Arduinos for the classroom*

Chrysostomos Valderanis (LMU)

* Δεν έχουμε Arduinos αυτήν την χρονιά

Βασισμένο σε ιδέες και υλικό των Andromachi Tsirou & Piero Giorgio Verdini

Introduction to using modern sensing devices

Chrysostomos Valderanis (LMU)

Βασισμένο σε ιδέες και υλικό των Andromachi Tsirou & Piero Giorgio Verdini

Τι χρειαζόμαστε -1;

- Laptop με Windows10 (11)
- Χρήστη με administrative rights ή να ξέρουμε το administrative password.
- Σύνδεση στο internet
- Θύρα USB type A (USB 2.0 είναι αρκετό)

- Συνδεόμαστε στη διεύθυνση (Piero) https://www.mycloud.swisscom.ch/s/S004F5C42F313CB2BBD3852F9AAB19E7CFC6020FC40
  - Σε αυτή τη διεύθυνση μπορείτε να βρείτε όλες τις βιβλιοθήκες, και ακόμη περισσότερα, που θα δοκιμάσουμε σήμερα.

Τι χρειαζόμαστε -1α;

- Αν χρησιμοποιείτε Linux.
  - Κάποιες σύγχρονες διανομές Linux φορτώνουν ένα daemon για braille displays.
  - Ο USB-> Serial μετατροπέας που υπάρχει στον uC χρησιμοποιείται και στα braille displays.
  - Πρέπει να πείσουμε το Linux να μην χρησιμοποιήσει την USB σύνδεση ως braille display
  - Η γρήγορη λύση είναι sudo apt-get remove –purge brltty
  - Όλες οι υπόλοιπες οδηγίες είναι ίδιες με αυτές των Windows 10

Τι χρειαζόμαστε -2;

- έναν driver για να μιλήσουμε με τον uC
  - Piero -> Device Drivers -> CH341SER.EXE – το κατεβάζουμε στο laptop
  - Για την εγκατάσταση πρέπει να είναι συνδεμένο το υλικό
- Ένα περιβάλλον για να γράφουμε τον κώδικα και να μπορούμε να τον μεταφέρουμε από το laptop στον uC
  - https://thonny.org/ Κατεβάζουμε την έκδοση για windows (thonny-4.1.4.exe)
- Την βιβλιοθήκη για το SHT30
  - https://github.com/rsc1975/micropython-sht30 - κατεβάζουμε το αρχείο στο laptop

Τι χρειαζόμαστε -3;

- Cable USB A -> USB C
- D1 Mini (ESP8266) CPU board
  - https://www.wemos.cc/en/latest/d1/d1_mini.html
- SHT30 shield (temperature humidity sensor)
- WS2812B RGB LED shield

- Προσφέρονται από εμάς

Μετράμε την θερμοκρασία στα πειράματα;

- ΝΑΙ
- Τα χαρακτηριστικά των ανιχνευτών εξαρτώνται από την θερμοκρασία και άρα η μέτρησή της μας βοηθάει να βελτιστοποιήσουμε την απόδοσή τους
- Τα ηλεκτρονικά των ανιχνευτών παράγουν θερμότητα και πρέπει να ξέρουμε σε τι θερμοκρασία λειτουργούν καθώς επηρεάζει το χρόνο ζωής τους
- Ο συνδυασμός θερμοκρασίας – υγρασίας μπορεί να οδηγήσει σε υγροποίηση μέσα στον όγκο του ανιχνευτή – σημείο δρόσου

Ένα παράδειγμα από το πείραμα ATLAS

- Στο φασματόμετρο μιονίων του πειράματος ATLAS υπάρχουν
  - 1k ανιχνευτικοί θάλαμοι με 10 αισθητήρες θερμοκρασίας / θάλαμο
  - 12k ηλεκτρονικές κάρτες με 1 αισθητήρα / κάρτα
  - 1k κάρτες συγκέντρωσης δεδομένων με 1 αισθητήρα / κάρτα
  - 3 αισθητήρες μαγνητικού πεδίου / θάλαμο με 1 αισθητήρα θερμοκρασίας
  - Σύνολο ~ 25000 αισθητήρες θερμοκρασίας

Μετράτε την θερμοκρασία στις τάξεις σας;

- Ελπίζω πως ναι
- Η μέτρηση της θερμοκρασίας μπορεί να χρησιμοποιηθεί από την μετεωρολογία μέχρι τον κήπο και στα περισσότερα πειράματα φυσικής
- Μπορείτε να πάρετε το σημερινό σύστημα μαζί σας.

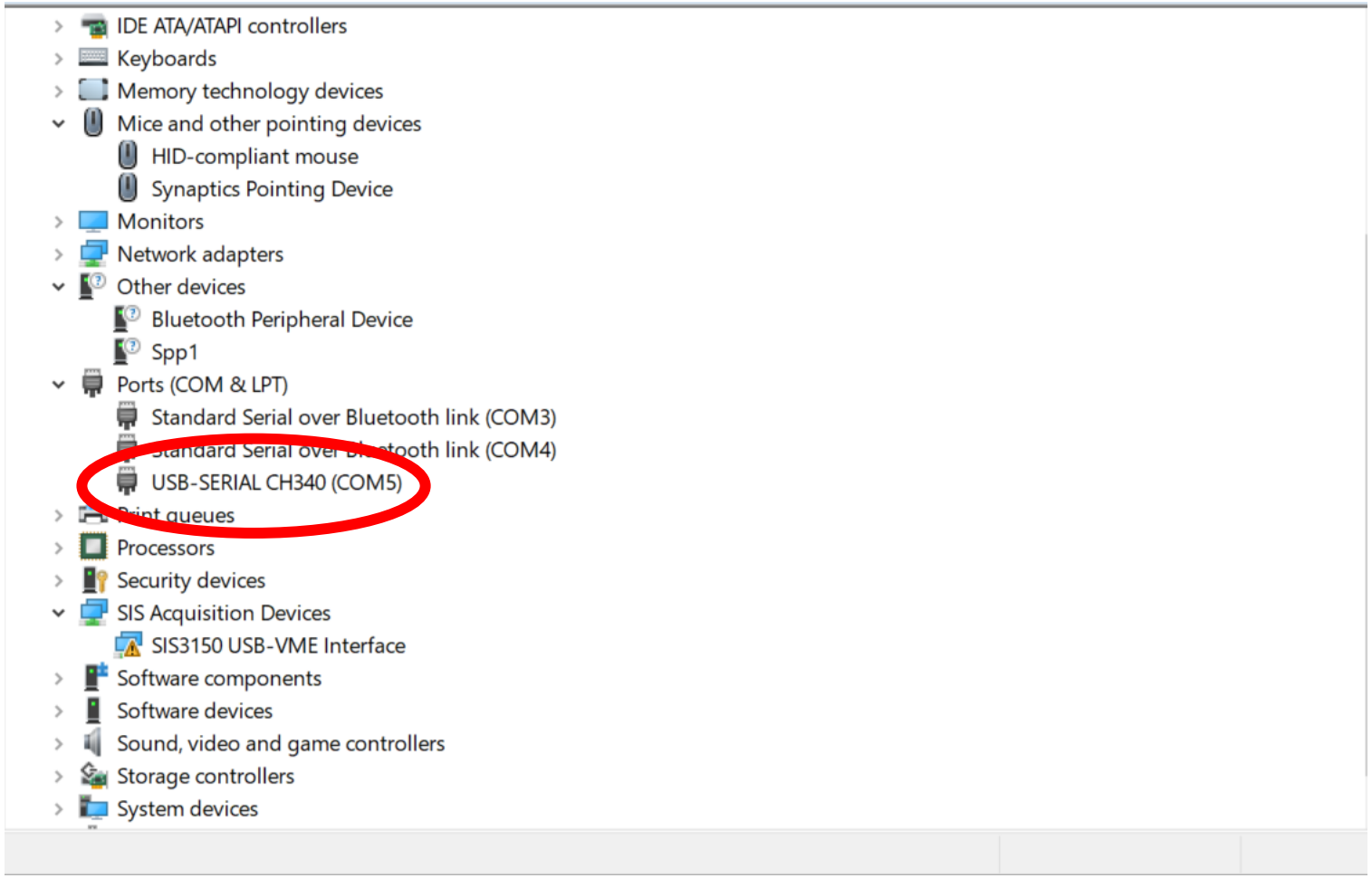
- Η μεγαλύτερη διαφορά μεταξύ των πειραμάτων και της τάξης είναι ότι τα πειράματα θέλουν να έχουν τον πλήρη έλεγχο του συστήματος. Ξεκινάμε από την αναλογική έξοδο των αισθητήρων, συνήθως τάση, και υλοποιούμε όλα τα περιφερειακά συστήματα που χρειάζονται για την υποστήριξη μέχρι την τελική αποθήκευση της μέτρησης.

Γιατί Python;

- Τυπικά τα περισσότερα συστήματα μετρήσεων προγραμματίζονται
  - είτε σε γραφικά περιβάλλοντα (scratch, blockly etc..) – κυρίως στα σχολεία
  - Είτε σε C, ειδικές γλώσσες – όλοι οι υπόλοιποι
- Σε πολλά περιβάλλοντα, η γλώσσα προγραμματισμού Python έχει κυριαρχήσει
  - Πάρα πολλές βιβλιοθήκες
  - Προχωρημένες δομές δεδομένων διαθέσιμες από την γλώσσα
  - Καλύτερος έλεγχος λαθών
- Μπορούμε να μεταφέρουμε και τον προγραμματισμό των uC σε python

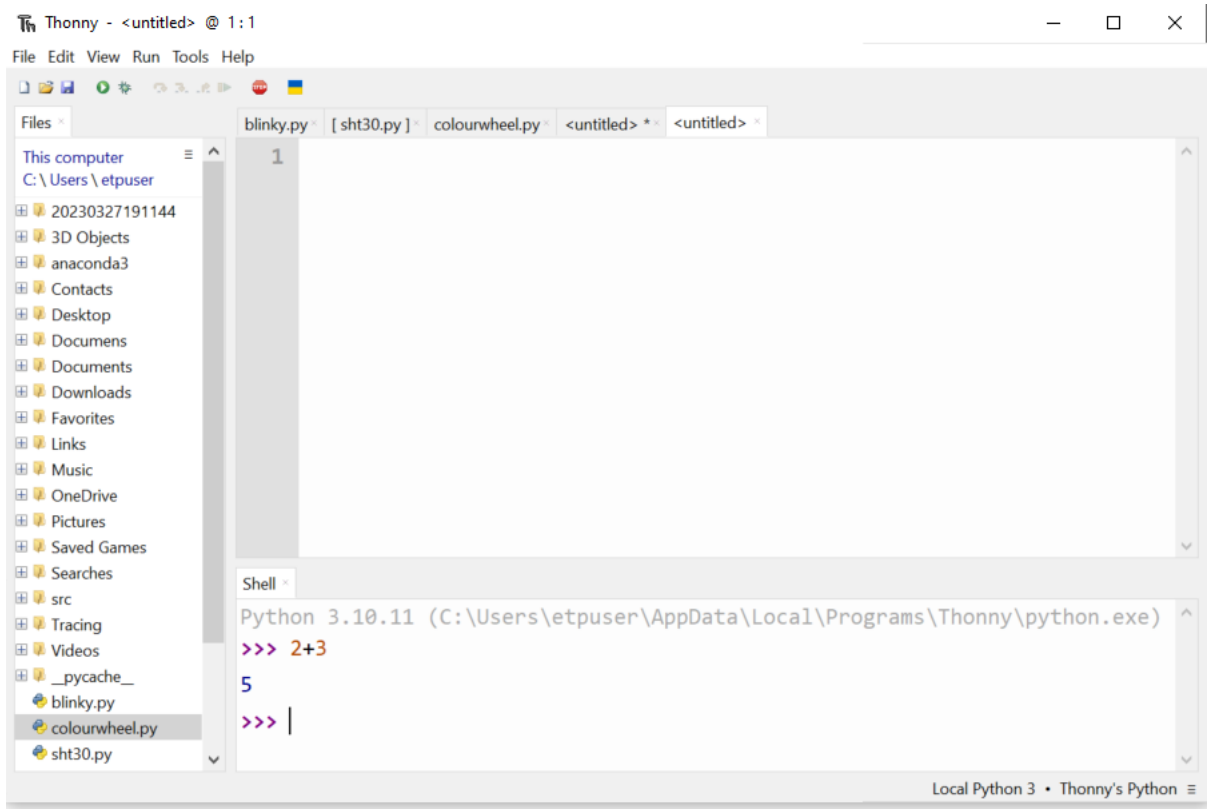
Εγκατάσταση driver για επικοινωνία laptop – D1 board

- Piero -> Device Drivers -> CH341SER.EXE – το κατεβάζουμε στο laptop
- Συνδέουμε την D1 board
- Δεξί κλικ και run as administrator…
- Μετά από την εγκατάσταση ο device manager των windows πρέπει να δείχνει



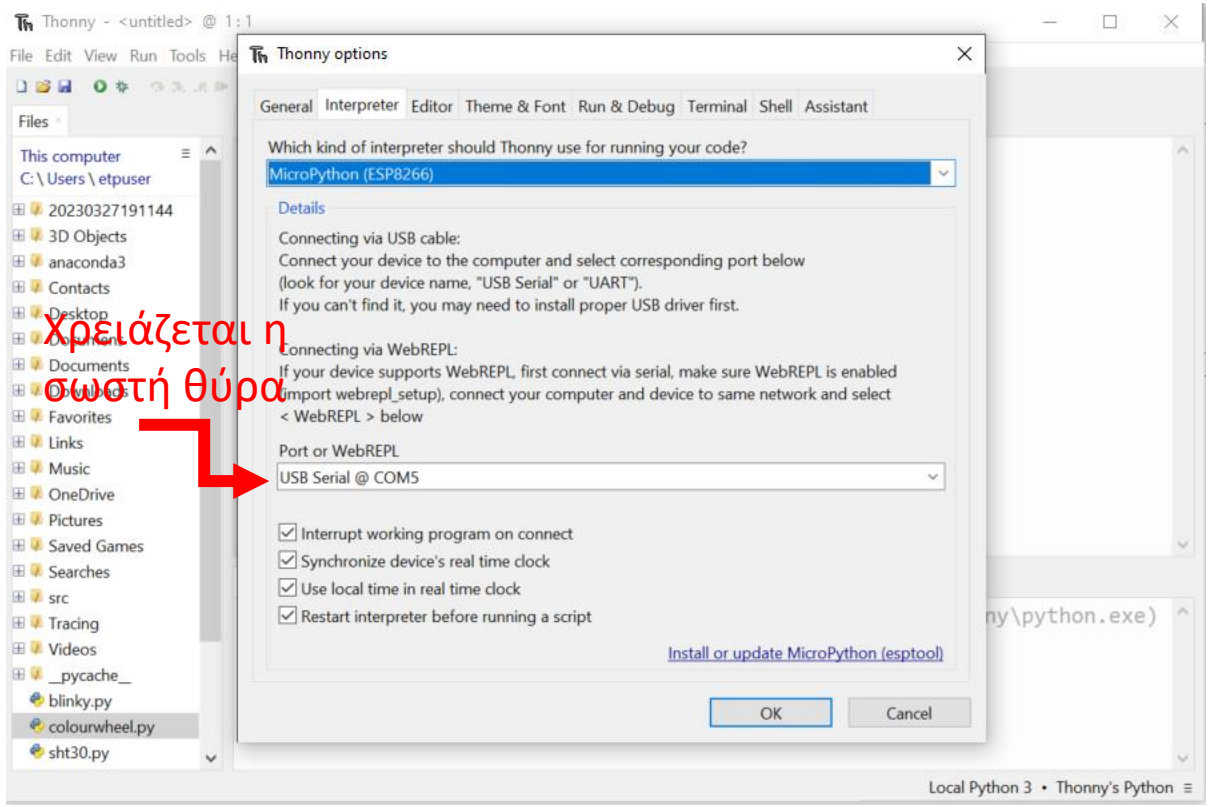
Εγκατάσταση Thonny

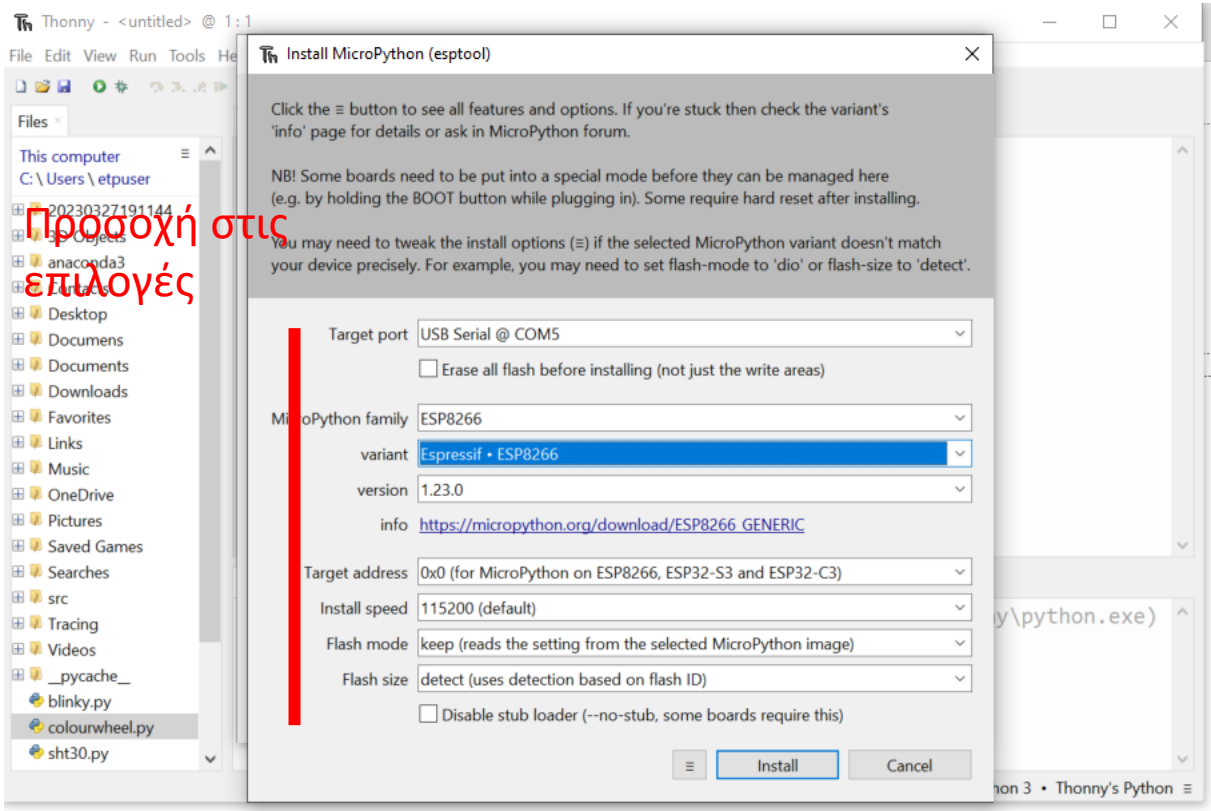
- Περιβάλλον για τον προγραμματισμό σε Python.
  - Δοκιμάστε κάτι στο διαθέσιμο κέλυφος



Μεταφορά ελέγχου στον uC

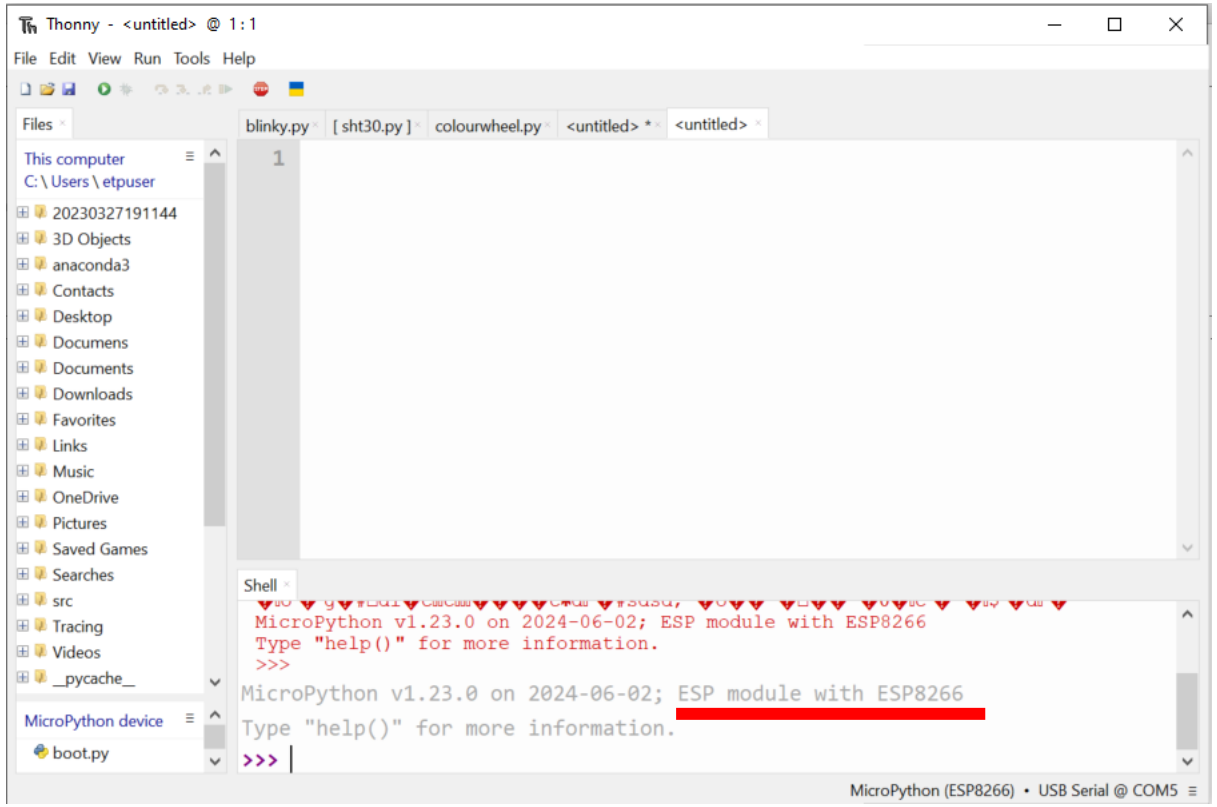
- Thonny -> Run -> Configure interpreter





Τελική κατάσταση Thonny

- Ο κώδικας (θα) τρέχει στον uC.



Πρώτη δοκιμή. Αναβοσβήνοντας ένα LED - 1

- Αντιγράψτε, με κάποιο τρόπο, το παρακάτω πρόγραμμα σε ένα καινούριο αρχείο στο Thonny
  - Από το powerpoint, από την σελίδα στο indico, δακτυλογραφώντας το

```
import time
from machine import Pin
ledpin = Pin(2, Pin.OUT)
while(True):
    ledpin.on()
    time.sleep(0.5)
    ledpin.off()
    time.sleep(0.5)
```

These two lines are similar to #include <library.h> in C/C++

here we define the LED pin

now we loop forever

turn the pin to 1 (LED off)

wait half a second

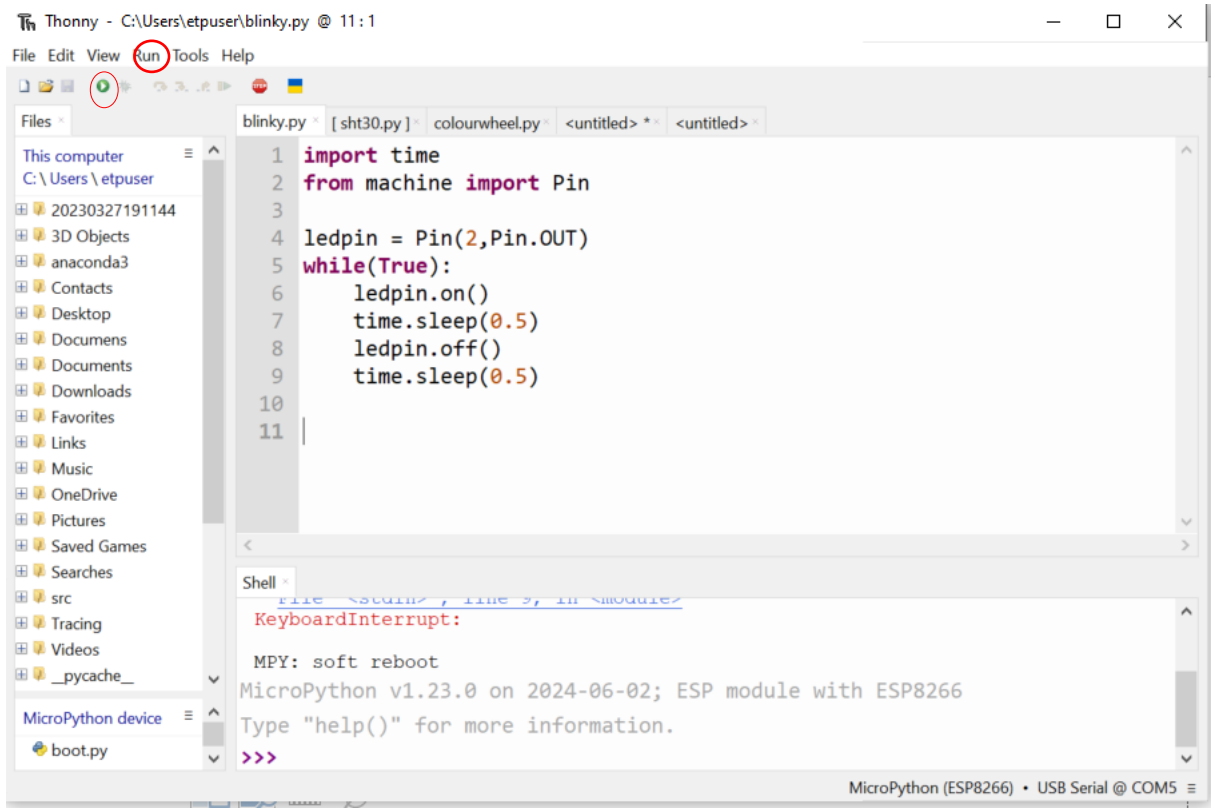
turn the pin to 0 (LED on)

wait half a second

Πώς ξέρουμε για το machine, Pin, 2 as magic number?
Από την τεκμηρίωση της κάρτας.
https://docs.micropython.org/en/latest/esp8266/quickref.html

https://www.wemos.cc/en/latest/tutorials/d1/get_started_with_micropython_d1.html

Πρώτη δοκιμή. Αναβοσβήνοντας ένα LED - 2



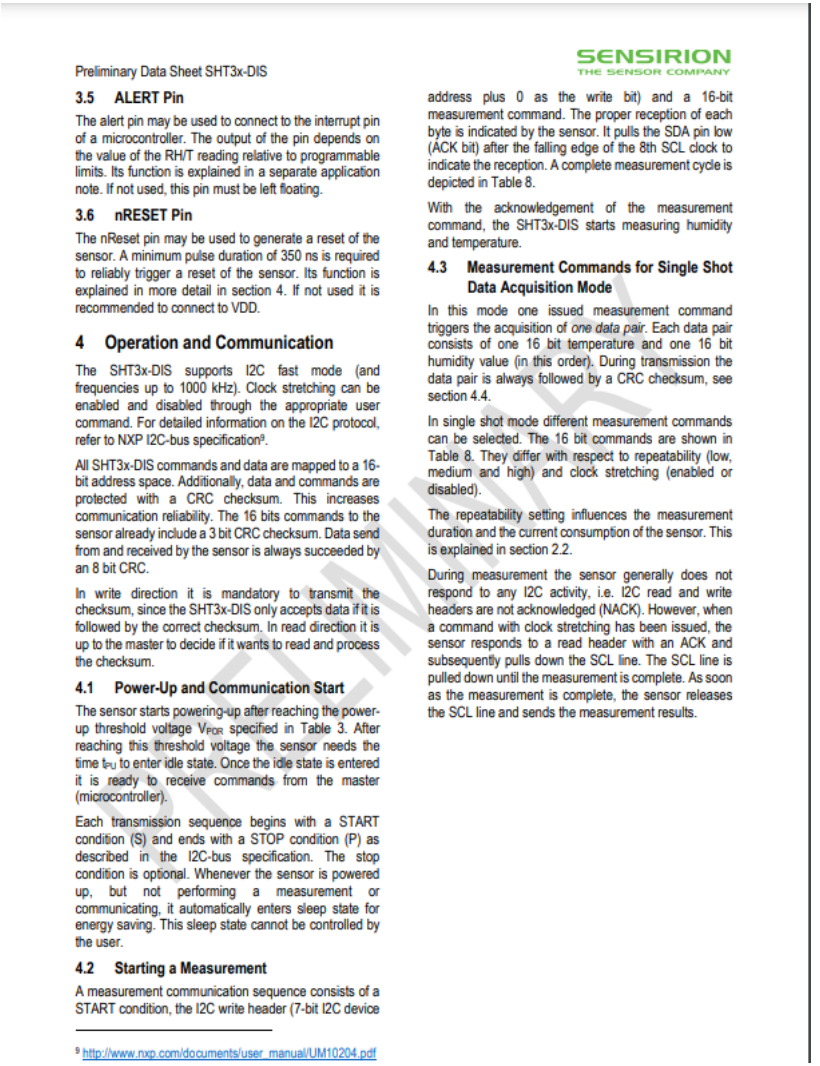
Προσοχή
1. στο indentation του python
2. Πού εκτελείται το πρόγραμμα

Το τρέχουμε
- Πατώντας F5
- Πατώντας το πράσινο βελάκι
- Run -> Run current script

- Βλέπετε το LED να αναβοσβήνει;
- Μπορείτε να βεβαιώσετε ότι πράγματι ledpin.on αντιστοιχεί σε φως;

Δεύτερη δοκιμή. Διαβάζοντας την θερμοκρασία - 1

- Χρησιμοποιούμε το SHT30 sensor shield
  - Η θερμοκρασία διαβάζεται μέσω του πρωτοκόλλου I2C
  - Κάποιος χρειάζεται να διαβάσει το datasheet και να το υλοποιήσει
    - Αυτό το κάνει ο συγγραφέας της library



Preliminary Data Sheet SHT3x-DIS

SENSIRION
THE SENSOR COMPANY

3.5 ALERT Pin

The alert pin may be used to connect to the interrupt pin of a microcontroller. The output of the pin depends on the value of the RH/T reading relative to programmable limits. Its function is explained in a separate application note. If not used, this pin must be left floating.

3.6 nRESET Pin

The nReset pin may be used to generate a reset of the sensor. A minimum pulse duration of 350 ns is required to reliably trigger a reset of the sensor. Its function is explained in more detail in section 4. If not used it is recommended to connect to VDD.

4 Operation and Communication

The SHT3x-DIS supports I2C fast mode (and frequencies up to 1000 kHz). Clock stretching can be enabled and disabled through the appropriate user command. For detailed information on the I2C protocol, refer to NXP I2C-bus specification[9].

All SHT3x-DIS commands and data are mapped to a 16-bit address space. Additionally, data and commands are protected with a CRC checksum. This increases communication reliability. The 16 bits commands to the sensor already include a 3 bit CRC checksum. Data send from and received by the sensor is always succeeded by an 8 bit CRC.

In write direction it is mandatory to transmit the checksum, since the SHT3x-DIS only accepts data if it is followed by the correct checksum. In read direction it is up to the master to decide if it wants to read and process the checksum.

4.1 Power-Up and Communication Start

The sensor starts powering-up after reaching the power-up threshold voltage V_{POR} specified in Table 3. After reaching this threshold voltage the sensor needs the time t_{PU} to enter idle state. Once the idle state is entered it is ready to receive commands from the master (microcontroller).

Each transmission sequence begins with a START condition (S) and ends with a STOP condition (P) as described in the I2C-bus specification. The stop condition is optional. Whenever the sensor is powered up, but not performing a measurement or communicating, it automatically enters sleep state for energy saving. This sleep state cannot be controlled by the user.

4.2 Starting a Measurement

A measurement communication sequence consists of a START condition, the I2C write header (7-bit I2C device address plus 0 as the write bit) and a 16-bit measurement command. The proper reception of each byte is indicated by the sensor. It pulls the SDA pin low (ACK bit) after the falling edge of the 8th SCL clock to indicate the reception. A complete measurement cycle is depicted in Table 8.

With the acknowledgement of the measurement command, the SHT3x-DIS starts measuring humidity and temperature.

4.3 Measurement Commands for Single Shot Data Acquisition Mode

In this mode one issued measurement command triggers the acquisition of *one data pair*. Each data pair consists of one 16 bit temperature and one 16 bit humidity value (in this order). During transmission the data pair is always followed by a CRC checksum, see section 4.4.

In single shot mode different measurement commands can be selected. The 16 bit commands are shown in Table 8. They differ with respect to repeatability (low, medium and high) and clock stretching (enabled or disabled).

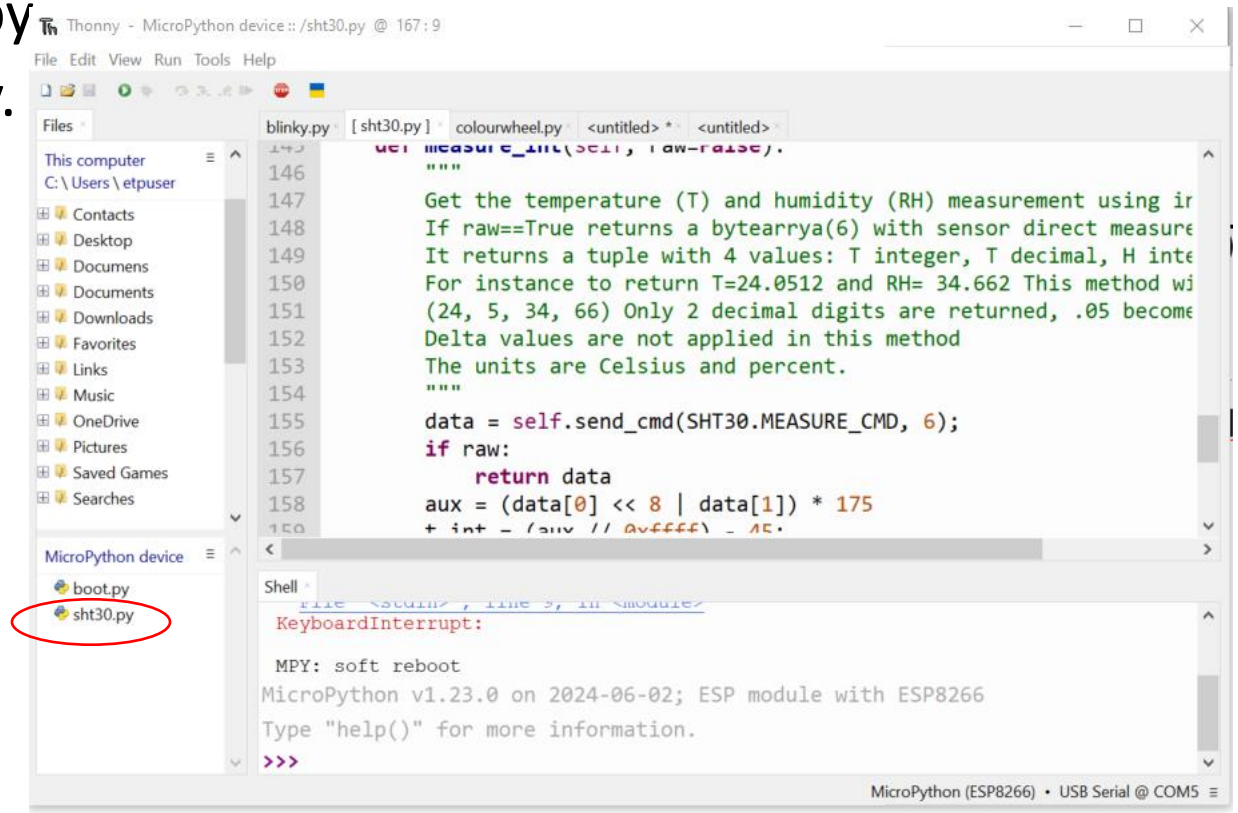
The repeatability setting influences the measurement duration and the current consumption of the sensor. This is explained in section 2.2.

During measurement the sensor generally does not respond to any I2C activity, i.e. I2C read and write headers are not acknowledged (NACK). However, when a command with clock stretching has been issued, the sensor responds to a read header with an ACK and subsequently pulls down the SCL line. The SCL line is pulled down until the measurement is complete. As soon as the measurement is complete, the sensor releases the SCL line and sends the measurement results.

[9] http://www.nxp.com/documents/user_manual/UM10204.pdf

Δεύτερη δοκιμή. Διαβάζοντας την θερμοκρασία - 2

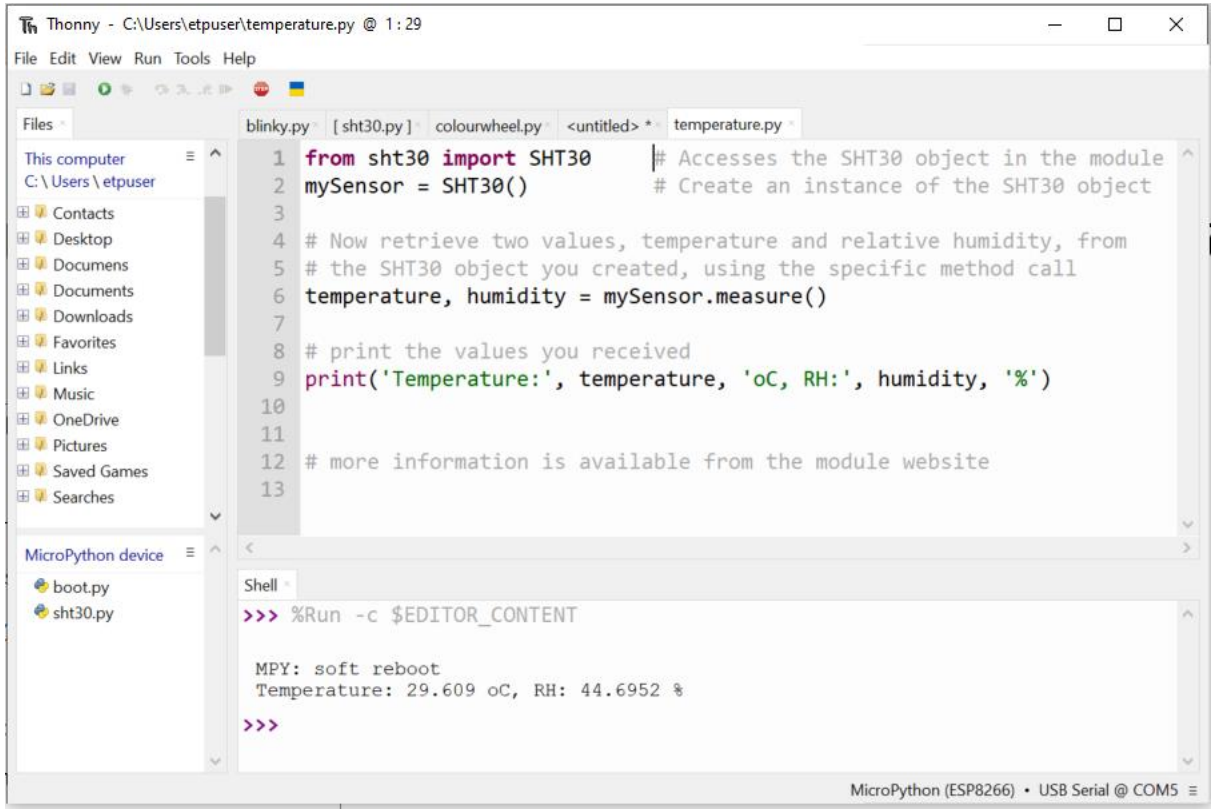
- Πρέπει να μεταφέρουμε το αρχείο sht30.py (η βιβλιοθήκη) στον uC.
  - Piero -> uPython Code -> sht30.py
  - Ανοίγουμε το αρχείο στο Thonny.
  - Save as … στον uC



Δεύτερη δοκιμή. Διαβάζοντας την θερμοκρασία - 3

Αντιγράψτε το παρακάτω πρόγραμμα σε ένα αρχείο στο Thonny

```
from sht30 import SHT30 # Accesses the SHT30 object in the module
mySensor = SHT30()      # Create an instance of the SHT30 object
# Now retrieve two values, temperature and relative humidity, from
# the SHT30 object you created, using the specific method call
temperature, humidity = mySensor.measure()
# print the values you received
print('Temperature:', temperature, 'ºC, RH:', humidity, '%')
# more information is available from the module website
```



Τρίτη δοκιμή. Μια γεννήτρια χρωμάτων

- Χρησιμοποιούμε το WS2812B RGB LED shield.
- Πού είναι η βιβλιοθήκη του; Είναι ενσωματωμένη στην διανομή που χρησιμοποιούμε.
- Ανοίξτε το αρχείο colourwheel.py στο Thonny. Δοκιμάστε τις διάφορες συναρτήσεις που έχει ορίσει.