

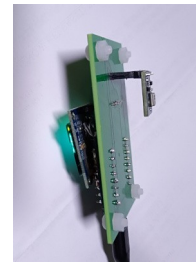


Dubtub Monitor

Thank you for your purchase of the BeneschTech dubtub monitor. This device will happily measure temperature and humidity within your tub every 15 seconds for hopefully many years. It is manufactured with pride in small batches in Colorado Springs, CO for home mycologists using all open source software. Did you know that this monitor is also wifi capable? It sure is. You can stream the readings to an app of your choice listening to UDP broadcast packets on port 2708 in JSON format, or ours. It is powered by a simple USB phone charger, and can also use the USB port for programming / set up. Unfortunately, it is the older Micro-USB type, but not to fear, it sips current and is nowhere near the current limit for that type of interface. If you have the 2026.08 version (blue board), there are also solder points to wire in 5 volts directly. The rear sensor is detachable for easy installation to a plastic tub top. The easiest way I've found is to use a 1/8" (3mm) drill bit and drill a hole in the center of the top, then push the drill to the sides to make a 3/8-1/2" (13mm) sideways slit to fit the connector through. Then, push the connector through and use a marker to mark the mounting holes locations (4). Drill those with a 4mm or 3/16" bit. You can then use M3/10mm bolts / nuts to hold it in place securely, and some clear silicone around the periphery of the connector to keep it rust free. Alternatively 1/8" dia/ 1/2" will probably work as well. We strongly advise using stainless steel or plastic mounting hardware as it is in a very humid environment.

Specifications

Type	Value
Sensor Type	AHT10 – 3.3v (removable)
Display Technology	1" OLED / SSD1306 - 5v
Bus Communication	I2C shared data lines
Microcontroller Type	ESP8266 D1 Mini Clone
Power	5V 300ma through USB cable, solder points (blue board only)
Mounting Holes (4)	M3 diameter
Operating system	Micropython 1.28.0
Wifi Compatibility	2.4 Ghz, WPA2/WEP/Open

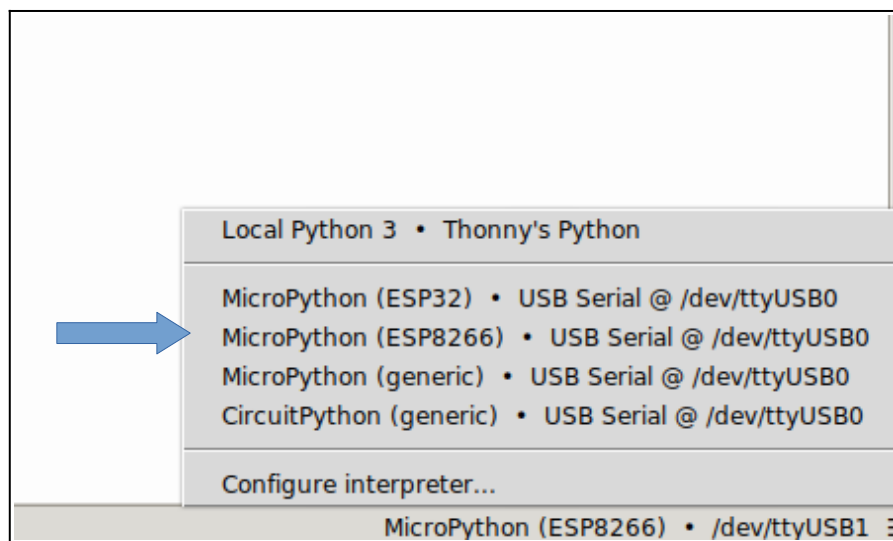


Setup / Programming

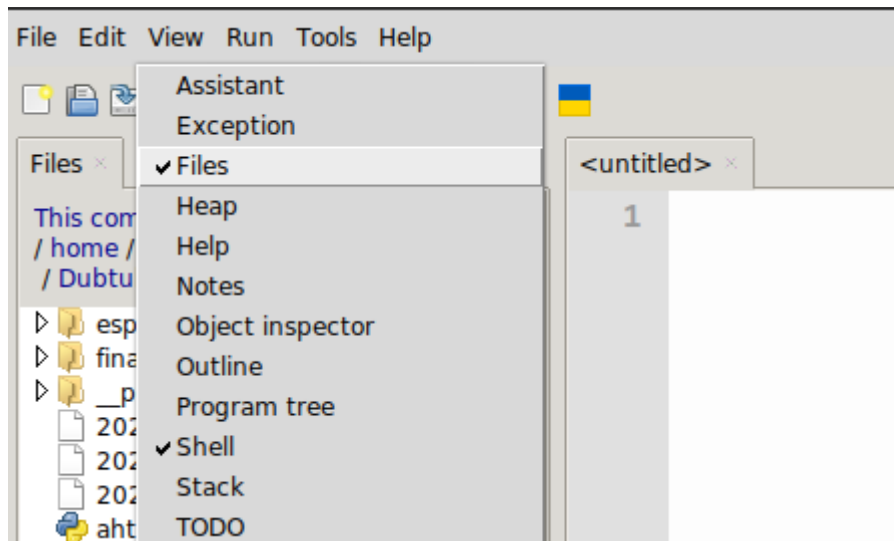
If you are fine with all of your grows being named "Test Tub 1" and no wifi hookup, you're done. However, I think most customers won't be satisfied with this. To set this up, you will have to modify some files on the device itself. Do not fear though, it is really easy. To work with most microcontrollers running MicroPython, especially this one, you will need a free software package called "Thonny". Don't worry, there's no tracking cookies, AI, subscription, or ads. Just good software the way it was meant to be written.

You can obtain it here → <https://thonny.org/>

The installation is pretty straightforward and is best followed by their directions. Plug the device into your PC using the provided USB cable. It should register via the CH34 driver as either a COM port on Windows, or /dev/ttyUSB device on Linux. Thonny will likely automatically detect which one is the right one, but if not you might need to help it find it. In the lower right hand side, you can click on the statusbar and select which one is the right one. You should choose the one marked ESP8266



After that, if it is not already, you want to open up the “Files” view. You can enable this by using the standard menu in the top left of the application under “View”



On the left, under the section “Micropython Device”, double click on “boot.py” to open it. From there you can modify the top 4 variables to your liking.

```
[ boot.py ] ×
1  TUBNAME="Test Tub 1"
2  INTERVAL=15000
3  WIFI_NETNAME=""
4  WIFI_PASSWD=""
5
6
7  from time import sleep
8  from machine import Pin,I2C,Timer,idle
9  from ssd1306 import SSD1306_I2C
10 from ahtx0 import AHT10
11 from network import WLAN
12 from bignumbers import Bignumbers
13 import gc
```

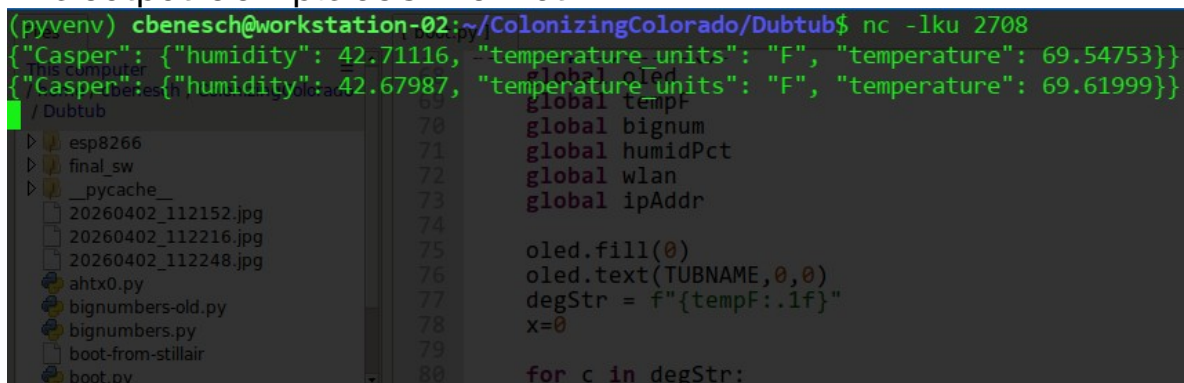
Variable Name	Meaning
TUBNAME	String - The text displayed above the temperature readout (Approx 20 chars)
INTERVAL	Integer - The number of milliseconds to wait between measurements. Eg 15000 - 15 seconds.
WIFI_NETNAME	String - The name of your Wifi network (if desired)
WIFI_PASSWD	String - The password to access your Wifi network

Common intervals

15000	Default - 15 seconds
60000	1 Minute
300000	5 Minutes
3600000	1 Hour
14400000	4 Hours

If you choose to connect your monitor to your wifi, you can then listen on any other connected device for measurement updates via UDP broadcast packets on port 2708. Or modifying the code in boot.py you can change it yourself to use a different port or transmission mechanism. This way was chosen as it is the easiest to work with. Check the <https://www.beneschtech.com> website under the Software section for compatible software to integrate a grow farm or continuous monitoring and historical values.

The output is simple JSON format



```
(pyvenv) cbenesch@workstation-02:~/ColonizingColorado/Dubtub$ nc -lku 2708
{"Casper": {"humidity": 42.71116, "temperature_units": "F", "temperature": 69.54753}}
{"Casper": {"humidity": 42.67987, "temperature_units": "F", "temperature": 69.61999}}
```

The screenshot also shows a file explorer on the left with the following files:

- esp8266
- final_sw
- __pycache__
- 20260402_112152.jpg
- 20260402_112216.jpg
- 20260402_112248.jpg
- ahtx0.py
- bignumbers-old.py
- bignumbers.py
- boot-from-stillair
- boot.py