

LTE-M を使って、 センサの値をクラウドで管理 (Rev.03)

本研究開発の成果の一部は、国立研究開発法人新エネルギー・産業技術総合開発機構（NEDO）委託事業「IoT 推進のための横断技術開発プロジェクト/トリリオンノード・エンジンの研究開発」の結果得られたものです。

リーフ、基板、治具等の配布物は、研究・評価の目的以外での使用は、行わないください

Revision Sheet

Revision	Date	Content	Name
00	2019-9-30	正式発行	東京大学

目次

1. LTE-Mリーフの組み方	4
1-1. 必要なリーフ	4
1-2. リーフの組み立て.....	4
1-3. リーフのネジ止め	5
1-4. 組み立て完成	5
2. IFTTT Platform(Maker)の設定方法	6
2-1. アカウント作成	6
2-2. 「this」の設定.....	9
2-3. 「that」の設定	12
3. 必要情報の確認	16
3-1. 「Key」の確認	16
3-2. 設定確認・修正の方法.....	18
4. スケッチについて	19
4-1. スケッチのダウンロード	19
4-2. ライブラリの準備.....	19
4-3. スケッチ加筆修正事項.....	21
4-4. スケッチ書き込み.....	22
5. IFTTT との通信	24
5-1. 「LTE-M」リーフのデータ送信	24
5-2. Google スプレッドシートの表記確認.....	25
5-3. 「LTE Mリーフデモ IFTTT の例」動画	25

1. LTE-Mリーフの組み方

1-1. 必要なリーフ



上から1段目：LTE-Mリーフ、2段目：Leaf×2リーフ、4-Sensorsリーフ、3段目：AVR MCUリーフ、USBリーフ、LCDリーフ

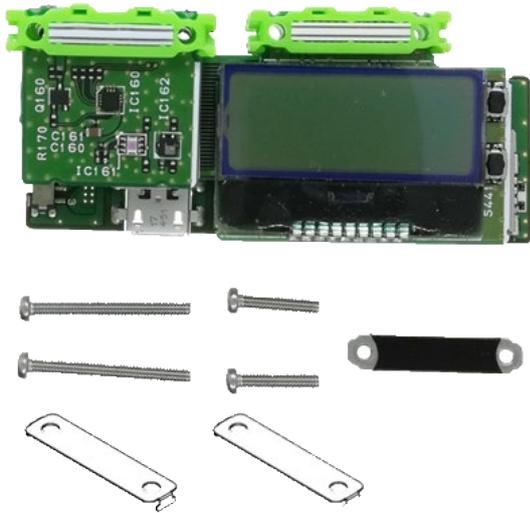
1-2. リーフの組み立て



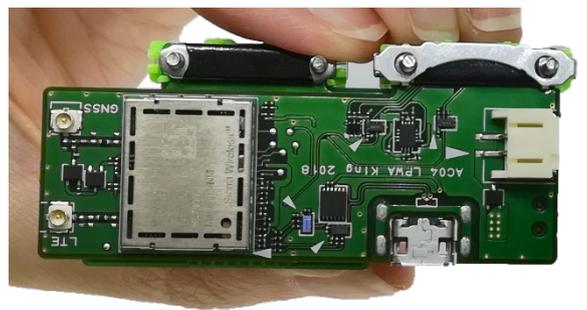
下から、LTE-Mリーフ、Leaf×2リーフを重ねます。

(写真左側) AVR MCUリーフ、USBリーフ、4-Sensorsリーフ (写真右側) LCDリーフ

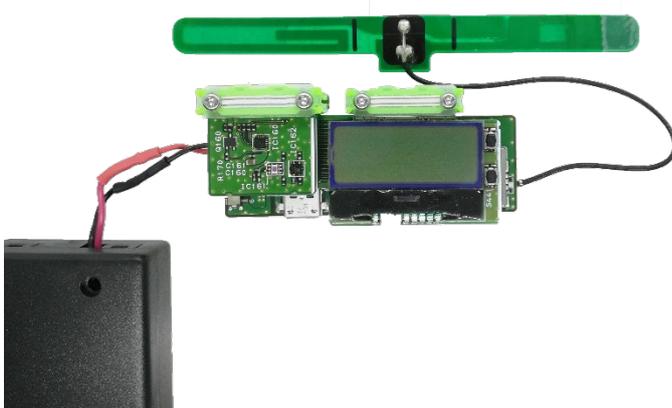
1-3. リーフのネジ止め



一番下に Nut Plate を固定し、コネクタに Connector Cover を装着したら、写真左側のリーフを M2×20mm で、写真右側を M2×10mm でネジ止めします。



1-4. 組み立て完成



LTE-M リーフ用アンテナ、SIM、電池ボックスを接続したら完成です。

2. IFTTT Platform(Maker)の設定方法

IFTTTとは、異なるプラットフォームやソーシャルメディアを連携し、指定した動作を自動化するサービスです。

また「IF This Then That」の略称で、「もしこれが起きたら、あれをする」といった意味をもち、「this」をトリガー（引き金）にして、「that」のアクションを起こします。

プラットフォーム等を連携させるためには、「My Applets」を作成し、動作に対する指示を作成する必要があります。

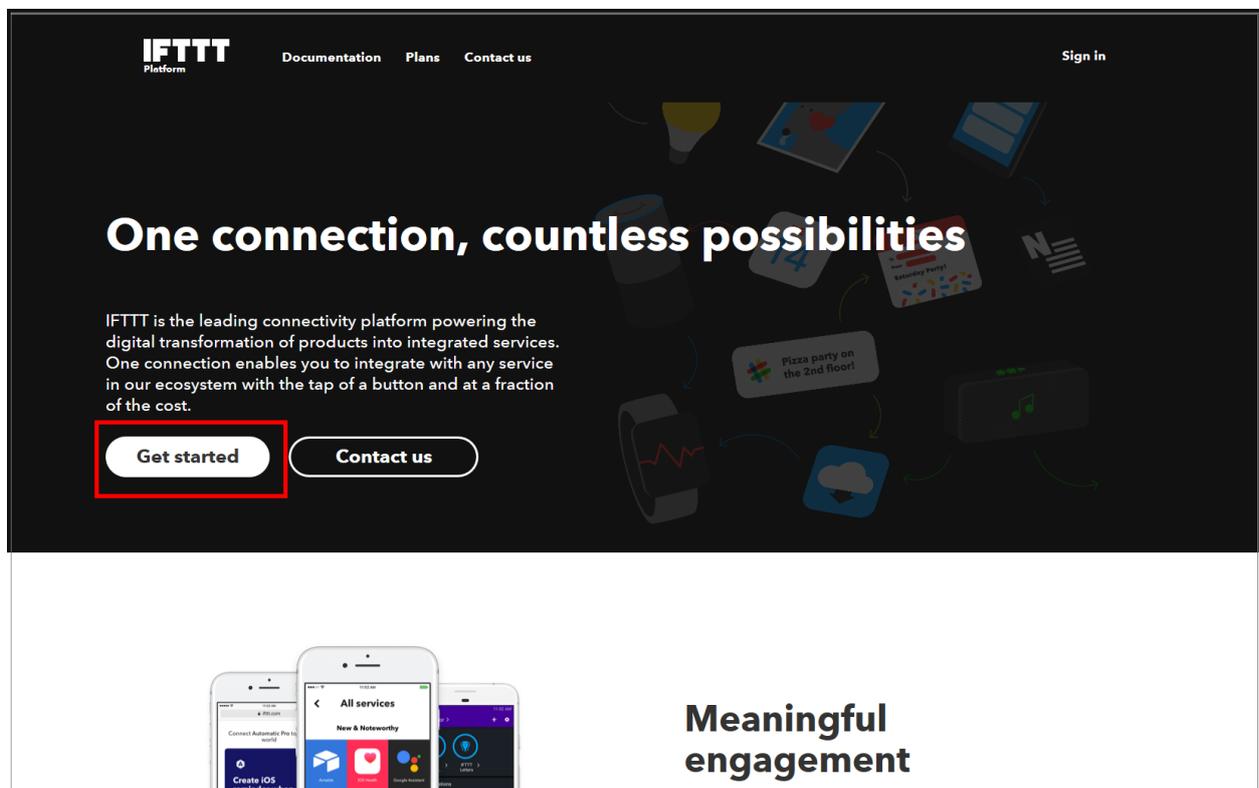
具体的に今回 IFTTT を使用して行うのは、「LTE-M リーフ」からセンサー情報を特定の URL に POST 送信し、そのデータを「Google スプレッドシート」に表示させることです。

2-1. アカウント作成

- 1) <https://platform.ifttt.com/maker> から、必要事項を登録します。

※今回は、「Google スプレッドシート」を使用するため、先に「Google アカウント」で Sign in してください。

（「Google アカウント」がない場合は、先に登録します。）



「Get started」を選択



Get started with IFTTT



Continue with Google



Continue with Facebook

Or use your password to [sign up](#) or [sign in](#)



Download on the
App Store



GET IT ON
Google Play

[Help](#)

[Careers](#)

[Terms & Privacy](#)

「Continue with Google」を選択、サインインして進みます。

2) 会社情報等、必要事項を入力します。(右上には Google アカウントが表示されます。)

IFTTT
Platform

Contact us Google アカウント名 ▼

Tell us about your company

Company name

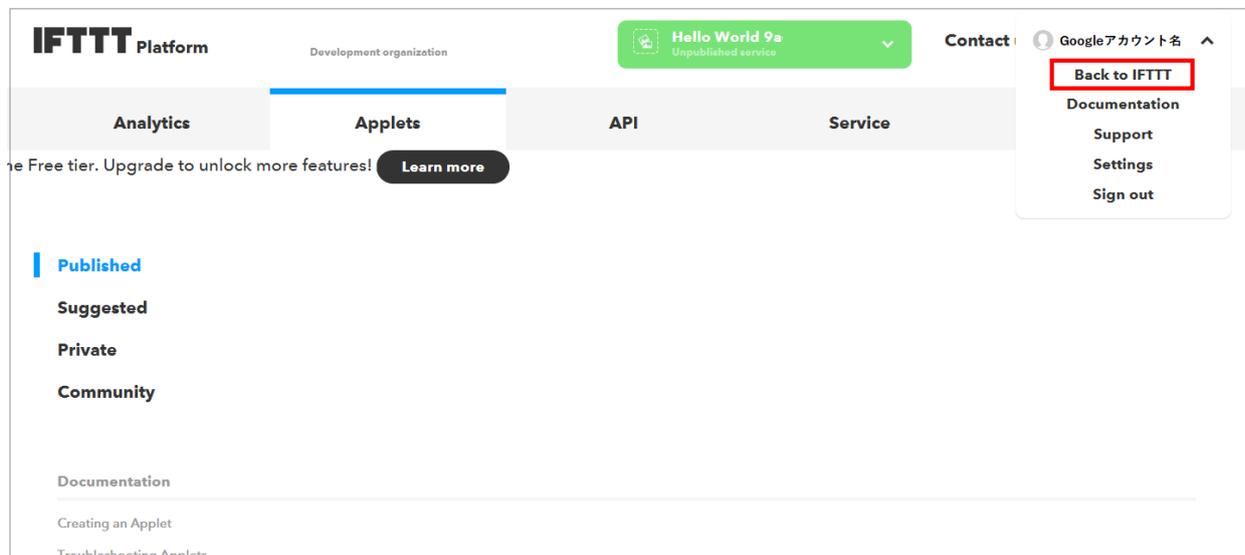
Get started

Documentation Plans Contact us Careers Terms

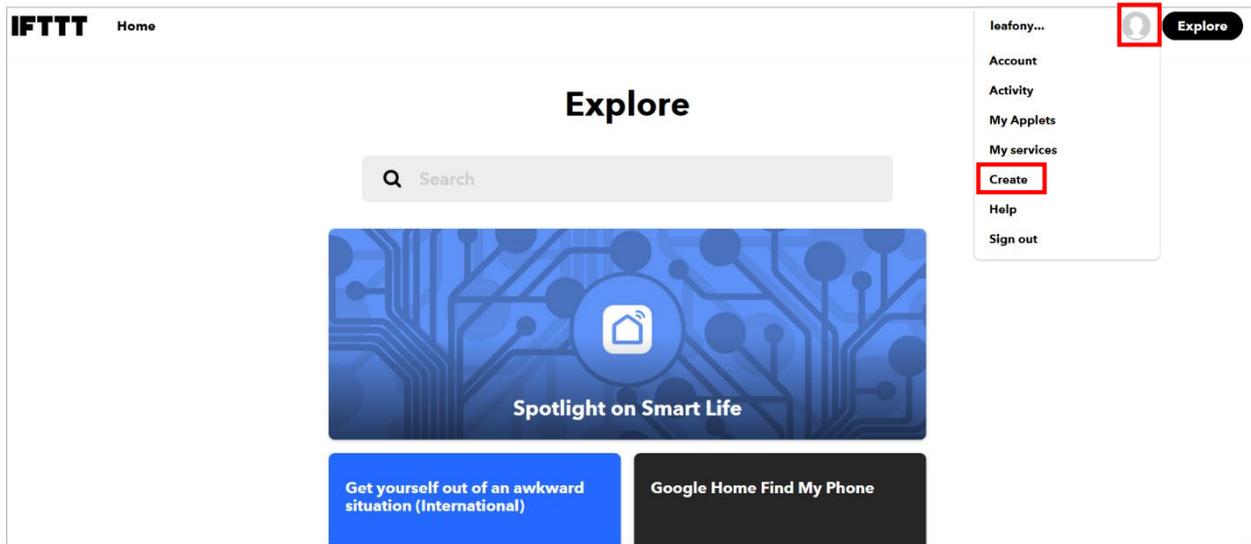
3) 下記、赤枠内ほか、必要事項を入力します。

The screenshot shows the 'General' configuration page for an IFTTT service. The service name is 'Hello World 9a48ac'. The 'Description' field is empty and highlighted with a red box. Below it, a note says 'A 3-4 sentence description of your service and what makes it great; use [Markdown](#) for links.' The 'Category' dropdown is empty. The 'Official web page' field contains 'http://example.com'. The 'Support email' field is highlighted with a red box and contains 'support@example.com'. Below it, a note says 'A valid support email address for communications from IFTTT or forwarded from IFTTT users.' The 'IFTTT service ID' field is empty.

4) 「SAVE」で設定を保存したら、「Back to IFTTT」から、<https://ifttt.com/discover> を表示します。



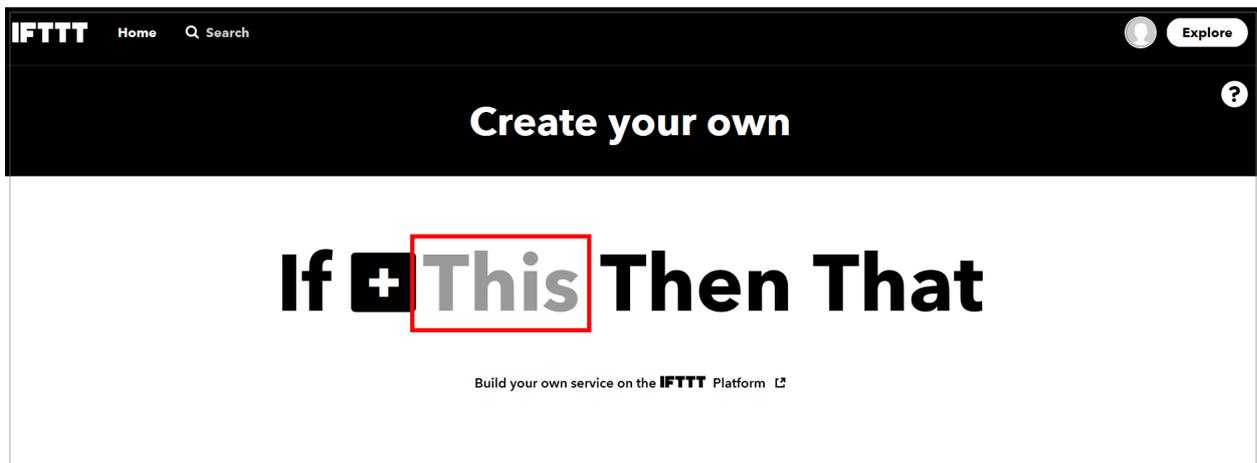
アイコンをクリックして、プルダウンの中から「Create」をクリックします。



2-2. 「this」の設定

1) 「if this then that」画面が開き、設定を開始します。

(「this」には、既存のサービスを選択するほか、アプリからの Web リクエストも可能。)



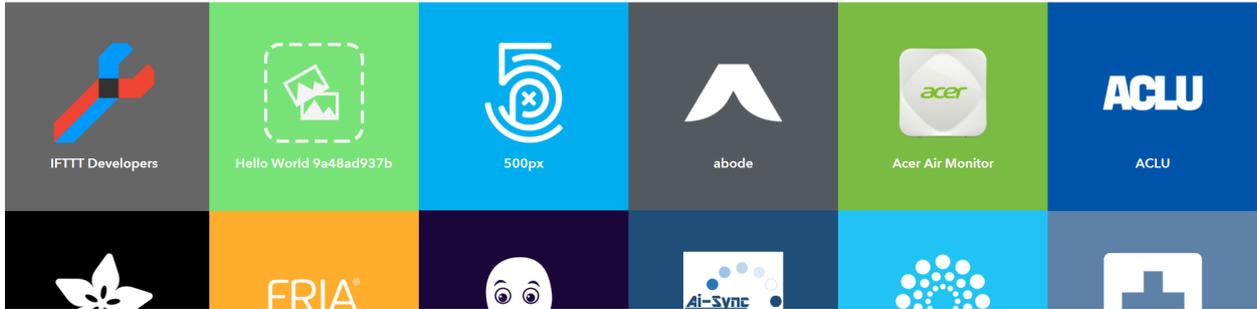
2) ここからは、LTE-M リーフのセンサー情報を Google スプレッドシートに書き込むための設定を行います。

まず、「this」(トリガー) をクリックし、検索窓に「Webhooks」と入力し、サービスを探します。

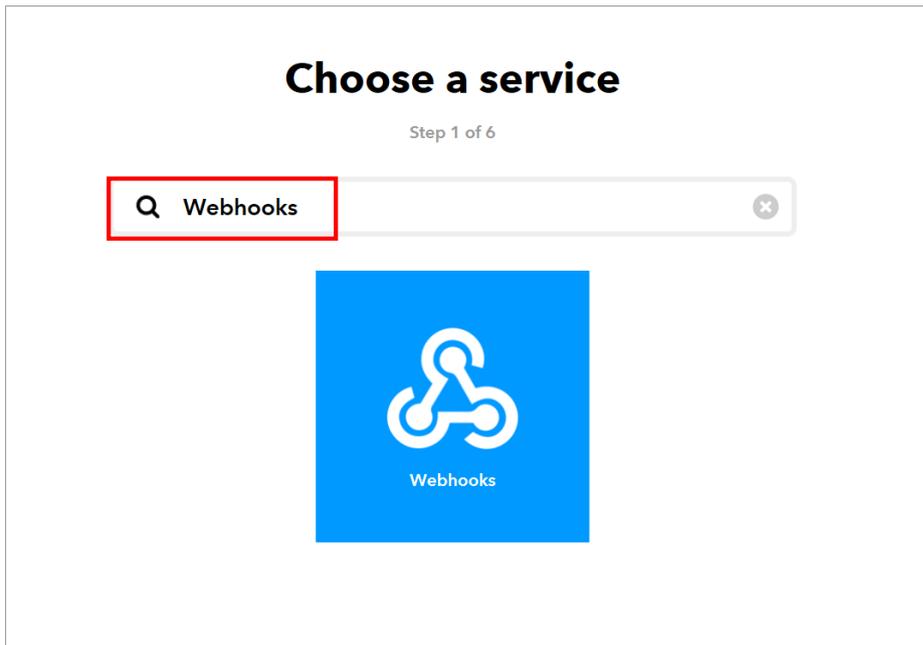
< Back

Choose a service

Step 1 of 6



画像を選択してください。



3) 画面が変わるので、トリガーを設定するために、「Receive a web request」をクリックします。

IFTTT Home Q Search

< Back

Choose trigger

Step 2 of 6

Receive a web request

This trigger fires every time the Maker service receives a web request to notify it of an event. For information on triggering events, go to your Maker service settings and then the listed URL (web) or tap your username (mobile)

Don't see what you're looking for? [Suggest a new trigger](#)

4) 「Event Name」に “ LTE-M Leaf ” と書き、トリガーの設定を完了します。

IFTTT Home Q Search

< Back

Complete trigger fields

Step 2 of 6

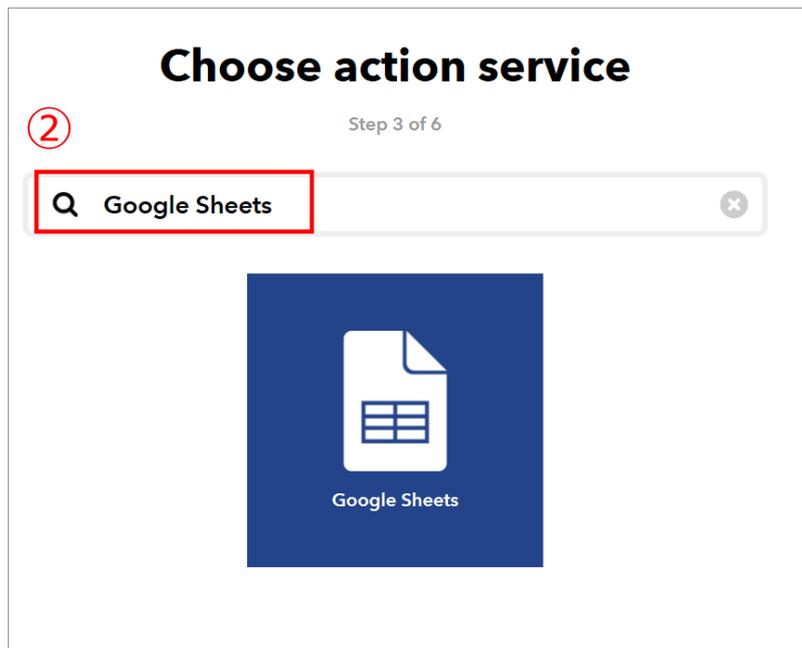
Event Name

The name of the event, like "button_pressed" or "front_door_opened"

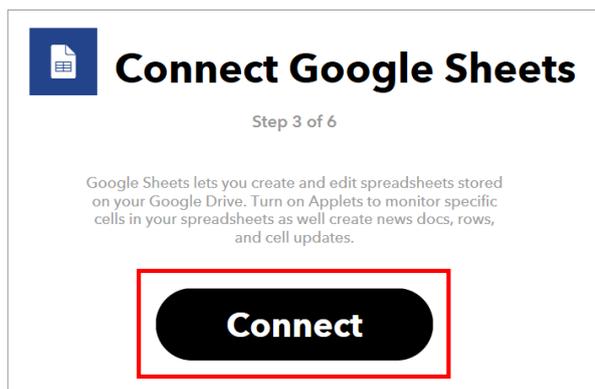
Create trigger

2-3. 「that」の設定

1) 「Create trigger」をクリックすると画面が移るので、①「that」(アクション) から、②「Google Sheets」を検索し、選択します。



2) 「Google Sheets」に「Connect」し、IFTTT のアクセスを許可します。

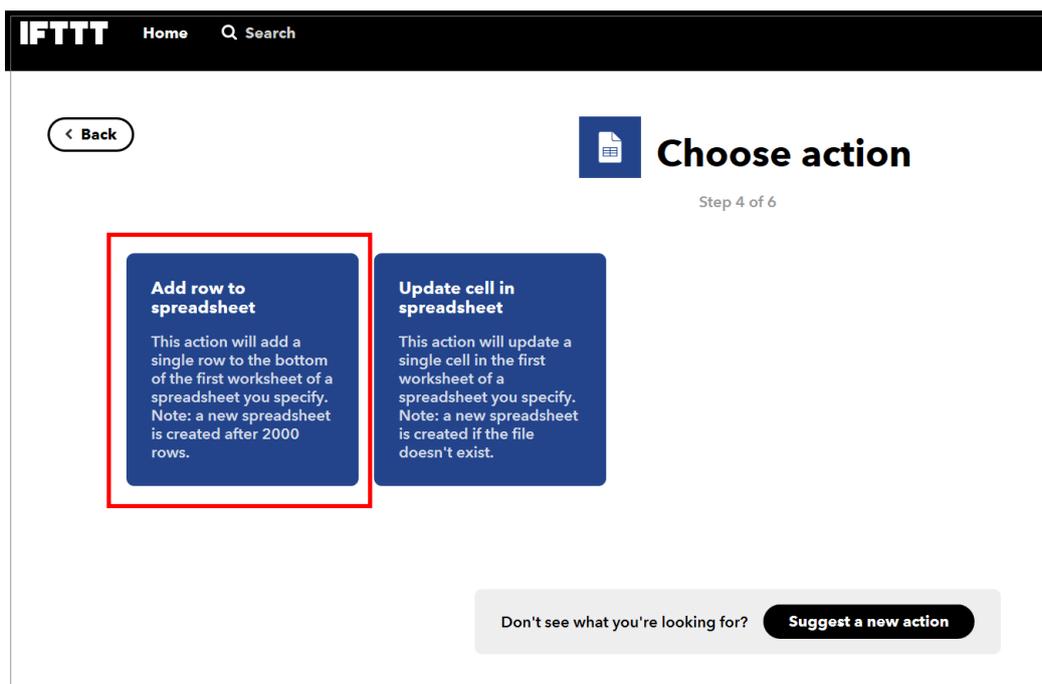




3) 次に、アクションを指定するために、「Add row to spreadsheet」をクリックします。

(指定のスプレッドシートに、行を追加する設定を選びます。

※2000 行を超えたら、新規スプレッドシートを作成するという内容になっています。)



4) 「action fields」を完成させるために、以下を入力していきます。

「Spreadsheet name」には、ここでは「IFTTT_LTE」（例）と入力。

5) 「Formatted row」では、Google スプレッドシートの記述方法を指定できます。

例)

今回は、「{{OccurredAt}}」で日時を表示し、=SPLIT 関数で、「, カンマ」区切りした値（文字列）を次のセルに記入するように、以下のスケッチを使用しました。

```
{{OccurredAt}} ||| {{Value1}} ||| =SPLIT("{{Value2}}",",")
```

参考 URL)

<https://qiita.com/komi360/items/35769ec65b4b06380d00>

Complete action fields

Step 5 of 6

Spreadsheet name

IFTTT_LTE

Will create a new spreadsheet if one with this title doesn't exist **Add ingredient**

Formatted row

OccurredAt ||| EventName ||| Value1 ||| Value2 ||| Value3

Use "|||" to separate cells **Add ingredient**

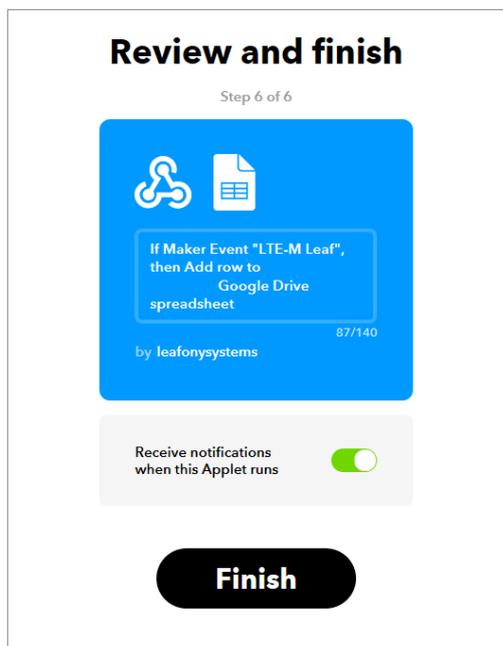
Drive folder path

IFTTT/MakerWebooks/ EventName

Format: some/folder/path (defaults to "IFTTT") **Add ingredient**

Create action

6) レビューが表示されるので、「Finish」をクリックします。



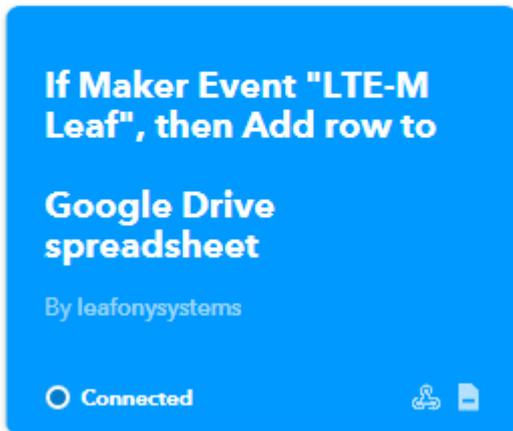
※この表示は、「My Applets」からも見るすることができます。

3. 必要情報の確認

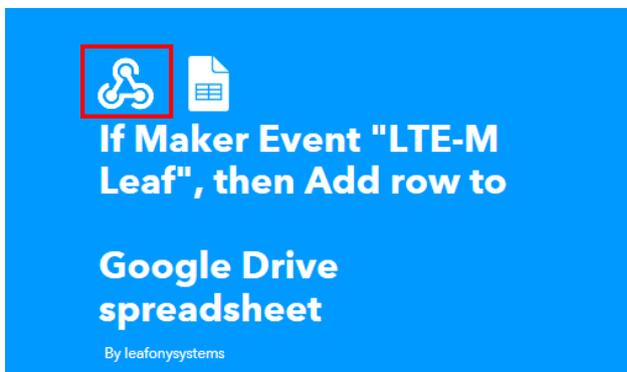
3-1. 「Key」の確認

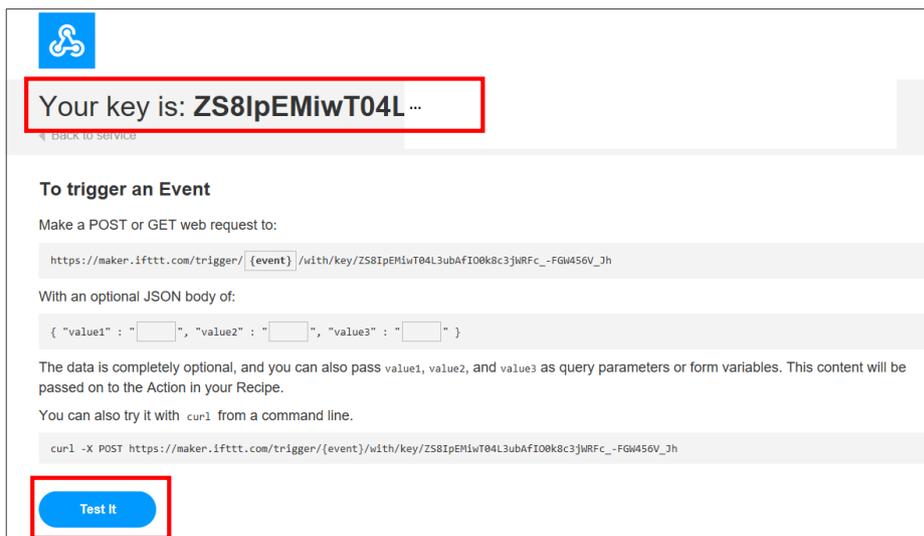
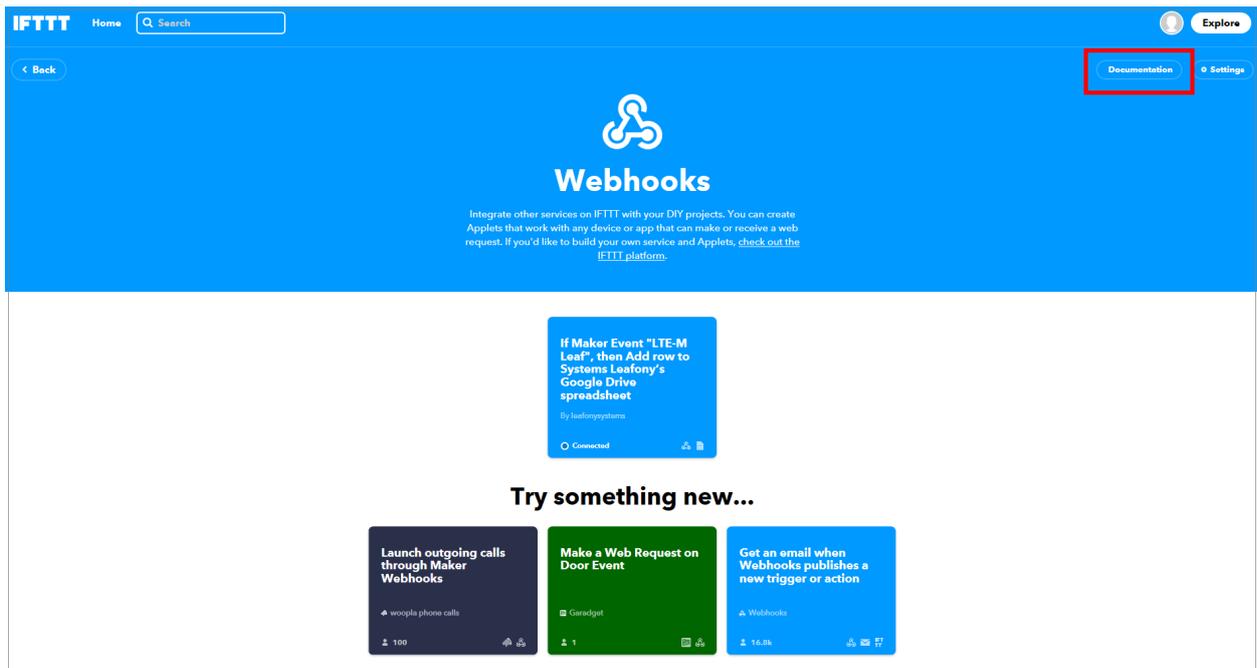
1) プルダウンから「My Applets」を選び、画像をクリックします。以下の表示から「Webhooks」マークをクリックします。

My Applets



2) 開いたページから、再度「Webhooks」マークをクリックし、右上の「Documentation」をクリックして「Key」を表示します。





3) ここでは、{evnet}を “ LTE-M Leaf ” と書き換えます。

4) {"value1":"□", "value2":"□", "value3":"□"} のスペース部分に、それぞれ **1**、**2**、**3** と書き込み、下記の表示のようにします。

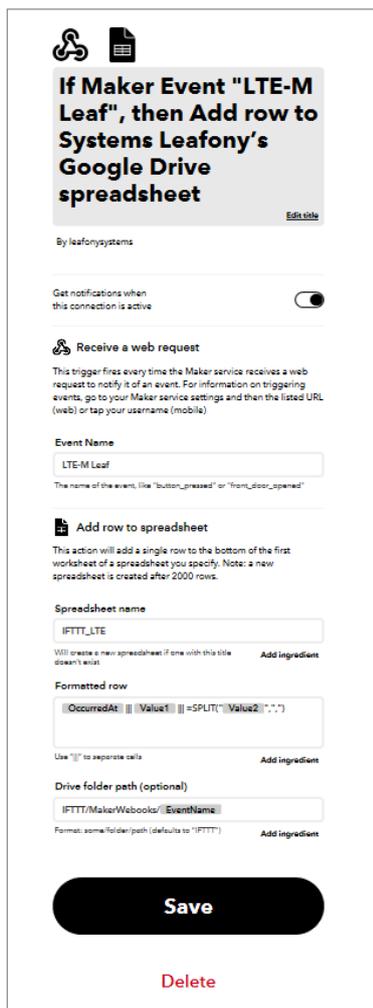
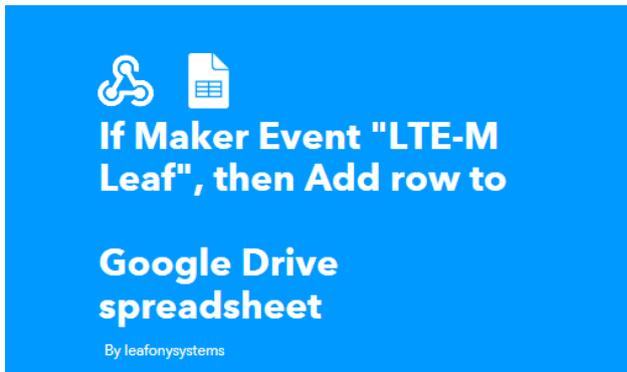
```
{"value1":"1", "value2":"2", "value3":"3"}
```

5) 「Test it」をクリックします。

Google ドライブ内にファイルが作成され、スプレッドシートに 1、2、3 と書き込まれていれば IFTTT の設定は完了です。

3-2. 設定確認・修正の方法

なお、ここまでの設定を確認、修正したい場合は、「My Applets」から「Webhooks」を選び、「Connected」をクリックしてください。



4. スケッチについて

注意：本スケッチは 1 時間あたり 150Kbyte 程度の送受信が発生します。通信容量にご注意ください。

4-1. スケッチのダウンロード

<https://trillion-node.org/lte-m/> 「LTE-M を使ってセンサの値をクラウドで管理しよう」(パスワード: trillion2019) から、「LPWA_IFTTT.ino」のスケッチをダウンロードし、「ドキュメント」の「Arduino」フォルダに置きます。

C:¥Users¥k¥Documents¥Arduino¥LPWA_IFTTT

4-2. ライブラリの準備

1) まず Arduino のライブラリは、通常以下の 2 か所に保存されます。

- ① C:¥Program Files (x86)¥Arduino¥libraries
- ② C:¥Users¥アカウント名¥Documents¥Arduino¥libraries

Arduino IDE をインストールした時、①にライブラリが保存されます (Arduino IDE Standard Libraries)。

※ 「Arduino IDE Standard Libraries」は、Arduino IDE をインストールした際のデフォルトです。

また、GitHub などからダウンロードしたライブラリ (ZIP ファイル) は、「スケッチ」→「ライブラリをインクルード」→「.ZIP 形式のライブラリをインストール」を選択し、②に保存してください。

2) 「Arduino IDE Standard Libraries」以外に必要なライブラリは、以下 URL 内より別途ダウンロードし、C:¥Users¥アカウント名¥Documents¥Arduino¥libraries に追加する必要があります。

<https://trillion-node.org/lte-m/> よりダウンロードできます。

※注意点も併記していますので、上記 URL をご参照ください。

■ 4-Sensors ライブラリ

- ・ユニファイドセンサドライバ include file: Adafruit_Sensor.h
- ・温湿度センサデータ取得 include file: HTS221.h
- ・照度センサデータ取得 include file: ClosedCube_OPT3001.h
- ・加速度センサデータ取得 include file : Adafruit_LIS3DH.h

■ LCD ライブラリ

- ・LCD の制御 include file: ST7032.h

■ AVR MCU ライブラリ

- ・タイマ割り込み include file: MsTimer2.h

3) 【重要】下記フォルダにある SoftwareSerial.h を一部、編集します。

<C:¥Users¥ユーザー名

¥AppData¥Local¥Arduino15¥packages¥arduino¥hardware¥avr¥1.8.1¥libraries¥SoftwareSerial¥src>

または

<C:¥Program Files (x86)¥Arduino¥hardware¥arduino¥avr¥libraries¥SoftwareSerial¥src>

※SoftwareSerial.h が見つからない場合は、プロパティの「表示」から、「隠しファイルを表示する」設定を選んでください。

「SoftwareSerial.h」を「書き込み可能な状態」にしてから、メモ帳などのソフトで開いてください。

下記「64」から「256」への変更を推奨します。

```
#define _SS_MAX_RX_BUFF 256 // RX buffer size
```

※本件の使用後は、設定を元に戻してください。

(下図、# 41～ 参照)



The screenshot shows the Arduino IDE interface with the file 'LPWA_IFTTT \$' open. The code in the editor is as follows:

```
14 // (c) 2019 Trillion-Node Study Group
15 // Released under the MIT license
16 // https://opensource.org/licenses/MIT
17 //
18 // Rev.00 2019/9/30 First release
19 //=====
20 //注意: 本スケッチは1時間あたり150Kbyte程度の送受信が発生します。通信容量にご注意ください
21 //=====
22 //use libraries
23 //Adafruit LIS3DH
24 //https://github.com/adafruit/Adafruit_LIS3DH
25 //※ Adafruit_LIS3DH.h
26 // uint8_t readRegister8(uint8_t reg);
27 // void writeRegister8(uint8_t reg, uint8_t value);
28 // をpublic:に移動する
29 //Adafruit Unified Sensor Driver
30 //https://github.com/adafruit/Adafruit_Sensor
31 //SmartEverything ST HTS221 Humidity Sensor
32 //https://github.com/ameltech/sme-hts221-library
33 //ClosedCube Arduino Library for ClosedCube OPT3001
34 //https://github.com/closedcube/ClosedCube_OPT3001_Arduino
35 //ST7032 - Arduino LiquidCrystal compatible library
36 //https://github.com/tomozh/arduino_ST7032
37 //MsTimer2 Library
38 //https://github.com/PaulStoffregen/MsTimer2
39 //=====
40 //=====
41 //※LTE-Mリーフへの1回の送信受信データが64byteを超える場合、SoftwareSerial.hで定義
42 // されている受信バッファサイズを変更する必要があります。
43 // 本スケッチでは256byteに変更することをお勧めします。
44 // 受信バッファサイズの変更は、SoftwareSerial.hを直接下記の通り変更します。
45 // #define _SS_MAX_RX_BUFF 64 → 256
46 //=====
```

The status bar at the bottom indicates 'COM74のArduino Pro or Pro Mini, ATmega328P (3.3V, 8 MHz)'.

4-3. スケッチ加筆修正事項

```

1 //=====
2 // Leafony Platform sample sketch
3 // Platform : LTE-M
4 // Processor : ATmega328P (3.3V /8MHz)
5 // Application : LPWA King demo
6 //
7 // Leaf configuration
8 // (1) AC04 LPWA King
9 // (2) AI01 4-Sensors
10 // (3) AI04 LCD
11 // (4) AP01 AVR MCU
12 // (5) AZ01 USB
13 //
14 // (c) 2019 Trillion-Node Study Group
15 // Released under the MIT license
16 // https://opensource.org/licenses/MIT
17 //
18 // Rev.00 2019/9/30 First release
19 //=====
20 //注意：本スケッチは1時間あたり150kbyte程度の送受信が発生します。通信容量にご注意ください
21 //=====
22 //use libraries
23 //Adafruit LIS3DH
24 //https://github.com/adafruit/Adafruit_LIS3DH
25 //※ Adafruit_LIS3DH.h
26 // uint8_t readRegister8(uint8_t reg);
27 // void writeRegister8(uint8_t reg, uint8_t value);
28 // をpublic:に移動する
29 //Adafruit Unified Sensor Driver
30 //https://github.com/adafruit/Adafruit_Sensor
31 //SmartEverything ST HTS221 Humidity Sensor
32 //https://github.com/ameletech/sme-hts221-library
33 //ClosedCube Arduino Library for ClosedCube OPT3001
34 //https://github.com/closedcube/ClosedCube_OPT3001_Arduino
35 //ST7032 - Arduino LiquidCrystal compatible library
36 //https://github.com/teensyduino/ST7032

```

1) スケッチを開き、# 78～に前述の IFTTT の「event_name」「Key」をコピー & ペーストします。

```

//=====
// IFTTT Webhook
// Webhook のイベント名と KEY 番号を記載
//=====
const char event_name[] = "LTE-M Leaf"; (例) // Webhook のイベント名
const char key[] = " "; // Webhook の KEY 番号

```

2) # 84～、SIM カードの情報を入力してください。(アクセスポイントネーム APN 名、ユーザー名、パスワード)

※スケッチに書き込み済み

```

//=====
// LTE 接続設定
// 契約した SIM カードの情報を入力する
//=====
const char apn[] = "soracom.io"; //APN 名
const char user_name[] = "sora"; //ユーザー名

```

```
const char password[] = "sora";          //パスワード
```

3) #72～、デバックやリーフの設定を変更できます。

- ・今回は LCD リーフを使用するため、「#define LCD」は常に「1」にしてください。 ※スケッチに設定済み
- ・「#define USB_EN 1」で、「USB Leaf からのシリアルモニタ表示への動作ログ出力 ON」
(※TeraTerm 等で動作状況が見られる設定。)

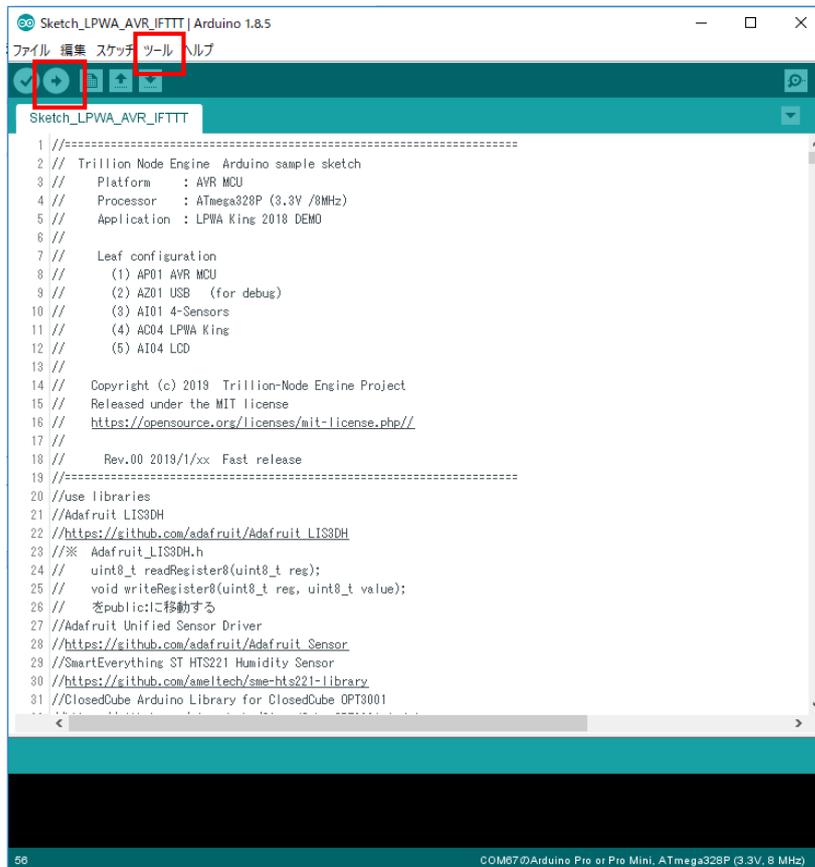
```
//=====
// デバック&リーフ設定
//=====
```

```
#define LCD 1          // LCD Leaf use:1  Grove leaf use:0
#define USB_EN 0     // USB Leaf use: 1 ( USB Leaf からログを出力する場合)
```

4) 設定を保存します。

4-4. スケッチ書き込み

- 1) 「USB」リーフにケーブルを繋ぎ、PC の COM ポートに接続します。
- 2) Arduino IDE で、「ツール」→「ボード」から、「Arduino Pro or Pro Mini」と「ATmega328P(3.3V,8MHz)」を選択。
次に、「ツール」→「シリアルポート」から、COM 番号を選択します。(AVR MCU 使用)



3) 「→」を押してスケッチを書き込み、以下の表示になったら、書き込み完了です。

```
7 // Leaf configuration
8 //   (1) AP01 AVR MCU
9 //   (2) AZ01 USB (for debug)
10 //   (3) AI01 4-Sensors
11 //   (4) AC04 LPWA King
12 //   (5) AI04 LCD
13 //
14 // Copyright (c) 2019 Trillion-Node Engine Project
15 // Released under the MIT license
16 // https://opensource.org/licenses/mit-license.php/
17 //
18 // Rev.00 2019/1/xx Fast release
19 //=====
20 //use libraries
21 //=====
```

ボードへの書き込みが完了しました。

```
avrdude: input file C:\Users\k\AppData\Local\Temp\arduino_build_649608/Sketch_LPWA_AVR_IFTTT_leafony.ino.hex contains 28010 bytes
avrdude: reading on-chip flash data:
Reading | ##### | 100% 11.81s
avrdude: verifying ...
avrdude: 28010 bytes of flash verified
avrdude done. Thank you.
```

COM87のArduino Pro or Pro Mini, ATmega328P (3.3V, 8 MHz)

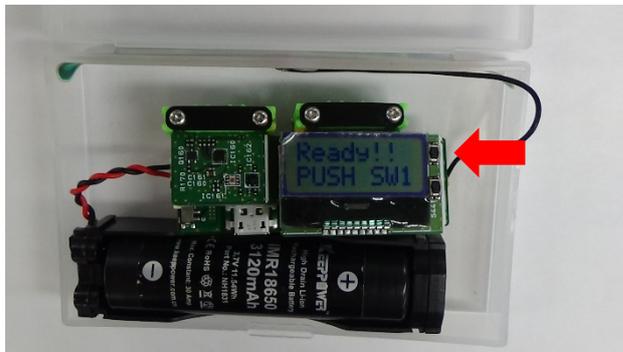
5. IFTTT との通信

5-1. 「LTE-M」リーフのデータ送信

- 1) 「LTE-M」リーフの電源を ON にすると、BOOT が始まります。



- 2) 「PUSH SW1」の表示が出るので、「LCD」リーフの上のボタンを押してください。



- 3) データ送信が始まります。



- 4) t:温度 h:湿度 l:照度 v:電圧が表示されます。



5-2. Google スプレッドシートの表記確認

1) Google スプレッドシートにセンサ情報が表示されているか確認します。

IFTTT_LTE ☆

ファイル 編集 表示 挿入 表示形式 データ ツール アドオン ヘルプ 最終編集: 数秒前

100% ¥ % .0 .00 123 Arial 10 B I U A

	A	B	C	D	E	F	G	H
1	日時	センサーデータ測定日時	温度	湿度	照度	電圧		
2	April 16, 2019 at 06:45PM	19/04/16,18:45:38	33.02	23.25	298	3.87		
3	April 16, 2019 at 06:48PM	19/04/16,18:48:39	33.43	23.01	289	3.87		
4	April 16, 2019 at 06:48PM	19/04/16,18:46:39	33.35	23.21	294	3.87		
5	April 17, 2019 at 11:17AM	19/04/17,11:17:35	33.54	31.3	701	3.84		
6	April 17, 2019 at 11:18AM	19/04/17,11:18:34	33.06	29.98	769	3.87		
7	April 17, 2019 at 11:19AM	19/04/17,11:19:34	33.1	29.95	242	3.87		
8	April 17, 2019 at 04:55PM	19/04/17,16:55:25	28.11	35.91	150	3.84		
9	April 17, 2019 at 05:22PM	19/04/17,17:21:55	28.41	47.84	463	3.84		
10								
11								
12								
13								
14								
15								
16								
17								
18								
19								
20								

データを送信した時分の値（測定日時、温度、湿度、照度、電圧）が記載されていたら完成です。

電波強度によっては、通信の途中でも途切れることがありますのでご注意ください。

5-3. 「LTE Mリーフデモ IFTTT の例」動画

<https://youtu.be/940YMXANZfy>

