

Raspberry Pi Pico W Base board 用

Plant Env Sensor board サンプルプログラム ユーザーガイド

Version 1.0

2024/7/13



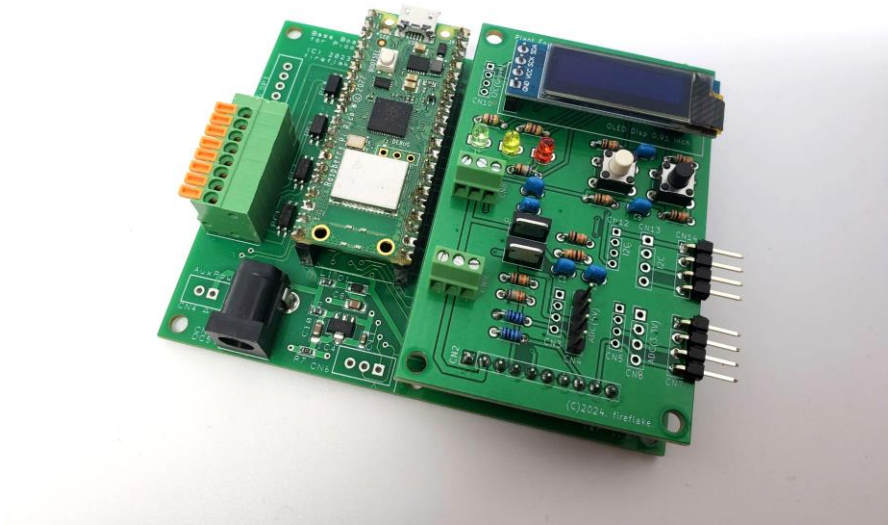
## 目次

1. はじめに .....	3
2. サンプルプログラムの概要 .....	3
3. プログラムのインストール方法 .....	4
3.1. PC から Pico W に接続してプログラムの読み書きが出来るようにする .....	4
3.2. Thonny などの IDE から Pico W にプログラムを書き込む.....	6
4. プログラムの実行方法 .....	9
5. 各プログラムの説明 .....	10
5.1. 単機能のプログラム .....	12
5.1.1. OLED に文字を表示する .....	12
5.1.2. DS18B20 から温度データを取得する .....	12
5.1.3. SEN0562 (BH1750) から照度データを取得する (+それを日射と積算日射データに変換する) ....	12
5.1.4. SEN0308 から土壌水分を取得する (+SEN0308 への電源供給をコントロールする) .....	13
5.1.5. DHT20 から温度データを取得する.....	14
5.1.6. 植物生育環境計測ボード上の LED を光らせる .....	14
5.1.7. 植物生育環境計測ボード上のタクトスイッチが押された事を検知する.....	14
5.2. 総合的なプログラム .....	16
5.2.1. 4 つのセンサーから環境データを取得し、それを OLED に表示する .....	16
5.2.2. 4 つのセンサーから環境データを取得し、それを OLED に表示する& Ambient にアップロードする .....	17
5.2.2.1. サンプルプログラムの説明 .....	17
5.2.2.2. Ambient にデータアップロードするための設定.....	18
6. ライセンスについて .....	22
7. おわりに.....	22

## 1. はじめに

この資料は、fireflake 製植物生育環境計測ボードのためのサンプルプログラム、の説明資料です。

サンプルプログラムを使う際には、お手元に、Raspberry Pi Pico W, fireflake 製ベースボード, fireflake 製植物生育環境計測ボードの 3 点があり、下の画像のように組み立ててあるものとします。



基本的に本資料の情報が正となりますが、サンプルプログラムのバージョンによっては、本資料の説明と動作が異なるかもしれません。その場合にはサンプルプログラムそれ自体を正とします。

## 2. サンプルプログラムの概要

植物生育環境計測ボード用サンプルプログラムは MicroPython で書かれています。MicroPython は組み込み向けに最適化された Python です。Pico W では C/C++ も動作しますが、MicroPython の方が手軽に使う事が出来るため、これを採用しました。お好みに応じて C, C++ を使うことも、もちろん可能です。

サンプルプログラムには、大きく分けて 2 種類のプログラムが含まれています。

ひとつは単機能のプログラムです。これは植物生育環境計測ボード上のセンサーモジュールからデータを取得する、あるいは OLED に文字を表示する、といったシンプルなプログラムです。

もうひとつは総合的なプログラムです。これは各センサーモジュールから取得したデータを OLED に表示する、といったループを繰り返すプログラムです。

どちらも、プログラム本体と関連ライブラリーを Pico W にインストールするだけで動作します。（ただしごく一部のプログラムは、別途設定が必要です。これについては各々のプログラムの説明のところで説明します。）

### 3. プログラムのインストール方法

ここでは、サンプルプログラムを Pico W にインストールする方法を説明します。

#### 3.1. PC から Pico W に接続してプログラムの読み書きが出来るようにする

この方法については、Raspberry Pi 公式サイトの説明が、（英語ですが）分かりやすくまとまっています。

<https://www.raspberrypi.com/documentation/microcontrollers/micropython.html>

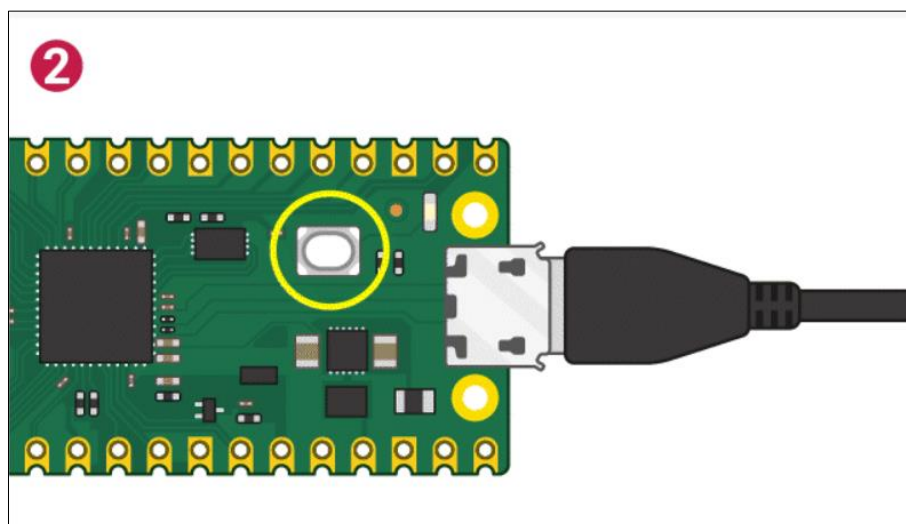
以下は、公式サイトの説明を抜粋したものになります。

まず、公式サイトから UF2 ファイルをダウンロードしてください。「Download the correct MicroPython UF2 file for your board:」と書いてある箇所で「Raspberry Pi Pico W」のダウンロードリンクをクリックしてください。

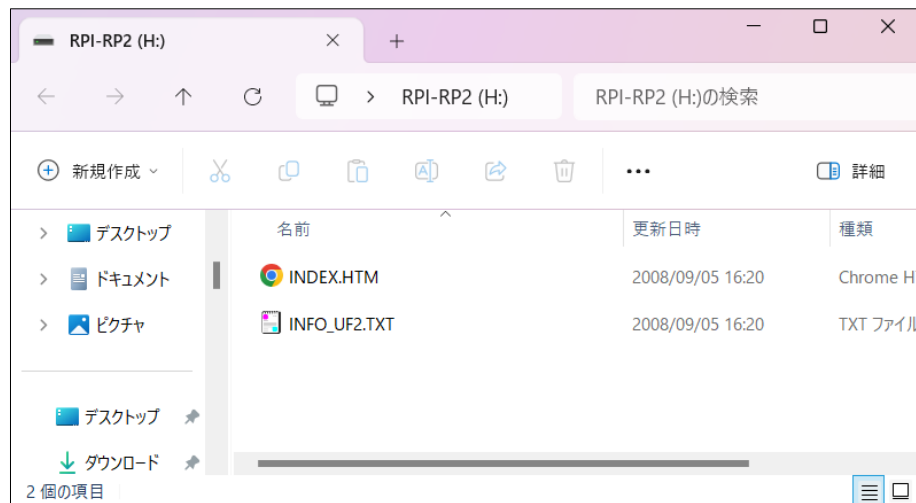
Download the correct MicroPython UF2 file for your board:

- Raspberry Pi Pico
- Raspberry Pi Pico W with Wi-Fi and Bluetooth LE support

Pico W の BOOTSEL ボタンを押したままにして、PC の USB ポートに接続します。接続出来たら、BOOTSEL ボタンをはなします。



Pico W が、RPI-RP2 という大容量ストレージデバイスとして認識されます。Windows の場合、RPI-RP2 のエクスプローラー画面が開きます。



さきほどダウンロードした MicroPython UF2 ファイルを RPI-RP2 のエクスプローラー画面にドラッグ&ドロップします。すると Pico W が再起動して、内部で MicroPython が実行され、USB シリアル経由でプログラムの読み書きと実行ができるようになります。

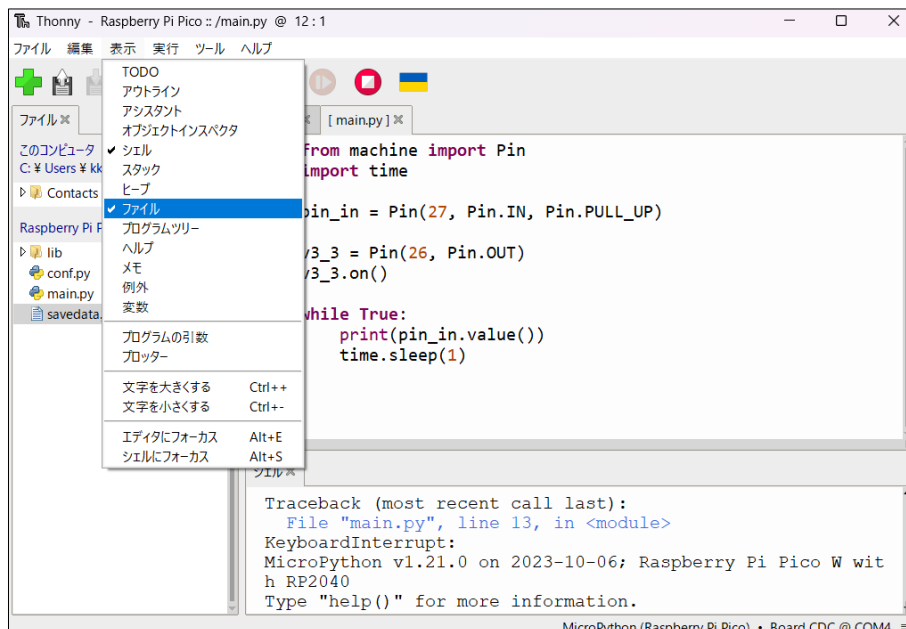
### 3.2. Thonny などの IDE から Pico W にプログラムを書き込む

サンプルプログラムを Pico W に書き込むには、MicroPython 向けの IDE を使うと便利です。ここでは Thonny を例にして説明します。

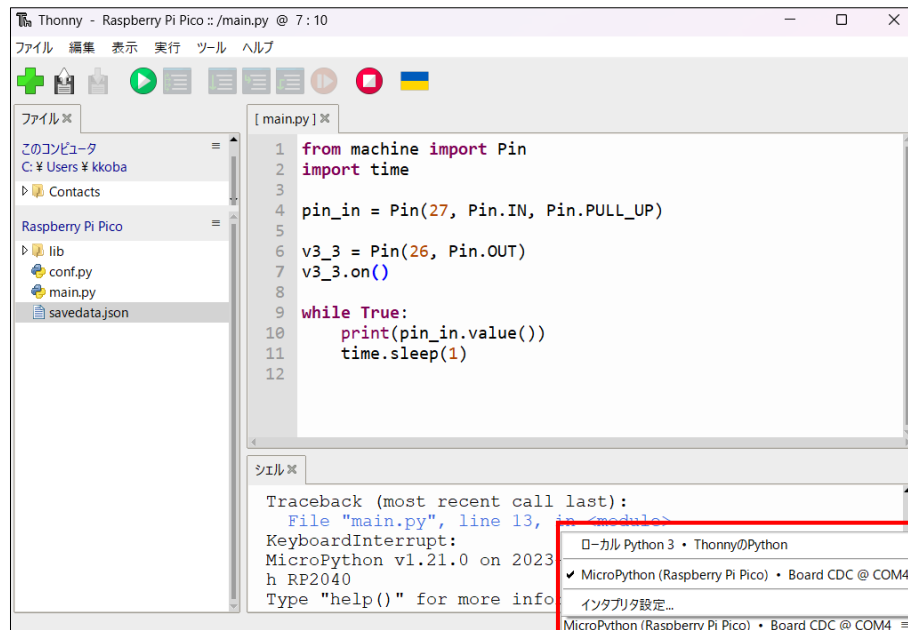
Thonny 公式サイト(<https://thonny.org/>)

Thonny については Web 上に使い方やインストール方法の情報が多くあるので、ここではその説明は行いません。ここでは Thonny のインストールと起動が済んでいるものとして説明します。

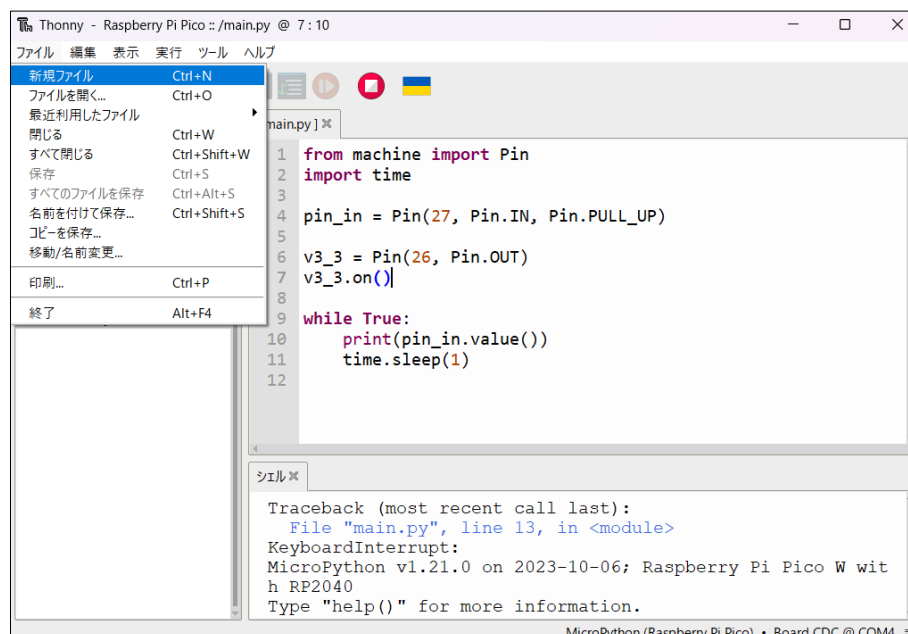
Thonny の表示メニューを開いて、ファイル表示にチェックを入れてください。すると左側にファイルツリーが表示されます。ここでファイルツリーの中に、先ほど PC に接続した Pico W が「Raspberry Pi Pico」として表示されているので、それを確認してください。



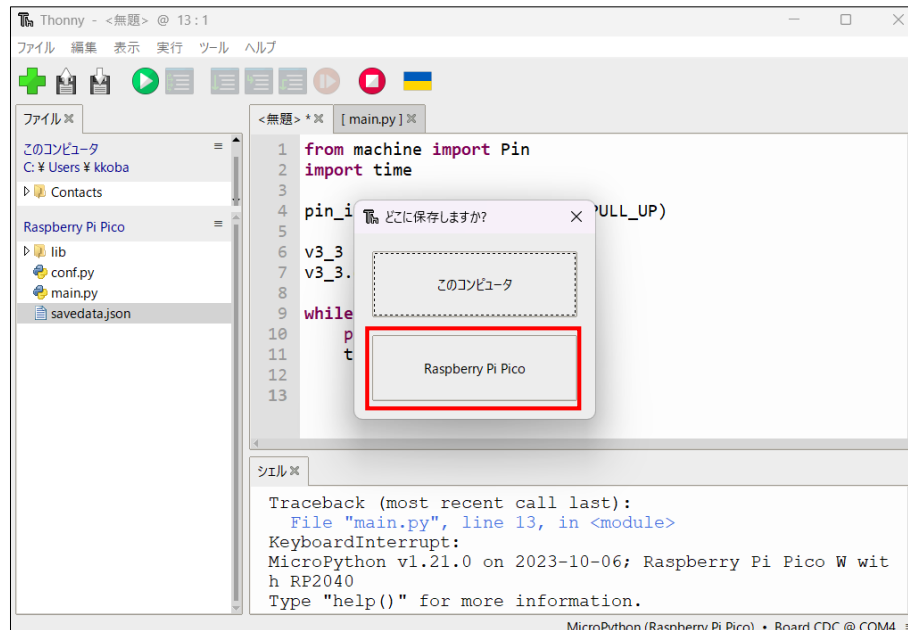
もしもない場合は、Thonny 右下のシリアルポート割り当てメニューを開いて、Pico W が接続されているシリアルポートを選択してください。



Pico W がある事を確認したら、ファイルメニューから「新規ファイル」を選びます。無題のファイルが開くので、インストールしたいサンプルプログラムのコードをそこにコピーします。



無題のファイルを、サンプルプログラムのファイルと同じ名前を付けて Pico W に保存します。これを繰り返して、動かしたいサンプルプログラムのコードをすべて Pico W に保存してください。lib フォルダの作成が必要なプログラムの場合は、ファイルツリーの中で右クリックして「新しいディレクトリ」メニューを選び、lib と名付けて OK ボタンを押すと lib フォルダが出来ます。サンプルプログラムをすべて Pico W に保存したらインストールは完了です。



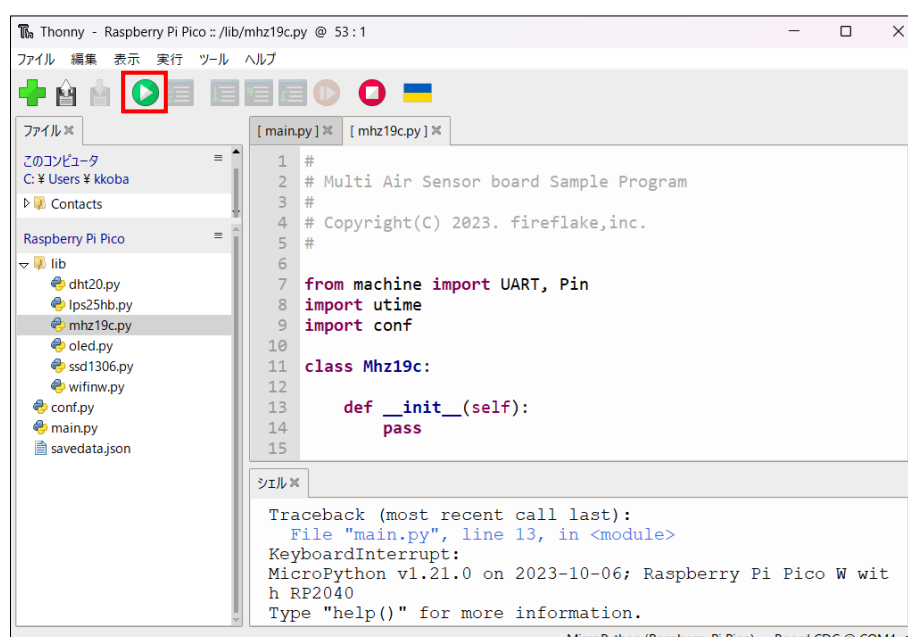


## 4. プログラムの実行方法

ここでは、サンプルプログラムを Pico W 上で実行する方法を説明します。

プログラムの実行には 2 種類の方法があります。ひとつは Pico W と Thonny を接続して、Pico W に入っているプログラムを Thonny で開いて編集して保存し※、その状態で実行ボタンを押す方法です。これを行うと Pico W 上でそのプログラムが実行されます。

※：Pico W に入っているプログラムを Thonny で開いて編集し、保存せずに実行ボタンを押すと、そのプログラムが Pico W のフラッシュメモリに書き込まれたうえで実行されます。ただし複数のファイルを編集してひとつも保存せずに実行ボタンを押すと、フロントに表示されているプログラムだけがフラッシュメモリに書き込まれて実行されます。



もうひとつは、Pico W にただ電源投入する方法です。分かりやすいイメージとしては、Pico W が載ったベースボードに AC アダプタから電源供給して、独立電源で動作させるというものです。これを行うと、自動的に Pico W にインストールされたプログラムが実行されます。この場合、プログラムは必ず main.py が実行されるという制約がある※ので注意してください。またこの方法で動かすと、エラーやログはどこにも表示されません。エラーやログを確認するには、それらを何らかのデバイス（例えば OLED）に表示するなどの仕組みを組んだうえでプログラムを実行する必要があります。

※：boot.py という名前のファイルが Pico W にインストールされていれば、それは main.py より先に動作しますが、べつの目的があって用意された仕組みなので、通常は main.py から実行してください

## 5. 各プログラムの説明

ここからは、各サンプルプログラムの説明をします。サンプルプログラムは、  
ff-pico-examples/plantenvsensors/src/以下に、下記の構成で置いてあります。  
(工事中と記載してあるものは今後置かれる予定です。)

src

└conf.py

└conf\_ambient.py

└main.py

└main\_ambient.py

└lib/

└ambient.py

└bh1750.py

└dht20.py

└dx18x20.py

└oled.py

└plantenv\_adc.py

└plantenv\_led.py

└plantenv\_sw.py

└ssd1306.py

サンプルプログラムの機能は下記になります。

#### ◆単機能のプログラム

センサー系のサンプルプログラムには、植物生育環境計測ボードにデフォルトで搭載する事を想定している DS18B20, SEN0308 ,SEN0562（センサーモジュール自体は BH1750）, DHT20 用のサンプルプログラムを入れています。

- OLED に文字を表示する
- DS18B20 から温度を取得する
- SEN0562（BH1750）から照度を取得する（+それを日射データ、積算日射データに変換する）
- SEN0308 から土壌水分を取得する（+SEN0308 への電源供給をコントロールする）
- DHT20 から温度を取得する
- LED を光らせる
- タクトスイッチが押された事を検知する

#### ※注意

SEN0562 などの上記の製品名は、植物生育環境計測ボードに載る事が想定されるモジュールの正式名称を正確に表していない場合があります。モジュールの正式名称や、それが買える EC サイトについては「**植物生育環境計測ボード仕様書**」をご確認ください。正しくないモジュールを植物生育環境計測ボードに接続した場合、サンプルプログラムは正しく動作しない場合があるのでご注意ください。

#### ◆総合的なプログラム

- 4 つのセンサーから植物生育環境データを取得し、それを OLED に表示する
- 4 つのセンサーから植物生育環境データを取得し、それを OLED に表示する & Ambient にアップロードする

## 5.1. 単機能のプログラム

### 5.1.1. OLED に文字を表示する

- このプログラムは、128x32 ドット、0.91 インチの有機 EL ディスプレイ（OLED）に文字を表示するプログラムです。OLED には最大で 4 行の表示が行えます。Pico W と OLED は I2C で通信します。
- Thonny から動作させる場合、下記の構成でサンプルプログラムを Pico W にインストールし、oled.py を開いている画面で、Thonny の実行ボタンを押してください。

.Raspberry Pi Pico（Pico W は Thonny 上ではこのように表記されます）

└oled.py

└ssd1306.py

- 独立電源で動作させる場合は、上記の構成でサンプルプログラムを Pico W にインストールし、oled.py のファイル名を main.py に変えたうえで、Pico W に電源投入してください。

### 5.1.2. DS18B20 から温度データを取得する

- このプログラムは、温度センサーの DS18B20 から温度を取得するプログラムです。Pico W と DS18B20 は 1-wire という方式で通信します。
- Thonny から動作させる場合、下記の構成でサンプルプログラムを Pico W にインストールし、lps25hb.py を開いている画面で、Thonny の実行ボタンを押してください。

.Raspberry Pi Pico（Pico W は Thonny 上ではこのように表記されます）

└ds18b20.py

- 独立電源で動作させる場合は、上記の構成でサンプルプログラムを Pico W にインストールし、ds18b20.py のファイル名を main.py に変えたうえで、Pico W に電源投入してください。ただこれをして、DS18B20 から取得したデータを OLED に表示する仕組みなどがなければ、見た目上は何も起きません。

### 5.1.3. SEN0562（BH1750）から照度データを取得する（+それを日射と積算日射データに変換する）

- このプログラムは、照度センサーの SEN0562（BH1750）から照度を取得するプログラムです。Pico W と SEN0562（BH1750）は I2C で通信します。
- このプログラムは、BH1750 から照度データを取得します。またそのデータを加工して、日射量や、積算日射量として表現します。
- Thonny から動作させる場合、下記の構成でサンプルプログラムを Pico W にインストールし、bh1750.py を開いている画面で、Thonny の実行ボタンを押してください。

.Raspberry Pi Pico（Pico W は Thonny 上ではこのように表記されます）

└bh1750.py

- 独立電源で動作させる場合は、上記の構成でサンプルプログラムを Pico W にインストールし、bh1750.py のファイル名を main.py に変えたうえで、Pico W に電源投入してください。ただこれにしても、SEN0562（BH1750）から取得したデータを OLED に表示する仕組みなどがなければ、見た目上は何も起きませ

#### 5.1.4. SEN0308 から土壌水分を取得する（+ SEN0308 への電源供給をコントロールする）

- このプログラムは、土壌水分センサーの SEN0308 から土壌水分を取得するプログラムです。SEN0308 は電圧出力をするセンサー（アナログセンサー）なので、Pico W はそれを ADC（アナログデジタルコンバーター）で受け取ります。
- このプログラムは、SEN0308 への電源供給コントロールを含みます。これは土壌水分センサーに常時通電する事によって、土壌水分センサー周辺の土中の塩類が、土壌水分センサーのそばに集まり、計測結果が狂うのを多少なりとも避けるためです。また、植物生育環境計測ボードは、SEN0308 以外のアナログセンサーの接続も想定していますので、この電源供給コントロール機能は、植物生育環境計測ボードに接続するアナログセンサーであれば使用できます。
- Thonny から動作させる場合、下記の構成でサンプルプログラムを Pico W にインストールし plantenv\_adc.py を開いている画面で、Thonny の実行ボタンを押してください。

.Raspberry Pi Pico（Pico W は Thonny 上ではこのように表記されます）

└plantenv\_adc.py

- 独立電源で動作させる場合は、上記の構成でサンプルプログラムを Pico W にインストールし、sen0308.py のファイル名を main.py に変えたうえで、Pico W に電源投入してください。ただこれにしても、SEN0308 から取得したデータを OLED に表示する仕組みなどがなければ、見た目上は何も起きません。

#### ※SEN0308 のキャリブレーションについて

土壌水分センサーは、どの土壌でも万能に土壌水分データを測定できるわけではありません。出来るだけ正確に土壌水分を測定するのであれば、まさに測定する土壌に合わせた「土壌水分センサーの出力電圧を、実際の土壌水分の値に変換する式」を作成する事が必須となります。サンプルプログラムにはごく簡易的な変換式を入れていますが、より正確に土壌水分を測定したい場合は、下記の資料などを参考にしながら、変換式を自作してプログラムに組み込んでください。

参考：土壌水分センサーの簡易校正法

<http://soillab.ag.saga-u.ac.jp/com/calib1.pdf>

#### 5.1.5. DHT20 から温度データを取得する

- このプログラムは、温度センサーの DHT20 から温度を取得するプログラムです。Pico W と DHT20 は I2C という方式で通信します。また、植物生育環境計測ボードを使う場合は DHT20 はベースボードの I2C チャンネルに接続する事になりますので、このプログラムもそのように組んでいます。
- Thonny から動作させる場合、下記の構成でサンプルプログラムを Pico W にインストールし、lps25hb.py を開いている画面で、Thonny の実行ボタンを押してください。

.Raspberry Pi Pico (Pico W は Thonny 上ではこのように表記されます)

└dht20.py

- 独立電源で動作させる場合は、上記の構成でサンプルプログラムを Pico W にインストールし、ds18b20.py のファイル名を main.py に変えたうえで、Pico W に電源投入してください。ただこれをして、DS18B20 から取得したデータを OLED に表示する仕組みなどがなければ、見た目上は何も起きません。

#### 5.1.6. 植物生育環境計測ボード上の LED を光らせる

- このプログラムは、植物生育環境計測ボード上の 3 つの LED を光らせるプログラムです。
- Thonny から動作させる場合、下記の構成でサンプルプログラムを Pico W にインストールし、multiair\_led.py を開いている画面で、Thonny の実行ボタンを押してください。

.Raspberry Pi Pico (Pico W は Thonny 上ではこのように表記されます)

└plantenv\_led.py

- 独立電源で動作させる場合は、上記の構成でサンプルプログラムを Pico W にインストールし、multiair\_led.py のファイル名を main.py に変えたうえで、Pico W に電源投入してください。

#### 5.1.7. 植物生育環境計測ボード上のタクトスイッチが押された事を検知する

- このプログラムは、植物生育環境計測ボード上の 2 つのタクトスイッチが押された事を検知し、押された場合はコンソールにそれを表示するプログラムです。
- Thonny から動作させる場合、下記の構成でサンプルプログラムを Pico W にインストールし、multiair\_sw.py を開いている画面で、Thonny の実行ボタンを押してください。

.Raspberry Pi Pico (Pico W は Thonny 上ではこのように表記されます)

└plantenv\_sw.py

- 独立電源で動作させる場合は、上記の構成でサンプルプログラムを Pico W にインストールし、multiair\_sw.py のファイル名を main.py に変えたうえで、Pico W に電源投入してください。ただこれをして、タクトスイッチが押された事を検知してそれを OLED に表示する仕組みなどがなければ、見た目上は何も起きません。

## 5.2. 総合的なプログラム

### 5.2.1. 4つのセンサーから環境データを取得し、それを OLED に表示する

- このプログラムは、タイトルにあるように「3つのセンサーから環境データを取得し、それを OLED に表示する」動作を行う総合的なプログラムです。
- このプログラムは、処理の流れが分かりやすいように、ある程度シンプルに組んであります。ですので長期稼働のための実用的な機構は含めていません。**エラーが出力された場合は、再起動などの処理はせずそこで停止します。**
- このプログラムは、周期的に4つのセンサーからデータを取得して OLED に表示する事を繰り返します。初期の動作周期は60秒です。
- このプログラムには、conf.py という設定用ファイルが含まれます。conf.py に含まれる設定は I2C チャンネル情報、I2C デバイスアドレス、ADC チャンネル情報、ループ待ち時間などです。conf.py は初期設定のまま使う事が可能です。必要があれば編集して使ってください。
- Thonny から動作させる場合、下記の構成でプログラムを Pico W にインストールし、main.py を開いている画面で、Thonny の実行ボタンを押してください。

.Raspberry Pi Pico (Pico W は Thonny 上ではこのように表記されます)

```
├conf.py
├main.py
└lib/
  ├──bh1750.py
  ├──dht20.py
  ├──dx18x20.py
  ├──oled.py
  └ssd1306.py
```

- 独立電源で動作させる場合は、上記の構成でプログラムを Pico W にインストールし、Pico W に電源投入してください。



## 5.2.2. 4つのセンサーから環境データを取得し、それを OLED に表示する & Ambient にアップロードする

### 5.2.2.1. サンプルプログラムの説明

- このプログラムは、無償でも使える IoT データの可視化サービスの Ambient と連携し、そこにセンサーから取得した環境データをアップロードするプログラムです。Ambient については、下記公式サイト「ドキュメント」メニューから多くの情報にアクセス出来ます。また Ambient は、fireflake にとって外部のサービスになるため、それに関するご質問などをいただいても応じられない場合があります。すみませんが、その点はご了承ください。

Ambient 公式サイト (<https://ambidata.io/>)

- このプログラムは、タイトルにあるように「4つのセンサーから環境データを取得し、それを OLED に表示 & Ambient にアップロードする」動作を行う総合的なプログラムです。
- このプログラムは、処理の流れが分かりやすいように、ある程度シンプルに組んであります。ですので長期稼働のための実用的な機構は含めていません。エラーが出力された場合は、再起動などの処理はせずそこで停止します。
- このプログラムには、conf.py という設定用ファイルが含まれます。conf.py に含まれる設定は I2C チャンネル情報、I2C デバイスアドレス、ADC チャンネル情報、ループ待ち時間、Wi-Fi の SSID とパスワード、Ambient の接続情報などです。このプログラムでは、conf.py は初期設定のまま使う事が出来ません。プログラムを正しく動かすためには、Wi-Fi 情報の設定と、Ambient 接続情報の設定が必要になります。詳しくは「Ambient にデータアップロードするための設定」を読んでください。
- Thonny から動作させる場合、下記の構成でプログラムを Pico W にインストールし、main.py を開いている画面で、Thonny の実行ボタンを押してください。

.Raspberry Pi Pico (Pico W は Thonny 上ではこのように表記されます)

└main.py (main\_ambient.py を左記に名称変更してインストールしてください)

└conf.py (conf\_ambient.py を左記に名称変更してインストールしてください)

└ lib/

└ambient.py

└bh1750.py

└dht20.py

└dx18x20.py

└oled.py

└ssd1306.py

- 独立電源で動作させる場合は、上記の構成でプログラムを Pico W にインストールし、Pico W に電源投入してください。

## 5.2.2.2. Ambient にデータアップロードするための設定

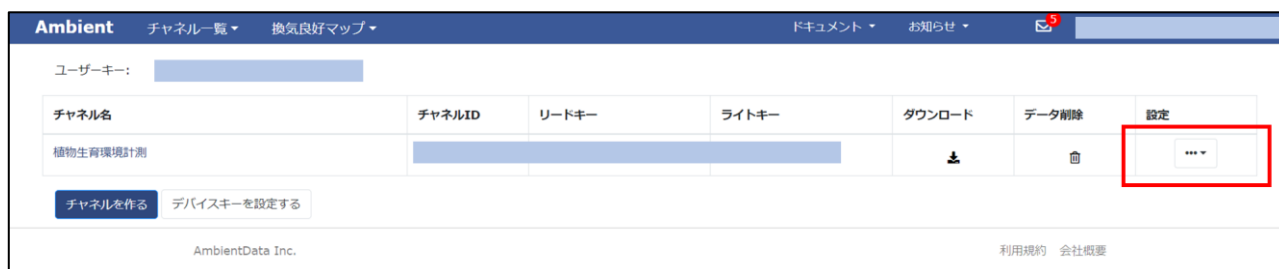
測定した環境データを Ambient にアップロードするためには、Ambient 側と Pico W 側の双方の設定をする必要があります。

### ◆ Ambient 側の設定

ここでの説明は粗いものになります。詳細は、下記のチュートリアルなど Ambient の公式ドキュメントを参照してください。

<https://ambidata.io/docs/>

1. Ambient にユーザー登録を行い、チャンネルをひとつ登録してください。
2. チャンネルを登録したら、設定のプルダウンリストをクリックし、表示される設定変更をクリックしてください。



The screenshot shows the Ambient web interface. At the top, there's a navigation bar with 'Ambient', 'チャンネル一覧', '換気良好マップ', 'ドキュメント', and 'お知らせ'. Below this, a user login section is visible. The main area displays a table of channels. The first channel is '植物生育環境計測'. The table has columns for 'チャンネル名', 'チャンネルID', 'リードキー', 'ライトキー', 'ダウンロード', 'データ削除', and '設定'. The '設定' column for the first channel has a dropdown menu icon, which is highlighted with a red box. Below the table, there are buttons for 'チャンネルを作る' and 'デバイスキーを設定する'. At the bottom, it says 'AmbientData Inc.' and '利用規約 会社概要'.

3. 表示される設定画面に下記のように入力し、チャンネル属性を設定するボタンを押してください。



The screenshot shows the 'Myチャンネル / 植物生育環境計測' settings page. The page title is 'Myチャンネル / 植物生育環境計測 ( ) / チャンネル設定'. The form includes the following fields and options:

- チャンネル名:** 植物生育環境計測
- 説明:** 説明
- データ1:** 日射量 (Blue color selection button)
- データ2:** 積算日射量 (Red color selection button)
- データ3:** 土壌温度 (Orange color selection button)
- データ4:** 土壌水分 (Purple color selection button)
- データ5:** 気温 (Green color selection button)
- データ6:** 湿度 (Cyan color selection button)
- データ7:** データー名7 (Pink color selection button)
- データ8:** データー名8 (Brown color selection button)
- 緯度:** 0
- 経度:** 0

4. チャネル名をクリックして、チャネル内部情報の画面に移動してください。



5. チャート作成ボタンを押して、1 つめのチャートを作ります。



6. チャートの設定は下記のようにします。作成例なので好きにカスタマイズしてください。設定が出来たら、設定するボタンを押して保存します。

### 日射



## 土壌

チャート設定変更

×

チャンネルデータ

位置

写真

チャート名

土壌

チャンネル

植物生育環境計測 ( )

透過度

z-index

チャート種類

折れ線グラフ

d1:日射量

☐ 表示なし
☐ 左軸
☐ 右軸

d2:積算日射量

☐ 表示なし
☐ 左軸
☐ 右軸

d3:土壌温度

☐ 表示なし
☒ 左軸
☐ 右軸

d4:土壌水分

☐ 表示なし
☐ 左軸
☒ 右軸

d5:気温

☐ 表示なし
☐ 左軸
☐ 右軸

d6:湿度

☐ 表示なし
☐ 左軸
☐ 右軸

d7

☐ 表示なし
☐ 左軸
☐ 右軸

d8

☐ 表示なし
☐ 左軸
☐ 右軸

左軸

最小値

-10

最大値

50

log?

☐

右軸

最小値

0

最大値

100

log?

☐

軸の最小値、最大値は空白のままにすれば自動的に設定されます。

表示件数

300

日付指定

☐

集計

の

設定せずに閉じる

設定する

## 空気

チャート設定変更

×

チャンネルデータ

位置

写真

チャート名

空気

チャンネル

植物生育環境計測 ( )

透過度

z-index

チャート種類

折れ線グラフ

d1:日射量

☐ 表示なし
☐ 左軸
☐ 右軸

d2:積算日射量

☐ 表示なし
☐ 左軸
☐ 右軸

d3:土壌温度

☐ 表示なし
☐ 左軸
☐ 右軸

d4:土壌水分

☐ 表示なし
☐ 左軸
☐ 右軸

d5:気温

☒ 表示なし
☐ 左軸
☐ 右軸

d6:湿度

☐ 表示なし
☐ 左軸
☒ 右軸

d7

☐ 表示なし
☐ 左軸
☐ 右軸

d8

☐ 表示なし
☐ 左軸
☐ 右軸

左軸

最小値

-10

最大値

50

log?

☐

右軸

最小値

0

最大値

100

log?

☐

軸の最小値、最大値は空白のままにすれば自動的に設定されます。

表示件数

300

日付指定

☐

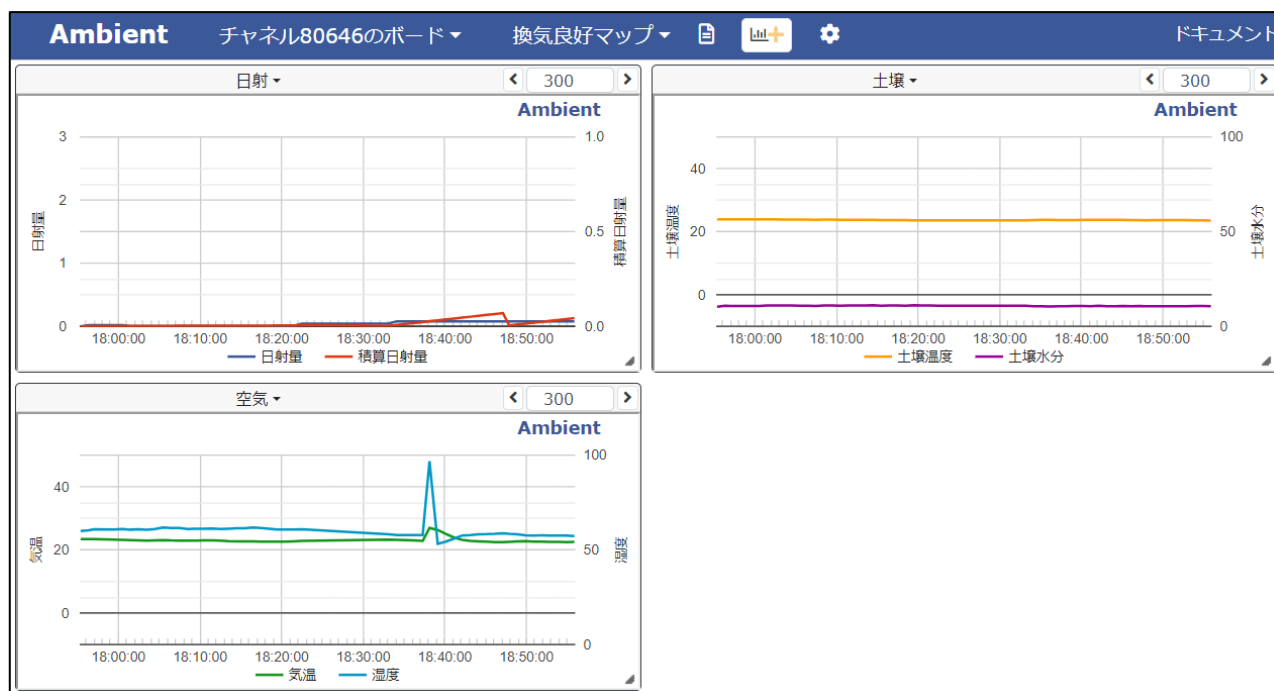
集計

の

設定せずに閉じる

設定する

7. データアップロードをして、このようにチャートが作成できれば成功です。（この画像は作成後しばらく経っているのデータが多少アップロードされています。）



#### ◆Pico W 側の設定

conf.py（conf\_ambient.py を左記に名称変更したもの）に、Pico W が Wi-Fi 接続するための値と、Ambient にデータアップロードするためのを追記します。以下の値を文字列として追記してください。

#### # Ambient 関連

```

AMBIENT_CHANNEL_ID = const("")    ←Ambient のチャンネル ID を入れてください。
AMBIENT_WRITE_KEY = const("")    ←Ambient のライトキーを入れてください。
WIFI_SSID = const("")            ←Pico W を接続する Wi-Fi の SSID を入れてください。
WIFI_PASSWORD = const("")        ←Pico W を接続する Wi-Fi のパスワードを入れてください。

```

これらの設定を行ったうえで、main.py（main\_ambient.py を左記に名称変更したもの）を動作させると、測定した空気環境データを Ambient にアップロードする事が出来ます。

## 6. ライセンスについて

サンプルプログラムのうち、fireflake 制作のプログラムは MIT ライセンスで配布されます。それ以外のプログラムは、それぞれのプログラムが配布されているライセンスに従います。fireflake 制作のプログラムは下記になります。

```
main.py
main_ambinet.py
conf.py
conf_ambient.py
bh1750.py
dht20.py
dx18x20.py
oled.py
plantenv_adc.py
plantenv_led.py
plantenv_sw.py
```

## 7. おわりに

本資料が、皆さんが作る事を楽しめる一助になれば幸いです。Happy Hacking!!