

ESP32でLINEへ画像投稿 [M5Stack-TimerCamera]

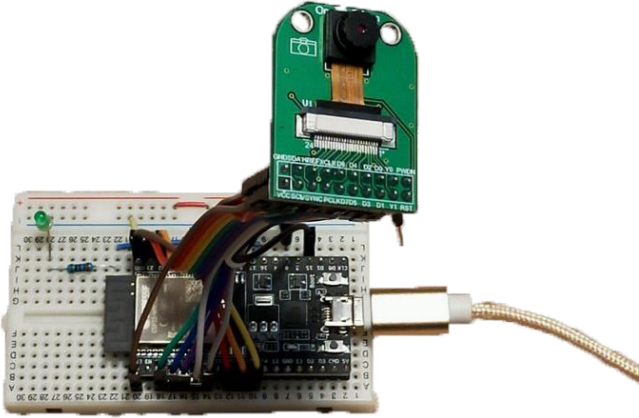

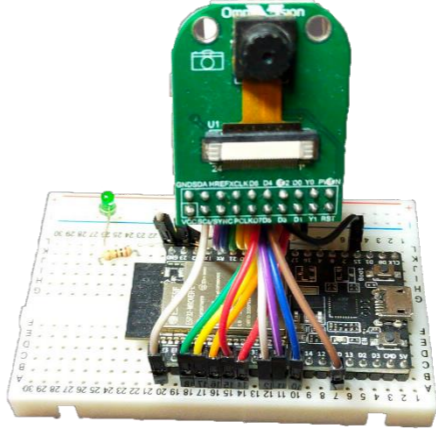

- HTTPS(TLS)クライアントの実装
- LINE-API利用による画像投稿

目次 《スマホで動画視聴》

1. 機器選定
2. 開発環境
3. ArduinoIDE設定
4. LINE設定(トークン取得)
5. Arduinoプログラム
6. プログラム書き込み、動作確認

1-1. 機器選定(4000円以下程度を目標)


※費用は時期により変動しますので参考です。

	同じハードウェア構成		近いハードウェア構成	
	①ESP32(WROOM)とOV2640	②M5Stack UnitCam (OV2640)	③ESP32(WROVER)とOV2640	④M5Stack TimerCamera (OV3660)
☒				
仕様	メモリ[SRAM]: 520kbyte、解像度: 2M pixel プログラム書込にはキットが必要*1		メモリ[SRAM]: 8Mbyte 解像度: 2M pixel	
用途	静止画		静止画、動画	
費用	3930円	M5Stack: UnitCam 18.95USD [marutsu: 2946円] + 1100円*1	4080円	M5Stack: F)19.95, X)17.95USD [SwitchScience: F)2860, X)2596円]
ソフト	ほぼ流用可能 (Arduinoのマザーボード設定やポートの使い方に違いがある)			
投稿	GoogleAPI, GoogleAppScript[GAS] による画像のGoogleDrive保存	-	-	スマホで動画視聴 ESP32でLINEへ画像投稿 今回

1-2. ESP32でのカメラ利用(価格詳細)

※費用は時期により変動しますので参考です。
 ※Hobby-ITサイトからExcelダウンロード可能

①ESP32(WROOM)とOV2640 【3930円】

NO	項目	数量	イメージ	商品名	URL	購入先	価格	備考
1	ESP32開発ボード WROOM	1		ESP32-DevKitC-3 2E ESP32-WROOM-32E 開発ボード 4MB	https://akizukidenshi.com/catalog/g/gM-15673/	秋月電子	1600	19Pin×2列仕様 (他社は15Pin×2列)
2	ブレッドボード 6穴版 EIC-3901	1		ブレッドボード 6穴版 EIC-3901	https://akizukidenshi.com/catalog/g/gP-12366/		460	
3	緑LED	1		3mm黄緑色LED 570nm 70度 OSG8HA3Z74A	https://akizukidenshi.com/catalog/g/gI-11637/		10	状態表示用
4	OV2640カメラモジュール	1		OV2640使用200万画素カメラ B0011	https://akizukidenshi.com/catalog/g/gM-13197/		1680	
5	ジャンパーケーブル	1		コネクタ付ケーブル 20cm 40P オススメ コネクタ付ケーブル 20cm長	https://akizukidenshi.com/catalog/g/gC-17228/		180	今回は手持ちを利用したのでコネクタ形状など未確認
総合計							3,930	別途送料が必要です

配線用のジャンパー線セットやLED抵抗は省略しました。

②M5Stack UnitCam 【4046円】

NO	項目	数量	イメージ	商品名	URL	購入先	価格	備考	
1	UnitCam	1		Unit Cam Wi-Fi Camera DIY Kit (OV2640)	https://shop.m5stack.com/collec-tions/m5-cameras https://www.marutsu.co.jp/pc/i/2228284/	M5Stack	\$18.95	UnitCam多数利用でも一つあれば良いです ソフトウェアの書込に必要	
2	USBシリアル変換モジュール	1		FT232XL 超小型USBシリアル変換モジュール	https://akizukidenshi.com/catalog/g/gM-08461/	marutsu	2946		
3	細ピンヘッダ 1×20	1		細ピンヘッダ 1×20	https://akizukidenshi.com/catalog/g/gC-04398/	秋月電子	20		
4	ジャンパーケーブル	1		コネクタ付ケーブル 20cm 40P オススメ コネクタ付ケーブル 20cm長	https://akizukidenshi.com/catalog/g/gC-17228/		180		今回は手持ちを利用したのでコネクタ形状など未確認
5	ブレッドボード	1		ブレッドボード BB-801	https://akizukidenshi.com/catalog/g/gP-05294/		220		
総合計							4,046	別途送料が必要です	



専用Uploaderもあるが、汎用性があるので今回はこの物品を選択

③ESP32(WROVER)とOV2640 【4080円】

NO	項目	数量	イメージ	商品名	URL	購入先	価格	備考
1	ESP32開発ボード WROVER	1		ESP32-DevKitC-V E ESP32-WROVER-E 開発ボード 8MB	https://akizukidenshi.com/catalog/g/gM-15674/	秋月電子	1750	19Pin×2列仕様 (他社は15Pin×2列)
2	ブレッドボード 6穴版 EIC-3901	1		ブレッドボード 6穴版 EIC-3901	https://akizukidenshi.com/catalog/g/gP-12366/		460	
3	緑LED	1		3mm黄緑色LED 570nm 70度 OSG8HA3Z74A	https://akizukidenshi.com/catalog/g/gI-11637/		10	状態表示用
4	OV2640カメラモジュール	1		OV2640使用200万画素カメラ B0011	https://akizukidenshi.com/catalog/g/gM-13197/		1680	
5	ジャンパーケーブル	1		コネクタ付ケーブル 20cm 40P オススメ コネクタ付ケーブル 20cm長	https://akizukidenshi.com/catalog/g/gC-17228/		180	今回は手持ちを利用したのでコネクタ形状など未確認
総合計							4,080	別途送料が必要です

配線用のジャンパー線セットやLED抵抗は省略しました。

④M5Stack TimerCamera(OV3660) 【2596/2860円】

NO	項目	数量	イメージ	商品名	URL	購入先	価格	備考
1	Timer Camera X	1		ESP32 PSRAM Timer Camera X (OV3660)	https://shop.m5stack.com/collec-tions/m5-cameras https://www.switch-science.com/products/6742	M5Stack	\$17.95	視野角 66.5°
					https://shop.m5stack.com/collec-tions/m5-cameras https://www.switch-science.com/products/6786	SWITCH SIENCE	2596	
1	Timer Camera F	1		ESP32 PSRAM Timer Camera F (OV3660)	https://shop.m5stack.com/collec-tions/m5-cameras https://www.switch-science.com/products/6786	M5Stack	\$18.95	視野角 120°
						SWITCH SIENCE	2860	
総合計							2,860	別途送料が必要です

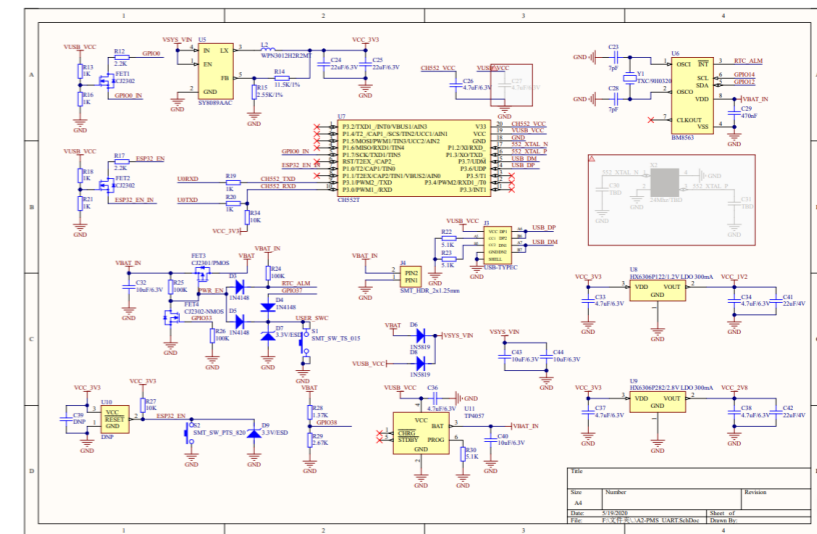
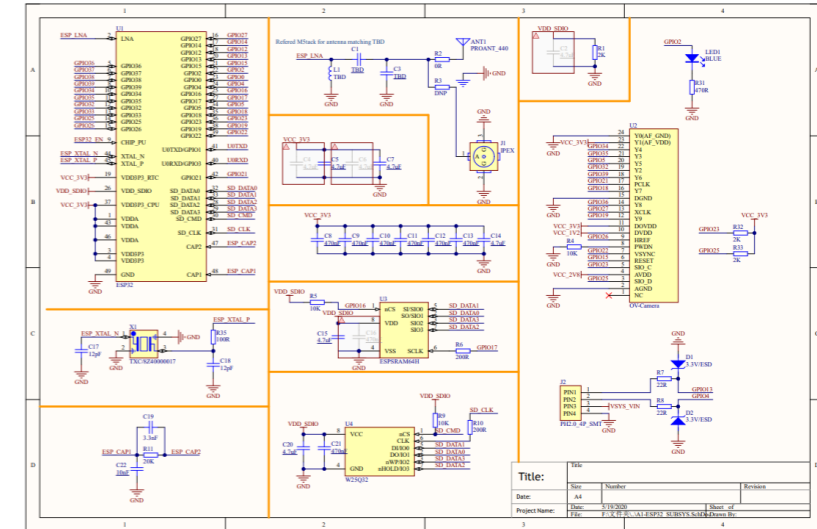
X/Fは視野角の違い
 マイクロUSBケーブル付きでパソコンがあれば開発可能

1 – 3. TimerCamera

● Pin Map

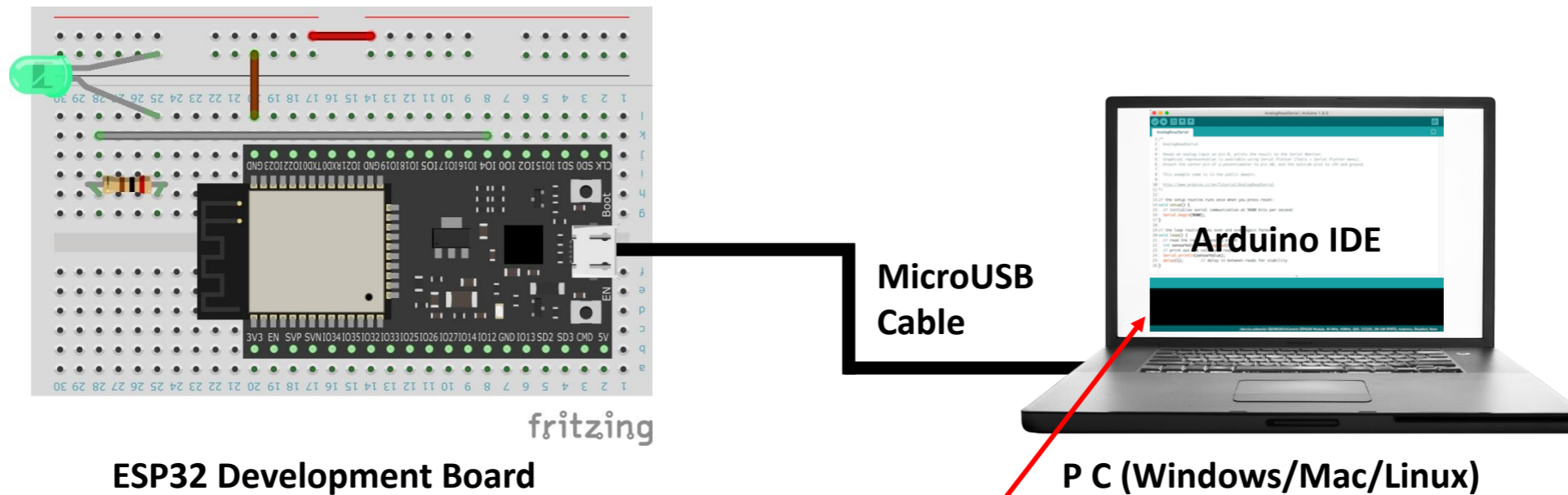
Interface	Camera Pin	TimerCamera
SCCB Clock	SIOC	IO23
SCCB Data	SIOD	IO25
System Clock	XCLK	IO27
Vertical Sync	VSYNC	IO22
Horizontal Reference	HREF	IO26
Pixel Clock	PCLK	IO21
Pixel Data Bit 0	D0	IO32
Pixel Data Bit 1	D1	IO35
Pixel Data Bit 2	D2	IO34
Pixel Data Bit 3	D3	IO5
Pixel Data Bit 4	D4	IO39
Pixel Data Bit 5	D5	IO18
Pixel Data Bit 6	D6	IO36
Pixel Data Bit 7	D7	IO19
Camera Reset	RESET	IO15
Camera Power Down	PWDN	-1
Power Supply 3.3V	3V3	3V3
Ground	GND	GND

● Schematic



2. 開発環境

開発環境はArduinoを利用していきます。



【Arduino Official site】

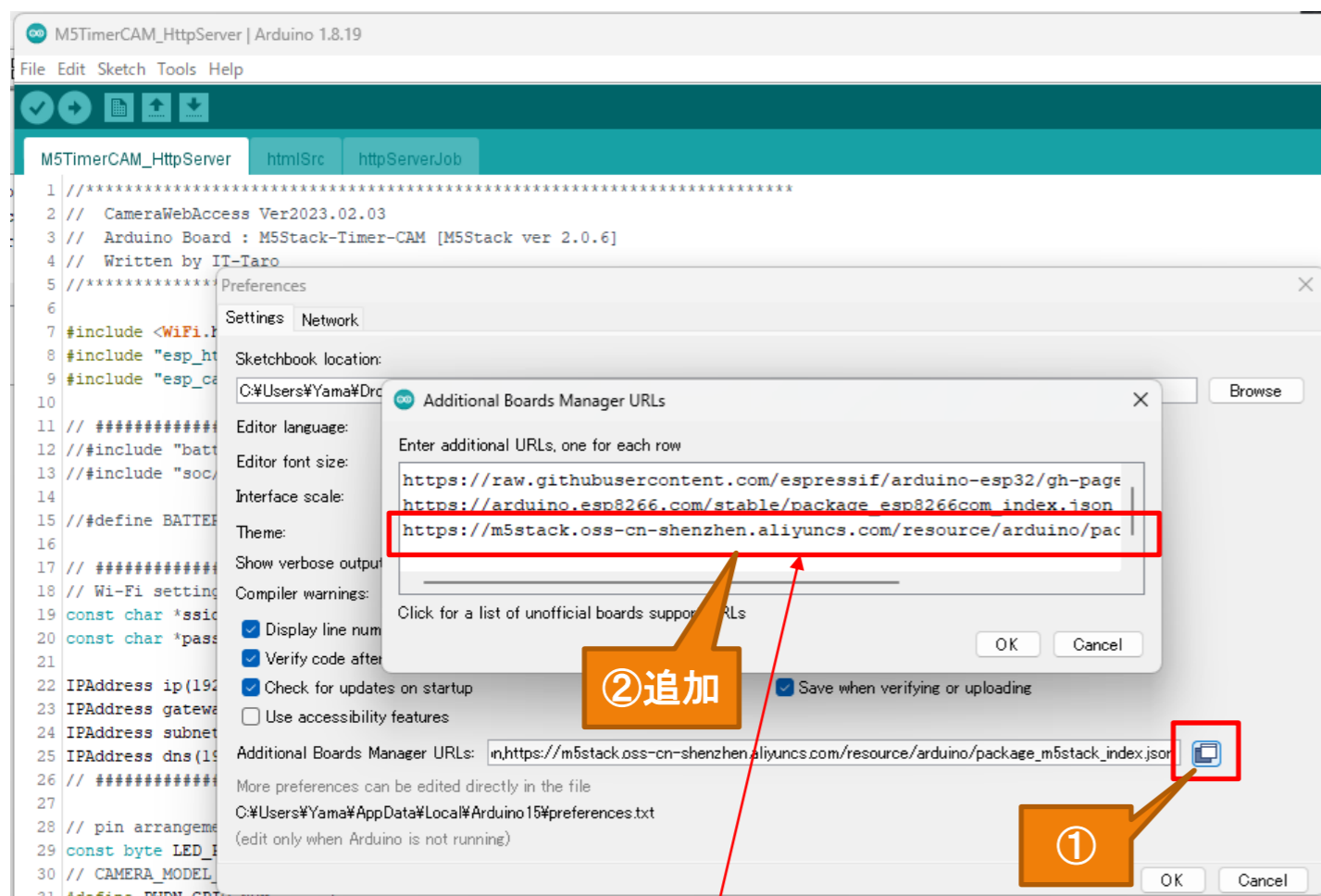
<https://www.arduino.cc/>

ダウンロード可能

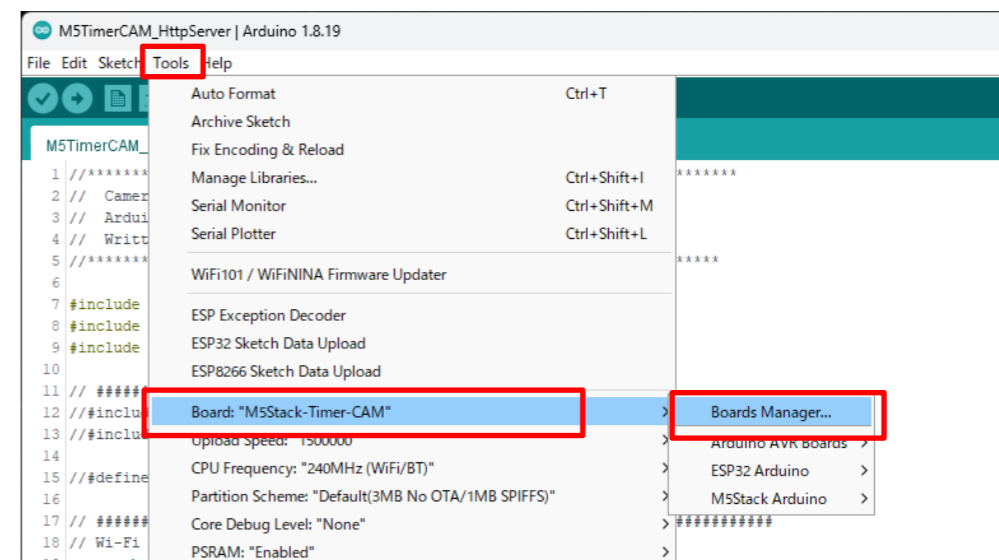
3-1. Arduino設定 (Board設定)

● M5Stack Official ArduinoIDE Setting
https://docs.m5stack.com/en/quick_start/timer_cam/arduino

1) ArduinoIDE設定からAdditional Board Manager設定を追加



2) Board Managerを起動



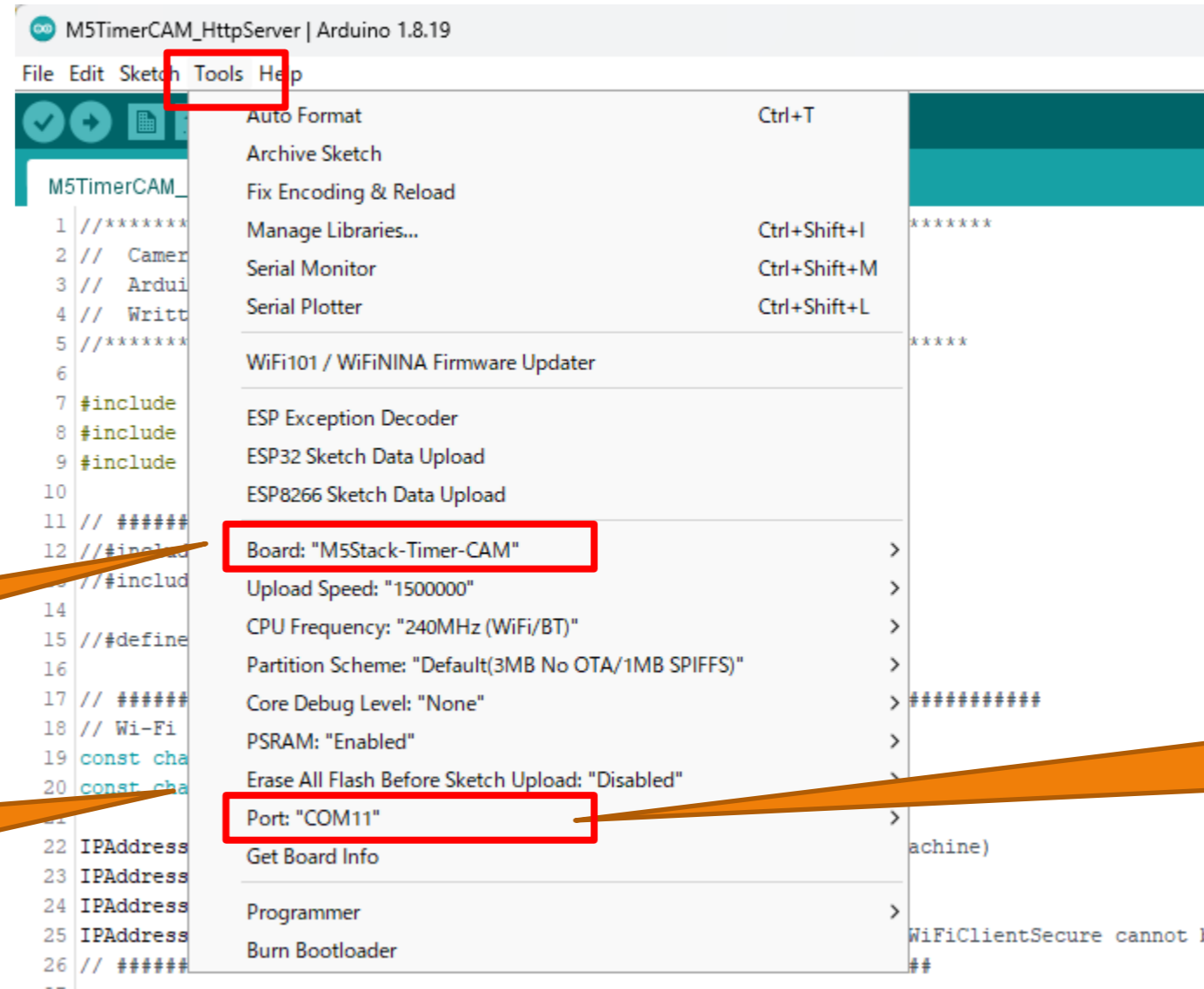
3) M5Stackをインストール



設定値:
https://m5stack.oss-cn-shenzhen.aliyuncs.com/resource/arduino/package_m5stack_index.json

3-1. Arduino設定 (Board設定)

4) Boardを「M5Stack-Timer-CAM」に設定



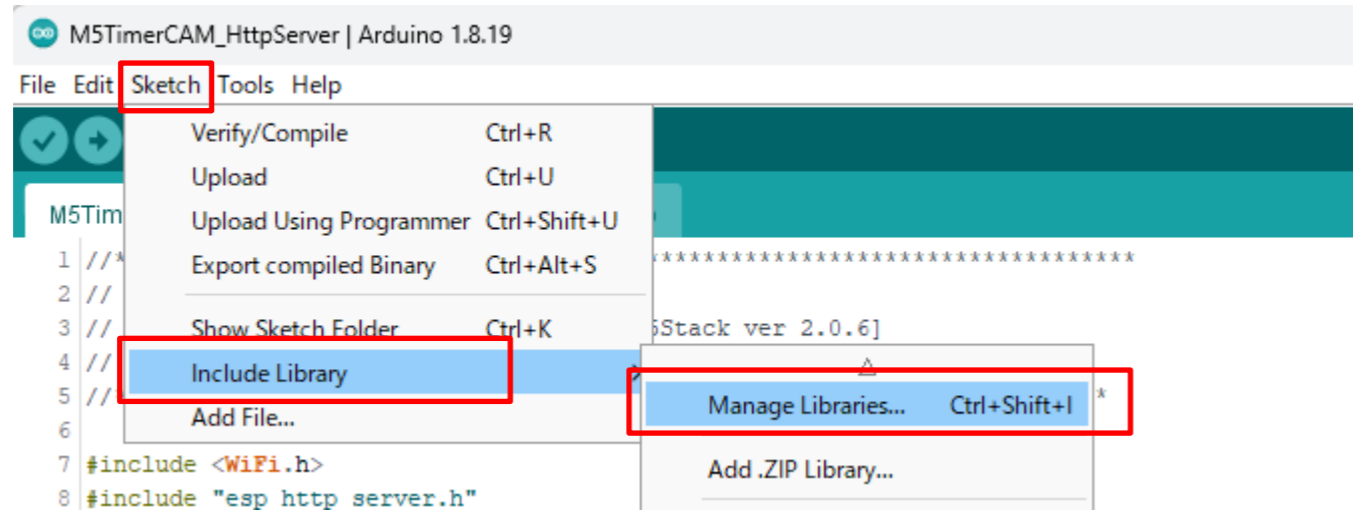
「M5Stack-Timer-CAM」を選択

他の設定は変更なし
(初期値のまま)

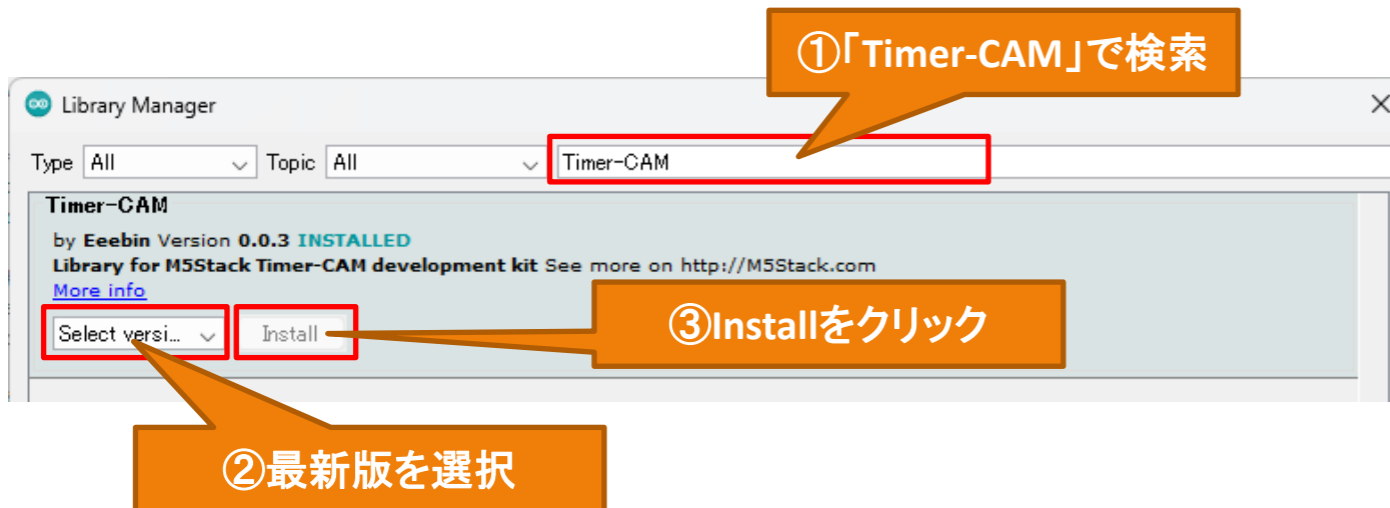
PortはTimerCameraが接続された
ポートを選択すること
【選択失敗時、書込エラー】

3-2. Arduino設定 (Library追加)

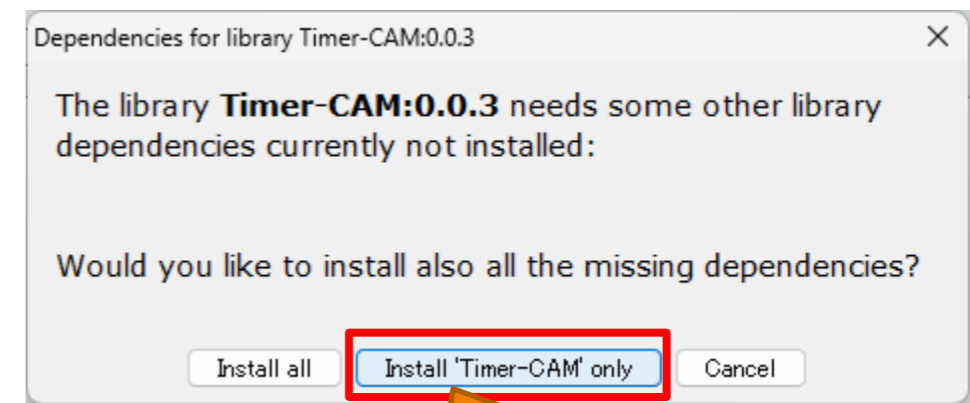
1) Library Managerを起動



2) 「Timer-CAM」をインストール



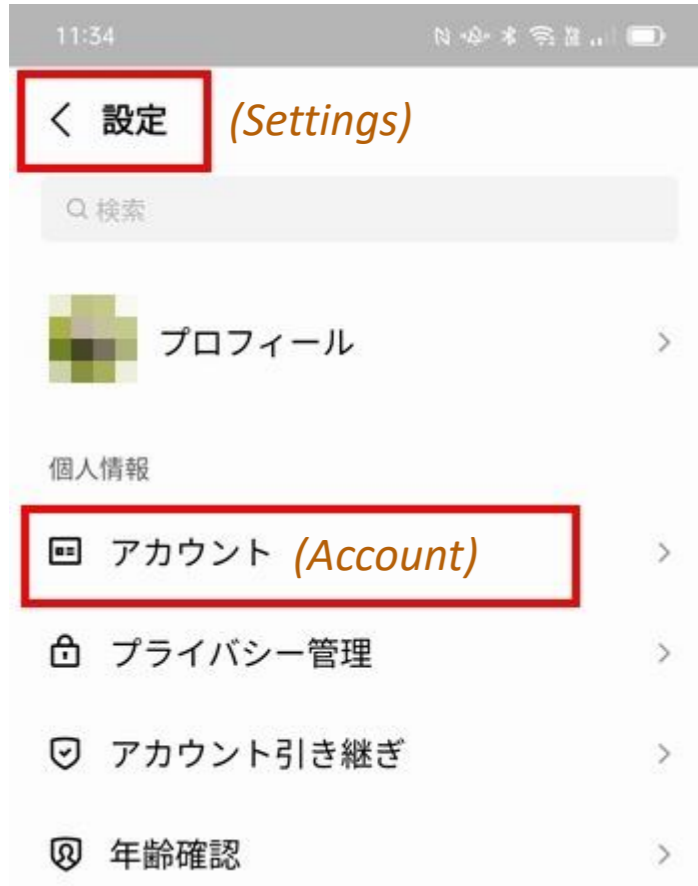
3) 「Timer-CAM」だけをインストール



これだけをインストール
(動作NG時、全てインストール)

4-1. LINE設定(トークン取得)

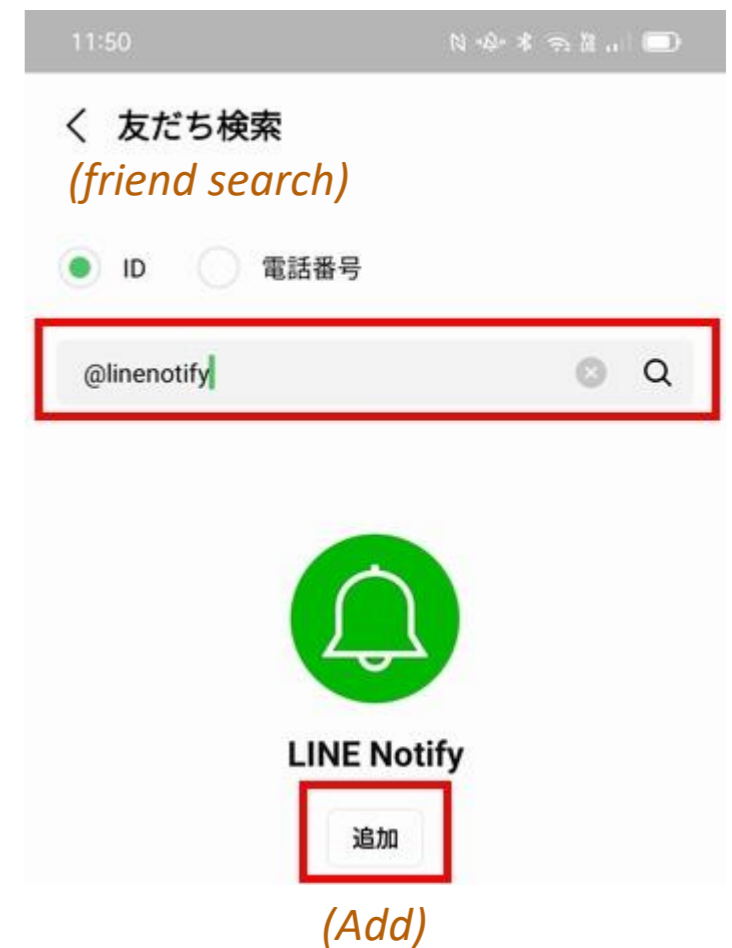
1) スマホLINEアプリの「設定」から
アカウントを選択



2) 「ログイン許可」をONにする

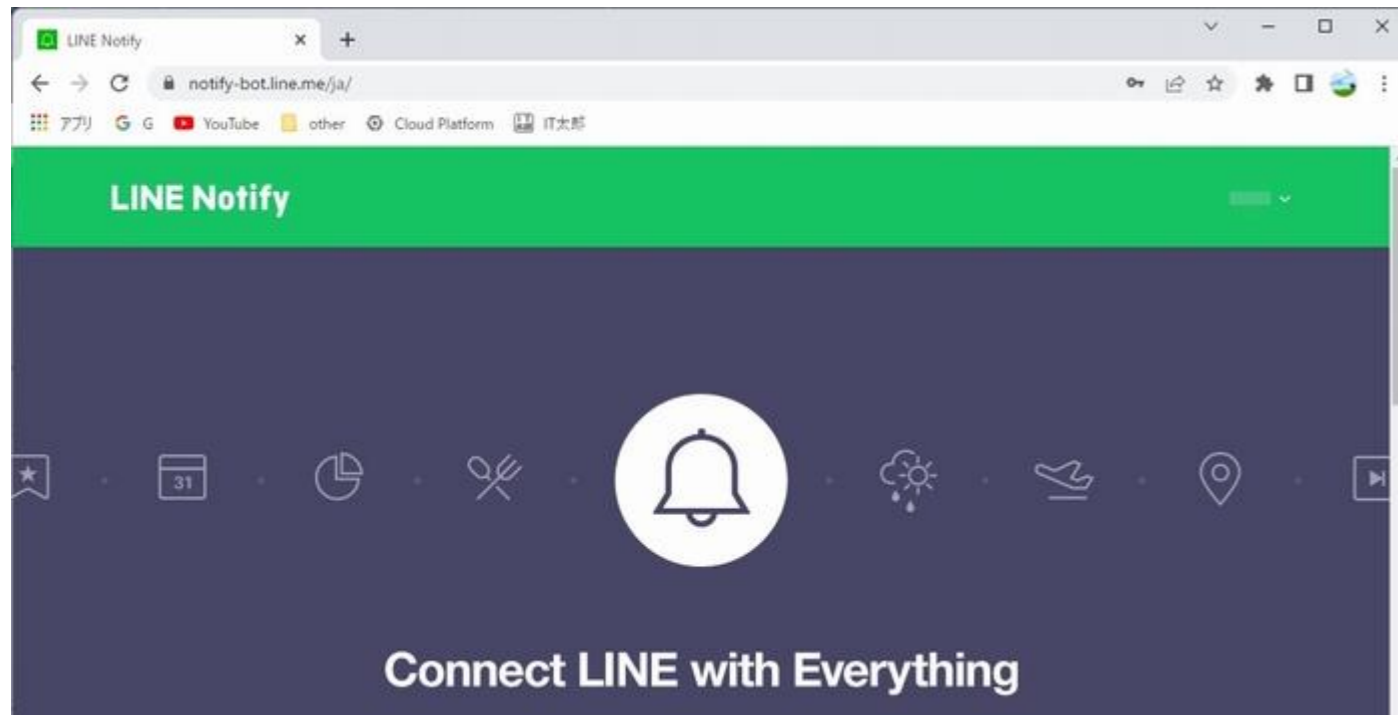


3) 友達追加で「@linenotify」で検索し
追加する



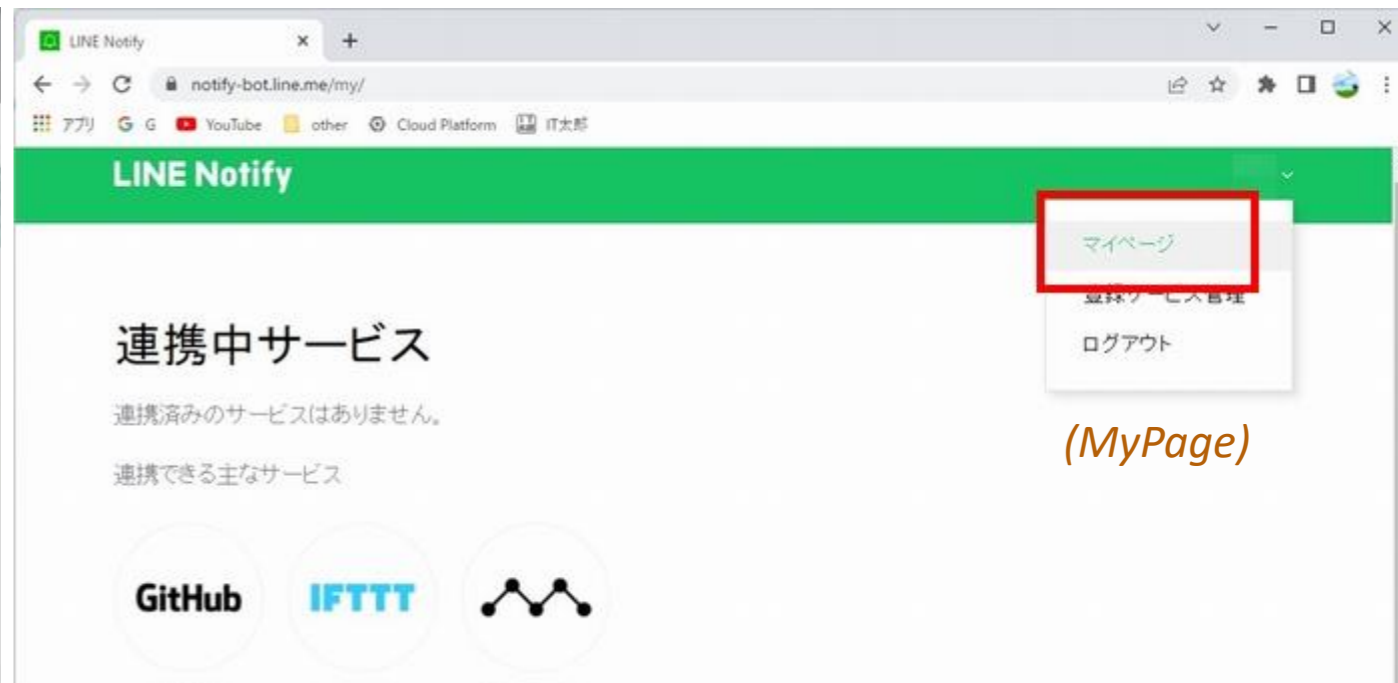
4-2. LINE設定(トークン取得)

4) パソコンでLINE NotifyにアクセスしLINEアカウントでログインする。



<https://notify-bot.line.me/ja/>

5) 右上のメニューから「マイページ」を選択



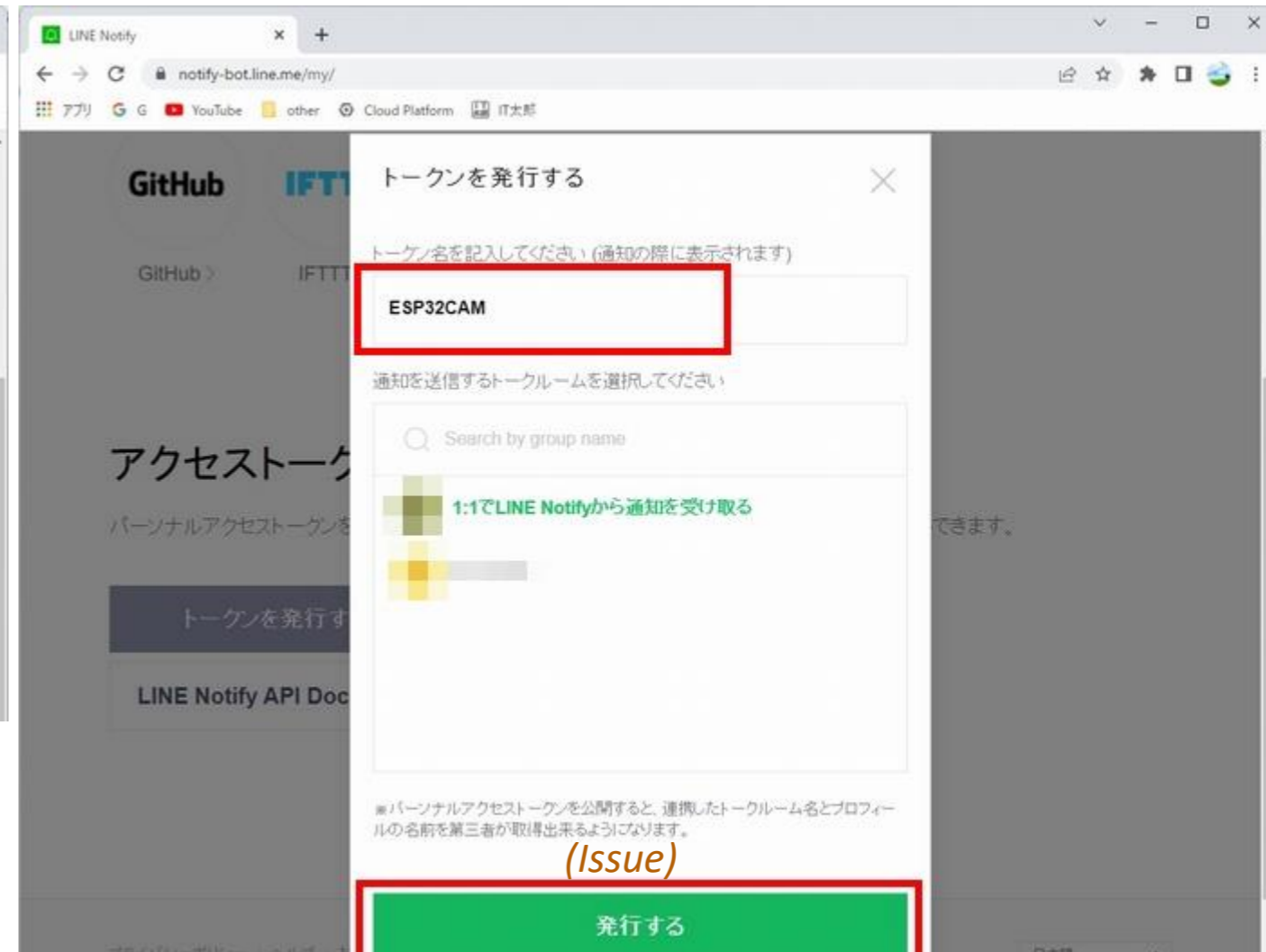
(MyPage)

4-3. LINE設定(トークン取得)

6) トークンを発行します。

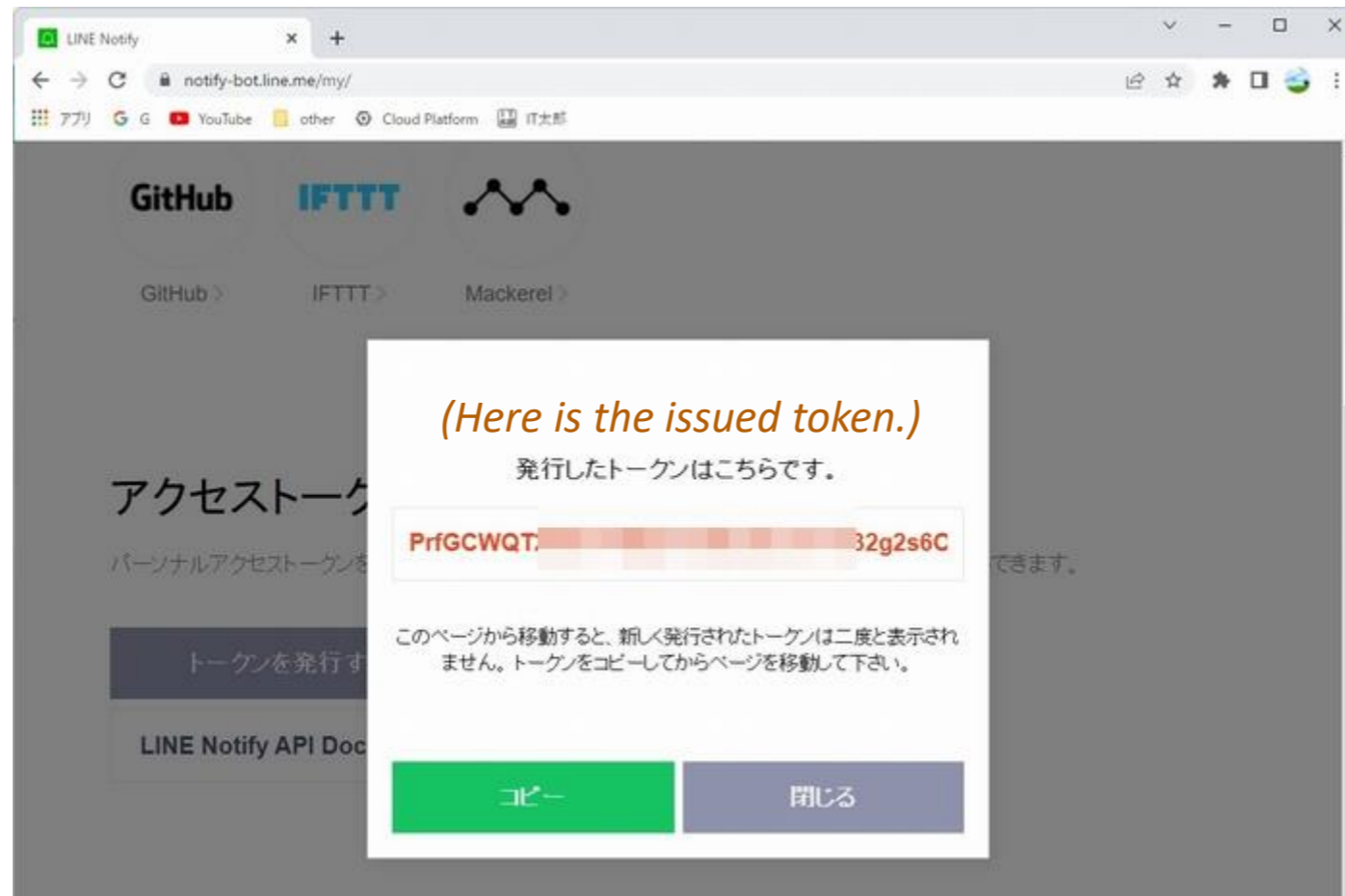


7) トークン名を入力して、発行します。



4-4. LINE設定(トークン取得)

8) 発行したトークンを取得します。(プログラムに記載するため記録します。)



5. Arduinoプログラム(グローバル定義)

```
7 #include <WiFi.h>
8 #include <WiFiClientSecure.h>
9 #include "esp_camera.h"
10
11 // ##### for Battery Use #####
12 //#include "battery.h"
13 //#include "soc/rtc_cntl_reg.h" // for BrouwnoutDetector Disable
14
15 //#define BATTERY_ENABLE
16
17 // ##### Line, Wi-Fi settings (Preferences) #####
18 String lineToken      = "#### TOKEN ####"; // [★change required]
19
20 const char *ssid      = "#### SSID ####"; // [★change required]
21 const char *password  = "### PASSWORD ###"; // [★change required]
22 // #####
23 const char* lineServer = "notify-api.line.me";
24
25 // LED Pin Setting
26 const byte LED_PIN    = 2; // Green LED
27
28 // CAMERA_MODEL_M5_UNIT_CAM
29 #define PWDN_GPIO_NUM    -1
30 #define RESET_GPIO_NUM  15
31 #define XCLK_GPIO_NUM    27
32 #define SIOD_GPIO_NUM    25
33 #define SIOC_GPIO_NUM    23
34
35 #define Y9_GPIO_NUM      19
36 #define Y8_GPIO_NUM      36
37 #define Y7_GPIO_NUM      18
38 #define Y6_GPIO_NUM      39
39 #define Y5_GPIO_NUM       5
40 #define Y4_GPIO_NUM      34
41 #define Y3_GPIO_NUM      35
42 #define Y2_GPIO_NUM      32
43 #define VSYNC_GPIO_NUM   22
44 #define HREF_GPIO_NUM    26
45 #define PCLK_GPIO_NUM    21
46
47 // Global Values
48 WiFiClientSecure httpsClient;
49 bool ledFlag          = true; // LED Control Flag
50 camera_fb_t * fb;
```

ライブラリの読み込み

LINE-Token, Wi-Fi設定

LED ポート設定

カメラ ポート設定

HTTPSクライアント、LED状態フラグ、カメラバッファ
の変数定義

設定変更必要

5. Arduinoプログラム (Setup関数)

```
52 // Setup Function
53 void setup() {
54     Serial.begin(115200);
55
56 // For Battery Use
57 #ifndef BATTERY_ENABLE
58     WRITE_PERI_REG(RTC_CNTL_BROWN_OUT_REG, 0); //disable detector
59     bat_init();
60     bat_hold_output();
61 #endif
62 //Serial.setDebugOutput(true);
63 //Serial.println();
64
65 // Camera Setting
66 camera_config_t config;
67 config.ledc_channel = LEDC_CHANNEL_0;
68 config.ledc_timer = LEDC_TIMER_0;
69 config.pin_d0 = Y2_GPIO_NUM;
70 config.pin_d1 = Y3_GPIO_NUM;
71 config.pin_d2 = Y4_GPIO_NUM;
72 config.pin_d3 = Y5_GPIO_NUM;
73 config.pin_d4 = Y6_GPIO_NUM;
74 config.pin_d5 = Y7_GPIO_NUM;
75 config.pin_d6 = Y8_GPIO_NUM;
76 config.pin_d7 = Y9_GPIO_NUM;
77 config.pin_xclk = XCLK_GPIO_NUM;
78 config.pin_pclk = PCLK_GPIO_NUM;
79 config.pin_vsync = VSYNC_GPIO_NUM;
80 config.pin_href = HREF_GPIO_NUM;
81 config.pin_sscb_sda = SIOD_GPIO_NUM;
82 config.pin_sscb_scl = SIOC_GPIO_NUM;
83 config.pin_pwdn = PWDN_GPIO_NUM;
84 config.pin_reset = RESET_GPIO_NUM;
85 config.xclk_freq_hz = 20000000;
86 config.pixel_format = PIXFORMAT_JPEG;
87 // Image size setting: QVGA(320x240), CIF(400x296), HVGA(480x320), VGA(640x480), SVGA(800x600), XGA(1024x768)
88 config.frame_size = FRAMESIZE_XGA;
89 config.jpeg_quality = 10;
90 config.fb_count = 2;
91
92 // camera init
93 esp_err_t err = esp_camera_init(&config);
94 if (err != ESP_OK) {
95     Serial.printf("Camera init failed with error 0x%x", err);
96     return;
97 }
```

シリアルモニタ開始

バッテリー利用時、BrownOUT（電圧低下）チェックをOFF
[すぐにエラーになり再起動してしまう場合に有効設定へ]

カメラ ポート設定及び初期化

画像サイズ設定
XGA(1024*768)

5. Arduinoプログラム (Setup関数)

```
98 /*sensor_t *s = esp_camera_sensor_get();
99 // initial sensors are flipped vertically and colors are a bit saturated
100 s->set_vflip(s, 1); // flip it back
101 s->set_brightness(s, 1); // up the blightness just a bit
102 s->set_saturation(s, -2); // lower the saturation*/
103
104 // ##### PIN setting start #####
105 pinMode ( LED_PIN, OUTPUT );
106
107 // ##### Wireless Wi-Fi connection #####
108 WiFi.begin ( ssid, password );
109 while ( WiFi.status() != WL_CONNECTED ) { // infinite loop until connected
110     // LED flashes every second while connected
111     ledFlag = !ledFlag;
112     digitalWrite(LED_PIN, ledFlag);
113     delay ( 1000 );
114     Serial.print ( "." );
115 }
116 // Wi-Fi connection completed (IP address display)
117 Serial.print ( "Wi-Fi Connected! IP address: " );
118 Serial.println ( WiFi.localIP() );
119 Serial.println ( );
120 // LED lights when Wi-Fi is connected (Wi-Fi connection status)
121 digitalWrite(LED_PIN, true);
122
123 // ##### HTTPS certificate check setting #####
124 // Skip Server certificate check (required since 1.0.5)
125 httpsClient.setInsecure();//skip verification
126 //httpsClient.setCACert(rootCA);// It is also possible to obtain a root certificate in advance using a web browser and
127
128 // ##### Get Image #####
129 Serial.println("Start get JPG");
130 getCameraJPEG();
131 // ##### Post to LINE CLOUD #####
132 Serial.println("Start Post Line");
133 postLine();
134
135 Serial.println("Line Completed!!!");
136 }
```

LED ポート設定

Wi-Fi接続処理

証明書チェックのスキップ処理
(暗号化は実施されるが、URLに対するサーバの正しさチェックをスキップ)

カメラでJPEG画像の取得処理

LINEへ画像の投稿処理

《参考》

証明書をチェックした場合の利用については、以下の投稿で実施
「屋外操作、AIスピーカ連携スマートリモコン」
<https://hobby-it.com/smartremo7/>

5. Arduinoプログラム (Setup関数)

```
148 // Post image to LINE
149 void postLine() {
150
151     // Connect to LINE Cloud
152     Serial.println("Connect to " + String(lineServer));
153     if (httpsClient.connect(lineServer, 443)) {
154         Serial.println("Connection successful");
155
156         String messageData = "--foo_bar_baz\r\n"
157             "Content-Disposition: form-data; name=\"message\"\r\n\r\n"
158             "ESP32CAM Post\r\n"; // message to display
159         String startBoundry = "--foo_bar_baz\r\n"
160             "Content-Disposition: form-data; name=\"imageFile\"; filename=\"esp32cam.jpg\"\r\nContent-Type: image/jpeg\r\n\r\n";
161         String endBoundry = "\r\n--foo_bar_baz--";
162
163         unsigned long contentsLength = messageData.length() + startBoundry.length() + fb->len + endBoundry.length();
164         String header = "POST /api/notify HTTP/1.0\r\n"
165             "HOST: " + String(lineServer) + "\r\n" +
166             "Connection: close\r\n" +
167             "content-type: multipart/form-data;boundary=foo_bar_baz\r\n" +
168             "content-length: " + String(contentsLength) + "\r\n" +
169             "authorization: Bearer " + lineToken + "\r\n\r\n";
170
171         Serial.println("Send JPEG DATA by API");
172         httpsClient.print(header);
173         httpsClient.print(messageData);
174         httpsClient.print(startBoundry);
175         // JPEG data is separated into 1000 bytes and POSTed
176         unsigned long dataLength = fb->len;
177         uint8_t* bufAddr = fb->buf;
178         for(unsigned long i = 0; i < dataLength ;i=i+1000) {
179             if ( (i + 1000) < dataLength ) {
180                 httpsClient.write(( bufAddr + i ), 1000);
181             } else if (dataLength%1000 != 0) {
182                 httpsClient.write(( bufAddr + i ), dataLength%1000);
183             }
184         }
185         httpsClient.print(endBoundry);
186
187         Serial.println("Waiting for response.");
188         while (httpsClient.connected()) {
189             String line = httpsClient.readStringUntil('\n');
190             if (line == "\r") {
191                 Serial.println("headers received");
192                 break;
193             }
194         }
195     }
196 }
```

サーバへの接続処理 (接続に成功すればIF文内へ)

ヘッダ作成処理

ヘッダ送信処理

カメラ画像送信処理
(1000Byte単位で送信)

画像送信終了のバウンダリー送信

サーバからの応答受信待ち (改行コード待ち)

5. Arduinoプログラム (Setup関数)

```
194 }
195 while (httpsClient.available()) {
196     char c = httpsClient.read();
197     Serial.write(c);
198 }
199 } else {
200     Serial.println("Connected to " + String(lineServer) + " failed.");
201 }
202 httpsClient.stop();
203 Serial.println();
204 Serial.println("Finish httpsClient");
205 }
206
207 // Get JPEG image with OV3660
208 void getCameraJPEG(){
209     fb = esp_camera_fb_get(); // Get JPEG image
210     if (!fb) {
211         Serial.printf("Camera capture failed");
212     }
213     Serial.printf("JPG: %uB ", (uint32_t)(fb->len));
214     Serial.println();
215     // Shooting end processing
216     esp_camera_fb_return(fb);
217 }
```

サーバからの受信内容をシリアルモニタへ表示

サーバ接続失敗時の処理

サーバ接続終了処理

カメラでのJPEG画像取得処理

5. Arduinoプログラム(送信データ)

HTTP(S)のPOSTで送信される

```
POST /api/notify HTTP/1.0
HOST: notify-api.line.me
Connection: close
content-type: multipart/form-data;boundary=foo_bar_baz
content-length: *****LENGTH*****
authorization: Bearer *****TOKEN*****
```

boundaryの設定

```
--foo_bar_baz
Content-Disposition: form-data; name="message"

ESP32CAM Post
```

boundary

投稿されるメッセージ

```
--foo_bar_baz
Content-Disposition: form-data; name="imageFile"; filename="esp32cam.jpg"
Content-Type: image/jpeg
```

```
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX JPEG Data XXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
--foo_bar_baz--
```

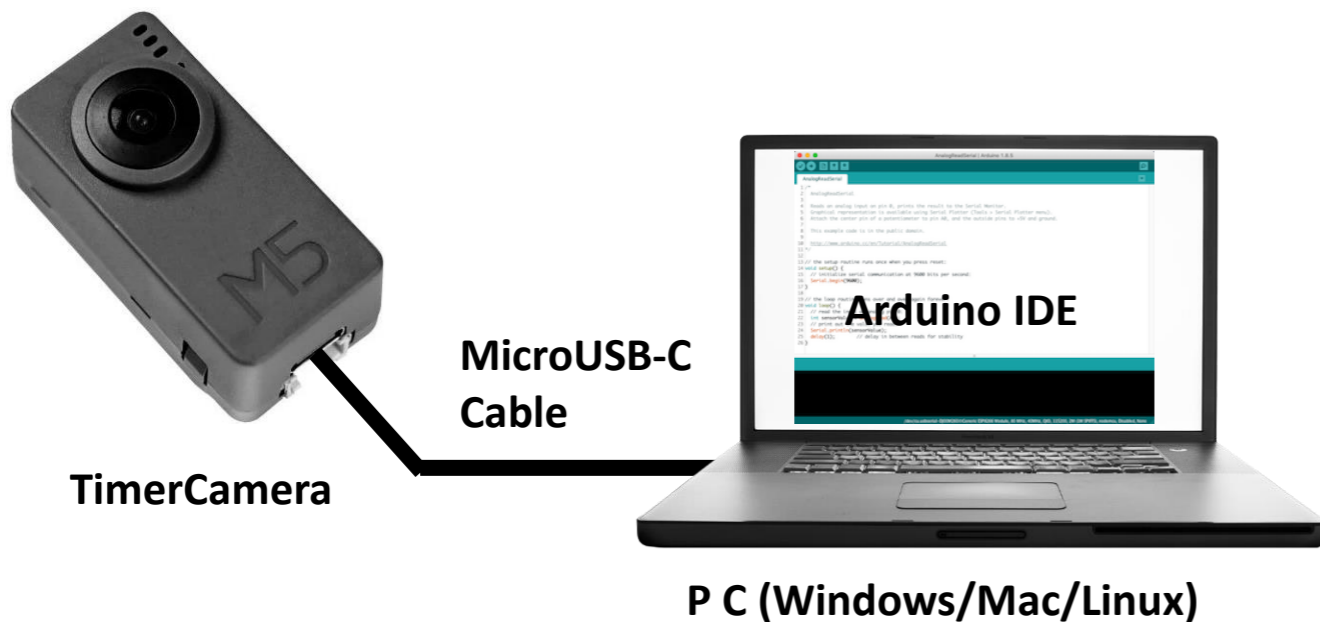
boundary

1000Byteずつ送信

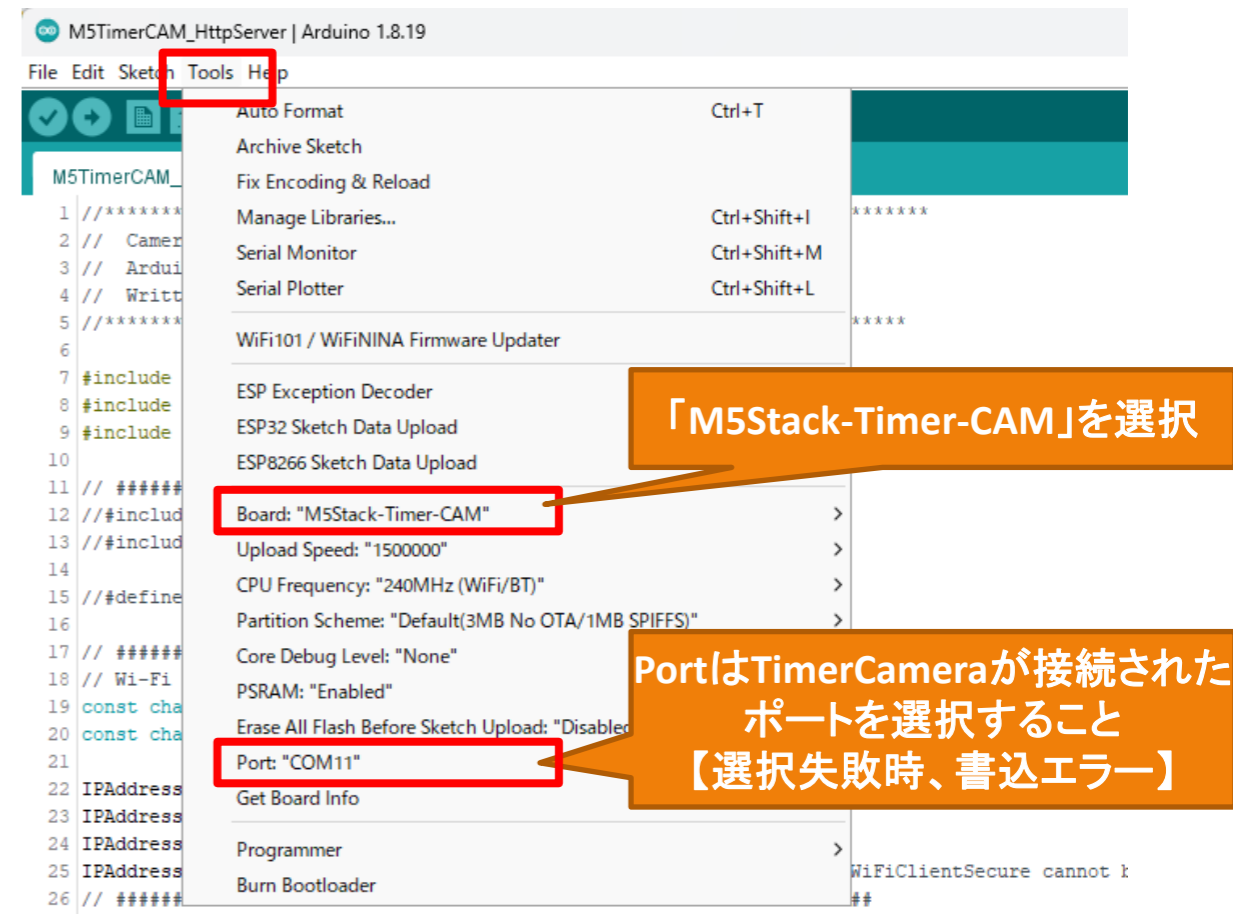
boundary

6-1. プログラム書き込み

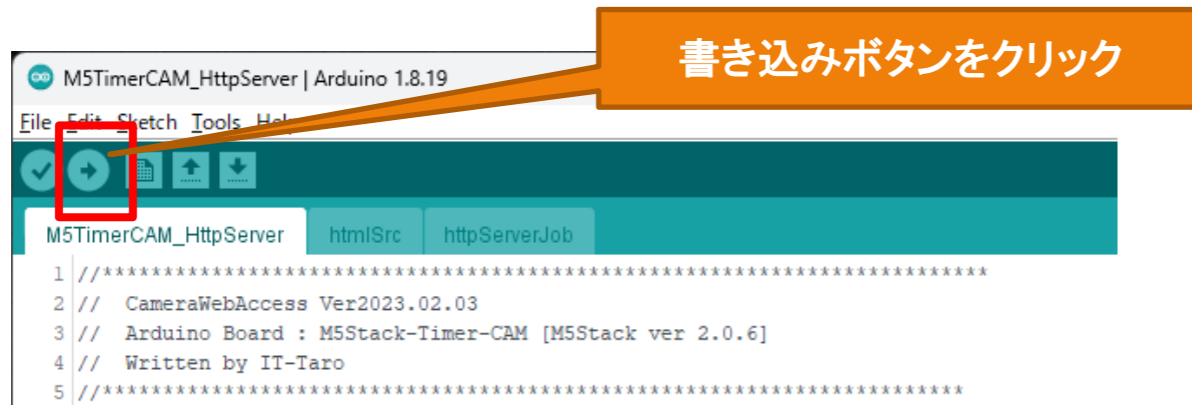
1) TimeCameraをマイクロUSB-Cケーブルで接続



2) ArduinoIDEでプログラムを開き、再度、設定確認。
(プログラムでWi-Fi設定[SSID、IPアドレスなど]は変更しておく。)

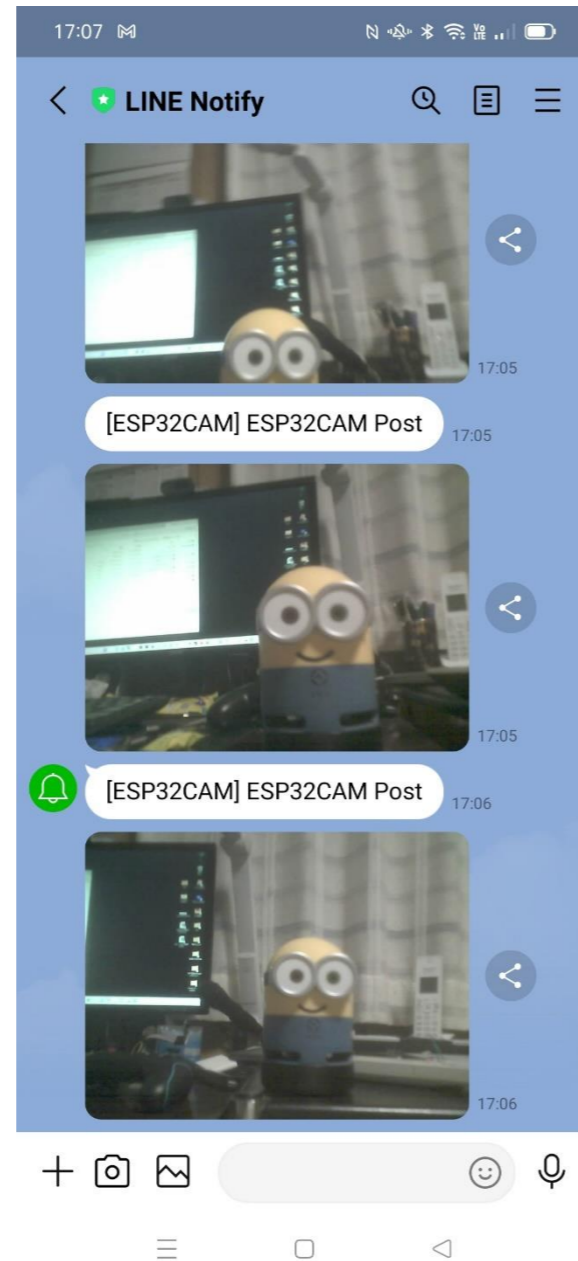


3) 書き込みボタンをクリック



6-2. 動作確認

TimerCameraが起動時に画像を取得し、LINEへ投稿します。



前々回の起動時に投稿

前回の起動時に投稿

TimerCameraによりLINEへ
投稿されたメッセージと画像画像