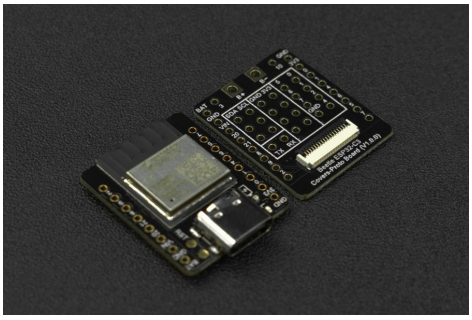
search

Controller
DFR0182 Wirless GamePad V2.0
DFR0100 DFRduino Beginner Kit For Arduino V3
DFR0267 Bluno
DFR0282 Beetle
DFR0283 Dreamer Maple V1.0
DFR0296 Bluno Nano
DFR0302 MiniQ 2WD Plus
DFR0304 BLE Wireless Gamepad V2
DFR0305 RoMeo BLE
DFR0351 Romeo BLE mini V2.0
DFR0306 Bluno Mega 1280
DFR0321 Wido-WIFI IoT Node
DFR0323 Bluno Mega 2560
DFR0329 Bluno M3
DFR0339 Bluno Beetle
DFR0343 UHex Low-power Controller
DFR0355 SIM808 with Leonardo mainboard
DFR0392 DFRduino M0 Mainboard Arduino Compatible
DFR0398 Romeo BLE Quad Robot Controller
DFR0416 Bluno M0 Mainboard
DFR0575 Beetle ESP32
DFR0133 X-Board
DFR0162 X-Board V2
DFR0428 3.5 inches TFT Touchscreen for Raspberry Pi
DFR0494 Raspberry Pi UPS HAT
DFR0514 DFR0603 IIC 16X2 RGB LCD KeyPad HAT V1.0
DFR0524 5.5 HDMI OLED-Display with Capacitive Touchscreen V2.0

SKU:DFR0868

1.



Introduction

Beetle ESP32-C3, mainly intended for IoT applications, is a controller based on ESP32-C3 RISC-V 32bit single-core processor.
On a coin-size board of 25*20.5 mm, there are up to 13 IO ports broken out, so you don't have to worry about running out of IO ports when making projects. Meanwhile, li-ion battery charging management function is integrated on the board which allows to directly connect li-ion battery without extra modules, while ensuring the application size and safety. The equipped expansion board for Beetle ESP32-C3 brings out more power sources without increasing product volume, more convenient to solder. Besides, the onboard easy-to-connect GDI saves the trouble of wiring when using a screen.
Beetle ESP32-C3 supports WiFi and Bluetooth 5(LE) dual-mode communication that reduces the difficulty of networking, and also both Bluetooth Mesh protocol and Espressif WiFi Mesh are supported for more stable communication and a larger coverage area.
Detailed tutorials for Beetle ESP32-C3 are provided for users to use the controller's WiFi function, such as connecting to IoT platforms like Aliyun, or IFTTT, etc. Or users can use various sensors and actuators from DFRobot to build up IoT systems.
Beetle ESP32-C3 can be programmed by Arduino IDE, ESP-IDF, MicroPython, C and Python are both supported.

2. Features

- Ultra-small size, the size is only 25×20.5mm(0.98×0.81 inch)
- Onboard lithium battery charging management, charging and discharging is safer
- The matching bottom plate makes it more convenient to make projects and use the screen
- RISC-V 32-bit core
- Supports Wi-Fi and Bluetooth 5 (LE) dual-mode communication

3. Specification

DFR0550 5" TFT-Display with Touchscreen V1.0

DFR0591 raspberry pi e-ink display module V1.0

DFR0592 DC Motor Driver HAT

DFR0604 I O Expansion HAT for Pi zero V1.0

DFR0566 IO Expansion HAT for Raspberry Pi

DFR0528 UPS HAT for Raspberry Pi Zero

DFR0331 Romeo for Edison Controller

DFR0453 DFRobot CurieNano - A mini Genuino Arduino 101 Board

TEL0110 CurieCore intel® Curie Neuron Module

DFR0478 FireBeetle ESP32 IOT Microcontroller(V3.0) Supports Wi-Fi & Bluetooth

DFR0483 FireBeetle Covers-Gravity I O Expansion Shield

FireBeetle Covers-24×8 LED Matrix

TEL0121 FireBeetle Covers-LoRa Radio 433MHz

TEL0122 FireBeetle Covers-LoRa Radio 915MHz

TEL0125 FireBeetle Covers LoRa Radio 868MHz

DFR0489 FireBeetle ESP8266 IOT Microcontroller

DFR0492 FireBeetle Board-328P with BLE4.1

DFR0498 FireBeetle Covers-Camera&Audio Media Board

DFR0507 FireBeetle Covers-OLED12864 Display

DFR0508 FireBeetle Covers-DC Motor & Stepper Driver

DFR0511 FireBeetle Covers-ePaper Black&White Display Module

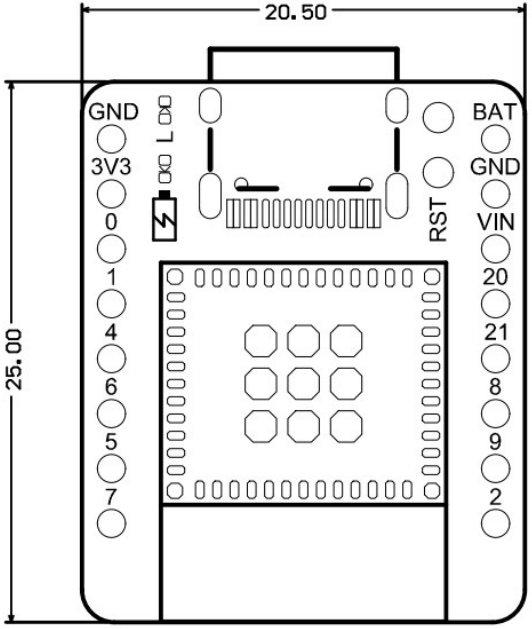
DFR0531 FireBeetle Covers-ePaper Black&White&Red Display Module

DFR0536 Micro bit Gamepad Expansion Board

DFR0548 Micro bit Driver Expansion Board

ROB0148 micro:Maqueen for micro:bit

Basic



parameters

- Operating voltage: 3.3V
- Type-C input voltage: 5V DC
- VIN input voltage: 5V DC
- Working current: 25mA
- Maximum charging current: 400mA
- Working temperature: -40~105°C
- Module size: 25x20.5 mm(0.98*0.81 inch)

Hardware information

- Processor: 32-bit RISC-V single-core processor
- Main frequency: 160 MHz
- SRAM: 400KB
- ROM: 384KB
- Flash: 4MB
- RTC SRAM: 8KB
- Clock: External (32 kHz) crystal, built-in fast RC oscillator clock 17.5 MHz (adjustable), and PLL clock
- USB: USB 2.0 up to 12Mbit/s

WIFI

- WIFI protocol: IEEE 802.11b/g/n
- WIFI bandwidth: 2.4 GHz band supports 20 MHz and 40 MHz bandwidth
- WIFI mode: Station mode, SoftAP mode, SoftAP+Station mode and promiscuous mode
- WIFI frequency: 2.4GHz
- Frame Aggregation: TX/RX A-MPDU, TX/RX A-MSDU

Bluetooth

- Bluetooth protocol: Bluetooth 5, Bluetooth mesh
- Bluetooth frequency: 125 Kbps, 500 Kbps, 1 Mbps, 2 Mbps

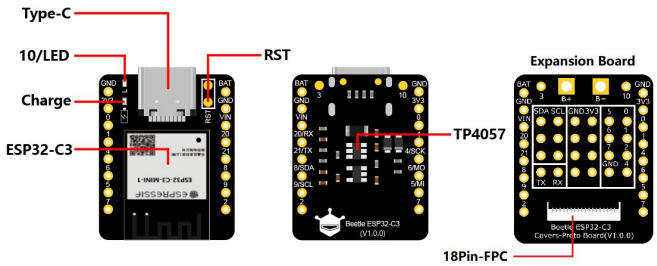
Interface pins

- Digital I/O x13
- LED PWM controller with 6 channels
- SPI x1

- UART x2
- I2C x1
- I2S x1
- Infrared receiver and transmitter: transmit channel x2, receive channel x2, (any pin)
- 2 × 12-bit SAR A/D converters, 6 channels
- DMA controller, 3 receive channels and 3 transmit channels

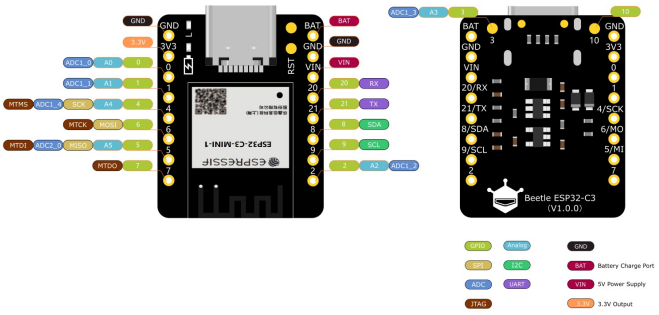
4. Schematic diagram of functional pins

Function Instructions



- Type-C: 5V
- 10/LED: Onboard LED pin
- ESP32-C3 module: the latest ESP32-C3 module launched by Espressif
- RST: reset pin, short contact point trigger reset
- TP4057: TP4057 lithium battery charge management chip
- Charge: charging indicator
  - Off: not plugged in power supply or fully charged
  - On: charging
  - Blinking: battery not connected
- 18Pin-FPC:GDI display interface

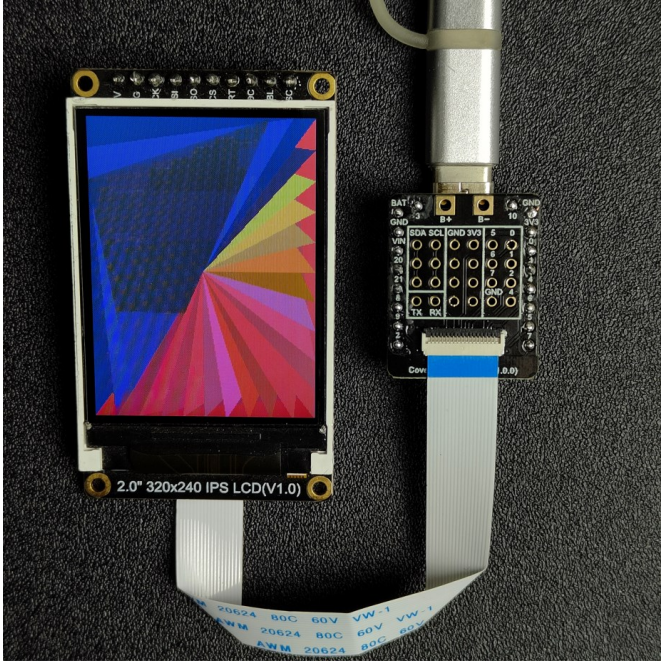
Pin Diagram



Pin Overview

- GPIO: regular pins
- Analog Port: Analog Input Pin
- JTAG: Debug Interface
- ADC: analog-to-digital conversion
- VIN: 5V power input
- BAT: battery access port

GDI display interface



This interface is a dedicated GDI display interface for DFRbot. It is connected with an 18pin-FPC cable, and a single core wire is used to connect the screen, providing you with the easiest way to use the screen. Below is a list of pins used by the GDI interface.

FPC PINS	Beetle ESP32 C3 Pins	Description
VCC	3V3	3.3V
BLK (PWM dimming)	10	Backlight
GND	GND	GND
SCLK	4/SCK	SPI clock
MOSI	6/MOSI	Host output, slave input
MISO	5/MISO	Host input, slave output
DC	1	Data/command
RES	2	Reset
CS	7	TFT Chip Select
SDCS	0	SD card chip select
FCS	NC	Font library
TCS	3	Touch
SCL	22/SCL	I2C clock
SDA	21/SDA	I2C data
INT	NC	INT
BUSY-TE	NC	ripstop pins
X1	NC	custom pin 1
X2	NC	custom pin 2

When using FPC to link the screen, you can configure the corresponding pin numbers according to the GDL demo. Normally, you only need to configure three pins according to different masters. Displays that support GDI:

- 1.54" 240x240 IPS wide viewing angle TFT display
- 1.8" 128x160 IPS TFT LCD Display
- 2.0" 320x240 IPS wide viewing angle TFT display
- 2.8" 320x240 IPS TFT resistive touch display
- 3.5" 480x320 IPS TFT capacitive touch display
- 1.51" OLED Transparent Display with Converter

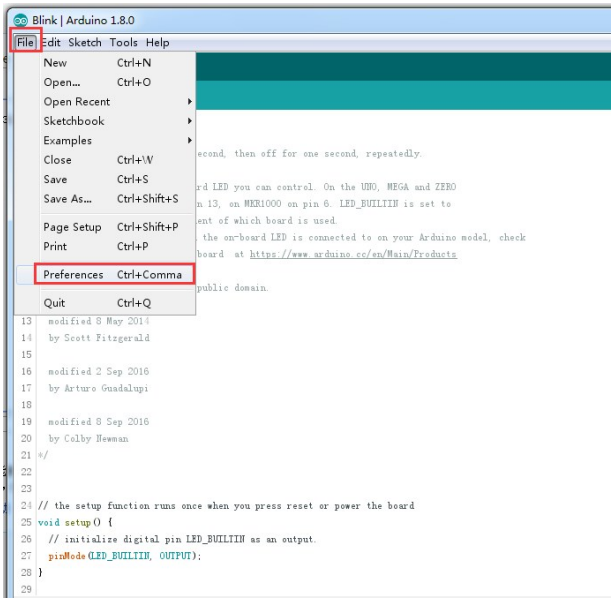
5. The first time to use

5.1 Arduino Environment Config

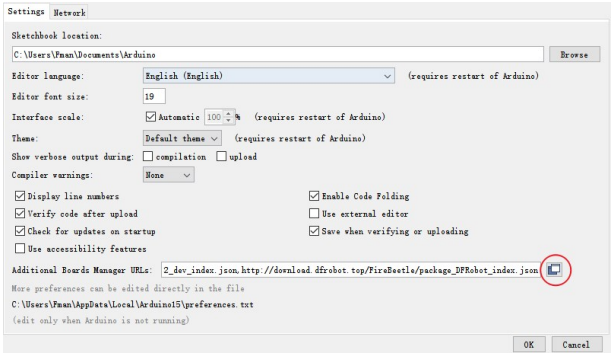
If you use Beetle-ESP32-C3 for the first time, you need to know the following steps

1. Add the json link in the IDE
2. Download the core of the MCU
3. Select the development board and serial port
4. Open the sample code and burn it
5. Get to know the serial monitor

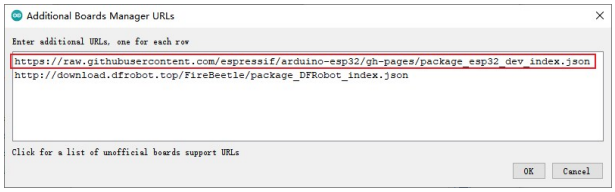
- Arduino IDE compiler environment config
- Configure URL to the Arduino IDE
- Open Arduino IDE and click File->Preferences, as shown below.



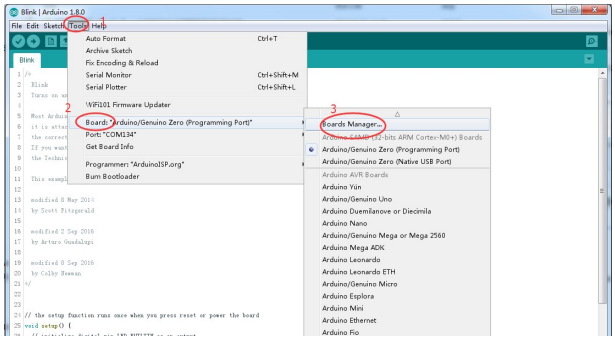
- In the newly opened interface, click the button in the red circle as shown below



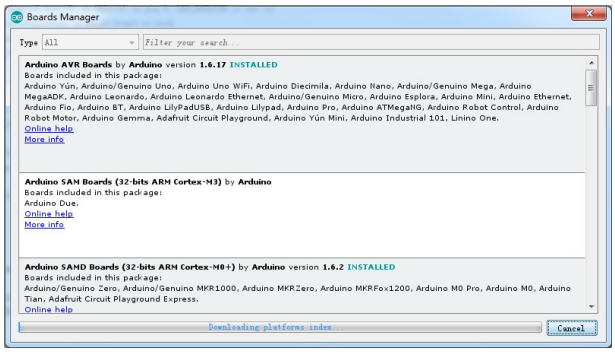
- Copy the following link into the new pop-up dialog box:https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json
- Note: If you have installed another environment before, you can press Enter key at the beginning or end of the previous link and paste the link into any line.



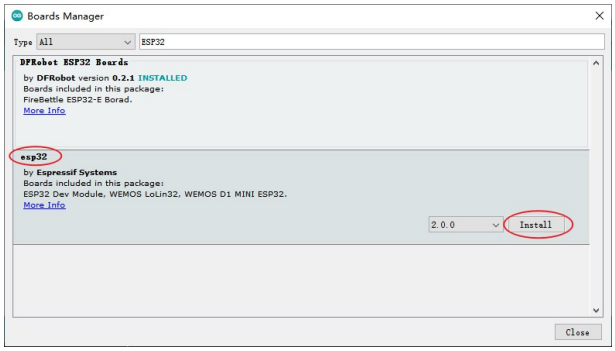
- Click OK
- Update the board
- Open Tools->Board:->Boards Manager... as shown below:



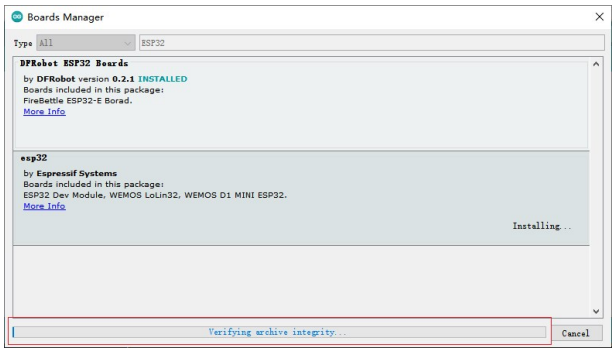
- Boards Manager will automatically update the boards as shown below:



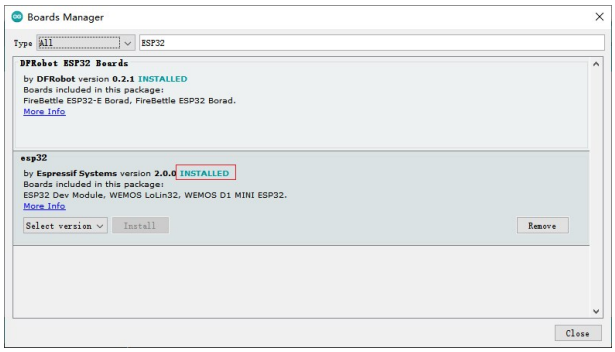
- After completing the update, you can enter esp32 at the top, select esp32 and click install when the following occurs (current version is 2.0.0):



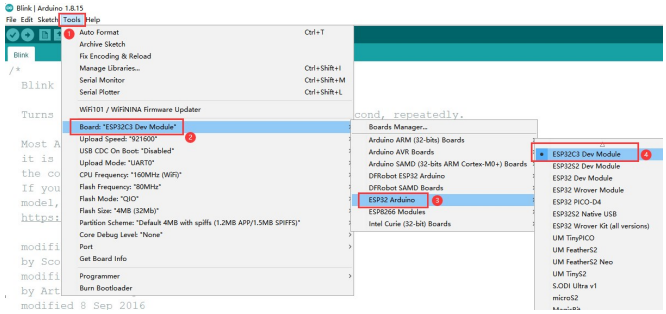
- Wait for the end of the following progress bar:



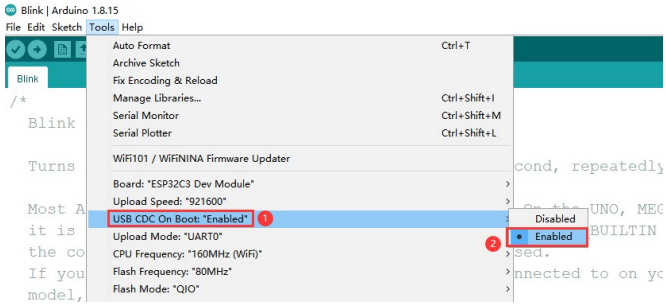
- After completing the installation, the list will show that the esp32 has been installed, as shown below:



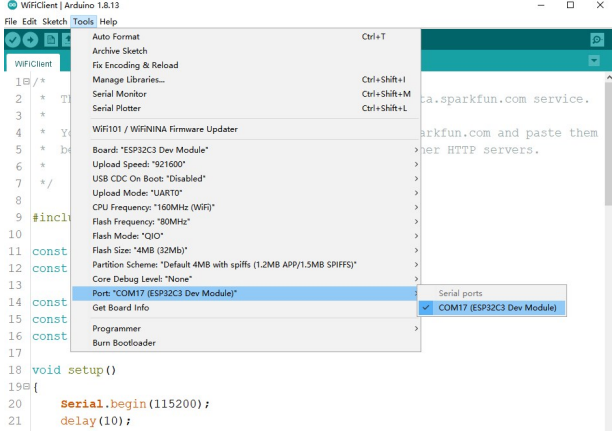
- Click Tools->Board: select ESP32C3 Dev Module (usually the first in the list)



- Before starting, you need to configure the following settings (when you select Disabled, the serial port is RX(20), TX(21), if you need to print on the Arduino monitor via USB, you need to select Enable)



- Click Port to select the corresponding serial port



5.2 LED Blink

Onboard LED is default to pin 10

Code

```
int led = 10;
void setup() {
  pinMode(led,OUTPUT);
}

void loop() {
  digitalWrite(led,HIGH);
  delay(1000);
  digitalWrite(led,LOW);
  delay(1000);
}
```

- Paste the above program into the program box
- Click the arrow to wait for the program to be compiled and burn it to the development board

Burn succeeded

- As shown above, burn succeeded.
- You will see the onboard LED blink.
- Can't burn? Click here.

6. Basic Tutorial (Introduce the difference between Arduino)

- Want to learn about Arduino basics? Click here.

6.1 PWM Output

The PWM function of ESP32C3 needs to be declared in advance.

- ledcAttachPin(GPIO, Channel)

  Description: specify which GPIO the signal will appear on

  Parameter:

  - GPIO: GPIO to output the signal
  - Channel: GPIO where the signal will be generated

- ledcWrite(Channel, dutyCycle)

  Description: set the PWM signal generation channel

  Parameter:

  - Channel: signal generation channel
  - dutyCycle: PWM value

- ledcSetup(Channel, freq, resolution)

  Description: set the PWM signal generation channel

  Parameter:

  - LedChannel: signal generation channel
  - freq: PWM frequency
  - resolution: PWM resolution

Sample

The ESPC3 PWM can be freely mapped to other ports for output, you need to set it up. This sample will be used to help you understand the operation steps, and you can see the LED gradually brighten and dim through it.

```
/*
 * LED breathing light sample
 */
const int ledPin = 10;  // Actually output pin afte

//Set PWM parameter
const int freq = 5000;//PWM frequency
const int ledChannel = 0;//GPIO for signal generati
const int resolution = 8;//8-bit resolution

void setup(){
  //PWM parameter setting
  ledcSetup(ledChannel, freq, resolution);

  //Attach the signal generation channel to the out
  ledcAttachPin(ledPin, ledChannel);
}

void loop(){
  //Start to brighten
  for(int dutyCycle = 0; dutyCycle <= 255; dutyCycl
    // changing the LED brightness with PWM
    ledcWrite(ledChannel, dutyCycle);
    delay(15);
  }

  //Start to dim
  for(int dutyCycle = 255; dutyCycle >= 0; dutyCycl
    // changing the LED brightness with PWM
    ledcWrite(ledChannel, dutyCycle);
    delay(15);
  }
}
```

6.2 Interrupt

ESP32C3 interrupts can be freely assigned

- pinMode(GPIO,INPUT_PULLUP);
  Description: external interrupt pin definition

Parameters:

- GPIO: the pin that ESP32C3 is to use as an interrupt.
- INPUT_PULLINGUP: set pull-up mode.

- attachInterrupt(digitalPinToInterrupt(pin), ISR, mode)
  Description: External interrupts
  Parameters:
  - pin: the Arduino pin number.
  - ISR: the ISR to call when the interrupt occurs; this function must take no parameters and return nothing. This function is sometimes referred to as an interrupt service routine.
  - mode: defines when the interrupt should be triggered. Three constants are predefined as valid values.
  - CHANGE trigger the interrupt pin when the pin level changes
  - RISING trigger the interrupt pin when the pin level changes from low to high
  - FALLING Trigger the interrupt pin when the pin level changes from high to low

- detachInterrupt(digitalPinToInterrupt(pin))
  Description: Turn off the given interrupt.
  Parameters:
  - pin: the interrupt pin to be disabled

- Interrupts ()
  Description: Re-enables interrupts (after they've been disabled by noInterrupts(). Interrupts allow certain important tasks to happen in the background and are enabled by default. Some functions will not work while interrupts are disabled, and incoming communication may be ignored. Interrupts can slightly disrupt the timing of code, however, and may be disabled for particularly critical sections of code.

- noInterrupts ()
  Description: Disables interrupts (you can re-enable them with interrupts()). Interrupts allow certain important tasks to happen in the background and are enabled by default. Some functions will not work while interrupts are disabled, and incoming communication may be ignored. Interrupts can slightly disrupt the timing of code, however, and may be disabled for particularly critical sections of code.

6.3 Setial Port

ESP32C3 serial port initialization requires mapping

- Serial1.begin(baud, config, rxPin, txPin);
  Description: Serial1 initialization
  Parameters:
  - baud: baud rate
  - config: data bit and stop bit setting
  - rxPin: RX pin

- txPin: TX pin

```
Serial1.begin(9600,SERIAL_8N1,/*rx =*/0,/*Tx =*/1);
```

6.4 Servo

ESP32-C3 doesn't support driving servo through the Servo library. Please search and install the ESP32_ISR_Servo library in Sketch->Include Library->Manage Library to drive servo.

7. Advanced Tutorial

7.1 Use SD Library

SD Class

- begin(cspin)
  Description: Initializes the SD library and card. This begins use of the SPI bus (digital pins 11, 12, and 13 on most Arduino boards; 50, 51, and 52 on the Mega) and the chip select pin, which defaults to the hardware SS pin (pin 10 on most Arduino boards, 53 on the Mega). Note that even if you use a different chip select pin, the hardware SS pin must be kept as an output or the SD library functions will not work.
  Parameter:
  - cspin : the Arduino pin connected to the chip select line of the SD card.

  Return: boolean type. True on success; false on failure
- exists()
  Description: Test whether a file or directory exists on the SD card.
  Syntax: SD. exists(filename)
  Parameter:
  - filename: the name of the file to test for existence, which can include directories (delimited by forward-slashes, /)

  Return: boolean type, true if the file or directory exists, false if not
- open()
  Description: Opens a file on the SD card. If the file is opened for writing, it will be created if it doesn't already exist (but the directory containing it must already exist).
  Syntax: SD.open(filename) SD.open(filename, mode)
  Parameter:
  - filename: the name the file to open, which can include directories (delimited by forward slashes, /) mode (optional): the mode in which to open the file, defaults to FILE_READ - byte. one of:
  - FILE_READ: open the file for reading;
  - FILE_WRITE: open the file for reading and writing.

Return: a File object referring to the opened file: if the file couldn't be opened, this object will evaluate to false in a boolean FILE_WRITE: open the file for reading and writing.
Return: a File object referring to the opened file;Return false if the file cannot be opened.

- remove() Description: Remove a file from the SD card. If the file didn't exist, the return value is unspecified, so it is better to use SD. Exists (file name) to detect whether the file exists before removing the file.
  Syntax: SD. remove(filename)
  Parameter:
    - filename：the name of the file to remove, which can include directories (delimited by forward-slashes, /)

  Return: boolean type. True if the removal of the file succeeded, false if not.

- mkdir(filename) Description: Create a directory on the SD card.
  Parameter:
    - filename: the name of the directory to create, with sub-directories separated by forward-slashes, /

  Return: boolean type. True if the creation of the directory succeeded, false if not.

- rmdir(filename) Description: Remove a directory from the SD card. The directory must be empty.
  Syntax: SD.rmdir(filename)
  Parameter:
    - filename: the name of the directory to remove, with sub-directories separated by forward-slashes, /

  Return: booleantype. True if the removal of the directory succeeded, false if not.

File Class

The file class provides the function of reading/writing files. The function of this class is very similar to the that of serial port related functions used before. The member functions are as follows.

- available()
  Description: Check if there are any bytes available for reading from the file.
  Syntax: file. available()
  Parameter:
    - file:an instance of the File class

  Return: the number of bytes available

- close()
  Description: Close the file, and ensure that any data written to it is physically saved to the SD card.
  Syntax: file. close()
  Parameter:
    - file:an instance of the File class

Return: none

- flush()
  Description: Ensures that any bytes written to the file are physically saved to the SD card. This is done automatically when the file is closed.
  Syntax: file.flush
  Parameter:
  - file: an instance of the File class

  Return: none

- peek()
  Description: Read a byte from the file without advancing to the next one.
  Parameter:
  - file: an instance of the File class

  Return: The next byte (or character), or -1 if none is available.

- position( )
  Description: Get the current position within the file (i.e. the location to which the next byte will be read from or written to).
  Syntax: file. position()
  Parameter:
  - file: an instance of the File class

  Return: the position within the file

- print()
  Description: Print data to the file, which must have been opened for writing.
  Syntax: file. print(data)file. print(data, BASE)
  Parameter:
  - file: an instance of the File class
  - data：the data to print (char, byte, int, long, or string)
  - BASE(optional): the base in which to print numbers: BIN for binary (base 2), DEC for decimal (base 10), OCT for octal (base 8), HEX for hexadecimal (base 16).

  Return: byte number to be transmitted.

- println()
  Description: Print data, followed by a carriage return and newline, to the File, which must have been opened for writing.
  Syntax: file. println(data)file,println(data, BASE)
  Parameter:
  - file:an instance of the File class
  - data (optional): the data to print (char, byte, int, long, or string)
  - BASE (optional): the base in which to print numbers: BIN for binary (base 2), DEC for decimal (base 10), OCT for octal (base 8), HEX for hexadecimal (base 16).

Return: byte number to be transmitted.

- seek()
  Description: Seek to a new position in the file, which must be between 0 and the size of the file (inclusive).
  Syntax: file. seek(pos)
  Parameter:
  - file : an instance of the File class
  - pos: the position to which to seek

  Return: true for success, false for failure (boolean)

- size()
  Description: Get the size of the file.
  Syntax: filue. size()
  Parameter:
  - file : an instance of the File class

  Return: the size of the file in bytes

- read()
  Description:Read from the file.
  Syntax: file.read Parameter:
  - file : an instance of the File class

  Return: The next byte (or character), or -1 if none is available.

- write()
  Description: Write data to the file.
  Syntax: file. write(data)file. write(buf, len)
  Parameter:
  - file : an instance of the File class
  - data : the byte, char, or string (char*) to write
  - buf: an array of characters or bytes
  - len : the number of elements in buf

  Return: the number of bytes written

- isDirectory()
  Description: Reports if the current file is a directory or not
  Syntax: file.isDirectory()
  Parameter:
  - file: an instance of the File class

  Return:boolean. True if the file is a directory, false if not

- openNextFile()
  Description: Reports the next file or folder in a directory.
  Syntax: file.openNextFile()
  Parameter:
  - file: an instance of the File class that is a directory

  Return: the next file or folder in the path

- rewindDirectory()
  Description: Back to the first file in the directory
  Syntax: file.rewindDirectory()

Parameter:

- file: an instance of the File class.

Return: None

7.2 ESP32-C3 Bluetooth Receive and Transmit

7.2.1 ESP32-C3 and mobile phone Bluetooth communication

This example demonstrates the data transmission between ESP32-C3 and the mobile phone. If you need to modify or use the data, you only need to change the data receiving part or the data transmitting part of the code.

```
/*
    Video: https://www.youtube.com/watch?v=oCMOYS71
    Based on Neil Kolban example for IDF: https://g
    Ported to Arduino ESP32 by Evandro Copercini

   Create a BLE server that, once we receive a conn
   The service advertises itself as: 6E400001-B5A3-
   Has a characteristic of: 6E400002-B5A3-F393-E0A9
   Has a characteristic of: 6E400003-B5A3-F393-E0A9

   The design of creating the BLE server is:
   1. Create a BLE Server
   2. Create a BLE Service
   3. Create a BLE Characteristic on the Service
   4. Create a BLE Descriptor on the characteristic
   5. Start the service.
   6. Start advertising.

*/

/* This example demonstrates the transparent transm
 * 1. You can see the data sent by ESP32-C3--see AP
 * 2. Data can be sent to ESP32-C3 through the inpu
 * This example is changed from the BLE_uart exampl
 */

#include <BLEDevice.h>
#include <BLEServer.h>
#include <BLEUtils.h>
#include <BLE2902.h>

BLEServer *pServer = NULL;
BLECharacteristic * pTxCharacteristic;
bool deviceConnected = false;
uint8_t txValue = 0;

// See the following for generating UUIDs:
// https://www.uuidgenerator.net/

#define SERVICE_UUID           "6E400001-B5A3-F393-
#define CHARACTERISTIC_UUID_RX "6E400002-B5A3-F393-
#define CHARACTERISTIC_UUID_TX "6E400003-B5A3-F393-


//Bluetooth connect/disconnect processing. Triggere
class MyServerCallbacks: public BLEServerCallbacks
    void onConnect(BLEServer* pServer) {   //This f
      Serial.println("Bluetooth connected");
      deviceConnected = true;
    };

    void onDisconnect(BLEServer* pServer) {  //This
      Serial.println("Bluetooth disconnected");
```

```
      deviceConnected = false;
      delay(500); // give the bluetooth stack the c
      pServer->startAdvertising(); // restart adver

    }
};

/***************Data receiving section***********
/*************************************/
//Bluetooth receive data processing. Triggered auto
class MyCallbacks: public BLECharacteristicCallback
    void onWrite(BLECharacteristic *pCharacteristic
      std::string rxValue = pCharacteristic->getVal

      //if(rxValue == "ON"){Serial.println("开灯");

      if (rxValue.length() > 0) {
        Serial.println("*********");
        Serial.print("Received Value: ");
        for (int i = 0; i < rxValue.length(); i++){
          Serial.print(rxValue[i]);
        }
        Serial.println();
        Serial.println("*********");
      }
    }
};
/*************************************/
/*************************************/


void setup() {
  Serial.begin(115200);
  BLEBegin();  //Initialize Bluetooth

}

void loop() {
/***************Data transmitting section*********
/*************************************/
  if (deviceConnected) {  //If there is a Bluetooth
    pTxCharacteristic->setValue("Hello");  //Send s
    pTxCharacteristic->notify();
    delay(10); // bluetooth stack will go into cong

    pTxCharacteristic->setValue("DFRobot");  //Send
    pTxCharacteristic->notify();
    delay(10); // bluetooth stack will go into cong
  }
/*************************************/
/*************************************/
}


void BLEBegin(){
  // Create the BLE Device
  BLEDevice::init(/*BLE name*/"UART Service");

  // Create the BLE Server
  pServer = BLEDevice::createServer();
  pServer->setCallbacks(new MyServerCallbacks());

  // Create the BLE Service
  BLEService *pService = pServer->createService(SER

  // Create a BLE Characteristic
  pTxCharacteristic = pService->createCharacteristi
                      CHARACTERISTIC_UUID_TX,
                      BLECharacteristic::PROPERTY_NOT
                    );

  pTxCharacteristic->addDescriptor(new BLE2902());
```

```
  BLECharacteristic * pRxCharacteristic = pService-
                      CHARACTERISTIC_UUID_RX,
                      BLECharacteristic::PROPERTY_W
                    );

  pRxCharacteristic->setCallbacks(new MyCallbacks()

  // Start the service
  pService->start();

  // Start advertising
  pServer->getAdvertising()->start();
  Serial.println("Waiting a client connection to no
}
```

7.2.2 Two ESP32C3 Bluetooth Communication

Use this example to demonstrate the data transmission between ESP32-C3 and ESP32-C3. If you need to modify or use the data, you only need to change the data receiving part or the data transmitting part of the code.

Host code

```
/**
 * A BLE client example that is rich in capabilitie
 * There is a lot new capabilities implemented.
 * author unknown
 * updated by chegewara
 */

#include "BLEDevice.h"
//#include "BLEScan.h"

// The remote service we wish to connect to.
static BLEUUID serviceUUID("4fafc201-1fb5-459e-8fcc
// The characteristic of the remote service we are
static BLEUUID    charTXUUID("beb5483e-36e1-4688-b7

static BLEUUID    charRXUUID("beb5483f-36e1-4688-b7

static boolean doConnect = false;
static boolean connected = false;
static boolean doScan = false;
static BLERemoteCharacteristic* pTXRemoteCharacteri
static BLERemoteCharacteristic* pRXRemoteCharacteri
static BLEAdvertisedDevice* myDevice;


/***************Data receiving section************
/*************************************/
//Bluetooth receive data processing, automatically
static void notifyCallback(BLERemoteCharacteristic*
    String BLEData = "";
    for(int i = 0; i < length; i++)  //
      BLEData += (char)pData[i];
    Serial.println("*********");
    Serial.print("Received Value: ");
    Serial.println(BLEData);
    Serial.println("*********");

    //if(BLEData == "ON"){Serial.println("Turn on t

    //Serial.print("Notify callback for characteris
    //Serial.print(pBLERemoteCharacteristic->getUUI
    //Serial.print(" of data length ");
    //Serial.println(length);
}
```

```
/**************************************/
/**************************************/

//Bluetooth connect/disconnect handling. Triggered
class MyClientCallback : public BLEClientCallbacks
  void onConnect(BLEClient* pclient) {
  }

  void onDisconnect(BLEClient* pclient) {
    connected = false;
    Serial.println("onDisconnect");
  }
};


/**
 * Scan for BLE servers and find the first one that
 */
 //Bluetooth scan processing event. Triggered autom
class MyAdvertisedDeviceCallbacks: public BLEAdvert
 /**
   * Called for each advertising BLE server.
   */
  void onResult(BLEAdvertisedDevice advertisedDevic
    //Serial.print("BLE Advertised Device found: ")
    //Serial.println(advertisedDevice.toString().c_

    // We have found a device, let us now see if it
    if (advertisedDevice.haveServiceUUID() && adver
      BLEDevice::getScan()->stop();
      myDevice = new BLEAdvertisedDevice(advertised
      doConnect = true;
      doScan = true;

    } // Found our server
  } // onResult
}; // MyAdvertisedDeviceCallbacks



void setup() {
  Serial.begin(115200);
  Serial.println("Starting Arduino BLE Client appli
  bleBegin();

}

void loop() {
  // If the flag "doConnect" is true then we have s
  // BLE Server with which we wish to connect.  Now
  // connected we set the connected flag to be true
  if (doConnect == true) {
    if (connectToServer()) {
      Serial.println("We are now connected to the B
    } else {
      Serial.println("We have failed to connect to
    }
    doConnect = false;
  }
/***************Data transmitting section*********
/**************************************/
  if (connected) {  //Send data when connected to b
    pTXRemoteCharacteristic->writeValue("I am the h
    pTXRemoteCharacteristic->writeValue("Hello clie
  }
  if(!connected){  //Rescan when not connected to b
    BLEDevice::getScan()->start(5,false);  // this
  }
/**************************************/
/**************************************/
  delay(1000);
}
```

```
void bleBegin()
{
  BLEDevice::init("");

  // Retrieve a Scanner and set the callback we wan
  // have detected a new device.  Specify that we w
  // scan to run for 5 seconds.
  BLEScan* pBLEScan = BLEDevice::getScan();
  pBLEScan->setAdvertisedDeviceCallbacks(new MyAdve
  pBLEScan->setInterval(1349);//Set the scan interv
  pBLEScan->setWindow(449);//Active scan time
  pBLEScan->setActiveScan(true);
  pBLEScan->start(5, false);//Scan time, in seconds
  }


//Bluetooth connection processing
bool connectToServer() {
    Serial.print("Forming a connection to ");
    Serial.println(myDevice->getAddress().toString(

    BLEClient*  pClient  = BLEDevice::createClient(
    Serial.println(" - Created client");

    pClient->setClientCallbacks(new MyClientCallbac

    // Connect to the remove BLE Server.
    pClient->connect(myDevice);  // if you pass BLE
    Serial.println(" - Connected to server");
    pClient->setMTU(517); //set client to request m

    // Obtain a reference to the service we are aft
    BLERemoteService* pRemoteService = pClient->get
    if (pRemoteService == nullptr) {
      Serial.print("Failed to find our service UUID
      Serial.println(serviceUUID.toString().c_str()
      pClient->disconnect();
      return false;
    }
    Serial.println(" - Found our service");


    // Obtain a reference to the characteristic in
    pTXRemoteCharacteristic = pRemoteService->getCh
    if (pTXRemoteCharacteristic == nullptr) {
      Serial.print("Failed to find our characterist
      Serial.println(charTXUUID.toString().c_str())
      pClient->disconnect();
      return false;
    }
    pRXRemoteCharacteristic = pRemoteService->getCh
    if (pRXRemoteCharacteristic == nullptr) {
      Serial.print("Failed to find our characterist
      Serial.println(charRXUUID.toString().c_str())
      pClient->disconnect();
      return false;
    }
    Serial.println(" - Found our characteristic");

    if(pRXRemoteCharacteristic->canNotify())
      pRXRemoteCharacteristic->registerForNotify(no

    connected = true;
    return true;
}
```

Slave code

```
/*
    Based on Neil Kolban example for IDF: https://g
    Ported to Arduino ESP32 by Evandro Copercini
    updates by chegewara
*/

#include <BLEDevice.h>
#include <BLEUtils.h>
#include <BLEServer.h>
#include <BLE2902.h>

// See the following for generating UUIDs:
// https://www.uuidgenerator.net/

#define SERVICE_UUID        "4fafc201-1fb5-459e-8fc
#define CHARACTERISTIC_UUID_RX "beb5483e-36e1-4688-
#define CHARACTERISTIC_UUID_TX "beb5483f-36e1-4688-
uint8_t txValue = 0;
bool deviceConnected = false;
BLECharacteristic *pTxCharacteristic;


//Bluetooth connect/disconnect processing. Triggere
class MyServerCallbacks: public BLEServerCallbacks
    void onConnect(BLEServer* pServer) {   //This f
      Serial.println("Bluetooth connected");
      deviceConnected = true;
    };

    void onDisconnect(BLEServer* pServer) {  //This
        Serial.println("Bluetooth disconnected");
        deviceConnected = false;
        delay(500); // give the bluetooth stack the
        BLEDevice::startAdvertising(); // restart a
    }
};
/***************Data receiving section************
/*************************************/
//Bluetooth receive data processing. Triggered auto
class MyCallbacks: public BLECharacteristicCallback
    void onWrite(BLECharacteristic *pCharacteristic
      std::string rxValue = pCharacteristic->getVal

      //if(rxValue == "ON"){Serial.println("Turn on

      if (rxValue.length() > 0) {
        Serial.println("*********");
        Serial.print("Received Value: ");
        for (int i = 0; i < rxValue.length(); i++)
              Serial.print(rxValue[i]);    //Print

        Serial.println();
        Serial.println("*********");
      }
    }
};
/*************************************/
/*************************************/

void setup() {
  Serial.begin(115200);
  Serial.println("Starting BLE work!");
  bleBegin();
}

/***************Data sending section************/
/*************************************/
void loop() {
  if(deviceConnected){  //Send data when a device i
    pTxCharacteristic->setValue("I am a slave");
    pTxCharacteristic->notify();
```

```
    pTxCharacteristic->setValue("Hello Sever");
    pTxCharacteristic->notify();
  }
  /*************************************/
  /*************************************/
  delay(1000);
}


void bleBegin()
{
  BLEDevice::init(/*BLE name*/"Long name works now"
  BLEServer *pServer = BLEDevice::createServer();
  pServer->setCallbacks(new MyServerCallbacks());
  BLEService *pService = pServer->createService(SER
  BLECharacteristic *pRxCharacteristic = pService->
                                          CHARACTERI
                                          BLECharact
                                         );
  pRxCharacteristic->setCallbacks(new MyCallbacks()

 pTxCharacteristic = pService->createCharacteristic
                                          CHARACTERI
                                          BLECharact
                                         );
 pTxCharacteristic->addDescriptor(new BLE2902());

  pService->start();
  // BLEAdvertising *pAdvertising = pServer->getAdv
  BLEAdvertising *pAdvertising = BLEDevice::getAdve
  pAdvertising->addServiceUUID(SERVICE_UUID);
  pAdvertising->setScanResponse(true);
  pAdvertising->setMinPreferred(0x06);  // function
  pAdvertising->setMinPreferred(0x12);
  BLEDevice::startAdvertising();

  }
```

7.3 WIFI control LED

ESP32C3 has WIFI function, the following example uses ESP32C3 to create a wifi server, use the client to connect to the server, and control the on and off of LED

Steps

1. Connect to WIFI "Beetle ESP32 C3", WIFI password has been set: 12345678
2. Visit the website http://192.168.4.1/ON to turn on the light Visit http://192.168.4.1/OFF to turn off the light
3. After the visit, click the up and down to conveniently control the on and off of the light without entering the URL.

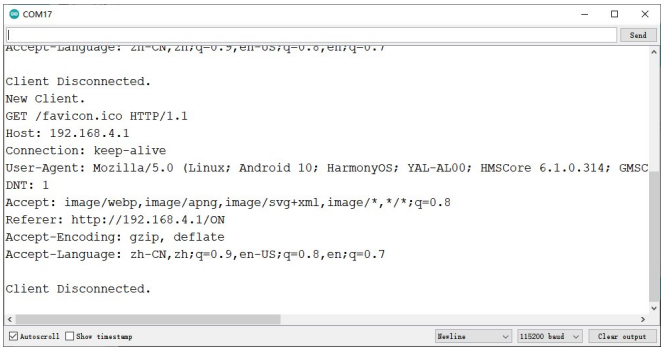
Code

```
/*
Steps:
1. Connect to WIFI "Beetle ESP32 C3", WIFI password
2. Visit the website http://192.168.4.1/ON to turn
3. After the visit, click the up and down to conven
*/

#include <WiFi.h>
#include <WiFiClient.h>
#include <WiFiAP.h>
```

```
#define myLED 10 //Set pin 10 as LED pin
// Set WIFI name and password
const char *ssid = "Beetle ESP32 C3";//WIFI name
const char *password = "12345678";//password

WiFiServer server(80);//The default web service por


void setup() {
  pinMode(myLED, OUTPUT);

  Serial.begin(115200);
  Serial.println();
  Serial.println("Configuring access point...");

  //If you want to open the network without a passw
  WiFi.softAP(ssid, password);
  IPAddress myIP = WiFi.softAPIP();
  Serial.print("AP IP address: ");
  Serial.println(myIP);
  server.begin();

  Serial.println("Server started");
}

void loop() {
  WiFiClient client = server.available();   // dete

  if (client) {                             // chec
    Serial.println("New Client.");
    String currentLine = "";                // crea
    while (client.connected()) {            // keep
      if (client.available()) {             // chec
        char c = client.read();             // read
        //Serial.write(c);                    // pr
        if (c == '\n') {                    // if n

          //end with a newline reminder
          if (currentLine.length() == 0) {
            client.println("HTTP/1.1 200 OK");
            client.println("Content-type:text/html"
            client.println();
             //connect characters with here
            client.print("Click <a href=\"/ON\">her
            client.print("Click <a href=\"/OFF\">he

            // HTTP response is blank
            client.println();
            // Break out of the loop
            break;
          } else {    // Clear variable cached data
            currentLine = "";
          }
        } else if (c != '\r') {  // If you get char
          currentLine += c;      // The obtained ch
        }

        // Check to see if /ON or /OFF
        if (currentLine.endsWith("/ON")) {
          digitalWrite(myLED, HIGH);
        }
        if (currentLine.endsWith("/OFF")) {
          digitalWrite(myLED, LOW);
        }
      }
    }
    // Disconnect
    client.stop();
    Serial.println("Client Disconnected.");
  }
}
```

Result

Use the mobile phone to connect to this wifi, access 192.168.4.1 through the browser, as shown in the figure, the ip address is 192.168.4.1, and the service is enabled.



Use a browser to access the ip address and get it as shown below

Click here to turn ON the LED.
Click here to turn OFF the LED.

Try clicking the links separately to control the LEDs

Member function

- WiFiServer server()
  Description: Set the server port
- softAP(ssid,password)
  Description: Configure WiFi as AP mode, and set the
  - ssid: wifi name in AP mode
  - password: wifi password in ap mode
- server.available()
  Description: Check whether the service port is connected (whether WIFI is connected)
- client.connected()
  Description: Check connection status
  Return value: true/false
- client.available()
  Description: Detect whether there is data input connected to WIFI
- client.read()
  Description: Read WIFI received data
- currentLine.endsWith()
  Description: Check whether the bracket content is obtained
- client.stop()
  Description: Disconnect

8. Application example expansion

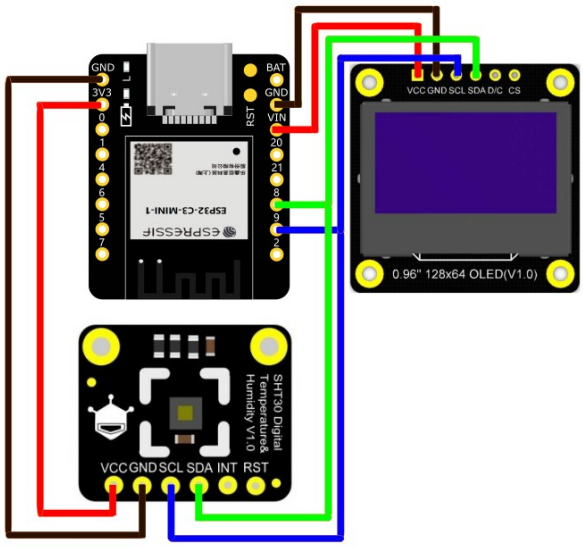
8.1 OLED display measured temperature and humidity

Getting the temperature and humidity displayed on the OLED screen is a very intuitive and interesting project. Below you will learn the basic OLED display and use the I2C interface to obtain the temperature and humidity sensor data. You need to prepare

- 0.96"128x64 IIC/SPI OLED monochrome display x1
- Jumper Wires
- FermionSHT30 temperature and humidity sensor x1 (or other SHT temperature and humidity sensor using the same SHT3x code base)

1. You need to install the SHT3x library first. For how to install the library, click here.

2. Wiring diagram



Connect the wires according to the picture above and copy the code below, you will see the temperature and humidity conditions around you.

```
#include <Arduino.h>
#include <U8g2lib.h> //Import font library
//#include <SPI.h>
#include <Wire.h>

#include <DFRobot_SHT3x.h>

/*
---Display hardware I2C interface---
U8G2_R0 does not rotate, landscape, drawing directi
U8G2_R1 is rotated 90 degrees clockwise, and the dr
U8G2_R2 is rotated 180 degrees clockwise, and the d
U8G2_R3 is rotated 270 degrees clockwise, and the d
U8G2_MIRROR displays mirrored content normally (use
U8x8_PIN_NONE means the pin is empty and the reset
---Display hardware SPI interface---
Connect cs by pin (pin can be selected by yourself)
Connect dc by pin (pin can be selected by yourself)
*/
U8G2_SSD1306_128X64_NONAME_F_HW_I2C u8g2(/* rotatic

//When ADR is connected to VDD, 0x45 can be selecte
//The default is 0x45, RST (reset pin) does not nee
```

```
DFRobot_SHT3x sht3x(&Wire,/*address=*/0x45,/*RST=*/

//To use SPI, you need to comment the above code an
//DFRobot_SHT3x   sht3x;

void setup() {
  Serial.begin(115200);
  u8g2.begin();
  u8g2.setFontPosTop();//When using drawStr to disp
  //Initialize the sensor
  while (sht3x.begin() != 0) {
        Serial.println("Failed to Initialize the ch
        delay(1000);
      }
      Serial.print("Chip serial number");
      Serial.println(sht3x.readSerialNumber());
      if(!sht3x.softReset()){
         Serial.println("Failed to Initialize the c
       }
}

void loop() {
  //Clean the screen
  u8g2.clearBuffer();
  //Use temperature and humidity read assignments f
  float   temp = sht3x.getTemperatureC();
  float   humi = sht3x.getHumidityRH();
  //Display temperature
  u8g2.setFont(u8g2_font_osb18_tf);    // Choose fo
  u8g2.drawStr(5,10,"Temp");//Write the character a
  u8g2.setFont(u8g2_font_t0_18b_tr);
  u8g2.setCursor(75, 15);//Display starts from this
  u8g2.print(temp);
  //Display humidity
  u8g2.setFont(u8g2_font_osb18_tf);
  u8g2.drawStr(5,40,"Humi");
  u8g2.setFont(u8g2_font_t0_18b_tr);
  u8g2.setCursor(75, 45);
  u8g2.print(humi);
  u8g2.sendBuffer();
  delay(1000);
}
```

Result

Member function

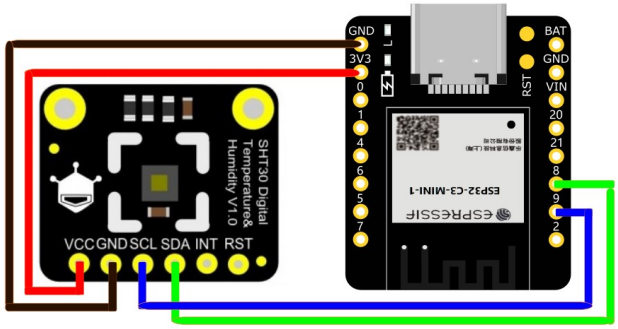
- u8g2.drawStr(x,y,"Temp")
  Description: Specify the screen position to display custom content
  Parameter:
  - X, Y: coordinates to start writing (the display font is displayed from the lower left corner to the upper right corner)
  - "Temp": English and numbers can be filled
- u8g2.setCursor(x,y) and u8g2.print()
  Description: Used together, the former is to display the starting position, and the latter has the same function as Arduino's print
  Parameter:
  - X, Y: coordinates to start writing (the display font is displayed from the lower left corner to the upper right corner)

8.2 Use WIFI to obtain temperature and humidity

This example is based on Example 8.3 to further learn the information transfer of Wifi under the LAN. You can learn how to access the IP address under the LAN to obtain the temperature and humidity sensor status of the SHT30 in another place. You need to prepare

- FermionSHT30 temperature and humidity sensor x1 (or other SHT temperature and humidity sensors using the same SHT3x code base)
- Jumper Wires You will also need to follow the steps in Example 9.1 to install the sensor library and connect it with the ESP32.

Wiring diagram



Steps

1. Connect to WIFI "Beetle ESP32 C3", WIFI password has been set: 12345678
2. Visit the website http://192.168.4.1/GET to get the temperature and humidity information in the LAN
3. In the temperature and humidity display page, you can update the sensor data by refreshing

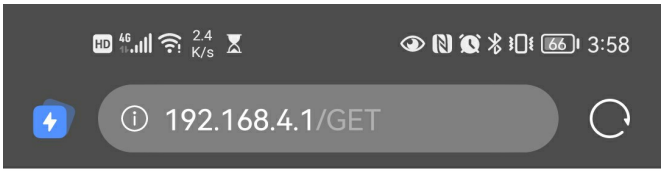
Code

```
/*
In this example, SHT30 is connected to ESPC3 to obt
*/

#include <WiFi.h>
#include <WiFiClient.h>
#include <WiFiAP.h>
#include <DFRobot_SHT3x.h>
//When ADR is connected to VDD, 0x45 can be selecte
//The default is 0x45, RST (reset pin) does not nee
DFRobot_SHT3x sht3x(&Wire,/*address=*/0x45,/*RST=*/

//To use SPI, you need to comment the above code an
//DFRobot_SHT3x   sht3x;

// Set WIFI name and password
const char *ssid = "Beetle ESP32 C3";//WIFI name
const char *password = "12345678";//password

WiFiServer server(80);//The default web service por

// Display the last sensor status feedback status

void setup() {
```

```
  //pinMode(myLED, OUTPUT);

  Serial.begin(115200);
  Serial.println();
  Serial.println("Configuring access point...");

  //If you want to open the network without a passw
  WiFi.softAP(ssid, password);
  IPAddress myIP = WiFi.softAPIP();
  Serial.print("AP IP address: ");
  Serial.println(myIP);
  server.begin();

  Serial.println("Server started");
  //Initialize the sensor
  while (sht3x.begin() != 0) {
        Serial.println("Failed to Initialize the ch
        delay(1000);
      }
      Serial.print("Chip serial number");
      Serial.println(sht3x.readSerialNumber());
      if(!sht3x.softReset()){
         Serial.println("Failed to Initialize the c
       }
}

void loop() {

WiFiClient client = server.available();   // detect

  if (client) {                             // chec
    Serial.println("New Client.");
    String currentLine = "";                // crea
    while (client.connected()) {            // keep
      if (client.available()) {             // chec
        char c = client.read();             // read
        //Serial.write(c);                    // pr
        if (c == '\n') {                    // if n

          //Clear the cached content
          if (currentLine.length() == 0) {
            client.print(" ");
            break;
          } else {    // Clear variable cached data
            currentLine = "";
          }
        } else if (c != '\r') {  // If you get char
          currentLine += c;      // The obtained ch
        }

        // Check if /GET is at the end
        if (currentLine.endsWith("/GET")) {
             //Read temperature and humidity
             float   temp = sht3x.getTemperatureC()
             float   humi = sht3x.getHumidityRH();
             //Print on web page
             client.print("temp (C): "); client.pri
             client.print("humi (%RH): "); client.p
        }
      }
    }
    // Disconnect
    client.stop();
    Serial.println("Client Disconnected.");
  }
  }
```

Result

You can access the website through a mobile phone, computer, etc. to obtain the following results (temperature and humidity under the LAN).




```
temp (C): 25.13
humi (%RH): 54.14
```

Member function

- WiFi.softAP(ssid, password)
  Description: Similar to opening the hotspot with the set WIFI account password
  Parameter:
  - ssid: WIFI name
  - password: WIFI password
- WiFi.softAPIP()
  Description: WIFI IP address

8.3 Get network time with WIFI

This example demonstrates getting the time from a network time server and using the RTC clock that comes with the ESP32 to keep the time updated. This example comes from CSDN blogger "Naisu Xu", the original link:

SEN0259 TF Mini LiDAR ToF Laser Range Sensor

SEN0214 20A Current Sensor

SEN0262 Gravity Analog Current to Voltage Converter for 4~20mA Application

SEN0291 Gravity: I2C Digital Wattmeter

DFR0027 DFRobot Digital Vibration Sensor V2

DFR0028 DFRobot Tilt Sensor

DFR0029 DFRobot Digital Push Button

DFR0030 DFRobot Capacitive Touch Sensor

DFR0032 Digital Buzzer Module

DFR0033 Digital magnetic sensor

DFR0034 Analog Sound Sensor

SEN0038 Wheel Encoders for DFRobot 3PA and 4WD Rovers

DFR0051 Analog Voltage Divider

DFR0052 Analog Piezo Disk Vibration Sensor

DFR0076 Flame sensor

DFR0053 Analog Slide Position Sensor

DFR0054 Analog Rotation Sensor V1

DFR0058 Analog Rotation Sensor V2

DFR0061 Joystick Module For Arduino

DFR0075 ADKeyboard Module

DFR0332 Fan Module

SEN0177 PM2.5 laser dust sensor

SEN0160 Weight Sensor Module

SEN0170 Wind Speed Sensor Voltage Type 0-5V

TOY0048 High Accuracy Dual Axis Inclinometer Sensor Arduino Gadgeteer Compatible

SEN0187 RGB and Gesture Sensor

SEN0186 Weather Station with Anemometer Wind vane Rain bucket

SEN0192 MicroWave Sensor

SEN0185 Hall sensor

FIT0449 DFRobot Speaker v1.0

https://blog.csdn.net/Naisu_kun/article/details/115627629

```
#include <WiFi.h>

const char *ssid = "********";    //WIFI name
const char *password = "********"; //WIFI password

const char *ntpServer = "pool.ntp.org";
const long gmtOffset_sec = 8 * 3600;
const int daylightOffset_sec = 0;

void printLocalTime()
{
    struct tm timeinfo;
    if (!getLocalTime(&timeinfo))
    {
        Serial.println("Failed to obtain time");
        return;
    }
    Serial.println(&timeinfo, "%F %T %A"); // forma
}

void setup()
{
    Serial.begin(115200);
    Serial.println();

    WiFi.begin(ssid, password);
    while (WiFi.status() != WL_CONNECTED)
    {
        delay(500);
        Serial.print(".");
    }
    Serial.println("WiFi connected!");

    // Get and set the time from the network time s
    // After the acquisition is successful, the chi
    configTime(gmtOffset_sec, daylightOffset_sec, n
    printLocalTime();

    WiFi.disconnect(true);
    WiFi.mode(WIFI_OFF);
    Serial.println("WiFi disconnected!");
}

void loop()
{
    delay(1000);
    printLocalTime();
}
```

struct tm structure

```
struct tm {
int tm_sec; // seconds, value 0~59;
int tm_min; // minutes, value 0~59;
int tm_hour; // hour, value 0~23;
int tm_mday; // The day of the month, the value is
int tm_mon; // month, value 0~11;
int tm_year; // year, whose value is equal to the a
int tm_wday; // week, value 0~6, 0 is Sunday, 1 is
int tm_yday; // The date of the year, value 0~365,
int tm_isdst; // Daylight saving time identifier, w
};
```

struct tm structure formatted output

SEN0203 Heart Rate Sensor
DFR0423 Self-Locking Swicth
SEN0213 Heart Rate Monitor Sensor
SEN0221 Gravity Hall Angle Sensor
SEN0223 Conductivity Switch Sensor
SEN0230 Incremental Photoelectric Rotary Encoder - 400P R
SEN0235 EC11 Rotary Encoder Module
SEN0240 Analog EMG Sensor by OYMotion
SEN0232 Gravity Analog Sound Level Meter
SEN0233 Air Quality Monitor PM 2.5, Formaldehyde, Temperature & Humidity Sensor
DFR0515 FireBeetle Covers-OSD Character Overlay Module
SEN0257 Gravity Water Pressure Sensor
SEN0289 Gravity: Digital Shake Sensor
SEN0290 Gravity: Lightning Sensor
DFR0271 GMR Board
ROB0003 Pirate 4WD Mobile Platform
ROB0005 Turtle 2WD Mobile Platform
ROB0025 NEW A4WD Mobile Robot with encoder
ROB0050 4WD MiniQ Complete Kit
ROB0111 4WD MiniQ Cherokey
ROB0036 6 DOF Robotic Arm Kit
FIT0045 DF05BB Tilt Pan Kit
ROB0102 Cherokey 4WD Mobile Platform
ROB0117 Basic Kit for Cherokey 4WD
ROB0022 4WD Mobile Platform
ROB0118 Basic Kit for Turtle 2WD
ROB0080 Hexapod Robot Kit
ROB0112 Devastator Tank Mobile Platform
ROB0114 Devastator Tank Mobile Platform
ROB0124 HCR Mobile Platform with

Formatting characters	Output
%a	Abbreviation for day of the week
%A	Full name of the day of the week
%b	Abbreviation for month
%B	Full name of month
%c	Standard date string
%C	Last two digits of the year
%d	Day of the month in decimal
%D	Month/day/year
%e	Day of the month in decimal in a two-character field
%F	Year-month-day
%g	The last two digits of the year, using a week-based year
%G	Years, use week-based years
%h	Abbreviated month name
%H	24 hour clock
%I	12 hour clock
%j	Day of the year in decimal
%m	Month in decimal
%M	Minutes in decimal
%p	Equivalent display of local AM or PM
%r	Time in 12 hours
%R	Display hours and minutes: hh:mm
%S	Second in decimal
%t	Horizontal tab
%T	Display hours, minutes and seconds: hh:mm:ss
%u	Day of the week, Monday is the first day (values from 0 to 6, Monday is 0)
%U	Week of the year, with Sunday as the first day (values from 0 to 53)
%V	Week of the year, using a week-based year
%w	Day of the week in decimal (values from 0 to 6, 0 for Sunday)
%W	Week of the year, with Monday as the first day (values from 0 to 53)
%x	Standard date string
%X	Standard time string
%y	Year in decimal without century (values from 0 to 99)

ROB0128 Devastator Tank Mobile Platform Metal DC Gear Motor

ROB0137 Explorer MAX Robot

ROB0139 FlameWheel Robot

DFR0270 Accessory Shield for Arduino

DFR0019 Prototyping Shield For Arduino

DFR0265 IO Expansion Shield for Arduino V7

DFR0210 Bees Shield

DFR0165 Mega IO Expansion Shield V2.3

DFR0312 Raspberry Pi GPIO Extension Board

DFR0311 Raspberry Pi Meet Arduino Shield

DFR0327 Arduino Shield for Raspberry Pi 2B and 3B

DFR0371 IO Expansion Shield for Bluno M3

DFR0356 Bluno Beetle Shield

DFR0412 Gravity IO Expansion Shield For DFRduino M0

DFR0375 Cookie I O Expansion Shield V2

DFR0334 GPIO Shield for Arduino V1.0

DFR0502 Gravity IO Expansion & Motor Driver Shield V1.1

DFR0518 Micro Mate- A Mini Expansion Board for micro bit

DFR0578 Gravity I O Expansion Shield for OpenMV Cam M7

DFR0577 Gravity I O Expansion Shield for Pyboard

DFR0626 MCP23017 IIC to 16 digital IO expansion module

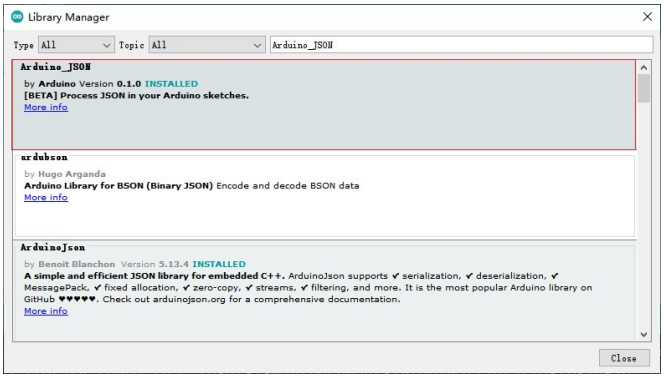
Formatting characters	Output
%Y	Year in decimal with century
%z	Time zone name, or return null if no time zone name is available

8.4 Get the weather with WIFI

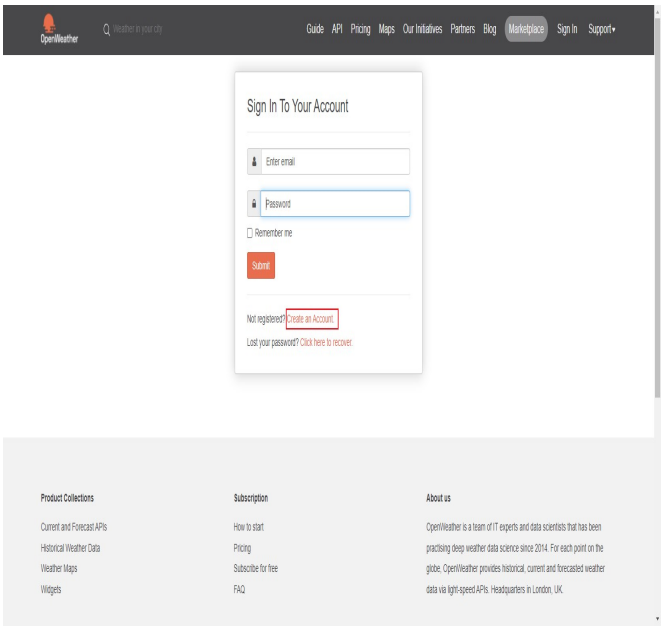
This example is used to let you learn how to get weather information and let you experience the information obtained in HTTP which extract data through Json and print it

- Before burning the code, we perform the following steps

1.You need to install the Arduino_JSON library. Enter Arduino_JSON in Arduino IDE Tools -> Manage Libraries and install the library



2.Register an OpenWeather account to get the weather information we want. Open your browser and go to https://openweathermap.org/appid/ press the sign up button and create a free account.



Click My API Keys to enter the interface for obtaining API

DFR0287 LCD12864 Shield

DFR0009 LCD KeyPad Shield For Arduino

DFR0063 I2C TWI LCD1602 Module Gadgeteer Compatible

DFR0154 I2C TWI LCD2004 Module Arduino Gadgeteer Compatible

DFR0202 RGB LED Matrix

DFR0090 3-Wire LED Module

TOY0005 OLED 2828 color display module .NET Gadgeteer Compatible

TOY0006 OLED 9664 RGB Display module

TOY0007 OLED 2864 display module

FIT0328 2.7 OLED 12864 display module

DFR0091 3-wire Serial LCD Module Arduino Compatible

DFR0347 2.8 TFT Touch Shield with 4MB Flash for Arduino and mbed

DFR0348 3.5 TFT Touch Shield with 4MB Flash for Arduino and mbed

DFR0374 LCD Keypad Shield V2.0

DFR0382 LED Keypad Shield V1.0

DFR0387 TELEMATICS 3.5 TFT Touch LCD Shield

DFR0459 8x8 RGB LED Matrix

DFR0460 64x32 RGB LED Matrix - 4mm Pitch/64x32 Flexible RGB LED Matrix-4mm Pitch/64x32 Flexible RGB LED Matrix-5mm Pitch

DFR0461 Gravity Flexible 8x8 RGB LED Matrix

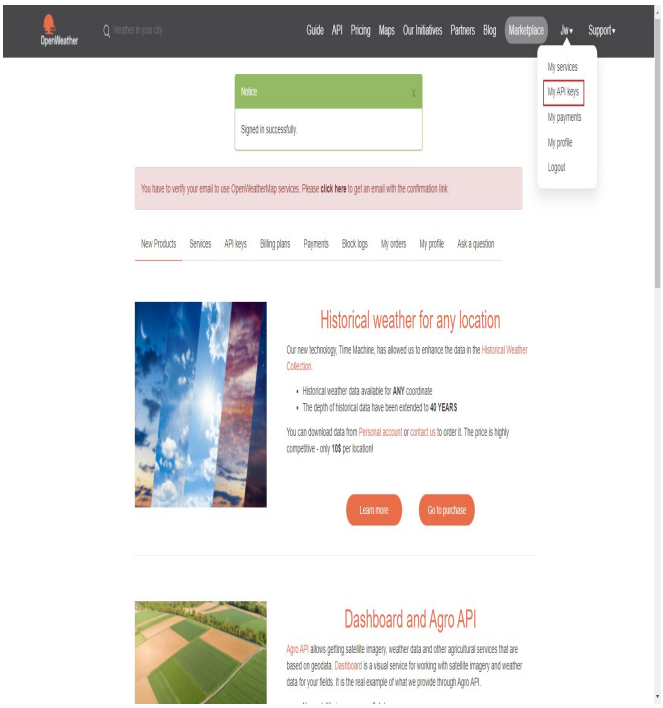
DFR0462 Gravity Flexible 8x32 RGB LED Matrix

DFR0463 Gravity Flexible 16x16 RGB LED Matrix

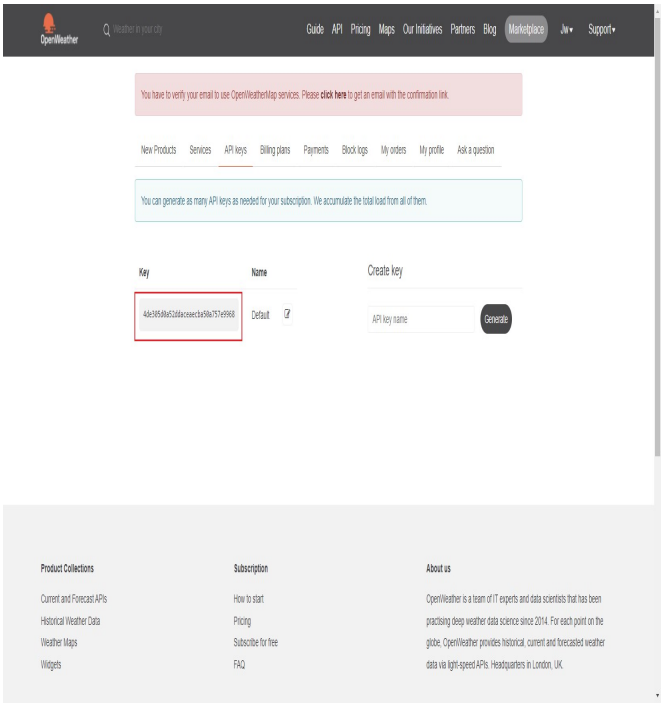
DFR0471 32x16 RGB LED Matrix - 6mm pitch

DFR0472 32x32 RGB LED Matrix - 4mm pitch

DFR0464 Gravity I2C 16x2 Arduino LCD with RGB Backlight Display



Copy the key here (this key is your only key to get weather information from OpenWeather)



You can fill the Key into the following URL and fill in the city name and its country to get the city weather information http://api.openweathermap.org/data/2.5/weather?q=yourCityName,yourCountryCode&APPID=yourAPIkey The following is an example to replace the city (such as Chengdu) of the data you want in yourCityName, yourCountryCode and the country code of the city (such as CN), fill in yourAPIkey which is the API key obtained earlier, the following is the URL of Chengdu, China with the API added: http://api.openweathermap.org/data/2.5/weather?q=ChengDu,CN&APPID=4de305d0a52ddaceaecba50a757e9968 Copying your URL into your browser will return a set of information corresponding to your local weather. On the day this tutorial was written, we had the following information about the weather in Chengdu, China.

DFR0499 64x64 RGB LED Matrix - 3mm pitch

DFR0506 7" HDMI Display with Capacitive Touchscreen

DFR0555\DF0556\DFR0557 Gravity I2C LCD1602 Arduino LCD Display Module

DFR0529 2.2 inches TFT LCD Display V1.0 (SPI Interface)

DFR0605 Gravity: Digital RGB LED Module

FIT0352 Digital RGB LED Weatherproof Strip 60LED m*3m

DFR0645-G DFR0645-R 4-Digital LED Segment Display Module

SKU DFR0646-G DFR0646-R 8-Digital LED Segment Display Module

DFR0597 7x71 Flexible RGB LED Matrix

DFR0231 NFC Module for Arduino

TEL0005 APC220 Radio Data Module

TEL0023 BLUETOOH BEE

TEL0026 DF-BluetoothV3 Bluetooth module

TEL0037 Wireless Programming Module For Arduino

TEL0044 DFRduino GPS Shield-LEA-5H

TEL0047 WiFi Shield V2.1 For Arduino

TEL0051 GPS GPRS GSM Module V2.0

TEL0067 WiFi Bee V1.0

TEL0073 BLE-Link

TEL0075 RF Shield 315MHz

TEL0078 WIFI Shield V3 PCB Antenna

TEL0079 WIFI Shield V3 RPSMA

TEL0084 BLEmicro

TEL0086 DF-Beacon EVB

TEL0087 USBBLE-LINK Bluno Wireless Programming Adapter

TEL0080 UHF RFID MODULE-USB

TEL0081 UHF RFID MODULE-RS485

{"coord":{"lon":104.0667,"lat":30.6667},"weather":[{"id":804,"main":"Clouds","description":"overcast clouds","icon":"04d"}],"base":"stations","main":{"temp":290.86,"feels_like":290.72,"temp_min":290.86,"temp_max":290.86,"pressure":1014,"humidity":78,"sea_level":1014,"grnd_level":957},"visibility":10000,"wind":{"speed":0.51,"deg":79,"gust":0.36},"clouds":{"all":94},"dt":1633655120,"sys":{"country":"CN","sunrise":1633647674,"sunset":1633689669},"timezone":28800,"id":1815286,"name":"Chengdu","cod":200}

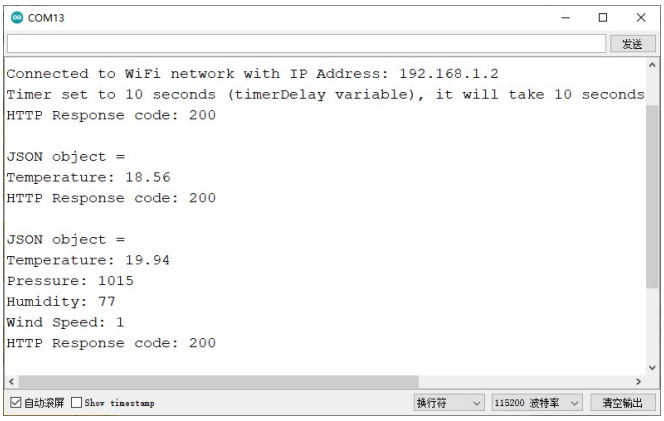
Code

```
/*
This example learns how to get weather information
*/

#include <WiFi.h>
#include <HTTPClient.h>
#include <Arduino_JSON.h>

//Modify WIFI name and password
const char* ssid = "******";//WIFI name
const char* password = "******";//WIFI password

//Fill in the API Key you got
String openWeatherMapApiKey = "4de305d0a52ddaceaecb
//Example:
//String openWeatherMapApiKey = "4de305d0a52ddaceae

// Fill in your city name and country abbreviation
String city = "ChengDu";
String countryCode = "CN";

//Example:
//String city = "ChengDu";
//String countryCode = "CN";

//Set the interval for obtaining information, the f
//You should limit the minimum interval of access t
unsigned long lastTime = 0;
//Set to get weather data every 10 minutes
//unsigned long timerDelay = 600000;
//Set to get weather data every 10 seconds
unsigned long timerDelay = 10000;

String jsonBuffer;

void setup() {
  Serial.begin(115200);
  WiFi.begin(ssid, password);
  Serial.println("Connecting");

  //Determine if WIFI is connected
  while(WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.print("Connected to WiFi network with IP A
  Serial.println(WiFi.localIP());
  Serial.println("Timer set to 10 seconds (timerDel
}
```

```
void loop() {
  //Send HTTP get request
  if ((millis() - lastTime) > timerDelay) {
    //Check if WIFI is connected
    if(WiFi.status()== WL_CONNECTED){
      String serverPath = "http://api.openweatherma

      //Put the combined URL into the httpGETReques
      jsonBuffer = httpGETRequest(serverPath.c_str(
      Serial.println(jsonBuffer);

      //Store the parsed Json object value in the J
      JSONVar myObject = JSON.parse(jsonBuffer);

      //Determine if the parsing was successful
      if (JSON.typeof(myObject) == "undefined") {
        Serial.println("Parsing input failed!");
        return;
      }

      Serial.print("JSON object = ");
      Serial.println(myObject);
      Serial.print("Temperature: ");
      //The obtained temperature is actually Kelvin
      //Kelvin = Celsius + 273.15
      double c = myObject["main"]["temp"];
      c = c-273.15;
      Serial.println(c);
      Serial.print("Pressure: ");
      //myObject["main"]["pressure"], the front is
      Serial.println(myObject["main"]["pressure"]);
      Serial.print("Humidity: ");
      Serial.println(myObject["main"]["humidity"]);
      Serial.print("Wind Speed: ");
      Serial.println(myObject["wind"]["speed"]);
    }
    else {
      Serial.println("WiFi Disconnected");
    }
    lastTime = millis();
  }
}

String httpGETRequest(const char* serverName) {
  WiFiClient client;
  HTTPClient http;

  //Connect URL
  http.begin(client, serverName);

  //Send HTTP site request
  int httpResponseCode = http.GET();

  //This array is used to store the obtained data
  String payload = "{}";

  //Put the obtained data into the array
  if (httpResponseCode>0) {
    Serial.print("HTTP Response code: ");
    Serial.println(httpResponseCode);
    payload = http.getString();
  }
  else {
    Serial.print("Error code: ");
    Serial.println(httpResponseCode);
  }

  //Release resources
  http.end();

  //Return the obtained data for Json processing
  return payload;
```

```
}
```

Result



Member function

- httpGETRequest(serverPath.c_str());
  Description: Parse the obtained object
- JSON.typeof(myObject)
  Description: Determine whether the object is in a parsed format

8.5 SmartConfig One-click network configuration + automatic reconnection

Through this code, Espressif's ESP-TOUCH can be used for one-click network configuration. Click to download the Android version For IOS version, please search for Espressif Esptouch in the App Store

```
#include <WiFi.h>

void SmartConfig()
{
  WiFi.mode(WIFI_STA);
  Serial.println("\r\nWait for Smartconfig...");
  WiFi.beginSmartConfig();
  while (1)
  {
    Serial.print(".");
    delay(500);                   // wait for a sec
    if (WiFi.smartConfigDone())
    {
      Serial.println("SmartConfig Success");
      Serial.printf("SSID:%s\r\n", WiFi.SSID().c_st
      Serial.printf("PSW:%s\r\n", WiFi.psk().c_str(
      break;
    }
  }
}

bool AutoConfig()
{
    WiFi.begin();
    for (int i = 0; i < 20; i++)
    {
        int wstatus = WiFi.status();
        if (wstatus == WL_CONNECTED)
        {
            Serial.println("WIFI SmartConfig Succes
            Serial.printf("SSID:%s", WiFi.SSID().c_
            Serial.printf(", PSW:%s\r\n", WiFi.psk(
```

```
            Serial.print("LocalIP:");
            Serial.print(WiFi.localIP());
            Serial.print(" ,GateIP:");
            Serial.println(WiFi.gatewayIP());
            return true;
        }
        else
        {
            Serial.print("WIFI AutoConfig Waiting..
            Serial.println(wstatus);
            delay(1000);
        }
    }
    Serial.println("WIFI AutoConfig Faild!" );
    return false;
}

void setup() {
  Serial.begin(115200);
  delay(100);
  if (!AutoConfig())
  {
      SmartConfig();
  }
}

void loop() {

}
```

FAQ

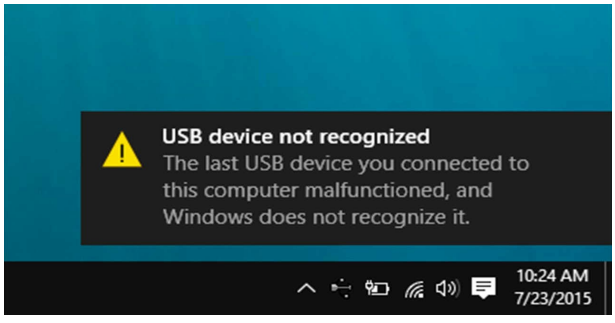
Burning error

Reason

- If the delay is too short or no delay is added in the loop, it will cause the programming to time out.

```
A fatal error occurred: Timed out waiting for packet header
A fatal error occurred: Timed out waiting for packet header
```

- Calling some functions incorrectly can cause the computer to fail to recognize the USB.



Resolutions

- Connect pin 9 to GND and pull down, power on again, and burn again

Serial port does not print

Resolutions

- Check if USB CDC is in Enable state
- Use other serial debugging assistants to view print information

DFR0222 X-Board Relay

DFR0017 Relay Module Arduino Compatible

DFR0289 RLY-8-POE Relay Controller

DFR0290 RLY-8-RS485 8 Relay Controller

DFR0144 Relay Shield for Arduino V2.1

DFR0473 Gravity Digital Relay Module Arduino & Raspberry Pi Compatible

For any questions, advice or cool ideas to share, please visit the DFRobot Forum.

More Documents

- ESP32-C3-mini-1 Datasheet
- Schematics Diagram
- Dimension Diagram

Get Beetle ESP32 C3 from DFRobot Store or DFRobot Distributor.

Turn to the Top