Welcome to the JCZN Workshop!

·····Table of contents·····

Getting Started

Introduction

The objective of this post is to explain how to upload an Arduino program to the ESP32-2432S022 module, from JCZN .

The ESP32 WiFi and Bluetooth chip is the latest generation of Espressif products. It has a dual-core 32-bit MCU, which integrates WiFi HT40 and Bluetooth/BLE 4.2 technology inside.
ESP wroom 32 has a significant performance improvement. It is equipped with a high-performance dual-core Tensilica LX6 MCU. One core handles high speed connection and the other for standalone application development. The dual-core MCU has a 240 MHz frequency and a computing power of 600 DMIPS.
In addition, it supports Wi-Fi HT40, Classic Bluetooth/BLE 4.2, and more GPIO resources.

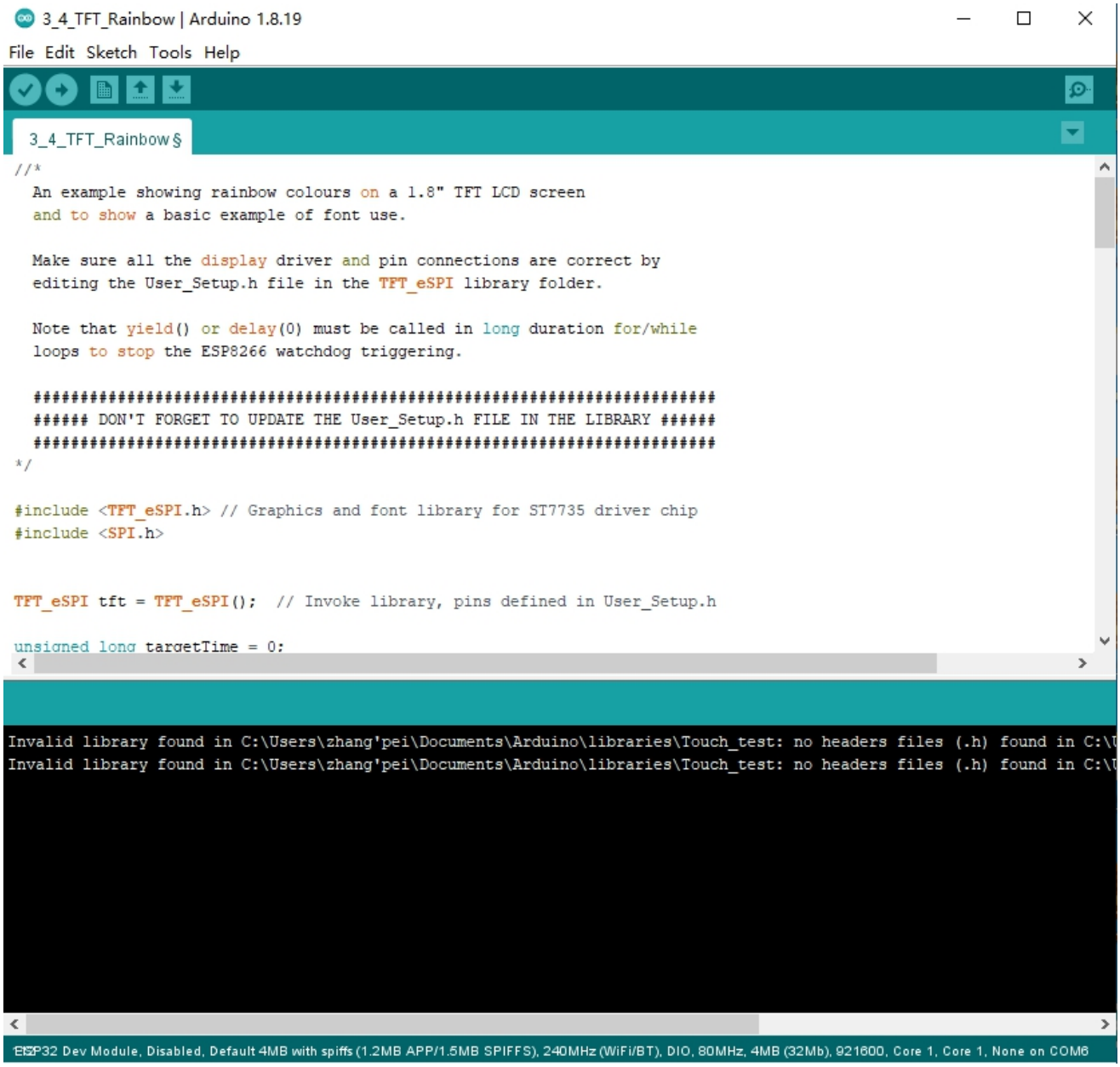
Installing using Arduino IDE

Programming the ESP32
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, 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

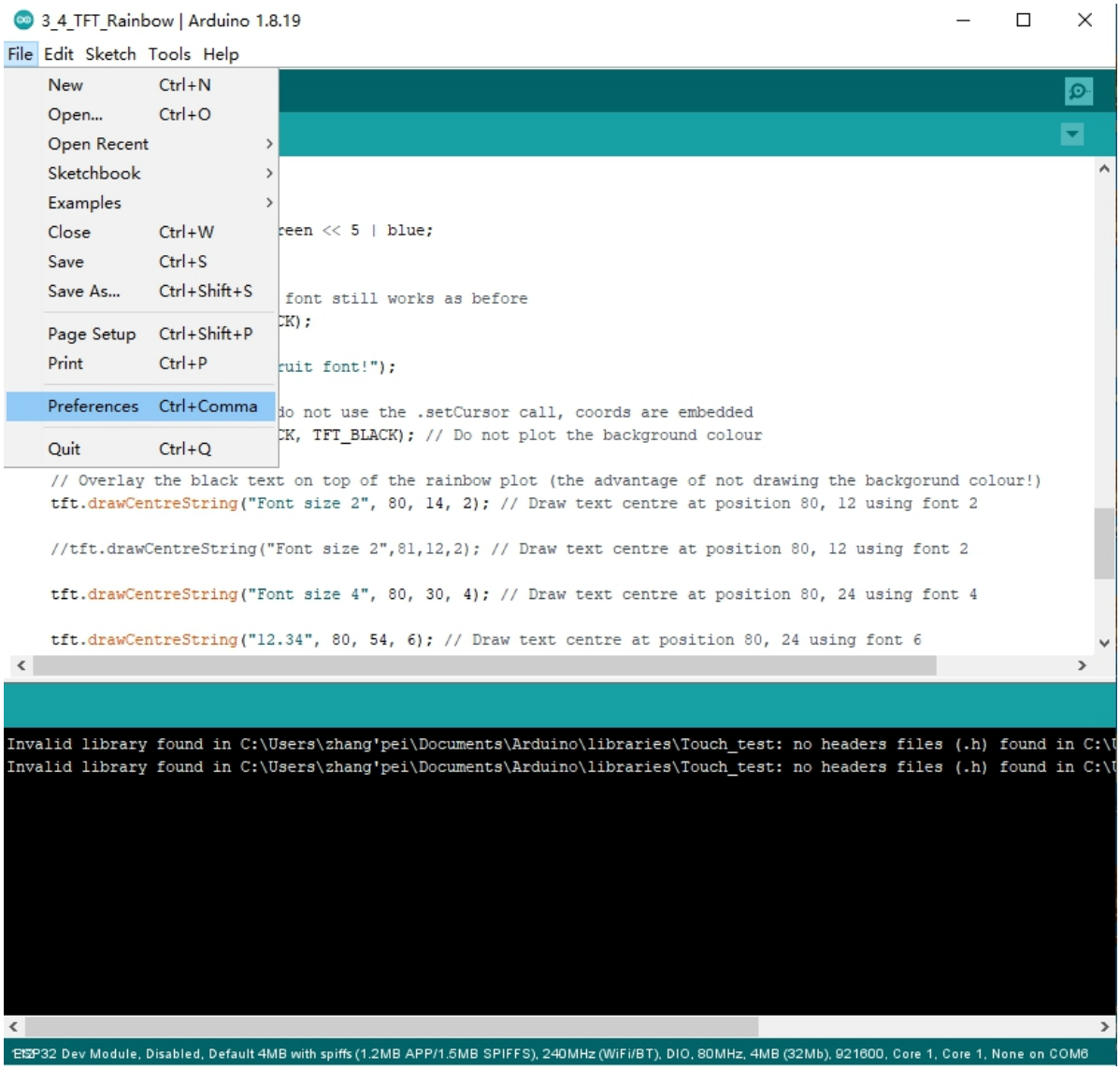
2, This is the way to install Arduino-ESP32 directly from the Arduino IDE.

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.



(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:

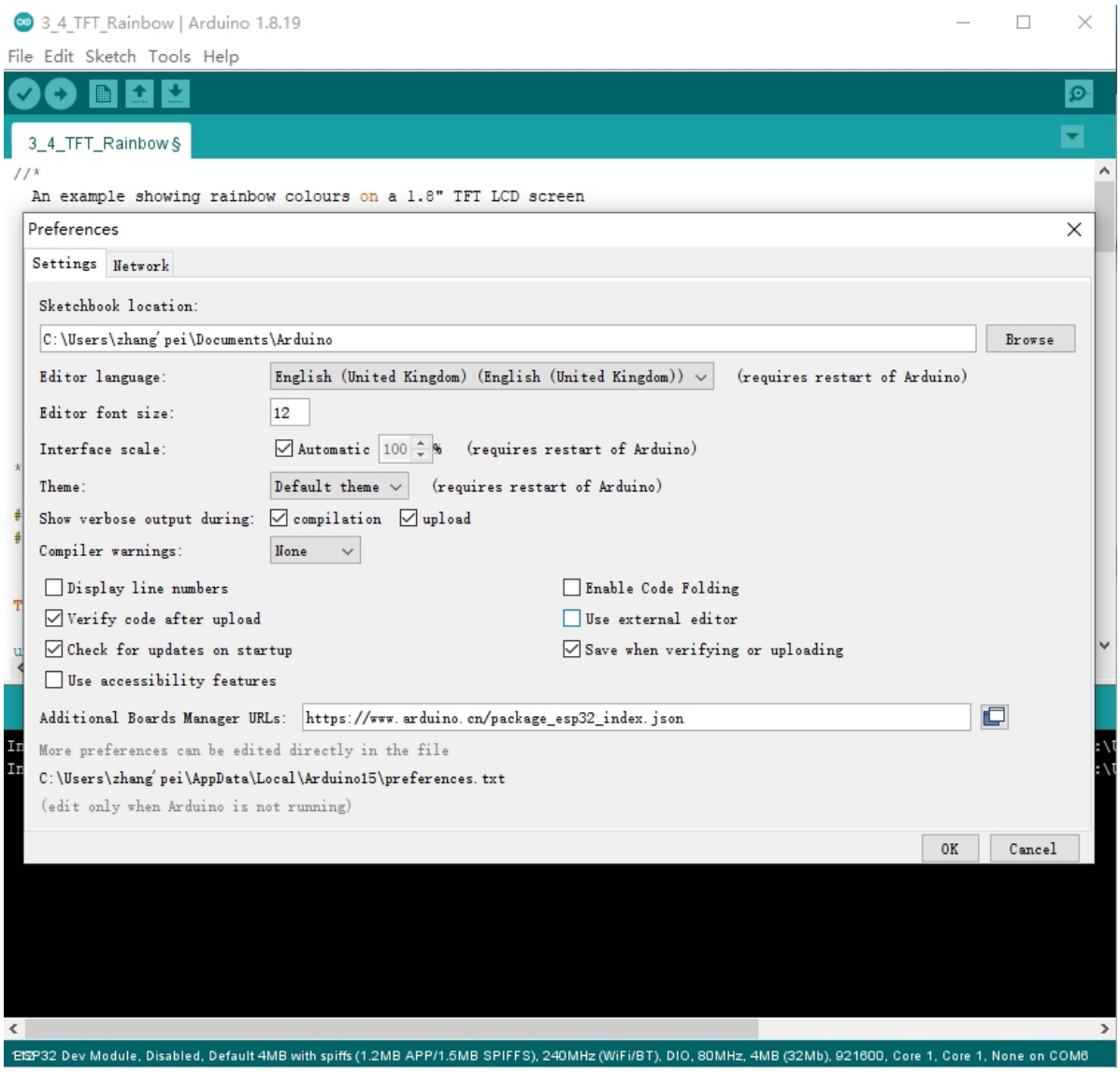
https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json

Development release link:

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:



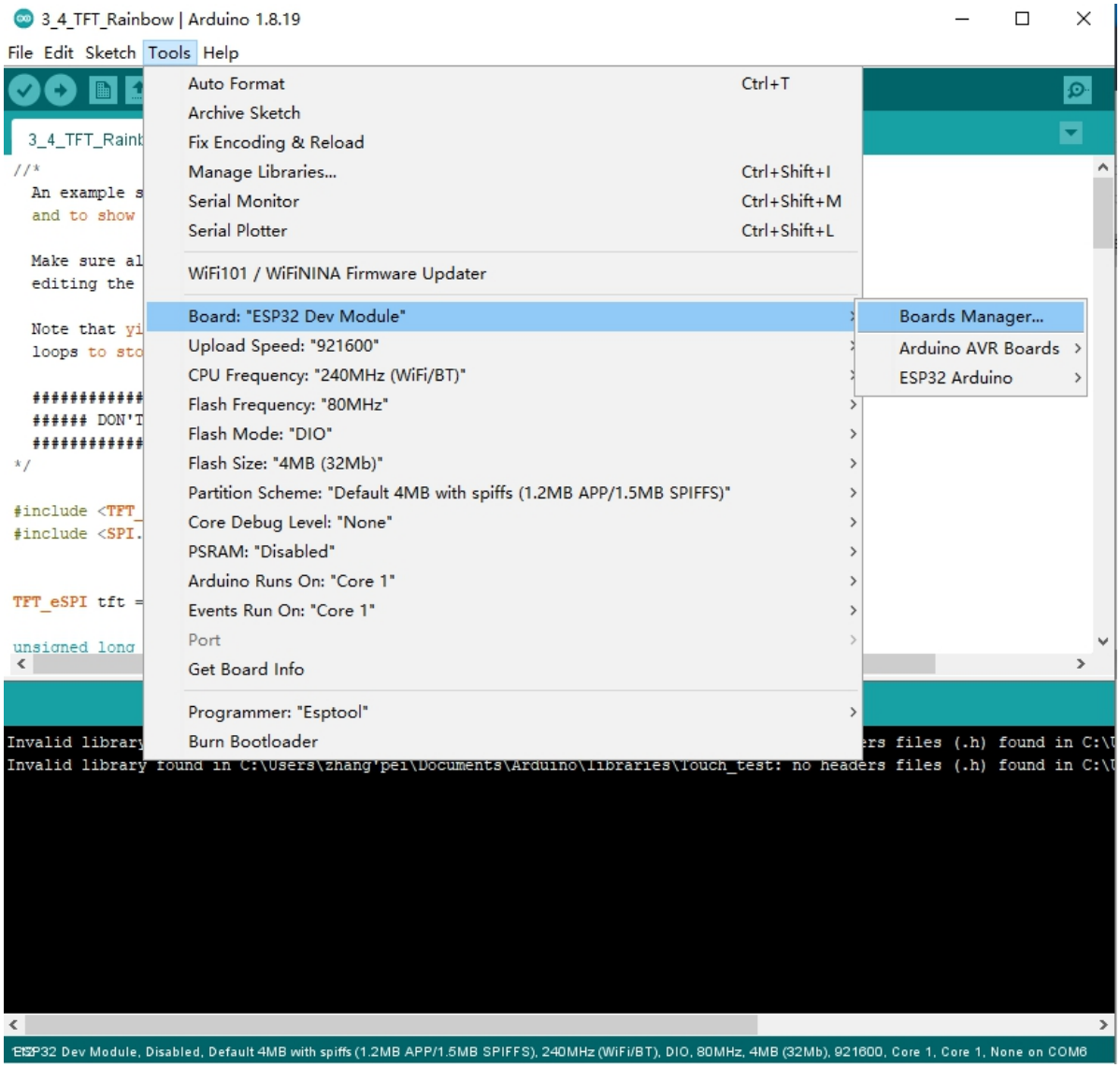
(9) In the Arduino IDE click on the Tools menu on the top menu bar.

(10) Scroll down to the Board: entry

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

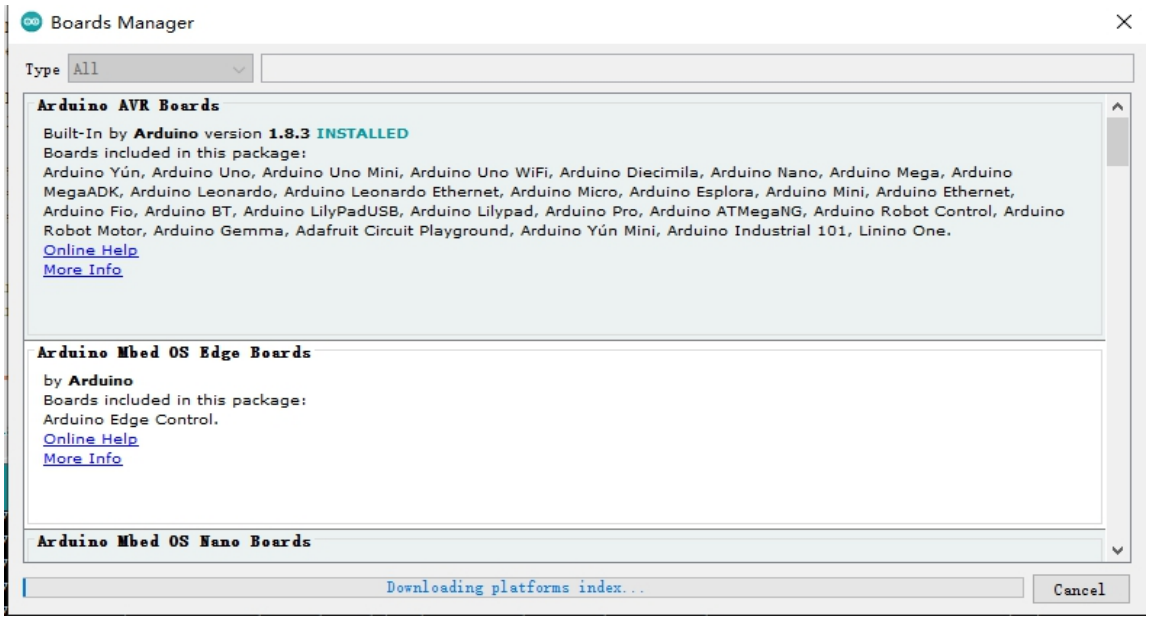
(12) At the top of the submenu is Boards Manager. Click on it to open the Boards Manager dialog box.

(13)I n the search box in the Boards Manager enter “esp32”.

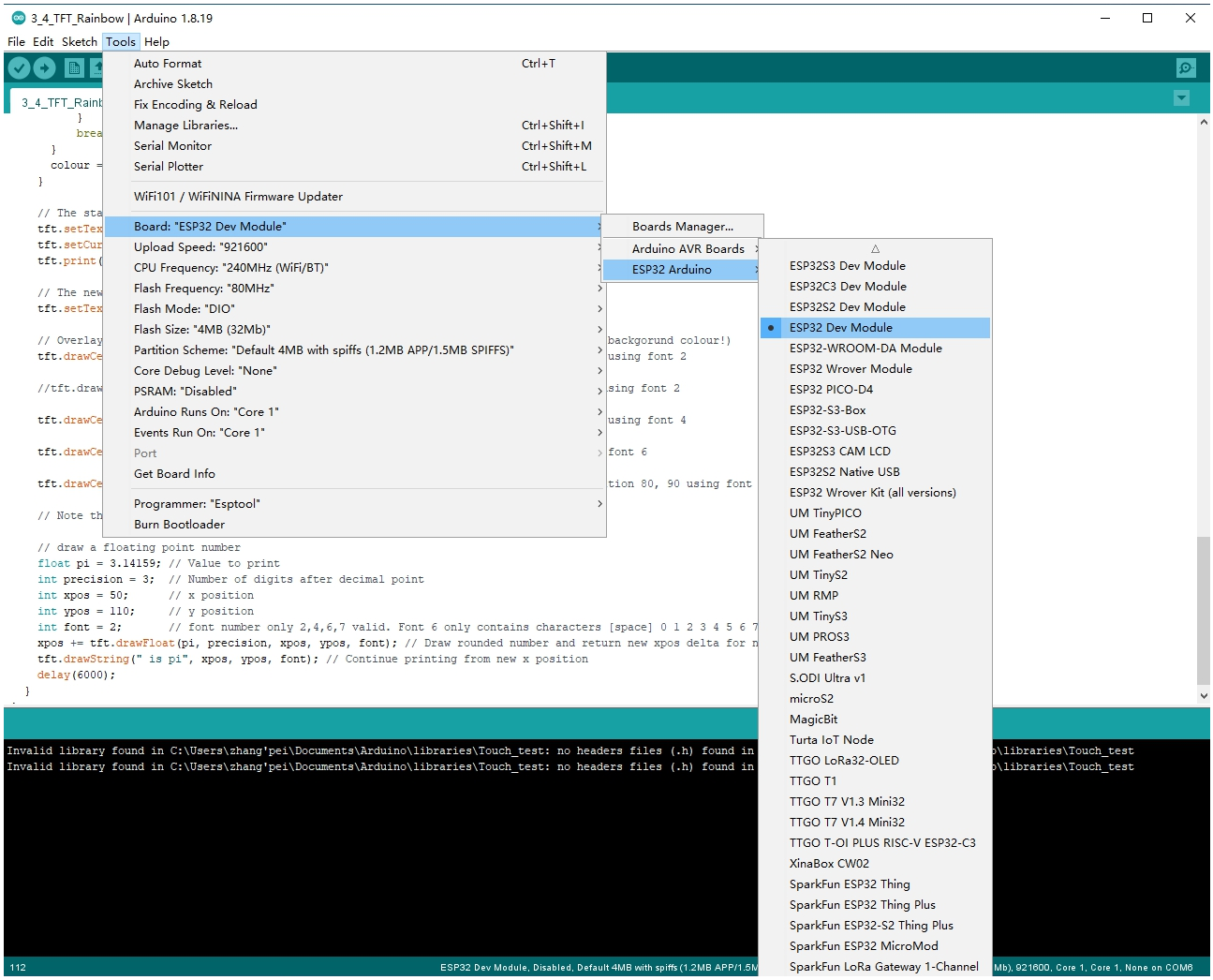


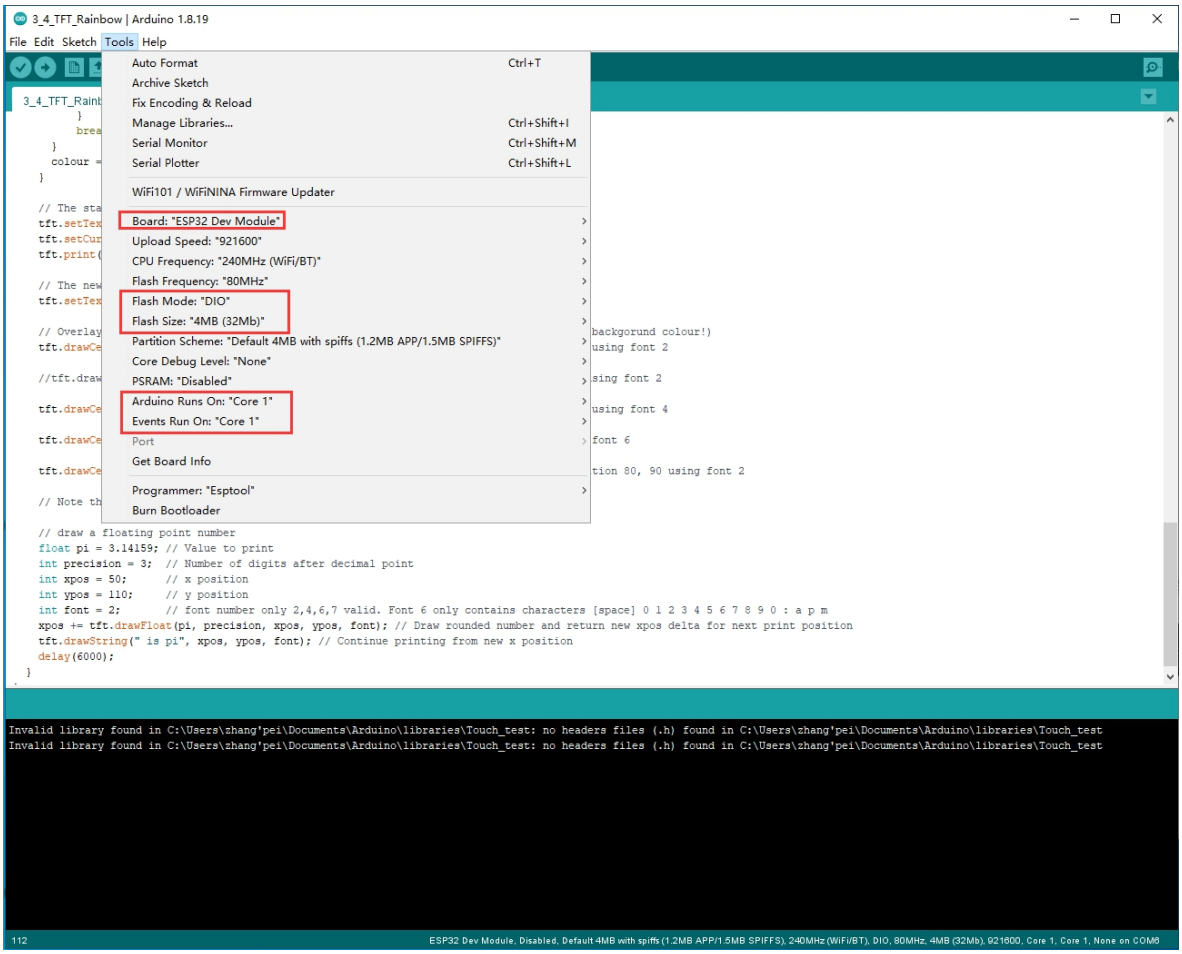
(14) You should see an entry for “esp32 by Espressif Systems”. Highlight this entry and click on the Install button.

This will install the ESP32 boards into your Arduino IDE

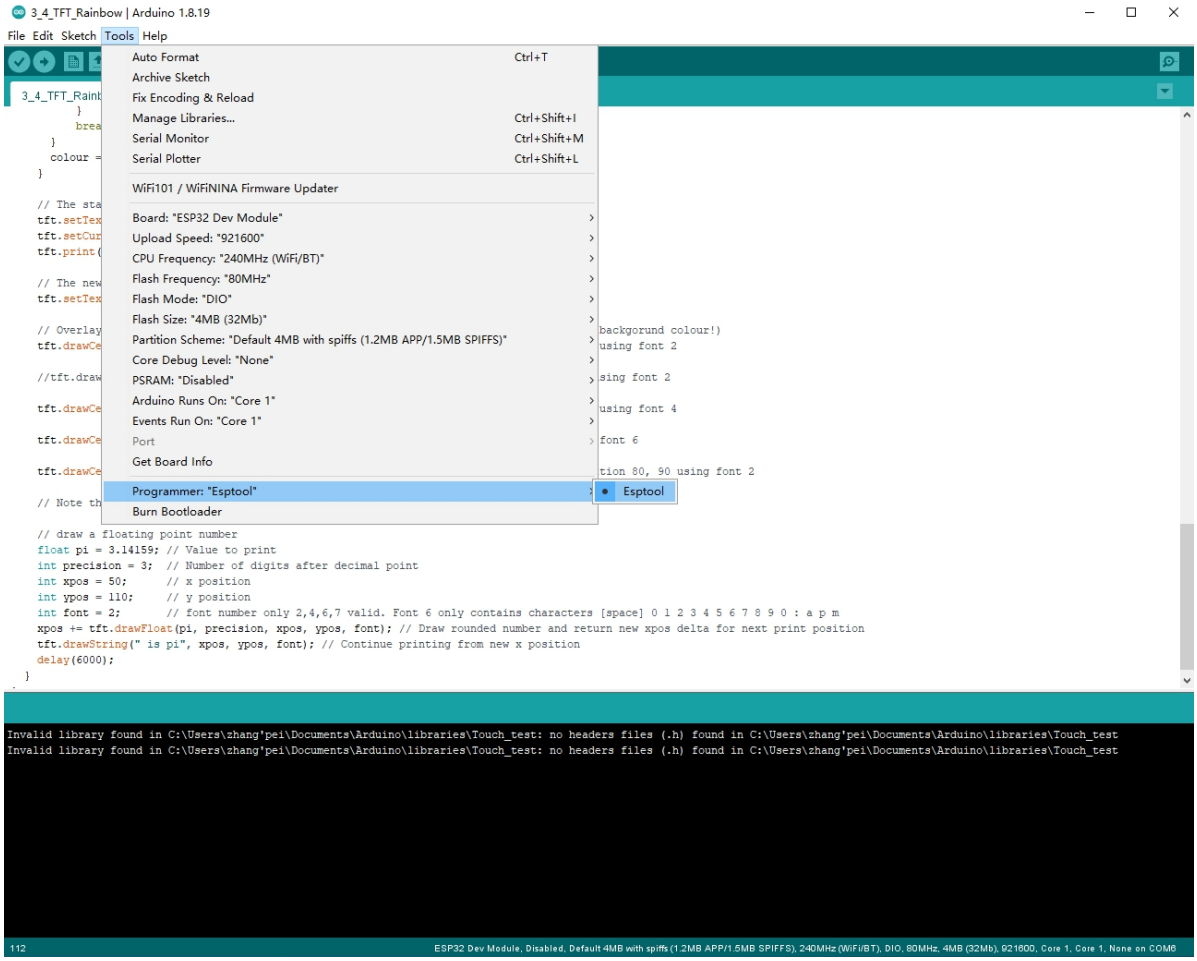


Once the installation completes, we need to select the correct board options for the "ESP32 Arduino" board. In the board type, in the tools tab, we choose “ESP32 Dev Module”.





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, wee need to reset the power of the device, in order to guarantee that it enters flashing mode again.

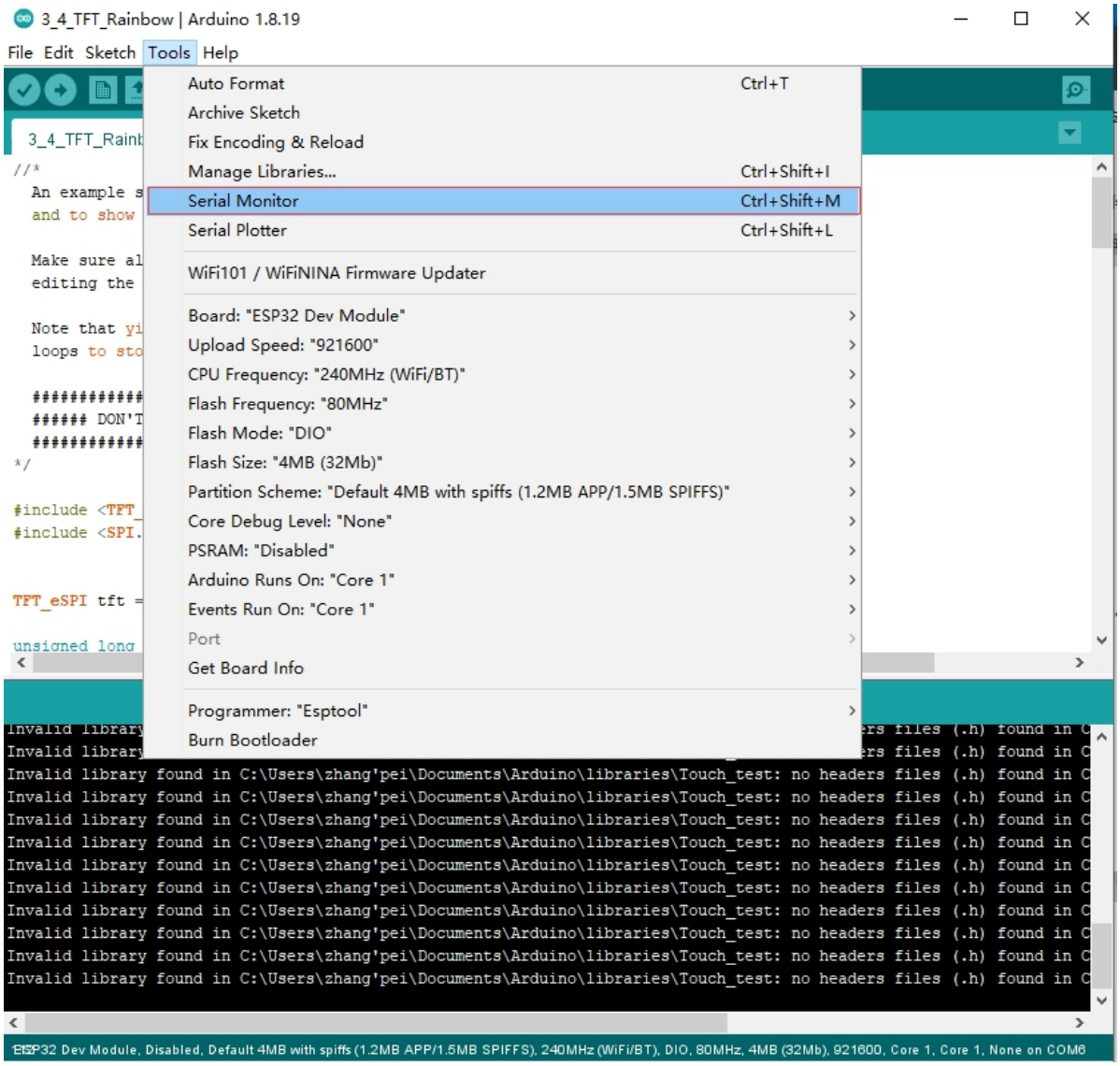
First program

Since this platform is based on Arduino, we can use many of the usual functions. As an example for the first program, the code bellow 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);
}
```

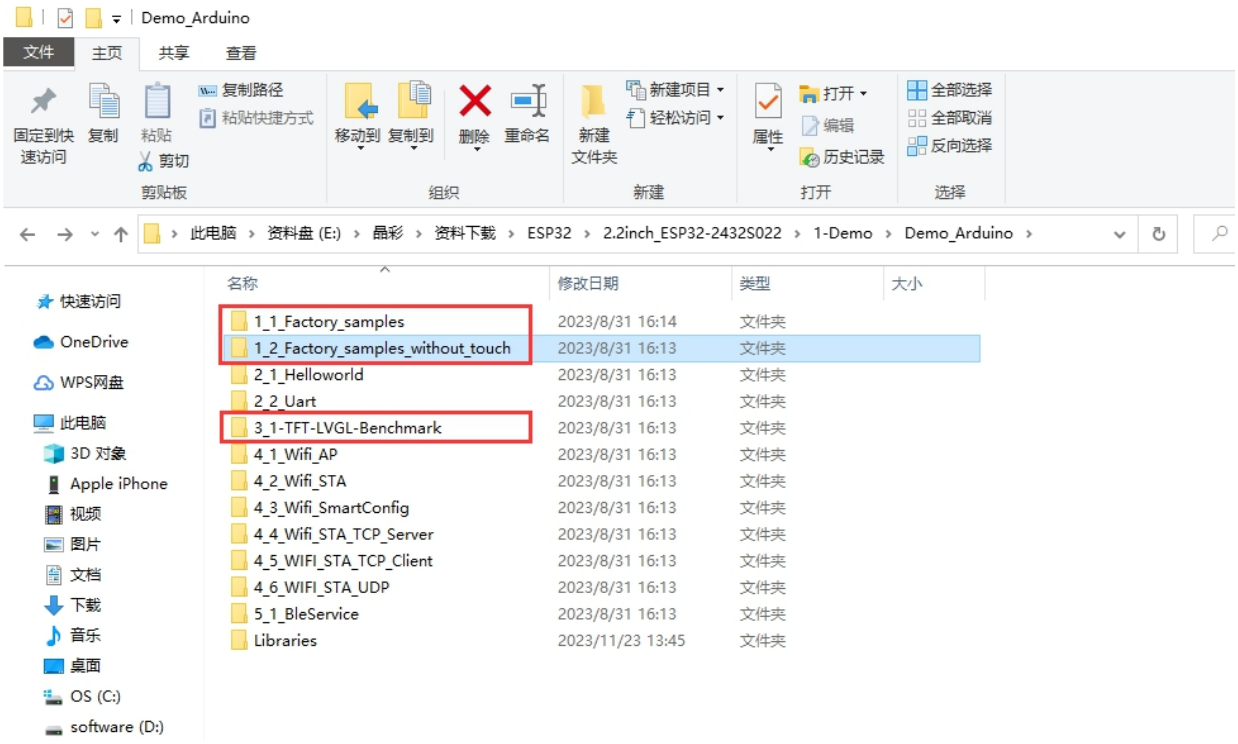
If everything is working fine, we will see the output in the serial console shown.



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

Sample program usage

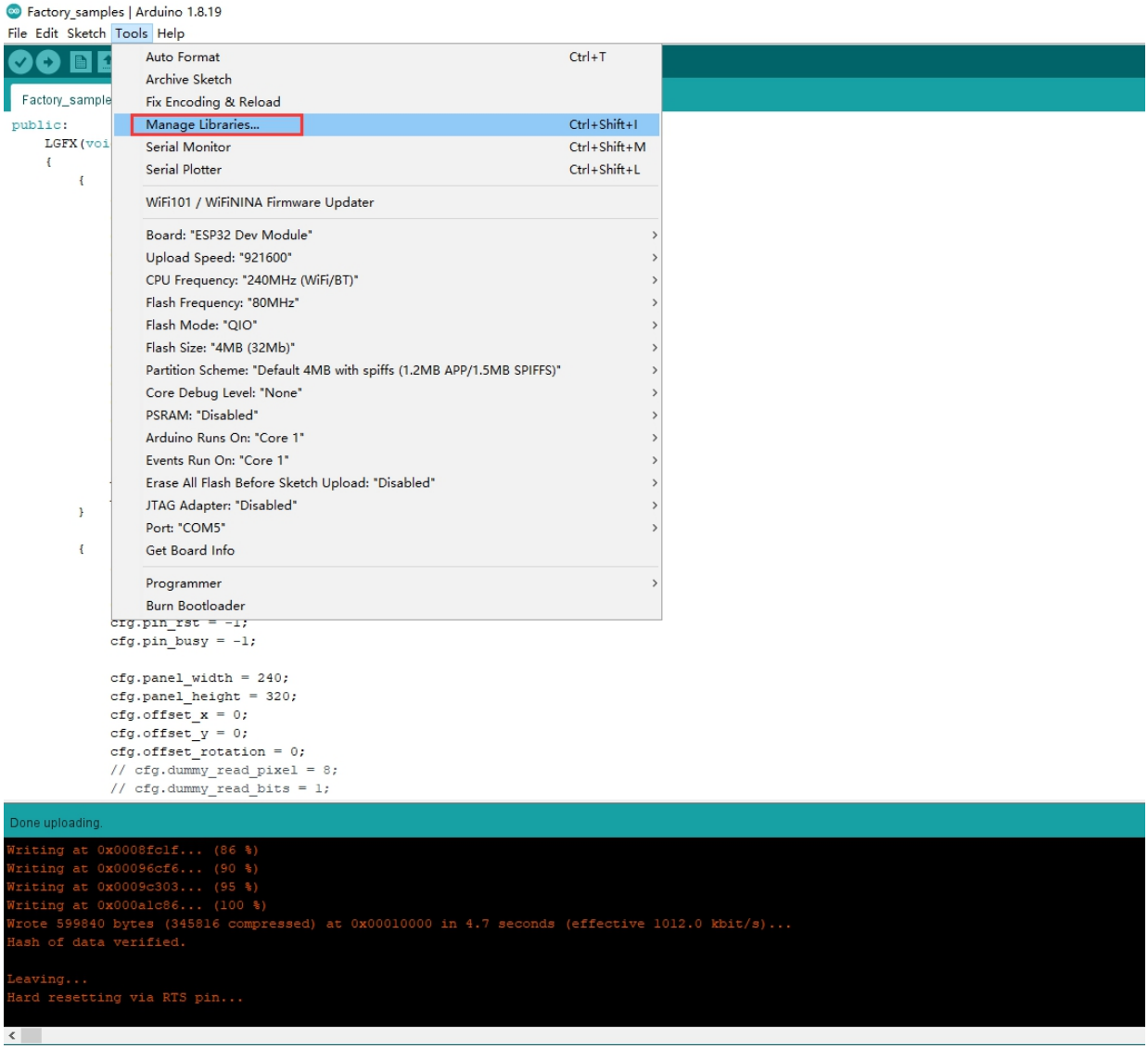
At present, only a preliminary explanation and introductory use are given to the samples displayed on the screen, and the corresponding examples in the data center are found, as shown in the figure:

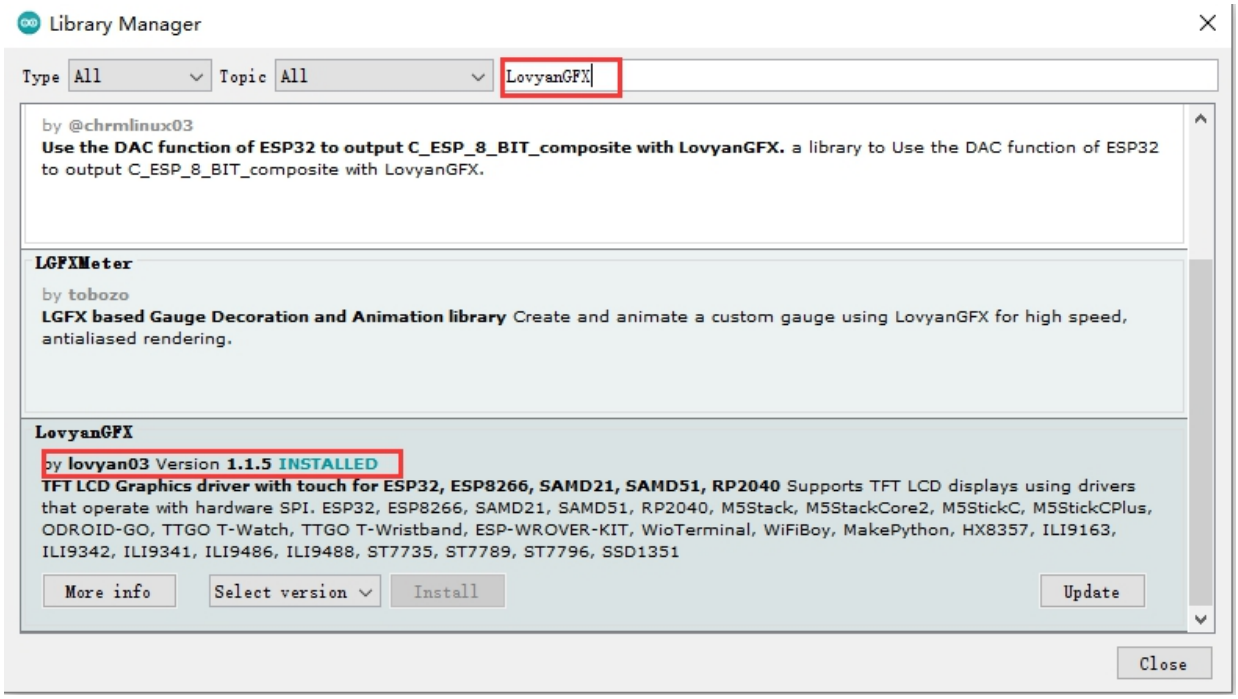


The examples in the red circle are all based on the LovyanGFX library as the basic application. This library supports various commonly used driver chips, such as ST7735, ST7789, ILI9341, etc., and has good compatibility.

LovyanGFX library file installation:

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


Factory_samples | Arduino 1.8.19
File Edit Sketch Tools Help
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Manage Libraries... Ctrl+Shift+I
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
WiFi101 / WiFiNINA Firmware Updater
Board: "ESP32 Dev Module"
Upload Speed: "921600"
CPU Frequency: "240MHz (WiFi/BT)"
Flash Frequency: "80MHz"
Flash Mode: "QIO"
Flash Size: "4MB (32Mb)"
Partition Scheme: "Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS)"
Core Debug Level: "None"
PSRAM: "Disabled"
Arduino Runs On: "Core 1"
Events Run On: "Core 1"
Erase All Flash Before Sketch Upload: "Disabled"
JTAG Adapter: "Disabled"
Port: "COM5"
Get Board Info
Programmer
Burn Bootloader
cfg.pin_busy = -1;
cfg.panel_width = 240;
cfg.panel_height = 320;
cfg.offset_x = 0;
cfg.offset_y = 0;
cfg.offset_rotation = 0;
// cfg.dummy_read_pixel = 8;
// cfg.dummy_read_bits = 1;
Done uploading.
Writing at 0x0008fc1f... (86 %)
Writing at 0x00096cf6... (90 %)
Writing at 0x0009c303... (95 %)
Writing at 0x000a1c86... (100 %)
Wrote 599840 bytes (345816 compressed) at 0x00010000 in 4.7 seconds (effective 1012.0 kbit/s)...
Hash of data verified.
Leaving...
Hard resetting via RTS pin...

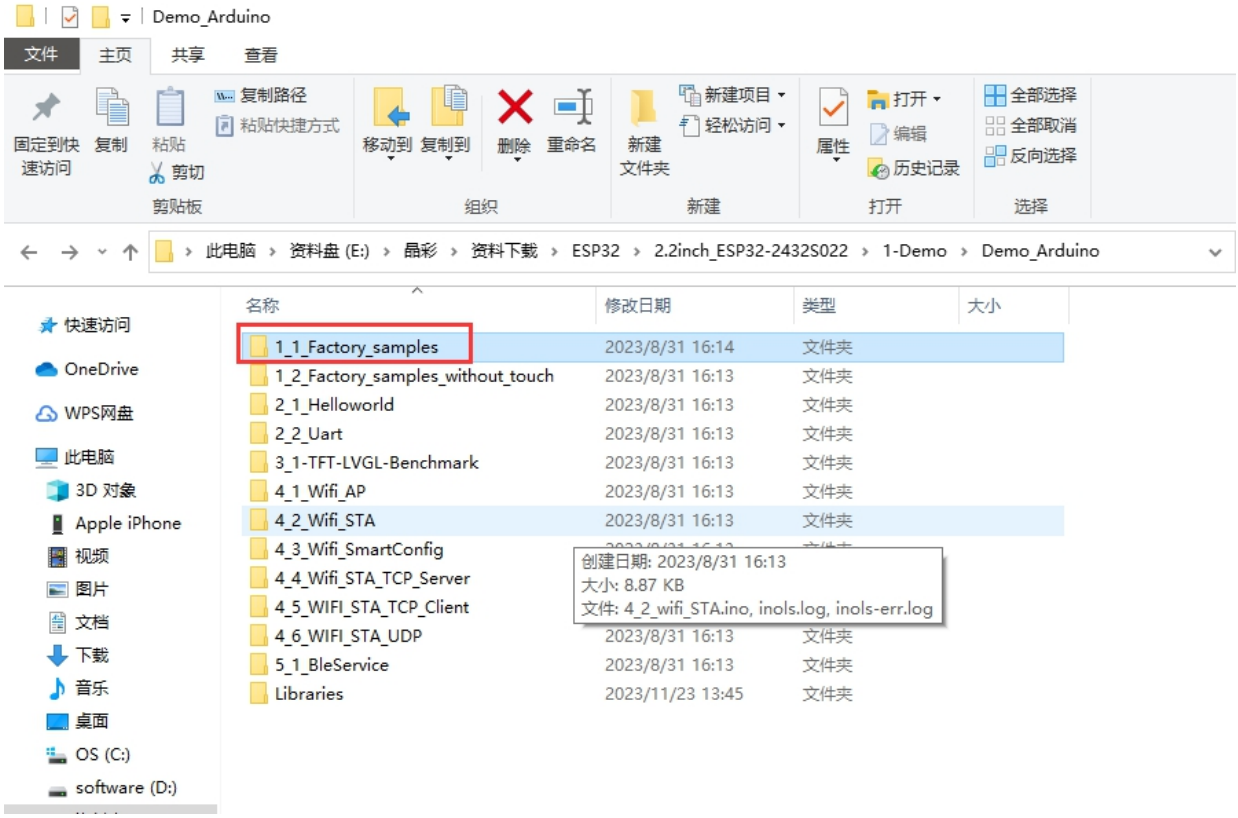


After configuring these, compile the arduino function in Factory_samples to light up the screen .

About the use of touch and LVGL:

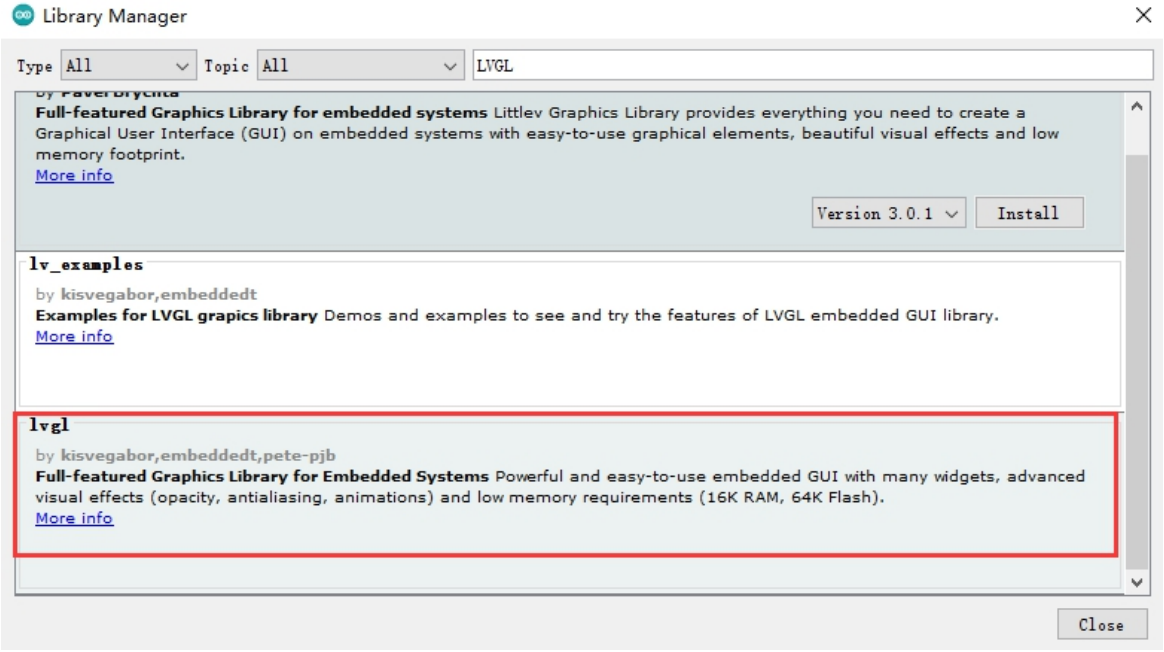
Find the data center Factory_samples

As shown:



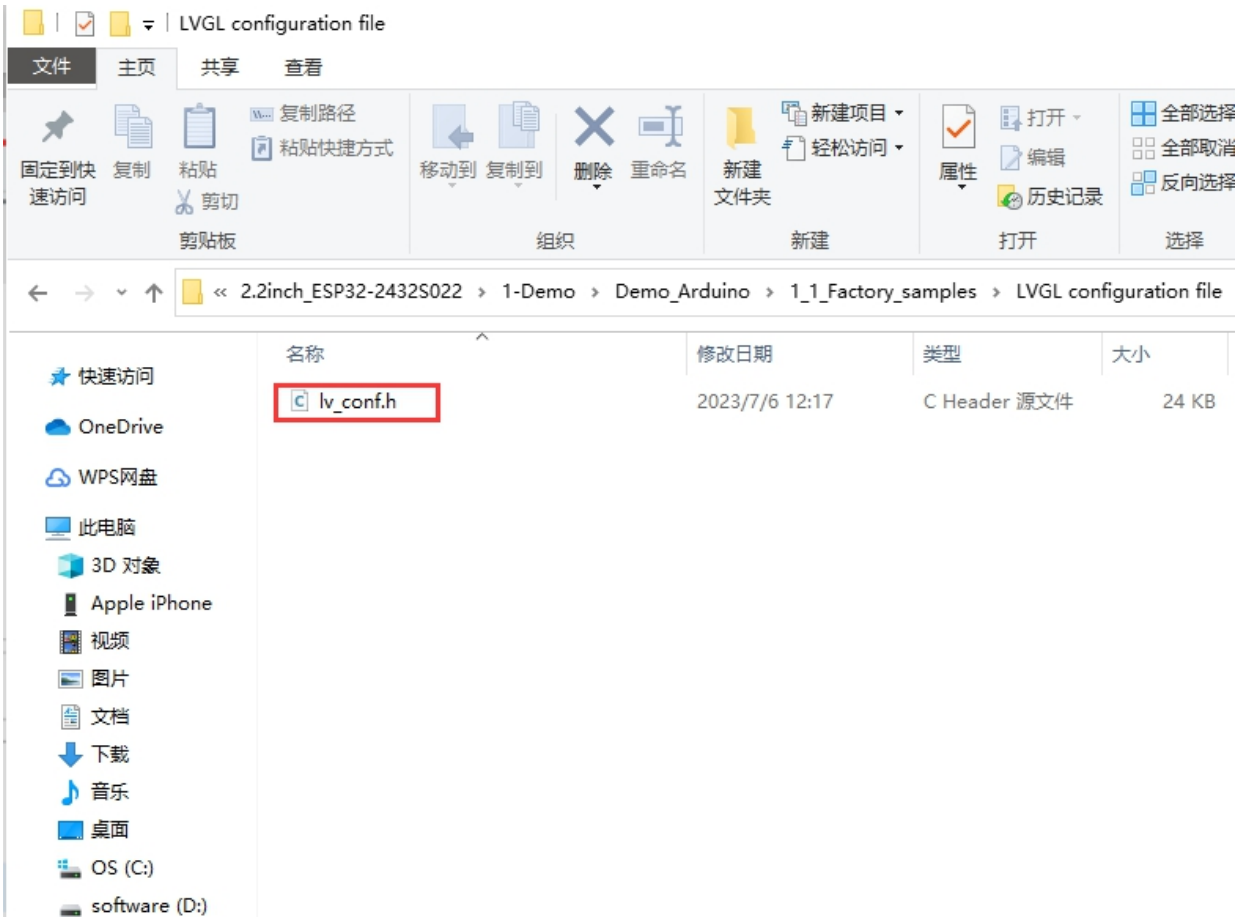
Download library files .

Lvgl



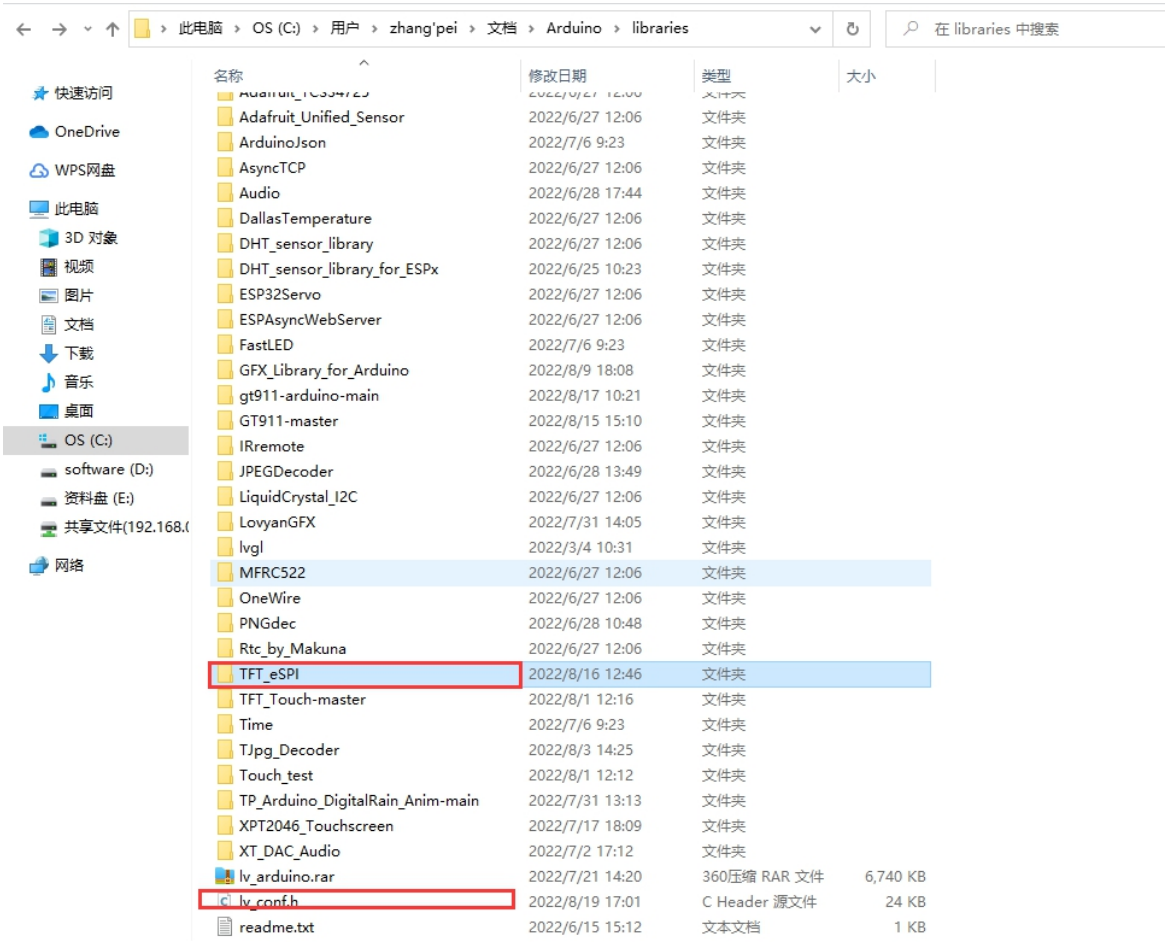
Copy the data center's lv_conf.h

As shown in the figure:



Put this file under the arduino library file, it must be in the same root directory as the library TFT_eSPI .

As shown:



After compiling, you can run LVGL and touch normally.