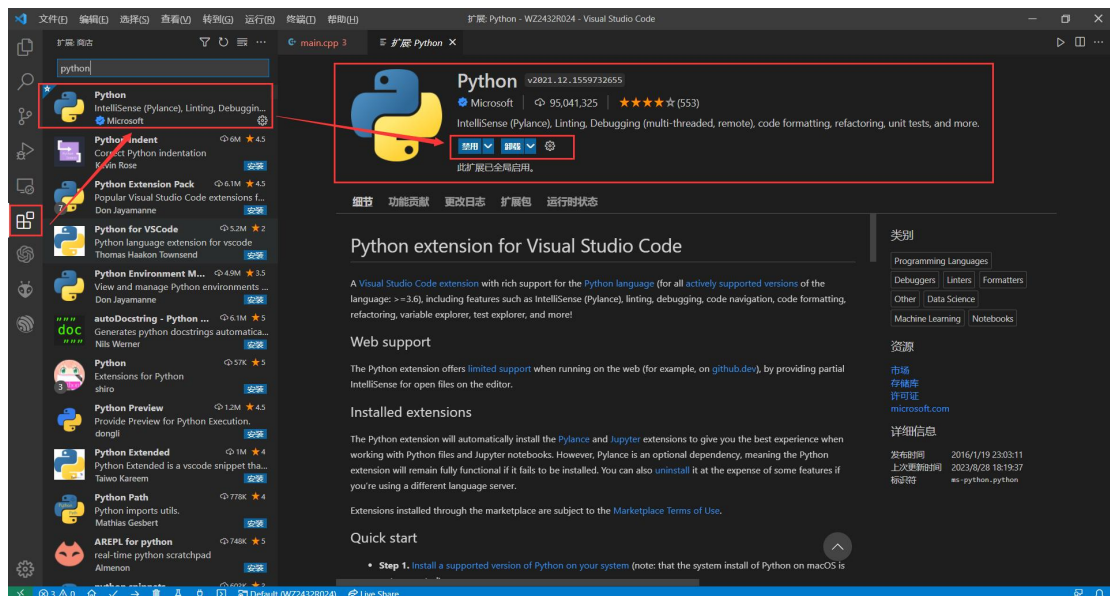


PlatformIO

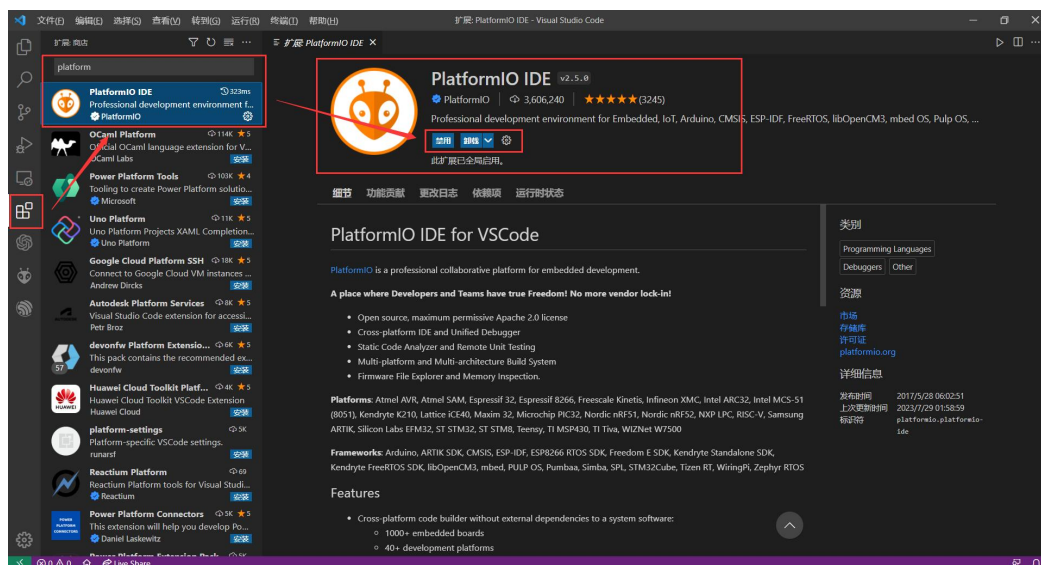
WZ2432R024 or WZ2432R028 or WZ2432R035

Take the WZ2432R024 as an example

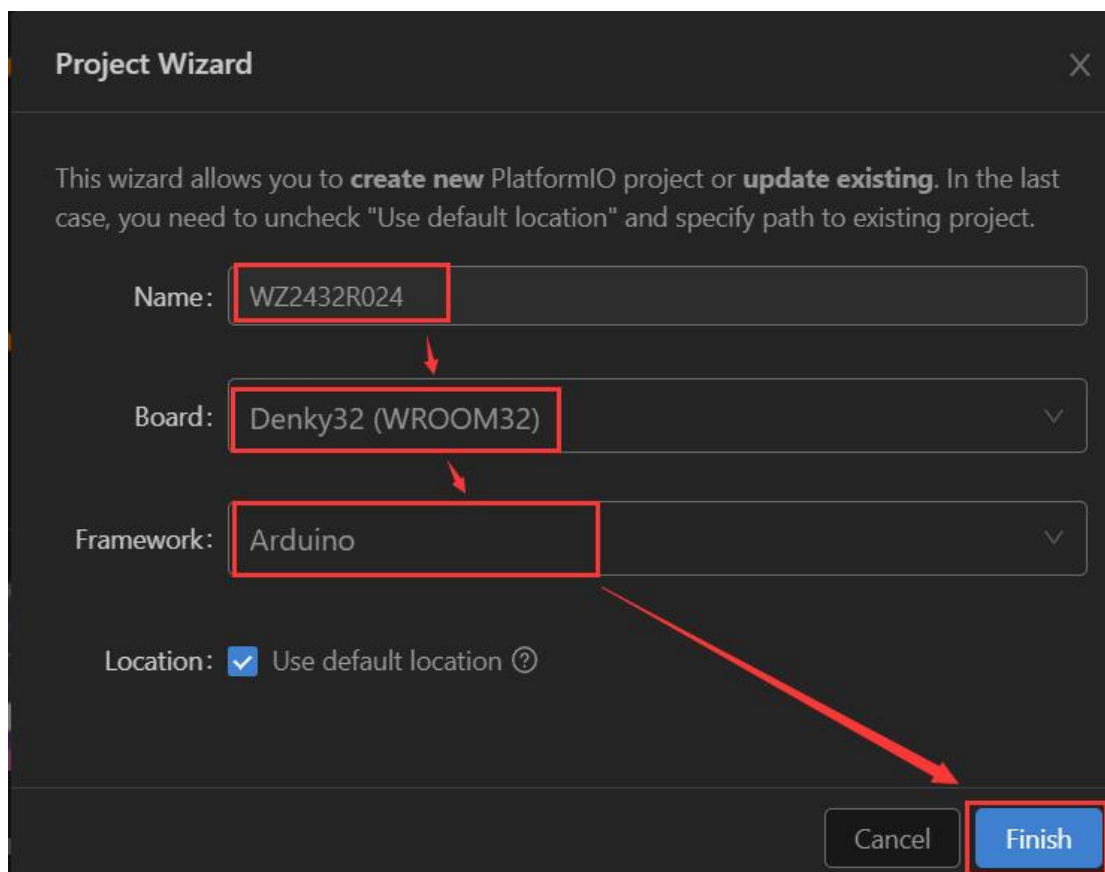
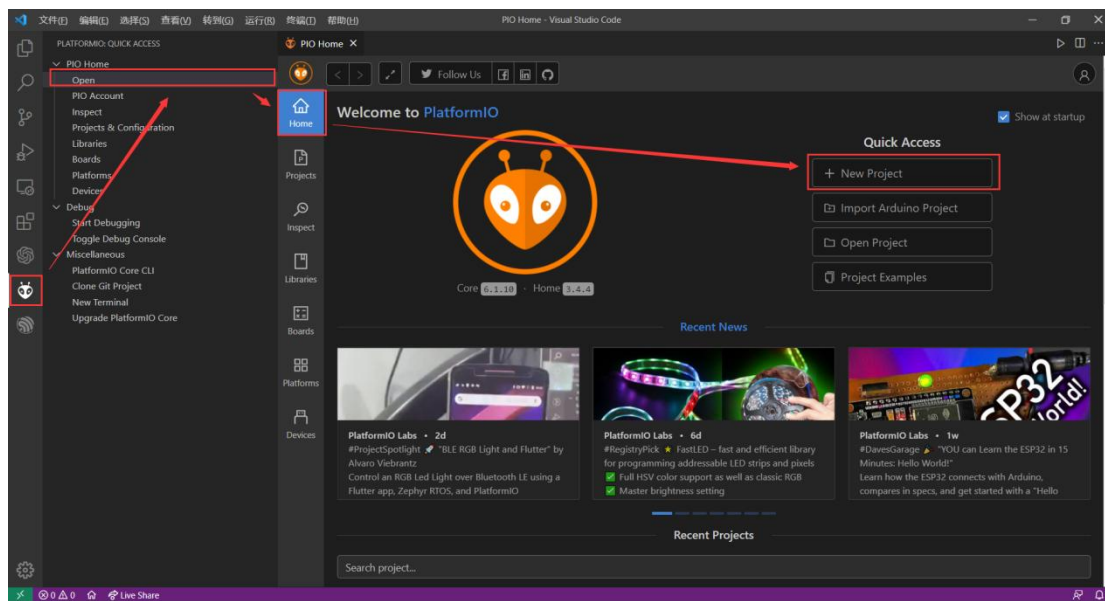
First open the VScode to check if the python is installed

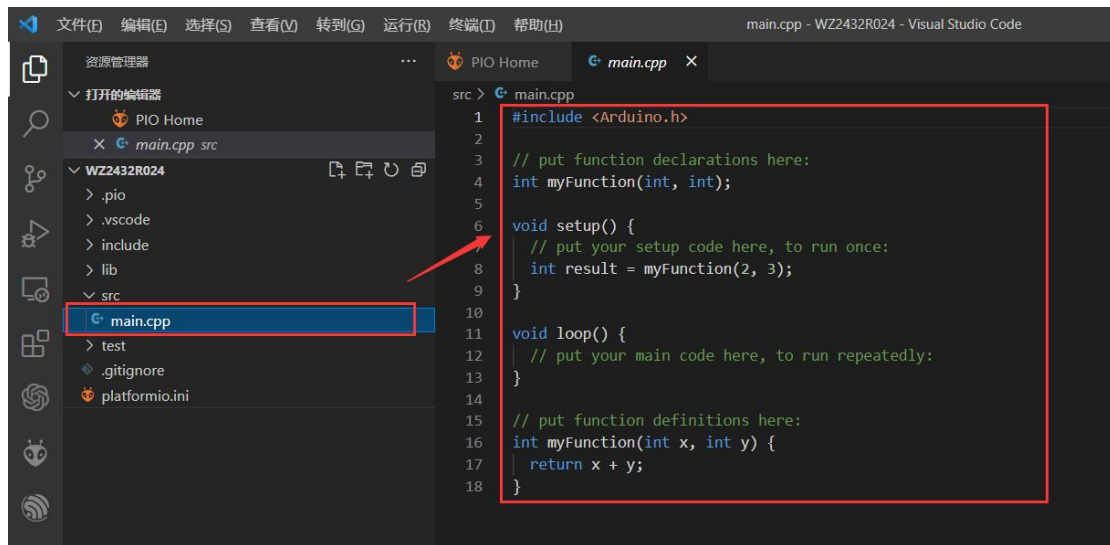


Open the VScode to download the PlatformIO

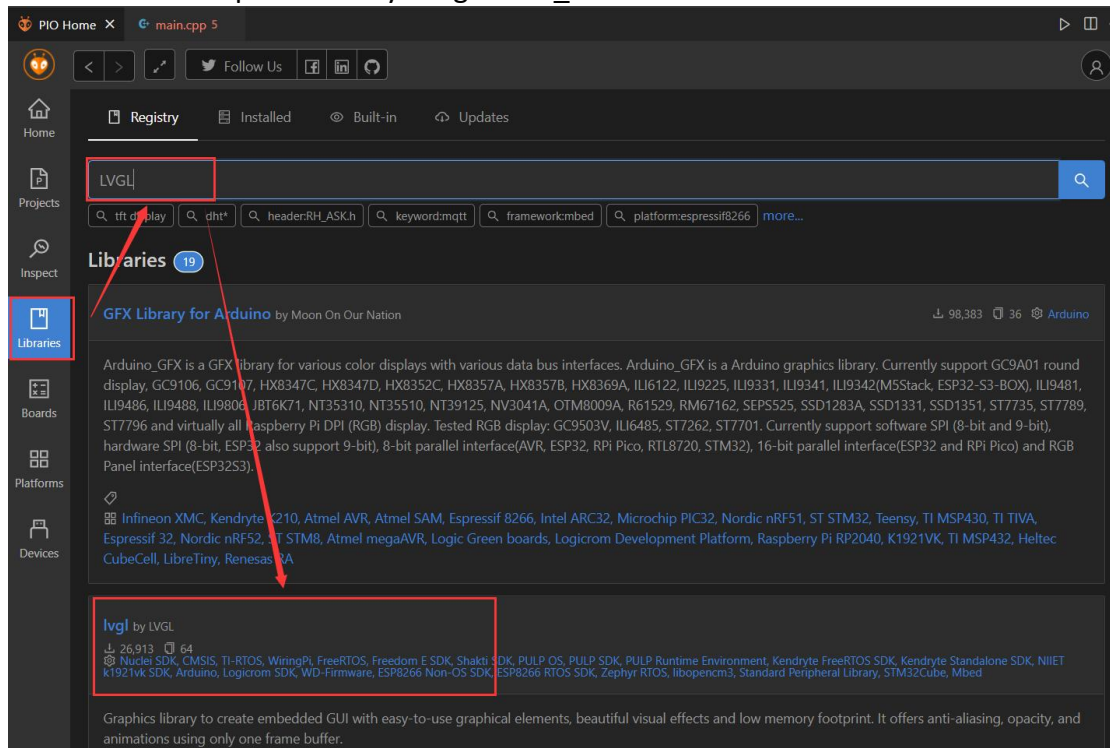


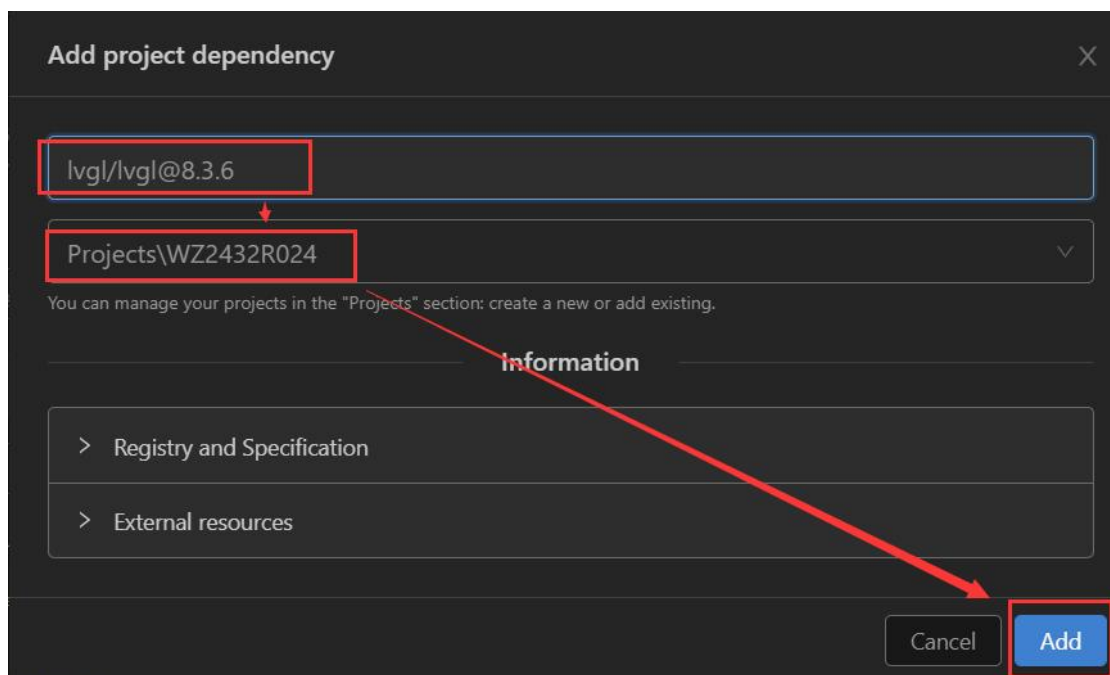
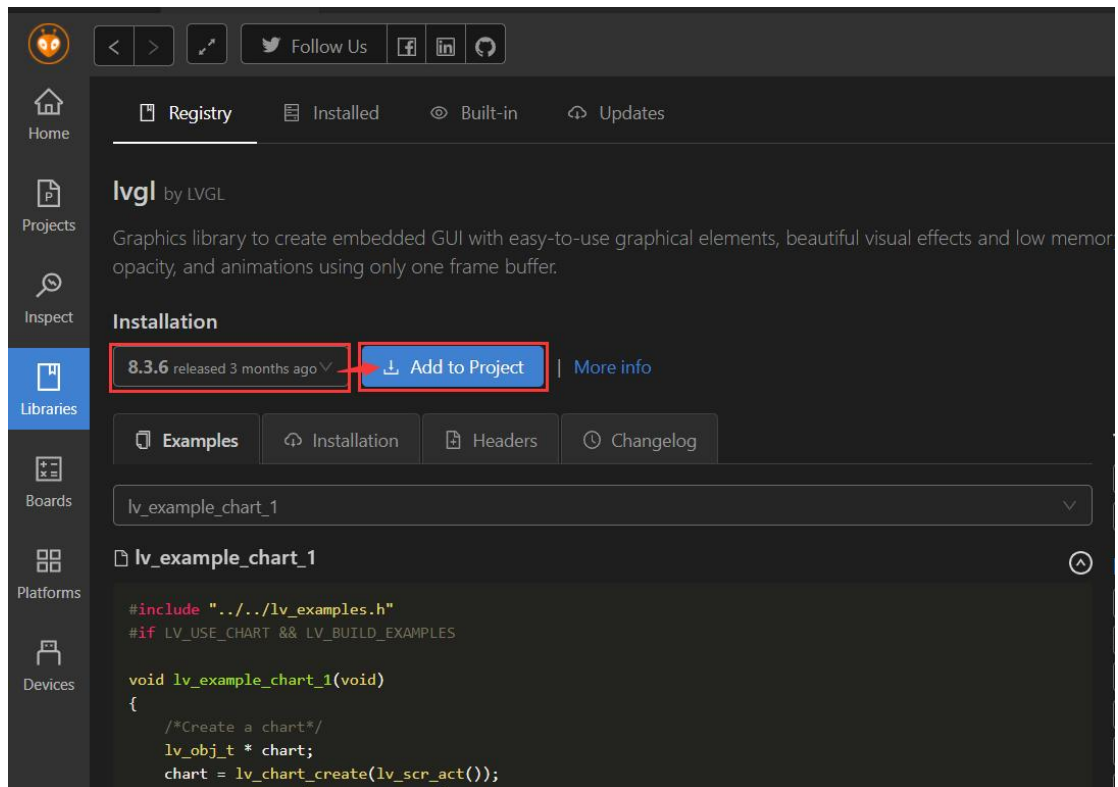
Create new projects





Download the required library (lvgl、TFT_eSPI)





The screenshot shows the Arduino IDE Libraries Manager interface. The search bar at the top contains the text 'TFT'. Below the search bar, a list of libraries is displayed. The first library is 'GFX Library for Arduino' by Moon On Our Nation, which has 98,383 downloads and 36 stars. The second library is 'TFT_eSPI' by Bodmer, which has 80,566 downloads and 152 stars. A red box highlights the 'TFT_eSPI' library, and a red arrow points from the search bar to it. The 'Libraries' tab is selected in the left sidebar.

Registry Installed Built-in Updates

TFT

tft-display dht* header.RH_ASK.h keyword.mqtt frameworkimbed platform.espressif8266 more...

Libraries 371

GFX Library for Arduino by Moon On Our Nation 98,383 36 Arduino

Arduino_GFX is a GFX library for various color displays with various data bus interfaces. Arduino_GFX is a Arduino graphics library. Currently support GC9A01 round display, GC9106, GC9107, HX8347C, HX8347D, HX8352C, HX8357A, HX8357B, HX8369A, ILI6122, ILI9225, ILI9331, ILI9341, ILI9342(M5Stack, ESP32-S3-BOX), ILI9481, ILI9486, ILI9488, ILI9806, JBT6K71, NT35310, NT35510, NT39125, NV3041A, OTM8009A, R61529, RM67162, SEPS525, SSD1283A, SSD1331, SSD1351, ST7735, ST7789, ST7796 and virtually all Raspberry Pi DPI (RGB) display. Tested RGB display: GC9503V, ILI6485, ST7262, ST7701. Currently support software SPI (8-bit and 9-bit), hardware SPI (8-bit, ESP32 also support 9-bit), 8-bit parallel interface(AVR, ESP32, RPi Pico, RTL8720, STM32), 16-bit parallel interface(ESP32 and RPi Pico) and RGB Panel interface(ESP32S3).

Infineon XMC, Kendryte K210, Atmel AVR, Atmel SAM, Espressif 8266, Intel ARC32, Microchip PIC32, Nordic nRF51, ST STM32, Teensy, TI MSP430, TI TIVA, Espressif 32, Nordic nRF52, ST STM8, Atmel megaAVR, Logic Green boards, Logicroom Development Platform, Raspberry Pi RP2040, K1921VK, TI MSP432, Heltec CubeCell, Libreminy, Renesas RA

TFT_eSPI by Bodmer 80,566 152 Arduino

A TFT and ePaper (SPI or parallel interface) graphics library with optimisation for Raspberry Pi Pico, RP2040, ESP8266, ESP32 and STM32 processors

arduino, tft, display, ttgo, lilyp, wt32-sc01, epaper, pico, rp2040 nano connect, rp2040, stm32, esp8266, nodemcu, esp32, m5stack, ili9341, st7735, ili9163, s6d02a1, ili9481

The screenshot shows the details of the 'TFT_eSPI' library by Bodmer. The version '2.5.31' is highlighted, along with the 'Add to Project' button. The 'Installation' tab is selected, showing the installation instructions. The code for the library is displayed in a text area.

Registry Installed Built-in Updates

TFT_eSPI by Bodmer

A TFT and ePaper (SPI or parallel interface) graphics library with optimisation for Raspberry Pi Pico, RP2040, ESP8266,

Installation

2.5.31 released about a month ago Add to Project More info

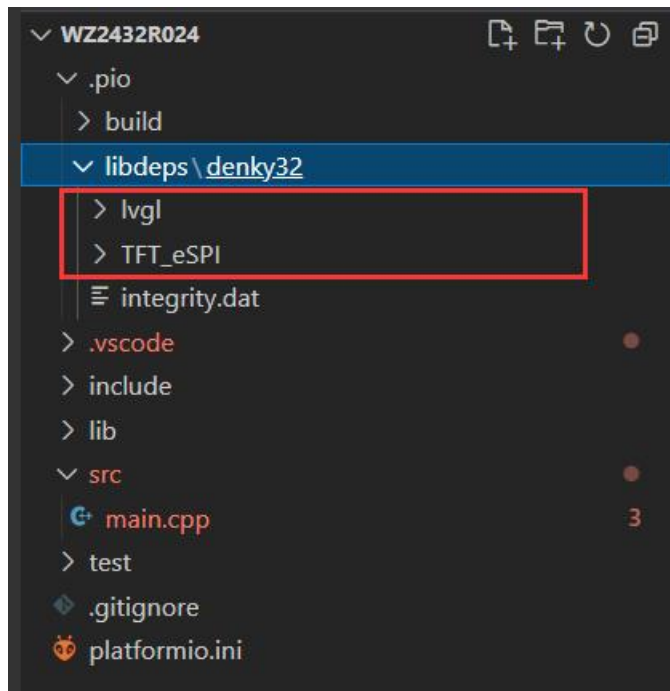
Examples Installation Headers Changelog

defaultEye

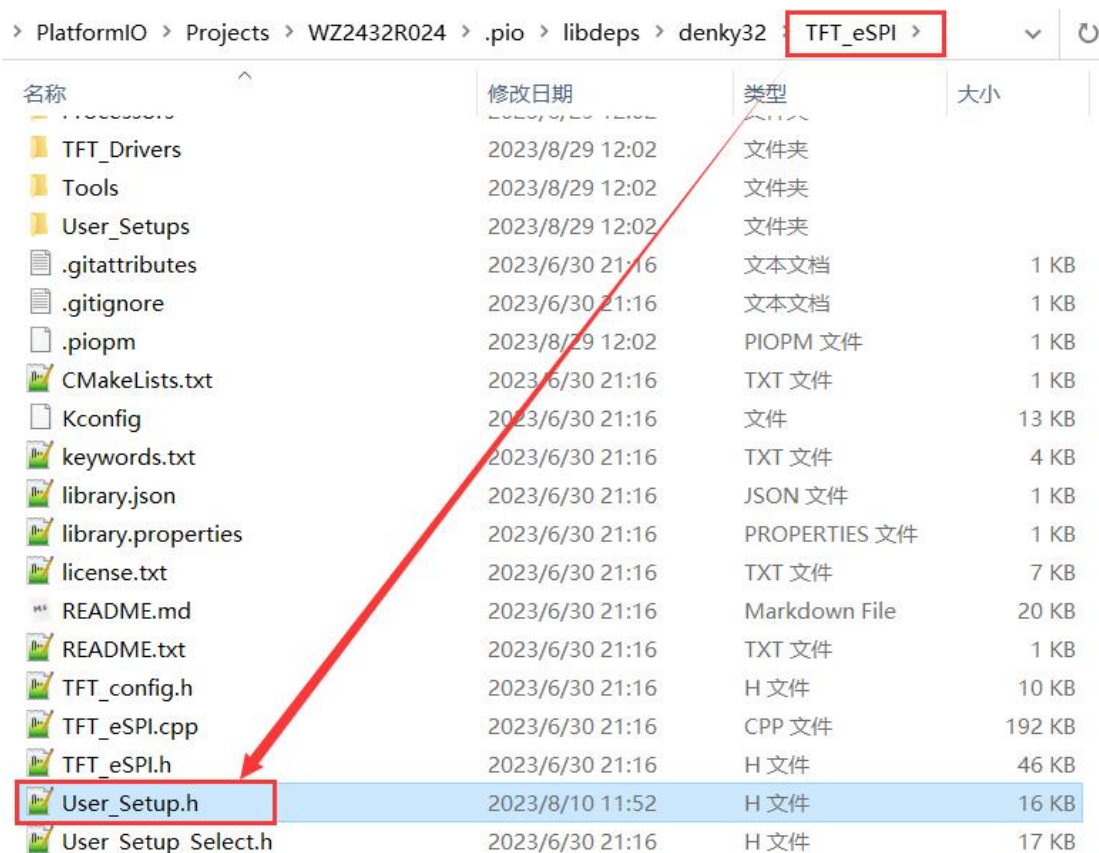
defaultEye

```
#define SCLERA_WIDTH 200
#define SCLERA_HEIGHT 200

const uint16_t sclera[SCLERA_HEIGHT * SCLERA_WIDTH] PROGMEM= {
    0X6901, 0X6901, 0X6901, 0X6901, 0X6901, 0X6901, 0X6901, 0X6901,
    0X6901, 0X6901, 0X6901, 0X6901, 0X6901, 0X6901, 0X6901, 0X6901,
    0X6901, 0X6901, 0X6901, 0X6901, 0X6901, 0X6901, 0X6901, 0X6901,
    0X6901, 0X6901, 0X6901, 0X6901, 0X6901, 0X6901, 0X6901, 0X6901,
    0X6901, 0X6901, 0X6901, 0X6901, 0X6901, 0X6901, 0X6901, 0X6901,
    0X6901, 0X6901, 0X6901, 0X6901, 0X6901, 0X6901, 0X6901, 0X6901,
    0X6901, 0X6901, 0X6901, 0X6901, 0X6901, 0X6901, 0X6901, 0X6901,
    0X6901, 0X6901, 0X6901, 0X6901, 0X6902, 0X6902, 0X6902, 0X7102,
    0X7122, 0X7122, 0X7122, 0X7122, 0X7122, 0X7142, 0X7142, 0X7142,
    0X7142, 0X7142, 0X7143, 0X7162, 0X7963, 0X7963, 0X7963, 0X7963,
```

Next, we want to configure the TFT _ eSPI library to replace the User_Setup.h in the library with the file we provide



Next, we want to configure the lvgl library, right-click to open the folder directory, and put the demo and examples folders into the src folder!



文档 > PlatformIO > Projects > WZ2432R024 > .pio > libdeps > denky32 > lvgl					
名称	修改日期	类型	大小		
.github	2023/8/29 11:55	文件夹			
demos	2023/8/29 11:55	文件夹			
env_support	2023/8/29 11:55	文件夹			
examples	2023/8/29 11:55	文件夹			
scripts	2023/8/29 11:55	文件夹			
src	2023/8/29 11:55	文件夹			
.codecov.yml	2023/4/3 16:35	YML 文件	1 KB		
.editorconfig	2023/4/3 16:35	EDITORCONFIG ...	1 KB		
.git	2022/8/8 19:00	GIT 文件	1 KB		
.gitignore	2023/5/3 15:55	文本文档	1 KB		
.piopm	2023/8/29 11:55	PIOPM 文件	1 KB		
.pre-commit-config.yaml	2023/5/3 15:55	YAML 文件	1 KB		
CMakeLists.txt	2022/12/4 0:42	TXT 文件	1 KB		
component.mk	2023/4/3 16:35	Makefile	2 KB		
idf_component.yml	2023/5/4 16:31	YML 文件	1 KB		
Kconfig	2023/5/3 15:55	文件	42 KB		
library.json	2023/5/4 16:32	JSON 文件	1 KB		
library.properties	2023/5/4 16:32	PROPERTIES 文件	1 KB		
LICENCE.txt	2022/12/4 0:42	TXT 文件	2 KB		

PlatformIO > Projects > WZ2432R024 > .pio > libdeps > denky32 > lvgl > src >

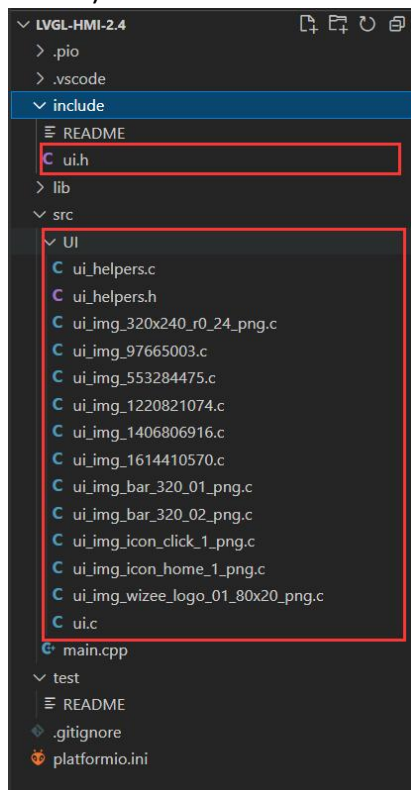
名称	修改日期	类型	大小
core	2023/8/29 11:55	文件夹	
demos	2023/8/29 12:20	文件夹	
draw	2023/8/29 11:55	文件夹	
examples	2023/8/29 12:20	文件夹	
extra	2023/8/29 11:55	文件夹	
font	2023/8/29 11:55	文件夹	
hal	2023/8/29 11:55	文件夹	
misc	2023/8/29 11:55	文件夹	
widgets	2023/8/29 11:55	文件夹	
lv_api_map.h	2023/4/3 16:35	H 文件	2 KB
lv_conf_internal.h	2023/5/4 16:32	H 文件	74 KB
lv_conf_kconfig.h	2023/5/3 15:55	H 文件	7 KB
lvgl.h	2023/4/3 16:35	H 文件	1 KB

Place the lv_conf.h file under this directory again

文档 > PlatformIO > Projects > WZ2432R024 > .pio > libdeps > denky32 >

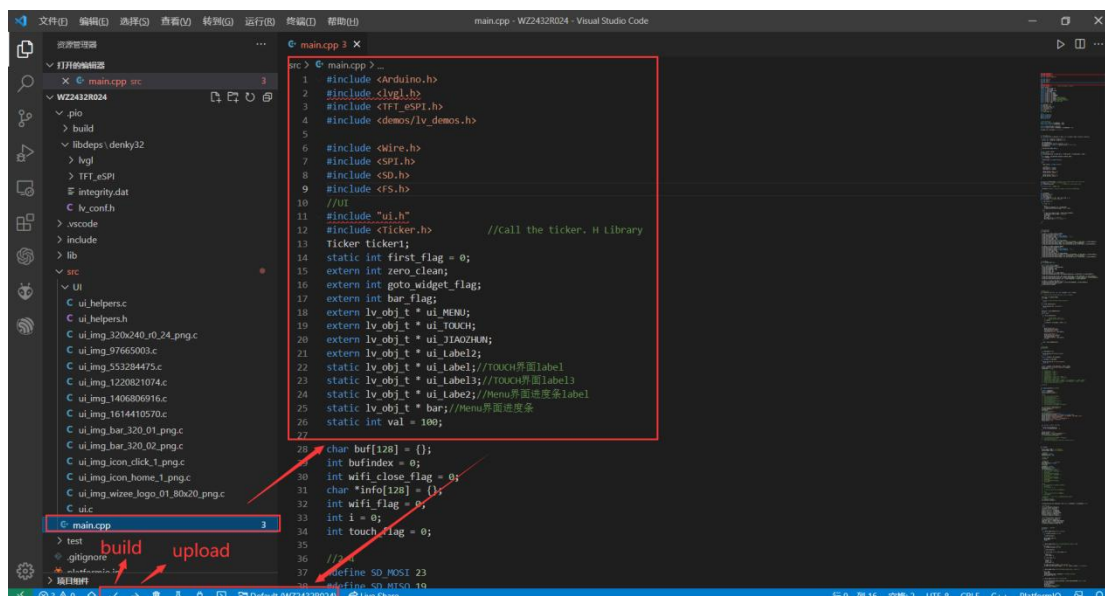
名称	修改日期	类型	大小
lvgl	2023/8/29 11:55	文件夹	
TFT_eSPI	2023/8/29 12:02	文件夹	
integrity.dat	2023/8/29 12:02	DAT 文件	1 KB
lv_conf.h	2023/8/25 17:25	H 文件	26 KB

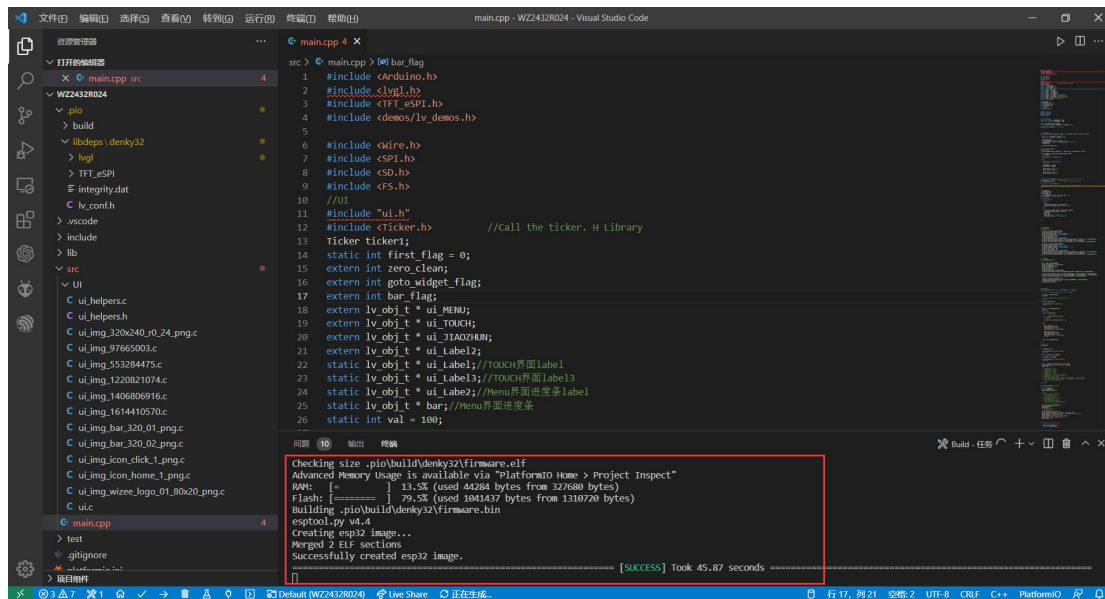
Next, let's configure our own UI files (the UI files are generated from the SquareLine Studio)



In the UI folder that will be generated. The c file is placed in the /src folder, and in the generated UI folder. Place the h file in the /include folder

At this time, we will complete all the configuration, write the code and start compiling the program





Next we began to burn the program, finished!

