Workshop 5: Google Home and ESP

1.1 Description

The Google Home as we know it is a useful device with which you can do a lot. However, additional features such as controlling lights cost a lot of money. In this workshop we describe a cheaper method to control an ESP via Google Home.

Part 1: turn on/off the lights

Part 2: Measuring temperature

1.2 Equipment

- ESP32 or ESP8266
- Google hub
- Smartphone
- Laptop
- LED
- Sensor (DHT-sensor)

1.3 Info

https://www.instructables.com/Connecting-My-ESP8266-With-Google-Home/

https://learn.adafruit.com/create-an-internet-of-things-dashboard-with-adafruit-dot-io

1.4 PART 1: turn on/off light

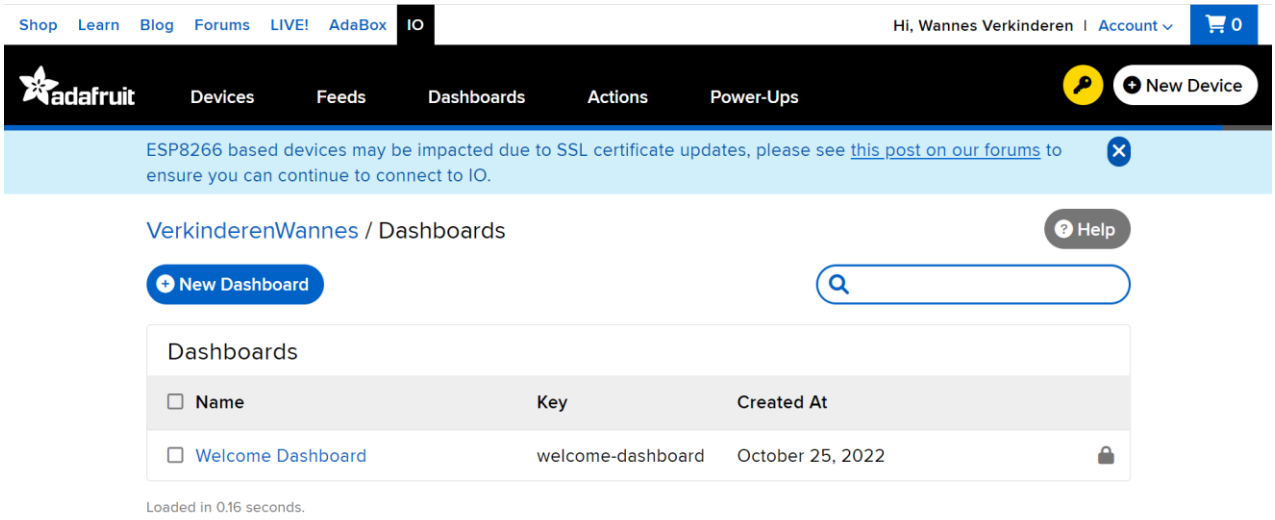
1.4.1 Set up

1.4.1.1 Necessary software

To combine Google Home and THE ESP, we use some external programs. We use IFTTT and adafruit.io to get the desired result. That's why it's important to sign up on both programs, preferably with the same email.

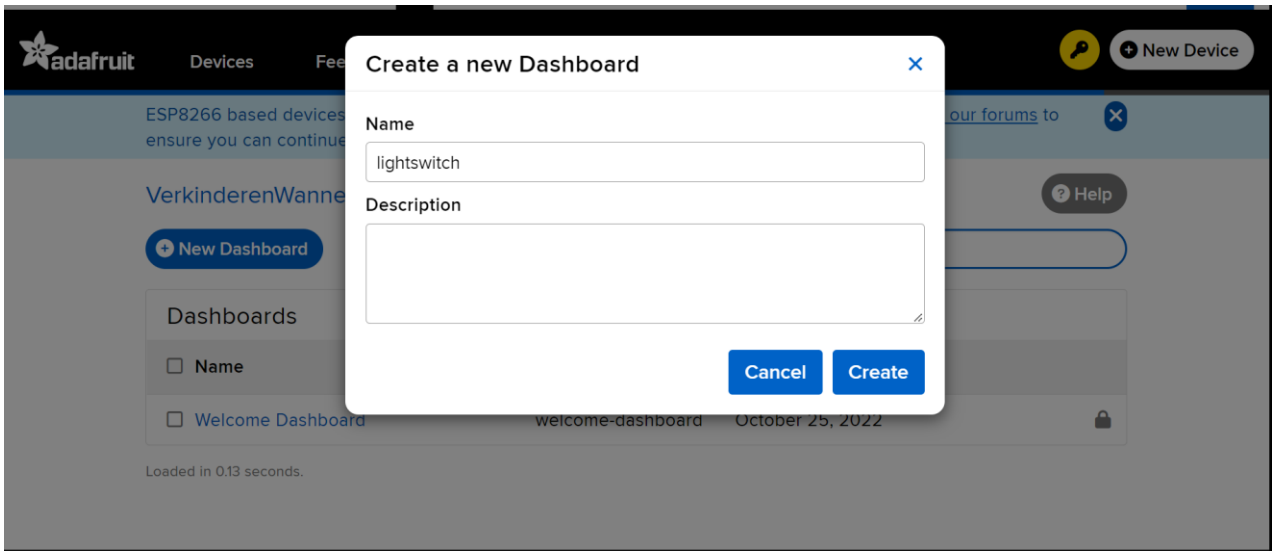
1.4.1.2 Set adafruit

First of all, we make sure that adafruit.io is ready for use so that we don't have to think about this later. Go to adafruit.io and log in. Normally you should see this screen now:

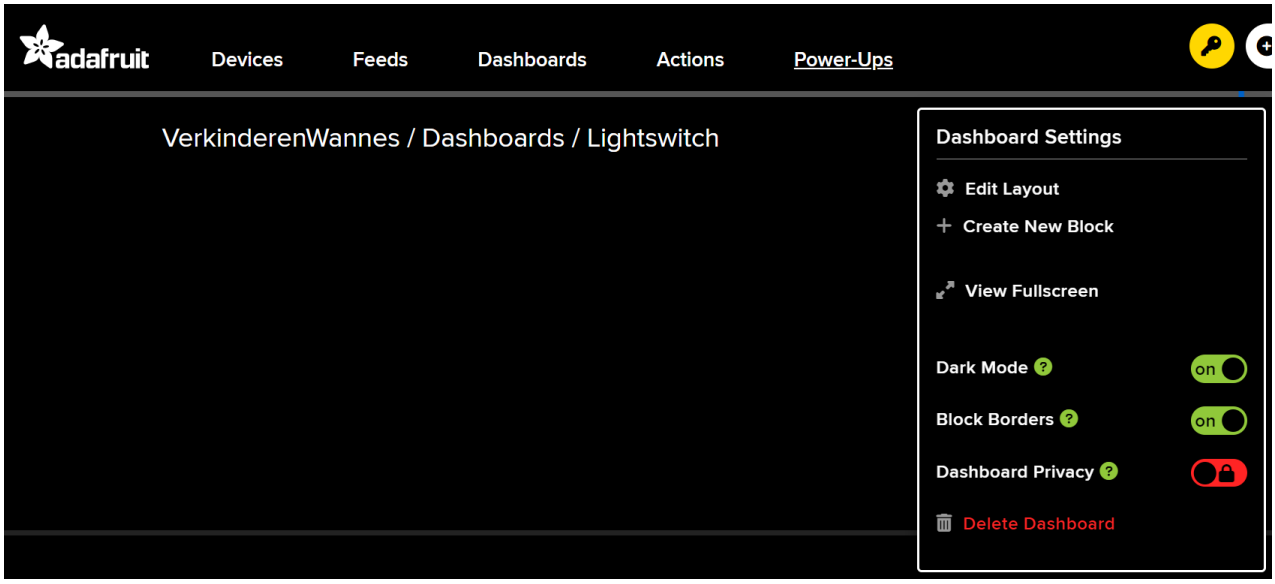


Then click "+ new dashboard".

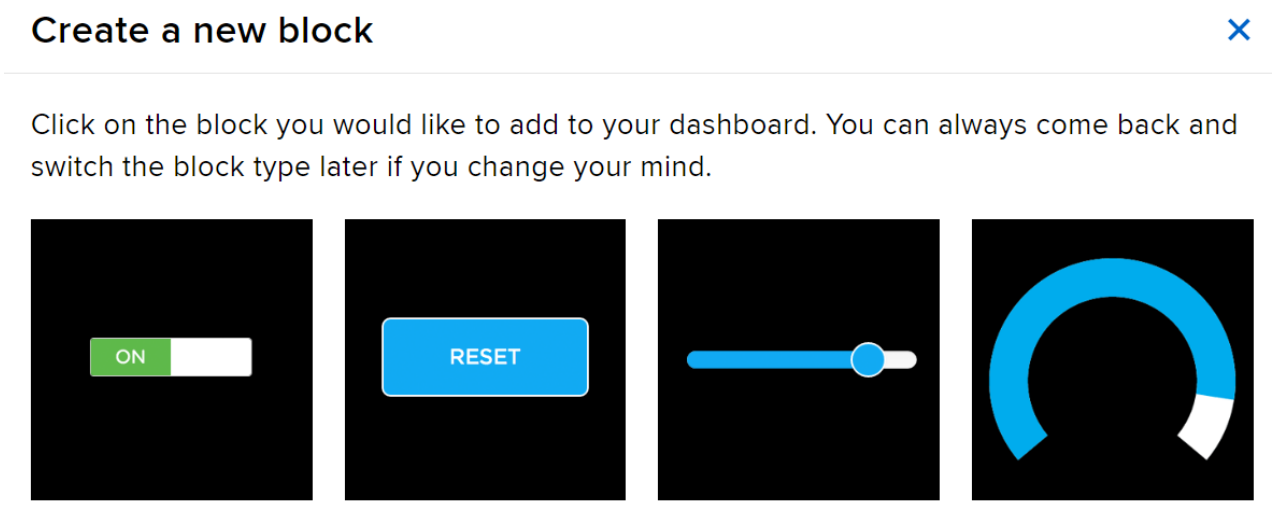
Then assign a name tour new dashboard. For example, call it "lightswitch."

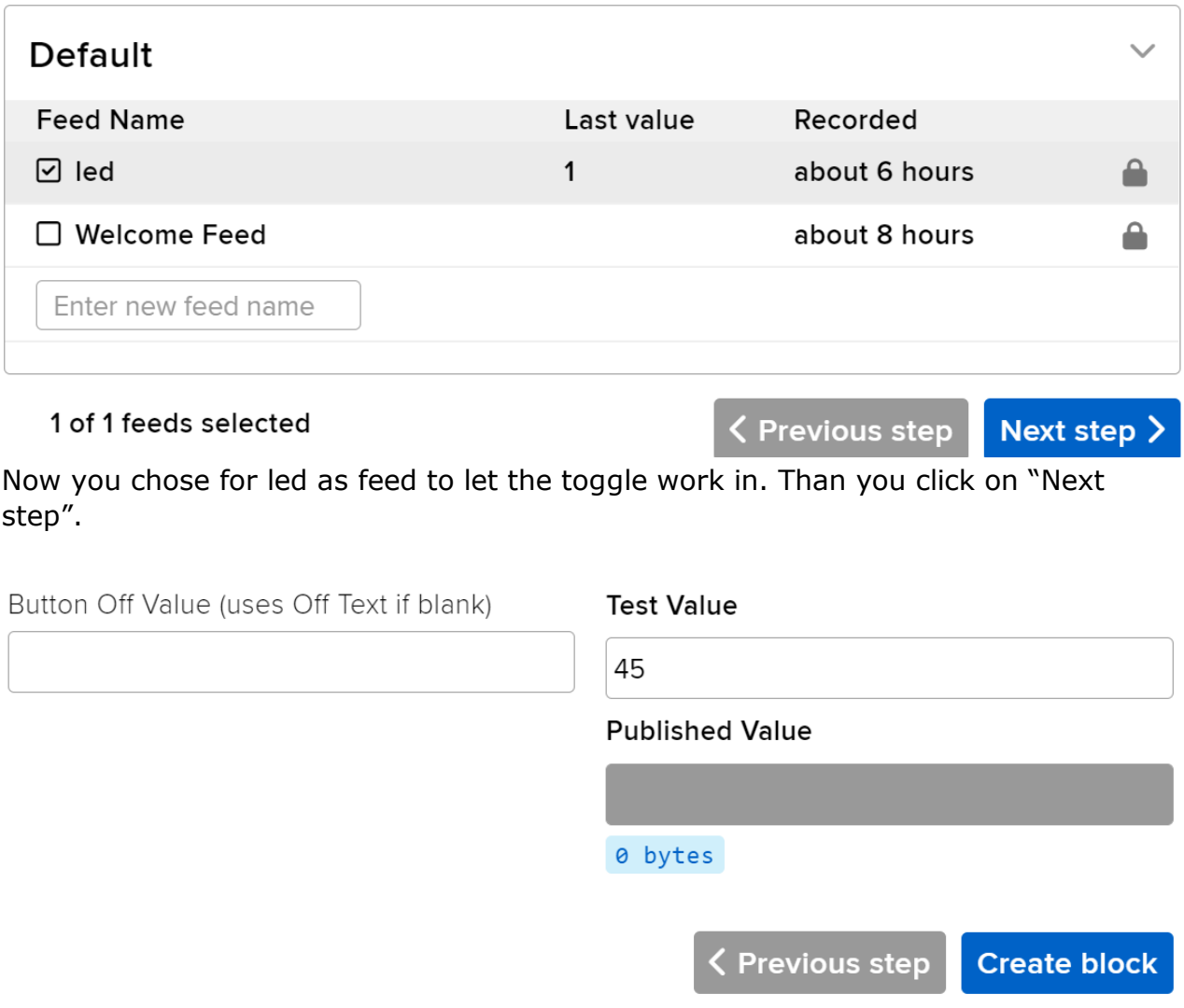


Then you end up on this screen without the push button. Then, in the upper-right corner of the screen you click on the gear.
Then you click on "Create New Block".



When you click on that you need to choose "Toggle".





Now you chose for led as feed to let the toggle work in. Than you click on "Next step".

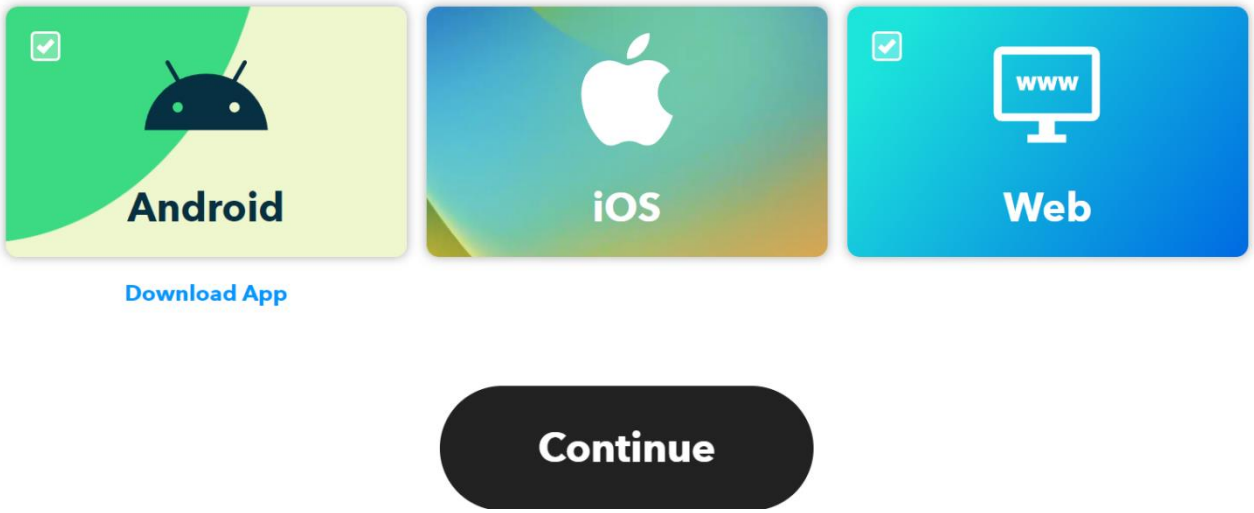
So to complete everything you need to click on "Create block"



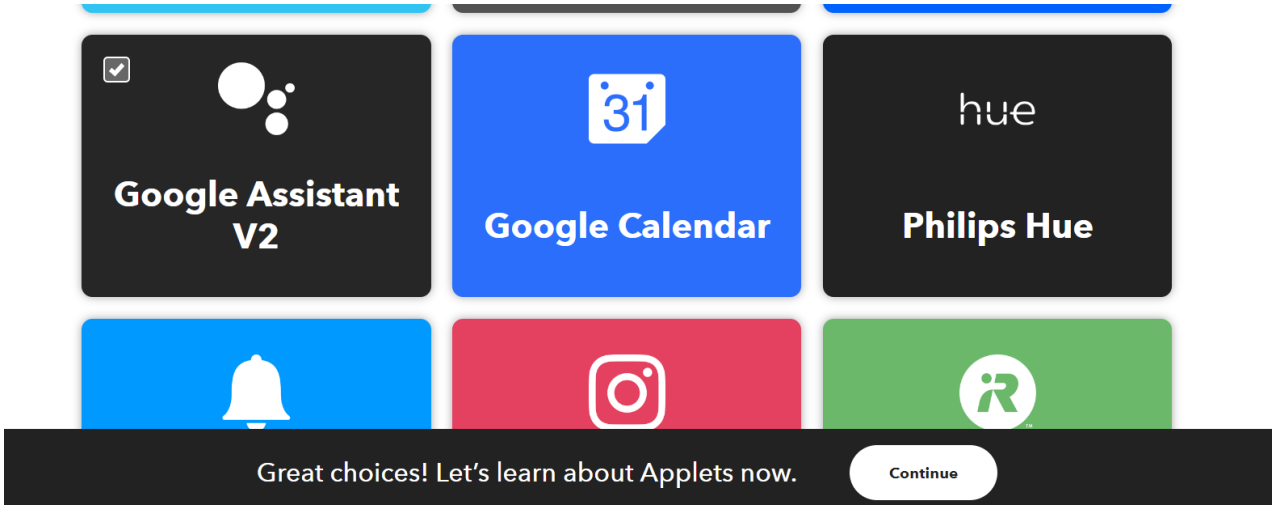
If you did it wright you will see something like this.

1.4.1.3 Set up IFTTT

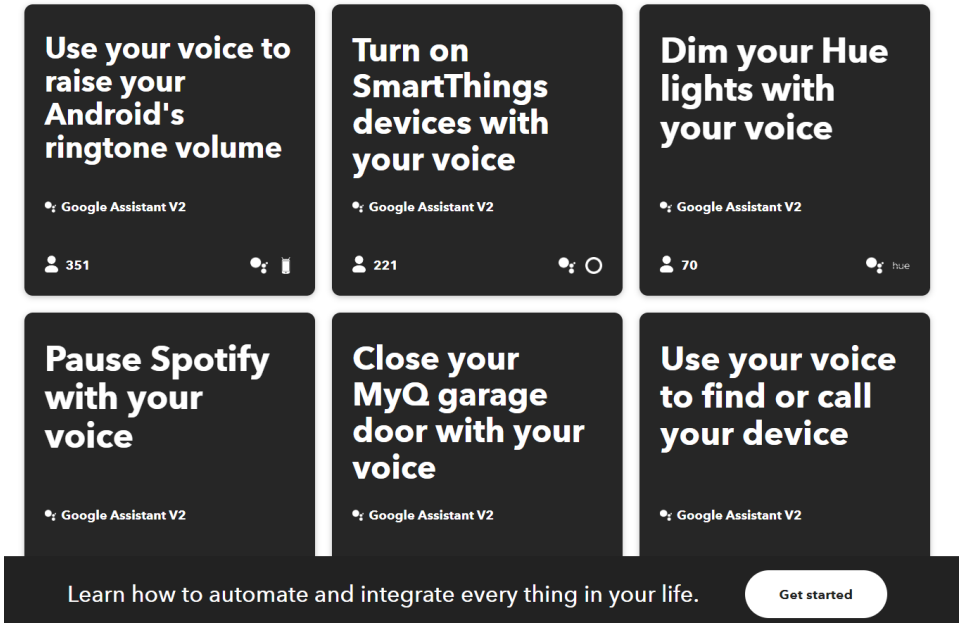
Now that we've set up Adafruit, it's time to set up IFTTT. Go to IFTTT and click on "Get started". There you need to make an account. Normally you should get next screen:



Then, on this screen we chose for "Android" and for "Web". Then you click on "Continue" and go to the next page.

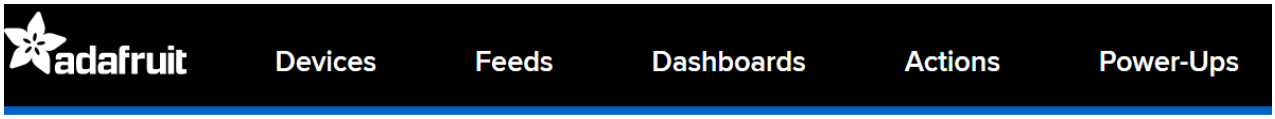


On this page you chose for "Google Assistant V2" and then click "Continue" again.

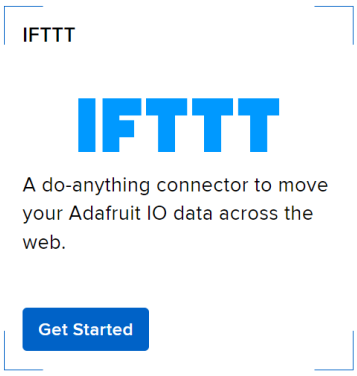


When you see this screen you just need to click on “Get started”. And go to the next screen.

After you created an Adafruit account and a IFTTT account you need to connect those 2 accounts with each other. So you need to go to Adafruit again and click on “Power-Ups”



Than you click on “Get Started” under the IFTTT.



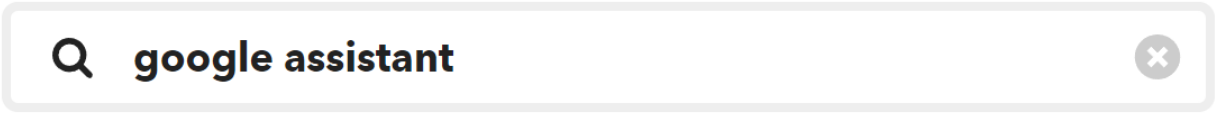
When you click on “Get Started” it will automatically connect to each other. Now you can create a trigger on IFTTT.



On this screen you go to the upper right corner and click on “Create”. There you need to add the google assistant.

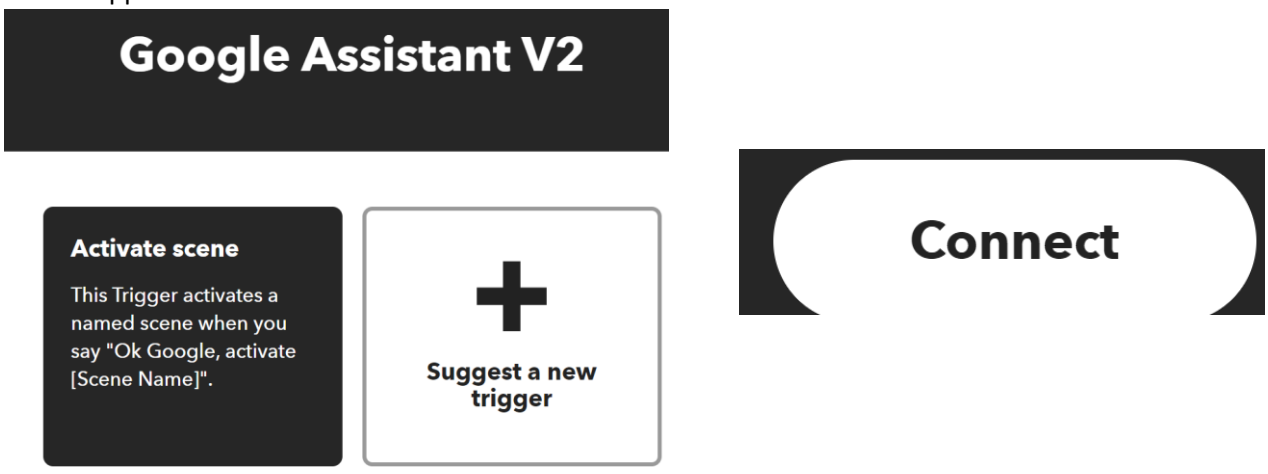


So you click on “Add” .

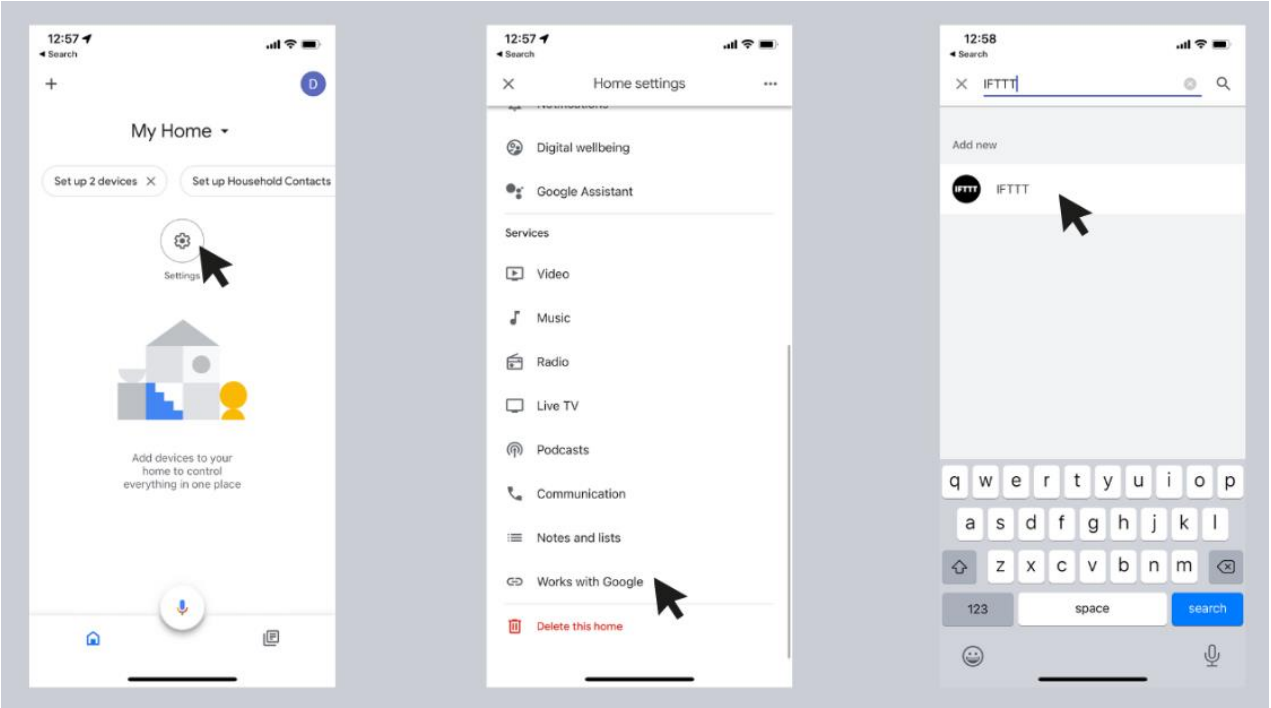




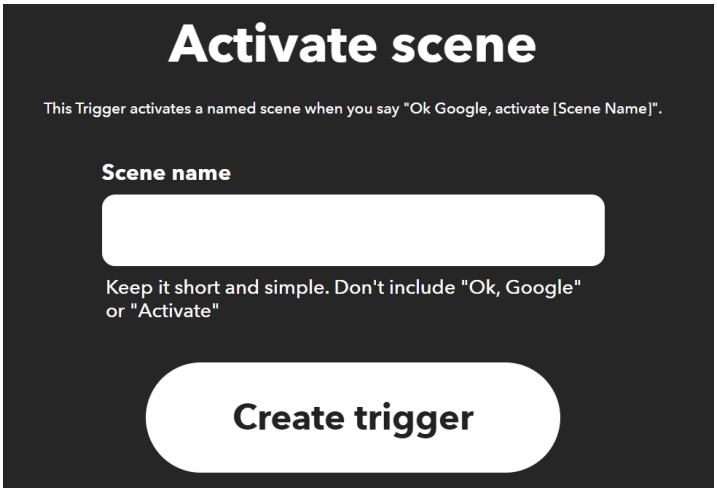
On the this screen you search for google assistant and click on the symbol.
On the next screen you just click on “activate scene” and then on “connect”. So with the following steps we will try to connect the google assistant with Adafruit. We will do that with the help of the “Google Home” app.



Now you need to open the Google Home app on your Android or iOS device. When you click on “Settings” you need to go to “Works with Google” and add FTTT.



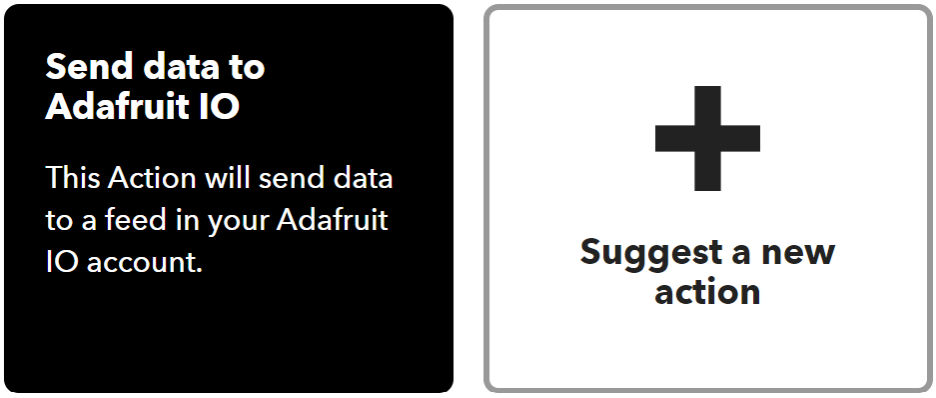
After you have don this you need to go back to IFTTT and start again. So you click again on “Create” and add the Google Assistant V2 to “IF THIS”



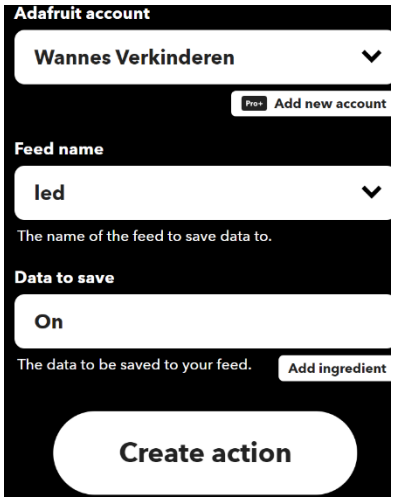
Now when you get this screen you can chose a name for your trigger and click on “Create trigger”.



Now you need to add Adafruit to “Then That” by clicking on “Add”.



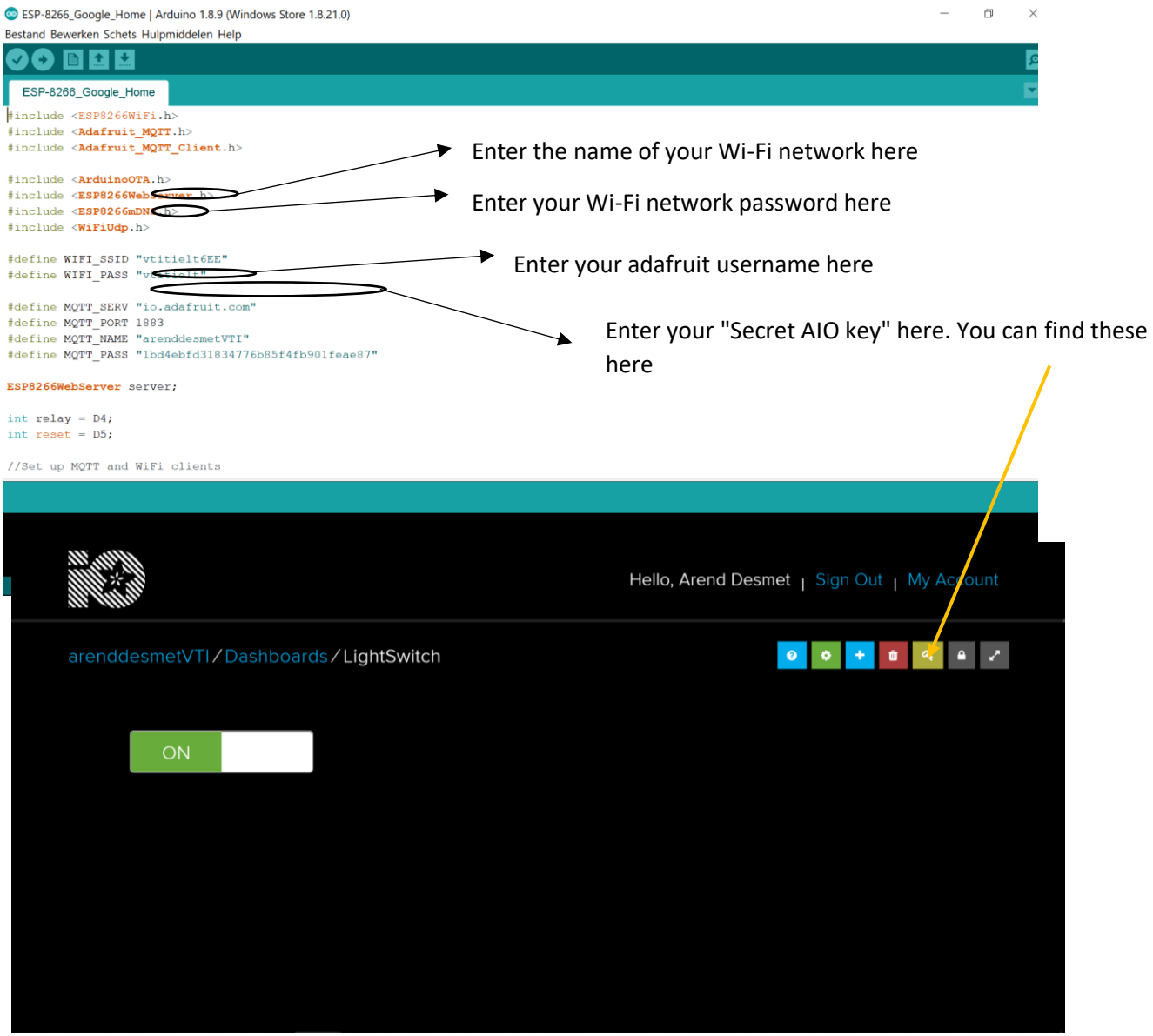
So for the next step you click on the black square and go to the next screen.



Now you make sure that you chose for your feed the same feed that you use in Adafruit by your display. And your data to save is “On”. When everything is correctly done you can create you action.

1.4.2 Program

The program sentence attached. Each line describes exactly what is happening. Some things still need to change. You can find the program basic by the second link that is placed at the start of this document.



First we will try to light up the led that is build in to the esp itself. So we don't need to make a electrical circuit. Now that we know what we need to do it is imported to make sure that your esp can communicate with you computer. We had some problems by ourselves to do that. If you didn't connect Adafruit, IFTTT and the Google Home app correctly you will have problems as well. First I have put the hole program underneath but after this I will explain the most imported parts apart.

Led_Builtin google home.ino

```
#include <WiFi.h>
#include "Adafruit_MQTT.h"
#include "Adafruit_MQTT_Client.h"

//#define Relay1   12
//#define Relay2   13
//#define Relay3   14
//#define Relay4   27
//int LED_BUILTIN = 2;

#define WLAN_SSID       "HoeveLanghof"             // Your SSID
#define WLAN_PASS       "vakantie"        // Your password

/************************ Adafruit.io Setup ********************************/

#define AIO_SERVER      "io.adafruit.com"
#define AIO_SERVERPORT  1883                   // use 8883 for SSL
#define AIO_USERNAME    "KarstenSchramme"            // Replace it with your username
#define AIO_KEY         "aio_vzal18limmvAM8MUGZRhJSNUpAJf"   // Replace with your Project Auth Key

/************ Global State (you don't need to change this!) ******************/

// Create an ESP8266 WiFiClient class to connect to the MQTT server.
WiFiClient client;
// or... use WiFiFlientSecure for SSL
//WiFiClientSecure client;

// Setup the MQTT client class by passing in the WiFi client and MQTT server and login details.
  Serial.print("Connecting to ");
  Serial.println(WLAN_SSID);

  WiFi.begin(WLAN_SSID, WLAN_PASS);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println();

  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());


  // Setup MQTT subscription for onoff feed.
  mqtt.subscribe(&Light1);
 // mqtt.subscribe(&Light3);
 // mqtt.subscribe(&Light2);
 // mqtt.subscribe(&Light4);
}

void loop() {

  MQTT_connect();


  Adafruit_MQTT_Subscribe *subscription;
  while ((subscription = mqtt.readSubscription(20000))) {
    if (subscription == &Light1) {
      Serial.print(F("Got: "));
      Serial.println((char *)Light1.lastread);
      int Light1_State = atoi((char *)Light1.lastread);
     // digitalWrite(Relay1, Light1_State);
      digitalWrite(LED_BUILTIN, Light1_State);

    }
   /* if (subscription == &Light2) {
      Serial.print(F("Got: "));
      Serial.println((char *)Light2.lastread);
      int Light2_State = atoi((char *)Light2.lastread);
      digitalWrite(Relay2, Light2_State);
    }
    if (subscription == &Light3) {
      Serial.print(F("Got: "));
      Serial.println((char *)Light3.lastread);
      int Light3_State = atoi((char *)Light3.lastread);
      digitalWrite(Relay3, Light3_State);
    }
    if (subscription == &Light4) {
      Serial.print(F("Got: "));
      Serial.println((char *)Light4.lastread);
```

```
    int Light4_State = atoi((char *)Light4.lastread);
    digitalWrite(Relay4, Light4_State);

    } */
  }


}

void MQTT_connect() {
  int8_t ret;

  // Stop if already connected.
  if (mqtt.connected()) {
    return;
  }

  Serial.print("Connecting to MQTT... ");

  uint8_t retries = 3;

  while ((ret = mqtt.connect()) != 0) { // connect will return 0 for connected
    Serial.println(mqtt.connectErrorString(ret));
    Serial.println("Retrying MQTT connection in 5 seconds...");
    mqtt.disconnect();
    delay(5000);  // wait 5 seconds
    retries--;
    if (retries == 0) {
      // basically die and wait for WDT to reset me
      while (1);
    }
  }
  Serial.println("MQTT Connected!");

}
```

In the beginning of our program are the libraries named that are included in the program.

```
#include <WiFi.h>
#include "Adafruit_MQTT.h"
#include "Adafruit_MQTT_Client.h"
```

So there are 2 libraries for Adafruit and normally there are some more libraries for Adafruit but we had a problem with putting them in the program so we used the library “Wifi.h” with some more Adafruit libraries included. So we bypassed that problem.

```
#define WLAN_SSID       "HoeveLanghof"             // Your SSID
#define WLAN_PASS       "vakantie"        // Your password
```

Here we needed to define the SSID so we can communicate with esp without any cable.

```
/************************* Adafruit.io Setup *********************************/

#define AIO_SERVER      "io.adafruit.com"
#define AIO_SERVERPORT  1883                   // use 8883 for SSL
#define AIO_USERNAME    "KarstenSchramme"            // Replace it with your username
#define AIO_KEY         "aio_vzal18limmvAM8MUGZRhJSNUpAJf"   // Replace with your Project Auth Key
```

This part is imported for Adafruit. When you define al of this it will connect you Adafruit account to the program so that the program can get information from and to Adafruit.

```
// Setup a feed called 'onoff' for subscribing to changes.
Adafruit_MQTT_Subscribe Light1 = Adafruit_MQTT_Subscribe(&mqtt, AIO_USERNAME"/feeds/LED_BUILTIN");
//Adafruit_MQTT_Subscribe Light1 = Adafruit_MQTT_Subscribe(&mqtt, AIO_USERNAME"/feeds/relay1"); // FeedName
//Adafruit_MQTT_Subscribe Light2 = Adafruit_MQTT_Subscribe(&mqtt, AIO_USERNAME "/feeds/relay2");
//Adafruit_MQTT_Subscribe Light3 = Adafruit_MQTT_Subscribe(&mqtt, AIO_USERNAME "/feeds/Relay3");
//Adafruit_MQTT_Subscribe Light4 = Adafruit_MQTT_Subscribe(&mqtt, AIO_USERNAME "/feeds/Relay4");
```

This is where he will need information of Adafruit. He get every feed that you have in Adafruit in one display and give it a name so it can be used in the program. Because or program is not fully used an we have only one feed we put “\\” for the other lines of program. For the led we use the “Led_BUILTIN” so we don’t need to connect relays or other lights.

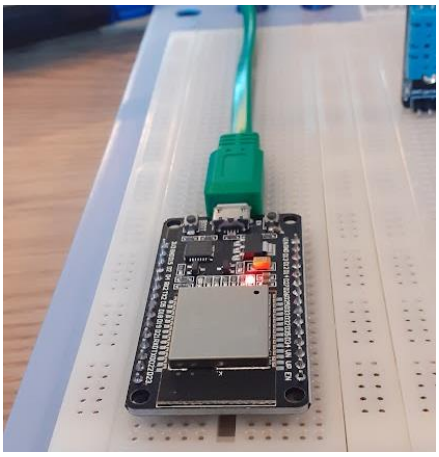
1.4.3 Google Hub

So we worked with a google hub. Normally we would have worked with a Google Nest but we were only able to work with a hub so we needed to work with that. Working with a Google Assistant device is most of the time simple but only if you let it work with other Google devices. From the moment you work with non-Google devices you will have some trouble with connecting the Google Home app and you device.



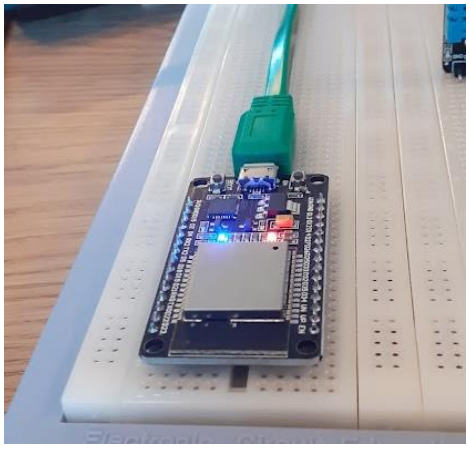


The first picture is a Google Nest and the second picture is a Google Hub. The biggest difference between them is that the Google Hub has a screen where you can see some information like what song is playing or how warm it is outside. But the Google Hub is also a lot bigger and is more expensive.



On this picture you see a red light. This is not the led that is built in the esp. It is a light that light up when the esp has connection with some power. So on this picture is the led that we will activate off.

When we say “hey Google turn on light one” the led will active and you will have a blue led shining.



Here we can see the blue led so there is been an action between the 2 pictures. Now if you want to deactivate the led you need to say “hey Google turn off light one”.

In the next link you can find a video were you can see the led working.

https://photos.google.com/photo/AF1QipOtckFAERMFtRhP-K6ASbVQnFUdsikD5af47AkQ

We went a bit further than that with a bigger light. We needed to work with a relay. It is not so different than wat we did before so I only took a video of it. You can find the video in the link underneath.

https://photos.google.com/photo/AF1QipMC4tD-IUkJLr6Lt-ZTm75Z9LDNdsvAcDSEwKKq

Here in the document it looked that we did this part very fast but that wasn’t in reality we have a few problem connecting the esp and the computer. We couldn’t load any program on the esp even “Blink” didn’t work.

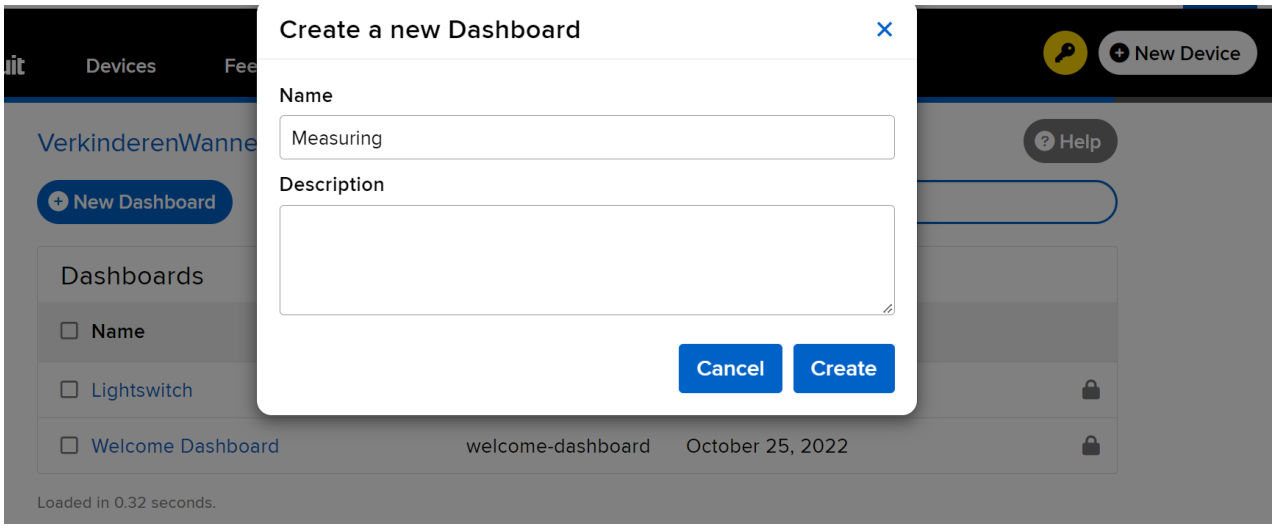
1.5 Part 2: measuring

1.5.1 Setting up Adafruit

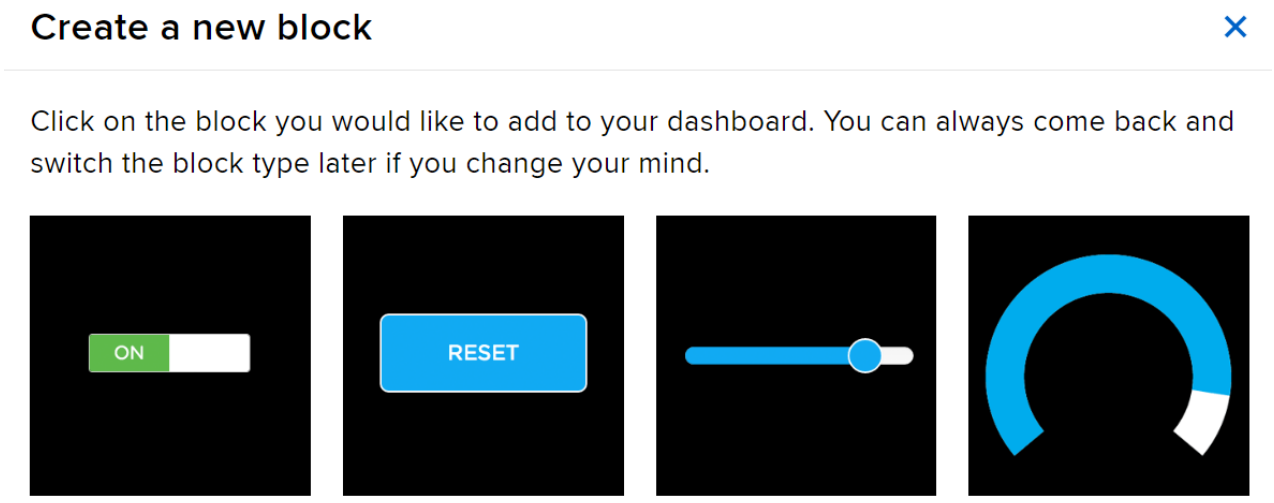
First we gone try to measure the temperature with DHT-11. We have also the DHT-20 and DHT-22 to our disposal but we don't gone you that one because it's not so different than the DHT-11.

So the steps that we need to do is a bit shorter because we have already connected Adafruit to IFTTT ant IFTTT to Google Assistant. So we don't need to do that anymore. And we can also use the same program that we used whit the led but than with some more lines of program.

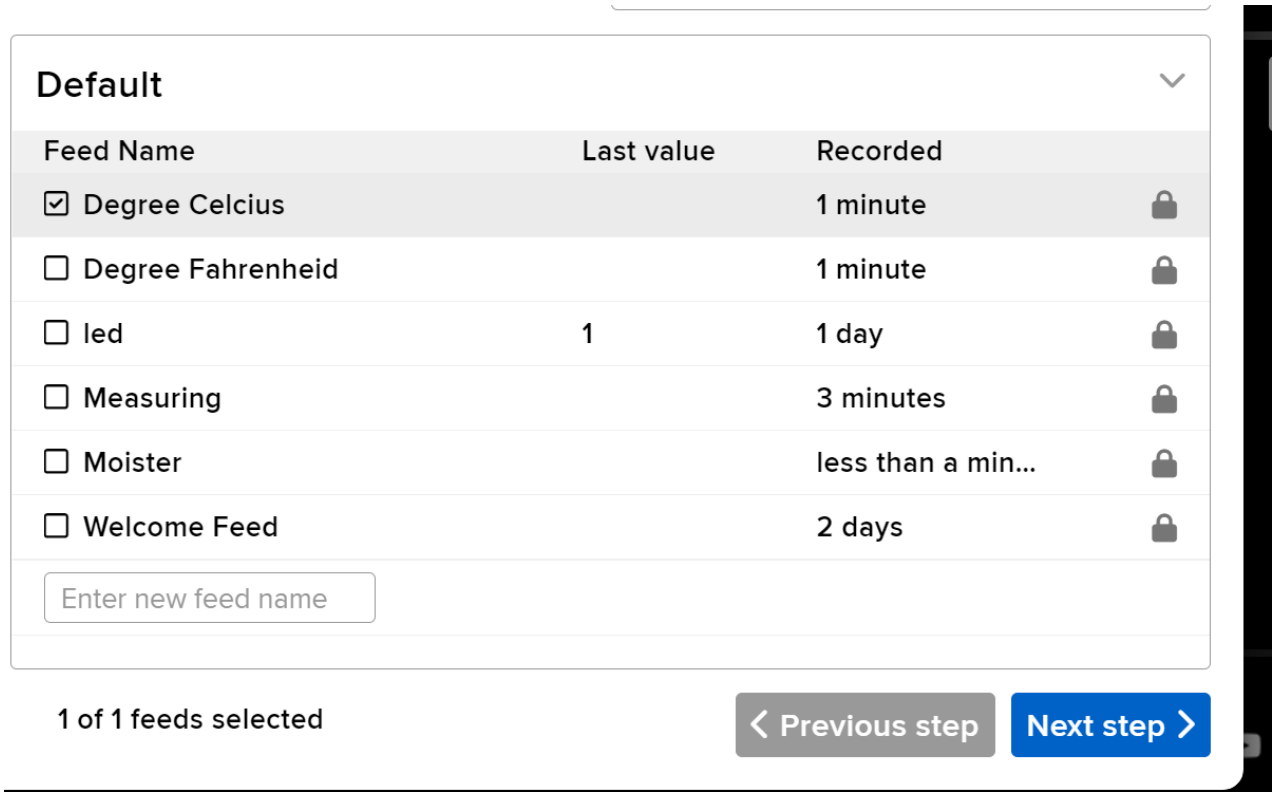
So first you need to make a new dashboard in Adafruit where you can put a new block in. For the new block you need to chose a Gauge. You are gone put the Gauge in a new feed that you need to make.



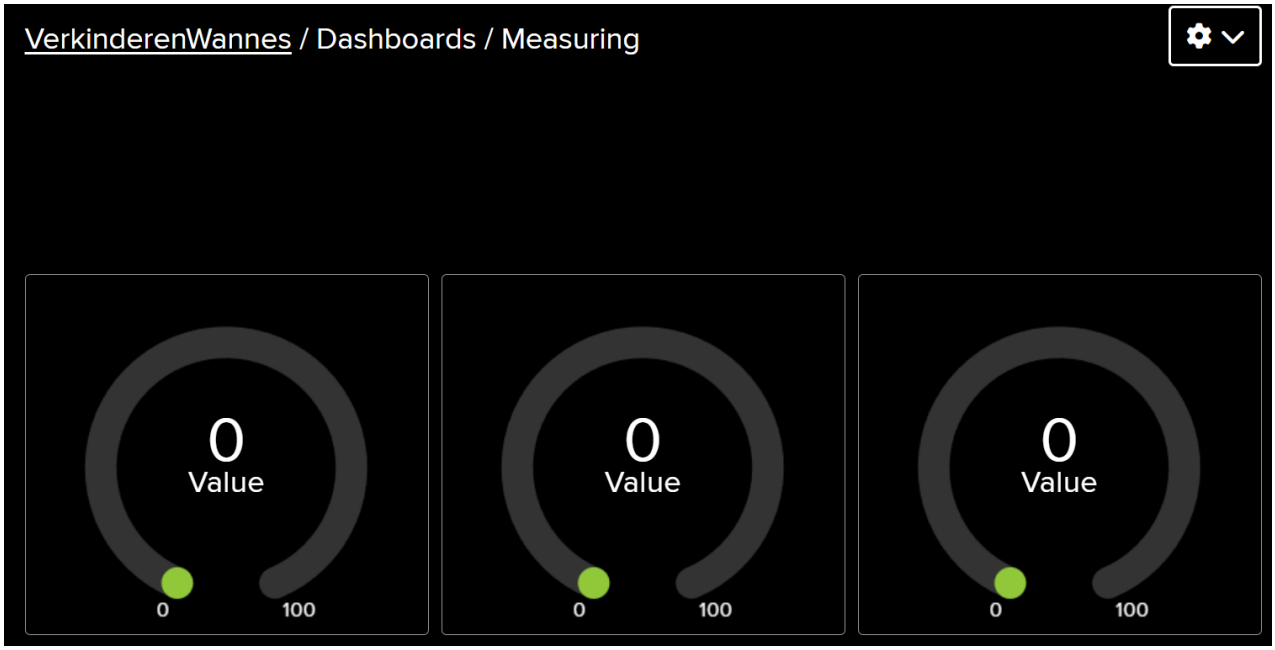
So here we make a new dashboard that we call "Measuring". We made a new dashboard so we can see easily where that al the project that we do with Adafruit are kept.



So as new block we need to take the Gauge.



So after you chose the Gauge as new block you need to make a new feed that wil work with the Gauge. Now you need to do this process three times because we need three Gauges. We need one for degree Celsius, one for degree Fahrenheit and one for the humidity. Every Gauge need to work with a new feed so you can't have more than one block in a feed.



So if you did everything right you will see this on Adafruit. But it is very imported that the work with different feeds.

1.5.2 Program

First I want to say that we didn't wright the program from top to bottom by ourselves. We found a basis program on the internet and we changed a few things so it did what we wanted to do it.

So for now we can't ask to google how warm it is or how much the humidity is because we didn't look to that. We did this because we think that we don't got time for that. We first will make sure that this document is completed and that we have made our presentation. So now we don't gone work with IFTTT and we will just put ore Adafruit in a program so we can see our measurement on the computer.

So first I will put the hole program underneath and then I will explain al the imported step for step.

```
22  #include <Adafruit_Sensor.h>
23  #include <DHT.h>
24  #include <DHT_U.h>
25
26  // pin connected to DH22 data line
27  #define DATA_PIN 2
28
29  // create DHT22 instance
30  DHT_Unified dht(DATA_PIN, DHT11);
31
32  // set up the 'temperature' and 'humidity' feeds
33  AdafruitIO_Feed *temperature = io.feed("temperature");
34  AdafruitIO_Feed *temperature_fahrenheit = io.feed("temperature fahrenheit");
35  AdafruitIO_Feed *humidity = io.feed("humidity");
36
37  void setup() {
38
39    // start the serial connection
40    Serial.begin(115200);
41
42    // wait for serial monitor to open
43    while(! Serial);
44
45    // initialize dht22
46    dht.begin();
47
48    // connect to io.adafruit.com
49    Serial.print("Connecting to Adafruit IO");
50    io.connect();
51
52    // wait for a connection
53    while(io.status() < AIO_CONNECTED) {
54      Serial.print(".");
55      delay(500);
56    }
```

```
57
58    // we are connected
59    Serial.println();
60    Serial.println(io.statusText());
61
62  }
63
64  void loop() {
65
66    // io.run(); is required for all sketches.
67    // it should always be present at the top of your loop
68    // function. it keeps the client connected to
69    // io.adafruit.com, and processes any incoming data.
70    io.run();
71
72    sensors_event_t event;
73    dht.temperature().getEvent(&event);
74
75    float celsius = event.temperature;
76    float fahrenheit = (celsius * 1.8) + 32;
77
78    Serial.print("celsius: ");
79    Serial.print(celsius);
80    Serial.println("C");
81
82    Serial.print("fahrenheit: ");
83    Serial.print(fahrenheit);
84    Serial.println("F");
85
86    // save fahrenheit (or celsius) to Adafruit IO
87    temperature->save(celsius);
88    temperature_fahrenheit->save(fahrenheit);
89
90    dht.humidity().getEvent(&event);
91
92    Serial.print("humidity: ");
93    Serial.print(event.relative_humidity);
94    Serial.println("%");
95
96    // save humidity to Adafruit IO
97    humidity->save(event.relative_humidity);
98
99    // wait 5 seconds (5000 milliseconds == 5 seconds)
100   delay(5000);
101
102 }
103
```

So this is the hole program. This program is a bit shorter than the previous program and that because this program is made for the right amount of measurements and the program for the led was made for more than one led.

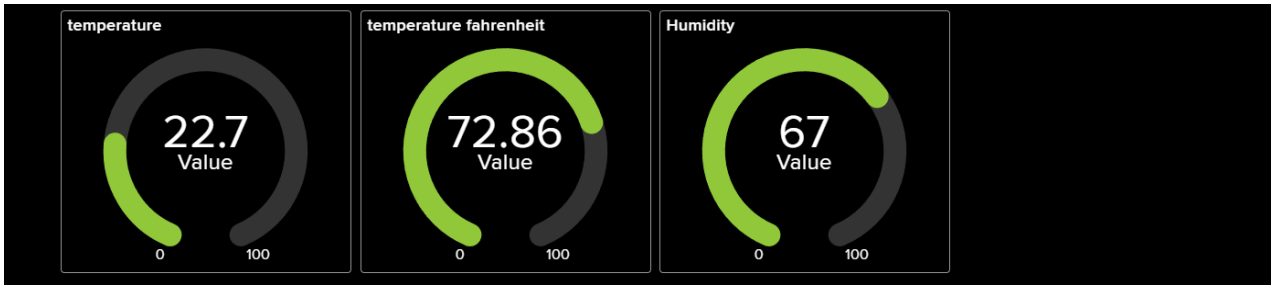
```
22  #include <Adafruit_Sensor.h>
23  #include <DHT.h>
24  #include <DHT_U.h>
```

In the beginning of the program we include the 3 different libraries. This time it wasn't a problem to put the libraries directly in the program. So the first libraries "Adafruit_Sensor.h" is a library that is in the program for the three Gauges that we use. The two other Libraries "DHT.h" and "DHT_U.h" are libraires for the temperature meter itself .

```
26  // pin connected to DH22 data line
27  #define DATA_PIN 2
28
29  // create DHT22 instance
30  DHT_Unified dht(DATA_PIN, DHT11);
31
32  // set up the 'temperature' and 'humidity' feeds
33  AdafruitIO_Feed *temperature = io.feed("temperature");
34  AdafruitIO_Feed *temperature_fahrenheit = io.feed("temperature fahrenheit");
35  AdafruitIO_Feed *humidity = io.feed("humidity");
```

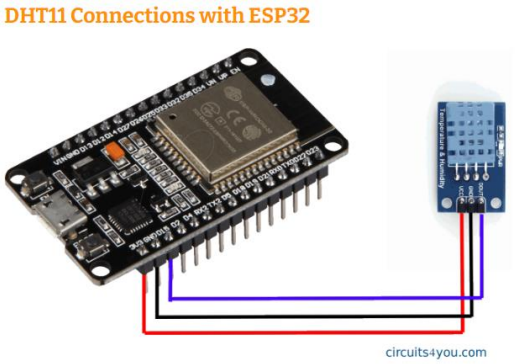
In line 27 we define the data-pin we use to connect our DHT-11 to. And in line 30 we define what for DHT we use. In ore case it is the DHT-11. And from line 33 to line 35 we say which Feed we use by every Gauge. This is why we need three different Feeds for the three Gauges.

The rest of the program is code for translating the data we get from our DHT-11 so we can send it to Adafruit and there is also a part that we send the measerments to Adafruit. So this program is not so difficult because we started with most of the codes that are filled in and than you just need to understand the program and what every part dous.



Here you see a screenshot from Adafruit after you loaded the program in to your esp. It is possible that the symbols are a bit different but that is because you can chouse your own symbol wen you make you blocks on Adafruit.

1.5.3 Electrical circuit



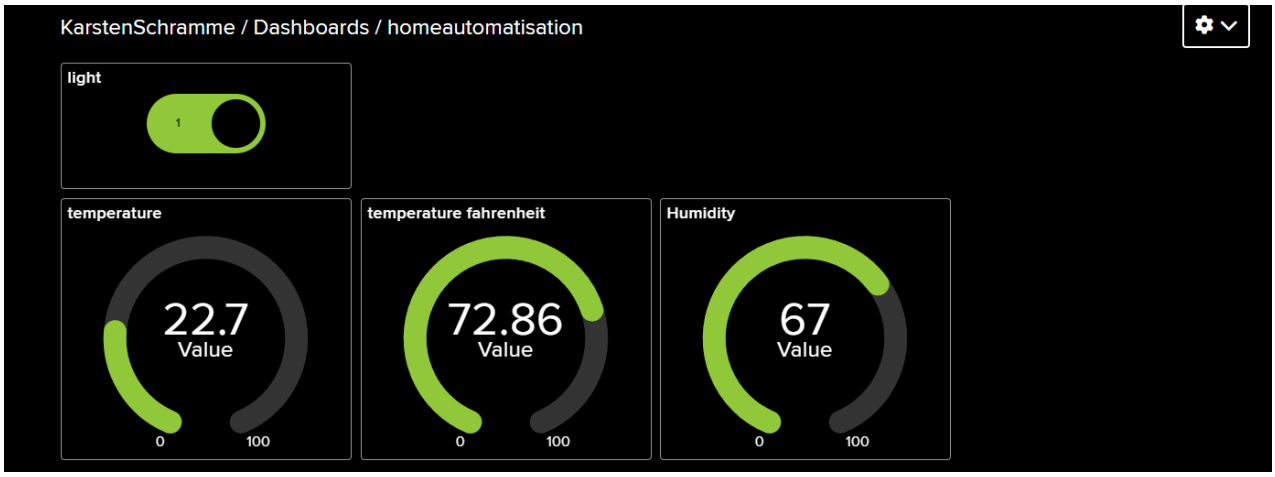
This is the electrical circuit. It's not difficult but you need the ports to put in the program. The DHT-11 is connected to the 3,3V pin and the ground.

1.6 Part 3: Putting together

Now for the last part of our workshop we want to put part 1 and part 2 together. I wanted to do that because we think its better that you can do the 2 different things at the same time. To put the 2 programs together you need to pick some code from the first program and some code from the second and put it together in an new program. For that part it is very imported that you know what every part of you 2 programs does.

1.6.1 Setup Adafruit

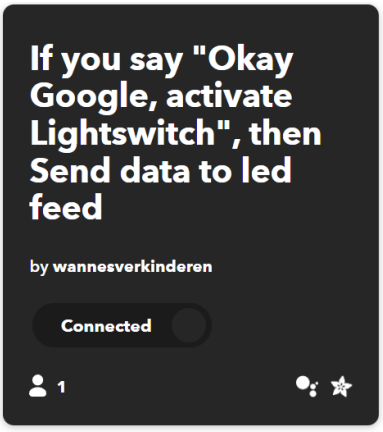
In Adafruit we will put the 2 dashboards together in 1. So first you need to make a new dashboard were you need to make 4 new blocks. You need to make 1 Toggle and 3 Gauges. This time you can use the Feeds from part 1 and part 2 but make sure you now what feed is by what block.



If you did it wright it need to look like this screen above. This is nothing new than the last 2 parts but you just need to combine it.

1.6.2 IFTTT

If you used the feed from part 1 for your Toggle than you can skip this. If you didn't do that than it is imported that you look again to part 1. You need to look at the part of IFTTT and you will see what of action you need to make.



So here you see my applet from the first part. You can see that I used the feed "led". So for this part I also used the feed led for my Toggle. So now I don't need to make a new IFTTT.

1.6.3 Program

First I will put the hole program in once in to the document. Than I will explain it.

```
#include "config.h"

/************************ Example Starts Here *******************************/
#include <Adafruit_Sensor.h>
#include <DHT.h>
#include <DHT_U.h>

#include <WiFi.h>
#include "Adafruit_MQTT.h"
#include "Adafruit_MQTT_Client.h"
// pin connected to DH22 data line
#define DATA_PIN 2           // data pin D2
#define Relay1   18           // licht/ relay has pin D18
// create DHT22 instance
DHT_Unified dht(DATA_PIN, DHT11);

// set up the 'temperature' and 'humidity' feeds
AdafruitIO_Feed *temperature = io.feed("temperature");
AdafruitIO_Feed *temperature_fahrenheit = io.feed("temperature fahrenheit");
AdafruitIO_Feed *humidity = io.feed("humidity");

/************ Global State (you don't need to change this!) ******************/

// Create an ESP8266 WiFiClient class to connect to the MQTT server.
WiFiClient client;
// or... use WiFiFlientSecure for SSL
//WiFiClientSecure client;

// Setup the MQTT client class by passing in the WiFi client and MQTT server and login details.
Adafruit_MQTT_Client mqtt(&client, AIO_SERVER, AIO_SERVERPORT, AIO_USERNAME, AIO_KEY);

/****************************** Feeds ***************************************/


// Setup a feed called 'onoff' for subscribing to changes.
Adafruit_MQTT_Subscribe Light1 = Adafruit_MQTT_Subscribe(&mqtt, AIO_USERNAME"/feeds/LED_BUILTIN");
Adafruit_MQTT_Subscribe Light2 = Adafruit_MQTT_Subscribe(&mqtt, AIO_USERNAME"/feeds/SOS");
//Adafruit_MQTT_Subscribe Light1 = Adafruit_MQTT_Subscribe(&mqtt, AIO_USERNAME"/feeds/relay1"); // FeedName
//Adafruit_MQTT_Subscribe Light2 = Adafruit_MQTT_Subscribe(&mqtt, AIO_USERNAME "/feeds/relay2");
//Adafruit_MQTT_Subscribe Light3 = Adafruit_MQTT_Subscribe(&mqtt, AIO_USERNAME "/feeds/Relay3");
//Adafruit_MQTT_Subscribe Light4 = Adafruit_MQTT_Subscribe(&mqtt, AIO_USERNAME "/feeds/Relay4");



void MQTT_connect();

void setup() {

  // start the serial connection
  Serial.begin(115200);
  pinMode(Relay1, OUTPUT);
  pinMode(LED_BUILTIN, OUTPUT);
  // wait for serial monitor to open

  // Connect to WiFi access point.
  Serial.println(); Serial.println();
  Serial.print("Connecting to ");
  Serial.println(WLAN_SSID);

  WiFi.begin(WLAN_SSID, WLAN_PASS);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println();

  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());


  // Setup MQTT subscription for onoff feed.
  mqtt.subscribe(&Light1);
  mqtt.subscribe(&Light2);


  while(! Serial);

  // initialize dht22
  dht.begin();

  // connect to io.adafruit.com
  Serial.print("Connecting to Adafruit IO");
  io.connect();

  // wait for a connection
  while(io.status() < AIO_CONNECTED) {
    Serial.print(".");
    delay(500);
  }

  // we are connected
  Serial.println();
  Serial.println(io.statusText());
```

```
}

void loop() {

  // io.run(); is required for all sketches.
  // it should always be present at the top of your loop
  // function. it keeps the client connected to
  // io.adafruit.com, and processes any incoming data.
  io.run();
  MQTT_connect();


  Adafruit_MQTT_Subscribe *subscription;
  while ((subscription = mqtt.readSubscription(20000))) {
    if (subscription == &Light1) {
      Serial.print(F("Got: "));
      Serial.println((char *)Light1.lastread);
      int Light1_State = atoi((char *)Light1.lastread);
      Serial.println(Light1_State);
      switch (Light1_State) {
        case 0:
          digitalWrite (Relay1,LOW);
          break;
        case 1:
          digitalWrite (Relay1,HIGH);
          break;
      }
    }
    if (subscription == &Light2) {
      Serial.print(F("Got: "));
      Serial.println((char *)Light2.lastread);
      int Light2_State = atoi((char *)Light2.lastread);
      switch (Light2_State) {
        case 0:
          digitalWrite (Relay1,LOW);
          break;
        case 2:
         if (Light2_State == 2){
          digitalWrite (Relay1,HIGH);
          delay(dit);
          digitalWrite (Relay1,LOW);
          delay(dit);

          digitalWrite (Relay1,HIGH);
          delay(dit);
          digitalWrite (Relay1,LOW);
          delay(dit);

          digitalWrite (Relay1,HIGH);
          delay(dit);
          digitalWrite (Relay1,LOW);
          delay(dah);

          digitalWrite (Relay1,HIGH);
          delay(dah);
          digitalWrite (Relay1,LOW);
          delay(dah);

          digitalWrite (Relay1,HIGH);
          delay(dah);
          digitalWrite (Relay1,LOW);
          delay(dah);

          digitalWrite (Relay1,HIGH);
          delay(dah);
          digitalWrite (Relay1,LOW);
          delay(dah);

          digitalWrite (Relay1,HIGH);
          delay(dit);
          digitalWrite (Relay1,LOW);
          delay(dit);

          digitalWrite (Relay1,HIGH);
          delay(dit);
          digitalWrite (Relay1,LOW);
          delay(dit);

          digitalWrite (Relay1,HIGH);
          delay(dit);
          digitalWrite (Relay1,LOW);
          delay(dit);

          digitalWrite (Relay1,HIGH);
          delay(dit);
          digitalWrite (Relay1,LOW);
          delay(dit);

          delay(longWait);
         }else{
          break;
         }
        break;
      }
    }
}
sensors_event_t event;
dht.temperature().getEvent(&event);

float celsius = event.temperature;
float fahrenheit = (celsius * 1.8) + 32;

Serial.print("celsius: ");
Serial.print(celsius);
Serial.println("C");
```

```
230
231   while ((ret = mqtt.connect()) != 0) { // connect will return 0 for connected
232     Serial.println(mqtt.connectErrorString(ret));
233     Serial.println("Retrying MQTT connection in 5 seconds...");
234     mqtt.disconnect();
235     delay(5000);  // wait 5 seconds
236     retries--;
237     if (retries == 0) {
238       // basically die and wait for WDT to reset me
239       while (1);
240     }
241   }
242   Serial.println("MQTT Connected!");
243
244 }
245
```

This is the hole program. It is very big because we made from 2 programs 1 program. I am not gone to explain the hole program but I gone explain a few thing. The things that I will explain we be the most imported ones.

```
1   #include "config.h"
2
3   /************************ Example Starts Here *******************************/
4   #include <Adafruit_Sensor.h>
5   #include <DHT.h>
6   #include <DHT_U.h>
7
8   #include <WiFi.h>
9   #include "Adafruit_MQTT.h"
10  #include "Adafruit_MQTT_Client.h"
```

This is were al the libraries are included in line 4, 9 and 10 are the libraries for Adafruit. The library “WiFi.h” is a library where we needed to put mor libraries for Adafruit in it. We needed to do that because we had the same problem that we had in the beginning. With doing that the problem is fixed.

```
11  // pin connected to DH22 data line
12  #define DATA_PIN 2            // data pin D2
13  #define Relay1   18            // licht/ relay has pin D18
14  // create DHT22 instance
15  DHT_Unified dht(DATA_PIN, DHT11);
16
17  // set up the 'temperature' and 'humidity' feeds
18  AdafruitIO_Feed *temperature = io.feed("temperature");
19  AdafruitIO_Feed *temperature_fahrenheit = io.feed("temperature fahrenheit");
20  AdafruitIO_Feed *humidity = io.feed("humidity");
21
```

In line 12 can you see that the “Data_pin” is connected to 2. It means that the DHT-11 is connected to D2. For the light we used pin 18. Now in line 18 to 20 are the feeds described. In that feeds are the blocks that we use on Adafruit.

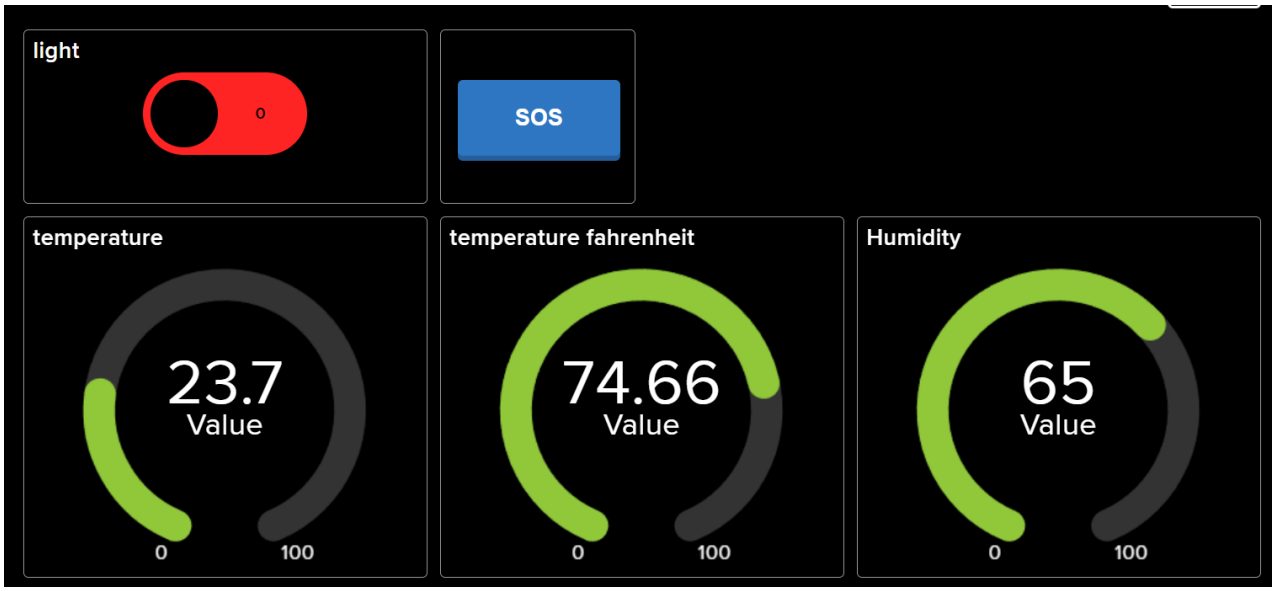
The rest of the program is a really difficult and I now in big lines how it works but I don’t now what every line means. So I don’t gone tell you what it means so I don’t say anything wrong.

1.7 Part 2: measuring

For an extension we made a SOS light. We made an electrical circuit with a relays and a led. I will not say much about it but I will explain you a few basic thing like the electrical circuit and the Adafruit. We had not much time left so it can be that at some point the document can stop.

1.7.1 Adafruit

So we use once again Adafruit. So we just ad a Toggle to the last dashboard. So we can put the SOS light in the program where the led and the DHT-11 also are.



This is wat your dashboard should look like. So you create a now block “Toggle” and let him work in a new feed. We picked the feed “SOS”. If you have an other layout of toggle you don’t need to panic. It is just an outer symbol that we have used for it.

1.7.2 IFTTT