ESP32-S3-GEEK

From Waveshare Wiki
Jump to: navigation, search

Overview

ESP32-S3-GEEK

(https://www.waveshare.com/esp32-s3-geek.htm)

ESP32-S3
I2C, UART, USB-A

Introduction

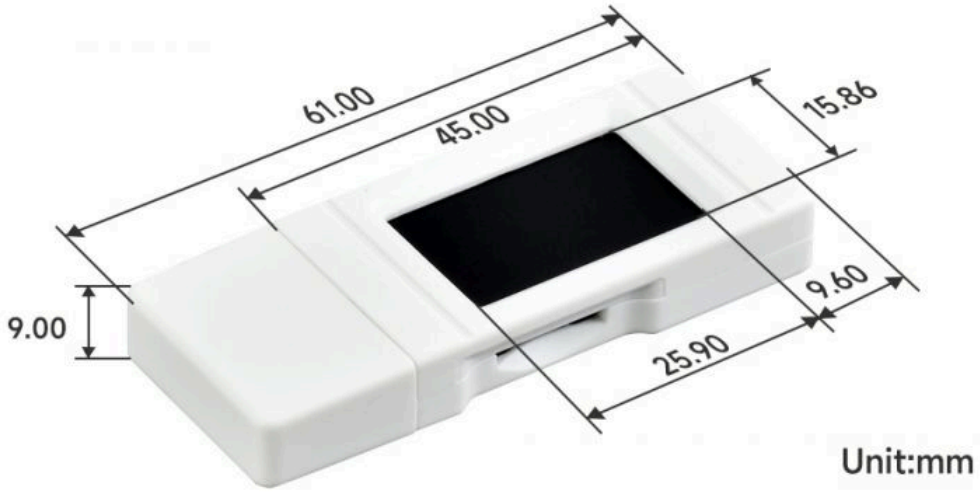
ESP32-S3-GEEK is a geek development board designed by Waveshare, onboard USB-A port, 1.14-inch LCD screen, a TF card slot, and other peripherals. Supports 2.4GHz WiFi and BLE 5, with integrated 16MB Flash & 2MB PSRAM, provides I2C port, UART port, and GPIO header, bringing more possibilities for your project.

Features

- Adopts **ESP32-S3R2** chip with Xtensa® 32-bit LX7 dual-core processor, capable of running at **240 MHz**.
- Built-in 512KB SRAM, 384KB ROM, 2MB of on-chip PSRAM, and onboard **16MB Flash memory**.
- Onboard 1.14inch 240×135 pixels 65K color **IPS LCD** display.
- Integrated **2.4GHz WiFi and Bluetooth LE** wireless communication.
  - WiFi supports Infrastructure BSS in Station, SoftAP, and Station + SoftAP modes.
  - WiFi supports 1T1R mode with a data rate of up to 150 Mbps.
  - Bluetooth supports high power mode (20dBm).
  - Internal co-existence mechanism between Wi-Fi and Bluetooth to share the same antenna.
- The onboard 3-pin **UART** interface can function as a USB-to-serial adapter.
- The onboard 3-pin **GPIO** interface can be used for debugging other modules or for ADC data collection.
- The onboard 4-pin **I2C** interface is available for testing the target board.
- Equipped with plastic case and cables.

- Provides online open-source demos and resources, more convenient for learning and development.

Dimensions



(/wiki/File:RP2-4-_geek_09.jpg)

Development Environment Setup

- **The following development system is Windows by default.**

ESP-IDF

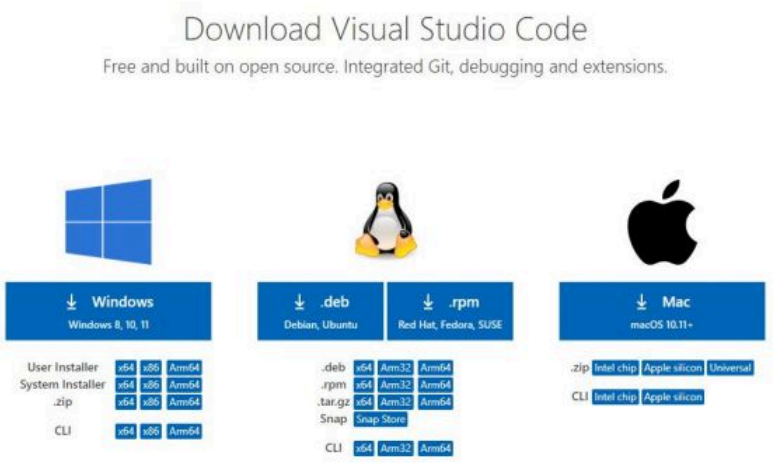
[Collapse]

- It is recommended to develop with the VSC plug-in.

Develop with VSCode Plug-in

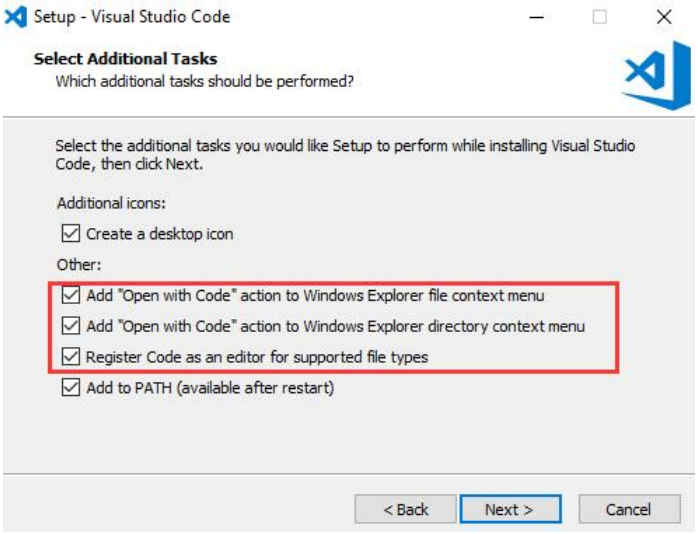
Install VSCode

1. Open the download page (https://code.visualstudio.com/download) of the official VSCode website, and select the corresponding system and system bit to download.



(/wiki/File:ESP32-S3-Pico_05.jpg)

2. After running the installation package, the rest can be installed by default, but here for the subsequent experience, it is recommended to check boxes 1, 2, and 3.



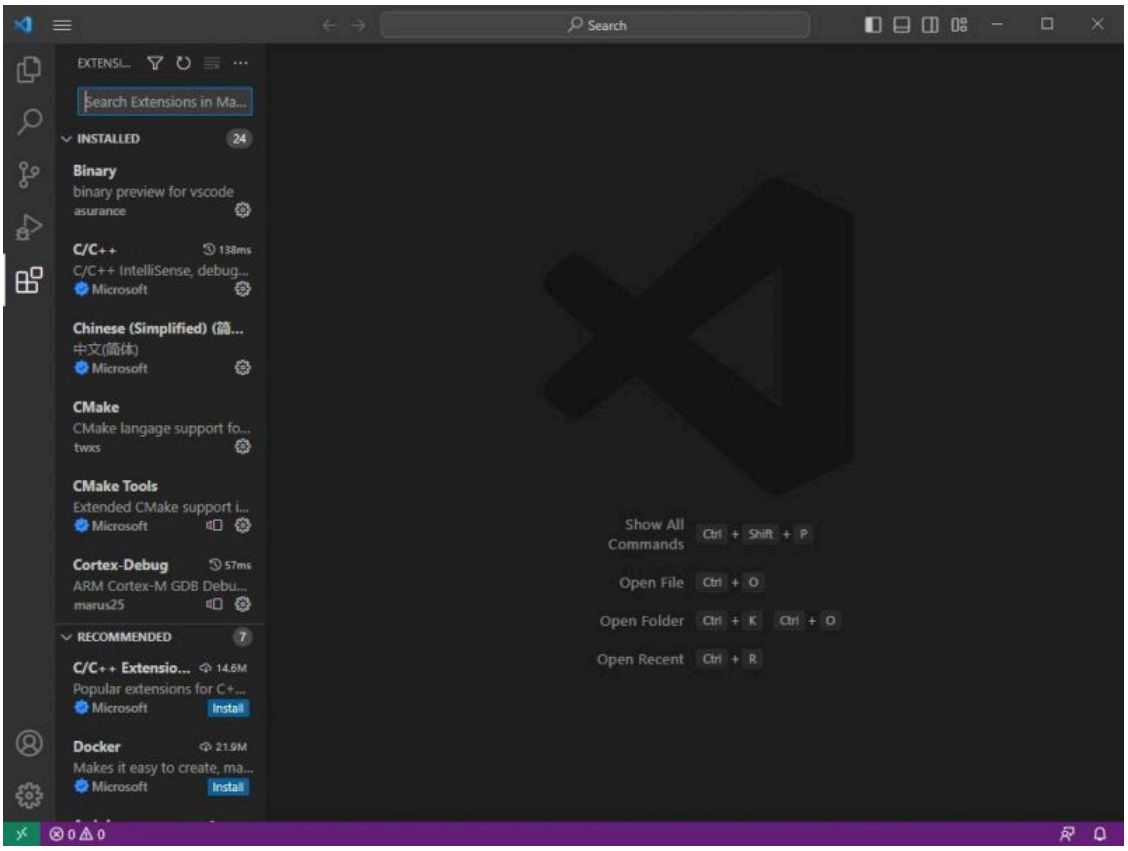
(/wiki/File:ESP32-S3-Pico_06.jpg)

- After the first and second items are enabled, you can open VSCode directly by right-clicking files or directories, which can improve the subsequent user experience.
- After the third item is enabled, you can select VSCode directly when you choose how to open it.

Install Espressif IDF Plug-in

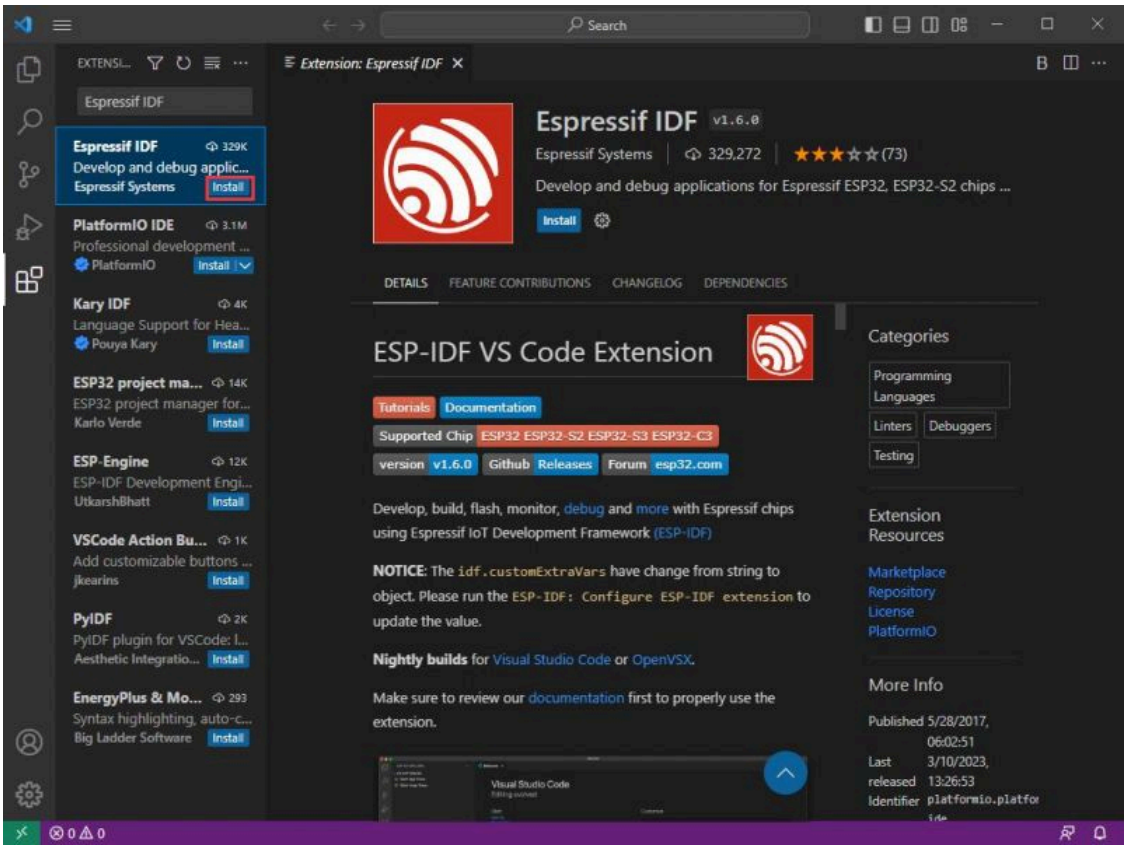
- Note: The latest version of the current plug-in is V1.6.0, for a consistent experience, users can choose the same version as us.

1. Open VSCode and use the shortcut key Shift + Ctrl + X to enter the plugin manager.



(/wiki/File:ESP32-S3-Pico_07.jpg)

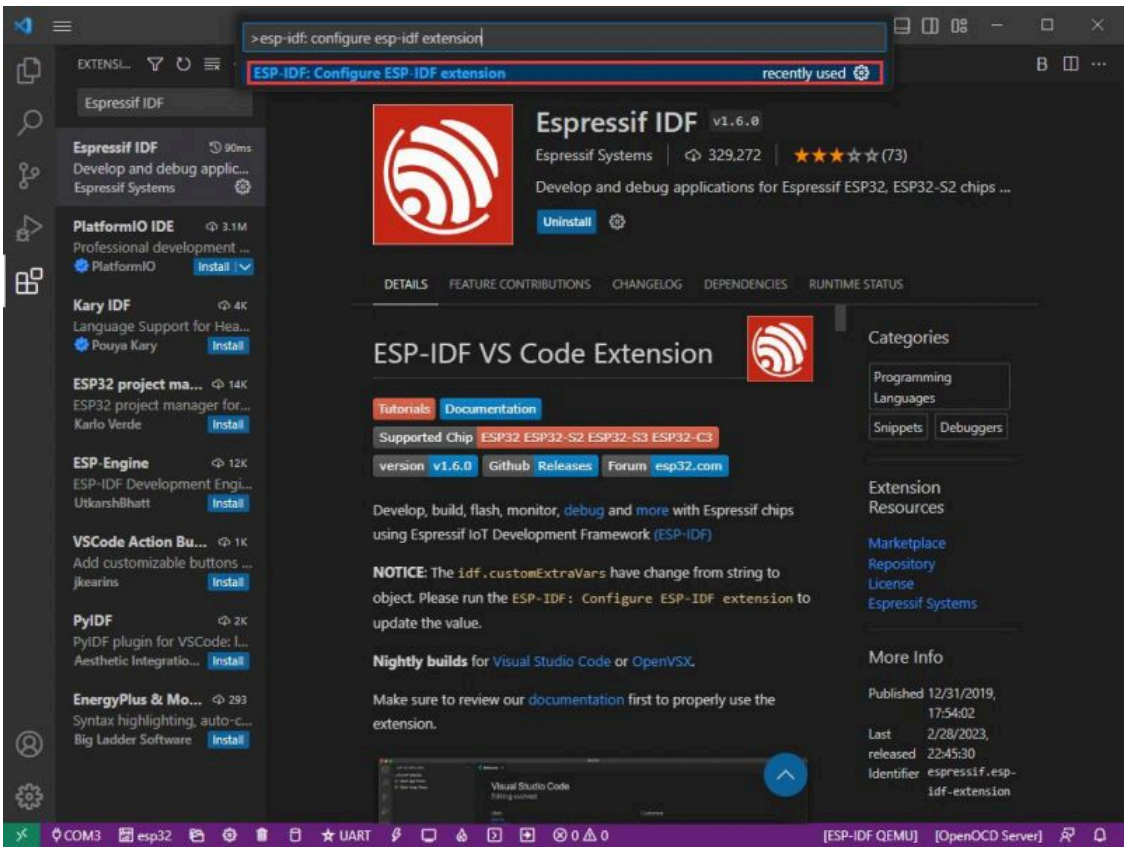
2. In the search bar, type Espressif IDF, select the corresponding plug-in, and click install.



(/wiki/File:ESP32-S3-Pico_08.jpg)

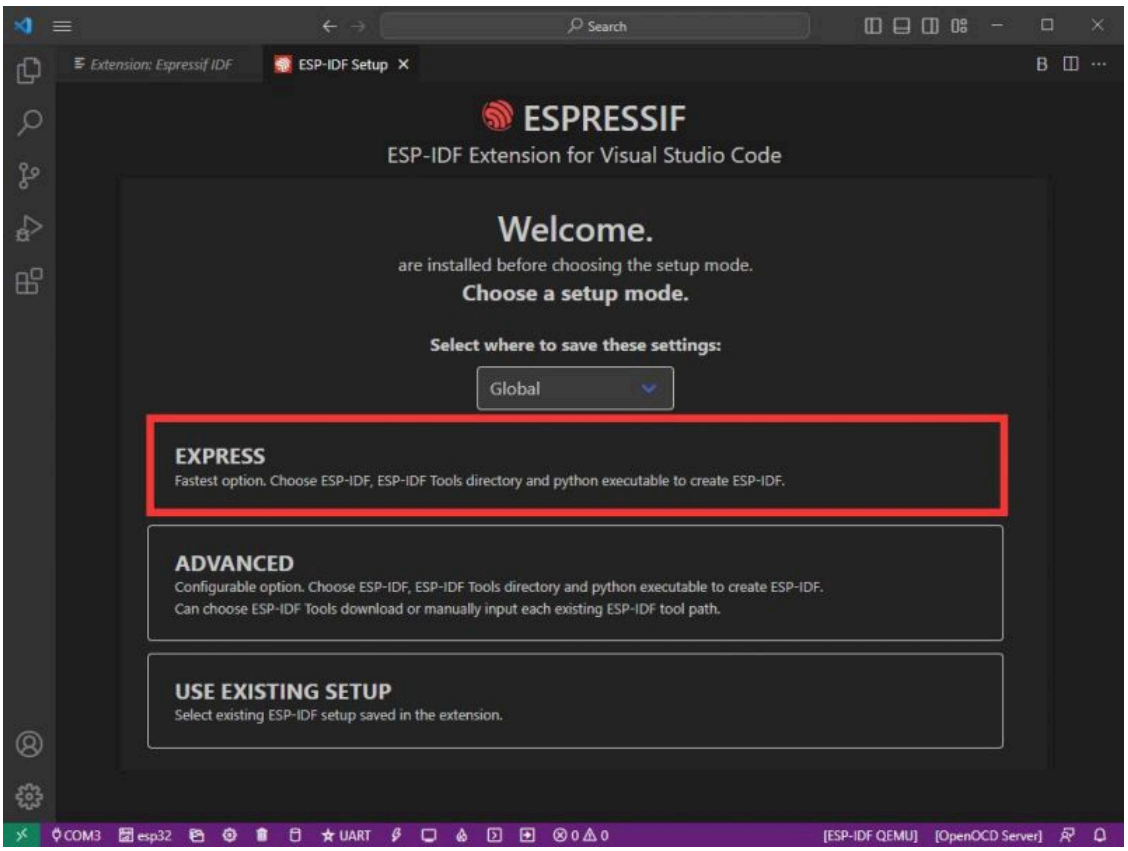
3. Press F1 to enter:

```
esp-idf: configure esp-idf extension
```



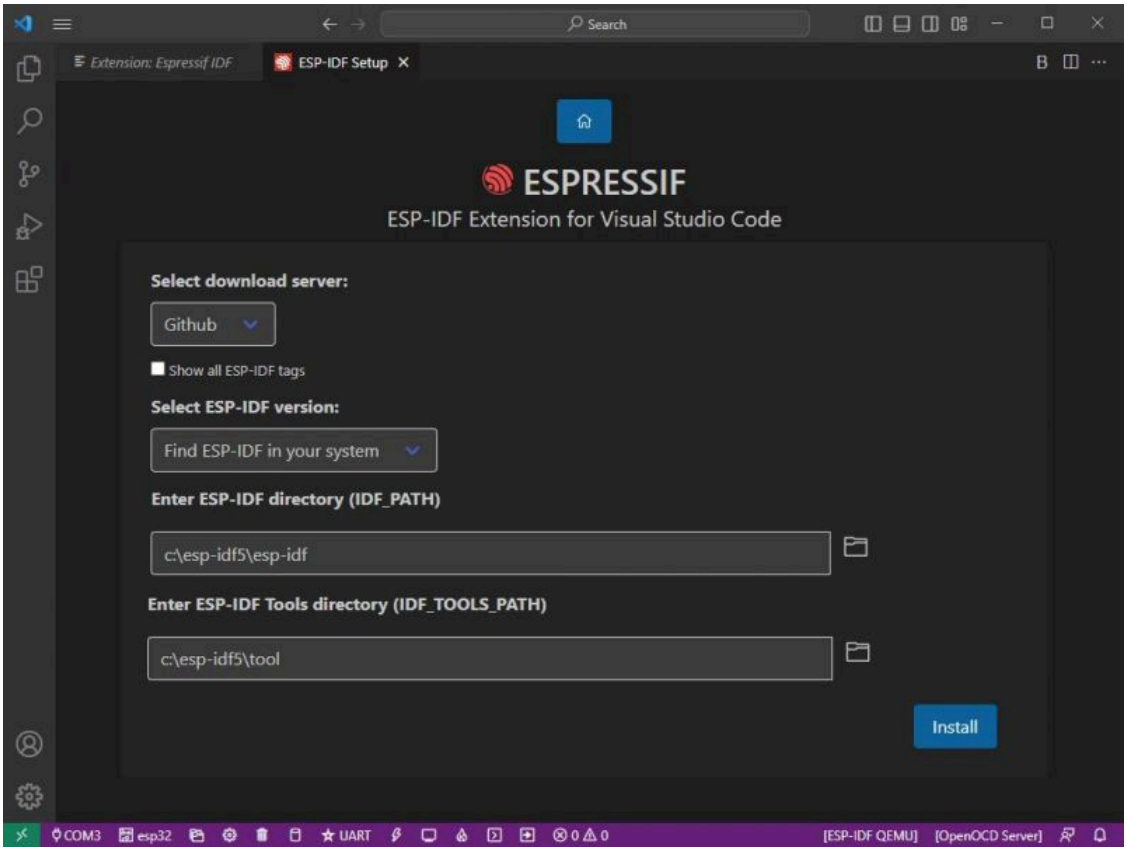
(/wiki/File:ESP32-S3-Pico_09.jpg)

4. Choose express (This tutorial is for first-time users, so only the first general installation tutorial is covered.)



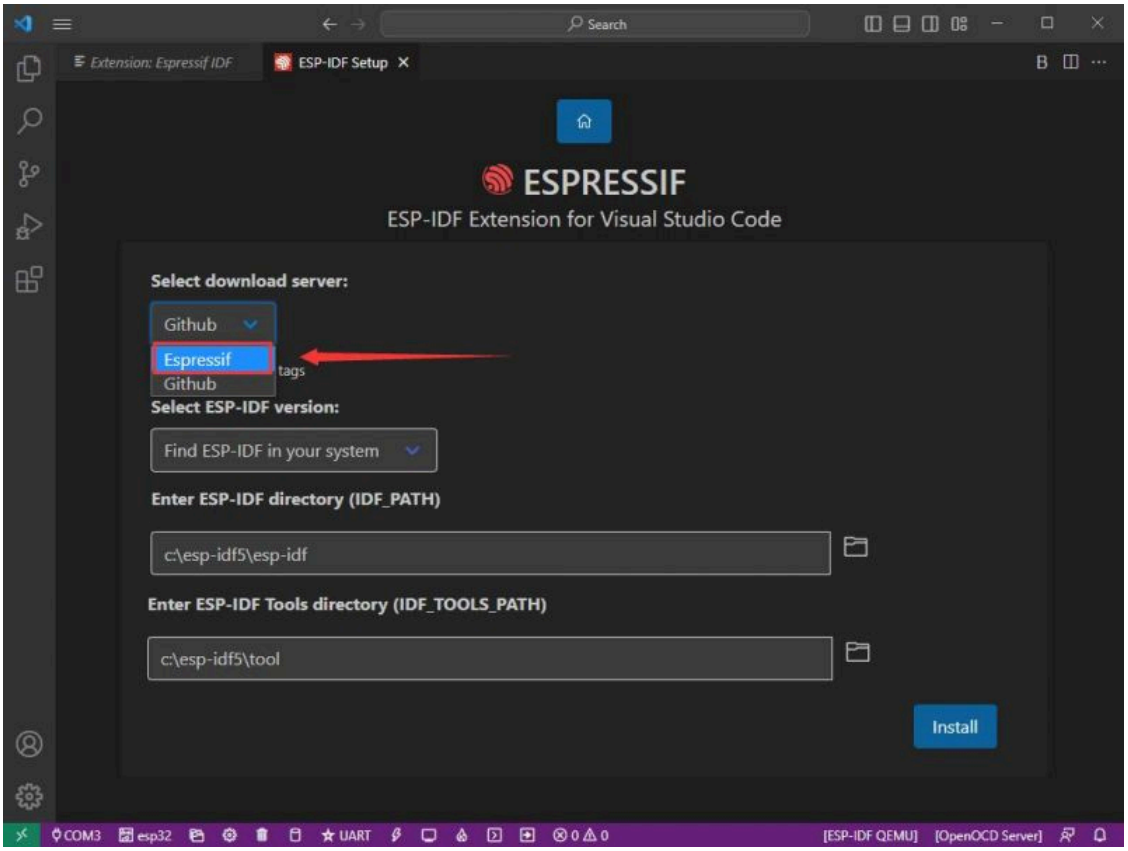
(/wiki/File:ESP32-S3-Pico_10.jpg)

5. Open and display this screen.



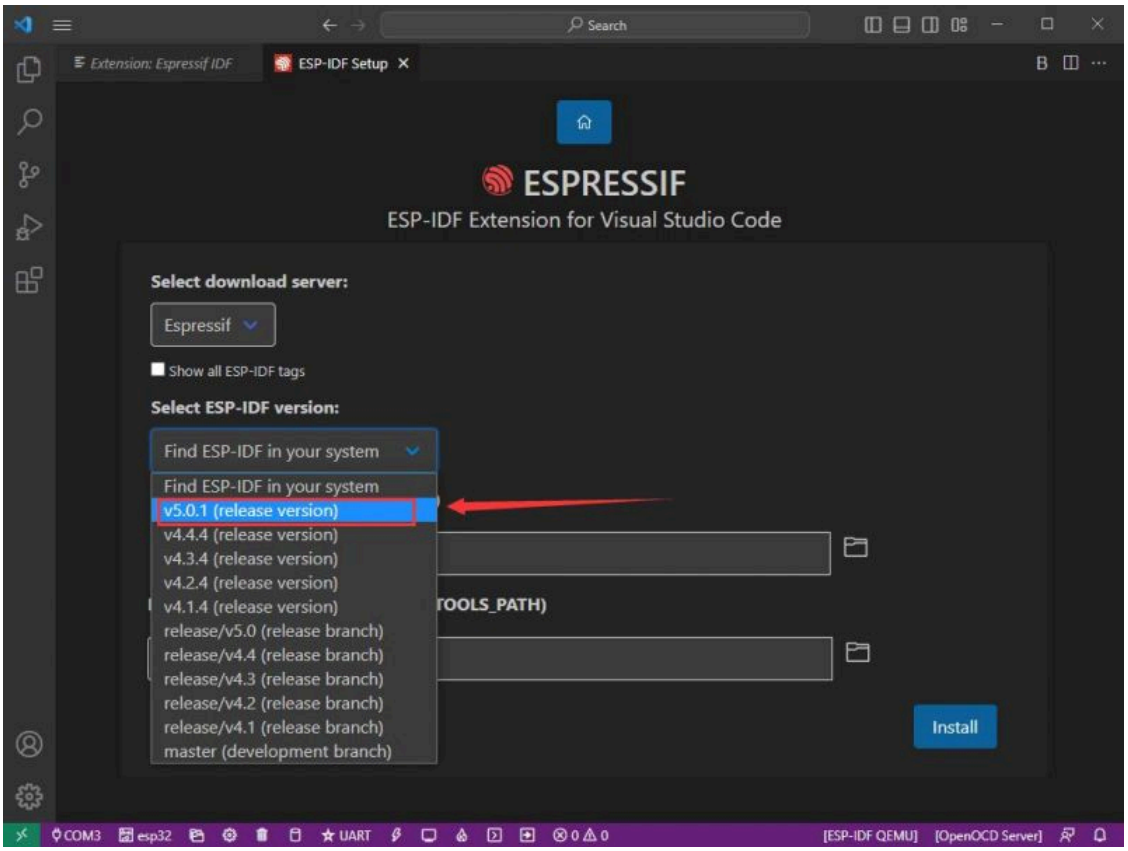
(/wiki/File:ESP32-S3-Pico_11.jpg)

6. Choose a server to download.



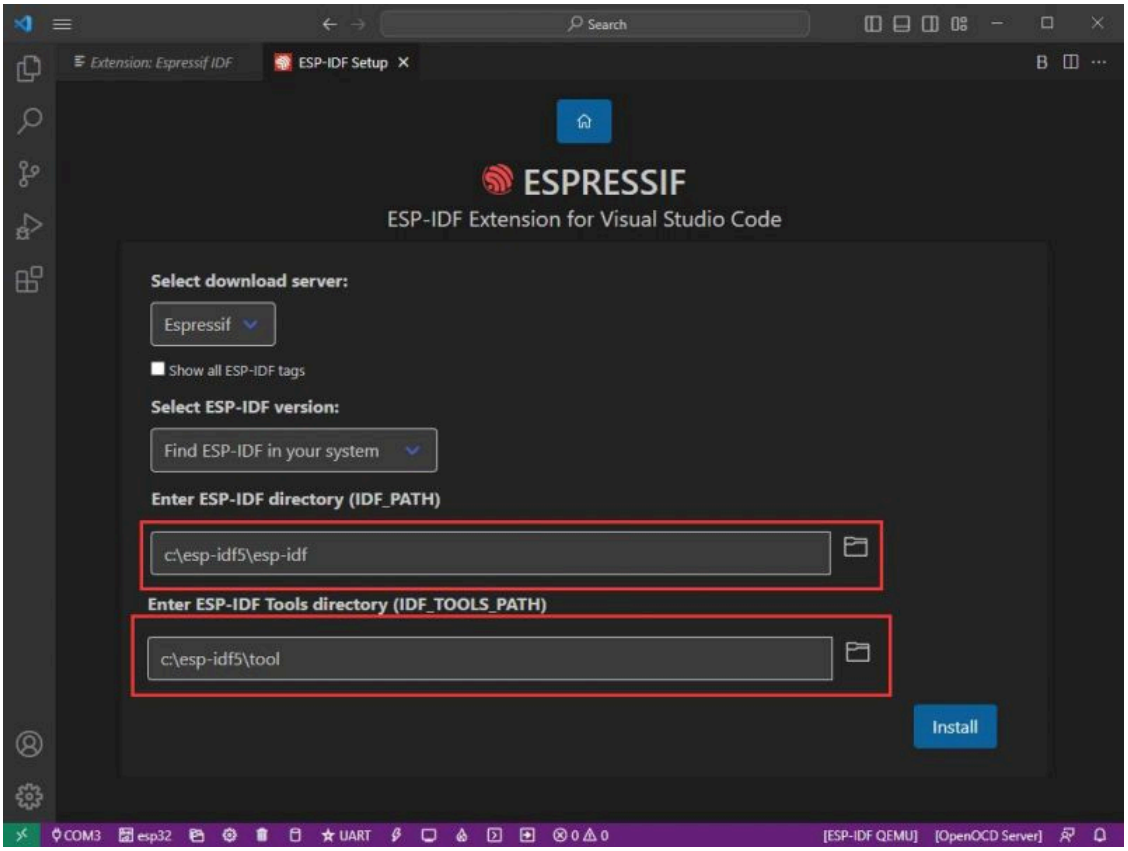
(/wiki/File:ESP32-S3-Pico_12.jpg)

7. Select the ESP-IDF version you want now, we choose the latest V5.0.1 (note that ESP-IDF started to support ESP32-S3 only after V4.4).



(/wiki/File:ESP32-S3-Pico_13.jpg)

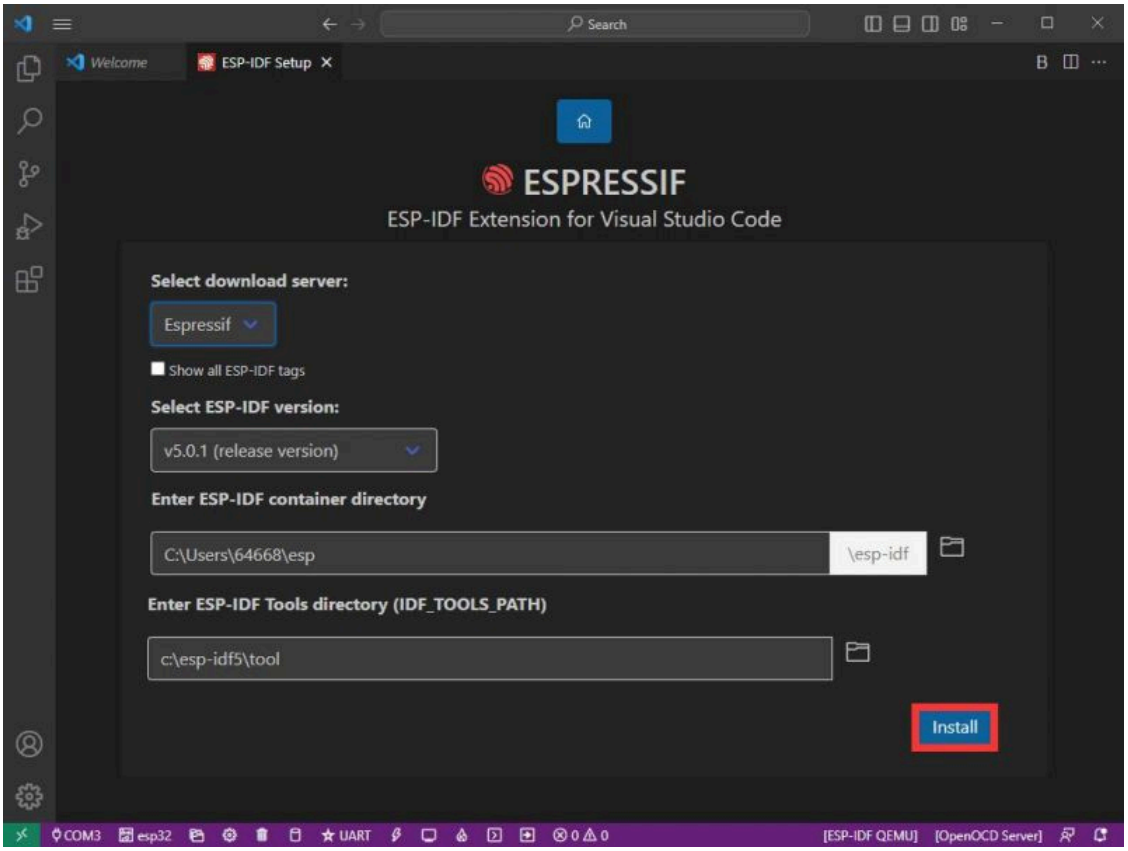
8. The following two are the ESP-IDF directory installation address and the ESP-IDF required tools installation address respectively.



(/wiki/File:ESP32-S3-Pico_14.jpg)

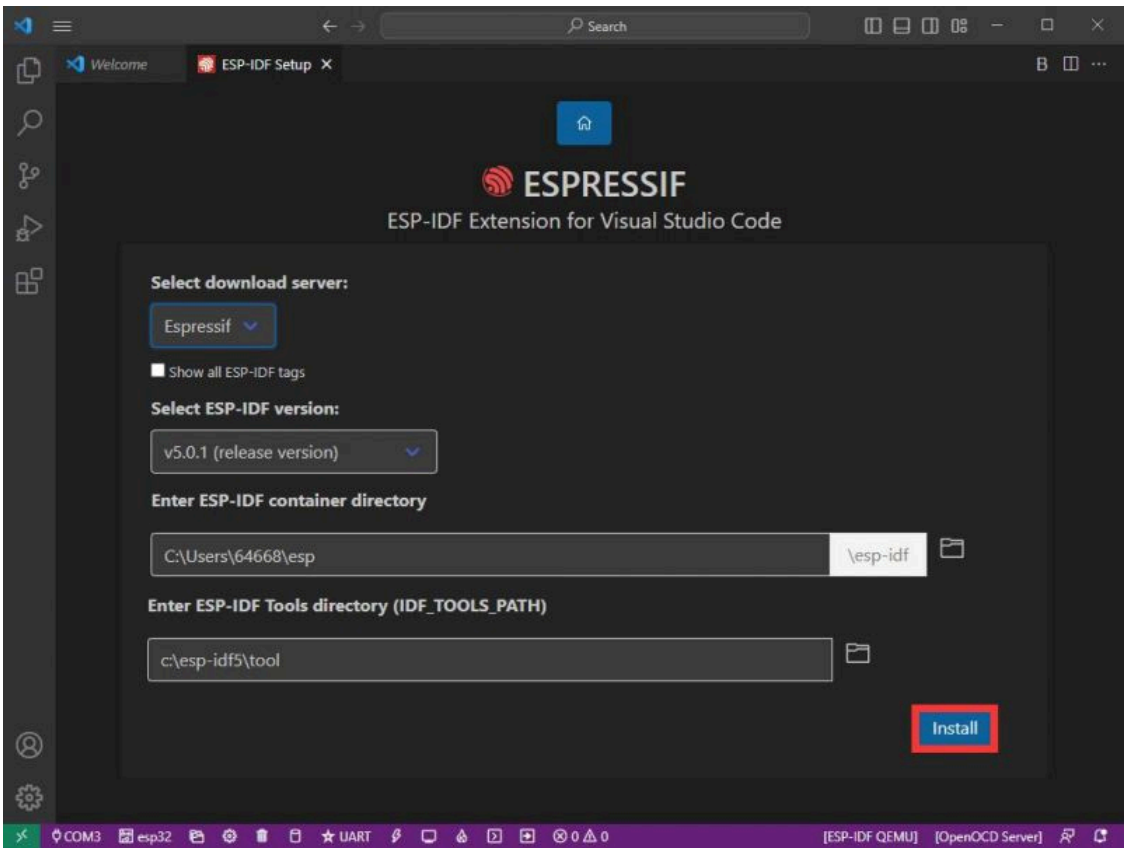
- **Note: If you have installed ESP-IDF before, or if it has failed, please make sure to delete the file completely or create a new path.**

9. Once the configuration is finished, click "install" to download.



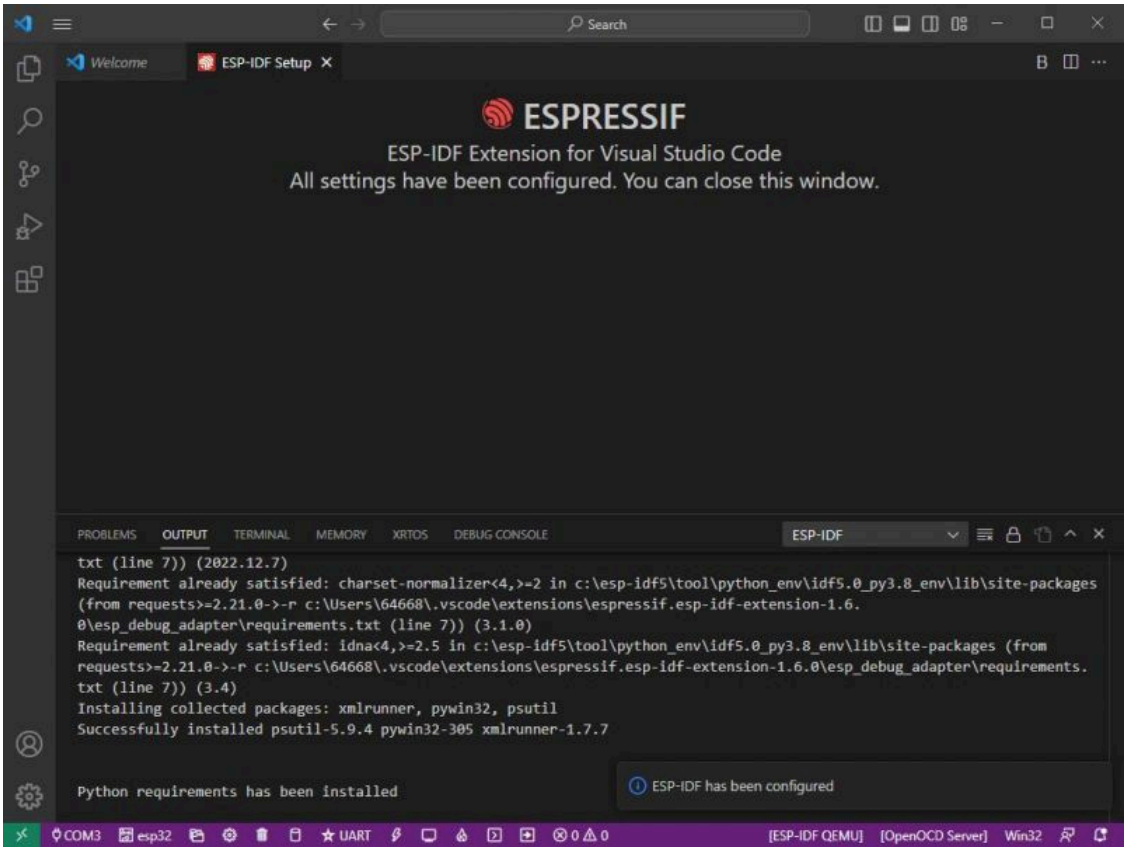
(/wiki/File:ESP32-S3-Pico_15.jpg)

10. Enter the download page, and it will automatically install the corresponding tools and environment, just wait a moment.



(/wiki/File:ESP32-S3-Pico_15.jpg)

11. After the installation is completed, it will enter the following screen, indicating that the installation is finished.



(/wiki/File:ESP32-S3-Pico_17.jpg)

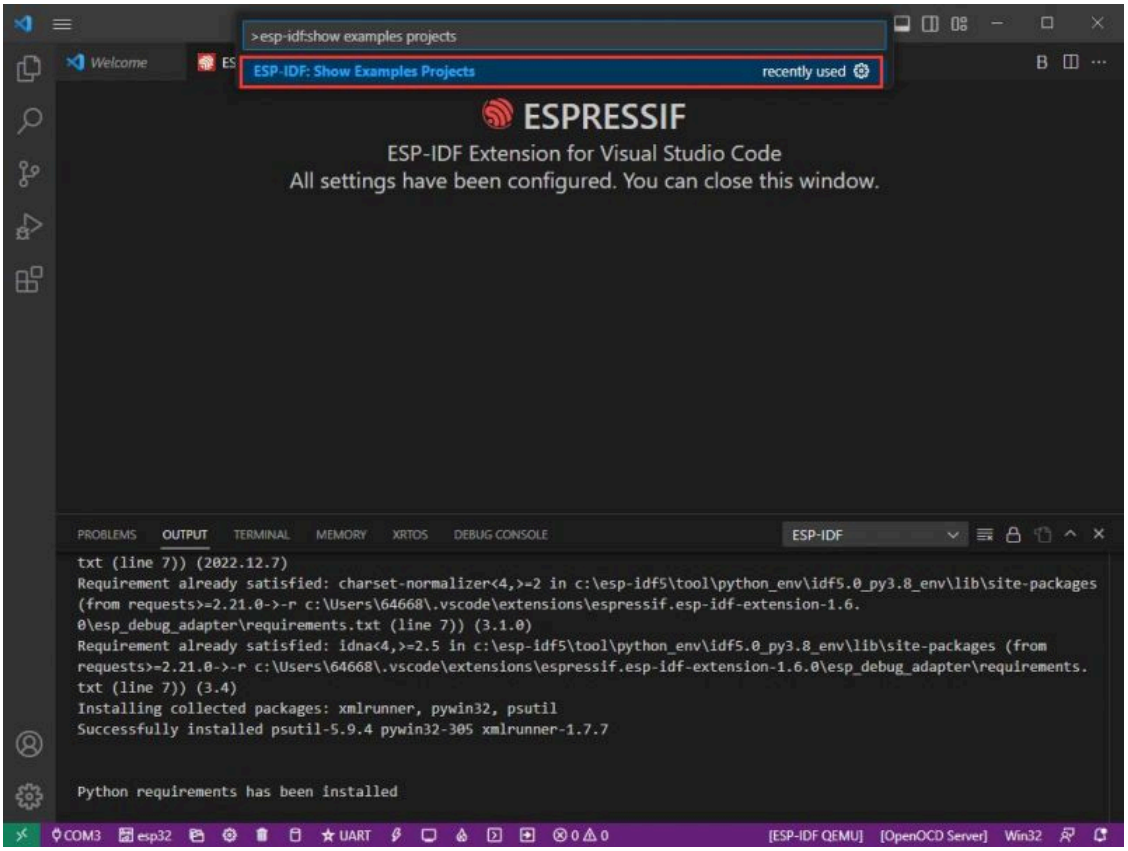
Official Demo Usage

- Click here (https://github.com/espressif/esp-idf/tree/master/examples) to view more details provided by the official ESP.

Creating a Demo

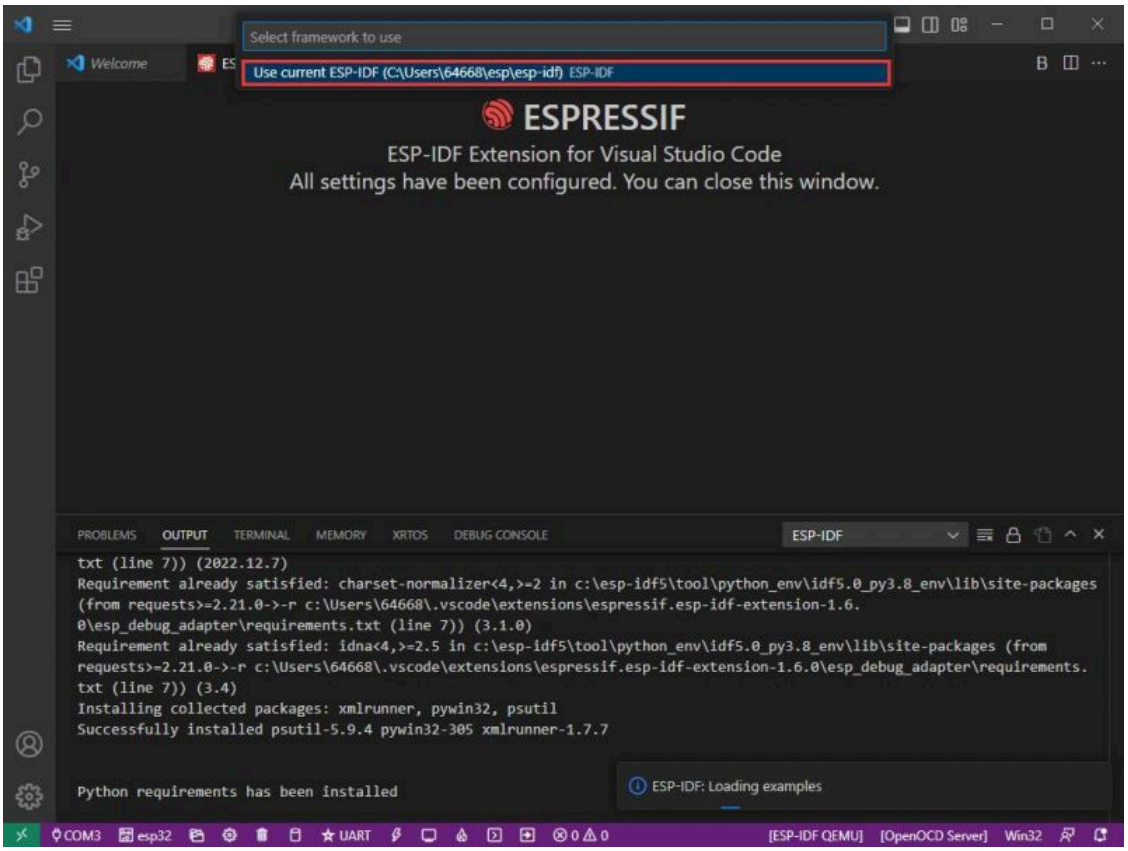
1. Using the shortcut F1, type:

```
esp-idf:show examples projects
```



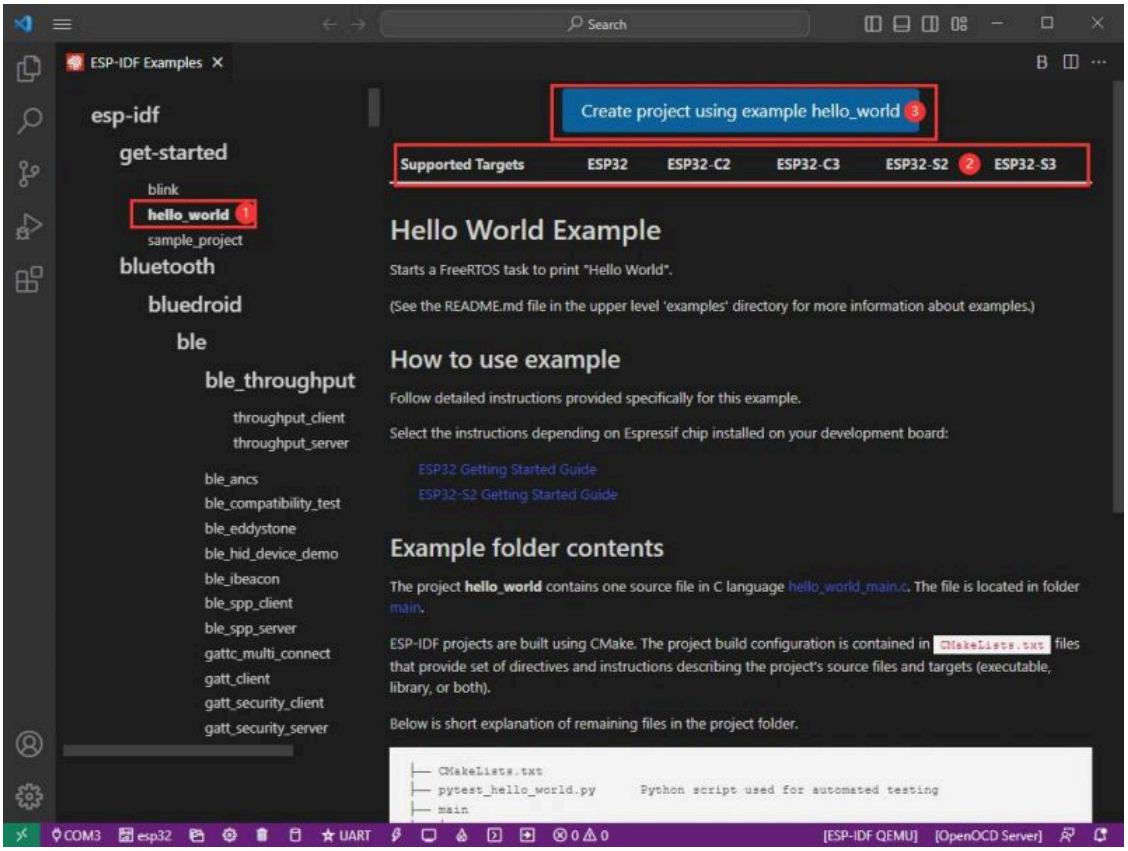
(/wiki/File:ESP32-S3-Pico_18.jpg)

2. Choose your current IDF version:



(/wiki/File:ESP32-S3-Pico_19.jpg)

3. Take "Hello World" as an example:



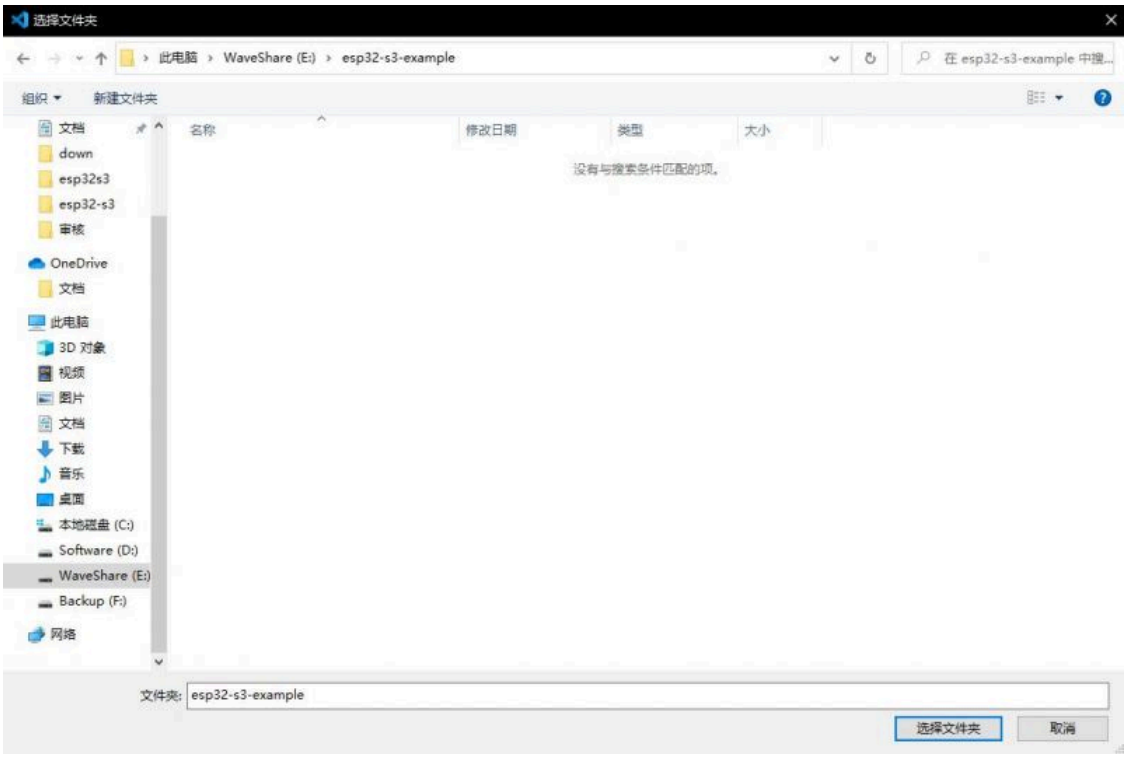
(/wiki/File:ESP32-S3-Pico_20.jpg)

4. Choose the corresponding demo.

5. The readme file will explain which chip the demo is suitable for (the following section will introduce how to use the demo and its file structure, which is omitted here).

6. Click to create the demo.

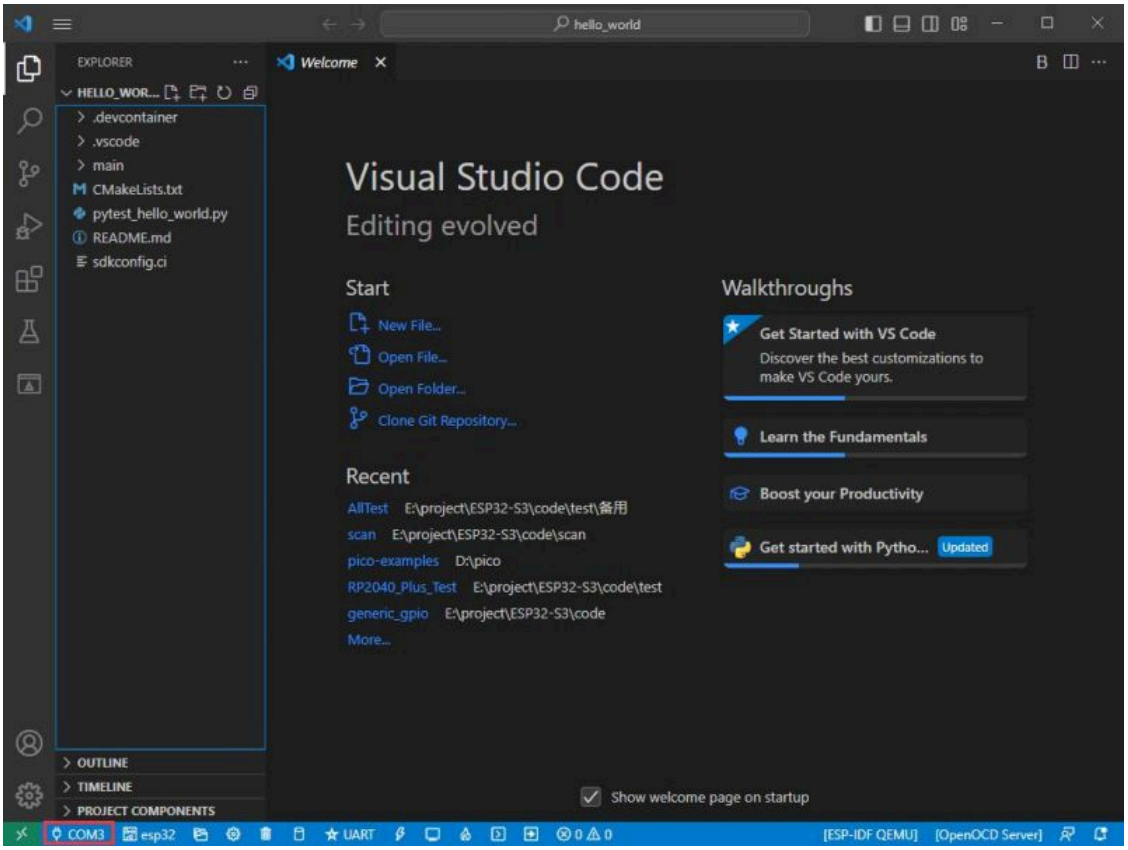
7. Choose the path to place the demo and ensure that there is no folder with the same name as the demo.



(/wiki/File:ESP32-S3-Pico_21.jpg)

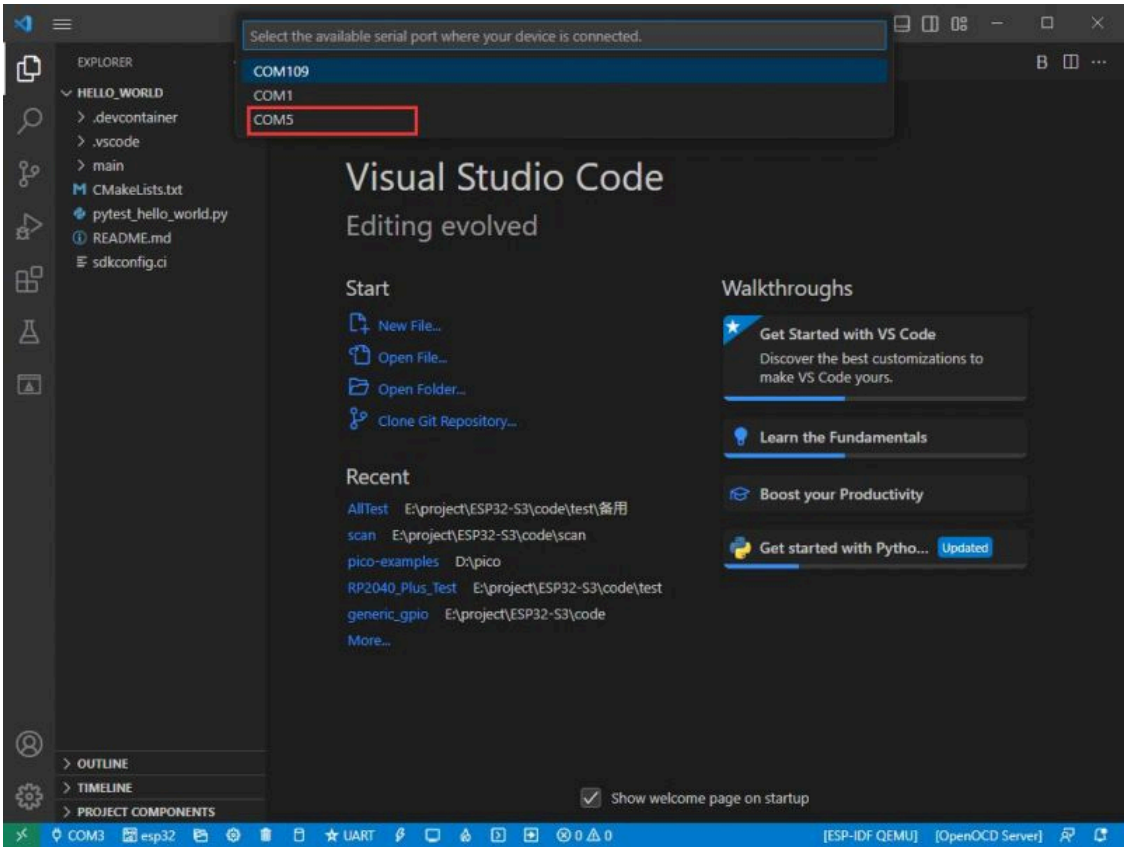
Modify COM Port

1. The corresponding COM port is displayed here, click on it to modify.



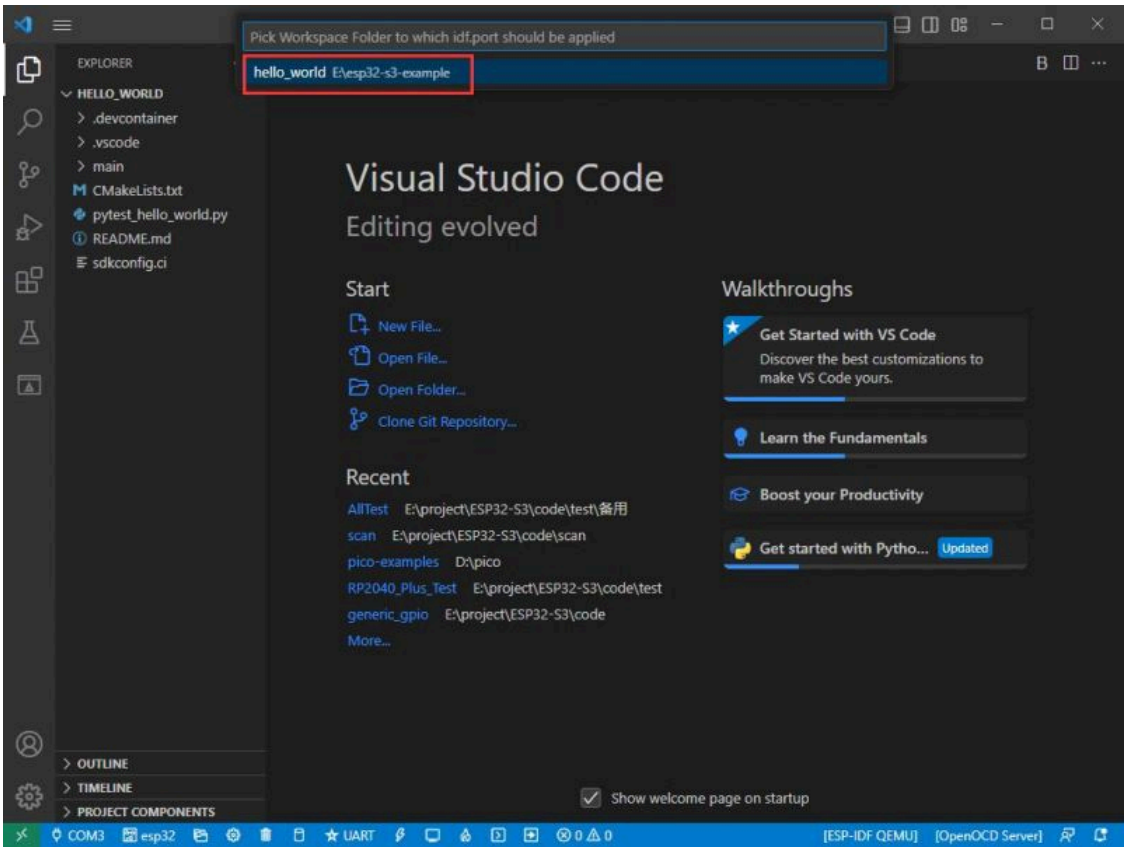
(/wiki/File:ESP32-S3-Pico_22.jpg)

2. We check the device manager COM port, and select COM5, please select your corresponding COM port:



(/wiki/File:ESP32-S3-Pico_23.jpg)

3. Choose the project and demo.

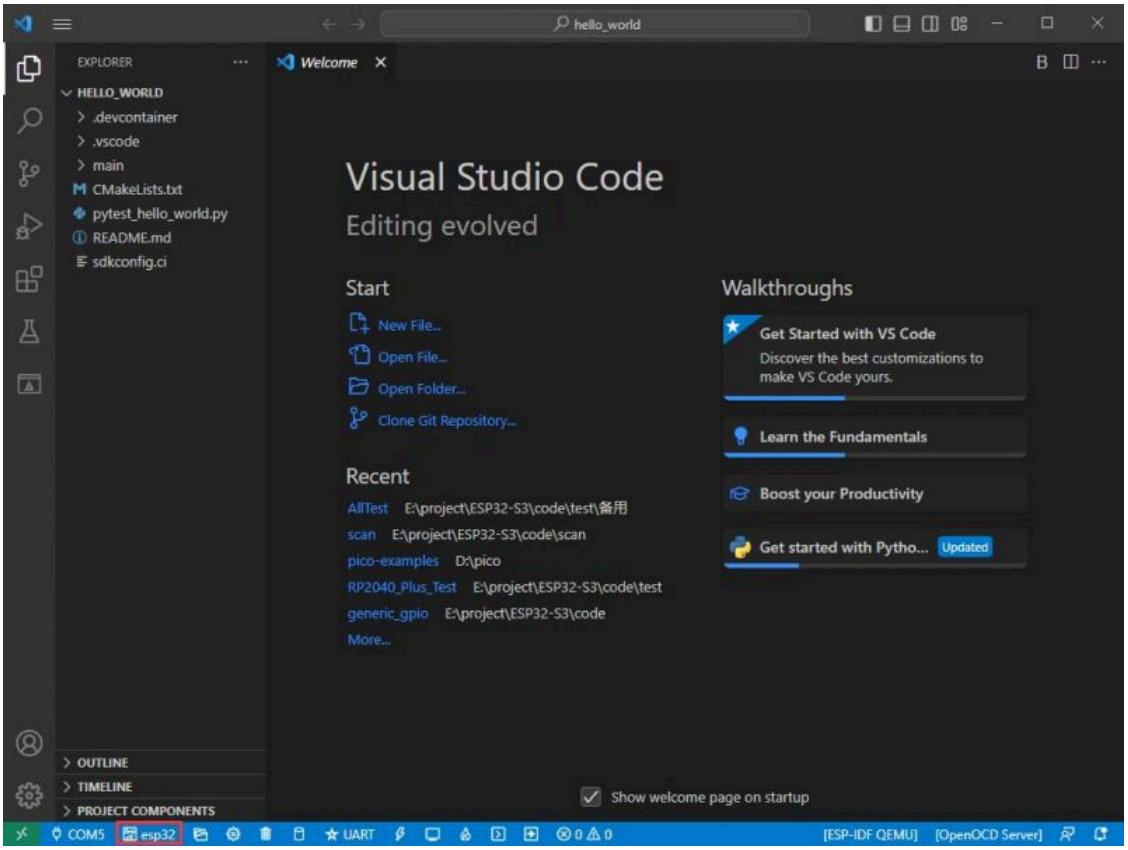


(/wiki/File:ESP32-S3-Pico_24.jpg)

4. Then the COM port is modified.

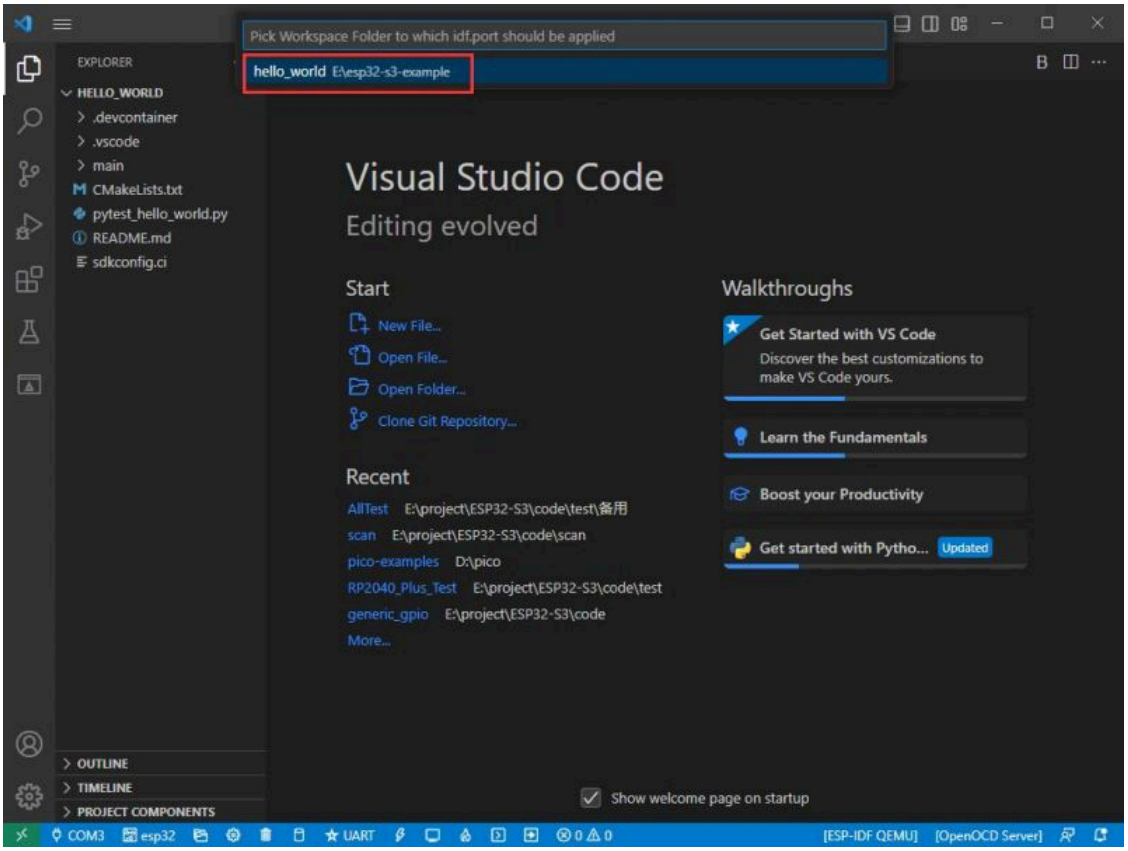
Modify the Driver

1. Here shows the driver used, click here to modify the corresponding driver:



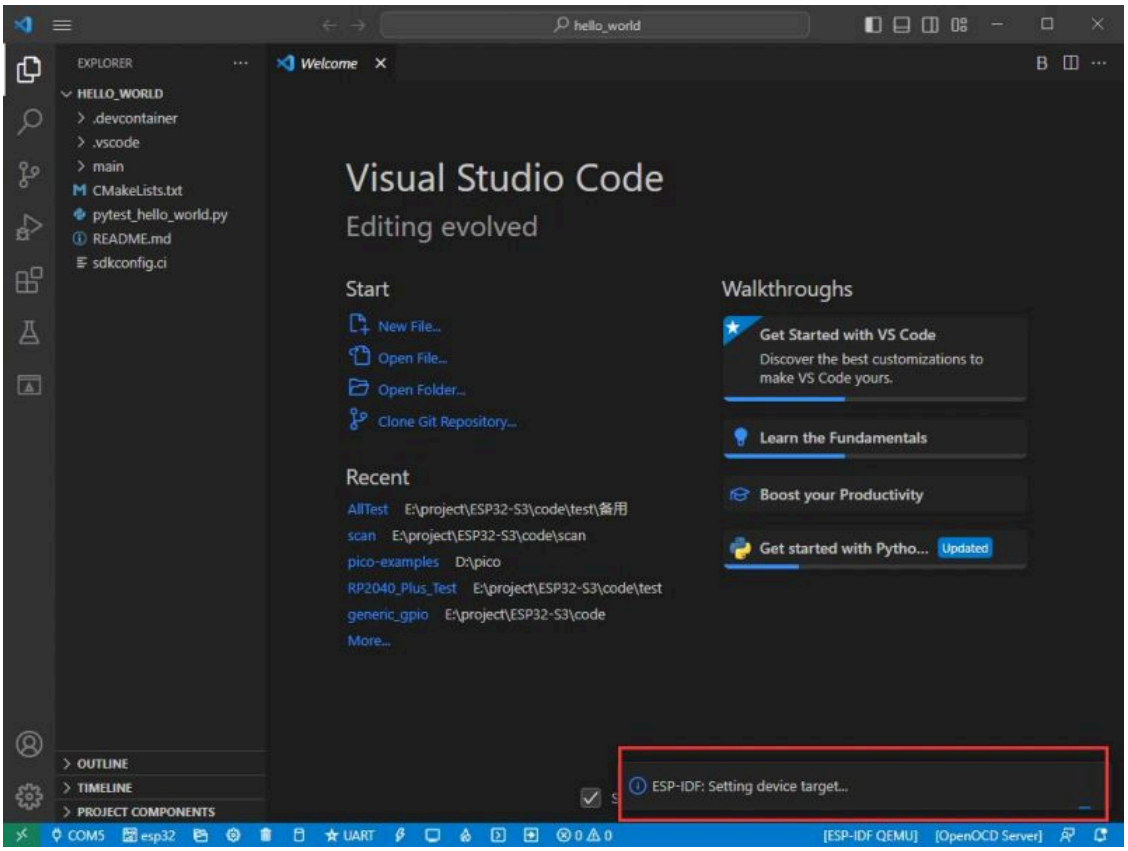
(/wiki/File:ESP32-S3-Pico_25.jpg)

2. Choose the project or demo:



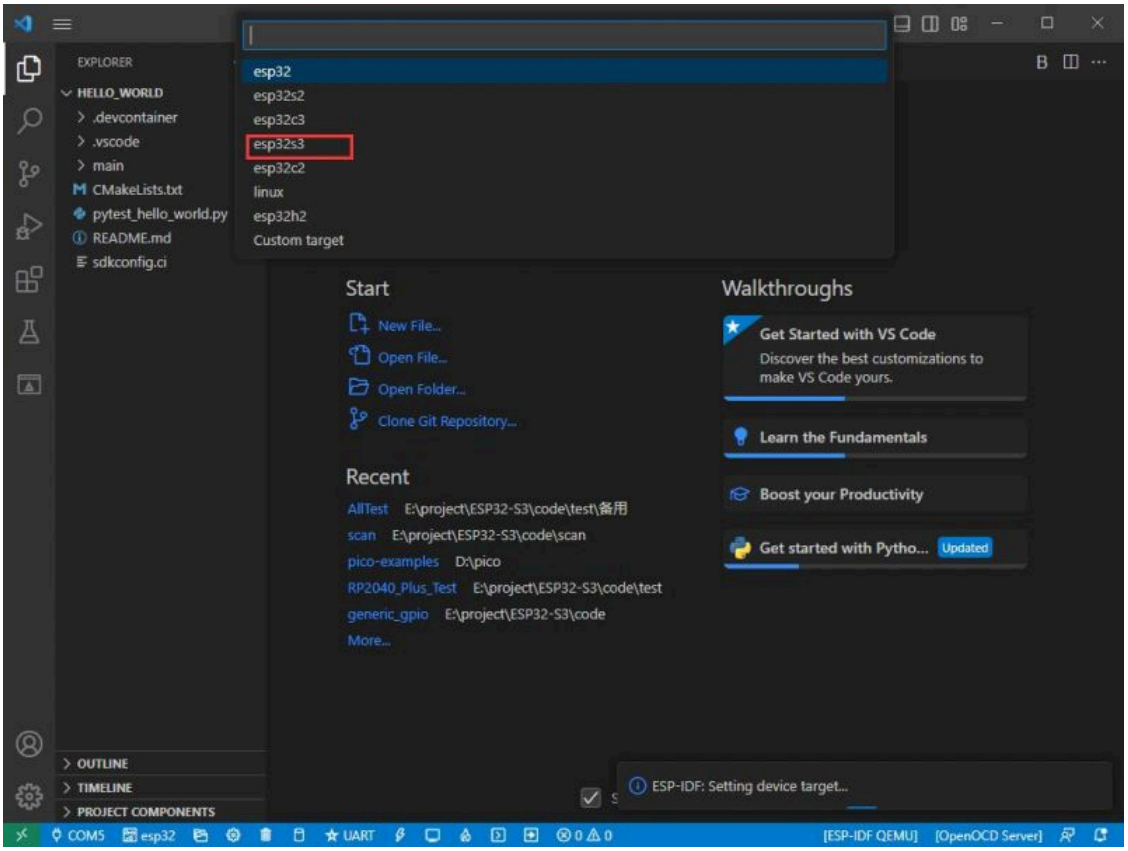
(/wiki/File:ESP32-S3-Pico_24.jpg)

3. Wait for a few seconds after clicking.



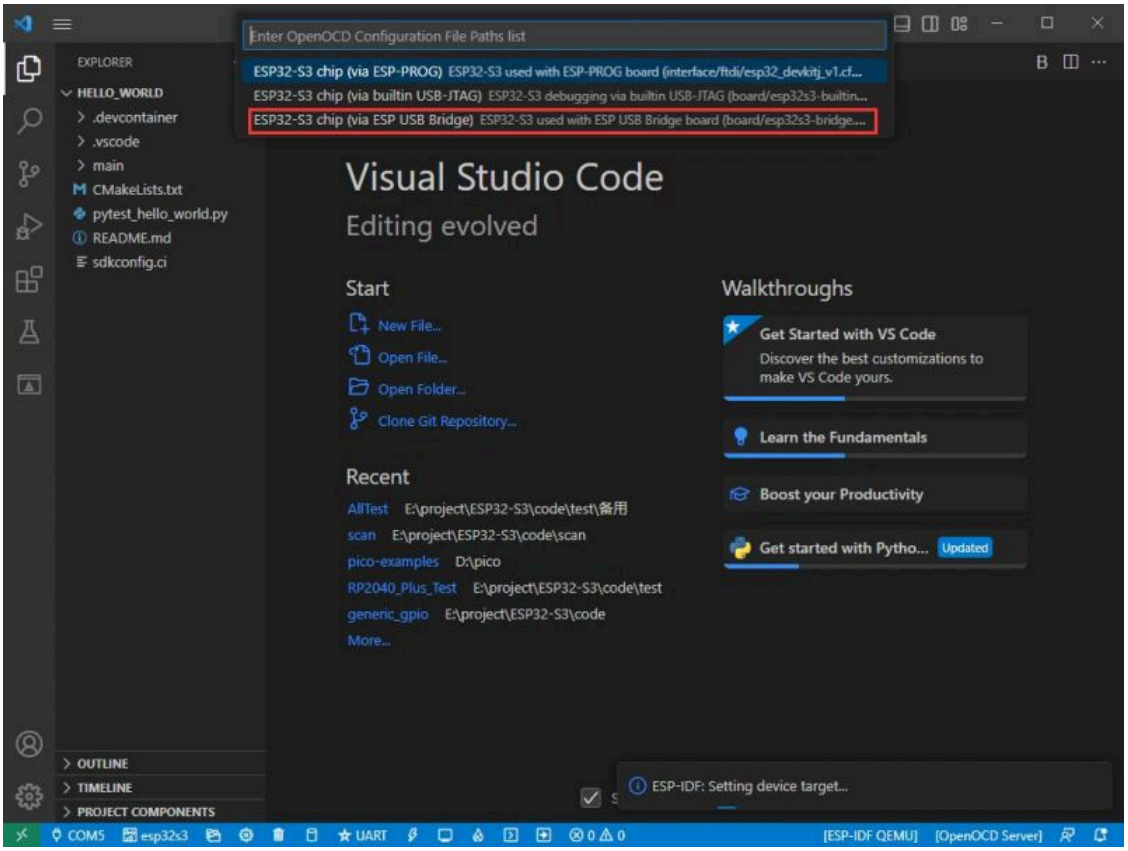
(/wiki/File:ESP32-S3-Pico_27.jpg)

4. Choose the driver we need, that is, the main chip ESP32S3.



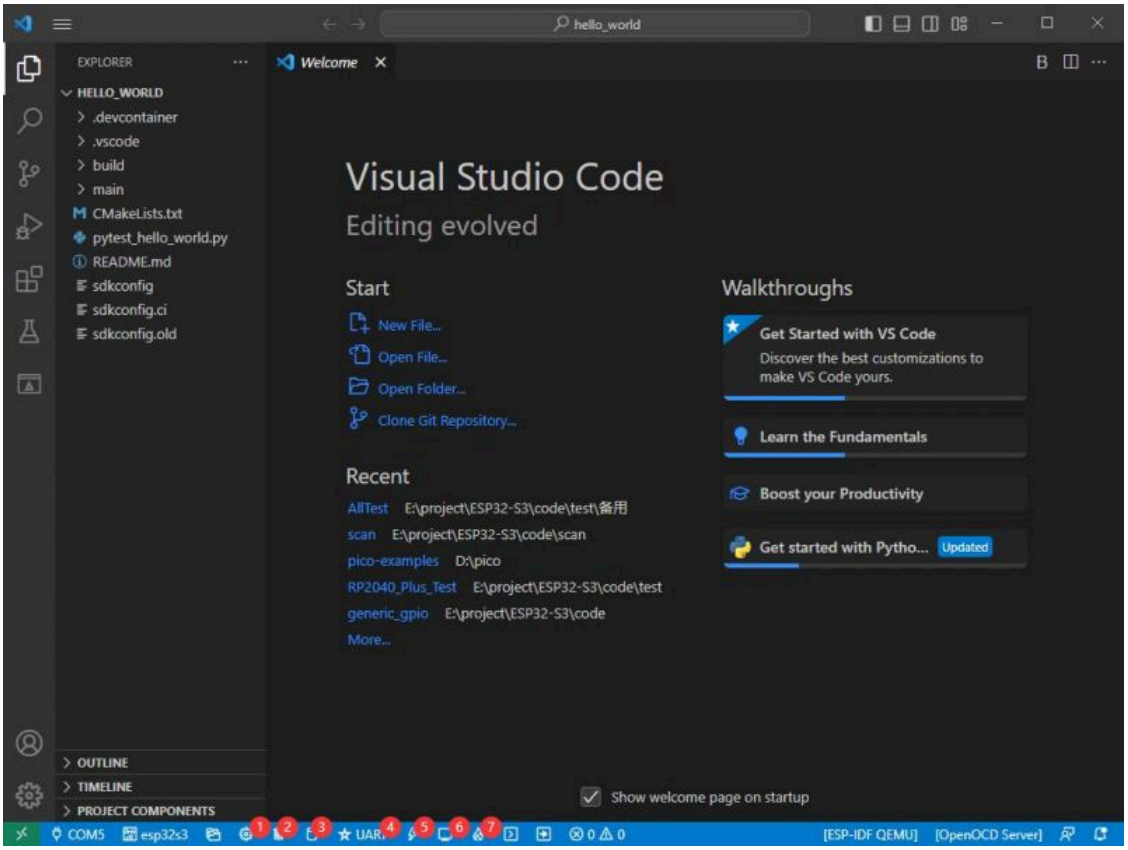
(/wiki/File:ESP32-S3-Pico_28.jpg)

5. Choose the openocd path, we can just choose one at random as it doesn't matter.



(/wiki/File:ESP32-S3-Pico_29.jpg)

The Rest of the Status Bar Introduction



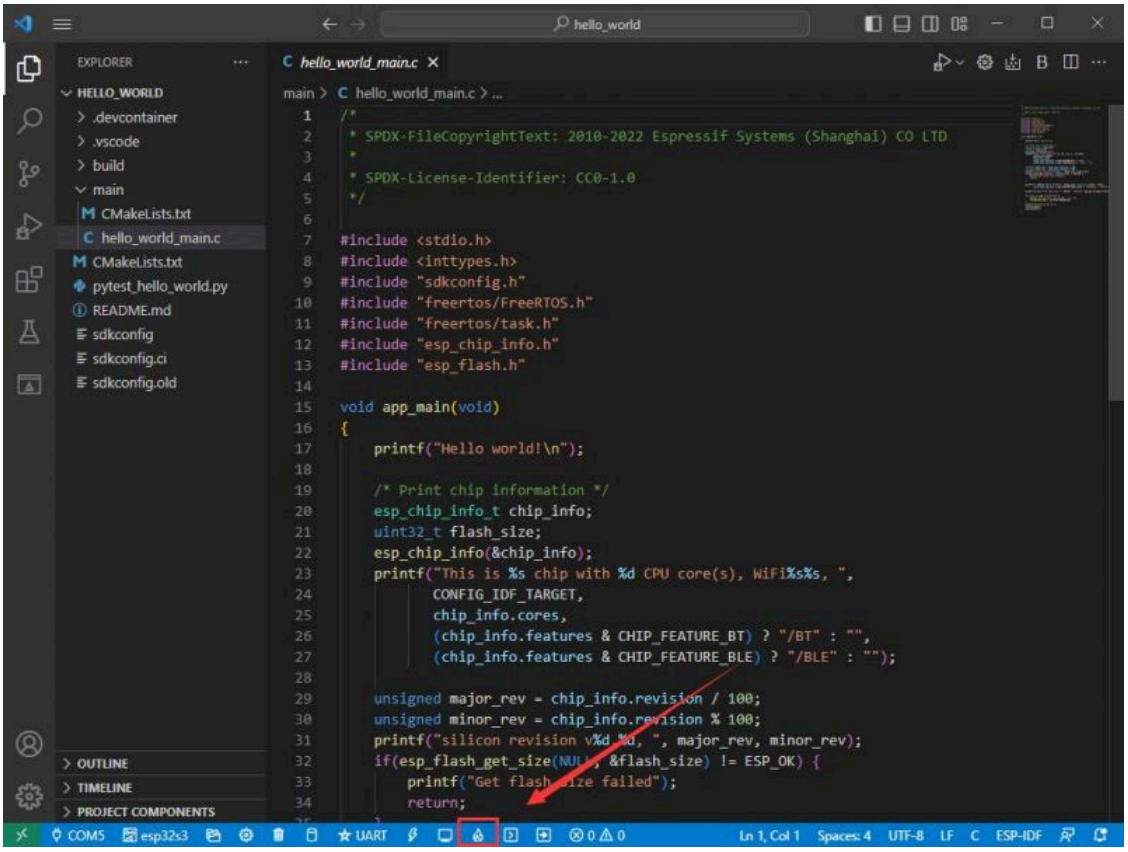
(/wiki/File:ESP32-S3-Pico_30.jpg)

- ① SDK configuration editor: many functions and configurations of ESP-IDF can be modified within it.
- ② Clean up everything and delete all compiled files.
- ③ Compile.

- ④ Current download method, default is UART.
- ⑤ Program the current firmware, please do it after compiling.
- ⑥ Open the serial monitor to view serial information.
- ⑦ Combined button for compiling, programming, and opening the serial monitor (most commonly used during debugging).

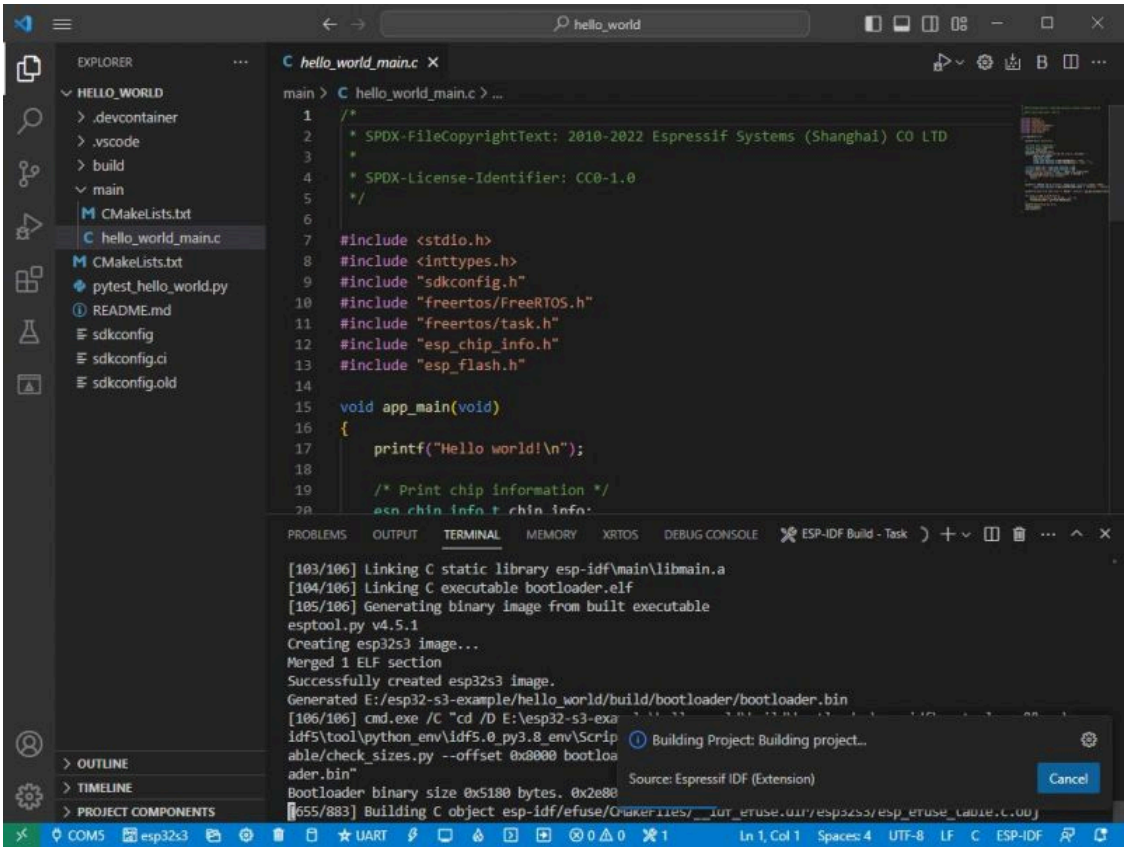
Compile, Program, and Serial Port Monitoring

1. Click on the Compile, Program, and Open Serial Monitor buttons we described earlier.



(/wiki/File:ESP32-S3-Pico_31.jpg)

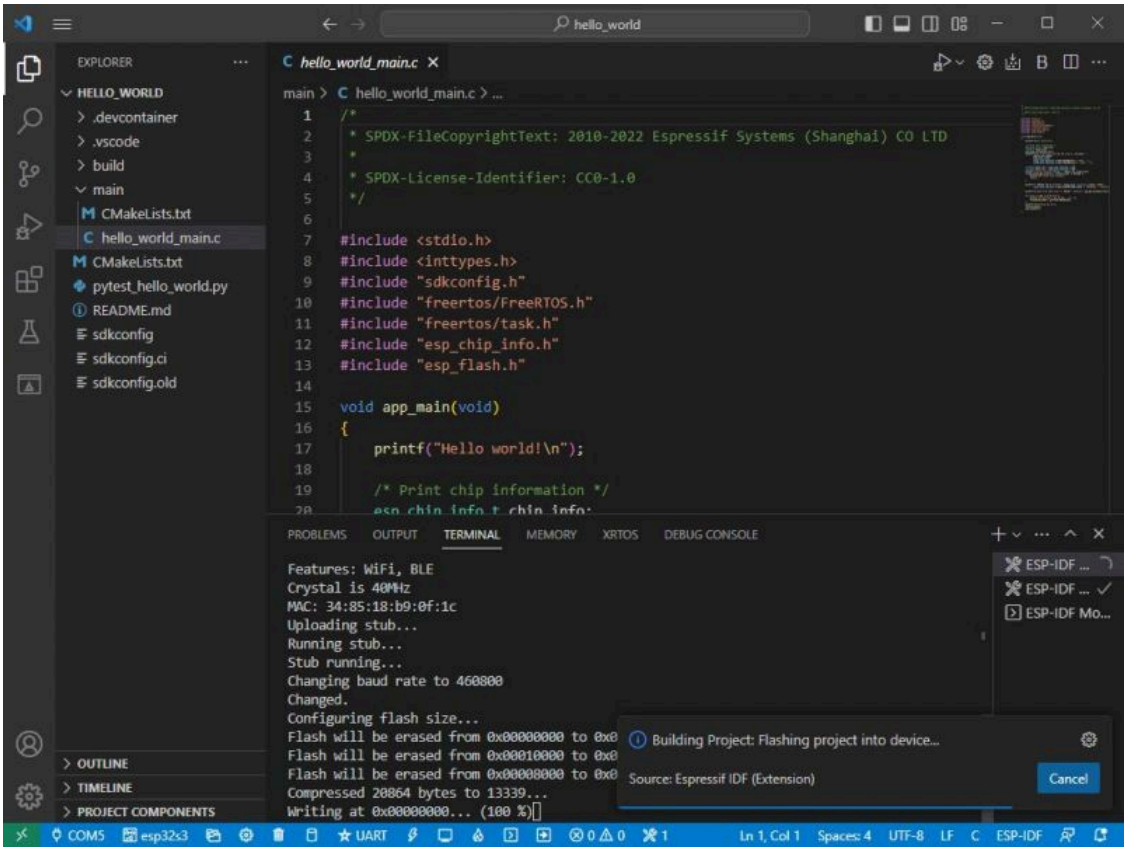
2. It may take a long time to compile, especially for the first time.



(/wiki/File:ESP32-S3-Pico_32.jpg)

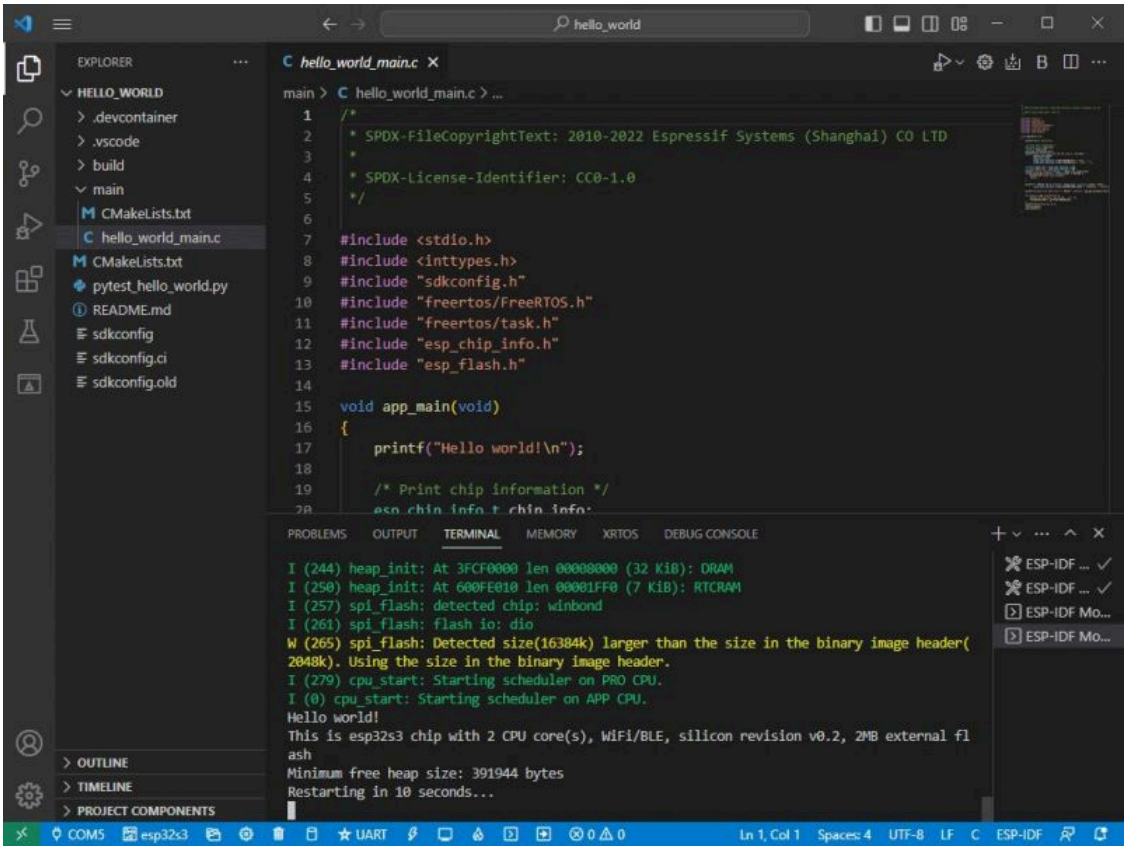
- During this process, ESP-IDF may take up a lot of CPU resources and therefore may cause system lag.

3. Because we use CH343 as a USB to serial port chip, and the on-board automatic download circuit, it can be downloaded automatically without manual operation.



(/wiki/File:ESP32-S3-Pico_33.jpg)

4. After successful download, it will automatically enter the serial monitor, and you can see the corresponding information output from the chip and prompt to reboot after 10s.



(/wiki/File:ESP32-S3-Pico_34.jpg)

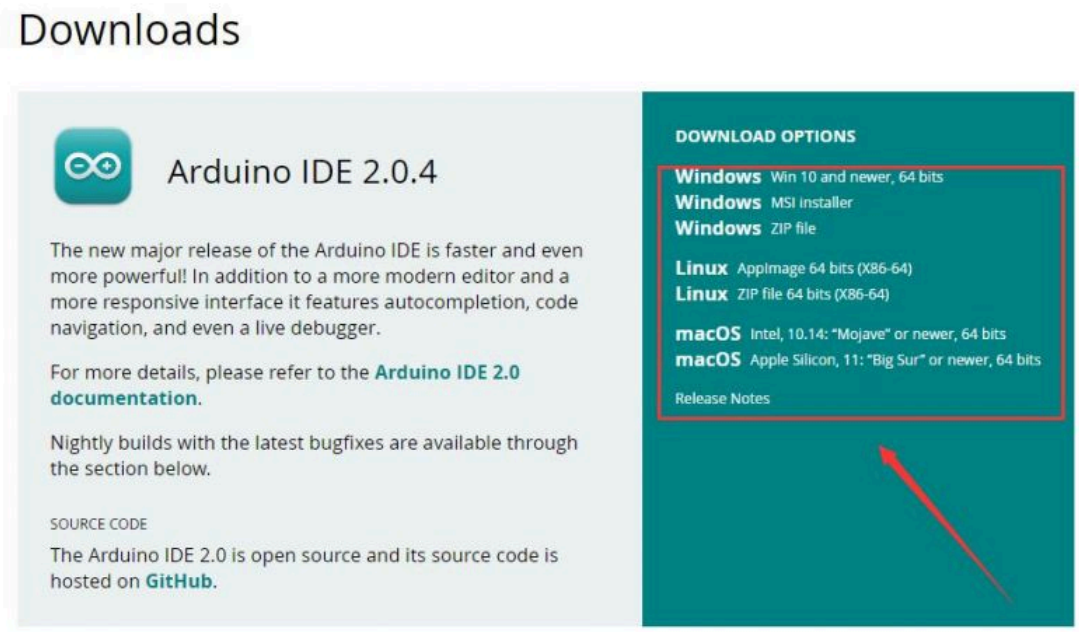
Arduino

[Collapse]

- **If you do not use arduino-esp32 before, you can refer to this link (https://docs.espressif.com/projects/arduino-esp32/en/latest/index.html).**

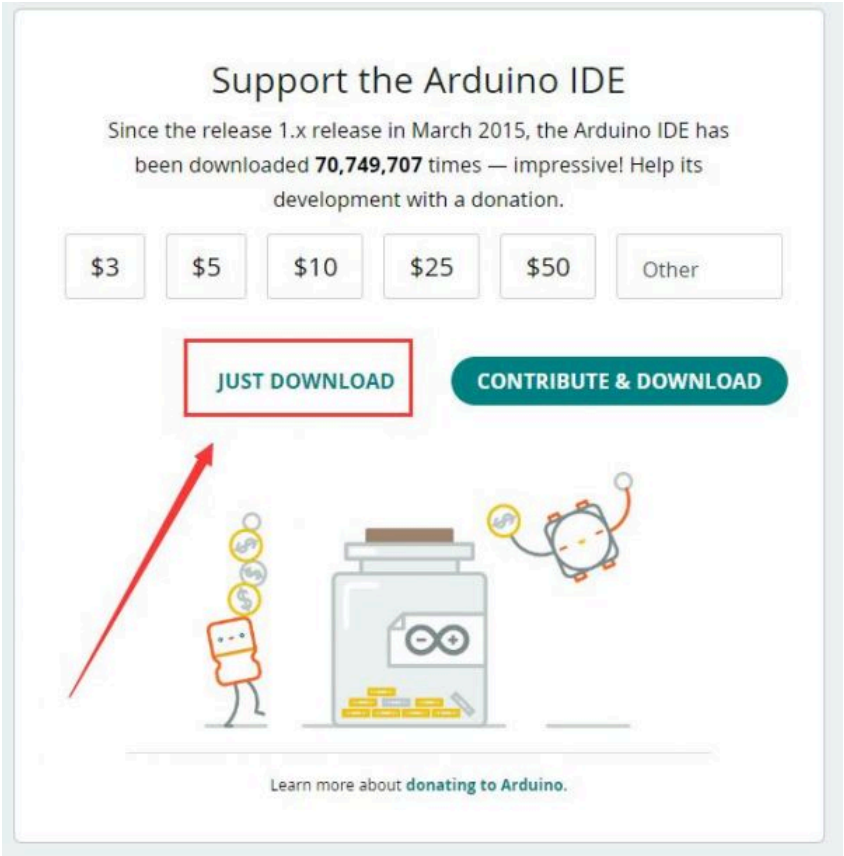
Install Arduino IDE

1. Open the official software download webpage (https://www.arduino.cc/en/software), and choose the corresponding system and system bits to download.



(/wiki/File:ESP32-S3-Pico_35.jpg)

2. You can choose "Just Download", or "Contribute & Download".

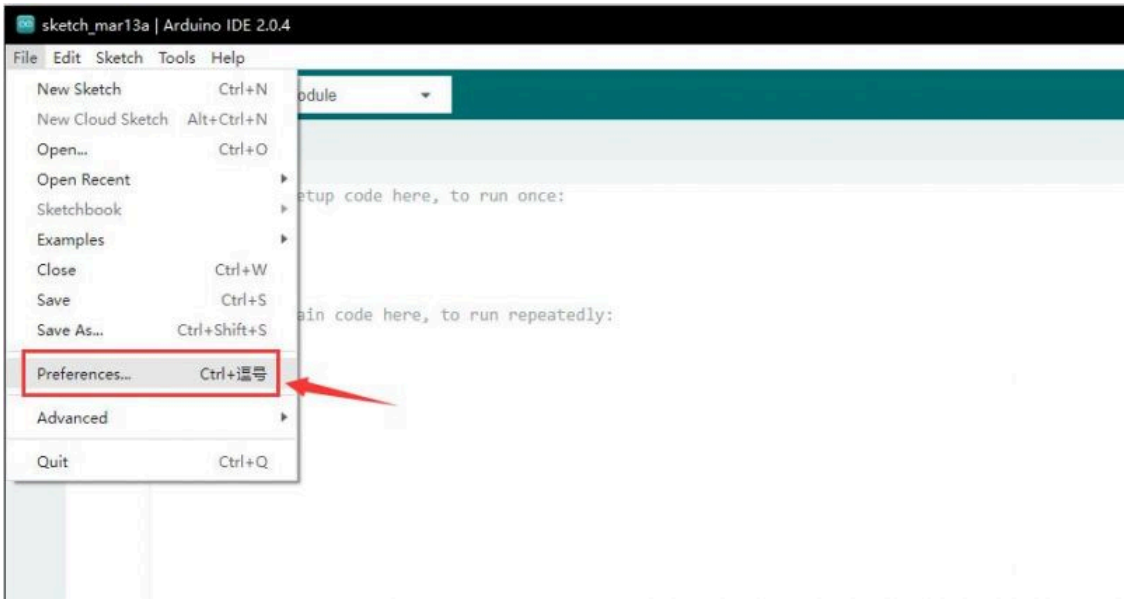


(/wiki/File:ESP32-S3-Pico_36.jpg)

3. Run to install the program and install it all by default.

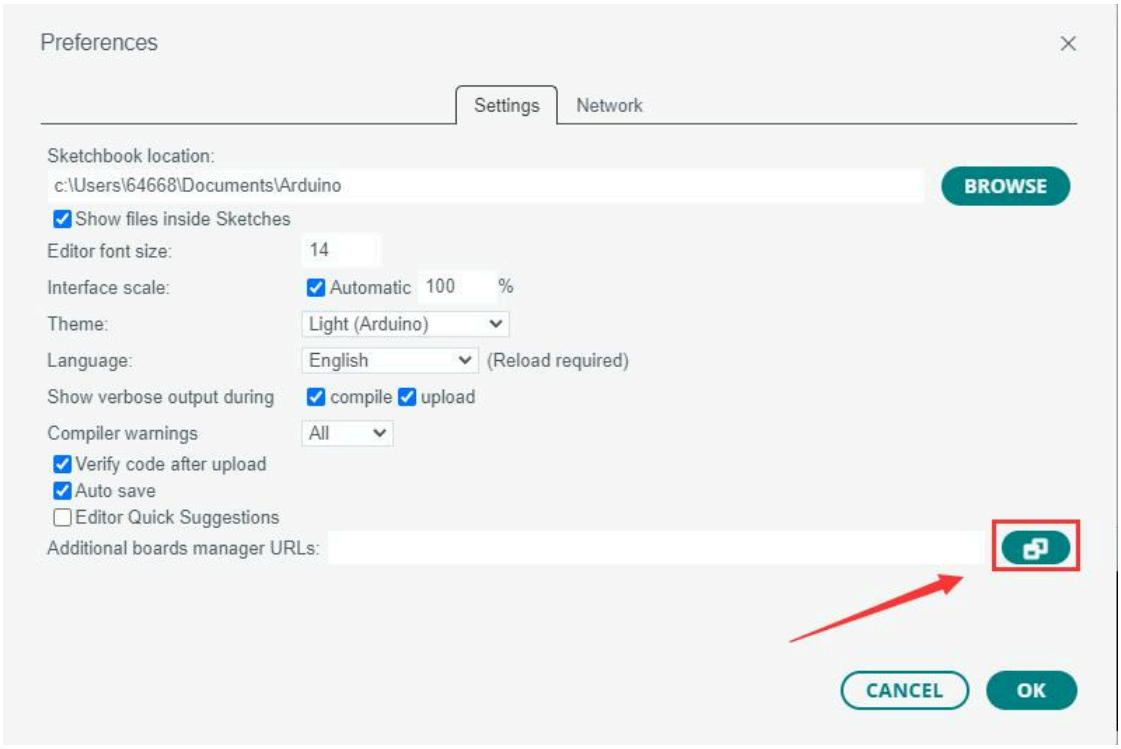
Install arduino-esp32 Online

1. Open Preferences.



(/wiki/File:ESP32-S3-Pico_37.jpg)

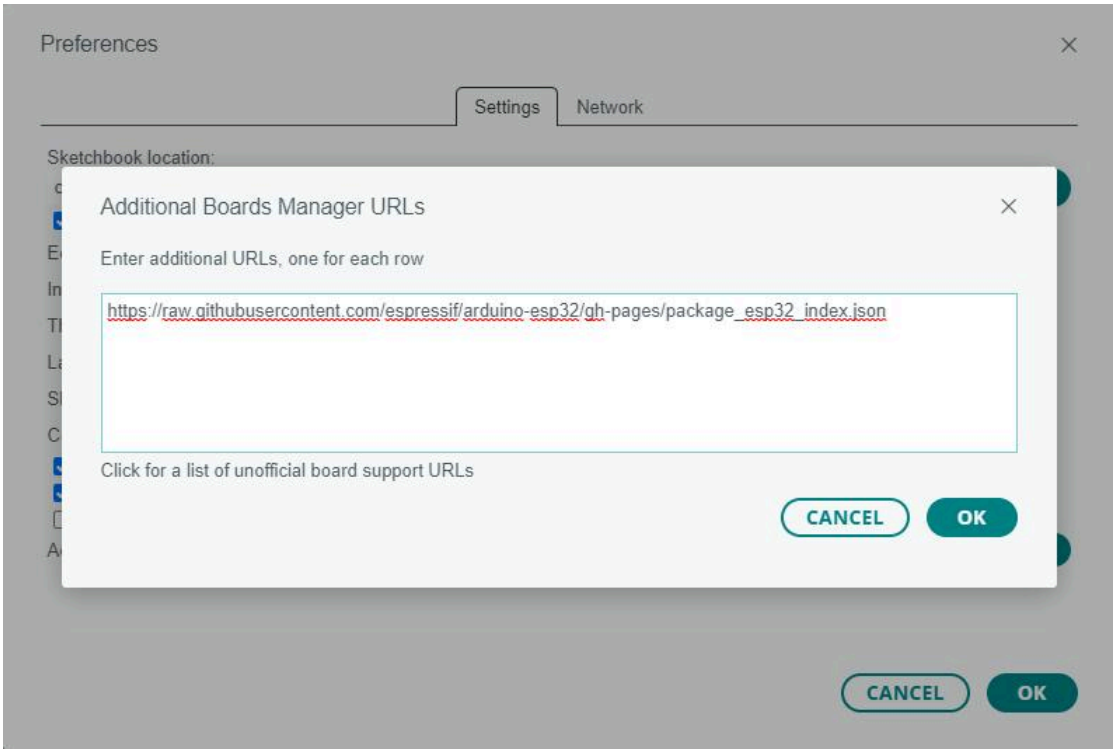
2. Add the corresponding board manager URLs and click the button.



(/wiki/File:ESP32-S3-Pico_38.jpg)

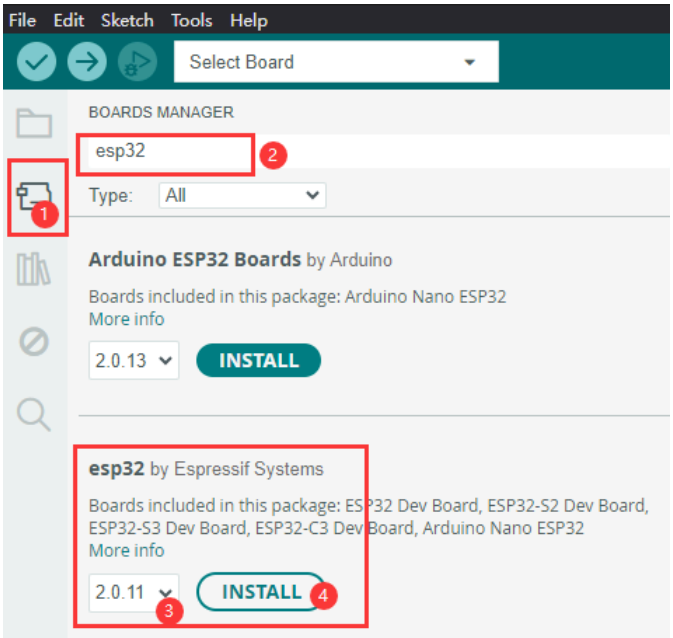
3. Add the following content in the first blank.

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



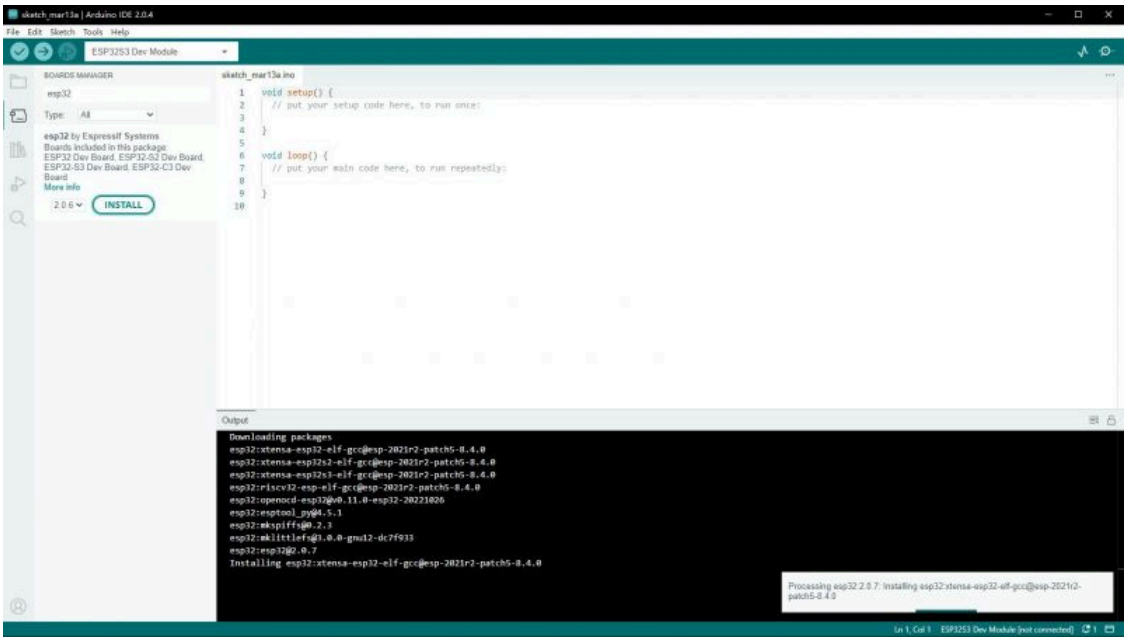
(/wiki/File:ESP32-S3-Pico_39.jpg)

4. Save the setting.
5. Open the board manager, enter ESP32 in the search bar, and select version **2.0.11**.



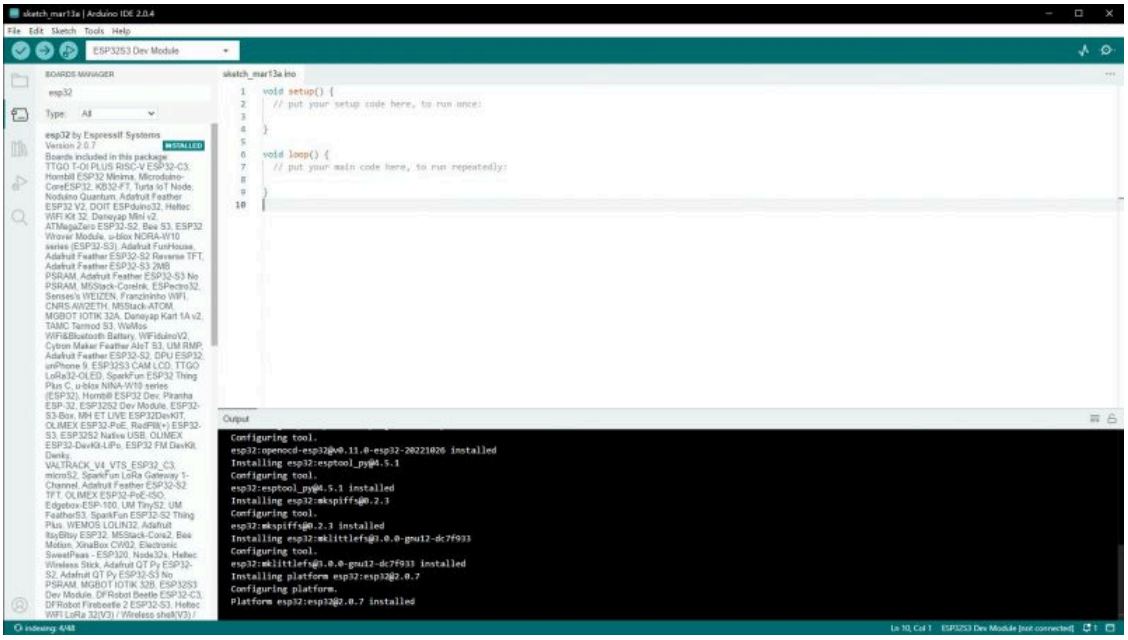
(/wiki/File:ESP32-S3-Pico-Ar-06-240708.png)

6. Wait for downloading.



(/wiki/File:ESP32-S3-Pico_41.jpg)

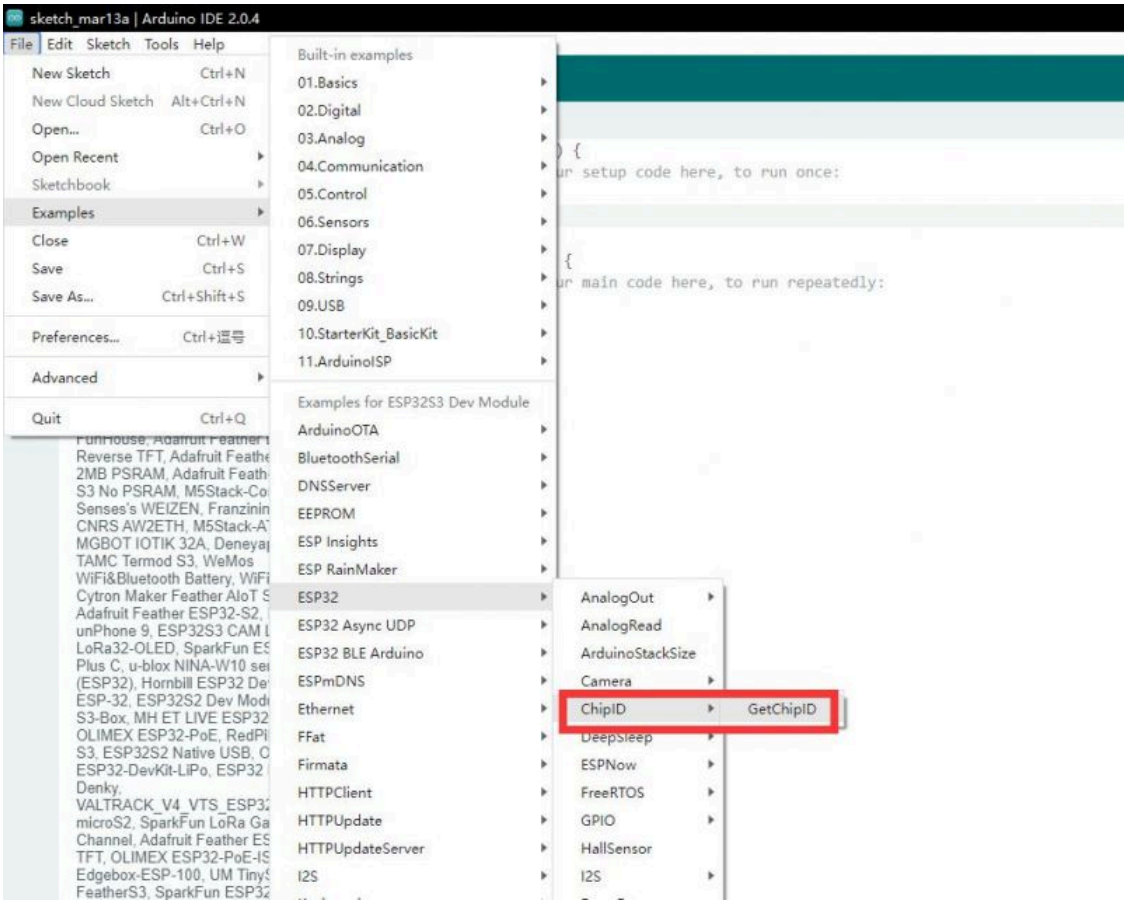
7. arduino-esp32 is downloaded.



(/wiki/File:ESP32-S3-Pico_42.jpg)

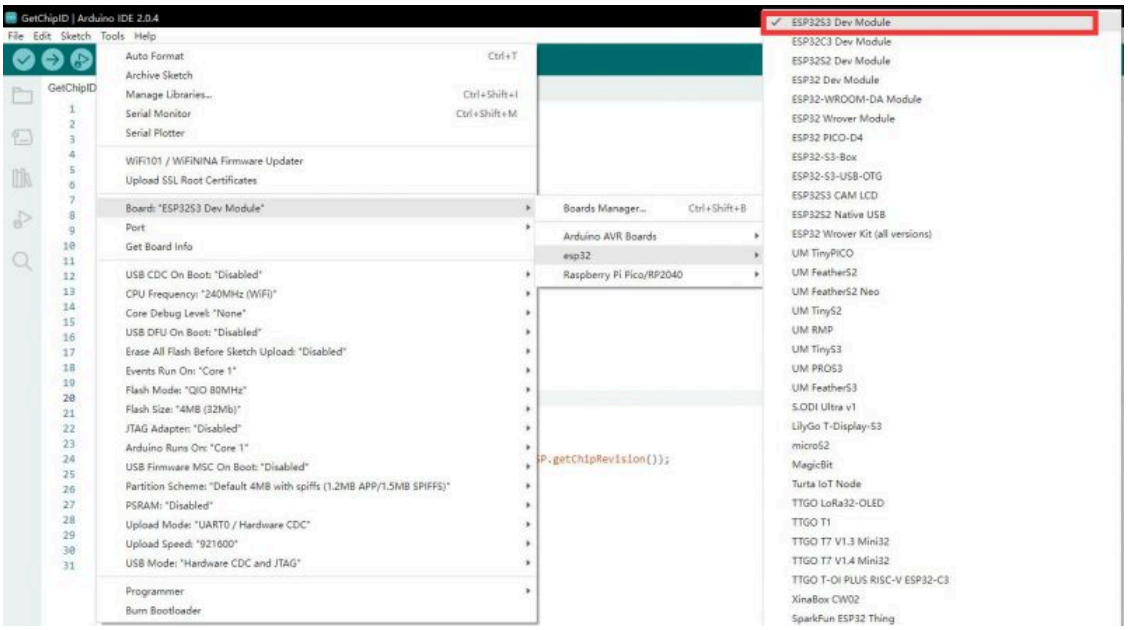
Use Arduino Demo

1. Select the demo, here we choose the demo to get the chip ID.



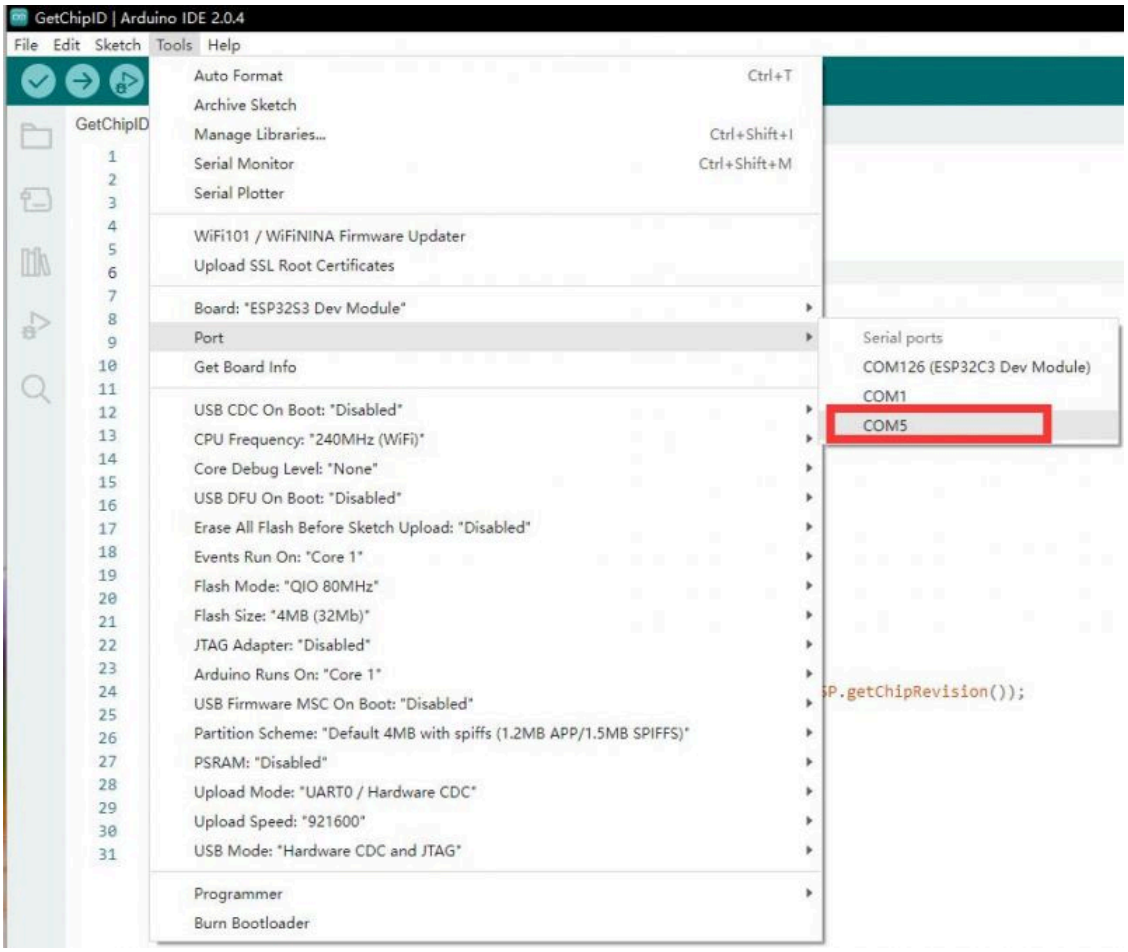
(/wiki/File:ESP32-S3-Pico_43.jpg)

2. Select the board as ESP32S3 Dev Module.



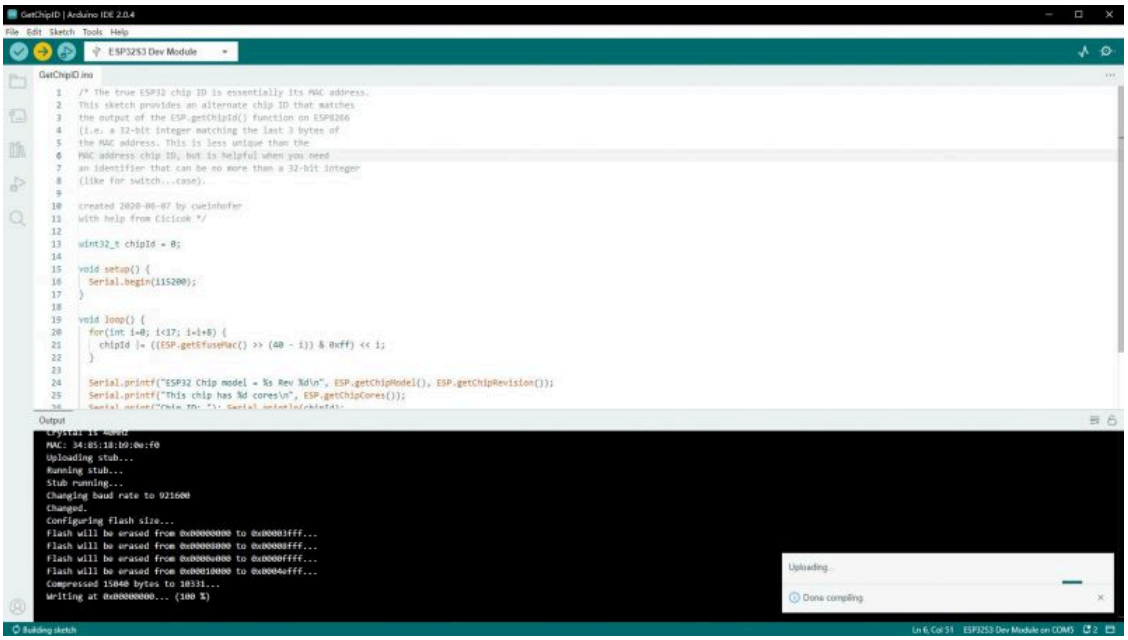
(/wiki/File:ESP32-S3-Pico_44.jpg)

3. Choose the COM5 port of ESP32-S3 USB.

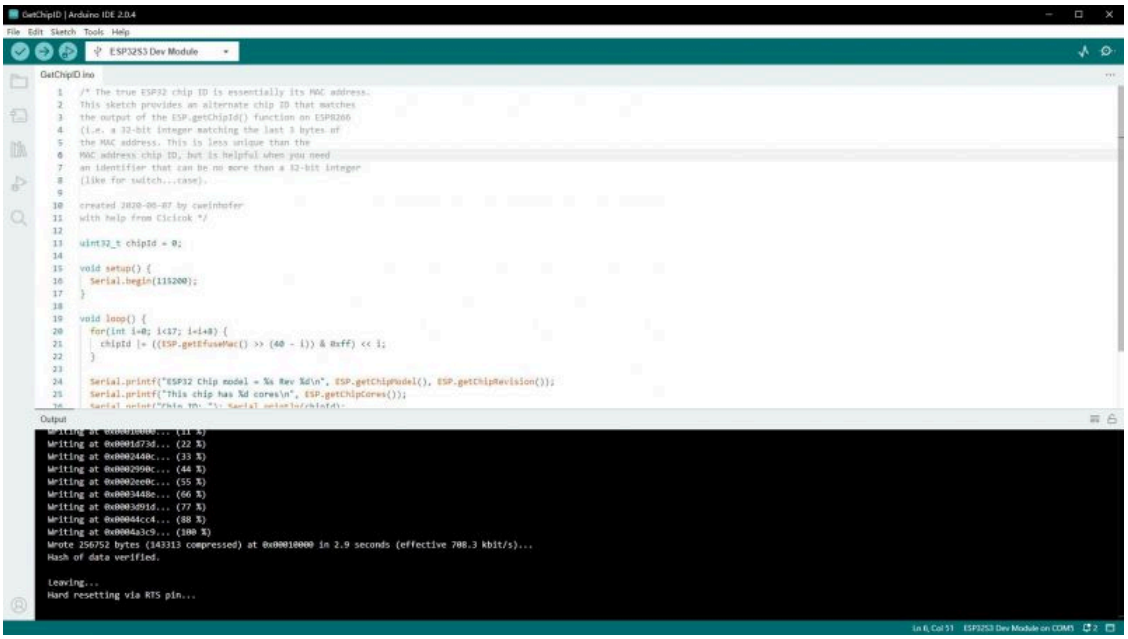


(/wiki/File:ESP32-S3-Pico_45.jpg)

4. Click the download button, then it compiles and downloads automatically.

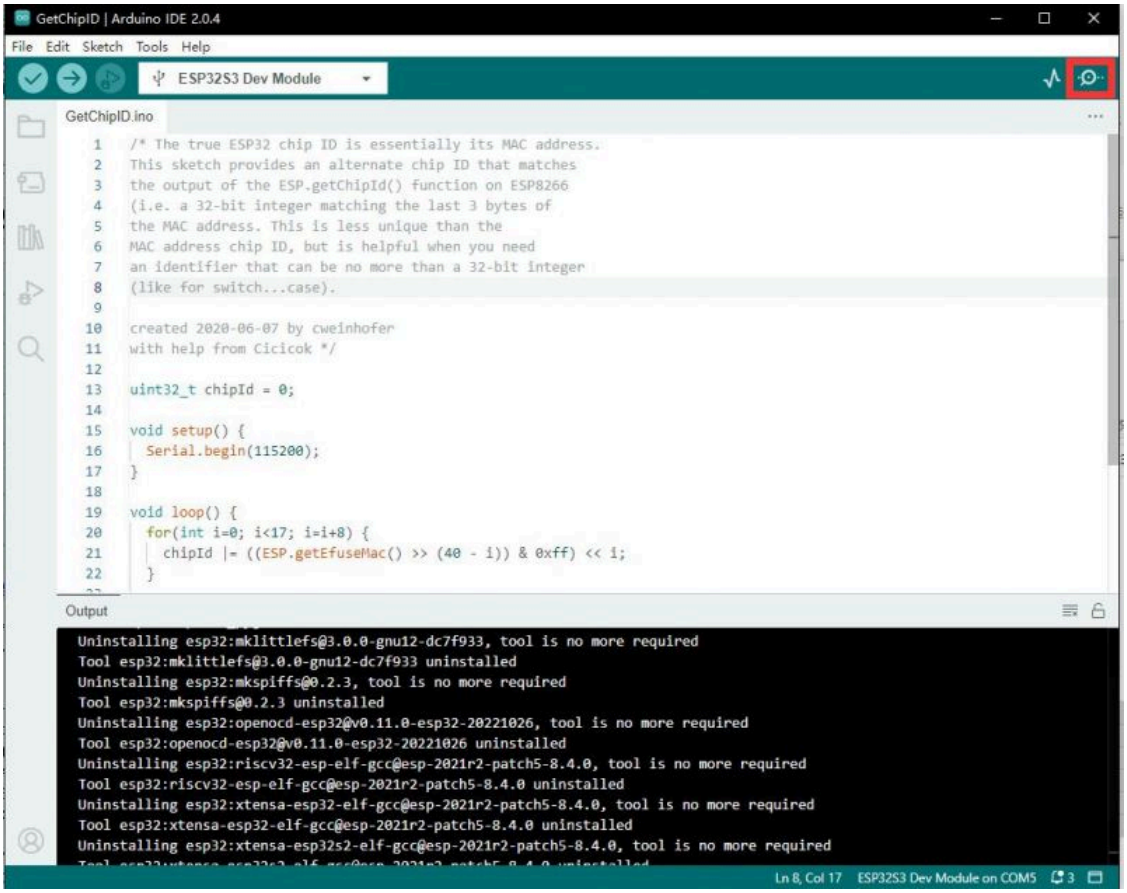

(/wiki/File:ESP32-S3-Pico_47.jpg)

5. Finish.



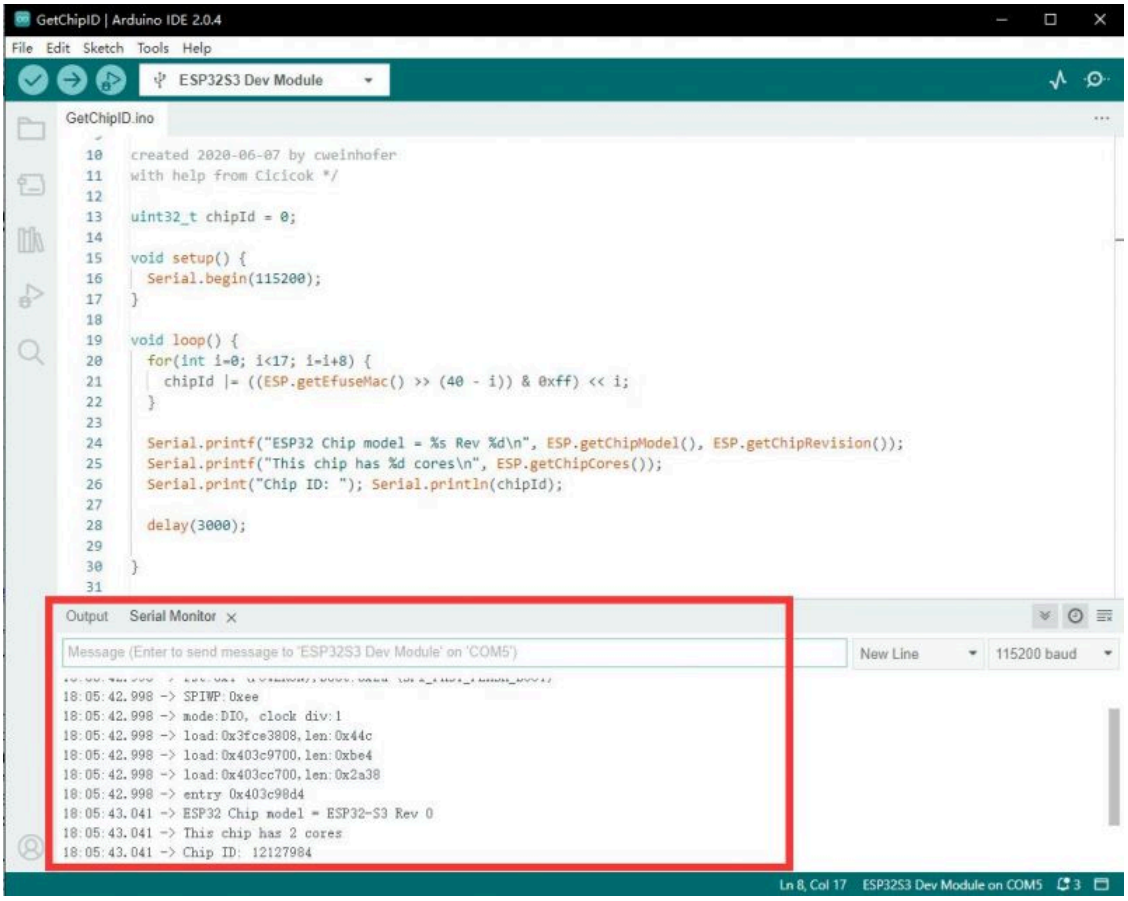
(/wiki/File:ESP32-S3-Pico_48.jpg)

6. Open the Serial Port Monitor.



(/wiki/File:ESP32-S3-Pico_49.jpg)

7. See the chip ID of the loop output.



(/wiki/File:ESP32-S3-Pico_50.jpg)

Note: This product uses USB to connect to Type-C. If it cannot output the chip information, please modify "Serial.print" function to "printf" function, and directly output to USB.

MicroPython

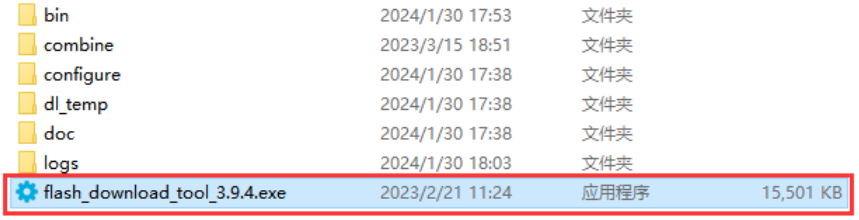
[Collapse]

Burn Firmware

1. Download the MicroPython firmware: ESP32-S3-GEEK-MPY-Flash Tool (https://files.waveshare.com/wiki/ESP32-S3-GEEK/ESP32-S3-GEEK-MPY-Flash.zip) or official micropython firmware (https://micropython.org/download/ESP32_GENERIC_S3/).
2. Unzip the downloaded package and enter:

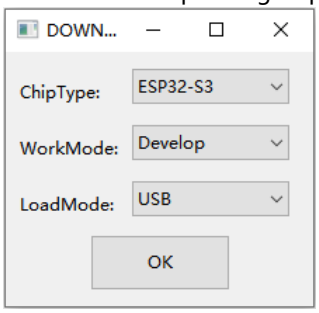
(/wiki/File:ESP32-S3-GEEK-MPY_04.png)

3. **Long-press the ESP32-S3-GEEK key to connect to the USB port of the PC and then release the boot key.**
4. Enter the flash_download_tool_3.9.4 file folder, open the flash_download_tool_3.9.4.exe.



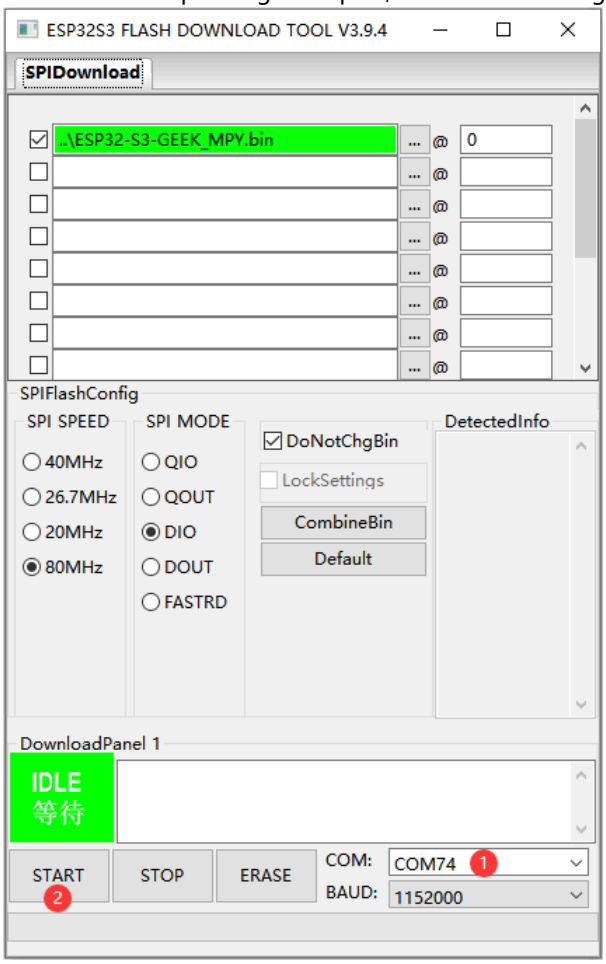
(/wiki/File:ESP32-S3-GEEK-MPY_05.png)

5. Select the corresponding chip, here we select the ESP32-S3 and USB.



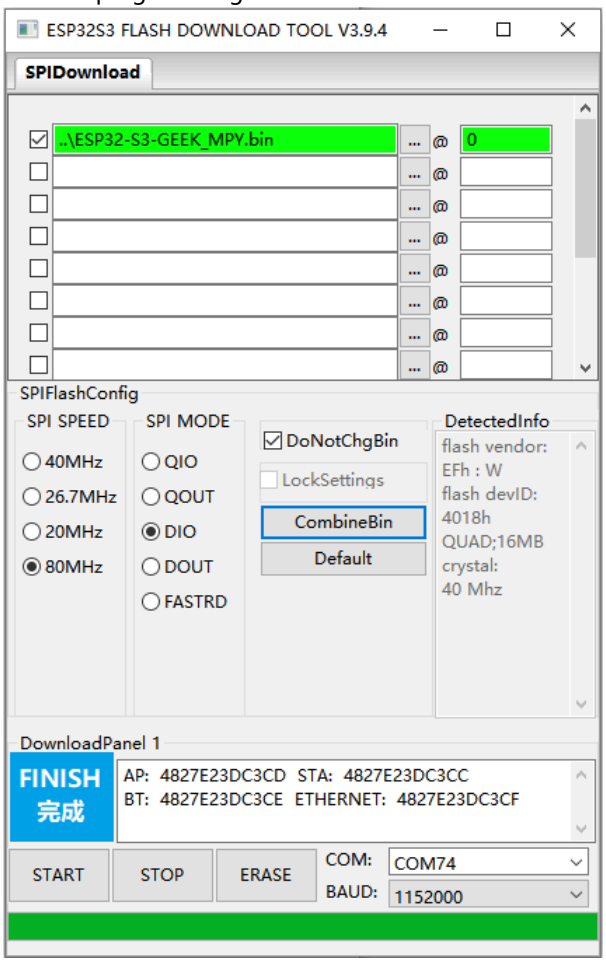
(/wiki/File:ESP32-S3-GEEK-MPY_02.png)

6. Select the corresponding COM port, and we have configured other information for you. Just click on "Start" to download.



(/wiki/File:ESP32-S3-GEEK-MPY_03.png)

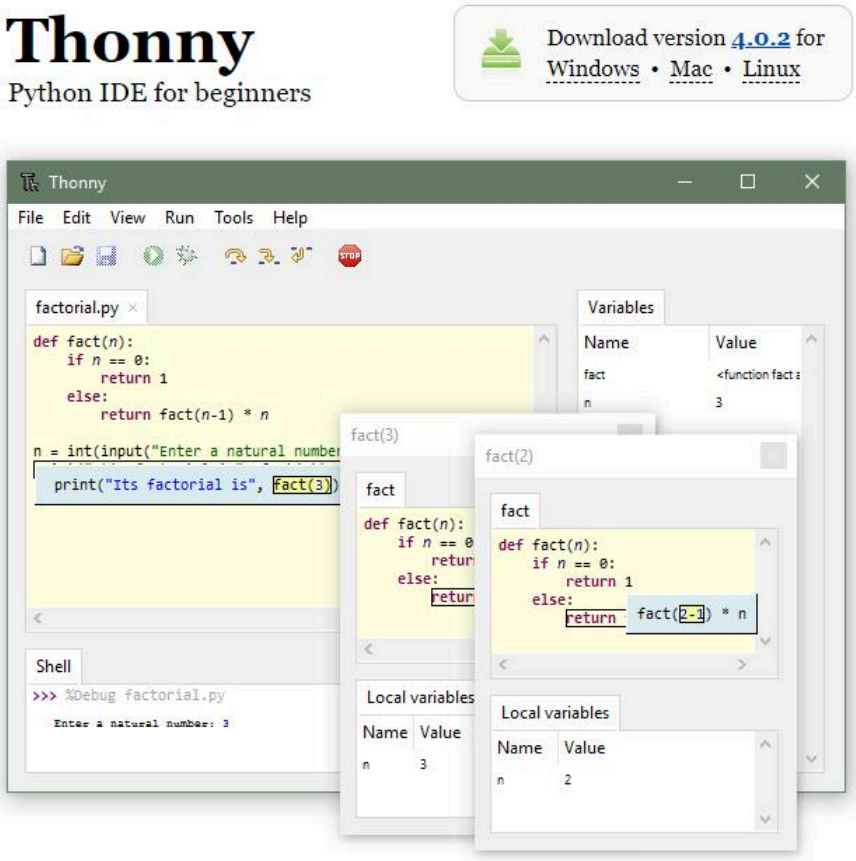
7. Wait for programming.



(/wiki/File:ESP32-S3-GEEK-MPY_01.png)

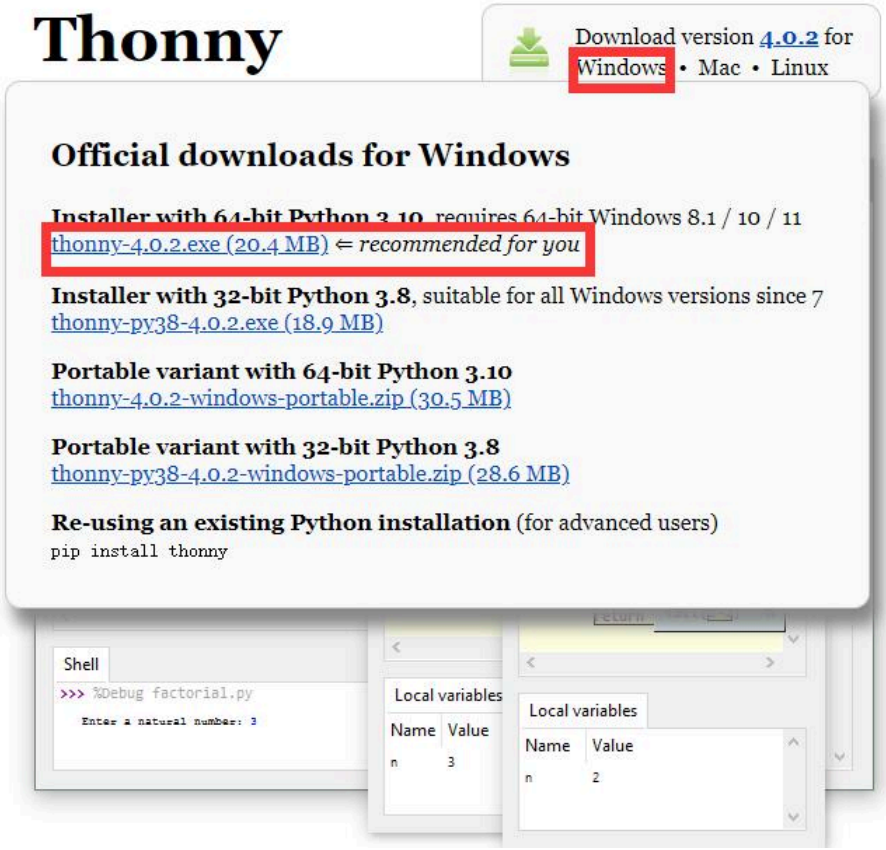
Install Thonny

1. Open official Thonny website (https://thonny.org/).



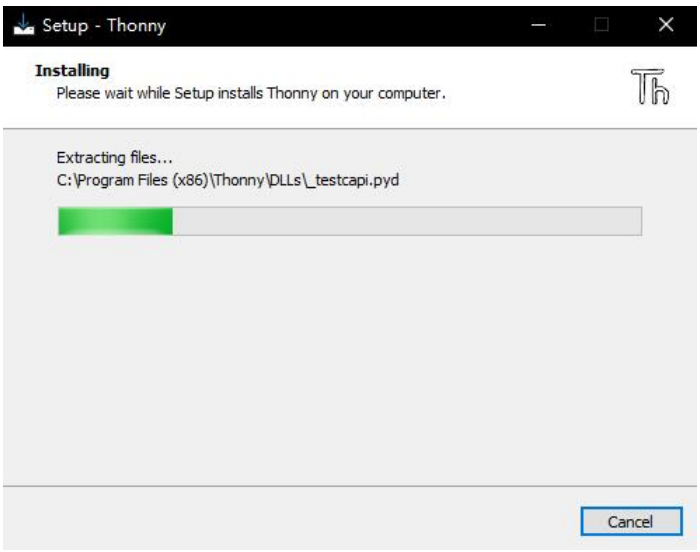
(/wiki/File:ESP32-S3-Pico_58.jpg)

2. Select the corresponding system and version, here I choose "Windows", and the mouse has to move to Windows, only then will the corresponding information be displayed.



(/wiki/File:ESP32-S3-Pico_55.jpg)

3. Just install it by default.



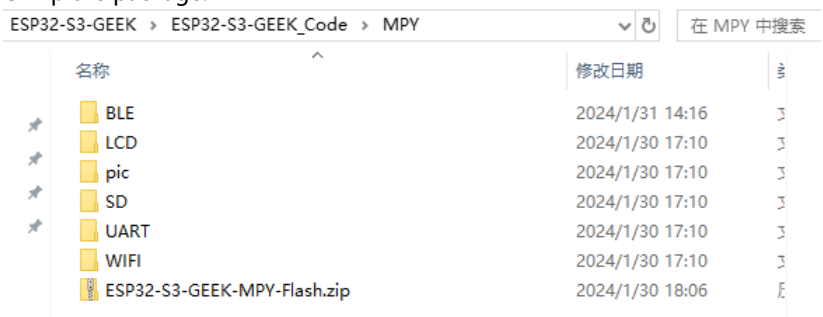
(/wiki/File:ESP32-S3-Pico_56.jpg)

4. Finish.

(/wiki/File:ESP32-S3-Pico_57.jpg)

Demo Download

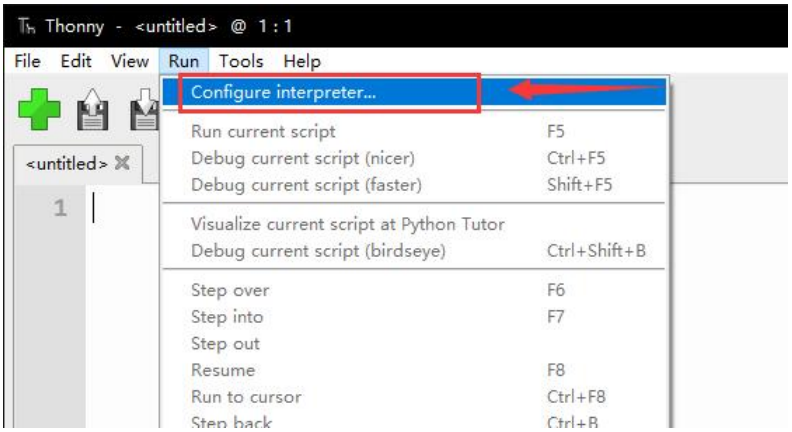
1. Click here to download (https://files.waveshare.com/wiki/ESP32-S3-GEEK/ESP32-S3-GEEK_Code.zip).
2. Unzip the package.



(/wiki/File:ESP32-S3-GEEK-MPY_09.png)

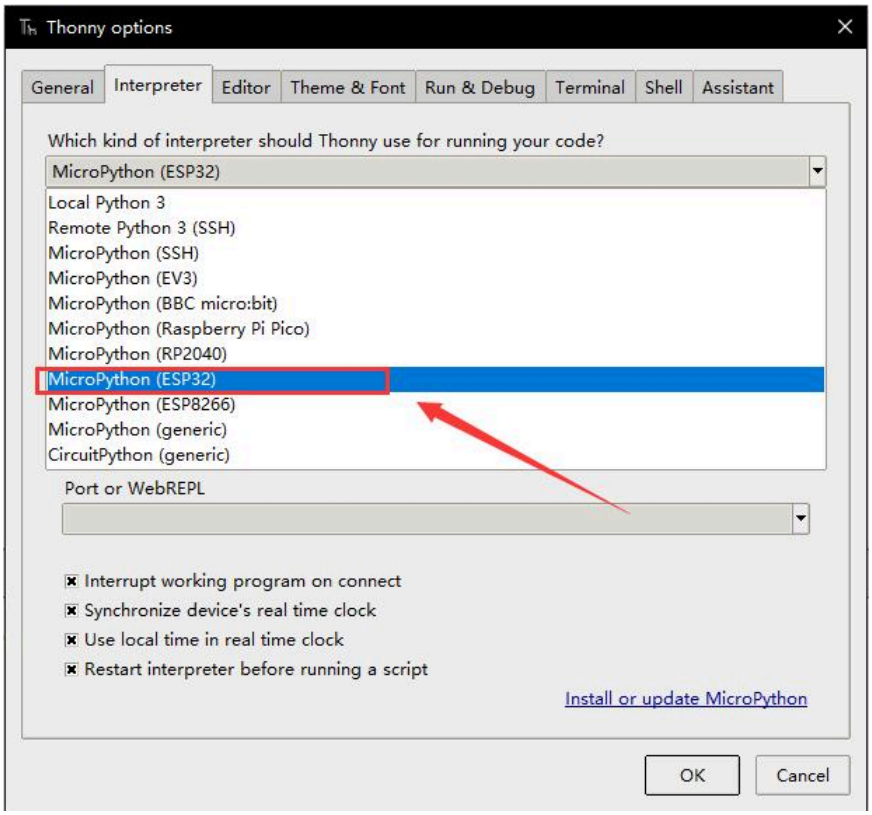
Run the Demo

1. Open Thonny and choose Configure Interpreter.



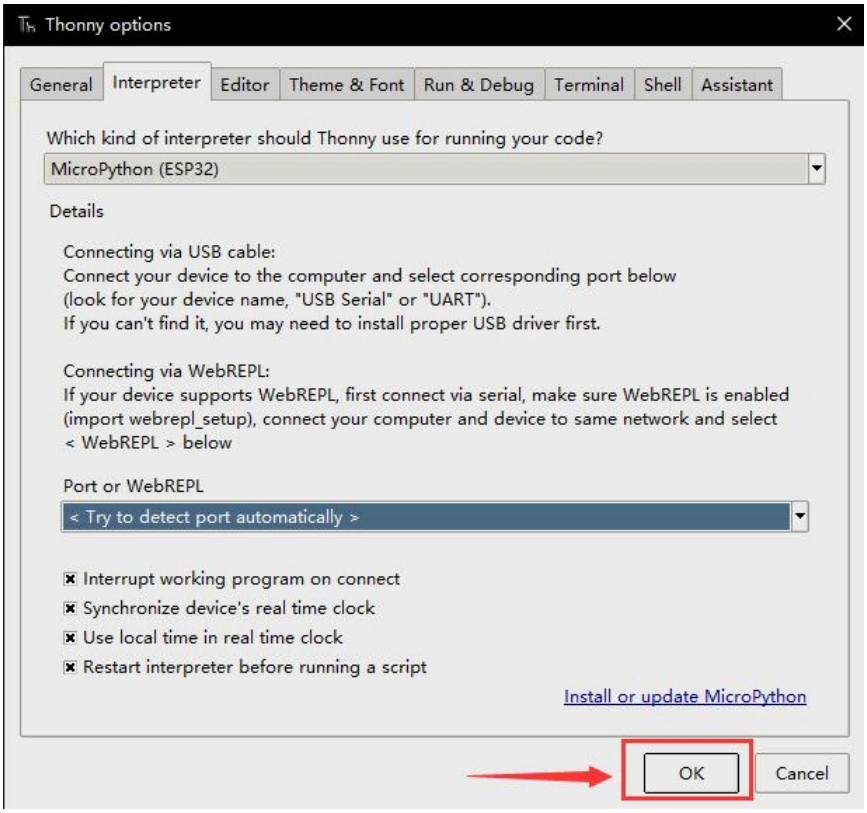
(/wiki/File:ESP32-S3-Pico_90.jpg)

2. Choose MicroPython (ESP32) as the Interpreter.



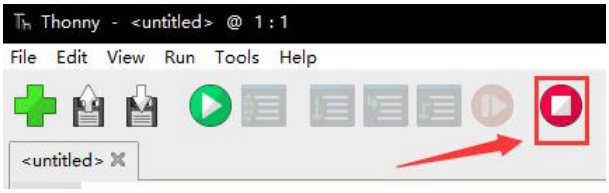
(/wiki/File:ESP32-S3-Pico_91.jpg)

3. Click OK to save.



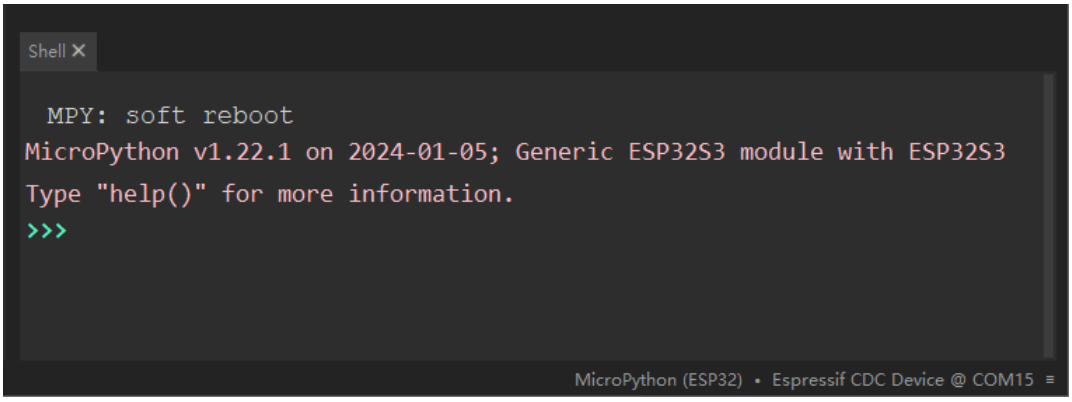
(/wiki/File:ESP32-S3-Pico_92.jpg)

4. Click Stop or Ctrl + F2.



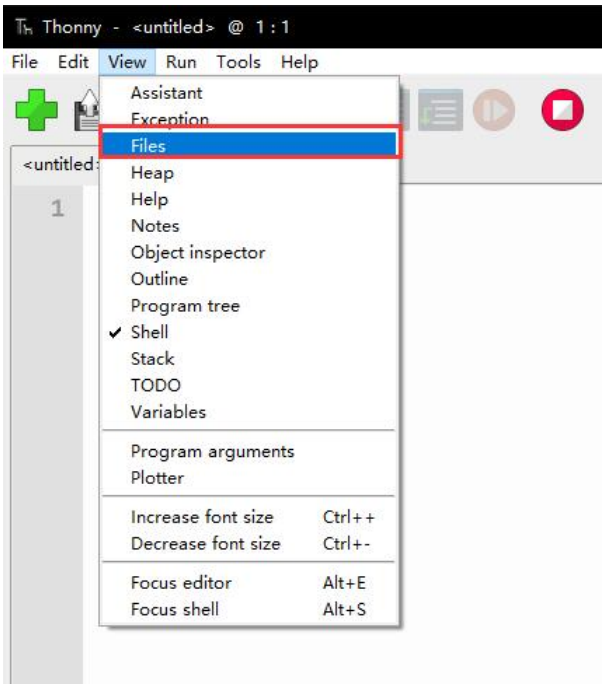
(/wiki/File:ESP32-S3-Pico_93.jpg)

5. You can see the MicroPython version information and the board name.



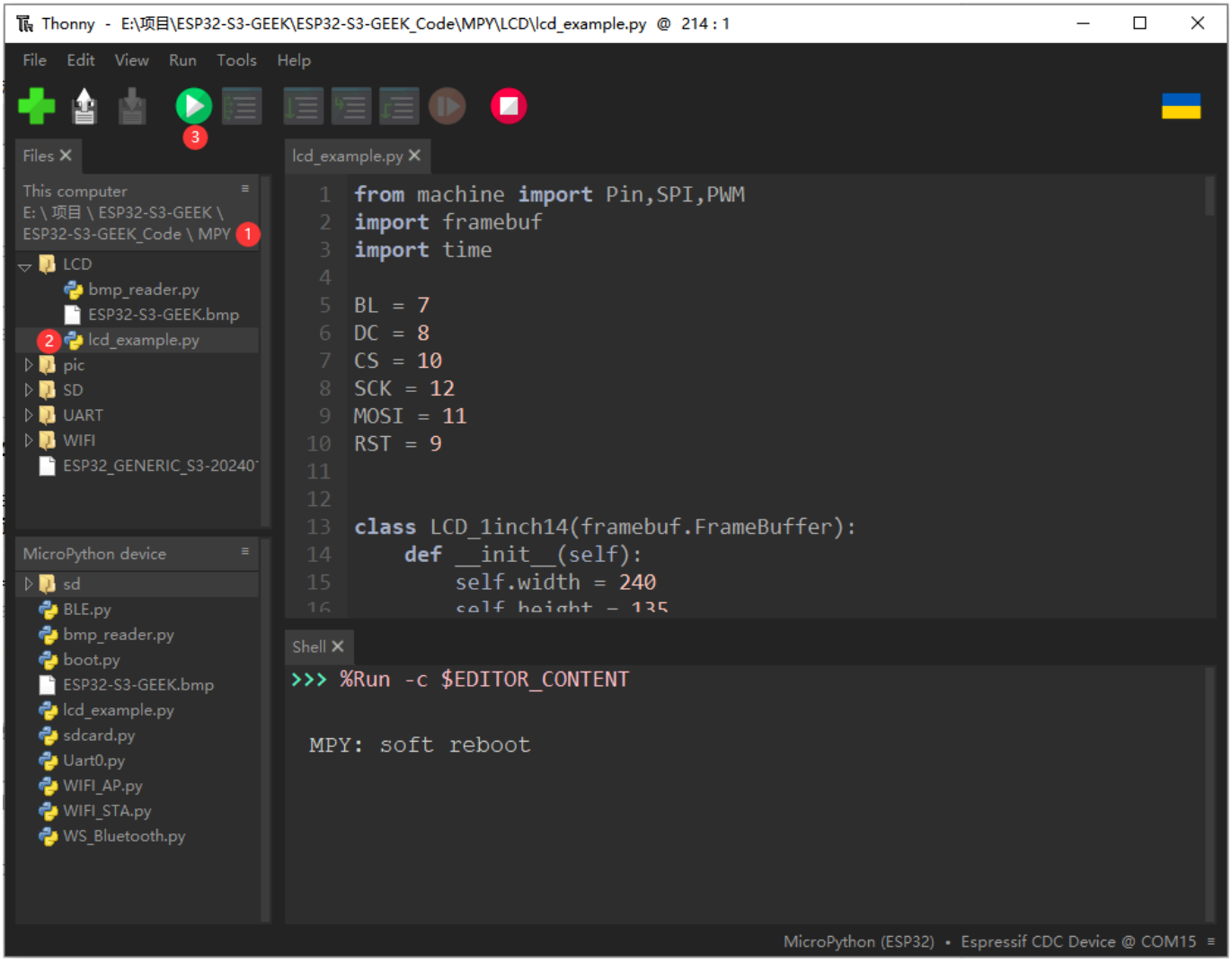
(/wiki/File:ESP32-S3-GEEK_94.png)

6. Open the files.



(/wiki/File:ESP-S3-Pico_95.jpg)

7. Choose our demo file and open a demo, here we choose a lcd_example.py demo.



(/wiki/File:ESP-32-GEEK_96.png)

- ①Current local file directory.
- ②The demo we want to open is lcd_example.py.

- ③Click to run or press "F5".

8. Wait a moment and you can see the LCD screen of the ESP32-S3-GEEK displaying the ESP32-S3-GEEK.bmp image.

Arduino Sample Demo

> **Note**: Before using the Arduino demo, please check whether the Arduino environment and the download setting are correctly configured. For more details, you can refer to Arduino (/wiki/ESP32-S3-GEEK#Arduino).

WIFI

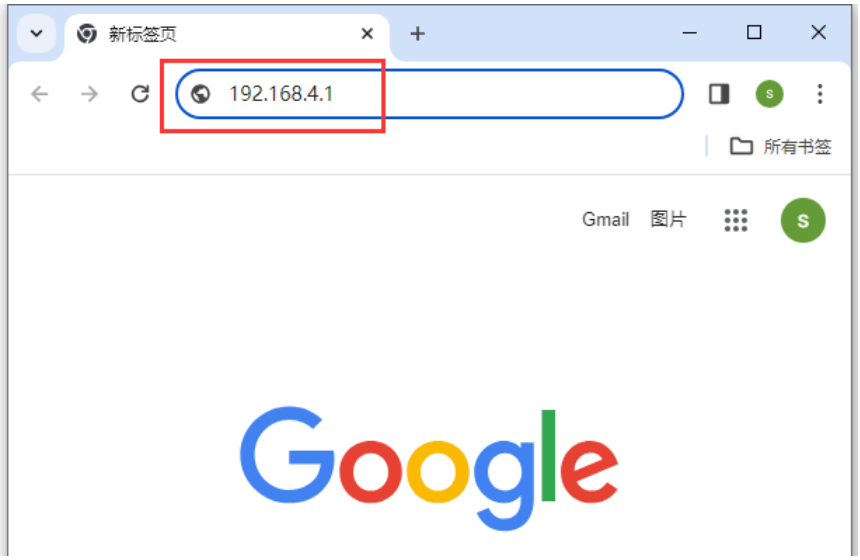
01-WIFI_AP_LCD

```
This code enables ESP32-S3-GEEK to open an Access Point (AP) mode for Wi-Fi. Once a PC is connected to its Wi-Fi network, you ca
n log in to its IP address via a web interface to control the display of images on ESP32-S3-GEEK's LCD.
```

- "ssid" is the AP name (ESP32-S3-GEEK) created by ESP32-S3-GEEK, and the "password" is the password (Waveshare) for connecting to AP.

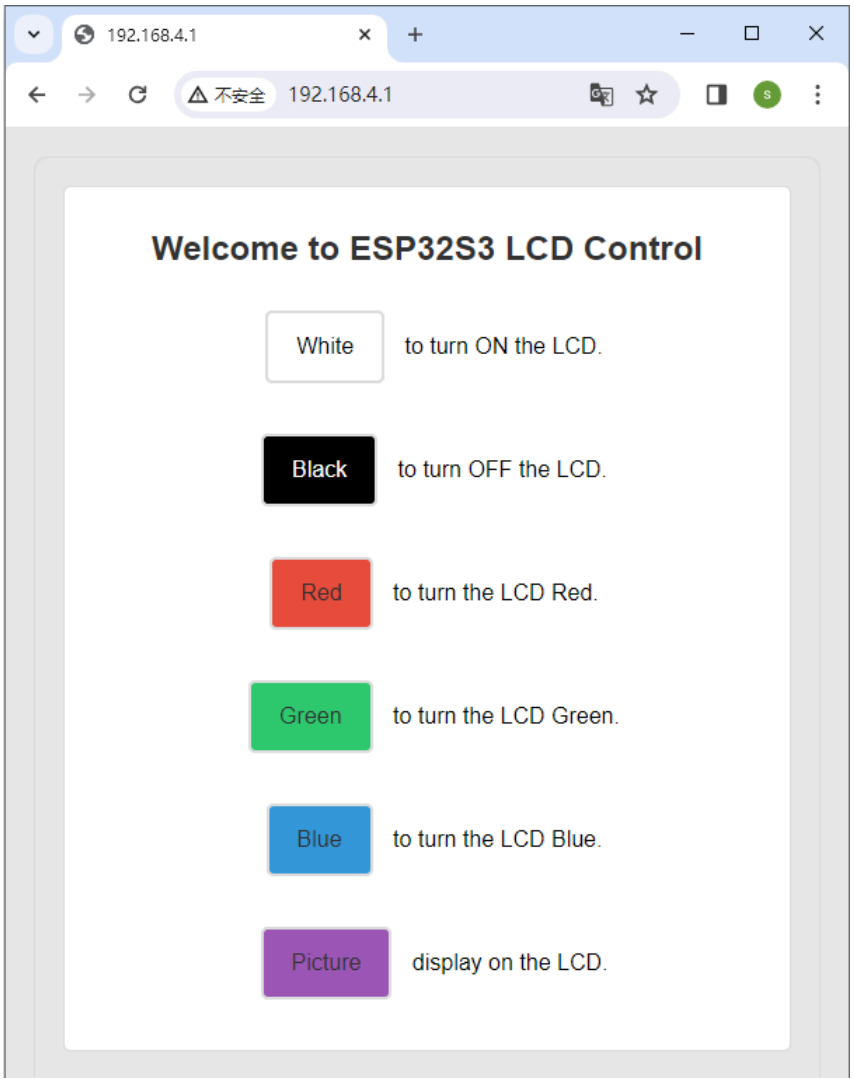
(/wiki/File:ESP32-S3-GEEK_Demo01.png)

- Connect to the AP of the ESP32-S3-GEEK Demo on the PC, and enter the password "Waveshare".
- The LCD will display the IP address of the HTTP server, and you can use the browser to log in the IP: 192.168.4.1.



(/wiki/File:ESP32-S3-GEEK_Demo03.png)

- Control the LCD of the ESP32-S3-GEEK through the button on the server, and you can observe the LCD by clicking on different buttons. For LCD function, you can refer to LCD demo description (https://www.waveshare.com/wiki/1.5inch_LCD_Module#Demo_Remarks_3).

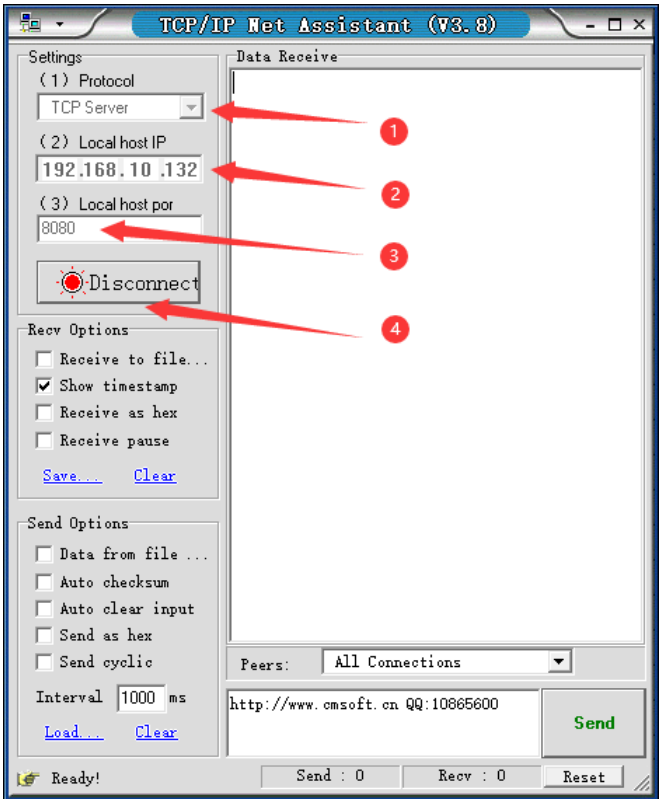


(/wiki/File:ESP32-S3-GEEK_Demo04.png)

02-WIFI_TCP_Client

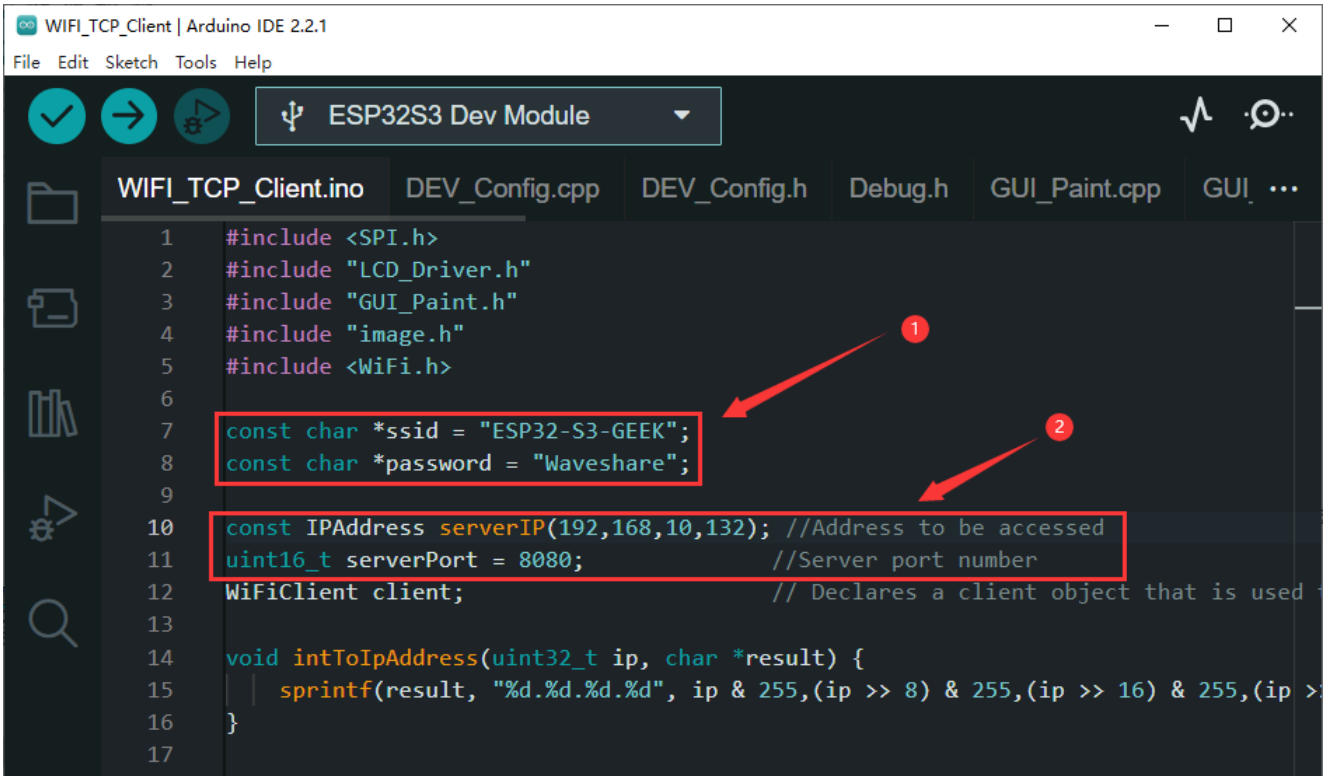
```
With this demo, ESP32-S3-GEEK enters WIFI's STA mode and connects to the same WIFI as the PC or mobile phone. As TCP Client acce
sses the TCP Server created by the PC or mobile phone, TCP communication is established, and the LCD displays the received conte
nt.
```

Using NetAssist.exe (https://files.waveshare.com/upload/f/f6/NetAssist.zip) to enter the TCP Server, the Local host IP is the default one, and the Local host Port is 8080. Click on "Connect" for TCP communication with ESP32-S3-GEEK(TCP Client).



(/wiki/File:ESP32-S3-GEEK_Demo06.png)

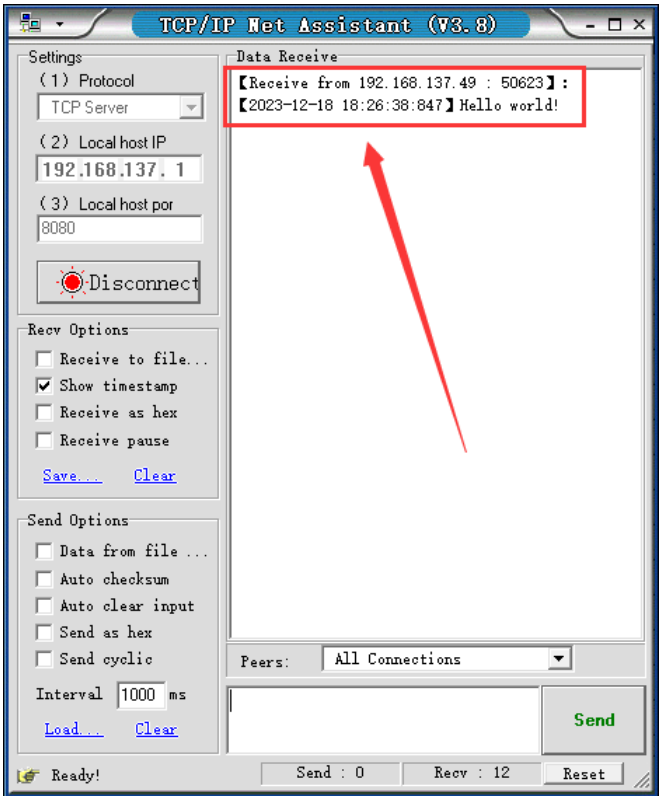
- The ESP32-S3-GEEK connects to the same WIFI network as the PC in STA mode. (**The WIFI network that ESP32-S3-GEEK connects to should operate on the 2.4GHz frequency band.** If there's no 2.4GHz frequency band available, the PC can create a hotspot with the network band set to "Any available frequency.") Remember to change the SSID and password to match the WIFI name and password you wish to connect to.



(/wiki/File:ESP32-S3-GEEK_Demo05.png)

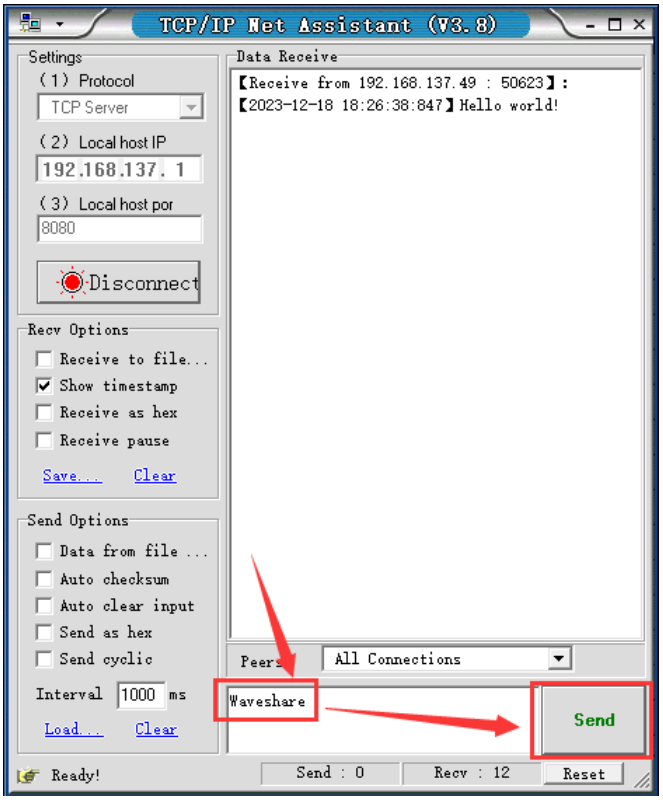
- **Modify the serverIP to the TCP Server's IP address obtained from using NetAssist.exe, and keep the serverPort as 8080.** Upon successful connection, the ESP32-S3-GEEK will send a TCP message "Hello world" to the TCP Server, and the LCD will

display "Access successful".



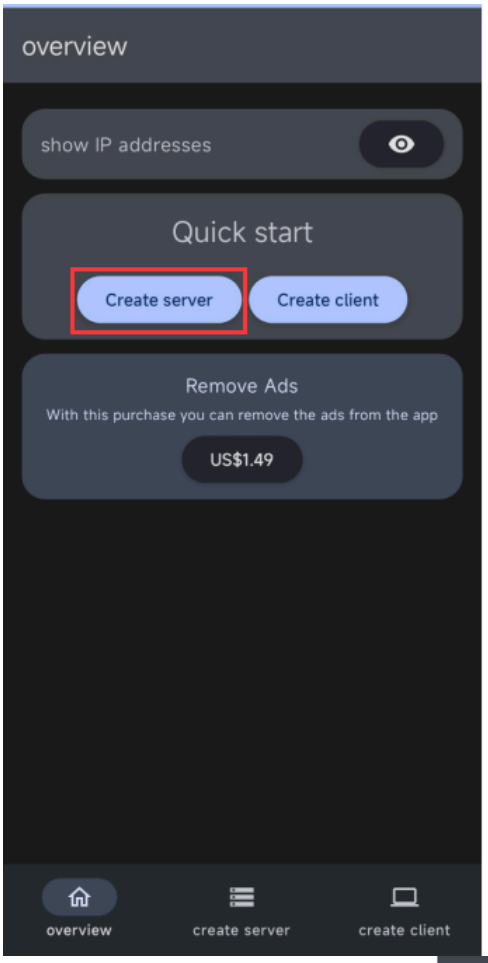
(/wiki/File:ESP32-S3-GEEK_Demo07.png)

- TCP Server sends the TCP message to the ESP32-S3-GEEK on the PC. If sent successfully, ESP32-S3-GEEK as the TCP Client can receive the message and display it on the LCD, and you can observe whether the LCD displays the message.

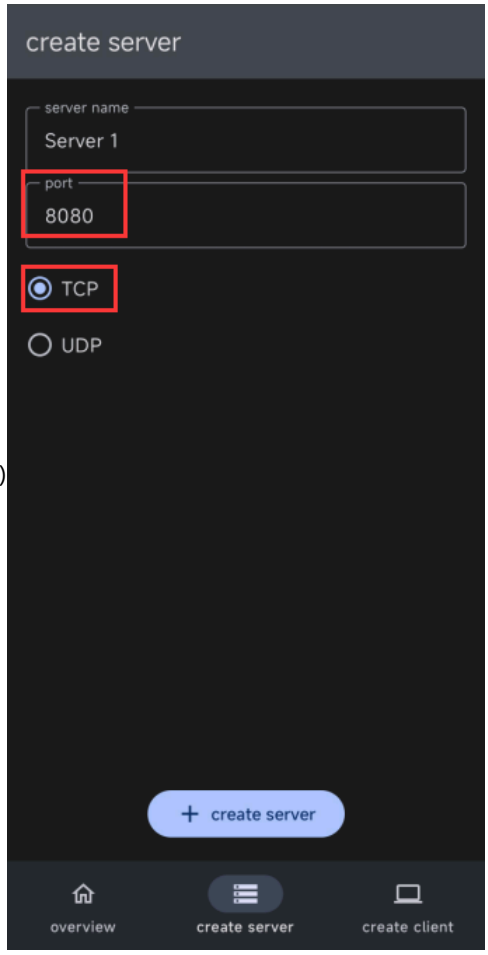


(/wiki/File:ESP32-S3-GEEK_Demo-tcP_07.png)

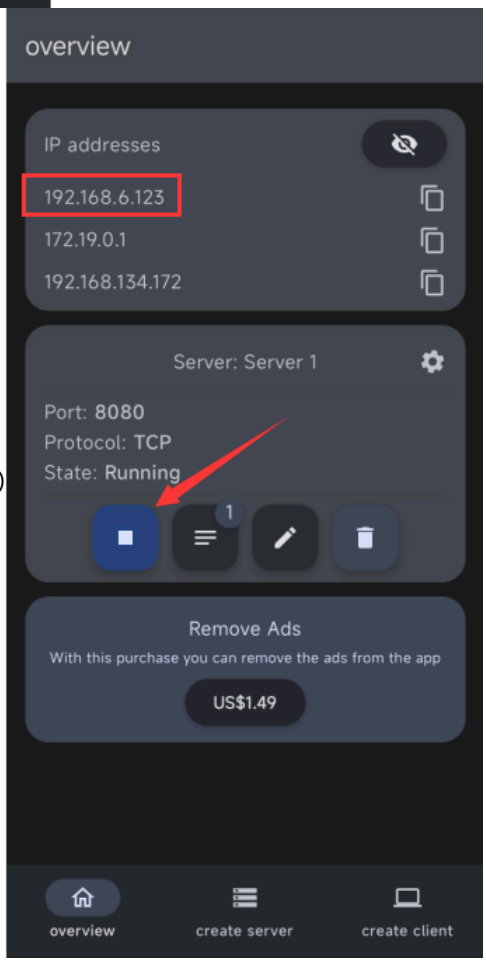
- You can also use your phone to create a hotspot with the name and password as "ssid" and "password", respectively, operating on the 2.4GHz frequency band. After setting up the hotspot, you can use a TCP debugging tool (https://files.waveshare.com/wiki/ESP32-S3-GEEK/Tcp%E8%B0%83%E8%AF%95%E5%8A%A9%E6%89%8B.zip) to establish TCP communication with ESP32-S3-GEEK.



(/wiki/File:ESP32-S3-GEEK_Demo08.png)

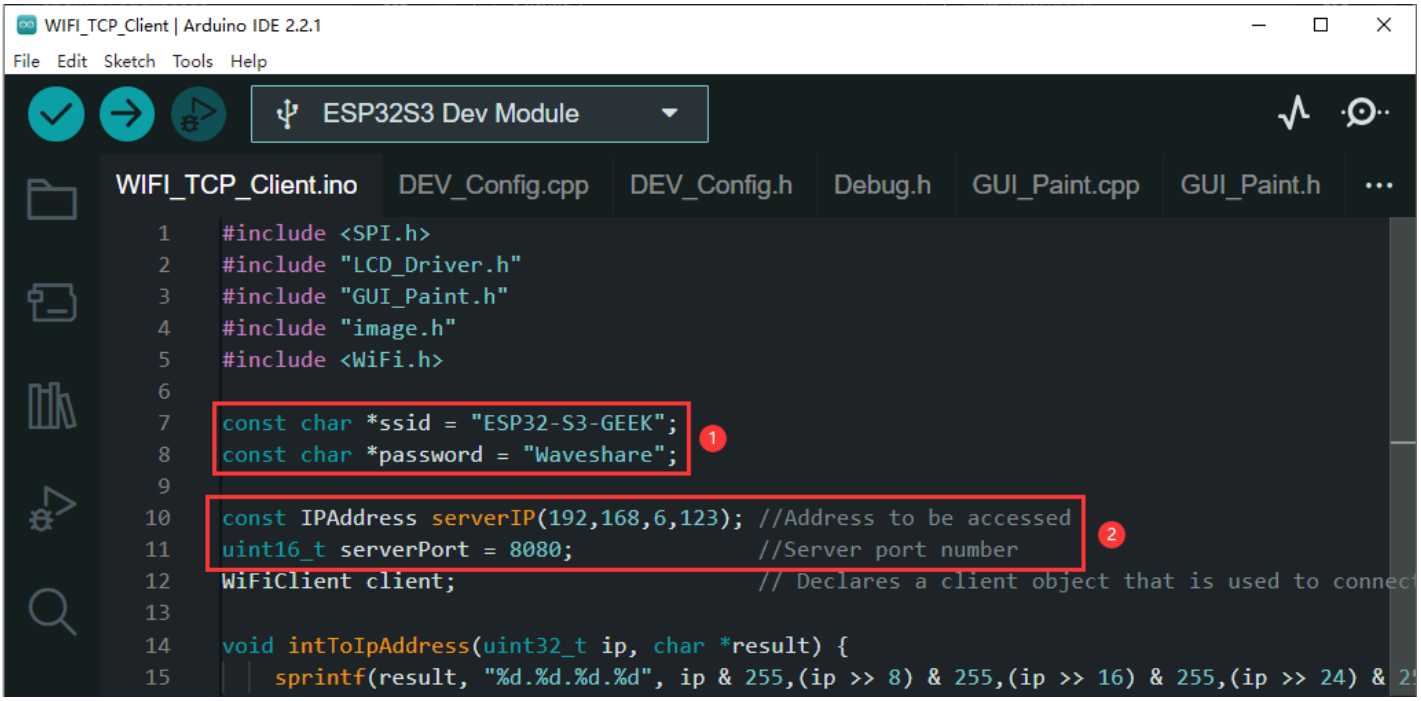


(/wiki/File:ESP32-S3-GEEK_Demo09.png)



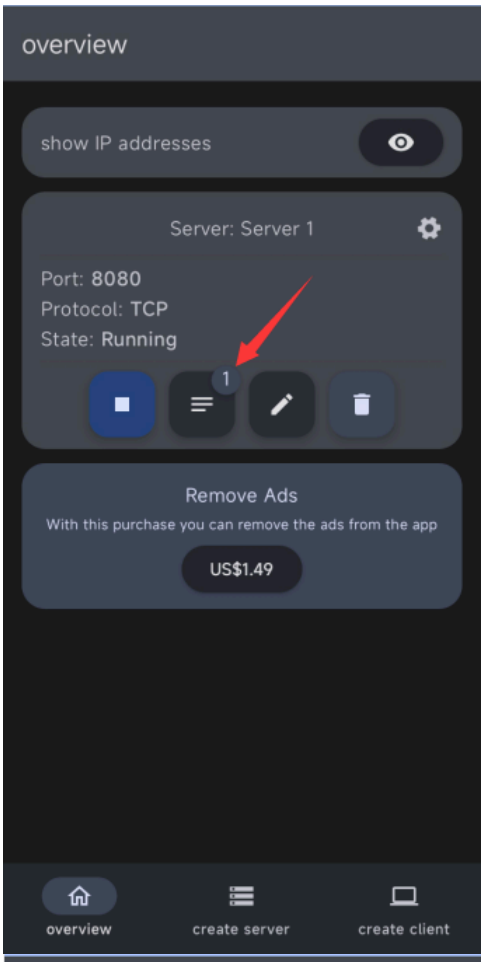
(/wiki/File:ESP32-S3-GEEK_Demo10.png)

- Modify the WIFI connected by the code as the hotspot of the smartphone, serverIP is the IP (192.168.6.123) of the TCPserver created by the smartphone in the above picture.

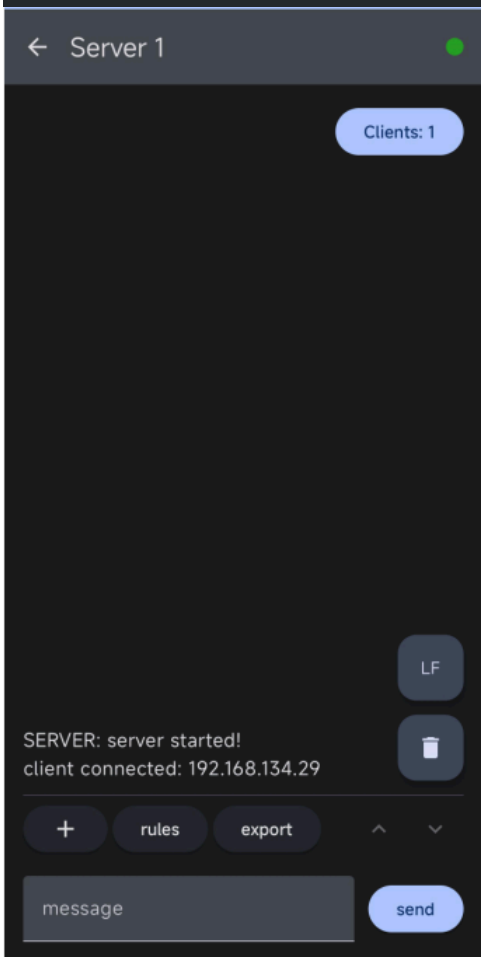


(/wiki/File:ESP32-S3-GEEK_Demo-IP07.png)

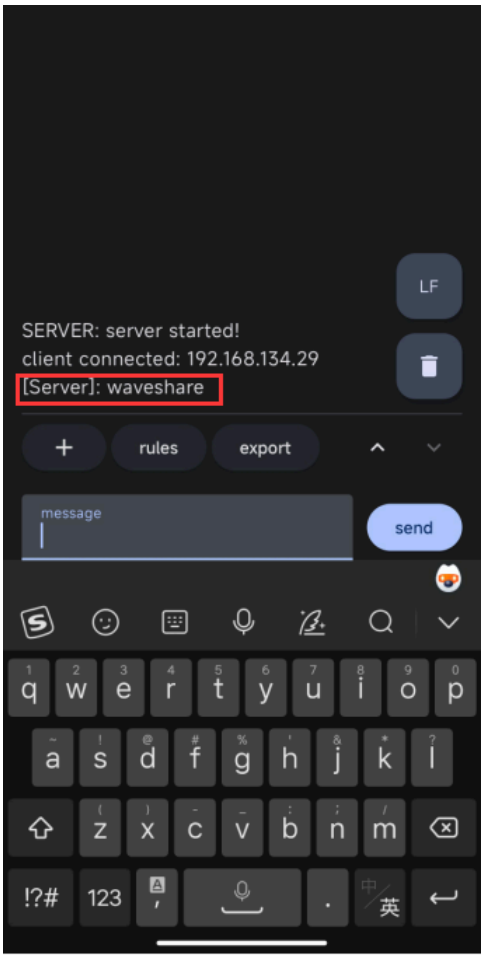
- After programming the demo, once connected, the TCP Server will receive a TCP message "Hello world" from ESP32-S3-GEEK. The LCD will display "Access successful". Using a mobile device's TCP Server, you can send TCP messages to ESP32-S3-GEEK. If the transmission is successful, ESP32-S3-GEEK, acting as a TCP Client, will receive the message and display its content on the LCD. You could observe whether the LCD shows the message.



(/wiki/File:ESP32-S3-GEEK_Demo-ip08.png)



(/wiki/File:ESP32-S3-GEEK_Demo-ip09.png)

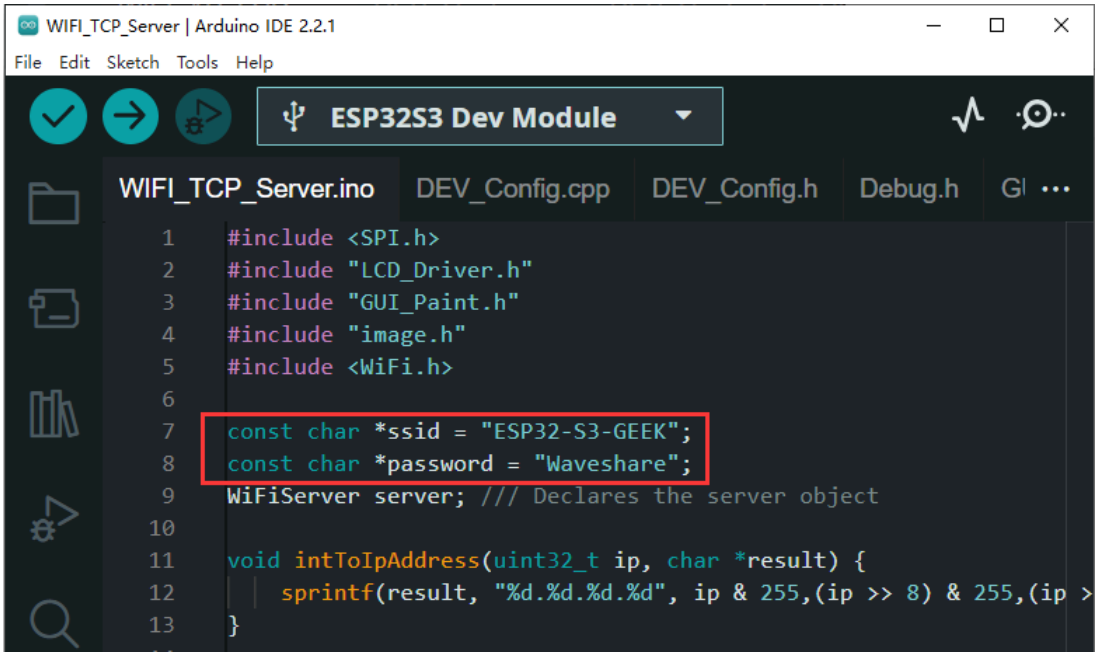


(/wiki/File:ESP32-S3-GEEK_Demo-ip10.png)

03-WIFI_TCP_Server

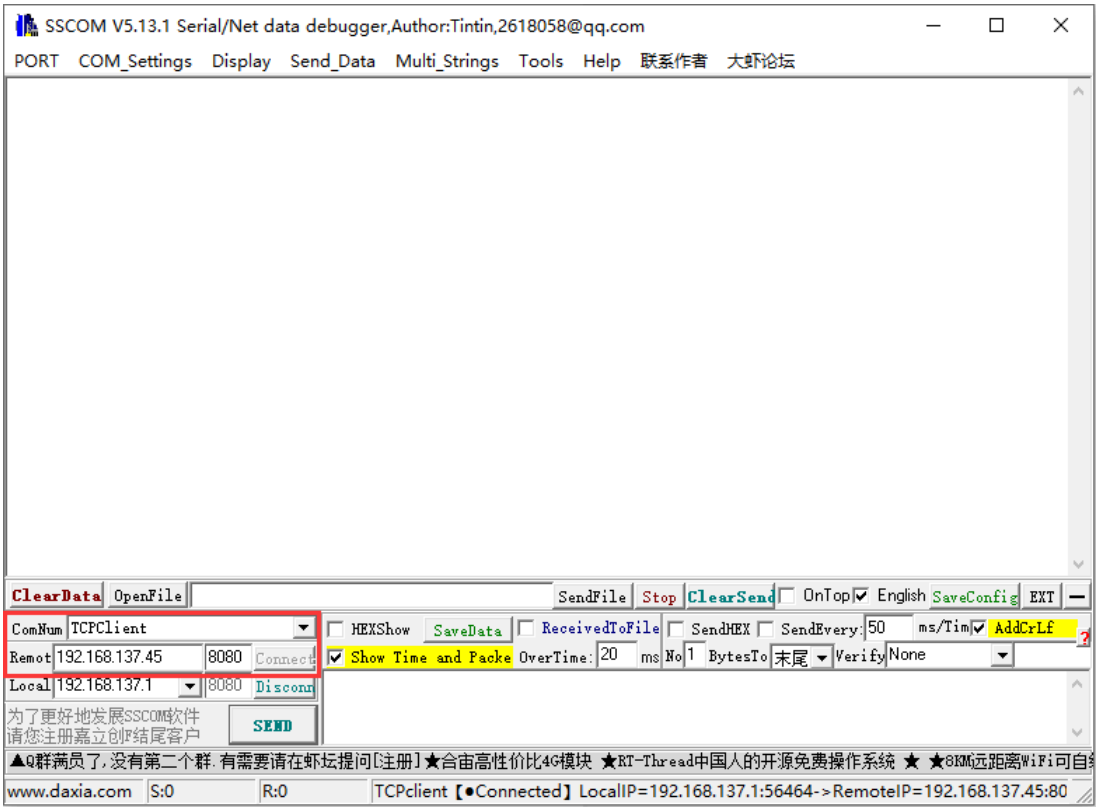
```
With this demo, ESP32-S3-GEEK enters WIFI's STA mode. After connecting the hotspot on the PC, creates a TCP Server, and the PC c
reates a TCP Client to access ESP-S3-GEEK. Then, the TCP communication is established between them, and GEEK displays the receiv
ed content on the LCD.
```

- ESP32-S3-GEEK connects to the same WIFI on the STA mode. (The WIFI network that ESP32-S3-GEEK connects to should operate on the 2.4GHz frequency band. If there's no 2.4GHz frequency band available, the PC can create a hotspot with the network band set to "Any available frequenc".) Remember to change the ssid and password to match the WIFI name and password you wish to connect to.



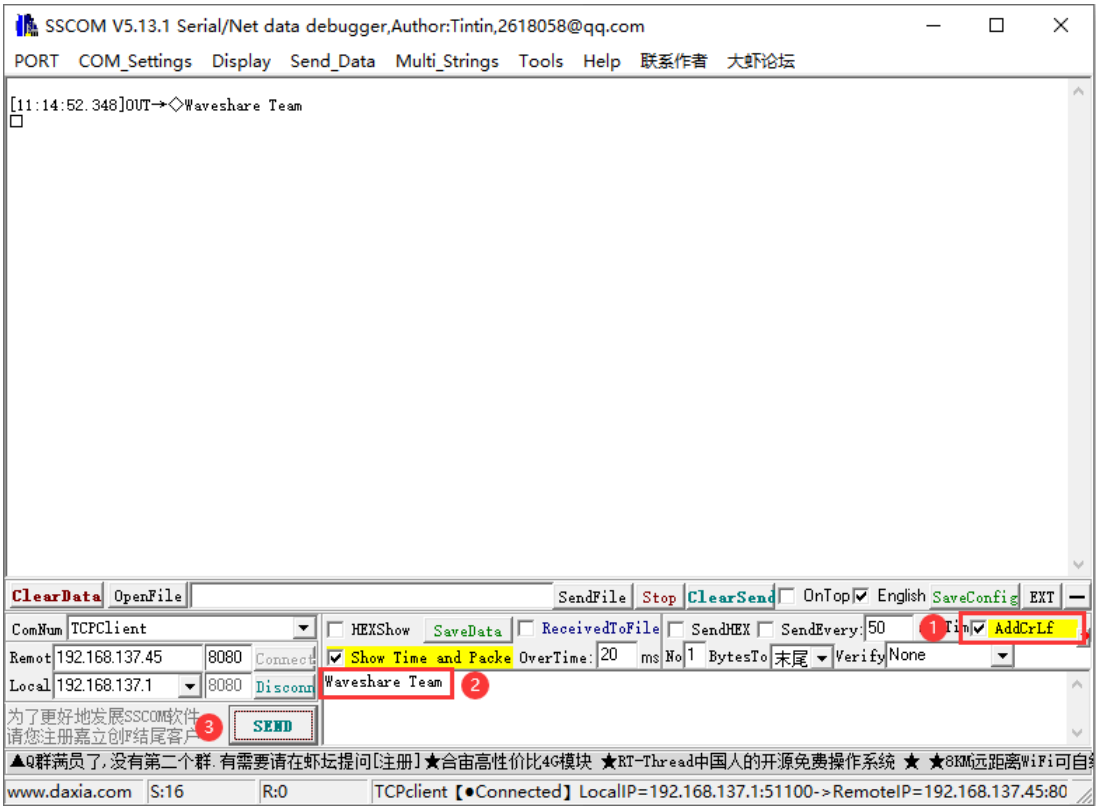
(/wiki/File:ESP32-S3-GEEK_Demo12.png)

- After connecting to the WIFI, **the LCD will display TCP ServerIP**, and use SSCOM to open TCP Client for ESP32-S3-GEEK(TCP Server) connection with TCP.



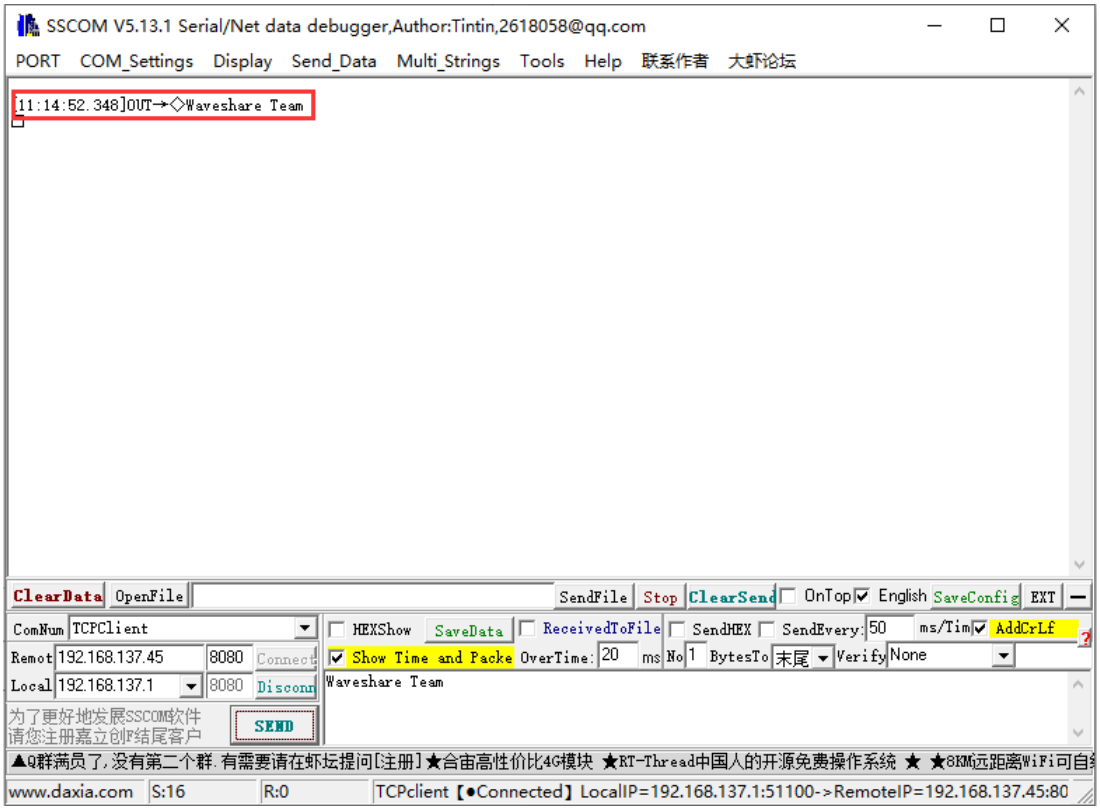
(/wiki/File:ESP32-S3-GEEK_Demo13.png)

- Send TCP message through TCP Client to ESP32-S3-GEEK(TCP Server), **check the "AddCrLf" options while sending the message**.



(/wiki/File:ESP32-S3-GEEK_Demo14.png)

- After sending successfully, ESP32-S3-GEEK as the TCP server receives the message and displays it on the LCD, and you can check whether the LCD displays the received message.

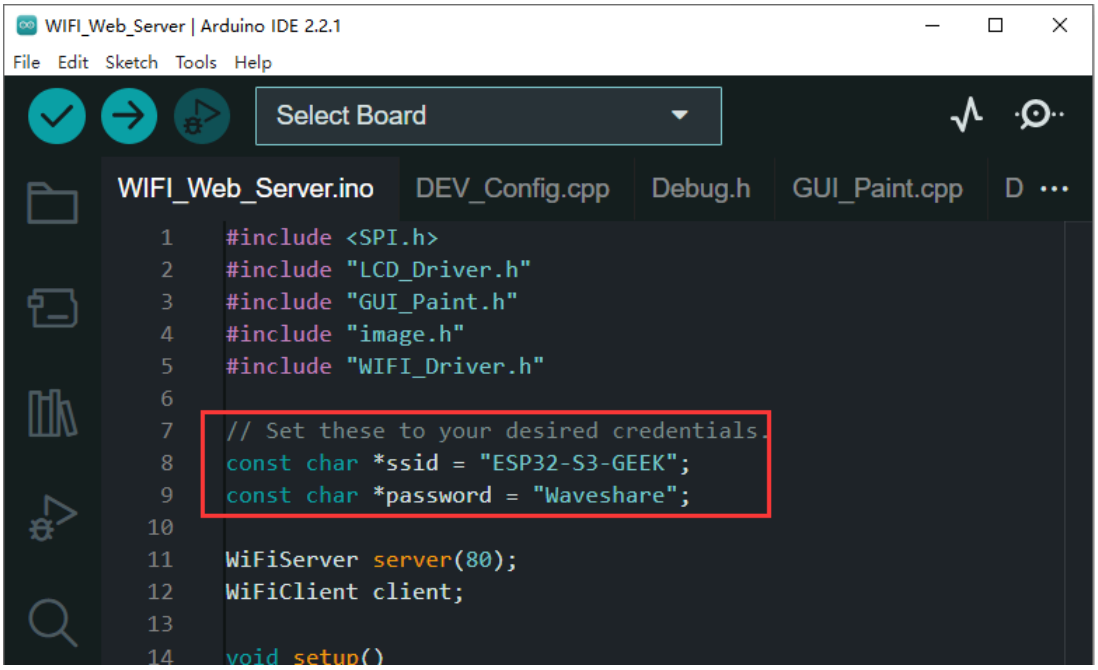


(/wiki/File:ESP32-S3-GEEK_Demo15.png)

04-WIFI_Web_Server

```
This code allows the ESP32-S3-GEEK to open its WIFI in AP mode. After a PC connects to this WIFI, open a serial debugging assist
ant. Through the HTTP webpage created by the ESP32-S3-GEEK, messages can be sent to the GEEK. Observe the received content on bo
th the serial debugging assistant and the LCD.
```

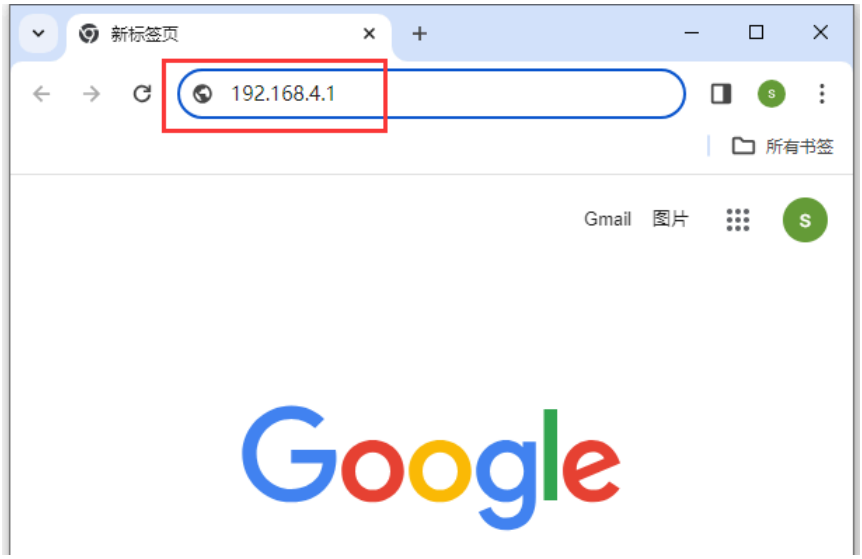
- The "ssid" is the name of the AP created by the ESP32-S3-GEEK, and the "password" is the password required to connect to the AP.



(/wiki/File:ESP32-S3-GEEK_Demo17.png)

- Using the PC to connect to the AP of the ESP32-S3-GEEK.

- The LCD will display the HTTP server's IP address. Use a web browser to log in to the IP address: 192.168.4.1.

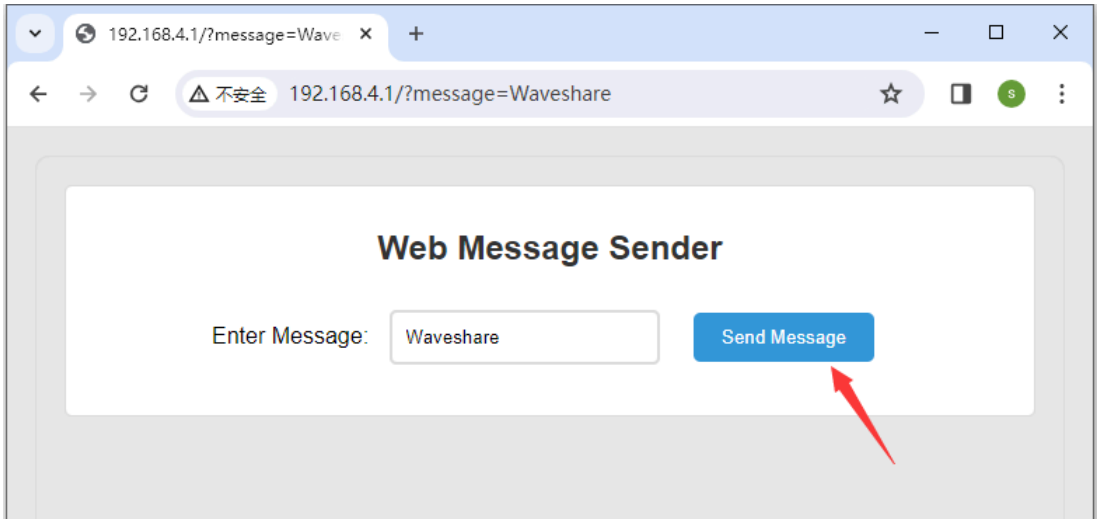


(/wiki/File:ESP32-S3-GEEK_Demo03.png)

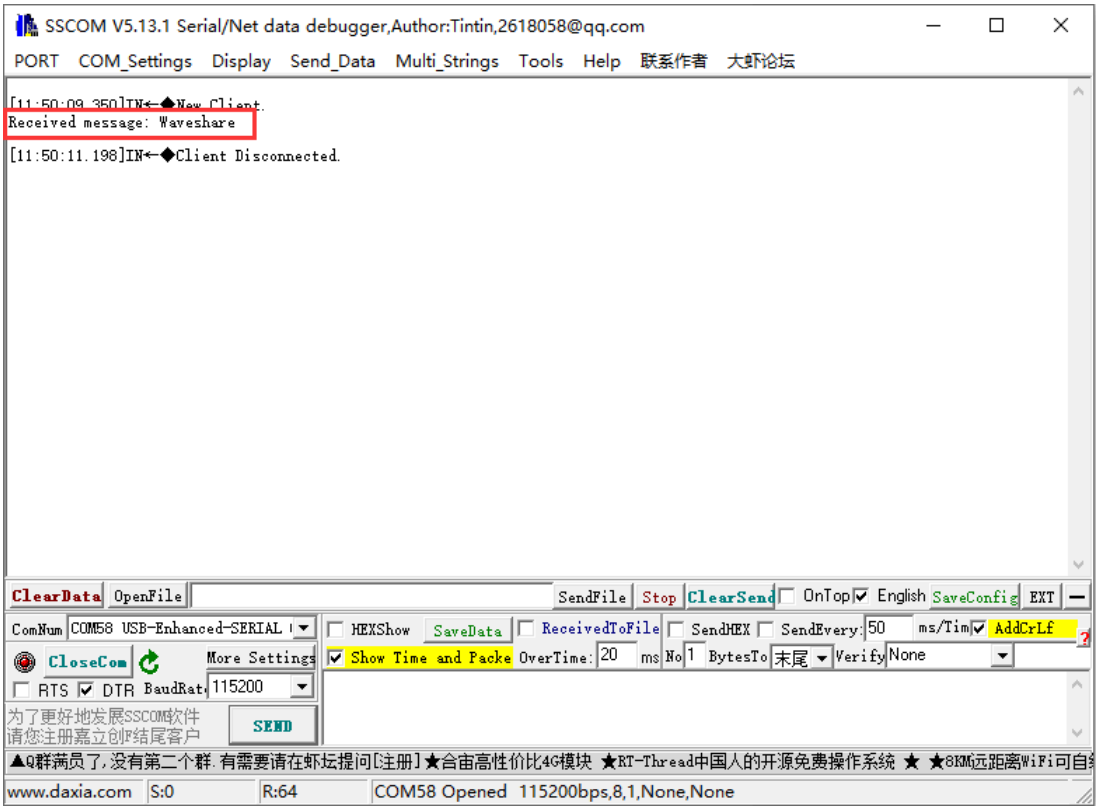
- Using the ESP32-S3-GEEK's UART interface, connect it to the PC via a USB to UART tool (https://www.waveshare.com/ch343-usb-uart-board.htm), then open a serial debugging assistant.

(/wiki/File:ESP32-S3-GEEK_Demo20.png)

- You can input text content on the HTTP web interface and send an HTTP request to the ESP32-S3-GEEK. The received content can be displayed on both the serial debugging assistant and the LCD.



(/wiki/File:ESP32-S3-GEEK_Demo21.png)



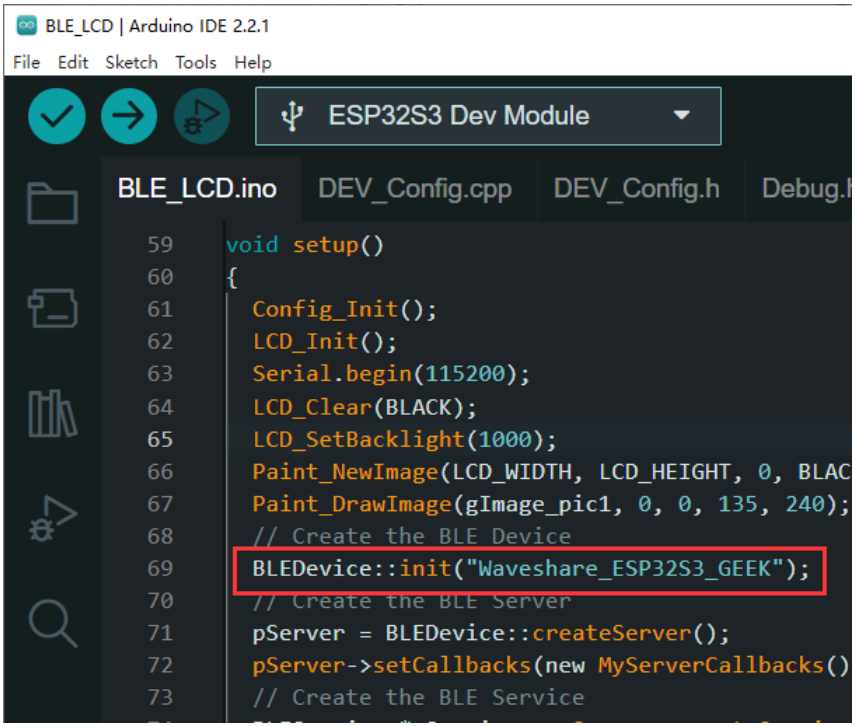
(/wiki/File:ESP32-S3-GEEK_Demo22.png)

BLE

01-BLE_LCD

```
With this demo, ESP32-S3-GEEK enters BLE. Using a mobile phone to open the Bluetooth debugging assistant, connect to ESP32-S3-GEEK, and establish BLE communication with the mobile phone. The message of sending and receiving is displayed on the LCD.
```

- In the code BLEDevice::init("Waveshare_ESP32S3_GEEK"), "Waveshare_ESP32S3_GEEK" is the Bluetooth device name.



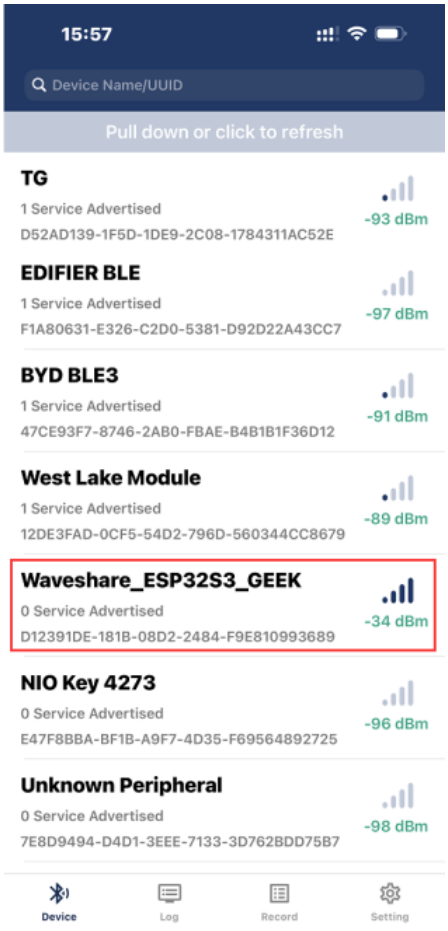
(/wiki/File:ESP32-S3-GEEK_Demo-20.png)

- Use the mobile phone to open the Bluetooth debugging assistant to scan and connect the device.

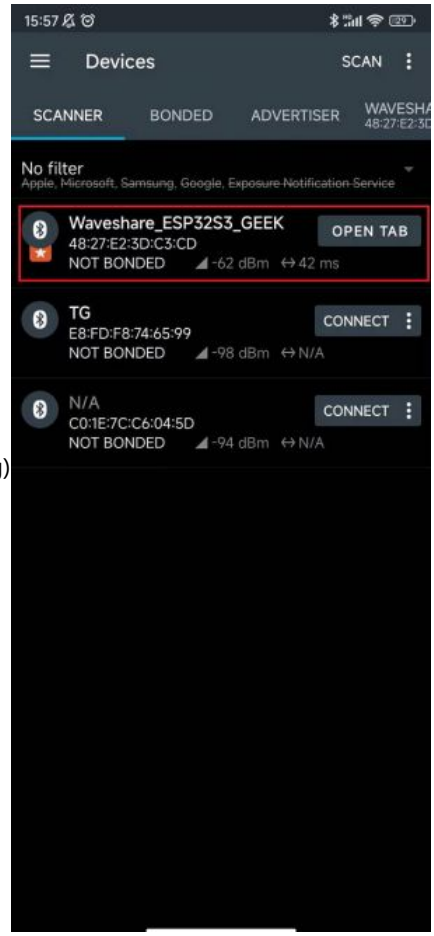
IOS:

Android:

(/wiki/File:ESP32-S3-GEEK_Demo-21.png)



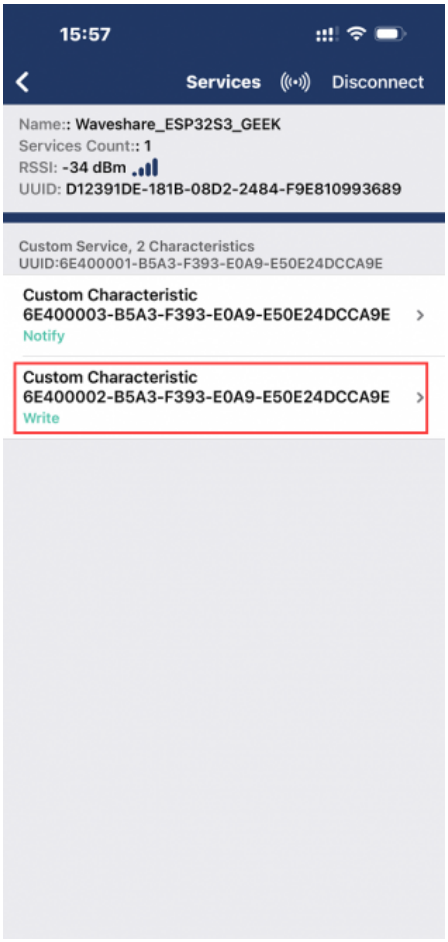
(/wiki/File:ESP32-S3-GEEK_Demo-22.png)



(/wiki/File:ESP32-S3-GEEK_Demo-122.jpg)

- Use the Bluetooth debugging assistant to send the message to ESP32-S3-GEEK, and the ESP32-S3-GEEK will display the message on the LCD after receiving it.

IOS:

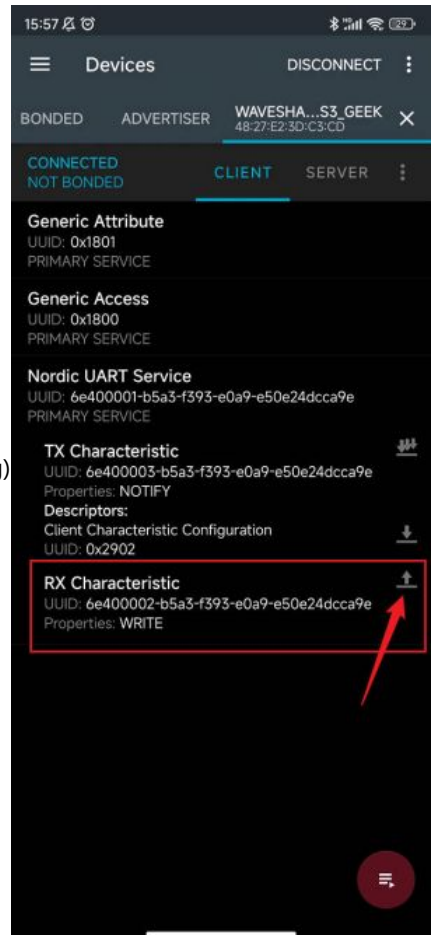


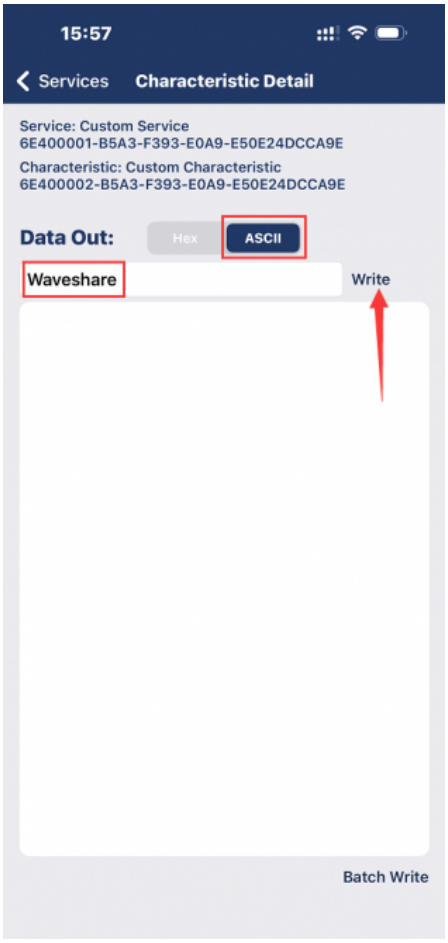
(/wiki/File:ESP32-S3-GEEK_Demo-24.jpg)

Android:

(/wiki/File:ESP32-S3-GEEK_Demo-21.png)

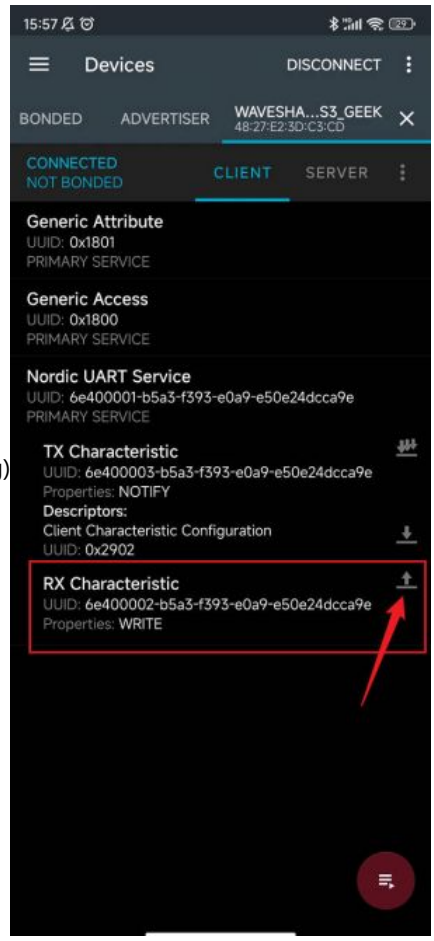
(/wiki/File:ESP32-S3-GEEK_Demo-23.png)





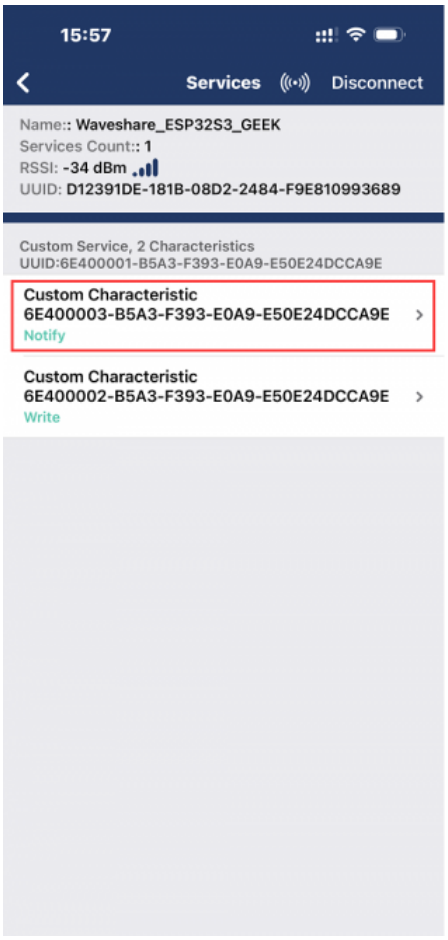
(/wiki/File:ESP32-S3-GEEK_Demo-24.jpg)

(/wiki/File:ESP32-S3-GEEK_Demo-132.png)



- Enter the receiving setting of the Bluetooth debugging assistant on the mobile phone:

IOS:

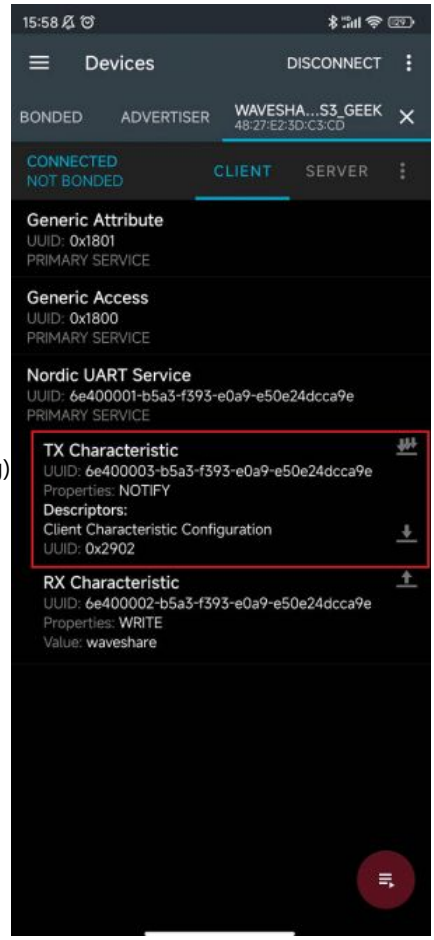


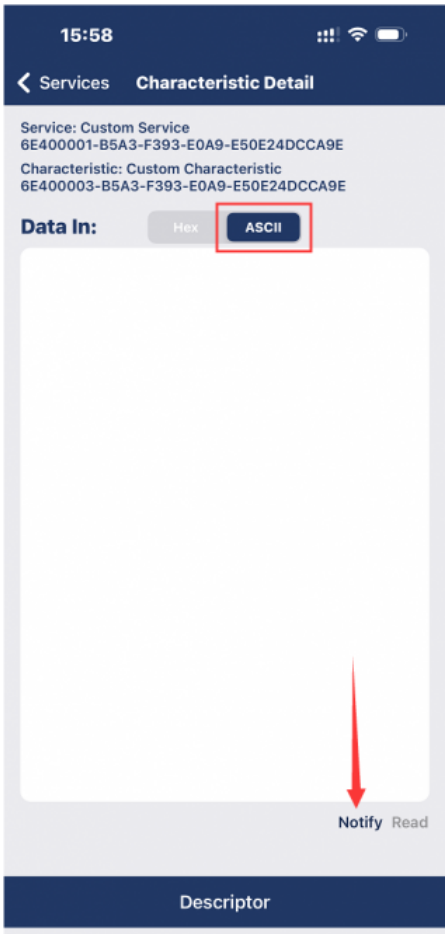
(/wiki/File:ESP32-S3-GEEK_Demo-26.jpg)

Android:

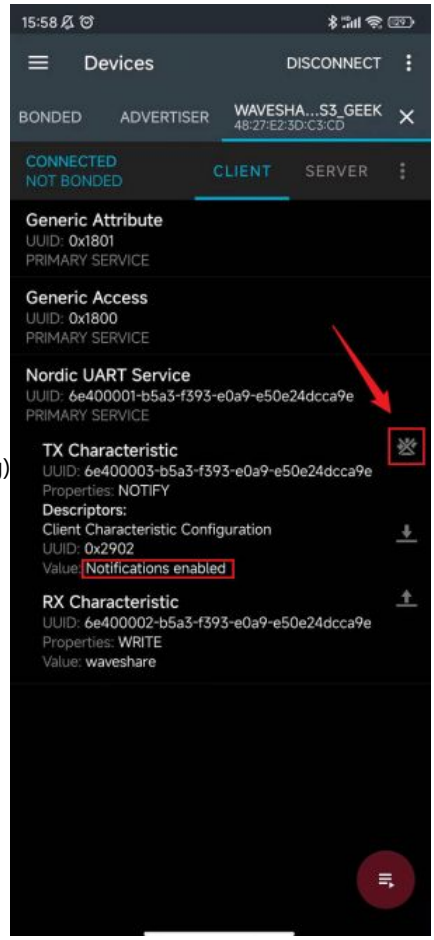
(/wiki/File:ESP32-S3-GEEK_Demo-21.png)

(/wiki/File:ESP32-S3-GEEK_Demo-25.png)



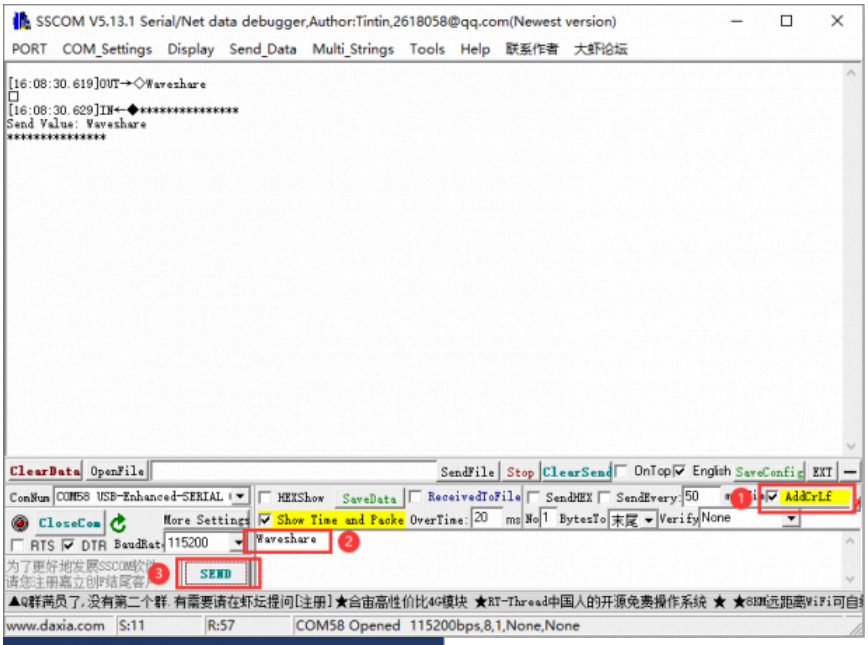


(/wiki/File:ESP32-S3-GEEK_Demo-125.png)

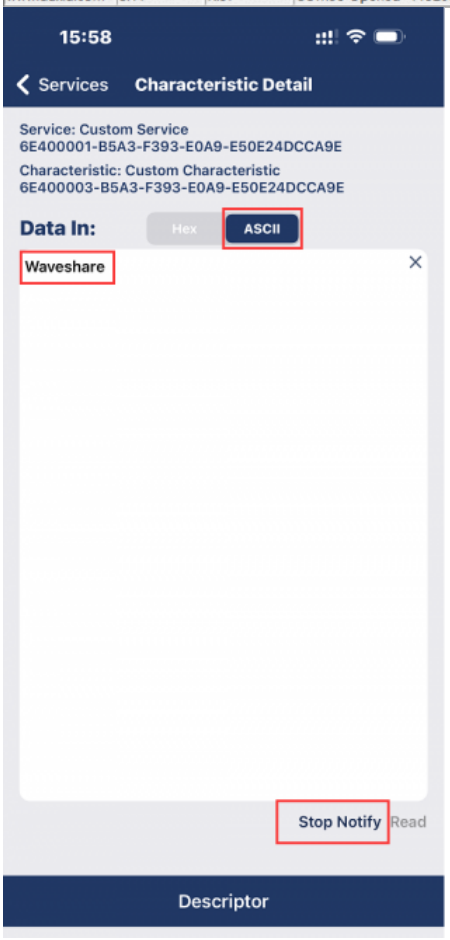


(/wiki/File:ESP32-S3-GEEK_Demo-126.jpg)

- ESP32-S3-GEEK uses USB to UART tool (https://www.waveshare.com/ch343-usb-uart-board.htm) to connect to the PC, open the serial debugging assistant, and send the UART message as a Bluetooth message to the mobile phone. Make sure to check "AddCrLf" options when sending the message. The sent message will be displayed on the LCD. On the mobile device, observe if it receives the Bluetooth message.

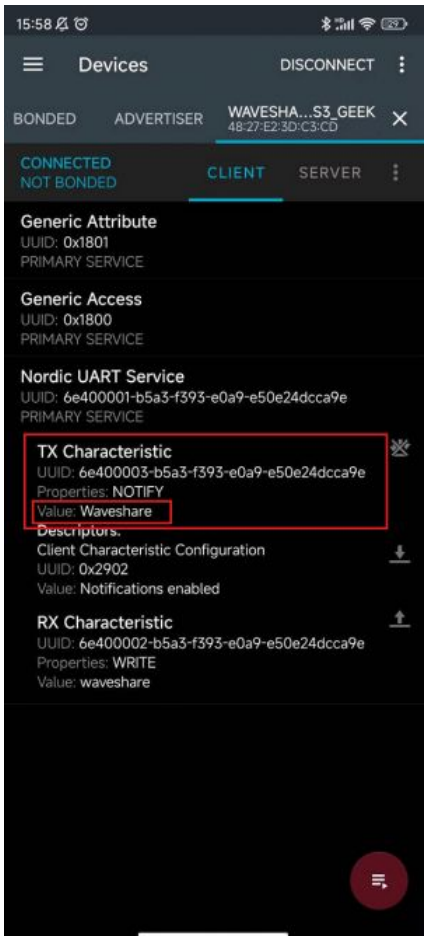


(/wiki/File:ESP32-S3-GEEK_Demo-29.png)



(/wiki/File:ESP32-S3-GEEK_Demo-27.jpg)

(/wiki/File:ESP32-S3-GEEK_Demo-28.png)



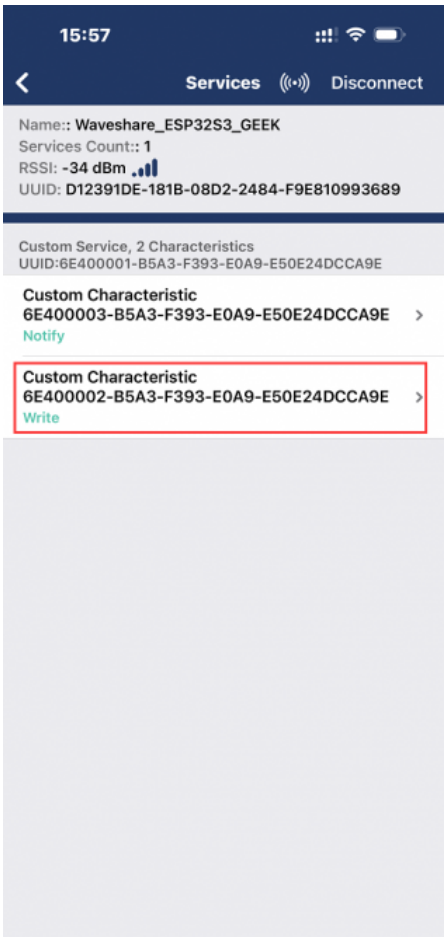
02-BLE_uart

```
This demo allows the ESP32-S3-GEEK to activate Bluetooth Low Energy (BLE). You can use a smartphone's Bluetooth debugging tool t
o connect to the ESP32-S3-GEEK, enabling BLE communication between the two devices. The sent and received messages will be displ
ayed on the serial monitor, similar to BLE_LCD. However, this version doesn't activate the LCD, reducing power consumption signi
ficantly by using UART to display message content.
```

- In the code BLEDevice::init("Waveshare_ESP32S3_GEEK"), "Waveshare_ESP32S3_GEEK" is the Bluetooth device name.
- With a mobile Bluetooth debugging tool, scan and connect to devices.

- Using the tool, send Bluetooth messages from the mobile device to the ESP32-S3-GEEK. The received messages will be displayed on the serial debugging assistant of the ESP32-S3-GEEK.

IOS:

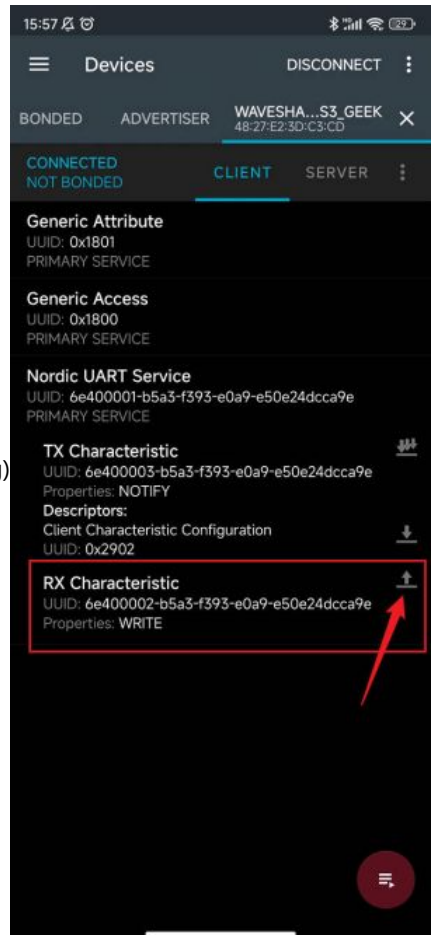


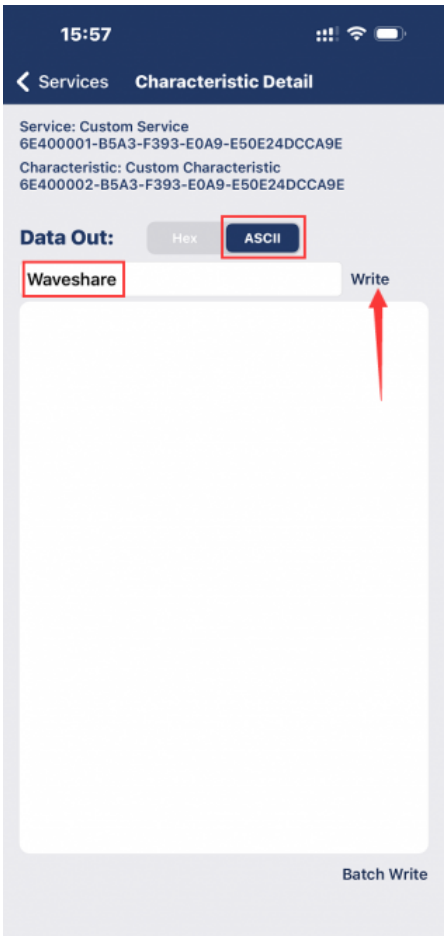
(/wiki/File:ESP32-S3-GEEK_Demo-24.jpg)

Android:

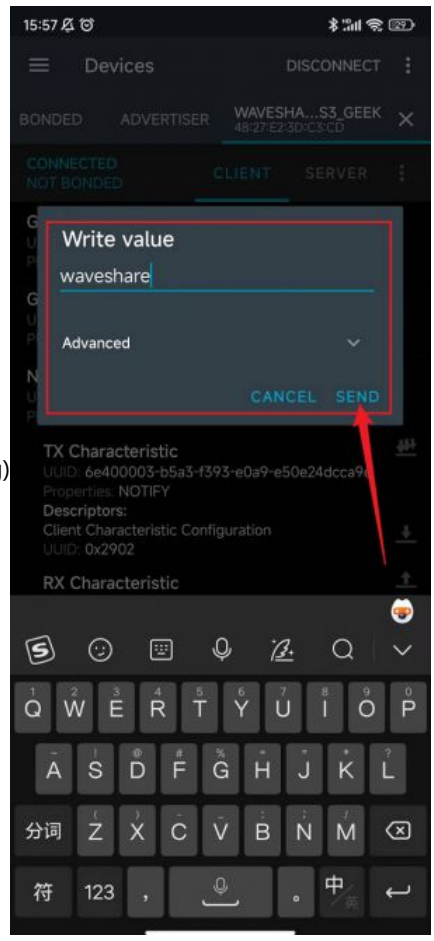
(/wiki/File:ESP32-S3-GEEK_Demo-21.png)

(/wiki/File:ESP32-S3-GEEK_Demo-23.png)

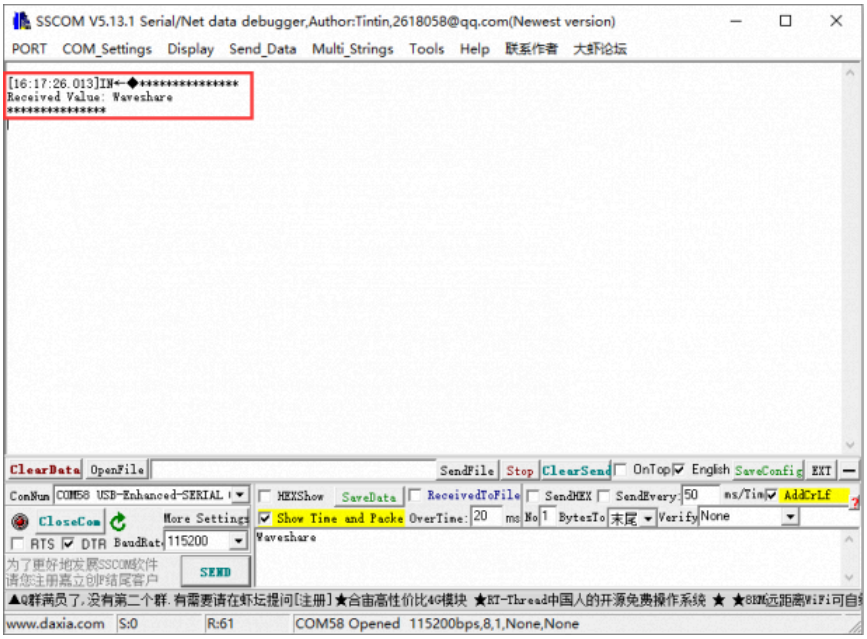




(/wiki/File:ESP32-S3-GEEK_Demo-132.png)



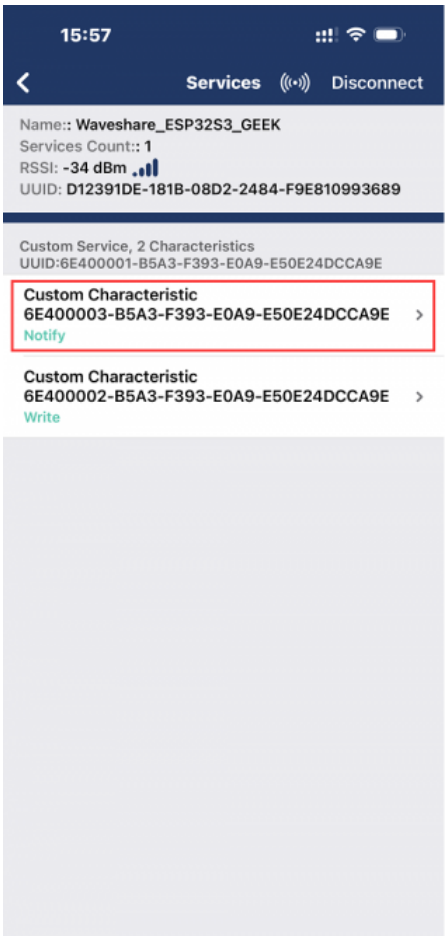
(/wiki/File:ESP32-S3-GEEK_Demo-133.jpg)



(/wiki/File:ESP32-S3-GEEK_Demo34.png)

- Enter the receiving setting of the Bluetooth debugging tool on the phone.

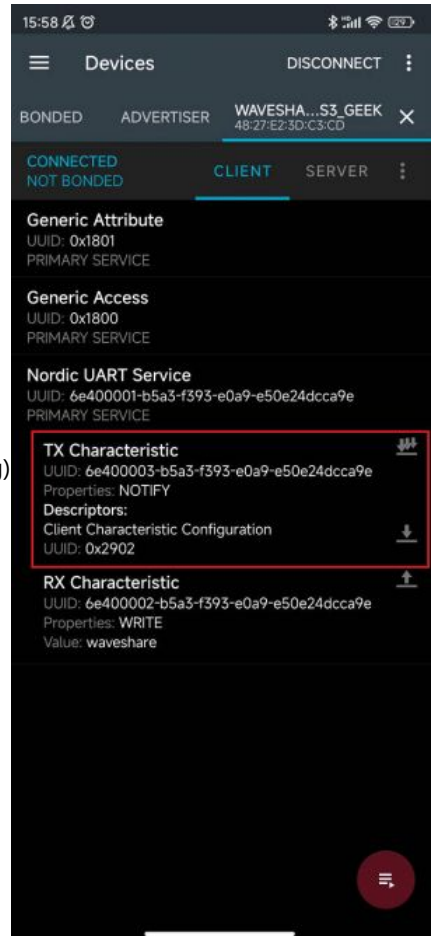
IOS:



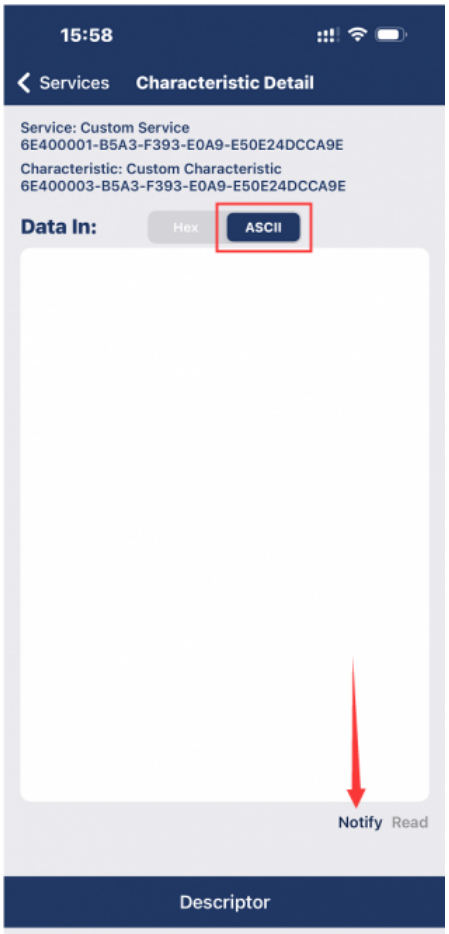
(/wiki/File:ESP32-S3-GEEK_Demo-26.jpg)

Android:

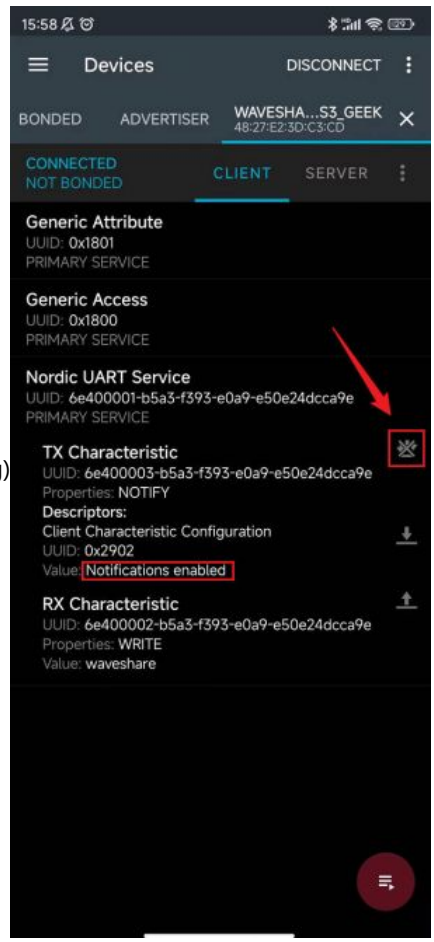
(/wiki/File:ESP32-S3-GEEK_Demo-21.png)



(/wiki/File:ESP32-S3-GEEK_Demo-25.png)

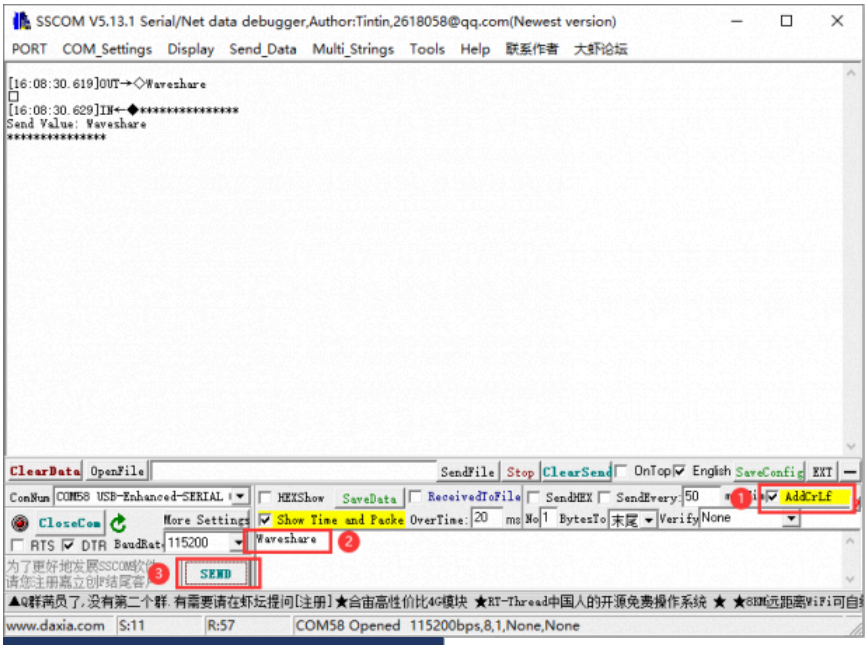


(/wiki/File:ESP32-S3-GEEK_Demo-125.png)

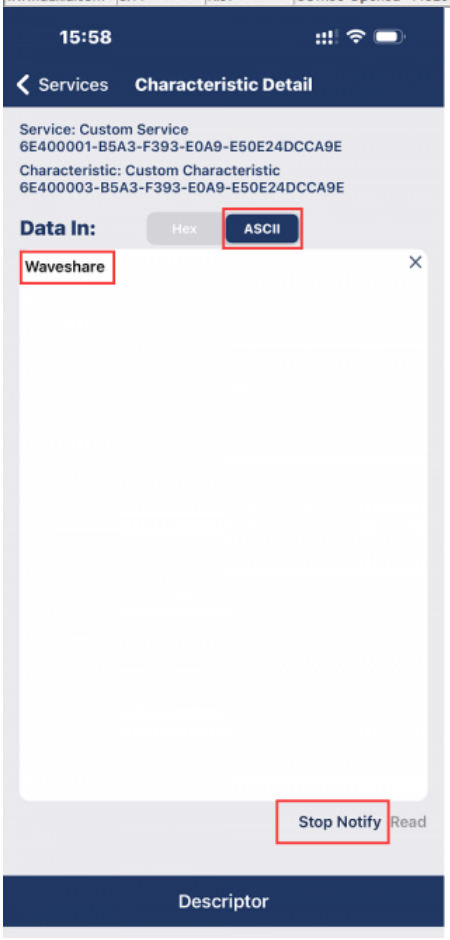


(/wiki/File:ESP32-S3-GEEK_Demo-126.jpg)

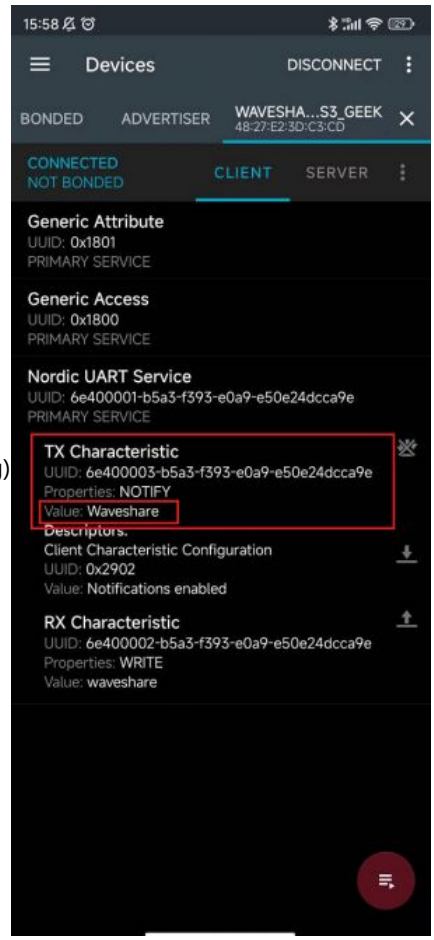
- ESP32-S3-GEEK, connected to the PC via CH343 USB UART Board (mini) (https://www.waveshare.com/ch343-usb-uart-board.htm), opens a serial debugging tool. Messages sent via serial are converted into Bluetooth messages to the phone. Ensure to check"AddCrLf". The message content will be displayed on the LCD. Check on the mobile device to observe if it receives the Bluetooth message.



(/wiki/File:ESP32-S3-GEEK_Demo-29.png)



(/wiki/File:ESP32-S3-GEEK_Demo-27.jpg)



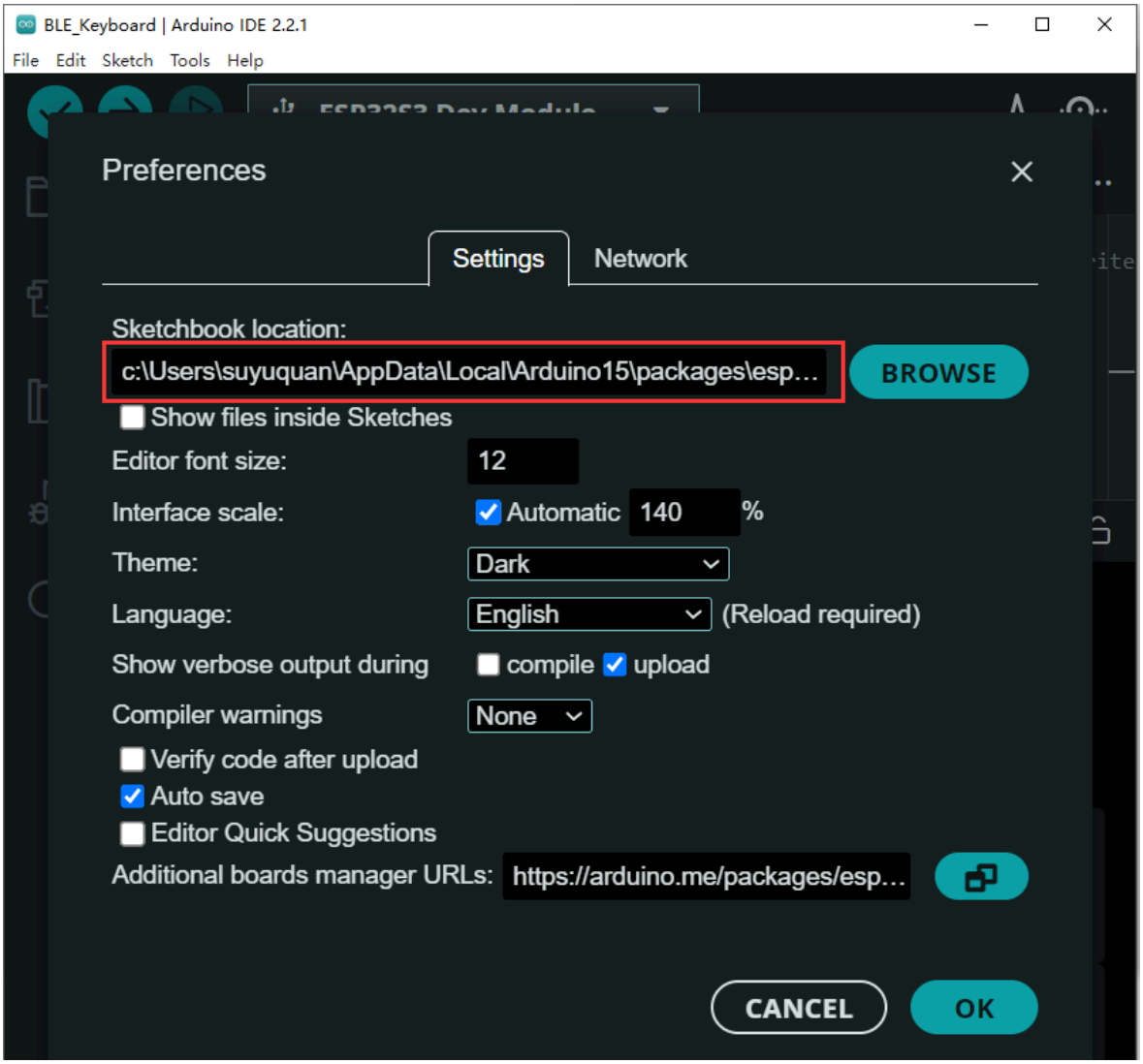
(/wiki/File:ESP32-S3-GEEK_Demo-28.png)

03-BLE_Keyboard

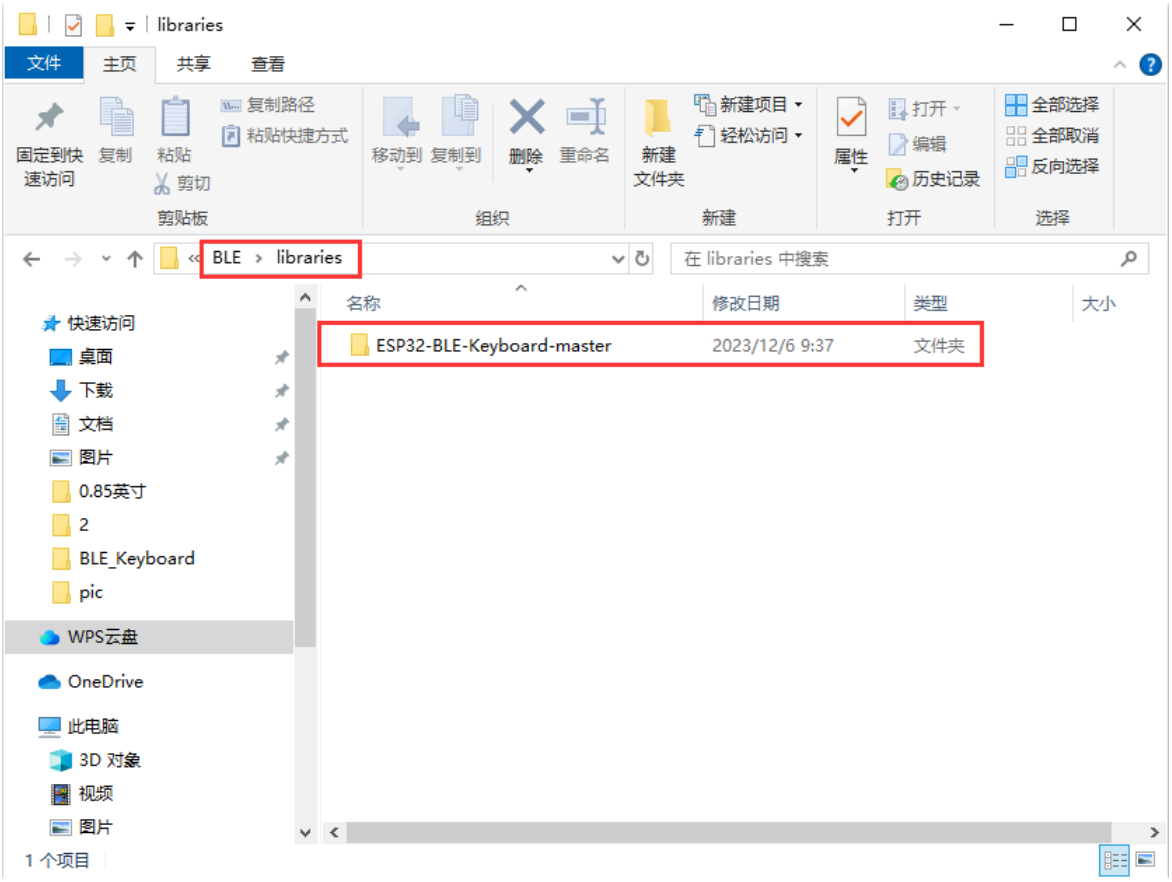
```
This code enables the ESP32-S3-GEEK to function as a Bluetooth keyboard. Once connected via the PC's Bluetooth, you can perform a series of single-key or combination-key operations. The ESP32-S3-GEEK will simulate these key presses and transmit correspondi ng keyboard inputs to the connected device.
```

- Before using BLE_Keyboard, you need to add the ESP32-BLE-Keyboard Arduino library folder from the libraries directory into the libraries folder within the installation directory of Arduino IDE.

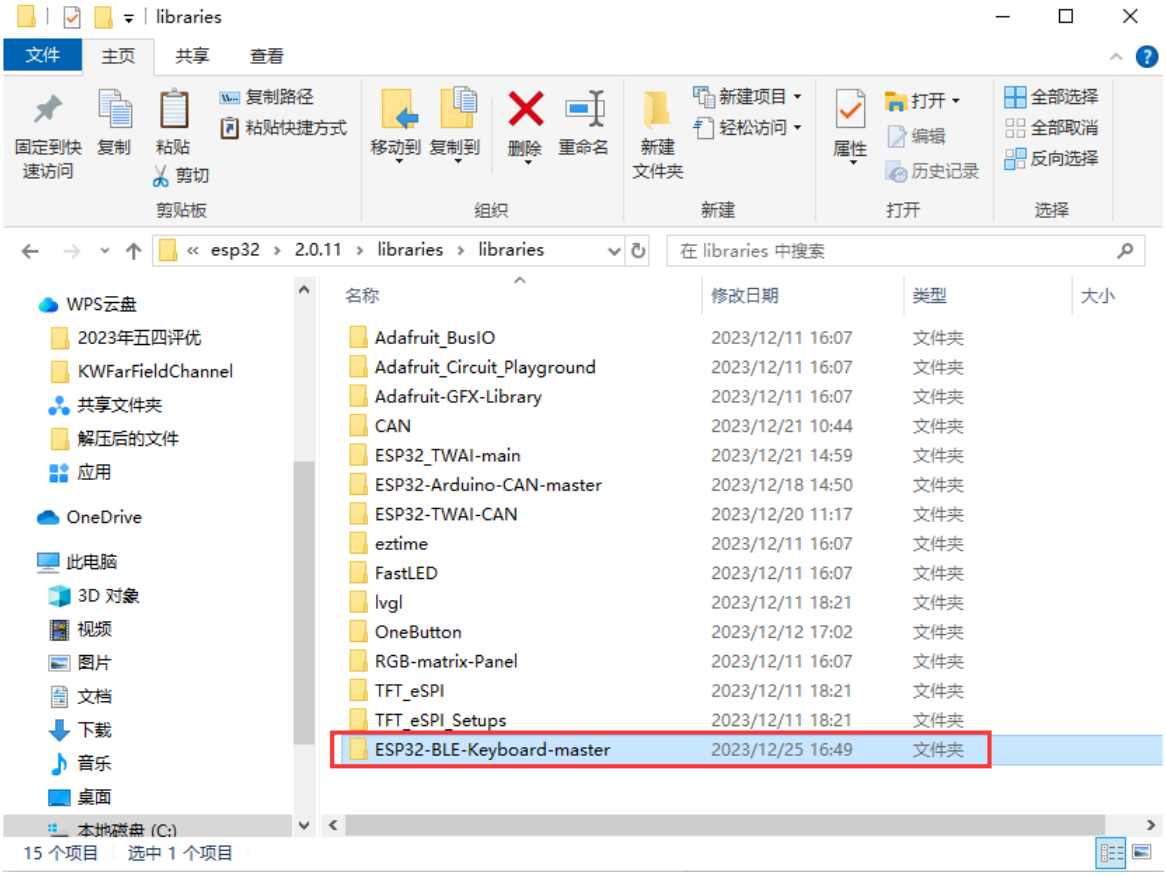
(/wiki/File:ESP32-S3-GEEK_Demo40.png)



(/wiki/File:ESP32-S3-GEEK_Demo41.png)



(/wiki/File:ESP32-S3-GEEK_Demo42.png)

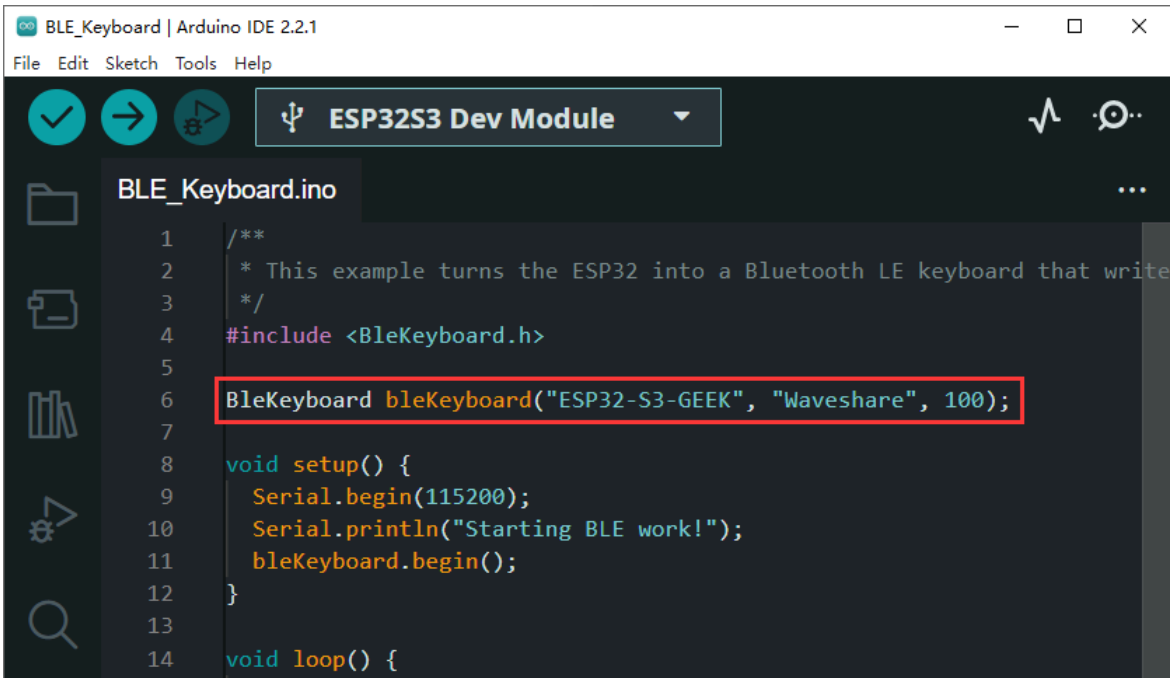


(/wiki/File:ESP32-S3-GEEK_Demo43.png)

- If the Arduino library of the ESP32-BLE-Keyboard has been installed, pay attention to modifying the BleKeyboard.c file as shown below:

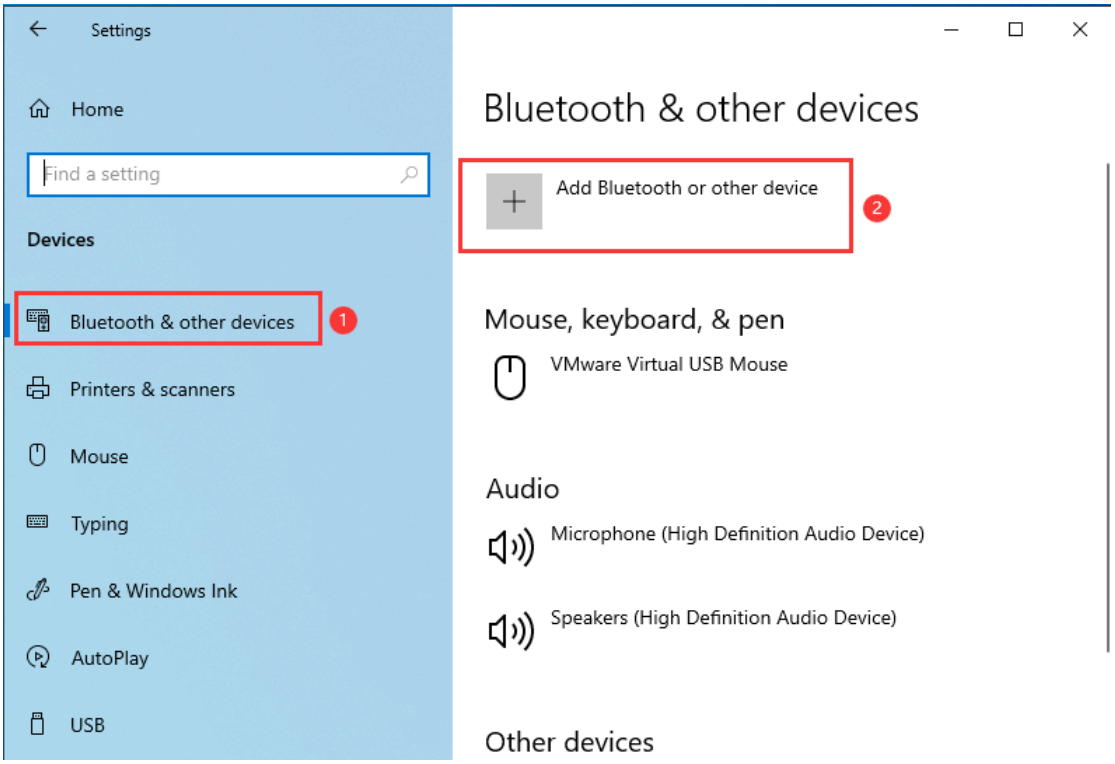
```
BLESecurity *pSecurity = new BLESecurity();
pSecurity->setAuthenticationMode(ESP_LE_AUTH_REQ_SC_MITM_BOND);

Modify it as:
BLESecurity *pSecurity = new BLESecurity();
pSecurity->setAuthenticationMode(ESP_LE_AUTH_BOND);
```

- ESP32-S3-GEEK is the name of the Bluetooth keyboard:

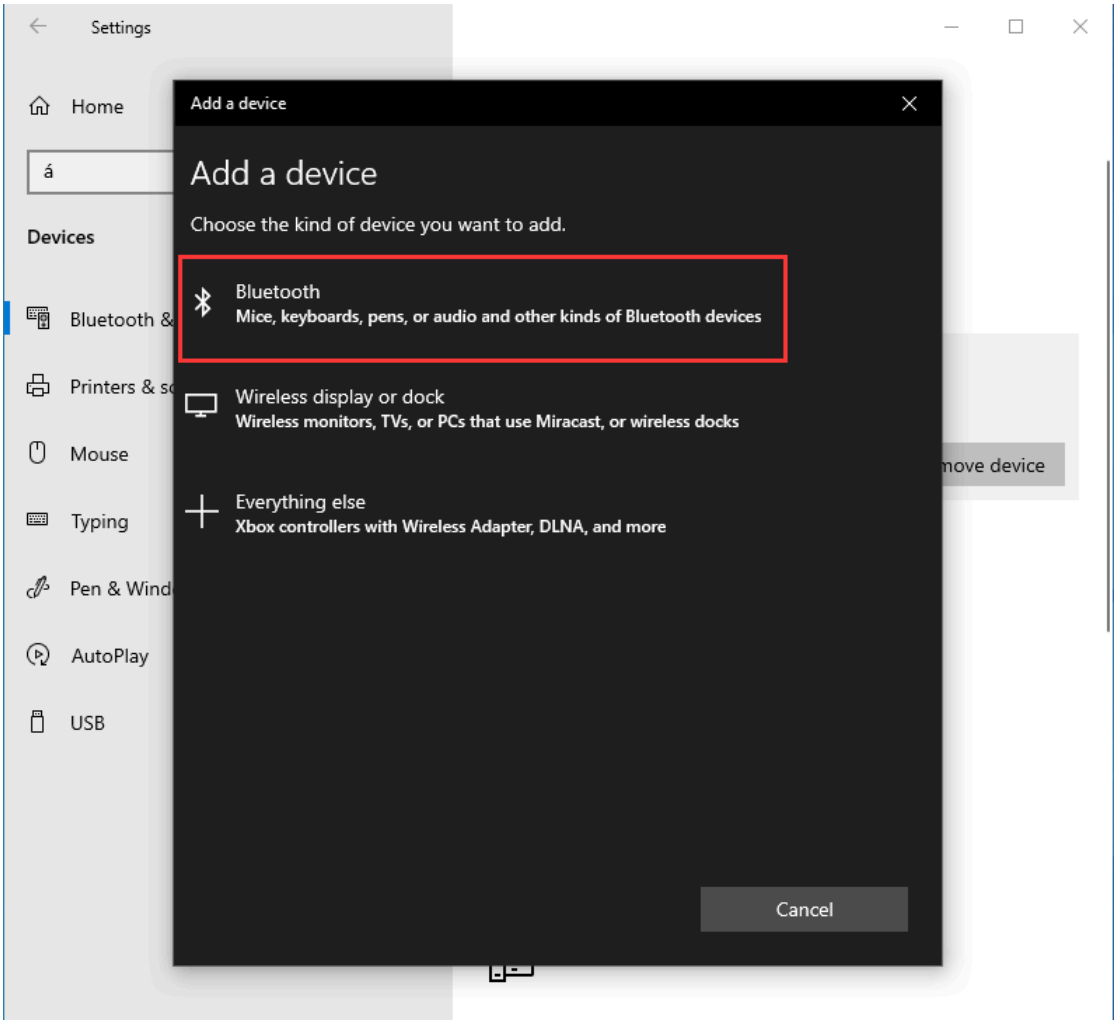


(/wiki/File:ESP32-S3-GEEK_Demo45.png)

- Use PC to enter Bluetooth to scan and connect the device:

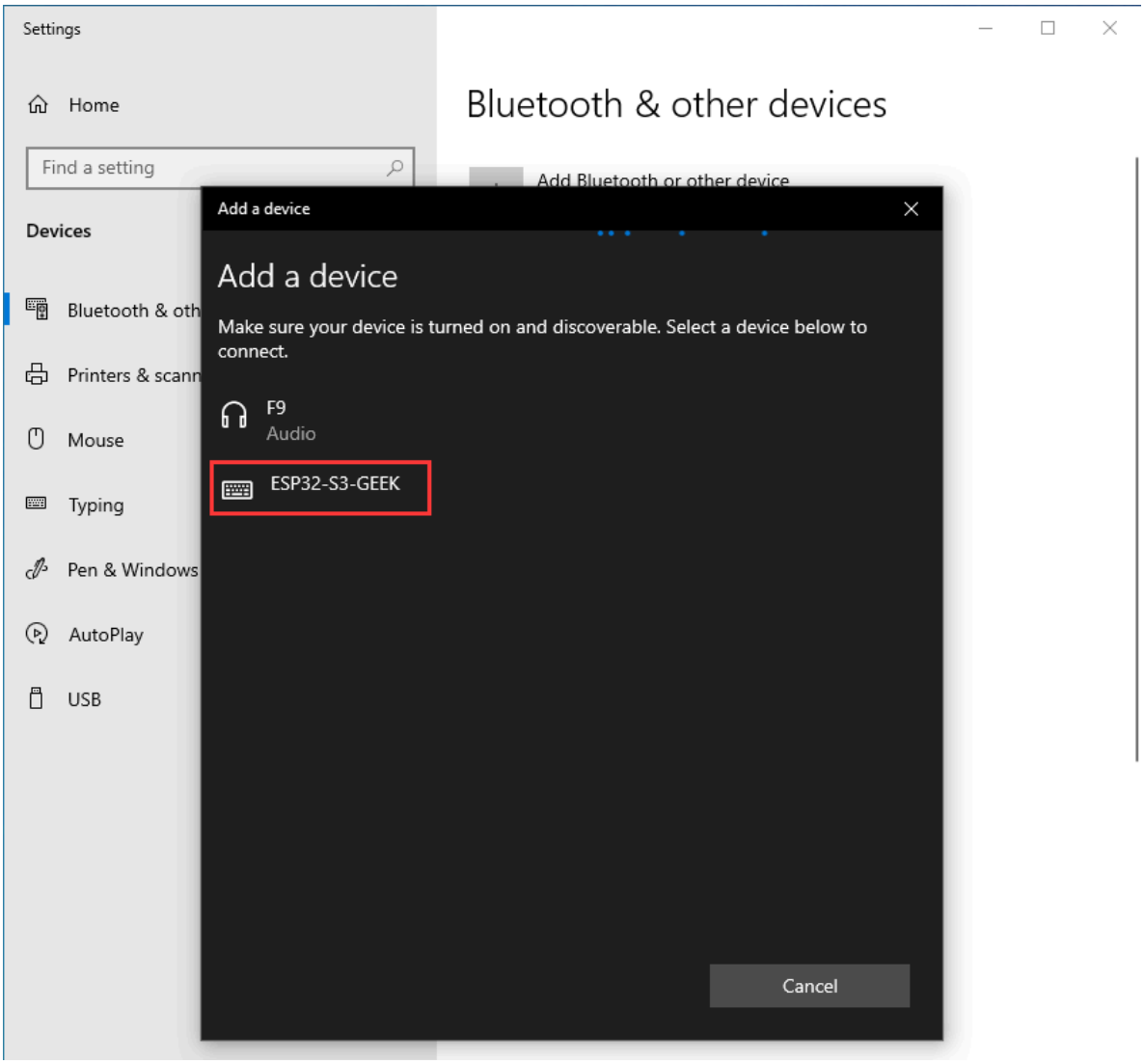


(/wiki/File:ESP32-S3-GEEK_Demo46.png)



(/wiki/File:ESP32-S3-

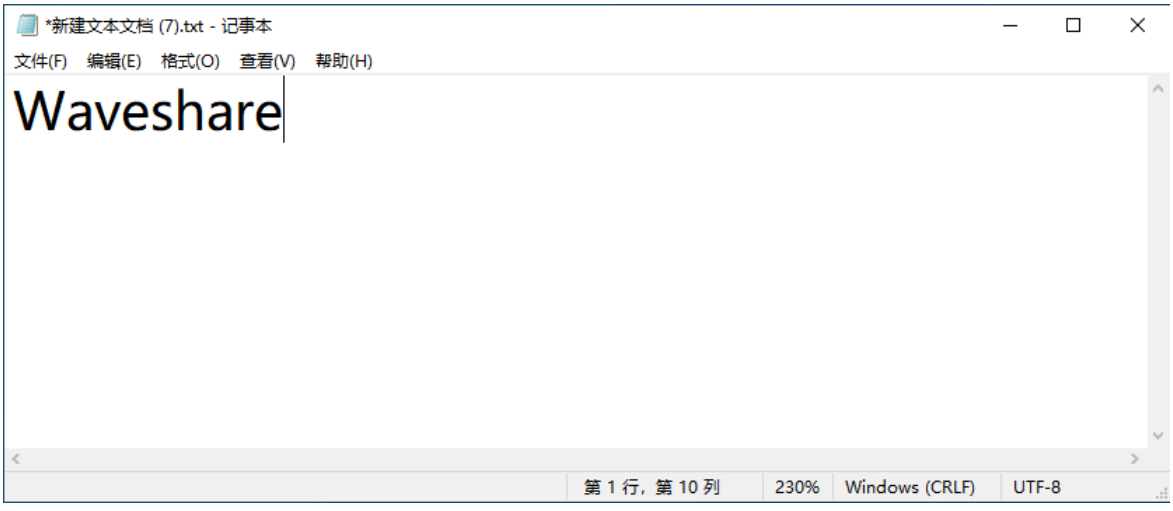
GEEK_Demo47.png)



(/wiki/File:ESP32-S3-

GEEK_Demo48.png)

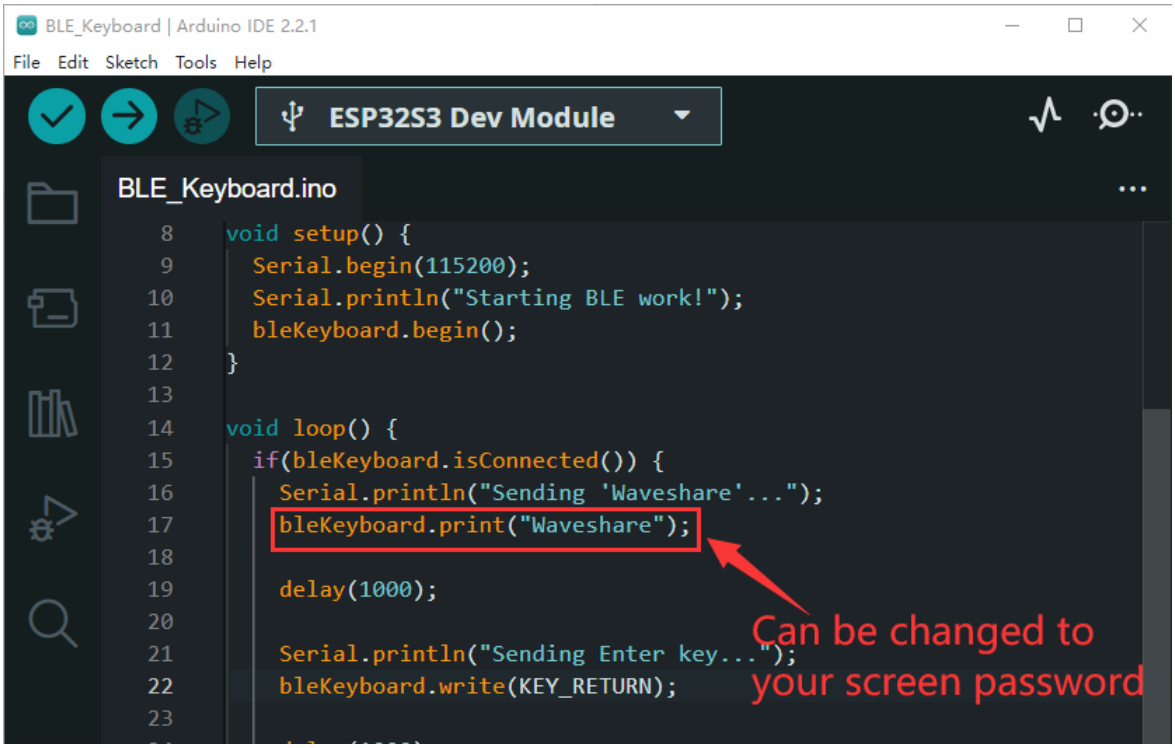
- After a successful connection, a series of keyboard actions are performed every 5 seconds (output "Waveshare", Ctrl+Alt+Delete).



(/wiki/File:ESP32-S3-

GEEK_Demo49.png)

- Modify the code with your unlock code to enable automatic locking and unlocking operations.



(/wiki/File:ESP32-S3-GEEK_Demo50.png)

- You can view the value of each key in the BleKeyboard.h file of the libraries file folder.

```
const uint8_t KEY_LEFT_CTRL = 0x80;
const uint8_t KEY_LEFT_SHIFT = 0x81;
const uint8_t KEY_LEFT_ALT = 0x82;
const uint8_t KEY_LEFT_GUI = 0x83;
const uint8_t KEY_RIGHT_CTRL = 0x84;
const uint8_t KEY_RIGHT_SHIFT = 0x85;
const uint8_t KEY_RIGHT_ALT = 0x86;
const uint8_t KEY_RIGHT_GUI = 0x87;

const uint8_t KEY_UP_ARROW = 0xDA;
const uint8_t KEY_DOWN_ARROW = 0xD9;
const uint8_t KEY_LEFT_ARROW = 0xD8;
const uint8_t KEY_RIGHT_ARROW = 0xD7;
const uint8_t KEY_BACKSPACE = 0xB2;
const uint8_t KEY_TAB = 0xB3;
const uint8_t KEY_RETURN = 0xB0;
const uint8_t KEY_ESC = 0xB1;
const uint8_t KEY_INSERT = 0xD1;
const uint8_t KEY_PRTSC = 0xCE;
const uint8_t KEY_DELETE = 0xD4;
const uint8_t KEY_PAGE_UP = 0xD3;
const uint8_t KEY_PAGE_DOWN = 0xD6;
const uint8_t KEY_HOME = 0xD2;
const uint8_t KEY_END = 0xD5;
const uint8_t KEY_CAPS_LOCK = 0xC1;
const uint8_t KEY_F1 = 0xC2;
const uint8_t KEY_F2 = 0xC3;
const uint8_t KEY_F3 = 0xC4;
const uint8_t KEY_F4 = 0xC5;
const uint8_t KEY_F5 = 0xC6;
const uint8_t KEY_F6 = 0xC7;
const uint8_t KEY_F7 = 0xC8;
const uint8_t KEY_F8 = 0xC9;
const uint8_t KEY_F9 = 0xCA;
const uint8_t KEY_F10 = 0xCB;
const uint8_t KEY_F11 = 0xCC;
const uint8_t KEY_F12 = 0xCD;
const uint8_t KEY_F13 = 0xF0;
const uint8_t KEY_F14 = 0xF1;
const uint8_t KEY_F15 = 0xF2;
const uint8_t KEY_F16 = 0xF3;
const uint8_t KEY_F17 = 0xF4;
const uint8_t KEY_F18 = 0xF5;
const uint8_t KEY_F19 = 0xF6;
const uint8_t KEY_F20 = 0xF7;
const uint8_t KEY_F21 = 0xF8;
const uint8_t KEY_F22 = 0xF9;
const uint8_t KEY_F23 = 0xFA;
const uint8_t KEY_F24 = 0xFB;

const uint8_t KEY_NUM_0 = 0xEA;
const uint8_t KEY_NUM_1 = 0xE1;
const uint8_t KEY_NUM_2 = 0xE2;
const uint8_t KEY_NUM_3 = 0xE3;
const uint8_t KEY_NUM_4 = 0xE4;
const uint8_t KEY_NUM_5 = 0xE5;
const uint8_t KEY_NUM_6 = 0xE6;
const uint8_t KEY_NUM_7 = 0xE7;
const uint8_t KEY_NUM_8 = 0xE8;
const uint8_t KEY_NUM_9 = 0xE9;
const uint8_t KEY_NUM_SLASH = 0xDC;
const uint8_t KEY_NUM_ASTERISK = 0xDD;
const uint8_t KEY_NUM_MINUS = 0xDE;
const uint8_t KEY_NUM_PLUS = 0xDF;
const uint8_t KEY_NUM_ENTER = 0xE0;
const uint8_t KEY_NUM_PERIOD = 0xEB;
```

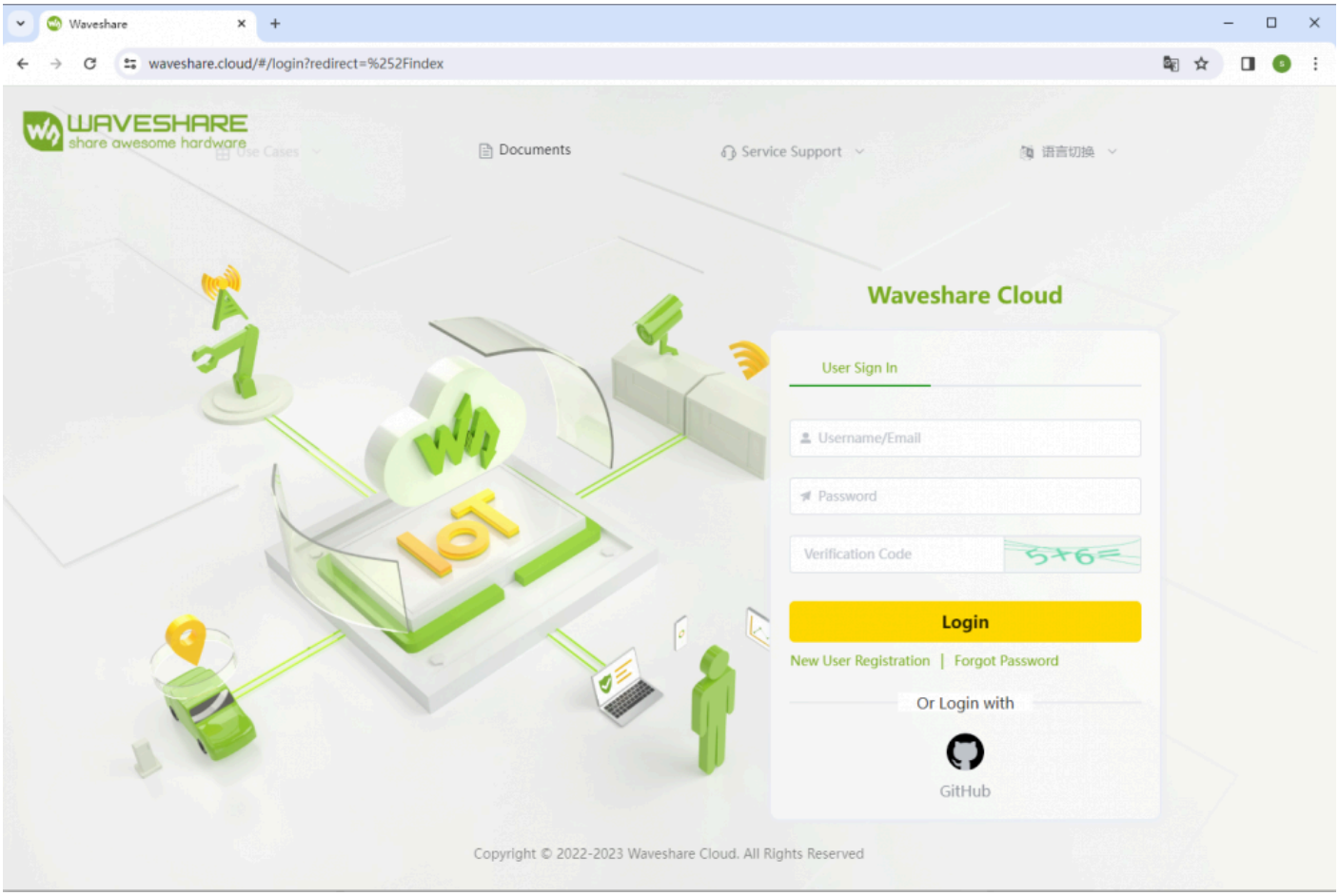
```
typedef uint8_t MediaKeyReport[2];

const MediaKeyReport KEY_MEDIA_NEXT_TRACK = {1, 0};
const MediaKeyReport KEY_MEDIA_PREVIOUS_TRACK = {2, 0};
const MediaKeyReport KEY_MEDIA_STOP = {4, 0};
const MediaKeyReport KEY_MEDIA_PLAY_PAUSE = {8, 0};
const MediaKeyReport KEY_MEDIA_MUTE = {16, 0};
const MediaKeyReport KEY_MEDIA_VOLUME_UP = {32, 0};
const MediaKeyReport KEY_MEDIA_VOLUME_DOWN = {64, 0};
const MediaKeyReport KEY_MEDIA_WWW_HOME = {128, 0};
const MediaKeyReport KEY_MEDIA_LOCAL_MACHINE_BROWSER = {0, 1}; // Opens "My Computer" on Windows
const MediaKeyReport KEY_MEDIA_CALCULATOR = {0, 2};
const MediaKeyReport KEY_MEDIA_WWW_BOOKMARKS = {0, 4};
const MediaKeyReport KEY_MEDIA_WWW_SEARCH = {0, 8};
const MediaKeyReport KEY_MEDIA_WWW_STOP = {0, 16};
const MediaKeyReport KEY_MEDIA_WWW_BACK = {0, 32};
const MediaKeyReport KEY_MEDIA_CONSUMER_CONTROL_CONFIGURATION = {0, 64}; // Media Selection
const MediaKeyReport KEY_MEDIA_EMAIL_READER = {0, 128};
```

MQTT

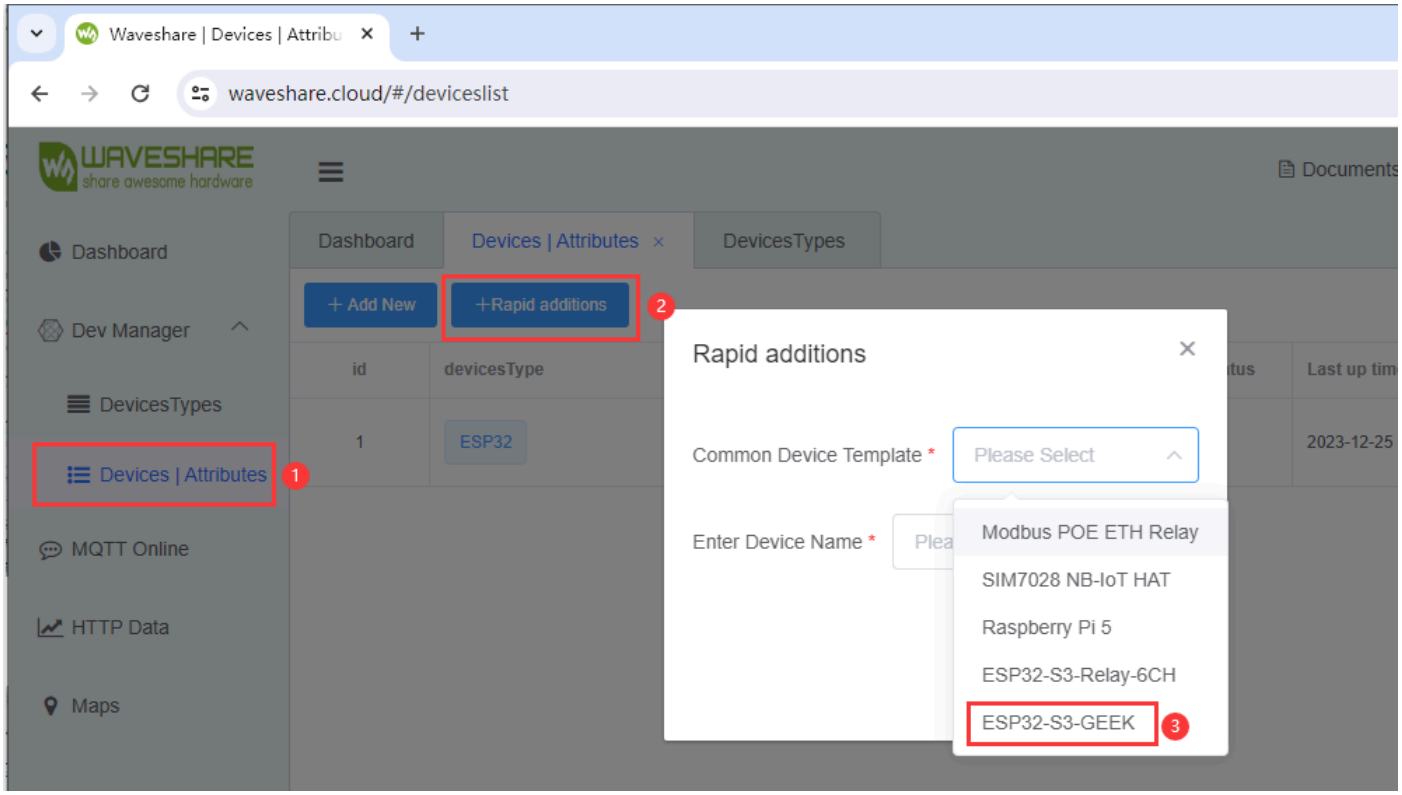
01-MQTT_sub_pub

```
The demo can use ESP32-S3-GEEK to open the STA mode of WIFI, connect to the WIFI, and then use the Waveshare cloud platform to c
arry out MQTT communication, subscribe and publish topics, and realize the long-distance transmission of information.
```

- When you are the first time to register on the Waveshare cloud (https://waveshare.cloud/#/login?redirect=%252Findex), you need to create a new device according to the tutorial (https://mqtt.waveshare.cloud/en/wavesharecloud-wiki.html).



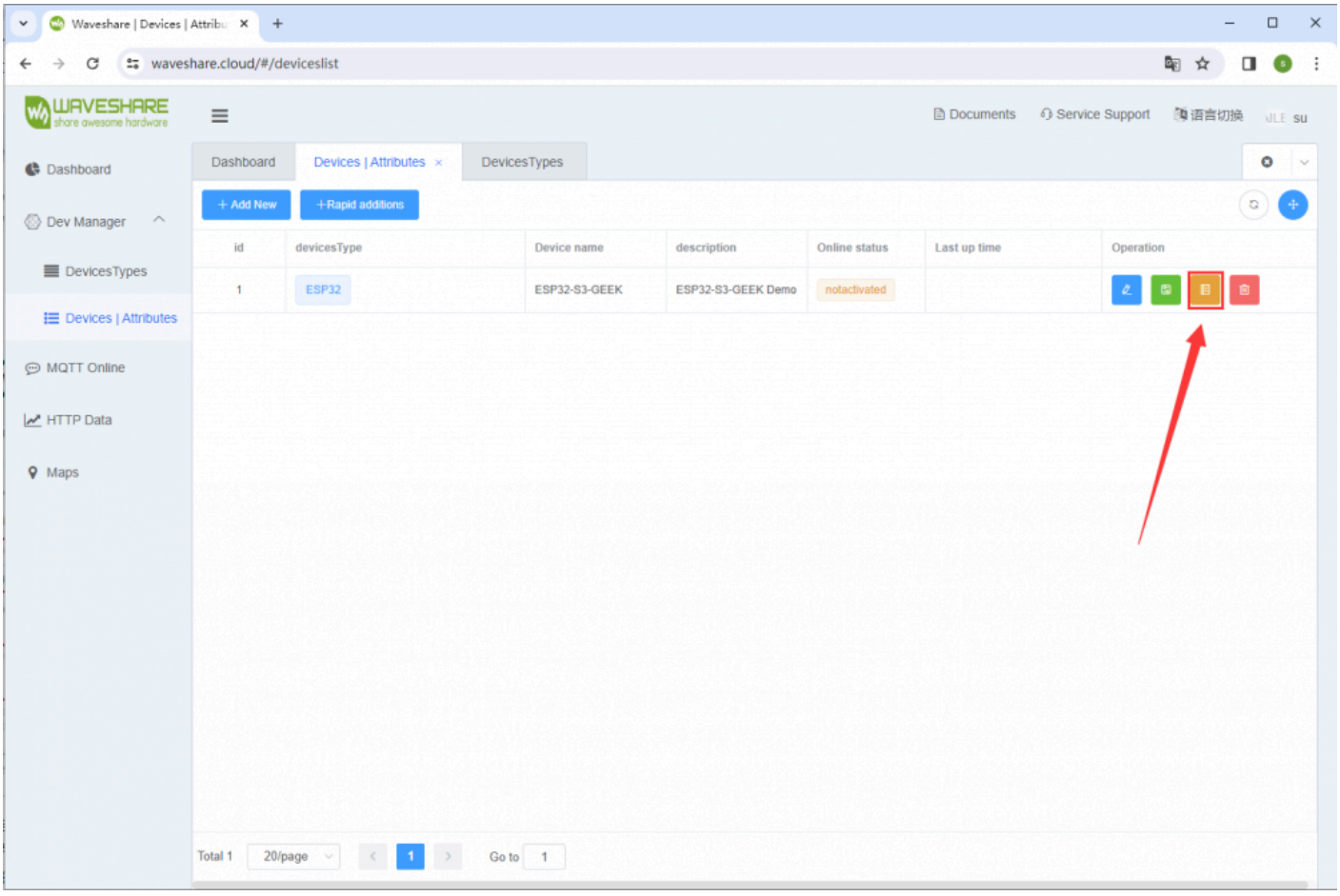
(/wiki/File:ESP32-S3-GEEK_SUB.png)



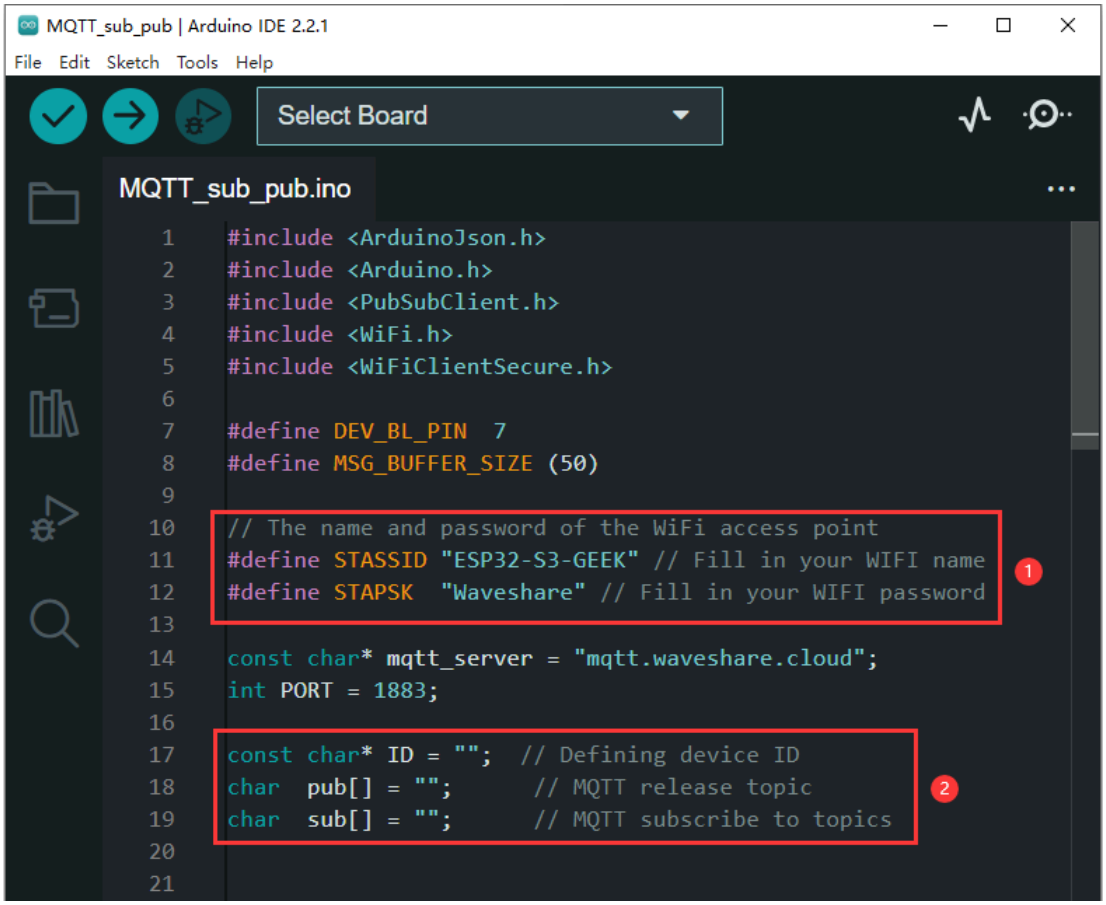
(/wiki/File:ESP32-S3-GEEK_SUB01.png)

- After creating, you can set the corresponding configuration in the Arduino IDE according to ESP32SDK tutorial (https://mqtt.waveshare.cloud/en/esp32-connect.html).

- After configuration, you can find the Client ID, Pub Topic, and Sub Topic of the newly created device on Waveshare's cloud platform under "View Address". You can assign these values in the code to facilitate ESP32-S3-GEEK's connection to your cloud platform device.
- Ensure that ESP32-S3-GEEK is set to STA mode and connected to the same Wi-Fi network as your device. (The Wi-Fi network ESP32-S3-GEEK connects to needs to support the 2.4GHz frequency band. In case it doesn't, you can use a PC to create a hotspot with the network frequency set to "Any available frequency".) Remember to modify the ssid and password with the name and password of the Wi-Fi network you want to connect to.

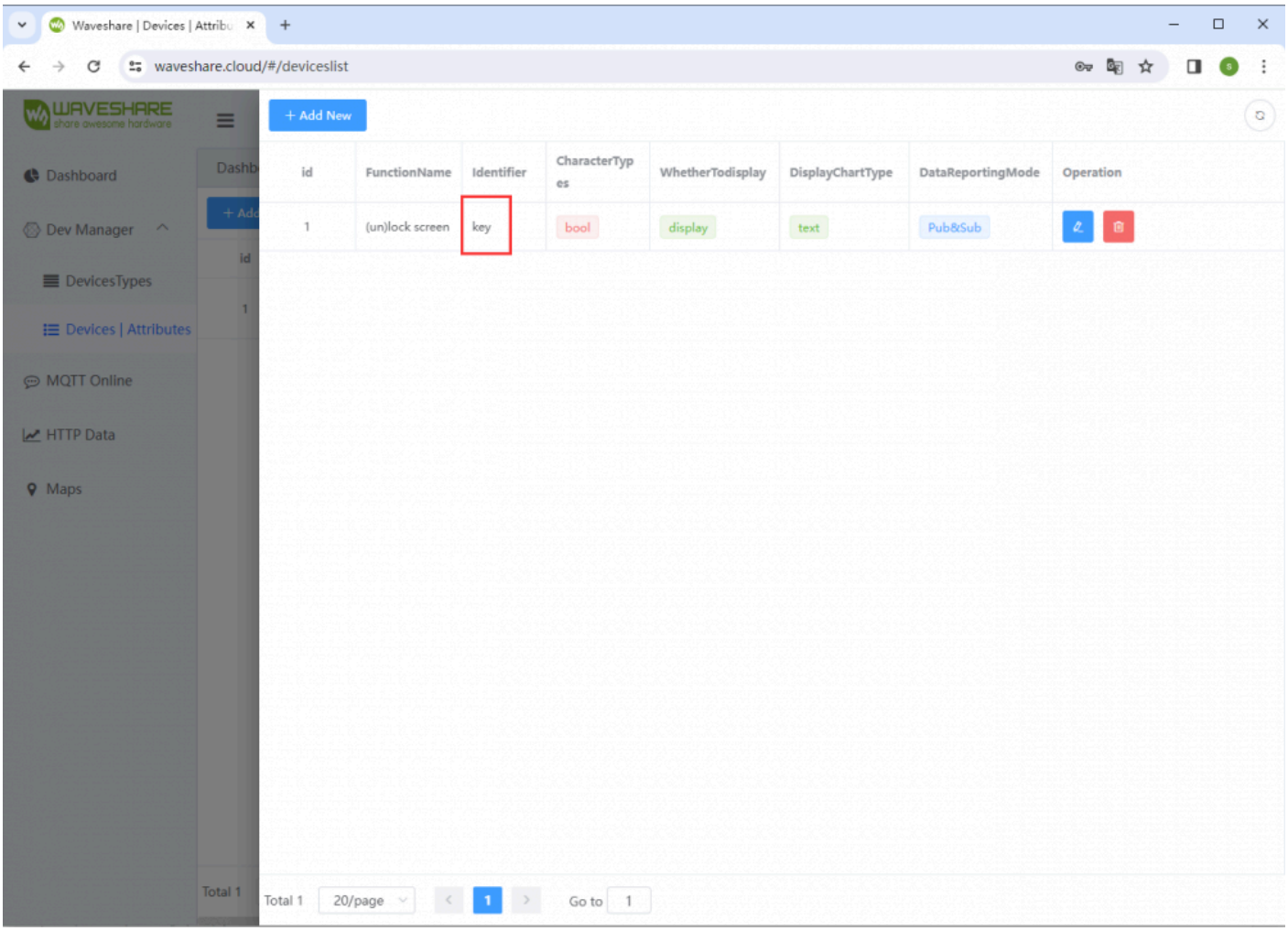


(/wiki/File:ESP32-S3-GEEK_SUB02.png)

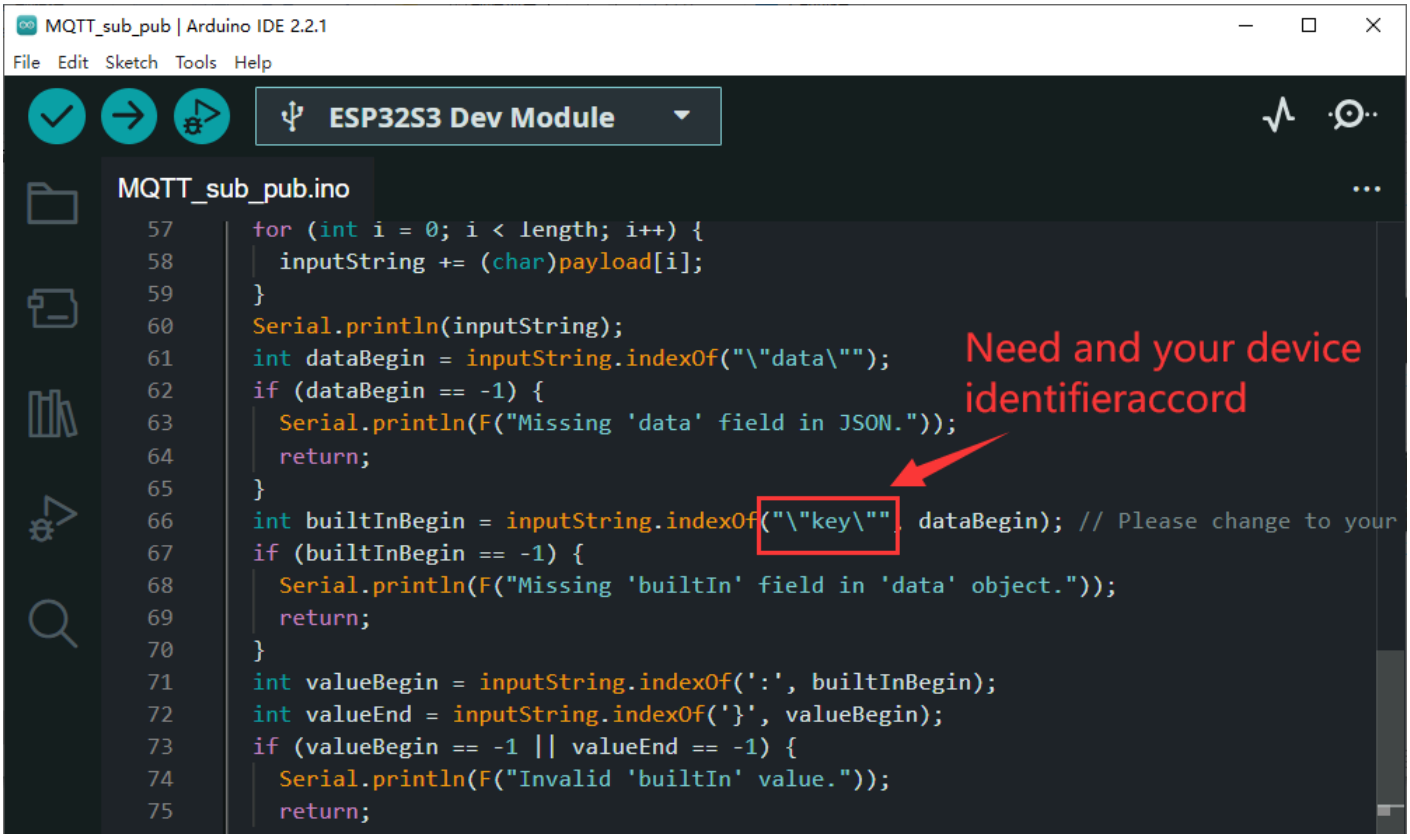


(/wiki/File:ESP32-S3-GEEK_SUB03.png)

- In the callback function, you can modify the identification tag to our device attribute identifier (https://mqtt.waveshare.cloud/en/waveshare cloud-wiki.html#3d7f7136_97) created on the cloud platform.

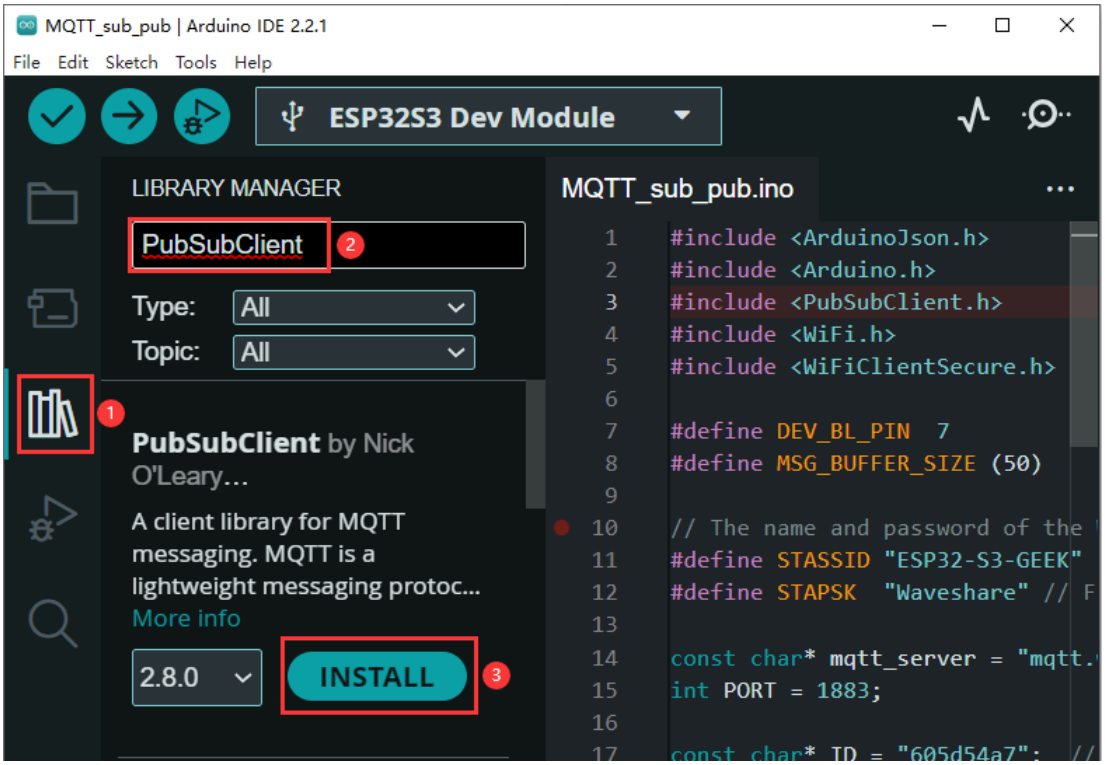


(/wiki/File:ESP32-S3-GEEK_SUB04.png)



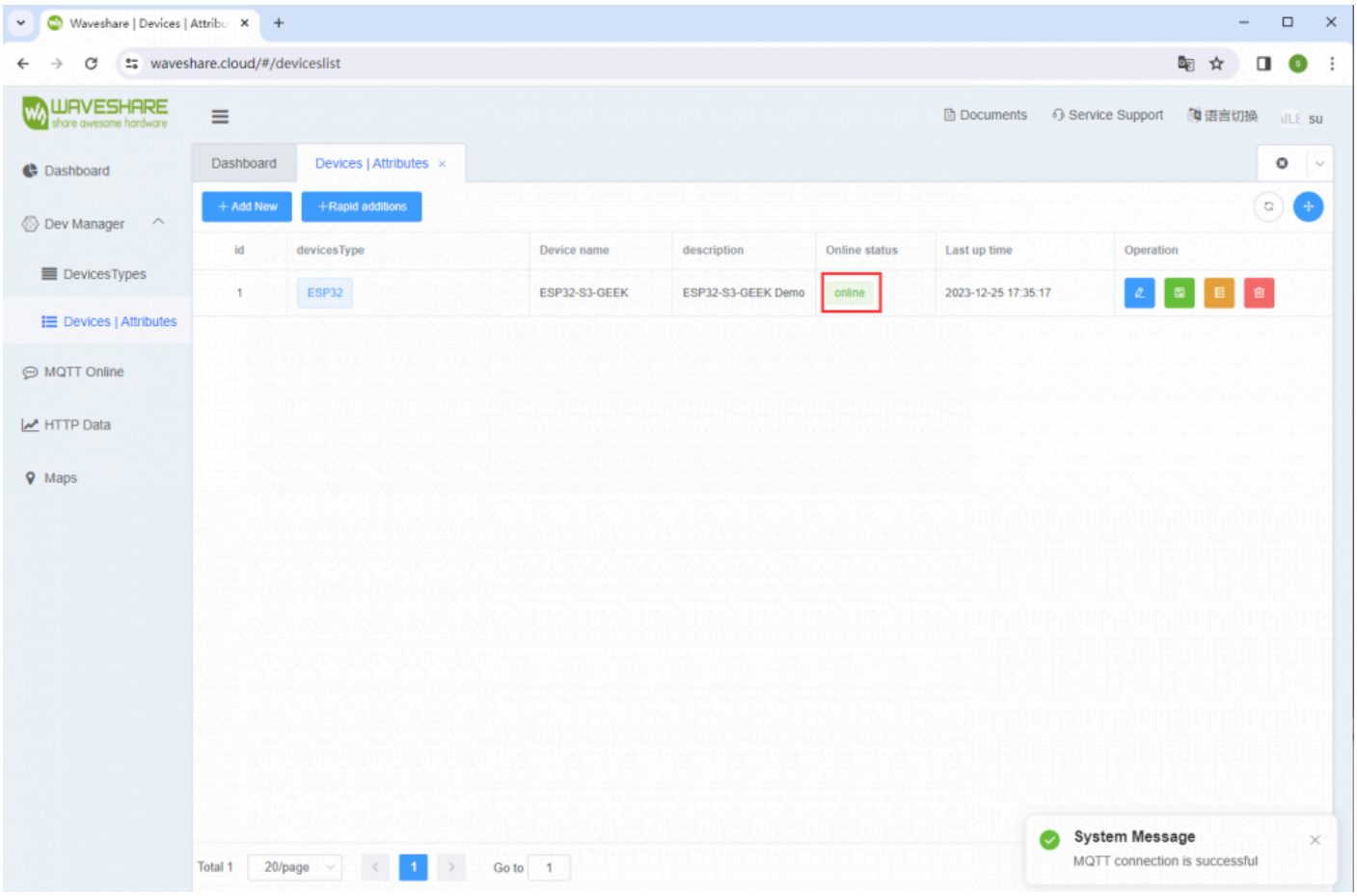
(/wiki/File:ESP32-S3-GEEK_SUB05.png)

- Install PubSubClient library, LIBRARY MANAGER -> Search PubSubClient -> INSTALL.

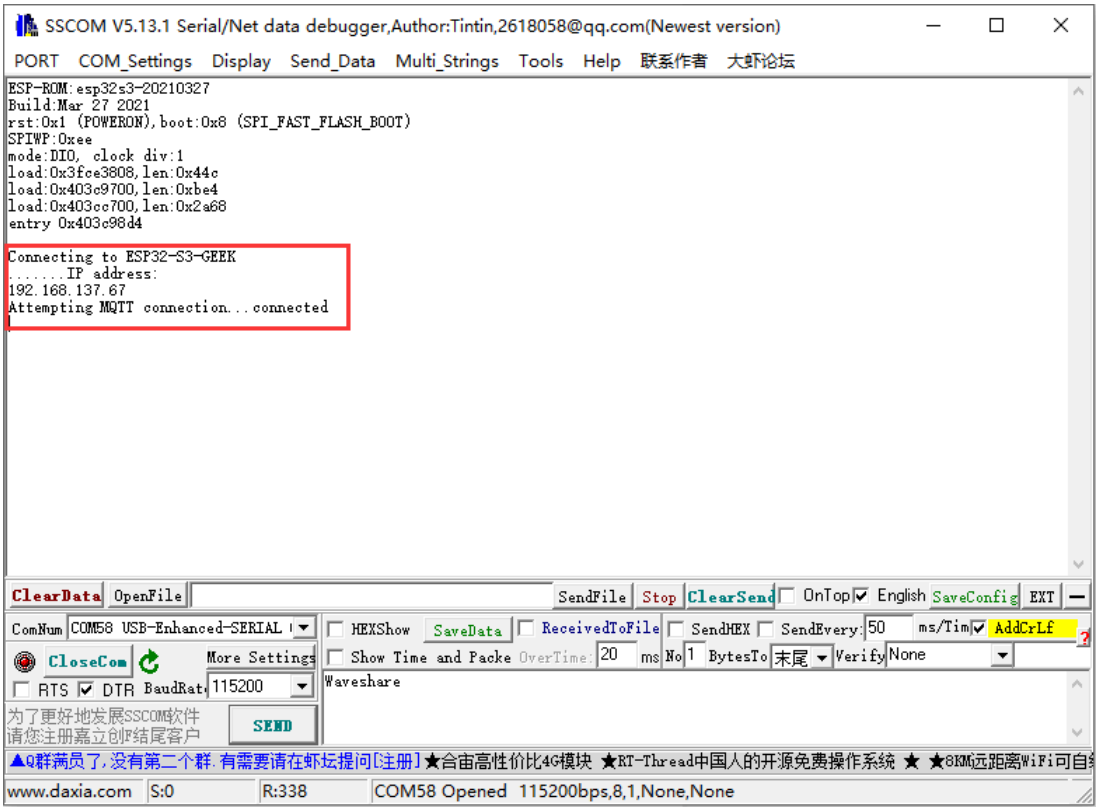


(/wiki/File:ESP32-S3-GEEK_SUB06.png)

- Program the code, and after connecting to the WIFI, you can observe whether the device on Waveshare cloud platform enters "online" status. If not, you can try to refresh the webpage, use CH343 USB UART Board (mini) (https://www.waveshare.com/ch343-usb-uart-board.htm) to connect to the PC, through the serial debugging assistant check the connection between the WIFI and MQTT. The LCD screen will display the connection situation between the WIFI and MQTT.

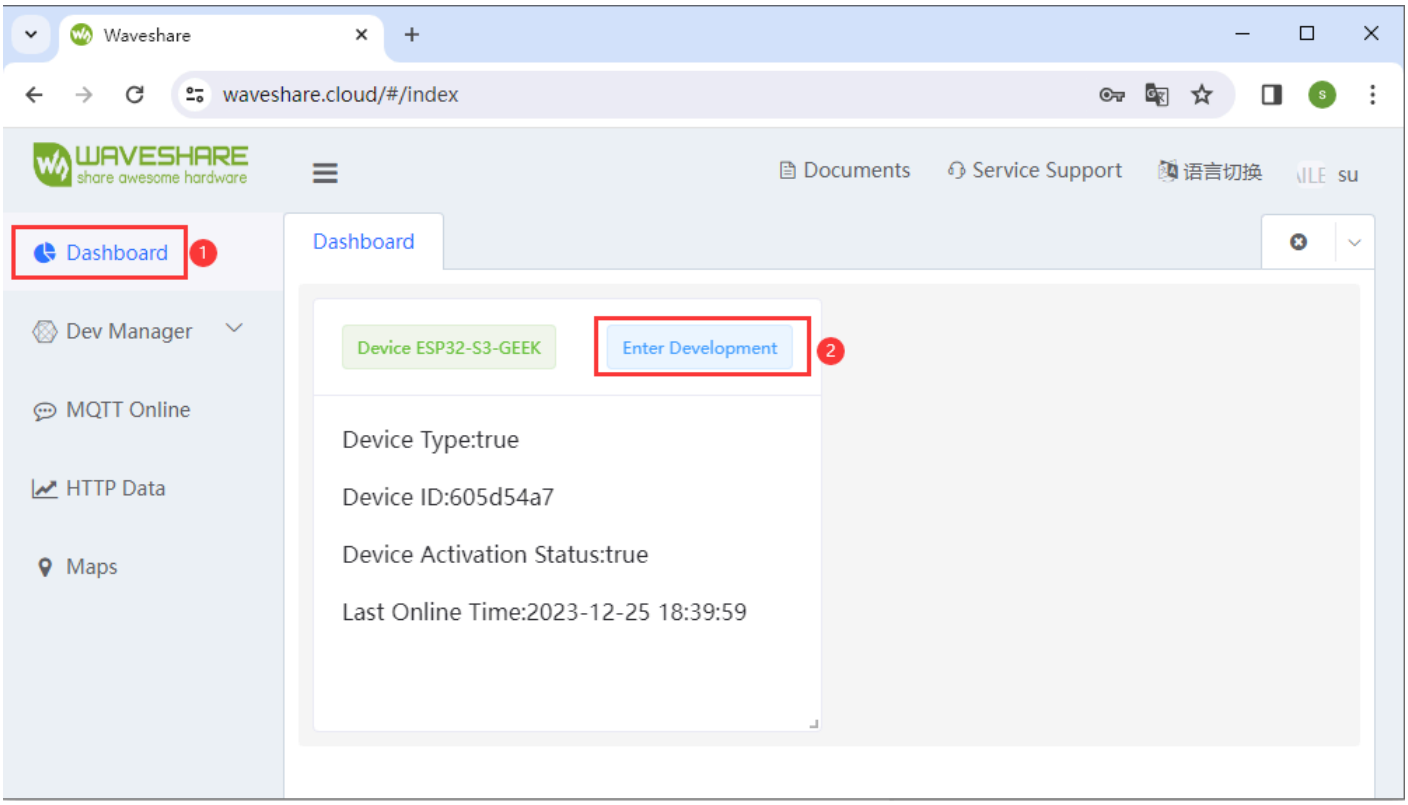


(/wiki/File:ESP32-S3-GEEK_SUB07.png)

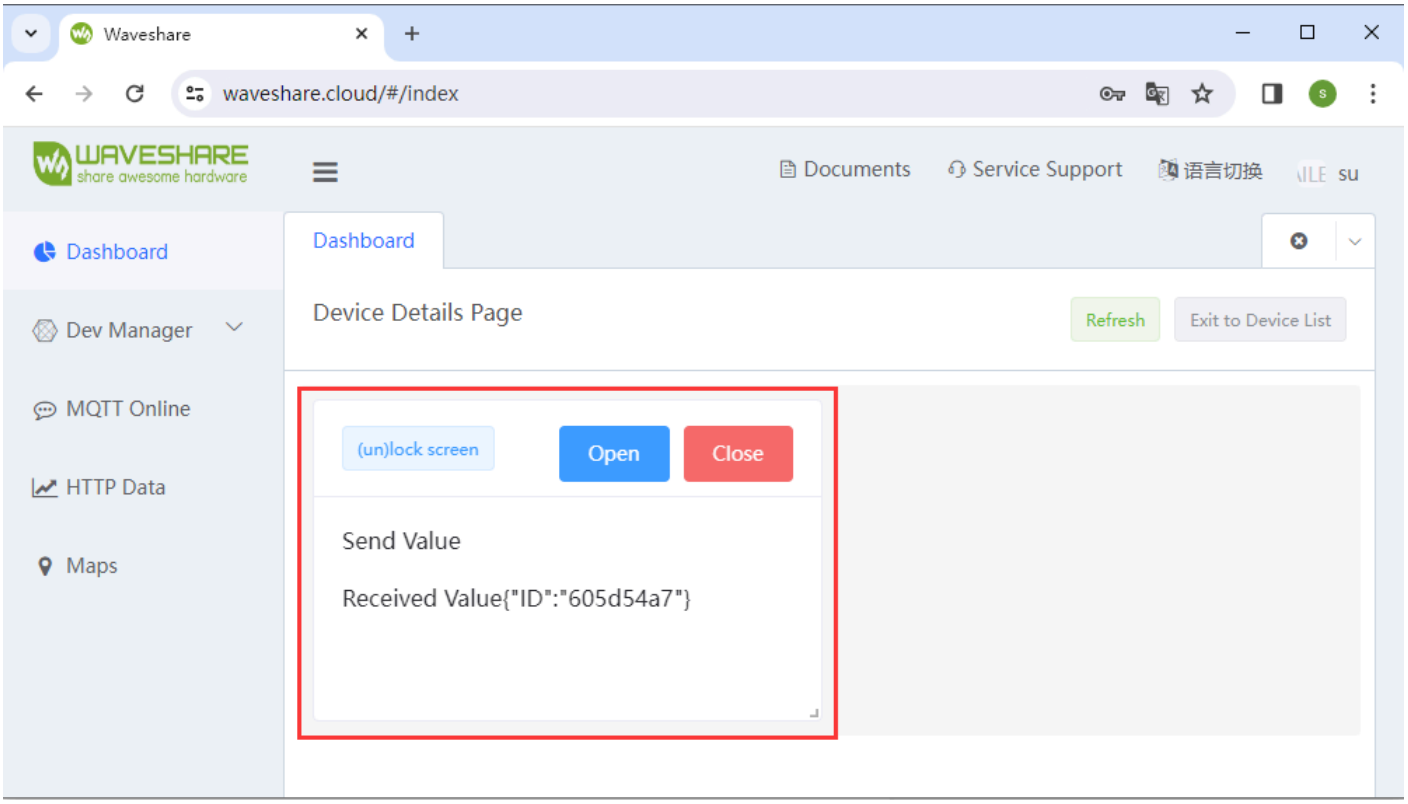


(/wiki/File:ESP32-S3-GEEK_SUB08.png)

- After connecting ESP32-S3-GEEK to Waveshare Cloud, you can directly send the MQTT message through the Dashboard.

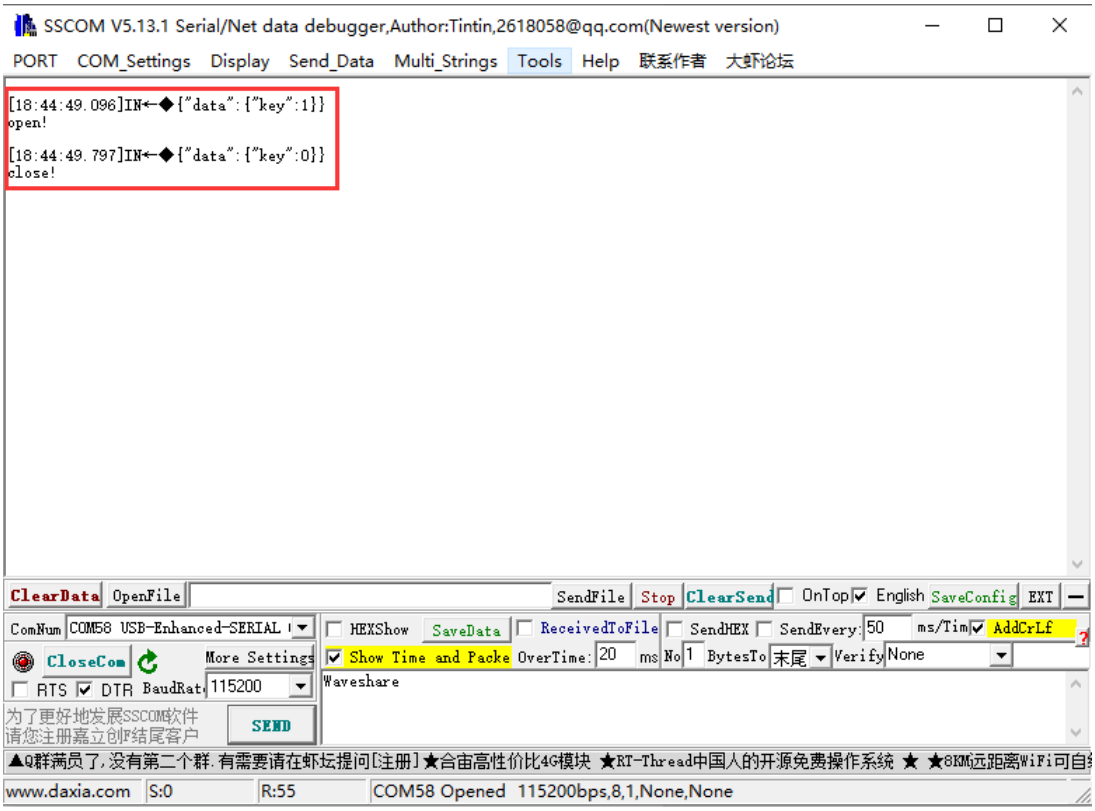


(/wiki/File:ESP32-S3-GEEK_SUB09.png)

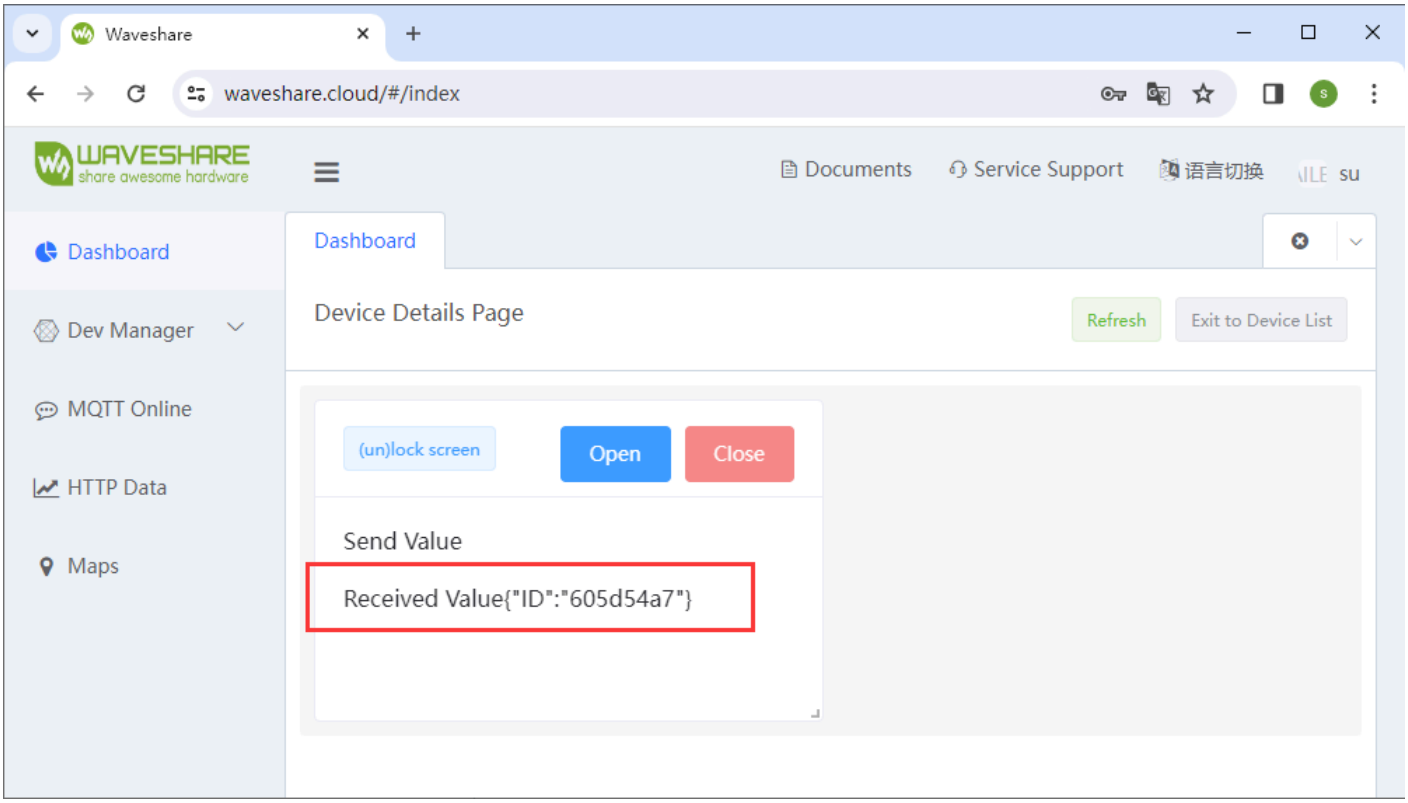


(/wiki/File:ESP32-S3-GEEK_SUB10.png)

- We can observe different feedback regarding changes in device attribute values (e.g., "key") on both the LCD and the serial debugging assistant. Moreover, within Waveshare's cloud platform, under the device's received values, we can view the data transmitted from ESP32-S3-GEEK to the Waveshare cloud device (the received value corresponds to its Client ID). Subsequently, one can send the return value or status of the 'key' back to the Waveshare cloud. This completes the implementation of MQTT for both data uplink and downlink, as well as topic subscription and publication.



(/wiki/File:ESP32-S3-GEEK_SUB11.png)

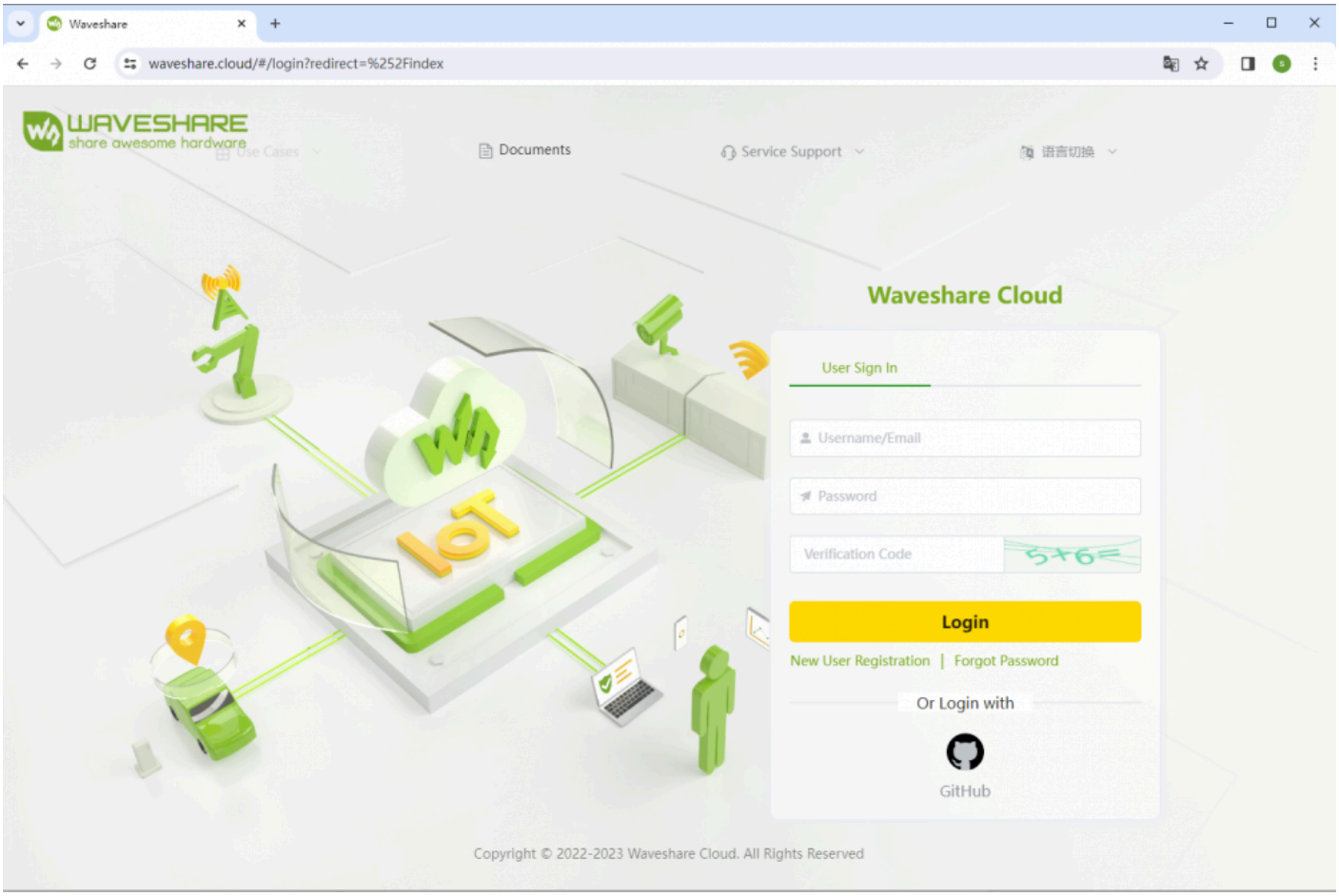


(/wiki/File:ESP32-S3-GEEK_SUB12.png)

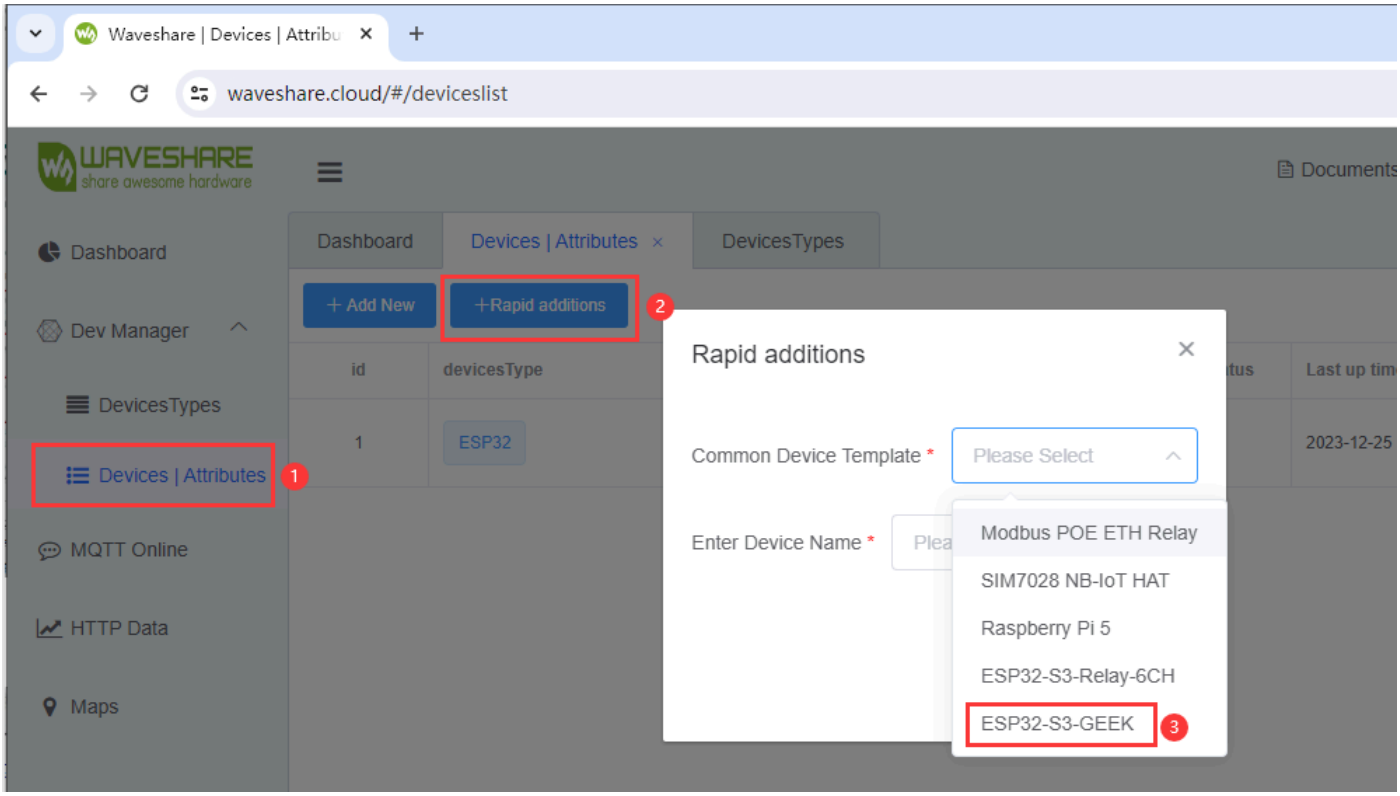
02-MQTT_BLE_Keyboard

This demo enables ESP32-S3-GEEK to enter the STA mode of the WIFI and Bluetooth. After connecting the WIFI and Bluetooth, on the Waveshare Cloud platform, it's possible to remotely lock the screen via Bluetooth and unlock it by entering a password. Additionally, there are more combinations of keys awaiting your development.

- When you are the first time to register on the Waveshare Cloud (https://waveshare.cloud/), you need to create a new device according to the tutorial (https://mqtt.waveshare.cloud/en/waveshare cloud-wiki.html).

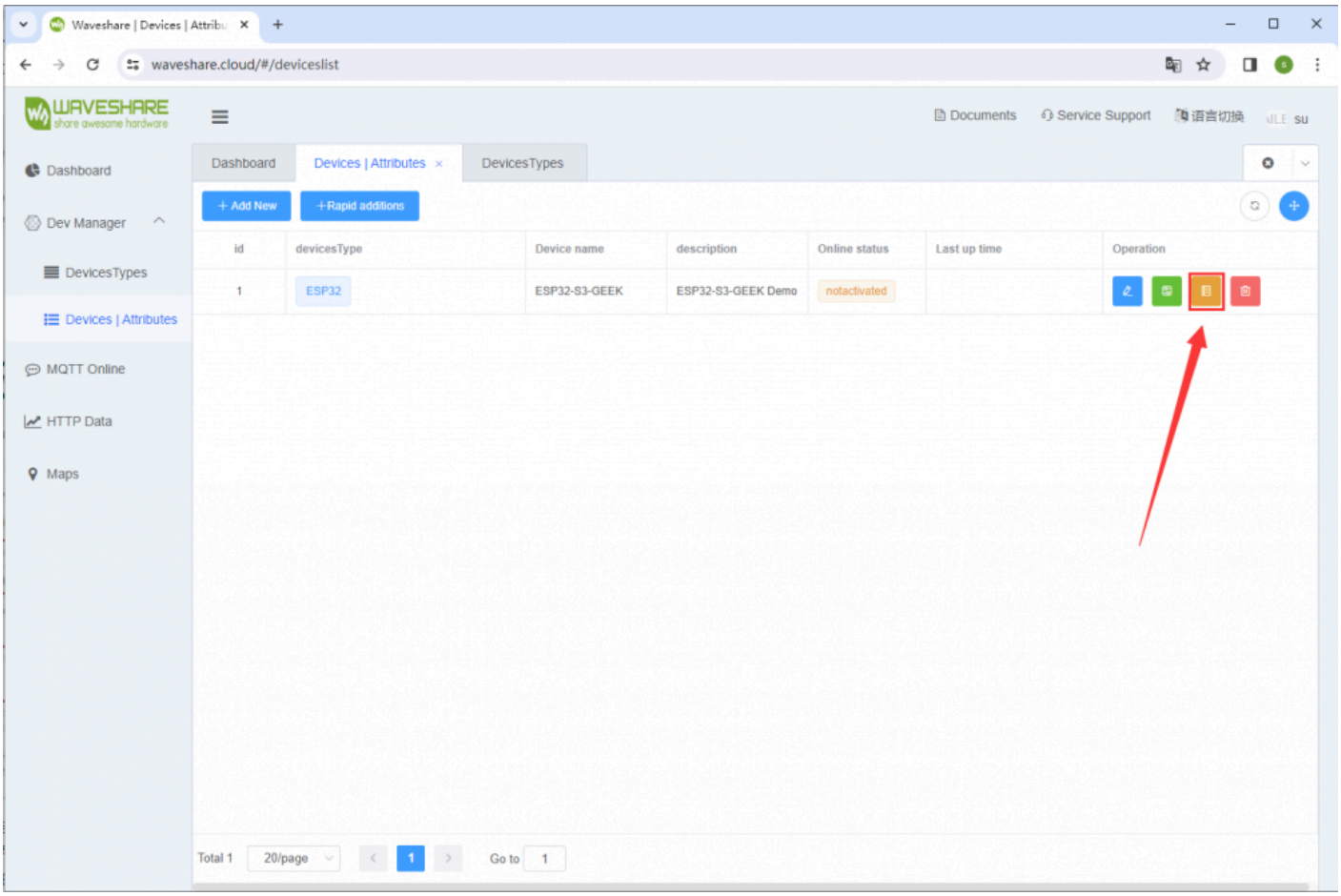


(/wiki/File:ESP32-S3-GEEK_SUB.png)

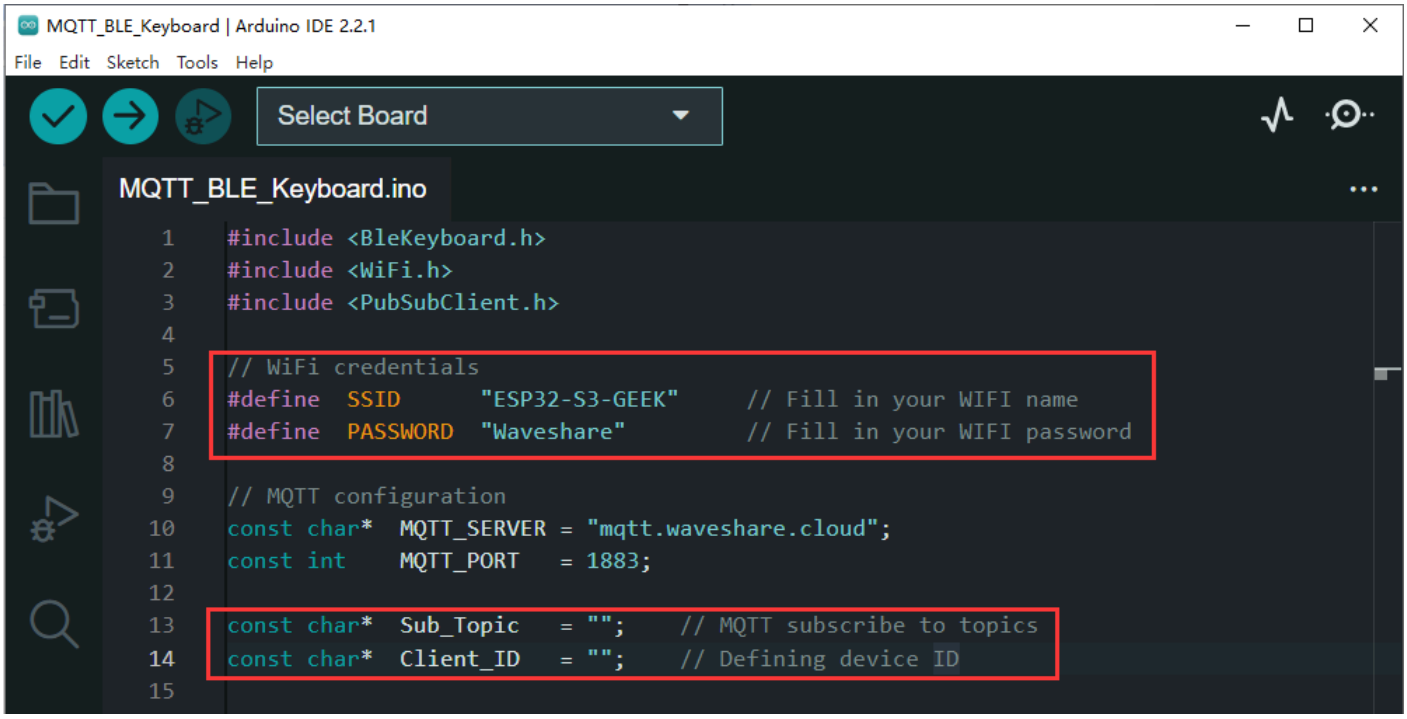


(/wiki/File:ESP32-S3-GEEK_SUB01.png)

- After creating, you can set the corresponding configuration in the Arduino IDE according to ESP32SDK tutorial (https://mqtt.waveshare.cloud/en/esp32-connect.html).
- After the configuration is complete, according to the "View Address" of the newly created device on Waveshare Cloud, you can see the "Client ID, Pub Topic, and Sub Topic" of the device, and you can write them into the demo for assigning values, which can be used for ESP32-S3-GEEK to connect to your cloud platform device.

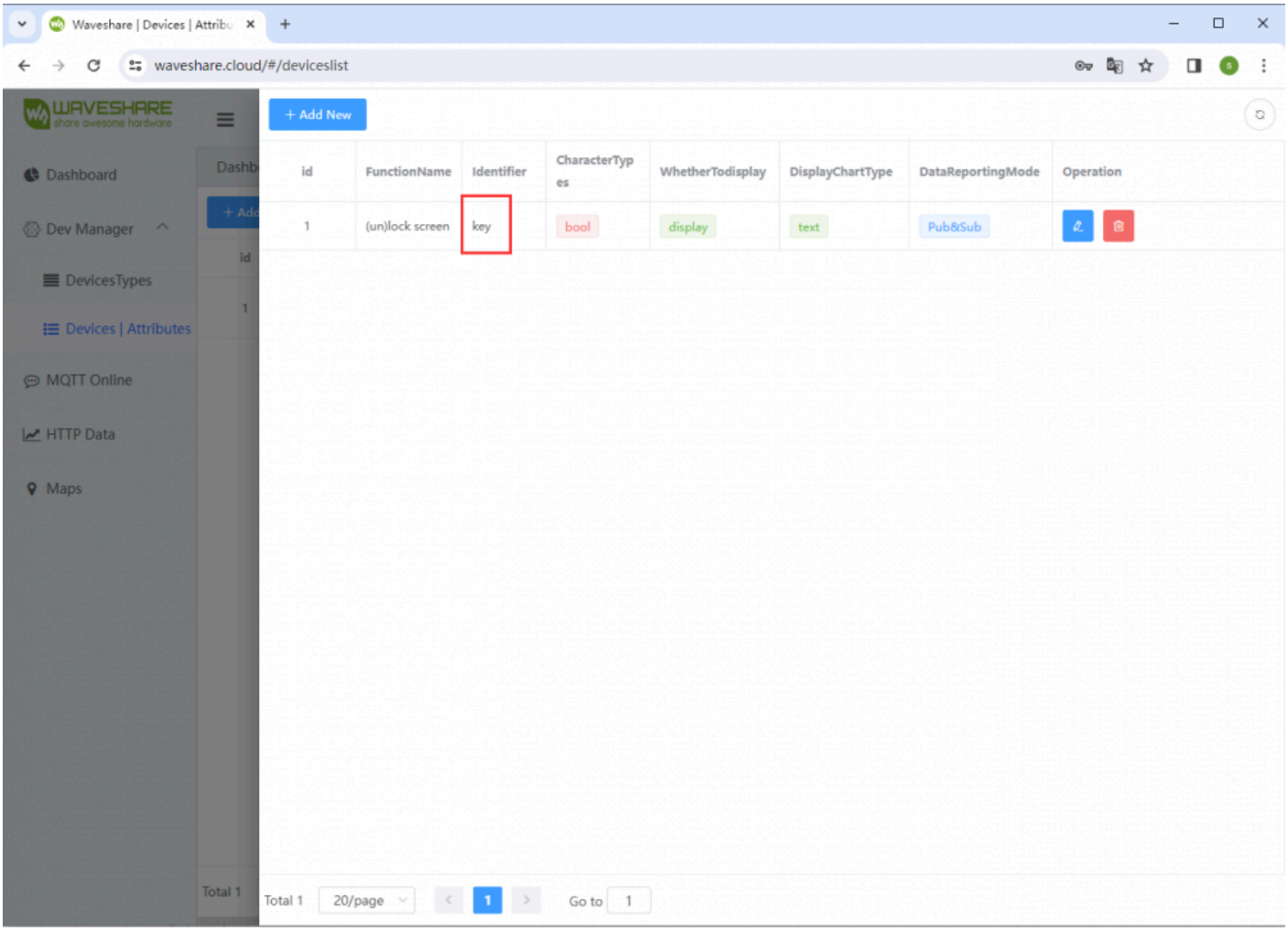


(/wiki/File:ESP32-S3-GEEK_SUB02.png)

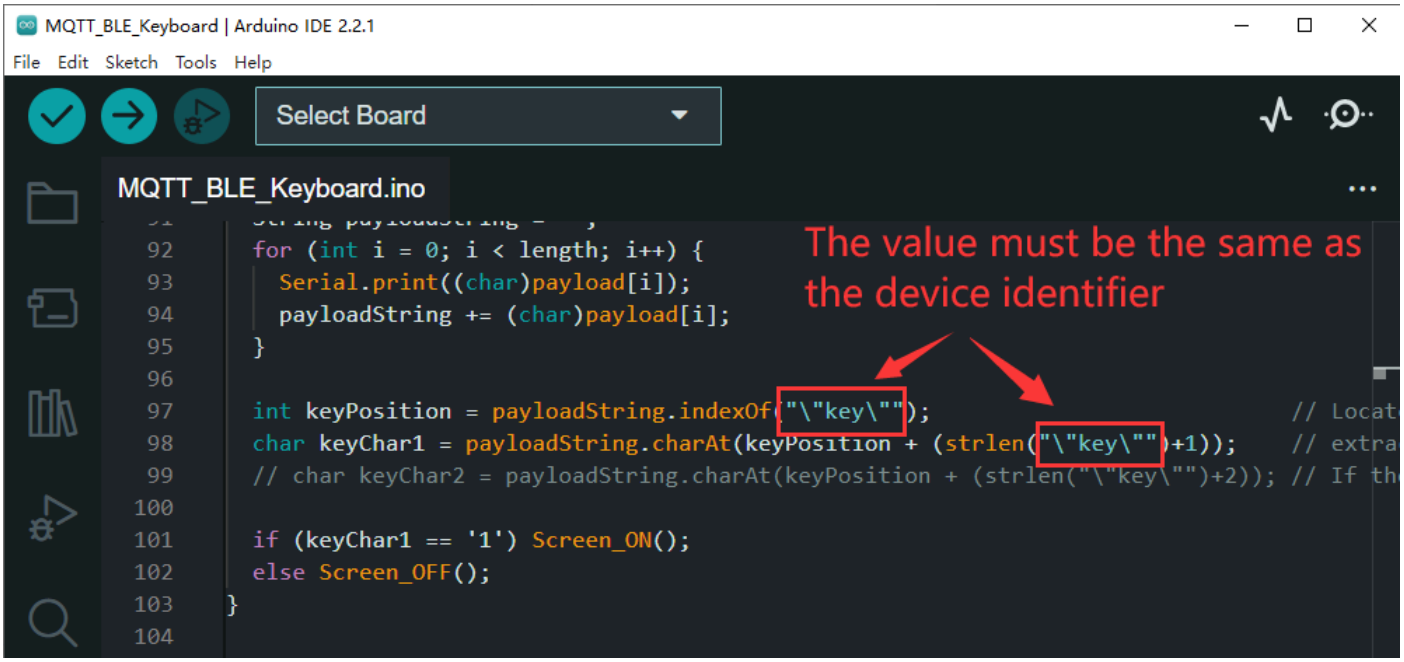


(/wiki/File:ESP32-S3-GEEK_Key03.png)

- ESP32-S3-GEEK connects in STA mode to the same Wi-Fi network (The Wi-Fi network ESP32-S3-GEEK connects to needs to support the 2.4GHz frequency band. If the network doesn't support 2.4GHz, you can create a hotspot on your PC with the network frequency set to "Any available frequency.") Remember to modify the ssid and password with the name and password of the Wi-Fi network you want to connect to.
- In the callback function, you can customize the identification tag to match the device attribute identifier (https://mqtt.waveshare.cloud/en/waveshare cloud-wiki.html#3d7f7136_97) you've created on our cloud platform.

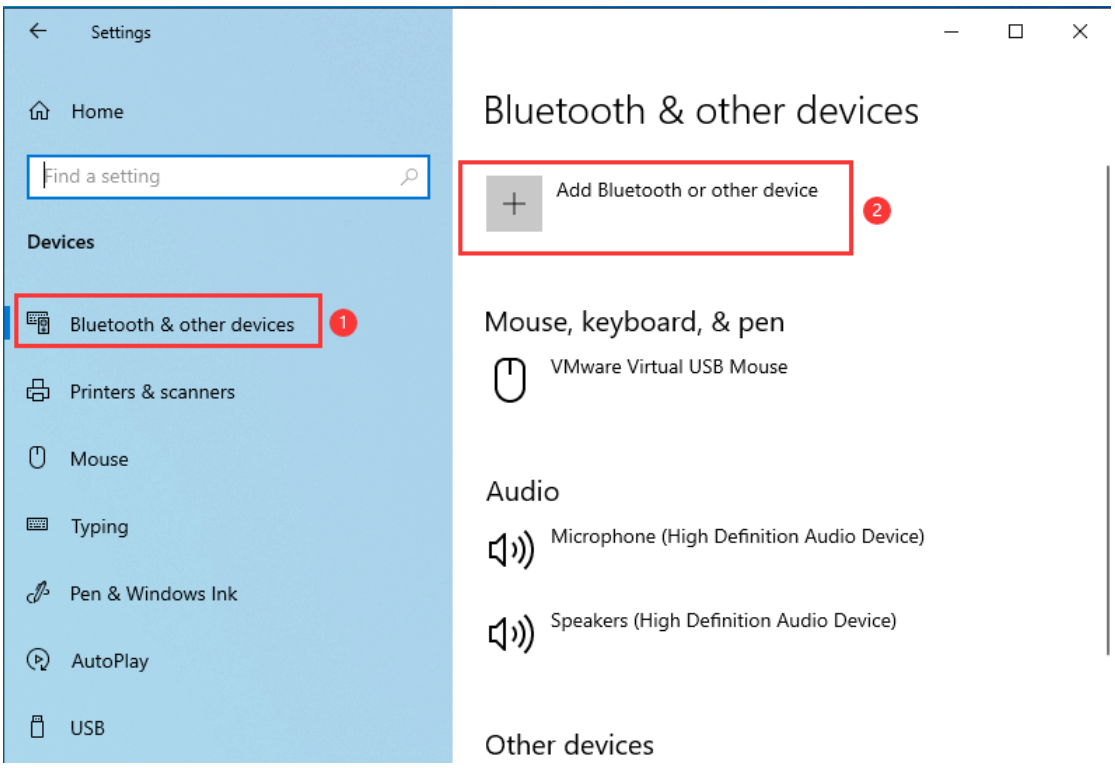


(/wiki/File:ESP32-S3-GEEK_SUB04.png)

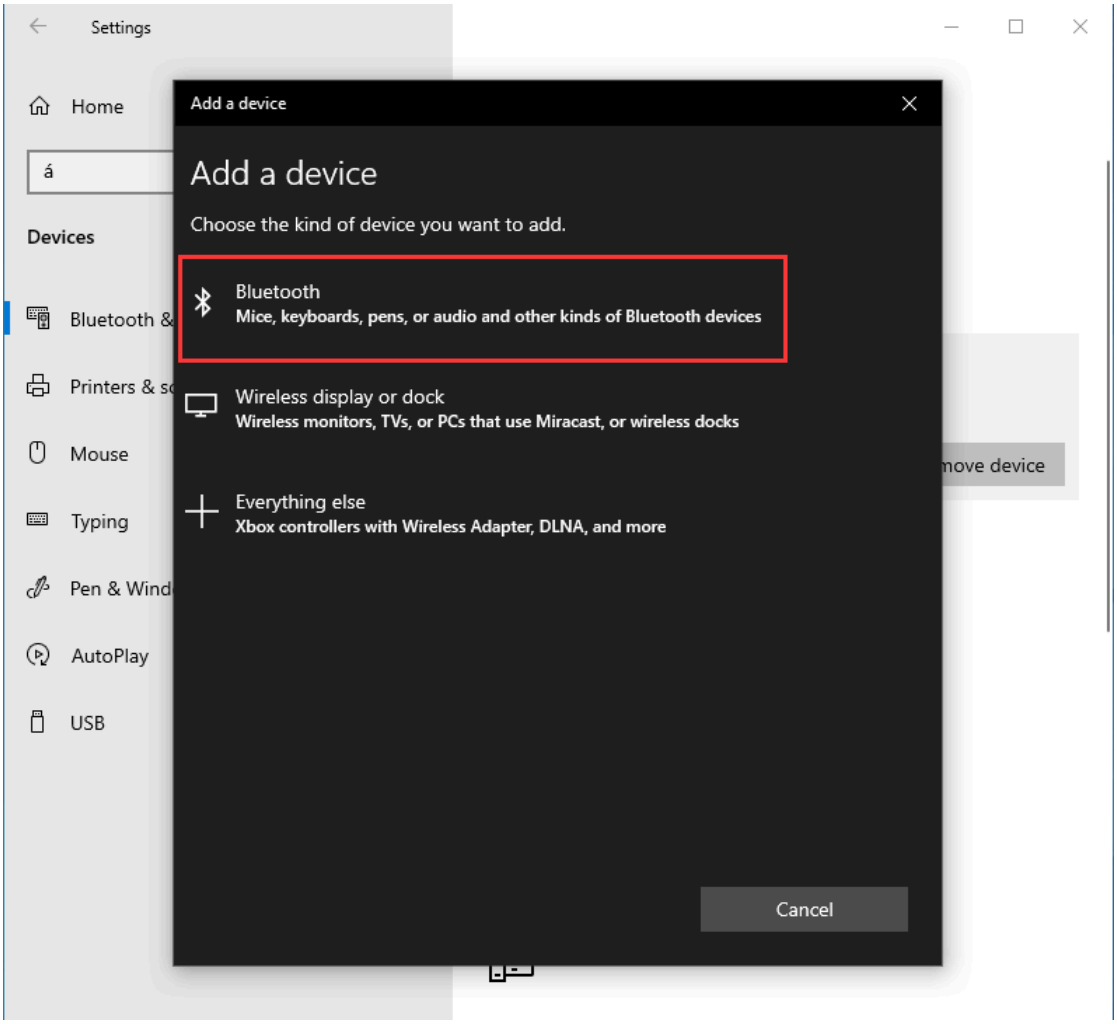


(/wiki/File:ESP32-S3-GEEK_Key05.png)

- Programming the code, after connecting to the WIFI, you can open PC Bluetooth to connect the ESP32-S3-GEEK.

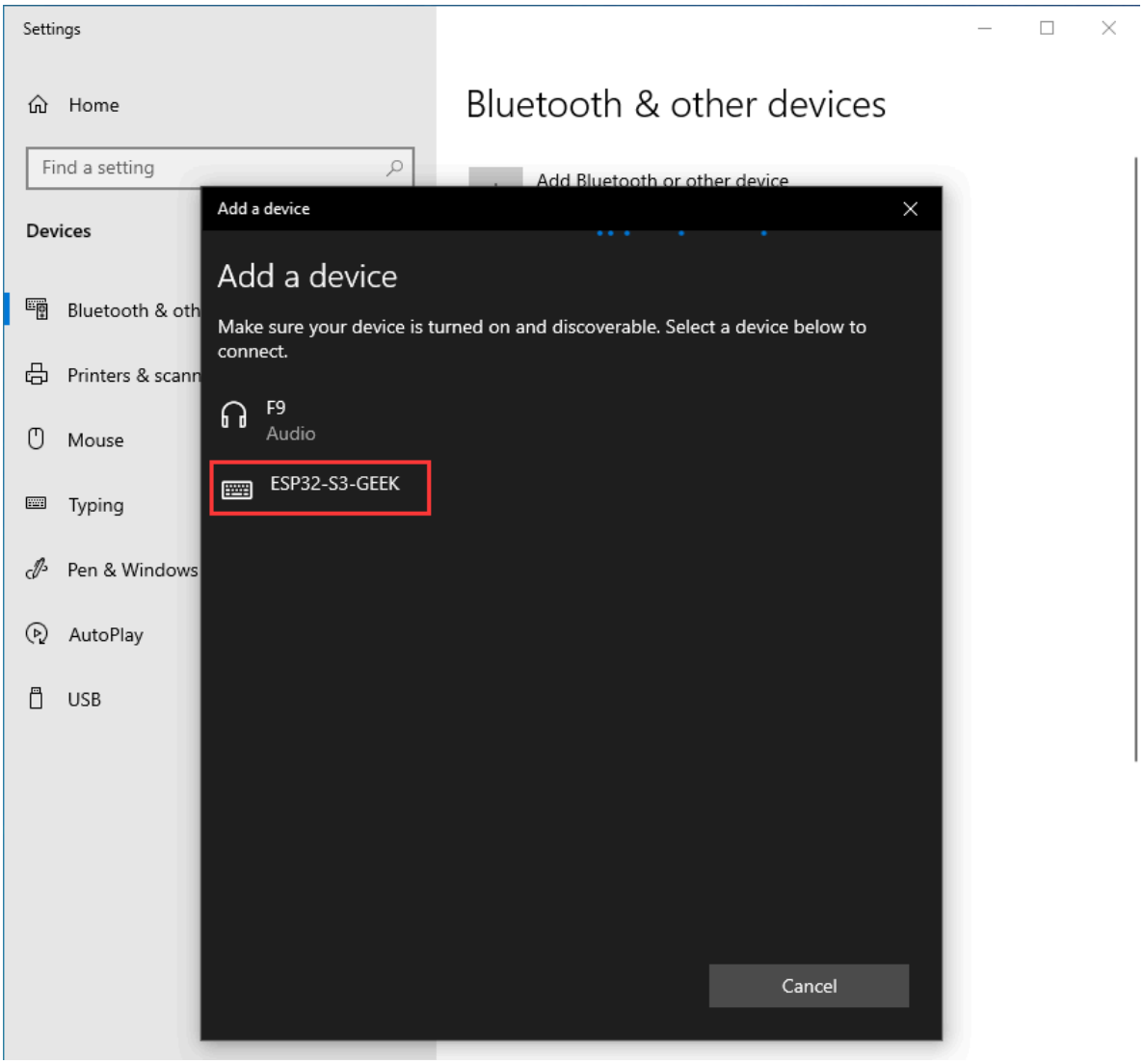


(/wiki/File:ESP32-S3-GEEK_Demo46.png)



(/wiki/File:ESP32-S3-

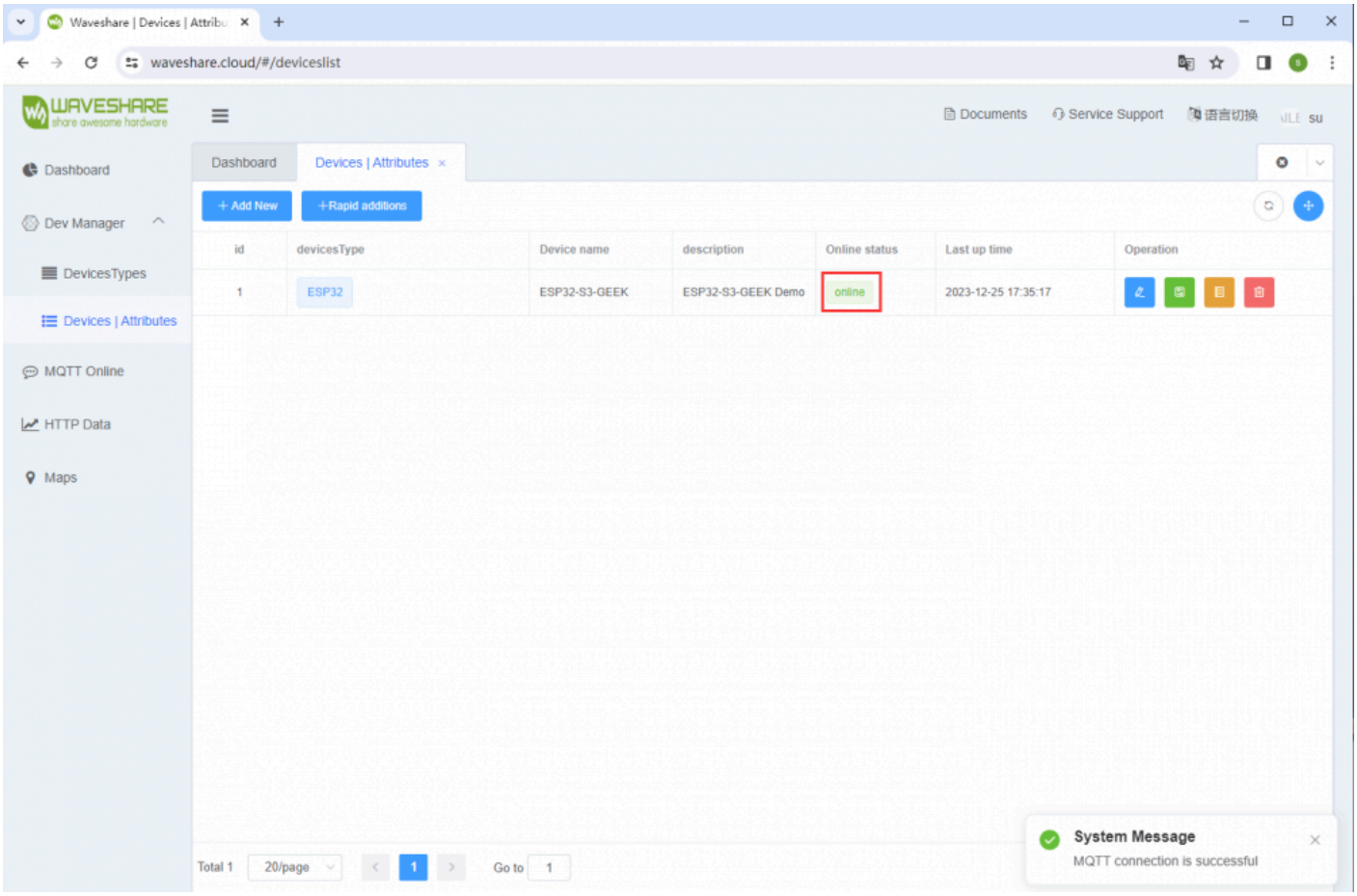
GEEK_Demo47.png)



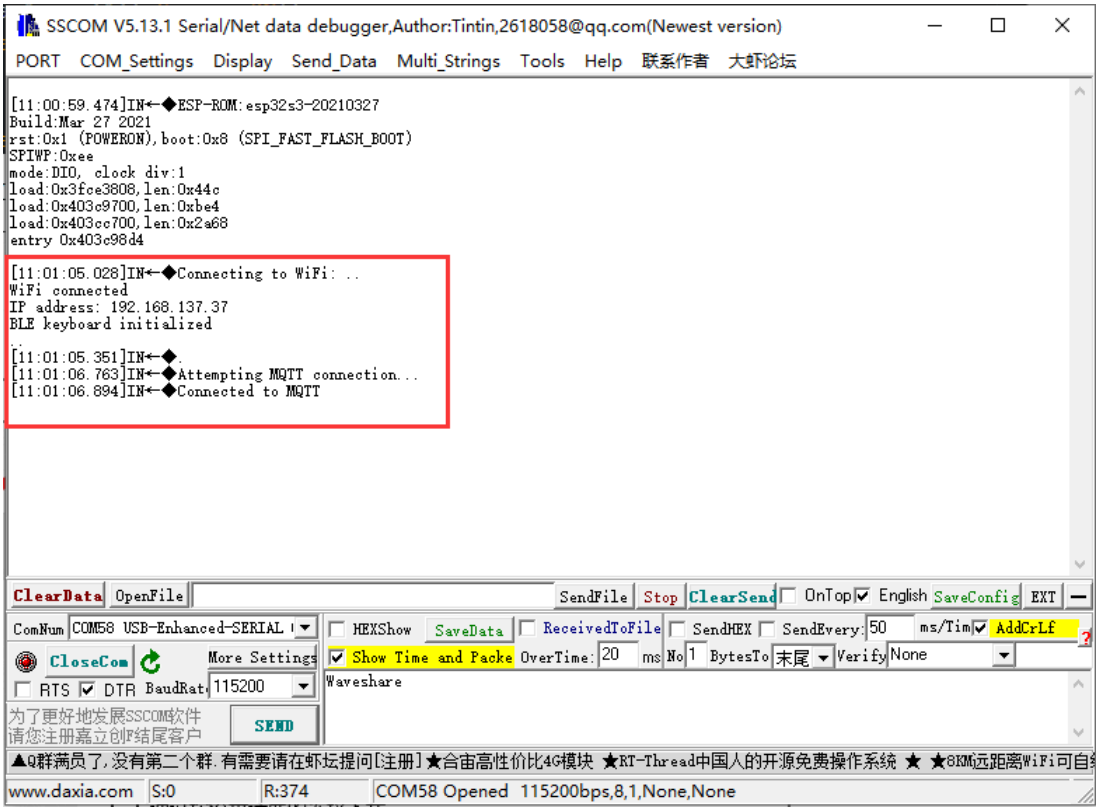
(/wiki/File:ESP32-S3-

GEEK_Demo48.png)

- Observe whether the device is on the Waveshare cloud platform in the online state, if not, you can try to refresh the web page with the use of CH343_USB_UART_Board (https://www.waveshare.com/ch343-usb-uart-board.htm)to connect to the PC through the serial debugging assistant to view the WIFI and MQTT connection, and the LCD will also display the WIFI and MQTT connections.

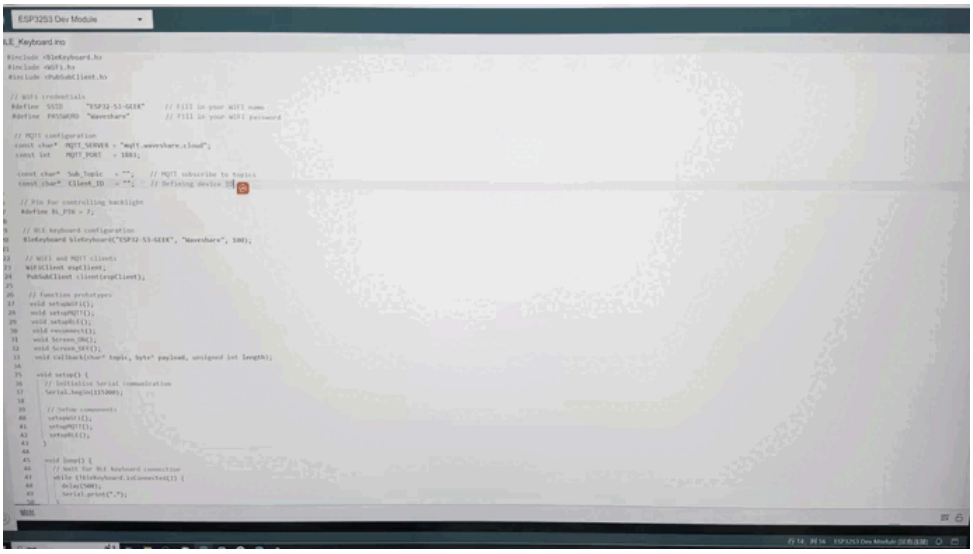


(/wiki/File:ESP32-S3-GEEK_SUB07.png)



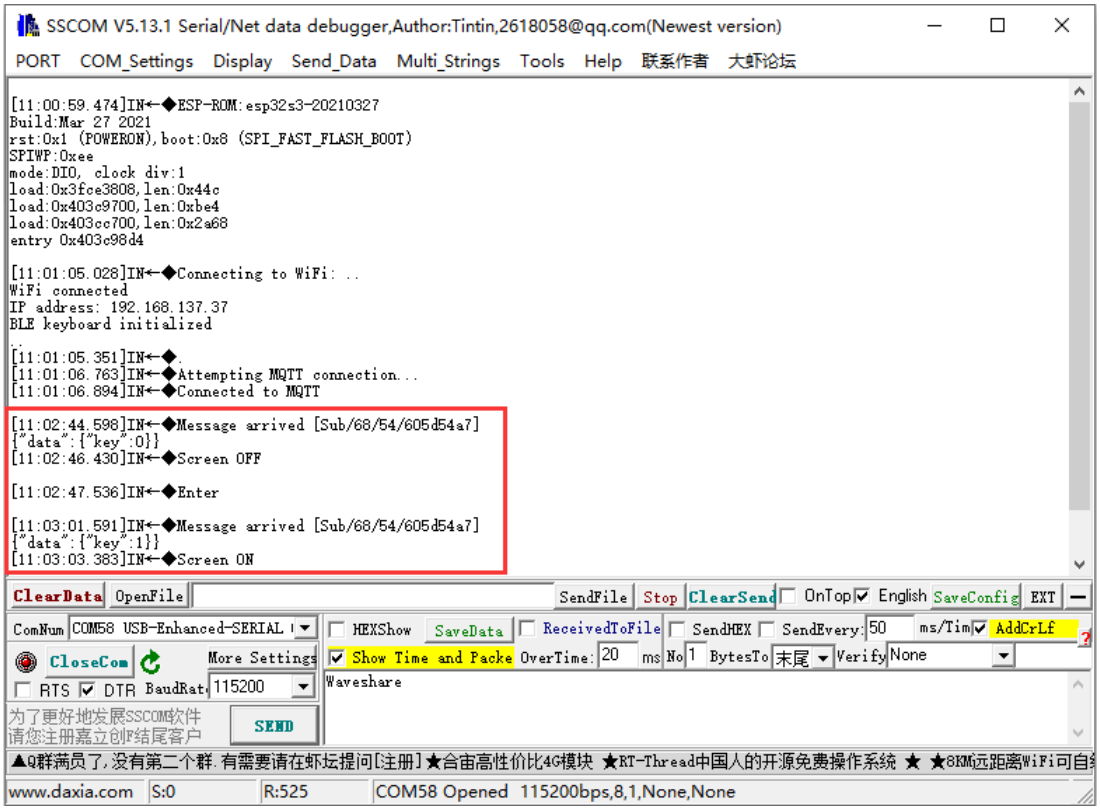
(/wiki/File:ESP32-S3-GEEK_G02.png)

- After the ESP32-S3-GEEK connects to the Waveshare Cloud, you can log in to the Waveshare Cloud Dashboard on your phone remotely lock the PC screen, and input a password for screen unlocking.



(/wiki/File:ESP32-S3-GEEK_G03.gif)

- We can observe different feedback regarding changes in device attribute values (e.g., "key") on both the LCD and the serial debugging assistant. Additionally, we can further customize the callback function to modify keys into combinations like Ctrl+C, and Ctrl+V, allowing for a personalized DIY remote-controlled Bluetooth keyboard.



(/wiki/File:ESP32-S3-GEEK_G04.png)

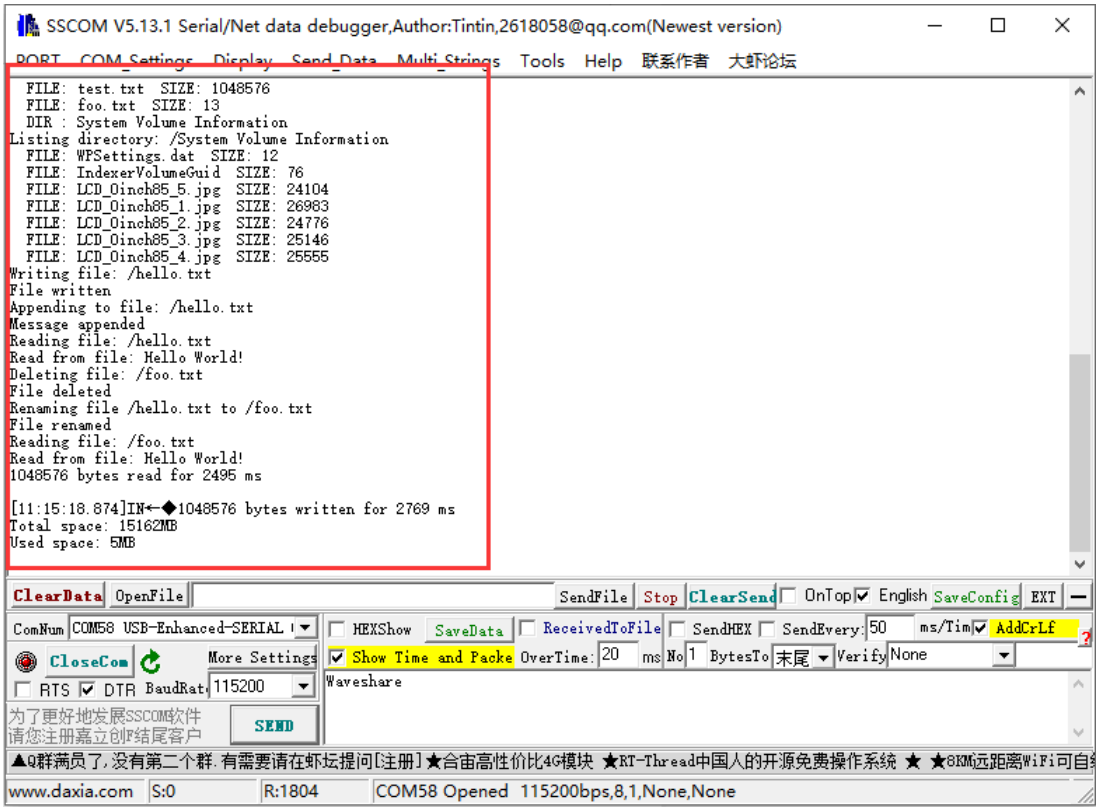
- Or you can enter keyboard tester (https://www.zfrontier.com/lab/keyboardTester) website to test the keys remotely pressed by the ESP32-S3-GEEK Bluetooth.

(/wiki/File:ESP32-S3-GEEK_G05.jpg)

SD

01-SD_Test

```
This demo allows the use of ESP32-S3-GEEK's SD card slot. Insert an SD card into the slot and open the serial debugging assistan
t. You'll be able to observe ESP32-S3-GEEK performing operations such as creating, deleting, modifying, and checking files on th
e SD card.
```



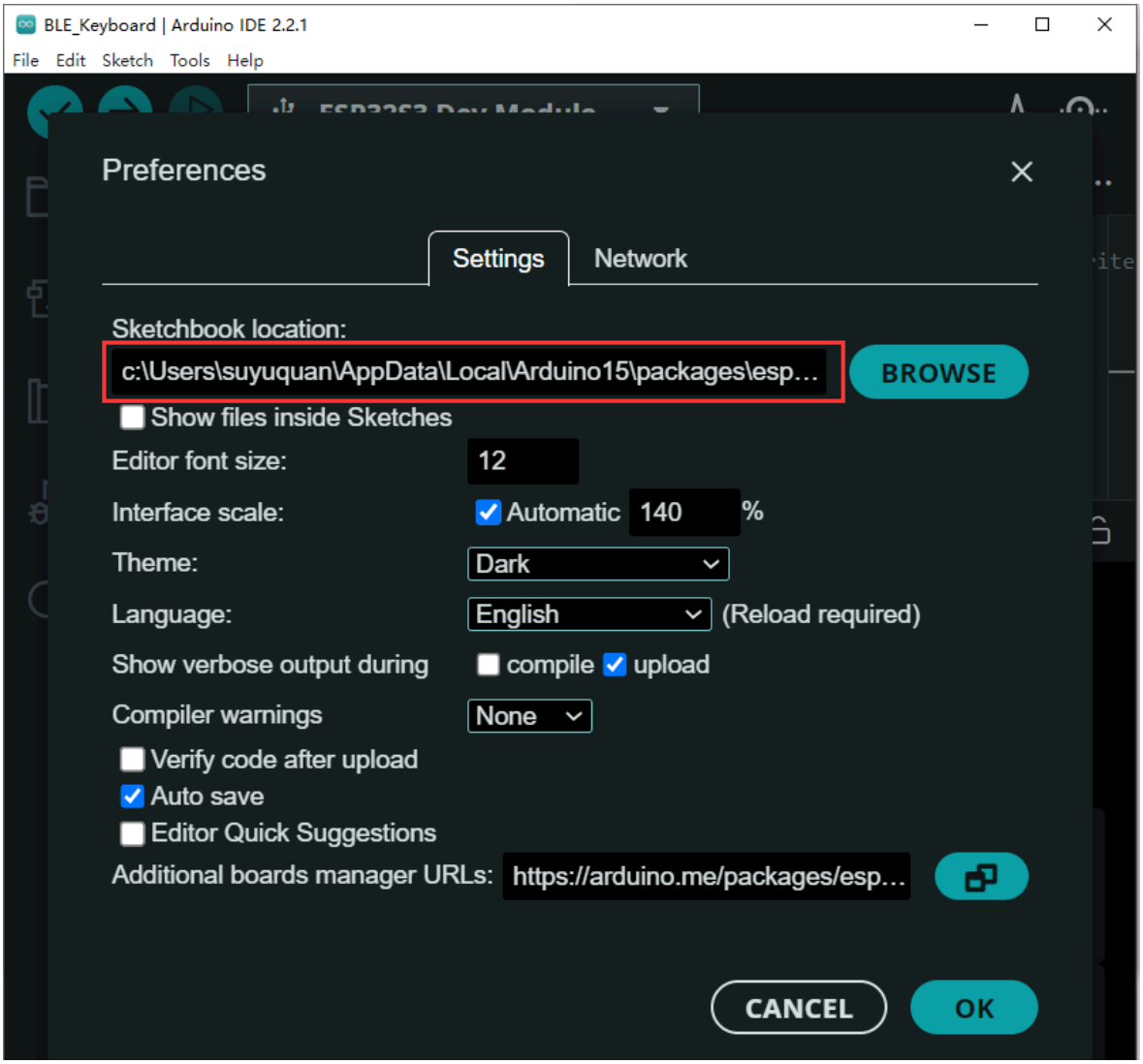
(/wiki/File:ESP32-S3-GEEK_SD.png)

02-SD_LCD

```
With this demo, the pictures of the SD card in the SD card slot of the ESP32-S3-GEEK can be read. Store the pictures on the SD c
ard, and insert it into the card slot. The ESP32-S3-GEEK can read and display the pictures of the SD card on the LCD, or it can
flash the pictures as a gif.
```

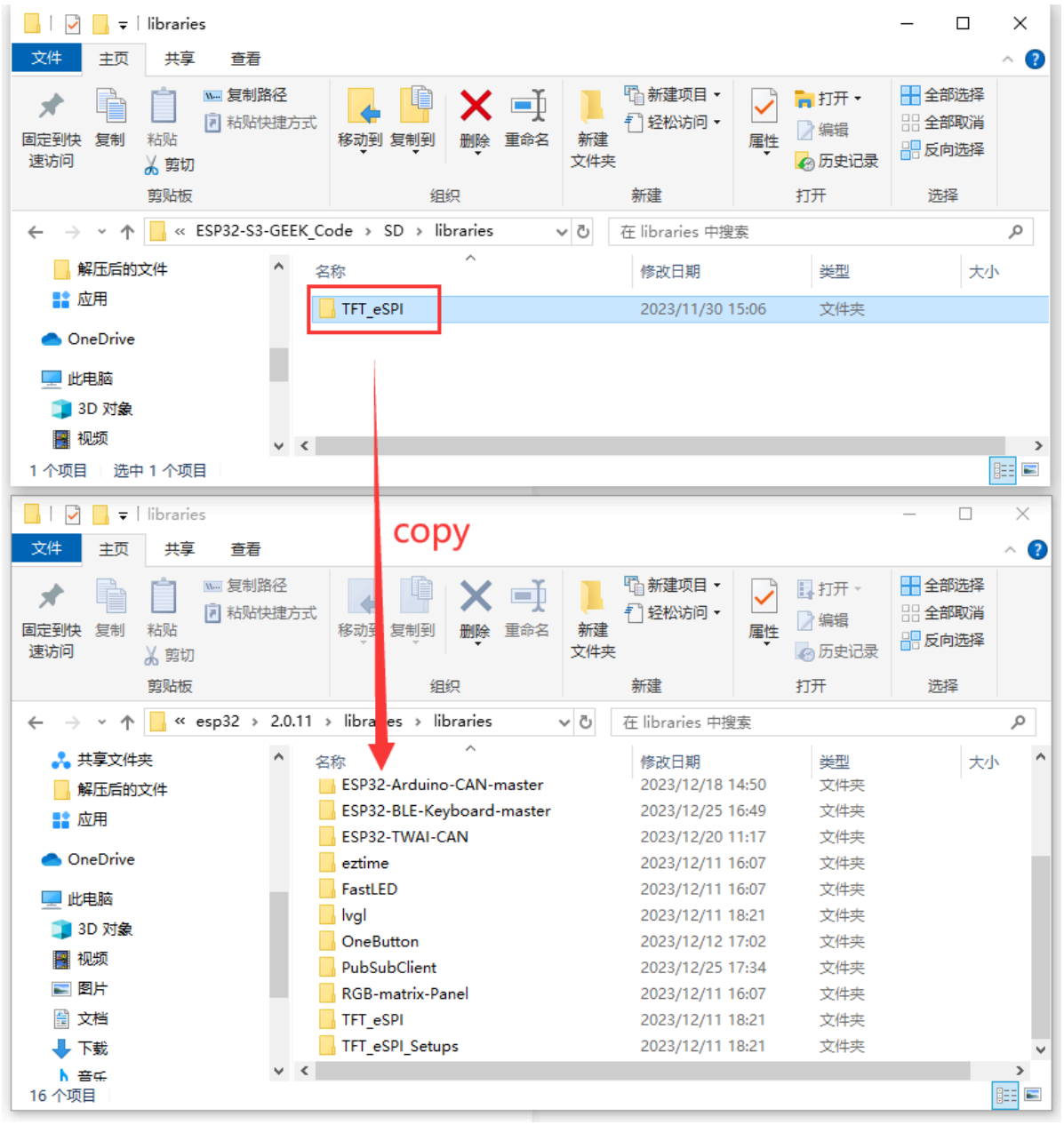
- Copy the TFT_eSPI file folder of the libraries to the installation directory of Arduino library.

(/wiki/File:ESP32-S3-GEEK_Demo40.png)



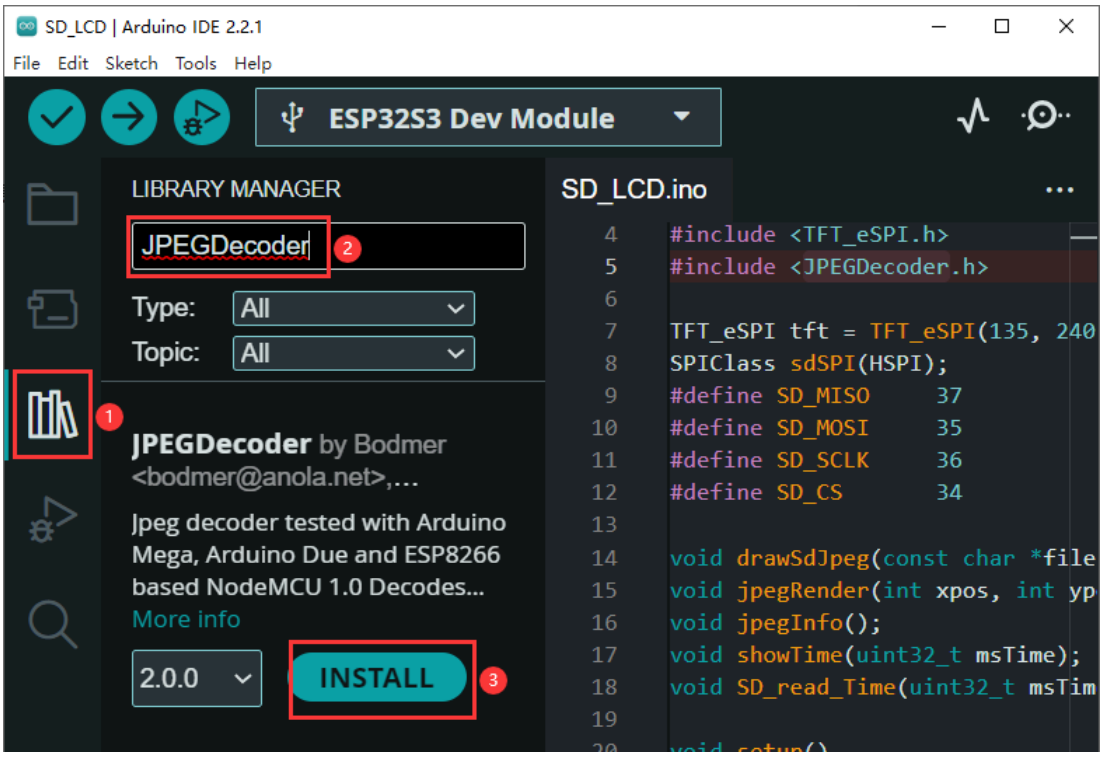
(/wiki/File:ESP32-S3-

GEEK_Demo41.png)



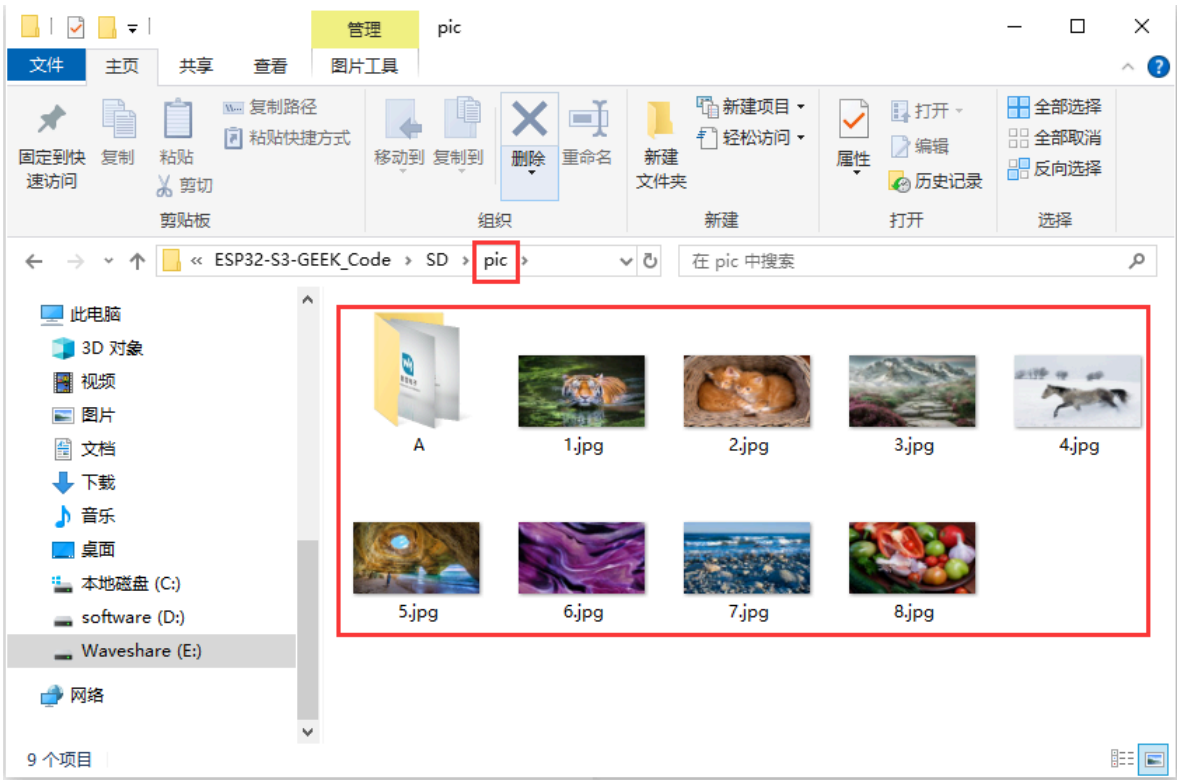
(/wiki/File:ESP32-S3-GEEK_LCD03.png)

- Install JPEGDecoder library, LIBRARY MANAGER -> search for JPEGDecoder -> INSTALL.



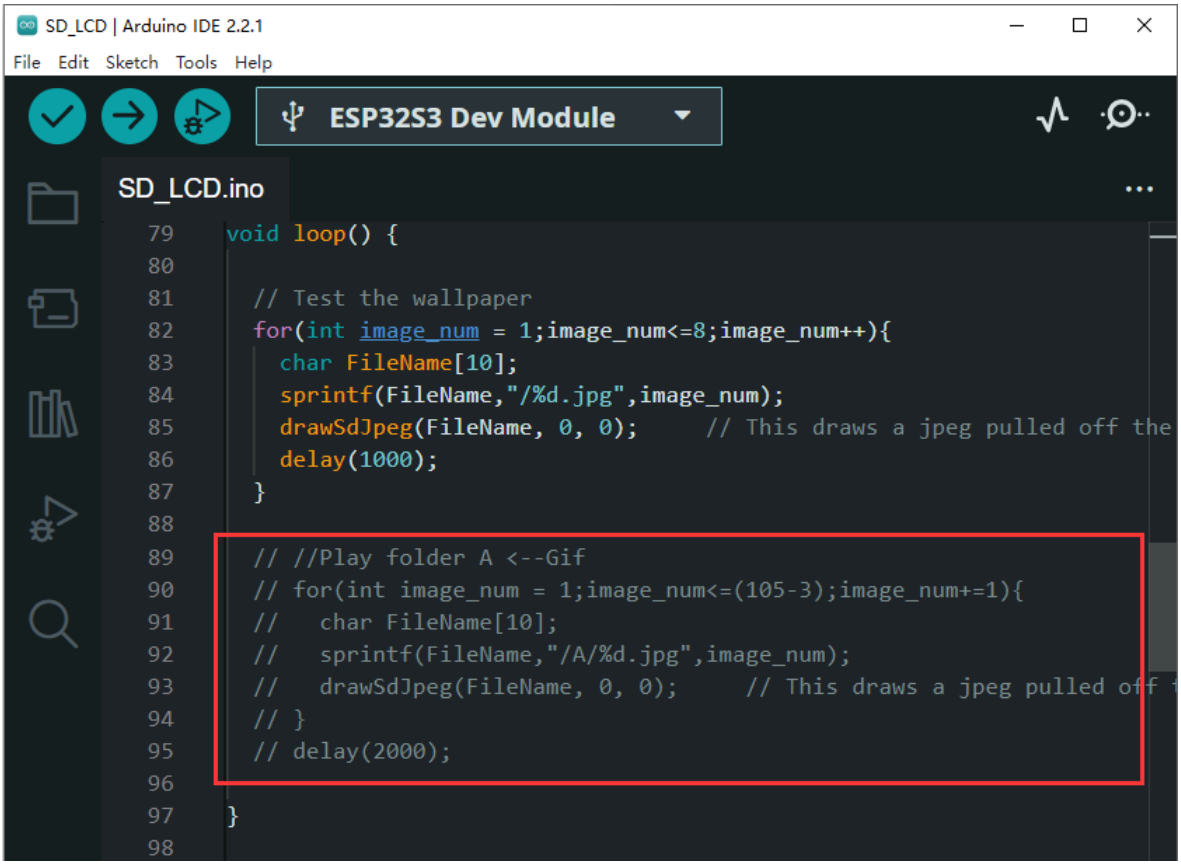
(/wiki/File:ESP32-S3-GEEK_LCD04.png)

- You can store the pictures of "pic" on the SD card, or you can store your pictures, and modify the picture size to 240×135, and the display effect is the best.



(/wiki/File:ESP32-S3-GEEK_LCD05.png)

- This part of the comment is removed to play the Waveshare startup animation.



(/wiki/File:ESP32-S3-GEEK_LCD06.png)

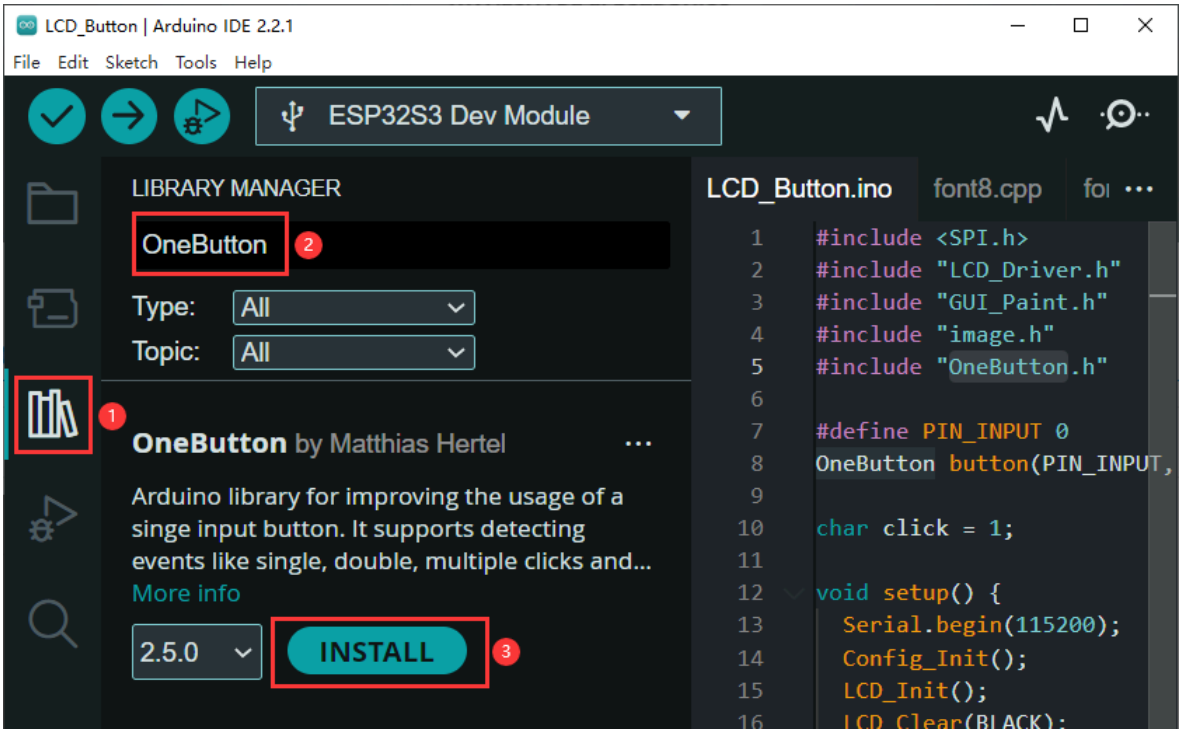

(/wiki/File:ESP32-S3-GEEK_LCD07.gif)

LCD

01-LCD_Button

This demo can use the boot button of ESP32-S3-GEEK to realize a short press to light up the LCD and switch to the next picture, and a long press to turn off the LCD.

- Install OneButton library, LIBRARY MANAGER -> search for OneButton -> INSTALL.

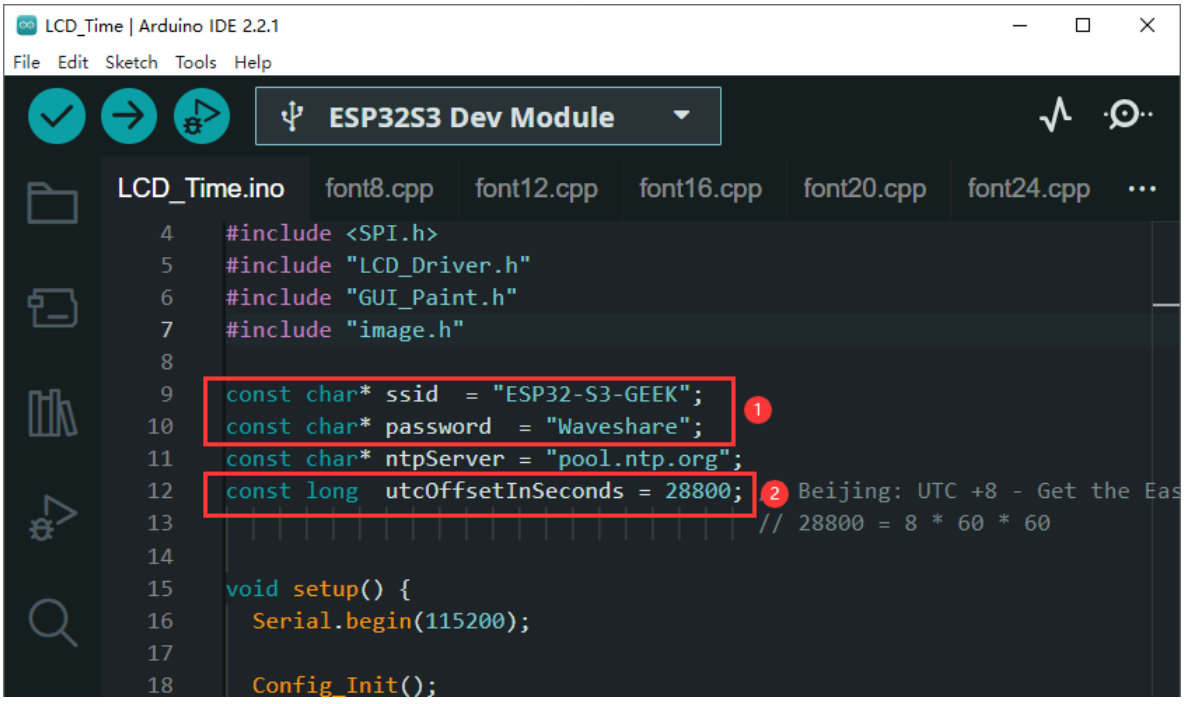


(/wiki/File:ESP32-S3-GEEK_LCD_Time03.png)

02-LCD_Time

```
This demo allows ESP32-S3-GEEK to connect to WiFi and obtain the current time. It then displays the date and time on both the LCD and the serial debugging assistant.
```

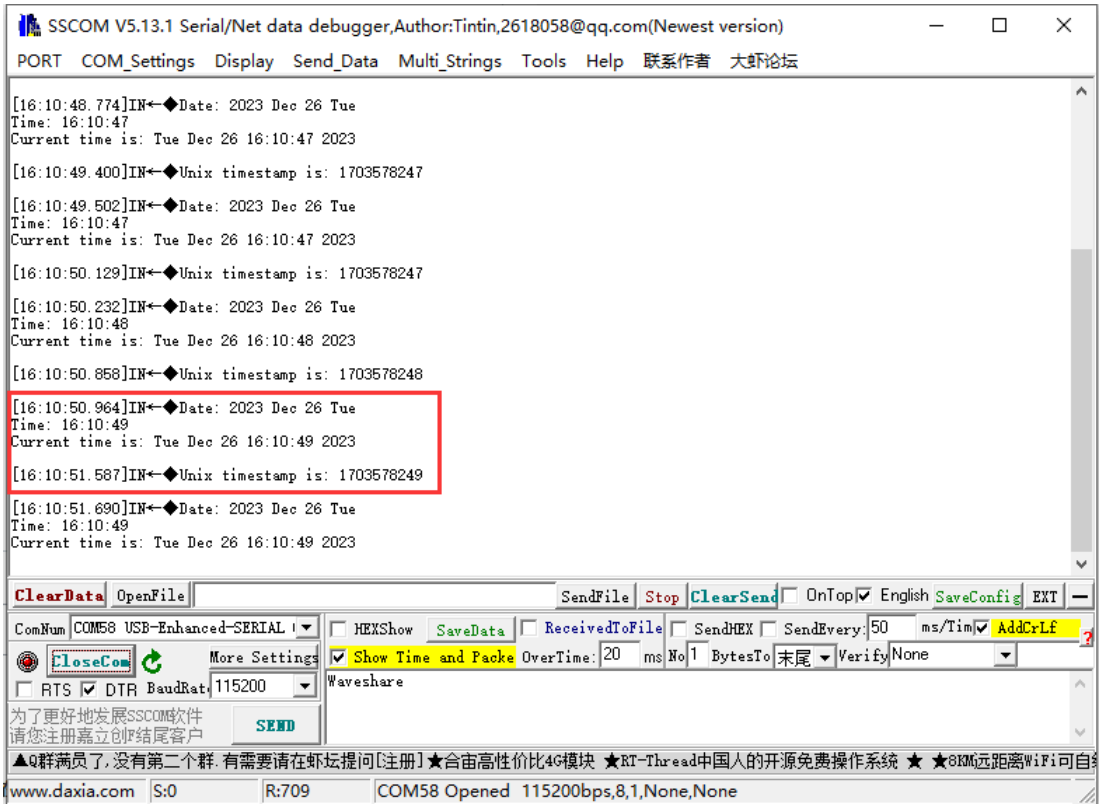
- Modify the ssid and password to the name and password of the Wi-Fi network you want. (The WIFI of the ESP32-S3-GEEK connected requires the Wi-Fi network of the 2.4GHz frequency band. If there's no 2.4GHz band available, create a hotspot on your PC with the network frequency set to "Any available frequency".) For instance, if you need the time zone for Beijing (GMT+8), set utcOffsetInSeconds to 28800.(8*60*60)



(/wiki/File:ESP32-S3-GEEK_LCD_Time01.png)

- After programming the code, you can see the LCD display the real-time time and date. ESP32-S3-GEEK uses USB to UART to connect to the PC, open the serial port debugging assistant, and you can see the obtained time and date printed on the serial

port debugging assistant.



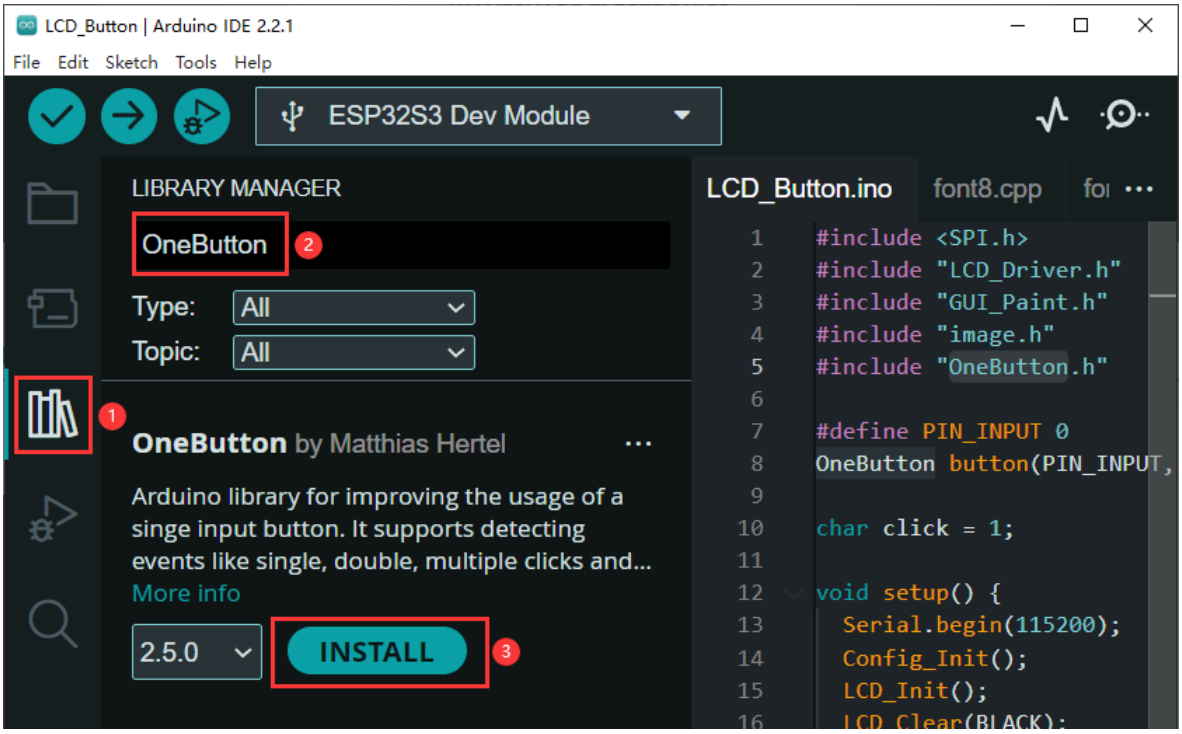
(/wiki/File:ESP32-S3-GEEK_LCD_Time02.png)

Button

01-OneButton

```
With this demo, the boot key on the ESP32-S3-GEEK is modified as a multi-functional key, capable of executing different operatio
ns based on single-click, double-click, or long-press actions.
```

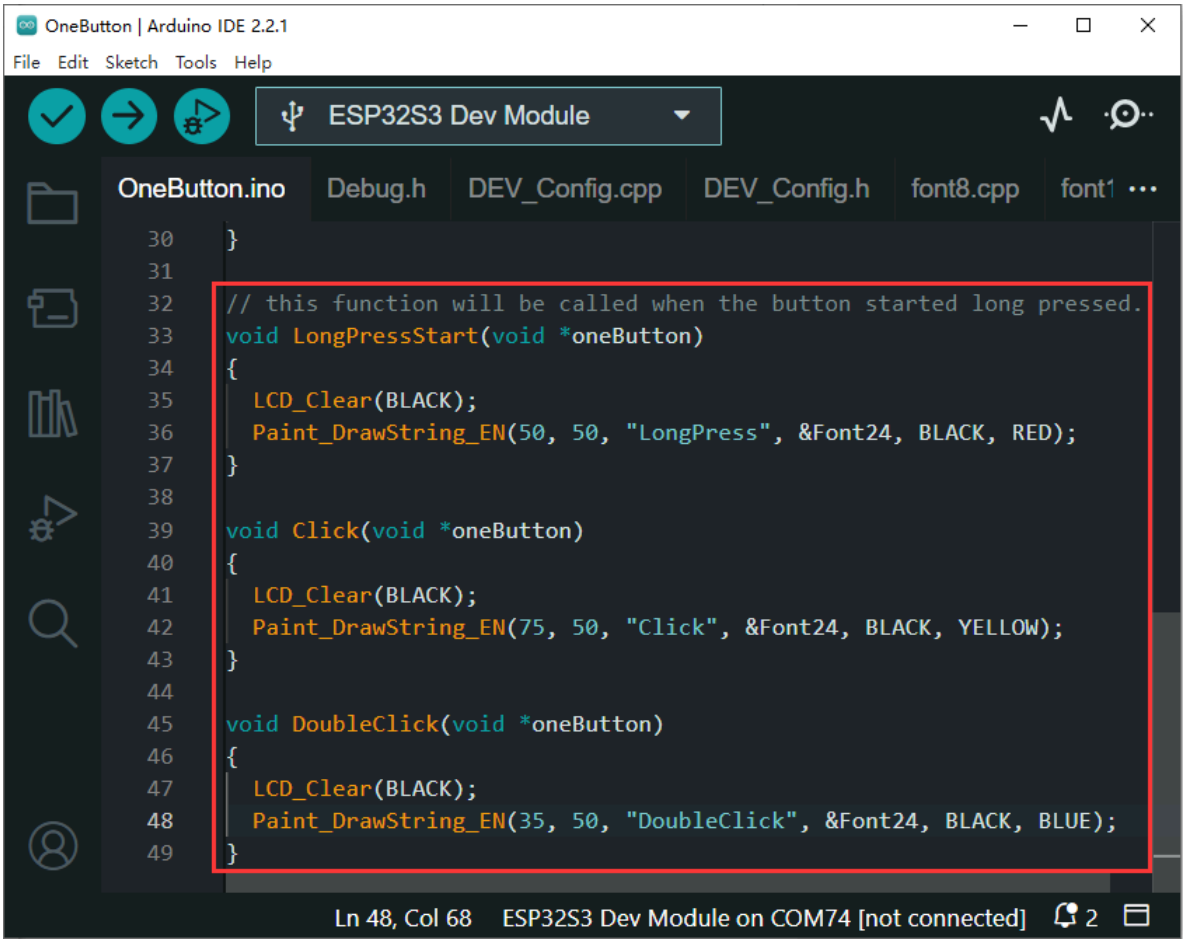
- Install OneButton library, LIBRARY MANAGER -> search for OneButton-> INSTALL.



(/wiki/File:ESP32-S3-GEEK_LCD_Time03.png)

- You can modify different processing functions by the following code. For more details, you can click here (https://github.com/mathertel/OneButton) to view.

```
Once a click is detected, it triggers: void Click(void *oneButton)
Once a double click is detected, it triggers: void DoubleClick(void *oneButton)
Once long press is detected, it triggers: void LongPressStart(void *oneButton)
```



(/wiki/File:ESP32-S3-GEEK_LCD_Time04.png)

UART

01-UART0

This demo uses ESP32-S3-GEEK to open UART0 and the serial port assistant for UART communication.

IIC

01-IIC_BME68X_Sensor

With this demo, the IIC hardware interface of the ESP32-S3-GEEK drives the IIC module.

- The demo uses the BME680 environmental sensor (https://www.waveshare.com/bme68x-environmental-sensor.htm). You can find detailed instructions and implementation effects in the tutorial (https://www.waveshare.com/wiki/BME680_Environmental_Sensor#Working_with_ESP32).

ADC

01-ADC_Read

```
This example allows the use of ESP32-S3-GEEK's GPIO interface for ADC sampling, enabling the reading of voltages within the 3.3V range. When using it, ensure a common ground and avoid exceeding the measurement range.
```

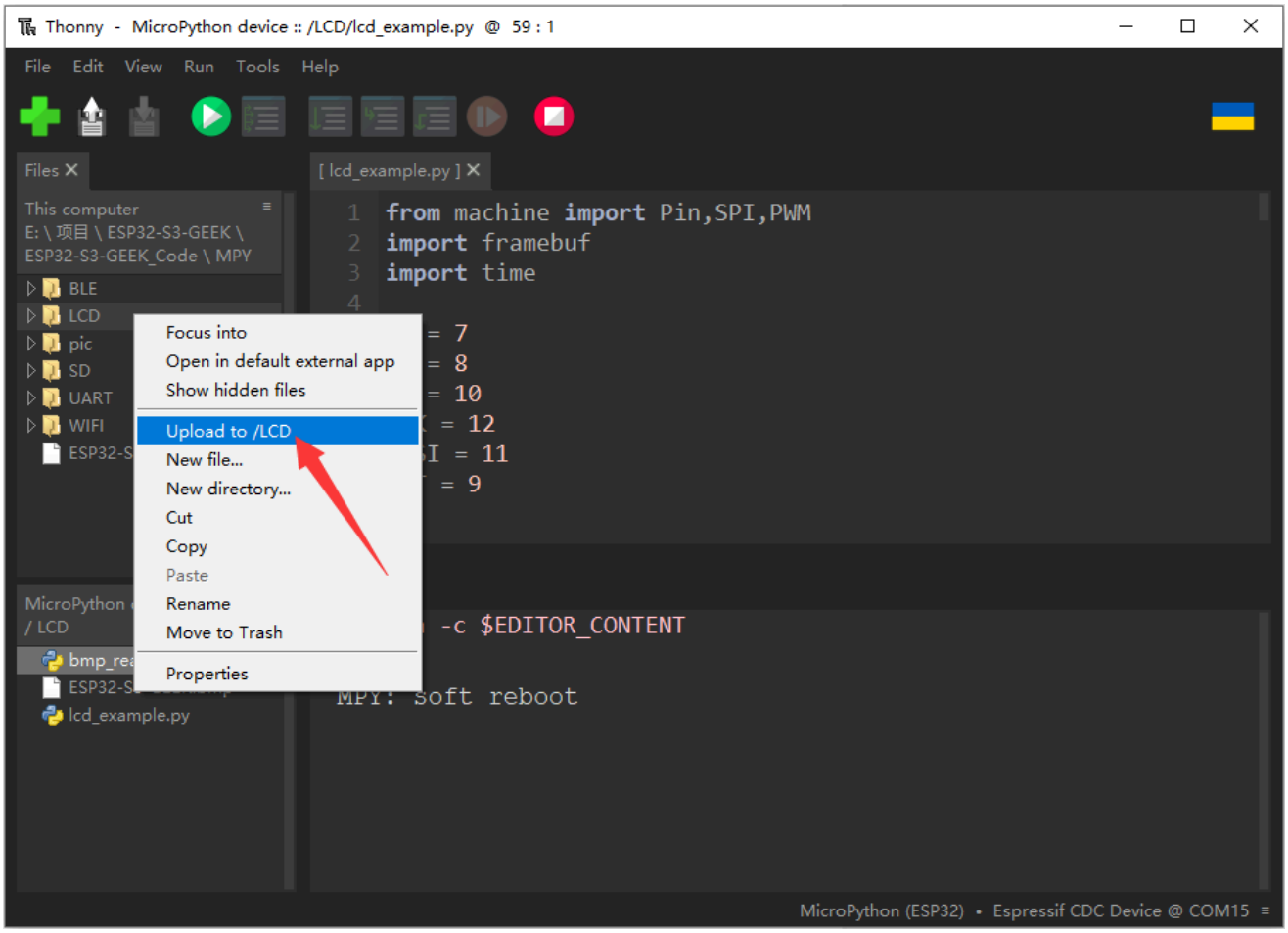
Micropython Sample Demo

> **Note**: Before using the Micropython demo, you need to check whether the Micropython environment and the download setting are correctly configured. For more details, you can refer to Micropython (/wiki/ESP32-S3-GEEK#MicroPython).

LCD

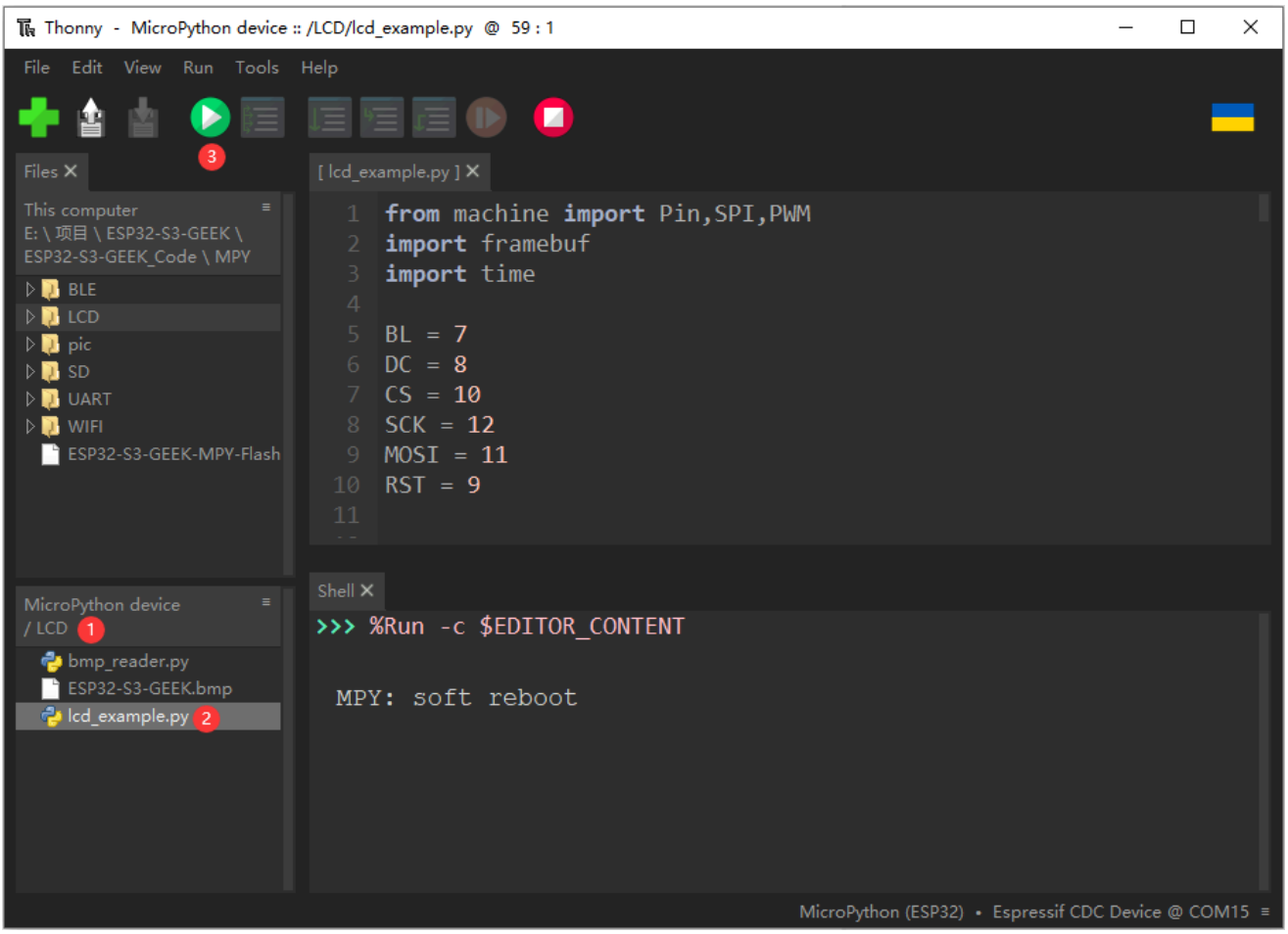
```
This demo uses the ESP32-S3-GEEK to open the LCD screen to display the text and BMP pictures.
```

- Upload the LCD file folder of the MPY file to ESP32-S3-GEEK.



(/wiki/File:ESP32-S3-GEEK-MPY_10.png)

- Enter the lcd_example.py of the ESP32-S3-GEEK, click to run, and observe the LCD screen display.

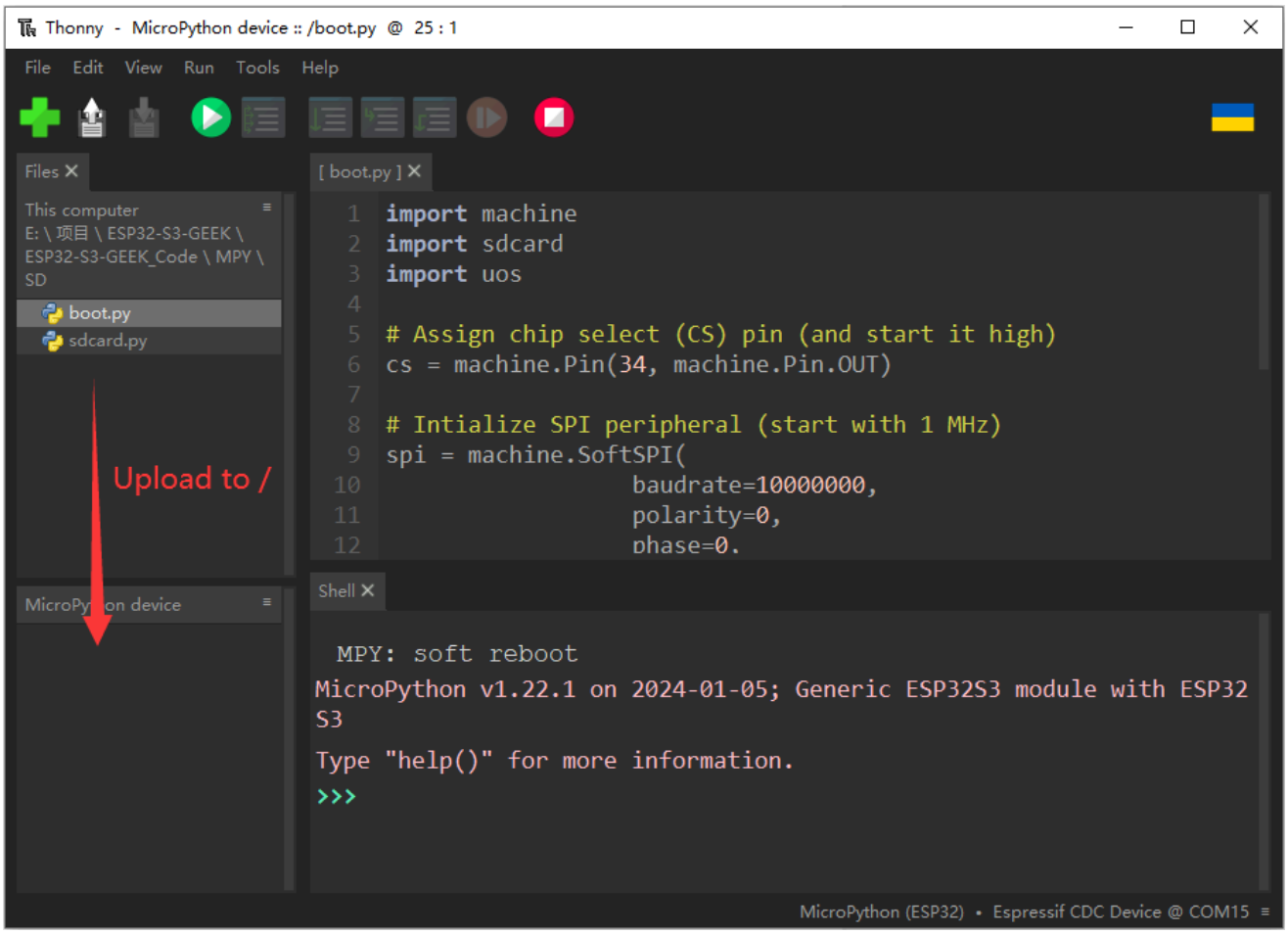


(/wiki/File:ESP32-S3-GEEK-MPY_11.png)

SD

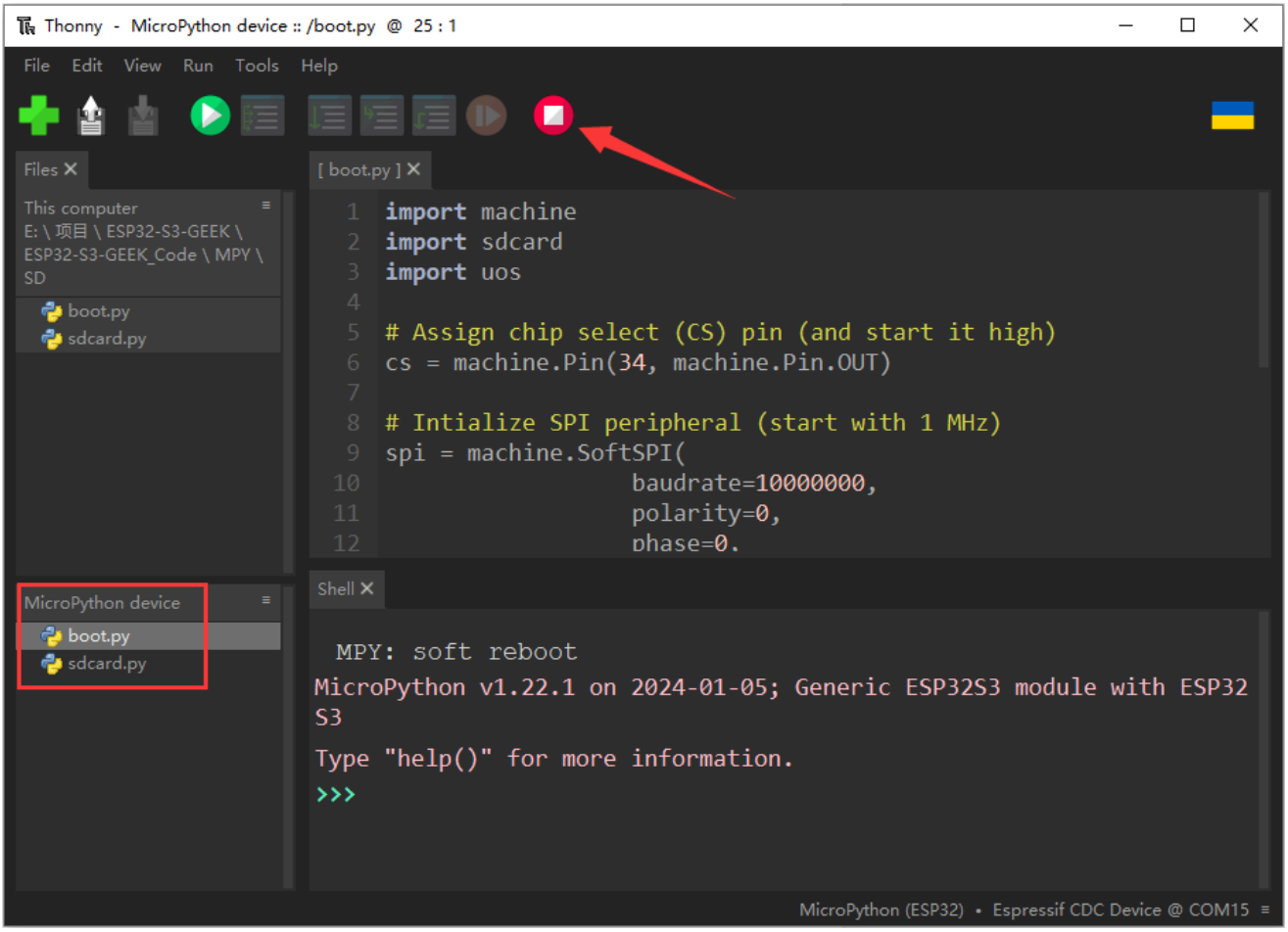
```
This demo can read the SD card slot of the ESP32-S3-GEEK, and directly open the SD file folder of the ESP32-S3-GEEK to browse th
e files. Before using it, you need to insert the SD card into the card slot.
```

- Upload two files to the ESP32-S3-GEEK:



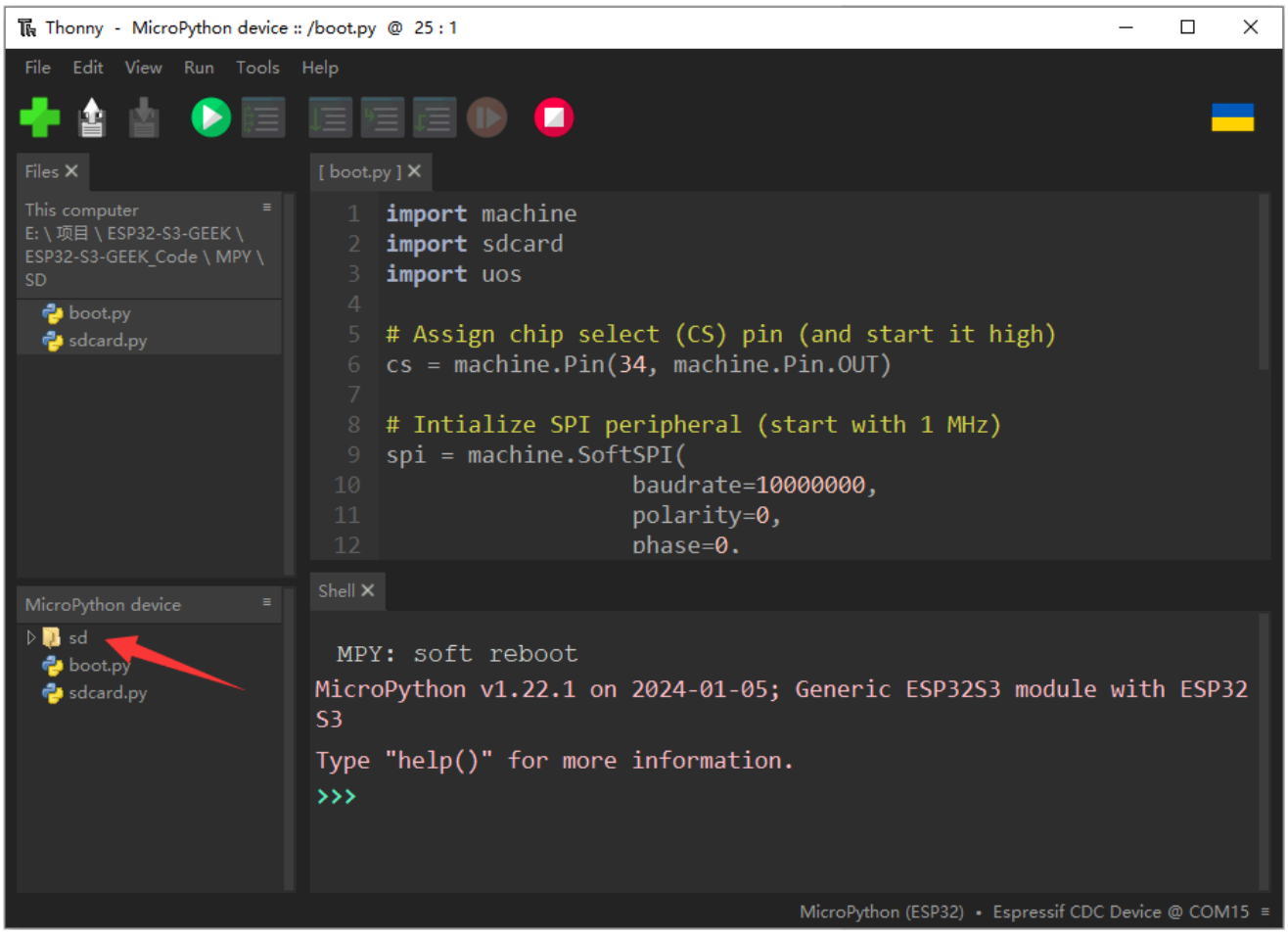
(/wiki/File:ESP32-S3-GEEK-MPY_12.png)

- After uploading, click on "STOP".



(/wiki/File:ESP32-S3-GEEK-MPY_13.png)

- You can see the SD file folder in the ESP32-S3-GEEK:



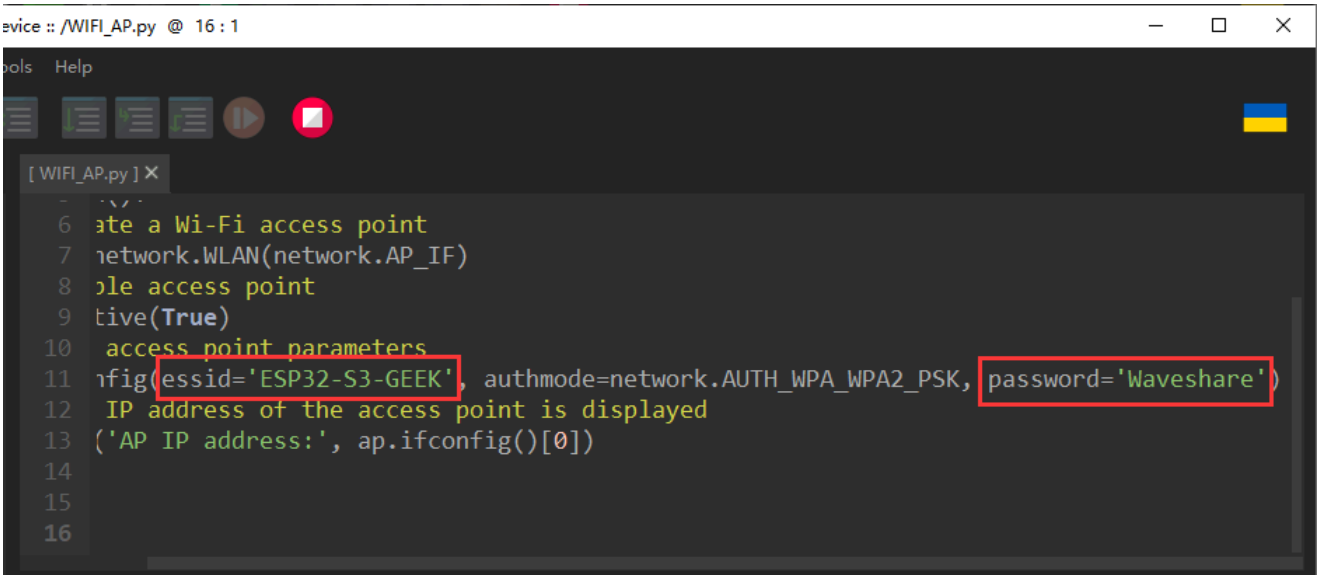
(/wiki/File:ESP32-S3-GEEK-MPY_14.png)

WIFI

01-WIFI_AP

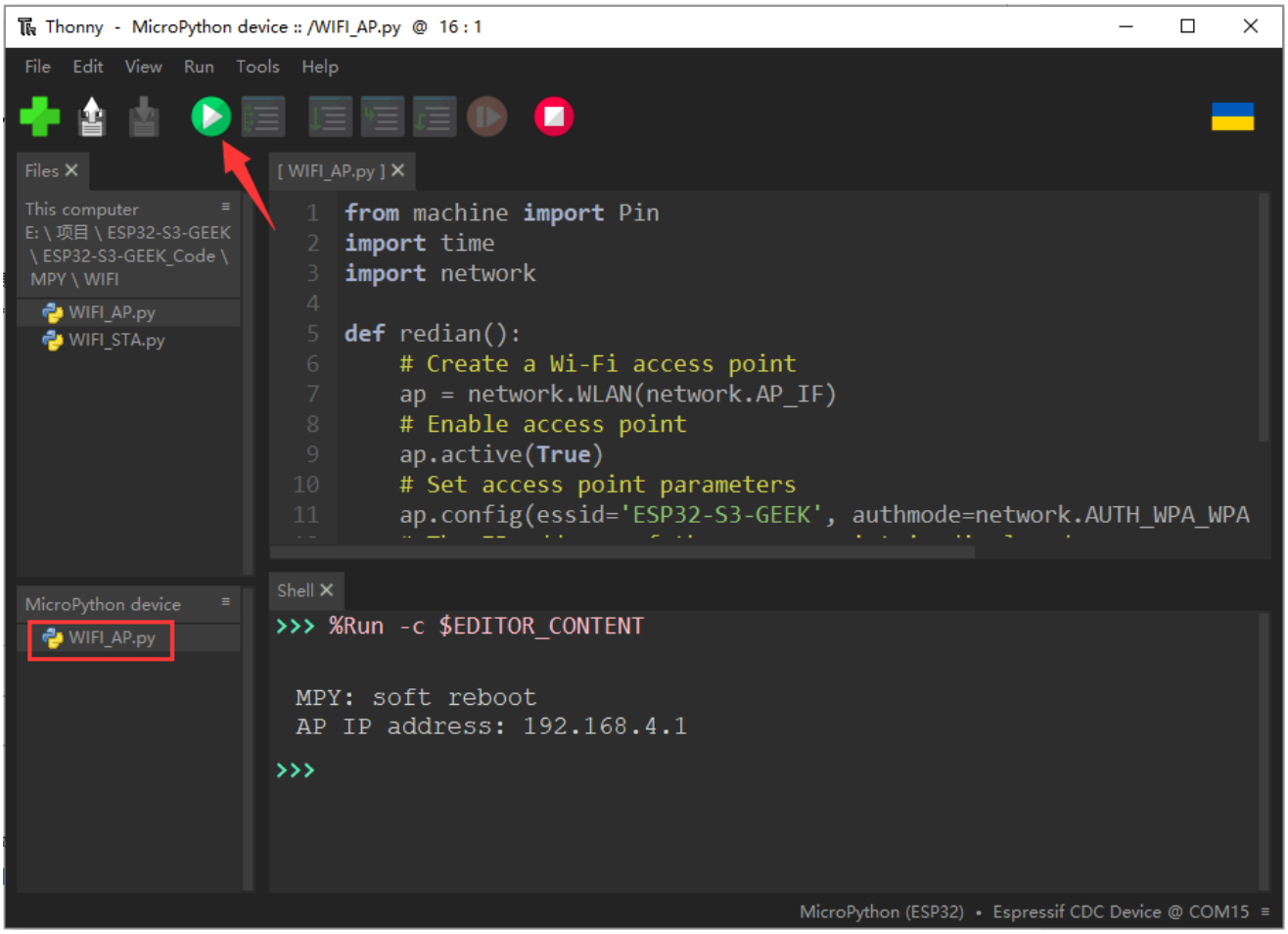
```
This demo turns on the AP mode of WIFI using the ESP32-S3-GEEK, and the PC can connect to its WIFI.
```

- essid is the AP name (ESP32-S3-GEEK) created by ESP32-S3-GEEK, the password (Waveshare) is the password used to connect to the AP.



(/wiki/File:ESP32-S3-GEEK-MPY_27.png)

- Upload the demo to the ESP32-S3-GEEK and click to run.



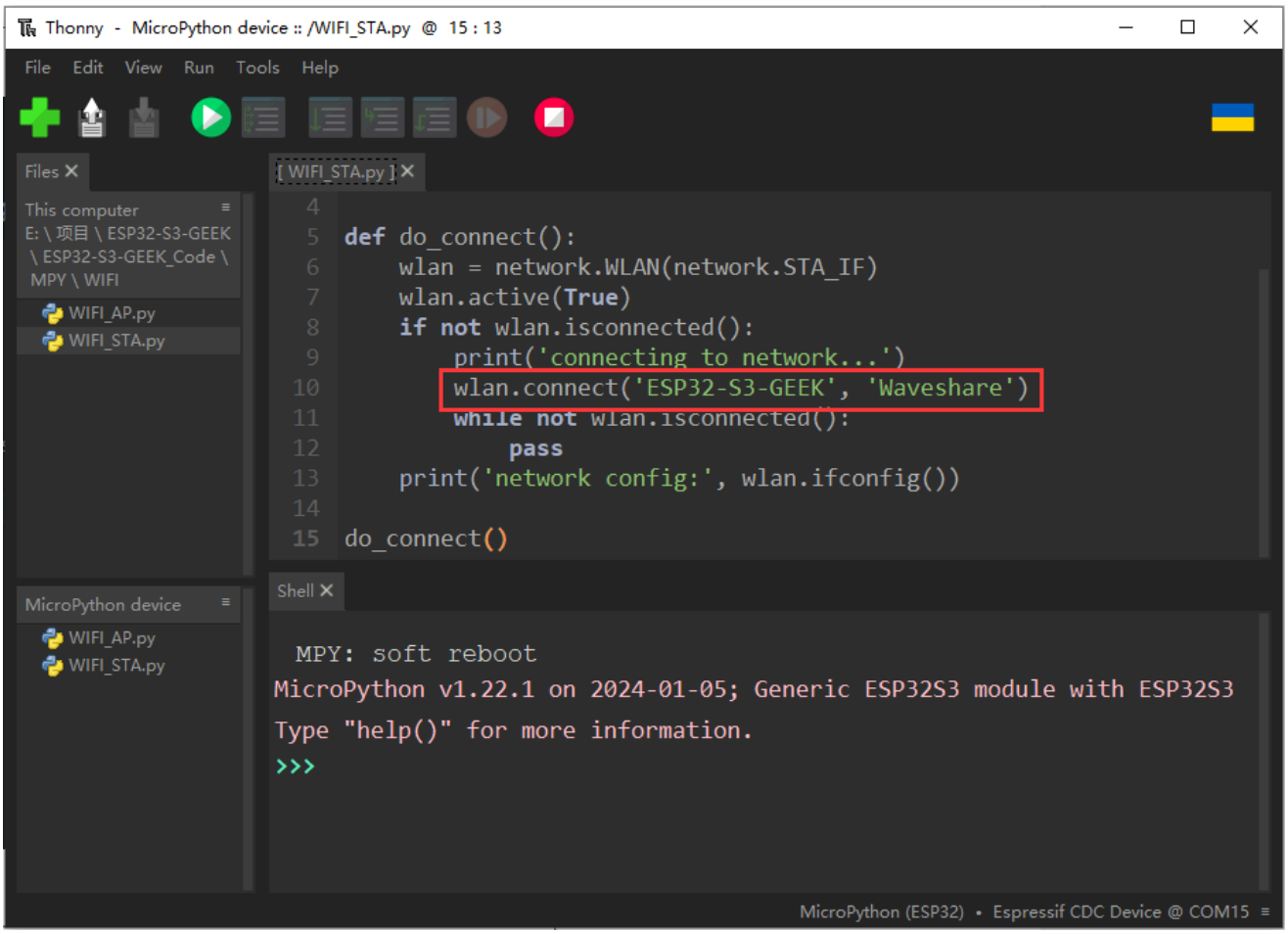
(/wiki/File:ESP32-S3-GEEK-MPY_15.png)

- Open the WIFI on the PC to connect, input the password: Waveshare.

02-WIFI_STA

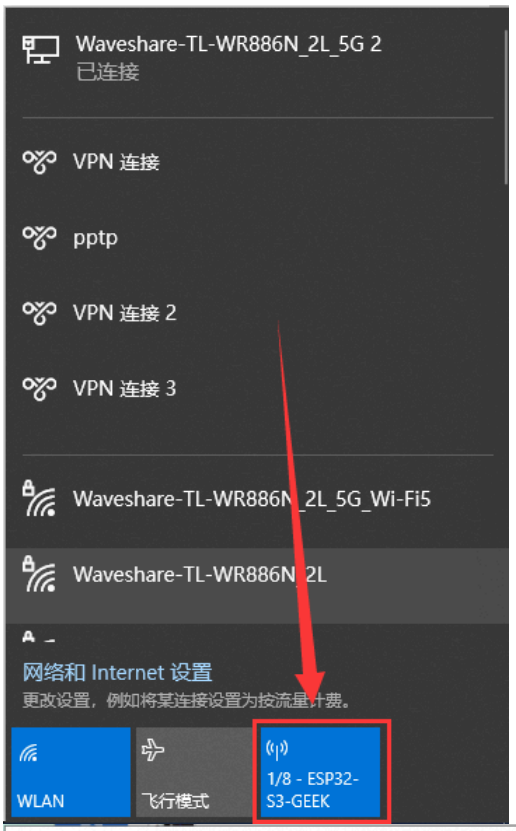
```
This demo can use ESP32-S3-GEEK to open the STA mode of WIFI, and it can connect to other WIFI or hotspot opened on the PC.
```

- In the STA mode, the ESP32-S3-GEEK connects to the WIFI with ssid of ESP32-S3-GEEK and password of Waveshare.

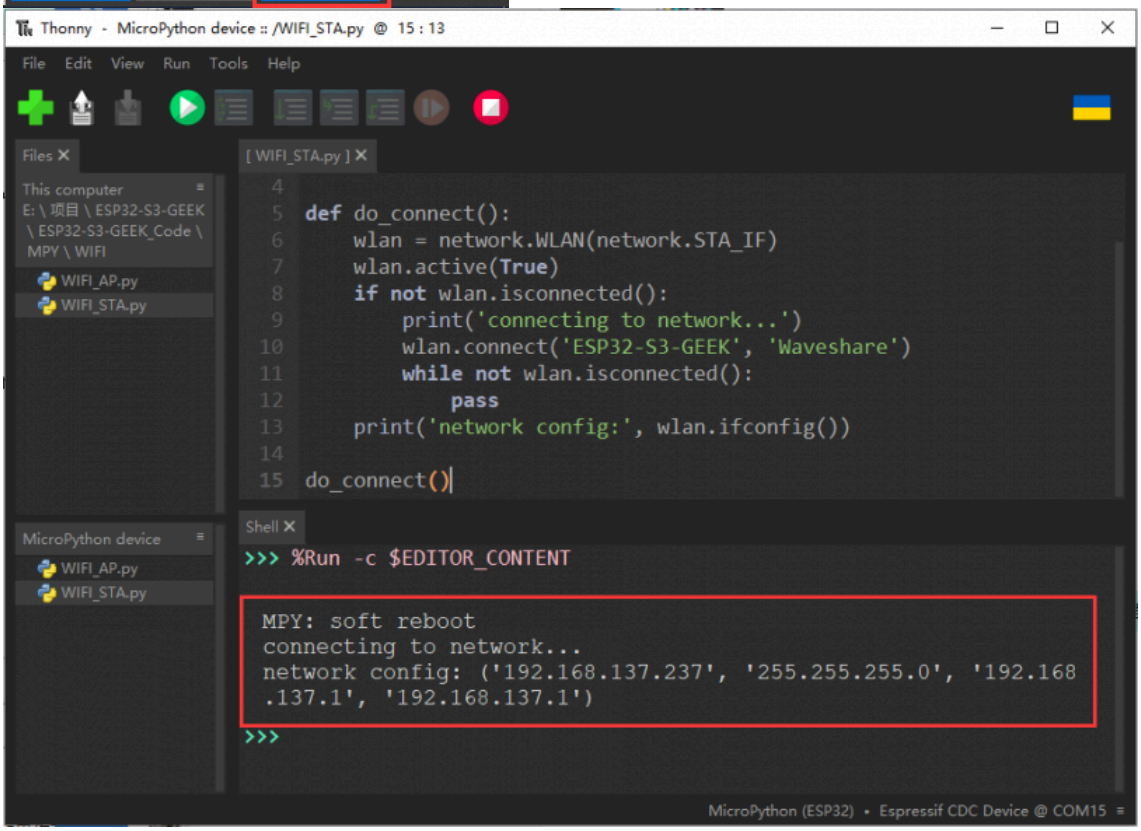


(/wiki/File:ESP32-S3-GEEK-MPY_18.png)

- PC opens a hotspot with ssid "ESP32-S3-GEEK" and password "Waveshare" (The WiFi network connected by ESP32-S3-GEEK needs to be on the 2.4GHz band. If there is no 2.4GHz band available, you can use the PC to open a hotspot, selecting the network band as "any available frequency"). Remember to modify the ssid and password to the desired WiFi network name and password. Upload the demo to ESP32-S3-GEEK and then click to run.



(/wiki/File:ESP32-S3-GEEK-MPY_19.png)

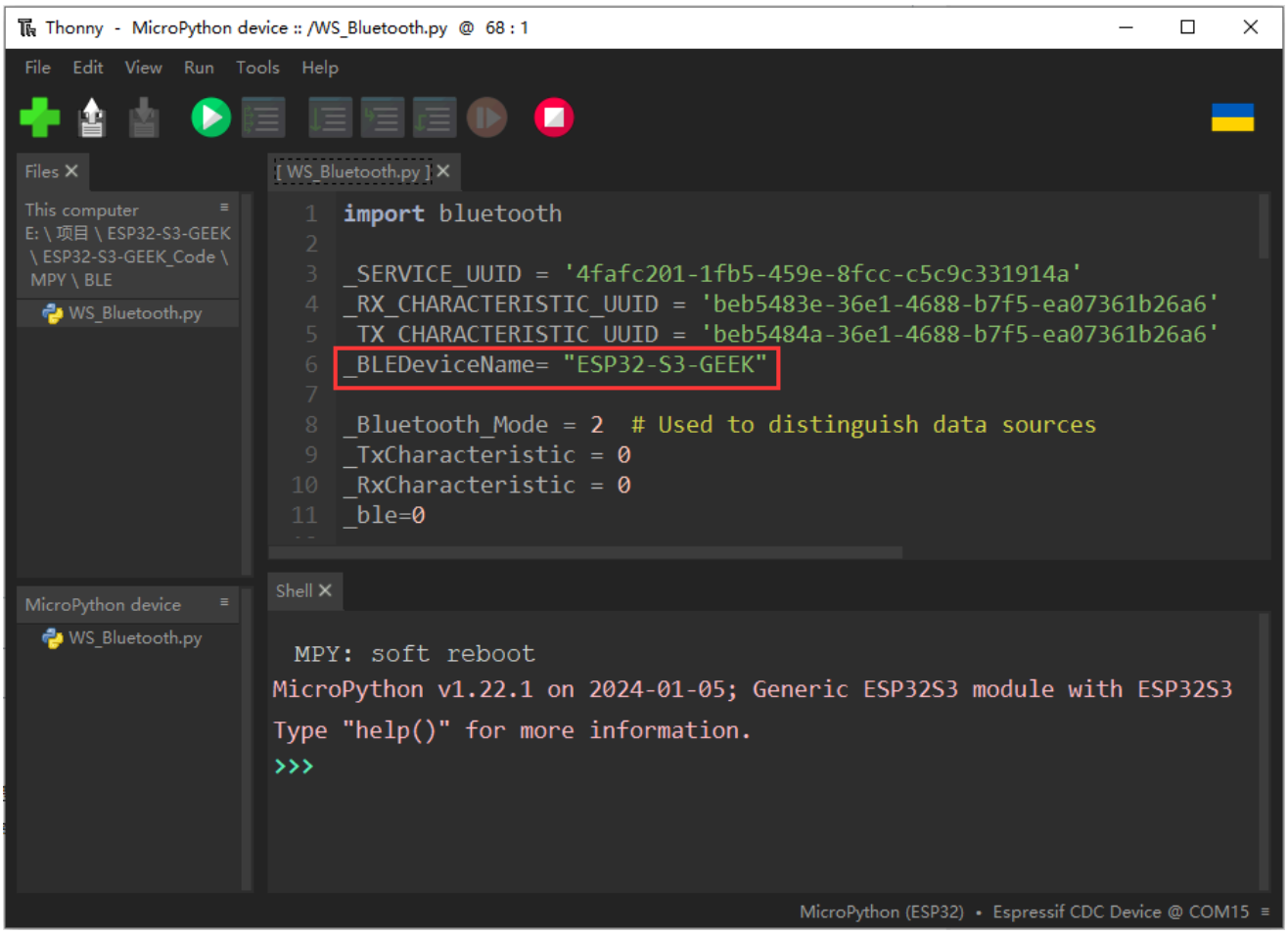


(/wiki/File:ESP32-S3-GEEK-MPY_20.png)

BLE

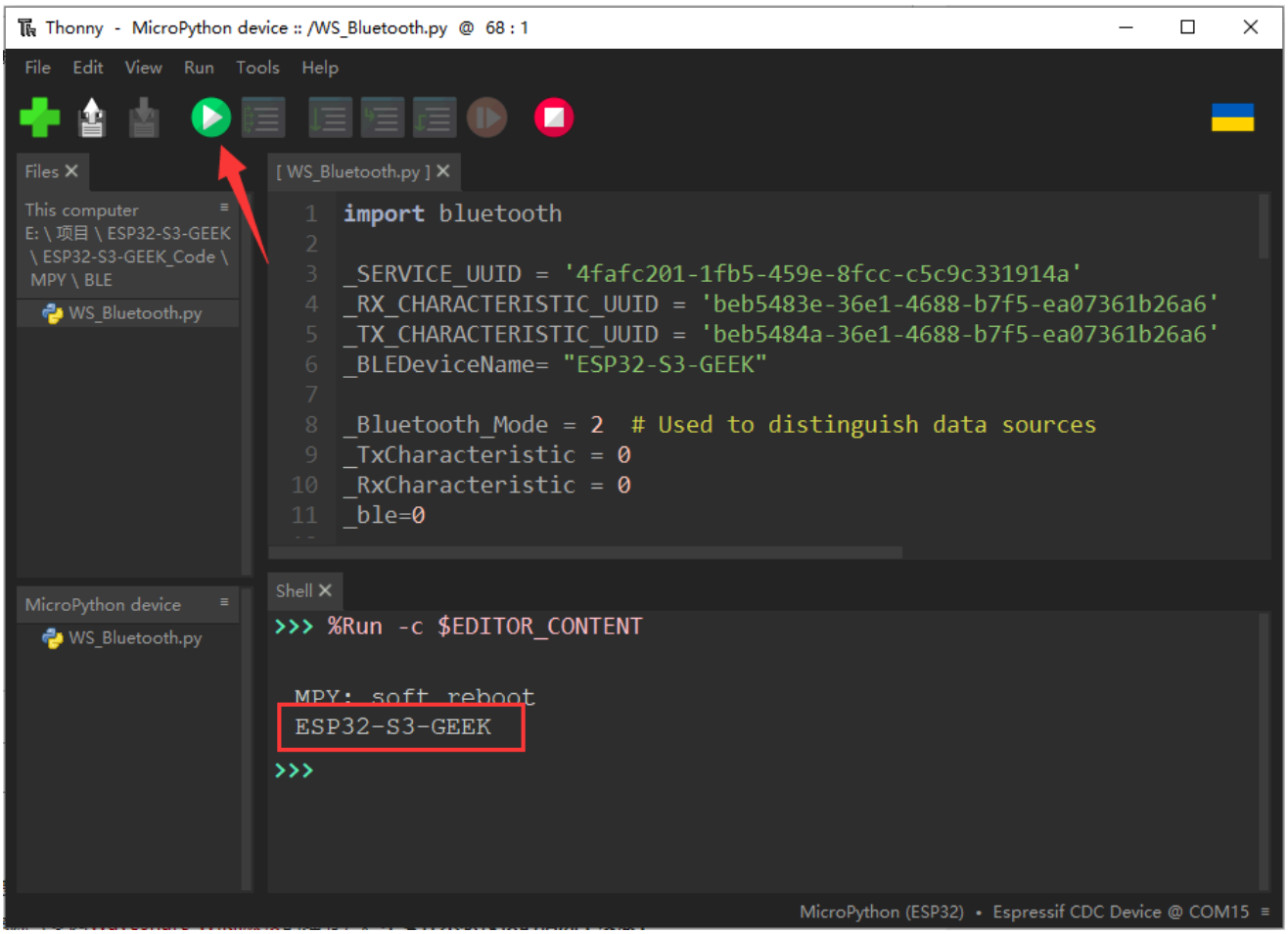
```
This demo opens the Bluetooth of the ESP32-S3-GEEK and uses the Bluetooth and the Bluetooth debugger on the phone to communicate.
```

- "ESP32-S3-GEEK" is the Bluetooth name.



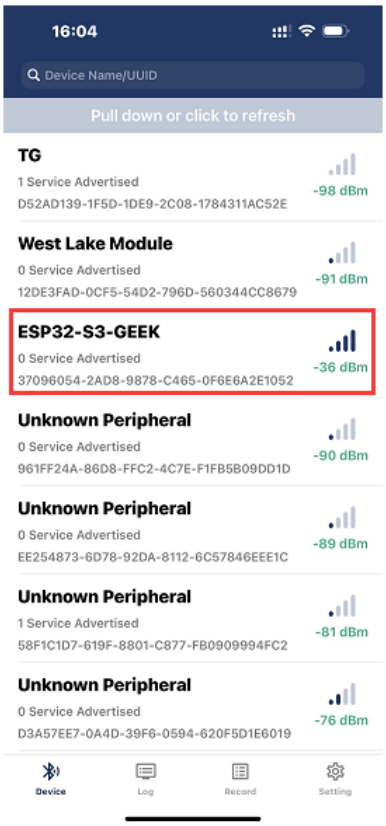
(/wiki/File:ESP32-S3-GEEK-MPY_21.png)

- Click the Run button to successfully open Bluetooth and the Bluetooth name will be printed.

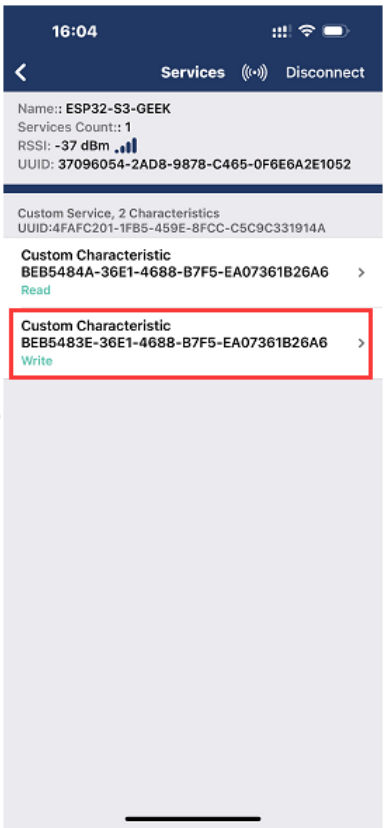


(/wiki/File:ESP32-S3-GEEK-MPY_22.png)

- Use your phone's Bluetooth debugging assistant to connect.

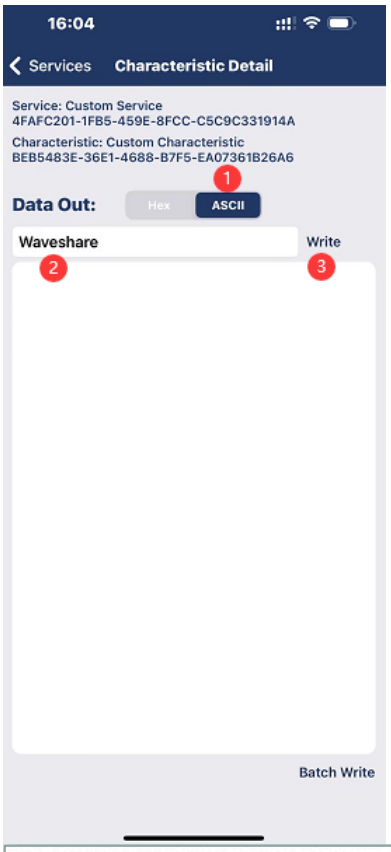


(/wiki/File:ESP32-S3-GEEK-MPY_24.png)

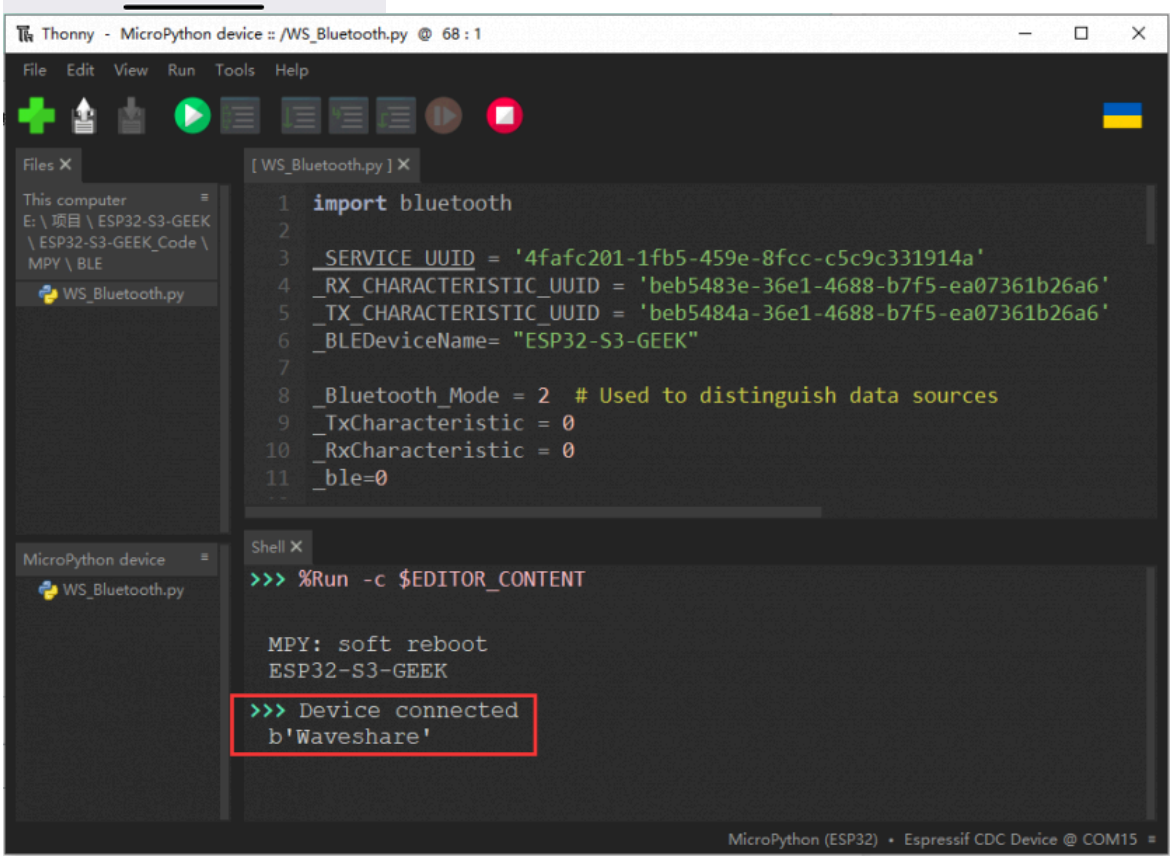


(/wiki/File:ESP32-S3-GEEK-MPY_25.png)

- Use your phone's Bluetooth debugging assistant to send a message, ESP32 receives the message and prints it out.



(/wiki/File:ESP32-S3-GEEK-MPY_26.png)



(/wiki/File:ESP32-S3-

GEEK-MPY_23.png)

UART

```
This demo can use the ESP32-S3-GEEK to open UART0, and open the serial port assistant for UART communication.
```

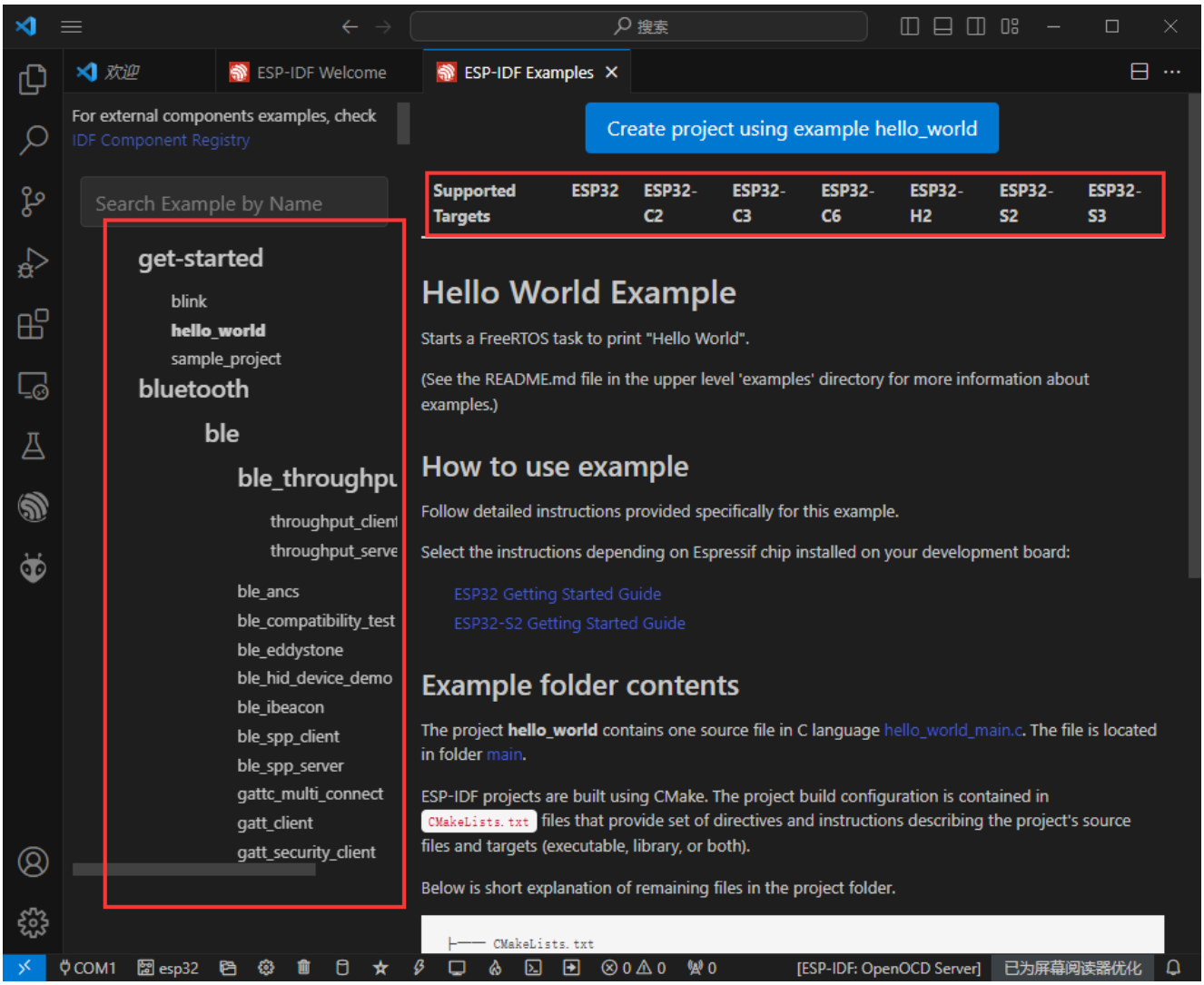
ESP-IDF Sample Demo

> Note: Before using ESP-IDF examples, ensure that the ESP-IDF environment and download settings are correctly configured. For specific instructions, refer to #ESP-IDF.

ESP-IDF Official Demo

```
Can be programmed in ESP-IDF using the official demos, only to provide the official routines to find the use of the method, the specific development needs to be the user's learning and programming.
```

The use of the official demo can be found in #Official Demo Usage, we need to pay attention to the use of the device COM port, chip selection, and if burning method is correct.

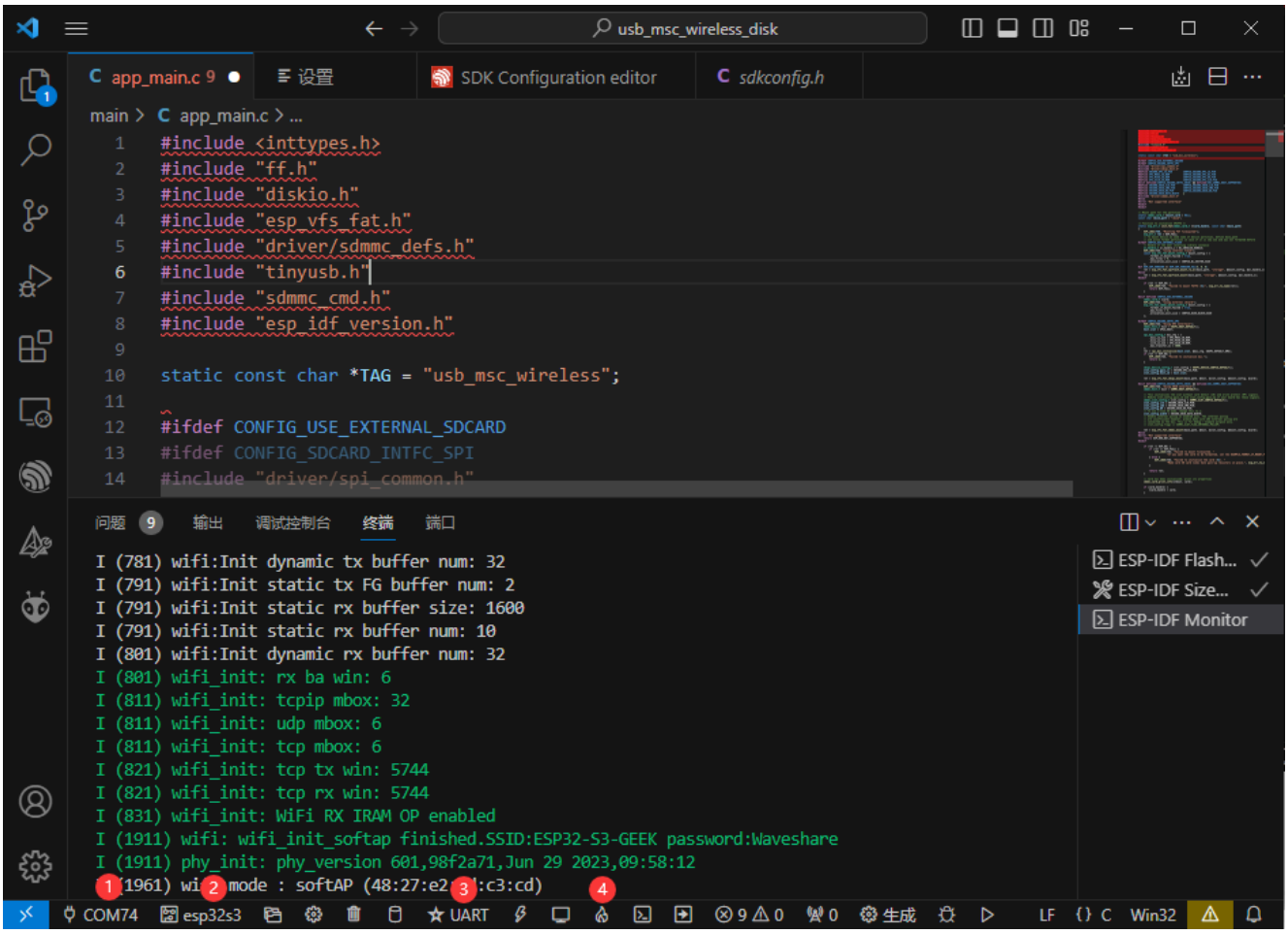


(/wiki/File:ESP32-S3-Geek_Demo_Official.png)

Wireless_USB_flash_drive

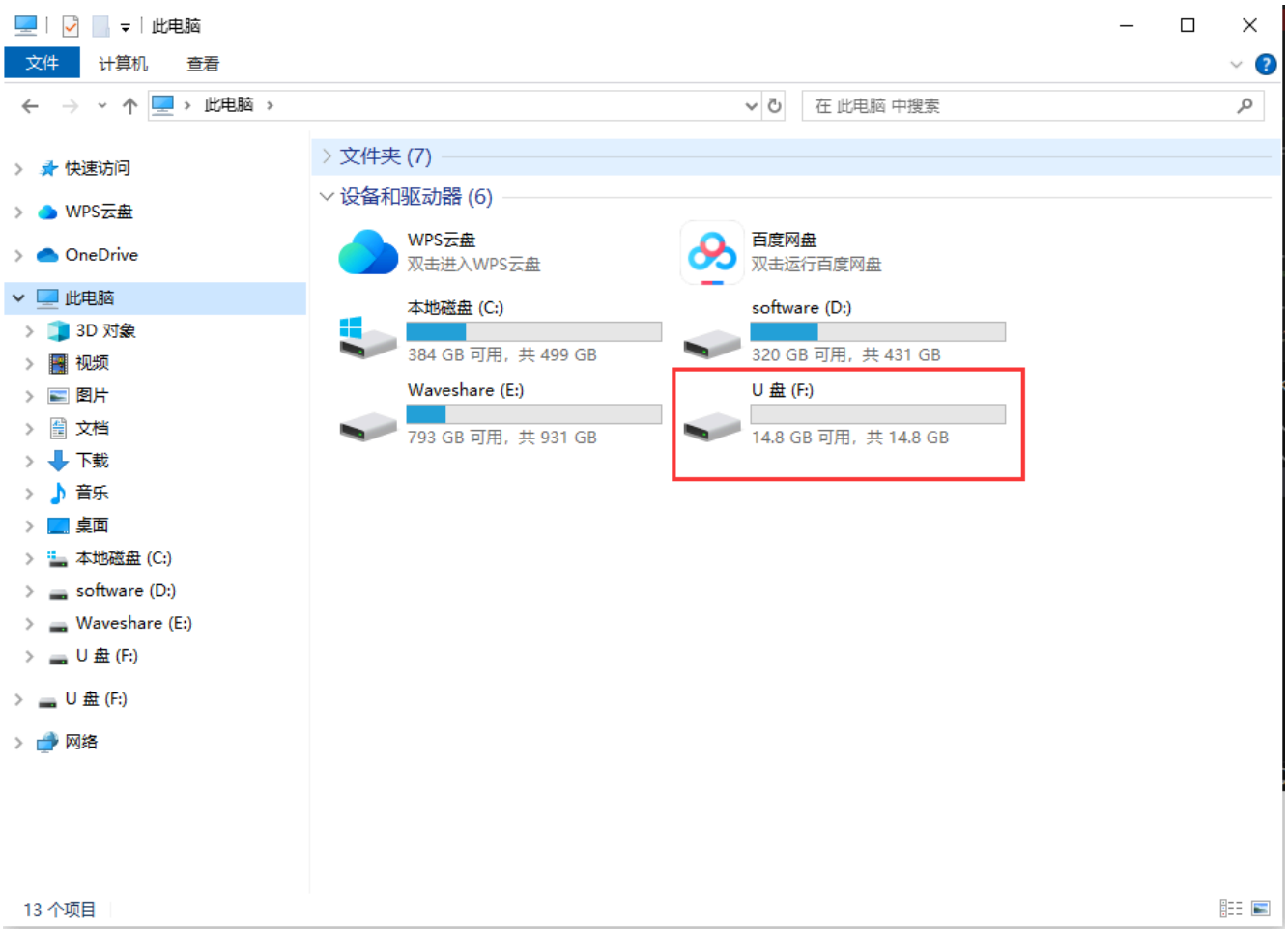
```
This demo allows ESP32-S3-GEEK to function as a USB flash driver with wireless access. When combined with an SD card, it transfo
rms into a high-capacity wireless storage unit. Additionally, it enables connecting to ESP32-S3-GEEK's hotspot to engage in HTTP
file server operations for both uploads and downloads, significantly enhancing user convenience.
```

- Before programming the demo, please insert the SD card into the SD card slot, and check whether the programming chip is esp32s3, and the COM port is correct or not. The programming method selects UART.



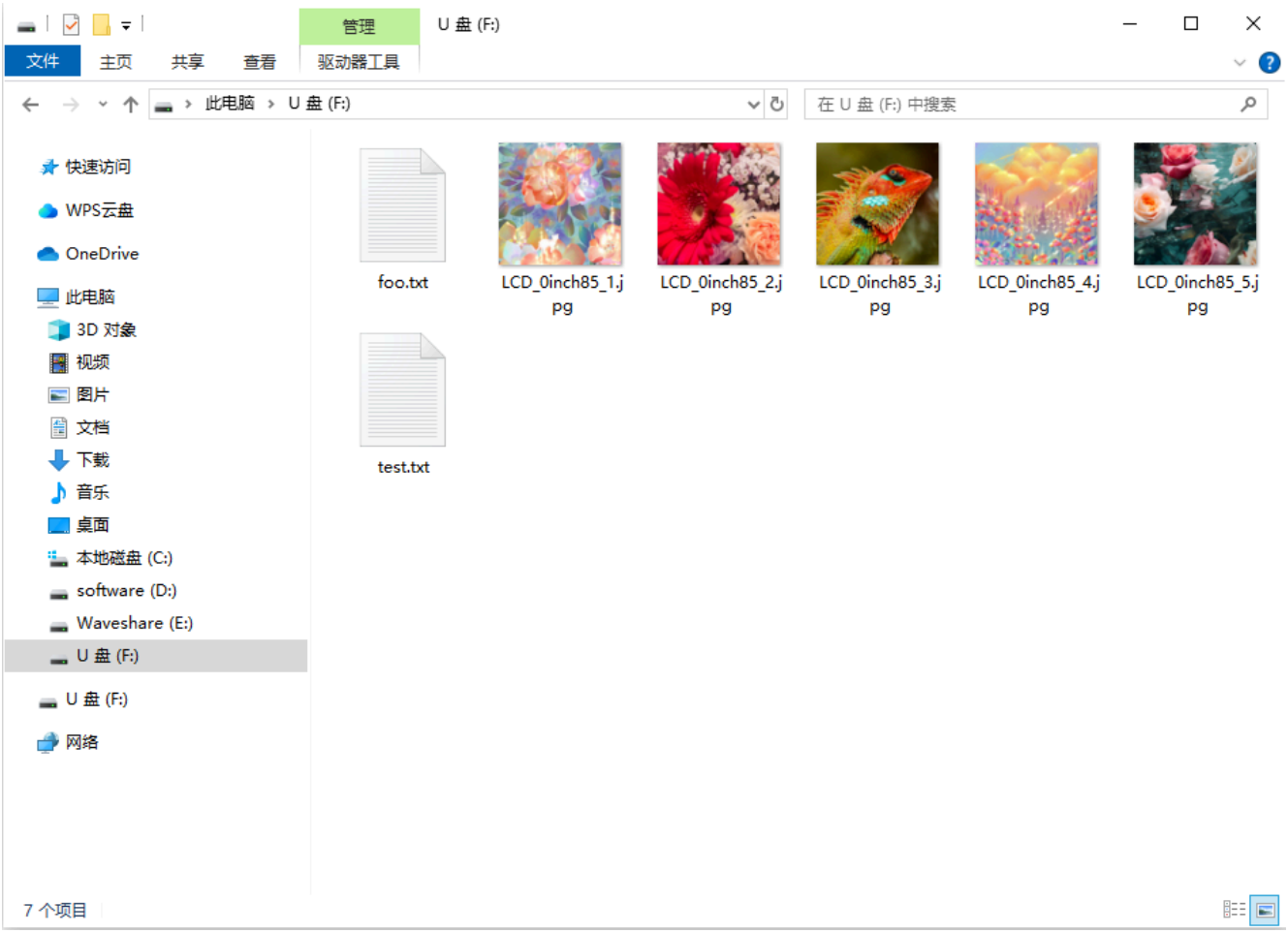
(/wiki/File:ESP32-S3_wireless_disk.png)

- After programming, unplug and plug the ESP32-S3-GEEK again, and you can see a new USB flash driver.



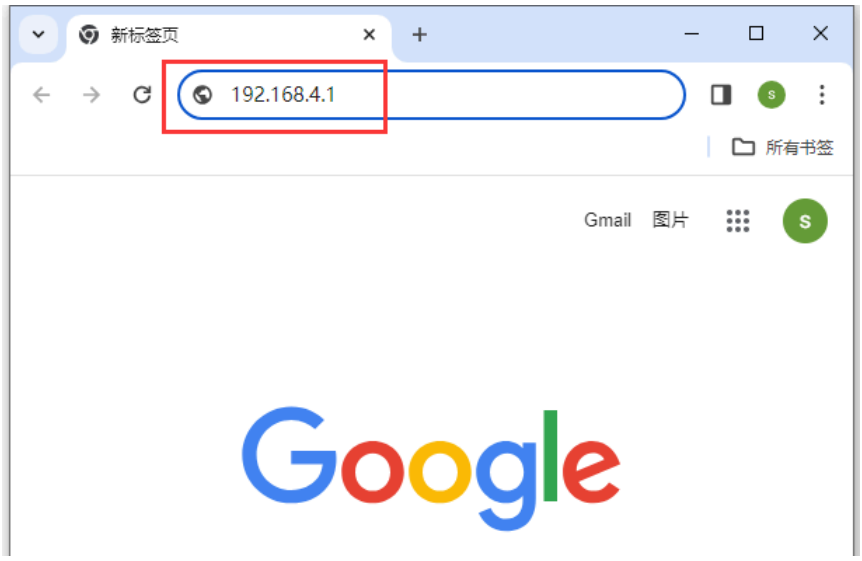
(/wiki/File:ESP32-S3_wireless_disk02.png)

- You can open the USB drive to browse through the files on the SD card and perform various operations such as adding, deleting, modifying, and checking files.



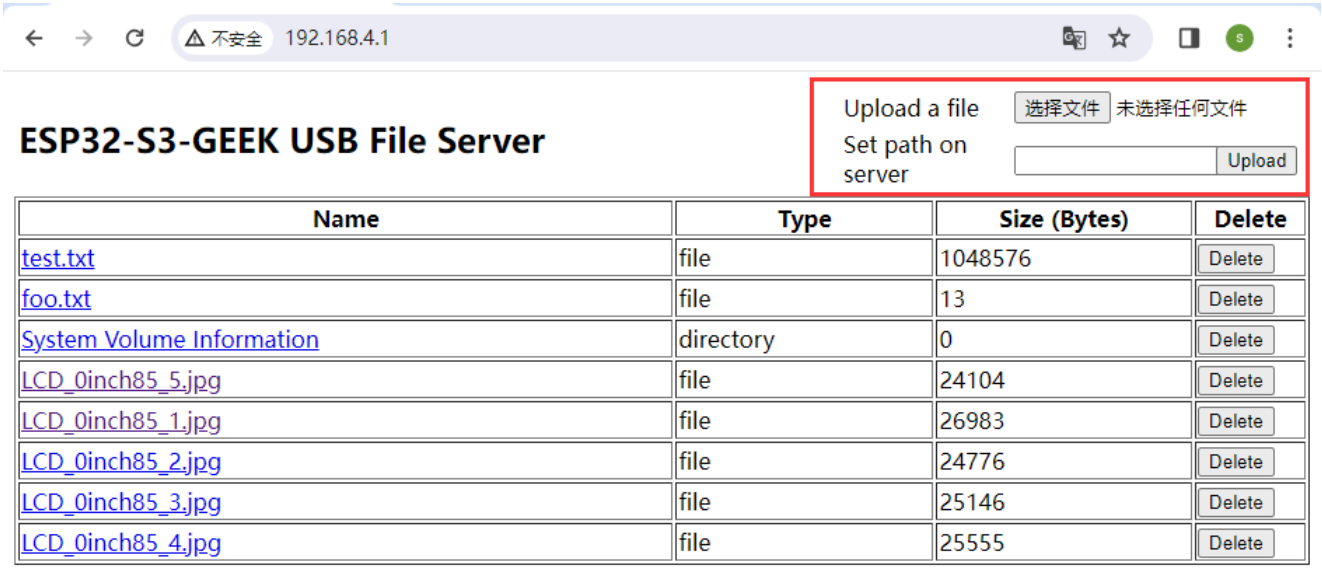
(/wiki/File:ESP32-S3_wireless_disk03.png)

- Enter the PC's WIFI setting, connect to the ESP32-S3-GEEK's AP, and input the password of "Waveshare".
- After connecting successfully, open the browser, and log in to IP: 192.168.4.1.



(/wiki/File:ESP32-S3-GEEK_Demo03.png)

- After login successfully, you can upload and download files wirelessly.



(/wiki/File:ESP32-S3_wireless_disk05.png)

Resource

Schematic

- Schematic (https://files.waveshare.com/wiki/ESP32-S3-GEEK/ESP32-S3-GEEK-Schematic1.pdf)

Demo

- Demo (https://files.waveshare.com/wiki/ESP32-S3-GEEK/ESP32-S3-GEEK_Code.zip)

Development Software

MicroPython

- ESP32-S3-GEEK-MPY-Flash (https://files.waveshare.com/wiki/ESP32-S3-GEEK/ESP32-S3-GEEK-MPY-Flash.zip)
- MicroPython Official Firmware (https://micropython.org/download/ESP32_GENERIC_S3/)

Image Modeling

- Zimo221.7z (https://files.waveshare.com/upload/c/c6/Zimo221.7z)
- Image2Lcd.7z (https://files.waveshare.com/upload/3/36/Image2Lcd.7z)
- Image_Extraction (https://www.waveshare.com/wiki/Image_Extraction)
- Ink Screen Font Library Tutorial (https://www.waveshare.com/wiki/Ink_Screen_Font_Library_Tutorial)

Debugging Tool

- sscom serial port assistant (https://files.waveshare.com/upload/b/b3/Sscom5.13.1.zip)
- NetAssist tool (https://files.waveshare.com/upload/9/95/Netassist.zip)
- IOS Bluetooth assistant (https://apps.apple.com/us/app/bluetoothassistant/id1536579599)
- Android Bluetooth debugging tool (https://files.waveshare.com/wiki/ESP32-S3-GEEK/Android%E8%93%9D%E7%89%99%E8%B0%83%E8%AF%95%E5%8A%A9%E6%89%8B.zip)

- Android TCP debugging tool (https://files.waveshare.com/wiki/ESP32-S3-GEEK/Tcp%E8%B0%83%E8%AF%95%E5%8A%A9%E6%89%8B.zip)
- Espressif APP (https://www.espressif.com.cn/en/support/download/apps?keys=&field_type_tid%5B%5D=842)

Datasheet

- ESP32-S3 Datasheet (https://files.waveshare.com/upload/f/f7/Esp32-s3_datasheet_en_%281%29.pdf)
- ESP32-S3 Technical Reference Manual (https://files.waveshare.com/upload/9/9e/Esp32-s3_technical_reference_manual_en_%281%29.pdf)
- ESP32-S3-WROOM-1 Datasheet (https://files.waveshare.com/upload/8/87/Esp32-s3-wroom-1_wroom-1u_datasheet_en.pdf)

ESP32 Official Document

- ESP32-Arduino Official Document (https://docs.espressif.com/projects/arduino-esp32/en/latest/index.html)
- ESP-IDF Official Document (https://docs.espressif.com/projects/esp-idf/en/release-v5.0/esp32s3/index.html)

MicroPython Official Document

- MicroPython Official Document (https://docs.micropython.org/en/latest/)

FAQ

Question:ESP32-S3-GEEK programming failed?

Answer:

① During programming, long-press the Boot button, insert the USB, and then release the button to enter the ESP32-S3-GEEK Download mode for the programming process. After programming, reconnecting the USB will enter the SPI_FAST_FLASH_BOOT mode, run the demo.
② Try turning off the PC's Bluetooth switch and then directly burn and run (indicating the conflict between the Bluetooth driver and the ESP32-S3-GEEK's COM driver results in the programming failure).

Question:ESP32-S3-GEEK successfully connected to Waveshare Cloud, why can't it send or receive MQTT messages?

Answer:

Please check if the Pub Topic and Sub Topic are correctly filled in, and verify if the device is online in the Waveshare Cloud. Both factors could impact the ability to send or receive MQTT messages.

Question:When the SD card is inserted into the ESP32-S3-GEEK, can the computer directly function as a USB flash driver? After inserting the SD card, can you directly view its files using Windows File Explorer when connected to the computer?

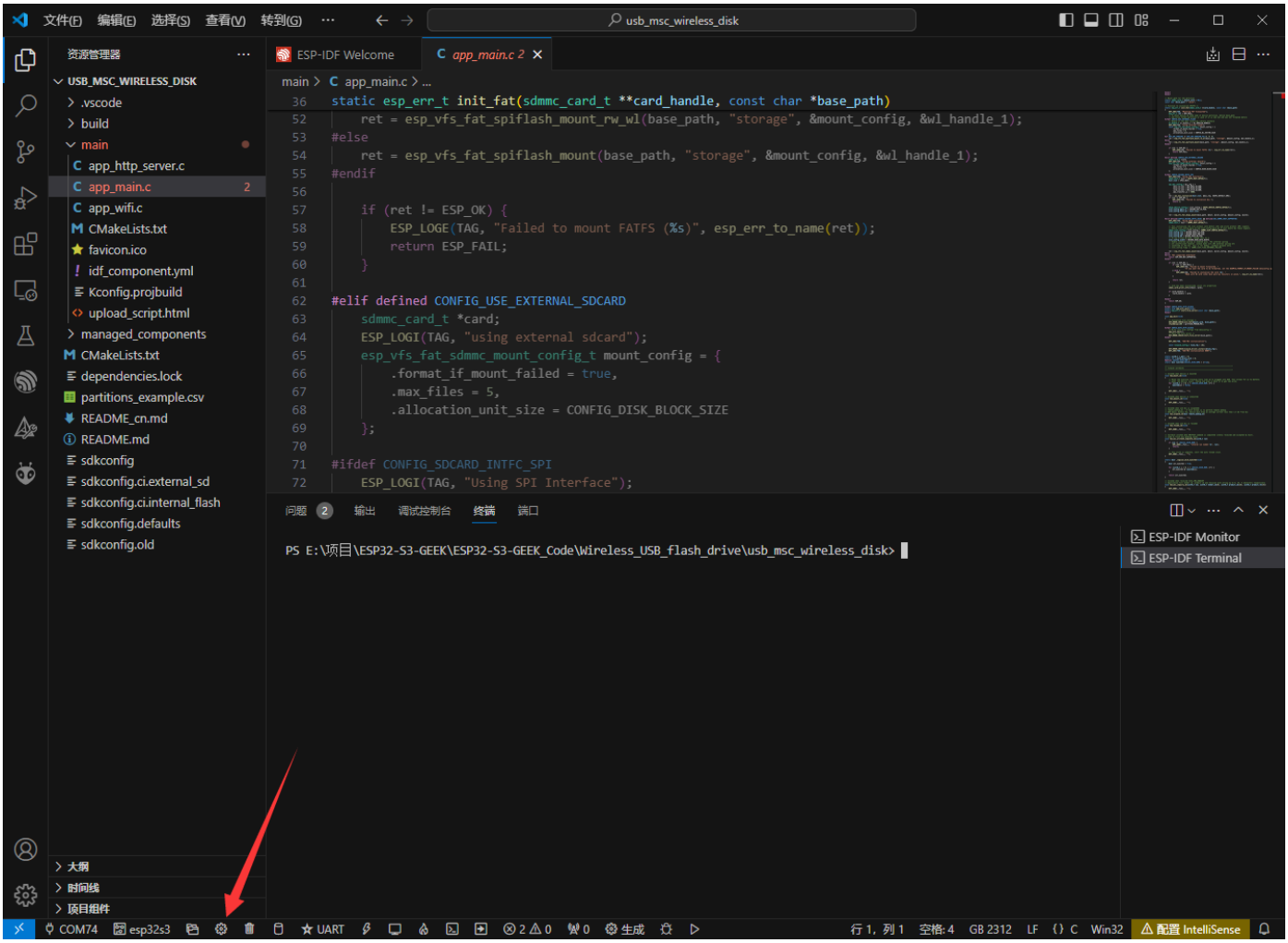
Answer:

Yes, you need to flash #Wireless_USB_flash_drive, in the SD card image, SPIFS only supports up to 32GB.

Question:ESP32-S3-GEEK burned the Wireless_USB_flash_drive demo, and the 16G MicroSD card is inserted into the SD card slot, but the display memory is only 1.4M?

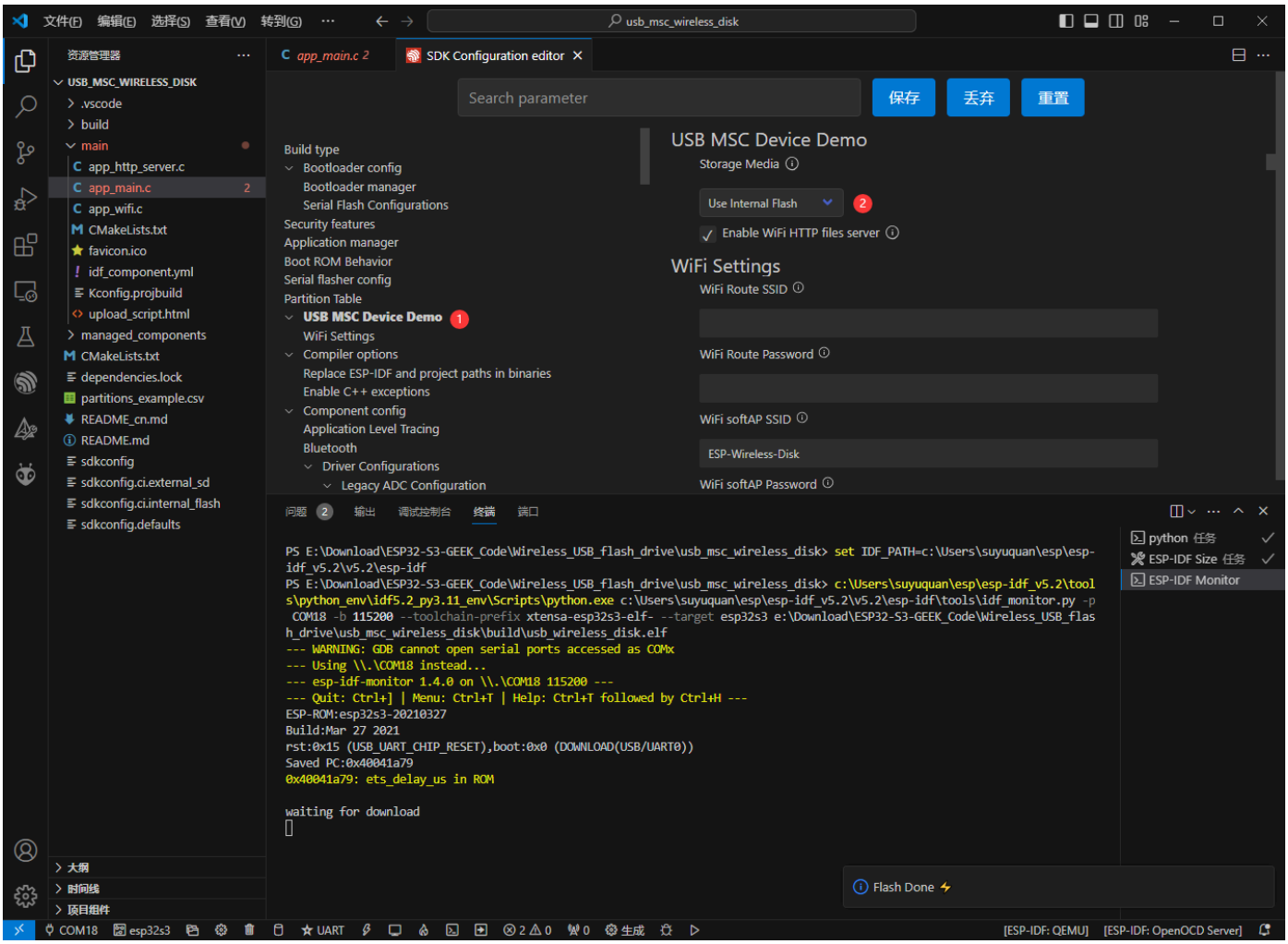
Answer:

- 1. Click Settings in the lower left corner.

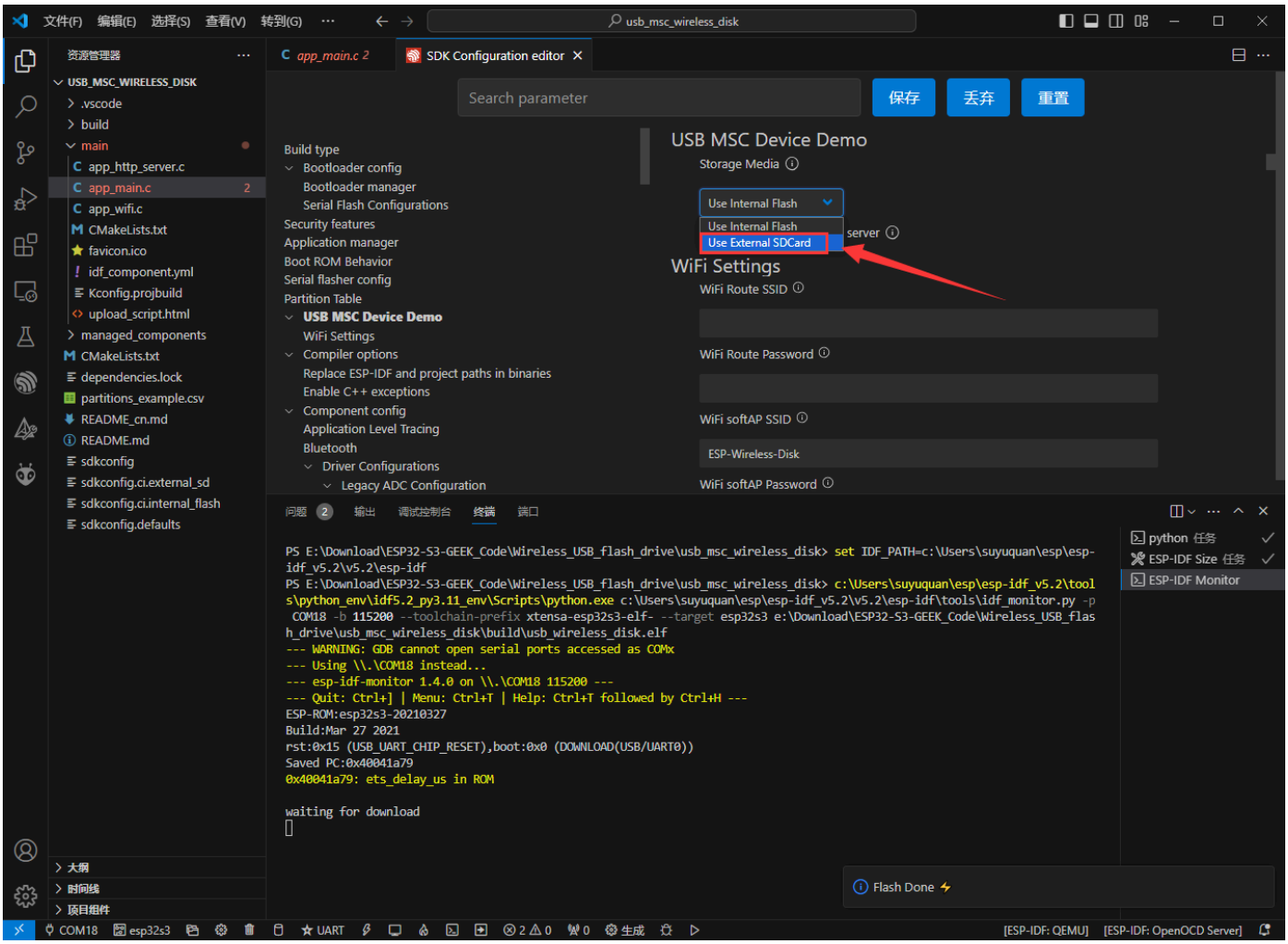


(/wiki/File:ESP32-S3-GEEK-FAQ1.png)

- 2. Modified "Use Internal Flash" to "Use External SDCard".

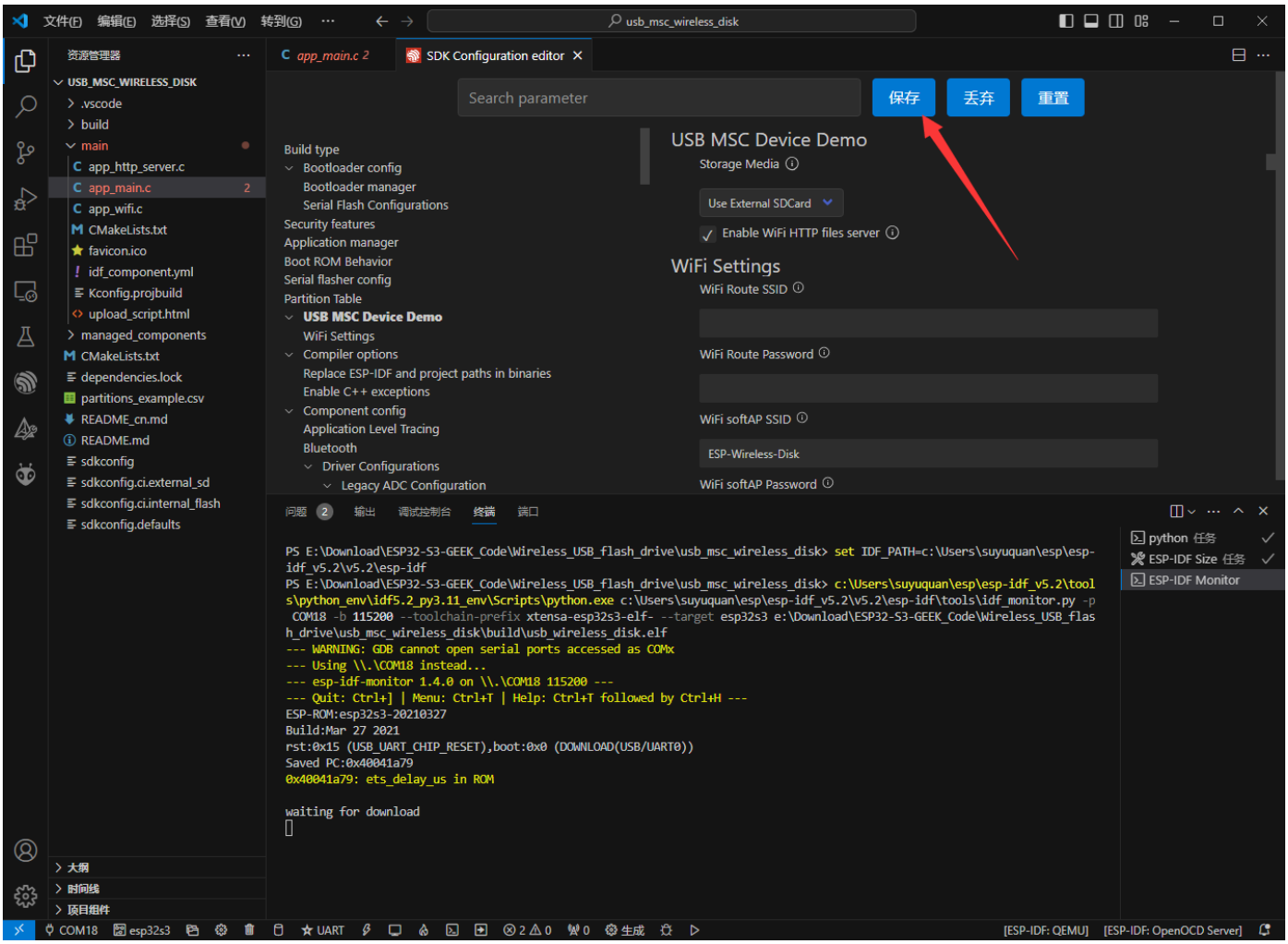


(/wiki/File:ESP32-S3-GEEK-FAQ2.png)



(/wiki/File:ESP32-S3-GEEK-FAQ3.png)

- 3. Save the settings and burn again.



(/wiki/File:ESP32-S3-GEEK-FAQ4.png)

Support

Technical Support

If you need technical support or have any feedback/review, please click the **Submit Now** button to submit a ticket, Our support team will check and reply to you within 1 to 2 working days. Please be patient as we make every effort to help you to resolve the issue.
Working Time: 9 AM - 6 PM GMT+8 (Monday to Friday)

Submit Now (https://service.waveshare.com/)

Retrieved from "https://www.waveshare.com/w/index.php?title=ESP32-S3-GEEK&oldid=87326 (https://www.waveshare.com/w/index.php?title=ESP32-S3-GEEK&oldid=87326)"