	Innovation et Développement Durable		1 <sup>ère</sup> STI2D
		<b>Tutoriel IOT - Arduino MKR1000</b> Envoi d'une notification d'alarme avec IFTTT		
		Séquence 4 : Les produits connectés		Tutoriel

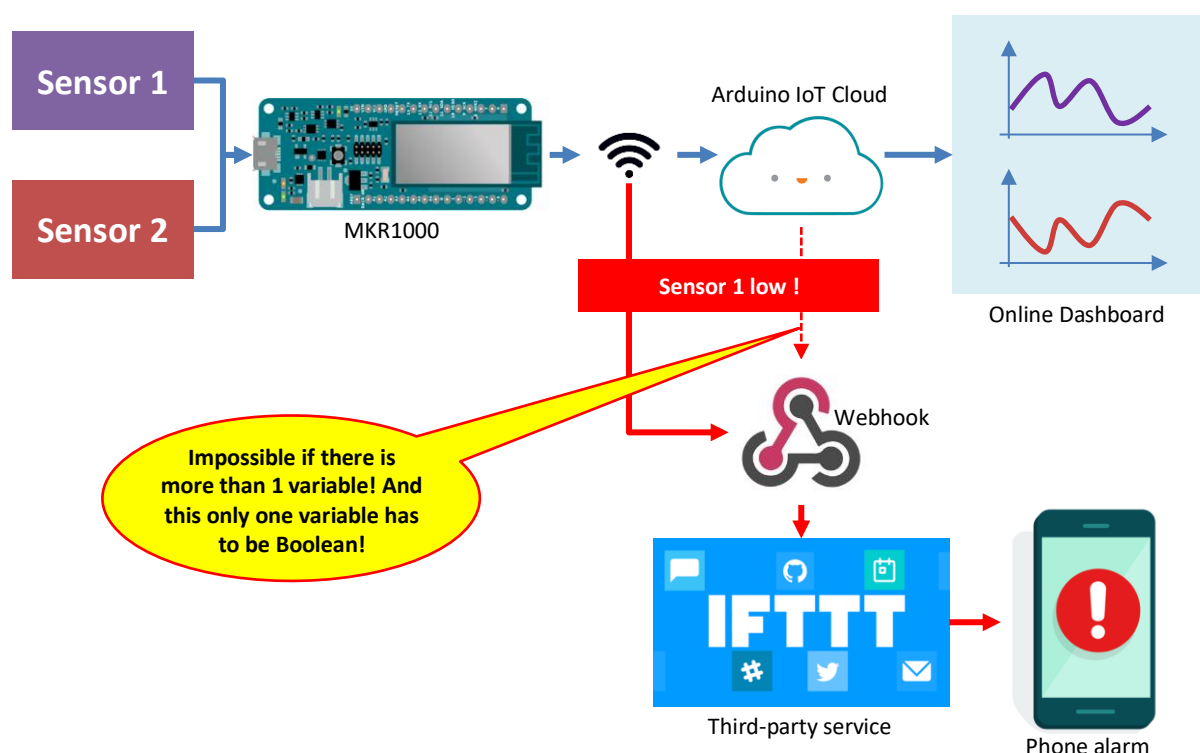
## 1. Introduction

### 1.1. Le projet

Arduino IoT Cloud est une formidable plateforme ! Associées aux cartes connectées comme la MKR1000, il est désormais possible de réaliser un objet connecté très facilement, et d'utiliser le « Dashboard » intégré à la plateforme pour interagir via Internet. Tout cela se fait gratuitement, le code et la mise en œuvre sont très simple, car la couche « Networking » est assistée de manière très intuitive.

Lire à distance la courbe d'évolution d'un capteur analogique, ou bien commander la commutation d'un relais avec son téléphone ne prend que quelques minutes d'installation et de code, sans réelle difficulté.

**Pour ce projet, on souhaite afficher les données de 2 capteurs sur le dashboard Arduino IoT Cloud, et déclencher une alerte sur un téléphone si la valeur du capteur 1 est en dessous d'un seuil.**



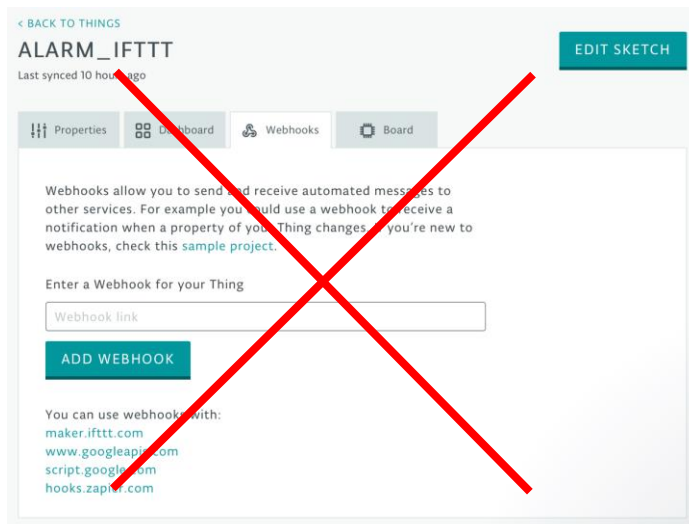
### 1.2. Problème technique à résoudre

Arduino IoT Cloud propose un service « Webhooks » qui permet d'envoyer les données de capteurs sur des applications tierces (Google Spreadsheets, IFTTT, Zapier, ...).







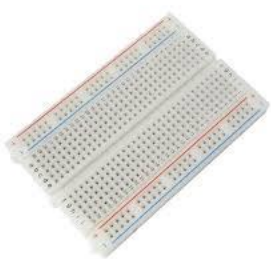
En l'absence de tout réglage, il semble que le Webhook soit déclenché à chaque fois qu'une donnée est envoyée sur le Cloud Arduino. A moins de faire du traitement côté service tiers, s'il y a plus d'un capteur à monitorer, l'alerte va se déclencher à chaque fois que les données vont changer. Ce n'est pas ce que nous désirons pour ce projet.




Le but de ce tutoriel est donc de **déclencher l'alerte et de réaliser l'envoi au webhook depuis le code Arduino**, sans passer par le service webhooks de la plateforme, dont nous avons énoncé les limitations apparentes plus tôt.

On conserve en outre la facilité de l'affichage des données sur le dashboard Arduino IoT Cloud.

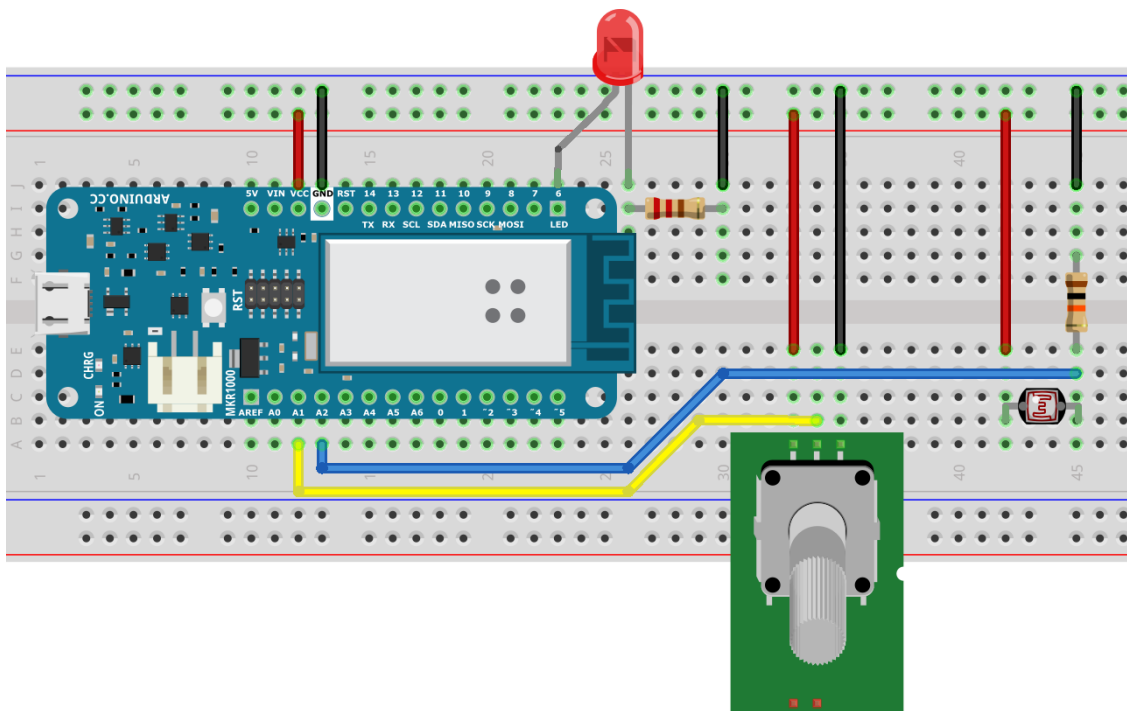


## 2. Matériel et logiciels nécessaires

Item	Illustration	Fonction
Arduino MKR1000		<ul style="list-style-type: none"> <li>Exécuter le programme du projet</li> <li>Envoyer les données en Wifi</li> </ul>
Potentiomètre		Capteur #1
LDR		Capteur #2
LED Rouge		Afficher l'alerte
Résistance 220 $\Omega$		Câblage de la LED
Résistance 10 k $\Omega$		Réaliser un pont diviseur de tension pour lire la LDR
Breadboard		Réaliser les branchements

Jumpers		Réaliser les branchements
Arduino IoT Cloud <a href="https://create.arduino.cc/iot/">https://create.arduino.cc/iot/</a>		<ul style="list-style-type: none"> <li>Interface de programmation (IDE) en ligne</li> <li>Visualisation des données (Dashboard)</li> </ul>
IFTTT <a href="https://ifttt.com">https://ifttt.com</a>		Visualiser une alarme à distance (sur un téléphone par exemple)

### 3. Schéma de câblage

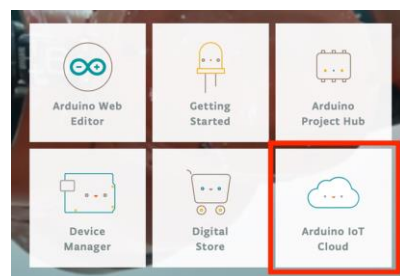


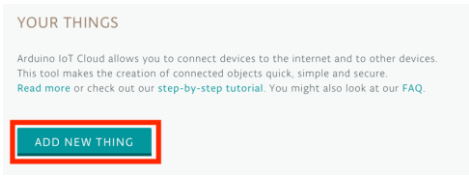
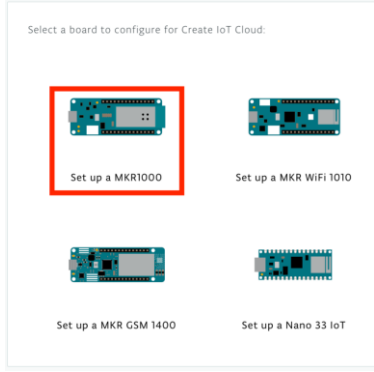
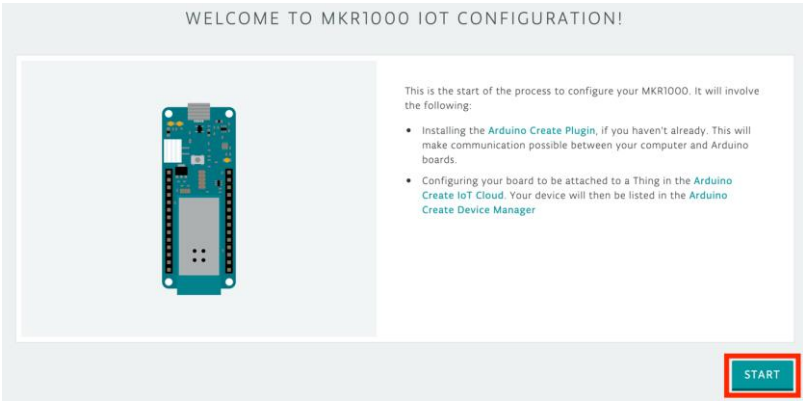
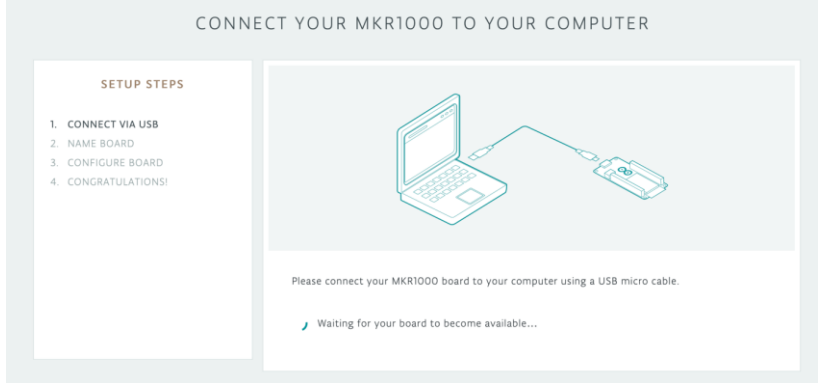
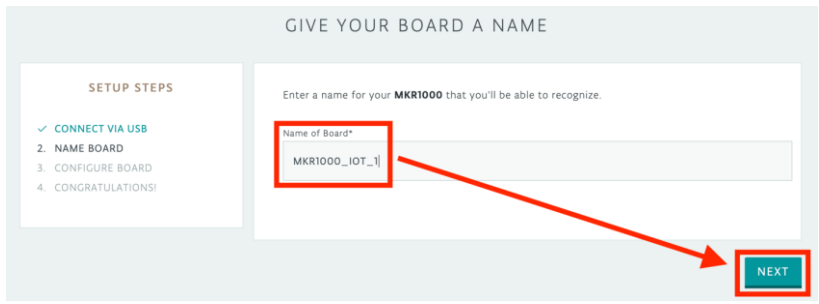
### 4. Arduino IoT Cloud

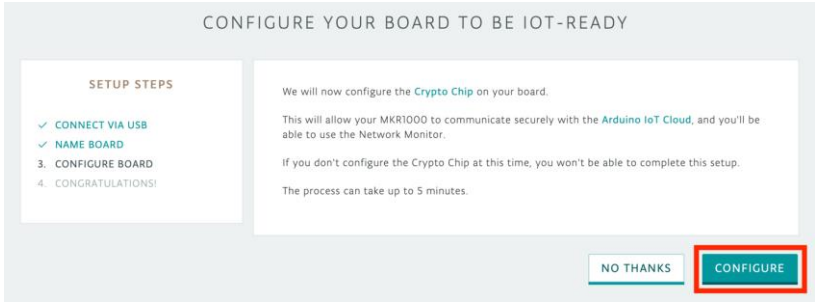
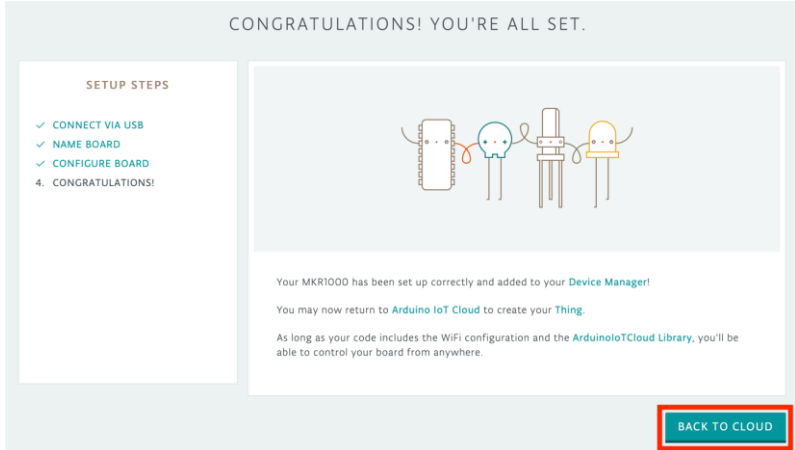
#### 4.1. Préparation

Se rendre sur <https://create.arduino.cc/iot/things> et créer un nouveau compte.

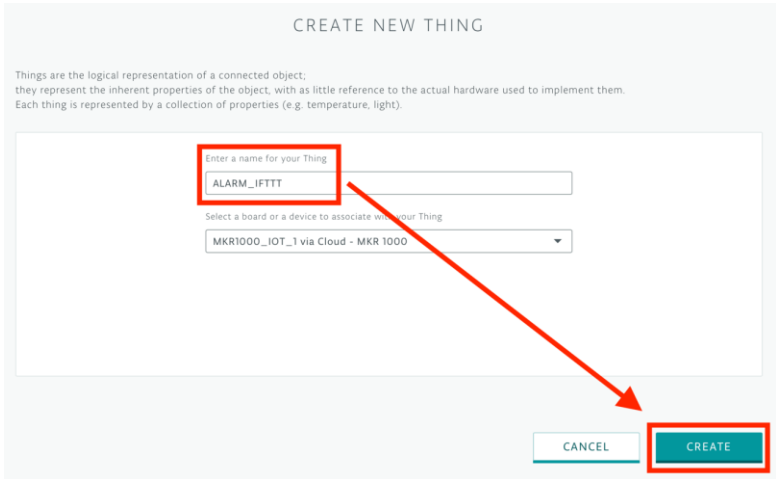
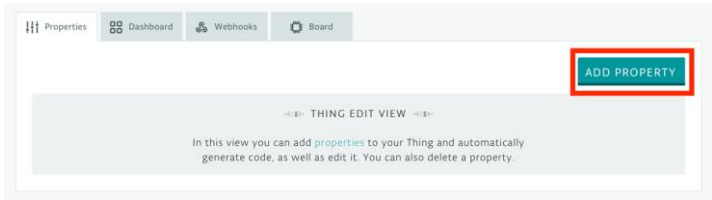
Cliquer sur  et choisir « Arduino IoT Cloud ».



Ajouter un nouvel objet « Thing ».	
Choisir la carte « MKR1000 ».	
Suivre la procédure (il faudra notamment installer le logiciel « Arduino Create Plugin »).	
Brancher le MKR1000 à l'ordinateur.	
Donner un nom à la carte.	

<p>Suivre la procédure (cette étape permet de configurer la puce de sécurité de la carte).</p>	
<p>Revenir à l'écran principal de l'interface du Cloud.</p>	

## 4.2. Création d'un objet connecté (« Thing ») et de propriétés (« Properties »)

<p>Donner un nom au nouvel objet (« <b>Thing</b> ») que nous allons créer pour ce projet, et choisir la carte que nous venons d'installer.</p>	
<p>Ajouter une nouvelle propriété (dans notre cas, il s'agit d'un des 2 capteurs du projets).</p> <p><b>Note :</b> chaque fonction du projet (capteur, actionneur...) doit être associée à une propriété (« <b>Property</b> »).</p>	

Configurer la propriété comme montré ci-contre (il s'agit de la propriété associée au potentiomètre).

**Note :** Les permissions sont particulièrement importantes :

- **Read & Write** : Lecture et écriture : il est possible de contrôler un objet depuis le cloud

- **Read Only** : Lecture seul : seul l'affichage des données sur le cloud est réalisé.

Dans notre cas, on souhaite simplement afficher la valeur d'un capteur, on choisira donc « Read Only ».

The screenshot shows the IFTTT property configuration form. The fields are as follows:

- Name: pot
- Variable Name: pot
- Type: Int
- Min value: 0
- Max value: 1023
- Permission: Read Only (selected)
- Update: When the value changes (selected)
- Delta: 0
- History: Show history visualization (checked)

Buttons at the bottom: CANCEL and ADD PROPERTY (highlighted with a red box).

Réaliser les 2 étapes précédentes afin de créer une propriété associée à la LDR, puis cliquer sur « Edit Sketch » afin de commencer à coder.

The screenshot shows the IFTTT dashboard for the sketch named ALARM\_IFTTT. The top bar includes a link to BACK TO THINGS and an EDIT SKETCH button (highlighted with a red box). Below the sketch name, there are tabs for Properties, Dashboard, Webhooks, and Board. The Properties tab is active, showing a table of properties:

NAME	TYPE	UPDATE	PERMISSION
ldr	Int	On change	RO
pot	Int	On change	RO

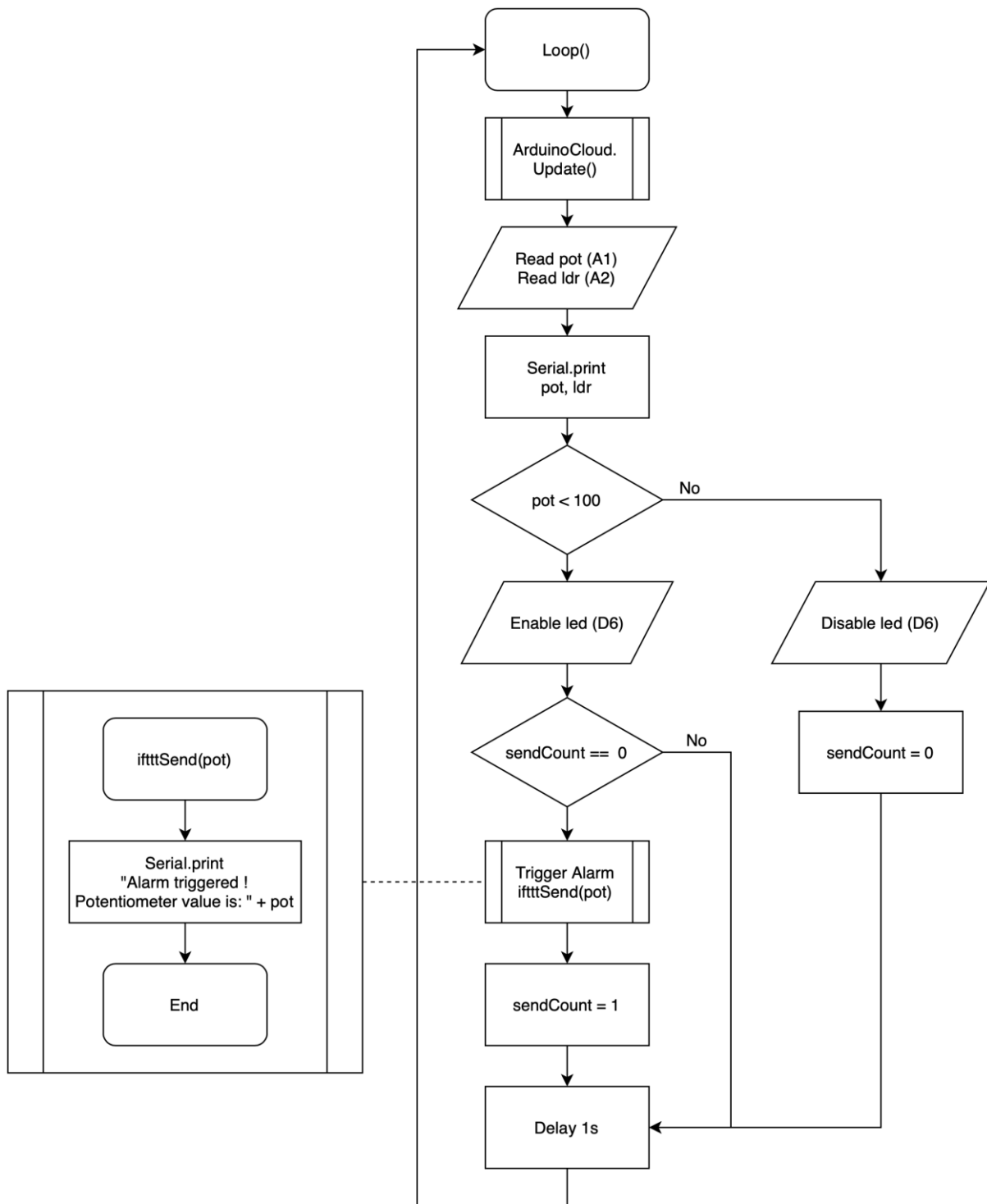
An ADD PROPERTY button is located at the top right of the table. The first two rows of the table are highlighted with a red dashed border.

### 4.3. Code pour afficher les valeurs des capteurs sur le Cloud

Dans un premier temps, nous allons réaliser le code pour :

- Lire les valeurs analogiques des capteurs
- Envoyer les données sur le Cloud
- Déclencher une alerte lorsque la valeur du potentiomètre est trop basse (*pour le moment envoyer un message sur le moniteur série + allumage d'une LED*)

#### 4.3.1. Algorithme



### 4.3.2. Codage sur le « Web Editor »

```
/*
  The following variables are automatically generated and updated when changes are made to the Thing
  properties

  int ldr;
  int pot;

  Properties which are marked as READ/WRITE in the Cloud Thing will also have functions
  which are called when their values are changed from the Dashboard.
  These functions are generated with the Thing and added at the end of this sketch.
*/

int sendCount = 0;

#define potPin 1
#define ldrPin 2
#define ledPin 6

#include "thingProperties.h"

void setup() {

  Serial.begin(9600);
  delay(1500);
  pinMode(6, OUTPUT);

  // Defined in thingProperties.h
  initProperties();
  // Connect to Arduino IoT Cloud
  ArduinoCloud.begin(ArduinoIoTPreferredConnection);

  /*
   The following function allows you to obtain more information
   related to the state of network and IoT Cloud connection and errors
   the higher number the more granular information youâ€™ll get.
   The default is 0 (only errors).
   Maximum is 4
  */
  setDebugMessageLevel(2);
  ArduinoCloud.printDebugInfo();
}

void loop() {

  ArduinoCloud.update();

  pot = analogRead(1);
  ldr = analogRead(2);

  Serial.println("pot = " + String(pot) + " ldr = " + String(ldr));

  if (pot < 100) {
    digitalWrite(ledPin, HIGH);
    if (sendCount == 0) {
      iftttSend(pot);
      sendCount = 1;
    }
  }

  else {
    digitalWrite(ledPin, LOW);
    sendCount = 0;
  }

  delay(1000);
}

void iftttSend(int val) {

  String str_val = String(val);
  Serial.println("Alarm triggered ! Potentiometer value is: " + str_val);
}
```

Dans l'éditeur Web, copier/coller le code précédent (remplacer le code par défaut).

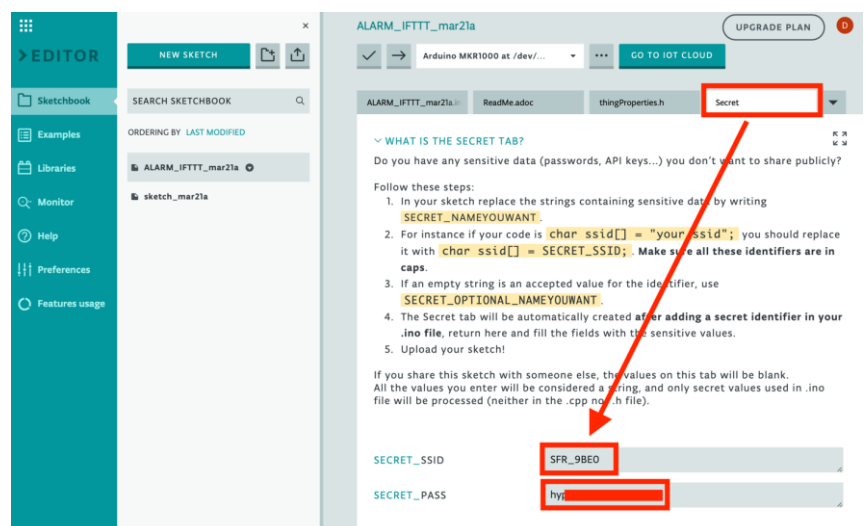
**Note :** le code par défaut comprend la majorité des commandes et fonctions propres à l'utilisation du cloud. Le code copié l'enrichi avec les éléments propres au projet.

```

1  /*
2  The following variables are automatically generated and updated when changes are
3  made to the Cloud Thing.
4  int ldr;
5  int pot;
6
7  Properties which are marked as READ/WRITE in the Cloud Thing will also have functions
8  which are called when their values are changed from the Dashboard.
9  These functions are generated with the Thing and added at the end of this sketch.
10 */
11
12 int sendCount = 0;
13
14 #define potPin 1
15 #define ldrPin 2
16 #define ledPin 6
17
18 #include "thingProperties.h"
19
20 void setup() {
21   Serial.begin(9600);
22   delay(1500);
23   pinMode(6, OUTPUT);
24
25   // Defined in thingProperties.h
26   initProperties();
27   // Connect to Arduino IoT Cloud
28   ArduinoCloud.begin(ArduinoIoTPreferredConnection);
29
30   /*
31   */
32 }

```

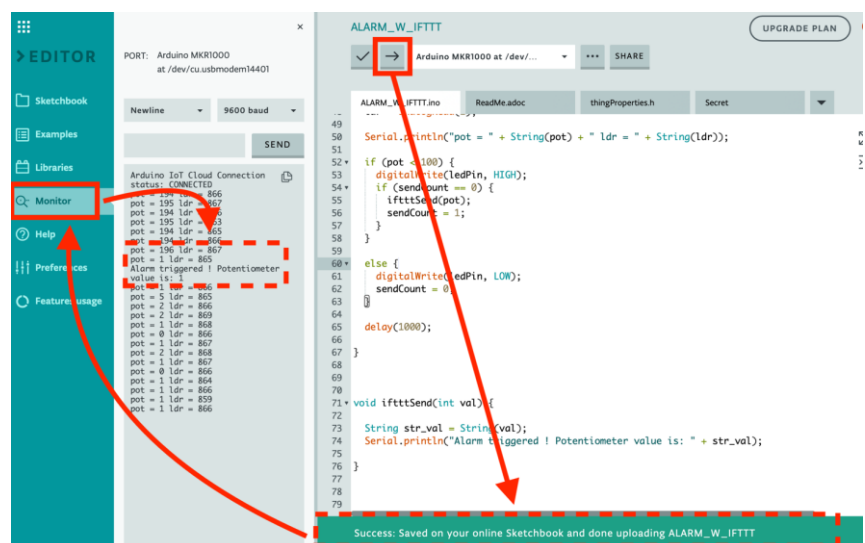
Aller dans l'onglet « Secret » et compléter les informations de connexion à votre point de connexion WIFI.



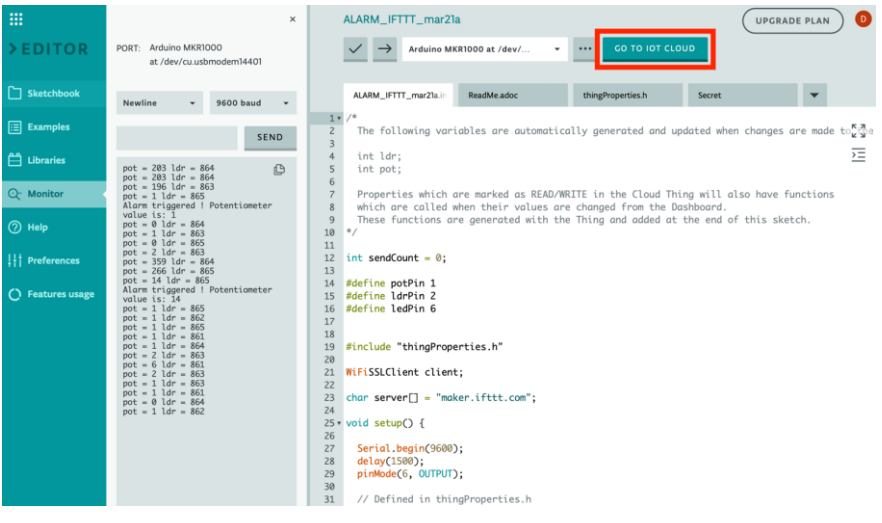

Cliquer sur le bouton « UPLOAD », puis attendre que le programme soit compilé et envoyé sur la carte.

Une fois le programme envoyé (bandeau vert), cliquer sur « Monitor » afin d'accéder aux messages envoyés sur le port série.

Actionner le potentiomètre, en dessous de 100, un message devrait indiquer que l'alarme a été déclenché.

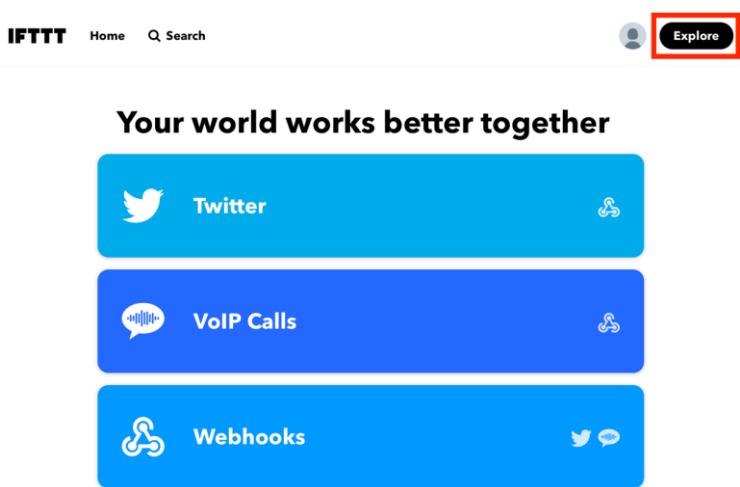


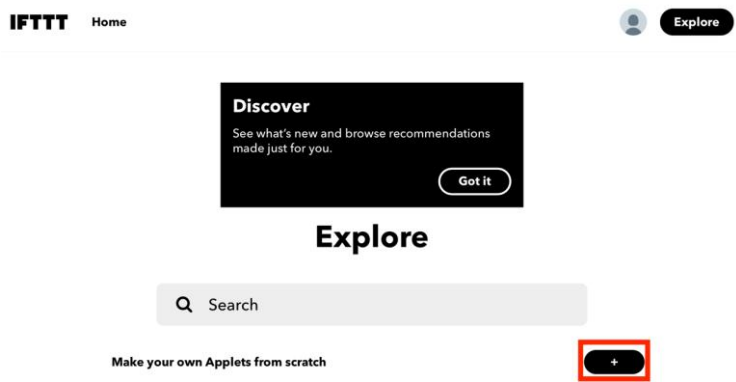
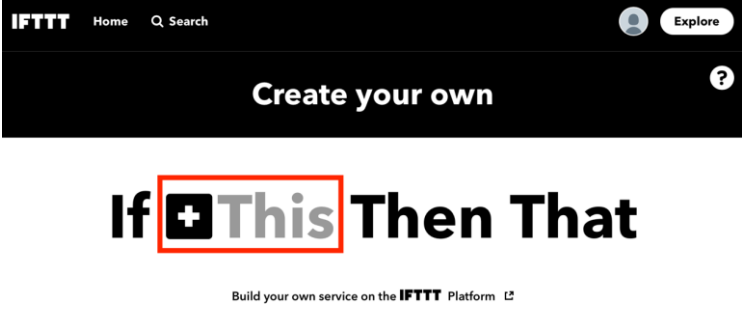
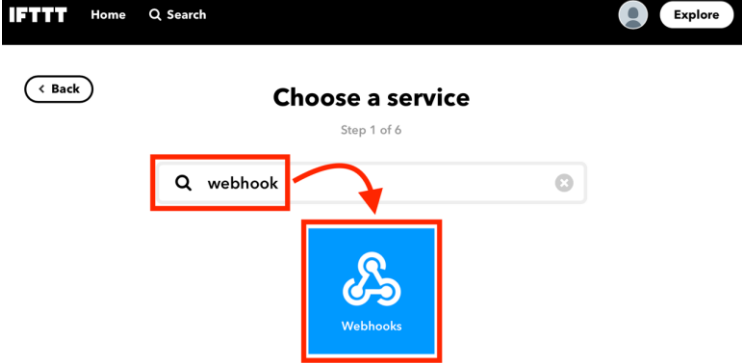
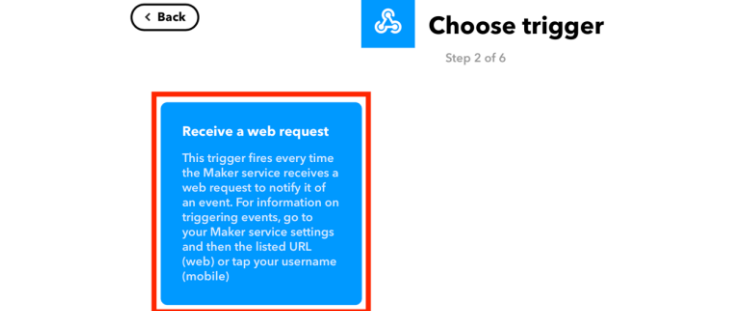
### 4.3.3. Dashboard

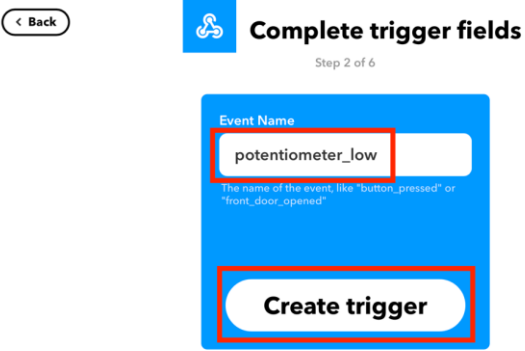

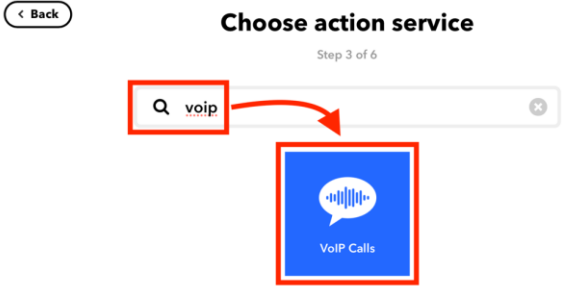
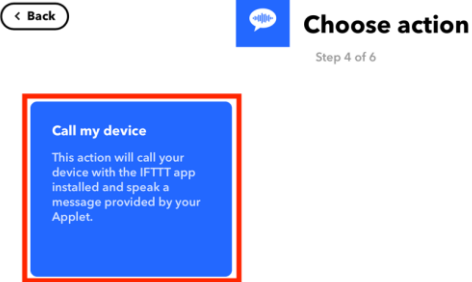
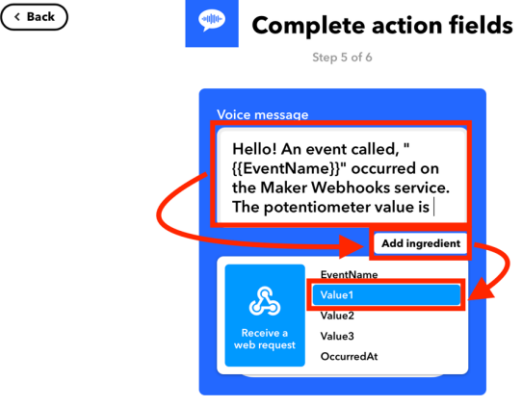
<p>Cliquer « GO TO IOT CLOUD » afin d'accéder aux données envoyées en ligne.</p>	
<p>Aller dans l'onglet « Dashboard », observer les valeurs évoluer... <b>Success!</b></p> <p>Afin de réaliser l'alarme sur le téléphone, revenir sur l'éditeur de sketches.</p>	

## 5. Déclenchement de l'alarme : IFTTT

**IFTTT** (if this then that) est un service web gratuit permettant à ses utilisateurs de créer des chaînes d'instruction simples appelées **applets**. Une applet est **déclenchée** par des changements qui interviennent au sein de services. [Wikipédia](https://fr.wikipedia.org/wiki/IFTTT)

<p>Aller à l'adresse <a href="https://ifttt.com">https://ifttt.com</a> et créer un compte.</p> <p>Cliquer ensuite sur « Explore ».</p>	
--	--

<p>Afin de créer une nouvelle « Applet », cliquer sur « + ».</p>	 <p>The screenshot shows the IFTTT Home page. At the top, there's a navigation bar with the IFTTT logo, 'Home', a search bar, and a user profile icon with an 'Explore' button. Below this, a 'Discover' section offers recommendations. The main heading is 'Explore', followed by a search bar. At the bottom, there's a section 'Make your own Applets from scratch' with a red box highlighting a '+' icon.</p>
<p>Cliquer sur « + This » afin de créer un nouvel événement déclencheur.</p>	 <p>The screenshot shows the 'Create your own' page on IFTTT. It features the 'If + This Then That' logo, with the '+' icon highlighted by a red box. Below the logo, it says 'Build your own service on the IFTTT Platform'.</p>
<p>Saisir « Webhooks » dans le moteur de recherche, et cliquer sur l'icône « Webhook ».</p> <p><b>Note :</b> un webhook est une fonction de rappel HTTP définie par l'utilisateur, qui récupère et stocke les données issues d'un événement, en général externe à votre application (<a href="https://www.inbenta.com/fr/blog/webhook-chatbot-api-tal/">https://www.inbenta.com/fr/blog/webhook-chatbot-api-tal/</a>).</p>	 <p>The screenshot shows the 'Choose a service' page (Step 1 of 6). A search bar contains the text 'webhook'. Below the search bar, a red box highlights the 'Webhooks' service icon, which is also pointed to by a red arrow from the search bar.</p>
<p>Continuer en cliquant sur l'icône « Receive a web request ».</p>	 <p>The screenshot shows the 'Choose trigger' page (Step 2 of 6). A red box highlights the 'Receive a web request' trigger option. The text below the trigger explains that it fires every time the Maker service receives a web request to notify it of an event.</p>

<p>Saisir le nom de l'évènement qui va donner lieu à l'alerte : « potentiometer_low », puis cliquer sur « Create trigger ».</p>	
<p>Cliquer sur « + That » afin de créer une action associée à l'évènement défini précédemment.</p>	
<p>Saisir « voip » dans le moteur de recherche, et cliquer sur l'icône « VoIP Calls ».</p> <p><b>Note :</b>  <i>La voix sur IP, ou « VoIP » pour Voice over IP, est une technique qui permet de transmettre la voix sur des réseaux IP filaires (câble/ADSL/fibre optique) ou non (satellite, Wi-Fi et réseaux mobiles), qu'il s'agisse de réseaux privés ou d'Internet.</i>  <a href="https://fr.wikipedia.org/wiki/Voix_sur_IP">https://fr.wikipedia.org/wiki/Voix_sur_IP</a></p>	
<p>Cliquer sur « Call my device ».</p>	
<p>Au besoin, modifier le texte du message à envoyer, puis rajouter la phrase « The potentiometer value is ».</p> <p>Cliquer ensuite sur « Add ingredient » et sélectionner « Value1 ».</p>	

Si tout va bien, le texte devrait prendre les variables « EventName » et « Value1 » en gris.

Cliquer ensuite sur « Create action ».

**Note :**

« Value1 », « Value2 », ... correspondent aux valeurs transmises par le Webhook. Ce sont des objets **JSON**, nous allons nous servir de « Value1 » pour transmettre la valeur du potentiomètre.

< Back



## Complete action fields

Step 5 of 6

Voice message

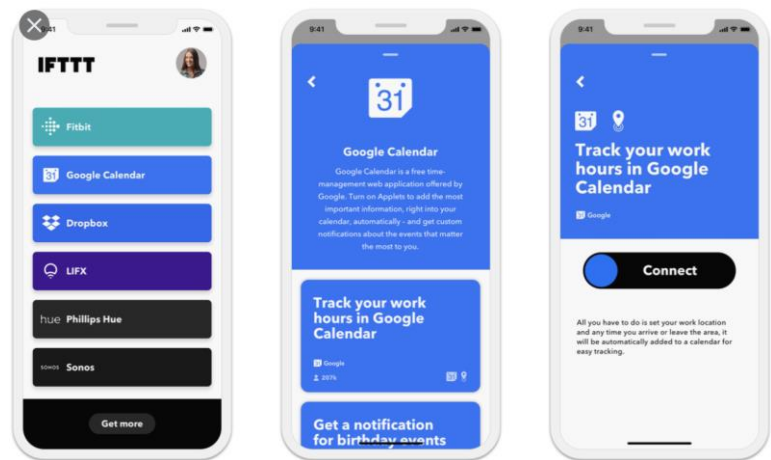
Hello! An event called, "  
EventName" occurred on  
the Maker Webhooks service.  
The potentiometer value is  
Value1

Add ingredient

Create action

A ce stade, il faut télécharger l'application IFTTT pour smartphone (Google Play ou Apple Store).

L'installer et connecter au compte IFTTT.



Revenir sur la page d'accueil d'IFTTT en cliquant sur l'icône en haut à gauche, puis cliquer sur « Webhooks ».



Home Q Search



Explore

## Your world works better together



Twitter

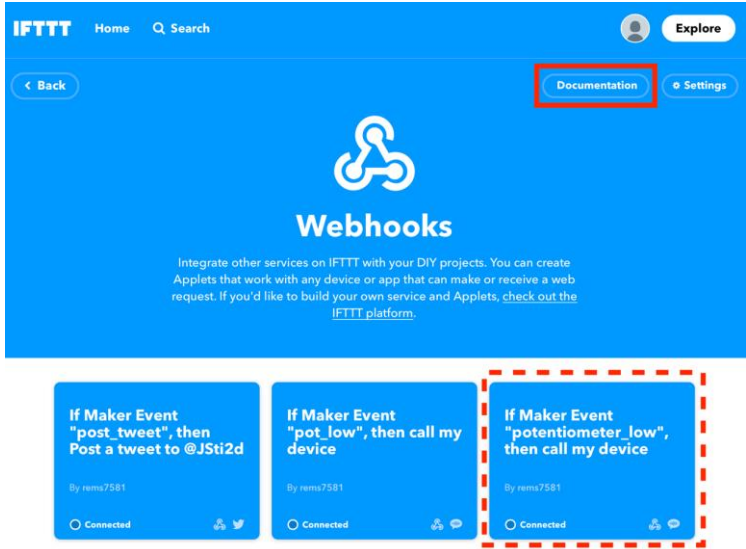
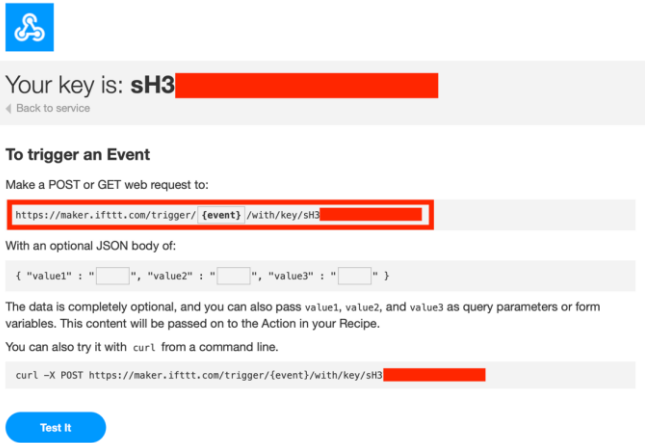


VoIP Calls



Webhooks



<p>On peut observer que le Webhook créé précédemment est bien présent.</p> <p>Cliquer sur « Documentation ».</p>	
<p>Cette page contient l'URL de la <b>requête HTTP</b> utilisée pour déclencher l'alerte.</p> <p>Pour la tester, la copier dans un navigateur Internet, puis :</p> <ul style="list-style-type: none"> <li>• Remplacer « {event} » par « <b>potentiometer_low</b> »</li> <li>• Ajouter à la fin « <b>?value1=255</b> »</li> </ul> <p>Si tout va bien, vous devriez recevoir un appel sur votre téléphone...</p>	

## 6. Finalisation du projet

### 6.1. Modification et importation de la bibliothèque « WIFI101 »

Il faut maintenant effectuer la requête HTTP depuis la carte MKR1000.

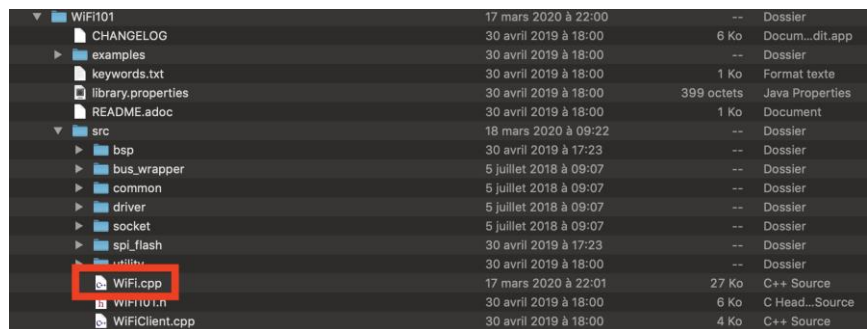
Les bibliothèques importées dans le code précédent font appel à la bibliothèque installée par défaut « **WIFI101** », qui permet notamment d'envoyer des requêtes HTTP.

**Cette bibliothèque nécessite cependant une modification pour fonctionner avec IFTTT.**

Avec le logiciel Arduino installé sur l'ordinateur, retrouver dans l'explorateur de fichier le dossier « **WiFi101** » :

Mac & PC : Documents/Arduino/librairies

Copier ce dossier sur le bureau, et ouvrir le fichier « **WiFi.cpp** » avec un éditeur de texte.



Remplacer la ligne **317** par :

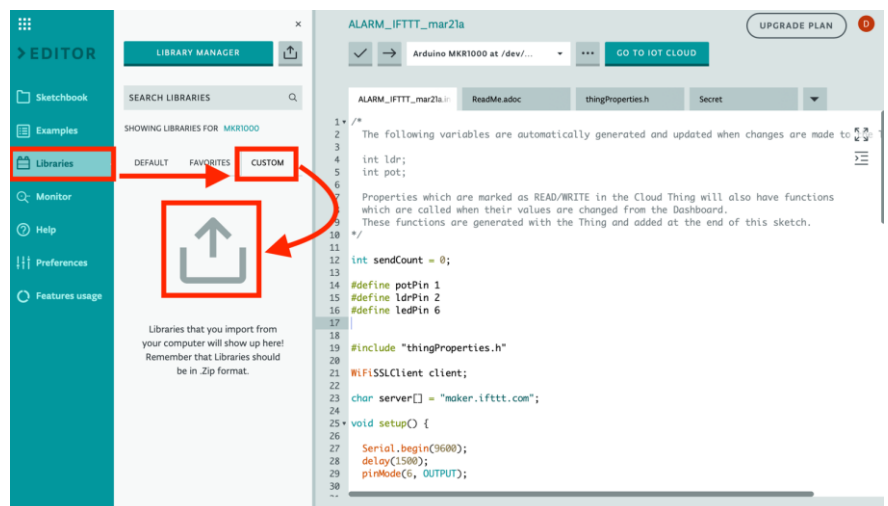
```
m2m_ssl_set_active_ciphersuites(SSL_CIPHER_RSA_WITH_AES_128_CBC_SHA | SSL_CIPHER_RSA_WITH_AES_128_CBC_SHA256 |
SSL_CIPHER_RSA_WITH_AES_128_GCM_SHA256 | SSL_CIPHER_RSA_WITH_AES_256_CBC_SHA |
SSL_CIPHER_RSA_WITH_AES_256_CBC_SHA256);
```

Puis sauvegarder le fichier.

Compresser le dossier WiFi101 afin de créer un fichier **WiFi101.zip**.

Source : <https://github.com/arduino-libraries/WiFi101/pull/274/commits/2c66be041a005236ebd95938afdfaf957a717349>

Revenir sur l'éditeur WEB, cliquer sur « **Librairies** » puis onglet « **Custom** » et importer « **WiFi101.zip** » précédemment créé.



## 6.2. Code

```
/*
  The following variables are automatically generated and updated when changes are made to the Thing
  properties

  int ldr;
  int pot;

  Properties which are marked as READ/WRITE in the Cloud Thing will also have functions
  which are called when their values are changed from the Dashboard.
  These functions are generated with the Thing and added at the end of this sketch.
*/

int sendCount = 0;

#define potPin 1
#define ldrPin 2
#define ledPin 6

#include "thingProperties.h"

WiFiSSLClient client;

char server[] = "maker.ifttt.com";

void setup() {

  Serial.begin(9600);
  delay(1500);
  pinMode(6, OUTPUT);

  // Defined in thingProperties.h
  initProperties();
  // Connect to Arduino IoT Cloud
  ArduinoCloud.begin(ArduinoIoTPreferredConnection);

  /*
  The following function allows you to obtain more information
  related to the state of network and IoT Cloud connection and errors
  the higher number the more granular information you'll get.
  The default is 0 (only errors).
  Maximum is 4
  */
  setDebugMessageLevel(2);
  ArduinoCloud.printDebugInfo();
}

void loop() {

  ArduinoCloud.update();

  pot = analogRead(1);
  ldr = analogRead(2);

  Serial.println("pot = " + String(pot) + " ldr = " + String(ldr));

  if (pot < 100) {
    digitalWrite(ledPin, HIGH);
    if (sendCount == 0) {
      iftttSend(pot);
      sendCount = 1;
    }
  }
  else {
    digitalWrite(ledPin, LOW);
    sendCount = 0;
  }

  delay(1000);
}
```

*Suite page suivante...*

```

void iftttSend(int val) {

  String str_val = String(val);

  Serial.println("Alarm triggered ! Potentiometer value is: " + str_val);

  String data = "{\"value1\": \"" + str_val + "\"}";

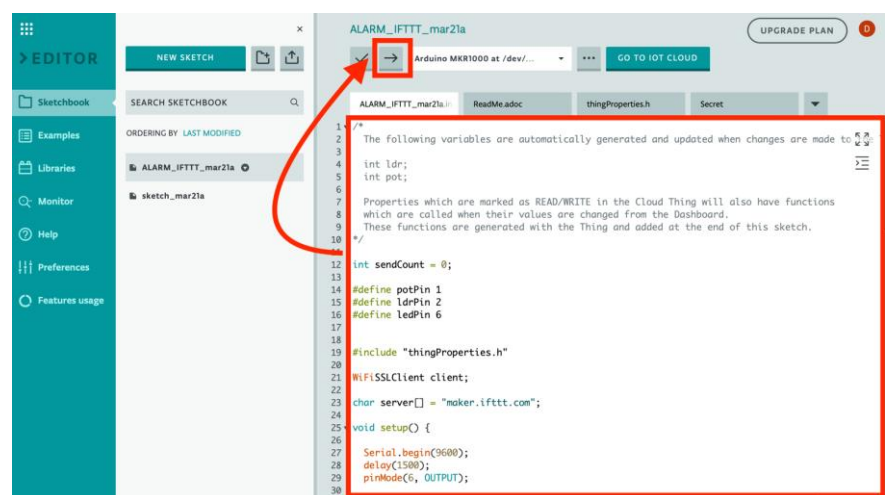
  // Connexion au serveur IFTTT
  Serial.println("Starting connection to server...");
  if (client.connectSSL(server, 443)) {
    Serial.println("Connected to server IFTTT, ready to trigger alarm...");
    // Make a HTTP request:
    client.println("POST /trigger/potentiometer_low/with/key/<YOUR_KEY> HTTP/1.1"); // Replace <YOUR_KEY>
    by your OWN IFTTT key
    client.println("Host: maker.ifttt.com");
    client.println("Content-Type: application/json");
    client.print("Content-Length: ");
    client.println(data.length());
    client.println();
    client.print(data);
    Serial.println("IFTTT alarm triggered !");
  }
  else {
    Serial.println("Connection at IFTTT failed");
  }
}
}

```

Dans l'éditeur Web, copier/coller le code précédent (remplacer le code précédent).

Modifier la ligne 90 avec votre propre clé IFTTT.

Télécharger le code sur la carte.



Une fois le programme envoyé (bandeau vert), cliquer sur « Monitor » afin d'accéder aux messages envoyés sur le port série.

Actionner le potentiomètre, en dessous de 100, un message devrait indiquer que l'alarme a été déclenché... et que l'alerte est partie sur IFTTT... *vous devriez recevoir un appel sur le téléphone !!*

*How about that?*

