

Projectnummer: 2023-1-PL01-KA220-SCH-000154043



Co-funded by  
the European Union

## **IoT4Schools**

**"Het internet der dingen in het onderwijs brengen middel  
om de uitdagingen van de 21<sup>ste</sup> eeuw aan te gaan"**

**Slimme afvalbakken: hoe kan het afvalbeheer in slimme steden  
worden verbeterd?**

## **Werkblad**

**Auteurs:** Angelika Tefelska, Dariusz Tefelski

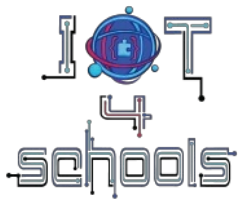
**Organisatie:** Technische Universiteit Warschau, Faculteit Natuurkunde

*Licentie: CC BY-NC 4.0 LEGAL CODE, Naamsvermelding-NietCommercieel 4.0 Internationaal*



Co-funded by  
the European Union

*De steun van de Europese Commissie voor de productie van deze publicatie houdt geen goedkeuring in van de inhoud, die uitsluitend de standpunten van de auteurs weergeeft, en de Commissie kan niet verantwoordelijk worden gehouden voor het gebruik van de informatie die erin is vervat.*



Team: .....

## 1. Tijd voor een brainstormsessie

Wat weet jij over afvalbeheer in de stad? (inzamelmethoden, verwerking, recyclingstatistieken, brandstofverbruikstatistieken voor vuilniswagens) Zoek naar informatie over dit onderwerp.

.....

.....

.....

.....

.....

.....

.....

Hoe kan het huidige afvalbeheersysteem worden verbeterd? Hoe kan het ophalen van afval uit huizen/appartementen worden verbeterd?

.....

.....

.....

.....

.....

..... Als je een slimme prullenbak wilt bouwen, hoe moet deze dan werken?

.....

.....

.....

.....

.....

## 2. Tijd om je eigen slimme afvalbak te ontwerpen

Bedenk een slimme afvallemmer eruit zou moeten zien, die als taak heeft om de mate van volheid van een afvallemmer te meten. Hiervoor gebruiken we een ultrasone afstandssensor die de afstand tot obstakels meet. Waar moet deze sensor in de prullenbak worden geplaatst om informatie over de volheid te verkrijgen? Daarnaast zal het project gebruik maken van drie LED diodes (rood, geel en groen). Waar plaats je deze diodes op de prullenbak zodat de gebruiker weet hoe vol de prullenbak is? Bij welke waarden van volheid gaan de afzonderlijke diodes branden? Is het versturen van gegevens over de mate van volheid van de prullenbak naar de cloud een goede oplossing?

.....

.....

.....

.....

.....

.....

.....

.....

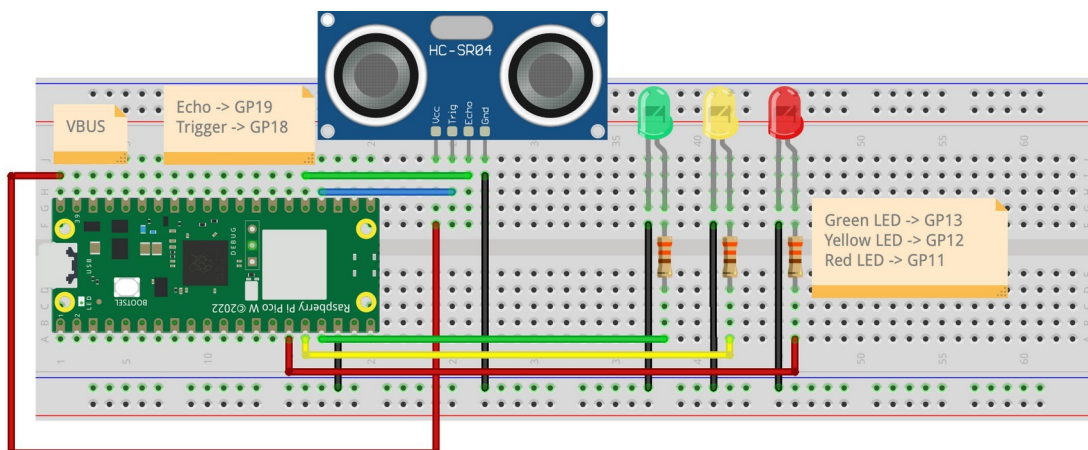
.....

.....

.....

## 3. Tijd voor opwarmactiviteiten

Voordat we verder gaan met het bouwen van een slimme prullenbak, moeten we eerst de elementen leren kennen die we gaan gebruiken. Om dit te doen, doen we twee korte opwarmopdrachten. De eerste gaat over verkeerslichten. De tweede gaat over de werking van de ultrasone afstandssensor. Bouw eerst de elektronische schakeling volgens onderstaande tekening. Deze schakeling wordt gebruikt voor twee opwarmopdrachten.



### 3.1 Eerste opwarmtaken: verkeerslichten

Elk programma geschreven in MicroPython voor het programmeren van de Raspberry Pi Pico ziet er uit:

```
1 import machine
2
3 #instructions that are to be executed only once here,
4 #e.g. configuration, creating variables
5
6 while True:
7     #instructions to be repeated over and over
```

Aan het begin voegen we de machinebibliotheek toe, die basisfuncties bevat om de microcontroller te bedienen. Vervolgens moeten we aan het begin instructies toevoegen die slechts eenmaal moeten worden uitgevoerd, zoals configuratie, het maken van variabelen of het maken van onze eigen functies. Dan hebben we altijd een oneindige lus, waarin we instructies plaatsen die steeds herhaald worden.

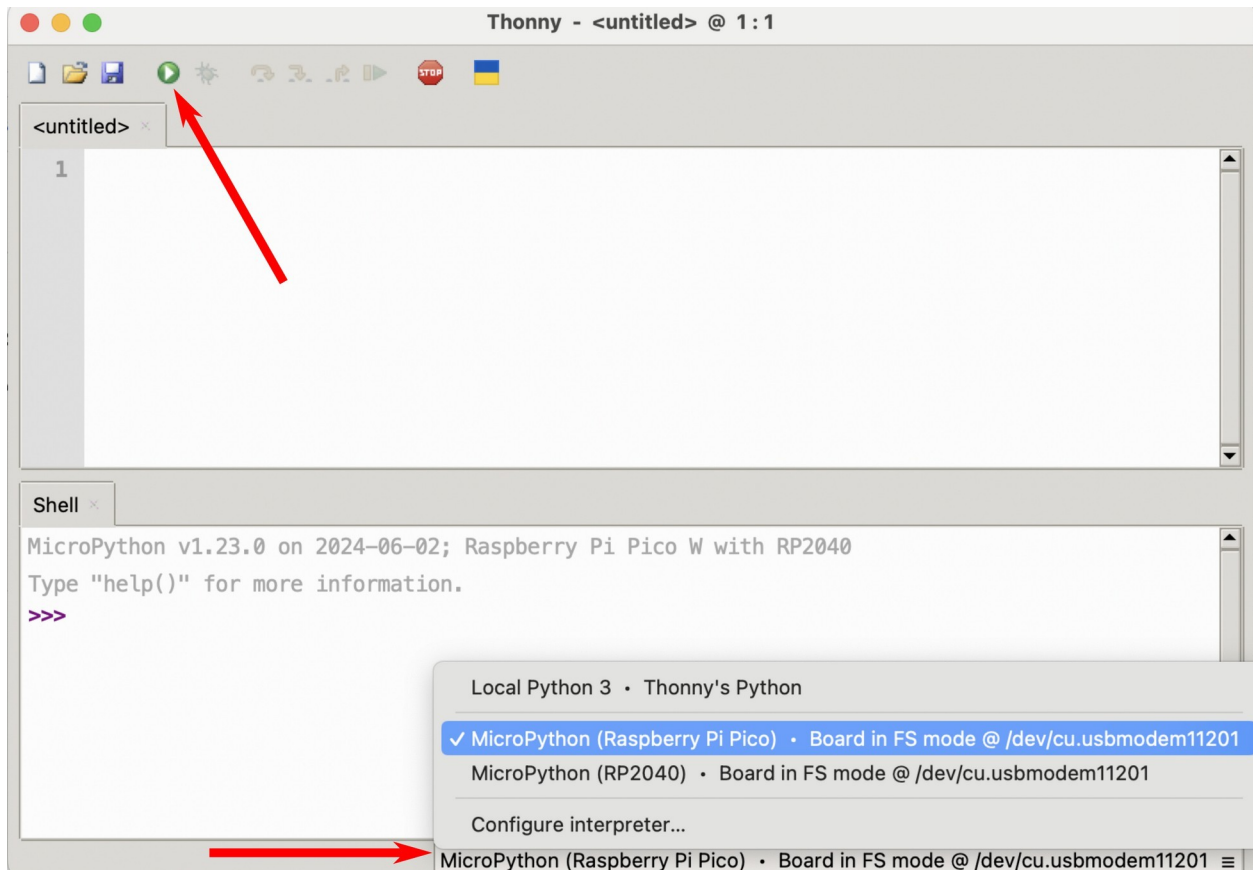
De basisfuncties die nodig zijn om de oefening te voltooien zijn:

- **machine.Pin(pinnummer, machine.Pin.IN of machine.Pin.OUT)** - configureert of een gegeven pin een ingang (machine.Pin.IN) of uitgang (machine.Pin.OUT) moet zijn. Ingangspinnen worden gebruikt als we gegevens willen lezen van een element dat op een pin is aangesloten, bijvoorbeeld als we een knop hebben aangesloten, lezen we of deze is ingedrukt (waarde 1) of niet (waarde 0). Output pinnen worden gebruikt als we een bepaald element dat op een pin is aangesloten willen aansturen, bijvoorbeeld een LED diode, of deze aan (waarde 1) of uit (waarde 0) moet zijn.
- **value(0 of 1)** - functie om de waarde op een gegeven pin in te stellen.
- **utime.sleep(tijd in seconden)** - functie die de programma-uitvoering voor een bepaalde tijd onderbreekt. Om deze functie te gebruiken, moet je de utime-bibliotheek gebruiken: importeer utime.

Dus als we willen dat de LED die is aangesloten op pin GP15 elke 1 seconde knippert, zou de code er als volgt uitzien:

```
1 import machine
2 import utime
3
4 led = machine.Pin(15, machine.Pin.OUT)
5
6 while True:
7     led.value(1)
8     utime.sleep(1)
9     led.value(0)
10    utime.sleep(1)
```

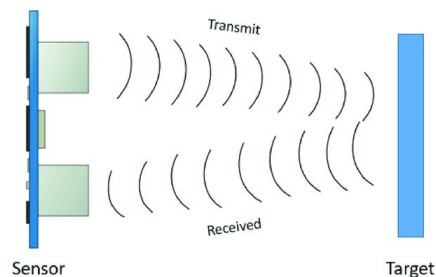
Open de Thonny-editor en selecteer vervolgens "MicroPython (Raspberry Pi Pico)" zoals weergegeven in de afbeelding. Als de verbinding met het bord correct is gemaakt, moet de groene knop Uitvoeren actief zijn (niet grijs weergegeven). Als het grijs is, selecteer dan "MicroPython (Raspberry Pi Pico)" opnieuw.



Probeer nu een programma te maken dat verkeerslichten simuleert.

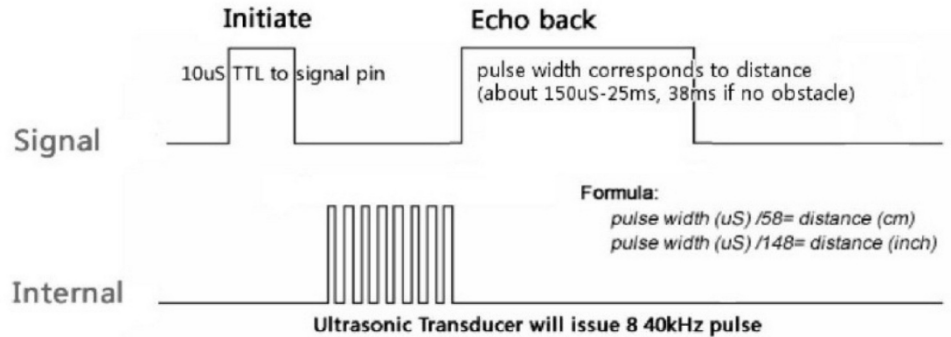
### 3.2 Opwarmopdracht nr. 2: ultrasone afstandssensor

Met de HC-SR04 ultrasone afstandssensor kun je de afstand tot een obstakel meten. Het meetidee wordt weergegeven in de onderstaande figuur:



Bron: [https://www.researchgate.net/figure/Blokdiagram-van-ultrasone-sensor-werkprincipes\\_fig5\\_344385811](https://www.researchgate.net/figure/Blokdiagram-van-ultrasone-sensor-werkprincipes_fig5_344385811)

Om te meten, volgens de sensordocumentatie (zie de afbeelding hieronder), zet u het signaal laag op de trigger, bijvoorbeeld 2  $\mu$ s. Zet dan het signaal 10  $\mu$ s hoog. Om vertragingen in microseconden te genereren, gebruik je de functie: **utime.sleep\_us()**. Stel in de volgende stap het lage signaal in op de trigger.



Bron: <https://www.electronicoscaldas.com/datasheet/HC-SR04.pdf>

Maak een programma dat de gemeten afstand van een ultrasone afstandssensor leest en weergeeft in de terminal. Om dit te doen:

1. Voeg de machine- en utime-bibliotheken toe.
2. Configureer de pin waarop de trigger is aangesloten als een uitgang.
3. Configureer de pin waarop de echo is aangesloten als ingang.
4. Stel in de while-lus de waarde van de triggerpen in op 0.
5. Wacht 2  $\mu$ s (**utime.sleep\_us()** functie).
6. Zet de waarde van de triggerpin op 1.
7. Wacht 10  $\mu$ s.
8. Zet de waarde van de triggerpin op 0.
9. Nu moeten we meten hoe lang het hoge signaal op de echo pin duurt, omdat de duur van het signaal op de echo pin gerelateerd is aan de afstand. Om dit te doen zullen we de functie **utime.ticks\_us()** gebruiken, die meet hoeveel tijd er verstreken is in  $\mu$ s sinds het programma werd gestart. Eerst maken we een while-lus die wordt uitgevoerd zolang het signaal laag is. Daarbinnen plaatsen we de functie **tick\_us()**. Op deze manier krijgen we informatie over wanneer het signaal voor het laatst laag was. Voeg dit codefragment toe aan je programma:

```
terwijl echo.value()==0:
```

```
    signal_off= utime.ticks_us()
```

10. Meet op dezelfde manier wanneer het signaal voor het laatst hoog was en sla de waarde op in een variabele met de naam bijv.

**signaal\_aan.**

11. Het verschil tussen de tijd van het laatste hoog en laag signaal is de duur van de hoge puls op de echo pin. Hoe vertalen we de pulsduur in afstand? In het begin is een

geluidsgolf uitgezonden, die weerkaatst vanaf het object en terugkeert naar de sensor. Daarom legt de golf in de gemeten tijd ( $t$ ) tweemaal de afstand tussen de sensor en het object af en beweegt met een snelheid van ongeveer 340 m/s (de geluidssnelheid in lucht). Daarom kunnen we de vergelijking voor de snelheid schrijven:

$$v = \frac{2d}{t}$$

$$d = v \frac{t}{2} = 0,034 \frac{\text{cm}}{\mu\text{s}} \frac{t}{2} \approx \frac{t [\mu\text{s}]}{58}$$

Daarom moet de verkregen pulsduur worden gedeeld door 58 om de afstand in centimeters te krijgen. Voeg de volgende regels toe aan het programma:

```
diff = signaal_aan-signaal_uit
afstand = diff/58,0
print("Afstand="+str(afstand))
```

Nu kun je de werking van het programma testen. Als je moeite hebt met het lezen van gegevens omdat ze te snel worden weergegeven, kun je een kleine vertraging toevoegen aan de afdrukfunctie.

## 4. Slimme afvalbak - niveau 1

Maak nu een programma waarin je de vaardigheden gaat gebruiken die je hebt opgedaan tijdens de opwarmtaken en meet de vulgraad van de afvalbak. Geef de vulgraad weer op de LED's volgens de percentagebereiken van de afvalbakbezetting die je hebt aangenomen in het stadium van het ontwerpen van de afvalbak.

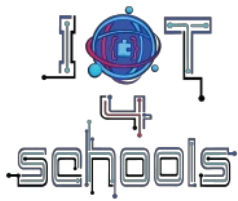
Maak een afvalbak van een ongebruikte verzend-/schoenendoos en test hoe je afvalbak werkt.

## 5. Slimme afvalbak - niveau 2

Laten we nu de prullenbak nog slimmer maken. Om dit te doen, sturen we gegevens naar de cloud over de volheid van de prullenbak en de locatie ervan. Deze gegevens kunnen later worden gebruikt om een algoritme te creëren dat een route voor de vuilniswagen ontwikkelt om alleen afval op te halen op die locaties waar de vuilnisbakken vol zijn, waardoor de ecologische voetafdruk wordt geminimaliseerd. Hiervoor zullen we de Adafruit IO cloud gebruiken.

Eerst moet je een gratis account aanmaken op <https://io.adafruit.com>. Vervolgens wil je twee gegevens naar de cloud sturen: de vulling van de afvalcontainer en de locatie. Om dit te doen, moet je twee feeds maken. Feeds zijn objecten die gegevens opslaan. Om een feed te maken, ga je naar het tabblad "Feed" en selecteer je de knop "Nieuwe feed".





Projectnummer: 2023-1-PL01-KA220-SCH-000154043



Co-funded by  
the European Union

Vervolgens verschijnt er een venster waarin je de naam van de feed moet invoeren, bijvoorbeeld Vullen of Locatie.

**Create a new Feed** ✕

**Name**  
  
Maximum length: 128 characters. Used: 0

**Description**

Cancel Create

Maak twee feeds aan. Zodra je dat hebt gedaan, zou je de feeds die je hebt gemaakt op je pagina moeten zien, zoals in de schermafbeelding hieronder:

[Shop](#) [Learn](#) [Blog](#) [Forums](#) **IO** [LIVE!](#) [AdaBox](#) Hi, Angelika Tefelska | Account ▾ 0

[Devices](#) [Feeds](#) [Dashboards](#) [Actions](#) [Power-Ups](#) [New Device](#)

[angtef / Feeds](#) Help

+ New Feed + New Group

Default + ⋮

Feed Name	Key	Last value	Recorded
<input type="checkbox"/> <a href="#">Filling</a>	filling-the-waste-bin	71.62069	about 5 hours ago
<input type="checkbox"/> <a href="#">Location</a>	location	52.2297,21.0122	about 6 hours ago

De volgende stap is het maken van een dashboard. Selecteer hiervoor het tabblad "Dashboards" en vervolgens "Nieuw Dashboard":

[Shop](#) [Learn](#) [Blog](#) [Forums](#) **IO** [LIVE!](#) [AdaBox](#) Hi, Angelika Tefelska | Account ▾ 0

[Devices](#) [Feeds](#) **[Dashboards](#)** [Actions](#) [Power-Ups](#) [New Device](#)

[angtef / Dashboards](#) Help

+ New Dashboard

Er verschijnt een venster waarin je de naam van het geselecteerde dashboard moet invoeren. Selecteer vervolgens aan de rechterkant het symbool Instellingen en kies "Nieuw blok maken".

[Shop](#) [Learn](#) [Blog](#) [Forums](#) **IO** [LIVE!](#) [AdaBox](#) Account ▾ 0

[Devices](#) [Feeds](#) [Dashboards](#) [Actions](#) [Power-Ups](#) [New Device](#)

[angtef / Dashboards / Test](#)

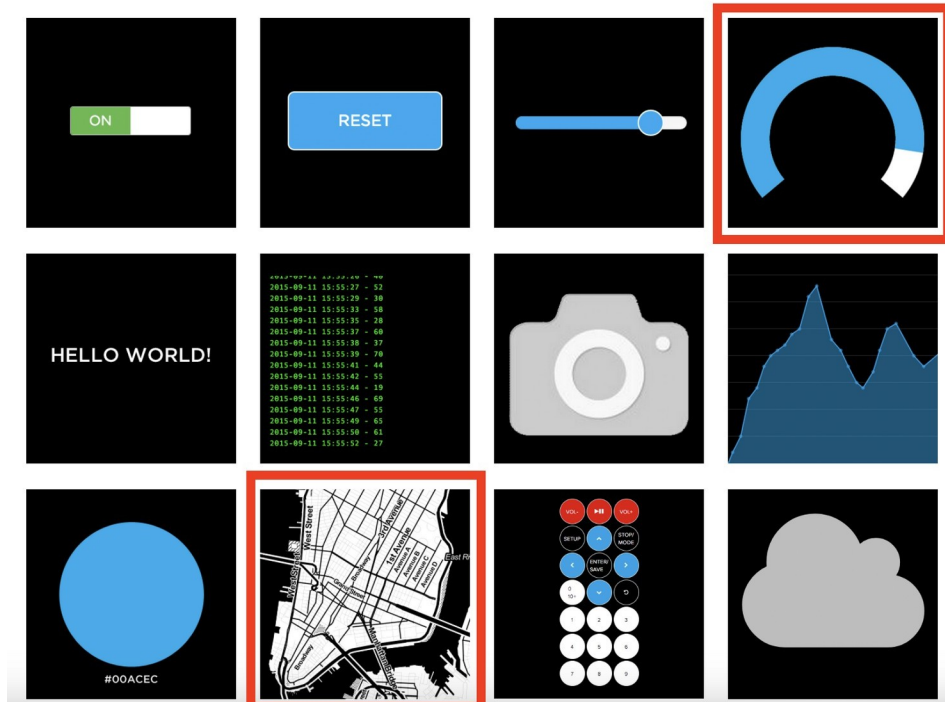


Adafruit IO heeft verschillende blokken beschikbaar om gegevens weer te geven (tekstvakken, meters, grafieken, kaarten, enz.). In ons geval kiezen we voor een meter en een kaart:

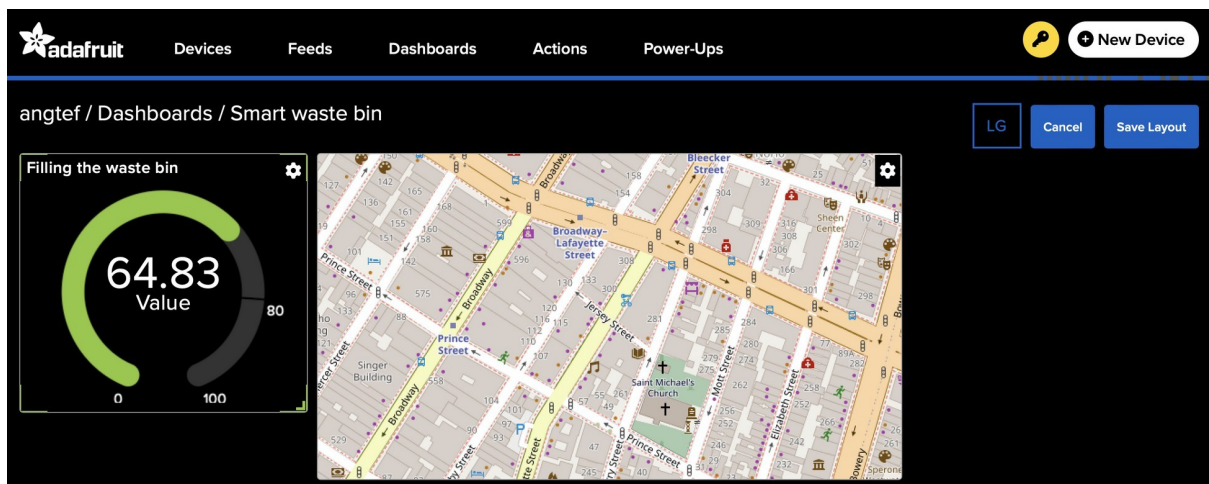
### Create a new block



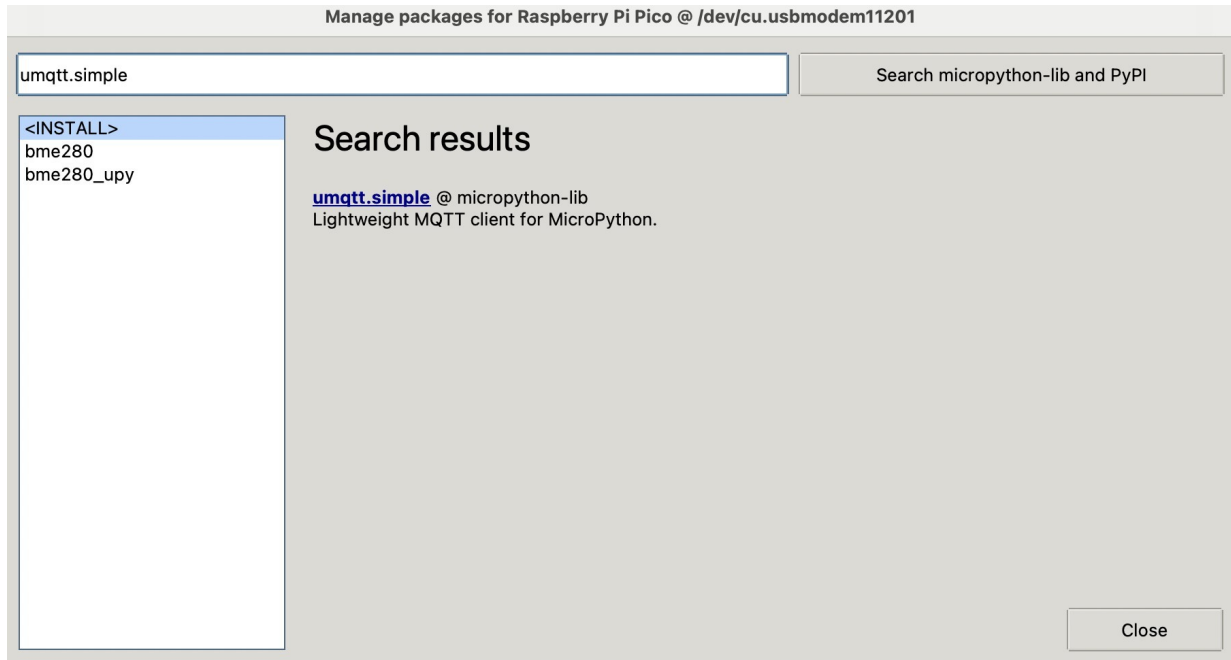
Click on the block you would like to add to your dashboard. You can always come back and switch the block type later if you change your mind.



Vervolgens verschijnt er een venster waarin wordt gevraagd met welke feed we het blok willen verbinden. Selecteer de feed die de afvalbakvulling definieert naar de indicator en de feed die de locatie definieert naar de kaart. Selecteer vervolgens weer het instellingenpictogram en kies "Lay-out bewerken". Rangschik de blokken naar wens op het scherm. Je kunt de blokken ook vergroten en verkleinen. Een voorbeeld van het uiterlijk zie je in de schermafbeelding hieronder:



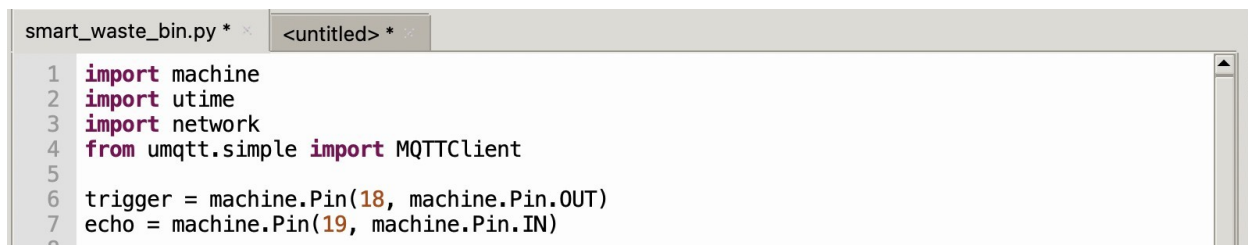
Laten we nu naar de Thonny editor gaan en de **umqtt.simple** bibliotheek installeren. Selecteer hiervoor "Tools" en vervolgens "Manage packages". Er verschijnt een venster waarin je de naam van de bibliotheek die je wilt installeren moet invoeren, in dit geval **umqtt.simple**:



Als je op de naam van de gevonden umqtt.simple bibliotheek klikt, wordt een venster met bibliotheekgegevens geopend en kun je de bibliotheek installeren door op de knop Installeren te klikken.

Nu kunnen we teruggaan naar onze code en de stukken toevoegen die nodig zijn om gegevens naar de cloud te sturen. Volg deze stappen om dit te doen:

1. Voeg de benodigde bibliotheken toe (netwerkbibliotheek en importeer het symbool MQTTClient uit de bibliotheek umqtt.simple):



2. Voeg een stukje code toe waarmee je verbinding kunt maken met je WIFI. Hier moet je de regels 15-18 invullen. Voer eerst de naam van je WIFI in en daarna het wachtwoord.

```
smart_waste_bin.py backup.py
1 import machine
2 import utime
3 import network
4 from umqtt.simple import MQTTClient
5
6 trigger = machine.Pin(18, machine.Pin.OUT)
7 echo = machine.Pin(19, machine.Pin.IN)
8
9 led_green = machine.Pin(13, machine.Pin.OUT)
10 led_yellow = machine.Pin(12, machine.Pin.OUT)
11 led_red = machine.Pin(11, machine.Pin.OUT)
12
13 depth = 100
14
15 WIFI_SSID = "your_wifi_name"
16 WIFI_PASSWORD = "your_wifi_password"
17 ADAFRUIT_IO_USERNAME = "username"
18 ADAFRUIT_IO_KEY = "key"
19
20 def connect_wifi():
21     wlan = network.WLAN(network.STA_IF)
22     wlan.active(True)
23     wlan.connect(WIFI_SSID, WIFI_PASSWORD)
24     while not wlan.isconnected():
25         print("Connecting to Wi-Fi...")
26         utime.sleep(1)
27     print("Connected to Wi-Fi:", wlan.ifconfig())
28
29 connect_wifi()
30
31 while True:
```

De laatste twee parameters zijn de gegevens van Adafruit IO. Ga terug naar de Adafruit website en klik op het sleutelsymbool. Wanneer je dit doet, verschijnen je login en sleutel. Kopieer deze gegevens naar het programma op regel 17-18:

### YOUR ADAFRUIT IO KEY


Your Adafruit IO Key should be kept in a safe place and treated with the same care as your Adafruit username and password. People who have access to your Adafruit IO Key can view all of your data, create new feeds for your account, and manipulate your active feeds.

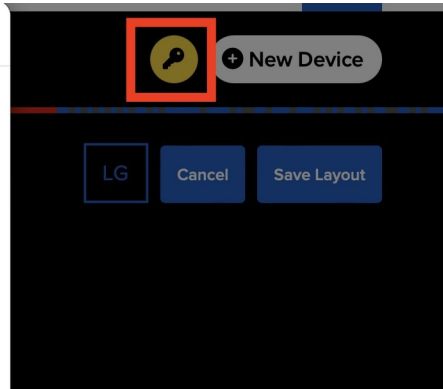
If you need to regenerate a new Adafruit IO Key, all of your existing programs and scripts will need to be manually changed to the new key.

Username

Active Key

REGENERATE KEY





3. Nu gebruiken we MQTT (Message Queuing Telemetry Transport), een lichtgewicht communicatieprotocol gebaseerd op het publish/subscribe model. Het is speciaal ontworpen voor het verzenden van gegevens in omgevingen met beperkte bronnen, zoals IoT-apparaten (Internet of Things). Om dit te doen, gebruikt u de onderstaande code, waarbij u alleen de feednamen in regel 33-34 wijzigt. Het voorbeeld gebruikt feeds met de namen: "Vullen" en "Locatie". Vervang ze door je eigen feeds. Vergeet niet om /csv te laten staan in het geval van Locatie, omdat je bij het gebruik van kaarten gegevens in csv-indeling moet aanleveren.

```
smart_waste_bin.py * x backup.py
13 depth = 100
14
15 WIFI_SSID = "your_wifi_name"
16 WIFI_PASSWORD = "your_wifi_password"
17 ADAFRUIT_IO_USERNAME = "username"
18 ADAFRUIT_IO_KEY = "key"
19
20 def connect_wifi():
21     wlan = network.WLAN(network.STA_IF)
22     wlan.active(True)
23     wlan.connect(WIFI_SSID, WIFI_PASSWORD)
24     while not wlan.isconnected():
25         print("Connecting to Wi-Fi...")
26         utime.sleep(1)
27     print("Connected to Wi-Fi:", wlan.ifconfig())
28
29 ADAFRUIT_IO_SERVER = "io.adafruit.com"
30 ADAFRUIT_IO_PORT = 1883 # Port MQTT
31 CLIENT_ID = "raspberrypi_pico"
32
33 FEED_FILL_LEVEL = "angtef/feeds/Filling"
34 FEED_LOCATION = "angtef/feeds/Location/csv"
35
36 client = MQTTClient(CLIENT_ID, ADAFRUIT_IO_SERVER, ADAFRUIT_IO_PORT, ADAFRUIT_IO_USERNAME, ADAFRUIT_IO_KEY)
37
38 def connect_mqtt():
39     client.connect()
40     print("Connected to Adafruit IO!")
41
42 connect_wifi()
43 connect_mqtt()
```

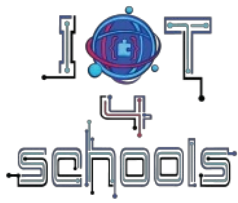
4. Laten we nu functies toevoegen om gegevens naar de cloud te sturen:

```
38 def connect_mqtt():
39     client.connect()
40     print("Connected to Adafruit IO!")
41
42 connect_wifi()
43 connect_mqtt()
44
45 def send_data(Filling, Location):
46     client.publish(FEED_FILL_LEVEL, str(Filling))
47     print("Fill level sent:"+str(Filling))
48     client.publish(FEED_LOCATION, Location)
49     print("Location sent:"+str(Location))
50
```

De laatste stap is om de gemeten bezetting en locatie door te geven aan de functie in de while-lus na je code. Je kunt je locatie bijvoorbeeld uitlezen van Google maps en deze vervangen door de variabele "location".

```
56 while True:
57     #your code
58
59     #value, latitude, longitude, altitude
60     location = "0, 52.2297,20.0122,0"
61     #fill_level - this is the measured fullness of the bin,
62     #correct the variable name to the one you used in your code
63     send_data(fill_level, location)
64
```

Test de slimme prullenbak. Vergeet niet om terug te keren naar de Adafruit IO-pagina op het dashboard dat je hebt gemaakt om te zien of je gegevens correct naar de cloud verstuurt.



## 6. : nadenken over functionaliteit en mogelijke verbeteringen

Denkt u dat het gebruik van slimme vuilnisbakken de afvalinzameling kan verbeteren en de ecologische voetafdruk kan verkleinen? In het project hebben we de route van de vuilniswagens niet geoptimaliseerd, maar je kunt je voorstellen dat er een systeem is dat gegevens uit de cloud leest van elke vuilnisbak en een optimale route voorbereidt. Als je geïnteresseerd bent in hoe je zo'n algoritme maakt, kun je kennismaken met de programma's van de website: <https://colab.research.google.com/drive/1aOq9jRh6c6fhaw1ahe0a1-yKVdMnO613?usp=sharing%2F>.

.....

.....

.....

.....

.....

Welke andere wijzigingen kunnen worden aangebracht om slimme bakken nog efficiënter te laten werken?

.....

.....

.....

.....

.....