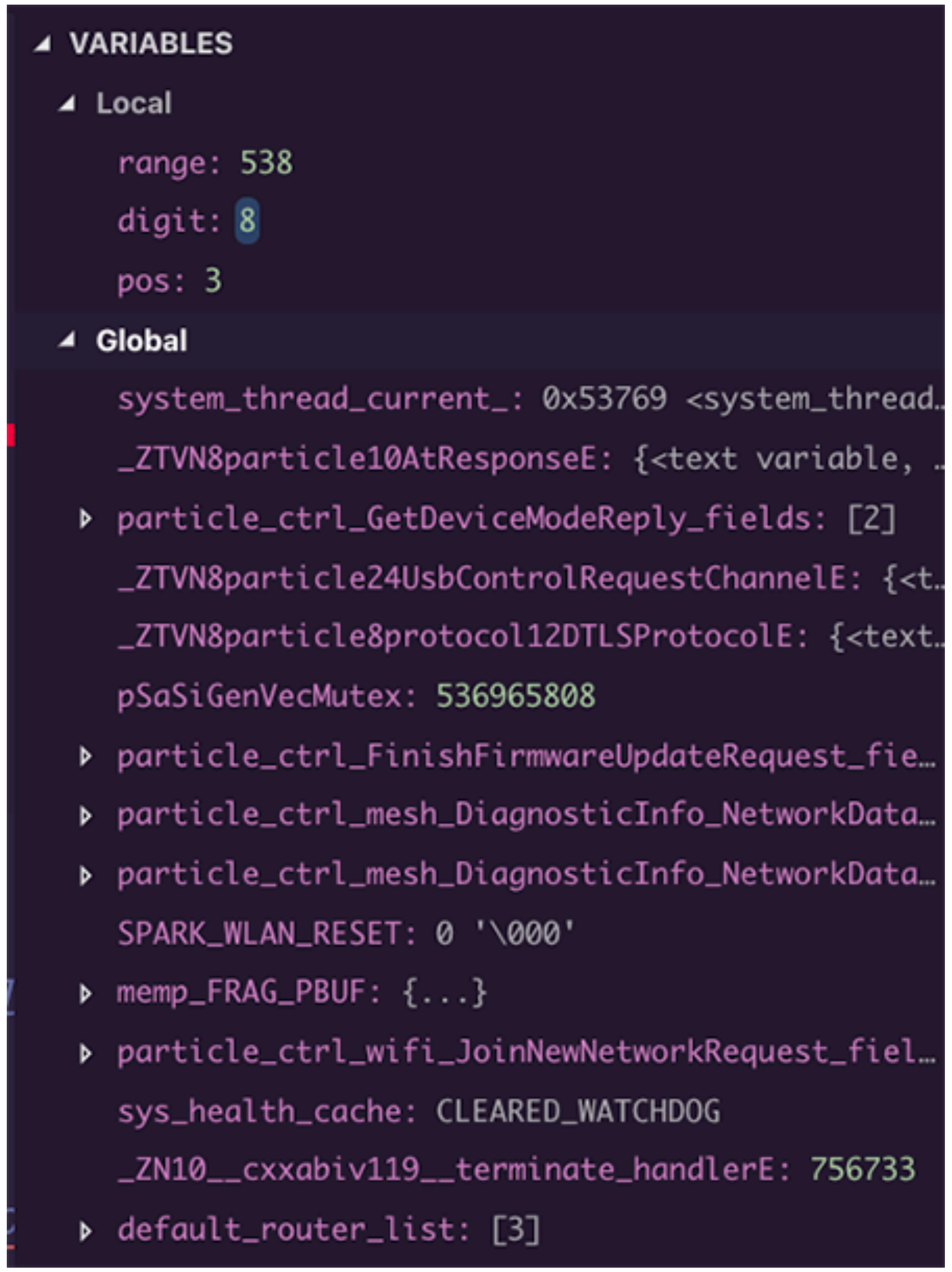
```
void loop()
{
  unsigned long currentMillis = millis();

  if (currentMillis - lastUpdate >= UPDATE_INTERVAL)
  {
    lastUpdate = millis();

    temp = (int)dht.getTempFarenheit();
    humidity = (int)dht.getHumidity();

    Serial.printlnf("Temp: %f", temp);
    Serial.printlnf("Humidity: %f", humidity);

    double lightAnalogVal = analogRead(A0);
    currentLightLevel = map(lightAnalogVal, 0.0, 4095.0, 0.0, 100.0);
      You, 15 days ago • initial commit
    createEventPayload(temp, humidity, currentLightLevel);
```

PARTICLE WORKBENCH DEBUGGING CAPABILITIES

- Breakpoints (incl. conditional)
- Step-debugging (into, over, etc.)
- Inspecting local, global and static variables
- Watching values
- Navigating the call stack
- Inspecting Registers and peripherals
- And more!

PARTICLE WORKBENCH DEBUGGING CAPABILITIES

- Breakpoints (incl. conditional)
- Step-debugging (into, over, etc.)
- Inspecting local, global and static variables
- Watching values
- Navigating the call stack
- Inspecting Registers and peripherals
- And more!



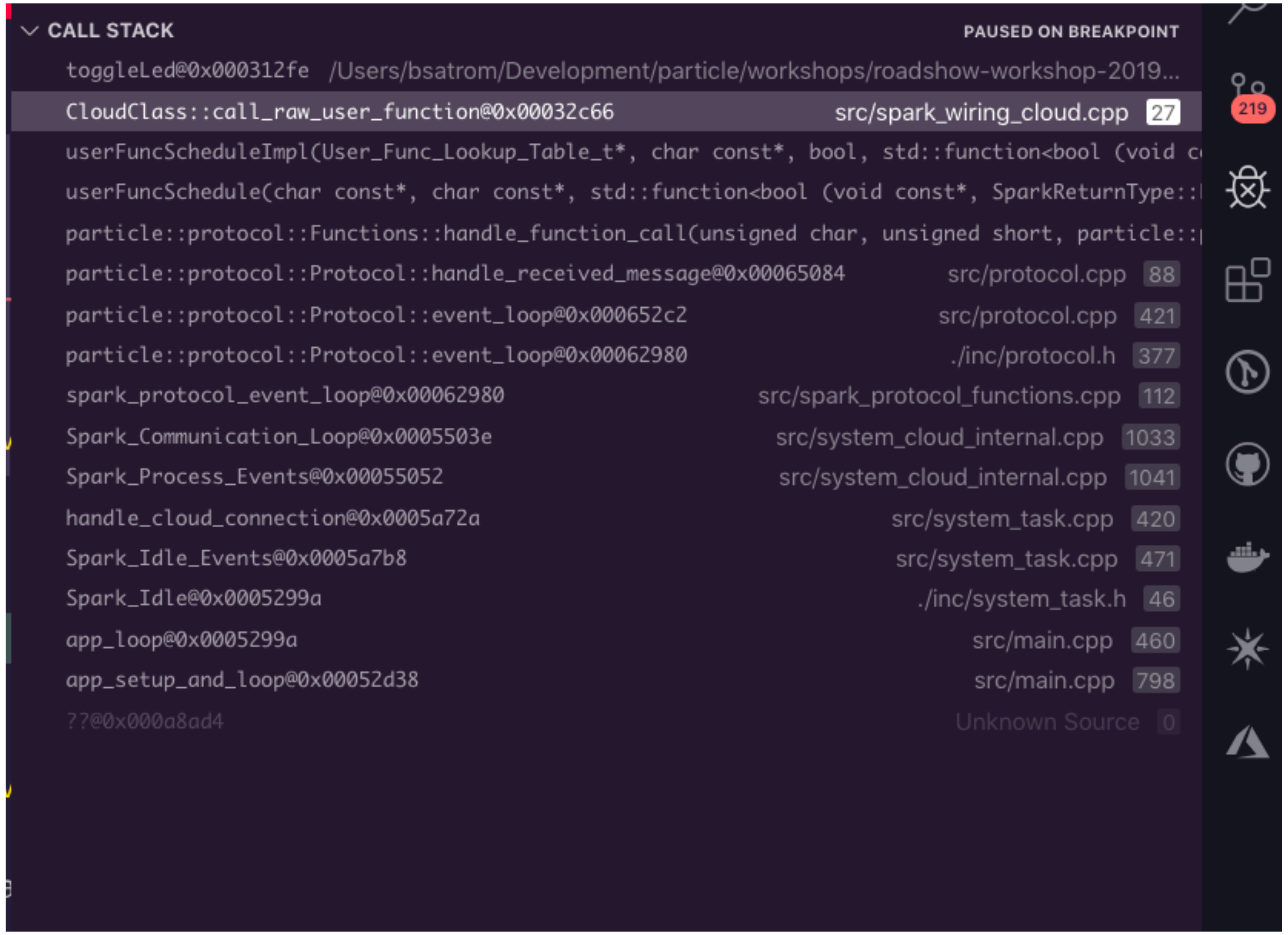
```
◢ VARIABLES
  ◢ Local
      range: 538
      digit: 8
      pos: 3
  ◢ Global
      system_thread_current_: 0x53769 <system_thread…
      _ZTVN8particle10AtResponseE: {<text variable, …
    ▸ particle_ctrl_GetDeviceModeReply_fields: [2]
      _ZTVN8particle24UsbControlRequestChannelE: {<t…
      _ZTVN8particle8protocol12DTLSProtocolE: {<text…
      pSaSiGenVecMutex: 536965808
    ▸ particle_ctrl_FinishFirmwareUpdateRequest_fie…
    ▸ particle_ctrl_mesh_DiagnosticInfo_NetworkData…
    ▸ particle_ctrl_mesh_DiagnosticInfo_NetworkData…
      SPARK_WLAN_RESET: 0 '\000'
    ▸ memp_FRAG_PBUF: {...}
    ▸ particle_ctrl_wifi_JoinNewNetworkRequest_fiel…
      sys_health_cache: CLEARED_WATCHDOG
      _ZN10__cxxabiv119__terminate_handlerE: 756733
    ▸ default_router_list: [3]
```

PARTICLE WORKBENCH DEBUGGING CAPABILITIES

- Breakpoints (incl. conditional)
- Step-debugging (into, over, etc.)
- Inspecting local, global and static variables
- Watching values
- Navigating the call stack
- Inspecting Registers and peripherals
- And more!

PARTICLE WORKBENCH DEBUGGING CAPABILITIES

- Breakpoints (incl. conditional)
- Step-debugging (into, over, etc.)
- Inspecting local, global and static variables
- Watching values
- Navigating the call stack
- Inspecting Registers and peripherals
- And more!



```
toggleLed@0x000312fe  /Users/bsatrom/Development/particle/workshops/roadshow-workshop-2019...
CloudClass::call_raw_user_function@0x00032c66                 src/spark_wiring_cloud.cpp 27
userFuncScheduleImpl(User_Func_Lookup_Table_t*, char const*, bool, std::function<bool (void c
userFuncSchedule(char const*, char const*, std::function<bool (void const*, SparkReturnType::
particle::protocol::Functions::handle_function_call(unsigned char, unsigned short, particle::
particle::protocol::Protocol::handle_received_message@0x00065084        src/protocol.cpp 88
particle::protocol::Protocol::event_loop@0x000652c2                     src/protocol.cpp 421
particle::protocol::Protocol::event_loop@0x00062980                     ./inc/protocol.h 377
spark_protocol_event_loop@0x00062980                src/spark_protocol_functions.cpp 112
Spark_Communication_Loop@0x0005503e                 src/system_cloud_internal.cpp 1033
Spark_Process_Events@0x00055052                     src/system_cloud_internal.cpp 1041
handle_cloud_connection@0x0005a72a                            src/system_task.cpp 420
Spark_Idle_Events@0x0005a7b8                                  src/system_task.cpp 471
Spark_Idle@0x0005299a                                         ./inc/system_task.h 46
app_loop@0x0005299a                                                src/main.cpp 460
app_setup_and_loop@0x00052d38                                      src/main.cpp 798
??@0x000a8ad4                                                    Unknown Source 0
```

PARTICLE WORKBENCH DEBUGGING CAPABILITIES

- Breakpoints (incl. conditional)
- Step-debugging (into, over, etc.)
- Inspecting local, global and static variables
- Watching values
- Navigating the call stack
- Inspecting Registers and peripherals
- And more!

ON-DEVICE DEBUGGING

DEMO

LET'S BUILD SOME INTEGRATIONS!

PLEASE RATE THIS SESSION!

https://part.cl/feedback