



# Welcome to the JCZN Workshop!

## .....Table of contents.....

一、 Introduction.....2

二、 Installing using Arduino IDE.....2

三、 sample program usage.....11



# Getting Started

## Introduction

The objective of this post is to explain how to upload an Arduino program to the ESP32-2424S012 module, from JCZN .

<http://www.jczn1688.com/zlxz>

The ESP32-C3-MINI-1U WiFi and Bluetooth chip is the latest generation of Espressif products. It has a Single Core 32-bit MCU, which Supporting IEEE 802.11b/g/n (2.4 GHz WiFi) and Bluetooth® 5 (LE). ESP32-C3-MINI-1U series of SoCs is an ultra-low-power and highly-integrated MCU-based solution that supports 2.4 GHz Wi-Fi and Bluetooth ® Low Energy (Bluetooth LE). The block diagram of ESP32-C3-MINI-1U is shown below. 32-bit RISC-V single-core processor with a four-stage pipeline that operates at up to 160MHz. Storage capacities ensured by 400 KB of SRAM (16 KB for cache) and 384 KB of ROM on the chip, and SPI, Dual SPI, Quad SPI, and QPI interfaces that allow connection to external flash.

## Installing using Arduino IDE

Programming the ESP32-C3-MINI-1U

An easy way to get started is by using the familiar Arduino IDE. While this is not necessarily the best environment for working with the ESP32-C3-MINI-1U, it has the advantage of being a familiar application, so the learning curve is flattened.

We will be using the Arduino IDE for our experiments.

### 1, Installing using Arduino IDE

we first need to install version 1.8.19 of the Arduino IDE (or greater), for example, the Arduino installation was in "C:/Programs(x86)/Arduino".

download release link:

<https://downloads.arduino.cc/arduino-1.8.19-windows.exe>

### 2, This is the way to install Arduino-ESP32 directly from the Arduino IDE.

Add Boards Manager Entry

Here is what you need to do to install the ESP32 boards into the Arduino IDE:

- (1) Open the Arduino IDE.



```
#define LGFX_USE_V1
#include "Arduino.h"
#include <lvgl.h>
#include "demos/lv_demos.h"
#include <LovyanGFX.hpp>
#include <Ticker.h>
#include "CST816D.h"
#include "do_mian.h"
#define I2C_SDA 4
#define I2C_SCL 5
#define TP_INT 0
#define TP_RST 1

#define off_pin 35
#define buf_size 120

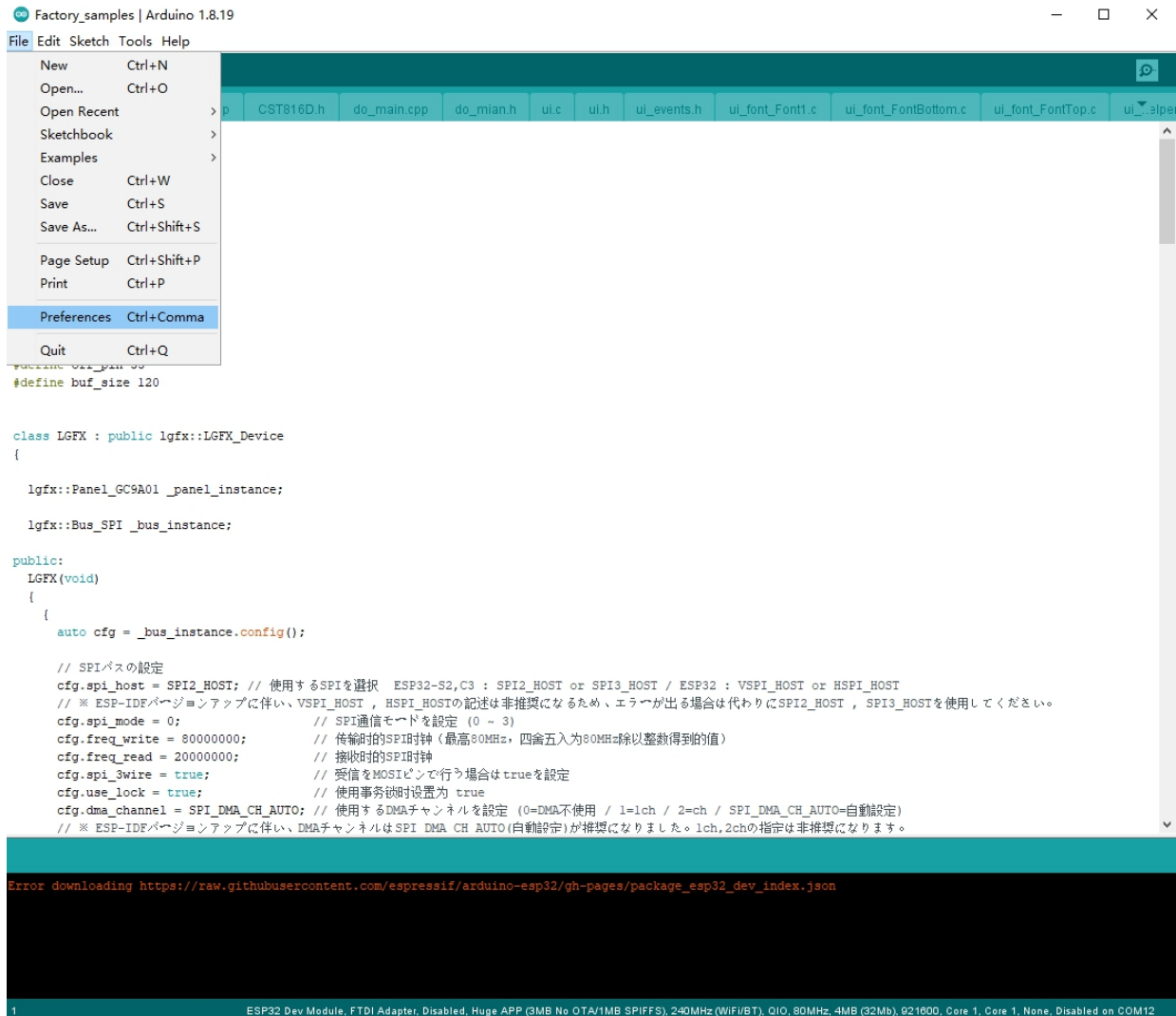
class LGFX : public lgfx::LGFX_Device
{
    lgfx::Panel_GC9A01 _panel_instance;

    lgfx::Bus_SPI _bus_instance;

public:
    LGFX(void)
    {
        {
            auto cfg = _bus_instance.config();

            // SPIバスの設定
            cfg.spi_host = SPI2_HOST; // 使用するSPIを選択 ESP32-S2,C3 : SPI2_HOST or SPI3_HOST / ESP32 : VSPi_HOST or HSPI_HOST
            // ※ ESP-IDFバージョンアップに伴い、VSPi_HOST , HSPI_HOSTの記述は非推奨になるため、エラーが出る場合は代わりにSPI2_HOST , SPI3_HOSTを使用してください。
            cfg.spi_mode = 0; // SPI通信モードを設定 (0 ~ 3)
            cfg.freq_write = 80000000; // 伝送時のSPI時鐘 (最高80MHz, 四舍五入为80MHz除以整数得到的値)
            cfg.freq_read = 20000000; // 接收時のSPI時鐘
            cfg.spi_3wire = true; // 受信をMOSIピンで行う場合はtrueを設定
            cfg.use_lock = true; // 使用事务锁时设置为 true
            cfg.dma_channel = SPI_DMA_CH_AUTO; // 使用するDMAチャンネルを設定 (0=DMA不使用 / 1=1ch / 2=ch / SPI_DMA_CH_AUTO=自動設定)
            // ※ ESP-IDFバージョンアップに伴い、DMAチャンネルはSPI DMA CH AUTO(自動設定)が推奨になりました。1ch,2chの指定は非推奨になります。
        }
    }
};
```

- (2) Click on the File menu on the top menu bar.
- (3) Click on the Preferences menu item. This will open a Preferences dialog box.



- (4) You should be on the Settings tab in the Preferences dialog box by default.
- (5) Look for the textbox labeled “Additional Boards Manager URLs”.
- (6) If there is already text in this box add a coma at the end of it, then follow the next step.
- (7) Paste the following link into the text box :

Stable release link:

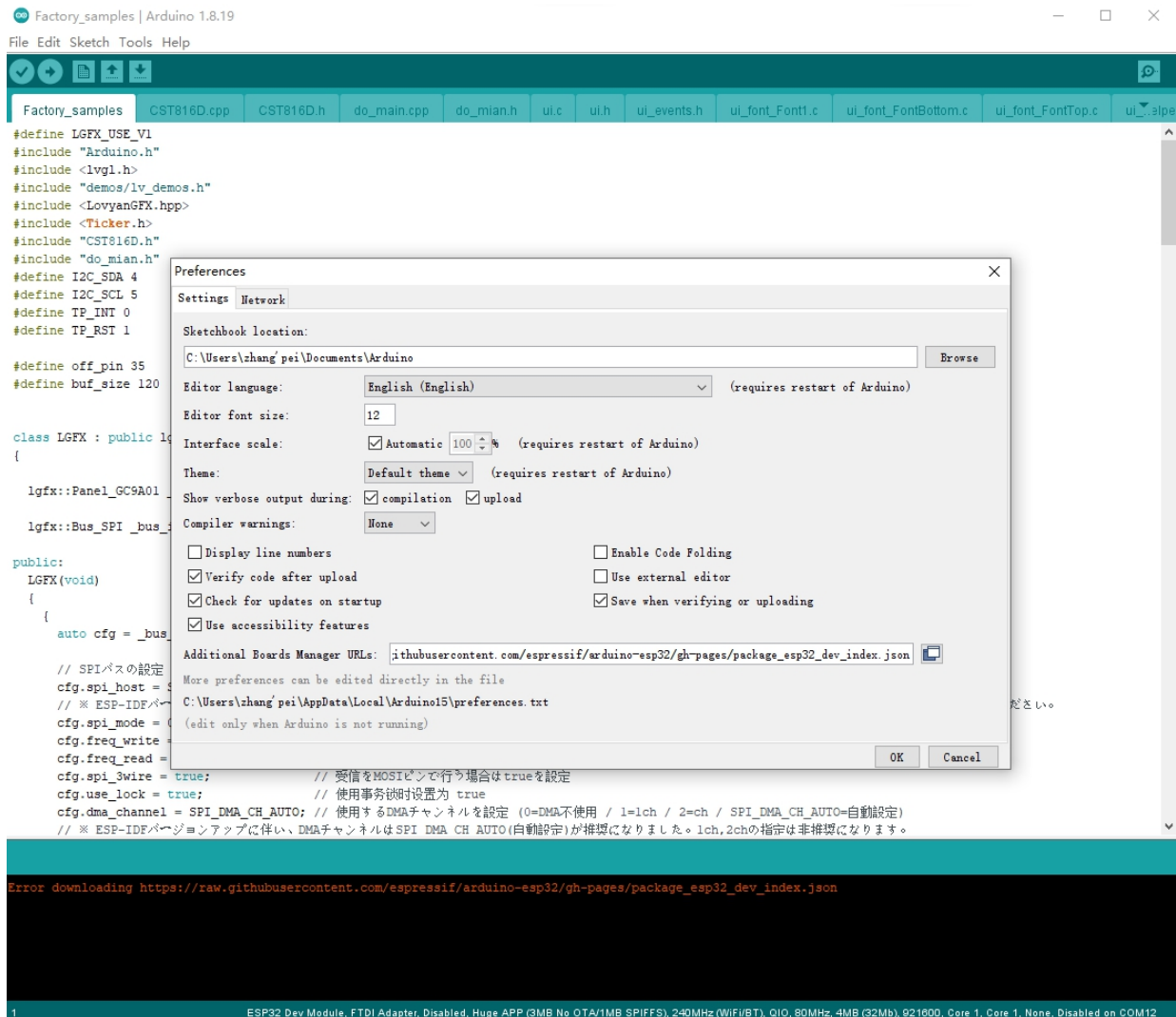
[https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package\\_esp32\\_dev\\_index.json](https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_dev_index.json)

Development release link:

[https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package\\_esp32\\_dev\\_index.json](https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_dev_index.json)

- (8) Click the OK button to save the setting.

The textbox with the JSON link in it is illustrated here:



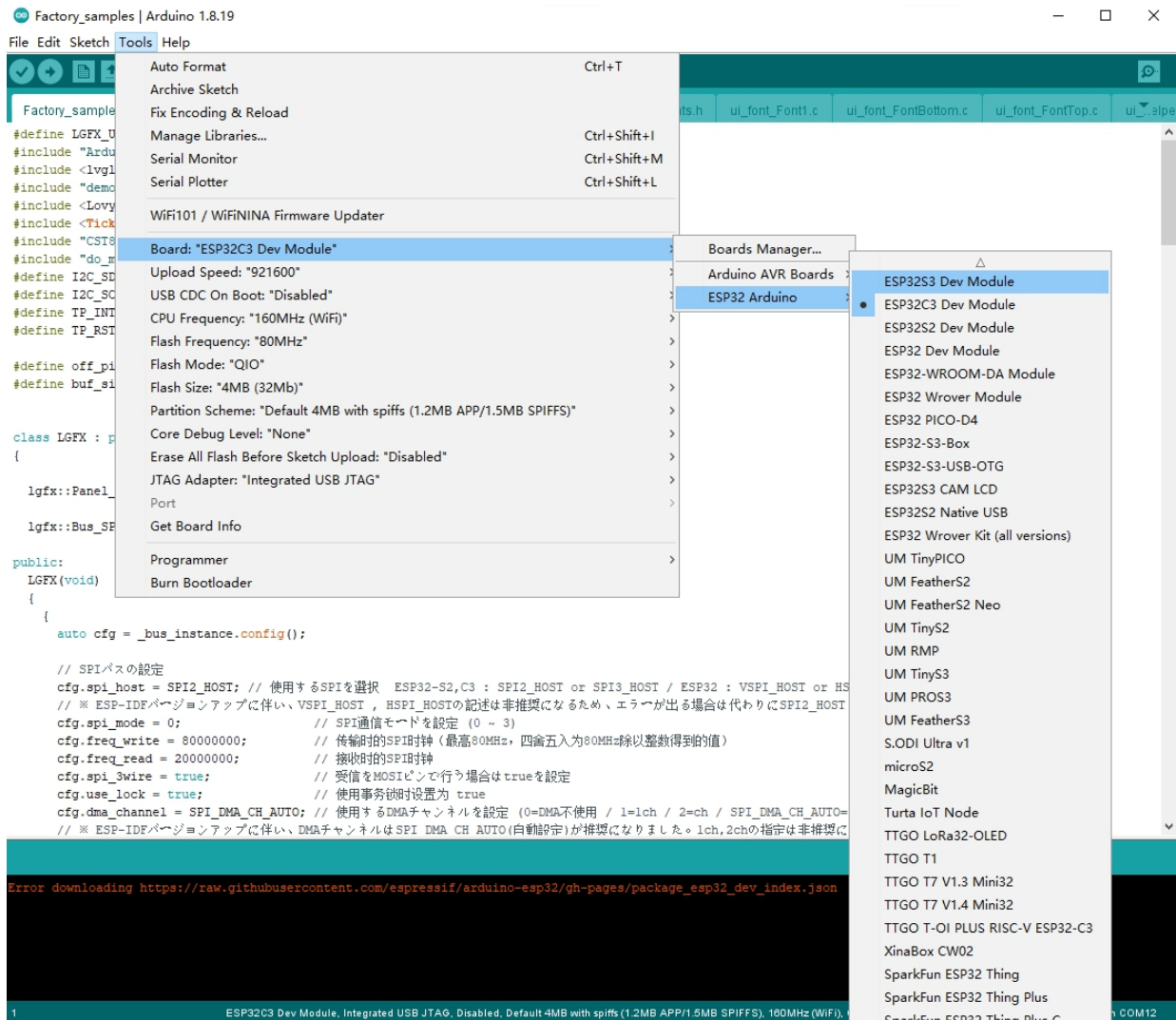
(9) In the Arduino IDE click on the Tools menu on the top menu bar.

(10) Scroll down to the Board: entry

(11) A submenu will open when you highlight the Board: entry.

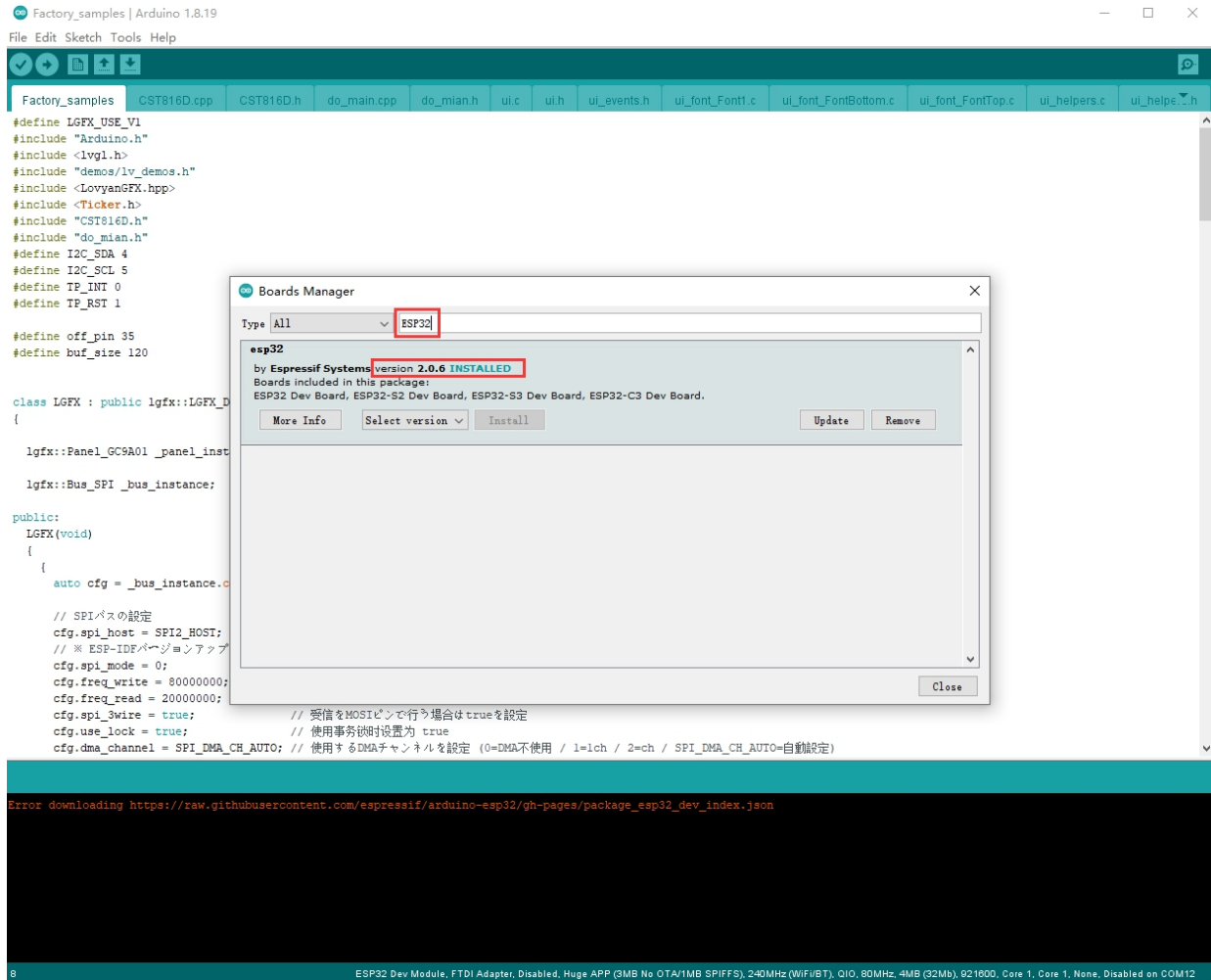
(12) At the top of the submenu is Boards Manager. Click on it to open the Boards Manager dialog box.

(13) In the search box in the Boards Manager enter "esp32".



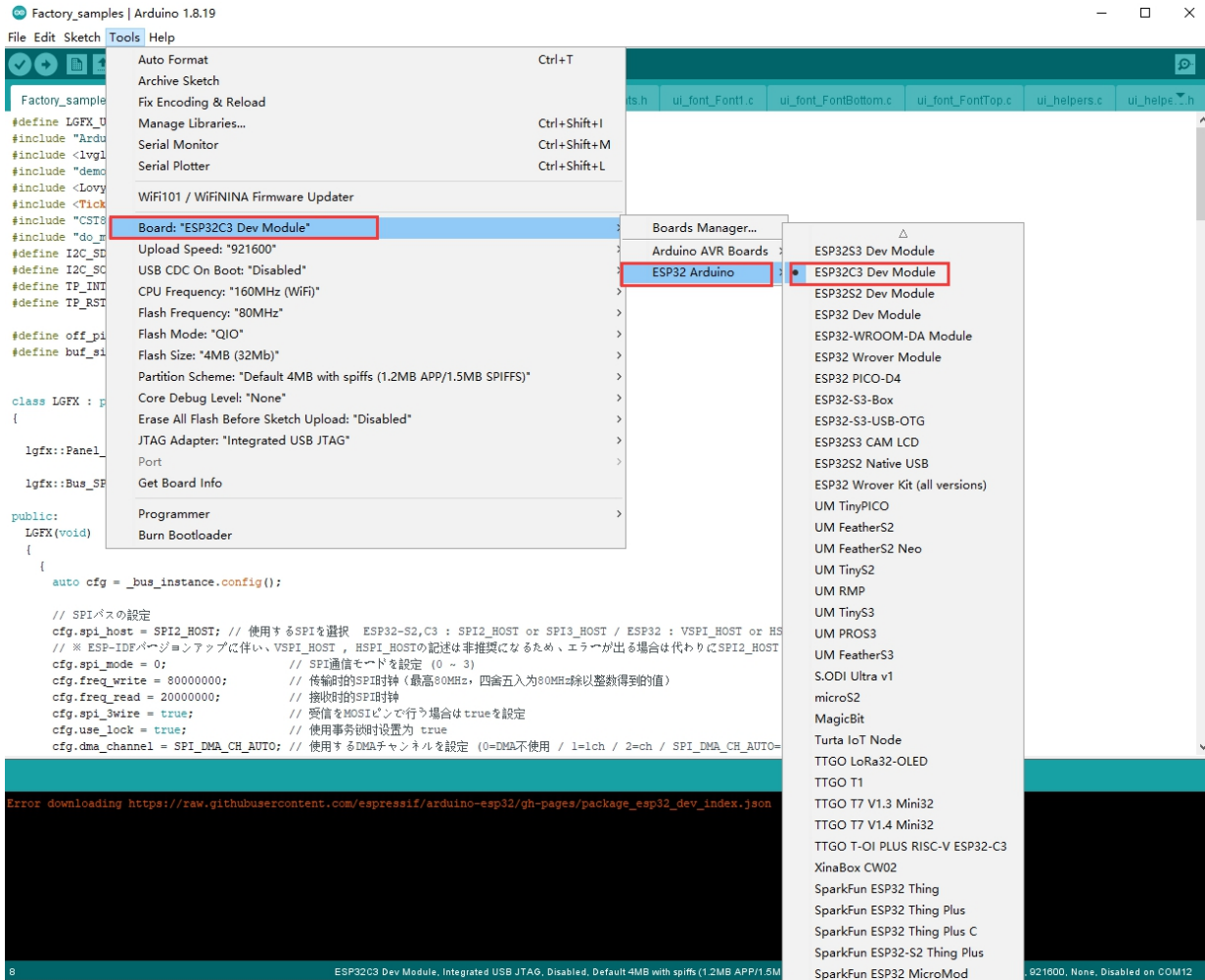
(14) You should see an entry for “esp32 by Espressif Systems”. Highlight this entry and click on the Install button.

This will install the ESP32 boards into your Arduino IDE

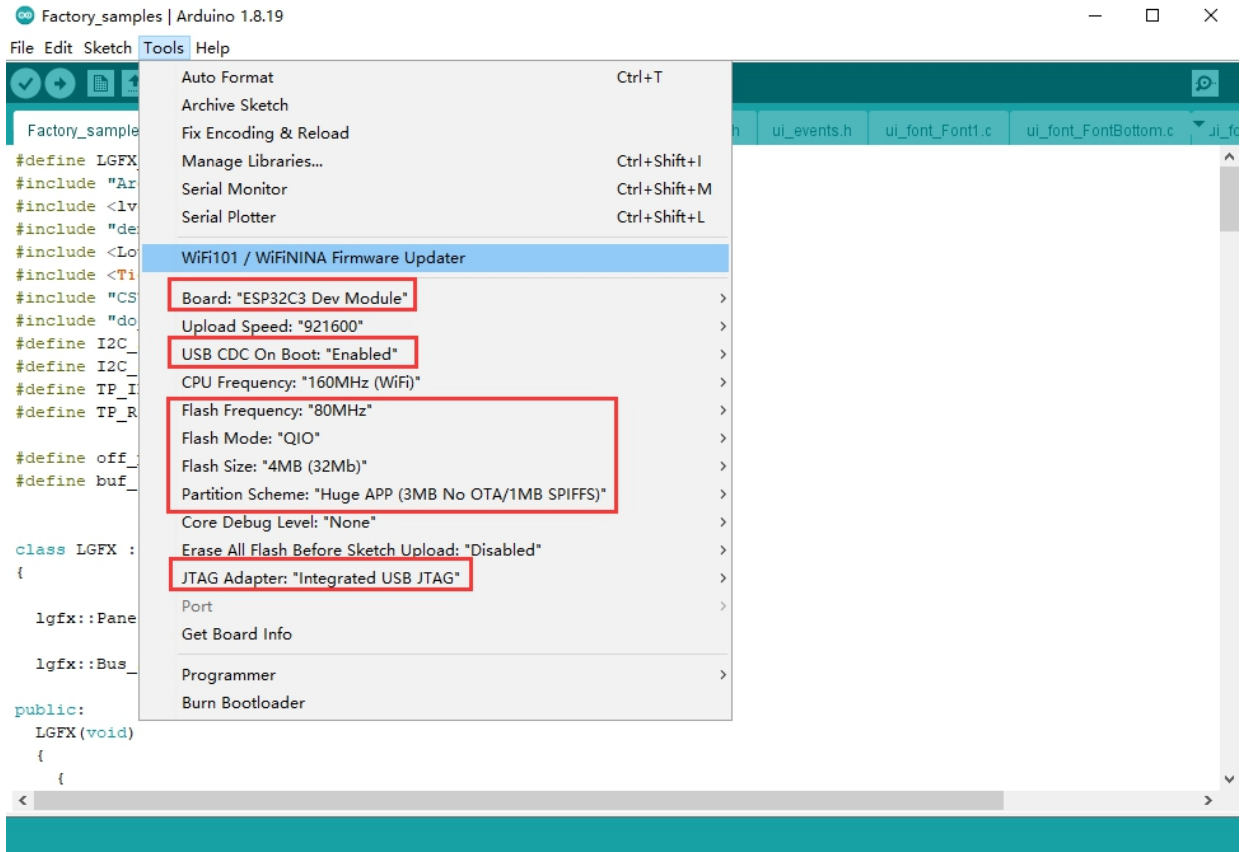


Once the installation completes, we need to select the correct board options for the "ESP32 Arduino" board. In the board type, in the tools tab, we choose "ESP32C3 Dev Module".

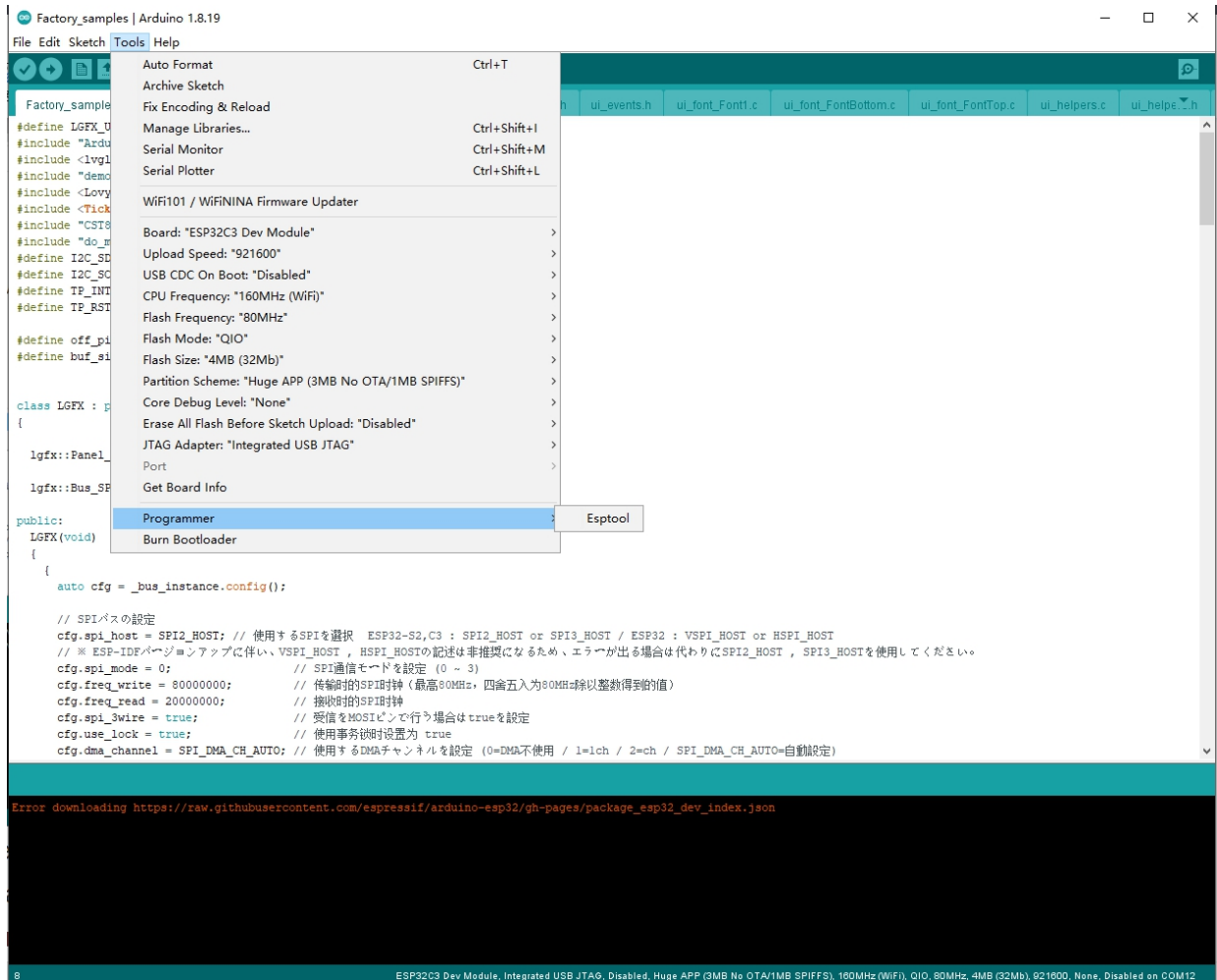








Set and In the programmer entry of the same tab, we choose "esptool".



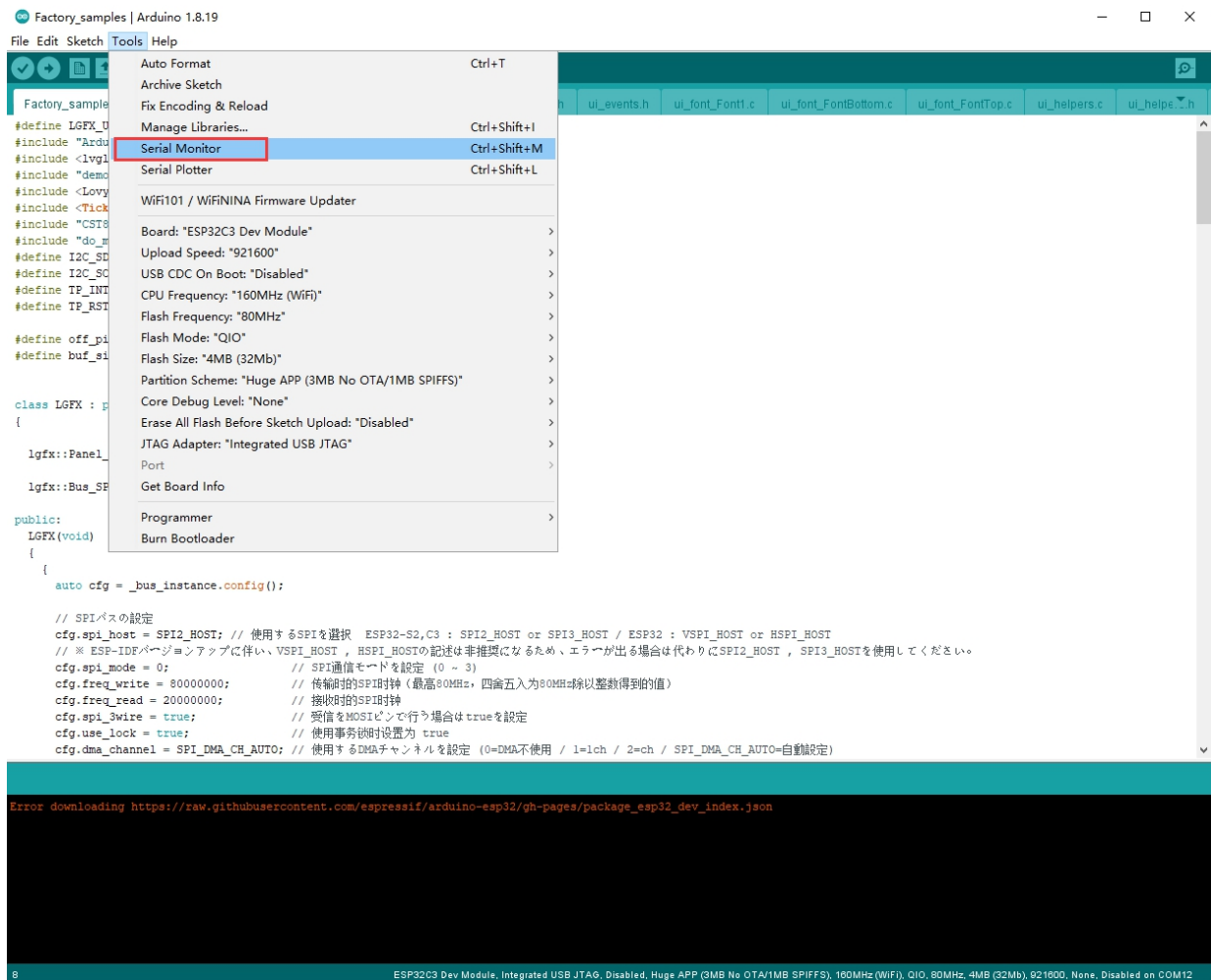
It's important to note that after the code is uploaded, the device will start to run it. So, if we want to upload a new program, we need to reset the power of the device, in order to guarantee that it enters flashing mode again.

First program

Since this platform is based on Arduino, we can use many of the usual functions. As an example for the first program, the code below starts the Serial port and prints "hello from ESP32" every second.

```
void setup() {  
    Serial.begin(115200);  
}  
  
void loop() {  
    Serial.println("hello from ESP32");  
    delay(1000);  
}
```

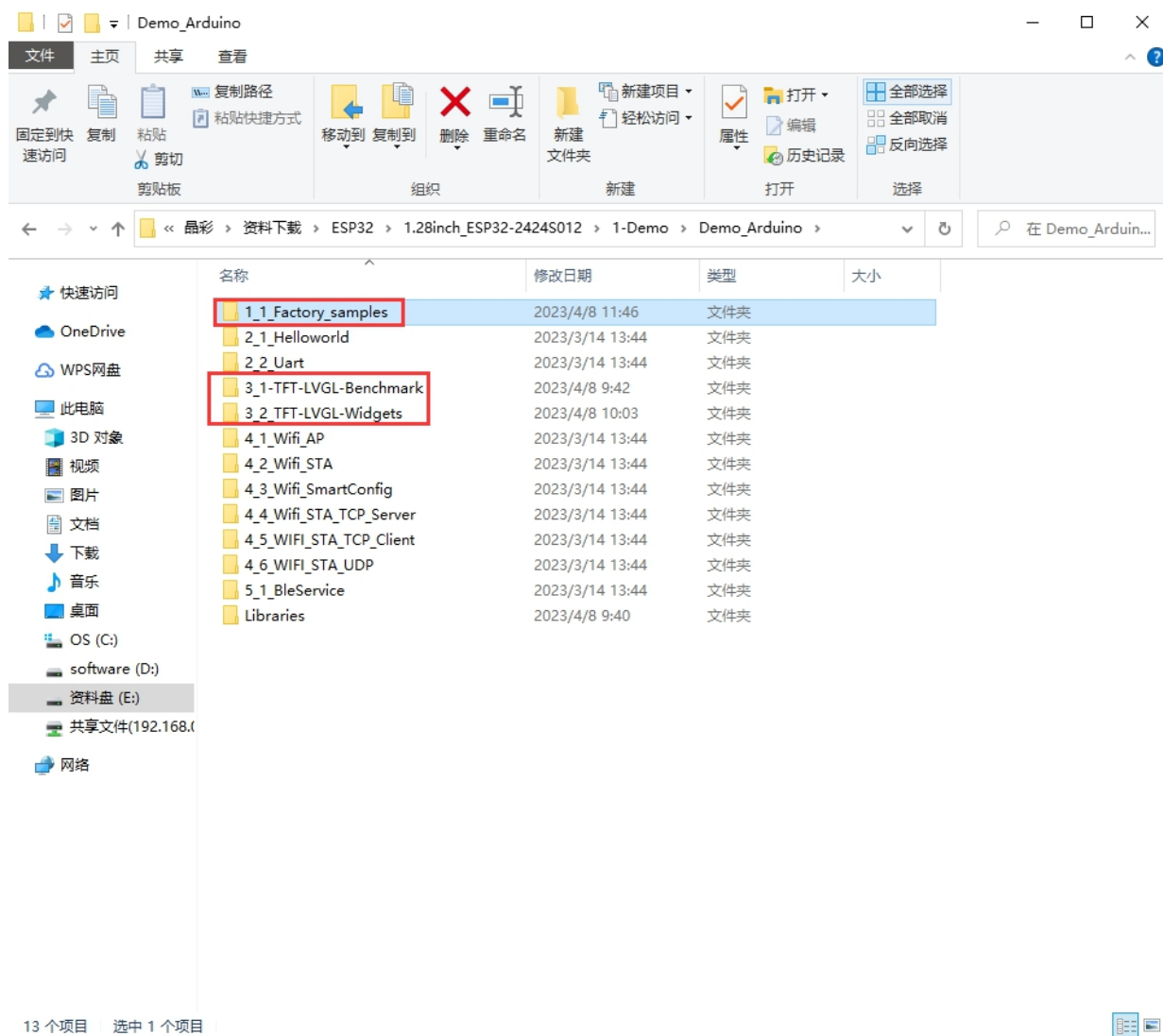
If everything is working fine, we will see the output in the serial console shown.



Again thank you for so much concern.. Hopefully, it's the beginning of a wonderful relationship!

## Sample program usage

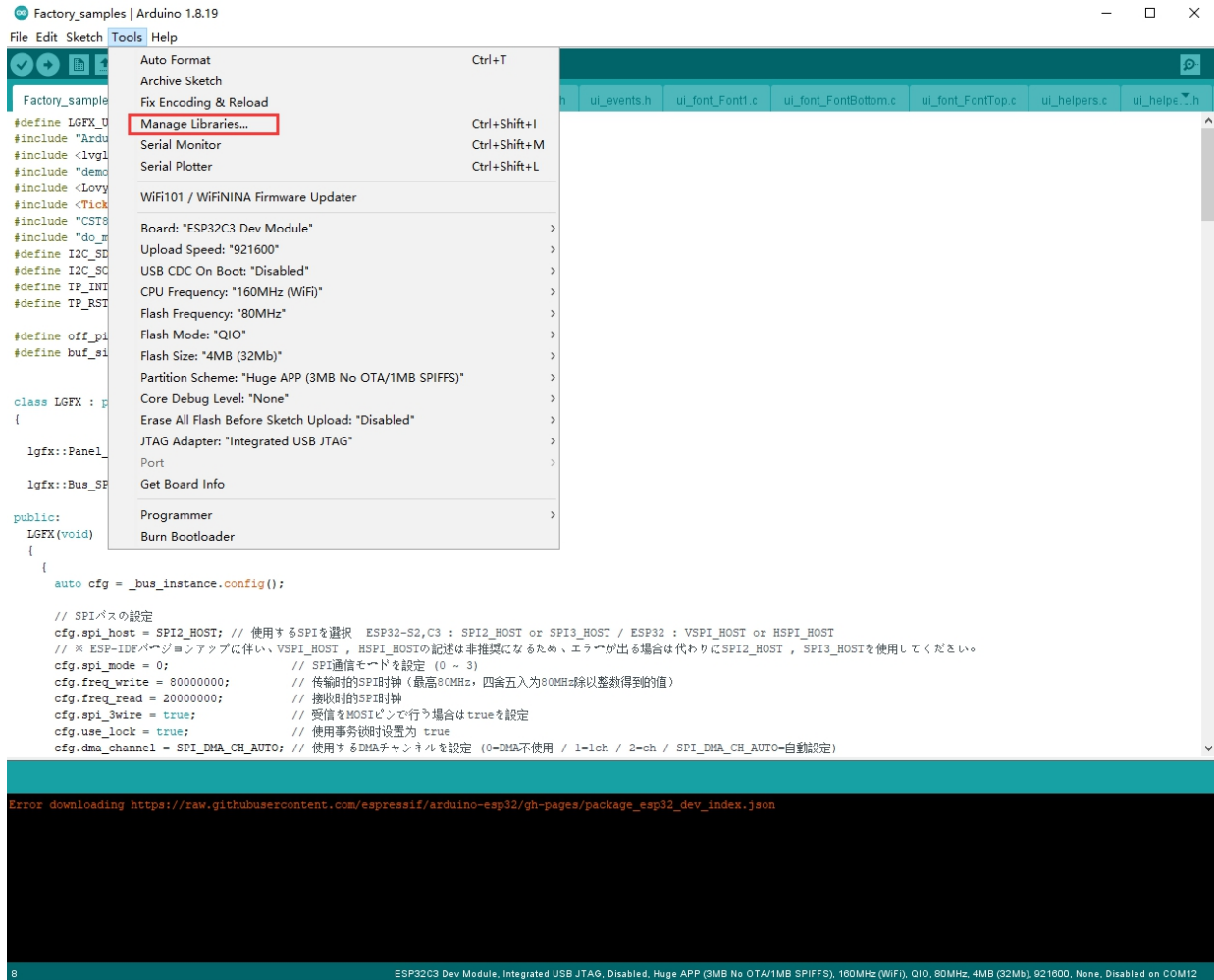
At present, only a preliminary explanation and introductory use are given to the samples displayed on the screen, and the corresponding examples in the data center are found, as shown in the figure:

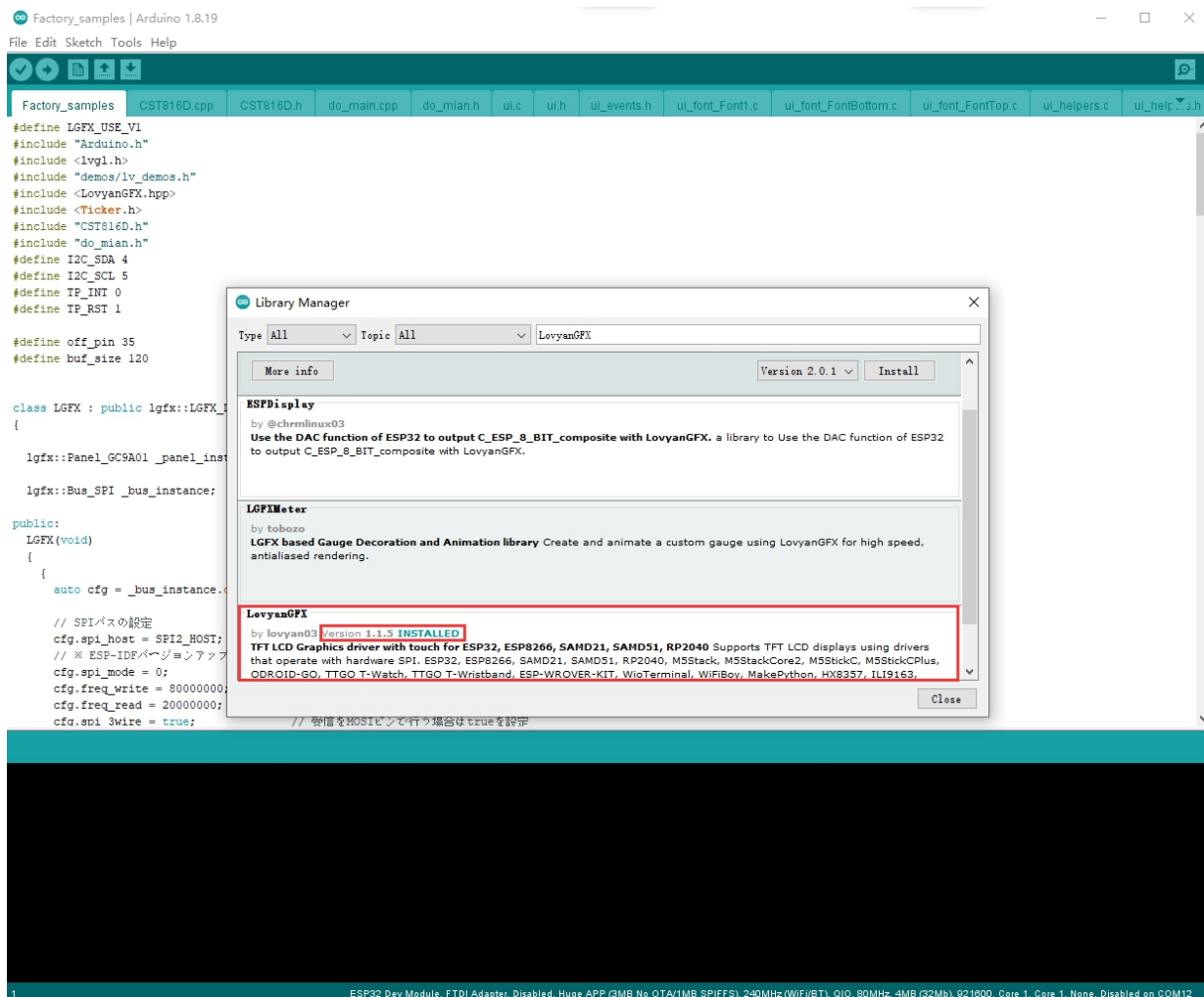


The examples in the red circle are all based on the LovyanGFX library as the basic application. This library supports various commonly used driver chips, such as ST7735, ST7789, ILI9341, etc., and has good compatibility.

LovyanGFX library file installation:

Open the library manager in Arduino, search for LovyanGFX, and click instal .



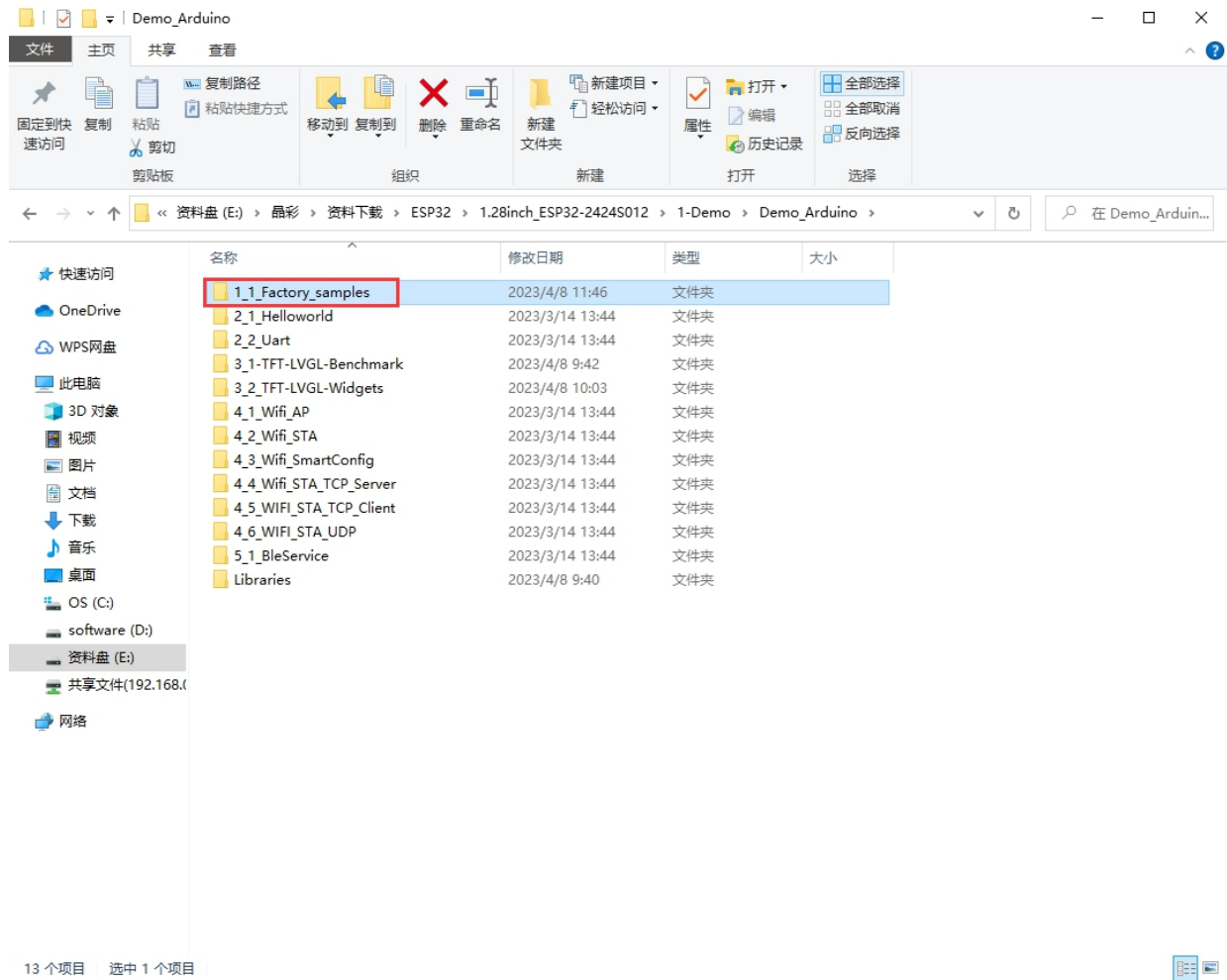


Although the Arduino\_GFX library has many advantages, it may also have a troublesome place for ordinary users, that is, after the installation

### About the use of Factory\_samples:

Find the data center 1\_1\_Factory\_samples

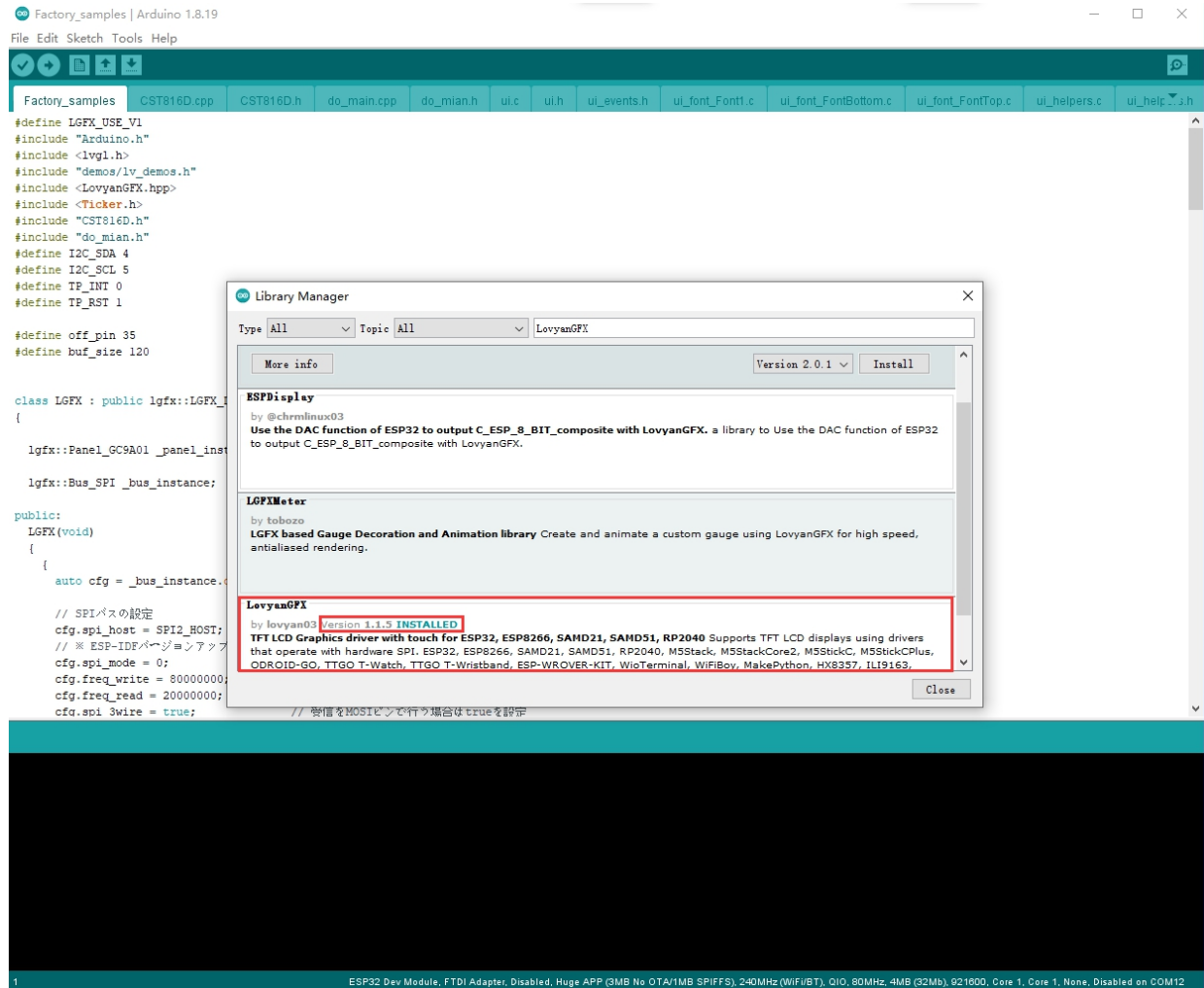
As shown:



Download library files .

LovyanGFX library





After compiling, you can run 1\_1\_Factory\_samples normally.