

Project number: 2023-1-PL01-KA220-SCH-000154043



Co-funded by
the European Union

IoT4Schools

**“Φέρνοντας το Διαδίκτυο των Πραγμάτων στη σχολική
εκπαίδευση ως εργαλείο για την αντιμετώπιση των
προκλήσεων του 21^{ου} αιώνα”**

**Έξυπνοι κάδοι απορριμμάτων: πώς να βελτιώσουμε τη
διαχείριση απορριμμάτων στις έξυπνες πόλεις;**

Φύλλο Έργου

Συγγραφείς: Angelika Tefelska, Dariusz Tefelski

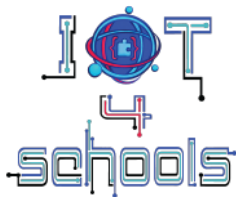
Οργανισμός: Warsaw University of Technology, Faculty of Physics

Άδεια: CC BY-NC 4.0 LEGAL CODE, Attribution-NonCommercial 4.0 International



Co-funded by
the European Union

Η υποστήριξη της Ευρωπαϊκής Επιτροπής για την παραγωγή αυτής της δημοσίευσης δεν συνιστά έγκριση του περιεχομένου, το οποίο αντικατοπτρίζει μόνο τις απόψεις των συγγραφέων και η Επιτροπή δεν μπορεί να θεωρηθεί υπεύθυνη για οποιαδήποτε χρήση των πληροφοριών που περιέχονται σε αυτήν.



Ομάδα:

1. Ωρα για μια συνεδρία καταιγισμού ιδεών

Τι γνωρίζετε για τη διαχείριση των απορριμμάτων στην πόλη; (μέθοδοι συλλογής, επεξεργασία, στατιστικά στοιχεία ανακύκλωσης, στατιστικά στοιχεία κατανάλωσης καυσίμου για απορριματοφόρα)
Αναζήτηση πληροφοριών για αυτό το θέμα:

.....

.....

.....

.....

.....

.....

.....

.....

Πώς μπορεί να βελτιωθεί το τρέχον σύστημα διαχείρισης απορριμμάτων; Πώς μπορεί να βελτιωθεί η συλλογή απορριμμάτων από σπίτια/διαμερίσματα;

.....

.....

.....

.....

.....

.....

.....

Αν θέλετε να φτιάξετε έναν έξυπνο κάδο απορριμμάτων, πώς πρέπει να λειτουργεί;

.....

.....

.....

.....

.....

.....

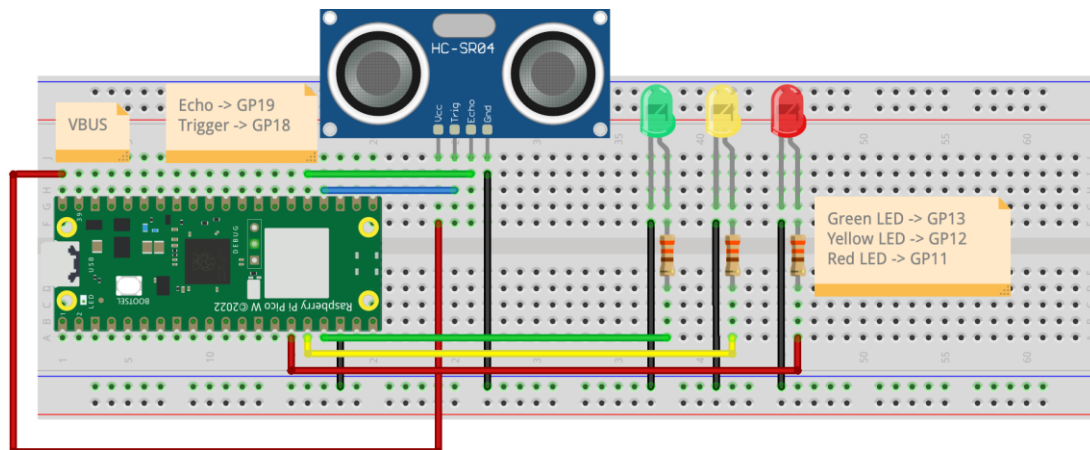
2. Ώρα για να σχεδιάσετε το δικό σας έξυπνο κάδο απορριμμάτων

Σκεφτείτε πώς θα πρέπει να είναι ένας έξυπνος κάδος απορριμμάτων, του οποίου η αποστολή θα είναι να μετρήσει το επίπεδο πληρότητας ενός κάδου απορριμμάτων. Για το σκοπό αυτό, θα χρησιμοποιήσουμε έναν αισθητήρα απόστασης υπερήχων που μετρά την απόσταση από τα εμπόδια. Πού πρέπει να τοποθετηθεί αυτός ο αισθητήρας στον κάδο απορριμμάτων για να ληφθούν πληροφορίες σχετικά με την πληρότητα; Επιπλέον, στο έργο θα χρησιμοποιηθούν τρεις δίοδοι LED (κόκκινο, κίτρινο και πράσινο). Πού θα τοποθετήσετε αυτές τις διόδους στον κάδο απορριμμάτων για να γνωρίζει ο χρήστης πόσο γεμάτος είναι ο κάδος; Σε ποιες τιμές πληρότητας κάδου θα ανάβουν οι μεμονωμένες διόδους; Είναι καλή λύση η αποστολή δεδομένων σχετικά με το επίπεδο πληρότητας του κάδου απορριμμάτων στο cloud;

[illegible]

3. Ώρα για μερικές δραστηριότητες προθέρμανσης

Πριν προχωρήσουμε στην κατασκευή ενός έξυπνου κάδου απορριμμάτων, ας γνωρίσουμε τα στοιχεία που θα χρησιμοποιήσουμε. Για να γίνει αυτό, θα κάνουμε δύο σύντομες εργασίες προθέρμανσης. Η πρώτη αφορά τα φανάρια. Η δεύτερη αφορά τη λειτουργία του αισθητήρα απόστασης υπερήχων. Αρχικά, κατασκευάστε το ηλεκτρονικό κύκλωμα σύμφωνα με το παρακάτω σχέδιο. Αυτό το κύκλωμα θα χρησιμοποιηθεί για δύο εργασίες προθέρμανσης.



3.1 Πρώτη εργασία προθέρμανσης: τα φανάρια

Κάθε πρόγραμμα γραμμένο σε MicroPython για τον προγραμματισμό του Raspberry Pi Pico μοιάζει με αυτό:

```
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
```

Στην αρχή προσθέτουμε τη βιβλιοθήκη μηχανών, η οποία περιέχει βασικές λειτουργίες για τη λειτουργία του μικροελεγκτή. Στη συνέχεια, πρέπει να προσθέσουμε εντολές που πρόκειται να εκτελεστούν μόνο μία φορά στην αρχή, όπως διαμόρφωση, δημιουργία μεταβλητών ή δημιουργία δικών μας συναρτήσεων. Τότε έχουμε πάντα έναν ατέρμονο βρόχο, στον οποίο τοποθετούμε εντολές που θα επαναλαμβάνονται ξανά και ξανά.

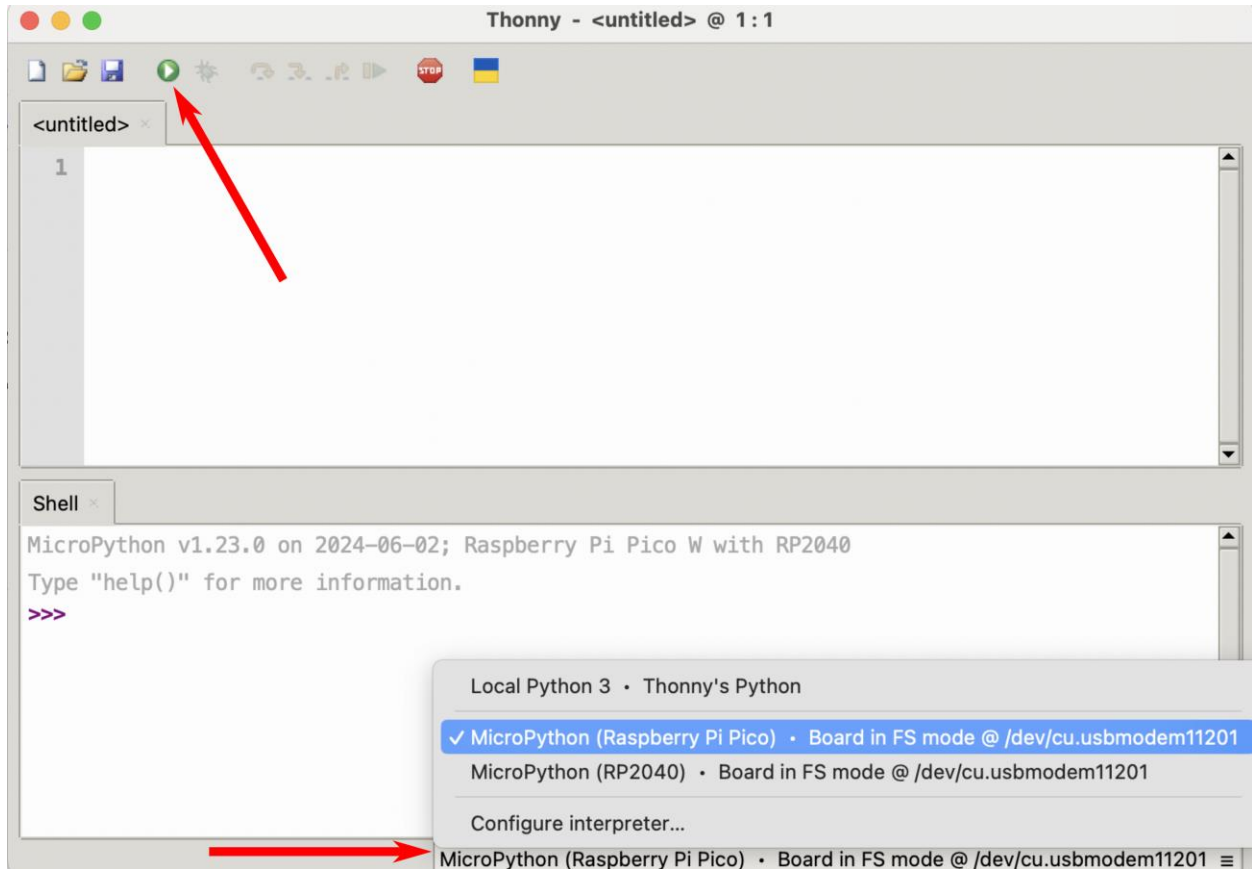
Οι βασικές λειτουργίες που είναι απαραίτητες για την ολοκλήρωση της άσκησης είναι:

- **machine.Pin (pin number, machine.Pin.IN ή machine.Pin.OUT)** - ρυθμίζει εάν μια συγκεκριμένη ακίδα θα πρέπει να είναι είσοδος (machine.Pin.IN) ή έξοδος (machine.Pin.OUT). Οι ακίδες εισόδου χρησιμοποιούνται όταν θέλουμε να διαβάσουμε δεδομένα από ένα στοιχείο συνδεδεμένο σε μια ακίδα, π.χ. όταν έχουμε συνδεδεμένο κουμπί, διαβάζουμε αν πατήθηκε (τιμή 1) ή όχι (τιμή 0). Οι ακίδες εξόδου χρησιμοποιούνται όταν θέλουμε να ελέγξουμε ένα συγκεκριμένο στοιχείο συνδεδεμένο σε έναν ακροδέκτη, π.χ. μια δίοδο LED, είτε θα πρέπει να είναι ενεργοποιημένη (τιμή 1) είτε απενεργοποιημένη (τιμή 0).
- **value(0 ή 1)** - λειτουργία για τη ρύθμιση της τιμής σε μια συγκεκριμένη ακίδα.
- **utime.sleep (χρόνος σε δευτερόλεπτα)** - λειτουργία που αναστέλλει την εκτέλεση του προγράμματος για μια δεδομένη χρονική στιγμή. Για να το χρησιμοποιήσετε, πρέπει να συμπεριλάβετε τη βιβλιοθήκη utime: import utime.

Έτσι, αν θέλαμε η λυχνία LED που είναι συνδεδεμένη στην ακίδα GP15 να αναβοσβήνει κάθε 1 δευτερόλεπτο, ο κώδικας θα έμοιαζε με αυτό:

```
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)
```

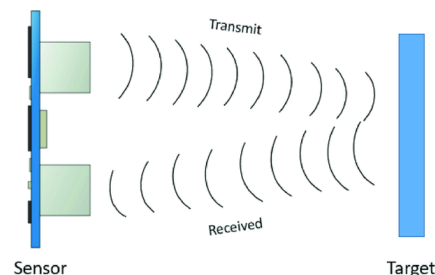
Ανοίξτε το πρόγραμμα επεξεργασίας Thonny και, στη συνέχεια, επιλέξτε "MicroPython (Raspberry Pi Pico)" όπως φαίνεται στην εικόνα. Μετά τη σωστή σύνδεση στην πλακέτα, το πράσινο κουμπί Run θα πρέπει να είναι ενεργό (όχι γκριζαρισμένο). Εάν είναι γκριζαρισμένο, επιλέξτε ξανά "MicroPython (Raspberry Pi Pico)".



Τώρα προσπαθήστε να δημιουργήσετε ένα πρόγραμμα που θα προσομοιώνει τα φανάρια.

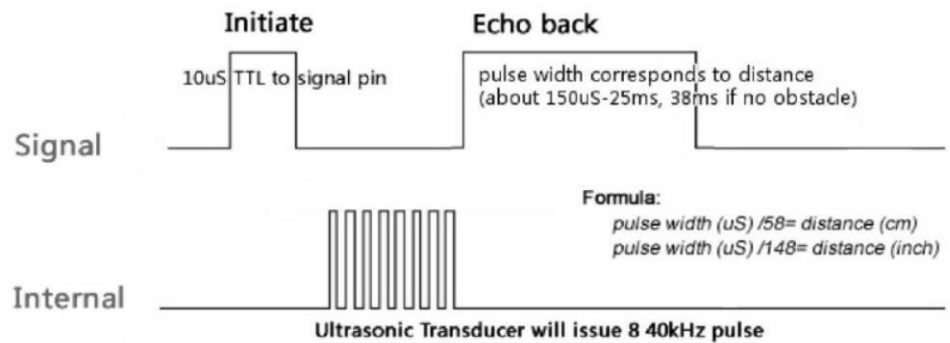
3.2 Εργασία προθέρμανσης no. 2: αισθητήρας απόστασης υπερήχων

Ο αισθητήρας απόστασης υπερήχων HC-SR04 σας επιτρέπει να μετράτε την απόσταση από ένα εμπόδιο. Η ιδέα της μέτρησης φαίνεται στο παρακάτω σχήμα:



Πηγή: https://www.researchgate.net/figure/A-block-diagram-of-Ultrasonic-sensor-working-principles_fig5_344385811

Για μέτρηση, σύμφωνα με την τεκμηρίωση του αισθητήρα (δείτε την παρακάτω εικόνα), ρυθμίστε το σήμα χαμηλά στην ακίδα trigger για μικρό χρονικό διάστημα, π.χ. 2μs. Στη συνέχεια, ρυθμίστε το υψηλό σήμα για 10μs. Για να δημιουργήσετε καθυστερήσεις σε μικροδευτερόλεπτα, χρησιμοποιήστε τη συνάρτηση: **utime.sleep_us()**. Στο επόμενο βήμα, ρυθμίστε το χαμηλό σήμα στην ακίδα trigger.



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

Δημιουργήστε ένα πρόγραμμα που διαβάζει τη μετρούμενη απόσταση από έναν αισθητήρα απόστασης υπερήχων και την εμφανίζει στο τερματικό. Για να το κάνετε αυτό:

1. Προσθέστε τις βιβλιοθήκες machine και utime.
2. Δηλώστε την ακίδα trigger ως έξοδο.
3. Διαμορφώστε την ακίδα στην οποία είναι συνδεδεμένη η ηχώ ως είσοδο.
4. Στον βρόχο while, ορίστε την τιμή της ακίδας trigger σε 0.
5. Περιμένετε 2 μs (συνάρτηση utime.sleep_us()).
6. Ορίστε την τιμή της ακίδας trigger σε 1.
7. Περιμένετε 10 μs.
8. Ορίστε την τιμή της ακίδας trigger σε 0.
9. Τώρα πρέπει να μετρήσουμε πόσο διαρκεί το υψηλό σήμα στον ακροδέκτη ηχούς, επειδή η διάρκεια του σήματος στον ακροδέκτη ηχούς σχετίζεται με την απόσταση. Για να το κάνουμε αυτό, θα χρησιμοποιήσουμε τη συνάρτηση utime.ticks_us(), η οποία μετρά πόσος χρόνος έχει περάσει σε μs από την έναρξη του προγράμματος. Αρχικά, θα δημιουργήσουμε έναν βρόχο while που θα εκτελείται όσο το σήμα είναι χαμηλό. Μέσα, θα τοποθετήσουμε τη συνάρτηση tick_us(). Με αυτόν τον τρόπο, θα λάβουμε πληροφορίες για το πότε το σήμα ήταν χαμηλό για τελευταία φορά. Προσθέστε αυτό το απόσπασμα κώδικα στο πρόγραμμά σας:

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

    signal_off = utime.ticks_us()
```

10. Ομοίως, μετρήστε πότε το σήμα ήταν υψηλό για τελευταία φορά και αποθηκεύστε την τιμή σε μια μεταβλητή που ονομάζεται π.χ. `signal_on`.

11. Η διαφορά μεταξύ του χρόνου της τελευταίας εμφάνισης του υψηλού και του χαμηλού σήματος είναι η διάρκεια του υψηλού παλμού στην ακίδα ηχούς. Πώς, όμως, μεταφράζουμε τη διάρκεια του παλμού σε απόσταση; Στην αρχή, εκπέμπεται ένα ηχητικό κύμα, το οποίο ανακλάται από το αντικείμενο και επιστρέφει στον αισθητήρα. Επομένως, στον μετρούμενο χρόνο (t), το κύμα διανύει τη διπλάσια απόσταση μεταξύ του αισθητήρα και του αντικειμένου και κινείται με ταχύτητα περίπου 340 m/s (η ταχύτητα του ήχου στον αέρα). Επομένως, μπορούμε να γράψουμε την εξίσωση για την ταχύτητα:

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

$$d = v \cdot \frac{t}{2} = 0.034 \frac{cm}{\mu s} \cdot \frac{t}{2} \approx \frac{t[\mu s]}{58}$$

Ως εκ τούτου, η λαμβανόμενη διάρκεια παλμού θα πρέπει να διαιρεθεί με το 58 για να ληφθεί η απόσταση σε εκατοστά. Προσθέστε τις ακόλουθες γραμμές στο πρόγραμμα:

```
diff = signal_on-signal_off
distance = diff/58.0
print("Distance="+str(distance))
```

Τώρα μπορείτε να δοκιμάσετε τη λειτουργία του προγράμματος. Εάν δυσκολεύεστε να διαβάσετε δεδομένα επειδή εμφανίζονται πολύ γρήγορα, μπορείτε να προσθέσετε μια μικρή καθυστέρηση στη λειτουργία εκτύπωσης.

4. Έξυπνος κάδος απορριμμάτων – επίπεδο 1

Τώρα δημιουργήστε ένα πρόγραμμα στο οποίο θα χρησιμοποιήσετε τις δεξιότητες που έχετε αποκτήσει κατά τη διάρκεια των εργασιών προθέρμανσης και θα μετρήσετε το επίπεδο πλήρωσης του κάδου απορριμμάτων. Εμφανίστε το επίπεδο πλήρωσης στις λυχνίες LED σύμφωνα με τις ποσοστιαίες περιοχές χωρητικότητας του κάδου που υποθέσατε στο στάδιο του σχεδιασμού του κάδου απορριμμάτων. Δημιουργήστε έναν κάδο απορριμμάτων από ένα χρησιμοποιήσιμο κουτί αποστολής/παπουτσιών και δοκιμάστε πώς λειτουργεί ο κάδος σας.

5. Έξυπνος κάδος απορριμμάτων – επίπεδο 2

Τώρα ας τροποποιήσουμε τον κάδο απορριμμάτων ώστε να είναι ακόμα πιο έξυπνος. Για να γίνει αυτό, θα στείλουμε δεδομένα στο cloud σχετικά με την πληρότητα του κάδου απορριμμάτων και τη θέση του.

Αυτά τα δεδομένα μπορούν αργότερα να χρησιμοποιηθούν για τη δημιουργία ενός αλγόριθμου που αναπτύσσει μια διαδρομή για το απορριματοφόρο ώστε να συλλέγει σκουπίδια μόνο από εκείνες τις τοποθεσίες όπου οι κάδοι είναι γεμάτοι, ελαχιστοποιώντας έτσι το αποτύπωμα άνθρακα. Για το σκοπό αυτό, θα χρησιμοποιήσουμε το Adafruit IO cloud. Αρχικά, πρέπει να δημιουργήσετε έναν δωρεάν λογαριασμό στη διεύθυνση <https://io.adafruit.com>. Στη συνέχεια, πρέπει να στείλετε δύο είδη δεδομένων: πλήρωση κάδου απορριμμάτων και τοποθεσία στο cloud. Για να το κάνετε αυτό, πρέπει να δημιουργήσετε δύο ροές (feeds). Οι ροές είναι αντικείμενα που αποθηκεύουν δεδομένα. Για να δημιουργήσετε μια ροή, μεταβείτε στην καρτέλα "Feed" και επιλέξτε το κουμπί "New Feed"..



Στη συνέχεια θα εμφανιστεί ένα παράθυρο όπου πρέπει να εισαγάγετε το όνομα της ροής, π.χ. Πλήρωση ή Τοποθεσία

Create a new Feed
✕

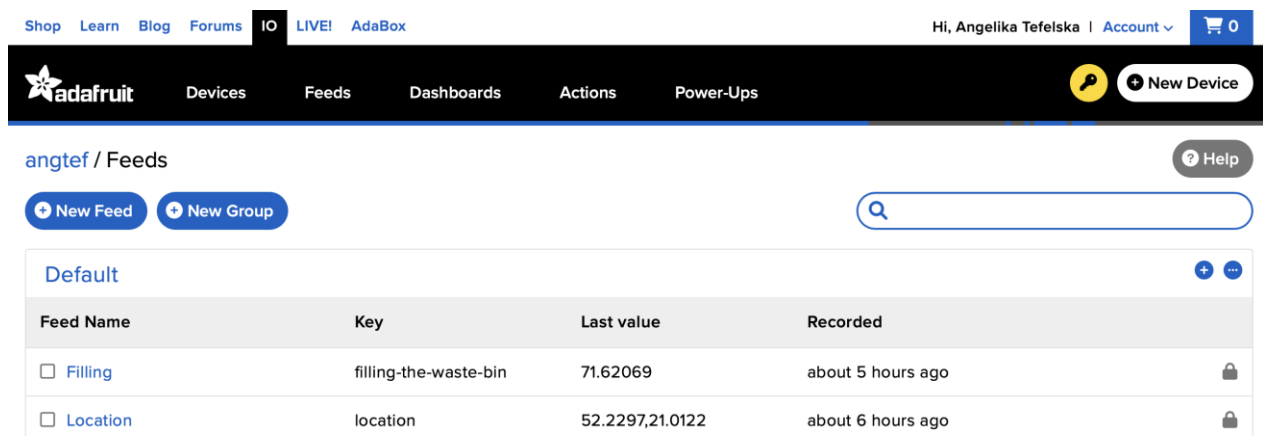
Name

Maximum length: 128 characters. Used: 0

Description

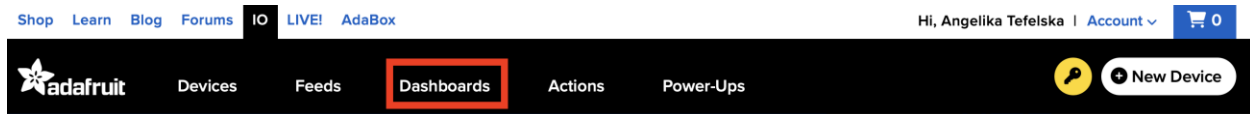
Cancel
Create

Δημιουργήστε δύο ροές. Μόλις το κάνετε αυτό, θα δείτε τις ροές που δημιουργήσατε στη σελίδα σας, παρόμοιες με το στιγμιότυπο οθόνης παρακάτω:



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

Το επόμενο βήμα είναι να δημιουργήσετε έναν πίνακα ελέγχου (dashboard). Για να το κάνετε αυτό, επιλέξτε την καρτέλα "Dashboards" και μετά "New Dashboard":



angtef / Dashboards

[New Dashboard](#)

Q |

Θα εμφανιστεί ένα παράθυρο όπου πρέπει να εισαγάγετε το επιλεγμένο όνομα του πίνακα ελέγχου. Στη συνέχεια, στη δεξιά πλευρά, επιλέξτε το σύμβολο ρυθμίσεων και επιλέξτε "Create New Block".

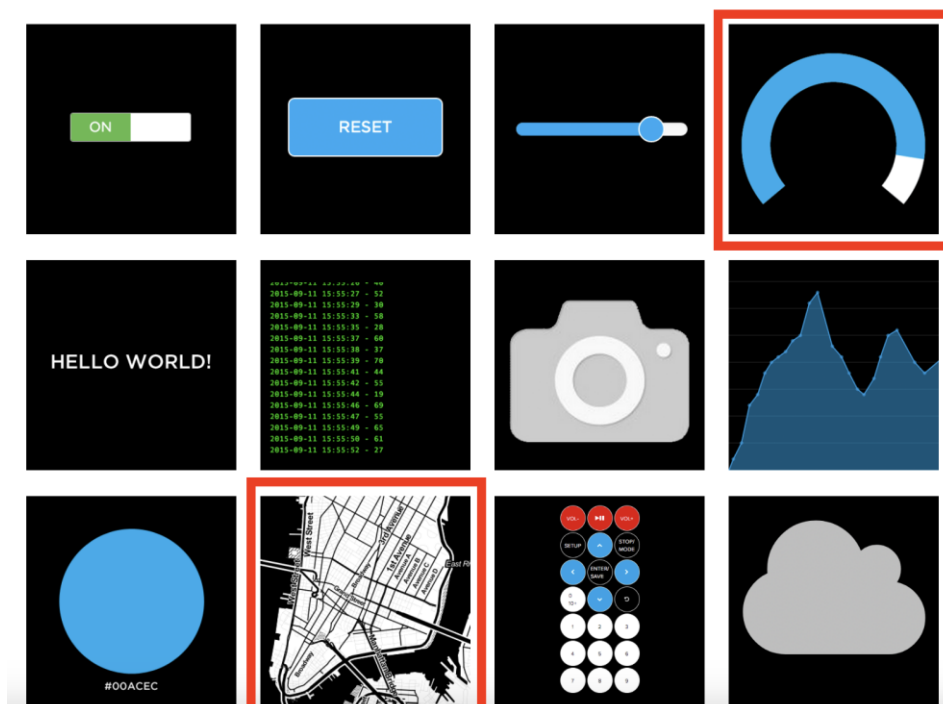


Το Adafruit IO διαθέτει διάφορα διαθέσιμα μπλοκ για την εμφάνιση δεδομένων (πλαίσια κειμένου, μετρητές, γραφήματα, χάρτες κ.λπ.). Στην περίπτωση μας, ας επιλέξουμε μετρητή και χάρτη:

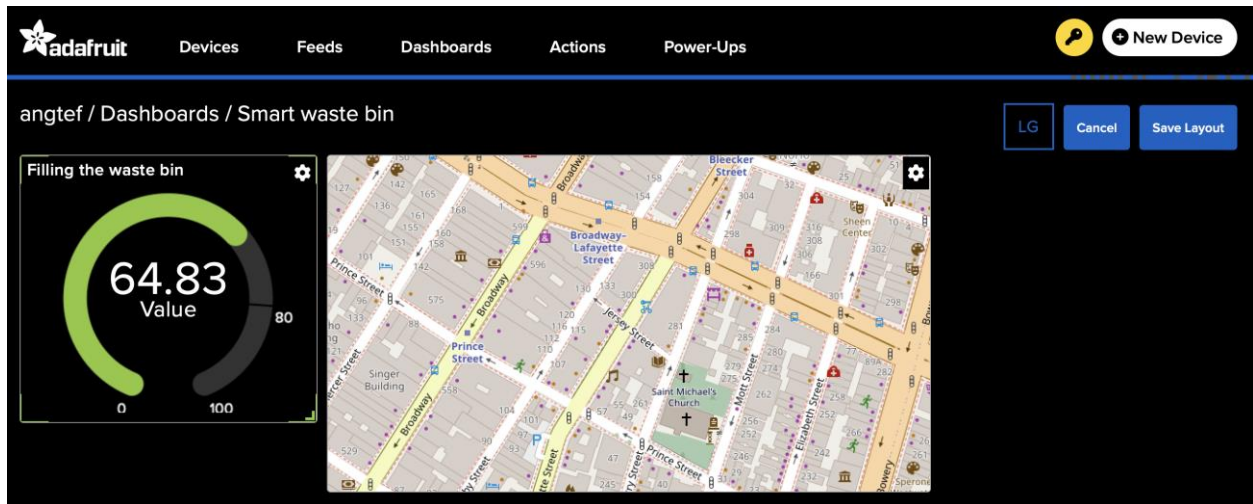
Create a new block



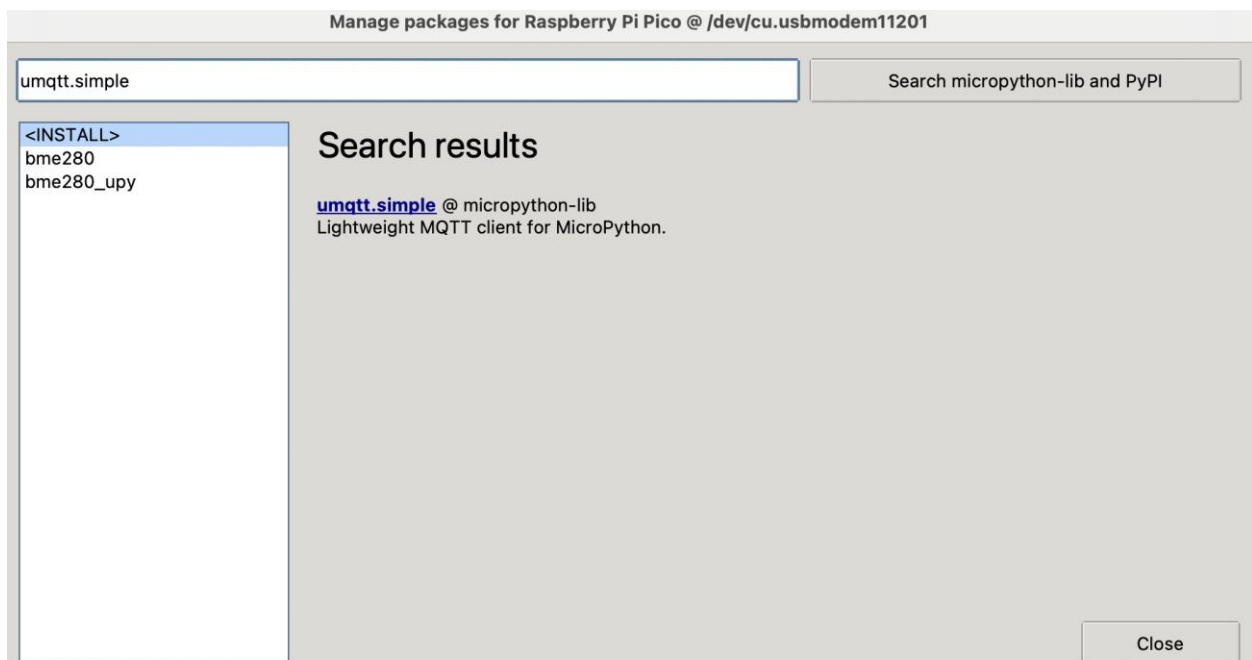
Click on the block you would like to add to your dashboard. You can always come back and switch the block type later if you change your mind.



Τότε θα εμφανιστεί ένα παράθυρο που ρωτά σε ποια ροή θέλουμε να συνδέσουμε το μπλοκ. Συνδέστε την ροή για το γέμισμα του κάδου απορριμμάτων στην αντίστοιχη ένδειξη και την ροή που καθορίζει τη θέση στον χάρτη. Στη συνέχεια, επιλέξτε ξανά το εικονίδιο ρυθμίσεων και επιλέξτε "Edit Layout". Τακτοποιήστε τα μπλοκ στην οθόνη όπως σας ταιριάζει. Μπορείτε επίσης να μεγεθύνετε και να σμικρύνετε τα μπλοκ. Ένα παράδειγμα της εμφάνισης φαίνεται στο στιγμιότυπο οθόνης παρακάτω:



Τώρα ας πάμε στον επεξεργαστή Thonny και ας εγκαταστήσουμε τη βιβλιοθήκη **umqtt.simple**. Για να το κάνετε αυτό, επιλέξτε «Tools» και μετά «Manage packages». Θα εμφανιστεί ένα παράθυρο όπου θα πρέπει να εισαγάγετε το όνομα της βιβλιοθήκης που θέλετε να εγκαταστήσετε, η οποία είναι η **umqtt.simple**.



Όταν κάνετε κλικ στο όνομα της βιβλιοθήκης umqtt.simple, θα ανοίξει ένα παράθυρο με δεδομένα βιβλιοθήκης και θα μπορείτε να εγκαταστήσετε τη βιβλιοθήκη κάνοντας κλικ στο κουμπί Install.

Τώρα μπορούμε να επιστρέψουμε στον κώδικά μας και να προσθέσουμε τα κομμάτια που είναι απαραίτητα για την αποστολή δεδομένων στο cloud. Για να το κάνετε αυτό, ακολουθήστε τα εξής βήματα:

1. Προσθέστε τις απαραίτητες βιβλιοθήκες (network library και import symbol MQTTClient από τη βιβλιοθήκη umqtt.simple):

```
smart_waste_bin.py * <untitled> *
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
```

2. Προσθέστε ένα κομμάτι κώδικα που θα σας επιτρέψει να συνδεθείτε στο WIFI σας. Εδώ πρέπει να συμπληρώσετε τις γραμμές 15-18. Πρώτα, πληκτρολογήστε το όνομα του WIFI σας και μετά τον κωδικό πρόσβασης.

```
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:
```

Οι δύο τελευταίες παράμετροι είναι τα δεδομένα από το Adafruit IO. Επιστρέψτε στον ιστότοπο Adafruit και κάντε κλικ στο σύμβολο του κλειδιού. Όταν το κάνετε αυτό, θα εμφανιστούν τα στοιχεία σύνδεσης και το κλειδί σας. Αντιγράψτε αυτά τα δεδομένα στο πρόγραμμα στις γραμμές 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.

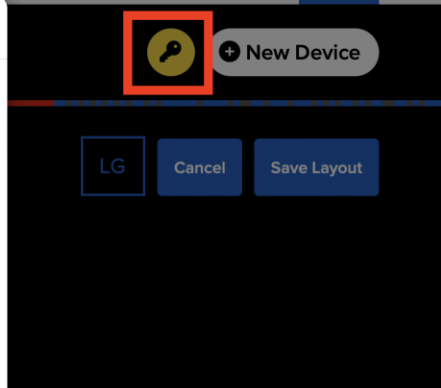
If you need to regenerate a new Adafruit IO Key, all of your existing programs and scripts will need to be manually changed to the new key.

Username

Active Key

REGENERATE KEY





3. Τώρα θα χρησιμοποιήσουμε το MQTT (Message Queuing Telemetry Transport), ένα ελαφρύ πρωτόκολλο επικοινωνίας που βασίζεται στο μοντέλο δημοσίευσης/εγγραφής. Σχεδιάστηκε ειδικά για την αποστολή δεδομένων σε περιβάλλοντα με περιορισμένους πόρους, όπως συσκευές IoT (Internet of Things). Για να το κάνετε αυτό, χρησιμοποιήστε τον παρακάτω κώδικα, αλλάζοντας μόνο τα ονόματα των ροών στις γραμμές 33-34. Το παράδειγμα χρησιμοποιεί ροές με το όνομα: "Filling" και "Location". Αντικαταστήστε τα με τα δικά σας. Απλώς θυμηθείτε να αφήσετε το /csv στην περίπτωση της ροής Location γιατί όταν χρησιμοποιείτε χάρτες, πρέπει να παρέχετε δεδομένα σε μορφή csv.

```
smart_waste_bin.py * 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. Τώρα ας προσθέσουμε συναρτήσεις για την αποστολή δεδομένων στο cloud:

```

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

```

Το τελευταίο βήμα είναι να μεταβιβάσετε τη μετρημένη πληρότητα και τη θέση του κάδου στη συνάρτηση στο βρόχο while μετά το #your code. Μπορείτε να διαβάσετε την τοποθεσία σας, για παράδειγμα, από τους χάρτες Google και να την αντικαταστήσετε στη μεταβλητή "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

```

Δοκιμάστε τον έξυπνο κάδο απορριμμάτων. Θυμηθείτε να επιστρέψετε στη σελίδα Adafruit IO στον πίνακα ελέγχου που δημιουργήσατε για να δείτε εάν στέλνετε σωστά τα δεδομένα σας στο cloud.

6. Ολοκλήρωση: προβληματισμός σχετικά με τη λειτουργικότητα και τις πιθανές βελτιώσεις

Πιστεύετε ότι η χρήση έξυπνων κάδων θα μπορούσε να βελτιώσει τη συλλογή απορριμμάτων και να μειώσει το αποτύπωμα άνθρακα; Στο έργο, δεν βελτιστοποιήσαμε τη διαδρομή των απορριμματοφόρων, αλλά μπορείτε να φανταστείτε ότι υπάρχει ένα σύστημα που διαβάζει δεδομένα από το cloud από κάθε κάδο και προετοιμάζει μια βέλτιστη διαδρομή. . Εάν ενδιαφέρεστε για το πώς να φτιάξετε έναν τέτοιο αλγόριθμο, μπορείτε να εξοικειωθείτε με τα προγράμματα από τον ιστότοπο: <https://colab.research.google.com/drive/1aOq9jRh6c6fhaw1ahe0a1-yKVdMnO613?usp=sharing%2F> .

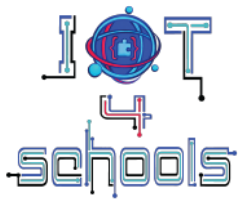
.....

.....

.....

.....

.....



Project number: 2023-1-PL01-KA220-SCH-000154043



**Co-funded by
the European Union**

Ποιες άλλες τροποποιήσεις μπορούν να γίνουν ώστε οι έξυπνοι κάδοι να λειτουργούν ακόμα πιο αποτελεσματικά;

.....

.....

.....

.....

.....