

品番：UNEKIT06-OLEDANIMATION

品名：OLED パラパラアニメ表示キット

■パラパラ画像を変える方法

白黒の PNG ファイルを作るツール：Piskel

<https://www.piskelapp.com/>

PNG ファイルを C 言語ソースに変換するツール：Image2cpp

<https://javl.github.io/image2cpp/>

開発ツール（コンパイル、書き込み）：Arduino IDE

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

「キーワード」で調べて、Arduino IDE を設定する。

ファイル→基本設定→追加のボードマネージャの URL：

https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json

ボード XIAO ESP32C3

ボードマネージャ esp32 by Espressif Systems 2.0.11

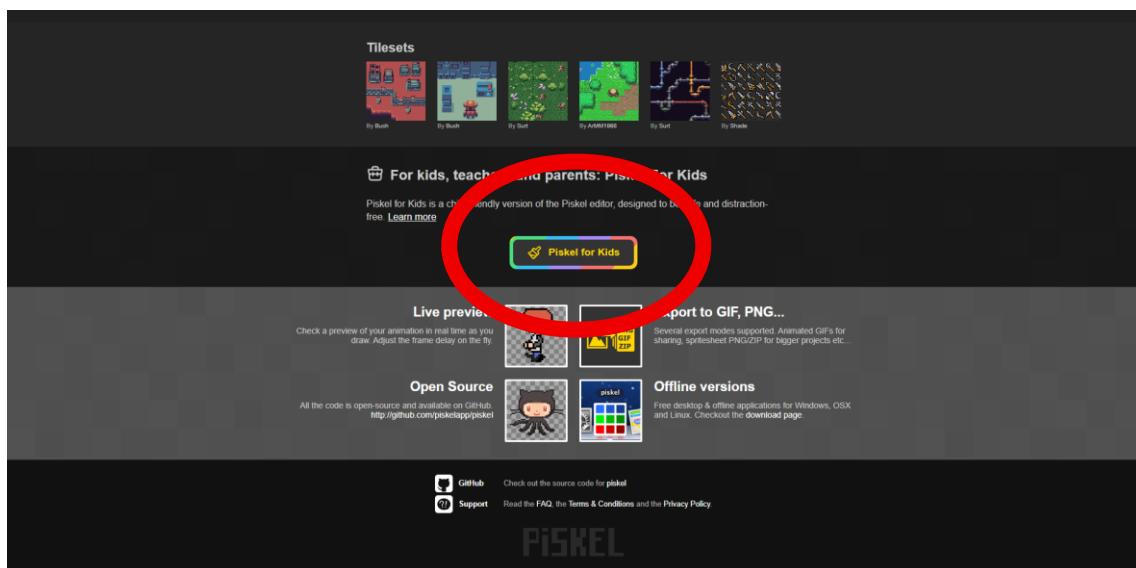
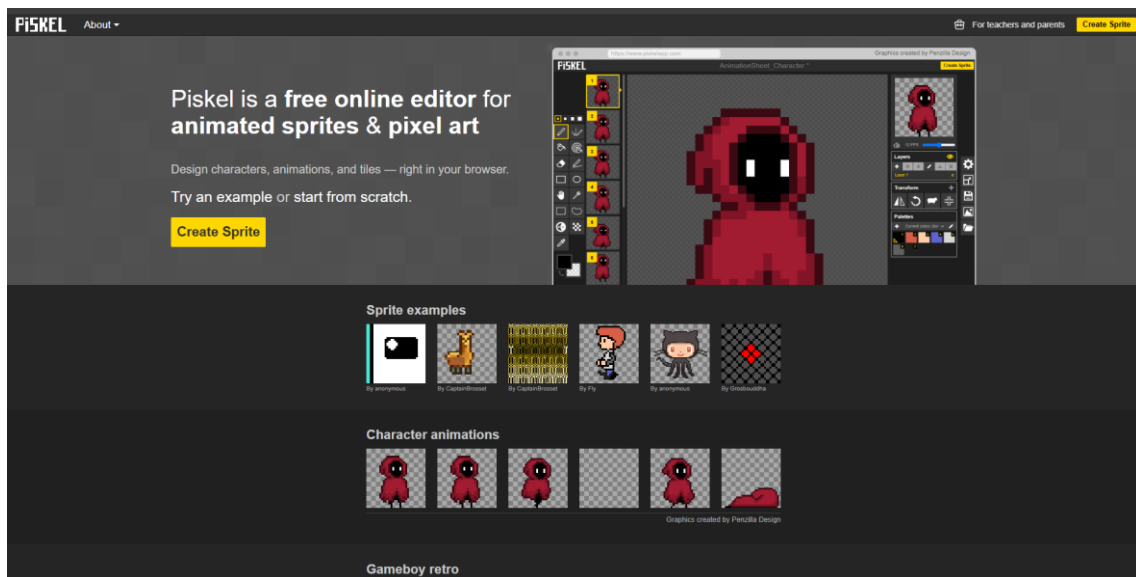
ライブラリ Adafruit SSD1306 by Adafruit 2.5.15

Piskel : 白黒の PNG ファイルを作るツール

<https://www.piskelapp.com/>

画面下の方の **Piskel for Kids** を使う

他の Create Sprite や、サンプルのアイコンは使わないこと。(不適切な CM が入る)



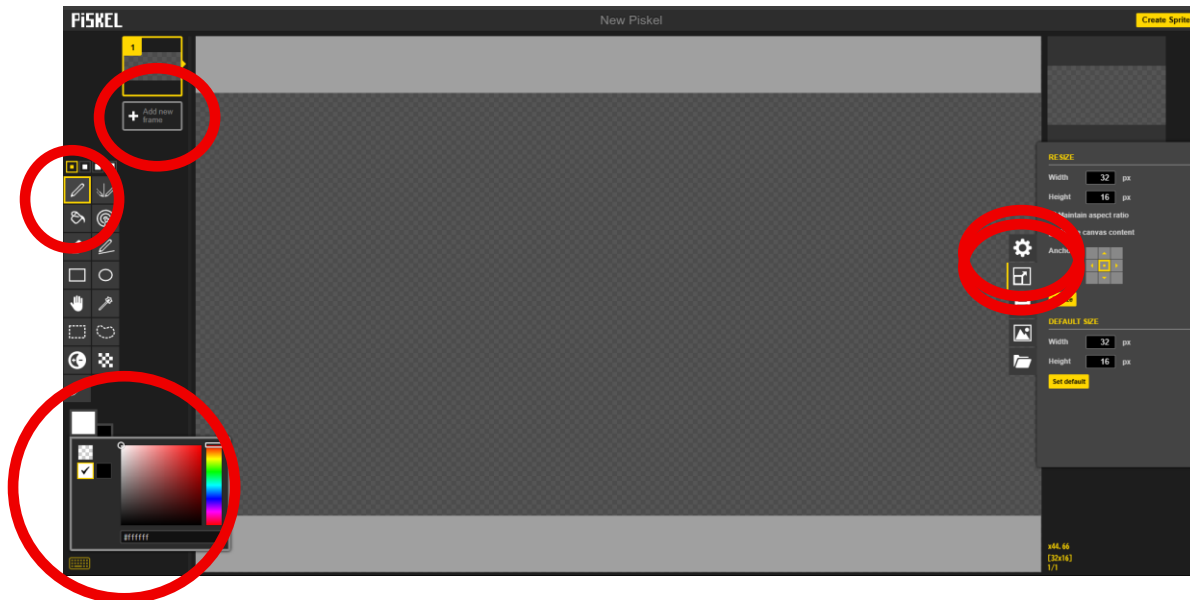
PREFERENCES SETTINGS の Grid で Enable grid を ☒ にする

RISIZE で、Width=32 , Height=16 に設定

描画色を、黒は黒のまま、透明を白(0xffffffff)に変更

塗りつぶしツールで背景を黒に、ペンツールを使い白でアニメを描く

Add New Frame で 8 枚に増やす



EXPORT で PNG ファイルを Download で書き出す。

保存先は (例) Download ¥Parapara_animation フォルダにする。

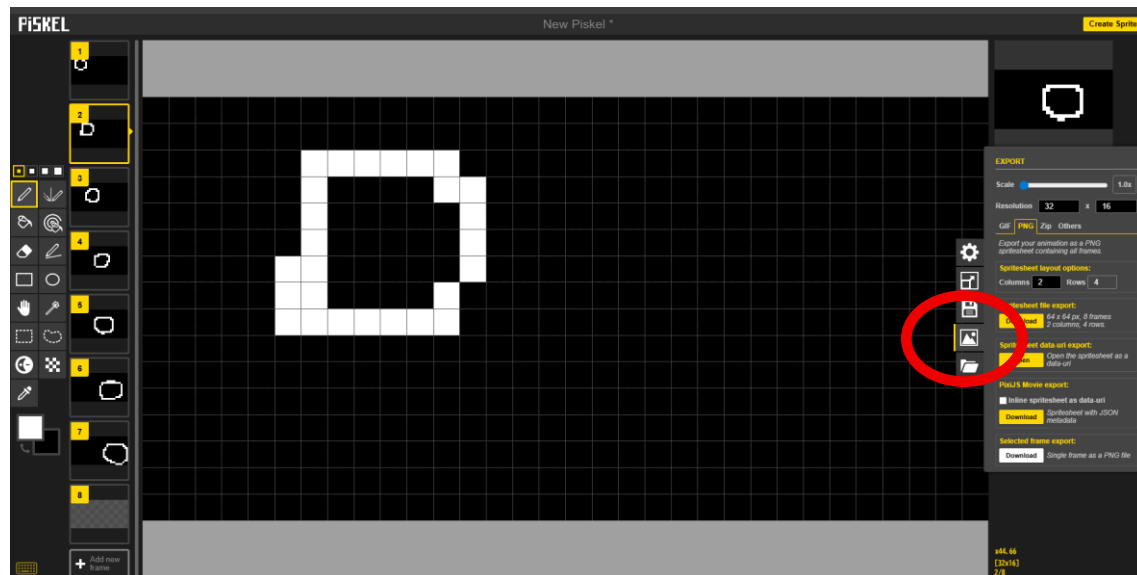


Image2cpp : PNG→ヘッダーファイル変換ツール

<https://javl.github.io/image2cpp/>

frameX.h の作り方 (例)

ファイル選択で (例) Download¥Parapara_animation フォルダから 8 枚のアニメ PNG ファイルを(0)1 から順番に(7)8 まで選択する。

image2cpp

image2cpp is a simple tool to change images into byte arrays (or arrays back into images) for use with (monochrome) displays such as OLEDs on your Arduino or Raspberry Pi. It was originally made to work with the Adafruit OLED library (for which you can find an example sketch for Arduino [here](#)) but has been expanded by the community to be useful in all kind of (embedded) projects.

More info (and credits) can be found in the [Github repository](#). This is also where you can report any [issues](#) you might come across.

Did you find this tool useful? Feel free to support my open source software on Github, especially if used commercially.

1. Select image

All images are loaded locally in your browser, and images are not uploaded or stored anywhere online.

ファイル選択

frame0.png

frame1.png

frame2.png

frame3.png

frame4.png

frame5.png

frame6.png

frame7.png

remove

remove

remove

remove

remove

remove

remove

or

1. Paste byte array

128 x 64 px

Read as horizontal

Read as vertical

Read images appear at step 3 below

2. Image Settings

Canvas size(s):

frame0.png (file resolution: 32 x 16)

32 x 16 glyph

remove

frame1.png (file resolution: 32 x 16)

32 x 16 glyph

remove

2. Image Settings

Canvas size(s):

frame0.png (file resolution: 32 x 16)

32 x 16 glyph

remove

frame1.png (file resolution: 32 x 16)

32 x 16 glyph

remove

frame2.png (file resolution: 32 x 16)

32 x 16 glyph

remove

frame3.png (file resolution: 32 x 16)

32 x 16 glyph

remove

frame4.png (file resolution: 32 x 16)

32 x 16 glyph

remove

frame5.png (file resolution: 32 x 16)

32 x 16 glyph

remove

frame6.png (file resolution: 32 x 16)

32 x 16 glyph

remove

frame7.png (file resolution: 32 x 16)

32 x 16 glyph

remove

Apply first image size to all images

Background color:

☒ White ☐ Black ☐ Transparent

Invert image colors

☐

Dithering:

Binary

Brightness / alpha threshold:

128

0 - 255, if the brightness of a pixel is above the given level the pixel becomes white, otherwise they become black. When using alpha, opaque and transparent are used instead.

Scaling:

original size

Center image:

☐ horizontally ☐ vertically

Centering the image only works when using a canvas larger than the original image.

Rotate image:

0 degrees

Flip image:

☐ horizontally ☐ vertically

3. Preview



Generate code クリックで、ヘッダーファイルが生成される。

3. Preview



4. Output

Code output format

Arduino code

Adds some extra Arduino code around the output for easy copy-paste into [this example](#). If multiple images are loaded, generates a byte array for each and appends a counter to the identifier.

Identifier/Prefix:

Draw mode: Horizontal - 1 bit per pixel

If your image looks all mossed up on your display, like the image below, try using a different mode.



☐ swap

Useful when working with the u8g2 library.

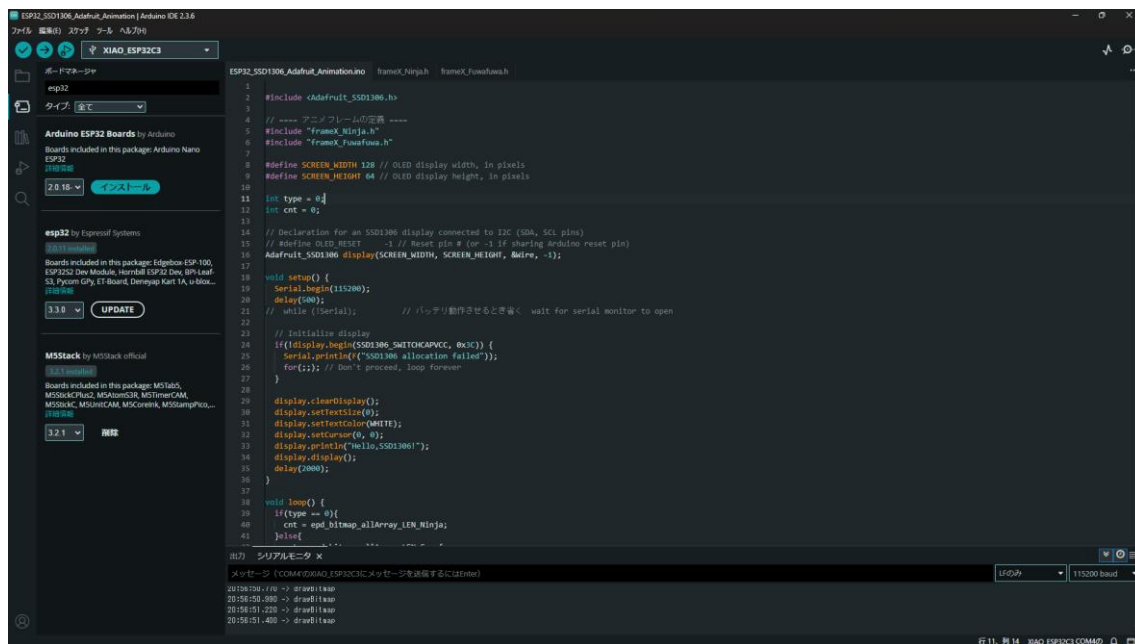
Generate code **Copy** Output Downloaded as binary file (.bin)

Copy Output クリックで、ヘッダーファイルの内容がクリップボードにコピーされる。
 Arduino IDE 開発用の新規タブをファイルを.h 拡張子で作り、貼り付ける。

[illegible]

開発ツール（コンパイル、書き込み）：Arduino IDE

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



Arduino の使い方をネットで調べて、自分でもわかるページを参考にインストールする。
バージョン：2.3.6

以降で、

「キーワード」で調べて、Arduino IDE を下記のように設定する。

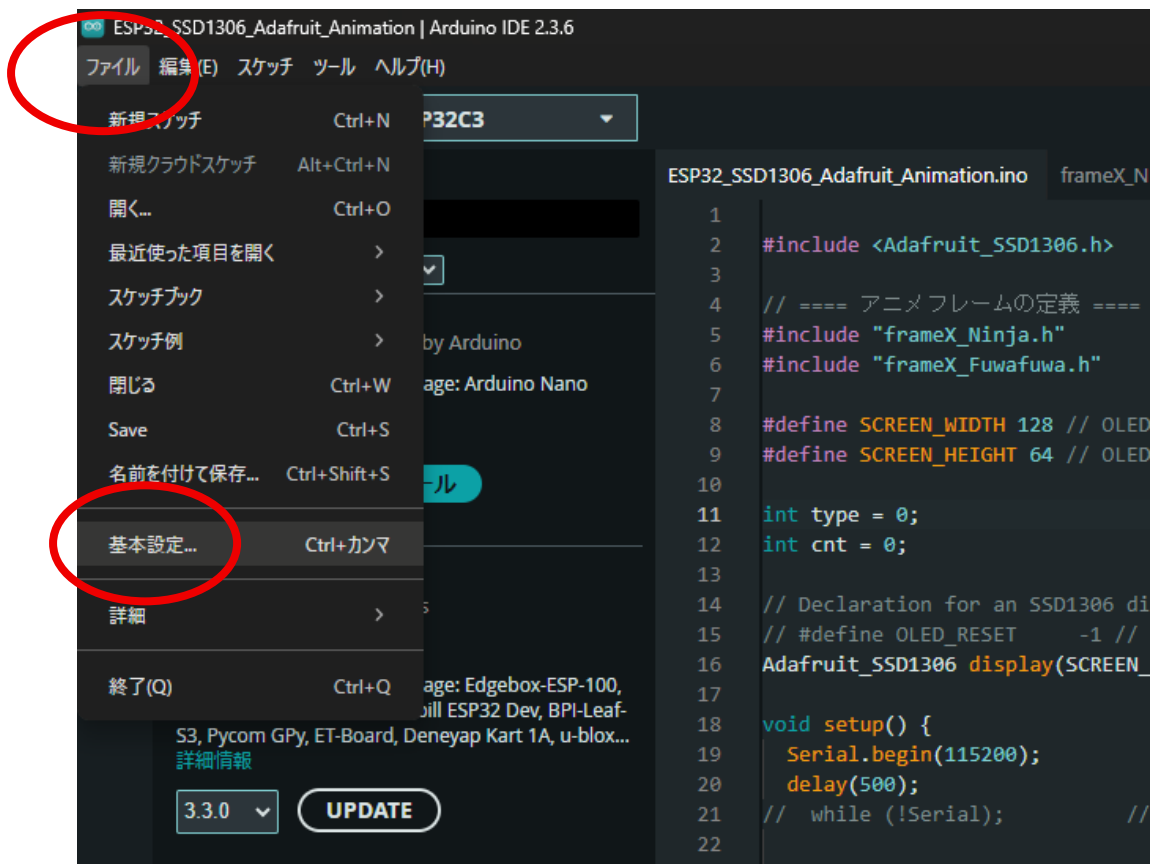
ファイル→基本設定→追加のボードマネージャの URL：

https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json

ボード XIAO ESP32C3

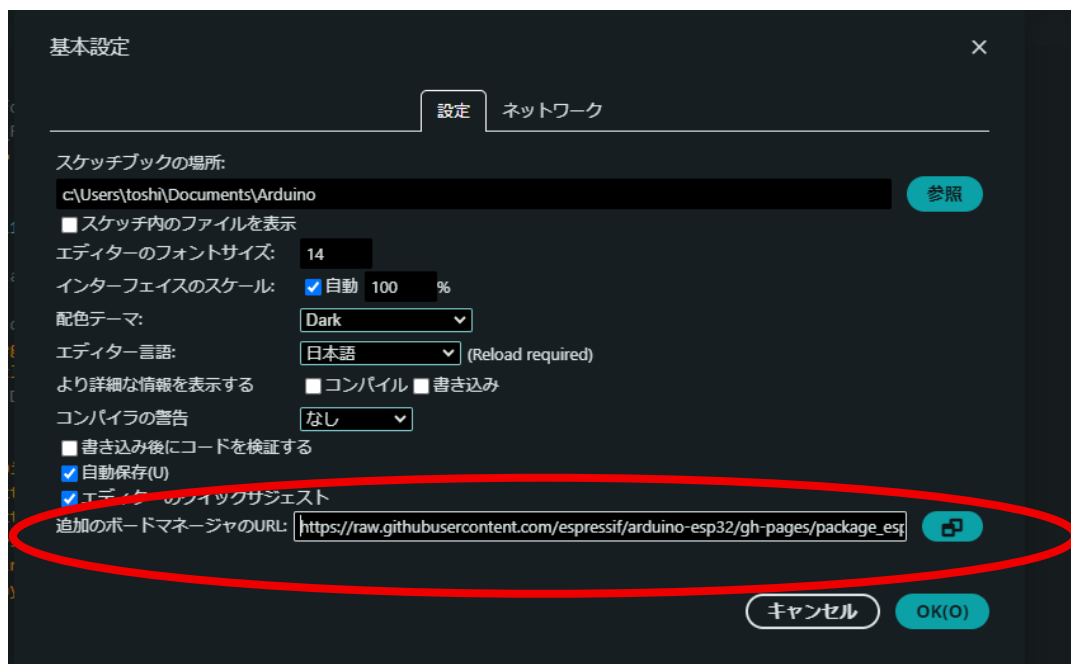
ボードマネージャ esp32 by Espressif Systems 2.0.11

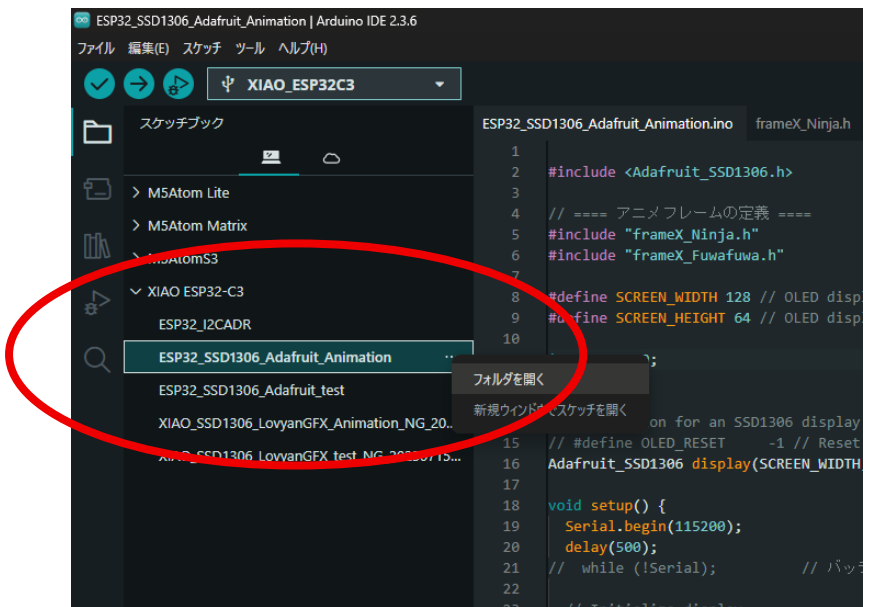
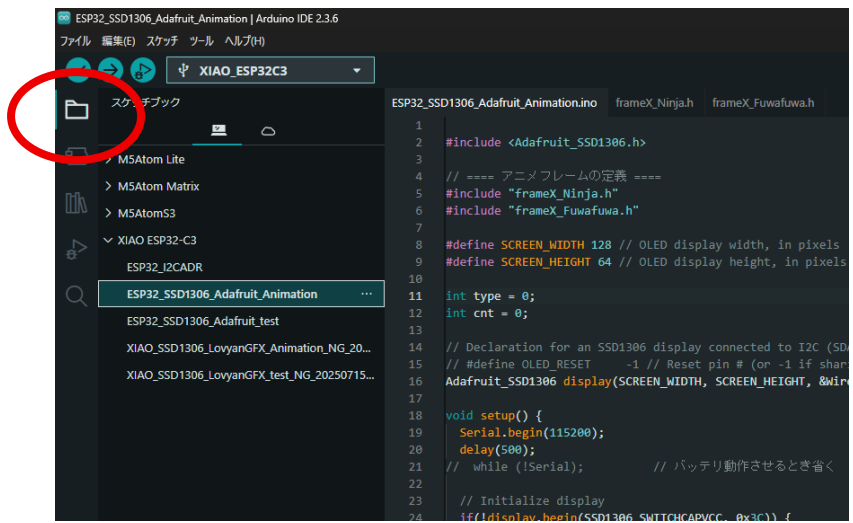
ライブラリ Adafruit SSD1306 by Adafruit 2.5.15



ファイル→基本設定→追加のボードマネージャの URL :

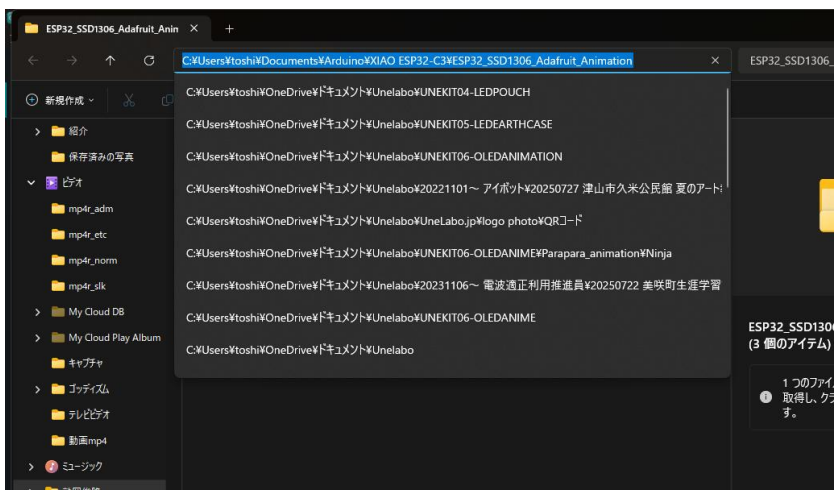
https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json

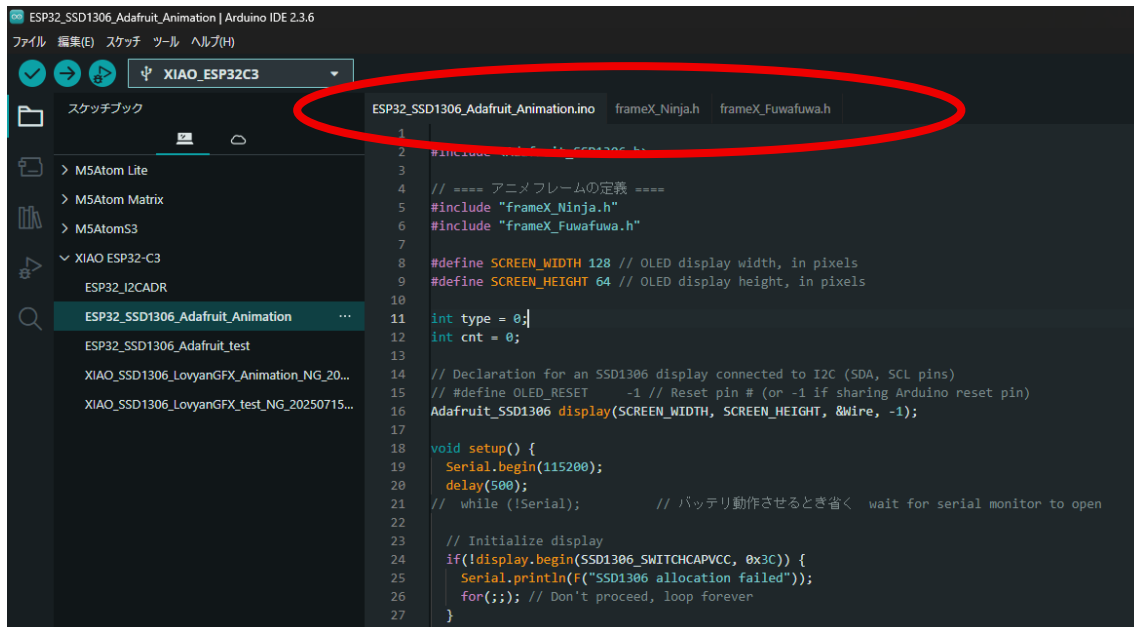




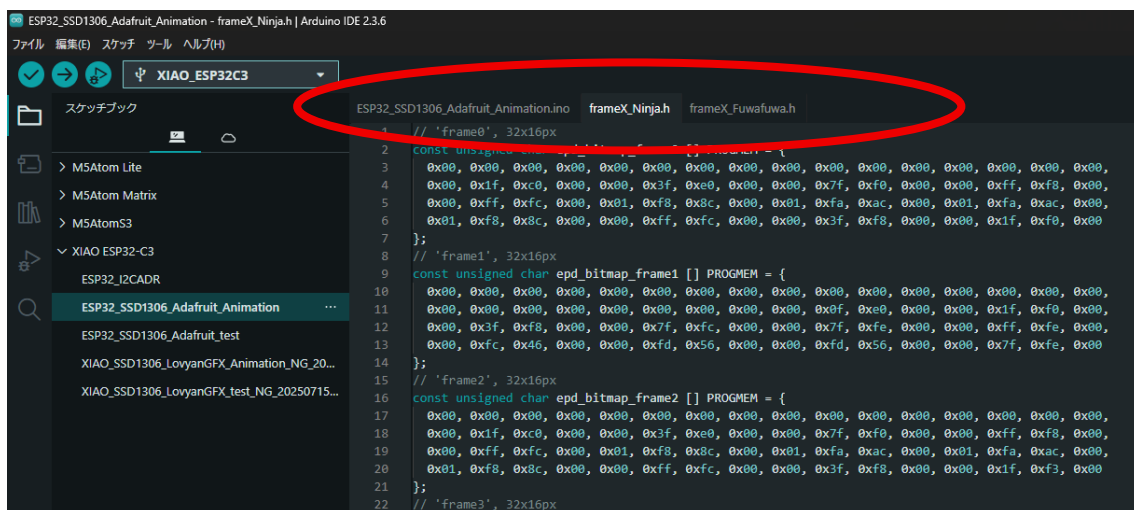
C:\Users\toshi\Documents\Arduino\XIAO
C3\ESP32_SSD1306_Adafruit_Animation

ESP32-

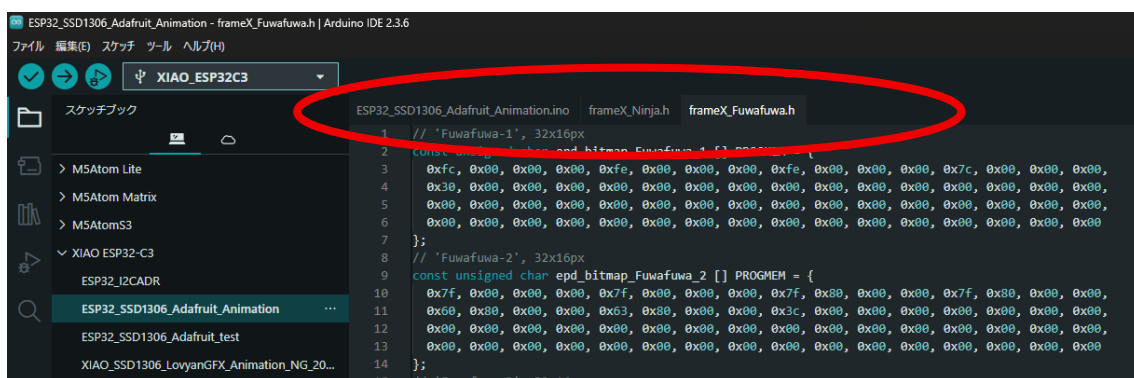




```
1 // ESP32_SSD1306_Adafruit_Animation.ino
2 #include "Adafruit_SSD1306.h"
3
4 // ==== アニメフレームの定義 ====
5 #include "frameX_Ninja.h"
6 #include "frameX_FuwaFuwa.h"
7
8 #define SCREEN_WIDTH 128 // OLED display width, in pixels
9 #define SCREEN_HEIGHT 64 // OLED display height, in pixels
10
11 int type = 0;
12 int cnt = 0;
13
14 // Declaration for an SSD1306 display connected to I2C (SDA, SCL pins)
15 // #define OLED_RESET -1 // Reset pin # (or -1 if sharing Arduino reset pin)
16 Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1);
17
18 void setup() {
19   Serial.begin(115200);
20   delay(500);
21   // while (!Serial); // バッテリー動作させるとき省く wait for serial monitor to open
22
23   // Initialize display
24   if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) {
25     Serial.println(F("SSD1306 allocation failed"));
26     for(;;); // Don't proceed, loop forever
27   }
28 }
```

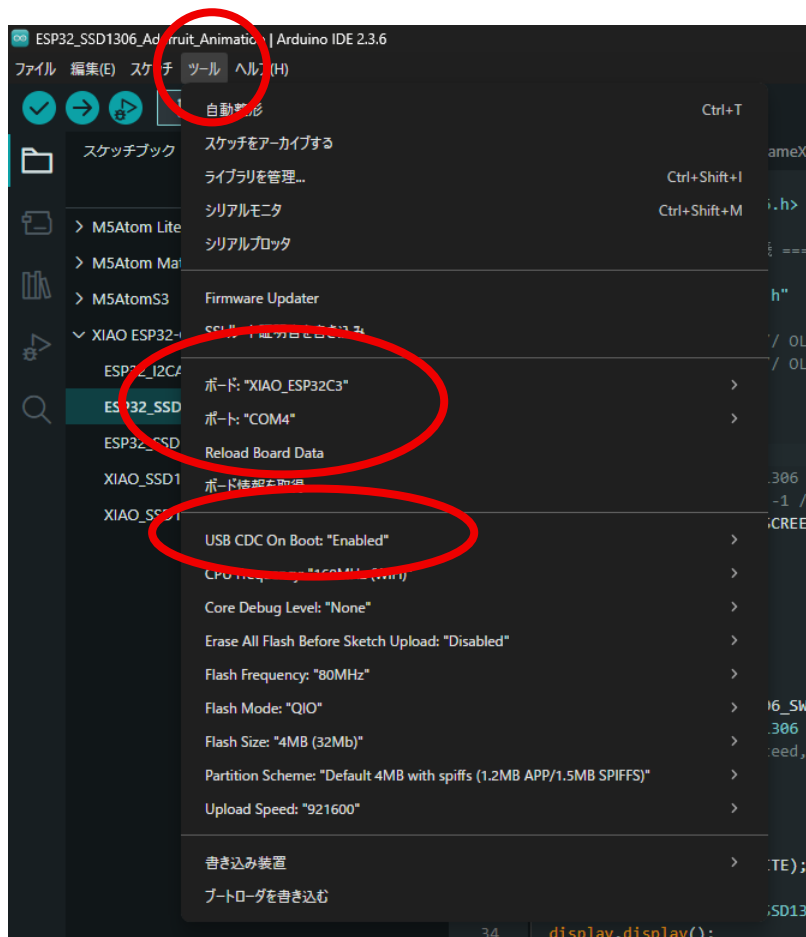
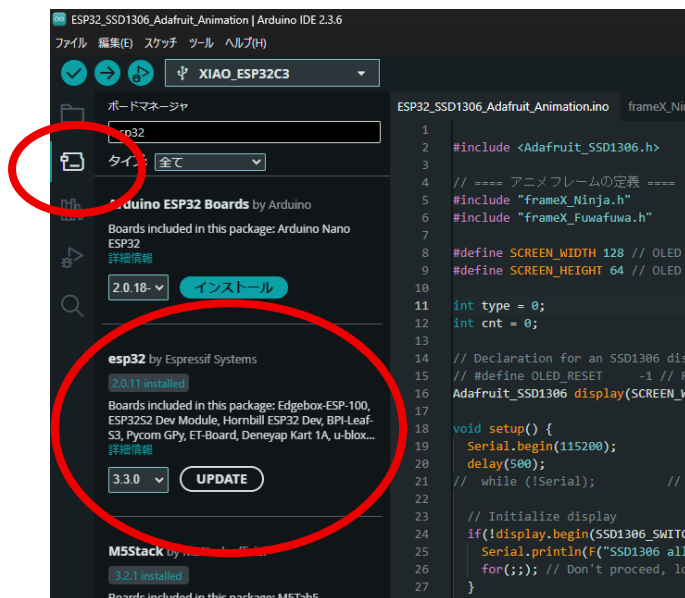


```
1 // 'frame0', 32x16px
2 const unsigned char epd_bitmap_frame0 [] PROGMEM = {
3   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
4   0x00, 0x1f, 0xc0, 0x00, 0x00, 0x00, 0x3f, 0xe0, 0x00, 0x00, 0x7f, 0xf0, 0x00, 0x00, 0xff, 0xf8, 0x00,
5   0x00, 0xff, 0xfc, 0x00, 0x01, 0xf8, 0x8c, 0x00, 0x01, 0xfa, 0xac, 0x00, 0x01, 0xfa, 0xac, 0x00,
6   0x01, 0xf8, 0x8c, 0x00, 0x00, 0xff, 0xfc, 0x00, 0x00, 0x3f, 0xf8, 0x00, 0x00, 0x1f, 0xf0, 0x00
7 };
8 // 'frame1', 32x16px
9 const unsigned char epd_bitmap_frame1 [] PROGMEM = {
10  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
11  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x0f, 0xe0, 0x00, 0x00, 0x00, 0x1f, 0xf0, 0x00,
12  0x00, 0x3f, 0xf8, 0x00, 0x00, 0x7f, 0xfc, 0x00, 0x00, 0x7f, 0xfe, 0x00, 0x00, 0xff, 0xfe, 0x00,
13  0x00, 0xfc, 0x46, 0x00, 0x00, 0xfd, 0x56, 0x00, 0x00, 0xfd, 0x56, 0x00, 0x00, 0x7f, 0xfe, 0x00
14 };
15 // 'frame2', 32x16px
16 const unsigned char epd_bitmap_frame2 [] PROGMEM = {
17  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
18  0x00, 0x1f, 0xc0, 0x00, 0x00, 0x00, 0x3f, 0xe0, 0x00, 0x00, 0x7f, 0xf0, 0x00, 0x00, 0xff, 0xf8, 0x00,
19  0x00, 0xff, 0xfc, 0x00, 0x01, 0xf8, 0x8c, 0x00, 0x01, 0xfa, 0xac, 0x00, 0x01, 0xfa, 0xac, 0x00,
20  0x01, 0xf8, 0x8c, 0x00, 0x00, 0xff, 0xfc, 0x00, 0x00, 0x3f, 0xf8, 0x00, 0x00, 0x1f, 0xf0, 0x00
21 };
22 // 'frame3', 32x16px
```



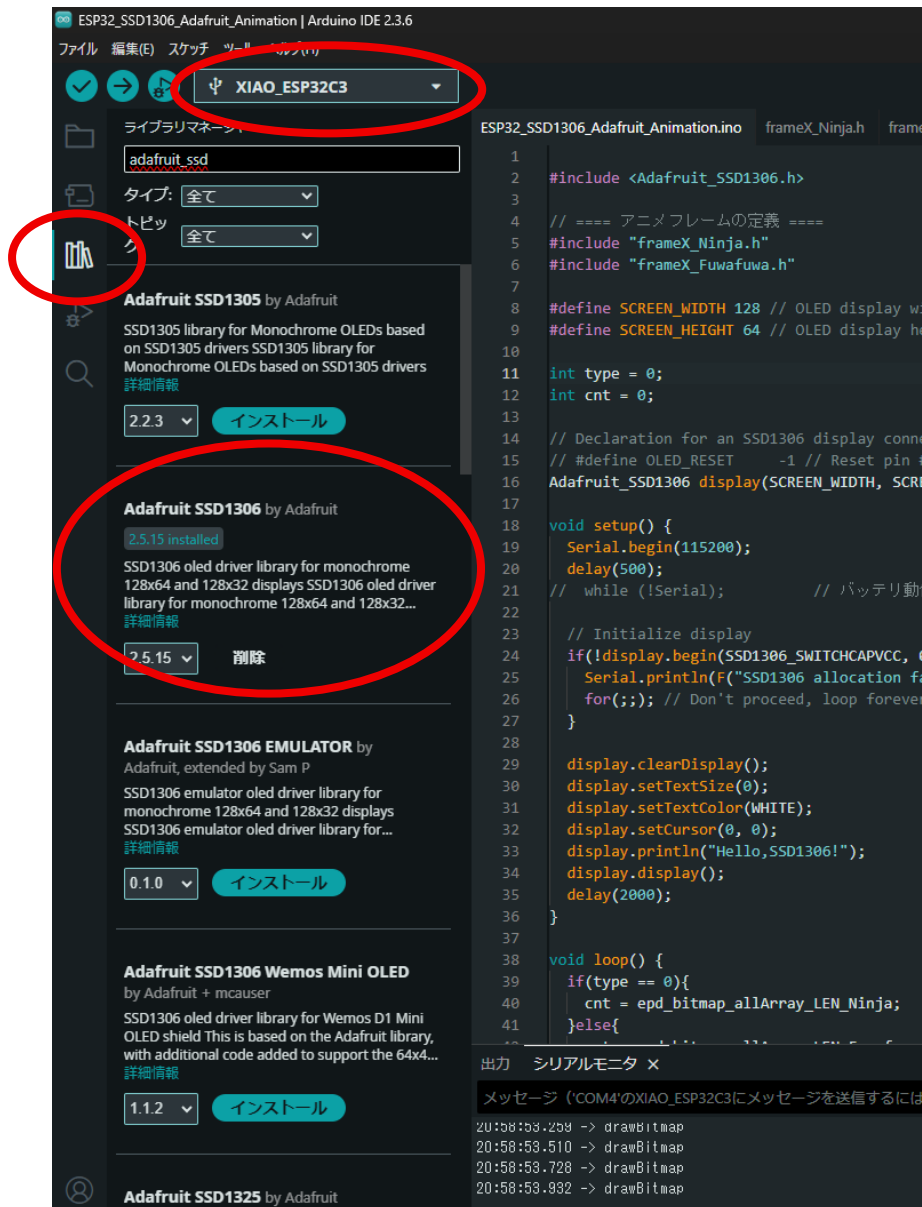
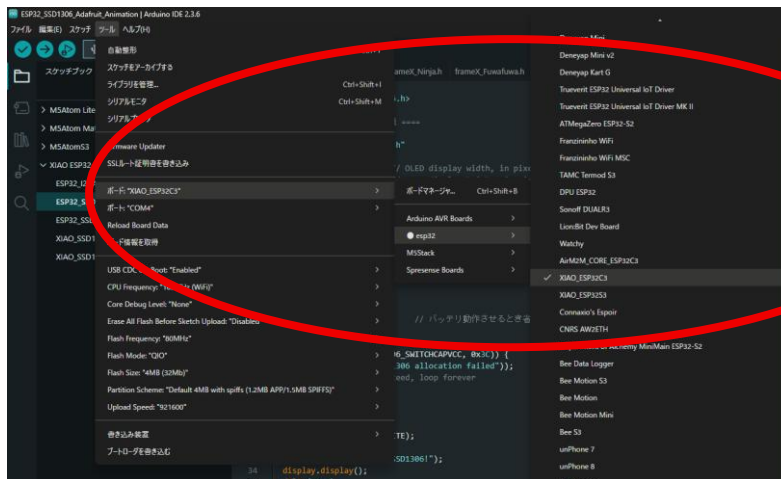
```
1 // 'FuwaFuwa-1', 32x16px
2 const unsigned char epd_bitmap_FuwaFuwa_1 [] PROGMEM = {
3   0xfc, 0x00, 0x00, 0x00, 0xfe, 0x00, 0x00, 0xfe, 0x00, 0x00, 0x7c, 0x00, 0x00, 0x00, 0x00, 0x00,
4   0x30, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
5   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
6   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
7 };
8 // 'FuwaFuwa-2', 32x16px
9 const unsigned char epd_bitmap_FuwaFuwa_2 [] PROGMEM = {
10  0x7f, 0x00, 0x00, 0x00, 0x7f, 0x00, 0x00, 0x7f, 0x00, 0x00, 0x7f, 0x00, 0x00, 0x00, 0x00, 0x00,
11  0x60, 0x00, 0x00, 0x00, 0x63, 0x00, 0x00, 0x3c, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
12  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
13  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
14 };
15 // 'FuwaFuwa-3', 32x16px
```

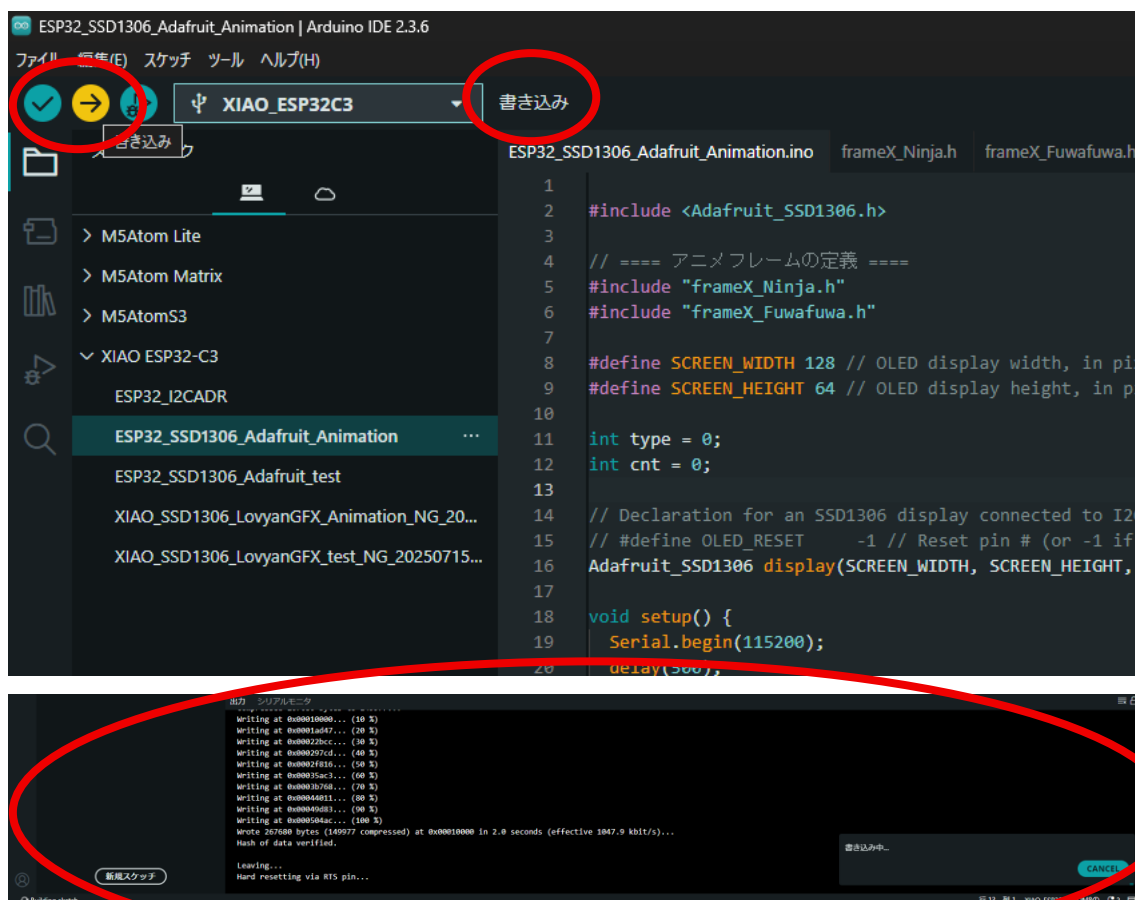
パラパラの画像を増やすには、このフォルダのタブに ○○.h ファイルを追加し、ソースコードを修正する。2枚のままで画像を変えるには、.h ファイルの中身を上書きする。



XIAO ESP32C3 を USB 接続して、COM ポート N に接続されるように COM ポートを選択する。

USB CDC On Boot “Enabled” にする。





以上、パラパラアニメの枚数を増やしたり、画像を差替えるための、Arduino IDE の設定、コンパイル、書き込み、の概要です。

基本的に、自力で調べていただくことを前提にしています。最近は、ネットに多く情報が出ていますし、Chat GPT 等でも教えてくれます。

初心者の方で、レクチャー希望の方は、個別にお問い合わせ、ご相談ください。

連絡先： 畝岡（うねおか）

WE B : <https://unelabo.jp>

