

joy-it DHT11 Temperature And Humidity Sensor User Manual

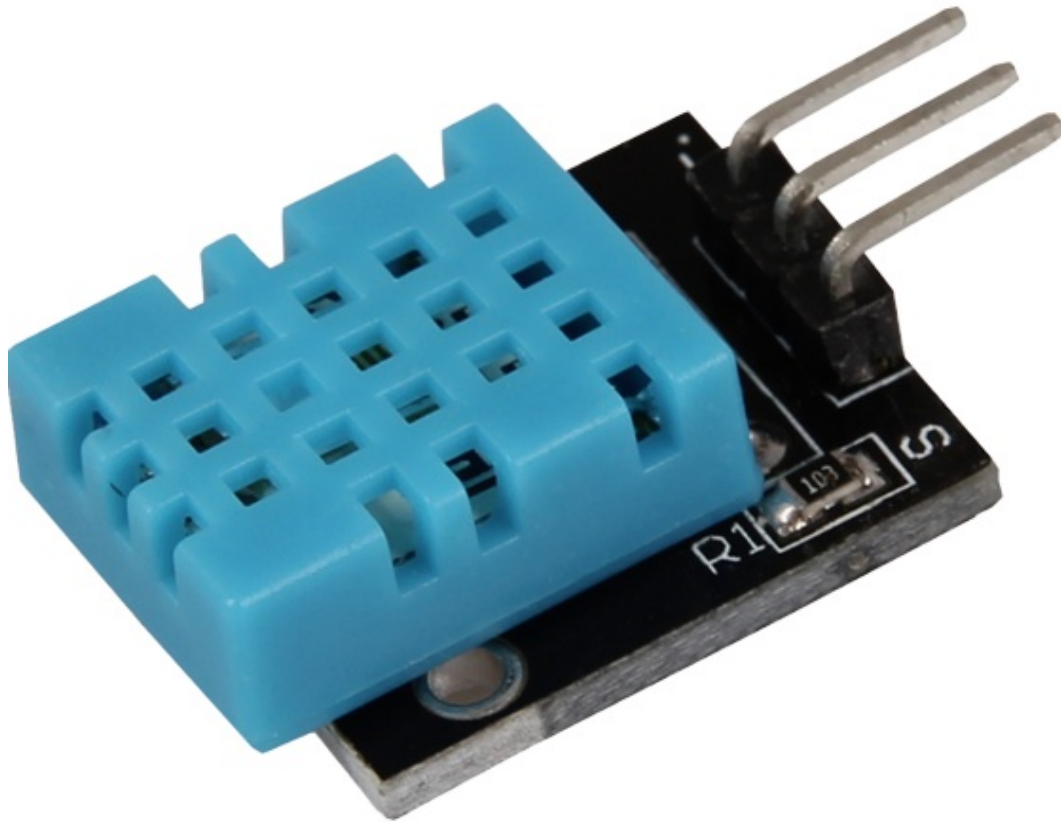
[Home](#) » [JOY-It](#) » joy-it DHT11 Temperature And Humidity Sensor User Manual 

Contents

- 1 joy-it DHT11 Temperature And Humidity Sensor
- 2 Product Information
- 3 Product Usage Instructions
- 4 GENERAL INFORMATION
- 5 BASIC INFORMATION
- 6 USE WITH THE ARDUINO
- 7 USE WITH THE RASPBERRY PI
- 8 USAGE WITH MICRO:BIT
- 9 USAGE WITH RASPBERRY PI PICO
- 10 ADDITIONAL INFORMATION
- 11 SUPPORT
- 12 Documents / Resources
 - 12.1 References



joy-it DHT11 Temperature And Humidity Sensor



Product Information

The product is a temperature and humidity sensor that uses the DHT11 sensor. It can be connected to a microcontroller or single board computer like the Raspberry Pi. The sensor provides accurate temperature and humidity readings.

Product Usage Instructions

For Arduino

1. Include the necessary libraries: `#include <Adafruit_Sensor.h>`, `#include <DHT.h>`, `#include <DHT_U.h>`
2. Define the DHT sensor pin: `#define DHTPIN <pin_number>`
3. Define the DHT sensor type: `#define DHTTYPE DHT11`
4. Initialize the device: `Serial.begin(9600);`
5. Begin the DHT sensor: `dht.begin();`
6. Print temperature sensor details: `Serial.println(F("DHTxx Unified Sensor Example"));`

For Raspberry Pi (MicroPython)

1. Load necessary libraries: `from machine import Pin`, `from utime import sleep`, `from dht import DHT11`
2. Initialize GPIO and DHT11 sensor: `dht11_sensor = DHT11(Pin(14, Pin.IN, Pin.PULL_UP))`
 - Perform measurement and read values: `dht11_sensor.measure()`, `temp = dht11_sensor.temperature()`, `humi = dht11_sensor.humidity()`
3. Output humidity value: `print("Humidity: {:.0f}%".format(humi))`
4. Wait for 3 seconds and repeat the loop

GENERAL INFORMATION

Dear Customer

Thank you for choosing our product. In the following, we will show you what to consider during commissioning and use. If you encounter any unexpected problems during use, please feel free to contact us. The DHT11 is a low-cost and straightforward digital temperature and humidity sensor that is particularly well suited for use with Raspberry Pi and Arduino.

BASIC INFORMATION

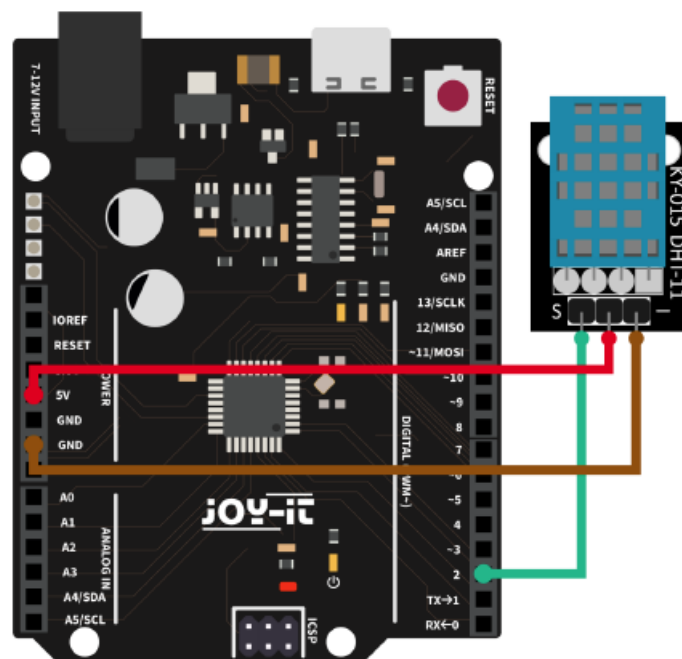
This sensor is a combination of temperature sensor and humidity sensor, united in a compact design. The sampling rate of the measurements is 2 seconds. This sensor is therefore particularly suitable for long-term measurements.

Technical Specifications

- **Chipset:** DHT11
- **Communication protocol:** 1-Wire
- **Measuring range temperature:** 0 °C to 50 °C
- **Measurement accuracy:** $\pm 2^{\circ}\text{C}$
- **Humidity measuring range:** 20-90 %RH
- **Measurement accuracy:** $\pm 50\%$ RH

USE WITH THE ARDUINO

Connection



ASSEMBLY AND INSTALLATION

Arduino/ DHT11

- **Pin D2:** Signal
- **5v:** +V
- **GND:** GND

Code Example

We offer a code example for the use with the Arduino, which you can download [here](#). This sensor does not output its measurement result as an analog signal to an output pin, but communicates it digitally and coded. There are several possibilities to control this sensor module.

The DHT-sensor-library published by the company Adafruit under the MIT-license, has proven to be particularly accessible. You can add the library directly in the Arduino IDE. To do this, click on Sketch → Include Library → Manage Libraries There you can search for the DHT sensor library in the search bar and install it. To use the sensor you now only need to copy the following modified code into your Arduino IDE and upload it to your Arduino. Before you start the code, make sure that the correct board is selected under Tools → Board: and the correct COM port is selected under Tools → Port.

```
#include <Adafruit_Sensor.h>
#include <DHT.h>
#include <DHT_U.h>

#define DHTPIN 2    // Digital pin connected to the DHT sensor

#define DHTTYPE    DHT11    // DHT 11

DHT_Unified dht(DHTPIN, DHTTYPE);

uint32_t delayMS;

void setup() {
  Serial.begin(9600);
  // Initialize device
  dht.begin();
  Serial.println(F("DHTxx Unified Sensor Example"));
  // Print temperature sensor details
  sensor_t sensor;
  dht.temperature().getSensor(&sensor);
  Serial.println(F("-----"));
  Serial.println(F("Temperature Sensor"));
  Serial.print (F("Sensor Type: ")); Serial.println(sensor.name);
  Serial.print (F("Driver Ver:  ")); Serial.println(sensor.version);
  Serial.print (F("Unique ID:   ")); Serial.println
(sensor.sensor_id);
  Serial.print (F("Max Value:   ")); Serial.print(sensor.max_value);
  Serial.println(F("°C"));
```

```

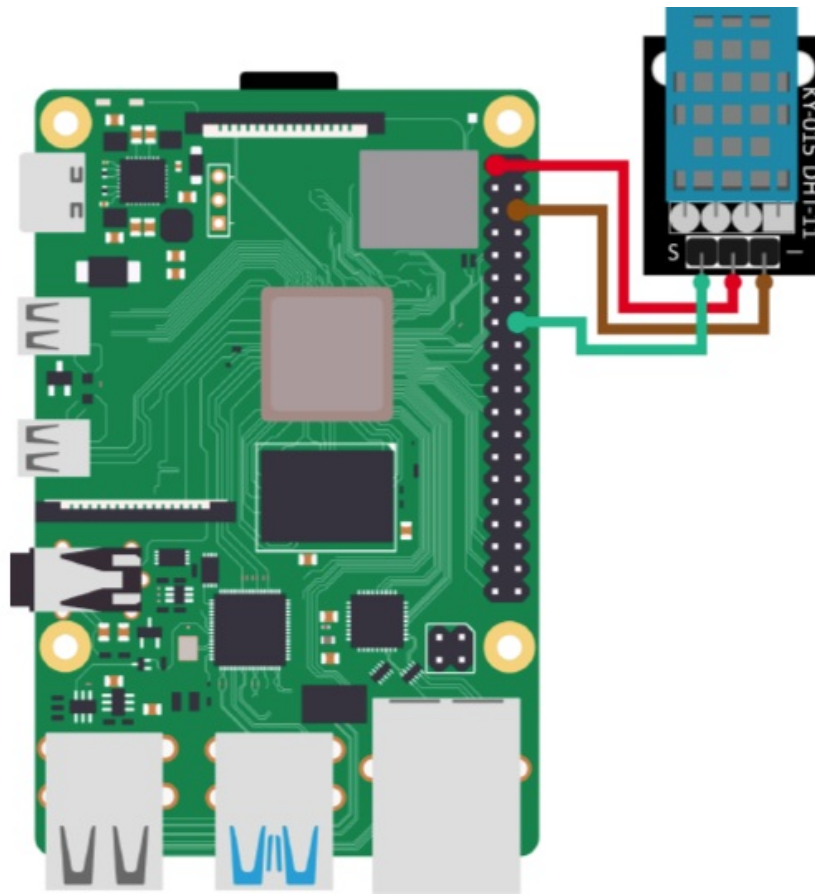
    Serial.print (F("Min Value:  ")); Serial.print(sensor.min_value);
    Serial.println(F("°C"));
    Serial.print (F("Resolution:  ")); Serial.print(sensor.resolution);
    Serial.println(F("°C"));
    Serial.println(F("-----"));
    // Print humidity sensor details
    dht.humidity().getSensor(&sensor);
    Serial.println(F("Humidity Sensor"));
    Serial.print (F("Sensor Type:  ")); Serial.println(sensor.name);
    Serial.print (F("Driver Ver:  ")); Serial.println(sensor.version);
    Serial.print (F("Unique ID:  ")); Serial.println
(sensor.sensor_id);
    Serial.print (F("Max Value:  ")); Serial.print(sensor.max_value);
    Serial.println(F("%"));
    Serial.print (F("Min Value:  ")); Serial.print(sensor.min_value);
    Serial.println(F("%"));
    Serial.print (F("Resolution:  ")); Serial.print(sensor.resolution);
    Serial.println(F("%"));
    Serial.println(F("-----"));
    // Set delay between sensor readings based on sensor details.
    delayMS = sensor.min_delay / 1000;
}

void loop() {
    // Delay between measurements.
    delay(delayMS);
    // Get temperature event and print its value.
    sensors_event_t event;
    dht.temperature().getEvent(&event);
    if (isnan(event.temperature)) {
        Serial.println(F("Error reading temperature!"));
    }
    else {
        Serial.print(F("Temperature:  "));
        Serial.print(event.temperature);
        Serial.println(F("°C"));
    }
    // Get humidity event and print its value.
    dht.humidity().getEvent(&event);
    if (isnan(event.relative_humidity)) {
        Serial.println(F("Error reading humidity!"));
    }
    else {
        Serial.print(F("Humidity:  "));
        Serial.print(event.relative_humidity);
        Serial.println(F("%"));
    }
}

```

USE WITH THE RASPBERRY PI

Connection



Raspberry Pi/ DHT11

- **GPIO 23 (Pin 16):** Signal
- **+3,3 V (Pin 1):** +V
- **GND (Pin 6):** GND

Code example

We provide a code example for use with the Raspberry Pi which you can download [here](#). This code example uses the Adafruit CircuitPython DHT library from Adafruit which is released under the MIT-License.

- `sudo apt-get update`
- `sudo apt-get install build-essential python-dev`
- `sudo apt install gpiod`
- `sudo apt install python3-pip`
- `sudo pip3 install adafruit-circuitpython-dht`

After you have executed the commands, you can also alternatively, if you have not downloaded the code sample, copy the following identical code sample into a newly created file. To do this, you must enter the following command into the console on your Raspberry Pi.

- `nano DHT11.py`

Once you have done this, all you need to do is paste the following code into the file you have just created.

```

import time
import board
import adafruit_dht

# Initialize the DHT with the data pin connected to pin 16 (GPIO 23)
of the Raspberry Pi:
dhtDevice = adafruit_dht.DHT11(board.D23)

# You can pass DHT22 use_pulseio=False if you do not want to use pulseio.
# This may be necessary on a Linux single board computer like the Raspberry Pi, but it will not work in CircuitPython.
# dhtDevice = adafruit_dht.DHT22(board.D18, use_pulseio=False)

while True:
    try:
        # Output of the values via the serial interface
        temperature_c = dhtDevice.temperature
        temperature_f = temperature_c * (9 / 5) + 32
        humidity = dhtDevice.humidity
        print("Temp: {:.1f} F / {:.1f} C    Humidity: {}% ".format(
            temperature_f, temperature_c, humidity))

    except RuntimeError as error:
        # Mistakes happen quite often, DHT's are hard to read, just move on
        print(error.args[0])
        time.sleep(2.0)
        continue
    except Exception as error:
        dhtDevice.exit()
        raise error

    time.sleep(2.0)

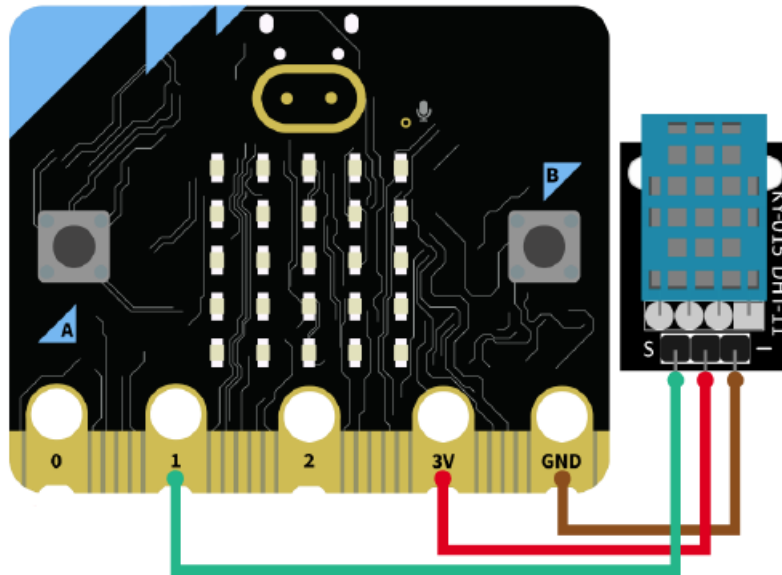
```

You can save the file with CTRL+O and then close it with CTRL+X.

- python3 DHT11.py

USAGE WITH MICRO:BIT

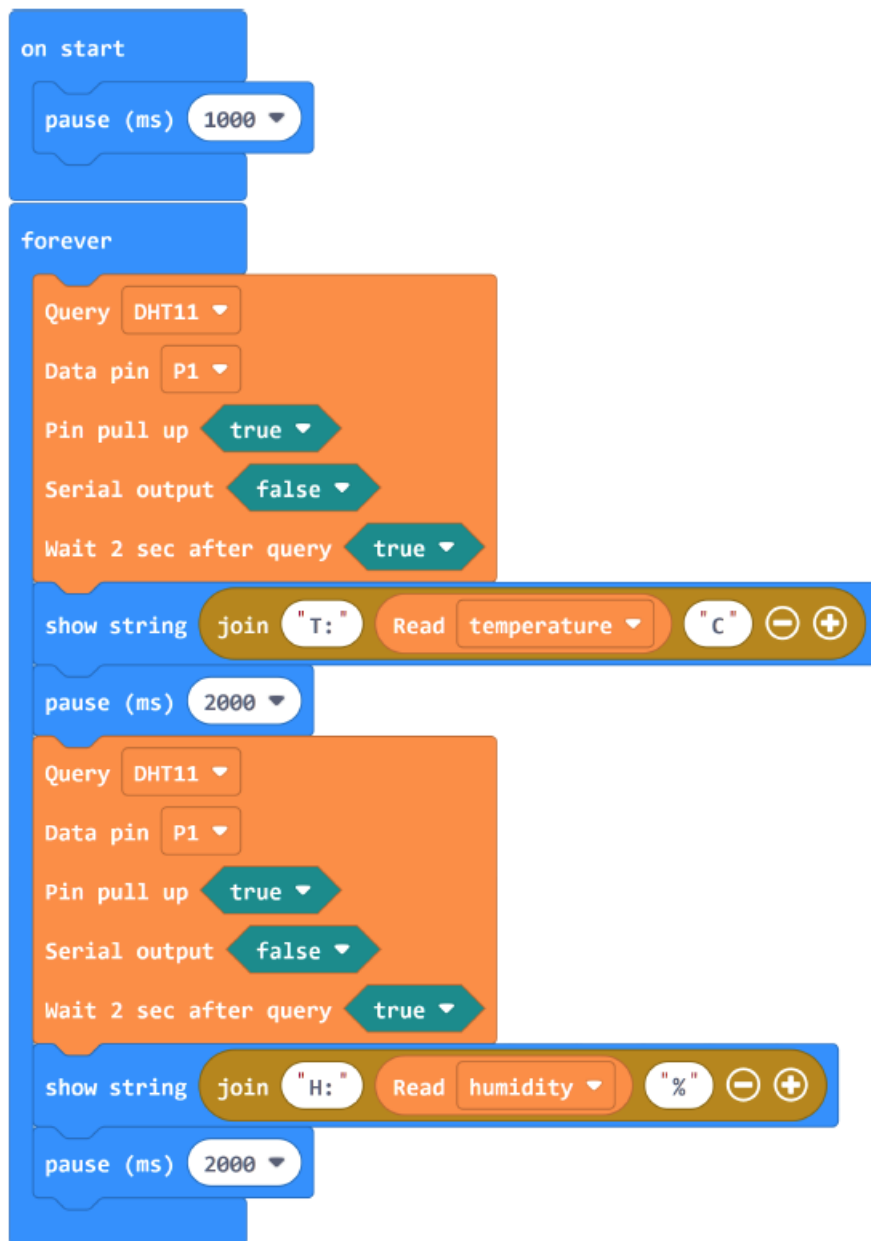
Connection



Micro:Bit: DHT11

- **Pin 1:** Signal
- **+3,3 V:** +V
- **GND:** GND

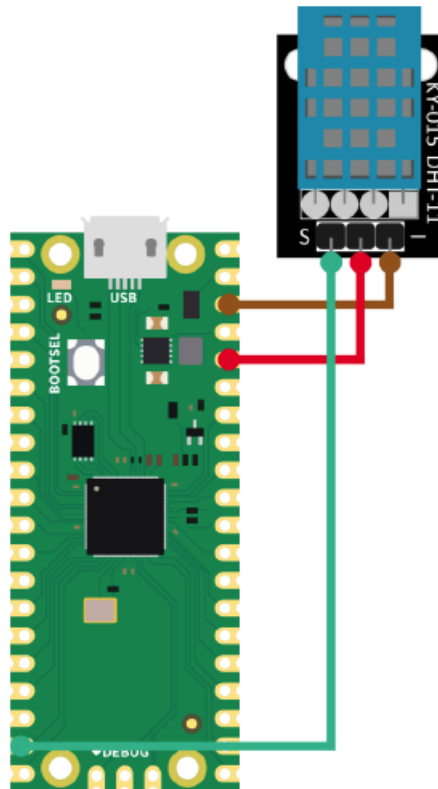
Code example



For the following code example an additional library is needed. To install it, click on “Add extension” in your make code sketch and search for “DHTII” here. Install here the „DHTII DHT22” library by alankrantas. Transfer the following sample code into your sketch, or import the .hex-file.

USAGE WITH RASPBERRY PI PICO

Connection



Raspberry Pi Pico: DHT11

- **GPIO14:** Signal
- **+3,3V:** +V
- **GND:** GND

Code example

For the following code example an additional library is needed: micropython-stubs by Jos Verlinde I published under the MIT license. Download the code sample [here](#) or transfer the following code completely to your Raspberry Pi Pico.

```

# Load libraries
from machine import Pin
from utime import sleep
from dht import DHT11

# Initialization GPIO14 and DHT11
sleep(1)
dht11_sensor = DHT11(Pin(14, Pin.IN, Pin.PULL_UP))

# Repeat (endless loop)
while True:
    # Perform measurement
    dht11_sensor.measure()
    # Read values
    temp = dht11_sensor.temperature()
    humi = dht11_sensor.humidity()
    # Output values
    print("Temperature: {}°C    Humidity: {:.0f}% ".format(temp, humi))
    print()
    sleep(3)

```

ADDITIONAL INFORMATION

Our information and take-back obligations according to the Electrical and Electronic Equipment Act (ElektroG)

Symbol on electrical and electronic equipment

This crossed-out dustbin means that electrical and electronic appliances do not belong in the household waste. You must return the old appliances to a collection point. Before handing over waste batteries and accumulators that are not enclosed by waste equipment must be separated from it.

Return options

As an end user, you can return your old device (which essentially fulfills the same function as the new device purchased from us) free of charge for disposal when you purchase a new device. Small appliances with no external dimensions greater than 25 cm can be disposed of in normal household quantities independently of the purchase of a new appliance.

Possibility of return at our company location during opening hours

- SIMAC Electronics GmbH, Pascalstr. 8, D-47506 Neukirchen-Vluyn, Germany

Possibility of return in your area

- We will send you a parcel stamp with which you can return the device to us free of charge. Please contact us by email at Service@joy-it.net or by telephone.

Information on packaging


- If you do not have suitable packaging material or do not wish to use your own, please contact us and we will send you suitable packaging.

SUPPORT












If there are still any issues pending or problems arising after your purchase, we will support you by e-mail, telephone and with our ticket support system.

- **E-Mail:** service@joy-it.net
- **Ticket system:** <http://support.joy-it.net>
- **Telephone:** +49 (0)2845 9360-50 (10-17 o'clock)
- **For further information please visit our website:** www.joy-it.net

Documents / Resources

	<p>joy-it DHT11 Temperature And Humidity Sensor [pdf] User Manual DHT11 Temperature And Humidity Sensor, DHT11, Temperature And Humidity Sensor, And Humidity Sensor, Humidity Sensor</p>
--	---

References

-  [Pin.in](#)
-  [Adafruit Industries · GitHub](#)
-  [GitHub - adafruit/Adafruit_CircuitPython_DHT: CircuitPython support for DHT11 and DHT22 type temperature/humidity devices](#)
-  [Adafruit_CircuitPython_DHT/LICENSE at main · adafruit/Adafruit_CircuitPython_DHT · GitHub](#)
-  [GitHub - adafruit/DHT-sensor-library: Arduino library for DHT11, DHT22, etc Temperature & Humidity Sensors](#)
-  [DHT-sensor-library/license.txt at master · adafruit/DHT-sensor-library · GitHub](#)
-  [GitHub - alankrantas/pxt-DHT11_DHT22: BBC micro:bit MakeCode extension for DHT11/DHT22 humidity and temperature sensors](#)
-  [micropython-stubs/LICENSE.md at main · Josverl/micropython-stubs · GitHub](#)
-  [GitHub - Josverl/micropython-stubs: Stubs of most MicroPython ports, boards and versions to make writing code that much simpler.](#)
-  [Joy-IT Helpdesk](#)
-  [For Makers and Professionals | Joy-IT](#)