My Book

cover sample for A5,
with bleed margin 3mm

published by Re:VIEW

はじめての IoT 講座

THEToilet　著

2021-07-22 版　　発行

はじめに

これは電子計算機研究会の IoT 講座用に作った技術同人誌です。

サークルに参加するメリットの一つに、興味があることについて学べる機会がある。これがあげられるとおもいます。

自分も一年の時にサークルの先輩から、いろいろな勉強会を開催していただき。自分の知見をひろげることができました。

(@THEToilet)

電子計算機研究会とは

芝浦工業大学公認のサークルであり、制作活動や日々の勉強を行っています

お問い合わせ先

本書に関するお問い合わせ：toilet.wc@gmail.com

目次

第 1 章

電子部品の準備

本章では本誌のサンプルを進めるにあたって必要な電子部品または、その購入方法について紹介します。

1.1 電子部品の購入の方法

電子部品の販売店が近くにあれば、直接商品を見ながら購入するのが一番ですが、お店が近くになかったり、コロナ渦の問題などで直接行くことが難しい場合は、通販での購入をおすすめします。下記の 5 つのサイトは電子部品を購入するにあたってよく使われている店おです。特に秋月電子通商、千石電商そして aitndo は秋葉原に店舗があるので、機会があれば行くことをおすすめします。

- 秋月電子通商
  - `https://akizukidenshi.com/catalog/`
- 千石電商
  - `https://www.sengoku.co.jp/`
- SWITCHSCIENCE
  - `https://www.switch-science.com/`
- Amazon.co.jp
  - `https://www.amazon.co.jp/`
- aitendo
  - `https://www.aitendo.com/`

1.2 本誌で利用する電子部品

筆者が本誌に使用するサンプルを作成するにあたって購入した商品を紹介します。本誌のサンプルを勧めるにあたって必要になるので、参考にして用意してください。

表 1.1: 必要な材料

品名	個数	参考価格	詳細情報
ESP32DevKitC	1 個	1230 円	
microUSB Type-B	1 本	約 300 円	
ブレッドボード	2 個	280 円× 2	
LED	1 袋	150 円	
ジャンプワイヤセット（オス・オス）	1 セット	220 円	
抵抗 100 Ω & 10k Ω	100 Ω: 1 袋、10k Ω: 1 袋	100 円× 2	
タクトスイッチ	1 個	10 円	
温湿度センサ	1 個	300 円	
ディスプレイ	1 個	580 円	
計		約 3550 円	

おすすめ製品

今回はすべて秋月の通販にて電子部品を購入をしましたが、同じ製品であればどこで購入するかは問いません。しかし、本誌は以下の製品で動作確認をしているため基本的には以下の製品を購入することをおすすめします。

ESP32DevKitC

ＥＳＰ３２－ＤｅｖＫｉｔＣ－３２Ｅ　ＥＳＰ３２－WROOM－３２E開発ボード４ＭＢ

https://akizukidenshi.com/catalog/g/gM-15673/

ブレッドボード

ブレッドボード　６穴版　ＥＩＣ－３９０１

https://akizukidenshi.com/catalog/g/gP-12366/

備考: ESP32DevKitC は幅が広いため、6 穴のブレッドボードを使うことをおすすめします。

LED

５mm赤色ＬＥＤ　６２５ｎｍ　７ｃｄ６０度　（１０個入）

https://akizukidenshi.com/catalog/g/gI-01318/

ジャンプワイヤセット（オス・オス）

ブレッドボード・ジャンパーワイヤ（オス－オス）セット　各種　合計６０本以上

https://akizukidenshi.com/catalog/g/gC-05159/

抵抗

カーボン抵抗（炭素皮膜抵抗）　１／４Ｗ１０ｋΩ　（１００本入）

https://akizukidenshi.com/catalog/g/gR-25103/

カーボン抵抗（炭素皮膜抵抗）　１／４Ｗ１００Ω　（１００本入）

https://akizukidenshi.com/catalog/g/gR-25101/

備考: 上記の抵抗は 100 本単位からしか購入できません。実際に使用するのはどちらの抵抗値とも 3 本以下なので必ずしも 100 本買う必要はありません。

タクトスイッチ

タクトスイッチ（緑色）

https://akizukidenshi.com/catalog/g/gP-03651/

備考: 色の選択は自由です。

温湿度センサ

温湿度センサ　モジュール　ＤＨＴ１１

https://akizukidenshi.com/catalog/g/gM-07003/

ディスプレイ

０．９６インチ　１２８×６４ドット有機ＥＬディスプレイ（ＯＬＥＤ）　白色

https://akizukidenshi.com/catalog/g/gP-12031/

第 2 章

環境構築

この章では ESP32 を利用するために必要な環境構築手順を説明します。
Windows 環境を想定しているので、Mac 環境の方は少しやり方が違うかもしれません。

2.1　ESP32 とは

ESP32 ってなに？？？ Espressif Systems 社が開発した SoC(System on a Chip) シリーズの名前環境開発環境として* Arduino IDE * ESP-IDF * MicroPython などがありますが、今回は Arduino IDE を用いて開発を進めていきたいと思います。

2.2　Arduino IDE のインストール

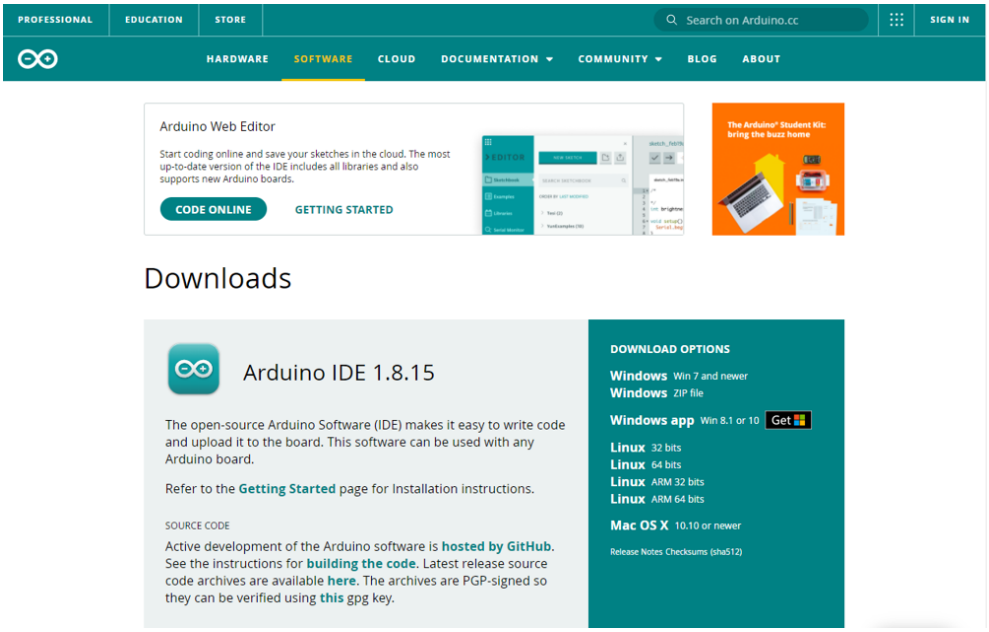

図 2.1: 1

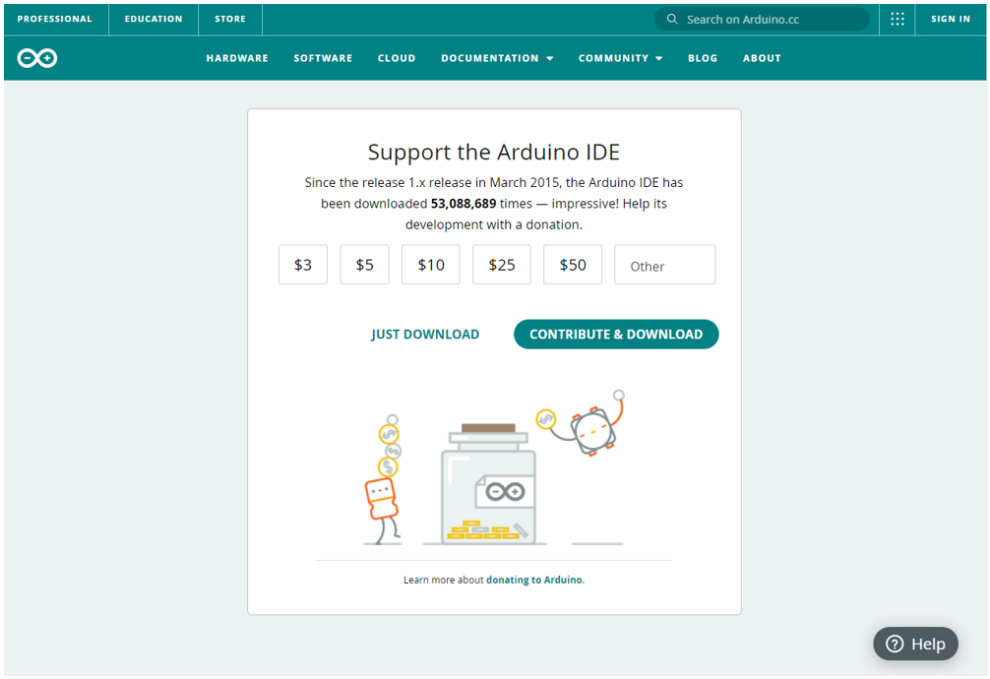
PROFESSIONAL
EDUCATION
STORE
Search on Arduino.cc
SIGN IN
HARDWARE
SOFTWARE
CLOUD
DOCUMENTATION
COMMUNITY
BLOG
ABOUT
Support the Arduino IDE
Since the release 1.x release in March 2015, the Arduino IDE has been downloaded 53,088,689 times — impressive! Help its development with a donation.
$3
$5
$10
$25
$50
Other
JUST DOWNLOAD
CONTRIBUTE & DOWNLOAD
Learn more about donating to Arduino.
Help

図 2.2: 2

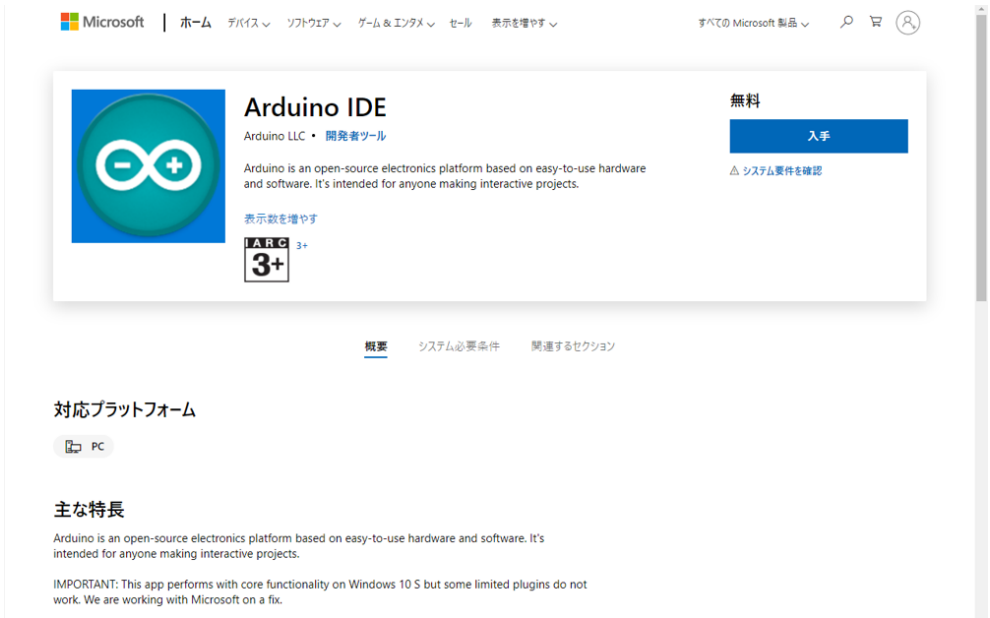
Microsoft
ホーム
デバイス
ソフトウェア
ゲーム & エンタメ
セール
表示を増やす
すべての Microsoft 製品
Arduino IDE
Arduino LLC • 開発者ツール
Arduino is an open-source electronics platform based on easy-to-use hardware and software. It's intended for anyone making interactive projects.
表示数を増やす
IARC 3+
無料
入手
システム要件を確認
概要
システム必要条件
関連するセクション
対応プラットフォーム
PC
主な特長
Arduino is an open-source electronics platform based on easy-to-use hardware and software. It's intended for anyone making interactive projects.
IMPORTANT: This app performs with core functionality on Windows 10 S but some limited plugins do not work. We are working with Microsoft on a fix.

図 2.3: 3

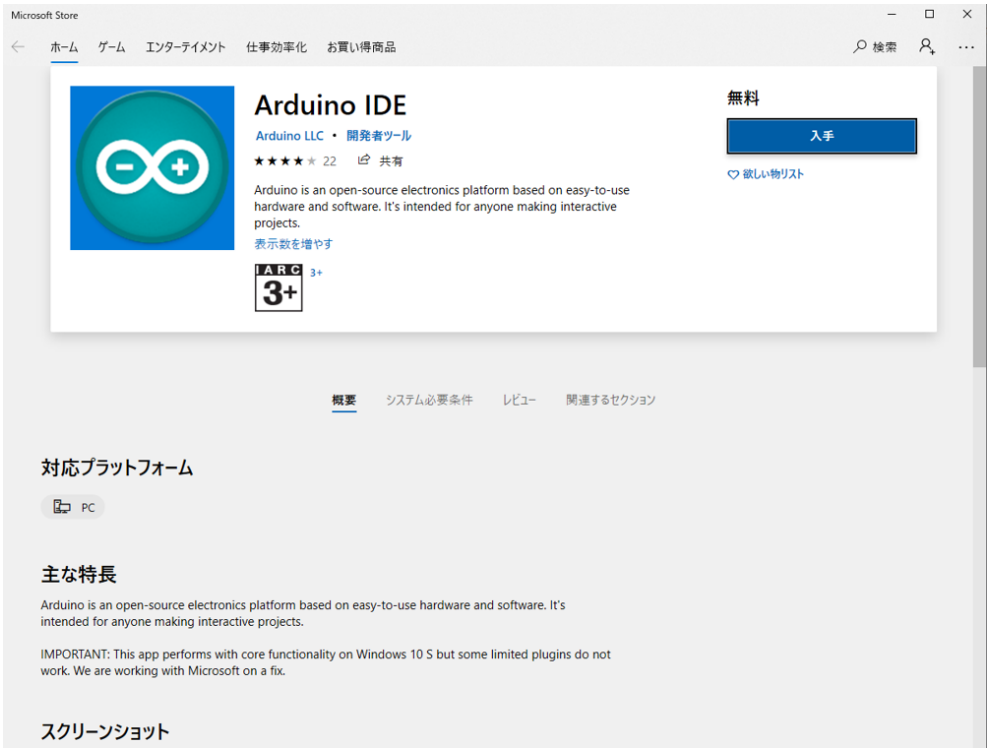
Microsoft Store
ホーム
ゲーム
エンターテイメント
仕事効率化
お買い得商品
検索
Arduino IDE
Arduino LLC • 開発者ツール
★★★★★ 22
共有
Arduino is an open-source electronics platform based on easy-to-use hardware and software. It's intended for anyone making interactive projects.
表示数を増やす
3+
無料
入手
欲しい物リスト
概要
システム必要条件
レビュー
関連するセクション
対応プラットフォーム
PC
主な特長
Arduino is an open-source electronics platform based on easy-to-use hardware and software. It's intended for anyone making interactive projects.
IMPORTANT: This app performs with core functionality on Windows 10 S but some limited plugins do not work. We are working with Microsoft on a fix.
スクリーンショット

図 2.4: 4

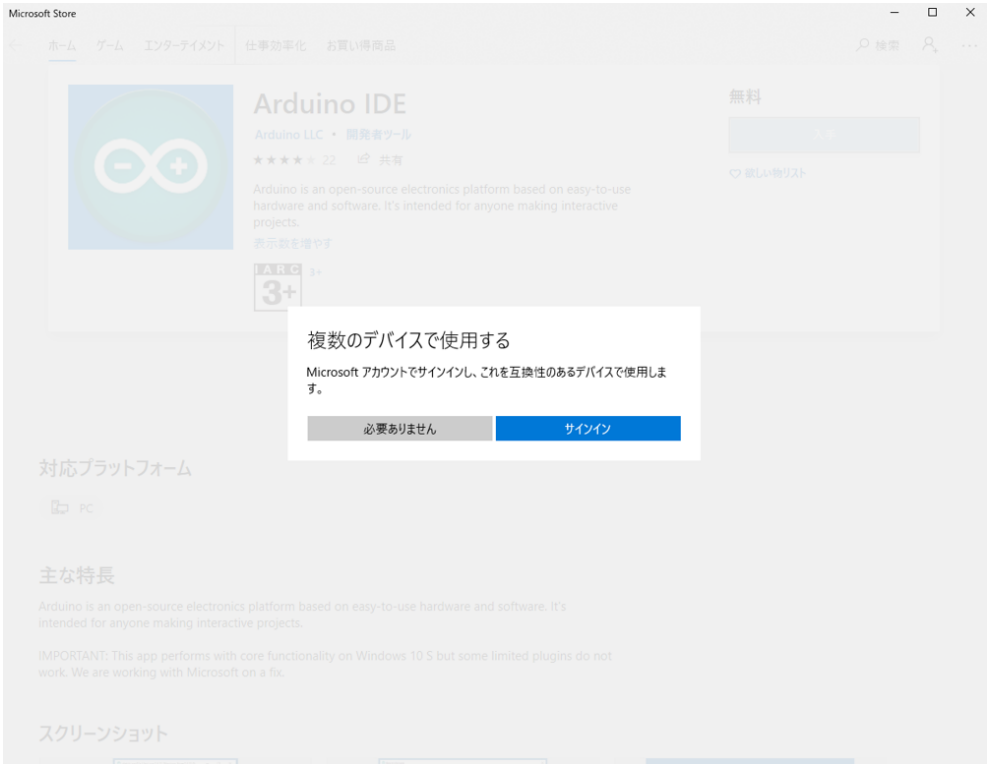
Microsoft Store
複数のデバイスで使用する
Microsoft アカウントでサインインし、これを互換性のあるデバイスで使用します。
必要ありません
サインイン

図 2.5: 5


Microsoft Store
ホーム
ゲーム
エンターテイメント
仕事効率化
お買い得商品
検索
マイ ライブラリ
すべて所有済み
インストール済み
インストールする準備ができました
ダウンロード
デバイスに含まれる
ダウンロードと更新
最新情報を取得する
最近のアクティビティ
Arduino IDE
アプリ
1.8.49.0
変更: 1 分以内
Xbox Game Bar
アプリ
5.721.6282.0
変更: 今日
スマホ同期
アプリ
1.21052.122.0
変更 2021/07/09

図 2.6: 6

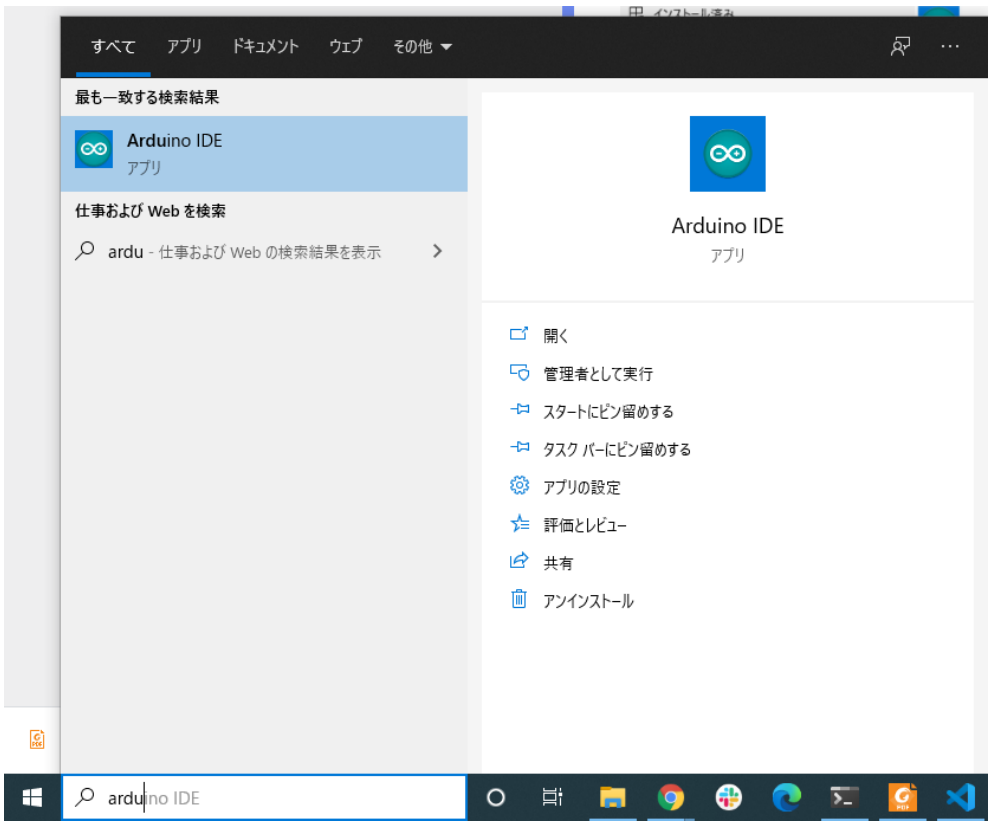
すべて
アプリ
ドキュメント
ウェブ
その他
最も一致する検索結果
Arduino IDE
アプリ
仕事および Web を検索
ardu - 仕事および Web の検索結果を表示
Arduino IDE
アプリ
開く
管理者として実行
スタートにピン留めする
タスク バーにピン留めする
アプリの設定
評価とレビュー
共有
アンインストール
arduino IDE

図 2.7: 7

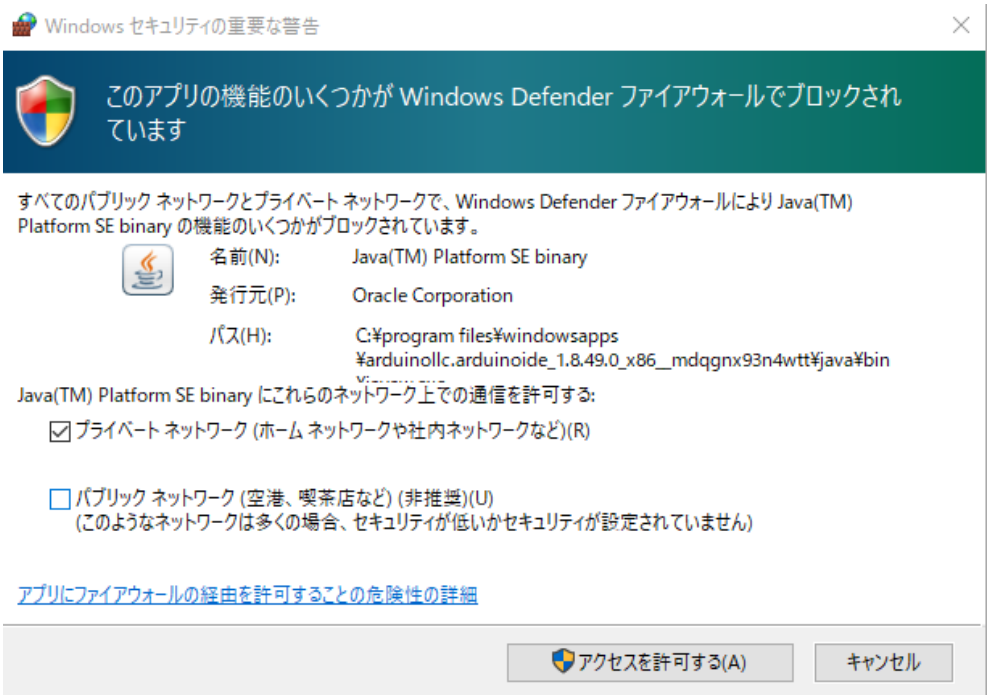
Windows セキュリティの重要な警告
このアプリの機能のいくつかが Windows Defender ファイアウォールでブロックされています
すべてのパブリック ネットワークとプライベート ネットワークで、Windows Defender ファイアウォールにより Java(TM) Platform SE binary の機能のいくつかがブロックされています。
名前(N): Java(TM) Platform SE binary
発行元(P): Oracle Corporation
パス(H): C:¥program files¥windowsapps ¥arduinollc.arduinoide_1.8.49.0_x86__mdqgnx93n4wtt¥java¥bin
Java(TM) Platform SE binary にこれらのネットワーク上での通信を許可する:
プライベート ネットワーク (ホーム ネットワークや社内ネットワークなど)(R)
パブリック ネットワーク (空港、喫茶店など) (非推奨)(U)
(このようなネットワークは多くの場合、セキュリティが低いかセキュリティが設定されていません)
アプリにファイアウォールの経由を許可することの危険性の詳細
アクセスを許可する(A)
キャンセル

図 2.8: 8

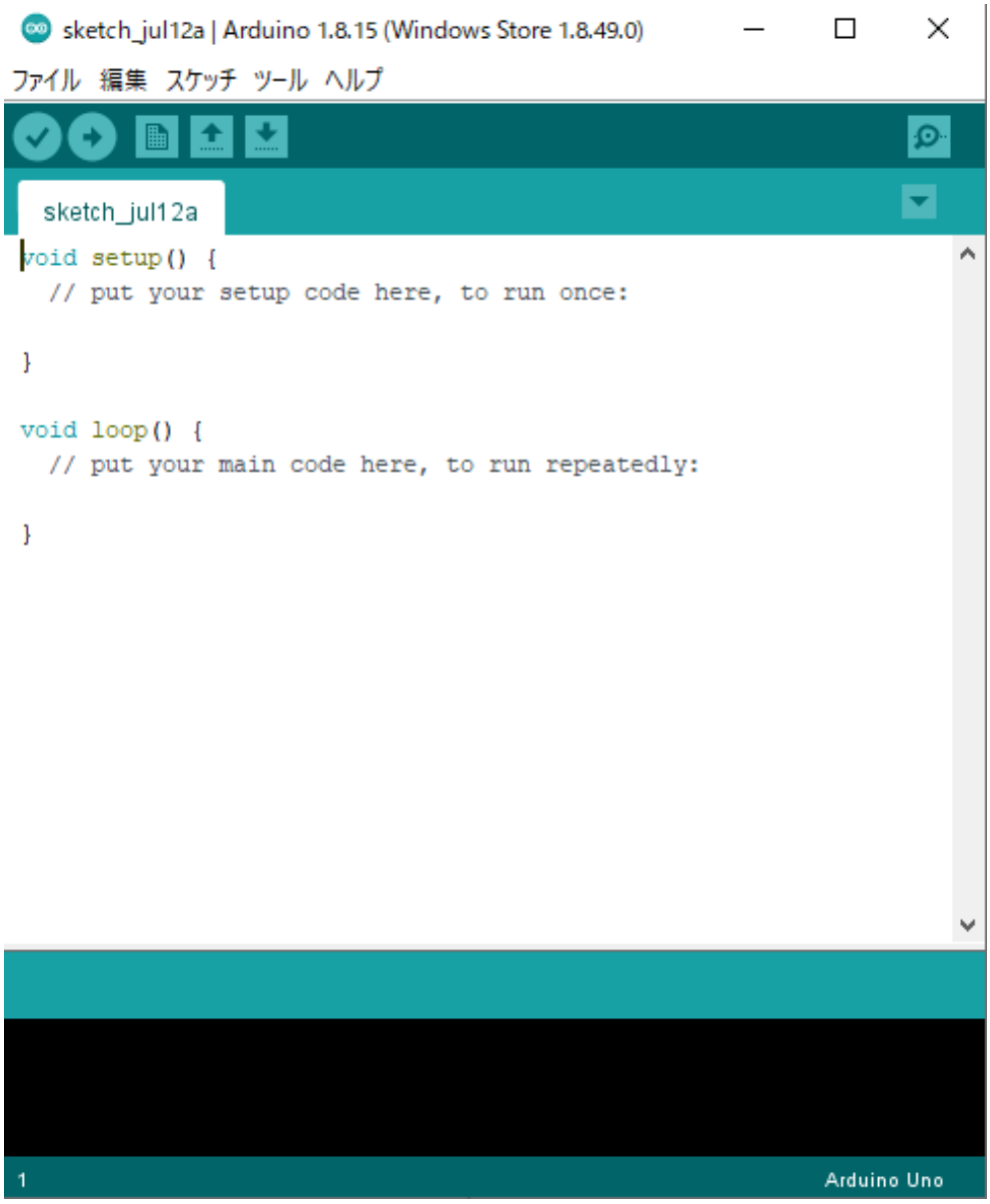
sketch_jul12a | Arduino 1.8.15 (Windows Store 1.8.49.0)
ファイル 編集 スケッチ ツール ヘルプ
sketch_jul12a
void setup() {
// put your setup code here, to run once:
}
void loop() {
// put your main code here, to run repeatedly:
}
1
Arduino Uno

図 2.9: 9

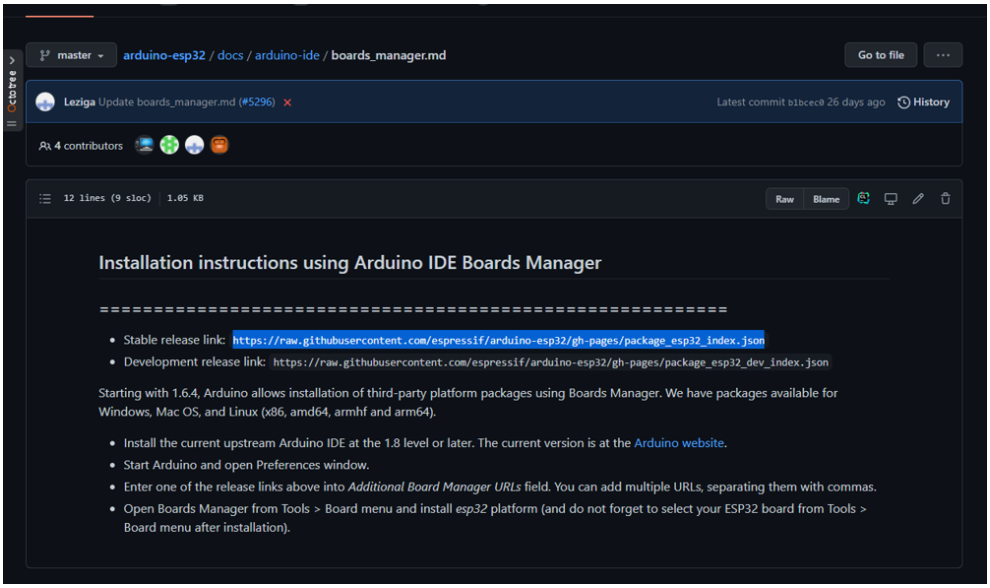

図 2.10: 10

図 2.11: 11

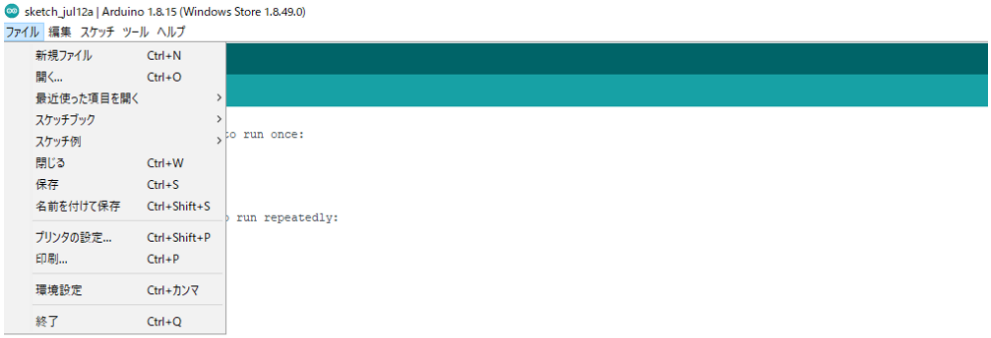

図 2.12: 12

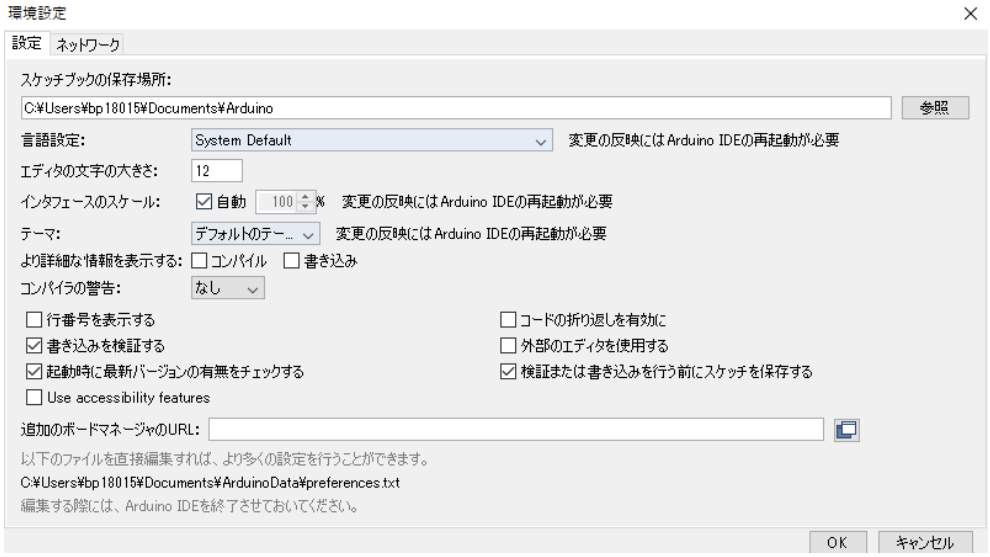

図 2.13: 13

2.3　ESP32 用ボードマネージャーのインストール

2.4　動作確認

ここで動作確認をするために定番の HelloWorld を行いましょう

ブレッドボード

まず ESP32 をブレッドボードにさしましょう

図 2.14: 14

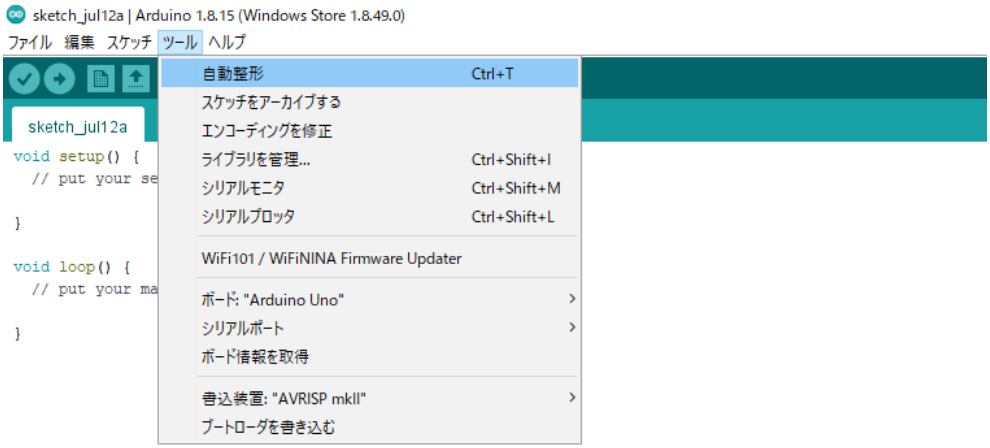

図 2.15: 15

図 2.16: 17

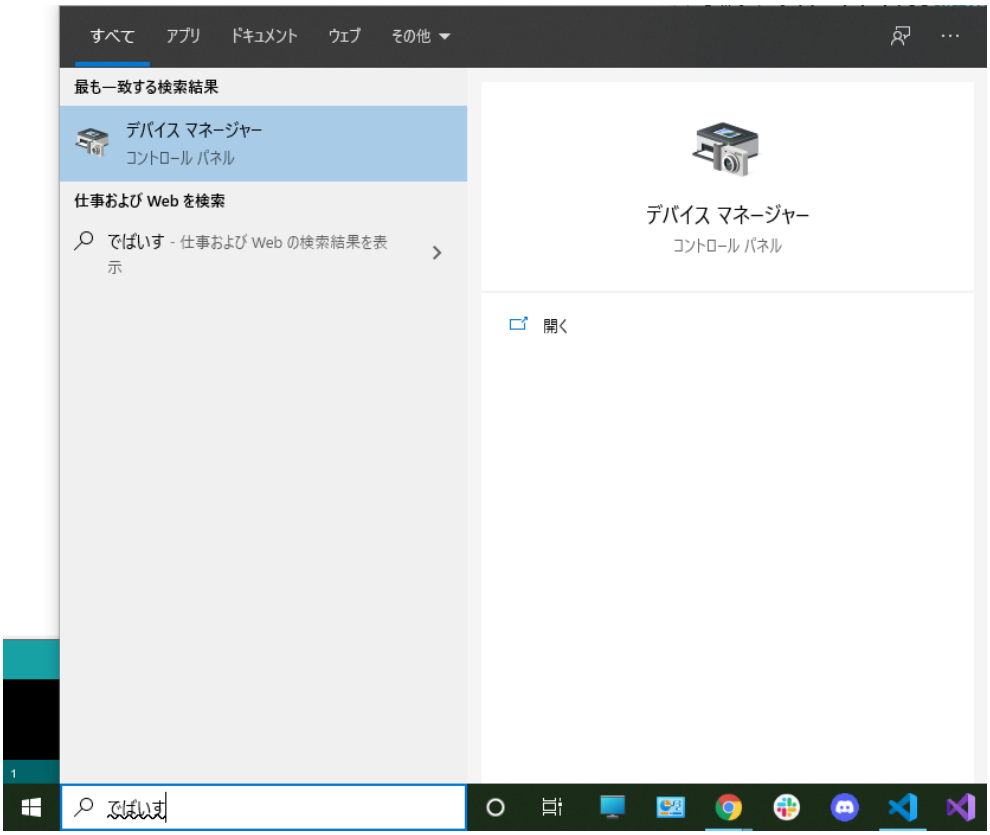

図 2.17: 18

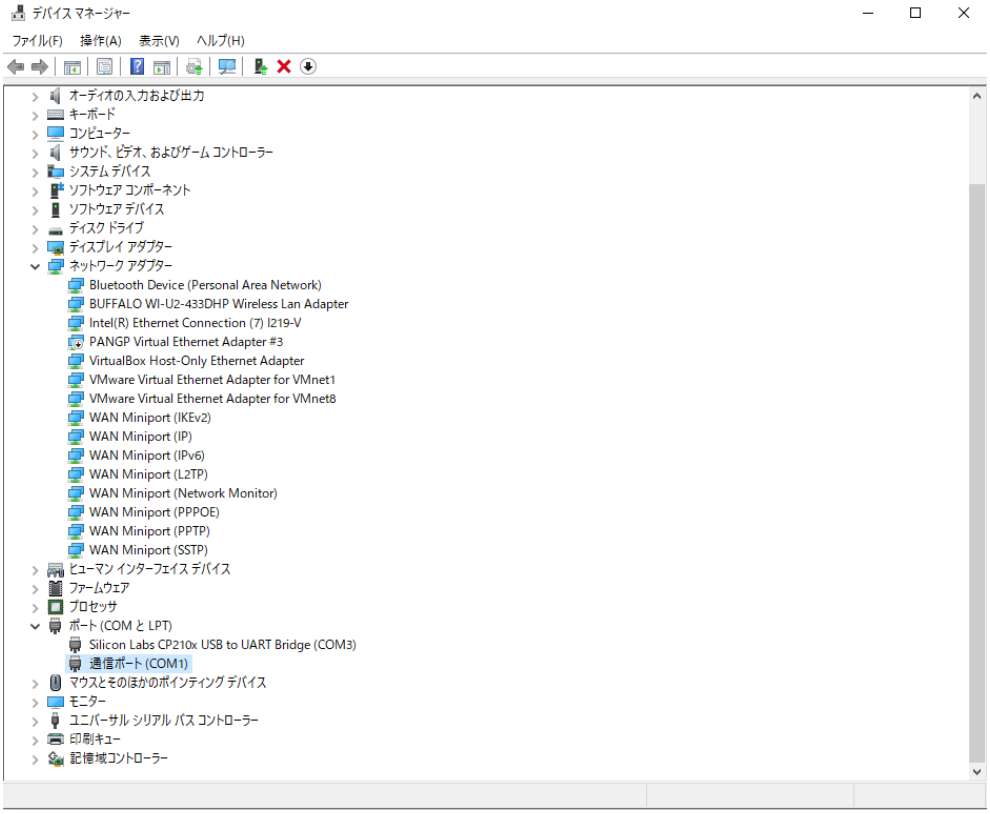
デバイス マネージャー
ファイル(F) 操作(A) 表示(V) ヘルプ(H)
オーディオの入力および出力
キーボード
コンピューター
サウンド、ビデオ、およびゲーム コントローラー
システム デバイス
ソフトウェア コンポーネント
ソフトウェア デバイス
ディスク ドライブ
ディスプレイ アダプター
ネットワーク アダプター
Bluetooth Device (Personal Area Network)
BUFFALO WI-U2-433DHP Wireless Lan Adapter
Intel(R) Ethernet Connection (7) I219-V
PANGP Virtual Ethernet Adapter #3
VirtualBox Host-Only Ethernet Adapter
VMware Virtual Ethernet Adapter for VMnet1
VMware Virtual Ethernet Adapter for VMnet8
WAN Miniport (IKEv2)
WAN Miniport (IP)
WAN Miniport (IPv6)
WAN Miniport (L2TP)
WAN Miniport (Network Monitor)
WAN Miniport (PPPOE)
WAN Miniport (PPTP)
WAN Miniport (SSTP)
ヒューマン インターフェイス デバイス
ファームウェア
プロセッサ
ポート (COM と LPT)
Silicon Labs CP210x USB to UART Bridge (COM3)
通信ポート (COM1)
マウスとそのほかのポインティング デバイス
モニター
ユニバーサル シリアル バス コントローラー
印刷キュー
記憶域コントローラー

図 2.18: 19

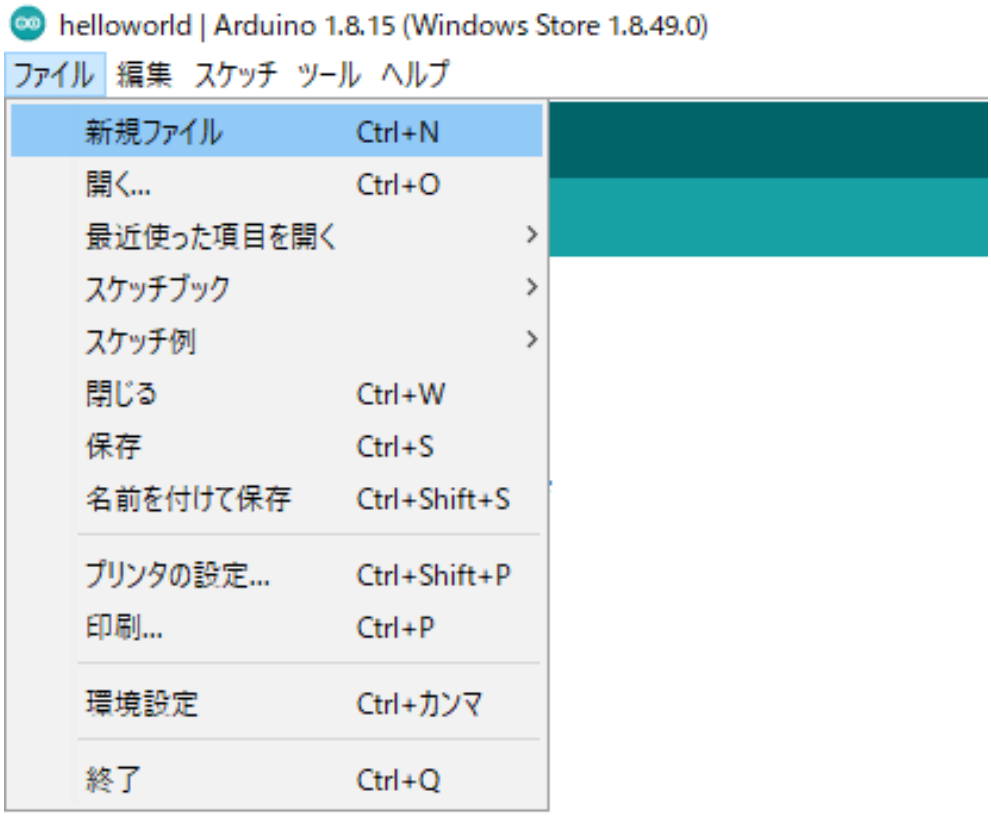
helloworld | Arduino 1.8.15 (Windows Store 1.8.49.0)
ファイル 編集 スケッチ ツール ヘルプ
新規ファイル Ctrl+N
開く... Ctrl+O
最近使った項目を開く
スケッチブック
スケッチ例
閉じる Ctrl+W
保存 Ctrl+S
名前を付けて保存 Ctrl+Shift+S
プリンタの設定... Ctrl+Shift+P
印刷... Ctrl+P
環境設定 Ctrl+カンマ
終了 Ctrl+Q

図 2.19: 20

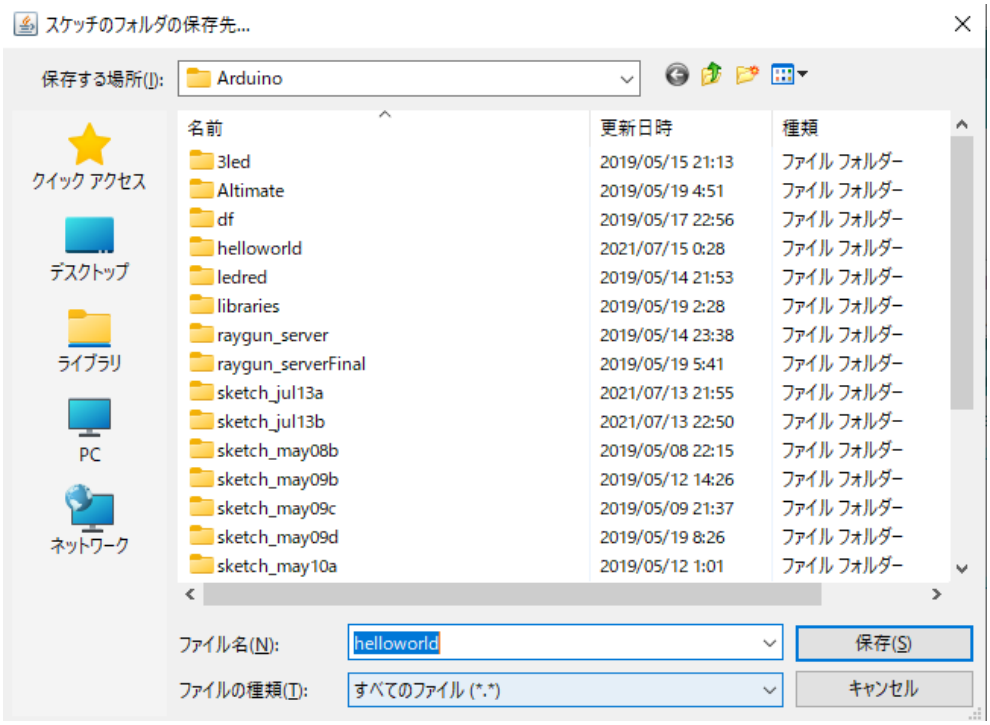

図 2.20: 21

図 2.21: 22

図 2.22: 23

図 2.23: 24

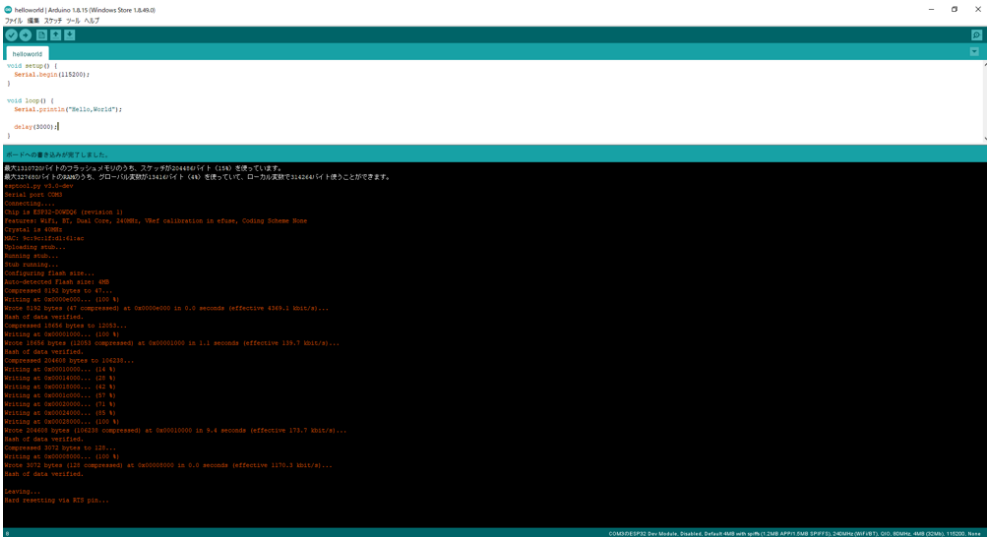

図 2.24: 25

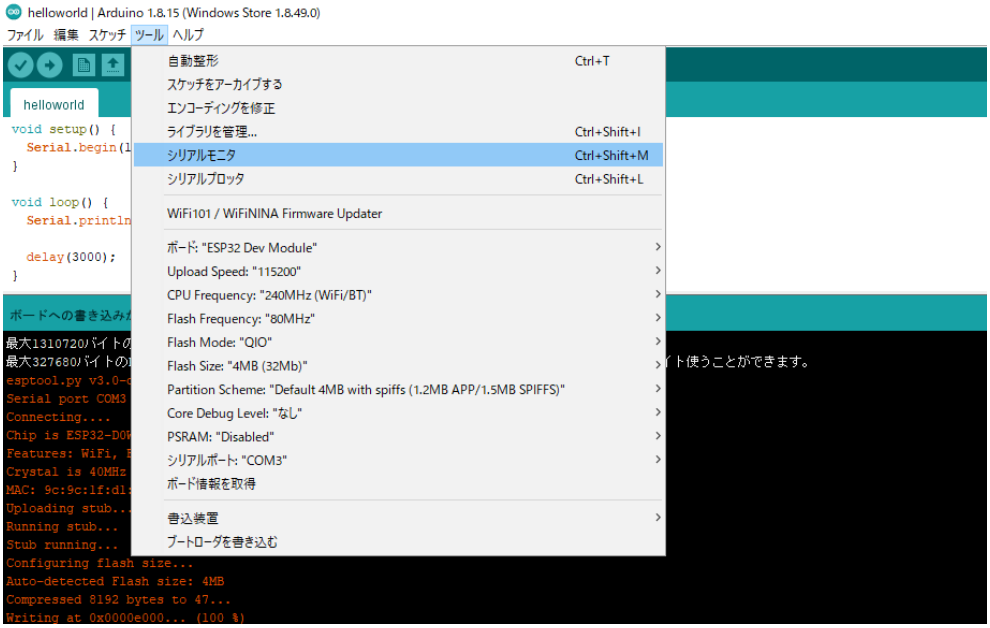

図 2.25: 26

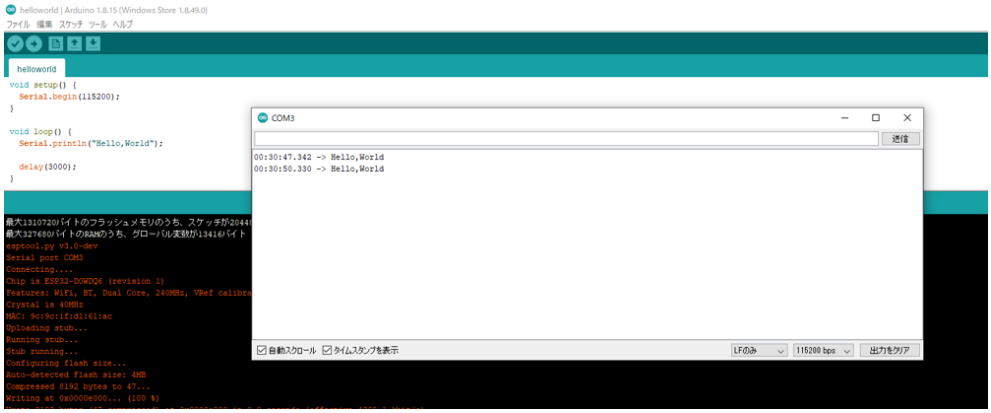

図 2.26: 27

リスト 2.1: 最初のプログラム

```
void setup() {
  Serial.begin(115200);
}

void loop() {
  Serial.println("Hello,World");
  delay(3000);
}
```

■コラム: シリアル通信とは

ArduinoIDE はシリアルモニタを備えていて、Arduino とコミュニケーションすることができます。

第 3 章

電子回路基礎

3.1　部品説明

LED

　　アノード　　極性は端子の長いほうをアノードと呼び　　電源の＋に接続する　　カソード　　端子の短いほうをカソードと呼ぶ　　カソード側は中の金属板が大きい　　点灯のために必要な情報　　順電圧（vf）　　順電流（lf）　　ラズパイで利用する場合は順電圧が 2v 程度，順電流が 20mA 程度=== ジャンプワイヤオスメス

抵抗

　抵抗見分け方

タクトスイッチ

　　プルアップとプルダウン　　スイッチを利用すれば 2 つの値を切り替えられる回路を作れます。　　しかし，スイッチがオフの場合では，出力する端子が解放状態（何も接続されてない状態）になる　　この場合周囲の雑音を拾ってしまい，値が安定しない状態になる　　そこで，プルダウンやプルアップを使って安定させる　　方法としては GND や Vdd（電源）に接続しておく方法　　こうしておくことでスイッチがオフ状態のとき，出力端子に接続されている抵抗を介して値を安定させる　　スイッチ OFF 時に 0v に安定させる方法をプルダウン　　電圧がかかった状態に安定させる方法をプルアップと呼ぶ

3.2 L チカしよう！

プログラムで L チカ

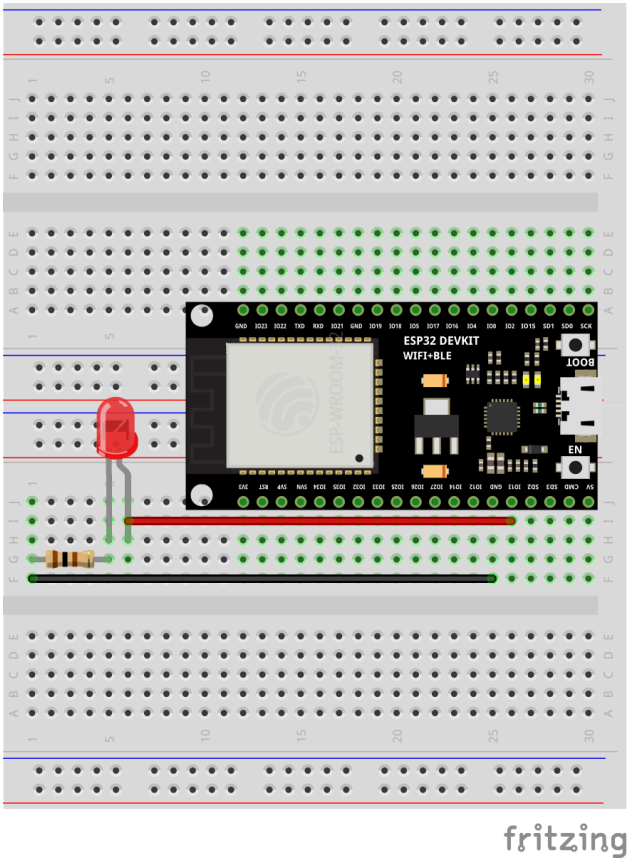

図 3.1: led2

リスト 3.1: Ltic

```
void setup() {
  pinMode(13, OUTPUT);
}
void loop() {
  digitalWrite(13, HIGH);
  delay(100);
  digitalWrite(13, LOW);
  delay(100);
}
```

タクトスイッチで L チカ

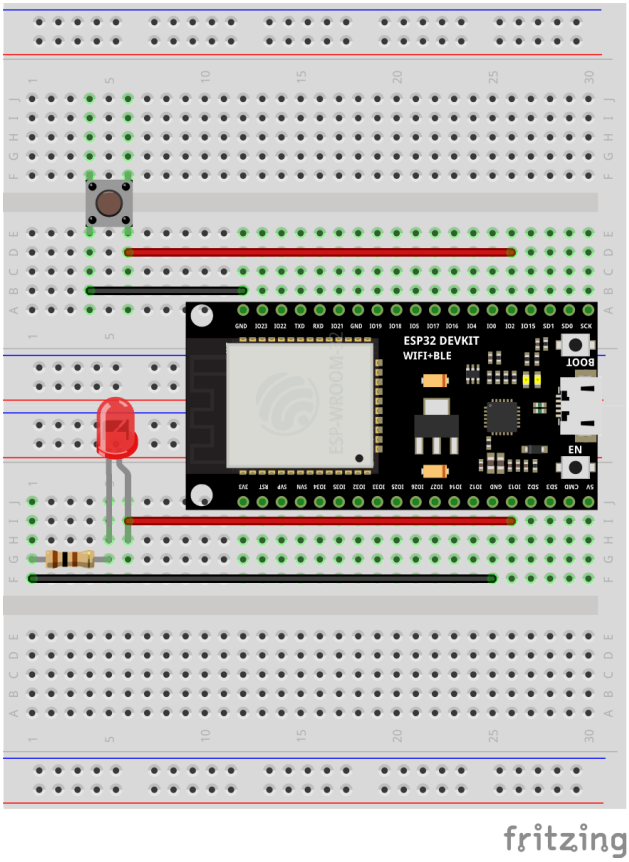

図 3.2: switch2

■コラム: チャタリング

第 4 章

取得データを Web に公開しよう！

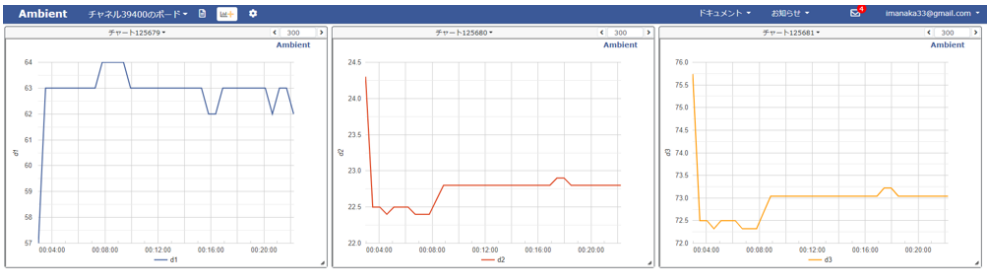

図 4.1: 6

4.1 センサーを使おう

I2C とは

温湿度センサー

LCD

4.2 Web に公開しよう

ambient について

Ambient は IoT データの可視化サービスです。https://ambidata.io/


Ambient
公開ボード
ドキュメント
お知らせ
ログイン
ユーザー登録(無料)
AmbientはIoTデータの可視化サービスです。
マイコンなどから送られるセンサデータを受信し、蓄積し、可視化(グラフ化)します。
Ambient
受信
可視化
マイコン
センサー
インターネット
チャネル
蓄積
簡単、でも強力なグラフ化機能
AmbientはIoTのアイデアをなるべく簡単にプロトタイプするお手伝いをします。細かな初期設定をしなくても送ったデータをリアルタイムでグラフ化します。例えばセンサから得られた温度と湿度のデータを初期設定をしないで送った場合でも、次のようにグラフが表示されます。

図 4.2: 1


Ambient
公開ボード
ドキュメント
お知らせ
ログイン
ユーザー登録(無料)
メールアドレス
パスワード
パスワード再入力
ユーザー登録(無料)
登録した時点で「Ambient利用規約」に書かれた内容に同意したものとします。
AmbientData Inc.
利用規約 会社概要

図 4.3: 2

図 4.4: 3

図 4.5: 4

チャネルを作成します。

図 4.6: 5

ライブラリのインストール

回路図

コーディング

第 5 章

API を使おう！

5.1　Weather API を使う

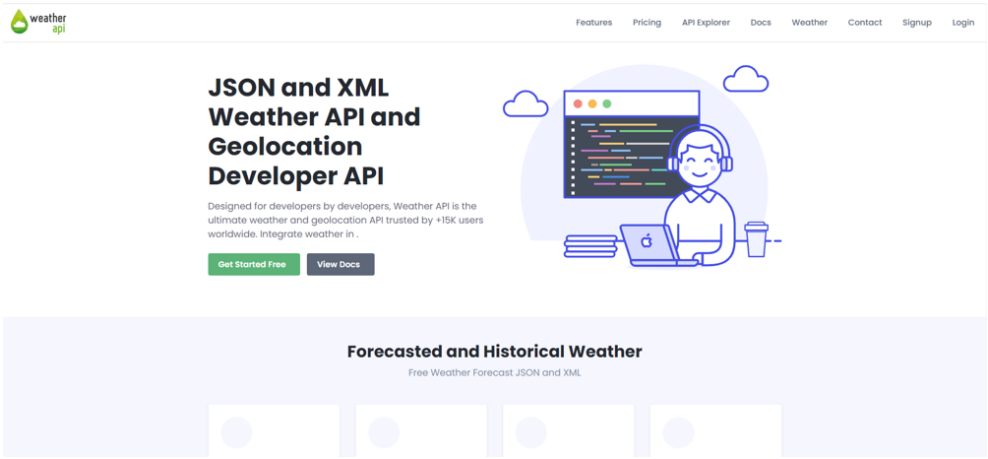

図 5.1: 1

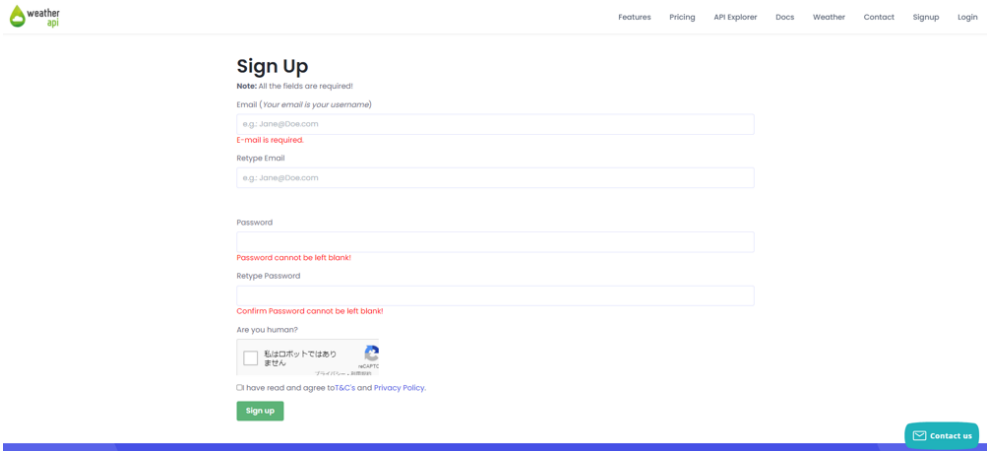

図 5.2: 2

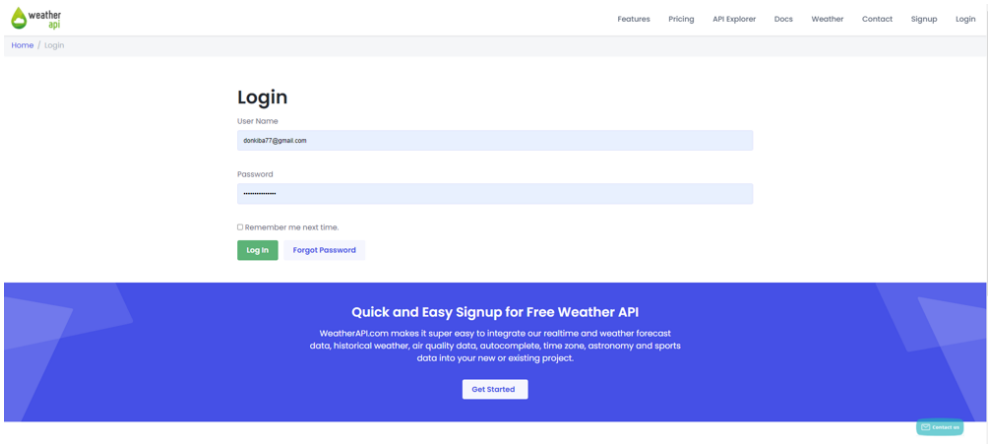

図 5.3: 3

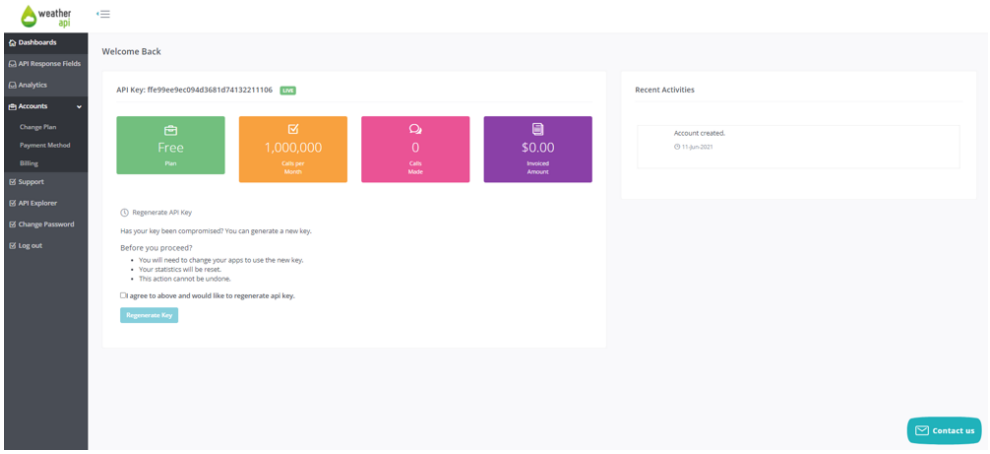

図 5.4: 4

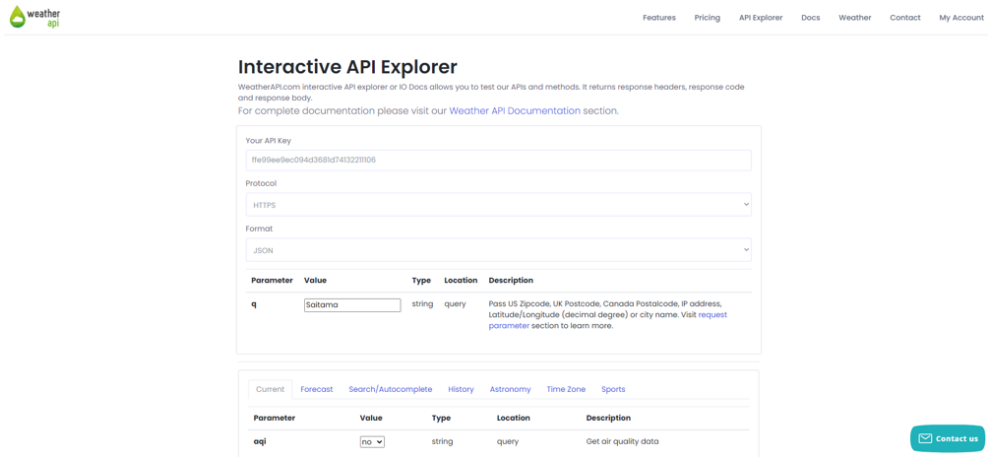

図 5.5: 5

Weather API https://www.weatherapi.com/

Call https://api.weatherapi.com/v1/current.json?key={key}&q=Saitama&aqi=no

ResponeseCode 200

リスト 5.1: ResponesHeader

```
{
    "Transfer-Encoding": "chunked",
    "Connection": "keep-alive",
    "Vary": "Accept-Encoding",
    "CDN-PullZone": "93447",
    "CDN-Uid": "8fa3a04a-75d9-4707-8056-b7b33c8ac7fe",
    "CDN-RequestCountryCode": "FI",
    "CDN-EdgeStorageId": "615",
    "CDN-CachedAt": "2021-07-12 14:05:36",
    "CDN-RequestPullSuccess": "True",
    "CDN-RequestPullCode": "200",
    "CDN-RequestId": "a45be49d32c7a76559a3f3920d337f53",
    "CDN-Cache": "MISS",
    "Cache-Control": "public, max-age=180",
    "Content-Type": "application/json",
    "Date": "Mon, 12 Jul 2021 12:05:36 GMT",
    "Server": "BunnyCDN-FI1-615"
}
```

リスト 5.2: ResponseBody

```
{
      "location": {
          "name": "Saitama",
          "region": "Saitama",
          "country": "Japan",
          "lat": 35.91,
          "lon": 139.66,
          "tz_id": "Asia/Tokyo",
          "localtime_epoch": 1626091536,
          "localtime": "2021-07-12 21:05"
      },
      "current": {
          "last_updated_epoch": 1626087600,
          "last_updated": "2021-07-12 20:00",
          "temp_c": 29.4,
          "temp_f": 84.9,
          "is_day": 0,
          "condition": {
              "text": "Partly cloudy",
              "icon": "//cdn.weatherapi.com/weather/64x64/night/116.png",
              "code": 1003
          },
          "wind_mph": 7.6,
          "wind_kph": 12.2,
          "wind_degree": 162,
          "wind_dir": "SSE",
          "pressure_mb": 1010.0,
          "pressure_in": 30.3,
```

```
        "precip_mm": 0.0,
        "precip_in": 0.0,
        "humidity": 61,
        "cloud": 47,
        "feelslike_c": 32.1,
        "feelslike_f": 89.8,
        "vis_km": 10.0,
        "vis_miles": 6.0,
        "uv": 7.0,
        "gust_mph": 9.2,
        "gust_kph": 14.8
    }
}
```

この形式を JSON（JavaScript Object Node）と言います。これらの形式が、Weather-API から帰ってくるため、ESP32 側で使えるようにしなければなりませんそこで、公開されているライブラリである arduinoJSON を利用します。

ライブラリのインストール

リスト 5.3: 最初のプログラム

```
// String input;

  StaticJsonDocument<1536> doc;

  DeserializationError error = deserializeJson(doc, input);

  if (error) {
    Serial.print(F("deserializeJson() failed: "));
    Serial.println(error.f_str());
    return;
  }

  JsonObject location = doc["location"];
  const char* location_name = location["name"]; // "Saitama"
  const char* location_region = location["region"]; // "Saitama"
  const char* location_country = location["country"]; // "Japan"
  float location_lat = location["lat"]; // 35.91
  float location_lon = location["lon"]; // 139.66
  const char* location_tz_id = location["tz_id"]; // "Asia/Tokyo"
  long location_localtime_epoch = location["localtime_epoch"]; // 1626533912
  const char* location_localtime = location["localtime"]; // "2021-07-17 23:58"

  JsonObject current = doc["current"];
  long current_last_updated_epoch = current["last_updated_epoch"]; // 1626533100
  const char* current_last_updated = current["last_updated"]; // "2021-07-17 23:45"
  float current_temp_c = current["temp_c"]; // 23.3
  float current_temp_f = current["temp_f"]; // 73.9
  int current_is_day = current["is_day"]; // 0

  JsonObject current_condition = current["condition"];
  const char* current_condition_text = current_condition["text"]; // "Clear"
  const char* current_condition_icon = current_condition["icon"];
  int current_condition_code = current_condition["code"]; // 1000
```

```
float current_wind_mph = current["wind_mph"]; // 3.8
float current_wind_kph = current["wind_kph"]; // 6.1
int current_wind_degree = current["wind_degree"]; // 250
const char* current_wind_dir = current["wind_dir"]; // "WSW"
int current_pressure_mb = current["pressure_mb"]; // 1019
float current_pressure_in = current["pressure_in"]; // 30.6
int current_precip_mm = current["precip_mm"]; // 0
int current_precip_in = current["precip_in"]; // 0
int current_humidity = current["humidity"]; // 88
int current_cloud = current["cloud"]; // 0
float current_feelslike_c = current["feelslike_c"]; // 24.9
float current_feelslike_f = current["feelslike_f"]; // 76.9
int current_vis_km = current["vis_km"]; // 16
int current_vis_miles = current["vis_miles"]; // 9
int current_uv = current["uv"]; // 1
float current_gust_mph = current["gust_mph"]; // 13.9
float current_gust_kph = current["gust_kph"]; // 22.3
```

API とは？

■コラム: サーバクライアント

サーバ？ クライアント？ とは何

Web サーバからの L チカ

第 6 章

応用編

6.1　外部からエアコンの電源を操作する

6.2　2 台の ESP32 を使ってピンポンする

6.3　VScode から ESP32 にスケッチを書き込む

付録 A

トラブルシューティング

A.1　シリアルモニタで文字化けがする

A.2　プログラムが書き込めない

A.3　プログラムを書き込んだが動作に反映されない

著者紹介

THEToilet / @THEToilet

あとがきみたいなのにあこがれていました。

はじめての IoT 講座

2021 年 7 月 12 日　初版第 1 刷　発行

著　者　THEToilet