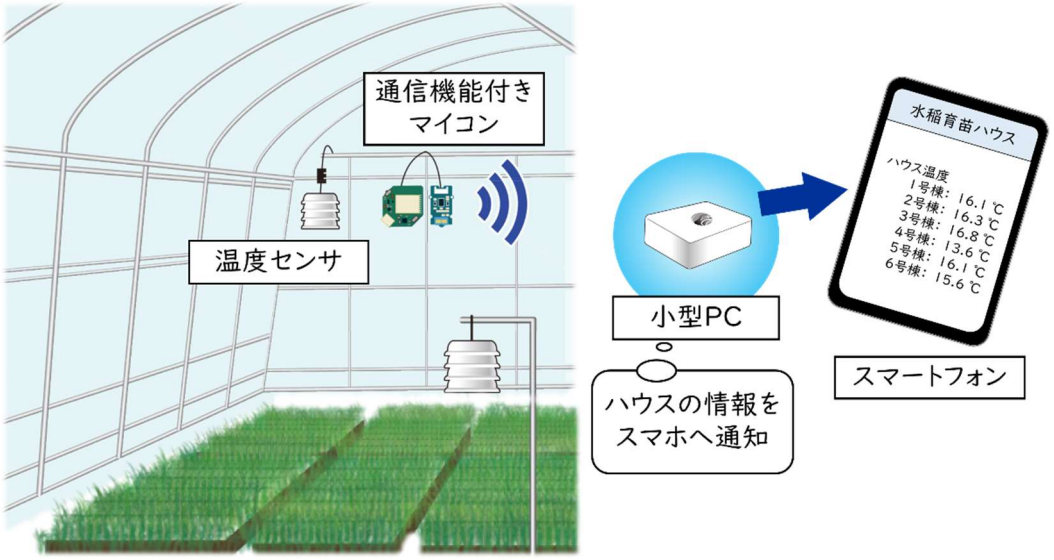
SOP23-202aK

禁転載

安価かつ簡便に
ハウス環境を遠隔監視できる
通い農業支援システム
標準作業手順書

HP 公開版



NARO 農研機構

改訂履歴

版　数	発行日	改訂者	改訂内容
第 1 版	2024 年 3 月 6 日	川口　健太郎	初版発行

最終更新日　2024 年 3 月 6 日

目次

はじめに **1**

免責事項 2

Ⅰ．通い農業支援システムの全体構成 **3**

1.通い農業支援システムの構成 3

2.通い農業支援システムの技術的な仕組み 4

（1）データの取得方法の仕組みと自動化について 4

（2）データ通知の仕組みについて 6

（3）通知するデータの種類 7

3.通い農業支援システムのコスト 7

（1）導入コスト 7

（2）ランニングコスト 9

（3）導入イメージ 10

Ⅱ．無線通信機能付きマイコンの導入 **11**

1.部品リスト 11

2.無線通信機能付きマイコンの製作と設定 11

（1）必要な材料の用意と確認 11

（2）無線通信機能付きマイコンの組立 12

（3）設定するための準備 12

（4）スマートフォンにアプリをインストールする 12

（5）アプリから Wio Node を登録する 15

（6）温度センサを Wio Node に登録する 20

（7）温度を測定する 21

（8）アクセストークンを含むデータ取得用 URL を保存する 22

（9）JSON 形式データの利用方法 23
（10）防水温度センサ以外のセンサを利用する方法 24
Ⅲ．スマートフォンのメッセージアプリの設定 26
（1）【スマートフォンで】事前準備 26
（2）【スマートフォンで】通知先の設定 26
（3）【Raspberry Pi で】LINE Notify にアクセスして、トークンを発行 26
（4）【スマートフォンで】通知したい LINE グループに LINE Notify を招待する 29
Ⅳ．小型パソコン Raspberry Pi でのプログラム 30
1.Raspberry Pi の設定 30
（1）【Windows パソコンで】Raspberry Pi OS ファイルをダウンロードする 30
（2）【Windows パソコンで】Raspberry Pi Imager をダウンロードする 31
（3）【Windows パソコンで】Raspberry Pi OS をインストールする 31
（4）【Raspberry Pi で】Raspberry Pi の設定を行う 34
（5）【Raspberry Pi で】グラフ通知プログラムを使用するための設定 36
（6）【Raspberry Pi で】プログラム言語 Python を使用するための設定 38
2.配布プログラムのダウンロード 40
（1）ブラウザ（Chromium）を開いて、以下の URL にアクセスする 40
（2）プログラムをダウンロードする 41
（3）ダウンロードしたプログラムを確認する 41
3.配布プログラムの設定 43
（1）事前準備 43
（2）データ通知プログラムを作成する 44
（3）グラフ通知プログラムを作成する 53
（4）警報通知プログラムを作成する 62
4.プログラムの自動実行設定 64
（1）crontab の初期設定を行う 64

（2）crontab を使ってプログラムの実行間隔を設定する 65
Ⅴ．現地での設置方法 69
1.Wio Node の防水方法 69
（1）半年程度の短期間使用する場合（簡易な方法） 69
（2）長期間使用する場合 71
2.100V 電源からの設置方法 71
（1）ハウス内のコンセント近くで利用する場合 71
（2）ハウス内のコンセントから遠い場所で利用する場合 72
（3）電源から設置場所が遠い場合の設置方法（USB ケーブルの延長） 73
3.温度センサの設置方法 75
（1）日よけを利用して温度を測定する方法 75
（2）強制通風筒を利用して温度（温湿度）を測定する方法 78
（3）温度センサ使用時の注意点 79
4.土壌水分センサの設置方法 80
用語解説 81
導入生産者の評価 82
参考資料 85
その他情報 86
担当窓口、連絡先 86

「はじめに」

東京電力福島第一原子力発電所事故から 10 年以上が経過し、被災地では大規模水稲生産法人やハウス農家が先行して営農を再開していますが、労働力の確保や担い手不足の問題が特に深刻です。また、生産者の管理するハウスの分散や、居住地から離れた場所へ「通い農業（参照 p.82）」を行っている事例もあることから、遠隔地でのハウス等農業施設を利用した農業再開のためには、リモートで生産現場の状況を確認可能な支援策が必要です。一方で、市販のハウス遠隔監視システムは通年で運用することを前提としており、水稲の育苗などの一時的な利用には適しておらず、被災地での運用はコスト的に見合わないことから、省力化のためにハウス遠隔監視システムの導入を検討している生産者も気軽に試すことができませんでした。即ち、「通い農業」を持続的に可能とする技術として、安価かつ簡便に運用できる IoT 機器が求められていました。

近年、スマート農業を支える IoT 技術に用いられるマイコンやセンサが手軽に入手・利用できるようになり、遠隔監視システムを安価に自作する一部生産者も現れています。しかし、依然として生産者にはハードルが高い状況です。そこで、生産者が安価に製作でき、スマートフォンで気軽にハウス内の状況を確認できる遠隔監視システムである「通い農業支援システム」を開発しました。

通い農業支援システムはモジュール化された簡便な IoT 機器を利用して、離れたハウスの温度等をスマートフォンで遠隔監視が可能です。ハウスごとに通信機能付きマイコンと温度センサを設置して遠隔監視システムを構築するには、IoT に関する知識や WEB サーバ（参照 p.82）の構築またはクラウドサービスの契約が必要であり、容易には作成できません。そこで、データ取得用・メッセージ通知用のサービス（Web API（参照 p.82））を利用することで、パソコンやスマートフォンを操作できる知識などがあれば利用できる、ハウス内温度等の遠隔監視を「安価かつ簡便に」実現するシステムを提案しています。

本手順書では誰もが通い農業を実現できるハウス遠隔監視システムを紹介します。

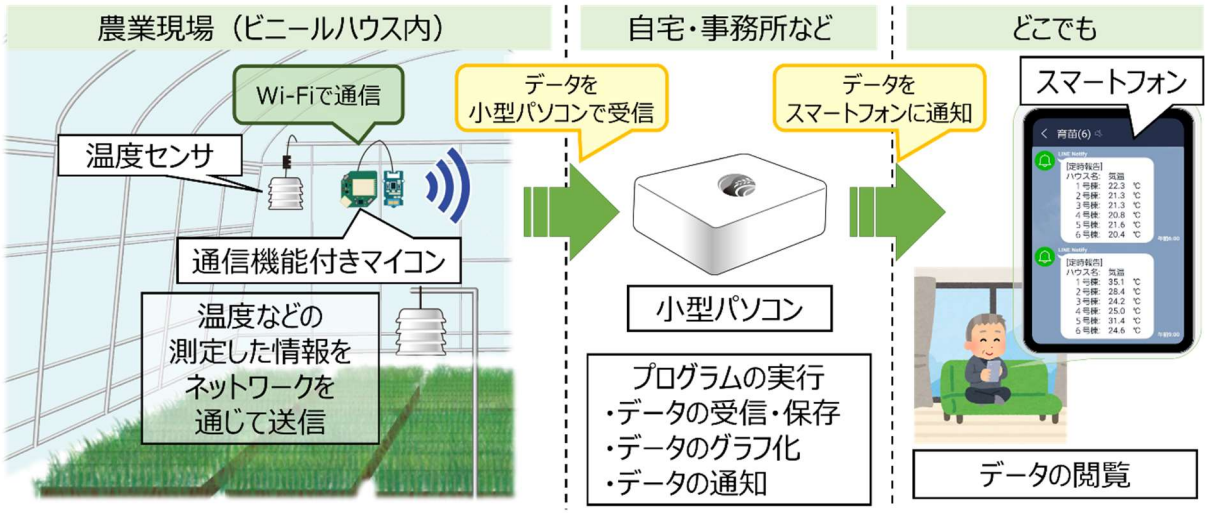
■ 免責事項

- 本手順書を用いて作成する「通い農業支援システム」、「配布プログラム」の利用又は利用不能で生じた直接又は間接的損害について、責任を負いません。
- 本手順書に記載した市販品の情報は執筆時点のものであり、製品の仕様・価格変更、生産・販売終了について農研機構は一切責任を負いません。また、市販品の使用方法について、メーカー指定外での利用や、過酷な環境下（高温、多湿、ちり、ほこり）での運用の結果について、農研機構は責任を負いません。
- 本手順書に記載した配布プログラムの商業的な利用又は配布は、農研機構に連絡して下さい。
- Wio Node の設定は Wio ver.2.5.6 (Android 13)、Wio Link ver.1.5.6（iOS 14.3）で動作を確認しています。
- 配布プログラムは Python 3.7.3、Numpy 1.20.1、Pandas 1.2.2、 Maplotlib 3.3.4 で動作を確認しています。
- 配布プログラムに関して不具合やエラーや障害が生じないことを保証しません。また、配布プログラムに欠陥があると判明した場合、訂正や補修する義務を負いません。
- 本手順書に記載したメッセージ通知用の Web API が用意されているアプリケーションには LINE や Slack があります。メッセージアプリの利用に際しては、サービス提供会社のプライバシーポリシーを十分に確認するとともに、自身の組織の情報セキュリティ管理方針に基づいて利用を判断して下さい。
- Wio Node は Seeed Technology Co., Ltd の商品です。
- Wio、Wio Link は Seeed Technology Co., Ltd が提供するソフトウェアです。
- Raspberry Pi はラズベリーパイ財団の登録商標です。
- Raspberry Pi Imager はラズベリーパイ財団が提供するソフトウェアです。
- LINE は LINE ヤフー株式会社の商標または登録商標です。
- Slack は Slack Technologies, Inc.の商標または登録商標です。
- Python は Python Software Foundation の商標または登録商標です
- Google Chrome、Chromium は Google LLC の商標または登録商標です。
- GitHub は、GitHub Inc.の商標または登録商標です。
- その他、会社名、商品名などは一般に各社の商標または登録商標です。

Ⅰ．通い農業支援システムの全体構成

1.通い農業支援システムの構成

通い農業支援システムは、遠隔地にあり頻繁に通うことが難しい現地のハウスの環境情報を Wi-Fi による無線ネットワークを通じてスマートフォンで遠隔監視するものです。市販の無線通信機能付きマイコンと温度等のセンサ、小型パソコン（Raspberry Pi）、スマートフォンで構成されます（図Ⅰ-１、図Ⅰ-２）。



図Ⅰ-１　通い農業支援システムの全体構成

図Ⅰ-２　現地での無線通信機能付きマイコンの設置例

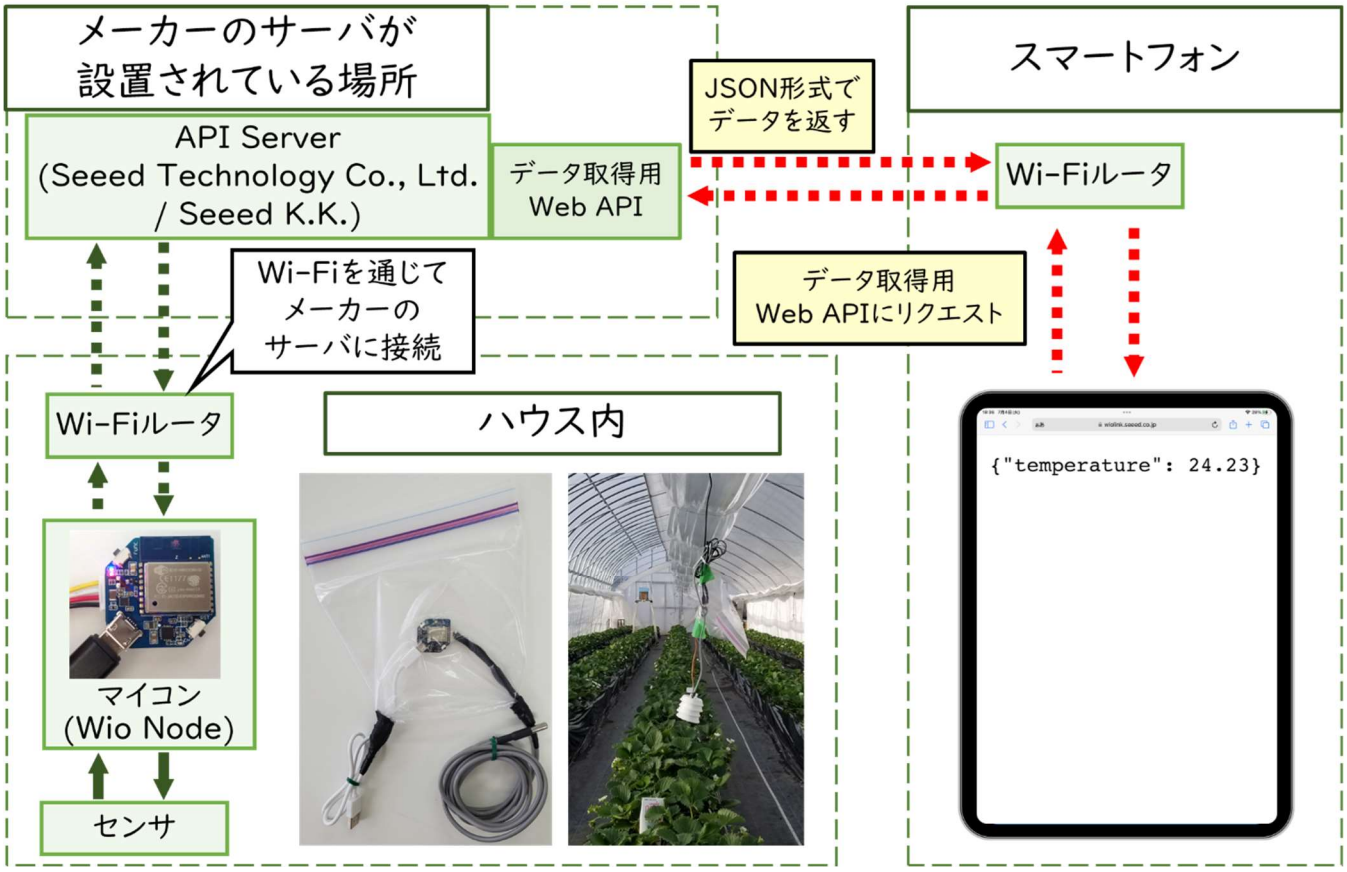
左：無線通信機能付きマイコンと温度センサ　右：水稲育苗ハウスでの設置例

2.通い農業支援システムの技術的な仕組み

「通い農業支援システム」では、データ取得用 Web API が用意されモジュール化されている IoT 機器を用いて遠隔地のデータを取得し、メッセージ通知用 Web API が用意されているメッセージアプリを用いてデータを通知します。

(1) データの取得方法の仕組みと自動化について

「通い農業支援システム」では「データの取得を簡易に行える Web API が用意されモジュール化されている IoT 機器（本手順書では Seeed 社の Wio Node）」を使用して作ります。データ取得用の Web API が用意されていると、図Ⅰ-3 のような仕組みで簡単にデータを取得できます。



図Ⅰ-3　Web API を利用したデータ取得の仕組み

スマートフォンからデータ取得用の Web API にアクセス（図Ⅰ-3 中の赤い点線）すると、自宅や事務所の Wi-Fi 等のインターネット回線を通じてメーカーのサーバにデータを取得

するよう指示（リクエスト）を出します。メーカーAPI サーバはハウス内に設置した Wio Node のセンサデータを取得し、スマートフォンにデータを返します。

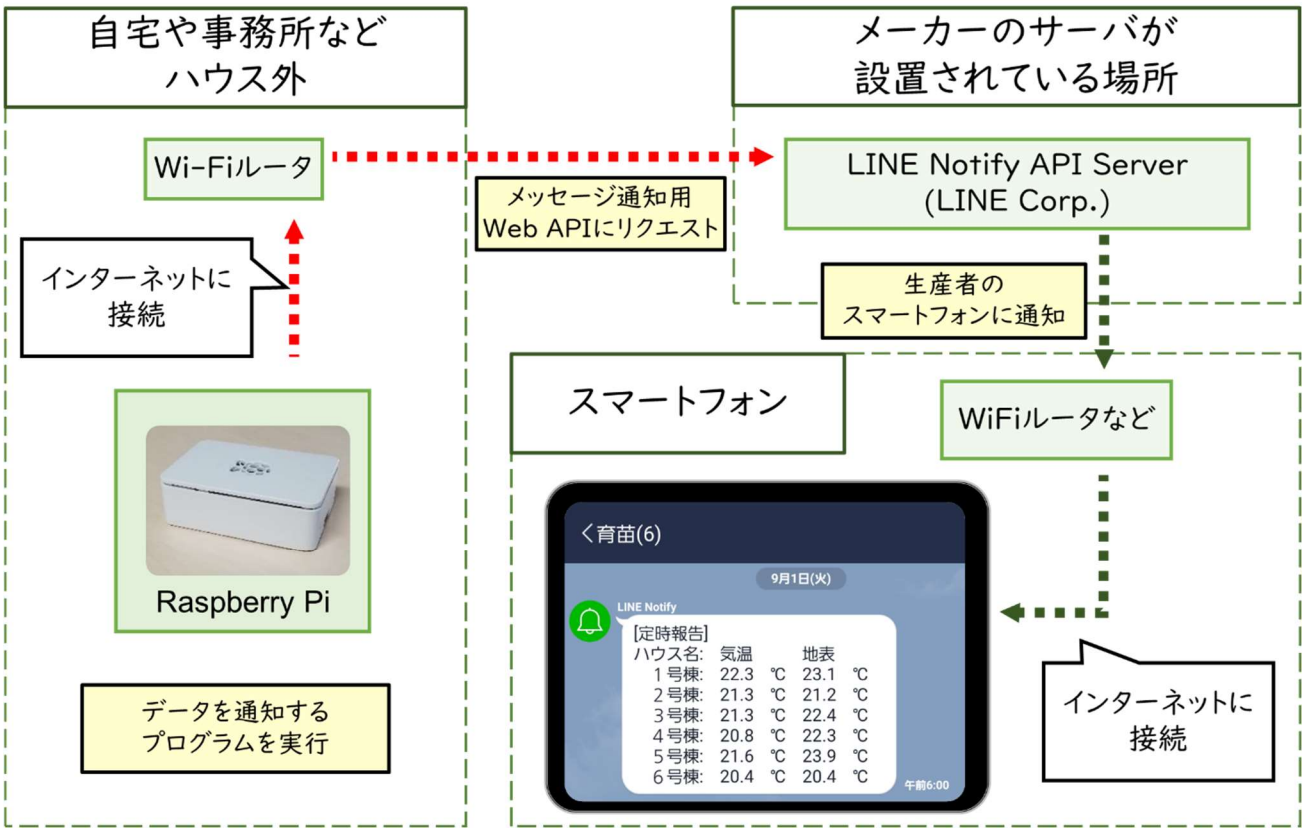
Wio Node ではスマートフォン上のアプリでマイコンを登録し、使用するセンサを設定することができます。登録した Wio Node ごとにアクセストークン（固有の文字列）が設定されます。設定したセンサごとにデータ取得用の URL（「https://から始まる URL にアクセストークンが含まれた文字列」）が発行されます。ブラウザでデータ取得用の URL にアクセスすることでデータの取得が可能です。しかし、これでは手動でしかデータを知ることができません。そこでこの操作を自動化して、スマートフォンに通知するプログラムを「通い農業支援システム」では用意しています。配布プログラム中では、センサごとに設定される「固有の値（URL）」を書き換えることで自動実行できるようになります。

図Ⅰ-4 データ取得用 Web API のアクセストークンの設定方法と自動化

（2）データ通知の仕組みについて

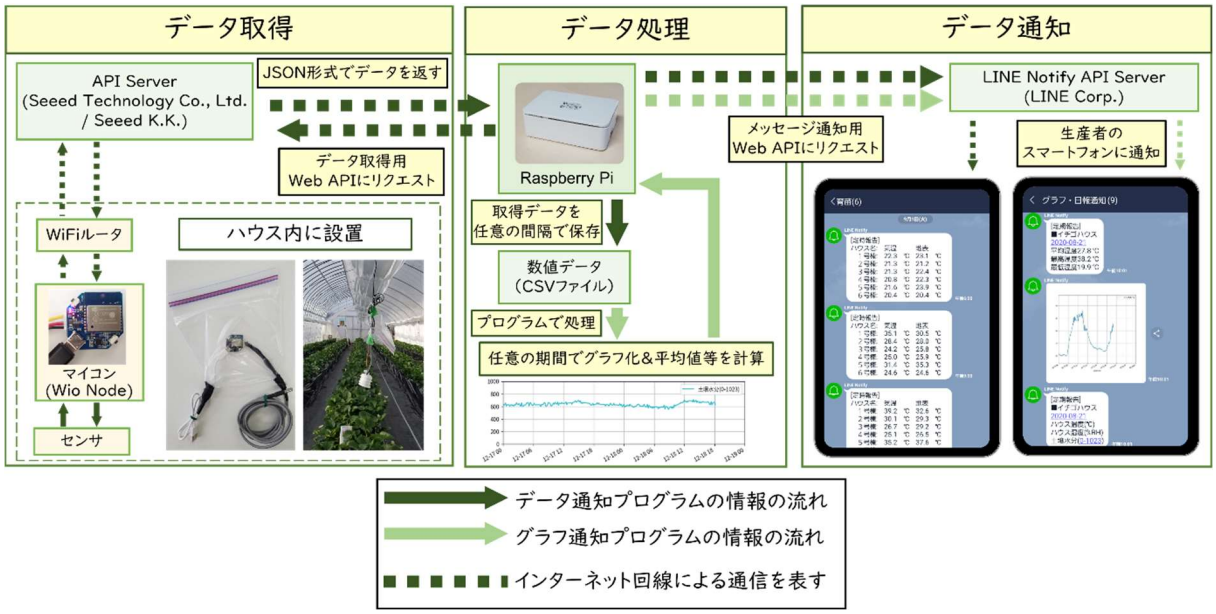
スマートフォンへのデータ通知にはメッセージ通知用の Web API が利用されているものを使用する必要があります。そのため、今回は LINE を利用することとしています。小型パソコン（本手順書では Raspberry Pi）でデータを通知するためのプログラムを実行すると、LINE 社のメッセージ通知用の API サーバにデータを渡します（赤い点線）（図Ⅰ-5）。これだけでスマートフォンにデータを通知することができます。ただし、LINE のどのグループに（あるいはどのユーザーに）通知するかを設定する必要があります。これには、Wio Node と同様に、LINE のメッセージ通知サービスである「LINE notify」を利用し、アクセストークンを発行する必要があります。アクセストークンの発行方法は後ほど説明しますので割愛しますが、LINE 社の Web API サーバに「通知するメッセージ」と「アクセストークン」を送ることで指定したグループにメッセージを通知することが可能です（図Ⅰ-6）。

図Ⅰ-5　メッセージ通知の仕組み



（3）通知するデータの種類

「10 分おきの温度」が知りたいのか、それとも「平均温度」や「グラフ」などデータを処理することで傾向を知りたいのかは生産者によって異なります。生産者にとって「利用しやすい形」でデータを通知する必要があります。そこで、「通い農業支援システム」ではデータの取得、保存と通知を行う「データ通知プログラム」と、10 分おきに保存したデータ（瞬時値）を「平均値」や「グラフ」に処理してから通知する「グラフ通知プログラム」を用意しています（図Ⅰ-6）。



図Ⅰ-6 「通い農業支援システム」の仕組み

3.通い農業支援システムのコスト

（1）導入コスト

ハウス 1 棟（本試算では 6 m×36 mを想定）から 6 棟を上限に温度センサを 1 つずつ設置するパターンについて説明します。ハウス 1 棟から 6 棟を上限に通い農業支援システムを導入した際のコスト試算は 1 棟では約 2.3 万円、上限の 6 棟の場合は 1 棟あたり約 1.2

万円となります。表Ⅰ-1の（１）黄色の網掛け部分は１棟ごとに必要となる資材となり、電源延長には（２）の資材を用います。

表Ⅰ-１　通い農業支援システムの導入コスト試算

	必要な資材	ハウスの棟数 1棟	3棟	6棟
（1）【データ取得】ハウス内に設置し、温度を取得するために使用	マイコン（Wio Node）	1,980	5,940	11,880
	防水温度センサ	1,600	4,800	9,600
	電源用USBケーブル	110	330	660
	Wi-Fiルータ*	5000	5000	5000
(2) ハウス内の電源からマイコン設置場所までの電源延長に使用	USB延長ケーブル	500	1,500	3,000
	100V電源延長コード他**	5,000	15,000	30,000
	USB ACアダプタ	1,000	3,000	6,000
(3) 【データ処理】自宅や事務所に設置し、プログラムの実行に使用	小型パソコン（Raspberry Pi 3B+）	5,000	5,000	5,000
	Raspberry Pi 3B+用ケース	1,000	1,000	1,000
	microSDカード***	1,000	1,000	1,000
	USBキーボード***	700	700	700
	USBマウス***	300	300	300
	HDMIケーブル***	300	300	300
	合計	¥23,490	¥43,870	¥74,440
	1棟あたりの費用	¥23,490	¥14,623	¥12,407

*Wi-Fiルータは100Vで動作する市場流通品のホームルータ（例えばSpeed Wi-Fi Home L02等）やモバイルルータを想定しています。
**100V 電源延長コード他の内訳は、「100V 電源延長コード（1,500 円）」、延長コードを入れてUSB ACアダプタを保護する「RVボックスなどのコンテナ（1,000円）」、「コンセント差し込み型の漏電ブレーカ（2,500円）」を想定しています。
***これらの周辺機器は自宅にある場合は必要ありません。なお、周辺機器一式を含むスターターキットも約１万円で販売されています。
（2023年４月時点の市場価格を基に試算）

Raspberry Piの周辺機器として、OSをインストールするmicroSDカード、キーボード、マウスが必要です。microSDカードは少なくともUHS-Iあるいはclass10の16GB以上を推奨します。Raspberry PiのOSインストールには、Windows（windows10で動作確認済み）、macOS、あるいはLinux（Ubuntu）が搭載されたPCが必要です。また、

Raspberry Pi に接続して使用するマウスとキーボードについては、自宅や事務所で過去に使用していたマウスやキーボードでかまいません。

ハウス内には Wio Node や温度センサを設置する必要があります。Wio Node の設置には電源が必要です。一般的なスマートフォンの充電器に使われる USB-AC アダプタ（100V 電源）を使って、USB ケーブルで接続します。データの取得や保存、グラフ化などといった処理を行う Raspberry Pi はインターネットを通じてデータを取得・通知します。そのため、ビニールハウス内やその周りに設置する必要は無く、インターネットが利用可能な自宅や事務所などに設置できます。データの閲覧にはメッセージの送受信に必要な Web API を備えるメッセージアプリであれば使用できます。今回は LINE 社のメッセージアプリ「LINE」を例にしますのでコスト試算には含めていません。

自宅の前にビニールハウスがある場合は、自宅の Wi-Fi ネットワークが利用できることがあります。その場合は Wi-Fi ルータは不要となります。既に自動灌水装置などを導入しているハウスであれば、ハウス内に配電盤があることが多いです。その場合はコンテナや 100V 延長コードを利用せず、配電盤内に USB-AC アダプタを設置することで 100V 電源延長コード等を省略することができます。また、配電盤のブレーカが漏電ブレーカであれば、コンセント差し込み型の漏電ブレーカを省略することが可能です。

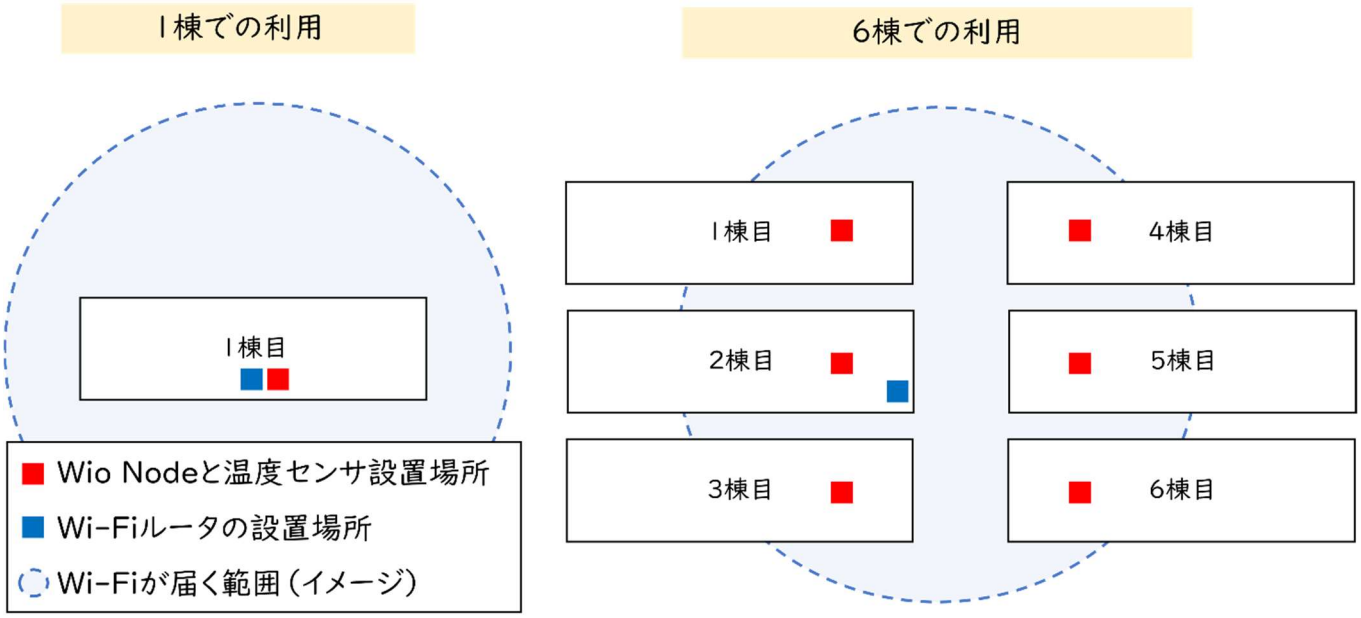
(2) ランニングコスト

通い農業支援システムで使用するインターネット回線は高速かつ大容量である必要はありません。そのため、格安 SIM と呼ばれる月額 1,000 円でデータ容量 3GB 程度の物を選択すれば十分に利用できます。使用する Wi-Fi ルータで利用可能な SIM を選択する必要があります。

(3)導入イメージ

複数棟で利用する際は、Wi-Fi ルータの設置場所を工夫して、できる限り 1 台の Wi-Fi ルータで無線ネットワークの範囲を広くすることがコツです。ただし、通常の Wi-Fi ルータは同時接続台数が 10 台（ホームルータでは最大 20 台ほど）ですので、温度以外にも他に測定したい項目がある、あるいは多点で計測したい場合は Wio Node の数が増えるため複数の Wi-Fi ルータが必要となることに注意してください。また、市販の Wi-Fi 中継器を用いて無線ネットワークの範囲を広げることが可能ですが、Wi-Fi 中継器自体も接続数にカウントされますので注意が必要です。

Wi-Fi ルータ電波の通信範囲は使用する Wi-Fi ルータ製品・使用環境によって異なりますが、ハウス内に設置した場合では通常、直線距離で約 30～40 m ほど届きます。用意した Wi-Fi ルータでどこまで電波が届くかは、事前に確認する必要があります。今回 6 棟で使用する場合は、中央のハウスの入り口に設置し、他の隣接するハウスでは電波が届く範囲に Wio Node を設置することを前提としています（図Ⅰ-7）。



図Ⅰ-7 データ取得・通信設備設置のイメージ

Ⅱ．無線通信機能付きマイコンの導入

1.部品リスト

ハウス内に設置する無線通信機能付きマイコンの制作と、設定に必要な資材を表Ⅱ-1 に示します。温湿度センサ、土壌水分センサは必須ではありません。

表Ⅱ-1 ハウス内で温度を測定する際のマイコン等一式の部品例

資材および部品	数量	規格･メーカー等
無線通信機能付きマイコン	1	Seeed社 Wio Node
防水温度センサ	1	Seeed社 ONE WIRE TEMPERATURE SENSOR
USB-ACアダプタ	1	Type-A用 Anker Anker PowerPort miniなど
USBケーブル	1	USBケーブル（Type-A - microB）
※温湿度センサ	(1)	Seeed社 Grove 温湿度センサ（SHT31/SHT35）
※土壌水分センサ	(1)	Seeed社 Grove 水分センサ

2.無線通信機能付きマイコンの制作と設定

（１）必要な材料の用意と確認

必要な材料は、無線通信機能付きマイコン（Wio Node）、防水温度センサ、Micro USB ケーブル、USB-AC アダプタ（図 II-1）です。ケーブルの長さは、設置場所への距離を目安に決めます。異なるセンサを利用する場合は注意が必要です（参照 p.24）

無線通信機能付きマイコン（Wio Node）


防水温度センサ


Micro USB
ケーブル


USB-AC
アダプタ

図Ⅱ-２　センサと電源用 micro USB ケーブルの接続方法

（２）無線通信機能付きマイコンの組立

センサと micro USB ケーブルを Wio Node に接続します。（図Ⅱ-1、Ⅱ-2）

（３）設定するための準備（初回は（４）のアプリを先にインストールしてください）

func ボタンを長押しするとボタン近くの青色 LED の点滅がゆっくりになります。Wio Node から設定用の Wi-Fi 電波（SSID（参照 p.82）は Wio_xxxxxx）が出るようになります。（図Ⅱ-3）

図Ⅱ-３　Wio Node の設定モード

（４）スマートフォンにアプリをインストールする

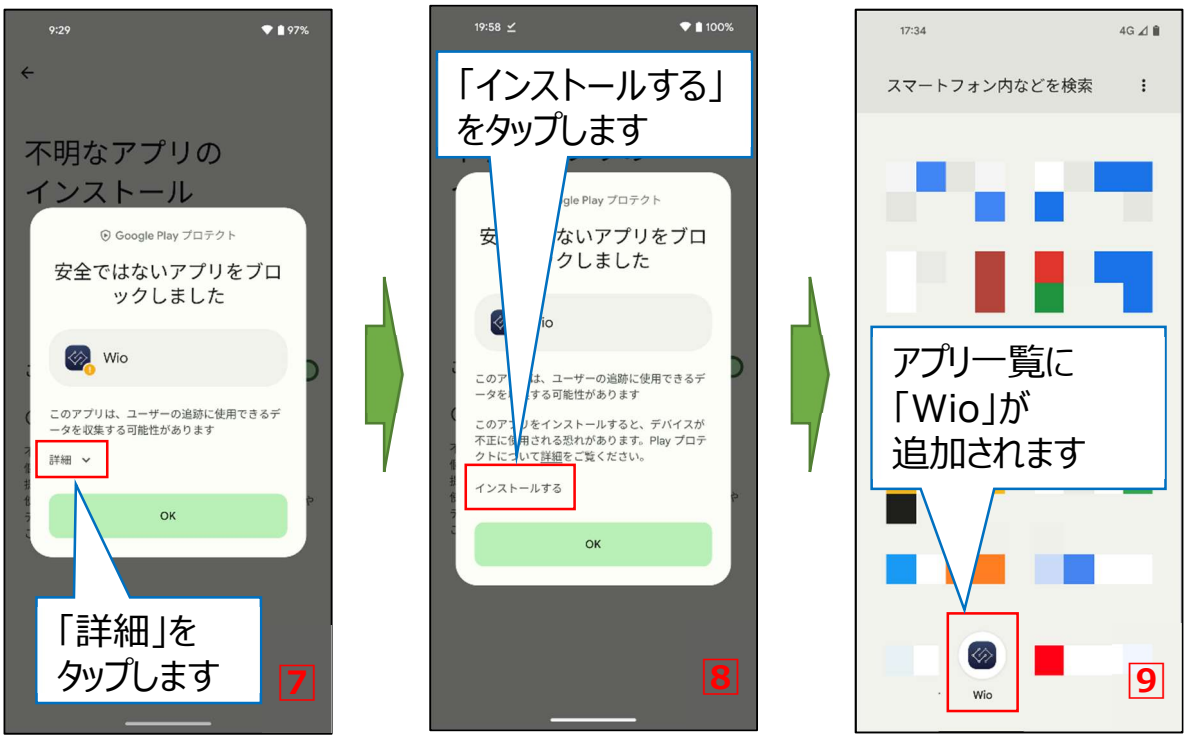
専用アプリ「Wio Link」をスマートフォンにインストールし、機器を登録します。iOS 版は App Store からインストールできます。Android 版は Google Play ではなく、 Seeed 株式会社が不具合修正や日本語化を行ったものが GitHub にて配布されております。ここでは、Android 版を例に、インストール方法と設定方法について説明します。

スマートフォンの Chrome 等のブラウザを用いて、ダウンロード先 URL（https://github.com/SeeedJP/Wio_Link_Android_App/releases）に直接アクセスするか、「Seeed エンジニアブログ Wio Link」で検索し、ブログ記事「Wio Link Android App のインストール手順」よりダウンロード先 URL にアクセスします。 その後、「app-2.5.6-release.apk」をダウンロードし、下記の手順でインストールします（図Ⅱ-4）。

図Ⅱ-4　Wio Link アプリのダウンロード方法

Android スマートフォンに標準インストールされている Files などのファイル管理アプリにて、ダウンロードしたファイルをインストールします。ここでは、Files での操作を例に説明します。

図Ⅱ-5　Wio Link アプリのインストール方法①



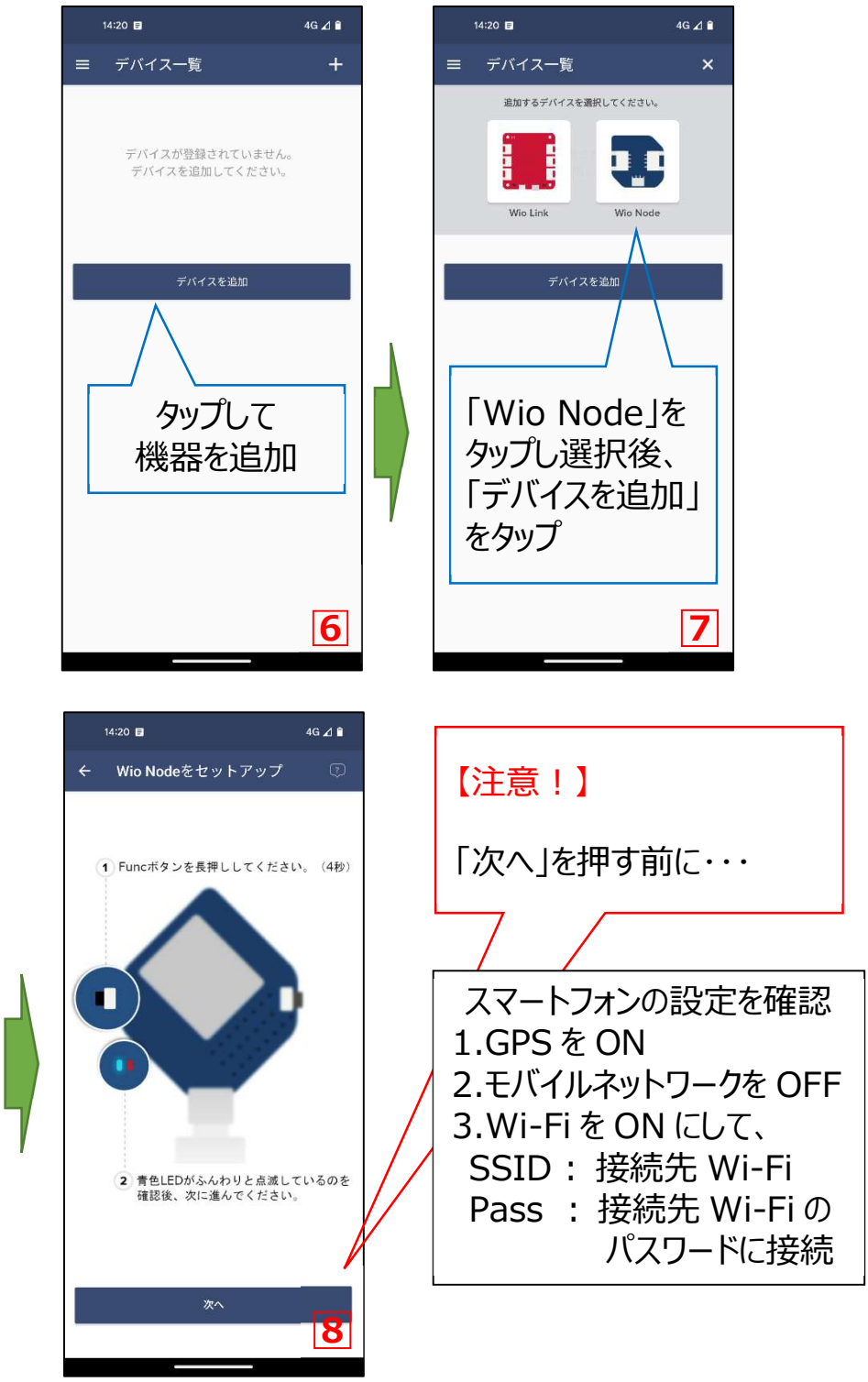
図Ⅱ-6　Wio Link アプリのインストール方法②

（５）アプリから Wio Node を登録する

アプリ一覧から「Wio」をタップし、以下の手順で日本サーバに設定し、アカウントを登録します（図Ⅱ-7）。その後、「デバイスを追加」から Wio Node を登録します（図Ⅱ-8、図Ⅱ-9）。

モバイルデータ通信を ON にして 15 のエラーが起きると、機器登録できないことがあります（図Ⅱ-10）。これは、最新の iOS や Android では、がインターネット接続できない SSID（Wio_xxxxxx）に接続されると、すぐに Wi-Fi の接続が切断されてしまうために起こる現象です。このため、Wio Node に設定情報を送信できません。これを回避するには 16 以降の設定を行います。

図Ⅱ-7　Wio Node の登録方法①



図Ⅱ-8　Wio Node の登録方法②

図Ⅱ-9　Wio Node の登録方法③

図Ⅱ-10　Wio Node の登録方法④

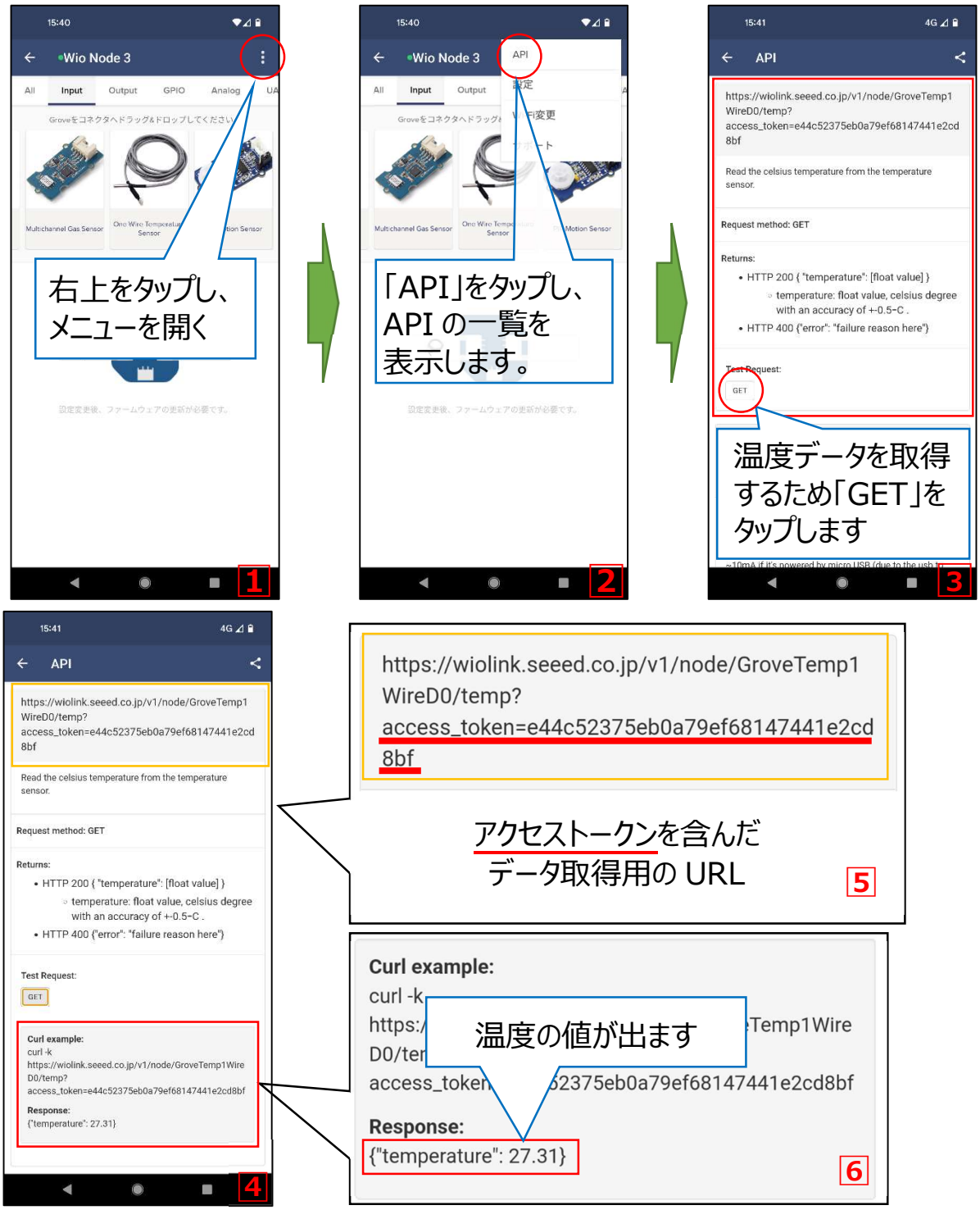
（６） 温度センサを Wio Node に登録する

図Ⅱ-11 の手順に沿ってセンサを選び、マイコンのファームウェアを更新します。センサごとに接続できるコネクタが異なります（参照 p.24）。

図Ⅱ-11　温度センサの設定方法

（７） 温度を測定する

図Ⅱ-12 の手順に沿って API 一覧を表示し、温度を測定します。



図Ⅱ-１２　温度の測定方法とアクセストークンの取得

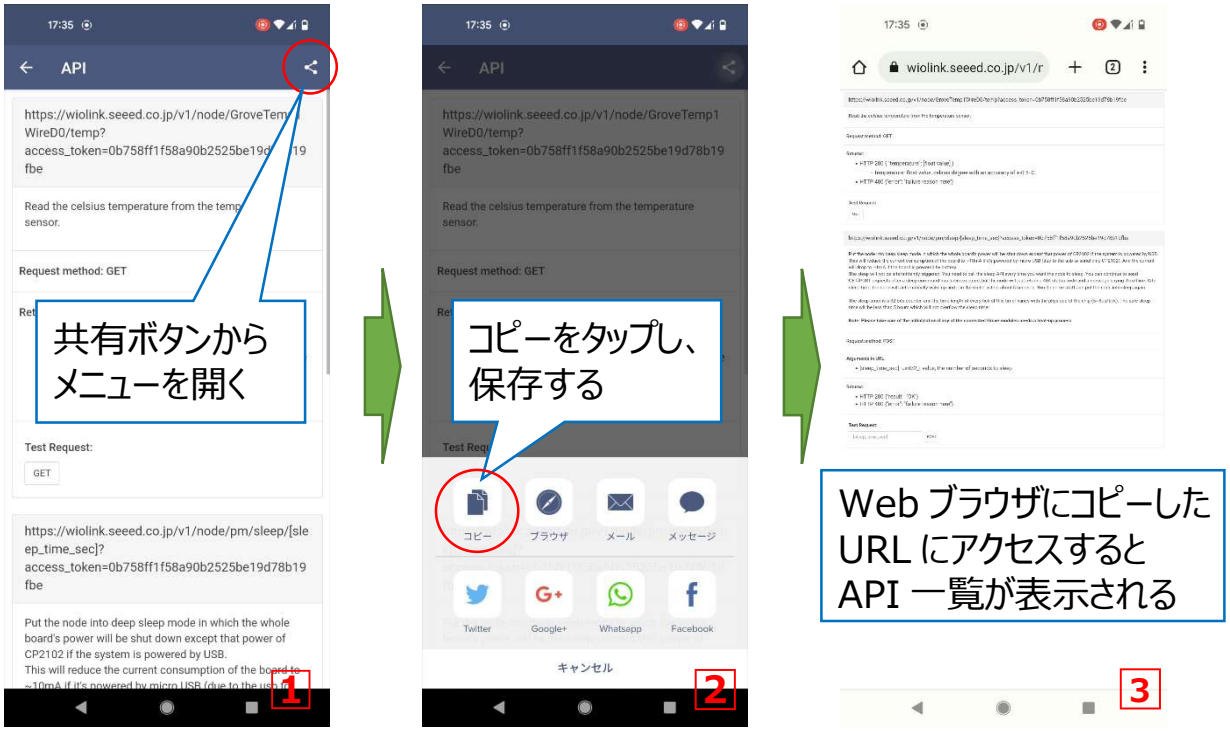
API 一覧の画面に出てくる https://wiolink.seeed.co.jp/‥‥/access_token=… の URL が重要です（図Ⅱ-12 5）。この URL はアクセストークン（参照 p.82）を含むデータ取得用の URL です。この URL をスマートフォンの Web ブラウザでアクセスすると、以下のように表示が出ます（図Ⅱ-13）。

図Ⅱ-13　アクセストークンを含んだデータ取得用 URL のブラウザ表示

Web ブラウザでアクセスすることで、メーカーのサーバ（Web API サーバ）を介してデータを取得しています。この操作を自動化する配布プログラムで使用するため、アクセストークンを含むデータ取得用 URL をセンサごとに保存する必要があります。

（８）　アクセストークンを含むデータ取得用 URL を保存する

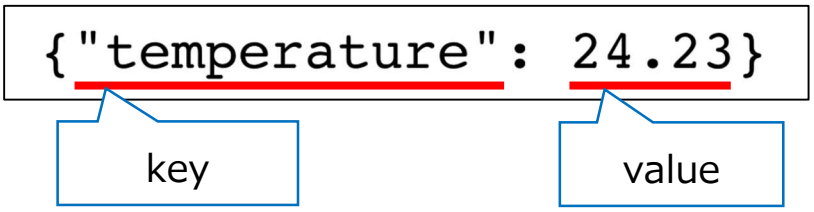
API 画面右上（図Ⅱ-14 1）に「共有ボタン」があるので、そこをタップすると API 一覧の画面にアクセスできる URL（https://wiolink.seeed.co.jp/v1/node/resources?access_token=アクセストークン）をコピーできます（図Ⅱ-14 2 3）。コピーして保存し、WEB ブラウザに貼り付けることで API 一覧を表示できます。この URL を「スマホのメールで自分に送る」「メモアプリ」に保存するなどすると便利です。しかし、データ取得用の URL や API 一覧の URL を知られてしまうと、他の人にもアクセスされてしまうので、保管方法には十分気を付けてください。



図Ⅱ-14　API 一覧の URL 保存方法

（9）　JSON 形式データの利用方法

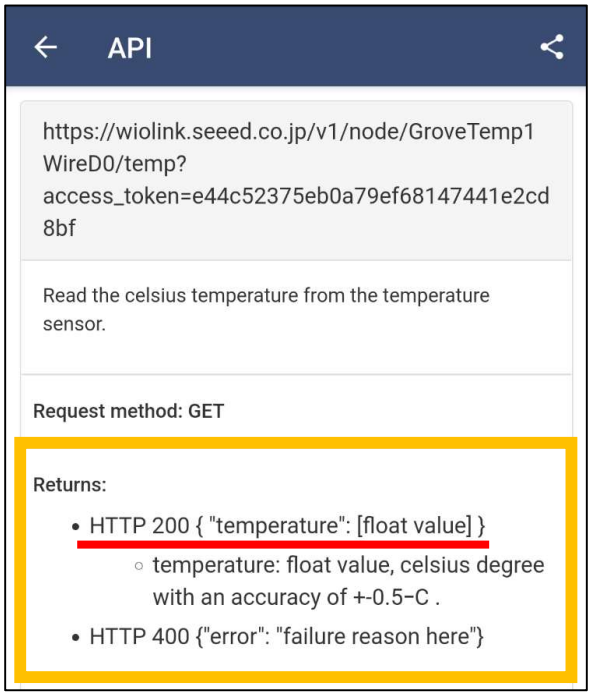
（7）で温度を測定し、図Ⅱ-15 のように表示されました。これは「JSON」と呼ばれる「名前（key）」と「データ（value）」の組み合わせの表示形式です。



図Ⅱ-15　JSON 形式のデータの見方について

温度のみ測定する場合は、”temperature”しか使いませんが、他のセンサを使う場合はそれぞれのセンサに応じた key が表示されます。たとえば、水分センサであれば”moisture”と表示されるため、水分センサを使う際はプログラム上で”moisture”と表記する必要があります。また、湿度の場合は”humidity”となります。

下図の Returns：という場所を見ると、これは https://wiolink.seeed.co.jp/‥から始まるアクセストークンを含むデータ取得用の URL にアクセスした際に表示される値です。HTTP 200 {“temperature”:[float value]} と書いてあり、key がわかります。



図Ⅱ-16　key の確認方法

（10）　防水温度センサ以外のセンサを利用する方法

Wio Node にはセンサを接続するコネクタが 2 つあり、最大 2 つのセンサを接続することができます。しかし、コネクタごとに対応するセンサの通信方式が異なり、Wio Link アプリ上でもコネクタに対応していないセンサは設定できません。たとえば、表Ⅱ-1 で紹介した防水温度センサはどちらのコネクタにも接続可能なため、Wio Node 1 つにつき 2 つ使用できますが、土壌水分センサは裏側から見て右側のコネクタ（図Ⅱ－2）のみ対応し 1 つ使用できます。一方、温湿度センサ（SHT31/35）はどちらのコネクタにも接続可能ですが、1 つしか利用できません。これは Wio Node の仕様（2023 年 4 月時点）で、「I2C」と呼ばれる方式のセンサは 2 つ同時に接続・設定するとデータが取得できないことがあるので注意が必要です。

参考：設定をする際のポイントと注意

- iOS 版では設定画面や設定手順が一部異なりますが、同様の設定ができます。Android 版の手順を開始する前に、接続用の Wi-Fi と接続しておく必要があります。「接続用の Wi-Fi と接続」⇒「Setup Your Wio Node まで進める」⇒「iOS の設定画面を開いて、Wio Node の SSID と接続する」⇒「アプリに戻って設定を進める」の順に進めてください。残りは Android 版とほぼ同様です。
- 複数の Wio Node を設定する際は 1 台ずつ行い、設定が完了した Wio Node は一度電源を切ってから次の Wio Node の設定を開始してください。（Wi-Fi ルータへの接続数が増えると回線が混雑して設定に失敗します）
- 接続する Wi-Fi は 2.4GHz の SSID と接続してください。5GHz の SSID とは接続できません。
- 一部のスマートフォン(Sony Xperia で確認)では、Wio Node に接続するセンサの登録（参照 p.20）ができるものの、Wio_Node 本体の登録（参照 p.17-19）ができません。このため、Wio Node 本体の登録には別のスマートフォンやタブレットを用意する必要があります。また、OS の更新がされた場合は Wio_Node 設定アプリのアップデートがされるまで設定できなくなることがあります。Wio_Node とスマートフォンの接続は、Wio_Node の Func ボタンを押すことで Wio_Node が設定用の電波を発し、スマートフォンが Wio_Node の SSID を選択し接続することで成立します。このため、SSID でインターネット通信できない場合に自動的に接続を切断してしまうタイプのスマートフォンでは Wio_Node の登録ができません。

Ⅲ．スマートフォンのメッセージアプリの設定

スマートフォンにデータを通知するために、メッセージアプリと連携させるための準備をします。メッセージアプリ「LINE」を例に、機能の１つである「LINE Notify」の設定を行っていきます。

（１）【スマートフォンで】事前準備

通知に使用したい LINE のアカウント（メールアドレス）とパスワードを確認しておきます。この際にスマートフォンの LINE アプリ上で PC でのログイン許可をオンにします。

（２）【スマートフォンで】通知先の設定

通知したいスマートフォンの LINE アプリでグループを作っておきます。

（３）【Raspberry Pi で】LINE Notify にアクセスして、トークンを発行

Raspberry Pi の Web ブラウザを開き、https://notify-bot.line.me/ja/ にアクセスし、メールアドレスとパスワードでログインします（図Ⅲ-1）。なお、複数回ログインに失敗するとしばらくログインできなくなるようです。LINE アプリの設定については、LINE 株式会社が提供する情報を参照してください（LINE みんなの使い方ガイド, https://guide.line.me/ja/）。

図Ⅲ-1　LINE Notify へのログイン方法

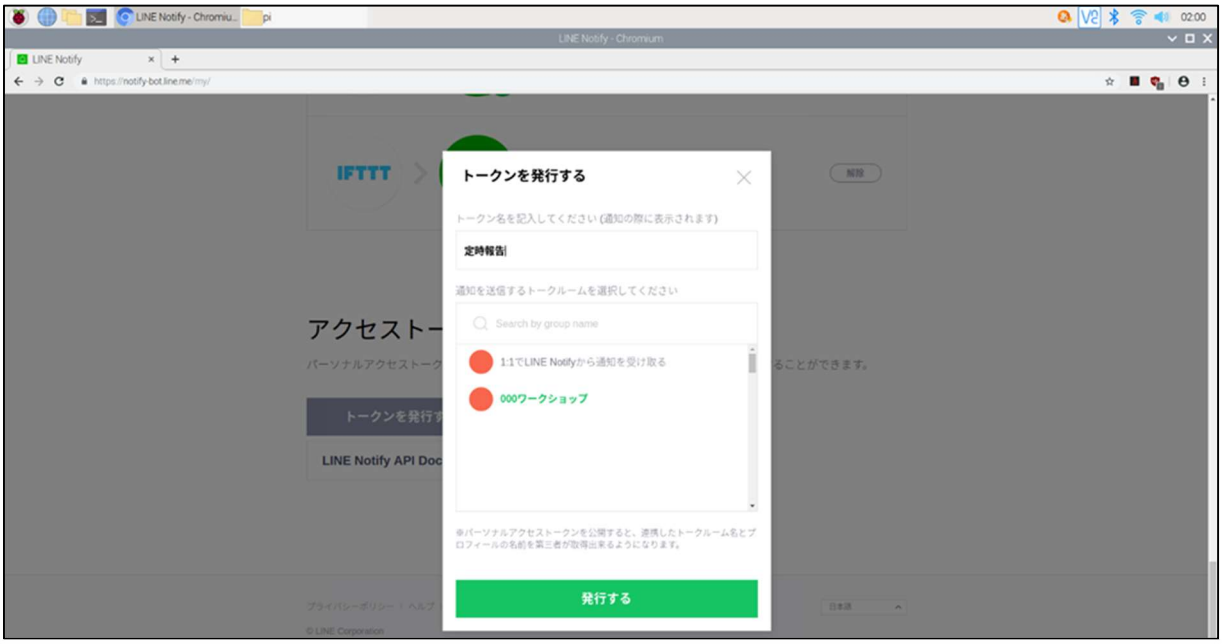
図Ⅲ-2 のような Line Notify のページが開きます。右上をクリックして、マイページにアクセスします。

図Ⅲ-２　LINE Notify 用のアクセストークンの発行方法①

下の方にアクセストークン発行とあるので、「トークンを発行する」をクリックします（図 III-3）。

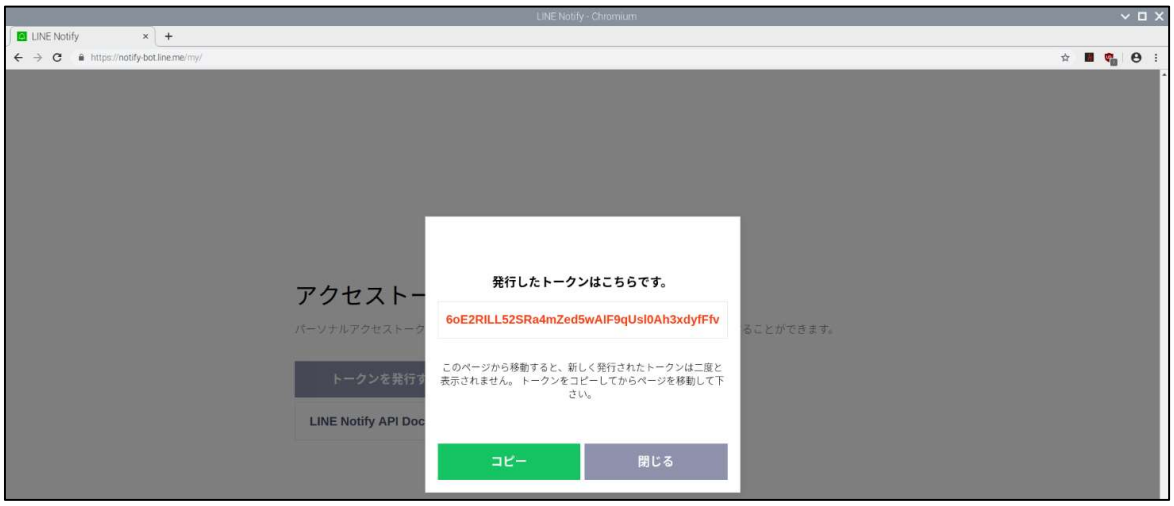
図Ⅲ-３　LINE Notify 用のアクセストークンの発行方法②

通知したいグループを選び、「トークン名」には「定時報告」と書き込み、「発行する」を押します（※トークン名はデータを通知する際に表示されます）（図 III-4）。



図Ⅲ-4　LINE Notify 用のアクセストークンの発行方法②

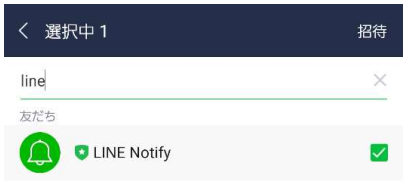
アクセストークンが発行されます（図Ⅲ-5）。コピーボタンを押し、Wio Node のアクセストークンと同様に確実に保存します。保存を尋ねる画面は 1 度しか表示されないので、必ず「テキストファイル」等に一時保存します。保存し忘れた場合は図Ⅲ-3 から発行手続きを再度行います。



図Ⅲ-5　LINE Notify 用のアクセストークンの発行方法②

（４）【スマートフォンで】通知したい LINE グループに LINE Notify を招待する

LINE グループのトークから招待する必要があります（図Ⅲ-6）。招待が終わればこれで終了です。なお、Line Notify のアクセストークンを再発行したい場合、図Ⅲ-3（p. 27）から再度行えば再発行可能です。



図Ⅲ-6 LINE Notify 用のアクセストークンの発行方法②

（参考）スマートフォンでアクセストークンを発行する方法

スマートフォンのブラウザ（Google Chrome など）の設定画面で、「PC 版サイト」を表示する項目をオンにすると、スマートフォン向けサイトでは設定することができないアクセストークンの発行が可能です。ただし、「PC 版サイト」を表示する設定は、ページを移動すると PC 版サイトを表示する設定が「オフ」になってしまうため、移動した際は再度「オン」にするよう注意してください。

Ⅳ. 小型パソコン（Raspberry Pi）でのプログラム

無線通信機能付きマイコンと、メッセージアプリを連携させるためのプログラムを作ります。プログラムは小型パソコン（Raspberry Pi）上で作成し定期実行させます（自由に設定可能（参照 p.68））。本項では Raspberry Pi の設定から行い、その後プログラムを設定していきます。なお、Raspberry Pi は、初めから Python というプログラミング言語が使用できる環境（Thonny Python：トニー パイソン）がインストールされており、プログラムの自動実行に必要な crontab と呼ばれる機能が使用できます。これらを利用して設定を進めていきます。安定して動作させるため、Raspberry Pi は本システム専用に用いることを推奨します。

1. Raspberry Pi の設定

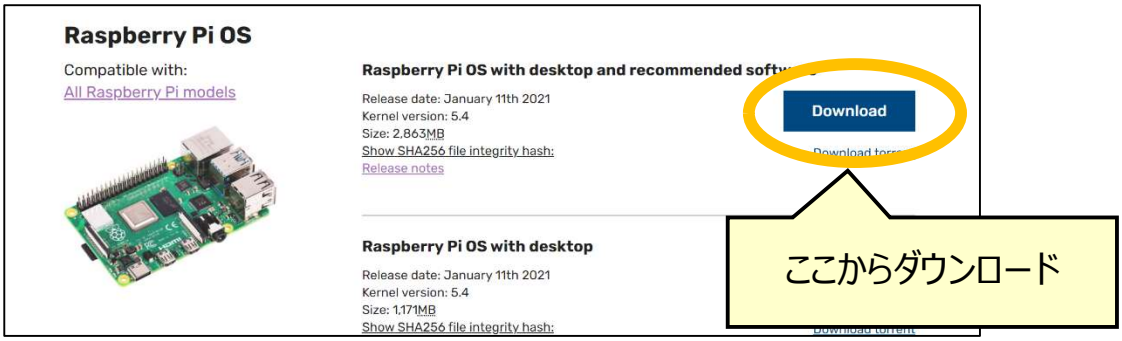
Raspberry Pi Imager と呼ばれる公式ソフトウェアを利用した方法を説明します。

（1）【Windows パソコンで】Raspberry Pi OS ファイルをダウンロードする

ラズベリーパイ財団のホームページのダウンロードページ（以下 URL）にアクセスします。

https://www.raspberrypi.org/software/operating-systems/

「Download」をクリックして、「Raspberry Pi OS with desktop and recommended software」をダウンロードします。ファイルが大きいため、時間がかかります。



引用：https://www.raspberrypi.org/software/operating-systems/ より

図Ⅳ-1 Raspberry Pi OS の OS ファイルのダウンロード

（２）【Windows パソコンで】Raspberry Pi Imager をダウンロードする

ラズベリーパイ財団のホームページのダウンロードページ（以下 URL）にアクセスします。

https://www.raspberrypi.org/software/

ここでは Windows 版を例に説明します。「Download for windows」をクリックしてダウンロードを開始してください。「imager_1.7.3.exe」というファイルがダウンロードされます。（ダウンロードの時期によりバージョンが異なり、ここでは 1.7.3 を例に説明します。）

図Ⅳ-2　Raspberry Pi Imager のダウンロード

（３）【Windows パソコンで】Raspberry Pi OS をインストールする

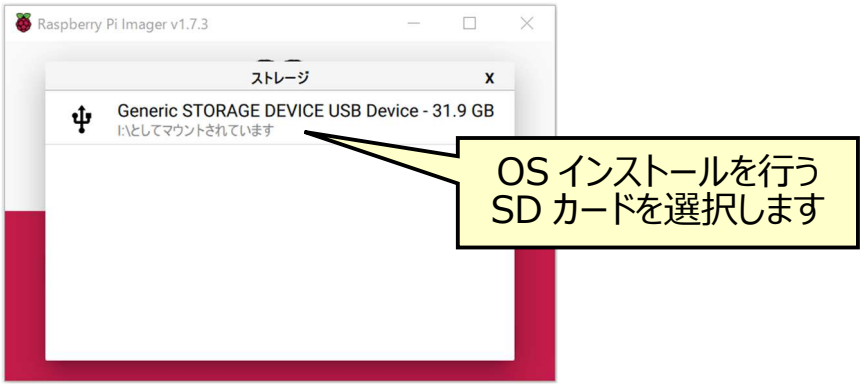
「imager_1.7.3.exe」から「Raspberry Pi Imager」をインストールします。その後、Raspberry Pi Imager を実行します。書き込みする microSD カードの選択とインストールする OS を選択してから OS のインストールを行います。

図Ⅳ-3　Raspberry Pi Imager の操作①

「カスタムイメージを使う」を選択し、ダウンロードした OS ファイル「20xx-xx-xx-raspios-buster-armhf-full.zip」を選択します。なお、「Raspberry Pi OS(other)」⇒「Raspberry Pi OS Full(32bit)」を選択することでもインストールが可能ですが、OS ファイルをダウンロードしながらインストールすることとなるため、長い時間がかかるほか、通信環境等の問題で失敗した場合はそれまでのインストールが無効となり、はじめからの再インストールが必要となります。そのため、今回は OS ファイルを事前にダウンロードする方法で行っています。

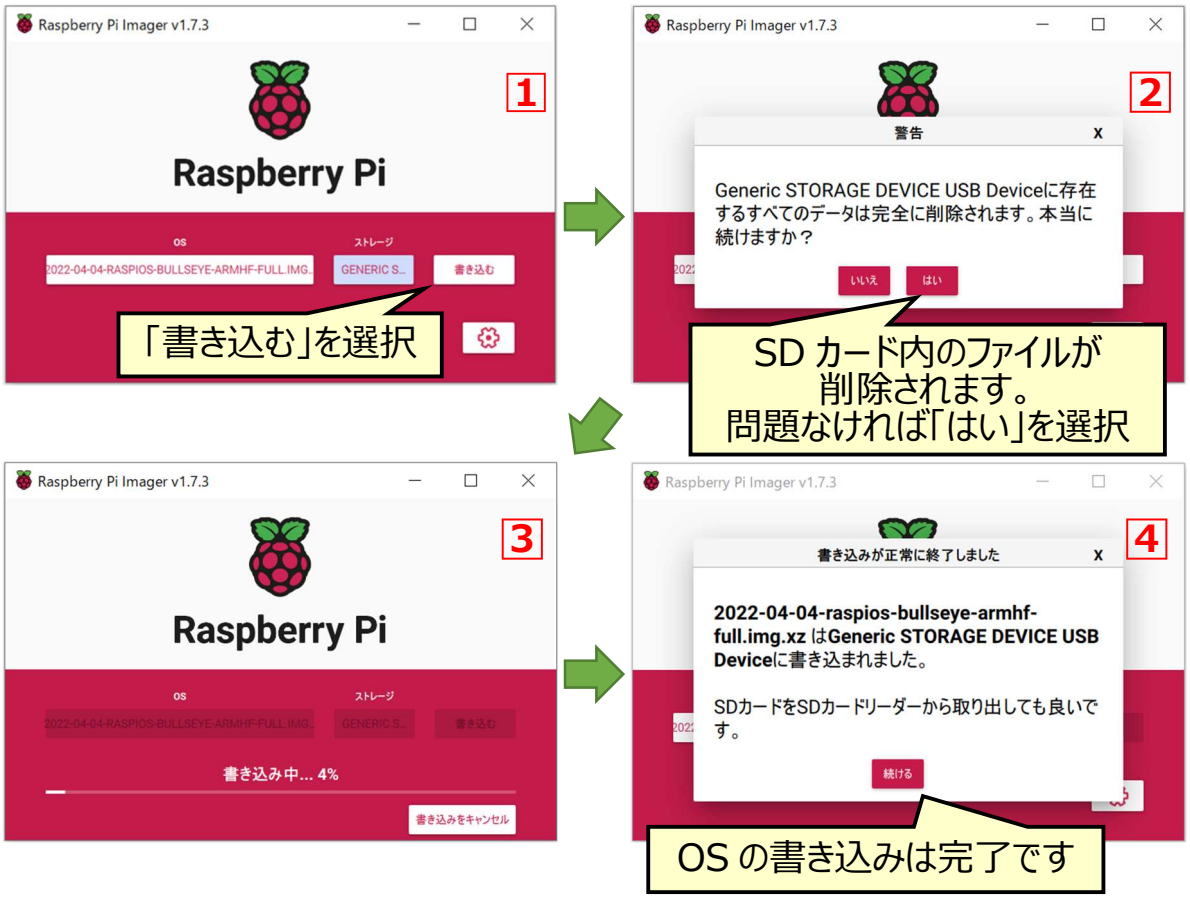
図Ⅳ-４　Raspberry Pi Imager の操作②

事前に使用する microSD カードを使用できるようマイクロ SD カードリーダーに接続しておきます。



図Ⅳ-５　Raspberry Pi Imager の操作③

OS ファイルと SD カードの選択が終わりましたので、WRITE を選択し、OS ファイルの書き込みを開始します。

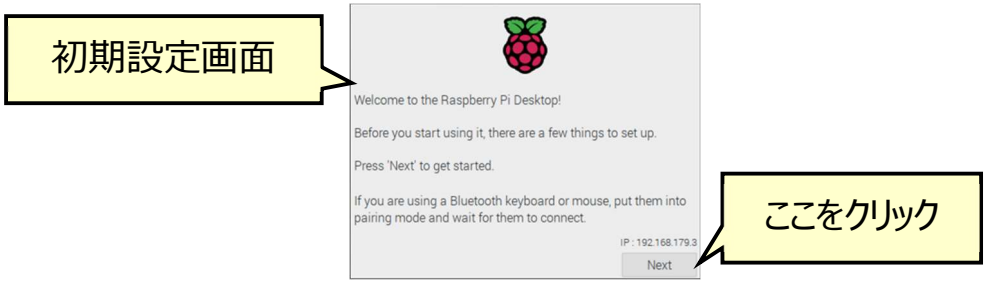


図Ⅳ-６　Raspberry Pi Imager の操作④

（4）【Raspberry Pi で】Raspberry Pi の設定を行う

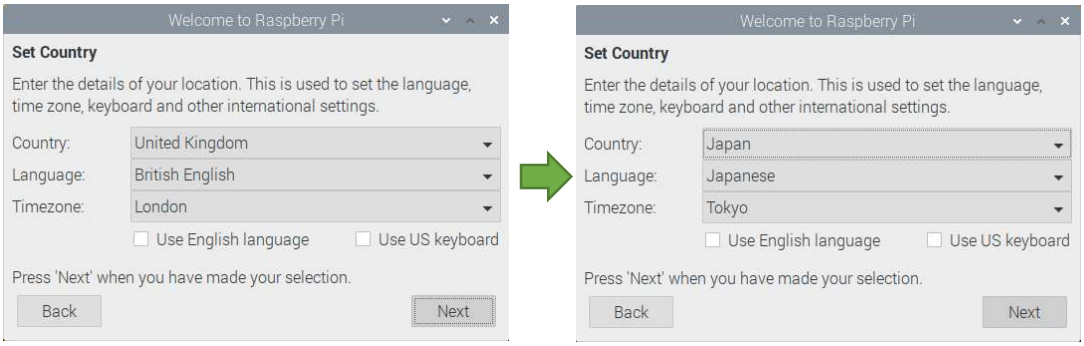
OS をインストールした microSD を Raspberry Pi にセットし、電源用の USB ケーブルをつなぎ Raspberry Pi に電源を入れます。Raspberry Pi には液晶モニタ（HDMI 入力を備えたテレビなど）、キーボードおよびマウスをあらかじめ接続しておいてください。

初期設定画面が開かれていますので、図 IV-7 の画面の Next をクリックして、画面に表示される指示に従って初期設定を進めていきます。



図Ⅳ-7 Raspberry Pi の初期設定①

「Country」を「Japan」に変更し、日本語表示とします。

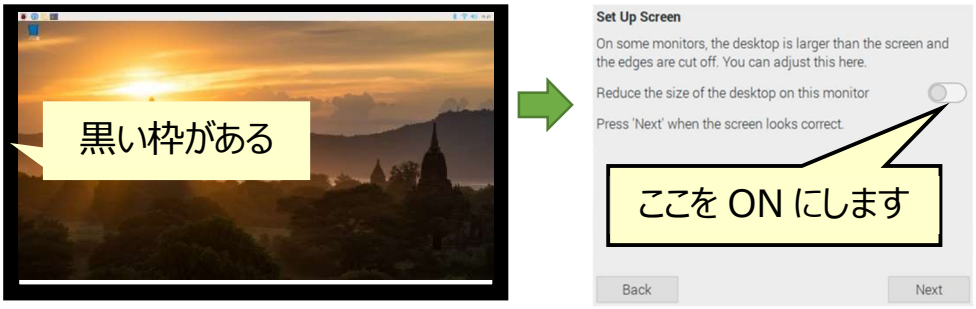


図Ⅳ-8 Raspberry Pi の初期設定②

username と password を入力します。

図Ⅳ-9 Raspberry Pi の初期設定③

表示画面に黒い枠がある場合はチェックを入れて進めてください。

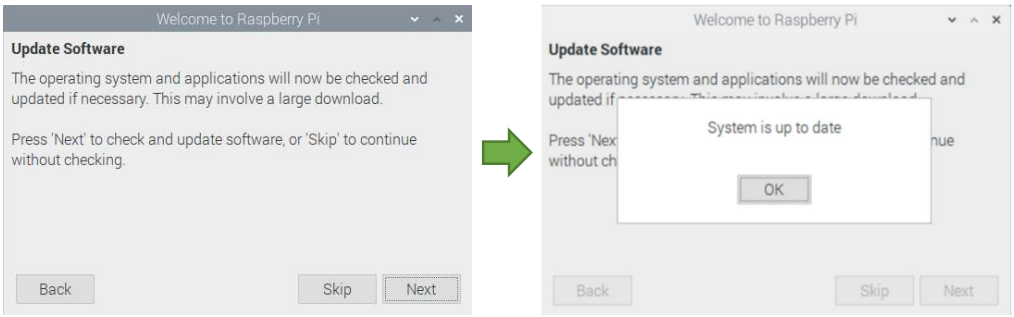


図Ⅳ-10　Raspberry Pi の初期設定④

Wi-Fi に接続します。

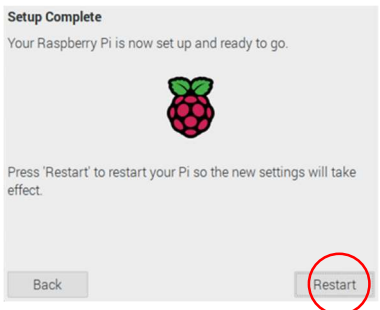
図Ⅳ-11　Raspberry Pi の初期設定⑤

ソフトウェアの更新を行います。ここで更新することで日本語入力ができるようになるので、必ず実行してください。アップデートに成功すると右の画面になります。



図Ⅳ-12　Raspberry Pi の初期設定⑥

“Restart”をクリックして再起動すると日本語表示、入力ができるようになります。

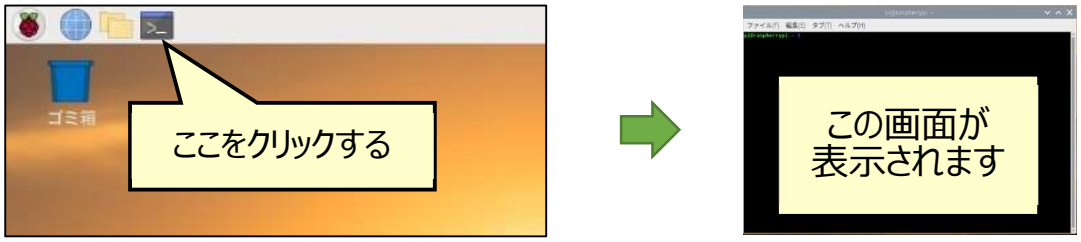


再度初期設定を行いたい場合、また日本語入力できない場合は、初期設定画面から再度ソフトウェアの更新を行う必要があります。
LXterminal（ターミナル)を使用して、sudo piwiz と入力して実行してください。
※LXterminal の説明は次ページ

図Ⅳ-13　Raspberry Pi の初期設定⑦

（5） 【Raspberry Pi で】グラフ通知プログラムを使用するための設定

LXterminal（ターミナル）（図Ⅳ-14）を使用して、グラフ通知プログラムに必要なモジュール（プログラミング言語 Python の機能を拡張するライブラリ）を 3 つインストールします。「NumPy」「pandas」「matplotlib」と呼ばれるものです。以下の手順に従って進めます。



図Ⅳ-14 ターミナルの実行方法

ターミナルに次のコマンド（下線部）を打ち込んで Enter を押してください。インストールに成功すると図Ⅳ-15 のような表示になります。

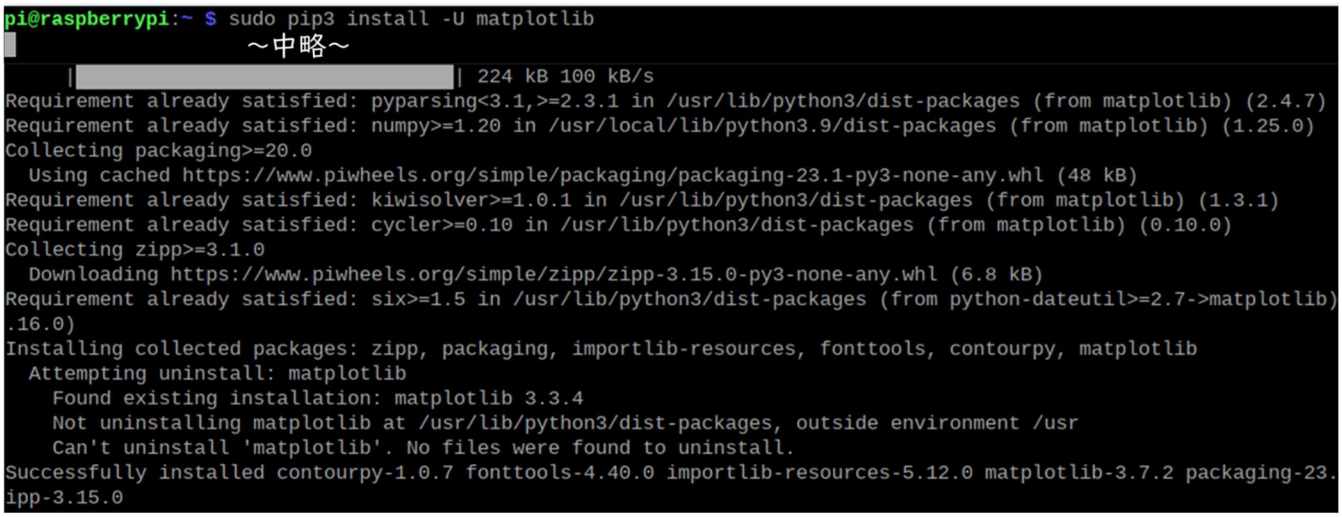
```
pi@raspberrypi: ~ $ sudo pip3 install pandas
```

```
pi@raspberrypi:~ $ sudo pip3 install pandas
Looking in indexes: https://pypi.org/simple, https://www.piwheels.org/simple
Collecting pandas
  Downloading https://www.piwheels.org/simple/pandas/pandas-2.0.3-cp39-cp39-linux_armv7l.whl (38.7 MB)
     |████████████████████████████████| 38.7 MB 3.3 kB/s
Collecting pytz>=2020.1
  Downloading https://www.piwheels.org/simple/pytz/pytz-2023.3-py3-none-any.whl (502 kB)
     |████████████████████████████████| 502 kB 102 kB/s
Collecting tzdata>=2022.1
  Downloading https://www.piwheels.org/simple/tzdata/tzdata-2023.3-py2.py3-none-any.whl (341 kB)
     |████████████████████████████████| 341 kB 234 kB/s
Collecting numpy>=1.20.3
  Downloading https://www.piwheels.org/simple/numpy/numpy-1.25.0-cp39-cp39-linux_armv7l.whl (12.5 MB)
     |████████████████████████████████| 12.5 MB 88 kB/s
Collecting python-dateutil>=2.8.2
  Downloading https://www.piwheels.org/simple/python-dateutil/python_dateutil-2.8.2-py2.py3-none-any.whl (247 kB)
     |████████████████████████████████| 247 kB 102 kB/s
Requirement already satisfied: six>=1.5 in /usr/lib/python3/dist-packages (from python-dateutil>=2.8.2->pandas) (1.16
.0)
Installing collected packages: tzdata, pytz, python-dateutil, numpy, pandas
  Attempting uninstall: python-dateutil
    Found existing installation: python-dateutil 2.8.1
    Not uninstalling python-dateutil at /usr/lib/python3/dist-packages, outside environment /usr
    Can't uninstall 'python-dateutil'. No files were found to uninstall.
  Attempting uninstall: numpy
    Found existing installation: numpy 1.19.5
    Not uninstalling numpy at /usr/lib/python3/dist-packages, outside environment /usr
    Can't uninstall 'numpy'. No files were found to uninstall.
Successfully installed numpy-1.25.0 pandas-2.0.3 python-dateutil-2.8.2 pytz-2023.3 tzdata-2023.3
```

図Ⅳ-15 モジュールのインストール①

引き続き、ターミナルに次のコマンド（下線部）を打ち込んで Enter を押してください。成功すると図Ⅳ-16 のような表示になります。

```
pi@raspberrypi: ~ $ sudo pip3  install -U matplotlib
```

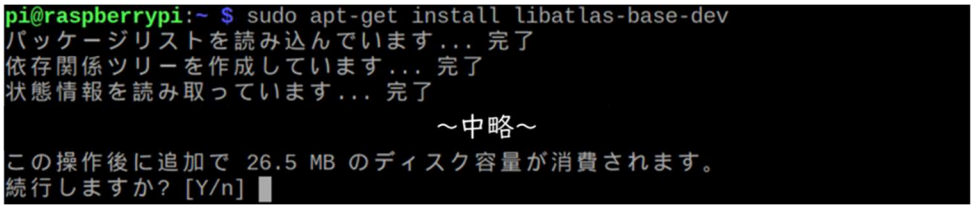


図Ⅳ-16　モジュールのインストール②

本手順書では matplotlib の version 3.3.2 以上を必要としており、これより古い場合は、グラフが日本語で表示されません。今回は 3.7.2 ですので、問題なく使用できます。

しかし、このままですと numpy がインストールされているのに使用できない状態になっています（2023 年 7 月現在）。そのため、以下のコマンドを実行し、使用できるようにします（図Ⅳ-17）。

```
pi@raspberrypi: ~ $ sudo apt-get install  libatlas-base-dev
```



図Ⅳ-17　モジュールのインストール③

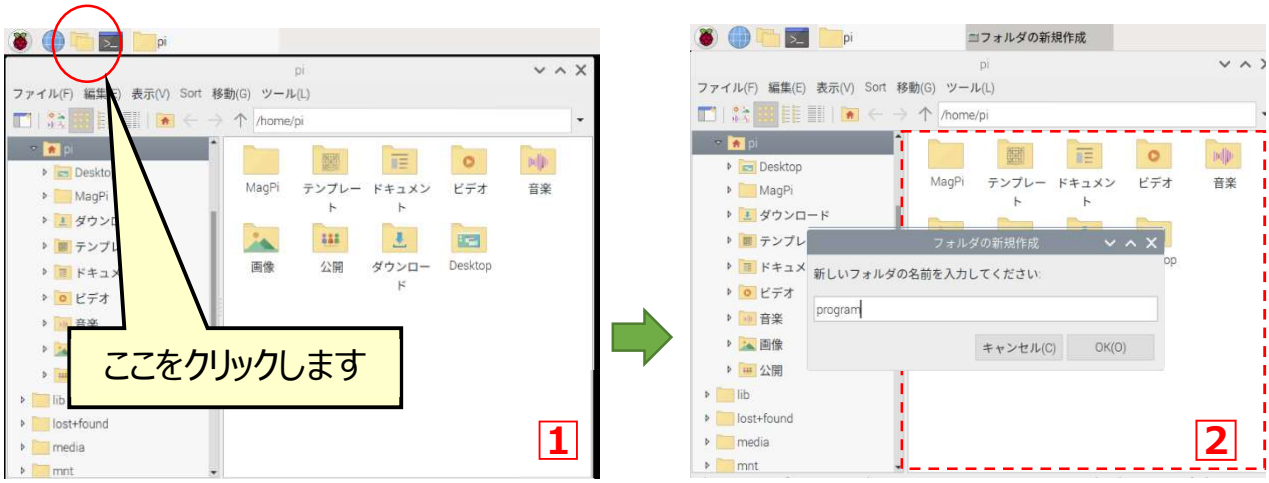
本手順書の通り実行しても、Raspberry Pi の OS アップデート状況やモジュールのアップデートにより、プログラムを実行してもモジュールを利用できないエラーが出る場合があります。この場合は、次のコマンドをターミナルで順に実行すると解決できます。

```
pi@raspberrypi: ~ $ sudo pip3 install -U numpy
pi@raspberrypi: ~ $ sudo pip3 install -U pandas
pi@raspberrypi: ~ $ sudo pip3 install -U matplotlib
pi@raspberrypi: ~ $ sudo apt-get install  libatlas-base-dev
```

（6）【Raspberry Pi で】プログラム言語 Python を使用するための設定

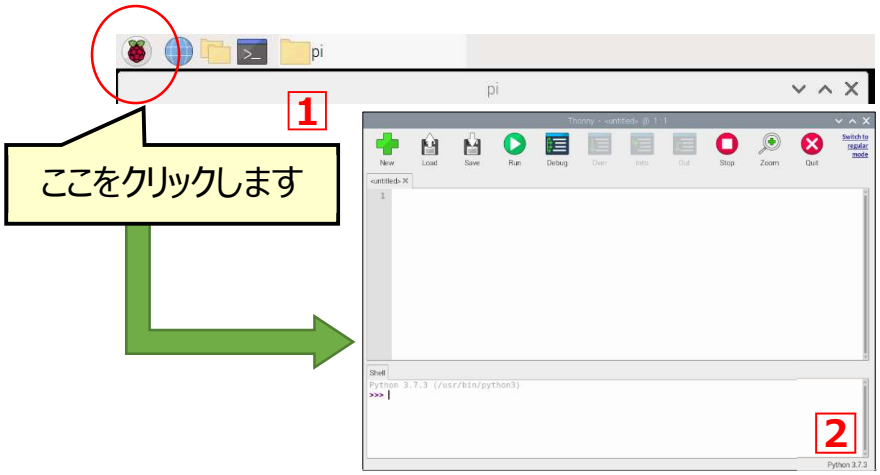
プログラムを入れるフォルダの準備と、プログラムを記述するソフトウェアの準備を行います。

図Ⅳ-18 1 の画面のフォルダをクリックすると pi フォルダが表示されます。Raspberry Pi（の OS である「Raspberry Pi OS」）では、通常使うフォルダは home フォルダの中に作られます。次に、図Ⅳ-18 2 の赤点線のエリア内で右クリックをし、新規作成（または「New Folder」を選択）することで、プログラムを入れる「program」フォルダを作ります。



図Ⅳ-18 プログラムを入れるフォルダの作成

図Ⅳ-19 1 の画面左上のラズベリー（赤い丸で囲まれたアイコン）をクリックし、プログラム⇒Thonny Python IDE をクリックすると図Ⅳ-19 2 の画面が出てきます。このままでも使用できますが、日本語での表示ができるように設定します。

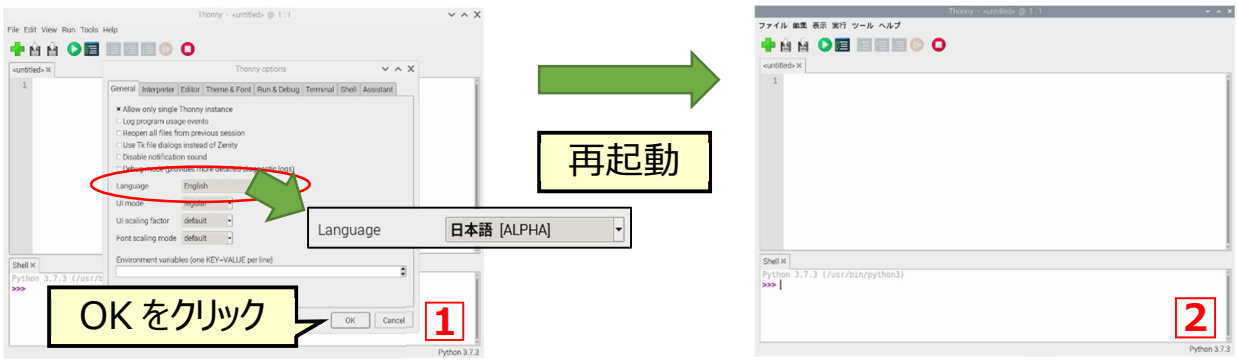


図Ⅳ-19　プログラム記述のための準備①

図Ⅳ-20 1の画面右上にある「Switch to regular mode」をクリックした後、OK を選択後、Thonny（トニー）を終了し、もう一度起動すると、図Ⅳ-20 2の画面になります。

図Ⅳ-20　プログラム記述のための準備②

図Ⅳ-21 1の Option 画面を表示するために、「Tool」から「Options」を選択します。「Language」を「日本語」に変更し、OK を選択した後、Thonny を再起動します。図Ⅳ-21 2のように、日本語化された画面が表示されます。



図Ⅳ-21　プログラム記述のための準備③

図Ⅳ-22 のように、プログラムを実行した際に間違っている箇所などを教えてくれる「アシスタント」を表示します。「表示」から「アシスタント」をクリックすると、画面の右側に「アシスタント」が表示されます。これで Python のプログラムを書くための設定は完了です。

図Ⅳ-22　プログラム記述のための準備④

2. 配布プログラムのダウンロード

以下の手順に沿ってプログラム配布サイトからプログラムをダウンロードします。

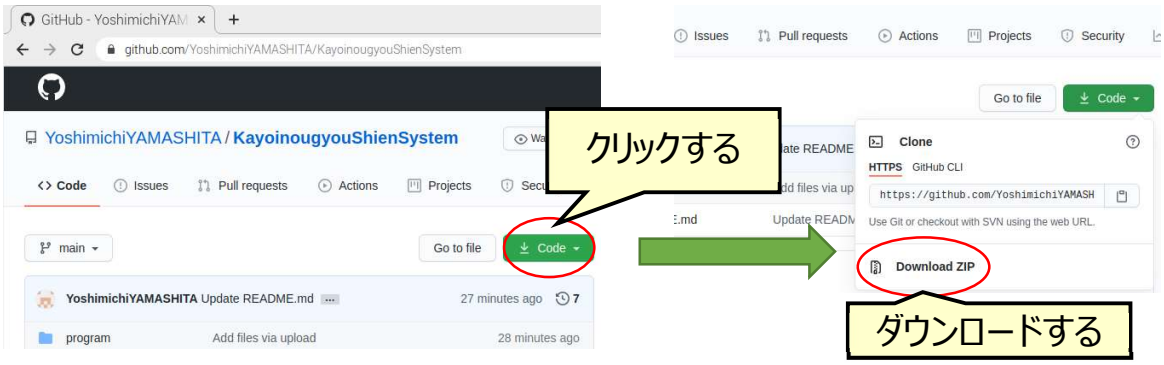
（１）　ブラウザ（Chromium）を開いて、以下の URL にアクセスする

https://github.com/YoshimichiYAMASHITA/KayoinougyouShienSystem にアクセスする（図Ⅳ-23）

図Ⅳ-23　配布プログラムのダウンロード①

（２） プログラムをダウンロードする

GitHub と呼ばれるサイトからプログラムをダウンロードできます。「↓ Code」をクリックして、「Download ZIP」を選択してダウンロードしてください（図Ⅳ-24）。



図Ⅳ-２４　配布プログラムのダウンロード②

pi フォルダ中の Downloads フォルダにプログラムがダウンロードされます。ダウンロードしたファイル「KayoinougyoShienSystem-main.zip」を右クリックして、「ここでファイルを展開」を選択すると圧縮されたファイルが解凍されます（図Ⅳ-25）。

図Ⅳ-25　配布プログラムのダウンロード③

（３） ダウンロードしたプログラムを確認する

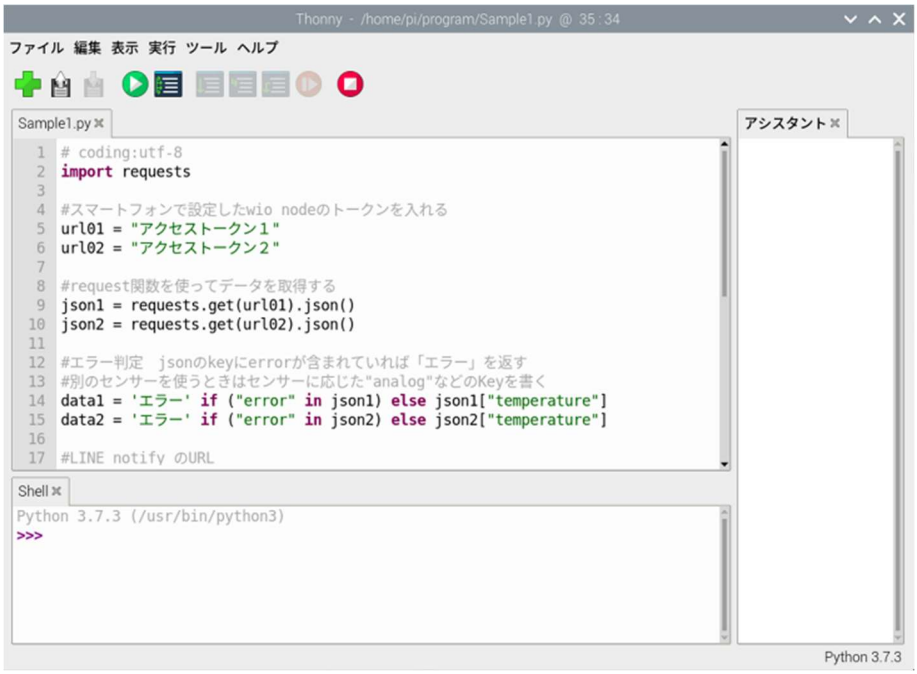
解凍したフォルダには以下のファイルが入っていることを確認してください。これら以外もいくつか資料が付属していますので、適宜確認し利用してください。

program1.py	…	プログラム①「定期通知プログラム」
program2.py	…	プログラム②「警報通知プログラム」
program3.py	…	プログラム③「定期通知プログラム」（再実行あり）
program4.py	…	プログラム④「データ保存プログラム」
Kayoi_data10.py	…	「データ通知プログラム」10分おきにデータを（通知）・保存する
Kayoi_daily_report.py	…	「グラフ通知プログラム」平均値、最大・最小値、グラフを通知する

これらのうち、本手順書では「警報通知プログラム」と「データ通知プログラム」、「グラフ通知プログラム」の３つを使用します。その他のプログラムについては、参考資料にあります「安価かつ簡便にハウスの遠隔監視に使える IoT 機器「通い農業支援システム」製作マニュアル」で詳細に解説しています。

「通い農業支援システム」はこれらの３つのプログラムの内、「データ通知プログラム」と「グラフ通知プログラム」で動かすことを前提としています。データ通知プログラムで 10 分ごとにデータを取得し、通知と保存を行います。グラフ通知プログラムはデータ通知プログラムで保存したデータから、前日の平均値・最大値・最小値とグラフを通知します。

これらのプログラムを p. 30 で作成した「program」フォルダに移動させてください。その後、program1.py をダブルクリックして開くと、図Ⅳ-26 のような画面が開くことを確認します。



図Ⅳ-26 Thonny Python でのプログラムの表示

3.配布プログラムの設定

先ほどダウンロードしたファイルを利用し、ハウスの環境（温度、湿度、地温、土壌水分）を測定するシステムを作成していきます。p．41 でダウンロードした「データ通知プログラム」、「グラフ通知プログラム」、「警報通知プログラム」を使用します。

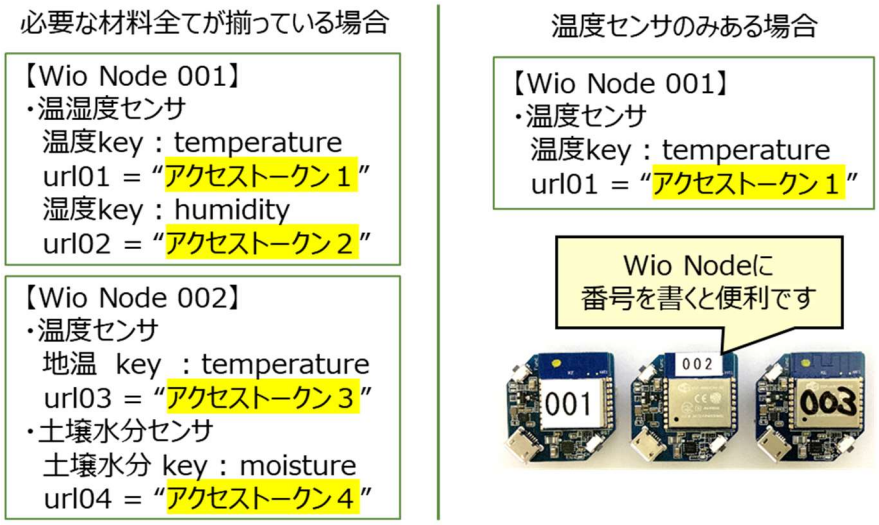
(1)事前準備

①本項で作成する遠隔監視システムの仕様

- ハウスの環境（温度、湿度、地温、土壌水分）を 1 時間毎に定期通知する。
- ハウスの温度が 30 ℃を超えた時に警報通知する。
- ハウスの温度について、前日の「平均温度」と「最高、最低温度」に加えて、前日の 0 時から通知する日の 24 時までの範囲のグラフを 12 時に通知する。
- 測定したデータは 10 分毎に保存する。

② Wio Node とセンサの準備

ハウスの温度、湿度、地温、土壌水分を測定するための資材を準備します（参照 p.11）。Wio Node は 1～2 個用意し、それぞれを「Wio Node 001/002」とします（図Ⅳ-27）。



図Ⅳ-27 Wio Node とセンサの構成

③ 通知する LINE グループの準備

p. 26 を参照し、以下の 3 つのグループを作り、トークンを発行します。

（1）001ハウス情報（定期通知）	…	1時間毎に通知する。
（2）002ハウス情報（日報・グラフ通知）	…	毎日12時に平均温度やグラフを通知する。
（3）003ハウス情報（警報通知）	…	ハウス温度が30 ℃を超えたら通知する。

図Ⅳ-28　LINE グループの構成

④ プログラムの準備を行う

以下の 4 つのプログラムを作成します。

（1）Kayoi_data10.py	…	10分間隔のデータ保存用。通知は行わない。
（2）Kayoi_data60.py	…	60分（1時間）間隔で「001ハウス情報（定期通知）」へ通知する。
（3）Kayoi_daily_report.py	…	（1）で保存した10分間隔データ(data10.txt)から平均温度と最高、最低温度に加えてグラフを作成し、「002ハウス情報（日報・グラフ通知）」へ通知する。
（4）Kayoi_alert.py	…	5分毎に温度をチェックし、ハウス温度が30 ℃を超えた際に「003ハウス情報（警報通知）」へ通知する。

図Ⅳ-29　プログラムの構成

※データを 1 時間毎ではなく、10 分毎に通知する場合は（2）は必要ありません。また、警報通知を必要としない場合は（4）は必要ありません。

p. 41 でダウンロードした「Kayoi_data10.py」は、これを元に「Kayoi_data60.py」を作成するため、事前にコピーを作ってください。次に「program3.py」をコピーして「Kayoi_alert.py」とファイル名を変更します。

(2)データ通知プログラムを作成する

Kayoi_data10.py を開いて、必要な箇所を修正します。以下のプログラムの灰色の網掛け部分を適宜修正します。1 行に収まらない場合などは一部表記を省略、改変しています。

10,13,16,18 行目　対応するアクセストークンに変更
119 行目　通知する LINE グループのアクセストークンに変更

```
1  # coding:utf-8
2  import requests #Web API にアクセスするために用いる
3  import time #時刻の取得に用いる
4  import os #CSV ファイルの保存に用いる
5  from decimal import Decimal, ROUND_HALF_UP #四捨五入に用いる
6
7  #スマートフォンで設定した wio node のトークンを入れる
8  #【Wio Node 001】
9  #  温湿度センサ-温度  key:temperature
10 url01 = "アクセストークン1"
11 #"""
12 #  温湿度センサ-湿度  key:humidity
13 url02 = "アクセストークン2"
14 #【Wio Node 002】
15 #  温度センサ-地温  key:temperature
16 url03 = "アクセストークン3"
17 #  土壌水分センサ-土壌水分  key:moisture
18 url04 = "アクセストークン4"
19 #"""
20
21 #request 関数と for 関数を使って最大 3 回  5 秒おきにデータを取得する
22 CONNECTION_RETRY = 3 #データ取得を行う最大試行数を設定する
23 INTERVAL_TIME = 5 #データ取得失敗時に何秒待つかを設定する
24
25 #data1（json1 の後には設定したセンサにあわせて「key キー」を記入する）
26 #1<= X < 4 なので  3  回試行する
27 for i1 in range(1, CONNECTION_RETRY+1):
28     json1 = requests.get(url01).json()
29     if ("error" in json1) :
30         if("Node is offline" in json1["error"]):
31             data1 = ' OFF'#マイコンか WiFi の電源がオフ
32             break
```

アルファベットの「URL」の小文字に数字の「01」です。

```
33          elif("timeout" in json1["error"]):
34              data1 = 'TIME'#マイコン再起動中か WiFi の調子が悪いです
35              time.sleep(INTERVAL_TIME)
36          elif("METHOD" in json1["error"]):
37              data1 = 'FWUP'#Wio アプリのセンサが正しいかを確認してください
38              break
39          elif("Unknown" in json1["error"]):
40              data1 = ' UNK'#センサが壊れているか、センサが抜けています
41              time.sleep(INTERVAL_TIME)
42          else:
43              data1 = 'FAIL'#データ取得に失敗しました。
44              time.sleep(INTERVAL_TIME)
45      else:
46          data1 = (Decimal(json1["temperature"]).quantize(Decimal('0.1')
            ,rounding=ROUND_HALF_UP))#データを四捨五入します
47          break
48
49  #"""
50  for i2 in range(1, CONNECTION_RETRY+1):
51      json2 = requests.get(url02).json()
52      if ("error" in json2) :
53          if("Node is offline" in json2["error"]):
54              data2 = ' OFF'#略
55              break
56          elif("timeout" in json2["error"]):
57              data2 = 'TIME'#略
58              time.sleep(INTERVAL_TIME)
59          elif("METHOD" in json2["error"]):
60              data2 = 'FWUP'#略
61              break
62          elif("Unknown" in json2["error"]):
63              data2 = ' UNK'#略
64              time.sleep(INTERVAL_TIME)
65          else:
66              data2 = 'FAIL'#略
67              time.sleep(INTERVAL_TIME)
68      else:
69          data2 = (Decimal(json2["humidity"]).quantize(Decimal('0.1')
            ,rounding=ROUND_HALF_UP))#データを四捨五入します
70          break
71
```

```
72  for i3 in range(1, CONNECTION_RETRY+1):
73      json3 = requests.get(url03).json()
74      if ("error" in json3) :
75          if("Node is offline" in json3["error"]):
76              data3 = ' OFF'#略
77              break
78          elif("timeout" in json3["error"]):
79              data3 = 'TIME'#略
80              time.sleep(INTERVAL_TIME)
81          elif("METHOD" in json3["error"]):
82              data3 = 'FWUP'#略
83              break
84          elif("Unknown" in json3["error"]):
85              data3 = ' UNK'#略
86              time.sleep(INTERVAL_TIME)
87          else:
88              data3 = 'FAIL'#略
89              time.sleep(INTERVAL_TIME)
90      else:
91          data3=(Decimal(json3["temperature"]).quantize(Decimal('0.1')
            ,rounding=ROUND_HALF_UP))#データを四捨五入します
92          break
93
94  for i4 in range(1, CONNECTION_RETRY+1):
95      json4 = requests.get(url04).json()
96      if ("error" in json4) :
97          if("Node is offline" in json4["error"]):
98              data4 = ' OFF'#略
99              break
100         elif("timeout" in json4["error"]):
101             data4 = 'TIME'#略
102             time.sleep(INTERVAL_TIME)
103         elif("METHOD" in json4["error"]):
104             data4 = 'FWUP'#略
105             break
106         elif("Unknown" in json4["error"]):
107             data4 = ' UNK'#略
108             time.sleep(INTERVAL_TIME)
109         else:
110             data4 = 'FAIL'#略
111             time.sleep(INTERVAL_TIME)
```

```
    else:
        data4=(Decimal(json4["moisture"]).quantize(Decimal('1')
        ,rounding=ROUND_HALF_UP))#データを四捨五入します
        break
#"""

#LINE notify の URL
url99 = "https://notify-api.line.me/api/notify"
token = '(1)001 ハウス情報(定期通知)のトークン' #LINE notify のトークン

#"""
#LINE に通知するメッセージを作る
message  = '¥n '
#'センサ名:'+str(data 番号)+'単位'+'¥n' センサ名、データ番号、単位を書く
message += '■ハウス環境'+'¥n'
message += '  温度:'+str(data1)+' ℃'+'¥n'
#"""
message += '  湿度:'+str(data2)+' %RH'+'¥n'
message += '  地温:'+str(data3)+' ℃'+'¥n'
message += '土壌水分:'+str(data4)+''
#"""

#LINE グループに通知を行う
payload = {'message' : message}
headers = {'Authorization' : 'Bearer '+ token}
r = requests.post(url99, data=payload, headers=headers)
#"""

#CSV で保存する
#CSV の保存のために時刻を取得
timestamp = 'date +%F" %H:%M"'
current_time = os.popen(timestamp).readline().strip()
#CSV で保存するデータを組み合わせる
data_set  = str(current_time)
data_set += ',' + str(data1)
#"""
data_set += ',' + str(data2)
data_set += ',' + str(data3)
data_set += ',' + str(data4)
#"""
data_set += '¥n'
```

```
152
153 #/home/pi/data.txt というファイルにデータを保存する
154 fout = open('/home/pi/data10.txt','at')
155 fout.write(data_set)
156 fout.close()
157
```

現在、「10 分間隔のデータを保存し、通知を行う」というプログラムになっています。p. 37 で説明した以下のプログラムを完成するには、通知を行わないようにする必要があります。

（1）Kayoi_data10.py　… 10 分間隔のデータ保存用。通知は行わない。

p. 43 で用意したセンサが「温度センサ」だけだった場合なども含めて、次ページの修正チャートを確認しながら、パターンに応じて修正を行ってください。

【パターン１：10 分間隔のデータを保存し、通知を行う」】

そのままで修正なし。通知間隔が 10 分で問題ない場合はこれで終了です。

【パターン２：10 分間隔のデータを保存し、通知は行わない】

目的の設定です。修正チャートに従って、（1）Kayoi_data10.py を完成してください。

【パターン３：温度センサが１つだけの場合で、通知を行う】

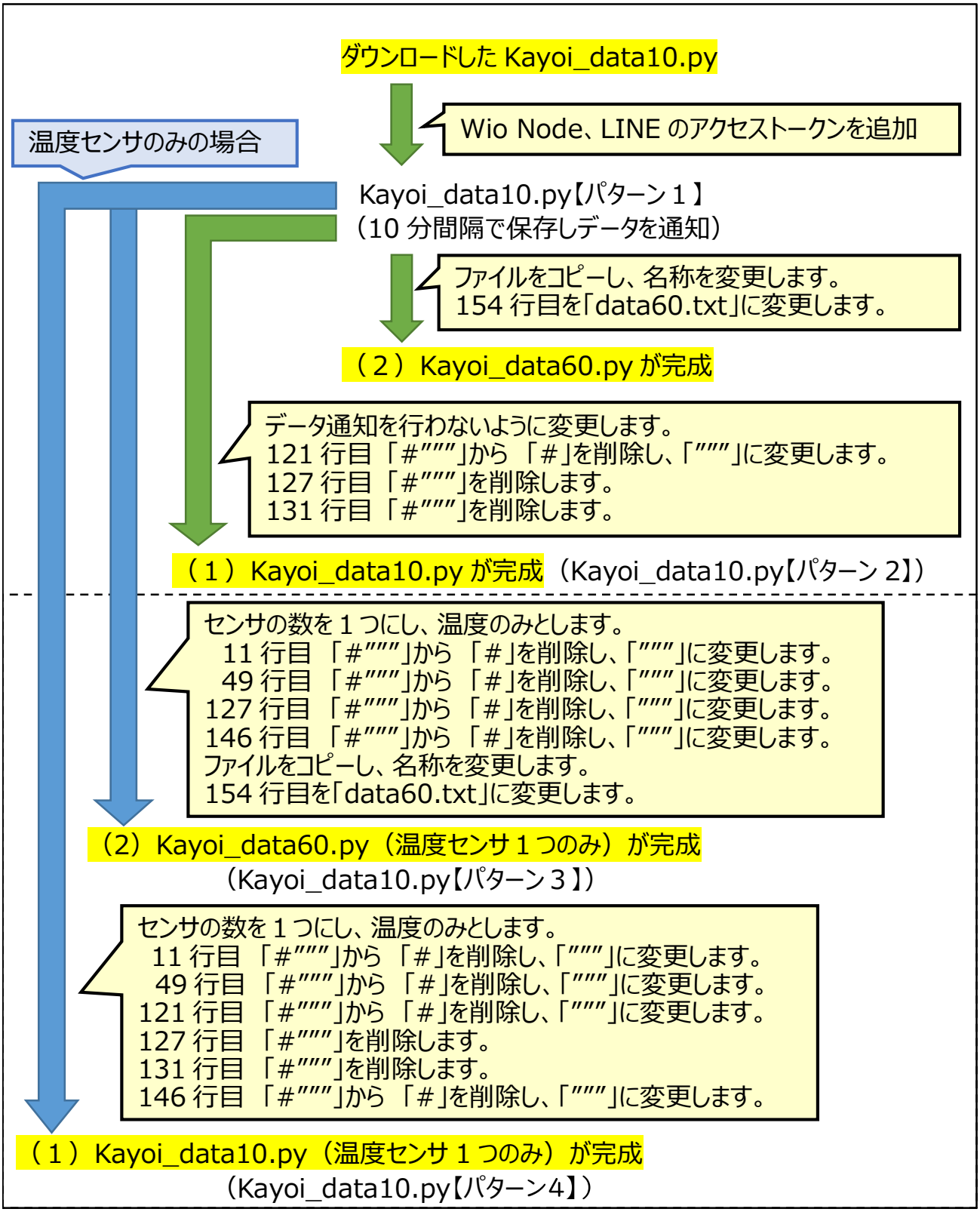
修正チャートに従って修正してください。

【パターン４：温度センサが１つだけの場合で、通知を行わない】

修正チャートに従って修正してください。

（2）Kayoi_data60.py…60 分（1 時間）間隔で「001 ハウス情報（定期通知）」に通知を作成します。

上記（1）Kayoi_data10.py のパターン１を修正して作成します。 154 行目 fout = open('/home/pi/data10.txt','at') の「data10.txt」を「data60.txt」に変更する。Kayoi_data10.py をコピーし、ファイル名を「Kayoi_data60.py」に変更する。



図Ⅳ-30 Kayoi_data10.py の修正チャート

参考：データ通知プログラムを活用する方法

- エラーメッセージからメンテナンスを行う

p. 45-46 の 29 行目から 44 行目を以下に抜き出しました。データ取得時に「エラー」となった場合、そのエラーの原因から通知するメッセージを変えています。

```
if ("error" in json1) :
    if("Node is offline" in json1["error"]):
        data1 = ' OFF'#マイコンか WiFi の電源がオフ
        break
    elif("timeout" in json1["error"]):
        data1 = 'TIME'#マイコン再起動中か WiFi の調子が悪いです
        time.sleep(INTERVAL_TIME)
    elif("METHOD" in json1["error"]):
        data1 = 'FWUP'#Wio アプリのセンサが正しいかを確認してください
        break
    elif("Unknown" in json1["error"]):
        data1 = ' UNK'#センサが壊れているか、センサが抜けています
        time.sleep(INTERVAL_TIME)
    else:
        data1 = 'FAIL'#データ取得に失敗しました
        time.sleep(INTERVAL_TIME)
```

プログラム中にもコメントで記載していますが、以下を参考にメンテナンスしてください。

OFF：マイコンか Wi-Fi の電源がオフなので、電源を確認します。

TIME：データ取得に時間がかかっているため、Wi-Fi の位置を調整します。

FWUP：設定センサと異なるセンサが接続されています。

UNK：センサが壊れているか、センサが抜けています。

FAIL：データ取得に失敗しました。Wi-Fi が混雑している可能性があります。

● データを四捨五入する

データを取得した際、温度センサは小数第 2 位までの値が表示されることがあります。しかし、実際には小数第 2 位まで必要な場合は少ないことと、通知時に数字の数がバラバラだと判断しにくいので、データを四捨五入して表示しています。たとえば、22 ℃前後の値の際、以下の 2 つのパターンのように表示されると読みにくいので、小数第 1 位までの値に変更します。

22.3 ℃
22.24 ℃

22.3 ℃
22.2 ℃

1 にする⇒小数第 1 位を四捨五入して表示
0.1 にする⇒小数第 2 位を四捨五入して表示
0.01 にする⇒小数第 3 位を四捨五入して表示

```
data1 = (Decimal(json1["temperature"]).quantize(Decimal('0.1')
,rounding=ROUND_HALF_UP))#データを四捨五入します
```

※実際のプログラム上では 2 行にせず、以下のように 1 行になります。

```
data1 = (Decimal(json1["temperature"]).quantize(Decimal('0.1'),rounding=ROUND_HALF_UP))#データを四捨五入します
```

これらを参考に、必要に応じてプログラムを修正して使用してください。

(3)グラフ通知プログラムを作成する

Kayoi_daily_report.py を開いて、必要な箇所を修正します。プログラムの灰色の網掛け部分（147 行）を修正します。1 行に収まらない場合は一部表記を省略、改変しています。

147 行目　通知する LINE グループのアクセストークンに変更
（（2）002 ハウス情報（日報・グラフ通知）用のアクセストークン）

```
1  # coding: utf-8
2  import pandas as pd #pandas をインポートする
3  import matplotlib.pyplot as plt #グラフを描画するのに用いる
4  import matplotlib as mpl #フォント名を指定するためにインポート
5  from datetime import date,timedelta #日付を扱うための標準ライブラリ
6  import requests
7  import datetime
8  from decimal import Decimal, ROUND_HALF_UP #四捨五入に用いる
9
10 #  読み込む csv ファイル名（絶対パスを推奨）
11 csv_file = "/home/pi/data10.txt"
12
13 # data10.txt にはヘッダーが無いので、列に名前をつける
14 col01 = "温度"
15 #"""
16 col02 = "湿度"
17 col03 = "地温"
18 col04 = "土壌水分"
19 #"""
20 colums_name = ["datetime"]
21 colums_name.append(col01)
22 #"""
23 colums_name.append(col02)
24 colums_name.append(col03)
25 colums_name.append(col04)
26 #"""
27
28 # csv_file をデータフレーム型で読み込む
29 df = pd.read_csv(csv_file, names = (colums_name),  index_col =
   "datetime", parse_dates = True, encoding = "UTF8")
30
31 #  データが取れなかった場合の警告があるところを NaN にする
```

```
df = df.replace(' OFF', 'NaN')
df = df.replace('TIME', 'NaN')
df = df.replace('FWUP', 'NaN')
df = df.replace(' UNK', 'NaN')
df = df.replace('FAIL', 'NaN')
# NaN を含む行を除外する
df = df.dropna(how = "any")
#  データが文字列になっているので float に変更
df = df.astype(float)

#  基準日を算出する（通常はプログラム実行日）
d = datetime.date.today()
#  日時を指定する方法
#d = date(2020,12,30)

#【基準日を含めた合計 48 時間を出力する】
d1 = d + timedelta(days = -1)#  基準日の前日を算出する
d2 = d + timedelta(days = +1)#  基準日の翌日を算出する

#matplotlib のフォントを日本語にする
mpl.rcParams['font.family'] = "Noto Sans CJK JP"

#全ての項目をグラフ化&画像として保存
#グラフ1：ハウス温度（1 つのグラフで）
df.plot.line( y = ["温度"], subplots = False, grid = True, figsize=(8,8), xlim
= [d1,d2], ylim = [0,40], colormap = "Set1")
plt.savefig("graph1.png")

#"""
#グラフ2：ハウス湿度（1 つのグラフで）
df.plot.line( y = ["湿度"], subplots = False, grid = True, figsize=(8,8), xlim
= [d1,d2], ylim = [0,100], colormap = "Set1")
plt.savefig("graph2.png")

#グラフ3：地温（1 つのグラフで）
df.plot.line( y = ["地温"], subplots = False, grid = True, figsize=(8,8), xlim
= [d1,d2], ylim = [0,40], colormap = "Set1")
plt.savefig("graph3.png")

#グラフ4：土壌水分（1 つのグラフで）
df.plot.line( y = ["土壌水分"], subplots = False, grid = True, figsize=(8,8),
```

```
    xlim = [d1,d2], ylim = [0,1023], colormap = "Set1")
70  plt.savefig("graph4.png")
71
72  #グラフ 5:ハウス温度、ハウス湿度、地温、土壌水分(それぞれのグラフを 1 枚で)
73  df.plot.line( y = ["温度","湿度","地温","土壌水分"], subplots = True, grid =
    True, figsize=(8,8), xlim = [d1,d2], colormap = "Set1")
74  plt.savefig("graph5.png")
75
76  #グラフ 6:ハウス温度、地温(1つのグラフで)
77  df.plot.line( y = ["温度","地温"], subplots = False, grid = True,
    figsize=(8,8), xlim = [d1,d2], ylim = [0,40], colormap = "Set1")
78  plt.savefig("graph6.png")
79  #"""
80
81  # DatetimeIndex の秒を切り捨て floor
82  print(df.index)
83  df.index = df.index.floor("min")
84  print(df.index)
85
86  # 【統計量を計算する(通知日から基準日までの区間)】
87  d_td = d
88  d_ld = d_td + timedelta(days = -1) # 基準日(通知日 1 日前とする)
89
90  # df.index の中身は timestamp 型であり、通知日で設定した date 型との比較では
    怒られるので、timestamp 型に変換する
91  d_td = pd.to_datetime(d_td)
92  d_ld = pd.to_datetime(d_ld)
93
94  #【統計量を計算する】
95  #平均、最高、最低を算出する。日平均気温は 1 時~24 時までの毎正時の平均値なの
    で、60min データを df3 とする
96  #24 時(00:00:00)のデータを計算に入れるために全体を 10 分前にずらす(8/10
    00:00:00 -> 8/9 23:50:00)
97  df2 = df
98  df3 = df
99  df2= df[(df.index >= d_ld) & (df.index <= d_td)]
100 df2.index = df2.index + timedelta(minutes = -10)
101
102 # 指定した範囲で df を抽出(置換)
103 df2= df2[(df2.index >= d_ld) & (df2.index <= d_td)]#10min ずらしたことで
    出てくる前後+1 日を削除
```

```
# 10 分データから最大値・最小値を算出する
df_max = df2.resample("1D").max()
df_min = df2.resample("1D").min()

#【平均値を算出するために 60min データを作成する】
df3 = df[(df.index >= d_ld) & (df.index <= d_td)]
df3 = df3.asfreq('1H') #60min データを作成
#  全体を 1 時間前にずらす（例：8/10 00:00:00 -> 8/9 23:00:00）
df3.index = df3.index + timedelta(hours = -1)
#  指定した範囲で df を抽出（置換）
df3 = df3[(df3.index >= d_ld) & (df3.index <= d_td)]

print(df3)

# 60 分データから日平均値を算出する
df_mean = df3.resample("1D").mean()

print(df_mean)

#  全データ（np_mean[日付 index, データ項目 column]の2次元配列）
np_mean = df_mean.values
np_max = df_max.values
np_min = df_min.values

print(np_mean)

#  前日のみ抽出 （インデックス-1 は最後の意味）
np_mean_yd = np_mean[-1]
np_max_yd = np_max[-1]
np_min_yd = np_min[-1]

#  変数に入れる(col1 を指定したい。0 から始まるので、1 列目のハウス内温度 1 は 0,2 列目の・・・は 1 のようになる）
#ハウスの温度
house_temp_mean_1 = Decimal(np_mean[-1, 0]).quantize(Decimal('0.1'), rounding = ROUND_HALF_UP)
house_temp_max_1 = np_max[-1, 0]
house_temp_min_1 = np_min[-1, 0]

#  日付を index から取得する
index = df_mean.index.date
```

```
index_str = df_mean.index.strftime('%Y-%m-%d')

# LINE notify の URL
url99 = "https://notify-api.line.me/api/notify"
token = '(2)002 ハウス情報(日報・グラフ通知)用のアクセストークン'

# LINE グループに通知を行う(1枚目の画像を送る)
message = '¥n'
message+= "■ハウス温度" + '¥n'
message+= str(index[-1]) + '¥n'
message+= '平均温度' + str(house_temp_mean_1) + " ℃" + '¥n'
message+= '最高温度' + str(house_temp_max_1) + " ℃" + '¥n'
message+= '最低温度' + str(house_temp_min_1) + " ℃"

payload = {'message' : message}
headers = {'Authorization' : 'Bearer '+ token}
files = {"imageFile":open("graph1.png","rb")}
r = requests.post(url99, data=payload, headers=headers, files = files)

#"""
# LINE グループに通知を行う(2枚目の画像を送る)
message = '¥n'
message+= "■ハウス湿度" + '¥n'
message+= str(index[-1])

payload = {'message' : message}
headers = {'Authorization' : 'Bearer '+ token}
files = {"imageFile":open("graph2.png","rb")}
r = requests.post(url99, data=payload, headers=headers, files = files)

# LINE グループに通知を行う(3枚目の画像を送る)
message = '¥n'
message+= "■地温" +'¥n'
message+= str(index[-1])

payload = {'message' : message}
headers = {'Authorization' : 'Bearer '+ token}
files = {"imageFile":open("graph3.png","rb")}
r = requests.post(url99, data=payload, headers=headers, files = files)

# LINE グループに通知を行う(4枚目の画像を送る)
```

```
message = '¥n'
message+= "■土壌水分" + '¥n'
message+= str(index[-1])

payload = {'message' : message}
headers = {'Authorization' : 'Bearer '+ token}
files = {"imageFile":open("graph4.png","rb")}
r = requests.post(url99, data=payload, headers=headers, files = files)

# LINE グループに通知を行う(5枚目の画像を送る)
message = '¥n'
message+= "ハウス情報まとめ" + '¥n'
message+= str(index[-1])

payload = {'message' : message}
headers = {'Authorization' : 'Bearer '+ token}
files = {"imageFile":open("graph5.png","rb")}
r = requests.post(url99, data=payload, headers=headers, files = files)

# LINE グループに通知を行う(6枚目の画像を送る)
message = '¥n'
message+= "ハウス温度と地温" + '¥n'
message+= str(index[-1])

payload = {'message' : message}
headers = {'Authorization' : 'Bearer '+ token}
files = {"imageFile":open("graph6.png","rb")}
r = requests.post(url99, data=payload, headers=headers, files = files)
#"""

```

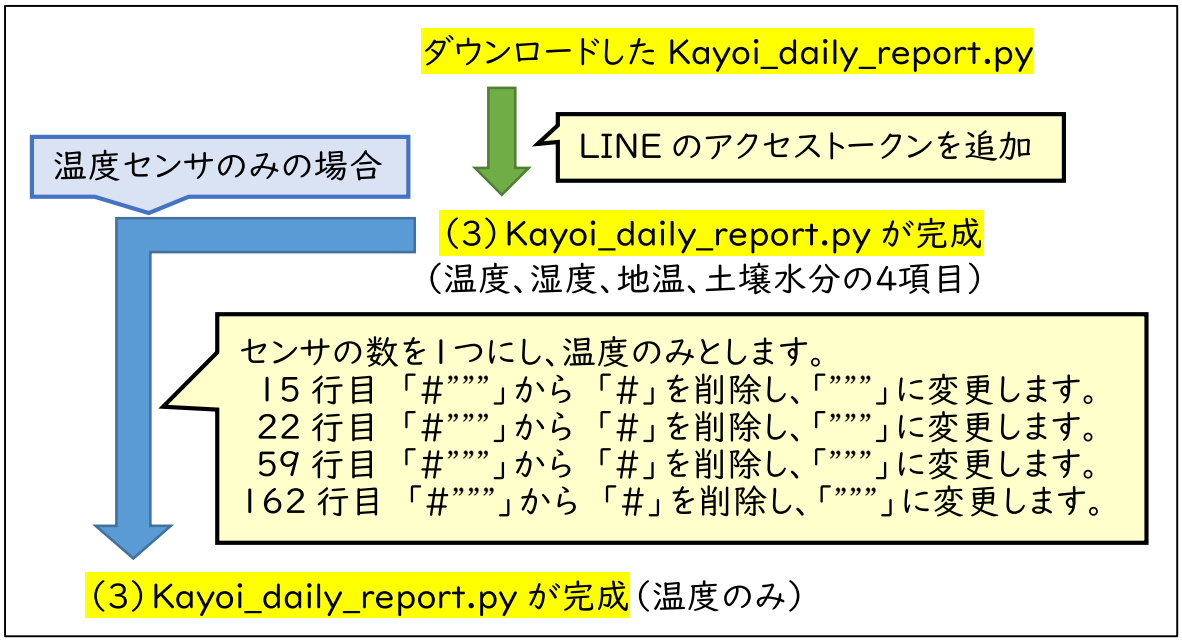
これで、（１）Kayoi_data10.py を実行することで作られる data10.txt から、平均温度や最高、最低温度に加えてグラフを通知するプログラム（3）Kayoi_daily_report.py が作成できました。p. 43 で用意したセンサが「温度センサ」だけだった場合は、次ページの修正チャートを確認しながら、以下のパターンに応じて修正を行ってください。

【パターン１：必要なセンサが全てそろっている場合】

そのままで修正なし。（3）Kayoi_daily_report.py は完成です。

【パターン２：温度センサ１つの場合】

温度センサ以外の部分を全てコメントに変えます。修正チャートに従って変更してください。



図Ⅳ-31　Kayoi_daily_report.py の修正チャート

参考：グラフの設定を変更する

温度のグラフを通知している、56-57 行目を例に説明します。

```
df.plot.line( y = ["温度"], subplots = False, grid = True, figsize=(8,8),
xlim = [d1,d2], ylim = [0,40], colormap = "Set1")
plt.savefig("graph1.png")
```

- 表示するデータを選択する

14-18 行目で設定した、それぞれのデータ名を、y=["データ名"]の中に書き込みます。今回は「温度」なので、y=["温度"]としています。たとえば、温度と地温を 1 つのグラフに表示する場合は、y=["温度","地温"]とします。

- 表示するデータを選択する

複数の項目（たとえば、温度と地温など）を表示する際に、複数の項目を 1 つのグラフの中に表示するか、項目ごとに複数のグラフを並べて表示するかを変更できます。

subplots = False　…　1 つのグラフで表示する

subplots = True　…　複数のグラフを並べて表示する

- Y 軸の範囲を変更する

今回は「ylim = [0,40]」と表示されています。これは Y 軸を 0 から 40 の間に設定していることとなります。たとえば、冬の間にハウス内で果樹などの永年性作物を越冬させたい場合は、氷点下になることもあります。その場合は「ylim = [-10,30]」とすると、Y 軸を-10 から 30 の間に設定することができます。土壌水分の場合は、ylim = [0,1023]としていますが、これは土壌水分センサが水分に応じて 0 から 1023 の間に値を通知するからです。

- グラフの線の色を変更したい

「colormap = "Set1"」を変更することで、線の色を変更することができます。Set1,Set2,Set3,tab10 などがありますので、変更して見て下さい。また、以下の URL に選択できるカラーマップについて詳しく書いてあります。英語で書かれていますが、カラーチャートとともに表示されているので、いくつか試してみることをお勧めします。

https://matplotlib.org/3.3.4/tutorials/colors/colormaps.html

- 通知したグラフの画像を保存したい

plt.savefig("graph1.png")と書いてありますが、graph1.png がグラフの画像です。これは「グラフ通知プログラム」を実行したフォルダに保存されます。今回は program フォルダ内にグラフ画像は保存されます。

- 平均温度の値をより正確に計算したい

Kayoi_data10.py では、プログラムの後半（141-142 行目）に時刻を取得し、データを保存しています。そのため、crontab（p. 66）で指定した実行時刻よりも少し遅れることがあり、遅れた場合の値は除外されたまま計算されてしまいます。そのため、プログラムの実行と同時に時刻を取得するように変更（たとえば 141-142 行目の内容を 6 行目に変更）することで、より正確に平均温度を計算することができます。

(4) 警報通知プログラムを作成する

program2.py を（4）Kayoi_alert.py に名前を変更します。

5 行目のアクセストークンを「温度センサ用のアクセストークン」に変更
16 行目は通知する「（3）003 ハウス情報（警報通知）」のトークンに変更
29 行目の設定温度を 30.0 に変更し、32、33 行目に「#」をつけコメントに変更

```
1  # coding:utf-8
2  import requests
3
4  #スマートフォンで設定した wio node のトークンを入れる
5  url01 = "アクセストークン1"
6
7  #request 関数を使ってデータを取得する
8  json1 = requests.get(url01).json()
9
10 #"temperature"のデータが取れたときに data に温度を入れる
11 data1 = json1["temperature"]
12
13 #LINE notify の URL
14 #LINE notify のトークン（通知先に応じて変更すること）
15 url99 = "https://notify-api.line.me/api/notify"
16 token = '通知する LINE グループのアクセストークン'
17
18 #高温用メッセージ
19 message1 = '高温注意!:'+str(data1)+'℃'
20 payload1 = {'message' : message1}
21
22 #低温用メッセージ
23 message2 = '低温注意!:'+str(data1)+'℃'
24 payload2 = {'message' : message2}
25
26 headers = {'Authorization' : 'Bearer '+ token,}
27
28 #警報を出したいときの温度を設定する
29 if data1 > 20.0:#高温用：この温度より「高い」と通知する
30     r = requests.post(url99,data=payload1,headers=headers)
31
32 if data1 < 10.0:#低温用：この温度より「低い」と通知する
33     r = requests.post(url99,data=payload2,headers=headers)
```

（4）Kayoi_alert.py は完成ですが、データ取得時にエラーだった場合に再実行するように修正する場合は、図Ⅳ-32 のように 7 行目から 11 行目を削除し、Program3.py の 14-24 行目をコピー＆ペーストしてください。なお、「1 2 3」回試行するとは（123 回ではなく、1 回、2 回、3 回と最大 3 回実行することを表しています。

```
#request 関数を使ってデータを取得する
json1 = requests.get(url01).json()

#"temperature"のデータが取れたときに data に温度を入れる
data1 = json1["temperature"]
```

データ取得時にエラーであった場合、
再実行するように変更

```
#data1
#1<= X < 4 なので 1 2 3 回試行する
for i1 in range(1, CONNECTION_RETRY+1):
    json1 = requests.get(url01).json()
    if ("error" in json1) :
        data1 = 'エラー'
#         print('data 1 error!')
        time.sleep(INTERVAL_TIME)
    else:
        data1 = json1["temperature"]
        break
```

図Ⅳ-32 データ取得時に再実行するように修正する方法

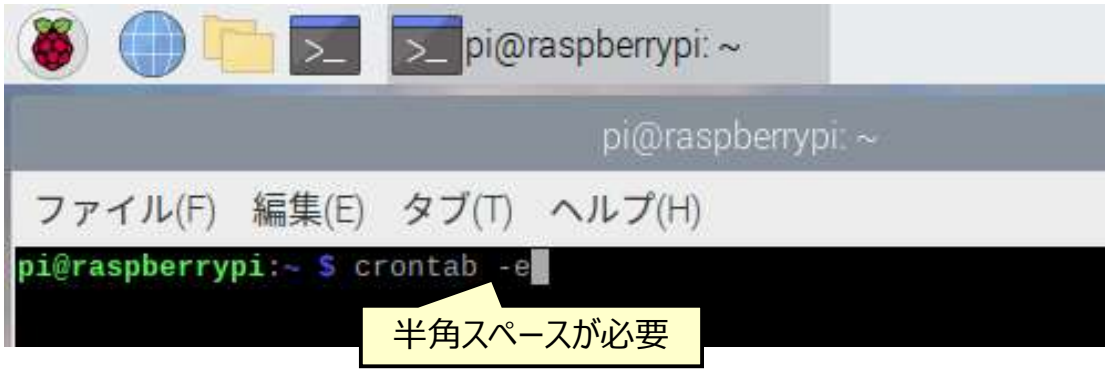
4.プログラムの自動実行設定

Raspberry Pi には、crontab（クロンタブ）と呼ばれる、プログラムを決めた時間に実行してくれる機能が搭載されています。これを使って、先ほど作成したプログラムを自分の好きなタイミングで動かすことができます。

(1) crontab の初期設定を行う

① ターミナルから crontab を開く

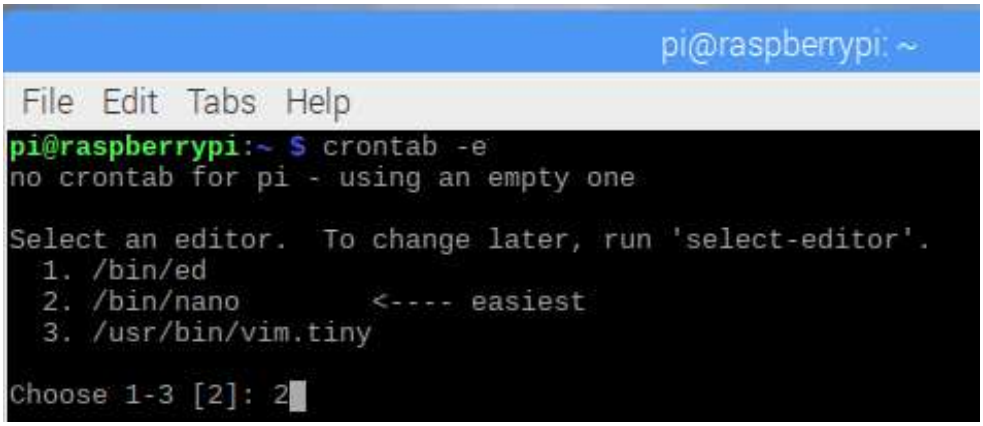
crontab -e と打って、Enter を押してください（図Ⅳ-33）。



図Ⅳ-33 crontab の設定方法①

② crontab の初期設定をする。（２回目以降は初期設定画面が出てきません）

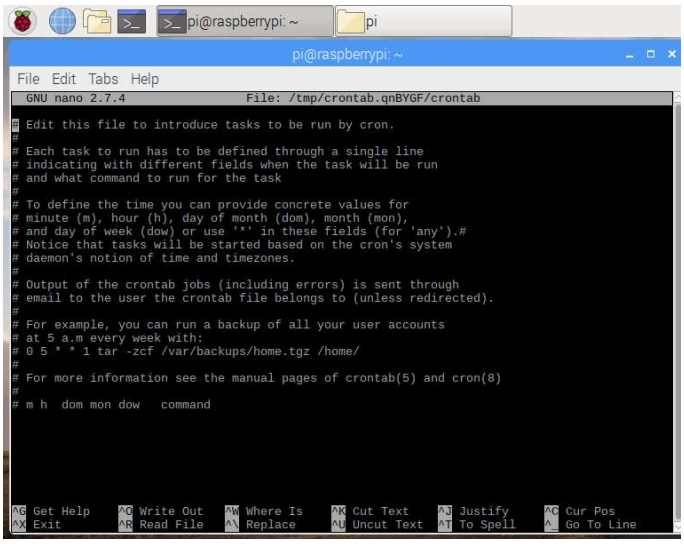
/bin/nano（と呼ばれるエディタ）を使います。画面の表示をよく見て、該当する番号を打ち Enter を押してください。（この画面の場合は「２」です。（図Ⅳ-34））



図Ⅳ-34 crontab の設定方法②

③ 図Ⅳ-35 になりましたら初期設定完了です。次に実行間隔の設定に移ります。

Ctrl を押しながら X を押すと終了し、元の画面に戻ります。

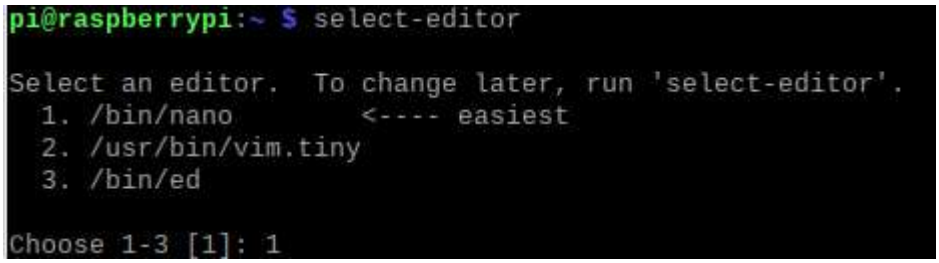


図Ⅳ-35 crontab の設定方法③

④ 誤って「②」の操作で/bin/nano 以外を選択してしまった場合

ターミナルを一度終了した後、ターミナルを再度起動して、以下の下線部のコマンドを実行し、/bin/nano を選択（番号 1 を入力）してください（図Ⅳ-36）。

pi@raspberrypi: ~ $ select-editor



図Ⅳ-36 crontab の設定方法④

(2) crontab を使ってプログラムの実行間隔を設定する

① ターミナルから crontab を開く

crontab -e と打って、Enter を押してください。以下の画面の□で囲んだ部分に実行間隔を設定する文を書きます。たとえば、次のように書くと、図Ⅳ-37 のようになります。

*/2 * * * * sudo python3 /home/pi/program/test.py

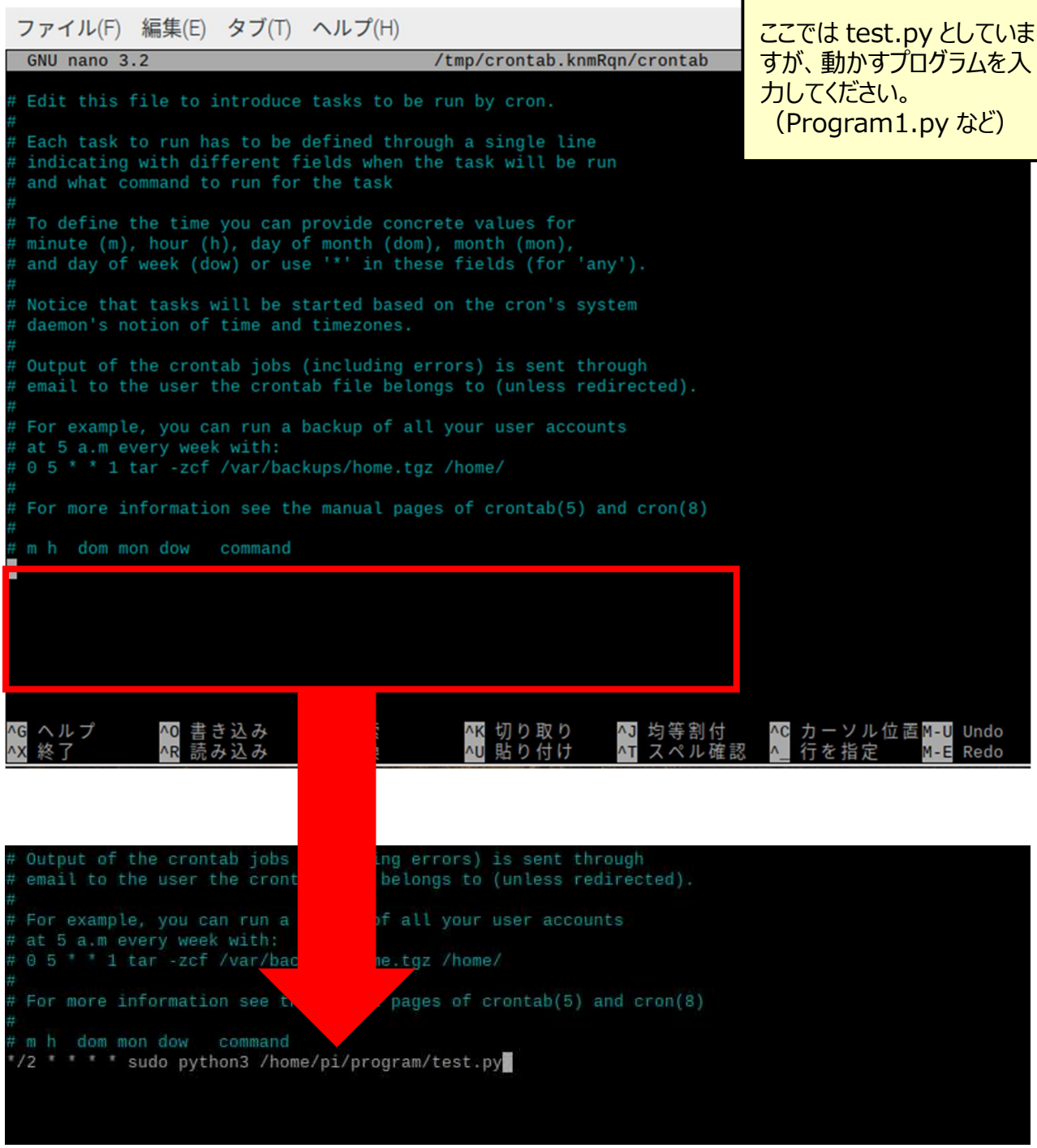
2 分間隔で実行

プログラムは Python を使う

プログラムのあるところ program はフォルダ名

ファイル名

ここでは test.py としていますが、動かすプログラムを入力してください。（Program1.py など）



図Ⅳ-37 crontab の設定方法⑤

② 作成したプログラムを定期実行できるように crontab を設定する

p. 44 で取り決めた設定で各プログラムを実行するようにします。

- LINE グループの設定

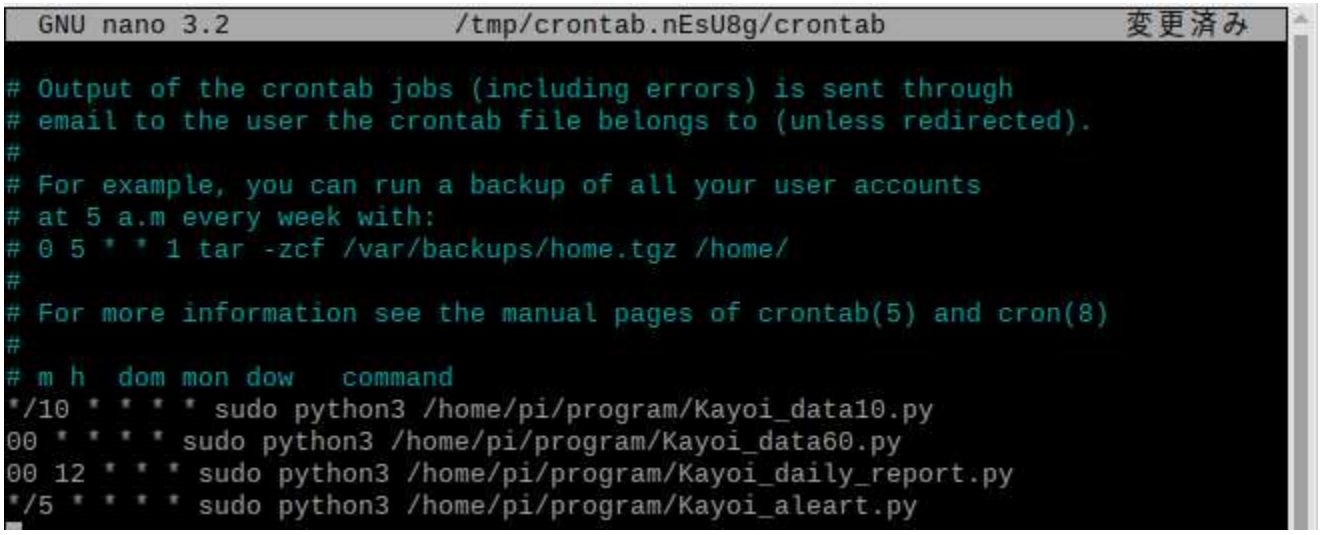
（1）001ハウス情報（定期通知）	…	1時間毎に通知する
（2）002ハウス情報（日報・グラフ通知）	…	毎日12時に平均温度やグラフを通知する
（3）003ハウス情報（警報通知）	…	ハウス温度が30 ℃を超えたら通知する

- プログラムの設定

（1）Kayoi_data10.py	…	10分間隔のデータ保存用。通知は行わない。
（2）Kayoi_data60.py	…	60分（1時間）間隔で「001ハウス情報（定期通知）」へ通知する。
（3）Kayoi_daily_report.py	…	（1）で保存した10分間隔データ(data10.txt)から平均温度と最高、最低温度に加えてグラフを作成し、「002ハウス情報（日報・グラフ通知）」へ通知する。
（4）Kayoi_alert.py	…	5分毎に温度をチェックし、ハウス温度が30 ℃を超えた際に「003ハウス情報（警報通知）」へ通知する。

上記①で開いた crontab を以下のように設定します（図Ⅳ-37）。

```
*/10 * * * * sudo python3 /home/pi/program/Kayoi_data10.py
00 * * * * sudo python3 /home/pi/program/Kayoi_data60.py
00 12 * * * sudo python3 /home/pi/program/Kayoi_daily_report.py
*/5 * * * * sudo python3 /home/pi/program/Kayoi_aleart.py
```



図Ⅳ-38 crontab の設定方法⑥

③ 警報通知を使用しない場合

先頭に＃「シャープ」をつけることで警報通知は動かなくなります。

```
#*/5 * * * * sudo python3 /home/pi/program/Kayoi_aleart.py
```

参考：crontab を用いた詳細な設定例

- 10 分おきに通知

  */10 * * * * sudo python3 以下略

- 1 時間おきに通知

  00 * * * * sudo python3 以下略

- 6 時から 18 時まで 10 分おきに通知

  */10 6-18 * * * sudo python3 以下略

- 6 時、9 時、10 時、12 時、13 時、14 時、15 時、22 時に通知

  00 6,9,10,12,13,14,15,16,22 * * * sudo python3 以下略

④ crontab の設定を保存する

キーボードの「Ctrl」キーを押しながら X を押します。保存するか聞かれるので「y」を押し、その次は「Enter」を押します。以下のような画面が出れば保存となります。

図Ⅳ-39 crontab の設定方法⑦

Ⅴ. 現地での設置方法

本項では現場に実際に設置する際の方法や使用できるセンサについて説明します。

1. Wio Node の防水方法

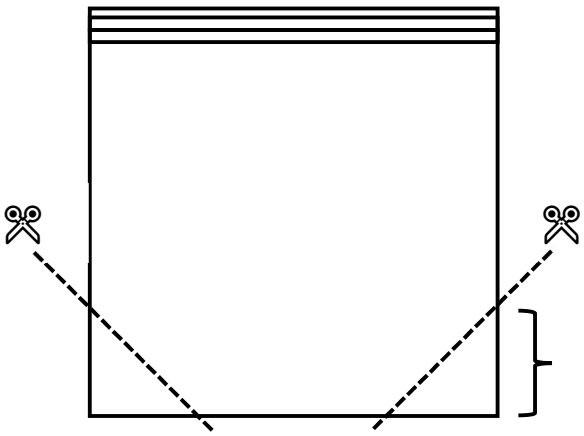
簡易な防水方法と、長期間使用する方法の 2 通りあります。表Ⅴ-1 の数量は、それぞれの防水方法を 1 つずつ施す場合に必要な数量です。

表Ⅴ-1 Wio Node の防水に必要な資材

資材および部品	数量	規格・メーカー等
ジッパーバッグ	1	ジップロック フリーザーバッグM（17.7 cm × 18.9 cm）など
ビニールテープ	1	ニトムズ nitoms J2577 [ビニルテープS 19×10 黒] など
自己融着テープ	1	日東シンコー、自己融着性ブチルゴムテープNo.15
防水バッグ	1	Aquapac Wire Through Case S (TC Clip付き) 型番548
■その他		
ハサミ、ケーブルタイ、ビニールタイ、ダブルクリップ、マイカ線など		

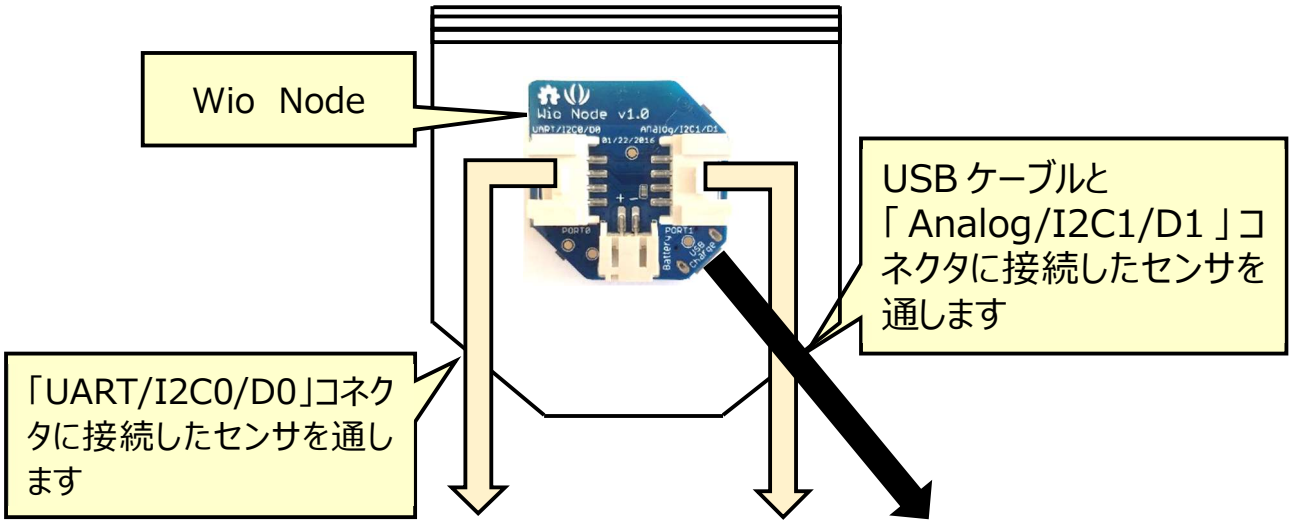
(1) 半年程度の短期間使用する場合（簡易な方法）

ジッパーパッグを用意して、袋の隅をカットします（図Ⅴ-1）。



図Ⅴ-1 ジッパーバッグの準備①

その後、袋の中に電源用の USB ケーブルとセンサをつなげた Wio Node を入れ、カットして穴をあけた部分から USB ケーブルとセンサを通します（図Ⅴ-2）。



図Ⅴ-2　ジッパーバッグの準備②

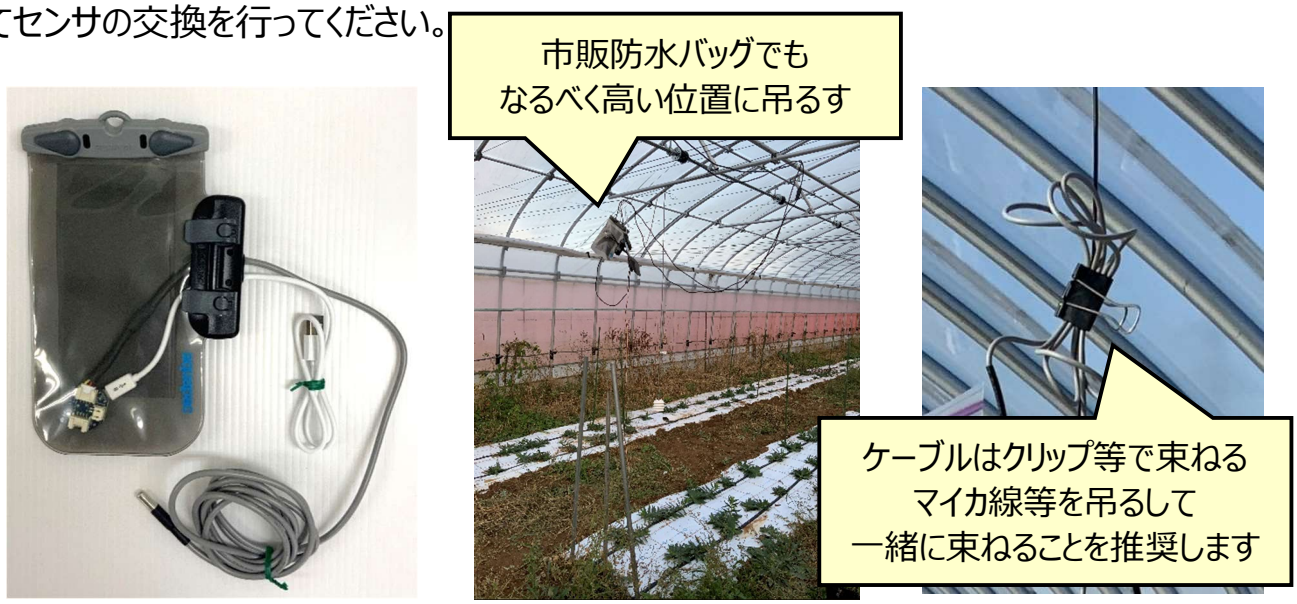
センサを通した後は、センサ部と袋の隅を「ビニールテープ」もしくは「自己融着テープ」で巻けば完成です。マイコンのコネクタ側を下にし、センサを１つだけつけた場合は図Ⅴ-3 1 の写真になります。

図Ⅴ-3　ジッパーバッグでの防水と設置例（失敗例）

袋ごと地面に置くと中に水が入ることがあるので、ハウス内に設置する際はマイカ線などを利用して、できるだけ吊るすようにします。なお、ミスト灌水などを行う施設ではマイコンが入ったジッパーバッグをミストがかからない高い位置に設置します。

(2) 長期間使用する場合

ハウスに１年中設置する場合も、ジッパーバッグによる簡易な防水袋を新たに作り変えればよいですが、作り変えるのが面倒であったり、高価でも長期間使用したい場合は、市販の防水バッグを利用します。図Ⅴ-4の左図のように、バッグ右側からセンサとUSBケーブルを出します。また、長期間利用される際には、「参考：温度センサの使用上の注意」を確認し、状況に応じてセンサの交換を行ってください。

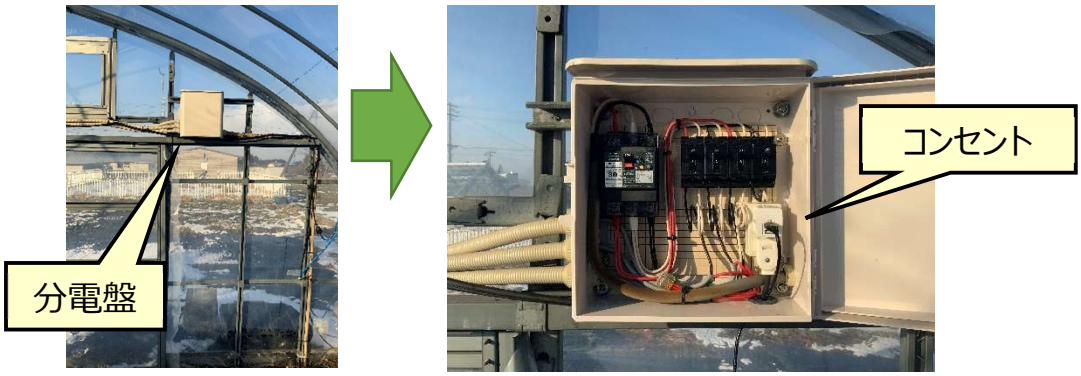


図Ⅴ-4　防水バッグでの防水と設置例（推奨例）

2. 100V 電源からの設置方法

(1) ハウス内のコンセント近くで利用する場合

分電盤内のコンセントに USB-AC アダプタを差し込み使用できます（図 V-5 右）。



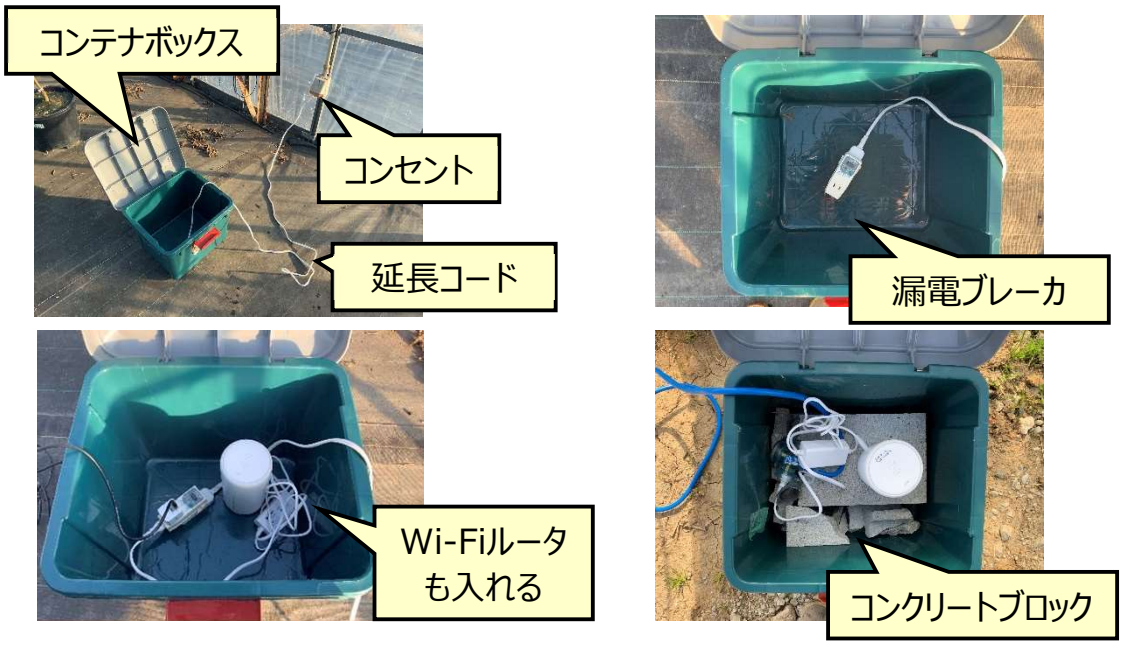
図Ⅴ-5　分電盤を利用する例

(2) ハウス内のコンセントから遠い場所で利用する場合

100V 電源延長ケーブルで延長します。電源ケーブルを地面に置いてしまうと、灌水時に漏電する危険性があるため、電源ケーブルはコンテナボックス内に設置します。コンテナ内にはコンクリートブロックなど適当な重りを入れて動かないようにします。表Ⅴ-2 の資材は 1 カ所のコンセントから 1 つの場所へ延長する場合の数量です。万が一の漏電を防ぐため、コンセントには漏電ブレーカを設置します。使用しない箇所はビニールテープで塞ぐか、防水延長コードや、普通のプラグを防水プラグ化するプラグカッパーなどを利用してください。

表Ⅴ-2 コンセントから電源を延長する場合の資材

資材および部品	数量	規格・メーカー等
コンテナボックス	1	アイリスオーヤマ 収納BOX　RVBOX400　など
100V電源延長ケーブル	1	屋内用・屋外用100V電源延長ケーブル、または電工ドラム
漏電ブレーカ	1	テンパールビリビリガード　プラグ型漏電遮断器
コンクリートブロック	1	コンテナボックスに入るサイズ
Wi-Fiルータ	1	Speed Wi-Fi HOME　Home L02など



図Ⅴ-6　コンテナボックスの設置方法

(3) 電源から設置場所が遠い場合の設置方法（USB ケーブルの延長）

Wio Node に接続する USB ケーブル（microB）と USB 延長ケーブルを接続してビニールテープを巻いて絶縁して利用します（図 V-7）。USB ケーブルは長くなればなるほど電圧が降下し、Wio Node が動作しません。（電圧は、電線の太さが細ければ細いほど、電線が長ければ長いほど低下します。そのため、細い電線を使った USB ケーブルでは延長できません。延長する際は Wio Node に接続する USB ケーブル（microB）には、急速充電対応と書いてあるものを使用することを推奨します。）表Ｖ-3 は 1 つの Wio Node を延長する際に必要な資材リストです。数量でのカッコ内の数字は、USB 延長ケーブルを自作する際の資材です。

図Ｖ-7　USB ケーブルの絶縁方法

表Ｖ-3 USB ケーブルを延長するための資材

資材および部品	数量	規格・メーカー等
USB延長ケーブル（5 m）	1	エレコム U2C-JE50BK [エコUSB2.0延長ケーブル 5 m]など
ビニールテープ	1	ニトムズ nitoms J2577 [ビニルテープS 19×10 黒] など
シールド線	(1)	富士電線工業 MVVS 0.5sq×2芯 ケーブルなど
USB延長ケーブル	(1)	USB充電通信延長ケーブル1 m（AT-CASTUSEX02）など
■その他		
ハサミ、ニッパー、熱収縮チューブ、圧着スリーブ、圧着工具、ヒートガンなど		

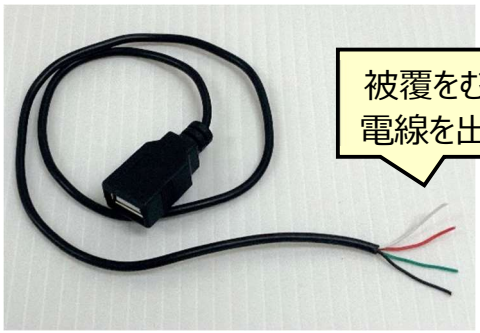
参考：USB 延長ケーブルを自作する

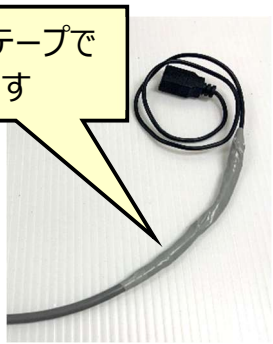
シールド線（1 巻 100　m）を用いて USB 延長ケーブルを自作することができます。

USB 延長ケーブル

100 m 巻を
4 等分して、
25 m にしたも

シールド線





太い電線を利用して、電圧降下を抑えた電源用の USB 延長ケーブル（最長 25 m ほど）を作ることができます。USB ケーブル内には 4 つの電線がありますが、赤い線（5V）と黒い線（GND）だけ使用します。USB ケーブルをカットして、シールド線を用いてつなぎ延長します。その際はハンダ付けか圧着スリーブを用いて、ショートしないように熱収縮チューブによる絶縁処理だけでなく、ビニールテープを用いて再度絶縁処理をするようにします。短い USB 延長ケーブルではなく、急速充電対応の USB ケーブルを用いると電線が太いため作りやすいです。うまく電線の被覆をはがせない、あるいは圧着スリーブを圧着しても電線が抜けてしまう場合は太い電線の USB ケーブルに変えることを検討します。

3.温度センサの設置方法

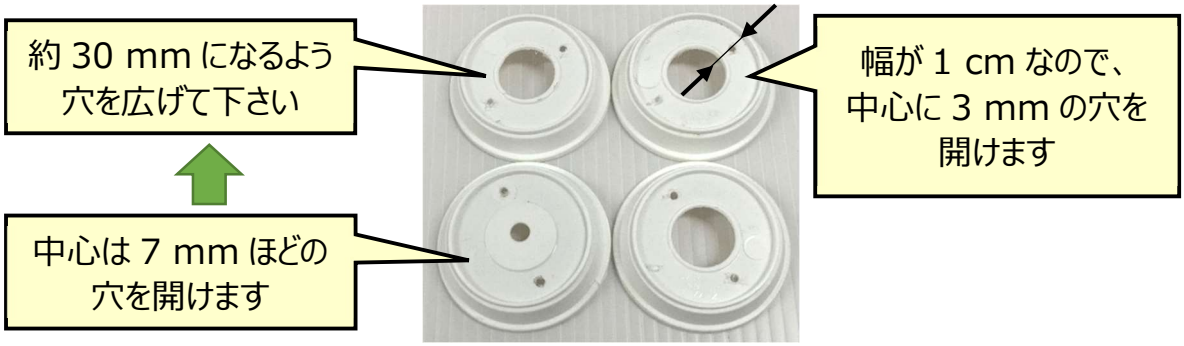
(1) 日よけを利用して温度を測定する方法

ハウス内の温度を精度よく測定するためには、センサに日射を遮るための日よけが必要になります。「ラジエーションシールド」で検索すると市販品が出てきますが、自分でも作成できます。作成方法はインターネット上でも手に入りますが、ここでは Wio Node と接続する温度センサ、温湿度センサを設置するための簡易なものを製作する方法を紹介します。ただし、閉め切ったハウス内では通風が無いことから熱がこもるため、温度を精度よく測定したい場合は後述する強制通風筒を利用します。表Ⅴ-4 は日よけを 1 つ作成するための必要個数です。

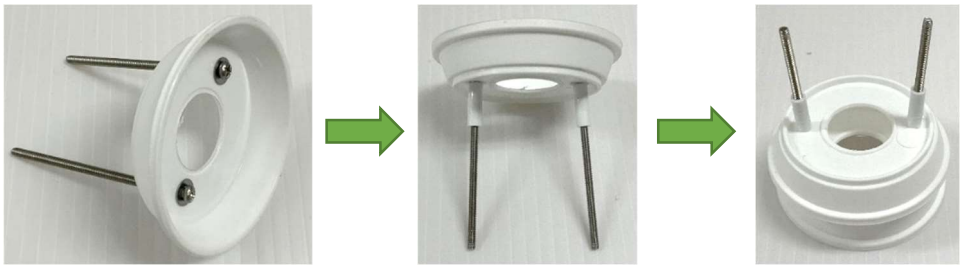
表Ⅴ-4 日よけ作成資材リスト

資材および部品	数量	規格・メーカー等
鉢皿	4	大和プラスチック 鉢皿サルーン 1号 φ65 mm×H15 mm ホワイト
M3ステンレスねじ 60 mm	1	十字穴付（+）ナベ小ねじ CSPPN-SUS-M3-60 など
M3ステンレスナット	4	六角ナットB250003 M3
M3ステンレスワッシャー	2	丸ワッシャー WSJ-SUS-M3
M3ステンレススプリングワッシャー	4	ばね座金 スプリングワッシャー 2号
アイロンビーズ	26	ウィルトンブランズ パーラービーズ 5001 単色 しろ
■その他		
ハサミ、ニッパー、ドライバー、電動ドリル、ニッパー、ホールソー、カッターナイフ、ビニールテープ、養生テープなど		

鉢皿を加工します。鉢皿全てに、温度センサを通すため穴を開けます。中心に 7 mm ほどの穴を開けてください（図Ⅴ-8）。1 枚はそのままにし、3 枚はホールソーなどで穴を広げて約 30 mm の穴にします。ホールソーが無い場合はニッパーで大まかに穴を広げ、調整はカッターナイフ等で行います。ステンレスねじを通す穴を開けるには 3 mm のドリルで穴を開けます。



図Ⅴ-8 日よけの作成方法①

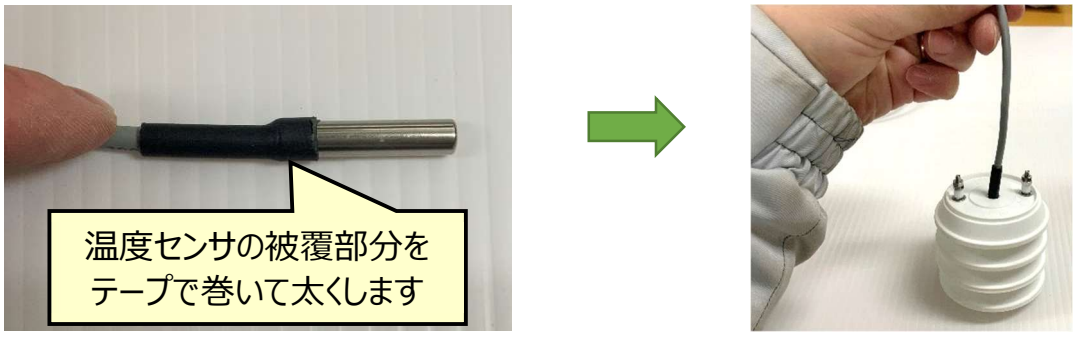


図Ｖ-9　日よけの作成方法②

穴（30 mm）を開けた鉢皿の下からネジを通します（図 V-9）。ネジにはスプリングワッシャー（無くてもかまいません）とワッシャーを入れてから鉢皿に通します。ネジにアイロンビーズ３つずつ通してスペーサーとします。その後鉢皿を通し、またアイロンビーズを３つずつ通してスペーサーとします。一番上の鉢皿には中心に 7 mm の穴を開けたものが来るようにします。

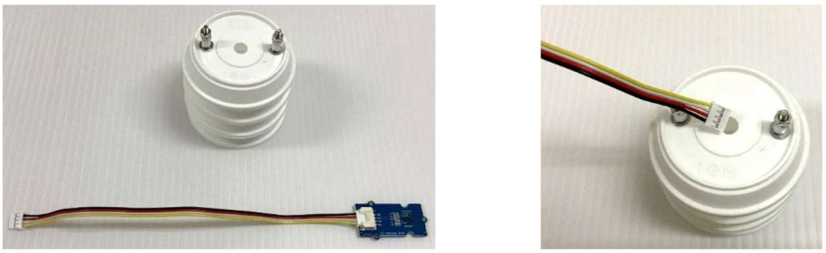
図Ｖ-10　日よけの作成方法③

最上段の鉢皿をつけたのち、ワッシャー、スプリングワッシャー（無ければ省略可）、アイロンビーズ、ナットの順で取り付け、ナットを回してしめつけていきます（図 V-10）。スプリングワッシャーがある場合はスプリングワッシャーがつぶれるまでしめます。無い場合は右図のようにナット２から３個分取り付けられる程度にしめます。ナットが外れないよう右図のようにダブルナットにします。



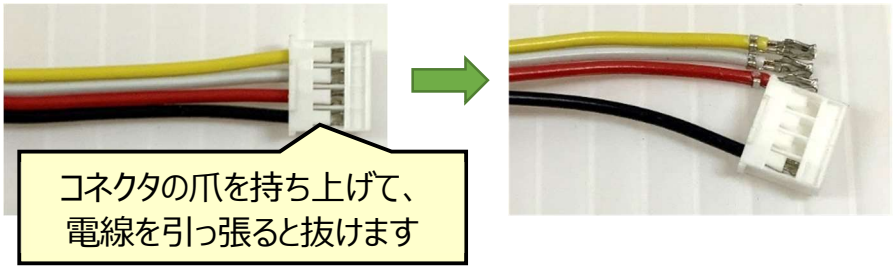
図Ｖ-11　日よけへのセンサ設置方法①

温度センサを穴に通した後、温度センサが抜けないようにします。温度センサの被覆部分を養生テープやビニールテープで 2 から 3 周分巻きます（図 V-11）。右図のように持ち上げても外れなければ完成です。



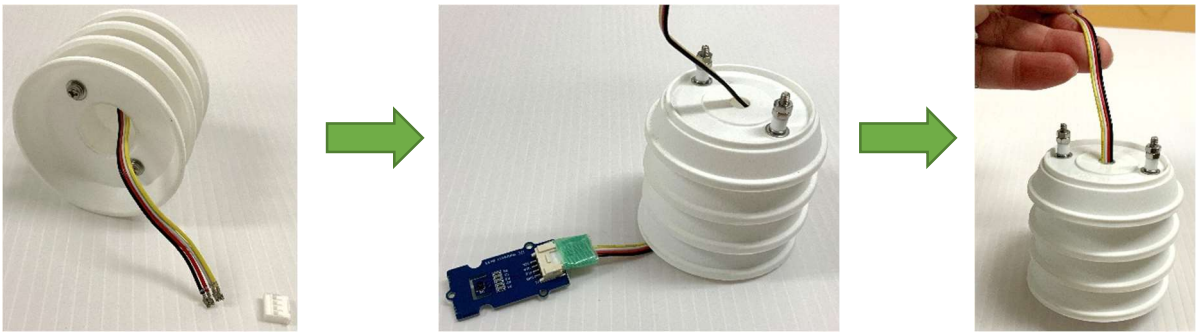
図Ｖ-12　日よけへのセンサ設置方法②

温湿度センサを使用する場合について説明します。温湿度センサはこのままの状態では、ケーブル部分を通すことができないため、コネクタを一度外す必要があります（図 V-12）。



図Ｖ-13　日よけへのセンサ設置方法③

コネクタを外すには、爪を持ち上げる必要があります（図 V-13）。カッターナイフでも行うことができますが、コネクタを破壊してしまう恐れがあるため、精密マイナスドライバなどで行ってください。



図Ｖ-14　日よけへのセンサ設置方法④

コネクタを外して、日よけに通した後はコネクタを再度取り付けて、温湿度センサを取り付けます。その際、コネクタ部を養生テープ等で巻いておき、日よけから落ちないようにします（図 V-14）。右のように持ち上げても外れなければ完成です。

接続用のケーブルが短いため、長いケーブルを購入して使用することを考えるかもしれませんが、ここで紹介した温度センサ、温湿度センサは「I2C」と呼ばれる通信方式のセンサであるため、ケーブルを長くするだけでは通信ができなくなってしまいます。少々短いですが、付属してくる長さのケーブルを使用してください。

(2) 強制通風筒を利用して温度（温湿度）を測定する方法

強制通風筒と呼ばれるタイプの日よけを利用すると、より精度よく測定することができます。強制通風筒も多くの自作方法がありますが、材料費約２万円で作成できる「NIAES-09S」は、センサの設置方法が簡単で、作成や設置方法等の詳しい資料*が公開されています。

*農業気象学会　の HP でダウンロード可能です。

連載講座「栽培環境における気温の観測技法と利用」

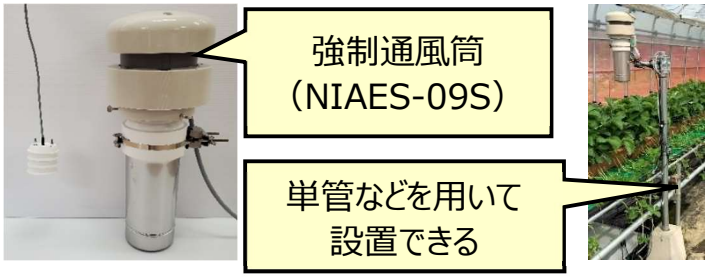
（4）NIAES-09S 改型強制通風筒の製作法（福岡峰彦ら）

http://agrmet.jp/wp-content/uploads/2019-A-2.pdf

連載講座「栽培環境における気温の観測技法と利用」

（5）観測用マストの建て方（福岡峰彦）

http://agrmet.jp/wp-content/uploads/2019-A-3.pdf



図Ｖ-１５　強制通風筒でのセンサ設置の様子

(3) 温度センサ使用時の注意点

温度センサを利用する際には、これまで使用していた温度センサとの差を確認してから使用してください。指標としていた温度がある場合に、これまでの温度センサと本システムの温度センサで大きな差分がある場合は、これまでの温度センサの値に問題があります。本システムを利用する際に温度指標にその差分を加えたうえで利用してください。あるいは、これまで温度校正を行った温度センサを用いていた場合は、使用する温度範囲でそのセンサとの差を比較することで校正し、ご利用ください。

温度センサの精度は重要です。防水温度センサが複数ある場合は、バケツに水を入れてその中に全ての温度センサを同時に入れて、それぞれの温度センサの値の差を見ると確認が簡単です。

なお、防水温度センサ（one wire temperature sensor、測定レンジ-55 ℃～+125 ℃）や温度センサ（Grove 温湿度センサ SHT31、測定レンジ-40 ℃～+125 ℃）では、一般的なハウス温度の範囲において±１ ℃の精度で測定できることを確認しておりますが、基板がむき出しのため、水をかけてしまったなどの原因で壊れたり、明らかにおかしい値が出るようになった場合は、新たにセンサを購入して交換してください。

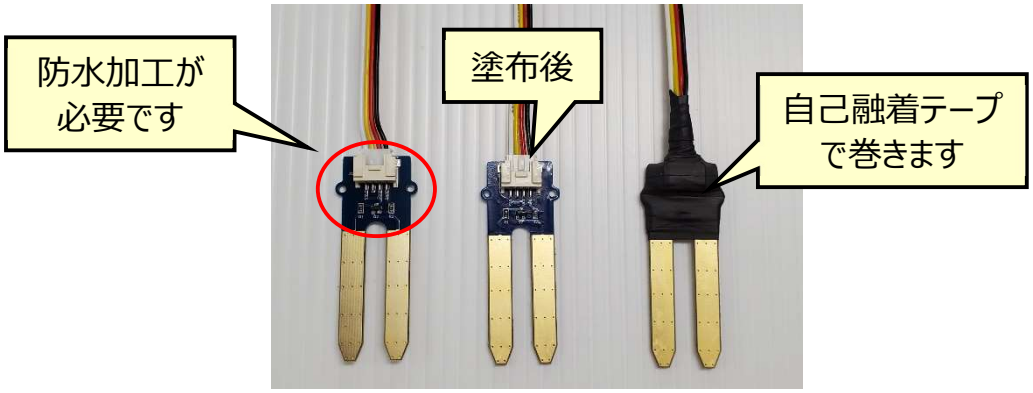
ハウスで使用後、事務所などに回収し、しばらく使用しないで保管していたものを次年度に再度利用する際は、新しい温度センサと比較して問題なく使用できるかを確認して下さい。

4. 土壌水分センサの設置方法

Wio Node には 2 つコネクタがありますが、土壌水分センサに利用できるものは Analog と書かれたコネクタ側のみとなりますので注意してください。アナログでの通信のため、ケーブルは延長可能です。右図の写真のような延長ケーブルを使用できます。表Ⅴ-5 は土壌水分センサを１組利用する際の必要個数です。

表Ⅴ-5 土壌水分センサ資材リスト

資材および部品	数量	規格・メーカー等
土壌水分センサ	1	Seeed社 Grove 水分センサ
Grove互換ケーブル	1	M5Stack用GROVE互換ケーブル 200 cm（1個入り）など
■その他		
ハサミ、RTVシリコーンスプレー、自己融着テープなど		



（１） （２） （３）

（１） 未処理
（２） ゴムスプレーを塗布
（３） 自己融着テープで防水

図Ⅴ-15 土壌水分センサへの防水加工方法

土壌水分センサは基板部分がむき出しのため、防水加工を施す必要があります。最も良い方法は、基板部分以外を養生テープでマスキングして、RTV シリコーンスプレーなどの基盤塗布用のゴムスプレーを塗布してから、自己融着テープを巻きます。RTV シリコーンスプレーが手に入らない場合は自己融着テープだけでもかまいません。

用語解説

通い農業：ここでは東京電力福島第一原子力発電所事故による避難後、避難先や新たな居住地から営農が再開された地域に通いながら農業を行うこと。

アクセストークン：ユーザーを判別する固有の値のこと。Web 上のサービスを利用する際に、ユーザー名とパスワードを入れることで、登録されたユーザーであることが認識され、サービスを受けることができます。本システムで利用している Wio Node や LINE では、ユーザー名とパスワードのような固有の値（文字列）を取得することで、メーカーのサーバにアクセスし、データの取得やメッセージの通知が行えるようになります。

SSID：無線 LAN（W-Fi）におけるアクセスポイントの名称のこと

Web API：HTTP からはじまる URL を用いてサーバ側のアプリケーションを実行する仕組み

WEB サーバ：WEB ブラウザからアクセスすることでデータを表示してくれるサーバコンピュータのこと

導入生産者の評価

■ 特徴

- 従来の温度確認を目的としたハウスの見回りを本システムによる自動遠隔監視に変えることで、灌水やハウスの側窓開閉などの管理作業要否の判断を遠隔で行うことができます。
- 市販の完成された遠隔監視システムは、データを蓄積できるものの測定点数やデータの保存間隔などに制限があります。これに対して通い農業支援システムは低価格であり、コストの問題で導入できなかった生産者にも利用しやすくなっています。また測定点数やデータの保存間隔などを選択して利用可能なため、データに基づいた農業を効率的に実施することができるようになります。
- 通い農業支援システムは設置場所を容易に変更できます。このため、将来市販のハウス遠隔監視システムを新たに導入した場合にも、通い農業支援システムを他のハウスや作業場等に移設して継続利用することが可能です。

■ 良い点

- ハウスにいなくても遠くから温度条件等の確認できますので、現場の実務者に対して曖昧な指示ではなくデータに基づく明確な指示ができます。
- 自宅にいながら遠隔地にあるハウスの状況を確認できますので、トラブル等によるハウス暖房停止に対し、適切に対応できます。
- どこにいてもハウス管理の判断ができますので、大規模な耕地を管理しながらハウスでの育苗作業の計画が立てやすくなります。実際に、本技術の導入によるハウス訪問の回

数が減ることが確認されていますので、耕地とハウスでの作業の効率的な割り当てが可能となり、農作業が忙しい時ほど本技術は役立つはずです。

- 従来の水稲育苗（ハウス、プール、芽出し機など）の温度測定は手動でしたが、本技術では自動測定が可能です。その結果、全員でデータをリアルタイムで共有でき、ハウスの急な温度上昇に適切に対応できるようになります。さらに、自宅に居ながらの温度監視が可能ですので、芽出し期などの夜間の見回りも不要になります。
- データに基づいた栽培管理指標を作るために役立てることができます。
- 切り花の調製作業を行うハウス内の作業スペースの温度が低すぎますと、花の色が変わり、品質低下につながります。本技術の利用により、暖房設定を適切に管理できるようになり、品質の安定化につながっています。これにより、さらに燃料費を削減できます。

■ 留意点

- 情報を共有して管理を共に行う必要のない熟練生産者は、ハウス遠隔監視システムを必要としないことが多いです。
- 対応にあたる県の普及組織や試験研究機関担当者は、本手順書を活用して実際にシステムを構築できる程度の知識が必要となります。
- 生産者によるシステム等の作成が技術的に困難な場合は、農研機構や普及所等の指導的立場の機関による少人数でのワークショップ等の指導や補助を生産者が受けることが望ましいです。
- 本技術の普及に関わる方は、システムを適用する管理作業の具体的な中身などを聞き出す作業を通して、生産者の利用イメージを明確にする必要があります。
- 温度、湿度、土壌水分などのセンサを利用することができますが、生産者間でデータを

共有するために使用する場合、測定条件・環境の差を小さくし、精度よく測定するため、温湿度であれば強制通風筒（参照 p. 78）を利用するなど、精度よく測定するための工夫が必要となります。

参考資料

1．成果情報：安価かつ簡便にハウスの遠隔監視に使える IoT 機器「通い農業支援システム（農研機構　普及成果情報　放射能対策技術　2020 年）
https://www.naro.go.jp/project/results/4th_laboratory/tarc/2020/20_067.html

2．安価かつ簡便にハウスの遠隔監視に使える IoT 機器「通い農業支援システム」製作マニュアル（農研機構東北農業研究センター、2021 年 3 月発行）
https://www.naro.go.jp/publicity_report/publication/files/KayoiManual_0706.pdf

3．山下善道ら（2021）営農再開後の通い農業を支援するハウス遠隔監視システムの開発とその展開．農研機構研究報告．（8）：211-230.
https://www.jstage.jst.go.jp/article/naroj/2021/8/2021_211/_pdf/-char/ja

4．簡便・安価な IoT 技術「通い農業支援システム」（農林水産省スマート農業推進フォーラム、2020 年）
https://www.maff.go.jp/j/kanbo/smart/forum/R2smaforum/horticulture/seika31.html

5．ハウスの情報をスマホで確認「通い農業支援システム」（農研機構 NAROchannel）
https://www.youtube.com/watch?v=zJd6nJEQi0k

6．「みどりの食料システム戦略」技術カタログ（Ver.3.0）～現在普及可能な新技術～（p. 77）．（農林水産省 2023 年 5 月）
https://www.maff.go.jp/j/kanbo/kankyo/seisaku/midori/attach/pdf/midori_catalog_all.pdf

7．プレスリリース：(研究成果) 安価かつ簡便にハウスの情報をスマートフォンで確認 -「通い農業支援システム」製作マニュアルを公開 -（農研機構 2021 年）
https://www.naro.go.jp/publicity_report/press/laboratory/tarc/141378.html

その他情報

本書の内容は、農林水産省委託事業「原発事故からの復興のための放射性物質対策に関する実証研究委託事業（2018～2020 年度）」によって実施された研究成果を取りまとめたものです。

担当窓口、連絡先

外部からの受付窓口：

農研機構　東北農業研究センター　研究推進部事業化推進室

電話：019-643-3407

E メール：jigyoka@ml.affrc.go.jp

「農研機構」は、国立研究開発法人 農業・食品産業技術総合研究機構のコミュニケーションネーム（通称）です。