

# M5Stack/M5Stick C の開発環境（Arduino IDE）のセットアップ方法・説明書

国野 亘 (<https://bokunimo.net/>)

## 本説明書について

本説明書では、M5Stack / M5Stick C 用の開発環境（Arduino IDE）を Windows 10 にセットアップする方法と、Arduino 言語の基礎について説明します。

## 準備 1. 統合開発環境 Arduino IDE をインストールする

Arduino 言語でプログラムを作成するための統合開発環境 Arduino IDE は、下記のサイトからダウンロードすることができます。

Arduino IDE:

<https://www.arduino.cc/en/Main/Software>

支払い画面では、\$3~\$50 の中から金額を選択し、クレジットカードで支払うことができます。初めて試すのであれば最低金額の\$3 で十分でしょう。クレジットカードを所持していない場合や、寄付したくない場合は、「JUST DOWNLOAD」を選択し、無料でダウンロードすることも出来ます。

## 準備 2. Wi-Fi マイコン ESP32 開発キットをインストールする

M5Stack や M5Stick C には、中国 Espressif Systems 社の Wi-Fi 搭載マイコン ESP32 が搭載されています。Arduino IDE にマイコン ESP32 用の開発キットを以下の手順で追加してください。

- ① Arduino IDE の「ファイル」メニューから[環境設定]を開きます。
- ② 「追加のボードマネージャの URL」に「[https://dl.espressif.com/dl/package\\_esp32\\_index.json](https://dl.espressif.com/dl/package_esp32_index.json)」を入力します (図 1)。
- ③ 「ツール」メニュー内の「ボード」から「ボードマネージャ」を開きます。
- ④ 検索ボックスに「ESP32」と入力し、「esp32 by Espressif Systems」を選択し、[インストール]をクリックします (図 2)。このインストールには環境によって 30 分くらいの時間がかかります。
- ⑤ Arduino IDE のメニューを「ツール」→「ボード」→「ESP32 Arduino」と進み、M5Stack の場合は [M5Stack-Core-ESP32] を、M5Stick C の場合は [M5Stick-C] を選択してください。

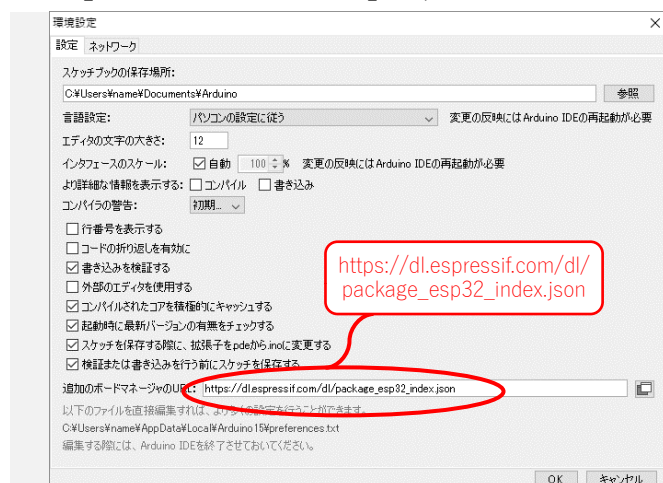


図 1 Arduino IDE の環境設定画面

Arduino IDE の「ファイル」メニューから[環境設定]を選択すると設定画面が表示される。「追加のボードマネージャの URL」に「[https://dl.espressif.com/dl/package\\_esp32\\_index.json](https://dl.espressif.com/dl/package_esp32_index.json)」を入力する

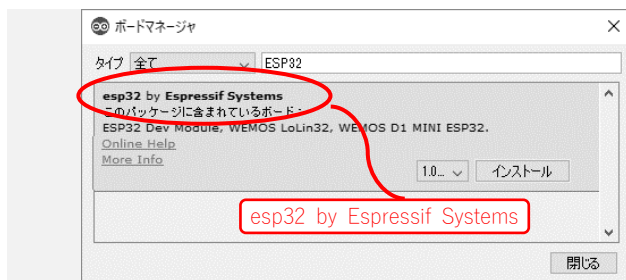


図 2 Arduino IDE のボードマネージャ

「ツール」メニュー内の「ボード」から「ボードマネージャ」を選択すると表示される。検索ボックスに「ESP32」と入力し、「esp32 by Espressif Systems」を選択し、[インストール]をクリックする

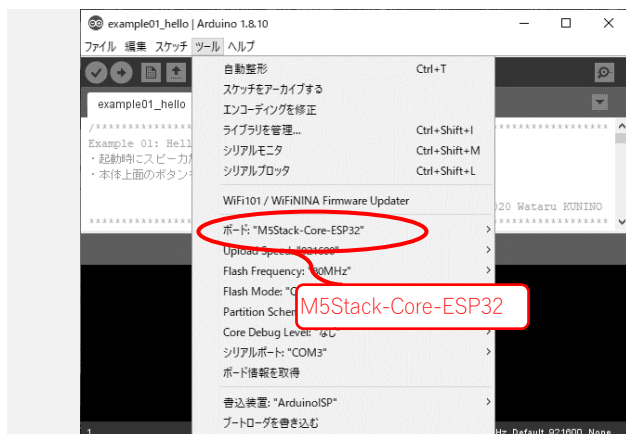


図 3 M5Stack を使用する場合の設定例

Arduino IDE の「ツール」メニュー内の「ボード」から「ESP32 Arduino」内の [M5Stack-Core-ESP32] または [M5Stick-C] を選択する

### 準備 3. M5Stack / M5Stick C 用ライブラリをインストールする

液晶ディスプレイなどの機能を、より簡単に扱うために、M5Stack や M5Stick C 専用ライブラリをインストールします。

- ① Arduino IDE の「スケッチ」メニューから「ライブラリをインクルード」を選択し、「ライブラリを管理」を選択すると、ウィンドウ「ライブラリマネージャ」が開きます。
- ② M5Stack の場合は、検索欄に「M5Stack」を入力し、「M5Stack by M5Stack」を選択します。M5Stick C の場合は、「M5StickC」を入力し、「M5StickC by M5StickC」を選択します（図 4）。
- ③ [インストール] ボタンを押すと、専用ライブラリがインストールされます。



図 4 Arduino IDE のライブラリマネージャ

M5Stack の場合は、検索欄に「M5Stack」を入力し、「M5Stack by M5Stack」を選択する。（M5Stick C の場合は、「M5StickC」を入力し、「M5StickC by M5StickC」を選択）

### 準備 4. サンプル・プログラムをダウンロードする

本稿用のサンプル・プログラム集を未だダウンロードしていない場合は、インターネット・ブラウザまたは Git コマンド (git clone https://github.com/bokunimowakaru/m5adc) でダウンロードしてください。

サンプル・プログラム集：

<https://github.com/bokunimowakaru/m5adc/archive/master.zip>

ダウンロードしたプログラム集 m5adc-master.zip 内の m5adc-master（または m5adc）フォルダは、Arduino IDE の「スケッチブックの保存場所」に展開（コピー）してください。「スケッチブックの保存

場所」が分からないときは、Arduino IDE の「ファイル」メニュー内の「環境設定」画面で確認します。

コピー後、Arduino IDE を再起動すると、「ファイル」メニュー内の「スケッチブック」に [m5adc-master] (または「m5adc」) が表示されます。M5Stack の場合は「m5stack」を、M5Stick C の場合は「m5stickc」を選択し、サンプル・プログラムを選択してください。まずは、図 5 のように、サンプル [example01\_serial] を選択してみましょう。

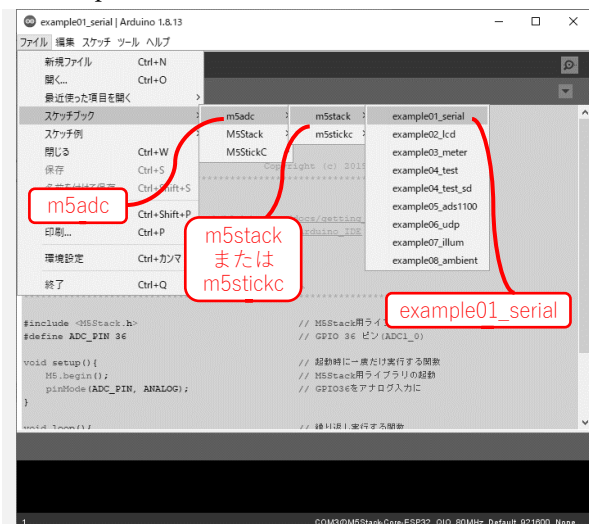


図 5 ダウンロードしたサンプル・プログラム集 m5adcを確認する

「ファイル」メニュー内の「スケッチブック」に表示される「m5adc-master」(または「m5adc」) を選択し、「m5stack」または「m5stickc」を選択してから [example01\_serial] を選択する。「m5adc」と同じ並びに「M5Stack」や「M5StickC」が表示されるので誤って選択しないよう注意する

## 準備 5. USB シリアル・ドライバのインストール

M5Stack /M5Stick C を、PC の USB 端子へ M5Stack /M5Stick C に付属するケーブルで接続すると、自動的に USB シリアル・ドライバがインストールされます。インストールには数分を要することもあります。Arduino IDE の「ツール」メニューの「シリアルポート」にポートが追加されれば準備完了です。複数の COM ポートがあるときは、M5Stack /M5Stick C の COM ポートを選択します。COM ポート番号を確認するには、有効な COM ポート番号を控えておき、一度、PC の USB 端子から本機を取り外します。非表示になった COM ポートが M5Stack /M5Stick C の COM ポートです。

サンプル・プログラムを書き込むには、Arduino IDE の上部に表示される右矢印ボタンをクリックします。ライブラリのコンパイルに数分を要しますが、2 回目以降は数十秒でコンパイルが完了します。Arduino IDE の右上のシリアル・モニタをクリックし、「adc=**nnn**, mv=**nnn**」のように A/D コンバータから取得した値が表示されることを確認してください。

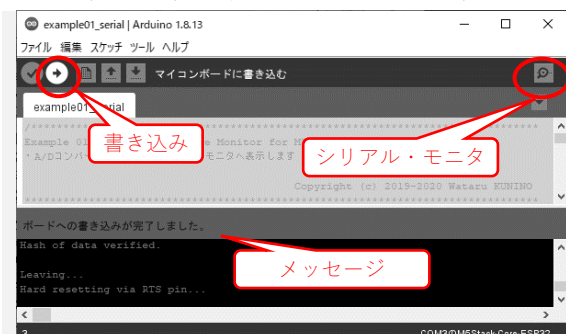


図 6 Arduino IDE を起動したときの様子の一例

コンパイルと M5Stack への書き込みを行うには左から 2 番目の右矢印ボタンをクリックする。中央の緑色のバーには、Arduino IDE の動作状態やメッセージが表示され、エラー発生時はオレンジ色に変化する

参考文献:M5Stack 社の公式サイト ([https://docs.m5stack.com/#/en/related\\_documents/Arduino\\_IDE](https://docs.m5stack.com/#/en/related_documents/Arduino_IDE))

## Arduino IDE 言語とは

Arduino 言語は、C/C++言語をベースにした組み込みマイコン向けプログラミング入門用に開発されたプログラミング言語です。C/C++言語は、IoT 対応機器に広く使われていますが、Python 言語などに比べ、自分で作成するプログラム量が多いことが課題でした。Arduino 言語であれば、ハードウェア専用のライブラリが標準装備されているので、主要機能の作成に集中し、簡潔なプログラムを作成することが出来ます。また、近年では Arduino 言語の影響を受けたり互換性を意識したりした C/C++言語用ライブラリも増えてきており、学習した知識は、本格的な IoT 対応機器の開発に役立つでしょう。

### Arduino 言語の特長：

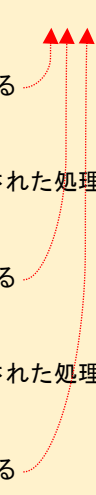
- ・組み込みマイコン向けプログラミング入門用
- ・IoT 対応機器で広く使われている C/C++言語ベース
- ・ハードウェア専用ライブラリで簡潔な記述が可能
- ・Arduino 言語の影響を受けたマイコン用ライブラリが増加中

## Arduino 言語のプログラムの基本的な構成

プログラミング言語は、マイコンの処理の手順を示すために人工的に作られた形式言語です。Arduino 言語のベースとなる C/C++言語では、主に「予約語」と呼ばれる基本的な命令や機能と、「関数」と呼ばれる小さなプログラムと、「ライブラリ」と呼ばれる関数の集まりで構成されます。

関数は、予約語や他の関数で構成された小さなプログラムで、繰り返し利用する手続きを切り出したものです。例えば、処理 A→処理 B→処理 A→処理 C→処理 A で構成されるプログラムがあった場合、処理 A を関数としてまとめることで、同じ処理 A のプログラムを何度も記述しなくて済むようになります。

### 関数の説明用プログラム（処理 A→B→A→C→A の処理 A を関数化）

<pre>void setup() {   // 処理 A 開始   // 予約語や関数で構成された処理手順   // 処理 A ここまで    // 処理 B 開始   // 予約語や関数で構成された処理手順   // 処理 B ここまで    // 処理 A 開始   // 予約語や関数で構成された処理手順   // 処理 A ここまで    // 処理 C 開始   // 予約語や関数で構成された処理手順   // 処理 C ここまで    // 処理 A 開始   // 予約語や関数で構成された処理手順   // 処理 A ここまで } void loop() { }</pre>	<pre>void a() { // 関数 a を定義する   // 処理 A   // 予約語や関数で構成された処理手順   // 処理 A ここまで }  void setup() {   a(); // 関数 a を実行する    // 処理 B   // 予約語や関数で構成された処理手順   // 処理 B ここまで    a(); // 関数 a を実行する    // 処理 C   // 予約語や関数で構成された処理手順   // 処理 C ここまで    a(); // 関数 a を実行する } void loop() { }</pre> 
---	---

関数の集まりのライブラリは、`#include` 命令でプログラムに組み込むことができます。例えば、`#include <M5Stack.h>`で M5Stack に搭載されたハードウェアを利用するためのライブラリを組み込むことが出来、M5Stack ライブラリ内の関数をプログラム内で使用することが出来るようになります。

### 液晶ディスプレイに Hello, world!を表示するプログラム

```
#include <M5Stack.h>
void setup() {
  M5.begin();
  M5.Lcd.println("Hello, world!");
}
void loop() {
}
```

ライブラリ M5Stack を組み込む  
波括弧「{」  
setup 関数  
コメント  
(プログラムではないことを示す)  
// M5Stack 用ライブラリの起動  
// LCD に Hello, world! を表示  
波括弧「}」  
セミコロン「;」  
(命令文の区切り)  
loop 関数

### Arduino 言語の setup 関数と loop 関数

Arduino 言語のプログラム内の「`void setup()`」と書かれた setup 関数は、マイコンの起動時に実行する初期設定処理部です。波括弧「{」から「}」で括られた範囲を実行します。M5Stack や M5Stick C では、setup 関数内に `M5.begin()` を記述し、M5Stack/M5Stick C 用のライブラリの初期化を行います。

「`void loop()`」と書かれた loop 関数は、setup 関数の処理後に、繰り返し実行する処理部です。マイコンのメイン処理となる通常動作を記述します。

プログラムを見やすく記述するには、命令や関数ごとに改行し、命令文や定義範囲を示す波括弧内にインデント（字下げ）を付与します。C/C++言語では、改行や、インデント、スペース文字は、プログラムの動作には（一部を除き）影響しませんが、見た目の不明確さがプログラムの誤りに繋がるので、なるべく守りましょう。

命令文の末尾には、命令の区切りを示すセミコロン「;」を付与します。前述のように、改行はプログラム上の作用がないので、命令文の区切りをセミコロンで明示する必要があります。

### C 言語の予約語

予約語には、プログラミング言語の基本的な命令や機能が多く含まれています。ライブラリの関数を使わずに、C 言語の予約語の組合せだけでプログラムを書くことも原理的には可能です。条件に応じた処理を行う if 構文や、繰り返し処理を行う for 構文、変数や関数の型を示す `int`, `float`, `char`, `void`, 処理の遷移を行う `return`, `break`, `continue` などがあります。C 言語の予約語は 30 個程度です。下表に予約語の一例を示します。これらの内容については、未だ理解できなくても大丈夫です。

#### C 言語の予約語の一例

予約語	構文の例	役割	主なサンプル
int	int a; int b=0; c=(int)b; int func(){処理}	整数型を示す	全サンプル
float	float a; float b=0.0; float func(){処理}	浮動小数点数型を示す	サンプル 5~8
void	void setup(){処理} void loop(){処理}	型が無いことを示す void setup	全サンプル
if	if(条件){処理} if(条件){処理}else{処理}	条件(不)一致時に処理を実行する	サンプル 5,7,8
for	for(初期化;条件;後処理){処理}	条件一致時に処理を繰り返す	サンプル 2,4,7,8
return	return; return a;	関数を終了し元の処理に戻る	サンプル 7~8



## C 言語の演算子

演算子は、最も基本的なコンピュータ処理です。変数への代入を行う代入演算子(=)や、四則演算(+, -, ×, ÷)などを行う算術演算子(+, -, \*, /), 比較を行うための比較演算子(==, <, >)などがあります。

変数 **a** に変数 **b** と **c** の値を加算して代入するには「**a** = **b** + **c** ;」のように記述し、変数 **a** と **b** を比較するには「if(**a** == **b**) { ~真のときに実行する処理 ~ }」のように記述します。「=」は代入を示し、「==」は比較を示す点に注意が必要です。使い方は、サンプル・プログラム 5, 7, 8 を参照してください。

## C 言語の変数

プログラム上で扱う数値や文字などのデータは、「変数」と呼ばれる容器を介在して、各命令や構文上で利用します。例えば、int **x** = 3; のように変数 **x** に 3 を代入し、次に int **y** = **x** + 1; のように変数 **x** に 1 を加算してから変数 **y** に代入するといった処理が行えます。

関数内で初めて変数を使うときは、int **x** = 3; や int **x**; のように、変数を定義する必要があります。「=」による代入を省略したときは、変数 **x** には何の値が入っているか分からない状態です。後に数値を代入してから使用します。

変数名にはアルファベットを使用します。通常、変数名の先頭には小文字を用いますが、定数など、通常の変数と区別するために意図的に大文字を使用することもあります。

変数には整数値を扱う整数型 int, 小数を含む数値を扱うことが出来る浮動小数点型 float などがあります。さらに Arduino 言語では文字列を扱うことが出来る String 型の変数もあります。

一度、定義した関数にデータを代入するときは **x** = 1; や **z** = **y** - **x**; のように変数型を付与せずに、直接、変数名に代入します。

### 変数の定義と代入

```
#include <M5Stack.h>

void setup() {
    M5.begin(); // M5Stack 用ライブラリの起動

    int x = 3; // 変数 x に 3 を代入する
    int y = x + 1; // 変数 y に x + 1 を代入する
    int z; // 変数 z を定義する

    z = y - x; // 変数 z に y - x を代入する
    M5.Lcd.printf("x=%d y=%d z=%d ¥n", x, y, z); // 変数 x, y, z の値を表示する

    x = 1; // 変数 x に 1 を上書きする
    y = 2; // 変数 y に 2 を上書きする
    z = x + y; // 変数 z に x + y を代入する
    M5.Lcd.printf("x=%d y=%d z=%d ¥n", x, y, z); // 変数 x, y, z の値を表示する
}

void loop() {
}
```

## サンプル・プログラムによる学習

本稿では、M5Stack / M5Stick C 用の Arduino 開発環境のセットアップ方法と、Arduino 言語の基本的な使い方について説明しました。Arduino 言語については、詳しく理解できなかったとしても、心配しないでください。引き続き、各サンプル・プログラムのコメントを参考に、Arduino 言語を使った IoT 対応機器のプログラミング学習を続けるうちに、徐々に理解が深まるでしょう。

《国野 亘》

本稿の二次使用を禁止します。本稿の改変を禁止します。損害について当方は一切補償しません。  
下記リポジトリ内の全複製（クローン含む）に限り、自由に複製・配布・利用することができます。

Git リポジトリ：<https://github.com/bokunimowakaru/m5adc>

Copyright (c) 2020 国野 亘 <https://bokunimo.net>