GUIDE de la carte AFF IoT

Version 1.0.2

Guide de l'AFF IoT

Votre guide à la carte AFF IoT

Bienvenue au *Guide de l'AFF IoT*

Ce livret contient les informations nécessaires à l'exploration des circuits du KIT AFF IoT.

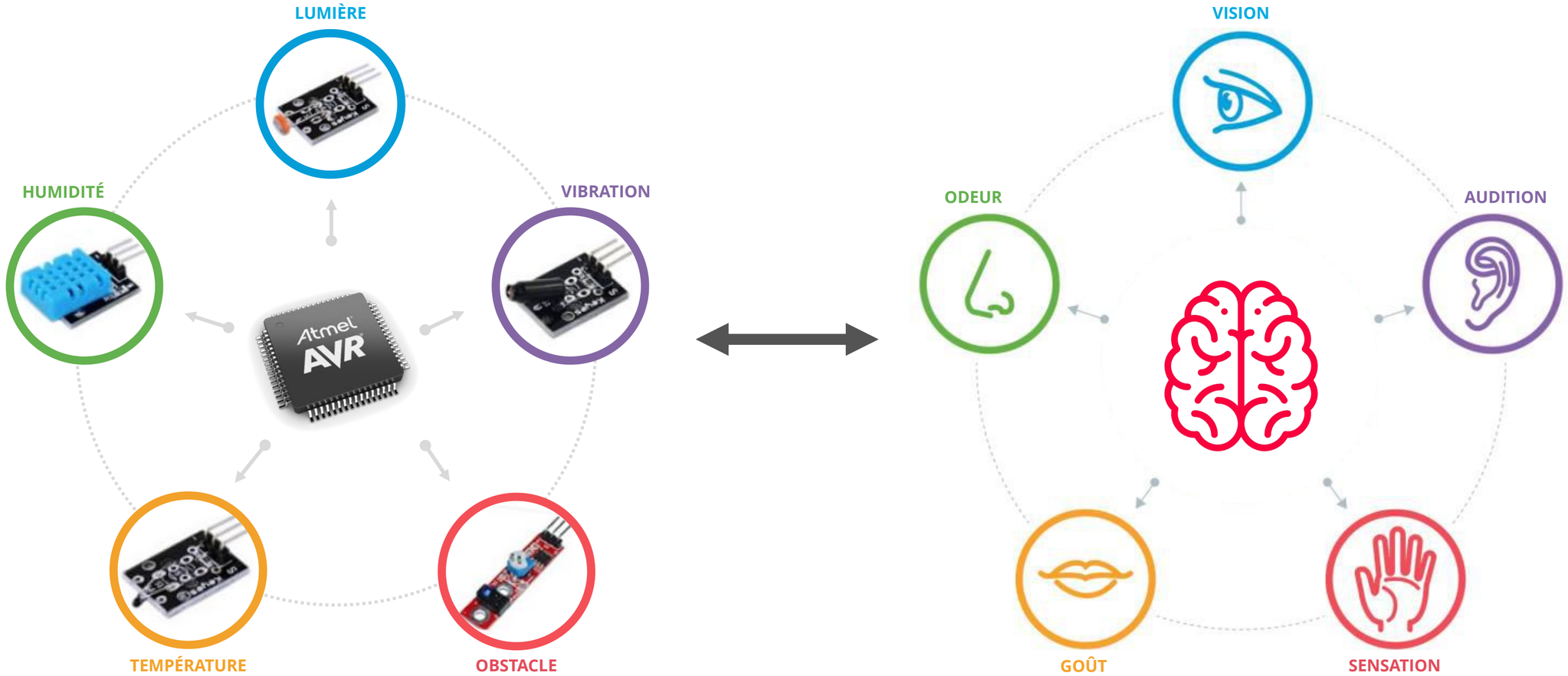
A la fin de ce livret, vous saurez comment créer vos propres projets et expériences.

COMMENÇONS ALORS!

Table des Matières

Concept fondamental

L'analogie entre l'Homme et le Microcontrôleur



Un **microcontrôleur** est un ordinateur présenté dans un seul circuit intégré, dédié à l'utilisation et à l'exécution de tâches. Il contient une mémoire, des périphériques d'entrée / sortie programmables ainsi que le cerveau. Il recueille les données dans les capteurs l'environnement, traite les données collectées selon les besoins du programmeur et effectue les actions en conséquence (contrôle-moteur, allumeur, etc.).

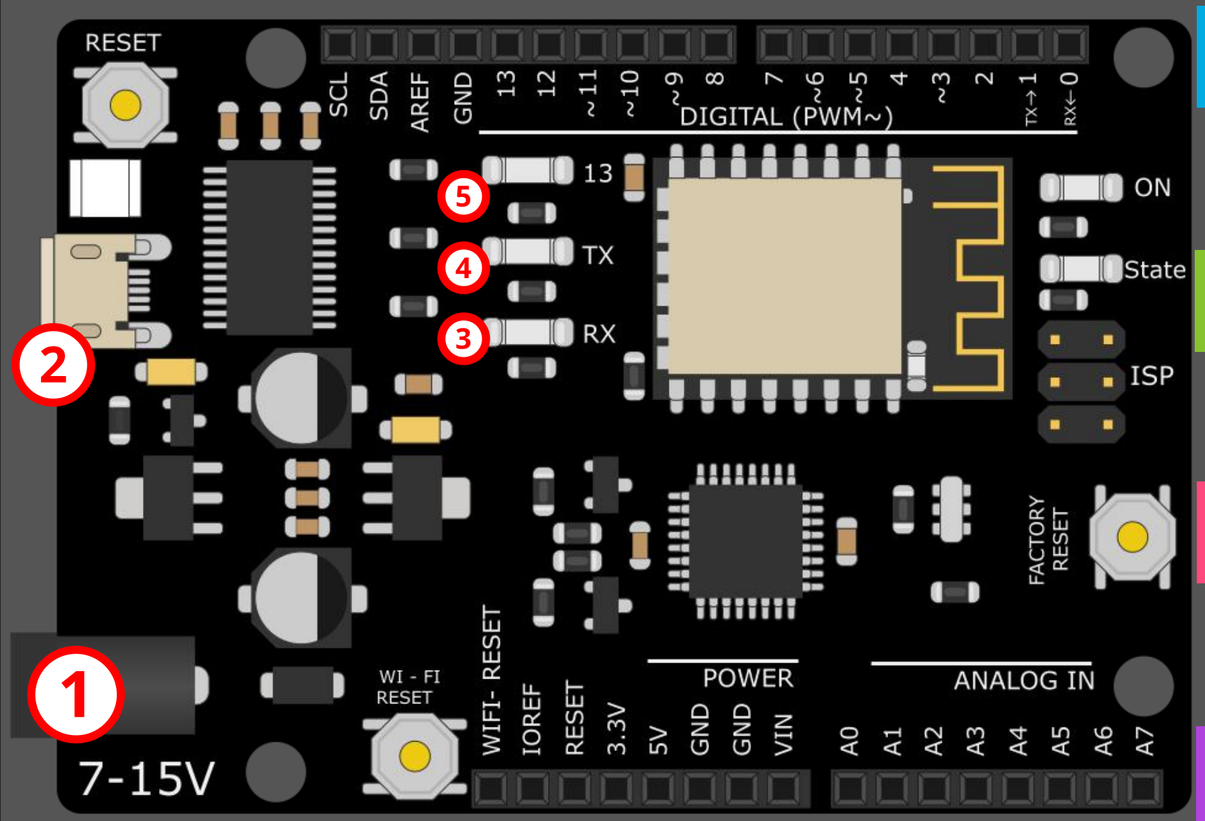
Le cerveau est un organe qui sert de centre au système nerveux. Il recueille des données de l'environnement en utilisant les organes des sens; traite les données collectées en fonction des besoins actuels et de circonstances antérieures similaires, puis envoie des ordres pour exécuter les actions en conséquence (promenade, conversation, etc.)

Qu’est ce que la carte AFF IoT?

- L’Internet des objets (Internet of Things IoT) comprend tout ce qui est connecté à Internet, qui peux être contrôlé/surveillé à tout moment et en tout lieu. Techniquement, il consiste à intégrer des capteurs et des appareils connectés à l'Internet via des réseaux fixes ou sans fil.
- Les appareils IoT font partie d'un concept plus vaste: la domotique. Il comprend les systèmes d'éclairage, de chauffage, de climatisation, de médias et de sécurité. Les avantages à long terme comprennent les économies d'énergie en garantissant automatiquement que les lumières et les composants électroniques sont éteints lorsqu'ils ne sont pas utilisés.
- La carte AFF IoT est une carte de microcontrôleur compatible avec Arduino qui comprend un module Wi-Fi intégré.

La carte **AFF IoT**

Composants



1

Entrée d’alimentation (Connecteur cylindrique)

Alimente la carte et fonctionne avec un adaptateur ou une batterie 9V ou 12V. La gamme d'alimentation est de 7V à 15V.

2

Port USB

Alimente la carte et connecte la carte à un ordinateur via USB.

3-4

LEDs (RX: Réception / TX: Transmission)

Ces LEDs indiquent que le microcontrôleur reçoit / envoie des bits de données de / vers l'ordinateur.

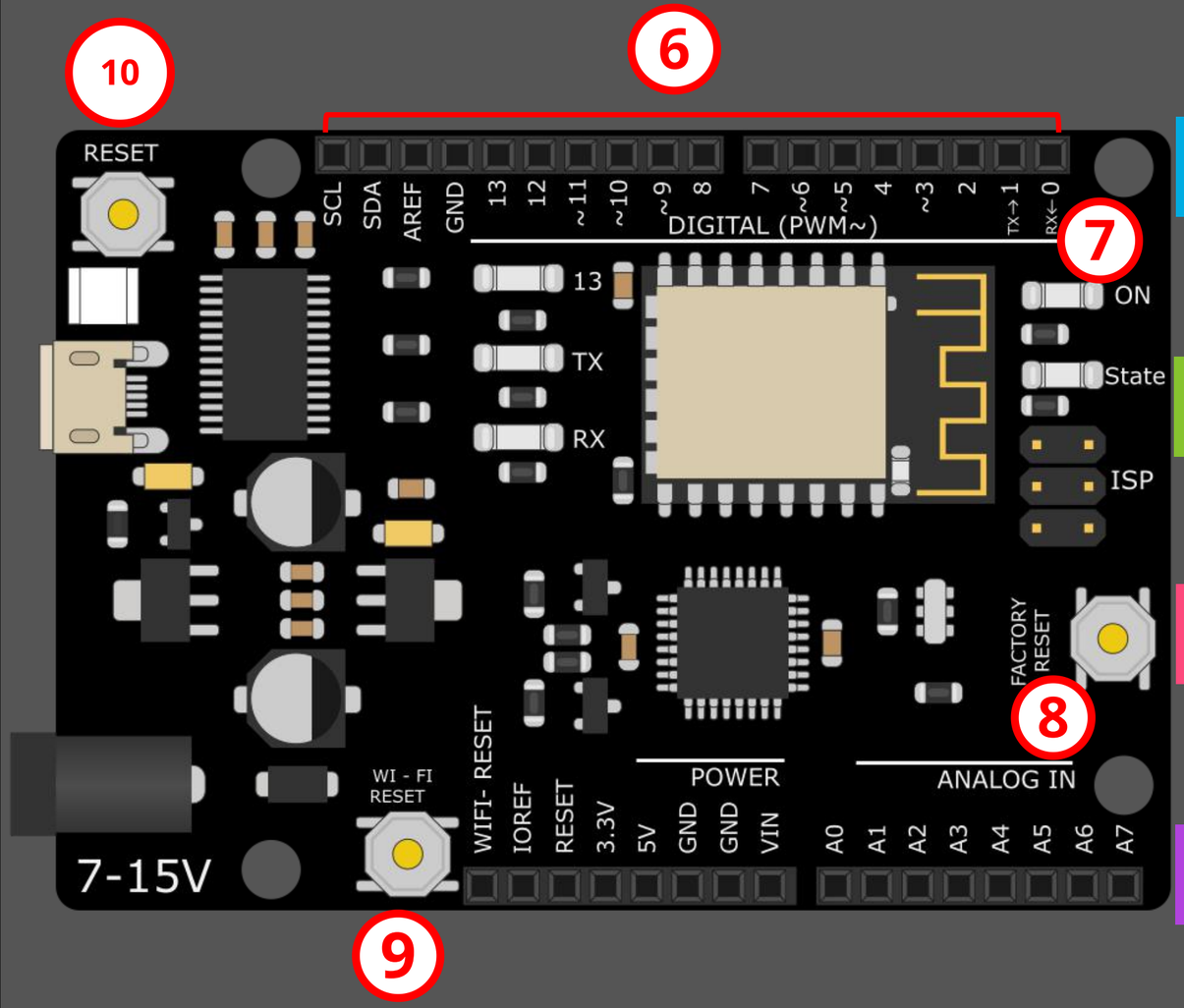
5

LED 13 (LED intégré)

Cette LED est liée au croquis pour montrer si le programme fonctionne correctement. Il est connecté via une résistance à la broche numérique 13.

La carte **AFF IoT**

Composants



6 Broches (Terre, Numérique, Rx, Tx)

Diverses broches peuvent être utilisées pour les entrées, les sorties numériques, et la communication série.

7 LEDs (Rouge et Bleu)

Le rouge est un simple indicateur d'alimentation. Le bleu indique l'état de la carte (configuré ou non).

8 Bouton “FACTORY RESET”

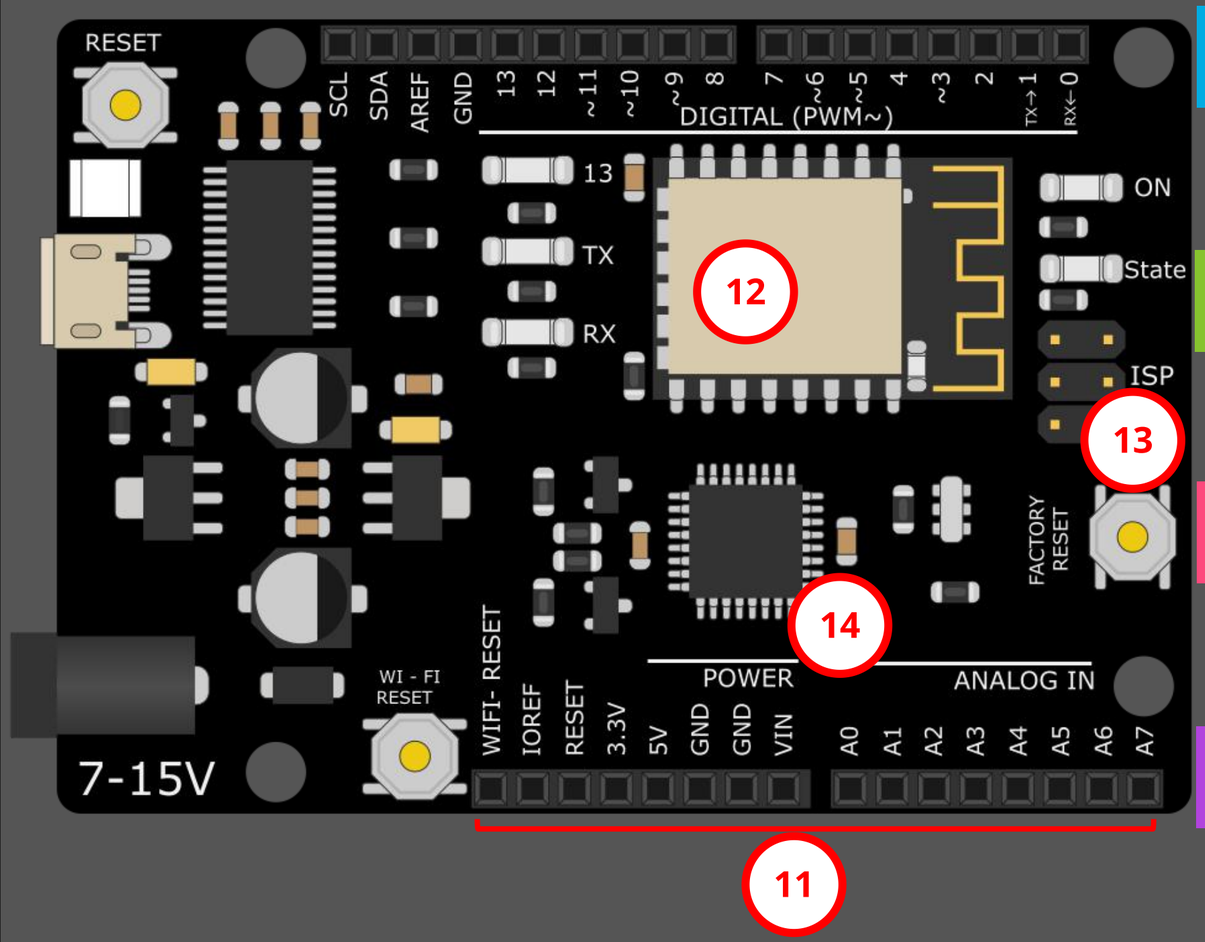
Buton réinitialisation d'usine. Ce bouton réinitialise la carte, supprime le Wi-Fi et les données du compte. (Appuyez pendant 5 secondes pour réinitialiser)

9-10 Bouton “Wi-Fi RESET” et Bouton “RESET”

Le premier redémarre le module Wi-Fi de la carte et se reconnecte au serveur, le second redémarre le microcontrôleur (redémarre le code) (Appuyer une seule fois).

La carte **AFF IoT**

Composants



11 Broches (“ANALOG IN”, “POWER”)

Différentes broches peuvent être utilisées pour les entrées analogiques, les sorties (3,3 V et 5 V) et la terre.

12 Le Module Wi-Fi ESP

Microcontrôleur avec protocole TCP / IP intégré qui donne à la carte AFF IoT un accès à Internet.

13 Les broches ICSP (In Circuit Serial Programming)

Ces broches sont utilisées pour programmer la carte via un kit d’outils de programmation.

14 Le Microcontrôleur AVR

Il s’agit du microcontrôleur de la carte AFF, similaire à celui utilisé dans la carte Arduino UNO traditionnelle.

Télécharger l'EDI Arduino

Accès Internet

1

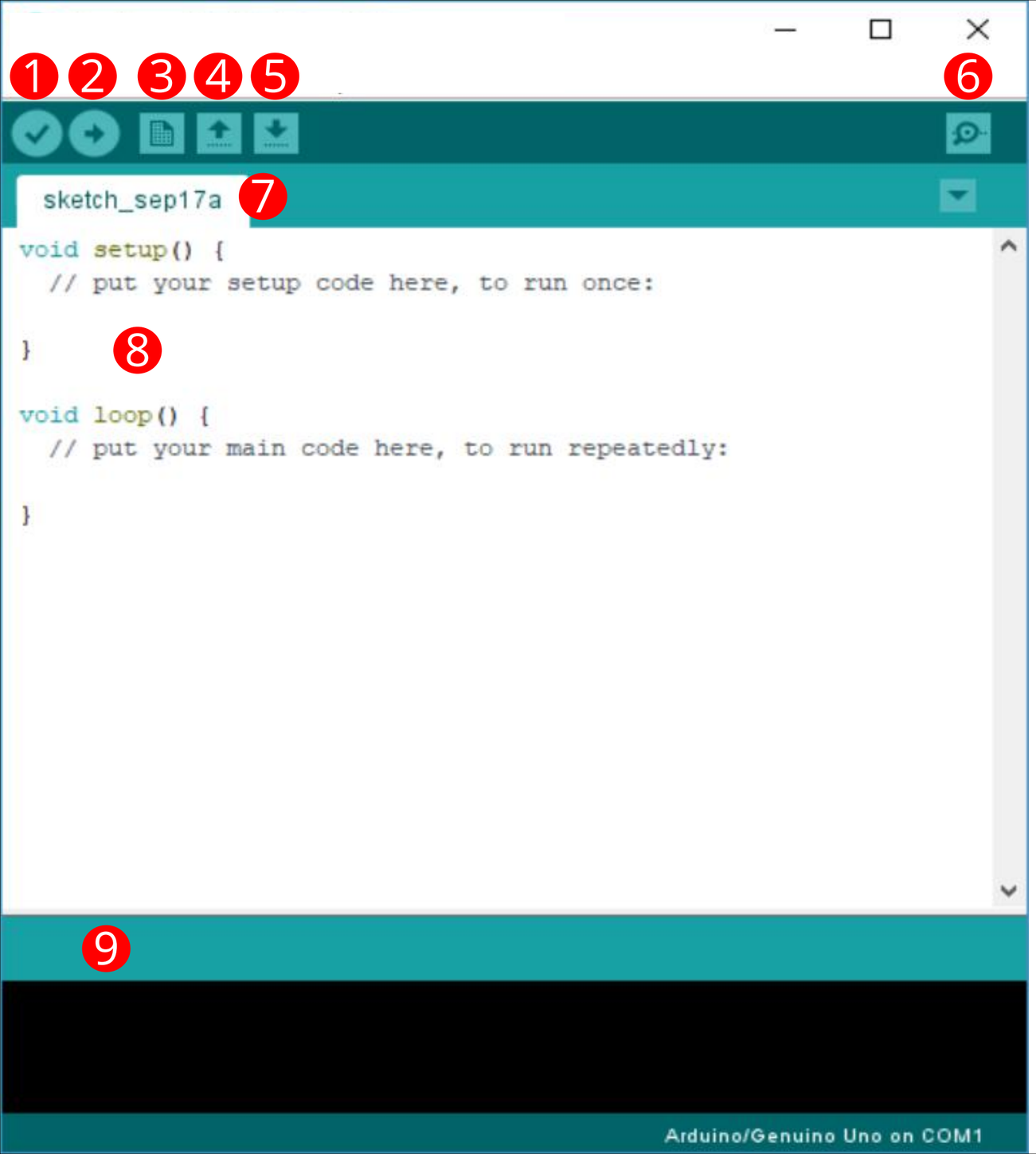
Téléchargez la dernière version du logiciel Arduino sur www.arduino.cc (c'est gratuit!).
Tapez l'URL suivante dans la barre d'adresse du navigateur: www.arduino.cc/en/Main/Software

2

Téléchargement

Choisissez le package d'installation du système d'exploitation approprié (Windows, MAC ou Linux).

* EDI: Environment de Development Intégrée



À propos de l'**EDI Arduino**

Ouvrez le logiciel Arduino sur votre ordinateur et vous verrez une page similaire à celle de gauche.

À propos de l' EDI Arduino

1

Vérifier

Compile et vérifie le code. Il détecte les erreurs telles que les points-virgules ou les parenthèses manquants.

2

Téléverser

Envoie le code à la carte AFF. Lorsque vous cliquez dessus, les deux LEDs jaunes (LEDs RX et TX) de la carte clignotent rapidement.

3

Nouveau

Ouvre une nouvelle fenêtre de code qui ne contient que deux fonctions : setup () et loop (), qui seront expliquées plus tard.

4

Ouvrir

Ouvre un croquis existant.

5

Enregistrer

Enregistre le croquis actuel ouvert.

6

Moniteur Série

Ouvre une fenêtre affichant toutes les informations série transmises par la carte AFF IoT. C'est très utile pour le débogage.

7

Nom du croquis

Affiche le nom du croquis

8

Zone de Code

Zone dédiée à la composition du code du croquis.

9

Zone de Message

Notifie le programmeur des erreurs dans le code une fois le code est vérifié.

```
void setup() {
  // put your setup code here, to run once:

}
```

La fonction `setup()`

Cette fonction est utilisée pour configurer et initialiser toutes les broches, interfaces et valeurs initiales pour les variables utilisées.
Cette fonction est exécutée une fois lors de la mise sous tension de la carte et lorsque vous appuyez sur le bouton de réinitialisation “RESET”.

La fonction `loop()`

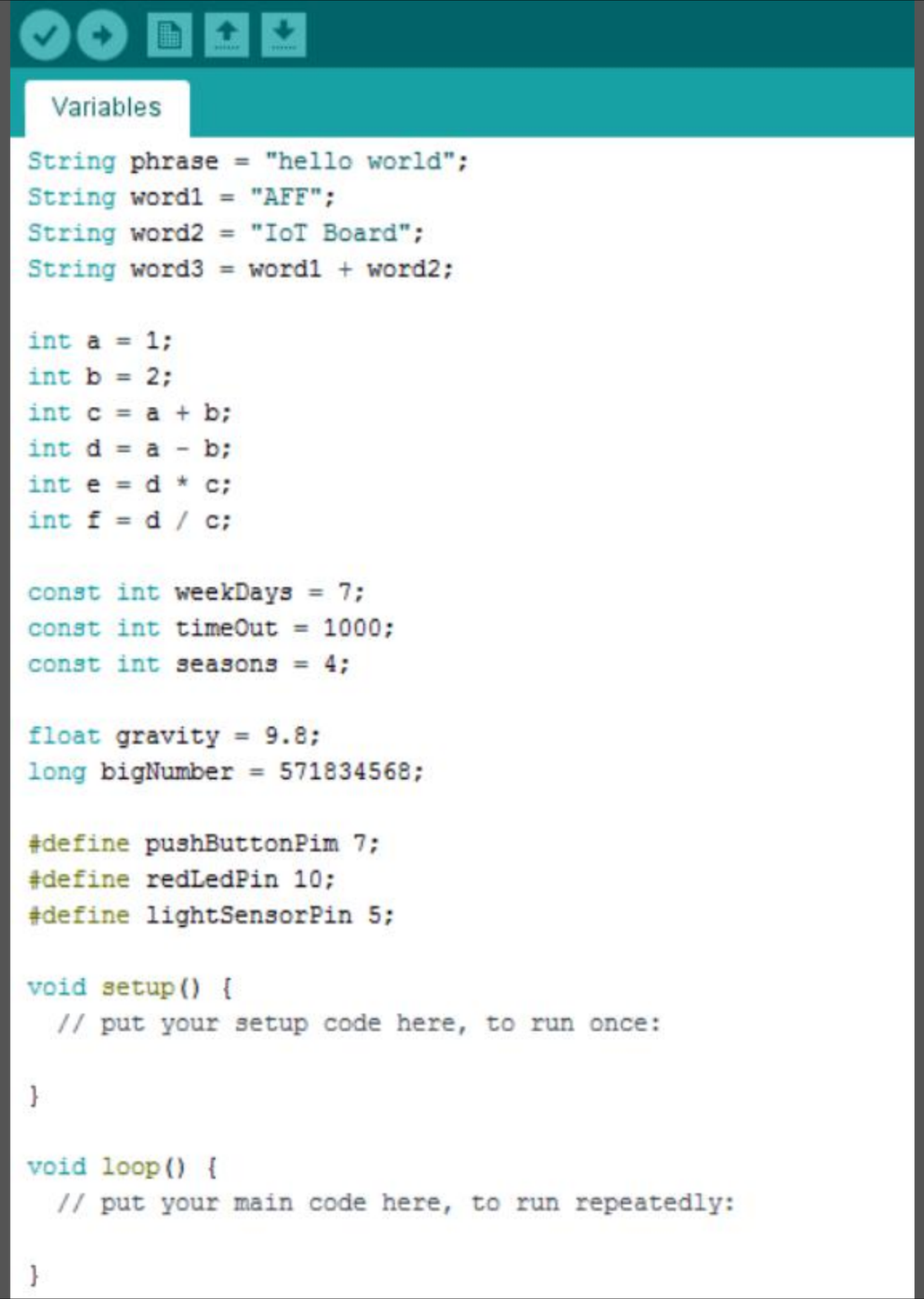
Toutes les lignes de code écrites dans cette fonction s'exécuteront de manière répétée tant que la carte sera alimentée.
Cette fonction est utilisée pour exécuter le code indéfiniment.

```
void loop() {
  // put your main code here, to run repeatedly:

}
```

Les fonctions essentielles de l’Arduino

```
// c'est pour les commentaires d'une seule ligne
// c’est bien de mettre une description au
// sommet et avant tout ce qui est "compliqué"
```

```
/ * c'est pour les commentaire multi-lignes
    Comme ça…
    Et ça….*/
```



Apprendre la programmation et les bases du codage

Dans la section suivante, tous les concepts de programmation nécessaires pour programmer la carte AFF IoT sont expliqués à l'aide des exemples illustratifs.

```
int a = 1;

int b = 2;

int c = a + b;

int d = a - b;

int e = d * c;

int f = d / c;
```

Variables Entières

- Une variable est comme un "compartiment", elle contient temporairement des nombres ou d'autres valeurs.
- Toute variable peut être assignée avec une valeur particulière en utilisant l'opérateur égal “=“.
- Toutes les opérations arithmétiques peuvent être utilisées et le résultat obtenu peut être stocké dans une autre variable.

Variables de type String (chaine de charactères)

- Ce type de variables est destiné à stocker les caractères composants un mot ou une phrase.
- En tant que type entier, les variables de type String peuvent être combinées pour composer une autre variable de type String à l'aide de l'opérateur plus «+».

```
String phrase = "hello world";

String word1 = "AFF";

String word2 = "IoT Board";

String word3 = word1 + word2;
```

```
const int weekDays = 7;

const int timeOut = 1000;

const int seasons = 4;
```

Variables Constantes

- Ce type de variables peut être attribué une fois et ne peut être modifié nulle part dans le code.
- Elle peut être déclarée en ajoutant le mot "const" avant son type.
- Essayer d'affecter une autre valeur à une variable constante provoque une erreur de compilation.

Variables Définies

- C'est similaire aux variables constantes.
- Elles sont généralement utilisées pour attribuer des numéros de broches aux composants physiques.
- Pas besoin de déclarer son type, elles peuvent être assignées directement
- Elles sont définies sans l'opérateur égal "="

```
#define pushButtonPim 7;

#define redLedPin 10;

#define lightSensorPin 5;
```

```
float gravity = 9.8;

long bigNumber = 571834568;
```

Variables longues et décimales

La variable décimale est déclarée en utilisant le type nommé «float».
La variable “long” est similaire à la variable entière, mais elle peut stocker des valeurs plus élevées.
-La gamme du variable de type “long” est définie entre: -2147483647 et +2147483646

Concept général

Fonctions et Bibliothèques

Fonctions

```
void function(){
  // read sensors
  // test conditions
  // perform actions
}

int add(int a, int b){
  int c = a + b;
  return c;
}
```

- Une fonction est un groupe d'instructions qui effectuent ensemble une tâche.
- Une fonction peut prendre certains paramètres (**a** et **b** dans l'exemple), effectuer certaines actions (addition dans l'exemple) et renvoyer la valeur (la somme de **a** et **b** qui est **c**).
- Elle peut également ne prendre aucun paramètre et ne pas avoir de valeur de retour (déclarée comme fonction “void”) telle que la fonction d'impression.

Bibliothèques

```
#include <Servo.h>

#include <EEPROM.h>

#include <NeoSWSerial.h>
```

- **Les bibliothèques** sont des fichiers écrits en C ou C++ (.c, .cpp) qui fournissent des fonctionnalités supplémentaires à vos croquis (par exemple, la possibilité de contrôler un servomoteur, de lire un encodeur, etc.). Ces instructions rendent **les fonctions** publiques et les constantes **définies** par **la bibliothèque** disponibles pour votre programme.
- Pour utiliser une bibliothèque existante dans un programme, allez simplement dans le menu Sketch, choisissez "Import Library", puis choisissez parmi les bibliothèques disponibles celle désirée. Cela insérera une instruction **#include** en haut du croquis pour chaque fichier d'en-tête (.h) du dossier de la bibliothèque.

Structure de contrôle

La déclaration la plus importante!

```
if (condition)
{
  //statement(s)
}

if (x > 120) digitalWrite(LEDpin, HIGH);

if (x > 120)
  digitalWrite(LEDpin, HIGH);

if (x > 120) {
  digitalWrite(LEDpin, HIGH);
}

if (x > 120) {
  digitalWrite(LEDpin1, HIGH);
  digitalWrite(LEDpin2, HIGH);
}
```

L'instruction « if » recherche une condition et exécute l'instruction en cours ou le jeu d'instructions si la condition est «vraie».
Les crochets peuvent être omis après une instruction if. Si cela est fait, la ligne suivante (définie par le point-virgule) devient la seule instruction conditionnelle.
Les instructions en cours d'évaluation entre parenthèses nécessitent l'utilisation d'un ou plusieurs opérateurs illustrés ci-dessous.
Opérateurs de comparaison:

x == y (x is equal to y)

x != y (x is not equal to y)

x < y (x is less than y)

x > y (x is greater than y)

x <= y (x is less than or equal to y)

x >= y (x is greater than or equal to y)

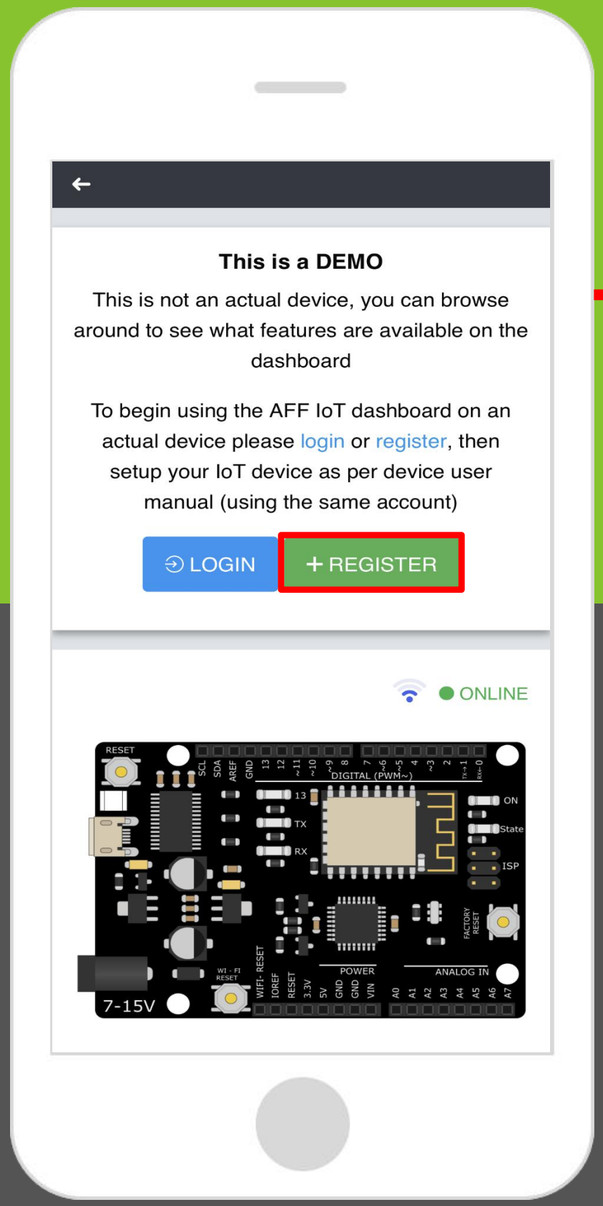
Faites attention à l'utilisation accidentelle du signe égal unique (par exemple, **`if(x = 10)`**). Le signe égale seule est l'opérateur d'assignation et définit x sur 10 (place la valeur 10 dans la variable x). Utilisez plutôt le double signe égal (par exemple, **`if(x == 10)`**), qui est l'opérateur de comparaison, et teste si x est égal à 10 ou non. La dernière déclaration n'est vraie que si x est égal à 10, mais la première déclaration sera toujours vraie.

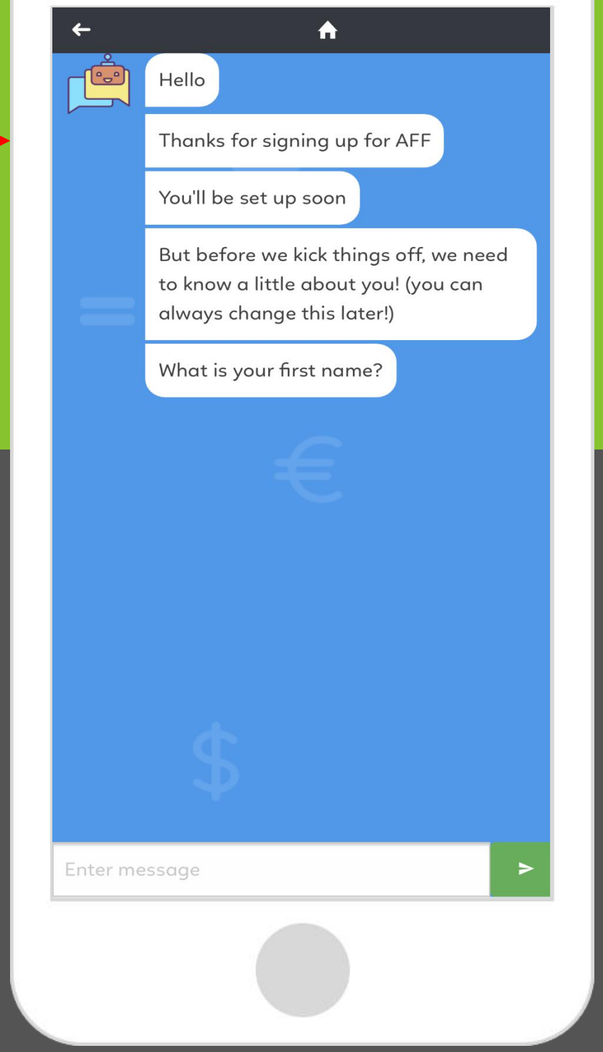
S'inscrire sur AFF IoT

(disponible sur Android et iOS)

Après avoir téléchargé l'application (AFF IoT), créez un nouveau compte. La création d'un compte est une procédure simple en 10 étapes et prend moins de 2 minutes. Si vous n’avez pas de smartphone, vous pouvez utiliser le navigateur sur un ordinateur tel que Chrome, Firefox, etc.

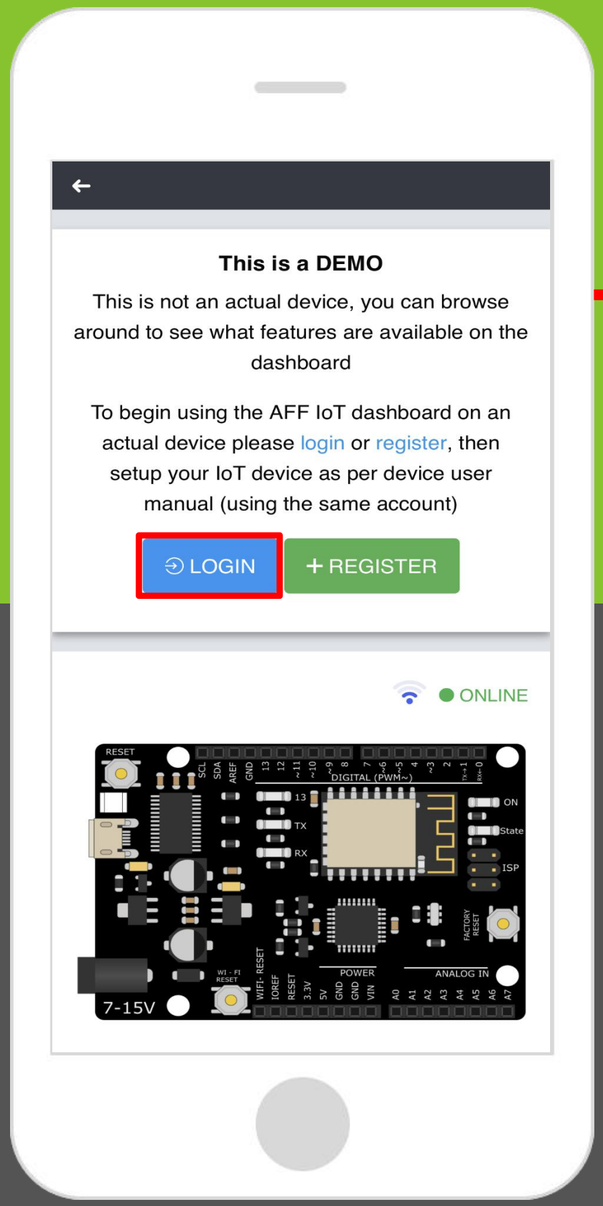
* Si vous rencontrez un problème lors de la création d'un nouveau compte ou lors de votre connexion, envoyez-nous un courrier électronique à l'adresse ***support@affcity.com***

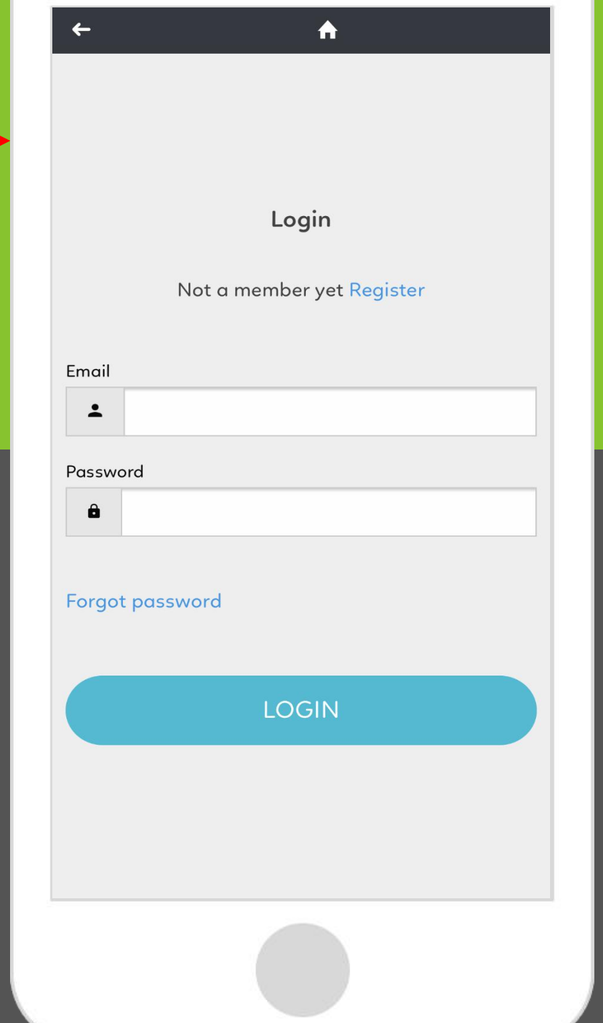




Application AFF IoT

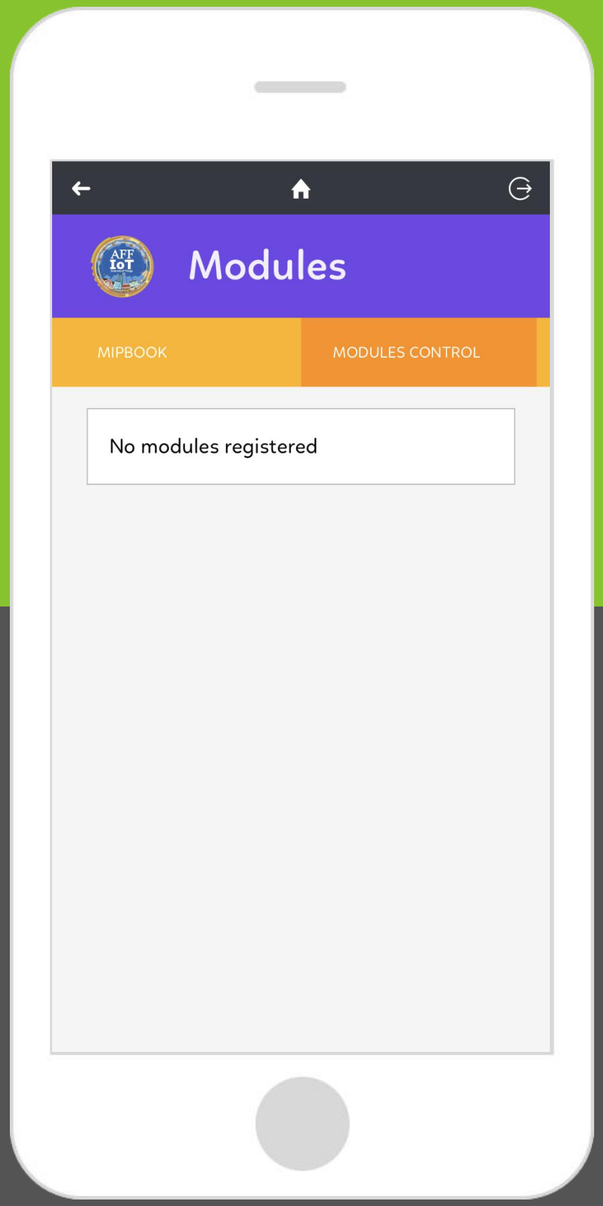
1. Créez un compte en cliquant sur le bouton vert "**REGISTER**".
2. Complétez les informations requises.
3. Ensuite, connectez-vous avec votre email et votre mot de passe.





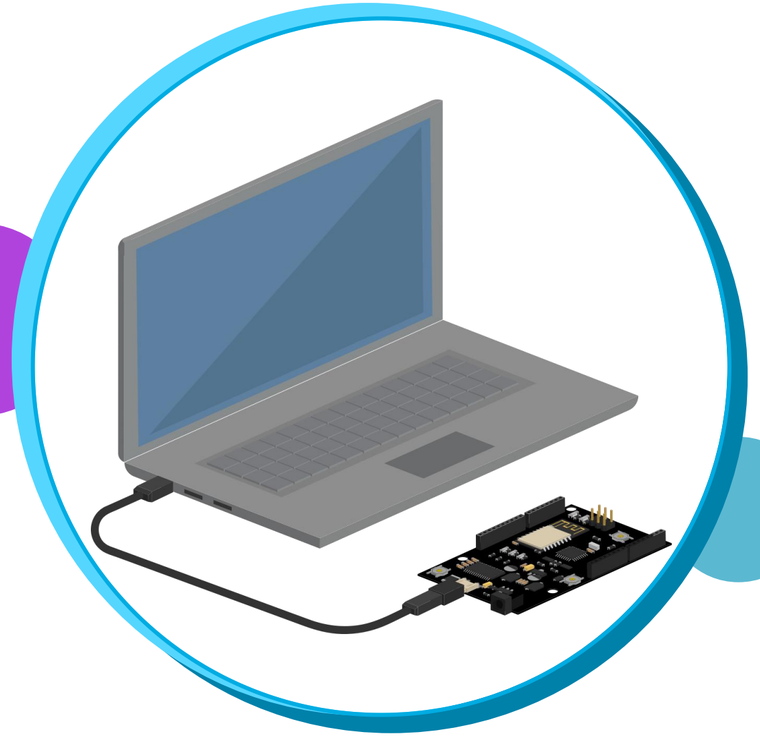
Application AFF IoT

1. Si vous avez un compte, appuyer sur le bouton bleu “**LOGIN**”.
2. Puis connectez-vous avec votre email et votre mot de passe.



Application AFF IoT

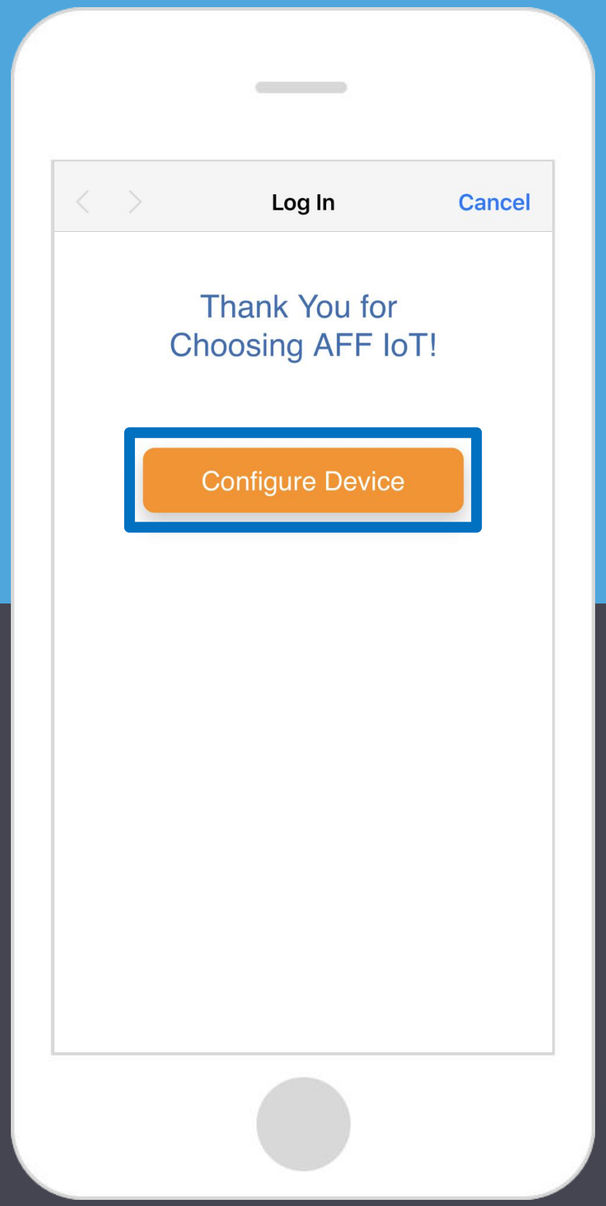
Après, enregistrez la carte AFF IoT (expliqué dans la section suivante). Ensuite, tout est prêt pour construire des projets!



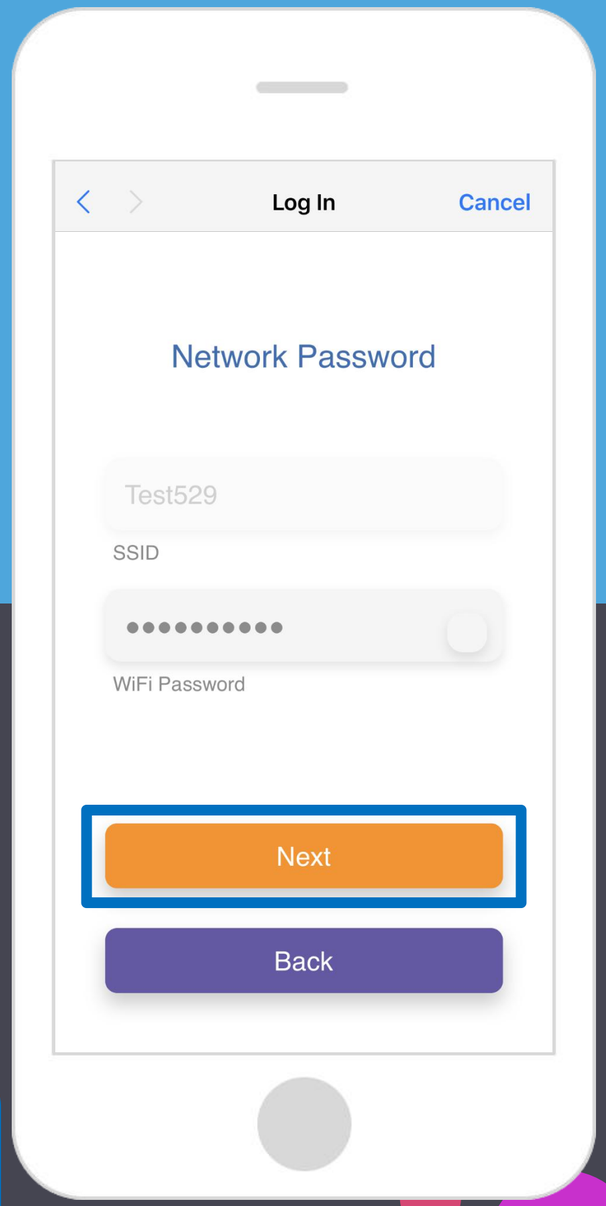
Connecter la carte AFF IoT à l'ordinateur

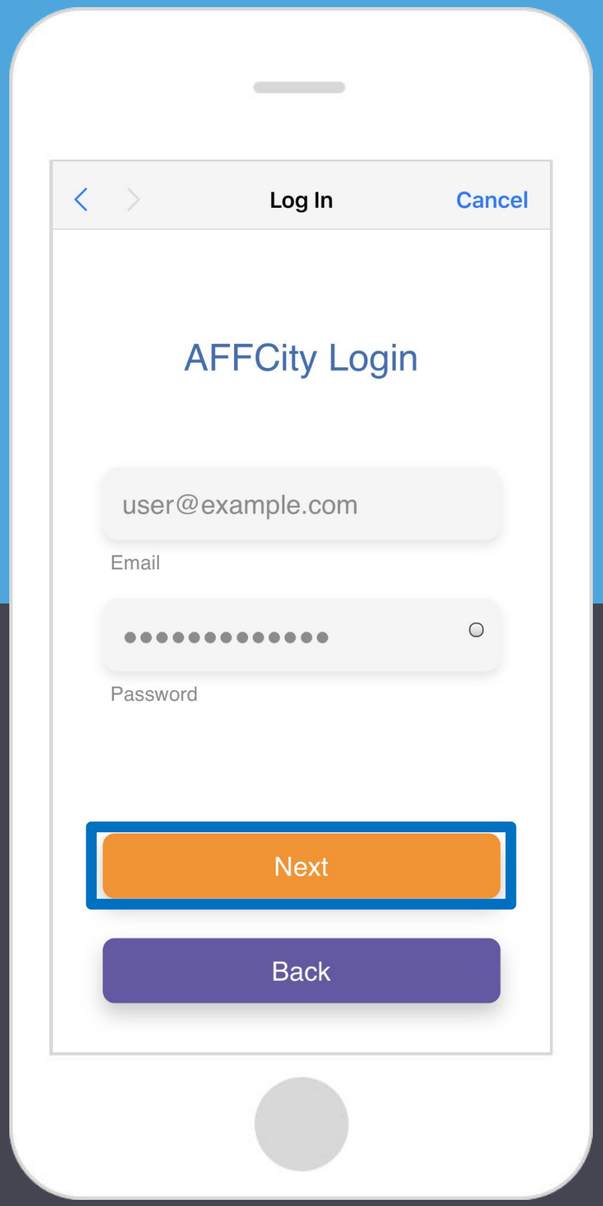
Utilisez un câble micro USB pour connecter la carte AFF IoT à l'un des ports USB de l'ordinateur. Ensuite ouvrez la page Wi-Fi sur votre Smartphone ou bien sur l'ordinateur.

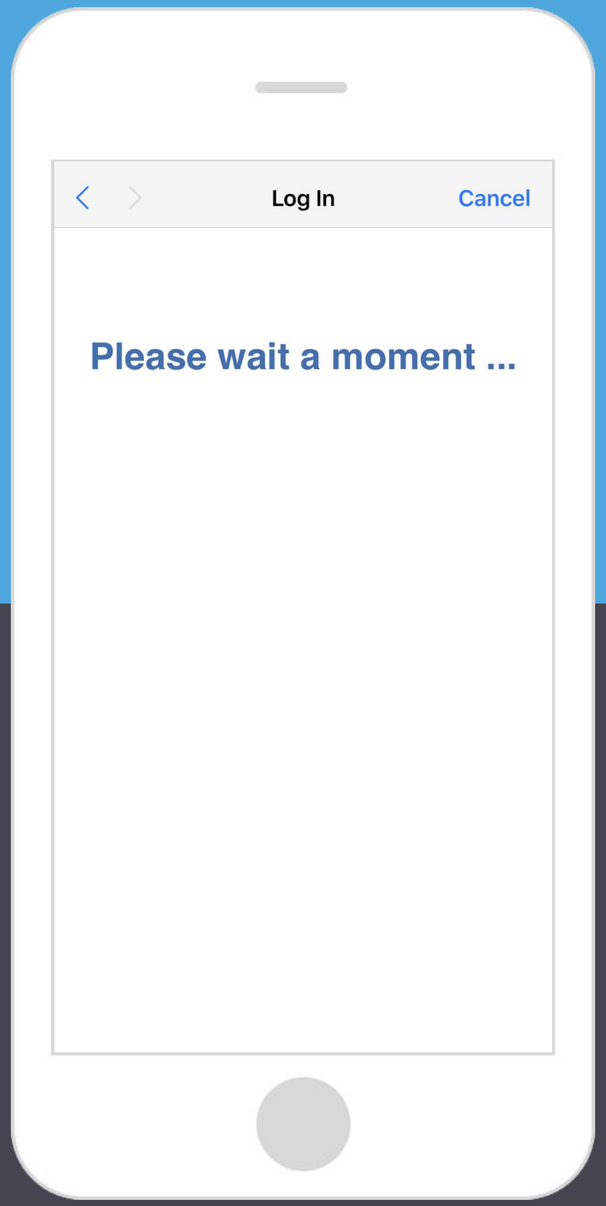
Choisissez votre carte AFF IoT du list des Wi-Fi

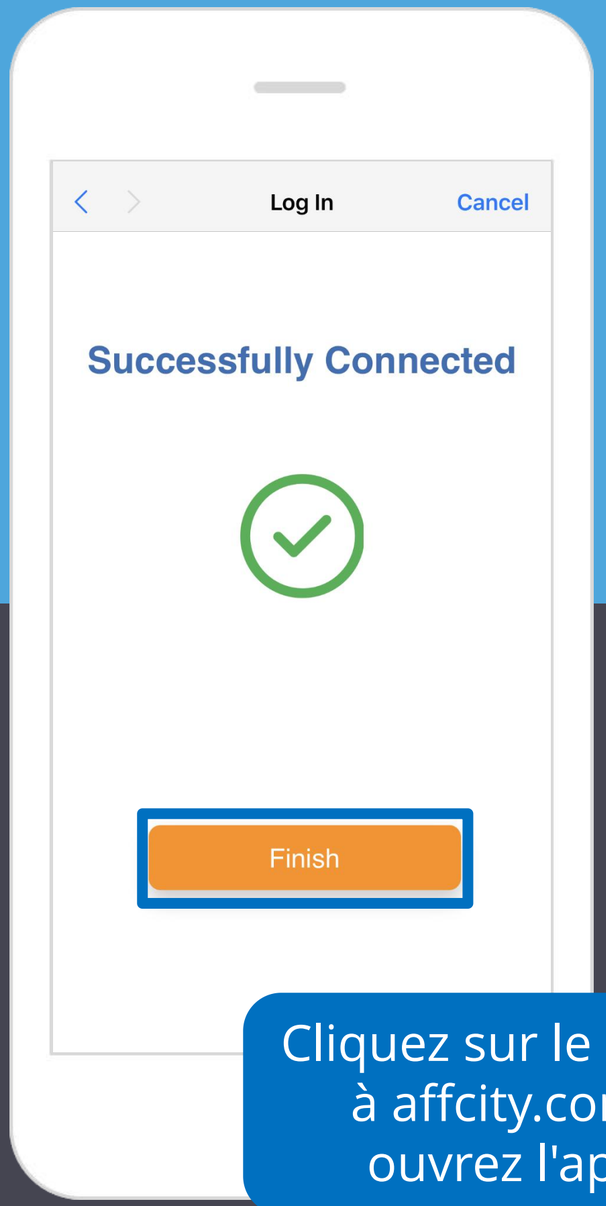


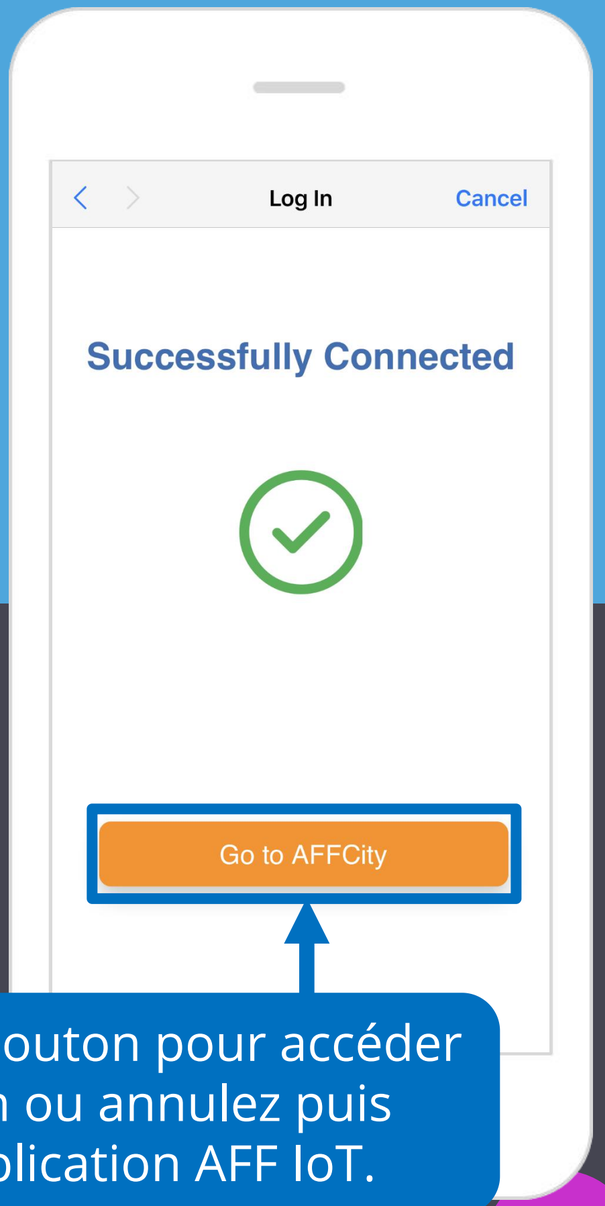
Si votre réseau ne figure pas dans la liste, cliquez sur «Scan Networks».



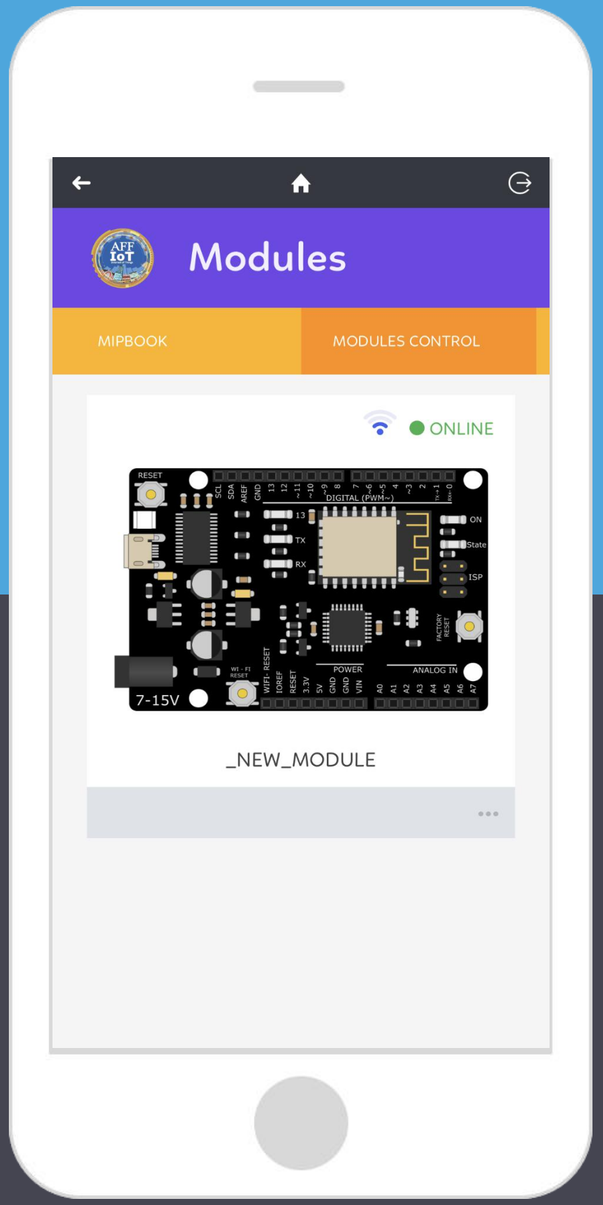






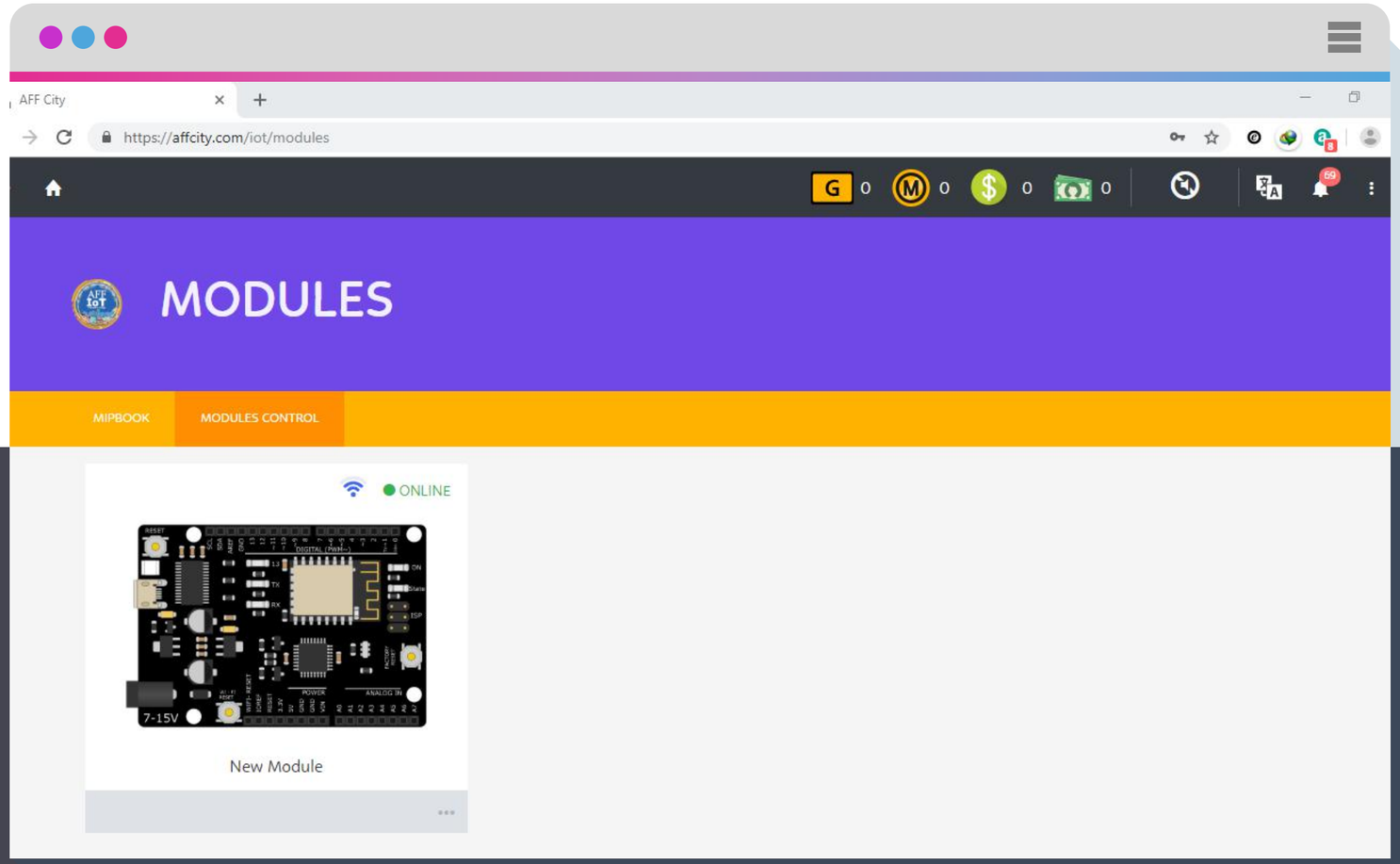


Cliquez sur le bouton pour accéder à affcity.com ou annulez puis ouvrez l'application AFF IoT.



Finalement...

Ouvrez l'application AFF IoT et vous devriez voir la carte AFF IoT connectée à Internet.
Il est nommé par défaut NEW MODULE et peut être édité.



Pour les utilisateurs de PC

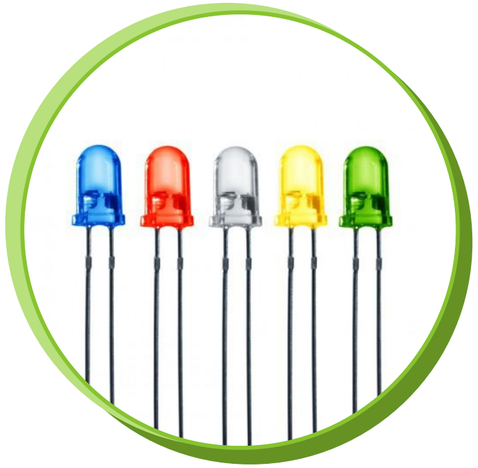
La procédure est la même, sauf qu'au lieu d'utiliser l'application AFF IoT, vous devez vous rendre sur https://affcity.com/iot/modules
Ici, vous pouvez continuer comme décrit avec l'application.

Construire 12 Projets

Le Kit AFF IoT **contient :**

Fils de connexion x20

Différents couleurs



LEDs x27

Rouge x5, Vert x5, Bleu x5,
Jaune x5, Blanc x5 et RVB x2

Bouton poussoir x9

Bouton poussoir sous la forme
d'un carré

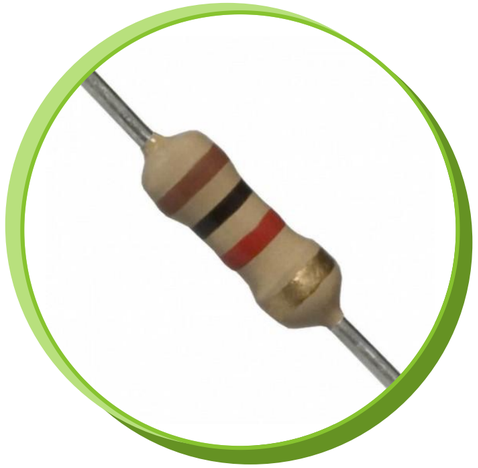
Potentiomètre x4

Résistance variable entre
0Ω et 5kΩ

Le Kit AFF IoT **contient :**

Résistance 10KΩ x6

0.5W 5%

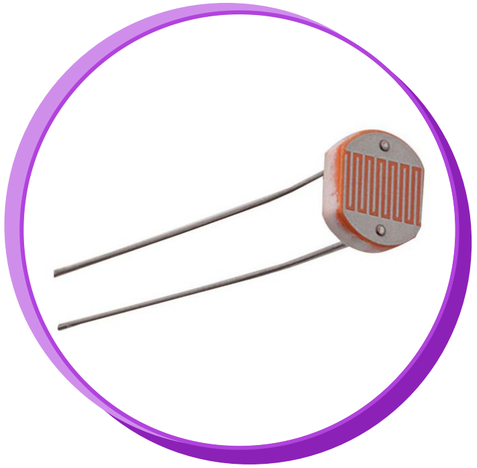


Résistance 1KΩ x4

0.5W 5%

Résistance 330Ω x25

0.25W 5%

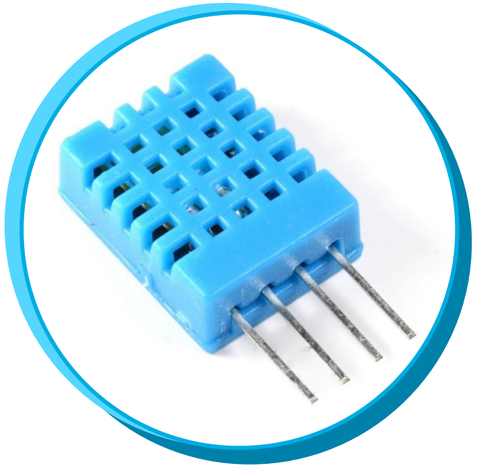


Photocellule x2

Résistance dépendante de la lumière

Light Dependent Resistor (LDR)

Le Kit AFF IoT **contient :**



DHT11 x1

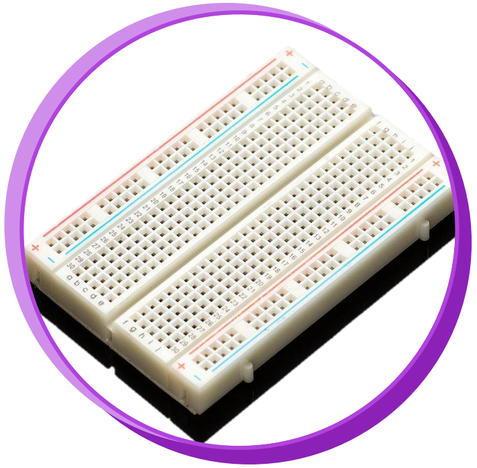
Capteur d'humidité et de température

Thermistance x3

1KΩ à 25°C

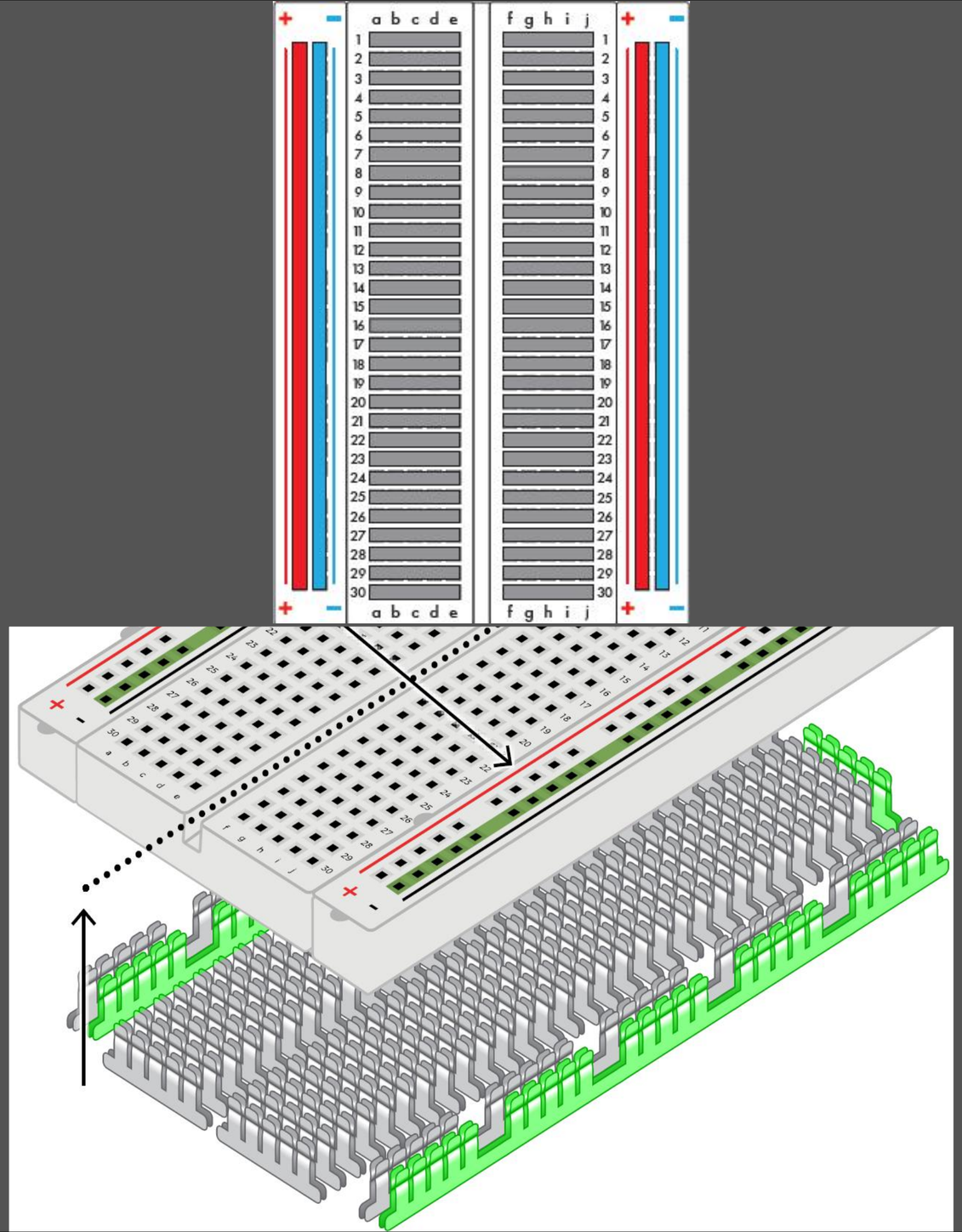
Bipeur piézo x2

Bippeur piézoélectrique polarisée



Plaque d’essai x1

Plaque d’essai sans sodure avec 400 broches



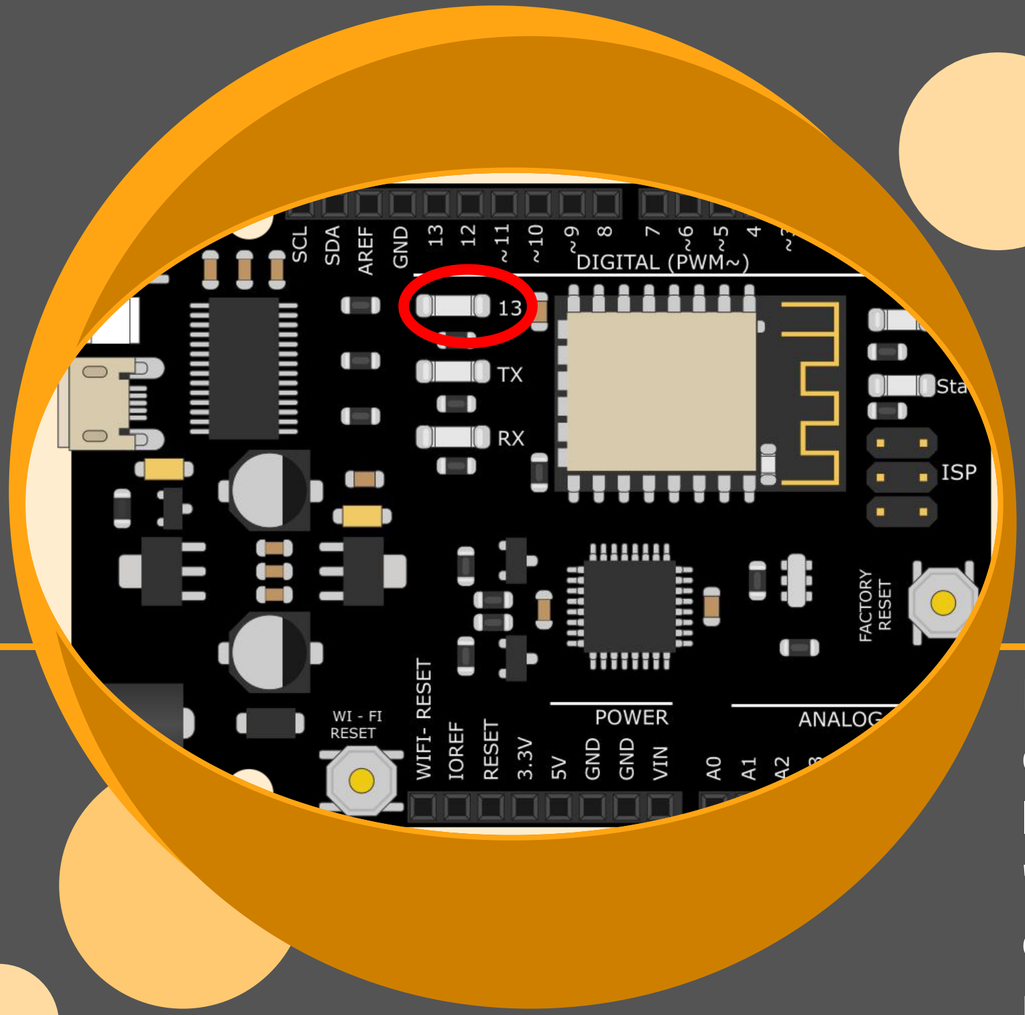
Plaque d’essai

La plaque d’essai est une base de construction pour le prototypage de l'électronique.

VCC:
Chaque signe + est alimenté n'importe où dans la colonne verticale.

GND:
Chaque signe - est relié à la terre n'importe où dans la colonne verticale.

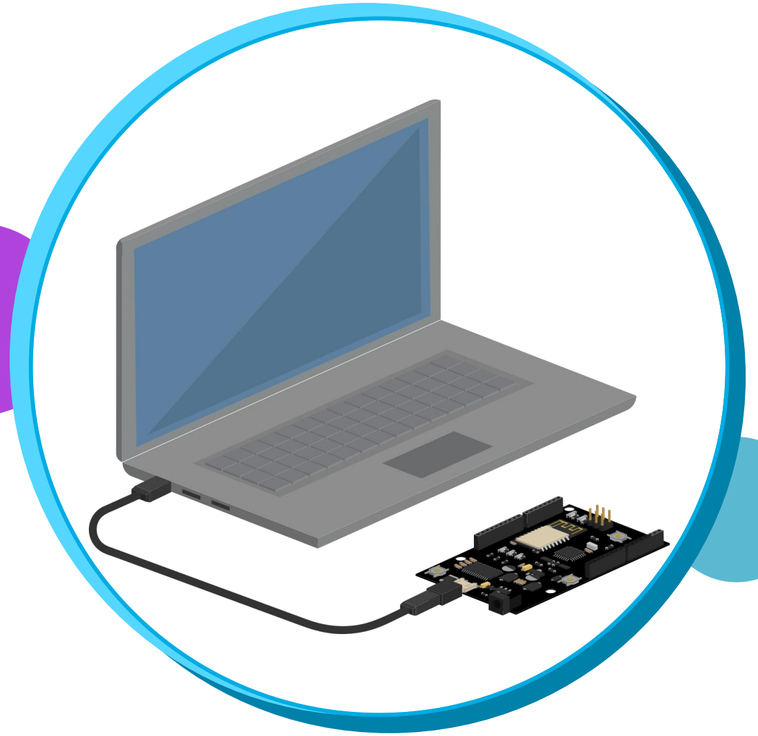
Rangées horizontales:(a-e & f-j):
Chacune de ces rangées numérotées de 1 à 30 est composée de cinq prises horizontales.
Les composants placés dans la même rangée seront connectés ensembles.



1 Clignoter le LED intégrée

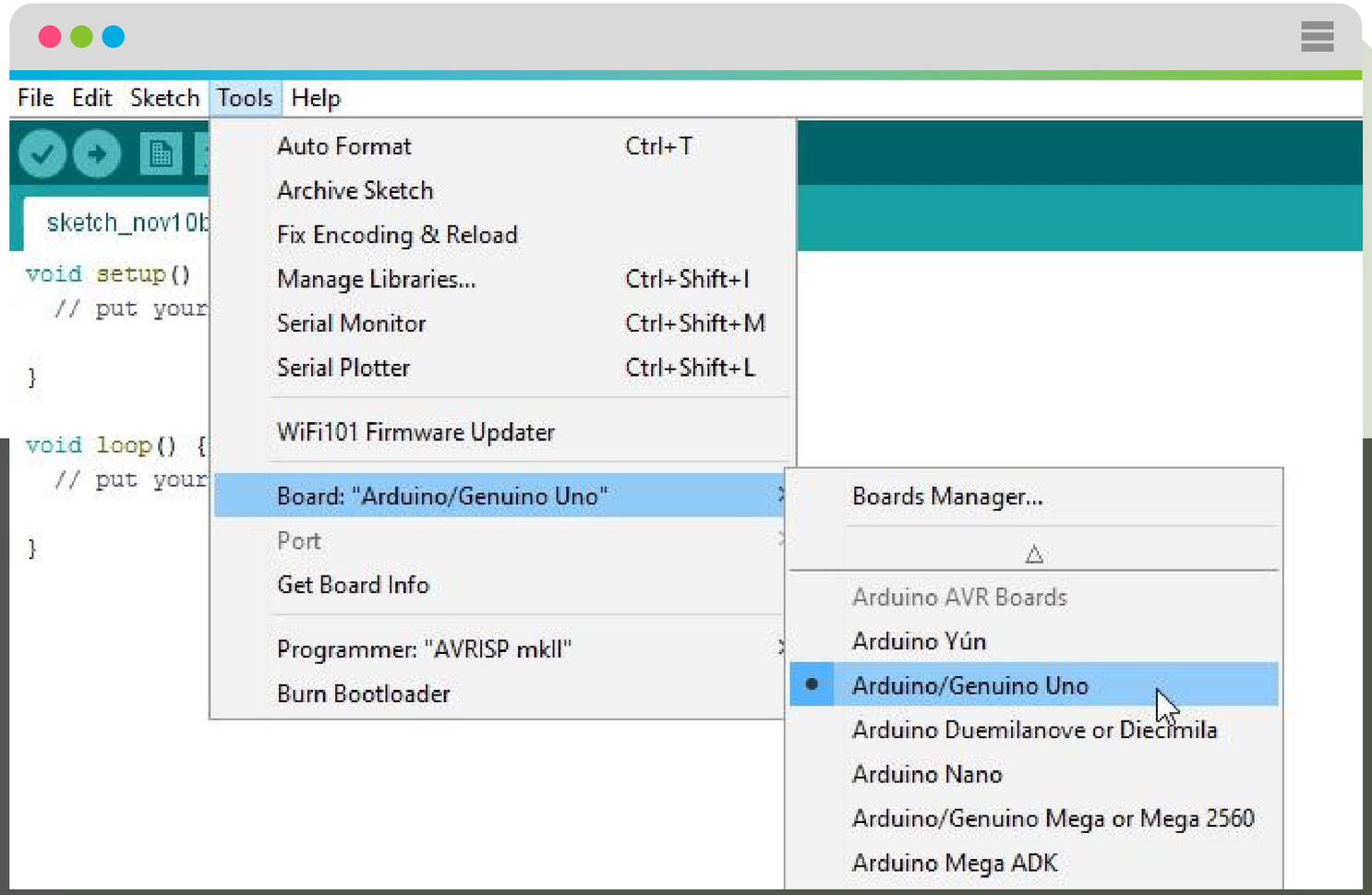
Le premier programme que chaque programmeur apprend consiste à écrire suffisamment de code pour que son code affiche la phrase "Hello World!" sur un écran. Le voyant clignotant est "Hello World!" de l'informatique physique. Dans ce tutoriel, le composant nécessaire est uniquement la carte AFF IoT et le câble micro USB.

Ce tutoriel consiste à allumer la LED intégrée (LED13) pendant une seconde, puis à l'éteindre pendant une seconde, à plusieurs reprises.



Connecter la carte AFF IoT à l'ordinateur

Utilisez un câble micro USB pour connecter la carte AFF IoT à l'un des ports USB de l'ordinateur.

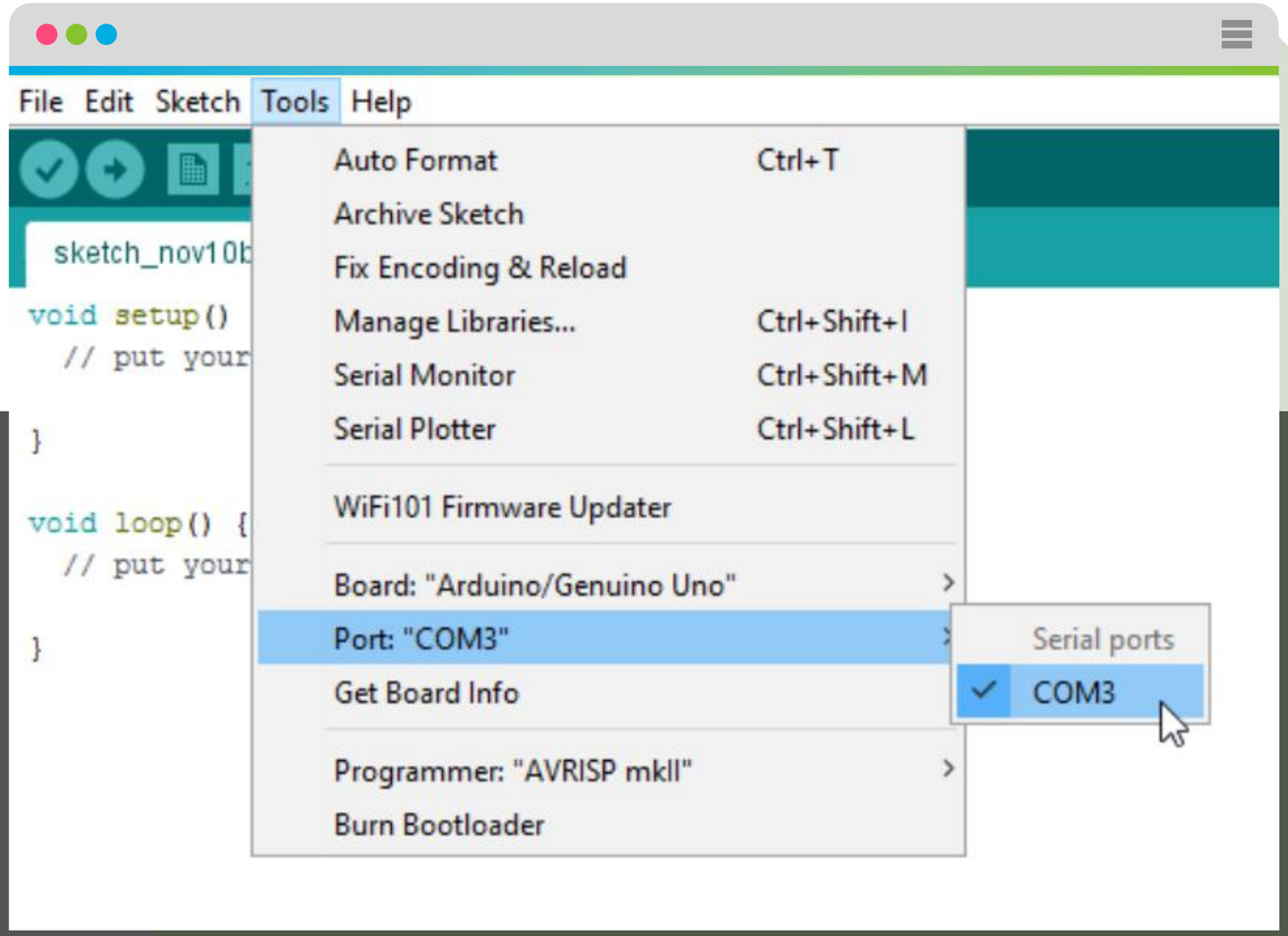


Séléctionner la carte: Arduino/Genuino Uno

Note:

La carte AFF IoT n'est pas listée dans le logiciel Arduino.

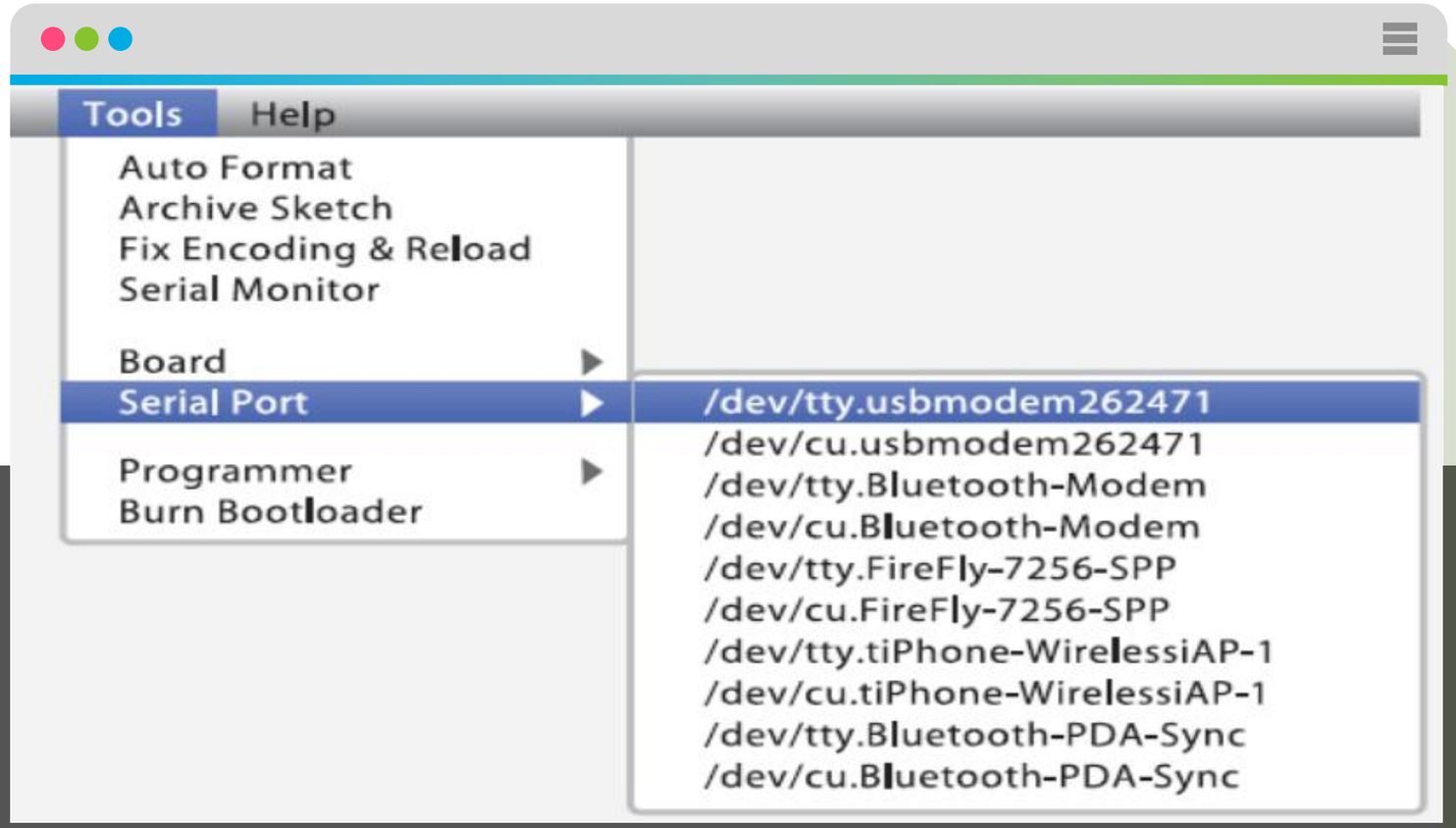
Séléctionner "Arduino/Genuino UNO" à sa place.



Séléctionner la périphérique série : (pour Windows)

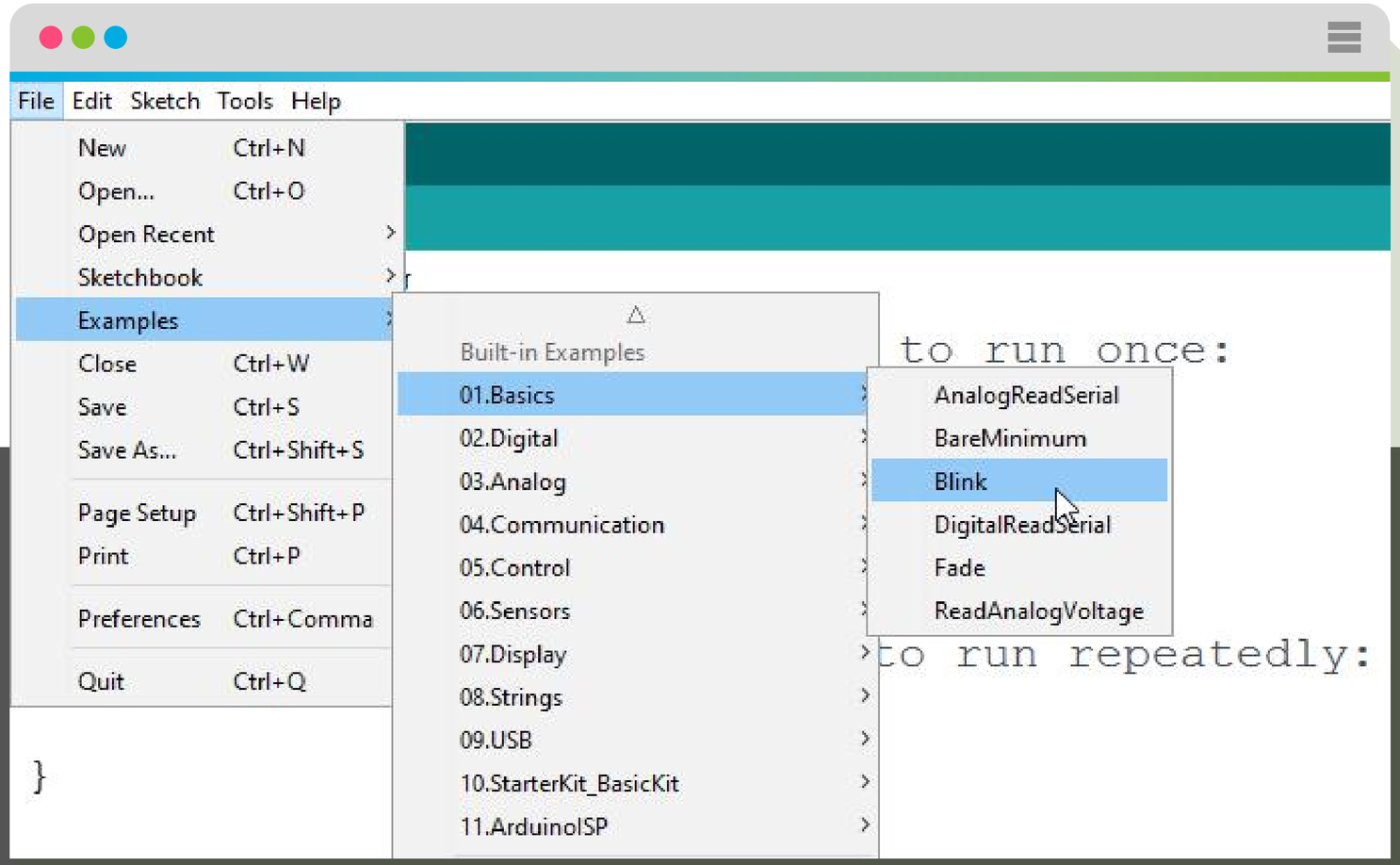
Sélectionnez le périphérique série de la carte dans le menu Tools → Serial Port. Pour savoir quel port, vous pouvez déconnecter la carte et rouvrir le menu. l'entrée qui disparaît devrait être votre carte. Reconnectez la carte et sélectionnez ce port série. Veuillez noter que le nombre de ports série peut différer de l'exemple utilisé.

Note: Si vous ne pouvez pas voir la carte, veuillez vous référer au lien suivant: https://learn.sparkfun.com/tutorials/how-to-install-ftdi-drivers/windows---in-depth



Séléctionner la périphérique série : (pour Mac)

Sélectionnez le périphérique série de la carte dans le menu Tools → Serial Port. Cela devrait être quelque chose comme **/dev/tty.usbmodem** ou **/dev/tty.usbserial**



Ouvrez l'exemple Blink

Après avoir connecté la carte AFF IoT à l'ordinateur, ouvrez l'exemple Blink répertorié dans les exemples écrits de l'EDI Arduino.

```
// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);                       // wait for a second
  digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);                       // wait for a second
}
```

L’exemple Blink

Cet exemple est l'alphabet du monde des microcontrôleurs!

Code à noter..

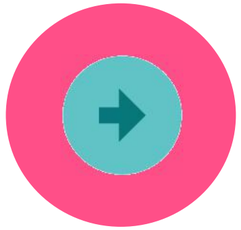
`pinMode(LED_BUILTIN, OUTPUT)` : Avant de pouvoir utiliser l'une des broches de la carte, celle-ci doit être spécifiée en tant qu'entrée (`INPUT`) ou sorite (`OUTPUT`). Nous utilisons une "fonction" intégrée appelée `pinMode()` pour définir le pin13 (auquel la LED13 est connectée) en tant que sortie (`OUTPUT`).

`digitalWrite(LED_BUILTIN, HIGH)` : Une broche de sortie numérique est réglée sur `HIGH` (sortie 5 volts) ou sur `LOW` (sortie 0 volt). Nous devons choisir la broche à contrôler dans le premier paramètre (LED_BUILTIN dans l'exemple) et la valeur voulu (HIGH dans l'exemple).

`delay(1000)` : Elle met le programme en pause pendant la durée (en millisecondes) spécifiée en tant que paramètre. (Il y a **`1000`** millisecondes dans une seconde).

Enfin..

Cela compile votre code. L'IDE change le texte en instructions que le microcontrôleur peut comprendre.



Cela envoie les instructions via le câble USB à la puce du microcontrôleur sur la carte.
La carte commencera alors à exécuter le code automatiquement.

Ce que vous **devriez voir**

Vous devriez voir votre voyant s'allumer pendant une seconde, puis s'éteindre pendant une seconde et ainsi de suite. Si ce n'est pas le cas, assurez-vous que vous avez correctement téléchargé le code sur la carte et qu'il n'y a pas d'erreur de compilation ou consultez les conseils de dépannage ci-dessous.

Dépannage

LED13 ne s'allume pas?
Assurez-vous que la carte est correctement branchée sur le PC

Le programme ne se charge pas ?
Cela arrive parfois, la cause la plus probable est de choisir un mauvais port série, vous pouvez changer cela dans Tools → Port.

Toujours ne se charge pas?
La solution commune pour presque tous les appareils électroniques est de l'éteindre puis de le rallumer.

2 Contrôler la LED avec des boutons poussoirs

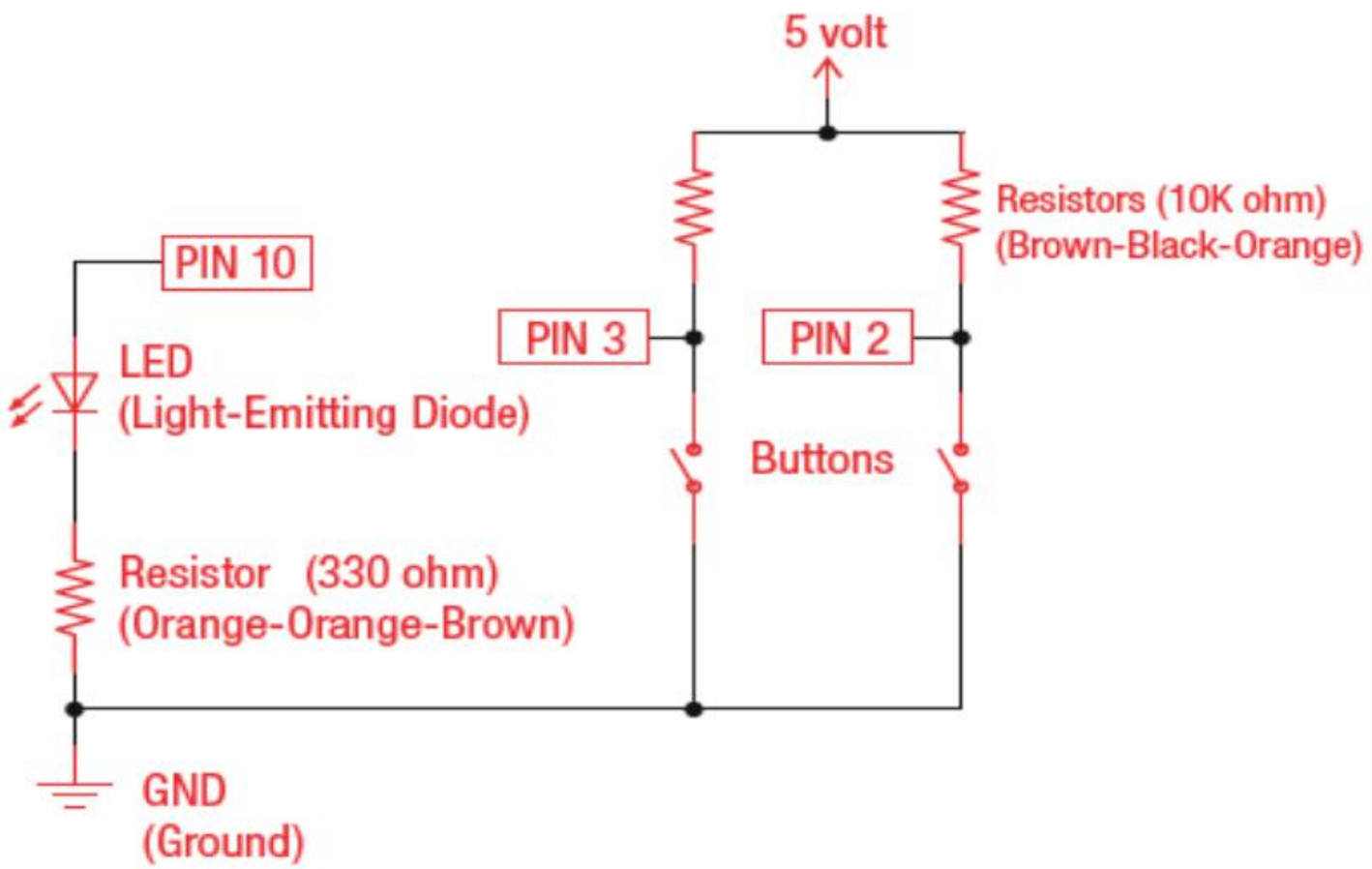
LED x1

Résistance 330Ω x1

Bouton poussoir x2

Résistance 10KΩ x2

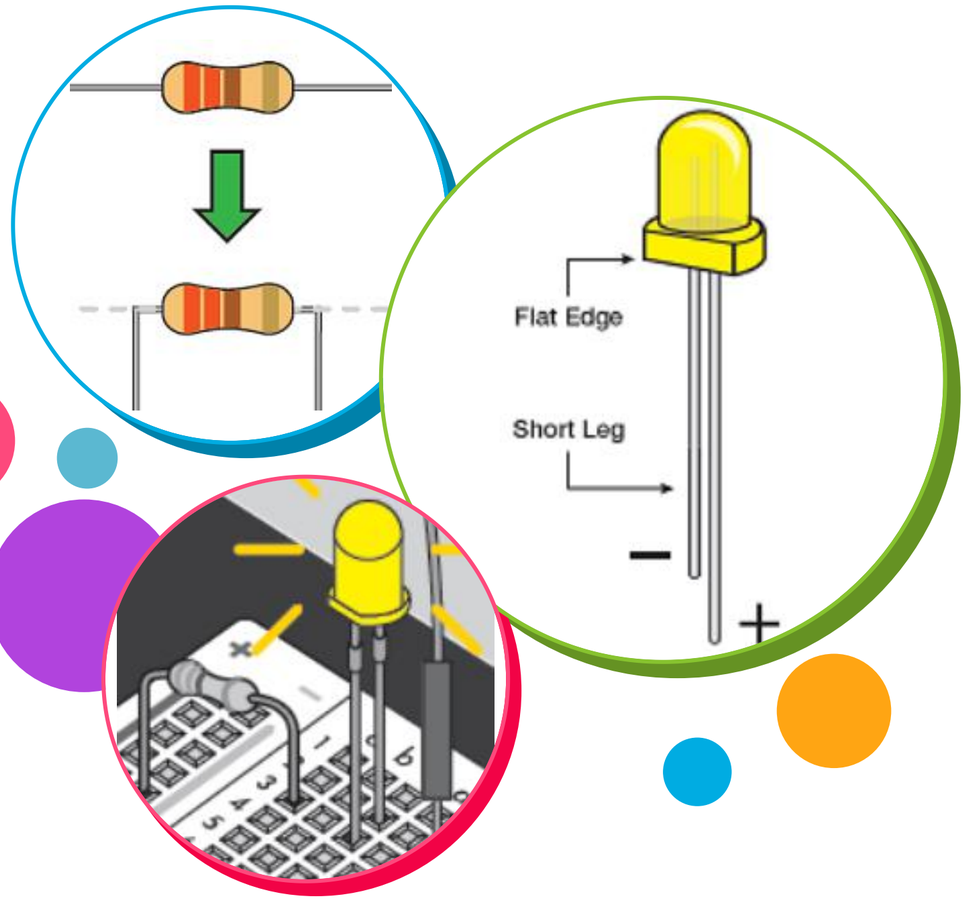
Fils de connexion x7



Bouton poussoir

Le bouton poussoir est l'une des entrées les plus courantes et les plus simples. La façon dont un bouton poussoir fonctionne avec la carte est que lorsque le bouton est enfoncé, la tension passe à 0V (LOW). La carte lit ceci et réagit en conséquence. Dans ce circuit, une résistance pull-up sera utilisée pour maintenir la tension élevée à 5V (HIGH) lorsque le bouton n'est pas enfoncé.

Attention!

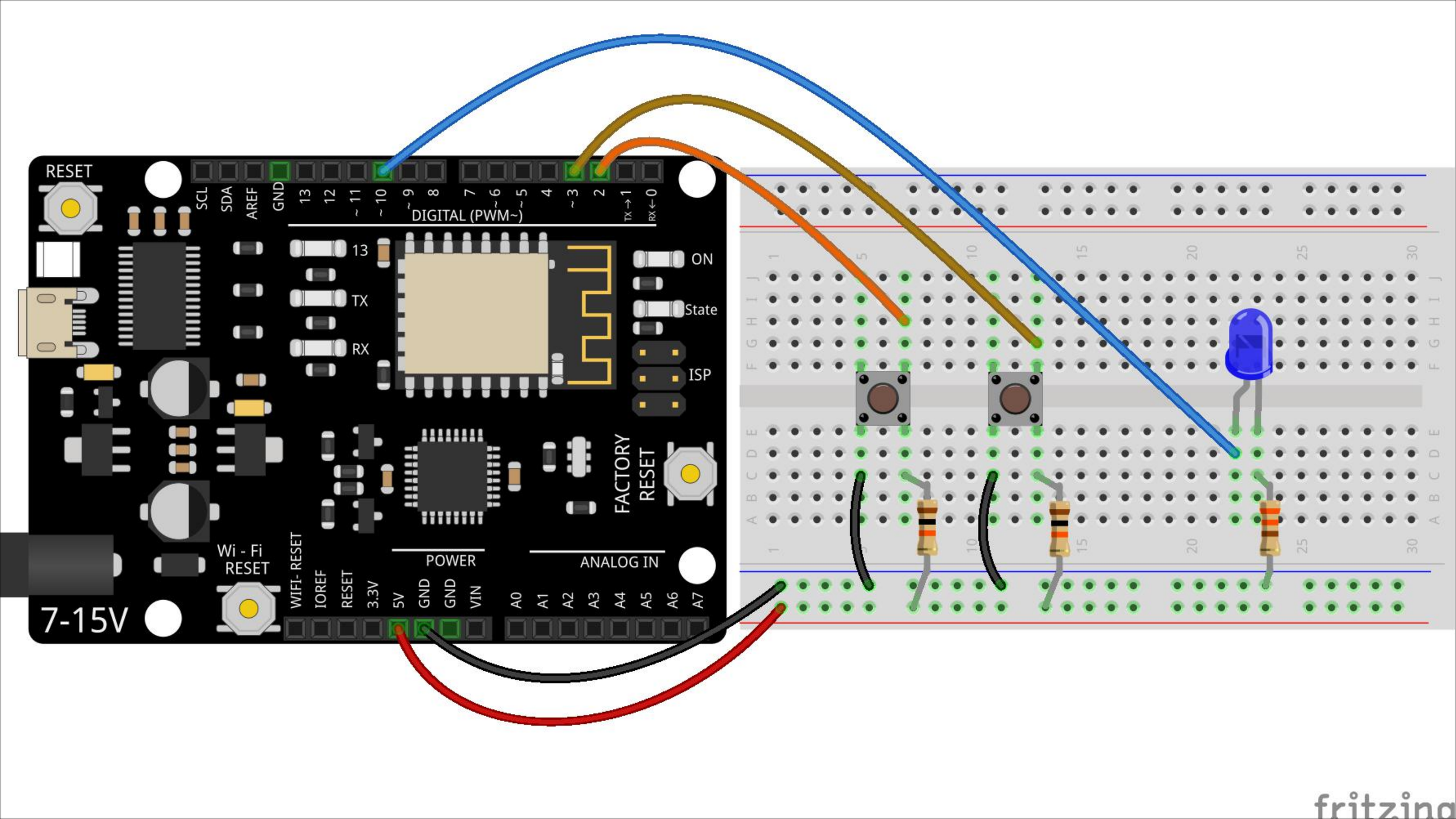


LED:

La jambe courte, marquée d'un côté plat (Flat Edge), passe en position négative (-).

Résistance 330Ω:

La bande de couleur doit être orange-orange-marron-or.
Les pattes des composants peuvent aller dans les deux trous.


RESET
SCL
SDA
AREF
GND
13
12
~ 11
~ 10
~ 9
8
7
~ 6
~ 5
4
~ 3
2
TX → 1
RX ← 0
DIGITAL (PWM~)
13
TX
RX
ON
State
ISP
FACTORY RESET
Wi - Fi RESET
WIFI- RESET
IOREF
RESET
3.3V
5V
GND
GND
VIN
POWER
ANALOG IN
A0
A1
A2
A3
A4
A5
A6
A7
7-15V
fritzing

==	ÉQUIVALENCE	A == B est vrai si A et B sont MÊME.
!=	DIFFÉRENCE	A != B est vrai si A et B NE sont PAS MÊME.
&&	ET	A && B est vrai si les deux A et B sont vrai
||	OU	A || B est vrai si A ou B ou les deux sont vrais
!	NON	!A est vrai si A est faux !A est faux si A est vrai

Comment Utiliser la LOGIQUE

Vous pouvez allumer un appareil de chauffage s'il fait trop froid, un ventilateur s'il fait trop chaud, arroser les plantes si elles deviennent trop sèches, etc. Pour prendre de telles décisions, il existe plusieurs opérations logiques. Elles peuvent être combiné pour construire des instructions `if()` complexes.

Par example:

```
if ((mode == chaffage) && ((temperature < seuil) || (temps == "nuit")))
        digitalWrite(CHAUFFAGE, HIGH);
```

AFF IoT Board folder > Circuits > Using_Pushbuttons

Using_Pushbuttons

```
#define Button1Pin  2
#define Button2Pin  3
#define BlueLedPin  10

void setup() {
  // Set up the pushbutton pins to be an input:
  pinMode(Button1Pin, INPUT); // configure the pin connected to the pushbutton 1 to be an input
  pinMode(Button2Pin, INPUT); // configure the pin connected to the pushbutton 2 to be an input
  pinMode(BlueLedPin, OUTPUT); // configure the pin connected to the Blue Led to be an output
}

void loop() {

  int Button1State, Button2State;  // variables to hold the pushbutton states
  Button1State = digitalRead(Button1Pin); // read the state of Button1 (LOW if pressed and HIGH if released)
  Button2State = digitalRead(Button2Pin); // read the state of Button2 (LOW if pressed and HIGH if released)

  if (((Button1State == LOW) || (Button2State == LOW))  // if we're pushing button 1 OR button 2
      && !                                               // AND we're NOT
      ((Button1State == LOW) && (Button2State == LOW))) // pushing button 1 AND button 2 simultaneously
                                                         // then...
  {
    digitalWrite(BlueLedPin, HIGH);  // turn the Blue LED on
  }
  else
  {
    digitalWrite(BlueLedPin, LOW);  // turn the Blue LED off
  }
}
```

Ouvrez votre croquis
Using_Pushbuttons

Code à noter..

`pinMode(Button1Pin, INPUT)` : Les broches numériques peuvent être utilisées comme entrées ou sorties. Il est spécifié comme INPUT afin de lire l'état du bouton.

`Button1State = digitalRead(Button1Pin)` : Pour lire une entrée numérique, nous utilisons la fonction `digitalRead()`. Il retournera `HIGH` (ou 1) s'il y a 5V sur la broche, ou `LOW` (ou 0) s'il y a 0V sur la broche.

`if (Button1State == LOW)` : Parce que nous avons connecté le bouton à `GND`, il sera lu `LOW` lorsqu'il sera enfoncé. L'opérateur "équivalence" ("`==`") est utilisé pour voir si le bouton est enfoncé.

`else:` l'instruction `else` signifie que si la condition de l'instruction `if` est fausse `(false)`, le code entre les crochets `else` est exécuté.

Ce que vous **devriez voir**

Vous devriez voir le LED s'allumer si vous appuyez sur l'un des boutons, et s'éteindre si vous appuyez sur les deux boutons (voir le code). Si cela ne fonctionne pas, assurez-vous d’avoir correctement assemblé le circuit, vérifié et téléchargé le code sur votre carte ou consultez les conseils de dépannage ci-dessous.

Dépannage

LED ne s'allume pas?
Le bouton poussoir est carré, et de ce fait, il est facile de le mettre dans le mauvais sens. Donnez-lui une rotation de 90 degrés et voyez si cela commence à fonctionner.

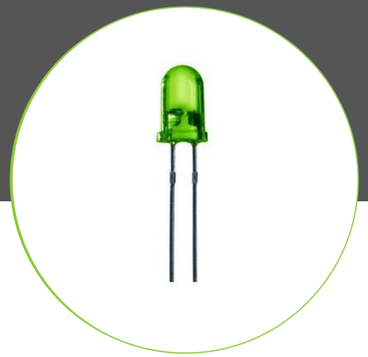
Toujours pas d’éclairage?
Les LED ne fonctionnent que dans un sens. Essayez de le sortir et en le tournant de 180 degrés (ne vous inquiétez pas, son installation dans le sens opposé n’endommage pas la LED).

Application dans la réalité

Les boutons dans la plupart des contrôleurs de jeux vidéo.

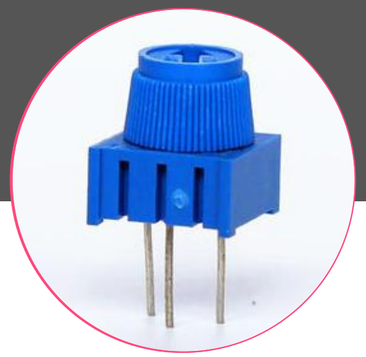
3 Atténuer une LED à l'aide d'un potentiomètre

Fils de connexion x7



LED verte x1

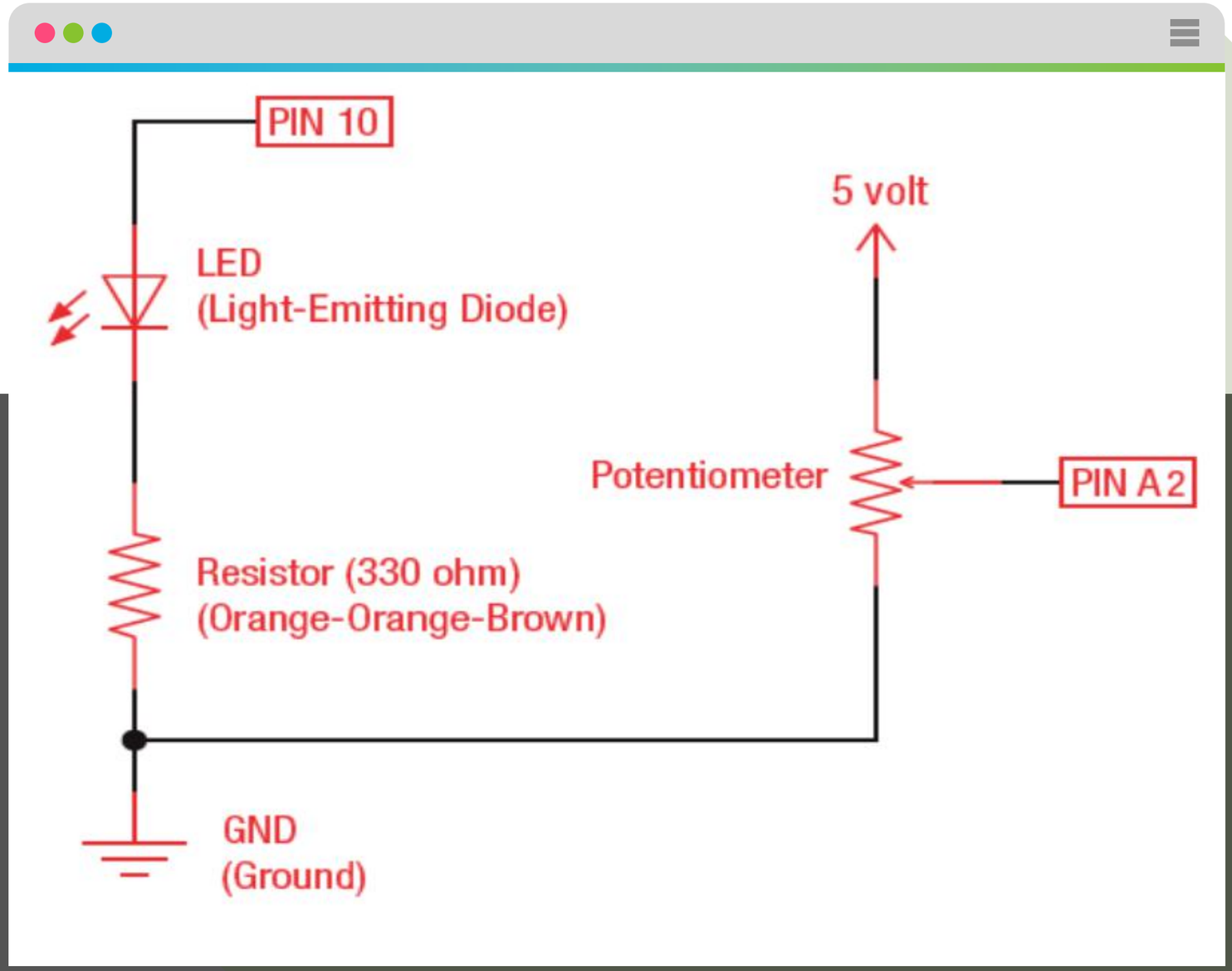
Résistance 330Ω x1



Potentiomètre x1

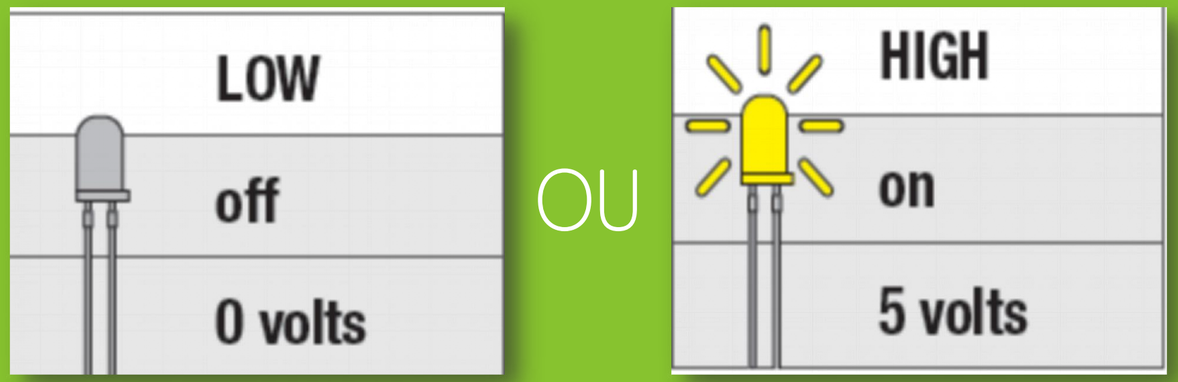
Notez que les broches analogiques A0 et A1 sont réservées à la communication avec le module Wi-Fi. Ne les utilisez donc pas, sinon la carte ne fonctionnera pas correctement!

Potentiomètre
(résistance variable)



Le potentiomètre a 3 broches, 2 broches externes et 1 interne. Lorsqu'il est connecté avec une tension de 5 volts sur ses deux broches extérieures, la broche du milieu fournit une tension comprise entre 0V et 5V, en fonction de la position du bouton sur le potentiomètre. Un potentiomètre est une démonstration d'un circuit diviseur de tension variable. Dans ce circuit, vous apprendrez à utiliser un potentiomètre pour contrôler la luminosité d'une LED.

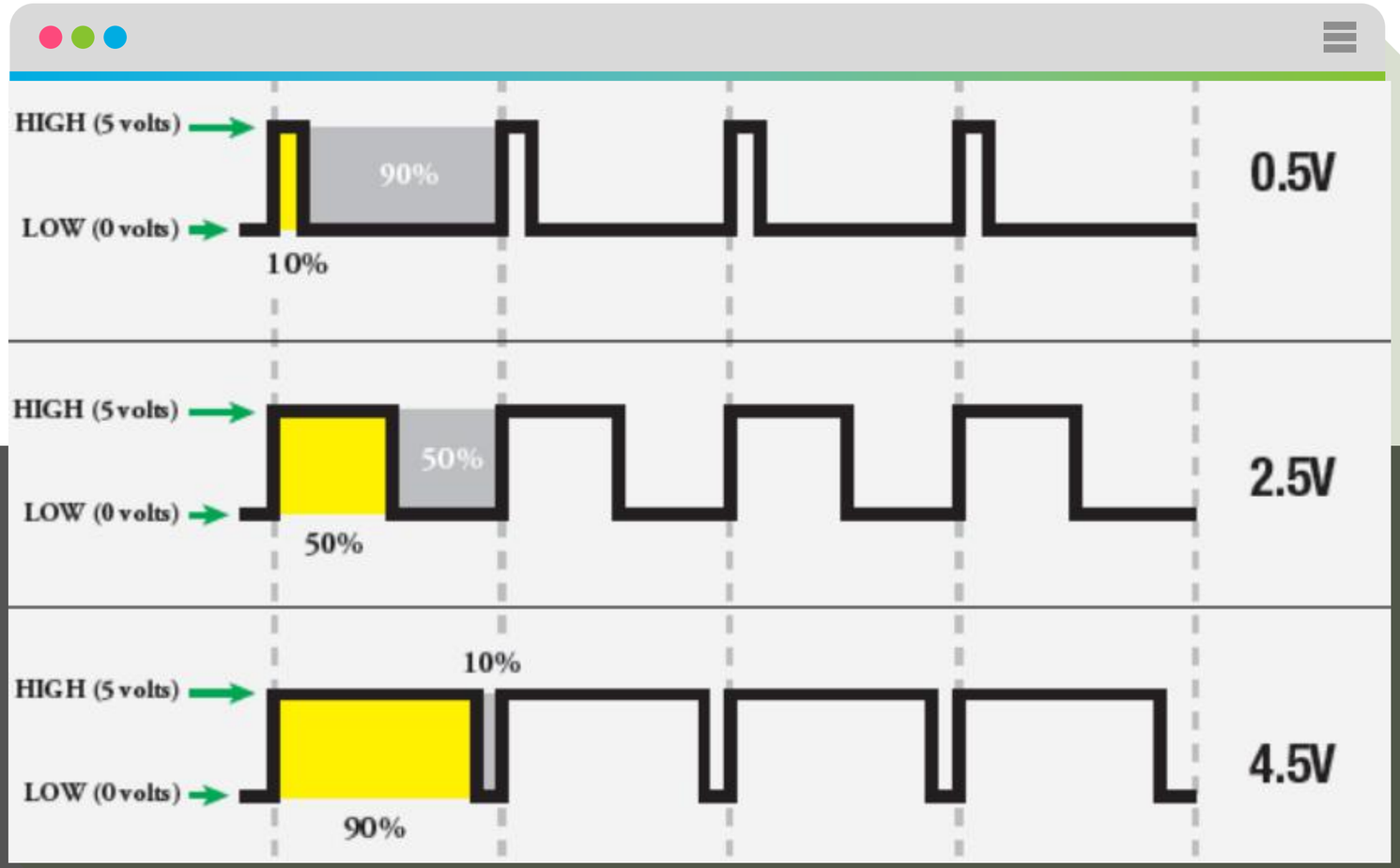
Numérique ou **Analogique**



0 volts	à	5 volts
0		1023

De nombreux dispositifs, tels que les LEDs et les boutons poussoirs, n'ont que deux états possibles: «allumé» et «éteint», ou encore appelé «HIGH» (5 volts) et «LOW» (0 volt). Les broches numériques d'une carte sont utiles pour émettre / recevoir des signaux de et vers le monde extérieur, et peuvent même faire des trucs comme la gradation simulée (en clignotant très vite) et les communications en série (transférer des données vers un autre appareil en les encodant sous forme des motifs de voltage HAUT "HIGH" et BAS "LOW").

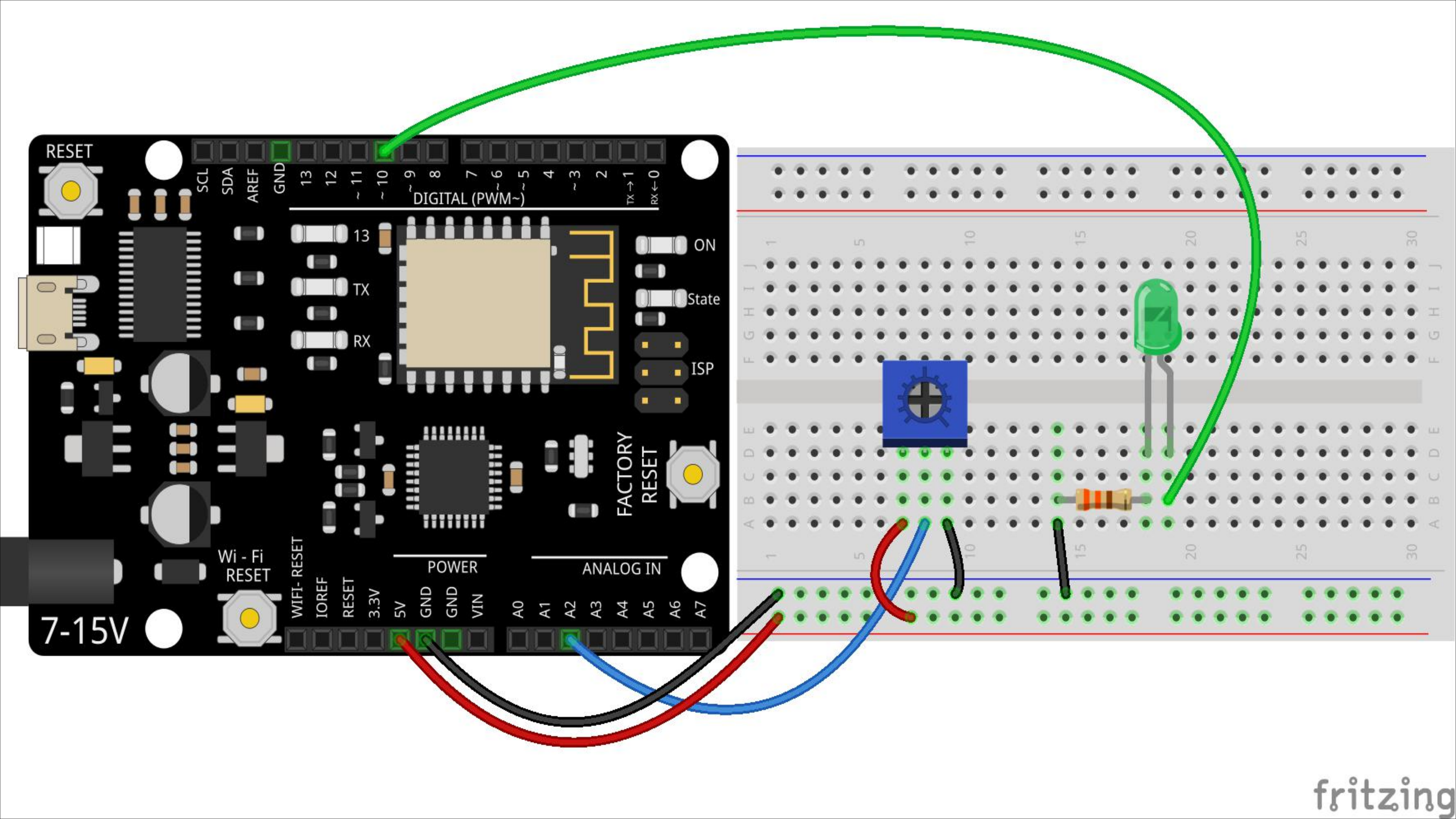
Mais il y a aussi beaucoup de choses qui ne sont pas simplement "allumées" ou "éteintes". Les niveaux de température, l'intensité lumineuse, etc. Tous ont une plage continue de valeurs entre "HIGH" et "LOW". Pour ceux-ci, la carte propose six entrées analogiques qui traduisent une tension d'entrée en une valeur comprise entre 0 (0 volt) et 1023 (5 volts). Les broches analogiques sont parfaites pour mesurer toutes ces valeurs du "monde réel".



Existe-t-il un moyen pour la carte AFF IoT d'émettre des tensions analogiques?

La carte AFF IoT n'a pas de véritable sortie de tension analogique. Mais, parce que la carte est si rapide, elle peut la simuler en utilisant quelque chose appelé **MLI** ("**Modulation de largeur d'impulsion**") (**PWM** *en anglais*). Les broches de la carte suivies de «~» sont compatibles avec la sortie analogique / MLI. Le microcontrôleur est si rapide qu'il peut faire clignoter une broche presque 1000 fois/s.

MLI va encore plus loin en faisant varier le temps passé par la broche en «HIGH» au temps qu'il passe en «LOW». S'il passe la majeure partie de son temps à l'état «HIGH», une LED connectée à cette broche apparaîtra lumineuse. S'il passe la plupart de son temps à l'état «LOW», le voyant est faible. Comme la broche clignote beaucoup plus rapidement que votre œil ne peut la détecter, la carte crée l'illusion d'une "vraie" sortie analogique.


RESET
SCL
SDA
AREF
GND
13
12
~ 11
~ 10
~ 9
8
7
~ 6
~ 5
4
~ 3
2
TX → 1
RX ← 0
DIGITAL (PWM~)
13
TX
RX
ON
State
ISP
FACTORY RESET
Wi - Fi RESET
WIFI- RESET
IOREF
RESET
3.3V
5V
GND
GND
VIN
POWER
A0
A1
A2
A3
A4
A5
A6
A7
ANALOG IN
7-15V
fritzing

AFF IoT Board folder > Circuits > Dimming_LED_using_Potentiometer

Dimming_LED_using_Potentiometer

```
#define GreenLedPin       10
#define PotentiometerPin  A2

void setup() {
  // put your setup code here, to run once:
  pinMode(GreenLedPin, OUTPUT); // configure the Green Led pin to be an output
}

void loop() {
  // put your main code here, to run repeatedly:
  int analogValue = analogRead(PotentiometerPin); // read the actual converted value (between 0 and 1023)
  // Note the analog pin can be read without declaring it as input

  analogValue = map(analogValue, 0, 1023, 0, 255); // transform the scale from 0 and 1023 to 0 and 255
  // becuase the analogWrite() function take parameter from 0 (0%) to 255 (100%) of duty cycle

  analogWrite(GreenLedPin, analogValue); // generate the PWM duty cycle according the needed intensity
}
```

Ouvrir votre croquis:

Dimming_LED_using_Potentiometer

Code à noter..

`int analogValue;` Les variables doivent être déclarées avant de les utiliser; nous déclarons ici une variable appelée analogValue, de type "int" (entier). N'oubliez pas que les noms de variables sont sensible aux majuscules et minuscules!

`analogValue = analogRead(PotentiometerPin);` La fonction `analogRead()` est utilisée pour lire la valeur sur une broche d'entrée analogique. `analogRead()` prend un paramètre, ici la broche analogique ("`PotentiometerPin`"), et retourne une valeur ("`analogValue`") comprise entre `0` (0 volt) et `1023` (5 volts).

`analogWrite(GreenLedPin, analogValue);` La fonction `analogWrite()` est utilisée pour générer un signal MLI (PWM) dans le `GreenLedPin` avec un rapport cyclique égale à `analogValue` (le rapport cyclique est le pourcentage de haute tension (5V) sur une période).

Code à noter..

`map(value, fromLow, fromHigh, toLow, toHigh)`

`value:` le variable à transformer
`fromLow:` la limite inférieure de la plage actuelle de la valeur
`fromHigh:` la limite supérieure de la plage actuelle de la valeur
`toLow:` la limite inférieure de la plage désirée de la valeur
`toHigh:` la limite supérieure de la plage désirée de la valeur

```
analogValue = map(analogValue, 0, 1023, 0, 255)
```

La lecture d'un signal analogique à l'aide de `analogRead()` retourne un nombre compris entre `0` et `1023`. Toutefois, pour commander une broche MLI (PWM) à l'aide de `analogWrite()` ,un nombre compris entre `0` et `255` est nécessaire en utilisant la fonction `map()`.

Ce que vous **devriez voir**

Vous devriez voir la LED devenir plus claire ou plus sombre en fonction de la position de votre potentiomètre. Si cela ne fonctionne pas, assurez-vous d'avoir correctement assemblé le circuit, vérifié et téléchargé le code sur votre carte ou consultez le dépannage ci-dessous:

Troubleshooting

Ne fonctionne pas
Assurez-vous de ne pas avoir accidentellement connecté l'élément résistif du potentiomètre, à la broche numérique 2 plutôt qu'à la broche analogique A2. (la rangée de broches sous les broches d'alimentation).

Travailler sporadiquement
Cela est probablement dû à une connexion légèrement louche avec les broches du potentiomètre. Essayez d'appuyer sur le potentiomètre.

La LED ne s'allume pas?
Les LED ne fonctionnent que dans un seul sens. Essayez de le sortir et en le tournant de 180 degrés (ne vous inquiétez pas, son installation dans le sens opposé n'endommage pas la LED)

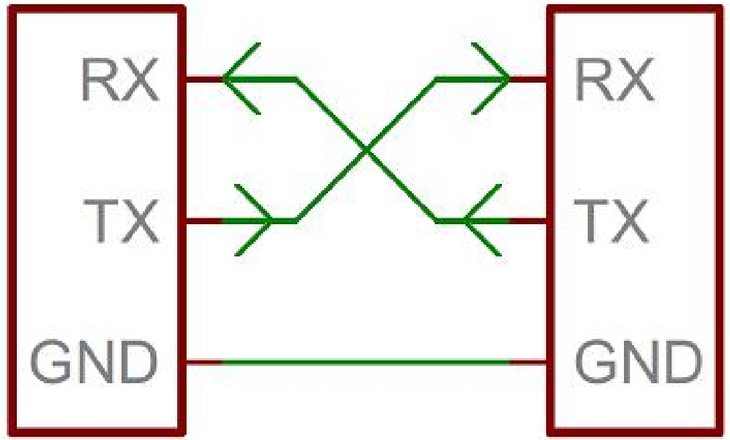
Application dans la réalité

La plupart des boutons de volume traditionnels utilisent un potentiomètre.

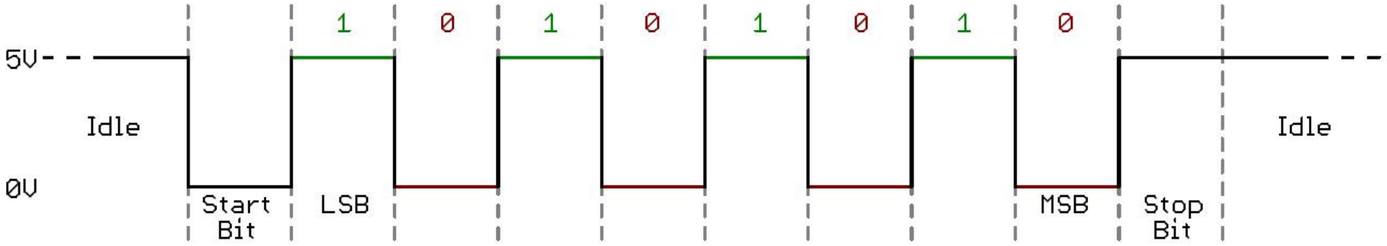
4 Utiliser le Moniteur Série

L’une des choses les plus importantes dans l’EDI Arduino est le “Moniteur Série” (Serial Monitor *en anglais*).

Communication Série



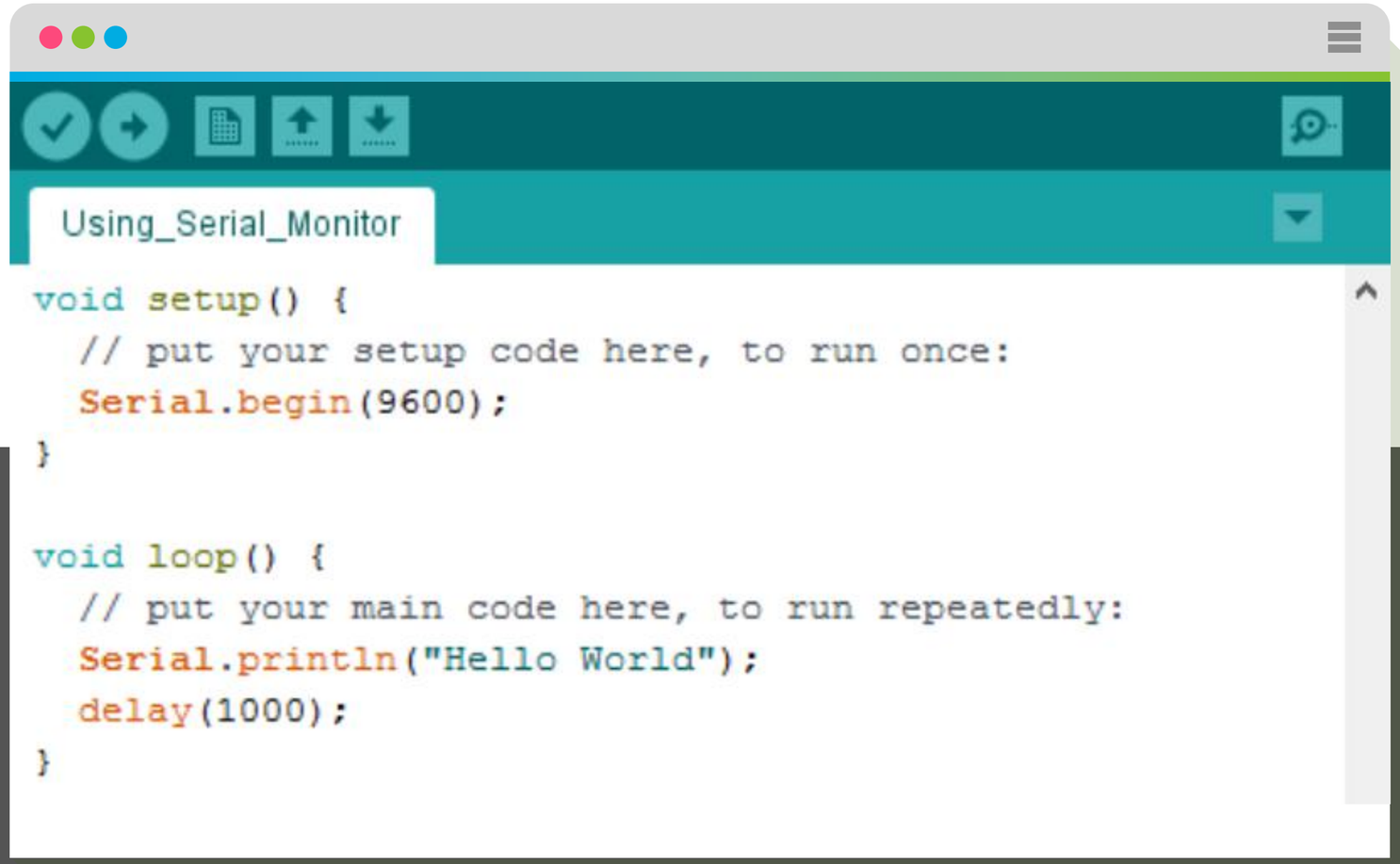
Un bus série ne comprend que deux fils - un pour l'envoi de données et un autre pour la réception. Pour cela, les périphériques série doivent avoir deux broches série: le récepteur, **RX**, et le transmetteur, **TX**.



En communication série, les données sont envoyées les unes après les autres jusqu'à la fin de tous les octets.

Les données sont envoyées de l'expéditeur au destinataire, par exemple en tirant des balles de tennis du tube du conteneur.

AFF IoT Board folder > Circuits > Using_Serial_Monitor

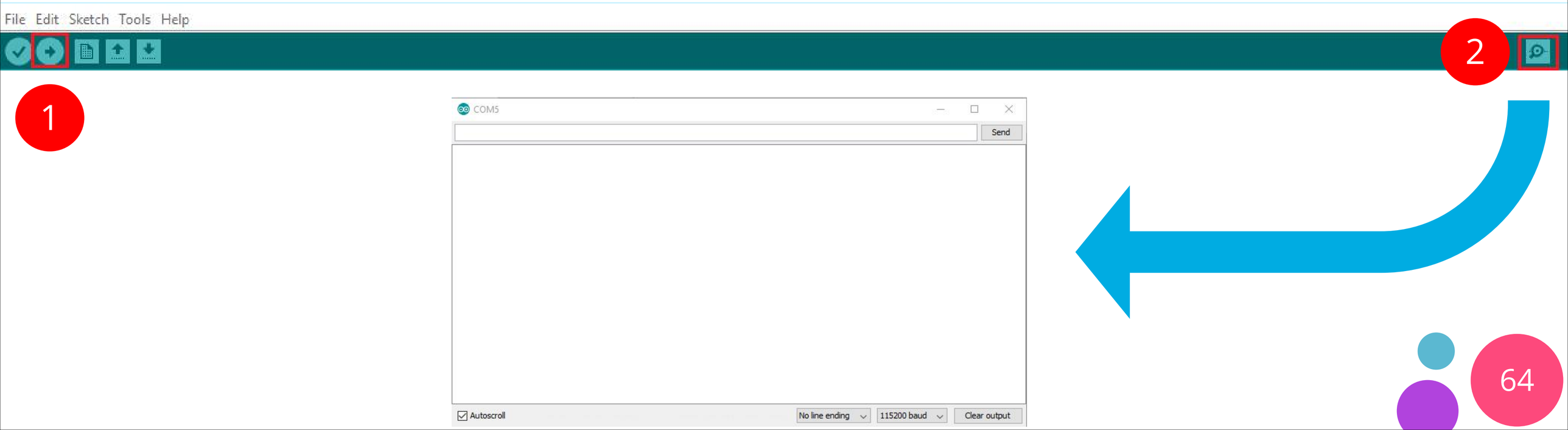


Ouvrir votre croquis:

Using_Serial_Monitor

Moniteur Série

Cet exemple utilise le **Moniteur série** de l'EDI Arduino. Pour l'ouvrir, commencez par télécharger le programme, puis cliquez sur le bouton qui ressemble à une loupe dans un carré. Pour que le moniteur série fonctionne correctement, il doit être réglé sur le même débit en bauds (vitesse en bits par seconde) que le code en cours d'exécution. Le code tourne à 9600 bauds; si le débit en bauds n'est pas défini sur 9600, veuillez le modifier en 9600.



Code à noter..

`Serial.begin(9600)` : Avant d'utiliser le moniteur série, vous devez appeler `Serial.begin()` pour l'initialiser. `9600` est le "débit en bauds" ou la vitesse de communication. Lorsque deux périphériques communiquent, les deux doivent être réglés à la même vitesse. Le moniteur série est également utilisé pour déboguer le code.

`Serial.println("Hello World")` : La commande `Serial.println()` imprime tout ce qui y est écrit entre parenthèse, y compris les variables de tous types, puis passe à la ligne suivante.

Tandis que, `Serial.print()` imprime tout sur la même ligne.

Pour plus d'informations, visitez http://arduino.cc/en/serial/print

Ce que vous **devriez voir**

Vous devriez voir la phrase «Hello World» imprimée sur le moniteur série chaque seconde.

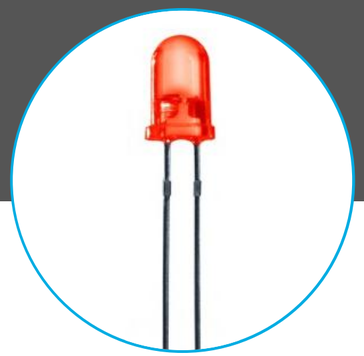
Troubleshooting

Rien n'est affiché
Essayez de quitter le moniteur série, puis redémarrez-le.

Charabia est affiché
Cela se produit parce que le moniteur série reçoit des données à une vitesse différente de celle attendue. Cliquez sur la liste déroulante "*** bauds" et remplacez-la par "9600 bauds".

Rien ne se passe lors de l'ouverture de Moniteur Série
Essayez de trouver la fenêtre du moniteur série dans la barre des tâches de la fenêtre en survolant l'icône Arduino avec la souris, puis en cliquant sur le moniteur série.

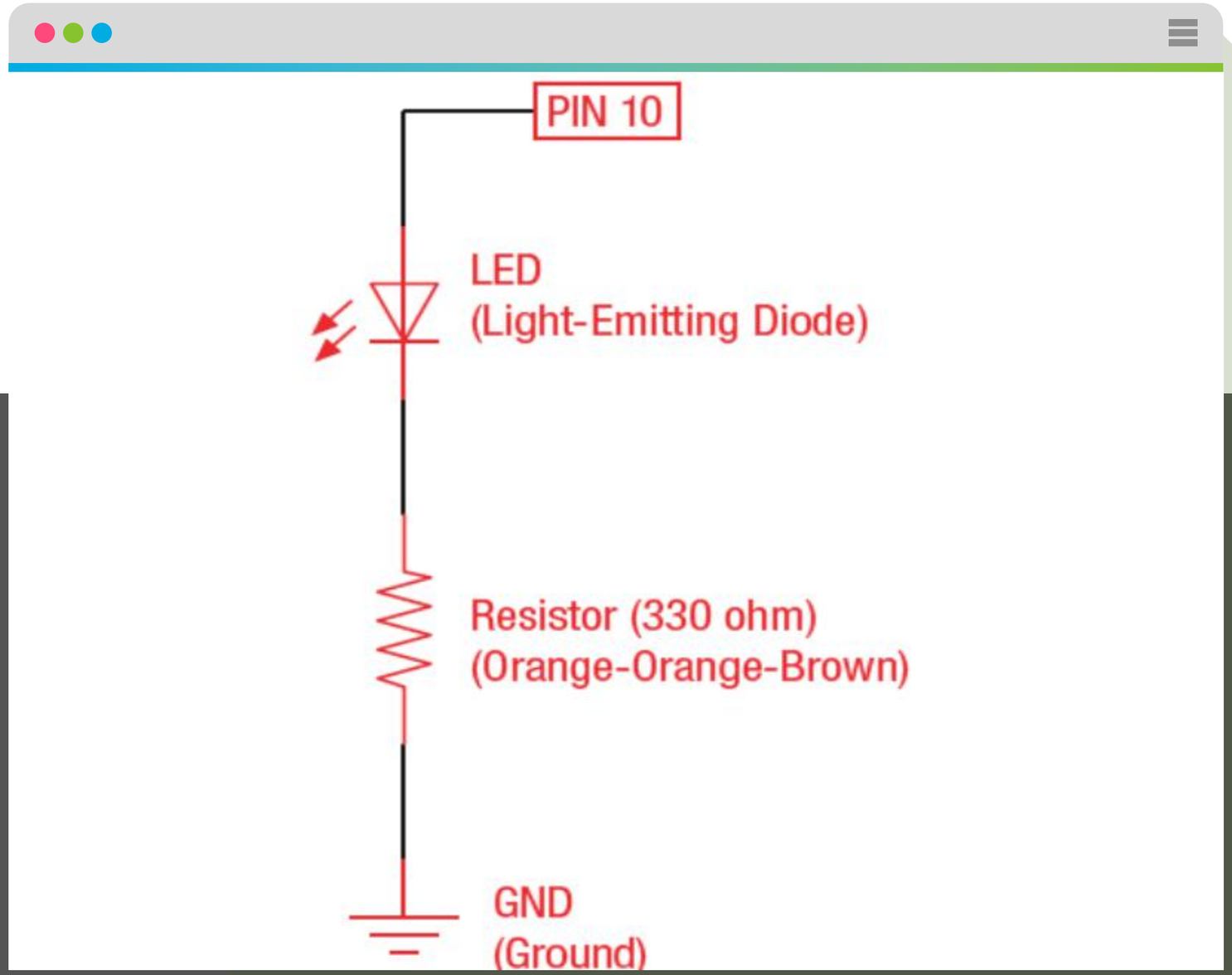
5 Contrôler une LED (via “Internet”)



LED rouge x1

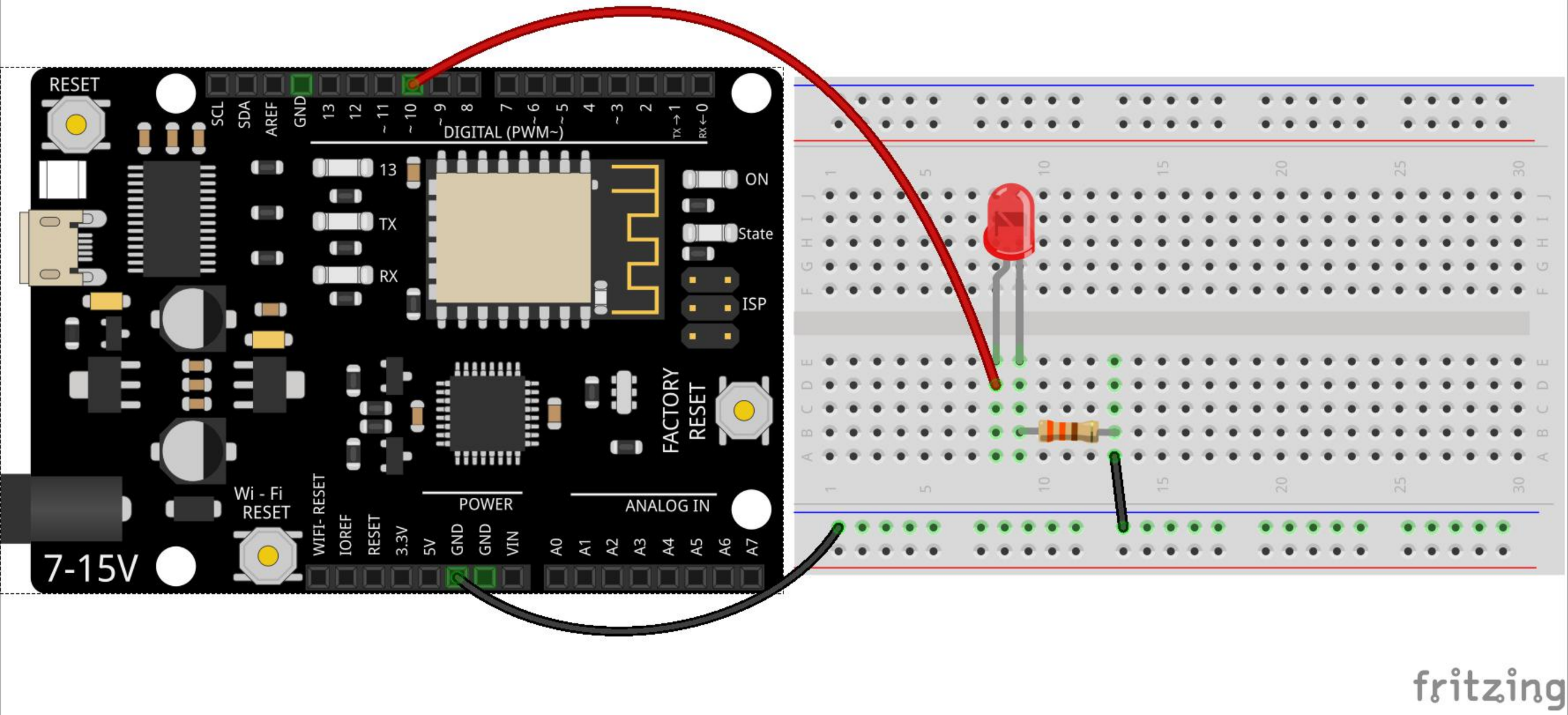
Résistance 330Ω x1

Fils de connexion x3

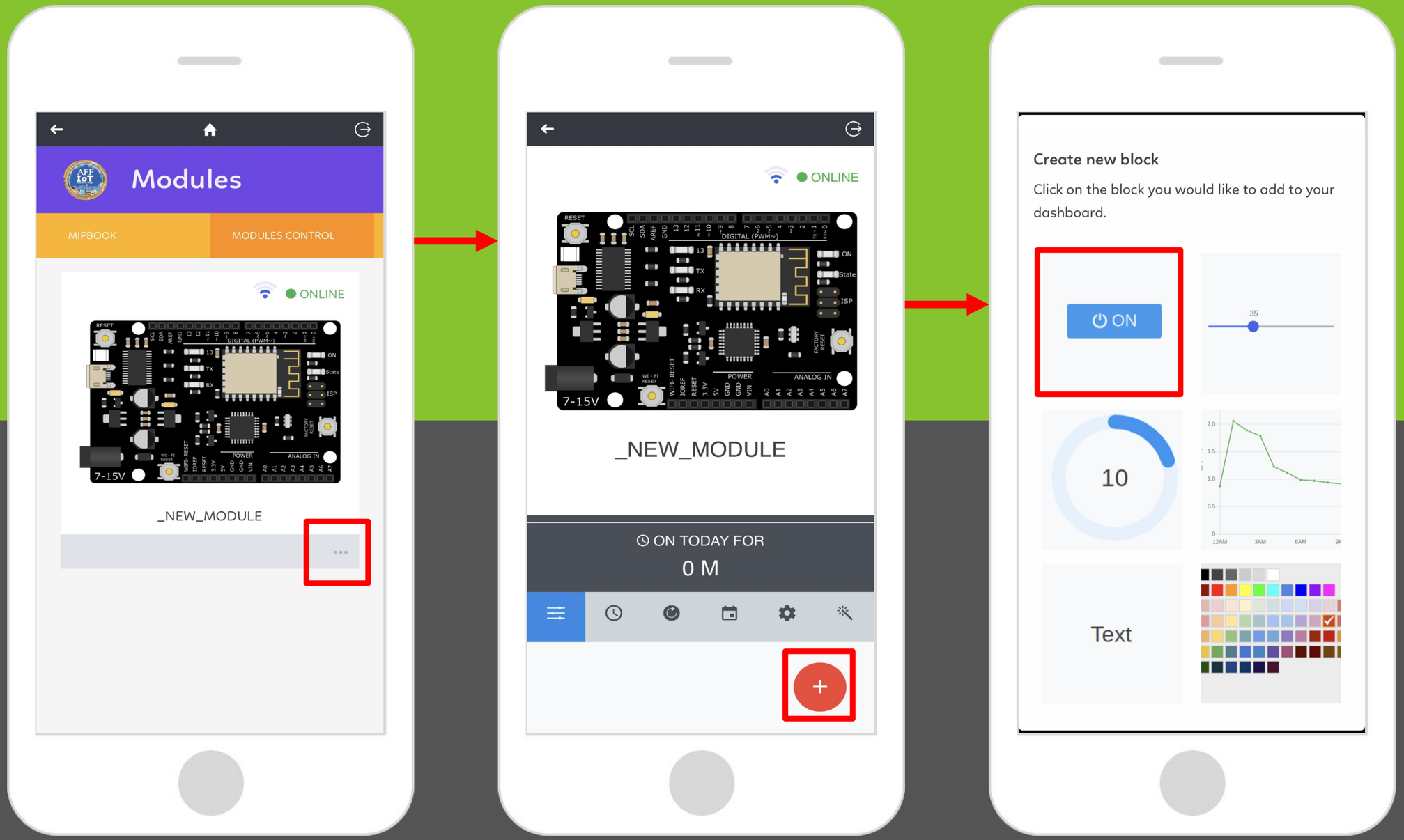


Contrôler une LED

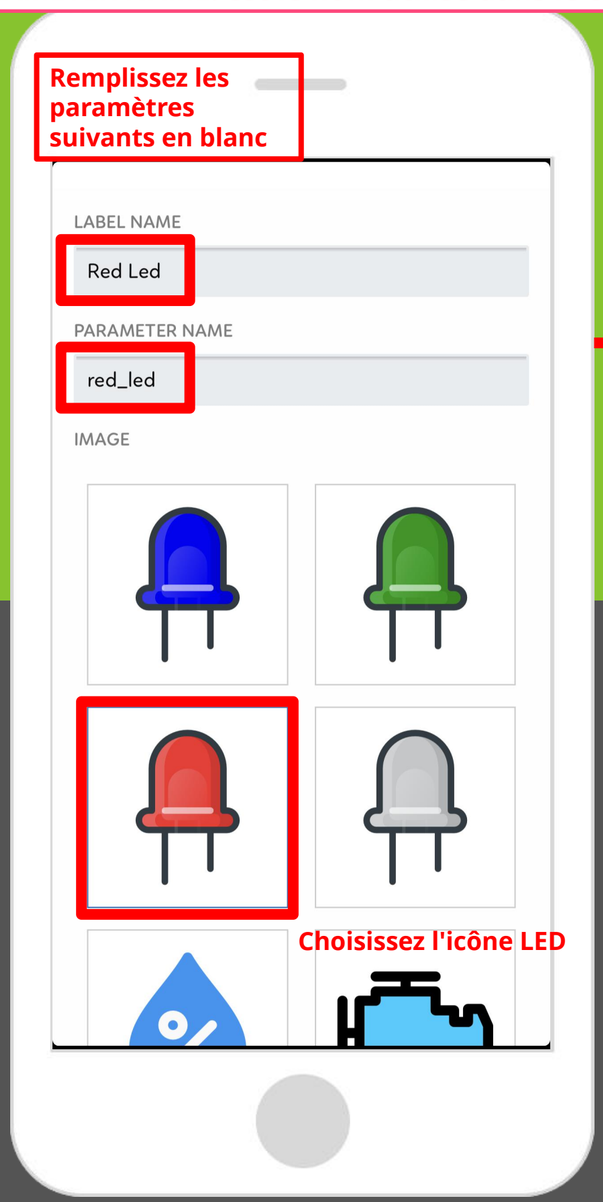
Les LEDs (DEL en français diode électro-luminescente) sont une source lumineuse petite mais puissante, utilisées dans de nombreuses applications différentes. Nous allons d'abord travailler sur le clignotement d'une LED. C'est une fonction simple et fondamentale, mais c'est la base pour d'autres fonctions complexes.

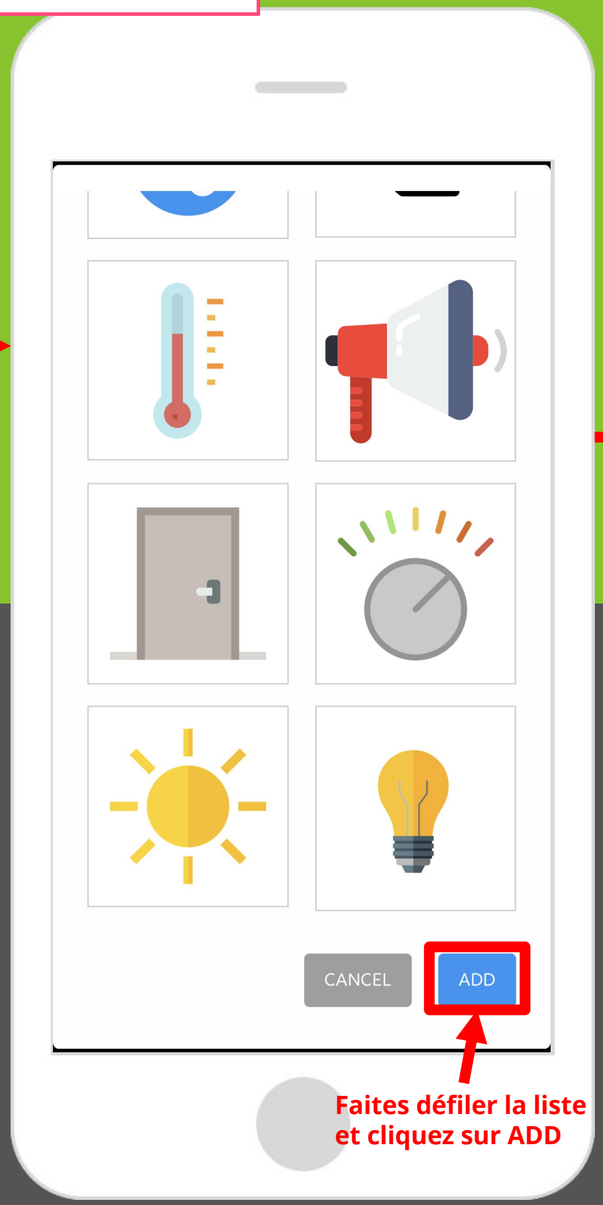

RESET
SCL
SDA
AREF
GND
13
12
~ 11
~ 10
~ 9
8
7
~ 6
~ 5
4
~ 3
2
TX → 1
RX ← 0
DIGITAL (PWM~)
13
TX
RX
ON
State
ISP
FACTORY RESET
Wi - Fi RESET
WIFI- RESET
IOREF
RESET
3.3V
5V
GND
GND
VIN
POWER
A0
A1
A2
A3
A4
A5
A6
A7
ANALOG IN
7-15V
A B C D E F G H I J
1 5 10 15 20 25 30

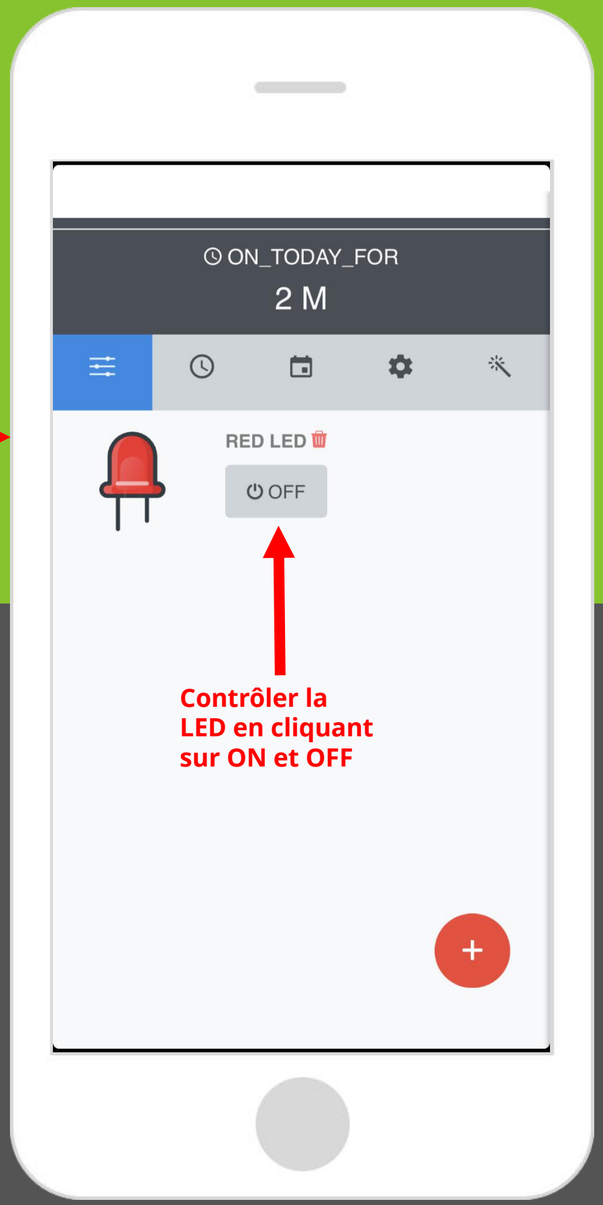
fritzing


Modules
MIPBOOK
MODULES CONTROL
ONLINE
_NEW_MODULE
ONLINE
_NEW_MODULE
ON TODAY FOR
0 M
Create new block
Click on the block you would like to add to your dashboard.
ON
35
10
Text

LABEL NAME: est le nom qui apparaît dans l'application.
PARAMETER NAME: est le nom utilisé dans le code.

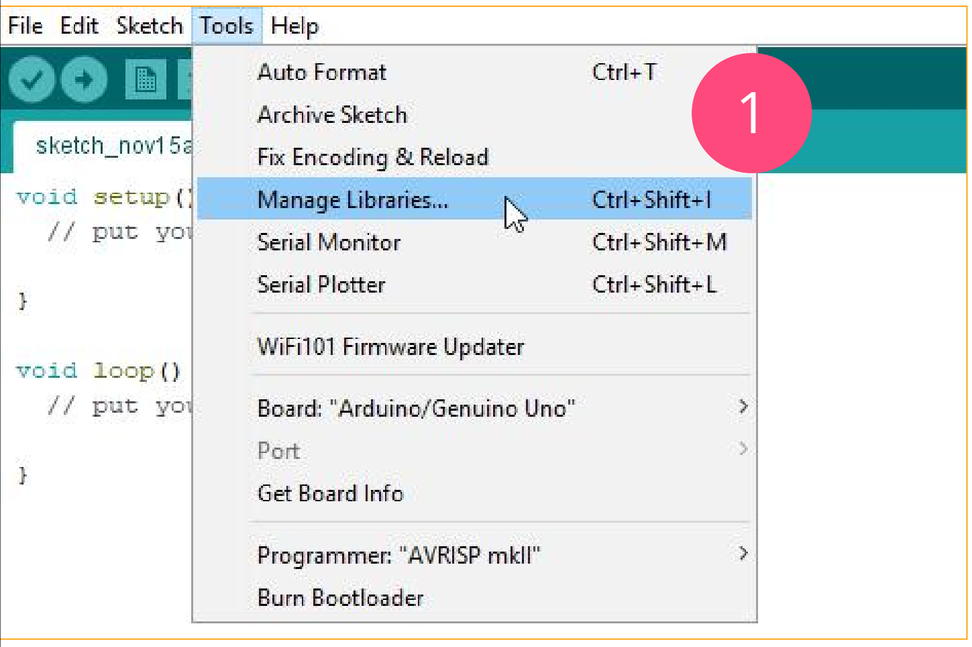


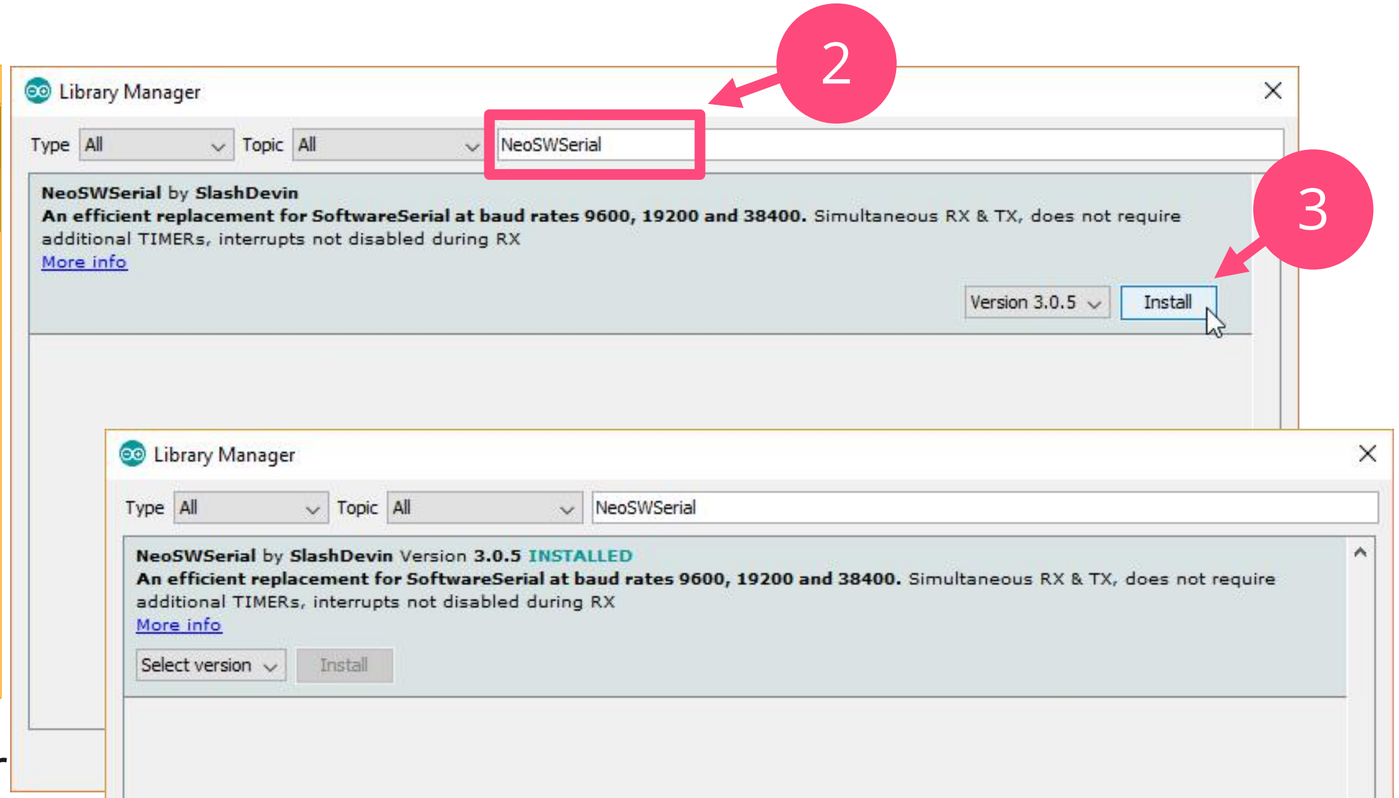




Avant de commencer!

Instructions pour ajouter une nouvelle bibliothèque (obligatoire pour communiquer avec le module Wi-Fi)





1. Ouvrez Arduino IDE et cliquez sur Tools, Manage Libraries.
2. Recherchez «NeoSWSerial».
3. Puis cliquez sur Install.
4. Une fois le téléchargement terminé, cliquez sur Close.

AFF IoT Board folder > Circuits > Controlling_an_LED

C'est ici que vous trouverez le code Arduino pour chaque circuit

Ouvrir votre croquis: Controlling_an_LED

Controlling_an_LED

```
/*Start of mandatory lines of codes in each sketch*/
#define RX A0 // define the Receive pin (RX) to communicate with the WiFi module
#define TX A1 // define the Transmit pin (TX) to communicate with the WiFi module
#include <NeoSWSerial.h> // including the library to use the Software Serial rather than the Hardware Serial (Serial)
NeoSWSerial WiFiModule(RX, TX); //initialize the variable to use in communication with the WiFi module
/*End of mandatory lines of code*/

#define RedLedPin   10

void setup() {
  // put your setup code here, to run once:
  WiFiModule.begin(19200); // begin the communication between the WiFi module and the microcontroller on the board
  pinMode(RedLedPin, OUTPUT); // configure the pin connected to the Red Led to be an output
}

void loop() {
  // put your main code here, to run repeatedly:
  if (WiFiModule.available() > 0) // if the WiFi module receive data from the server
  {
    String Command = WiFiModule.readStringUntil('\n'); // read the command sent from the WiFi module to the microcontroller

    if (Command.indexOf("red_led=1") >= 0) // if the received command contains "red_led=1" turn on the LED
    {
      digitalWrite(RedLedPin, HIGH); // turn on the LED
    }
    if (Command.indexOf("red_led=0") >= 0) // if the received command contains "red_led=0" turn it off
    {
      digitalWrite(RedLedPin, LOW); // turn off the LED
    }
  }
}
//(for more info about indexOf function see https://www.arduino.cc/reference/en/language/variables/data-types/string/functions/indexof/)
```

Code à noter..

`#include <NeoSWSerial.h>`: Cette ligne de code consiste en une importation de la bibliothèque de la communication série logiciel, cette bibliothèque consiste à définir deux broches quelconques en tant que broche de transmission (Tx) et en recevoir une (Rx) d'une communication série entre la carte et un autre périphérique (module Wi-Fi intégré dans la carte AFF IoT).
La fonction de cette bibliothèque est similaire à la normale `Serial` matériel.

`NeoSWSerial WiFiModule(Rx, Tx)`: Ici une variable de type "`NeoSWSerial`" nommée "`WiFiModule`" est déclarée. Elle attribue aux deux broches `A0` et `A1` les broches de transmission et de réception pour la communication série avec le module Wi-Fi.

Code à noter..

`WiFiModule.available()` : cette fonction teste si le module Wi-Fi a reçu une commande d'Internet et retourn le nombre d'octets reçus.
Remarque: s'il n'y a pas de données, la valeur de retour de cette fonction est 0.

`String command = WiFimodule.readStringUntil('\n');` Ici, une variable de type `String` nommée `command` est déclarée et elle sera affectée par le texte de la commande reçu du module Wi-Fi. `readStringUntil('\n')` signifie que la carte continue à recevoir des données jusqu'à ce que le caractère `'\n'` soit reçu (`'\n'` est la représentation de la touche ENTER du clavier).
Remarque: utilisez toujours la même ligne de code que cette ligne pour obtenir la commande envoyée par le module Wi-Fi.

`if(command.indexOf("red_led") >= 0);` La fonction `indexOf()` teste si la chaine de caractère entre parenthèses (comme paramètre) existe dans la chaine de caractère d'appel (le variable `command` dans l'exemple) et retourne l'indice là où il existe. Si la chaîne en paramètre n'existe pas dans la chaîne appelant, la fonction retourn -1.

Ce que vous **devriez voir**

Vous devriez voir votre LED s'allumer et s'éteindre lorsque vous cliquez sur le bouton bascule du tableau de bord dans l'application AFF IoT. Si ce n'est pas le cas, assurez-vous d'avoir correctement assemblé le circuit, vérifié et téléchargé le code sur votre carte ou consultez les conseils de dépannage ci-dessous.

Dépannage

La LED ne s'allume pas?
Les LED ne fonctionnent que dans un sens. Essayez de le sortir et en le tournant de 180 degrés (ne vous inquiétez pas, son installation dans le sens opposé n'endommage pas la LED).

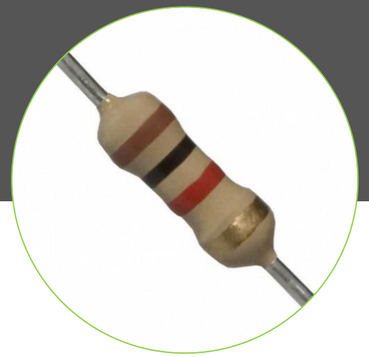
Le programme ne se charge pas?
Cela arrive parfois, la cause la plus probable est de choisir un mauvais port série, vous pouvez changer cela dans Tools → Serial Port.

Application dans la réalité

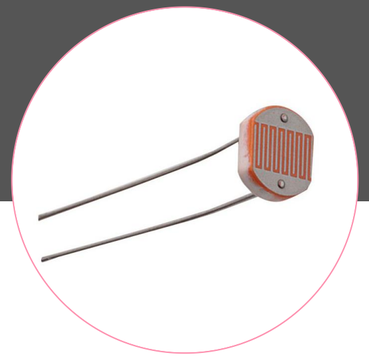
Indicateur marche/arrêt dans divers appareils électroniques

6 Lecture de l'intensité lumineuse (via «Internet»)

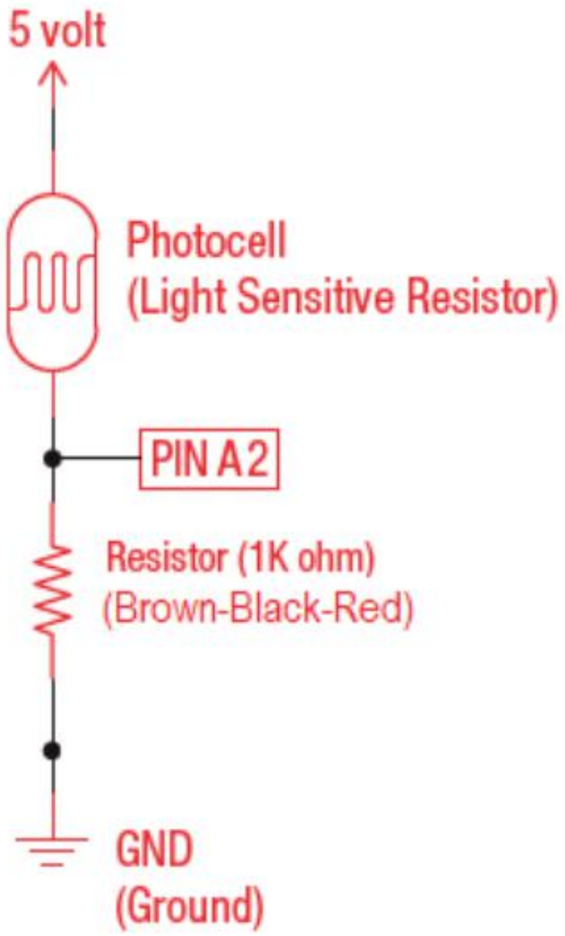
Fils de connexion x5



Résistance 1KΩ x1



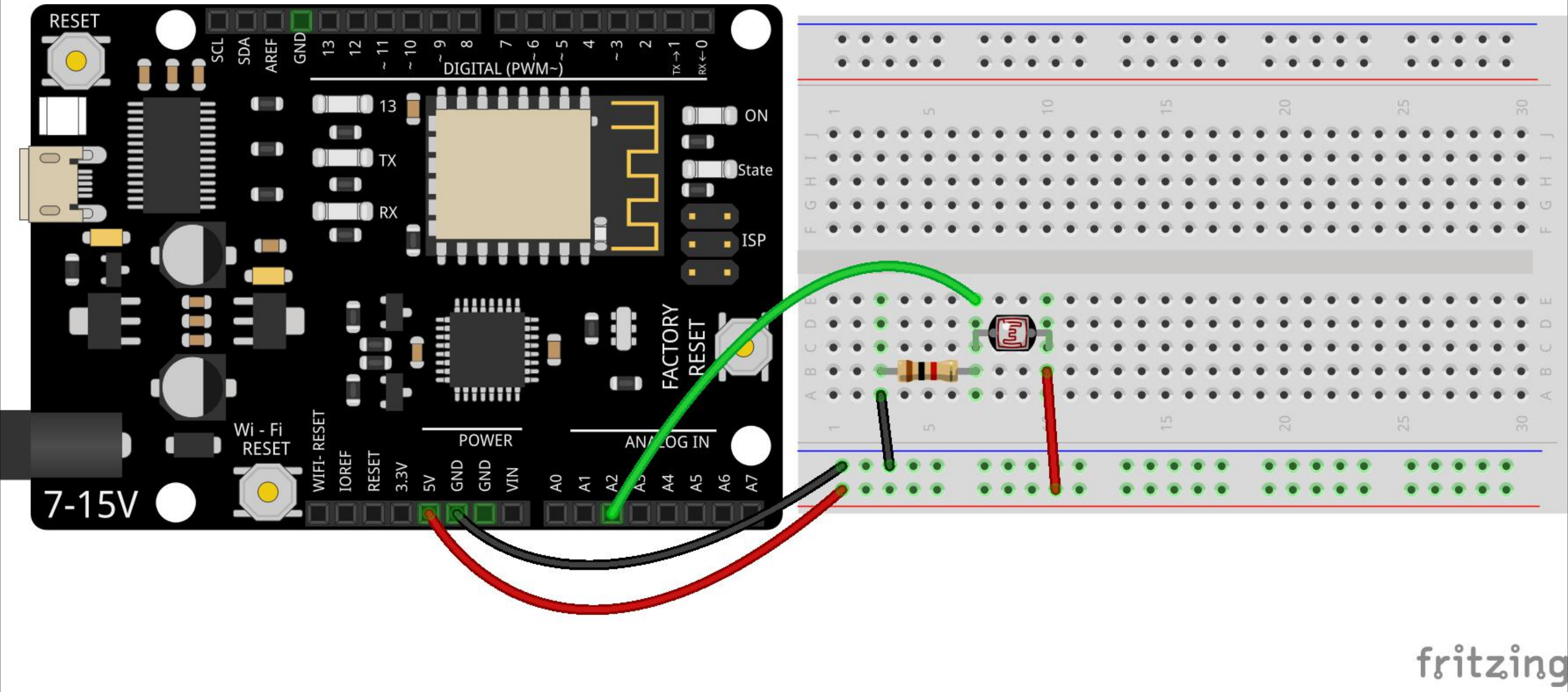
Photocellule x1

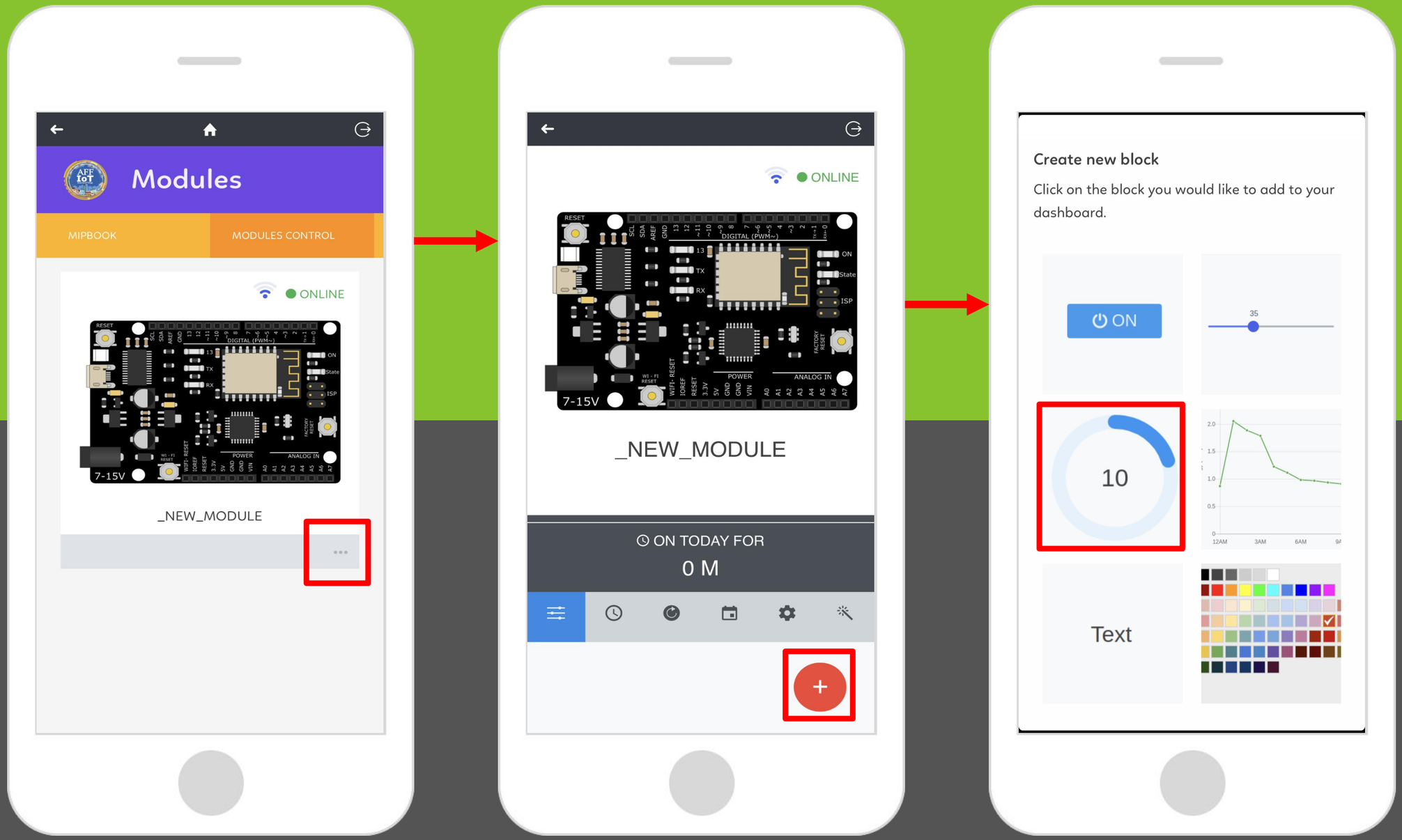


Photocellule

Une cellule photoélectrique est une résistance dont la valeur varie avec le changement d'intensité lumineuse. Lorsqu'elle est brillante, la résistance diminue et la tension (dans l'exemple de circuit) augmente.

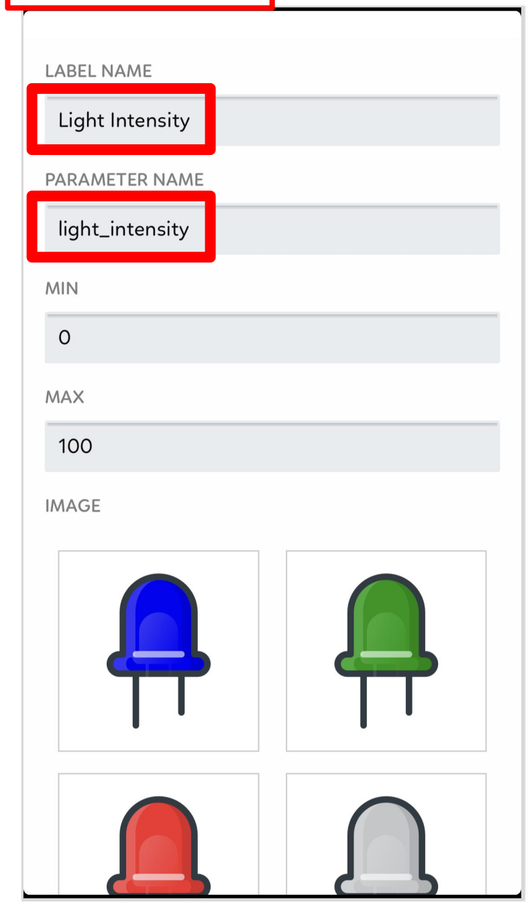
Quand il fait sombre, la résistance augmente et la tension (dans l'exemple de circuit) diminue.

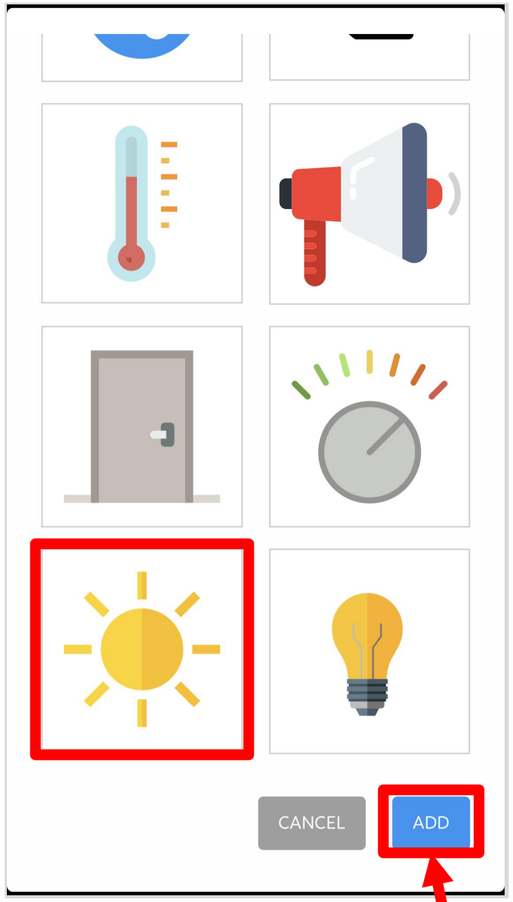

RESET
SCL
SDA
AREF
GND
13
12
~ 11
~ 10
~ 9
8
7
~ 6
~ 5
4
~ 3
2
TX → 1
RX ← 0
DIGITAL (PWM~)
13
TX
RX
ON
State
ISP
FACTORY RESET
Wi - Fi RESET
WIFI- RESET
IOREF
RESET
3.3V
5V
GND
GND
VIN
POWER
A0
A1
A2
A3
A4
A5
A6
A7
ANALOG IN
7-15V
fritzing


Modules
MIPBOOK
MODULES CONTROL
ONLINE
_NEW_MODULE
ONLINE
_NEW_MODULE
ON TODAY FOR
0 M
Create new block
Click on the block you would like to add to your dashboard.
ON
10
Text

LABEL NAME: est le nom qui apparaît dans l'application.
PARAMETER NAME: est le nom utilisé dans le code.

Remplissez les paramètres suivants en blanc





Faites défiler la liste et cliquez sur ADD

vous pouvez surveiller l'intensité de la lumière dans la chambre

AFF IoT Board folder > Circuits > Reading_Light_Intensity

Reading_Light_Intensity

```
/*Start of mandatory lines of codes in each sketch*/
#define RX A0 // define the Receive pin (RX) to communicate with the WiFi module
#define TX A1 // define the Transmit pin (TX) to communicate with the WiFi module
#include <NeoSWSerial.h> // including the library to use the Software Serial rather than the Hardware Serial (Serial)
NeoSWSerial WiFiModule(RX, TX); //initialize the variable to use in communication with the WiFi module
/*End of mandatory lines of code*/

#define LightSensorPin   A2

void setup() {
  // put your setup code here, to run once:
  WiFiModule.begin(19200); // begin the communication between the WiFi module and the microcontroller on the board
}

void loop() {
  // put your main code here, to run repeatedly:
  int lightIntensity; // declare an integer to read the voltage

  lightIntensity = analogRead(LightSensorPin); // read the actual converted value (between 0 and 1023)
  lightIntensity = map(lightIntensity, 0, 1023, 0, 100); // transform the scale from 0 and 1023 to 0% and 100%

  WiFiModule.println("light_intensity=" + String(lightIntensity)); // send the light intensity value to the server

  delay(3000); // wait for 3 second (3000ms = 3s)
}
```

Ouvrir votre croquis:
Reading_Light_Intensity

Ce que vous **devriez voir**

Vous devriez voir la valeur de jauge augmenter ou diminuer en fonction de la quantité de lumière que lit la photocellule. Si cela ne fonctionne pas, assurez-vous d'avoir correctement assemblé le circuit, vérifié et téléchargé le code sur votre carte ou consultez les conseils de dépannage ci-dessous.

Troubleshooting

Il ne répond pas aux changements de lumière
Etant donné que l'espacement des fils sur la cellule photoélectrique n'est pas standard, il est facile de l'égarer sur la plaque d'essai. Vérifiez qu'il est au bon endroit.

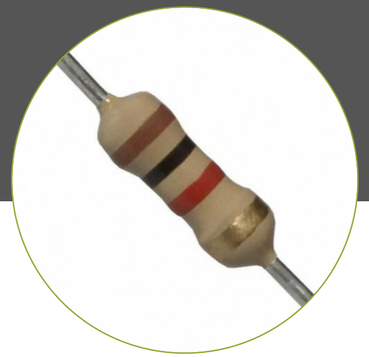
Pas encore assez efficace
Vous êtes peut-être dans une salle trop lumineux ou trop sombre.
Essayez d'allumer ou d'éteindre les lumières pour voir si cela aide. Ou si vous avez une lampe de poche à proximité, essayez-la.

Application dans la réalité

Un lampadaire utilise un capteur de lumière pour détecter le moment d'allumer la lumière

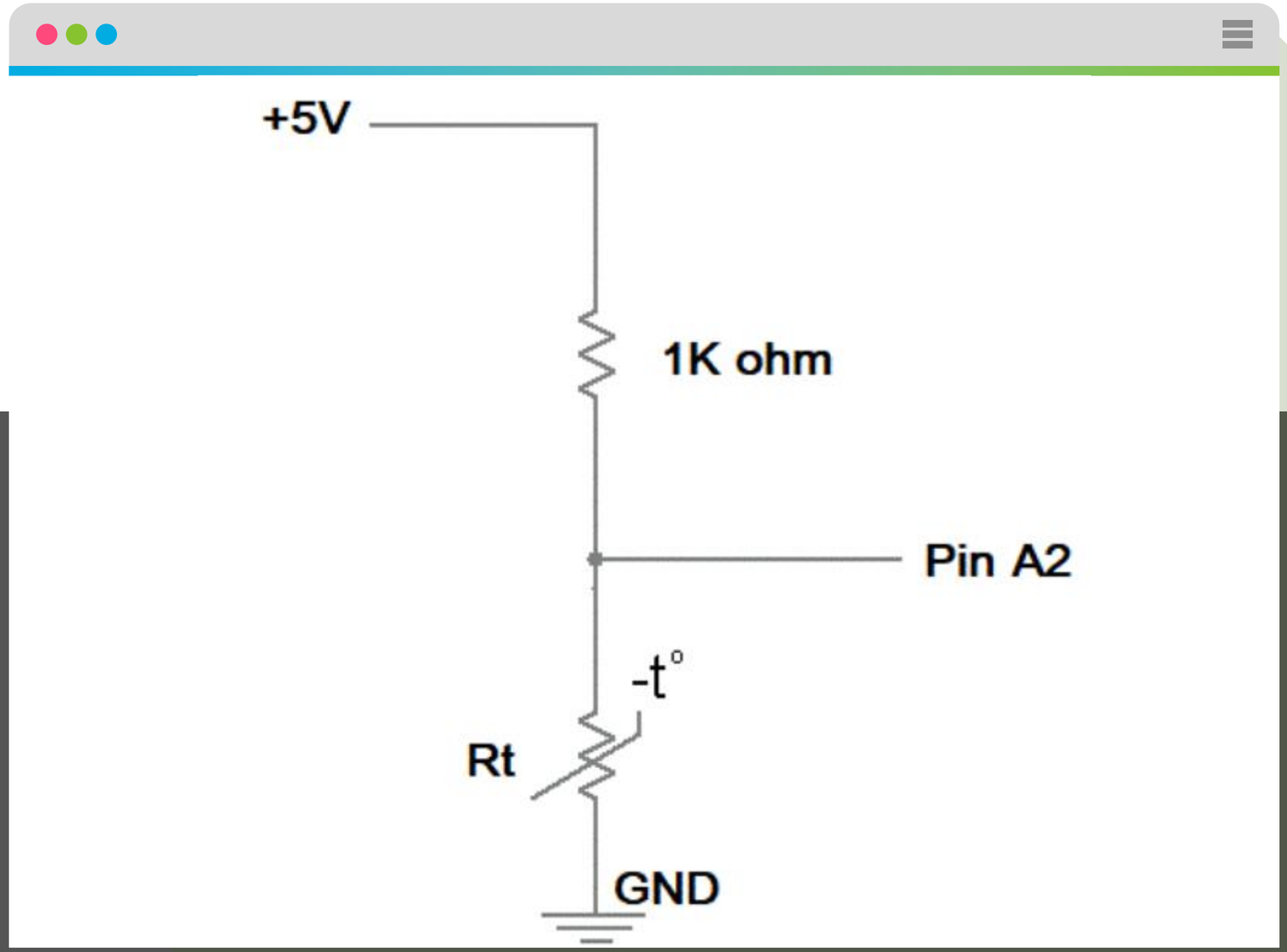
7 Surveillance de la température (via «Internet»)

Thermistance x1



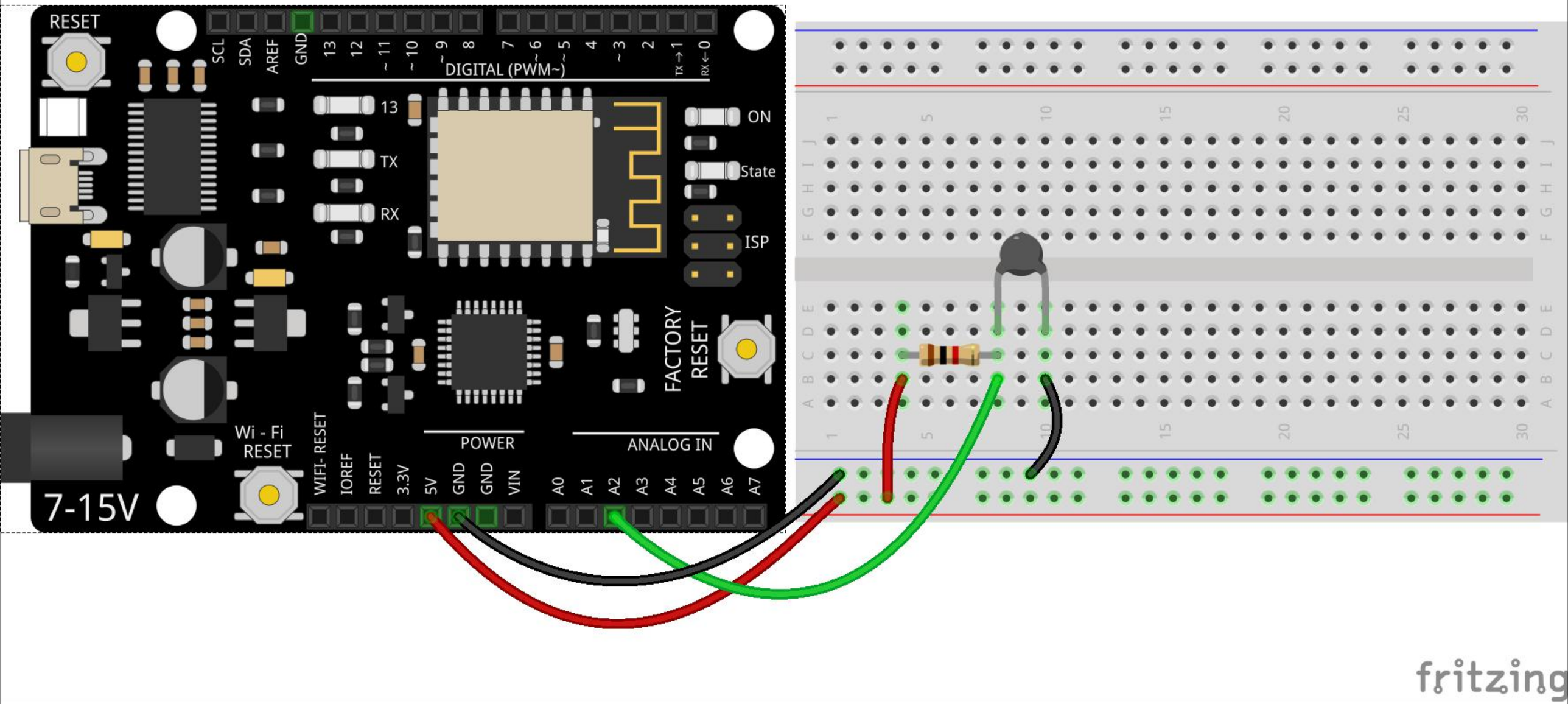
Résistance 1KΩ x1

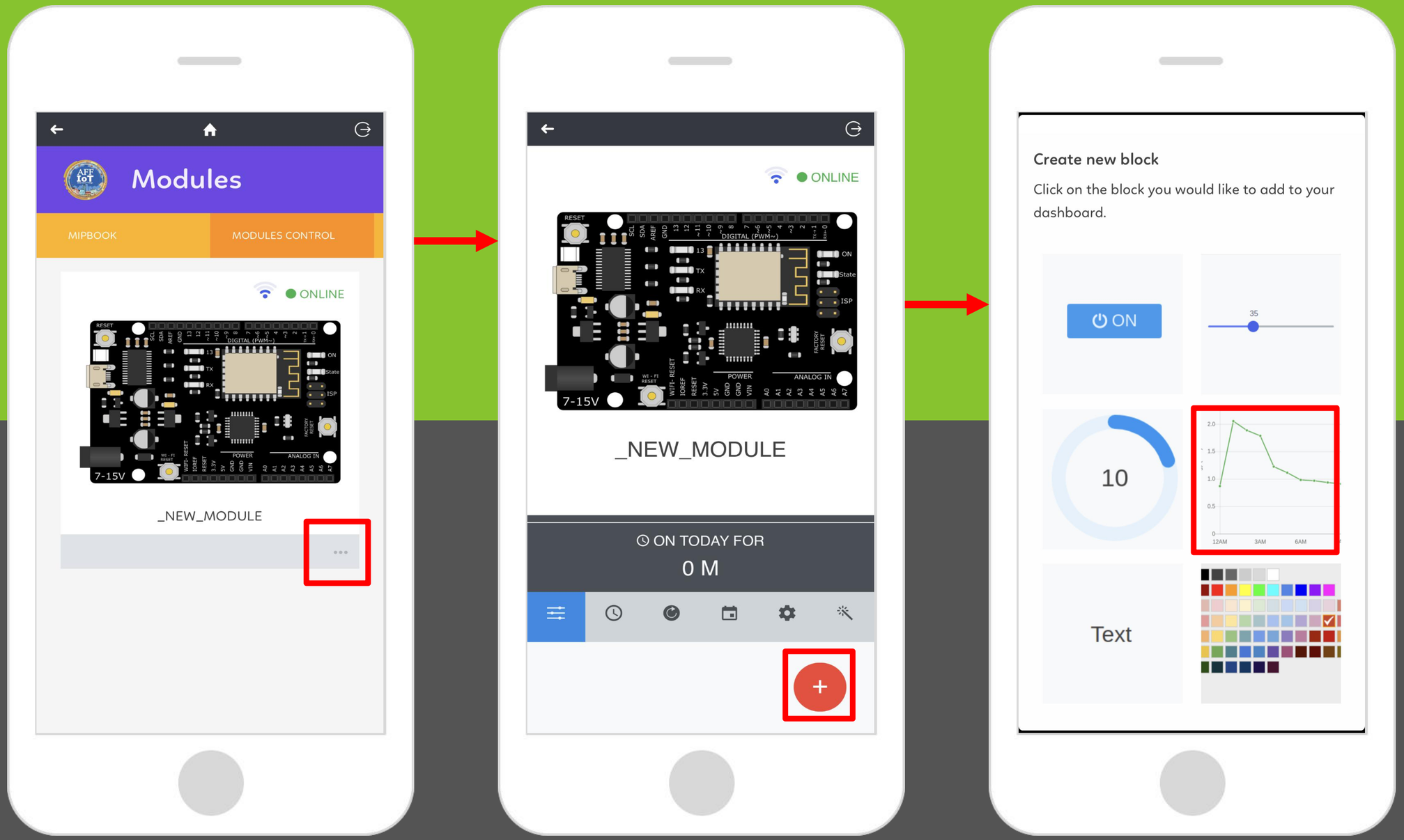
Fils de connexion x5



Thermistance

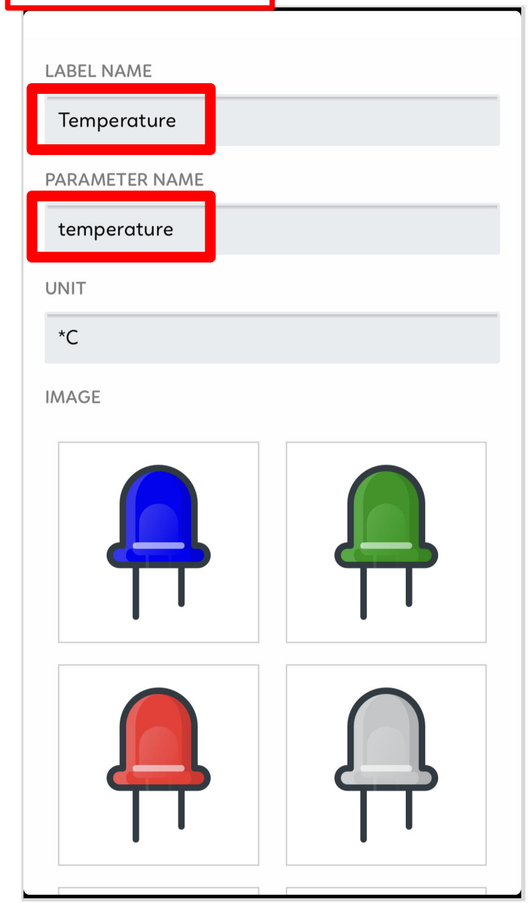
Une **thermistance** est un type de résistance dont la résistance dépend de la température, plus sensible que les résistances standard. Le mot est un mots-valise de thermique et de résistance. Les thermistances sont largement utilisées en tant que capteurs de température (type de coefficient de température négatif ou type NTC), en tant qu'éléments chauffants à autorégulation (type de coefficient de température positif ou type PTC). Avec les thermistances NTC, la résistance diminue lorsque la température augmente.

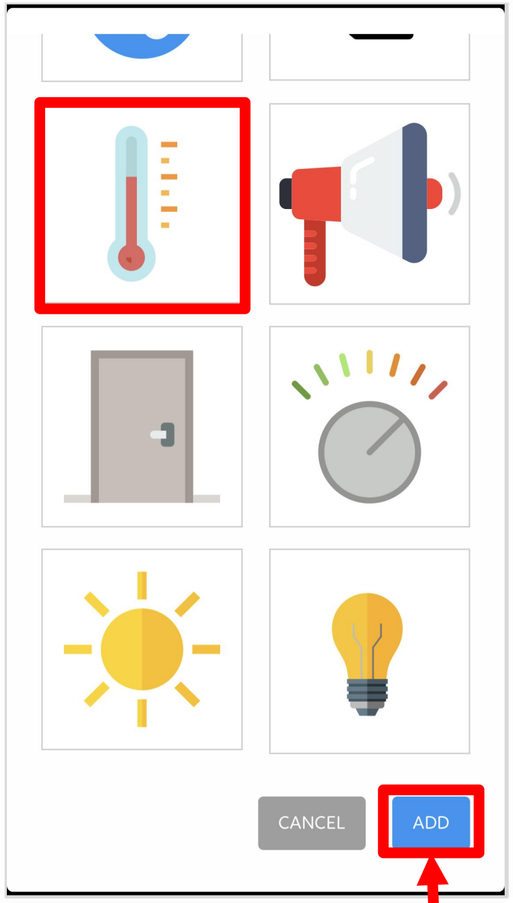

RESET
SCL
SDA
AREF
GND
13
12
~ 11
~ 10
~ 9
8
7
~ 6
~ 5
4
~ 3
2
TX → 1
RX ← 0
DIGITAL (PWM~)
13
TX
RX
ON
State
ISP
FACTORY
RESET
Wi - Fi
RESET
WIFI- RESET
IOREF
RESET
3.3V
5V
GND
GND
VIN
POWER
ANALOG IN
A0
A1
A2
A3
A4
A5
A6
A7
7-15V


Modules
MIPBOOK
MODULES CONTROL
ONLINE
_NEW_MODULE
ONLINE
_NEW_MODULE
ON TODAY FOR
0 M
Create new block
Click on the block you would like to add to your dashboard.
ON
35
10
Text

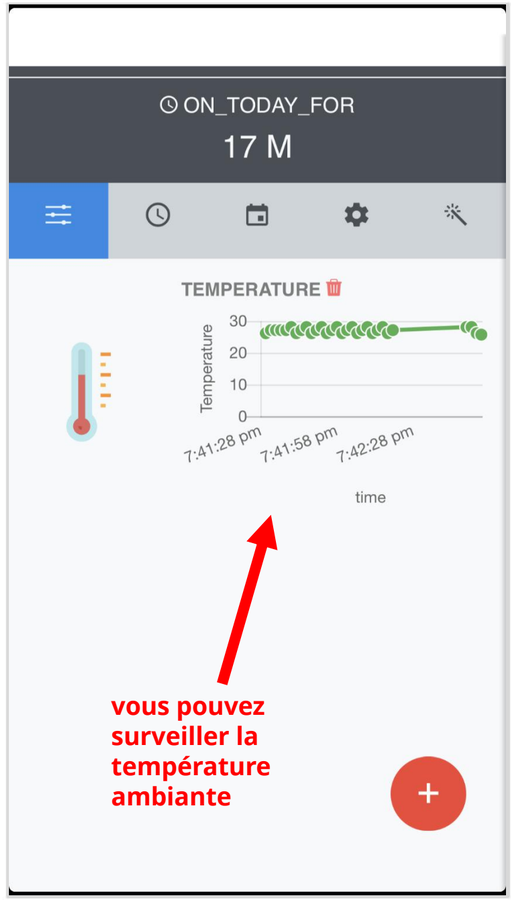
LABEL NAME: est le nom qui apparaît dans l'application.
PARAMETER NAME: est le nom utilisé dans le code.

Remplissez les paramètres suivants en blanc





Faites défiler la liste et cliquez sur ADD



vous pouvez surveiller la température ambiante

AFF IoT Board > Circuits > Monitoring_Temperature

Monitoring_Temperature

```
/*Start of mandatory lines of codes in each sketch*/
#define RX A0 // define the Receive pin (RX) to communicate with the WiFi module
#define TX A1 // define the Transmit pin (TX) to communicate with the WiFi module
#include <NeoSWSerial.h> // including the library to use the Software Serial rather than the Hardware Serial (Serial)
NeoSWSerial WiFiModule(RX, TX); //initialize the variable to use in communication with the WiFi module
/*End of mandatory lines of code*/

#define TemperatureSensorPin   A2

void setup() {
  // put your setup code here, to run once:
  WiFiModule.begin(19200); // begin the communication between the WiFi module and the microcontroller on the board
  Serial.begin(9600); // begin the communication between the board and the PC
}

void loop() {
  // put your main code here, to run repeatedly:
  float Voltage; // declare a decimal variable to store the read voltage
  int Temperature; // declare an integer variable to calculate the temperature
  Voltage = analogRead(TemperatureSensorPin) * 0.0048828125; // voltage = analog_value * (5/1024);

  Temperature = -21.231 * (Voltage - 3.765);
  // this equation is the linearized form of the temperature equation between 0 and 50 degrees (specific to the thermistor in the kit)

  Serial.print("temperature: "); // print on the serial monitor "Temperature: " and stay on the same line
  Serial.println(Temperature); // print on the serial monitor the actual temperature then go to the next line

  WiFiModule.println("temperature=" + String(Temperature)); // send the Temperature value to the server

  delay(3000); // wait for 3 second (3000ms = 3s)
}
```

Ouvrir votre croquis:
Monitoring_Temperature

Ce que vous **devez voir**

Vous devriez pouvoir lire la température détectée par votre capteur de température, sur le moniteur série de l'EDI Arduino et sur votre application. Si cela ne fonctionne pas, assurez-vous d'avoir correctement assemblé le circuit, vérifié et téléchargé le code sur votre carte ou consultez les conseils de dépannage.

Dépannage

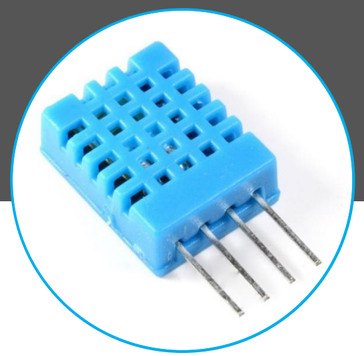
Charabia est affiché
Cela se produit parce que le moniteur série reçoit des données à une vitesse différente de celle attendue. Cliquez sur la liste déroulante "*** bauds" et remplacez-la par "9600 bauds".

La valeur de la température ne change pas
Essayez de frotter le capteur avec vos doigts pour le réchauffer ou appuyez le avec un sac de glace pour le refroidir.

Application dans la réalité

Les systèmes de climatisation utilisent un capteur de température pour surveiller et maintenir les paramètres de refroidissement/chauffage.

8 Measurer l'Humidité et la Température (via «Internet»)

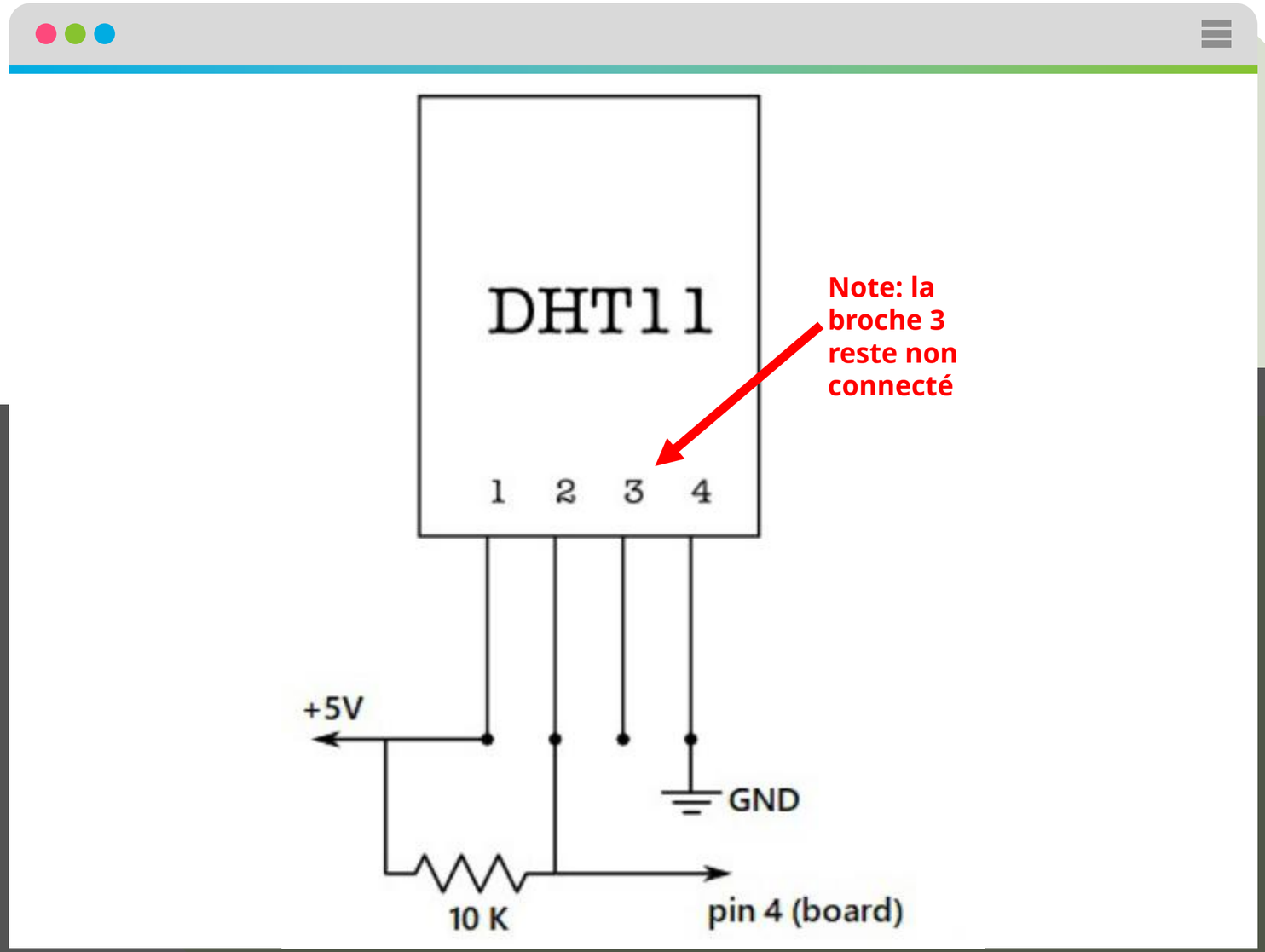


DHT11 x1

"Capteur d'humidité et de Temperature"

Résistance 10KΩ x1

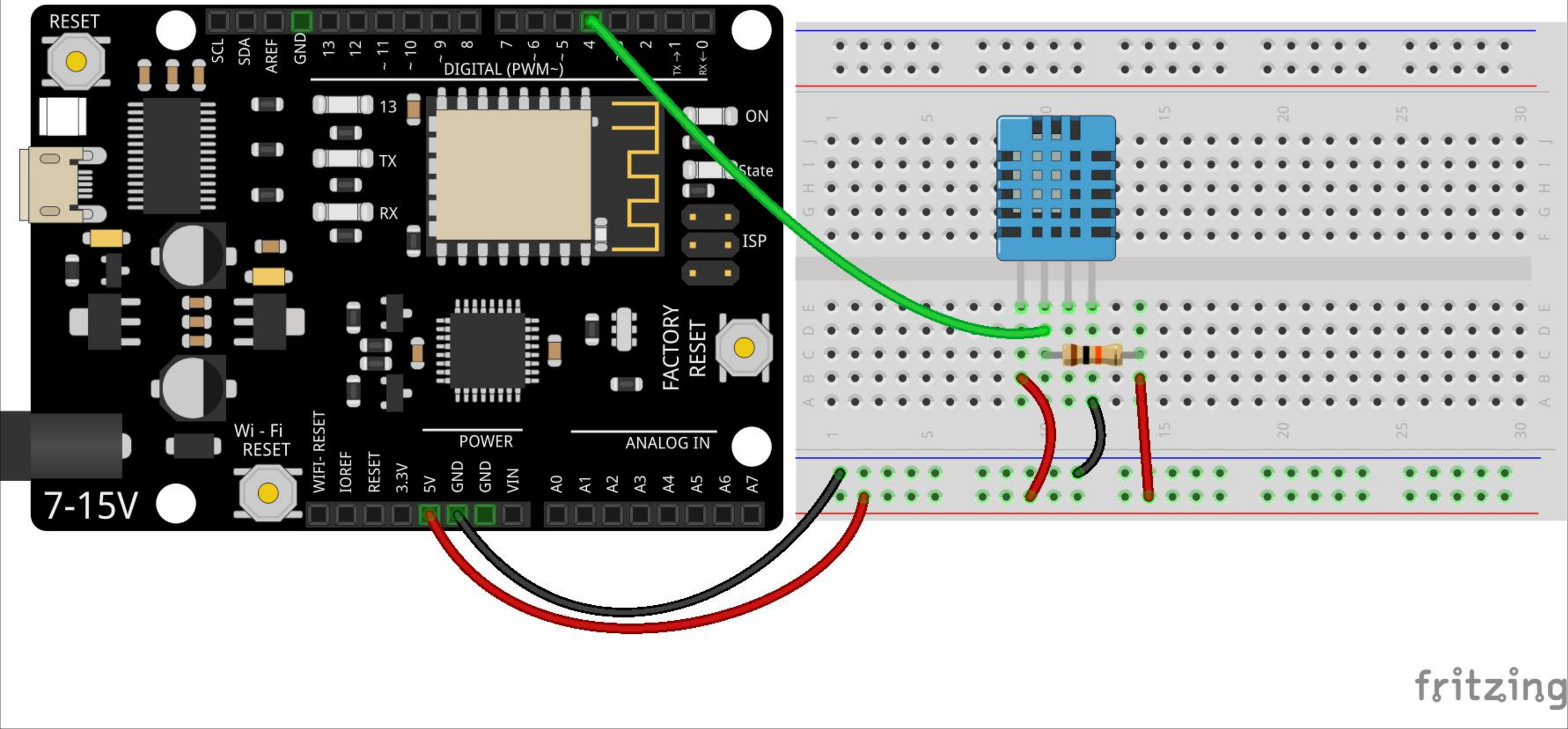
Fils de connexion x6

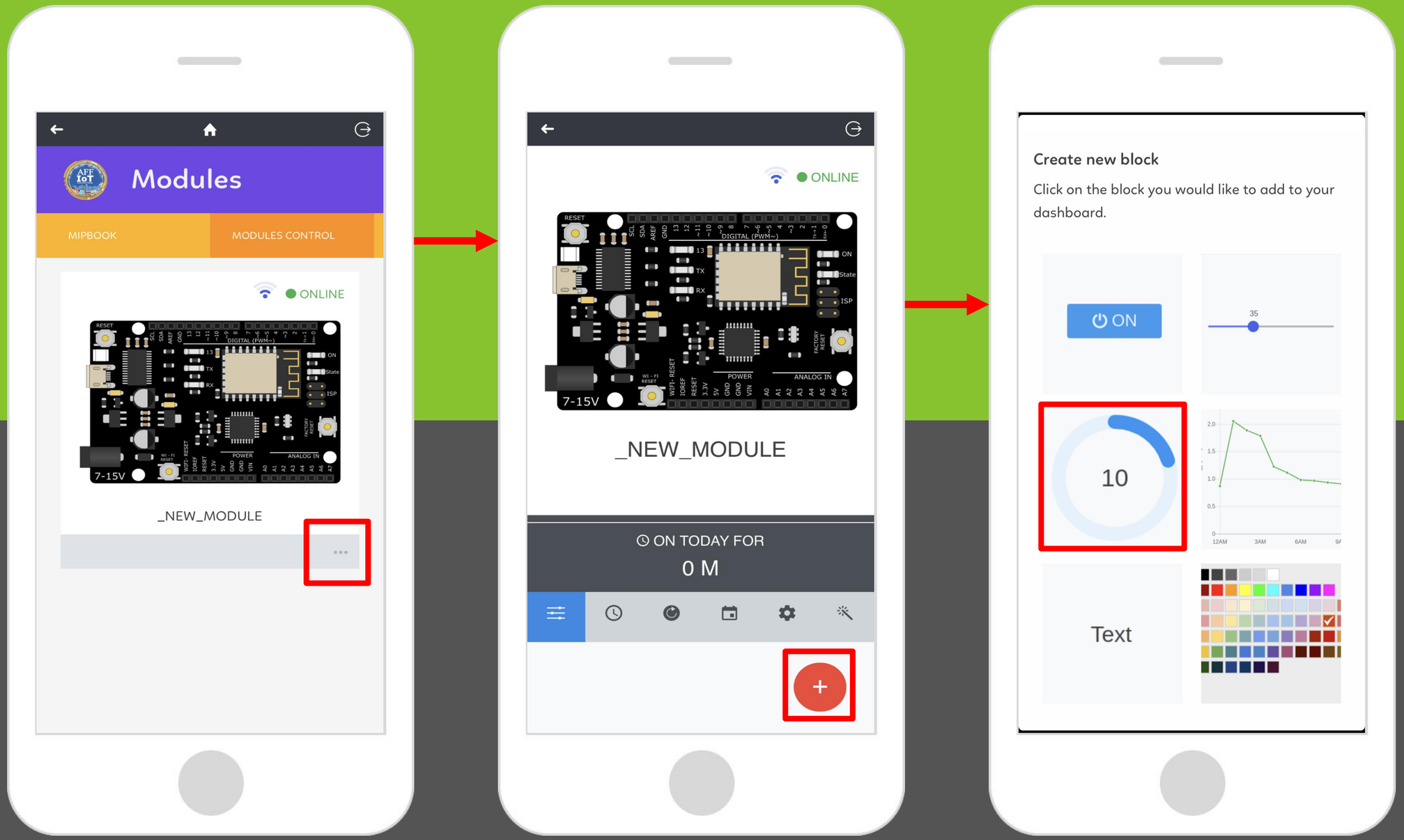


Capteur DHT11

DHT11 Capteur de Temperature et d'Humidité

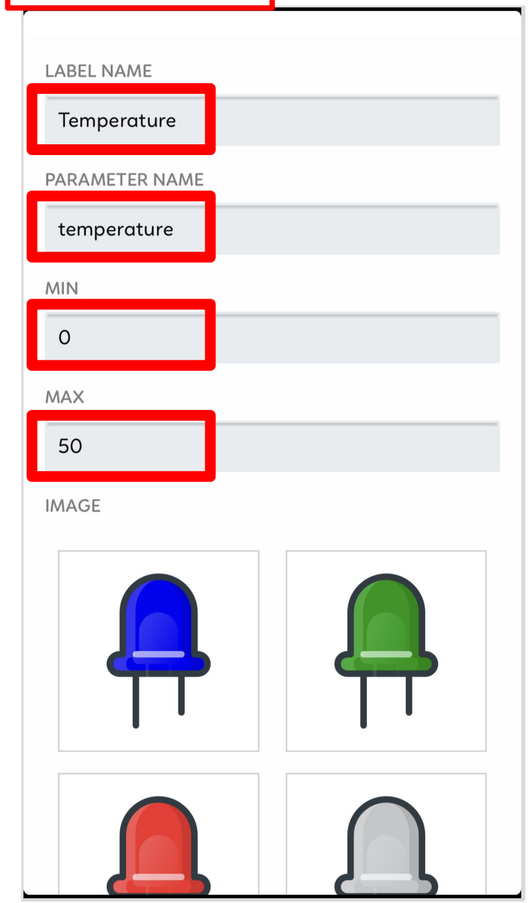
Comporte un complexe de capteurs de température et d'humidité avec une sortie de signal numérique calibrée.
Le capteur comprend un composant de mesure d'humidité et un composant NTC de mesure de température.

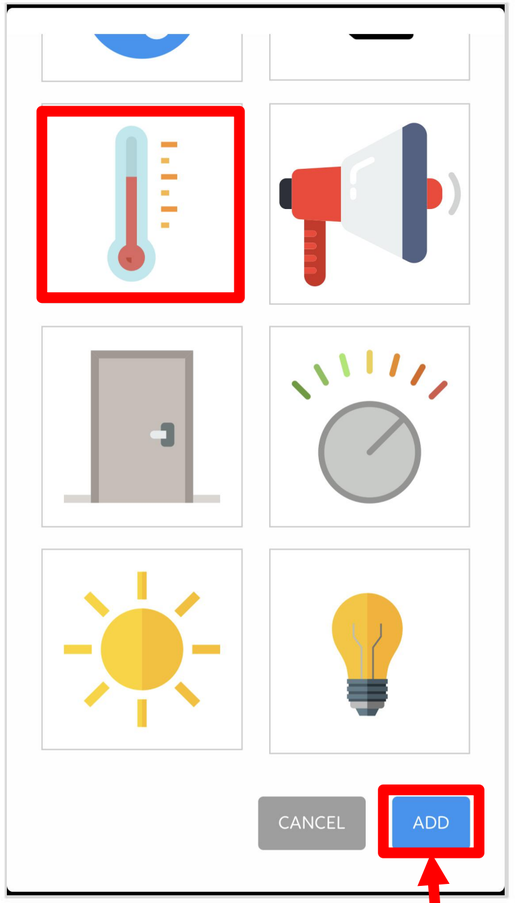

RESET
SCL
SDA
AREF
GND
13
12
~ 11
~ 10
~ 9
8
7
~ 6
~ 5
4
2
TX → 1
RX ← 0
DIGITAL (PWM~)
13
TX
RX
ON
State
ISP
FACTORY RESET
Wi - Fi RESET
WIFI- RESET
IOREF
RESET
3.3V
5V
GND
GND
VIN
POWER
ANALOG IN
A0
A1
A2
A3
A4
A5
A6
A7
7-15V
fritzing


Modules
MIPBOOK
MODULES CONTROL
ONLINE
_NEW_MODULE
ONLINE
_NEW_MODULE
ON TODAY FOR
0 M
Create new block
Click on the block you would like to add to your dashboard.
ON
35
10
Text

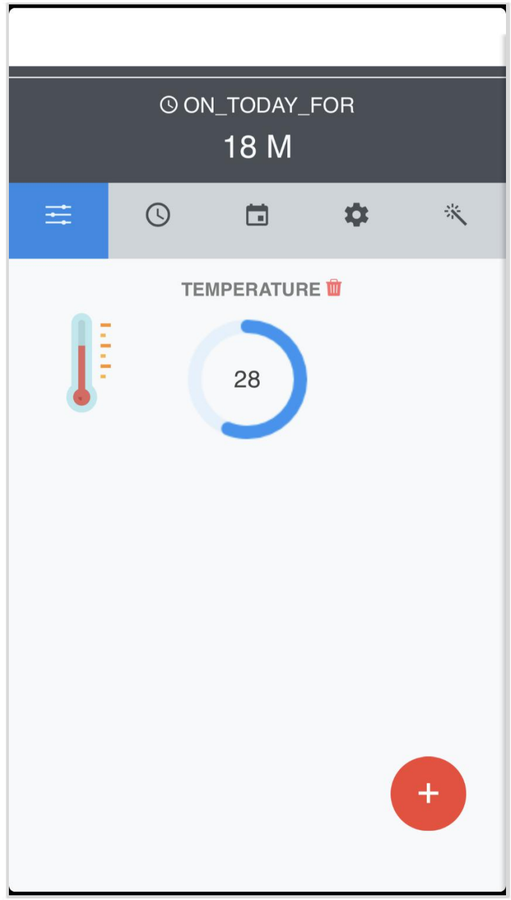
LABEL NAME: est le nom qui apparaît dans l'application.
PARAMETER NAME: est le nom utilisé dans le code.

Remplissez les paramètres suivants en blanc



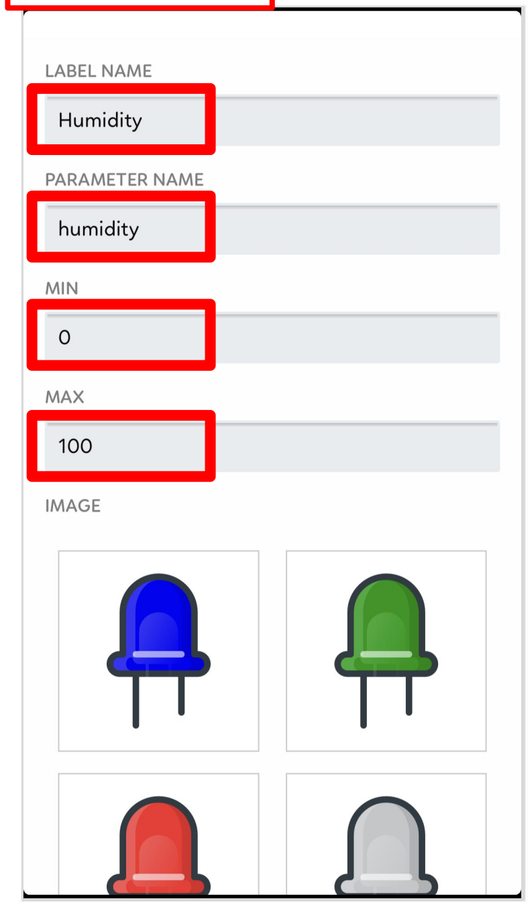


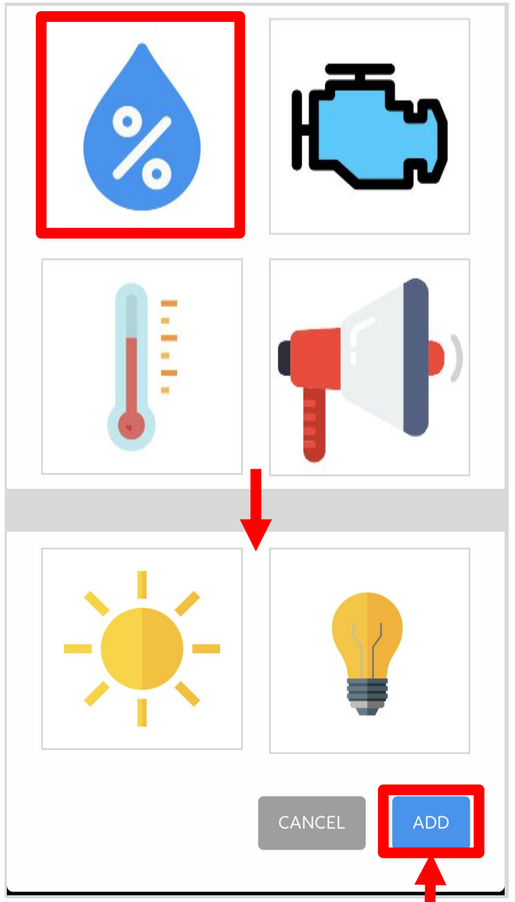
Faites défiler la liste et cliquez sur ADD



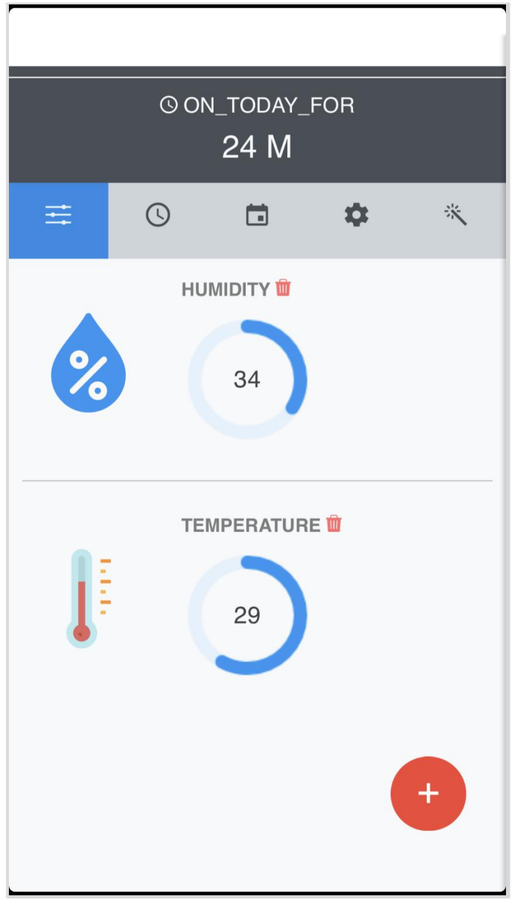
LABEL NAME: est le nom qui apparaît dans l'application.
PARAMETER NAME: est le nom utilisé dans le code.

Remplissez les paramètres suivants en blanc





Faites défiler la liste et cliquez sur ADD

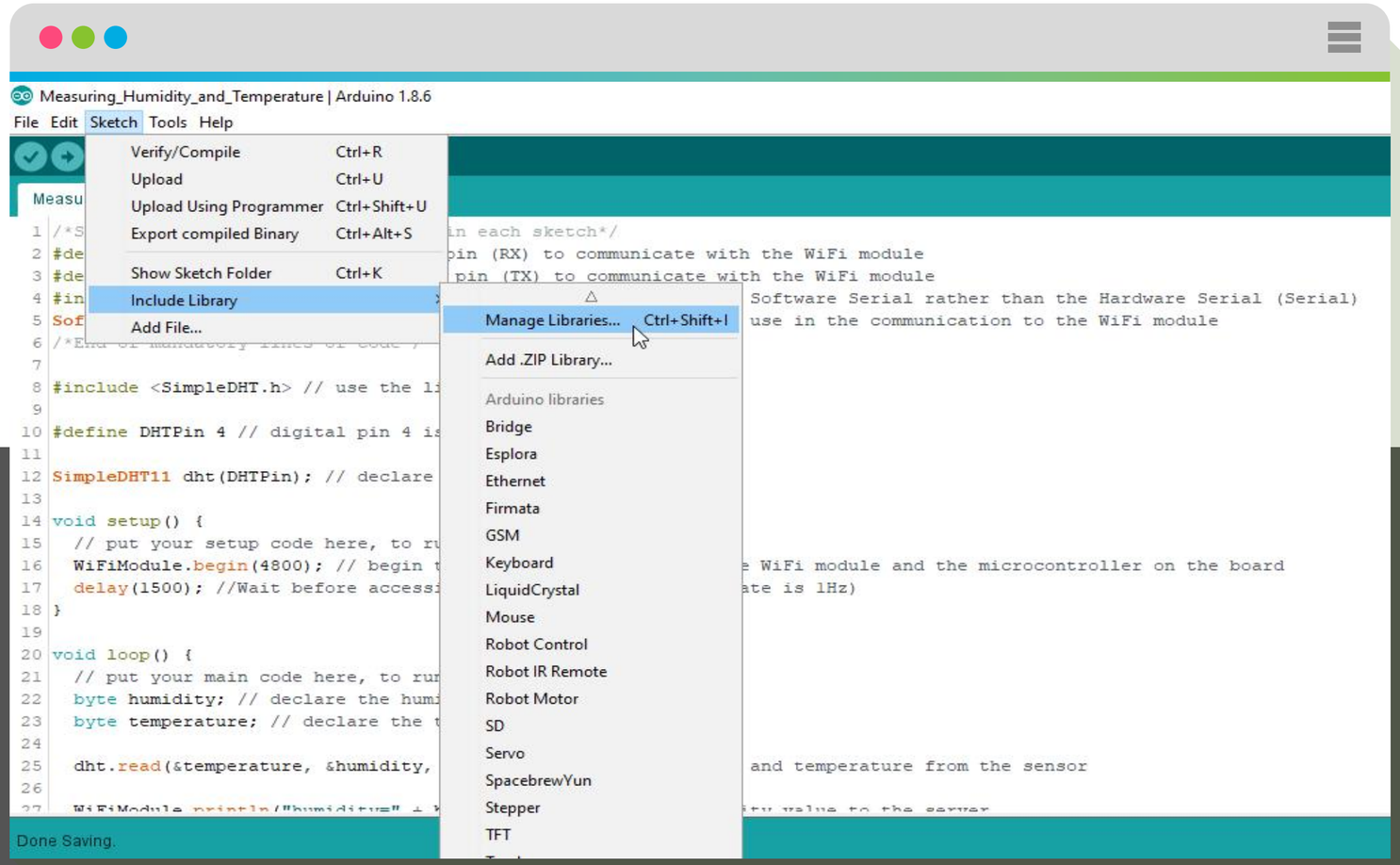


AFF IoT Board > Circuits > Measuring_Humidity_and_Temperature

Measuring_Humidity_and_Temperature

```
/*Start of mandatory lines of codes in each sketch*/
#define RX A0 // define the Receive pin (RX) to communicate with the WiFi module
#define TX A1 // define the Transmit pin (TX) to communicate with the WiFi module
#include <NeoSWSerial.h> // including the library to use the Software Serial rather than the Hardware Serial (Serial)
NeoSWSerial WiFiModule(RX, TX); //initialize the variable to use in communication with the WiFi module
/*End of mandatory lines of code*/

#include <SimpleDHT.h> // use the library functions in the code

#define DHTPin 4 // digital pin 4 is connected to the sensor

SimpleDHT11 dht(DHTPin); // declare the DHT11 instance

void setup() {
  // put your setup code here, to run once:
  WiFiModule.begin(19200); // begin the communication between the WiFi module and the microcontroller on the board
  delay(1500); //Wait before accessing Sensor (DHT11 sampling rate is 1Hz)
}

void loop() {
  // put your main code here, to run repeatedly:
  byte humidity; // declare the humidity variable
  byte temperature; // declare the temperature variable

  dht.read(&temperature, &humidity, NULL); // read both humudity and temperature from the sensor

  String data = "";
  data += "humidity=" + String(humidity); //example: humidity=70
  data += ","; // all parameters should be comma delimited
  data += "temperature=" + String(temperature); //example: temperature=27
  //data = "humidity=70,temperature=27"

  WiFiModule.println(data);// send the Humidity and Temperature values to the server

  delay(3000);//Wait 3 seconds before accessing sensor again.
}
```

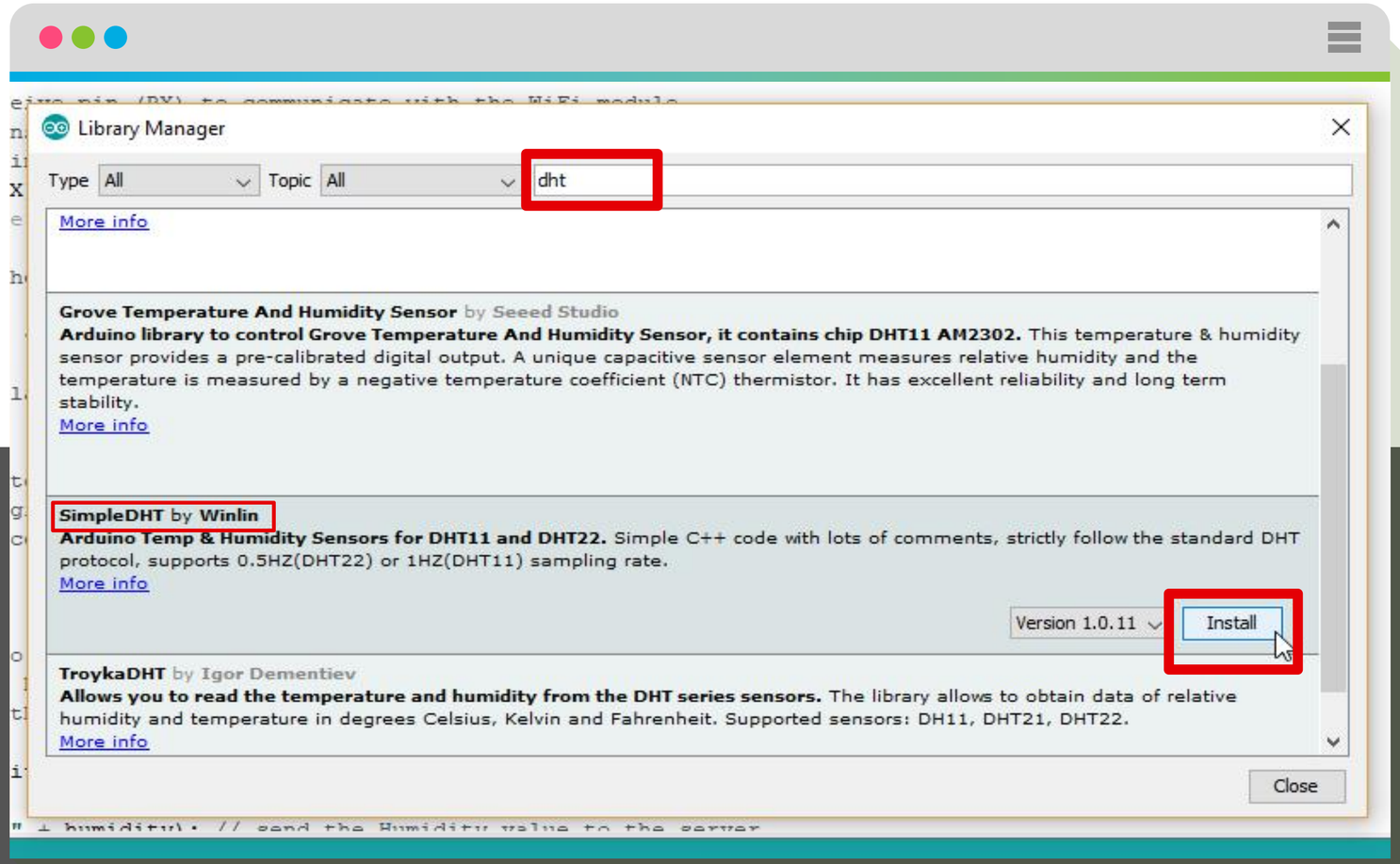
Ouvrir votre croquis: Measuring_Humidity_and_Temperature



Avant de Compiler le code

Télécharger la bibliothèque DHT

Sketch > Include Library > Manage Libraries



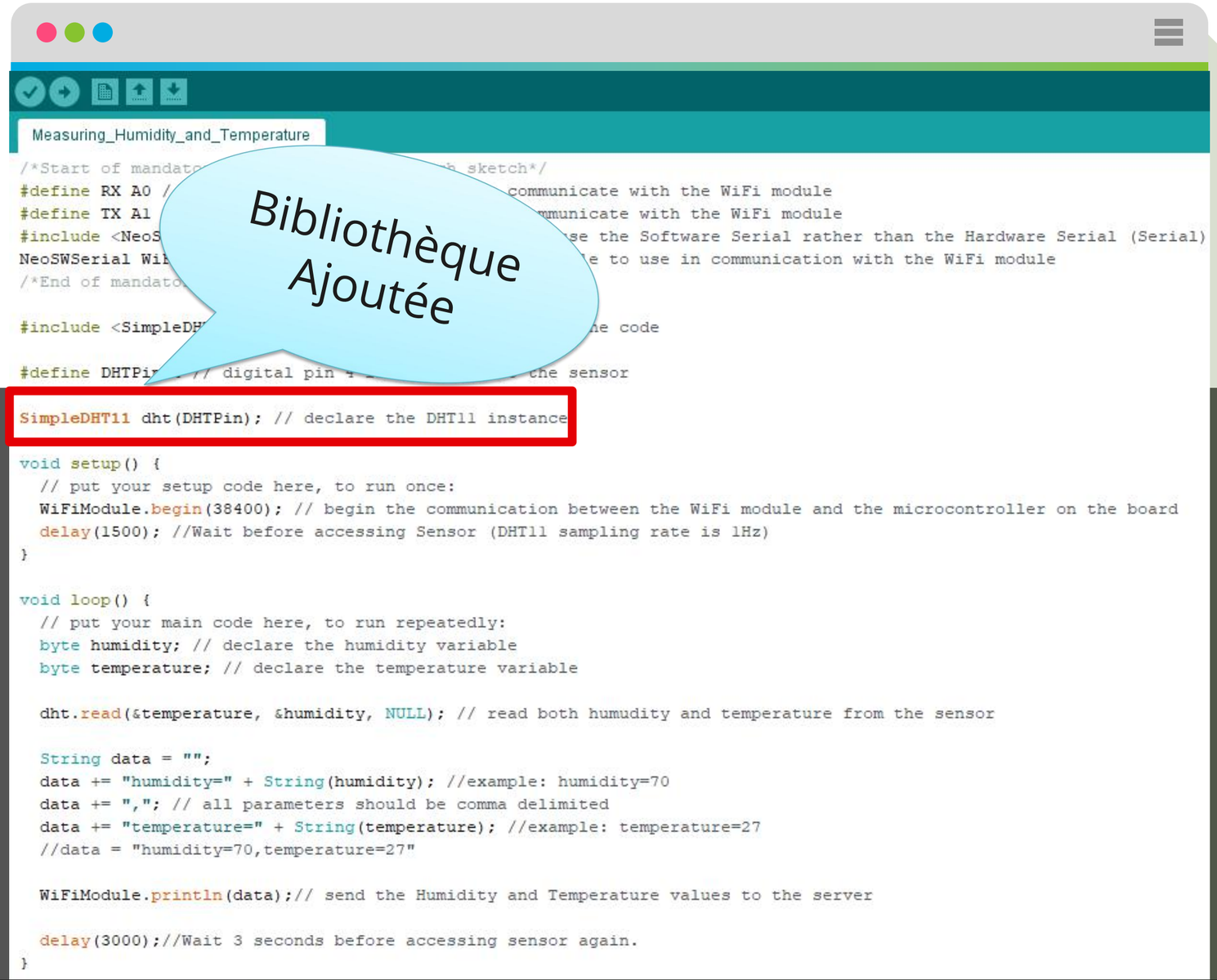
Avant de Compiler le code

Télécharger la bibliothèque DHT

Recherchez “dht”, sélectionnez la dernière version de “SimpleDHT library” et cliquez sur “Install”.

Avant de Compiler le code

Télécharger la bibliothèque DHT



Après avoir cliqué sur le bouton “Close”, nous pouvons voir que l'EDI Arduino reconnaît le type de SimpleDHT et change sa couleur en orange.

Code à noter..

```
dht.read(&temperature, &humidity, NULL);
```

Cette ligne de code appelle la fonction de lecture pour obtenir l'humidité et la température du capteur DHT11, et met les valeurs dans les variables «température» et «humidité».

Ce que Vous **devez voir**

Vous devriez voir les valeurs réelles d'humidité et de température. Si cela ne fonctionne pas, assurez-vous d'avoir correctement assemblé le circuit, vérifié et téléchargé le code sur votre carte ou consultez les conseils de dépannage ci-dessous.

Dépannage

Rien ne se passe
Ce programme n'a aucune indication externe qu'il fonctionne. Pour voir les résultats, vous devez ouvrir l'application AFF IoT.

Les valeurs ne changent pas
Essayez de frotter le capteur avec vos doigts pour le réchauffer ou appuyez sur un sac de glace pour le refroidir.

Application dans la réalité

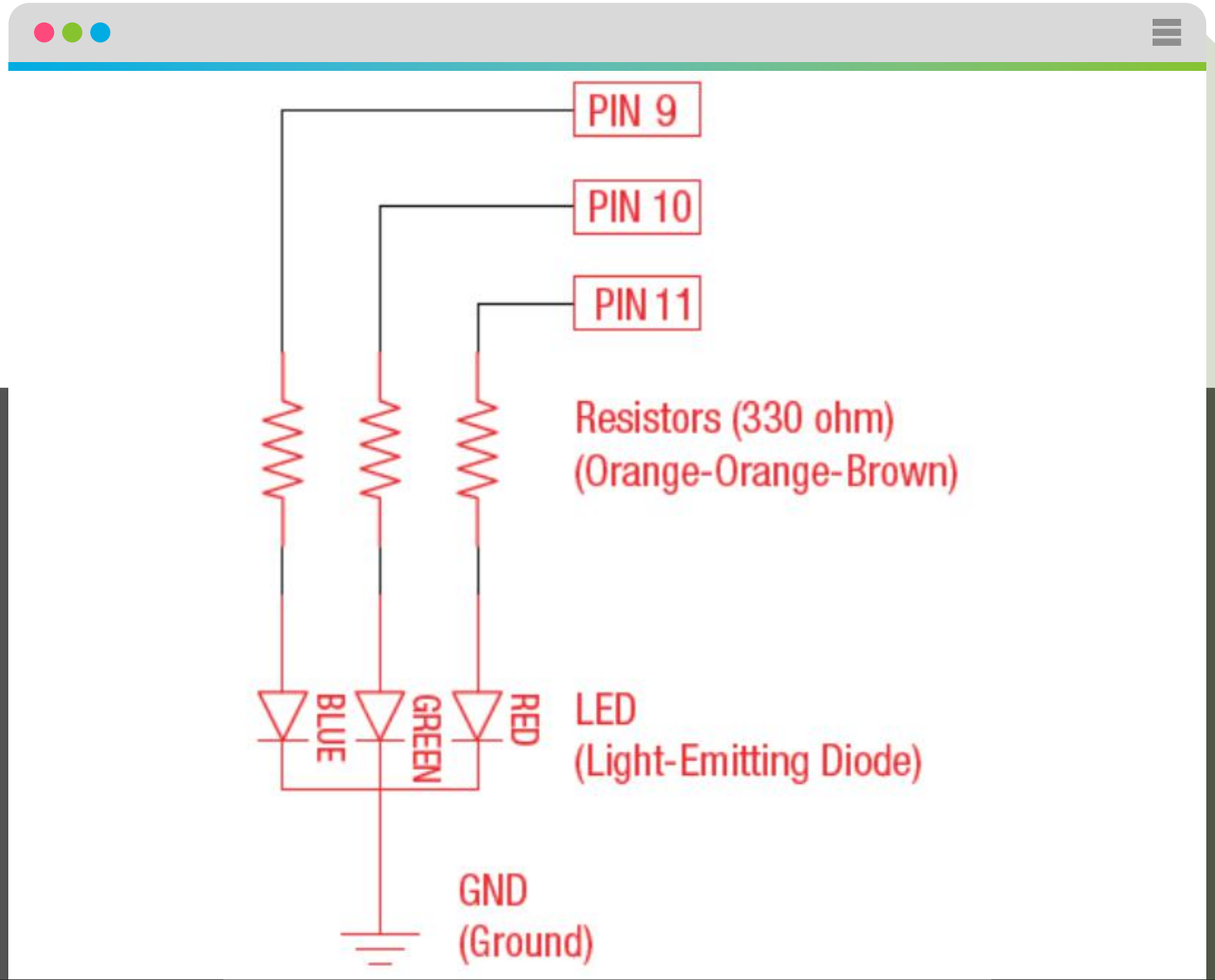
Mesurer l'humidité el la temperature dans les locaux de stockage pour éviter d'endommager les objets stockés

9 Réglage de la couleur des LED RVB (via «Internet»)

RVB LED x1

Résistance 330Ω x3

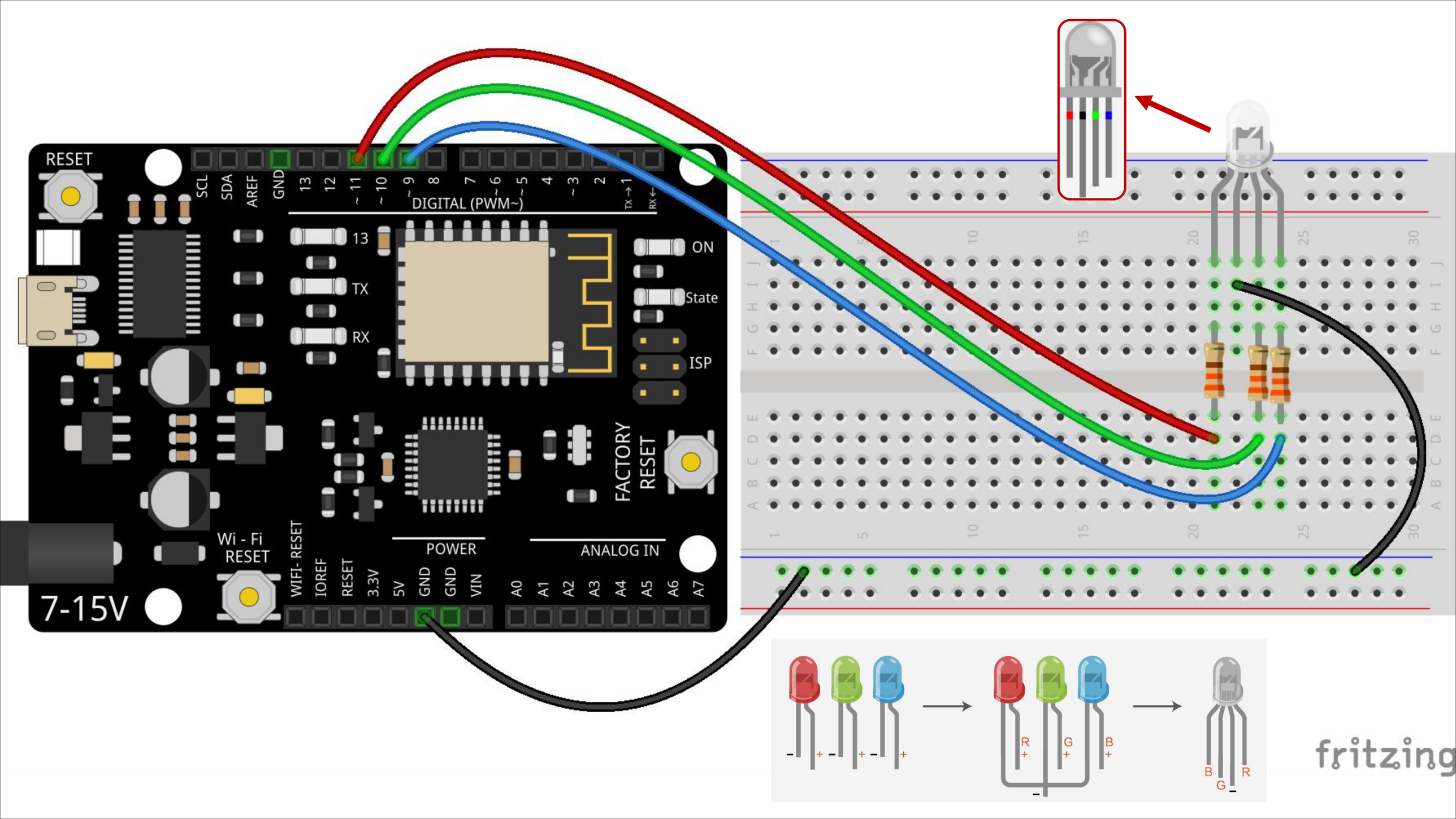
Fils de connexion x5

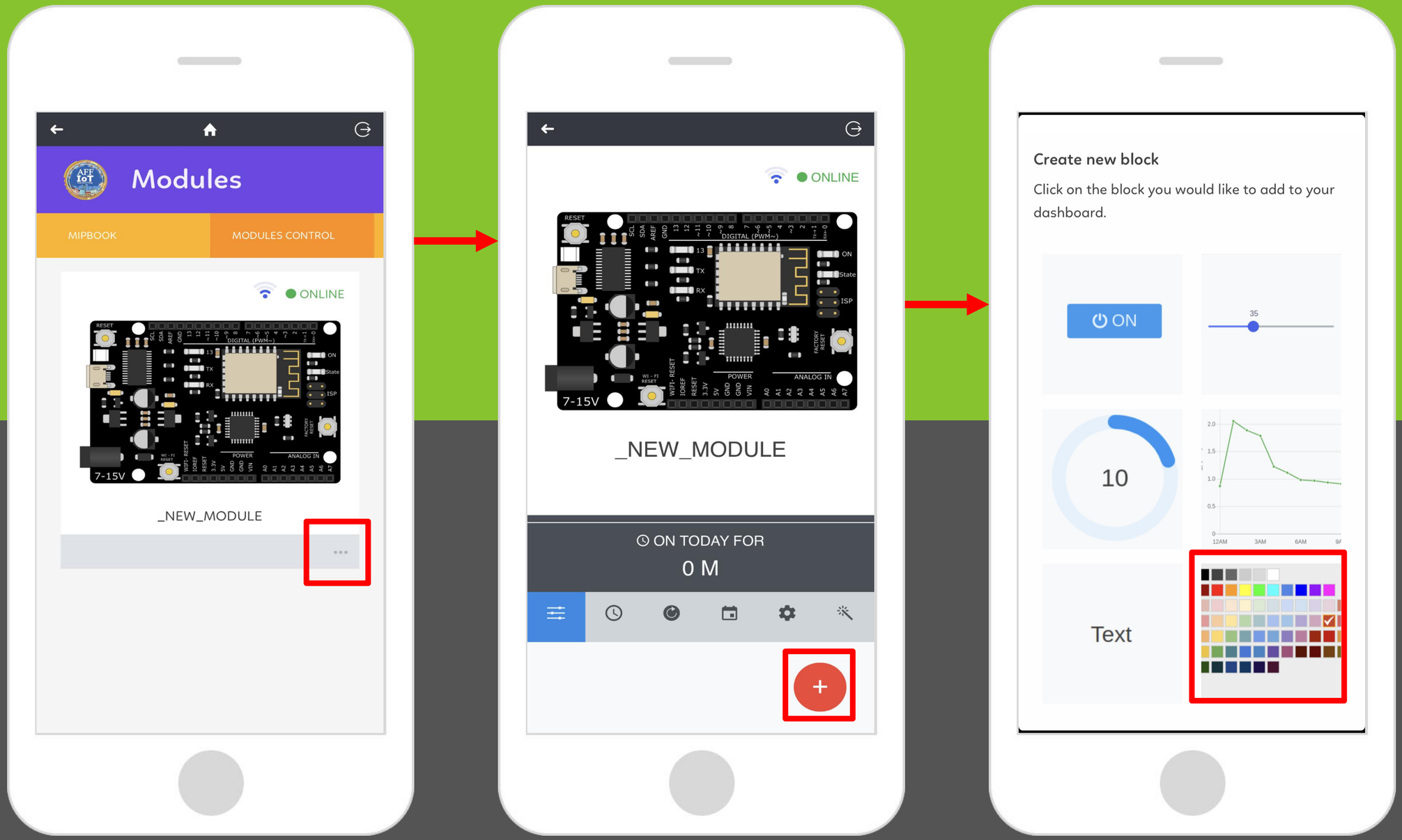


RVB LED

Les LED RVB ou rouge-vert-bleu ont trois diodes électroluminescentes différentes qui peuvent être combinées pour créer toutes sortes de couleurs. Dans ce circuit, vous apprendrez à utiliser une LED RVB pour créer des combinaisons de couleurs uniques.

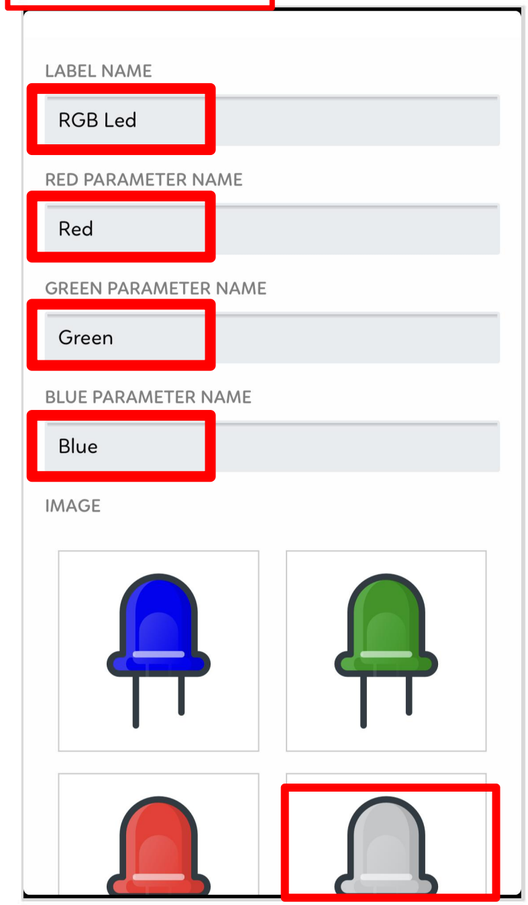
Selon la luminosité de chaque diode, presque toutes les couleurs sont possibles!

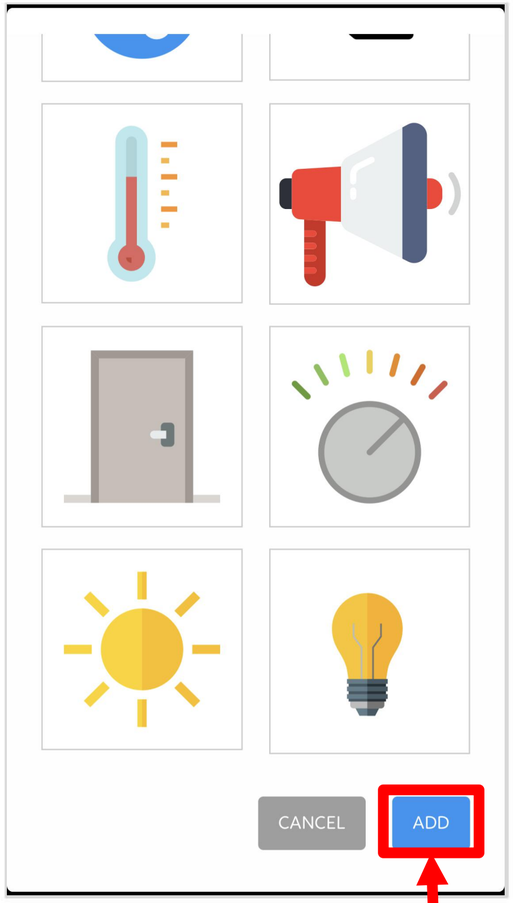

RESET
SCL
SDA
AREF
GND
13
12
~ 11
~ 10
~ 9
8
7
~ 6
~ 5
4
~ 3
2
TX → 1
RX ←
DIGITAL (PWM~)
13
TX
RX
ON
State
ISP
FACTORY RESET
Wi - Fi RESET
WIFI- RESET
IOREF
RESET
3.3V
5V
GND
GND
VIN
POWER
ANALOG IN
A0
A1
A2
A3
A4
A5
A6
A7
7-15V
R
G
B
B
G
R
fritzing


Modules
MIPBOOK
MODULES CONTROL
ONLINE
_NEW_MODULE
ONLINE
_NEW_MODULE
ON TODAY FOR
0 M
Create new block
Click on the block you would like to add to your dashboard.
ON
35
10
Text

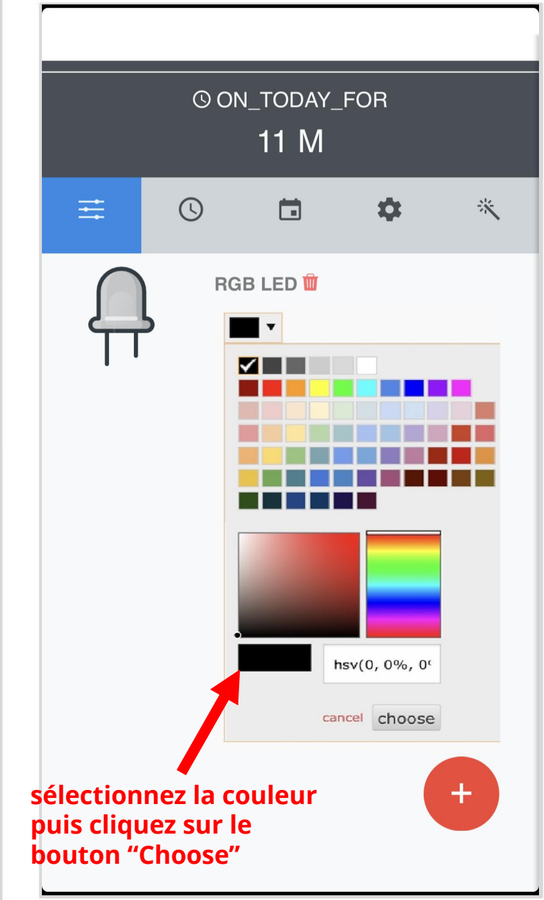
LABEL NAME: est le nom qui apparaît dans l'application.
PARAMETER NAME: est le nom utilisé dans le code.

Remplissez les paramètres suivants en blanc





Faites défiler la liste et cliquez sur ADD



sélectionnez la couleur puis cliquez sur le bouton “Choose”

AFF IoT Board folder > Circuits > Adjusting_RGB_LED_Color

Adjusting_RGB_LED_Color

```
/*Start of mandatory lines of codes in each sketch*/
#define RX A0 // define the Receive pin (RX) to communicate with the WiFi module
#define TX A1 // define the Transmit pin (TX) to communicate with the WiFi module
#include <NeoSWSerial.h> // including the library to use the Software Serial rather than the Hardware Serial (Serial)
NeoSWSerial WiFiModule(RX, TX); //initialize the variable to use in communication with the WiFi module
/*End of mandatory lines of code*/

#define RedLedPin    11
#define GreenLedPin  10
#define BlueLedPin   9

String RedStringValue = "";
String GreenStringValue = "";
String BlueStringValue = "";

int RedValue = 0;
int GreenValue = 0;
int BlueValue = 0;

void setup() {
  // put your setup code here, to run once:
  WiFiModule.begin(19200); // begin the communication between the WiFi module and the microcontroller on the board
  pinMode(RedLedPin, OUTPUT); // configure the pin connected to the Red Led to be an output
  pinMode(GreenLedPin, OUTPUT); // configure the pin connected to the Green Led to be an output
  pinMode(BlueLedPin, OUTPUT); // configure the pin connected to the Blue Led to be an output
  Serial.begin(9600);
}
```

1

Ouvrir votre croquis:
Adjusting_RGB_LED_Color

AFF IoT Board folder > Circuits > Adjusting_RGB_LED_Color

```
void loop() {
  // put your main code here, to run repeatedly:
  if (WiFiModule.available() > 0) // if the WiFi module receive data from the server
  {
    String Command = WiFiModule.readStringUntil('\n'); // read the command sent from the WiFi module to the microcontroller
    Serial.println(Command); // print the command on the Serial monitor for debugging
    if (Command.indexOf("Red=") >= 0) // if the received command contains "Red=" so we should change the Red intensity
    {
      String RedString = Command.substring(0, Command.indexOf(',')+1); // extract the command for "Red" parameter
      RedStringValue = Command.substring(Command.indexOf('=') + 1); // extract the Red intensity string value
      RedValue = RedStringValue.toInt(); // transform the string to an integer value
      analogWrite(RedLedPin, RedValue); // generate the PWM duty cycle according to the needed intensity
      Command.replace(RedString, ""); // remove the executed command
      Serial.println(Command); // print the command on the Serial monitor for debugging
    }
    if (Command.indexOf("Green=") >= 0) // if the received command contains "Green=" so we should change the Green intensity
    {
      String GreenString = Command.substring(0, Command.indexOf(',')+1); // extract the command for "Green" parameter
      GreenStringValue = Command.substring(Command.indexOf('=') + 1); // extract the Green intensity string value
      GreenValue = GreenStringValue.toInt(); // transform the string to an integer value
      analogWrite(GreenLedPin, GreenValue); // generate the PWM duty cycle according to the needed intensity
      Command.replace(GreenString, ""); // remove the executed command
      Serial.println(Command); // print the command on the Serial monitor for debugging
    }
    if (Command.indexOf("Blue=") >= 0) // if the received command contains "Blue=" so we should change the Blue intensity
    {
      String BlueString = Command.substring(0, Command.indexOf(',')+1); // extract the command for "Blue" parameter
      BlueStringValue = Command.substring(Command.indexOf('=') + 1); // extract the Blue intensity string value
      BlueValue = BlueStringValue.toInt(); // transform the string to an integer value
      analogWrite(BlueLedPin, BlueValue); // generate the PWM duty cycle according to the needed intensity
      Command.replace(BlueString, ""); // remove the executed command
      Serial.println(Command); // print the command on the Serial monitor for debugging
    }
  }
}
//(for more info about indexOf function see https://www.arduino.cc/reference/en/language/variables/data-types/string/functions/indexof/)
//(for more info about substring function see https://www.arduino.cc/en/Tutorial/StringSubstring)
```

2

Ouvrir votre croquis:
Adjusting_RGB_LED_Color

Code à noter..

`RedValue = RedStringValue.toInt()` : la fonction `toInt()` convertit une chaîne caractère valide en un entier. La chaîne en entrée doit commencer par un nombre entier. Si la chaîne contient des nombres non entiers, la fonction cessera d'effectuer la conversion.

`Command.substring(index)` : Avec un seul paramètre, il recherche une sous-chaîne donnée de l'indice affecté à la fin de la chaîne (`Command` dans cet exemple).

Ce que vous **devez voir**

Vous devriez voir votre LED s'allumer, mais cette fois dans de nouvelles couleurs folles! Si ce n'est pas le cas, assurez-vous d'avoir correctement assemblé le circuit, vérifié et téléchargé le code sur votre carte ou consultez les conseils de dépannage ci-dessous.

Dépannage

Voyant rouge
La diode rouge dans la LED RVB peut être un peu plus lumineuse que les deux autres. Pour rendre vos couleurs plus équilibrées, utilisez une résistance ohmique supérieure pour le rouge. Ou ajuster dans le code.

```
analogWrite(RedLedPin, RedValue);
```

à `analogWrite(RedLedPin, RedValue/3);`

La LED reste sombre ou affiche une couleur incorrecte
Avec les quatre broches de la LED si proches les unes des autres, il est parfois facile de les égarer. Vérifiez de nouveau chaque broche où est inséré.

Application dans la réalité

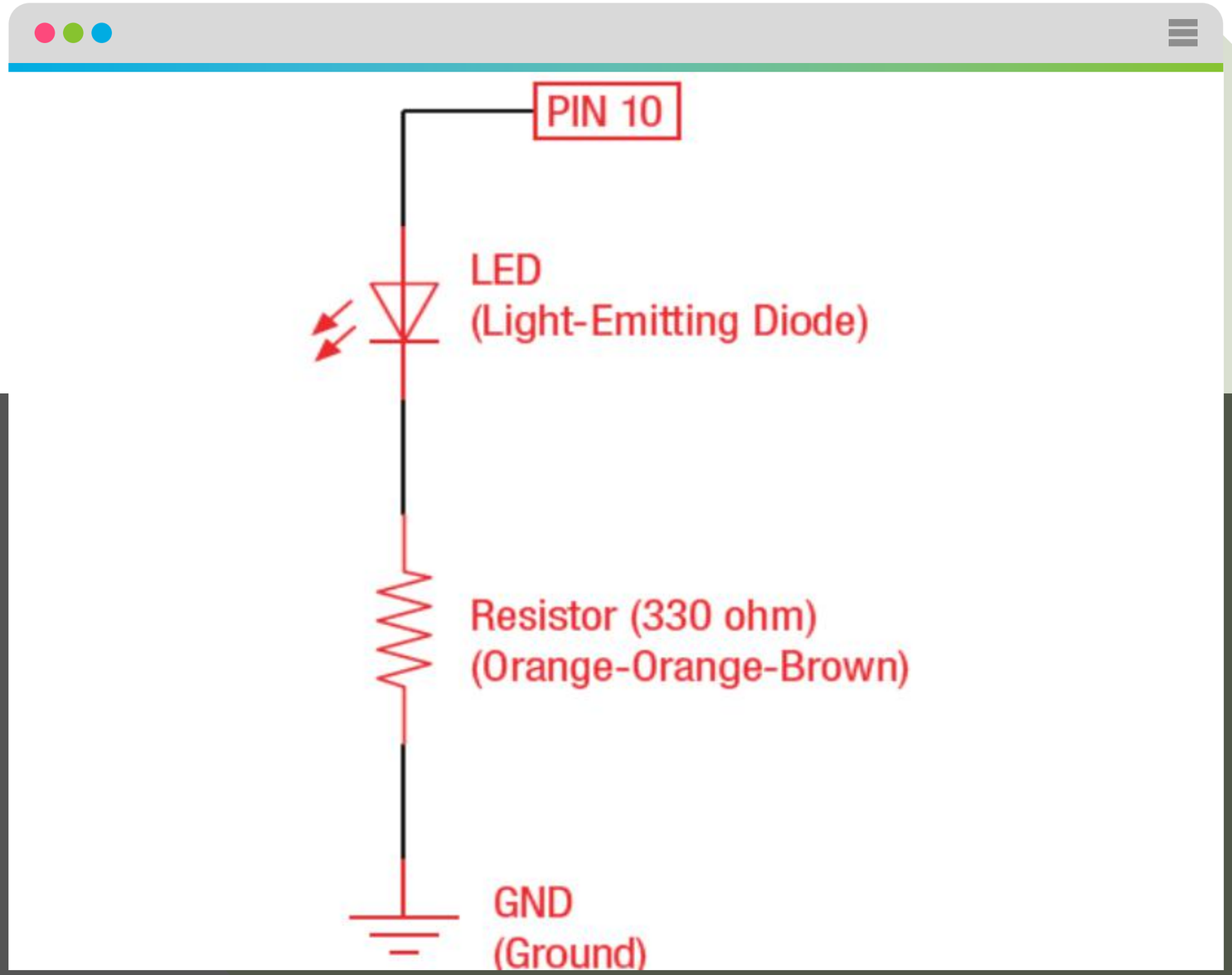
De nombreux appareils électroniques, tels que les consoles de jeux vidéo, utilisent des LED RVB pour afficher différentes couleurs au sein d'une même source de lumière.

10 Planification des Contrôles

LED blanche x1

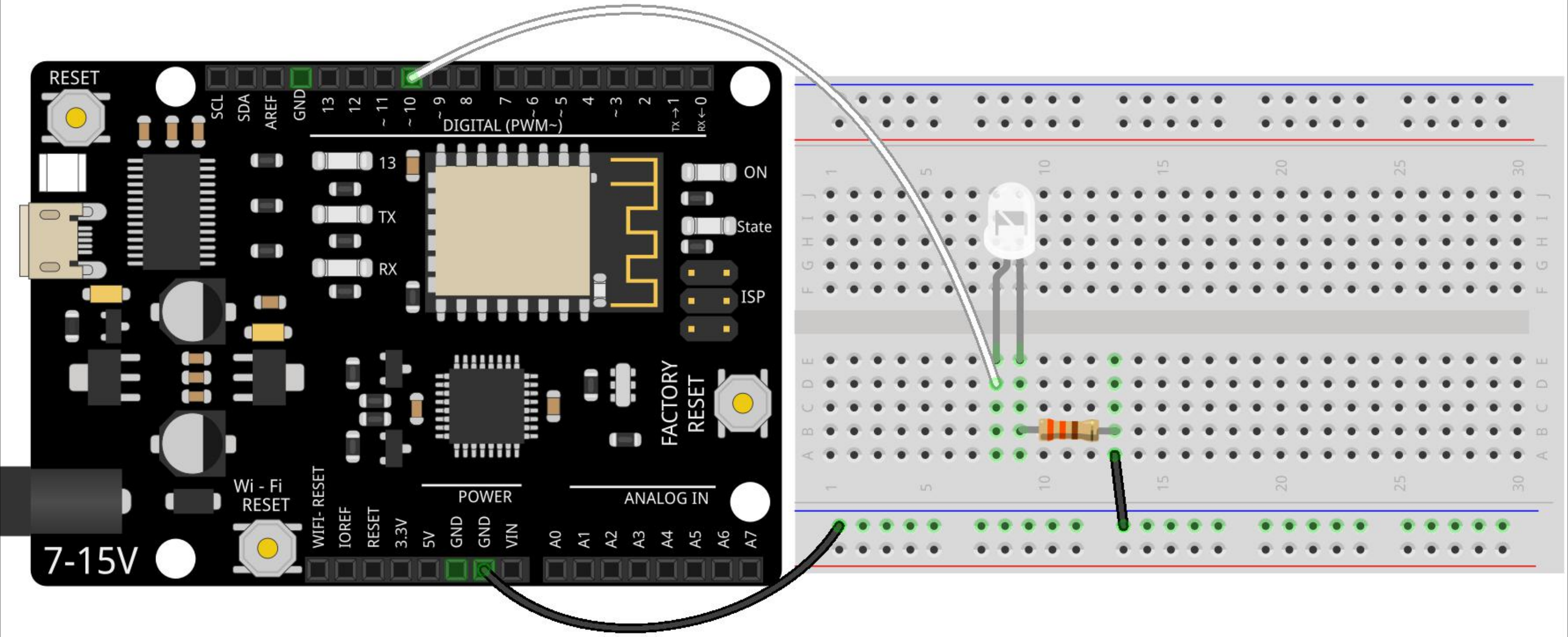
Résistance 330Ω x1

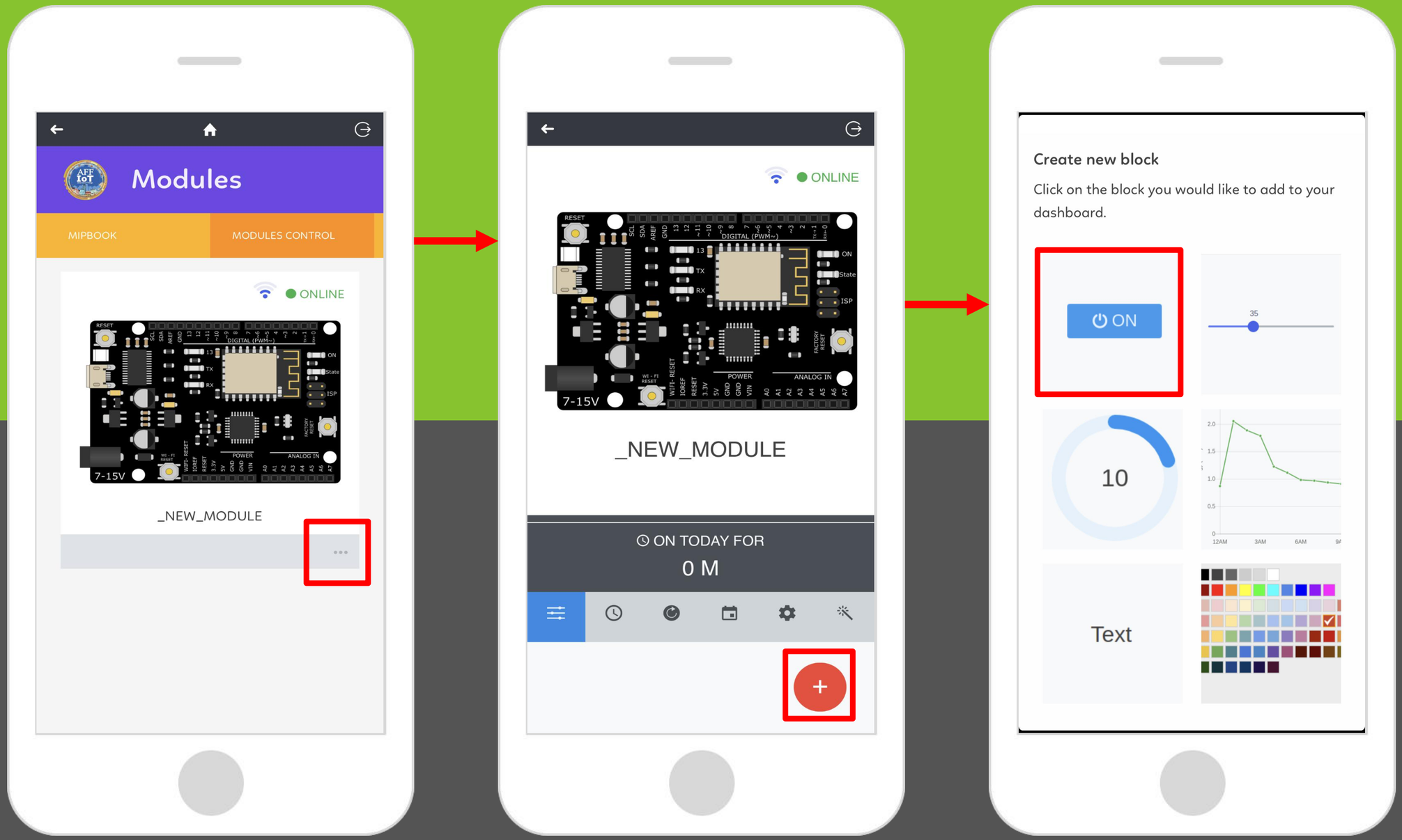
Fils de connexion x3



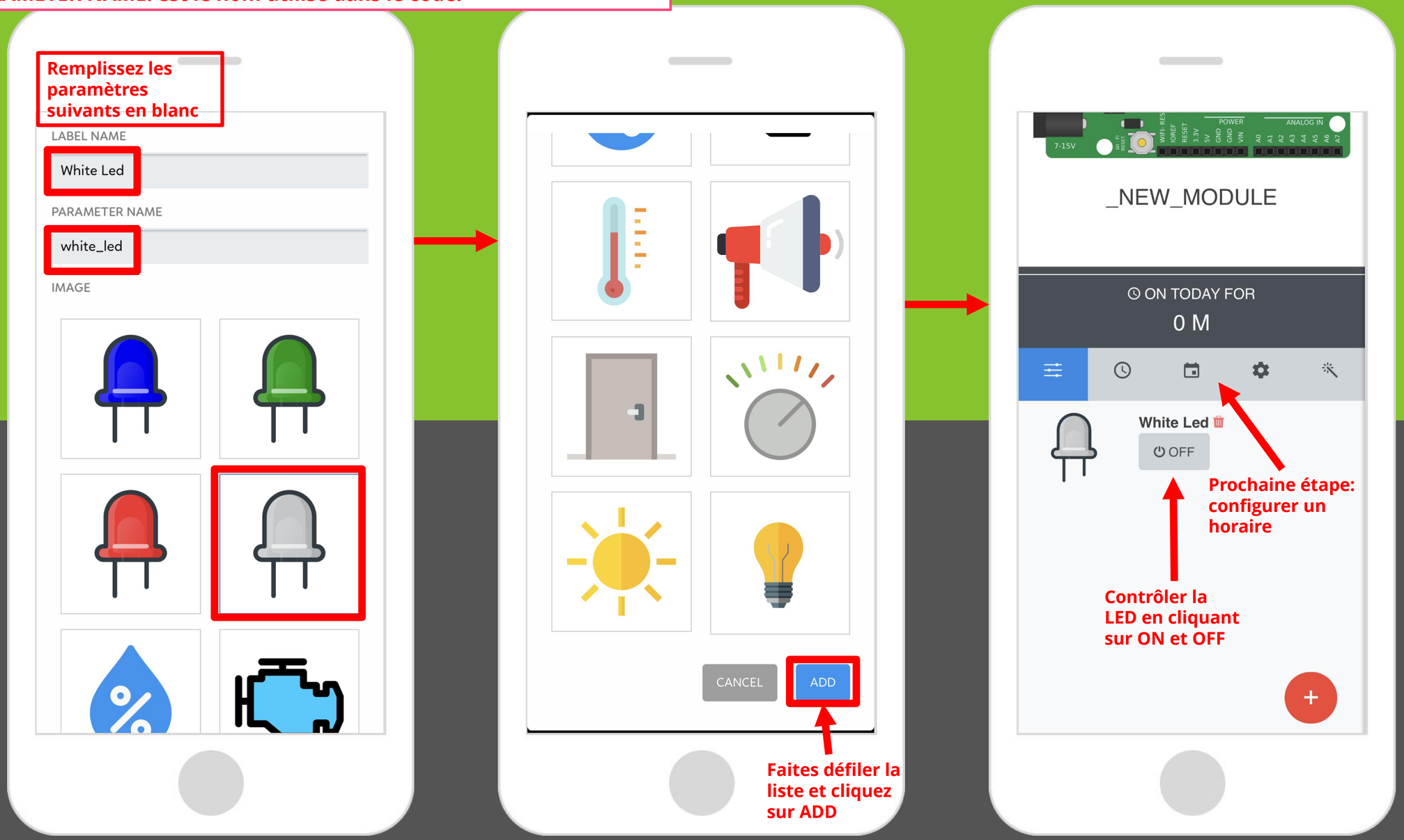
Allumer/éteindre une LED

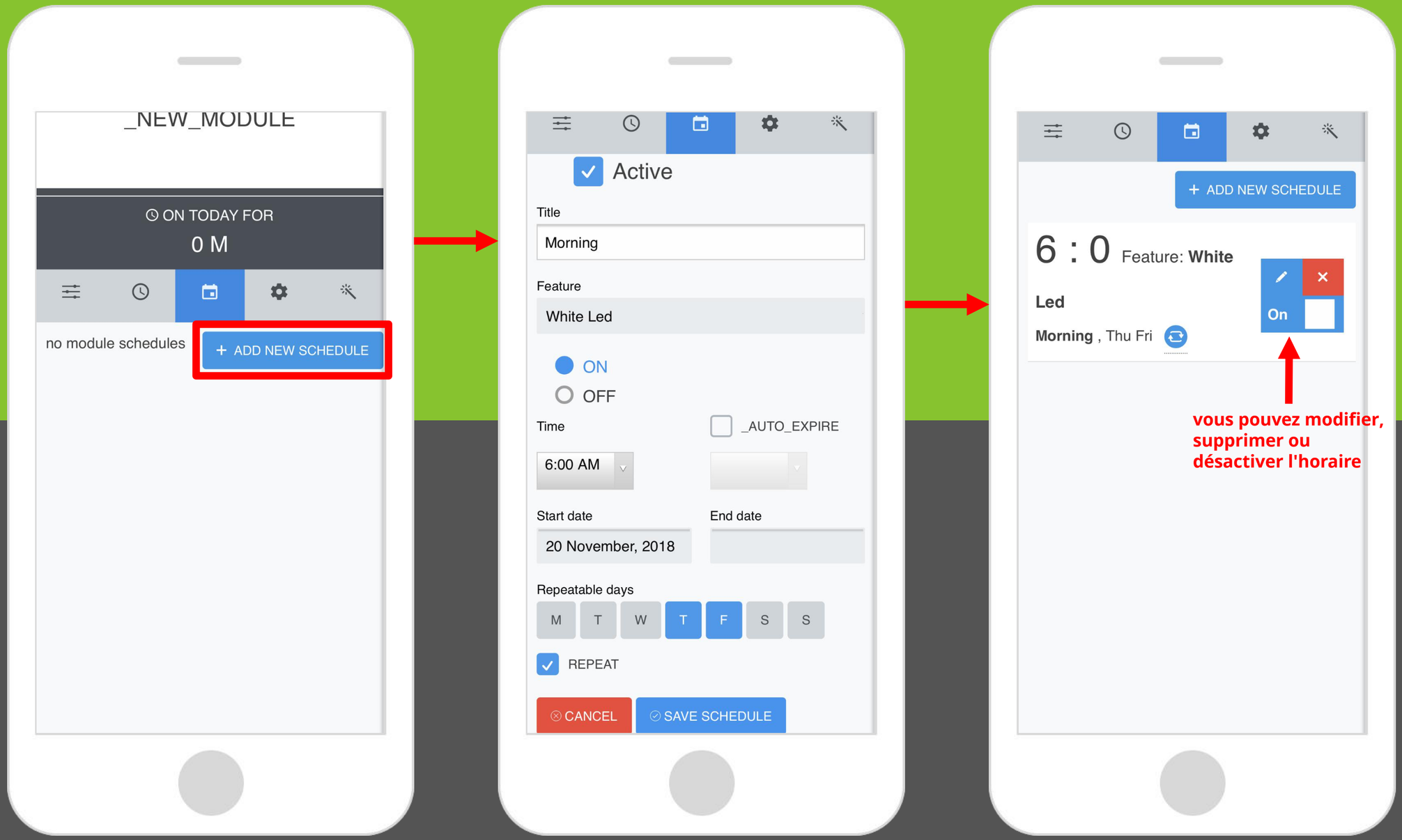
Les LEDs (diodes électroluminescentes) sont une source lumineuse petite mais puissante, utilisée dans de nombreuses applications différentes. Nous allons d'abord travailler sur le contrôle d'une LED. C'est une fonction simple et fondamentale, mais c'est la base pour les autres fonctions complexes.

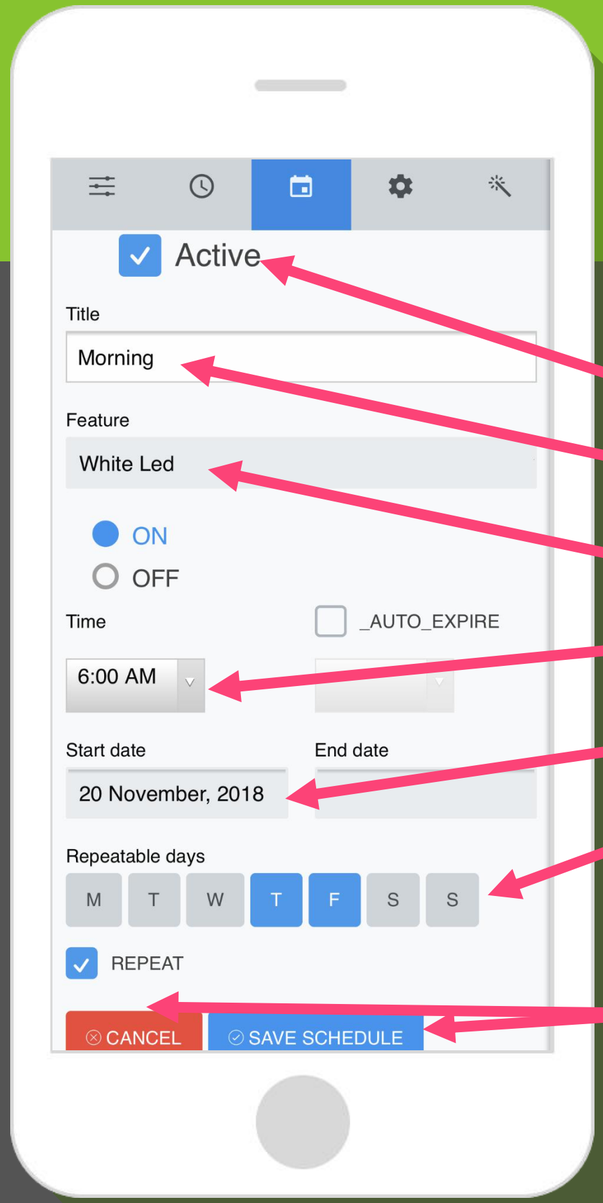

RESET
SCL
SDA
AREF
GND
13
12
~ 11
~ 10
~ 9
8
7
~ 6
~ 5
4
~ 3
2
TX → 1
RX ← 0
DIGITAL (PWM~)
13
TX
RX
ON
State
ISP
FACTORY
RESET
Wi - Fi
RESET
WIFI- RESET
IOREF
RESET
3.3V
5V
GND
GND
VIN
POWER
ANALOG IN
A0
A1
A2
A3
A4
A5
A6
A7
7-15V


Modules
MIPBOOK
MODULES CONTROL
ONLINE
_NEW_MODULE
ONLINE
_NEW_MODULE
ON TODAY FOR
0 M
Create new block
Click on the block you would like to add to your dashboard.
ON
35
10
Text

LABEL NAME: est le nom qui apparaît dans l'application.
PARAMETER NAME: est le nom utilisé dans le code.

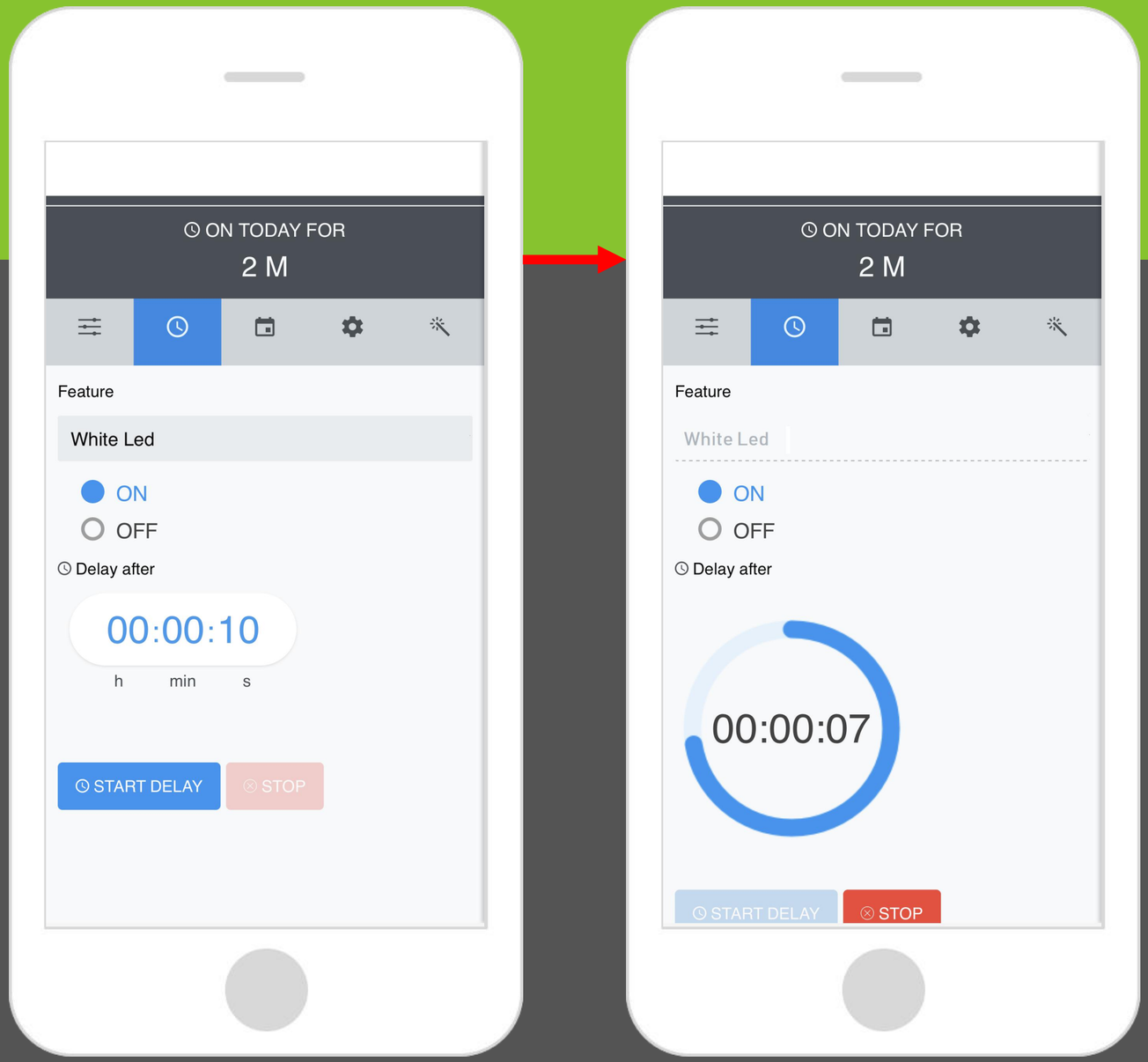



_NEW_MODULE
ON TODAY FOR
0 M
no module schedules
+ ADD NEW SCHEDULE
Active
Title
Morning
Feature
White Led
ON
OFF
Time
_AUTO_EXPIRE
6:00 AM
Start date
20 November, 2018
End date
Repeatable days
M T W T F S S
REPEAT
CANCEL
SAVE SCHEDULE
+ ADD NEW SCHEDULE
6 : 0 Feature: White
Led
Morning , Thu Fri
On
vous pouvez modifier, supprimer ou désactiver l'horaire



Explication

- Active: sélectionnez l'état d'activation du programme (actif ou non).
- Title: est le titre du horaire créé.
- Feature: est la commande à programmer (Activer ou Désactiver).
- Time: temps à laquelle l'action sera exécutée.
- Start date: est la date à laquelle le programme sera activé.
-Repeatable days: si la case REPEAT est cochée, vous pouvez choisir quels sont les jours où l'horaire est actif.
- Enregistrer le calendrier ou l’annuler.



Minuteur

L'option de minuterie consiste à choisir une commande et à définir un délai. Après ce délai, le contrôle sera exécuté.

Vous pouvez également arrêter le chronomètre en cliquant sur le bouton STOP, l'action différée sera annulée.

AFF IoT Board folder > Circuits > Examples > Scheduling_Controls

Scheduling_Controls

```
/*Start of mandatory lines of codes in each sketch*/
#define RX A0 // define the Receive pin (RX) to communicate with the WiFi module
#define TX A1 // define the Transmit pin (TX) to communicate with the WiFi module
#include <NeoSWSerial.h> // including the library to use the Software Serial rather than the Hardware Serial (Serial)
NeoSWSerial WiFiModule(RX, TX); //initialize the variable to use in communication with the WiFi module
/*End of mandatory lines of code*/

#define WhiteLedPin   10

void setup() {
  // put your setup code here, to run once:
  WiFiModule.begin(19200); // begin the communication between the WiFi module and the microcontroller on the board
  pinMode(WhiteLedPin, OUTPUT); // configure the pin connected to the White Led to be an output
}

void loop() {
  // put your main code here, to run repeatedly:
  if (WiFiModule.available() > 0) // if the WiFi module receive data from the server
  {
    String Command = WiFiModule.readStringUntil('\n'); // read the command sent from the WiFi module to the microcontroller

    if (Command.indexOf("white_led=1") >= 0) // if the received command contains "white_led=1" turn on the LED
    {
      digitalWrite(WhiteLedPin, HIGH); // turn on the LED
    }
    if (Command.indexOf("white_led=0") >= 0) // if the received command contains "white_led=0" turn it off
    {
      digitalWrite(WhiteLedPin, LOW); // turn off the LED
    }
  }
}
//(for more info about indexOf function see https://www.arduino.cc/reference/en/language/variables/data-types/string/functions/indexof/)
```

Ouvrir votre croquis:
Scheduling_Controls

Ce croquis fonctionnera de la même manière que dans l'exemple de Contrôler une LED (projet 5) mais avec un changement de nom de LED.

Ce que vous **devez voir**

Vous devriez voir la LED s'allumer après 10 secondes. Si vous sélectionnez l'option OFF, vous devriez voir la LED s'éteindre une fois le délai écoulé.

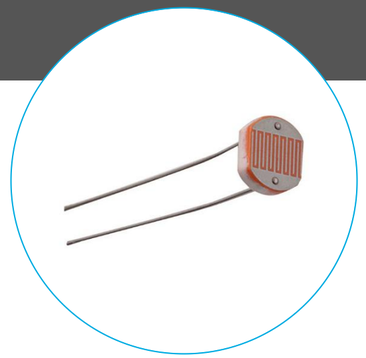
Dépannage

La LED ne s'allume pas?
Les LEDs ne fonctionneront que dans un sens. Essayez de la sortir et en la tournant de 180 degrés (ne vous inquiétez pas, son installation dans le sens opposé ne l’endommage pas).

Real World Application

La plupart des appareils IoT sont construits avec des minuteries intégrées

11 Simulation d'un système d'éclairage automatisé



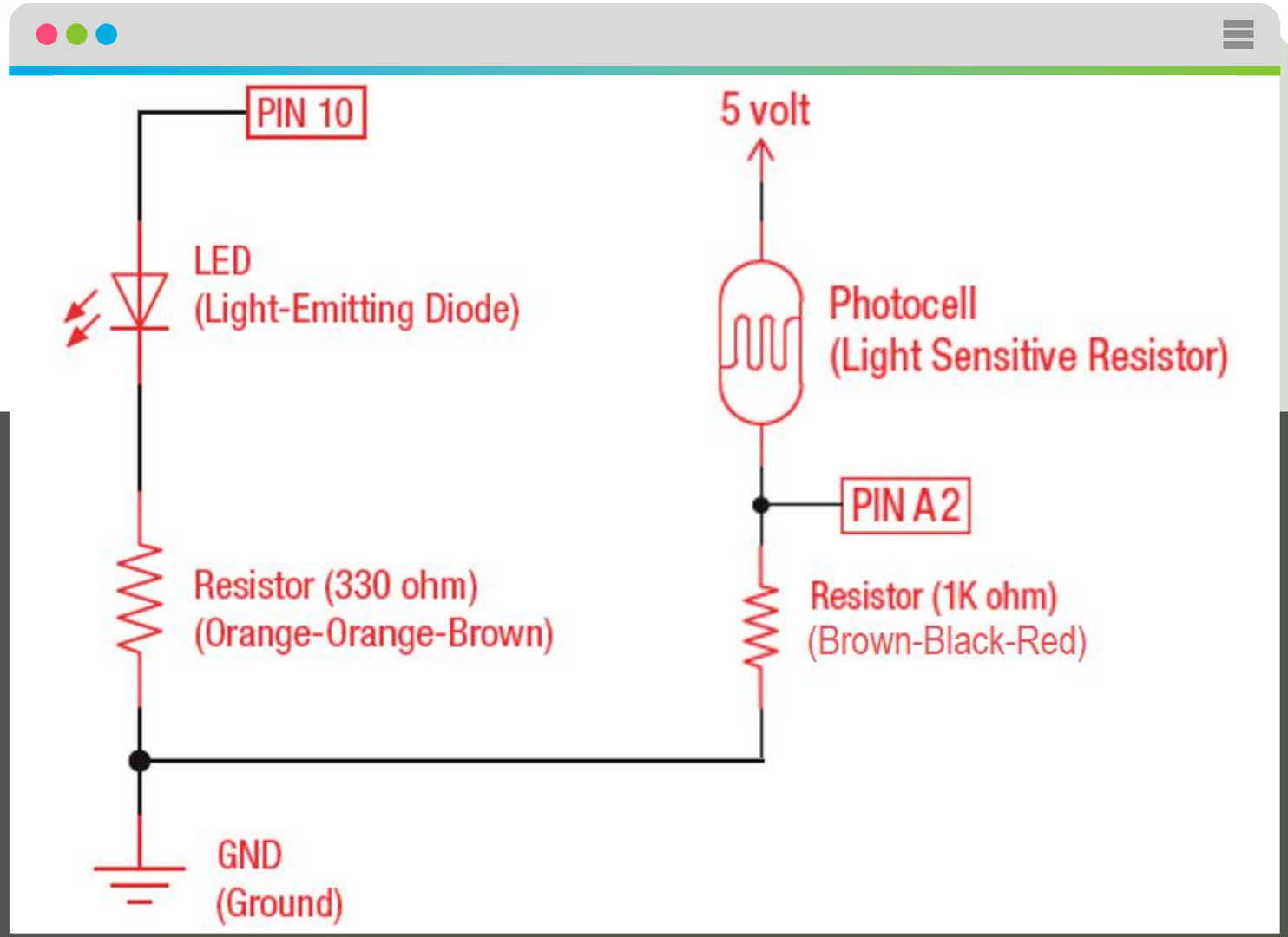
Photocellule x1

LED jaune x1

Résistance 1KΩ x1

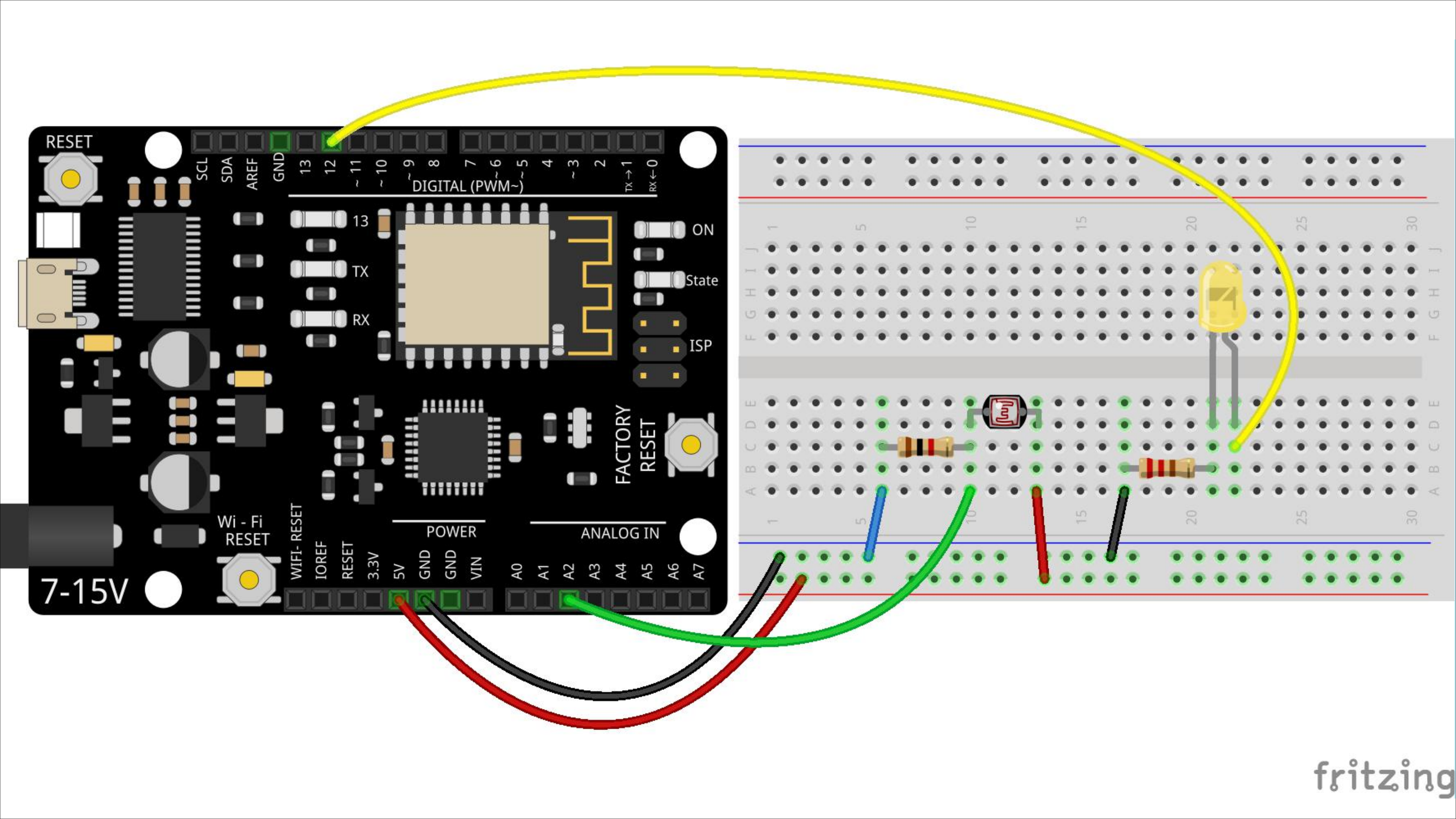
Résistance 330Ω x1

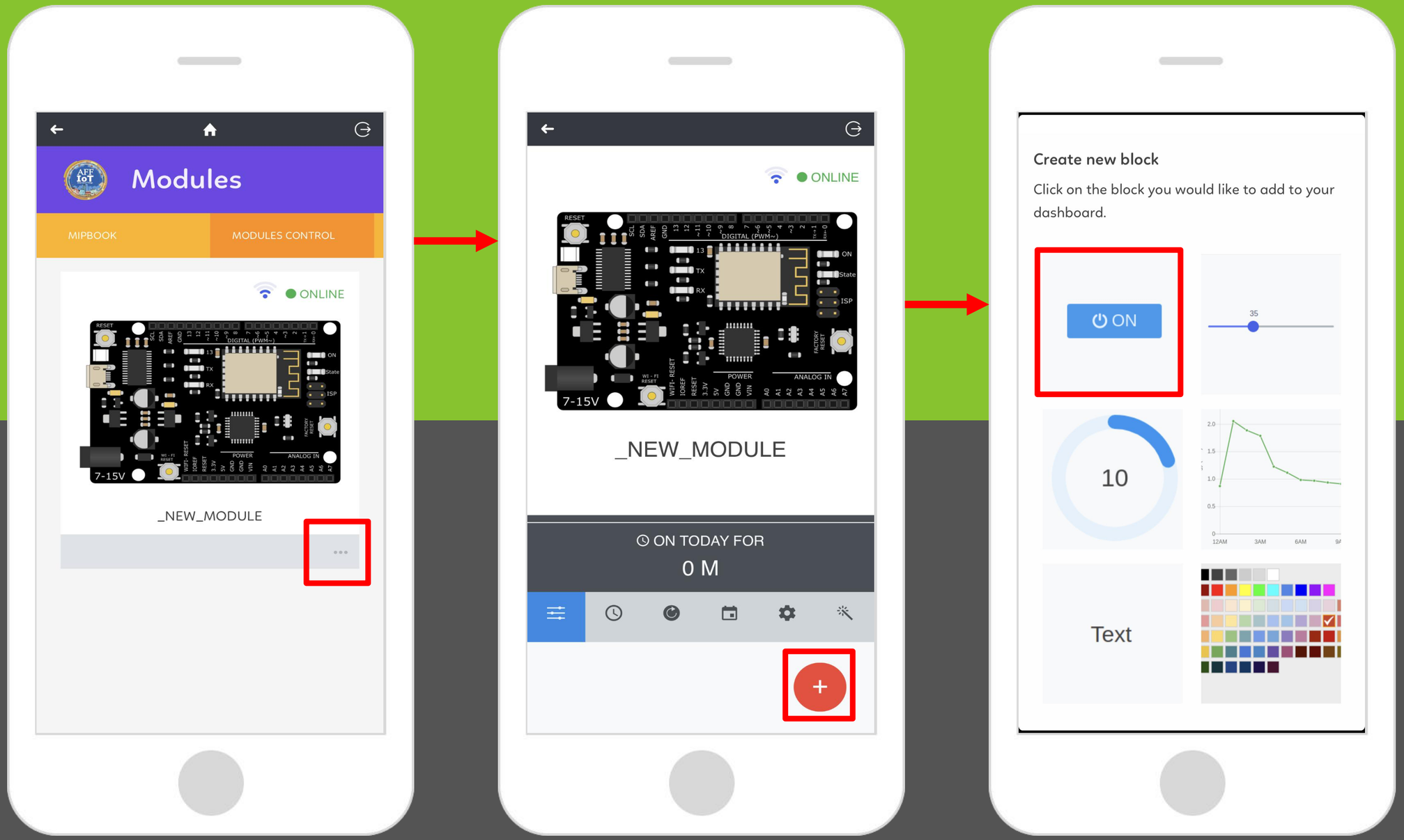
Fils de connexion x5



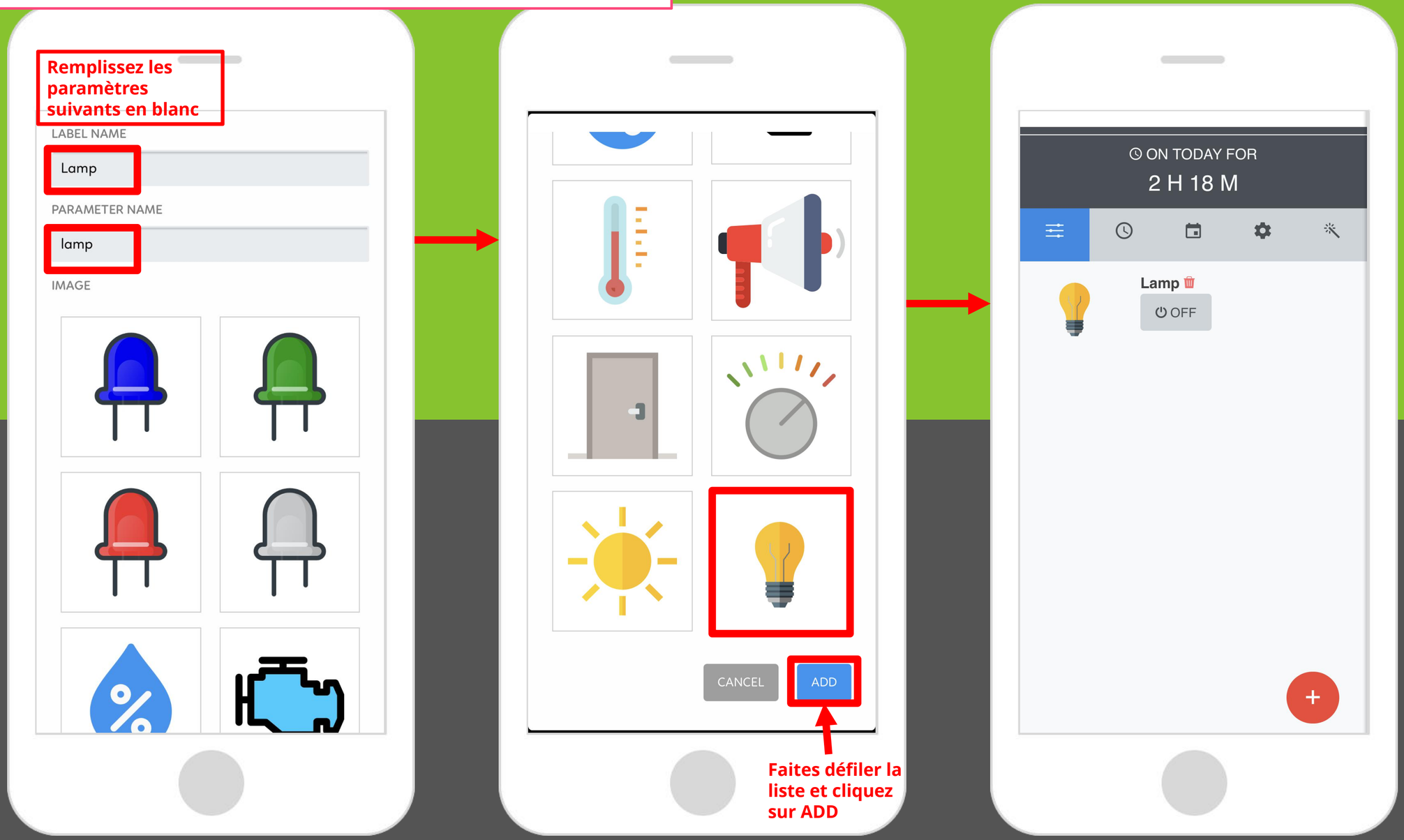
Simulation de système réel

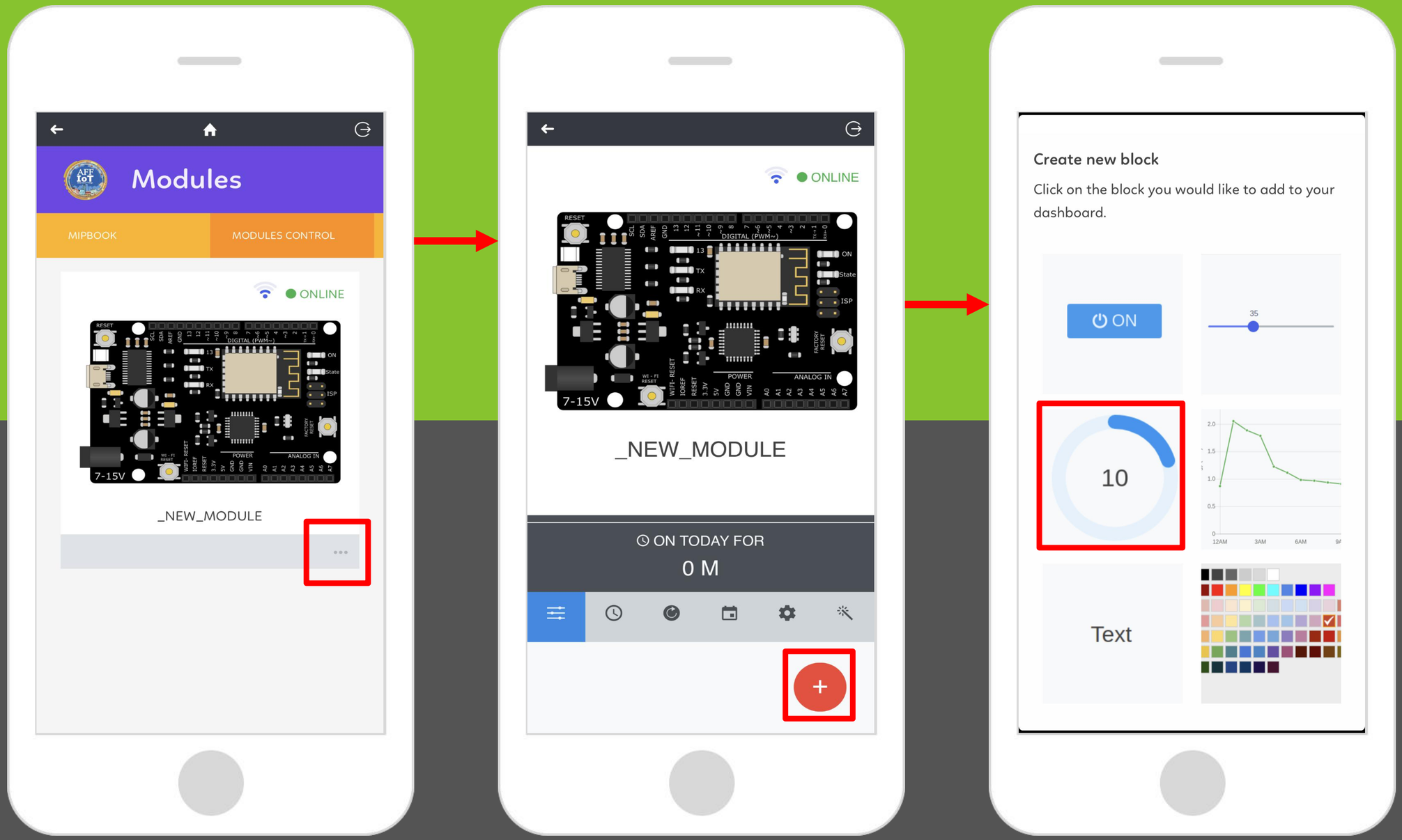
Une simulation est une imitation d'une opération du monde réel.
Ainsi, dans le monde réel, une ampoule est utilisée à la place de la LED.
La LED est utilisée à la place de l'ampoule pour des raisons de sécurité lors de plusieurs essais.


RESET
SCL
SDA
AREF
GND
13
12
~ 11
~ 10
~ 9
8
7
~ 6
~ 5
4
~ 3
2
TX → 1
RX ← 0
DIGITAL (PWM~)
13
TX
RX
ON
State
ISP
FACTORY RESET
Wi - Fi RESET
WIFI- RESET
IOREF
RESET
3.3V
5V
GND
GND
VIN
POWER
A0
A1
A2
A3
A4
A5
A6
A7
ANALOG IN
7-15V
fritzing

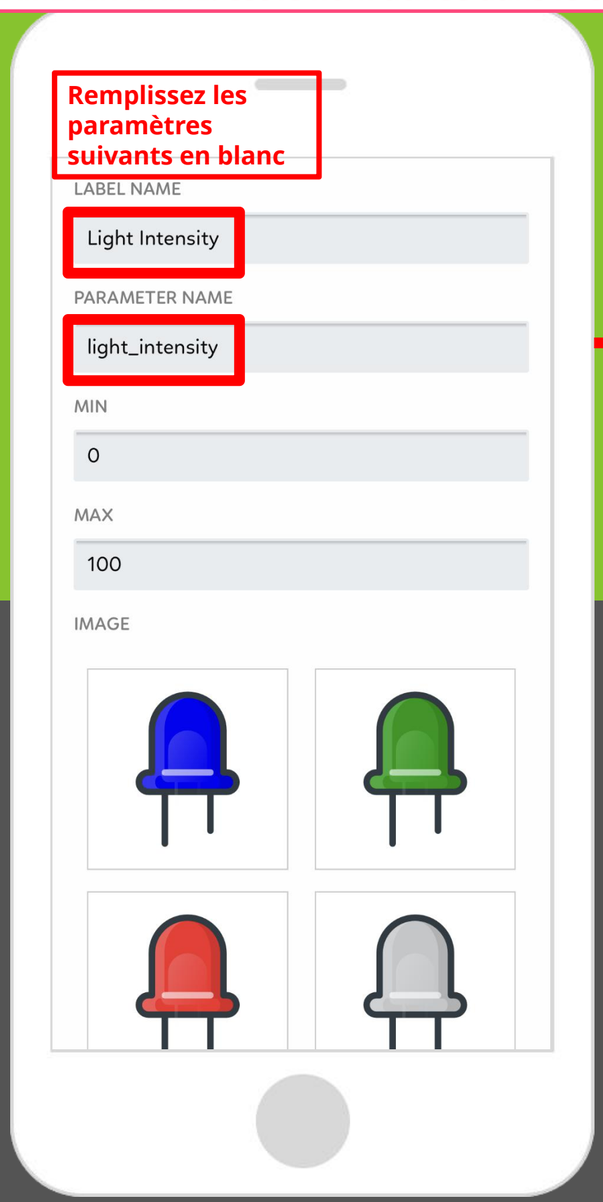

Modules
MIPBOOK
MODULES CONTROL
ONLINE
_NEW_MODULE
ONLINE
_NEW_MODULE
ON TODAY FOR
0 M
Create new block
Click on the block you would like to add to your dashboard.
ON
35
10
Text

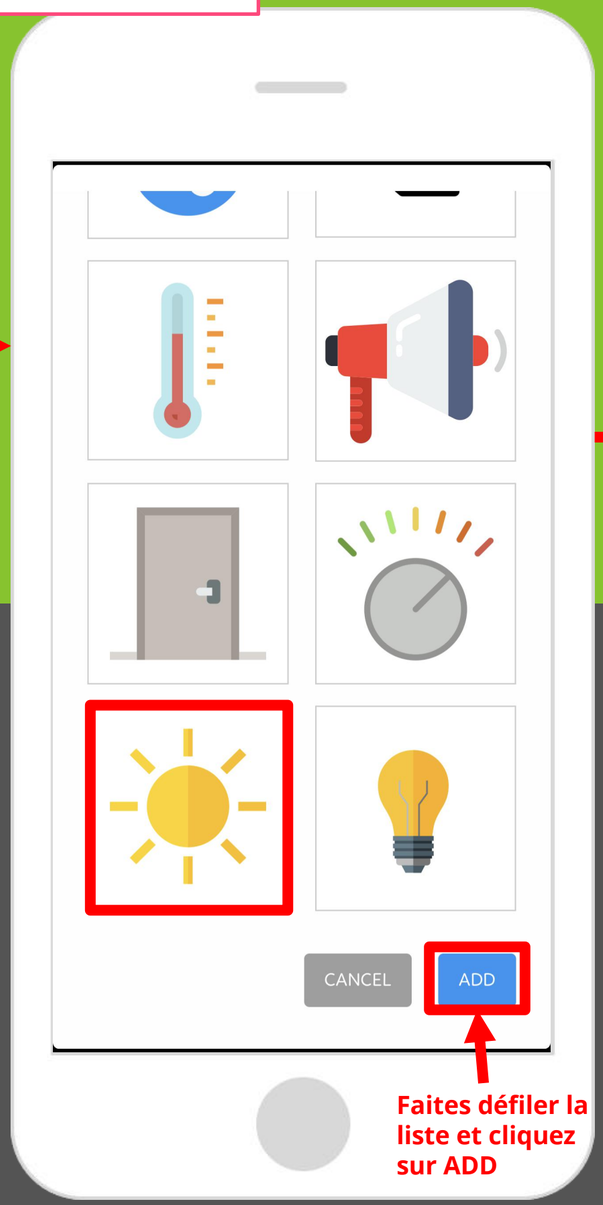
LABEL NAME: est le nom qui apparaît dans l'application.
PARAMETER NAME: est le nom utilisé dans le code.

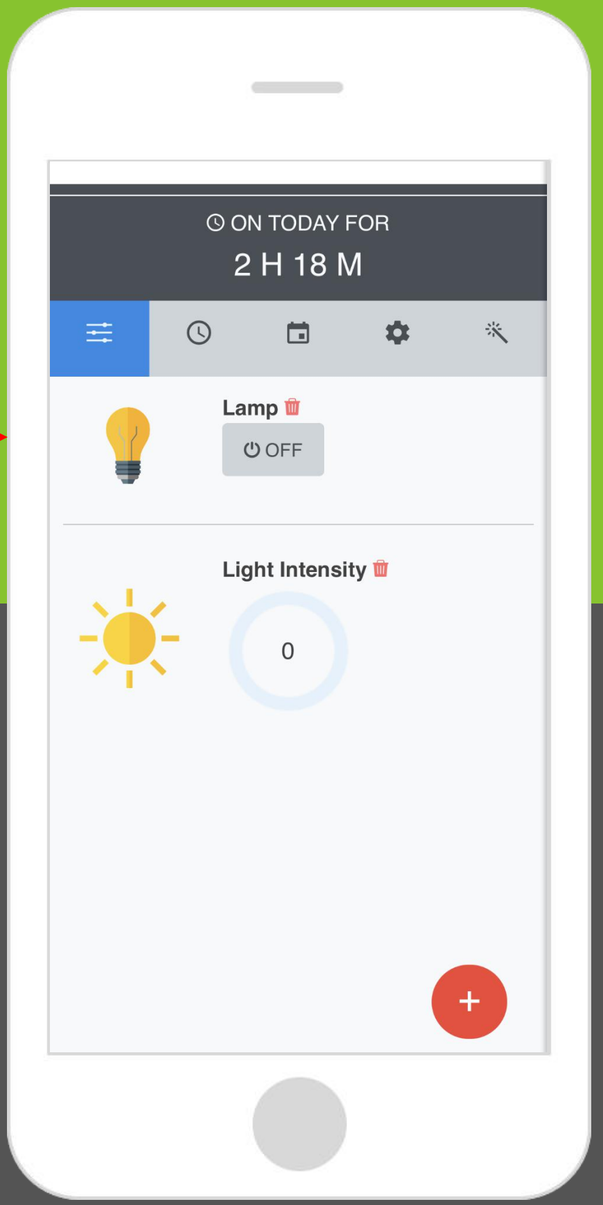


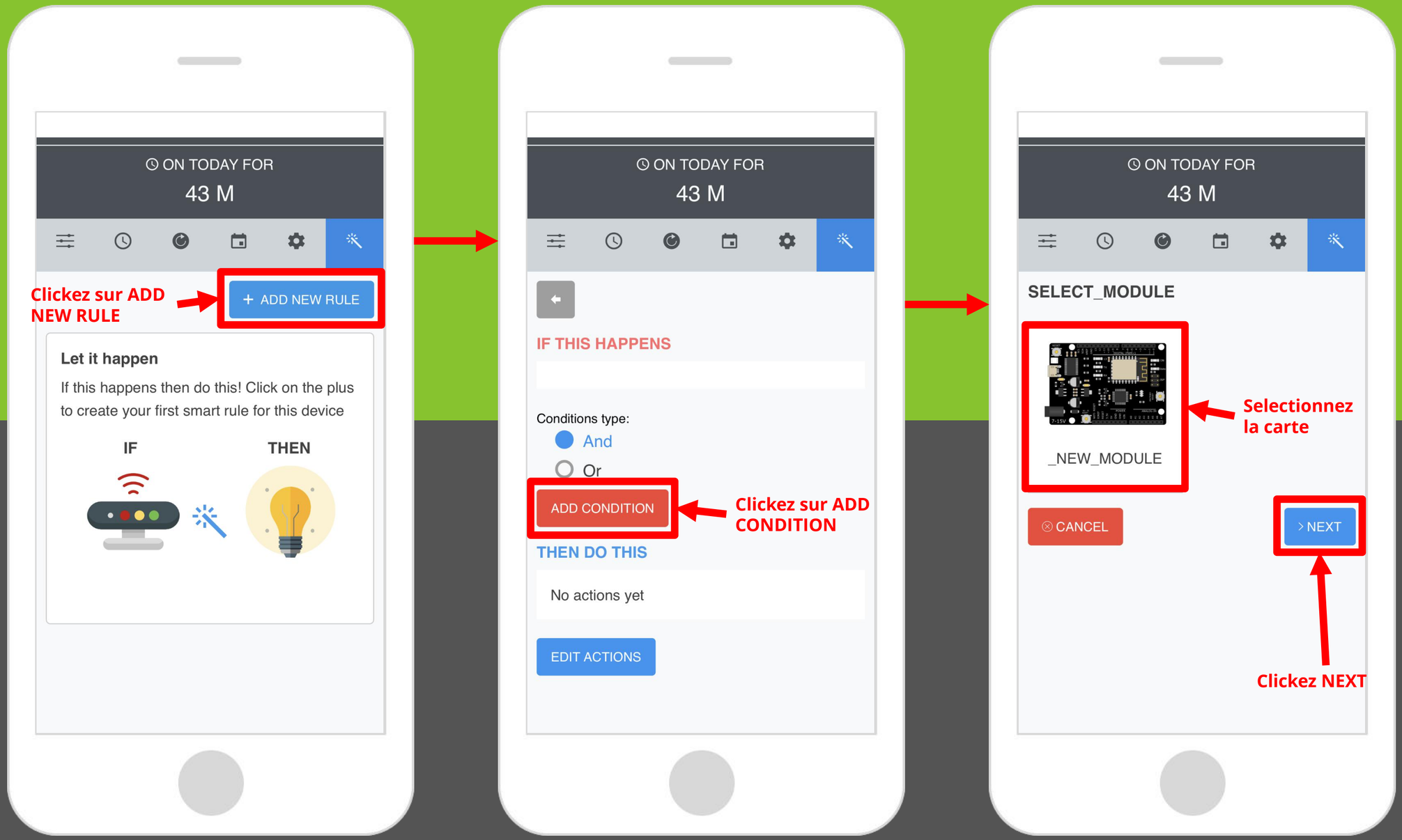

Modules
MIPBOOK
MODULES CONTROL
ONLINE
_NEW_MODULE
ONLINE
_NEW_MODULE
ON TODAY FOR
0 M
Create new block
Click on the block you would like to add to your dashboard.
ON
35
10
Text

LABEL NAME: est le nom qui apparaît dans l'application.
PARAMETER NAME: est le nom utilisé dans le code.



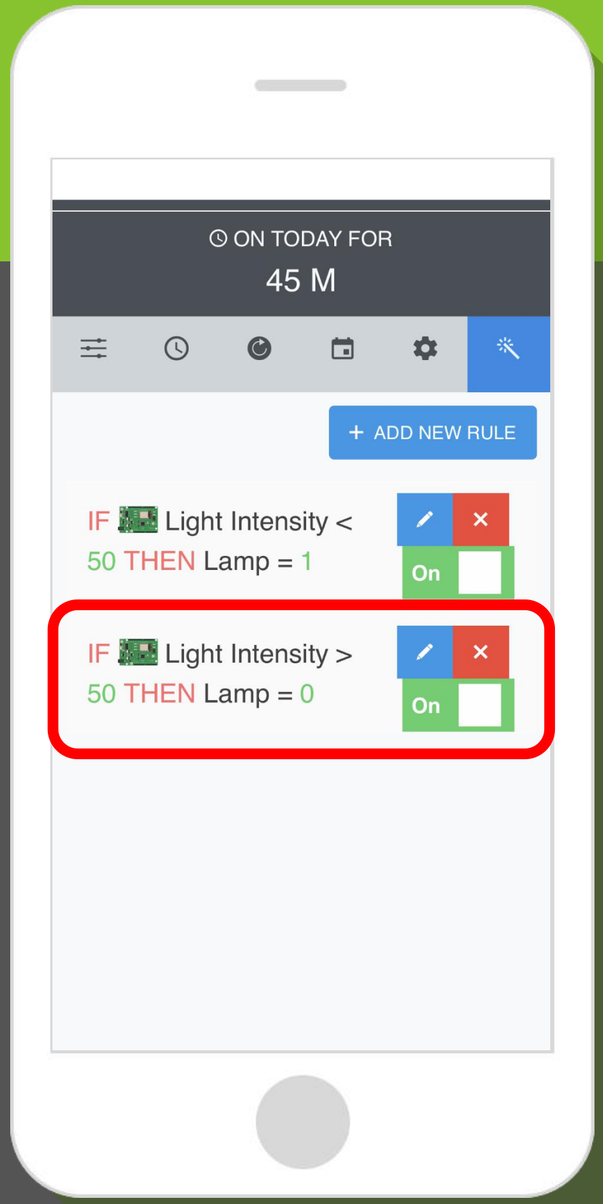





ON TODAY FOR
43 M
Clickez sur ADD NEW RULE
+ ADD NEW RULE
Let it happen
If this happens then do this! Click on the plus to create your first smart rule for this device
IF
THEN
ON TODAY FOR
43 M
IF THIS HAPPENS
Conditions type:
And
Or
ADD CONDITION
Clickez sur ADD CONDITION
THEN DO THIS
No actions yet
EDIT ACTIONS
ON TODAY FOR
43 M
SELECT_MODULE
_NEW_MODULE
Selectionnez la carte
CANCEL
NEXT
Clickez NEXT


ON TODAY FOR
43 M
IF
ONLINE
Lamp
Light Intensity
Selectionnez Light Intensity
IS
=
>
<
50
CANCEL
BACK
FINISH
ON TODAY FOR
43 M
IF THIS HAPPENS
Light Intensity < 50
Conditions type:
And
Or
ADD CONDITION
THEN DO THIS
EDIT ACTIONS
Clickez EDIT ACTIONS
43 M
THEN
1
FEATURE 1
CANCEL
FINISH
3
Done
2
Lamp


ON TODAY FOR
43 M
THEN
Allumer la lampe
FEATURE 1
Lamp
ON
OFF
FEATURE 2
CANCEL
FINISH
Click FINISH
ON TODAY FOR
45 M
Cliquez sur Retour pour revenir
IF THIS HAPPENS
Light Intensity < 50
Conditions type:
And
Or
ADD CONDITION
THEN DO THIS
Lamp 1
EDIT ACTIONS
ON TODAY FOR
45 M
+ ADD NEW RULE
IF Light Intensity < 50 THEN Lamp = 1
On
Vous pouvez modifier, supprimer ou désactiver la règle
Maintenant, lorsque l'intensité lumineuse est inférieure à 50%, la LED jaune (lampe réelle) s'allume.



Finalement

Répétez le processus, mais sélectionnez plutôt une intensité lumineuse inférieure à 50%, et choisissez “OFF” comme action au lieu du “ON”.
Vous pouvez régler le pourcentage d'intensité lumineuse en fonction de la lumière de votre salle.

AFF IoT Board folder > Circuits > Examples > Automated_Lighting_System

Automated_Lighting_System

```
/*Start of mandatory lines of codes in each sketch*/
#define RX A0 // define the Receive pin (RX) to communicate with the WiFi module
#define TX A1 // define the Transmit pin (TX) to communicate with the WiFi module
#include <NeoSWSerial.h> // including the library to use the Software Serial rather than the Hardware Serial (Serial)
NeoSWSerial WiFiModule(RX, TX); //initialize the variable to use in communication with the WiFi module
/*End of mandatory lines of code*/

#define LightSensorPin   A2
#define YellowLedPin     10

void setup() {
  // put your setup code here, to run once:
  WiFiModule.begin(19200); // begin the communication between the WiFi module and the microcontroller on the board
}

void loop() {
  // put your main code here, to run repeatedly:
  if (WiFiModule.available() > 0) // if the WiFi module receive data from the server
  {
    String Command = WiFiModule.readStringUntil('\n'); // read the command sent from the WiFi module to the microcontroller

    if (Command.indexOf("lamp=1") >= 0) // if the received command contains "lamp=1" turn on the LED
    {
      digitalWrite(YellowLedPin, HIGH); // turn on the LED
    }
    if (Command.indexOf("lamp=0") >= 0) // if the received command contains "lamp=0" turn it off
    {
      digitalWrite(YellowLedPin, LOW); // turn off the LED
    }
  }

  int lightIntensity; // declare an integer to read the voltage

  lightIntensity = analogRead(LightSensorPin); // read the actual converted value (between 0 and 1023)
  lightIntensity = map(lightIntensity, 0, 1023, 0, 100); // transform the scale from 0 and 1023 to 0% and 100%

  WiFiModule.println("light_intensity=" + String(lightIntensity)); // send the light intensity value to the server

  delay(3000); // wait for 3 second (3000ms = 3s)
}
```

Ouvrir votre croquis:
Automated_Lighting_System

Ce que vous **devez voir**

Vous devriez voir la LED jaune s'allumer lorsque la luminosité de la salle diminue et s’éteindre lorsque la luminosité augmente.

Dépannage

Reportez-vous aux procédures de dépannage pour les projets ‘Contrôler une LED’ et ‘Lecture d'intensité d’une LED’.

Application dans la réalité

Récemment, la plupart des lampes sont équipées d'un capteur de lumière intégré. Ils s’allument lorsque la lumière est inférieure au seuil prédéfini et s’éteignent dans le cas contraire.

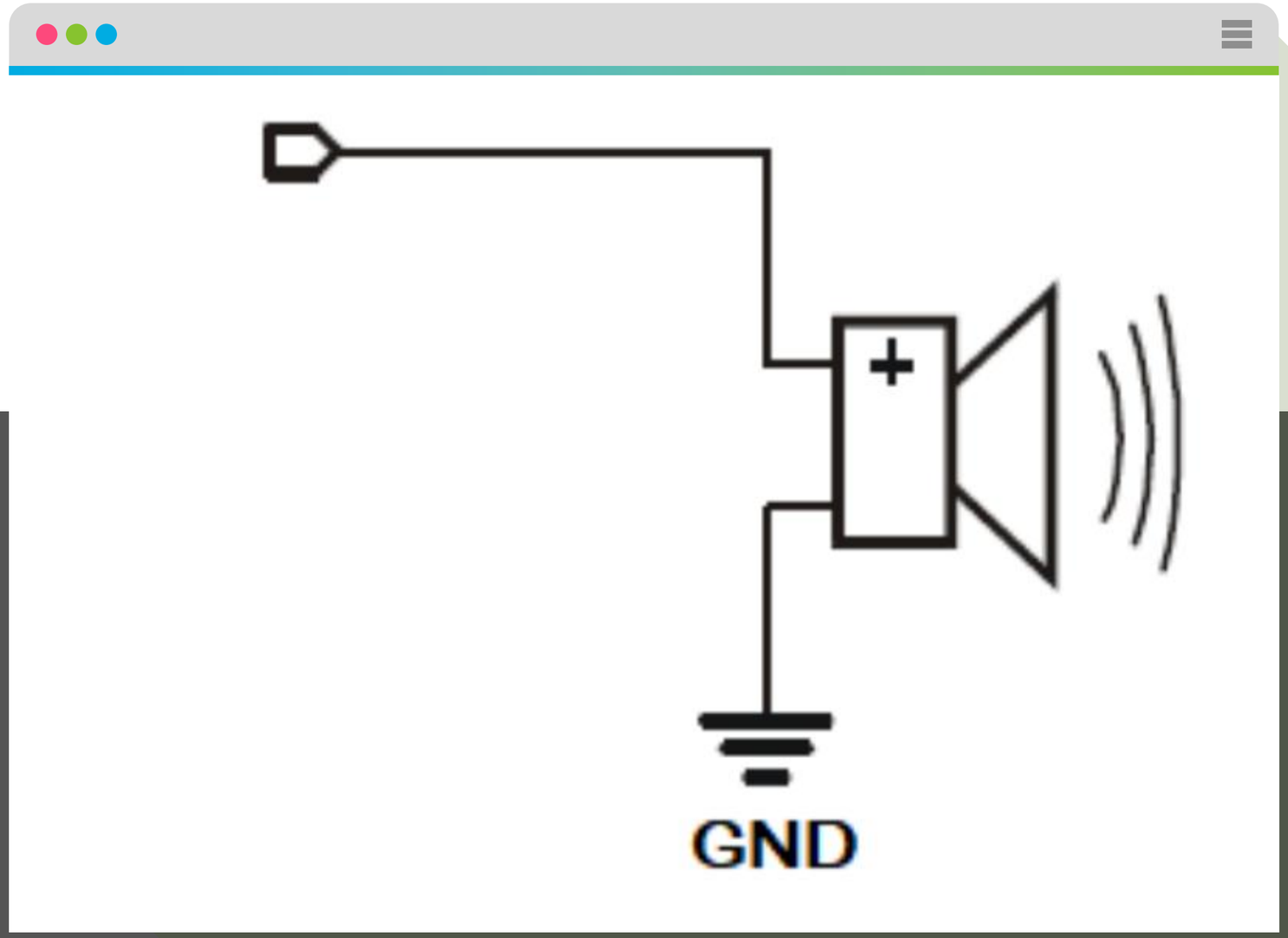
12 Circuit d'alarme de surchauffe d’un bâtiment

Bippeur x1

Thermistance x1

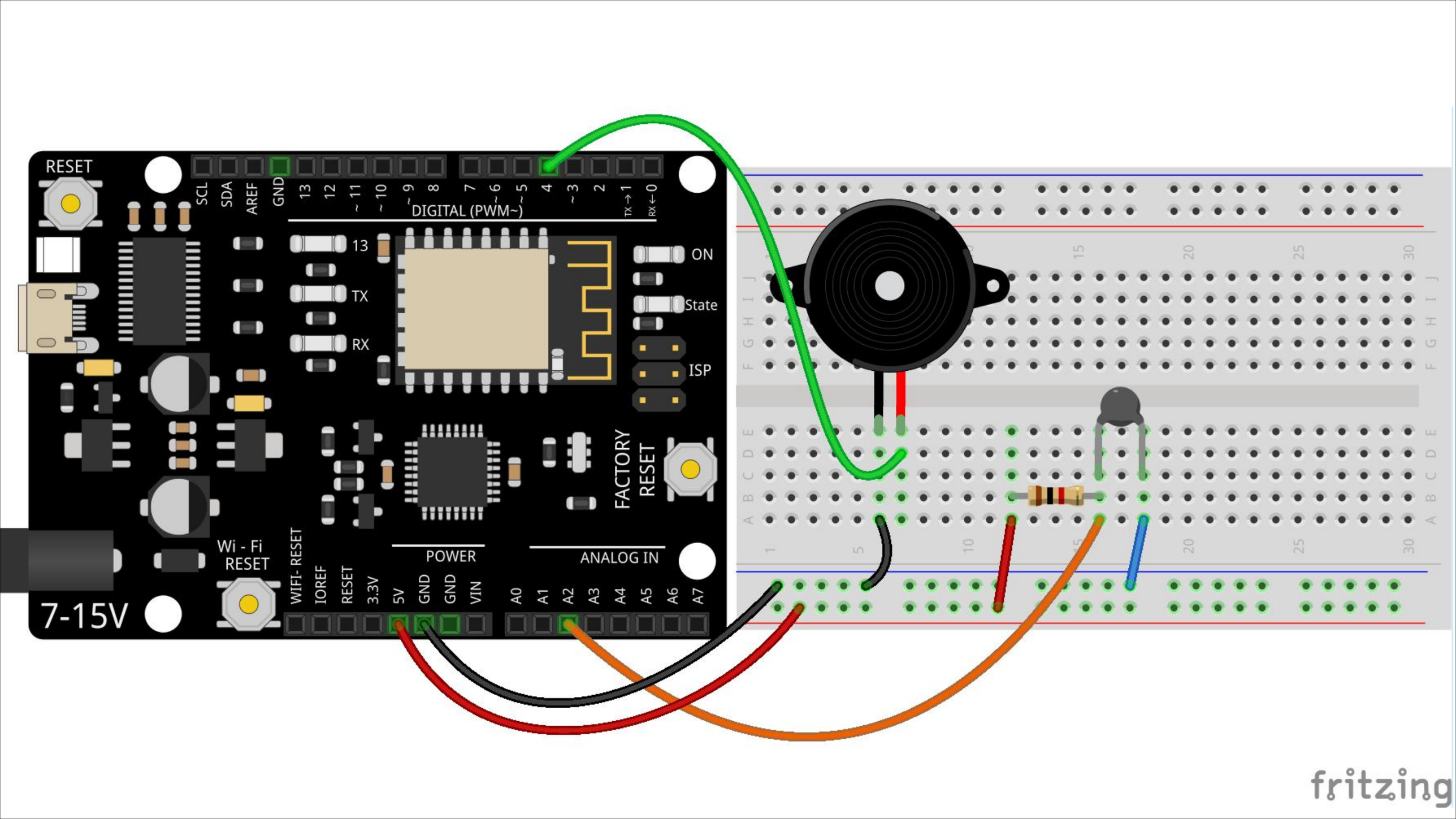
Resistance 1KΩ x1

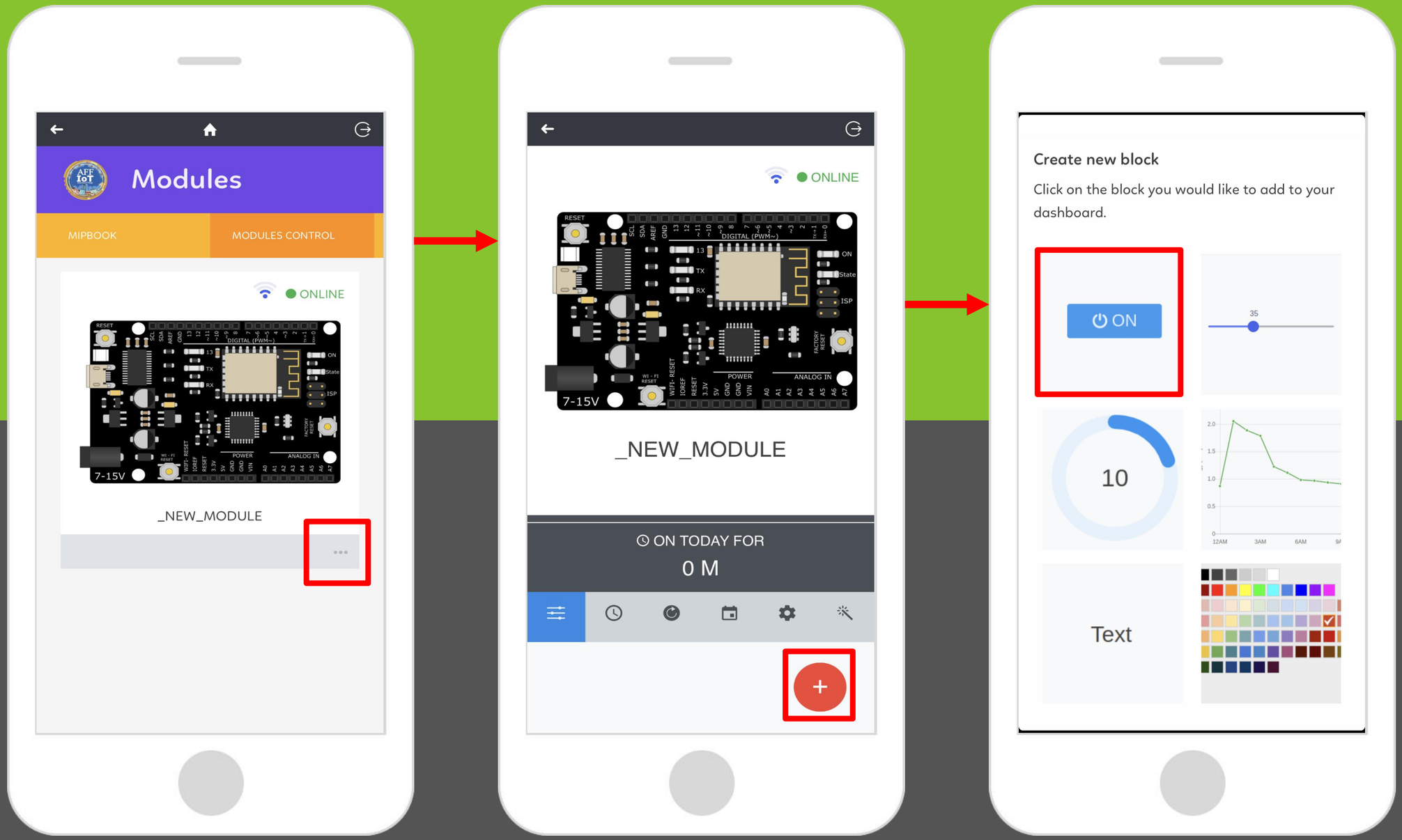
Fils de connexion x7



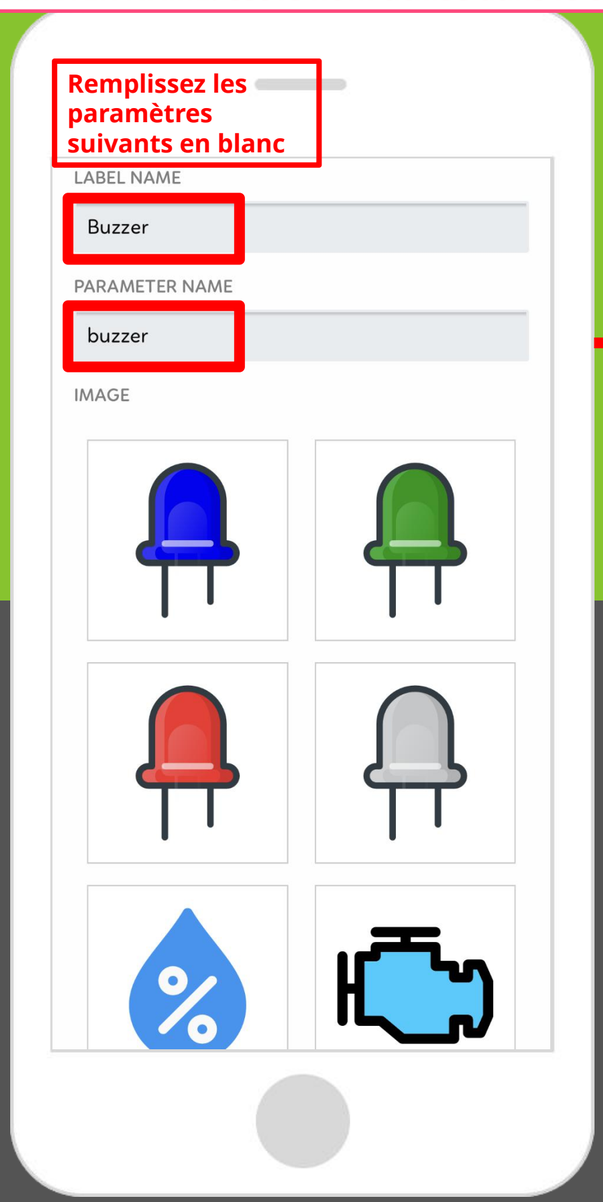
Bippeur

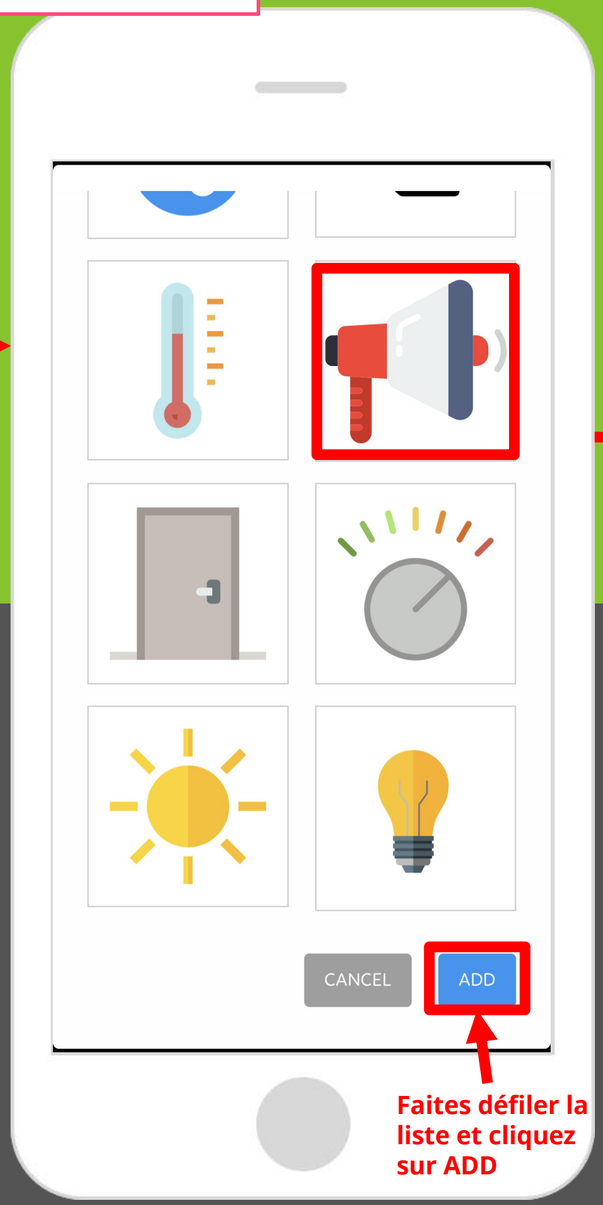
Nous utiliserons un bippeur qui émettra un son lorsque nous lui appliquerons une tension!
Ce tutoriel est l'introduction aux systèmes d'alarme.

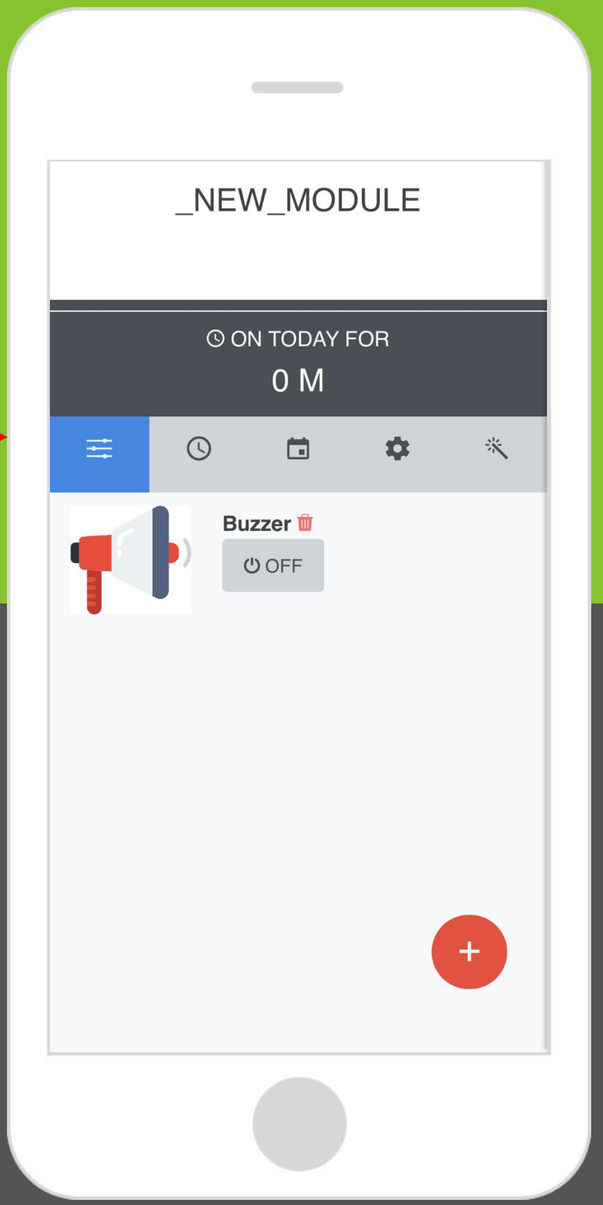

RESET
SCL
SDA
AREF
GND
13
12
~ 11
~ 10
~ 9
8
7
~ 6
~ 5
4
~ 3
2
TX → 1
RX ← 0
DIGITAL (PWM~)
13
TX
RX
ON
State
ISP
FACTORY RESET
Wi - Fi RESET
WIFI- RESET
IOREF
RESET
3.3V
5V
GND
GND
VIN
POWER
A0
A1
A2
A3
A4
A5
A6
A7
ANALOG IN
7-15V
fritzing

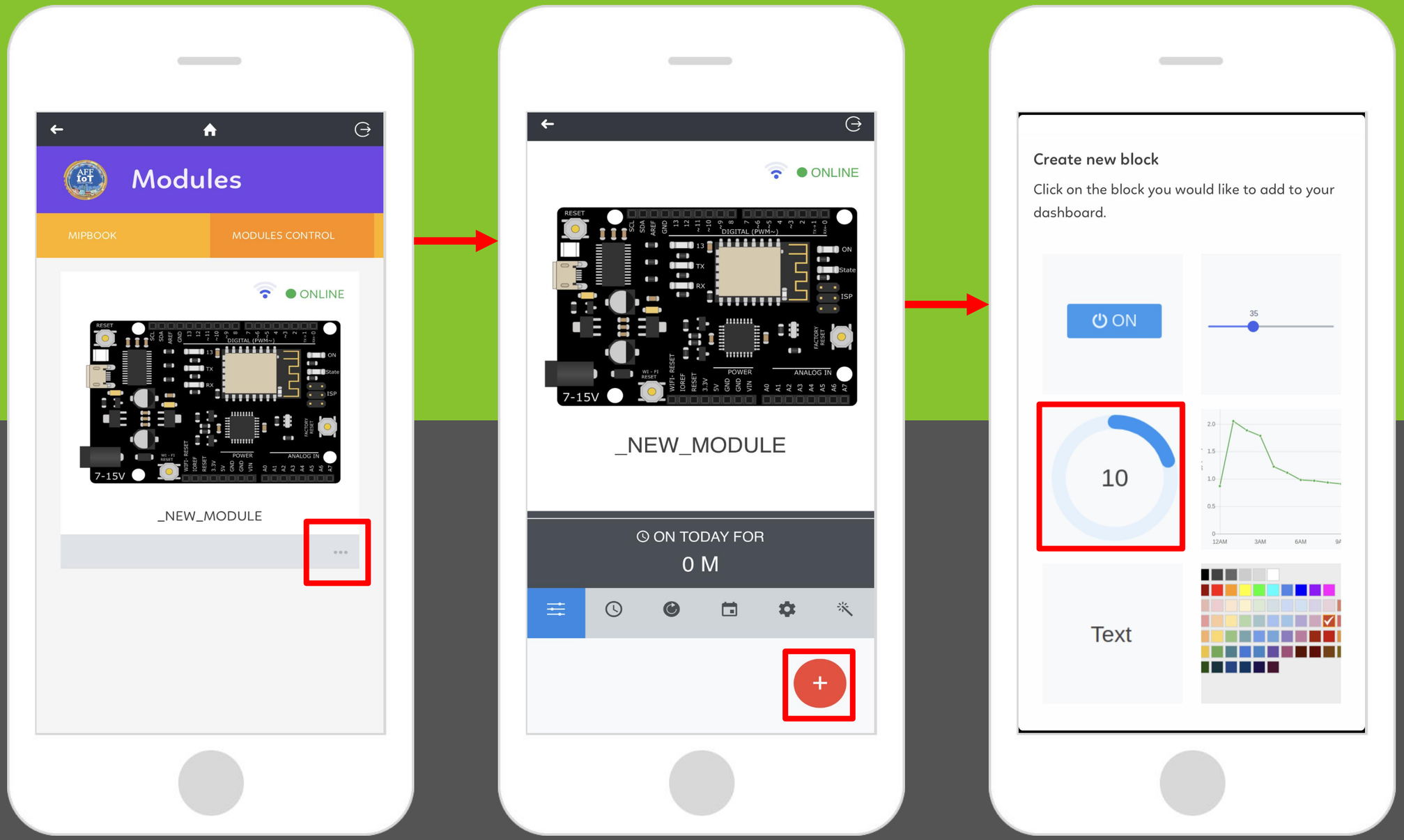

Modules
MIPBOOK
MODULES CONTROL
ONLINE
_NEW_MODULE
ON TODAY FOR
0 M
Create new block
Click on the block you would like to add to your dashboard.
ON
35
10
Text

LABEL NAME: est le nom qui apparaît dans l'application.
PARAMETER NAME: est le nom utilisé dans le code.



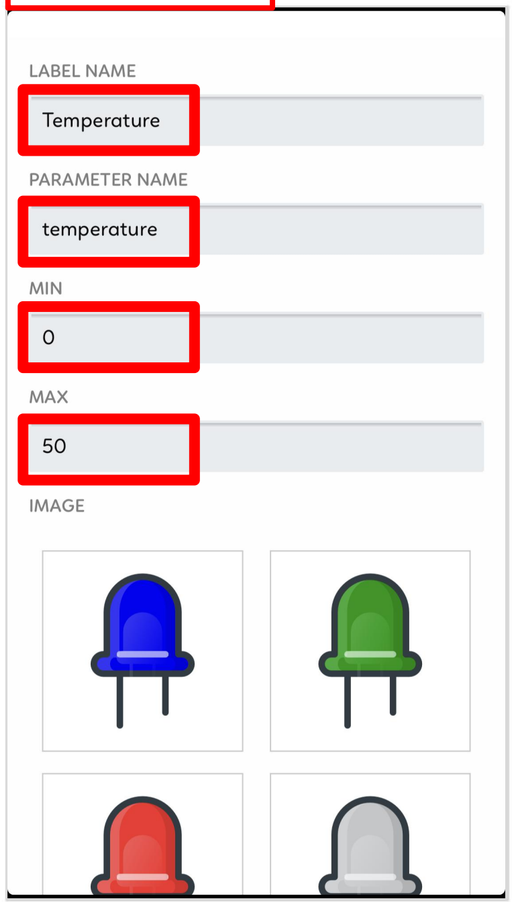


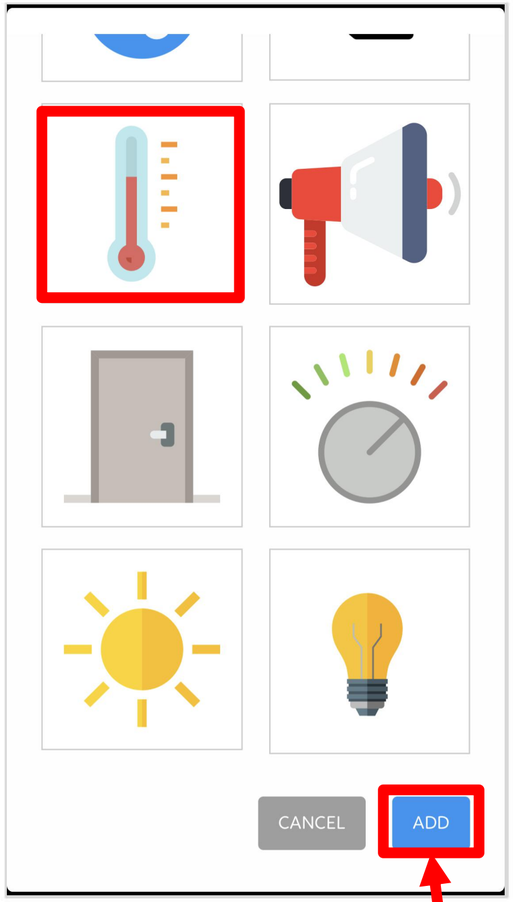



Modules
MIPBOOK
MODULES CONTROL
ONLINE
_NEW_MODULE
ONLINE
_NEW_MODULE
ON TODAY FOR
0 M
Create new block
Click on the block you would like to add to your dashboard.
ON
35
10
Text

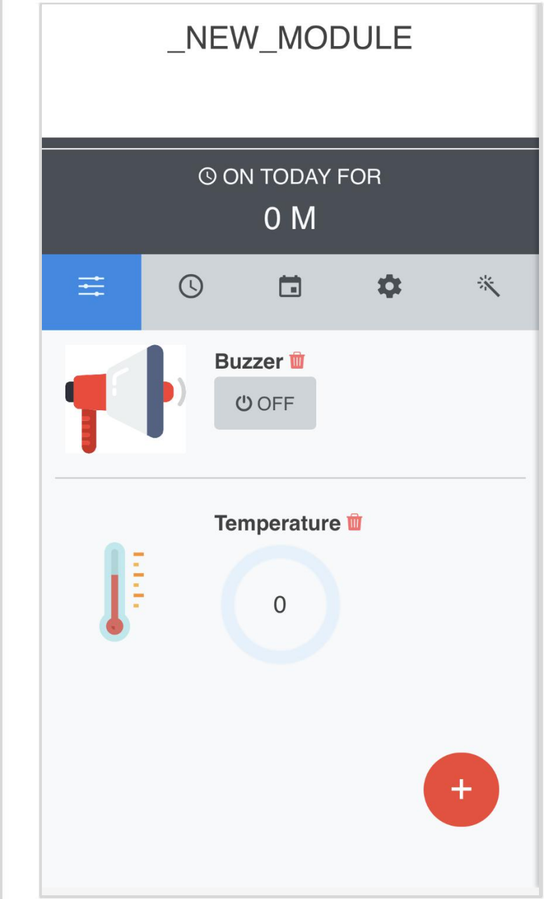
LABEL NAME: est le nom qui apparaît dans l'application.
PARAMETER NAME: est le nom utilisé dans le code.

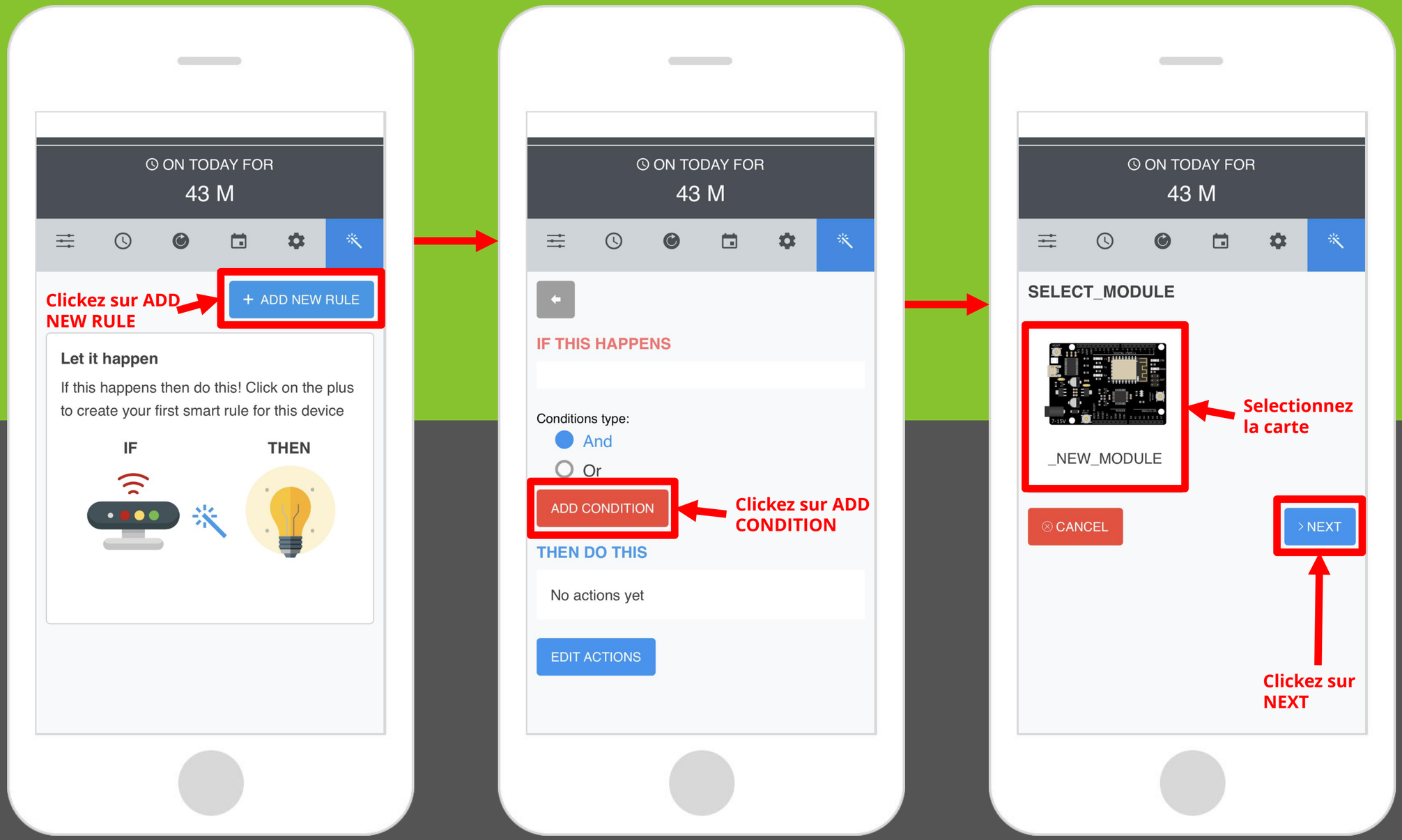
Remplissez les paramètres suivants en blanc





Faites défiler la liste et cliquez sur ADD




ON TODAY FOR
43 M
Clickez sur ADD NEW RULE
+ ADD NEW RULE
Let it happen
If this happens then do this! Click on the plus to create your first smart rule for this device
IF
THEN
ON TODAY FOR
43 M
IF THIS HAPPENS
Conditions type:
And
Or
ADD CONDITION
Clickez sur ADD CONDITION
THEN DO THIS
No actions yet
EDIT ACTIONS
ON TODAY FOR
43 M
SELECT_MODULE
_NEW_MODULE
Selectionnez la carte
CANCEL
NEXT
Clickez sur NEXT


ON TODAY FOR
48 M
IF
ONLINE
Buzzer
Temperature
Selectionnez Temperature
IS
=
>
<
35
CANCEL
BACK
FINISH
ON TODAY FOR
48 M
IF THIS HAPPENS
Temperature > 35
Conditions type:
And
Or
ADD CONDITION
THEN DO THIS
EDIT ACTIONS
Clickez sur EDIT ACTIONS
48 M
THEN
1
FEATURE 1
CANCEL
FINISH
3
Done
2
Buzzer


ON TODAY FOR
48 M
Cliquez sur Retour pour revenir
IF THIS HAPPENS
Temperature > 35
Conditions type:
And
Or
ADD CONDITION
THEN DO THIS
Buzzer =1
EDIT ACTIONS
ON TODAY FOR
48 M
+ ADD NEW RULE
IF Temperature > 35 THEN Buzzer = 1
On
Vous pouvez modifier, supprimer ou désactiver la règle

AFF IoT Board folder > Circuits > Examples > Overheat_Alarm

Overheat_Alarm

```
/*Start of mandatory lines of codes in each sketch*/
#define RX A0 // define the Receive pin (RX) to communicate with the WiFi module
#define TX A1 // define the Transmit pin (TX) to communicate with the WiFi module
#include <NeoSWSerial.h> // including the library to use the Software Serial rather than the Hardware Serial (Serial)
NeoSWSerial WiFiModule(RX, TX); //initialize the variable to use in communication with the WiFi module
/*End of mandatory lines of code*/

#define BuzzerPin   4
#define TemperatureSensorPin A2


void setup() {
  // put your setup code here, to run once:
  WiFiModule.begin(19200); // begin the communication between the WiFi module and the microcontroller on the board
  pinMode(BuzzerPin, OUTPUT); // configure the pin connected to the Buzzer to be an output
}
void loop() {
  // put your main code here, to run repeatedly:
  float Voltage; // declare a decimal variable to store the read voltage
  int Temperature; // declare an integer variable to calculate the temperature
  Voltage = analogRead(TemperatureSensorPin) * 0.0048828125; // voltage = analog_value * (5/1024);

  Temperature = -21.231 * (Voltage - 3.765);
  // this equation is the linearized form of the temperature equation between 0 and 50 degrees (specific to the thermistor in the kit)

  WiFiModule.println("temperature=" + String(Temperature)); // send the Temperature value to the server

  delay(3000); // wait for 3 second (3000ms = 3s)

  if (WiFiModule.available() > 0) // if the WiFi module receive data from the server
  {
    String Command = WiFiModule.readStringUntil('\n'); // read the command sent from the WiFi module to the microcontroller
    if (Command.indexOf("buzzer=1") >= 0) // if the received command contains "buzzer=1" trigger on the Buzzer
    {
      digitalWrite(BuzzerPin, HIGH); // turn on the LED
    }
    if (Command.indexOf("buzzer=0") >= 0) // if the received command contains "buzzer=0" trigger it off
    {
      digitalWrite(BuzzerPin, LOW); // turn off the LED
    }
  }
}
```

Ouvrir votre croquis : Overheat_Alarm

Dans ce tutoriel, nous avons supposé qu'un capteur de température est installé dans la salle. Lorsque la température augmente pour atteindre 35°C, le bipeur doit déclencher une alarme.
Ce code est une combinaison du code permettant de contrôler le bipeur et du code permettant de surveiller la température.

Ce que vous **devez voir**

Vous ne devriez rien voir! Mais vous devriez entendre un son lorsque vous approchez un briquet en direction du capteur. Si cela ne fonctionne pas, assurez-vous d'avoir correctement assemblé le circuit, vérifié et téléchargé le code sur votre carte ou consultez les conseils de dépannage ci-dessous.

Dépannage

Pas de Son
Compte tenu de la taille et de la forme du bippeur, il est facile de manquer les bons trous sur la planche d'essai. Vérifiez sa position.

Encore pas de Son !
Ce bippeur est polarisé, veillez donc à ce que la broche avec l'indication «+» soit connectée à la broche numérique et l'autre à GND.

Application dans le monde Réel

Tous les systèmes de sécurité utilisent des avertisseurs sonores (ou sirènes) lors du déclenchement d'une alarme.

Crédit

Tous les schémas de circuit sont créés avec Fritzing (une initiative matérielle à code source ouvert qui rend l'électronique accessible à tous, en tant que matériel créatif).

Merci au guide de "Sparkfun Inventor" pour ses images de clarification.