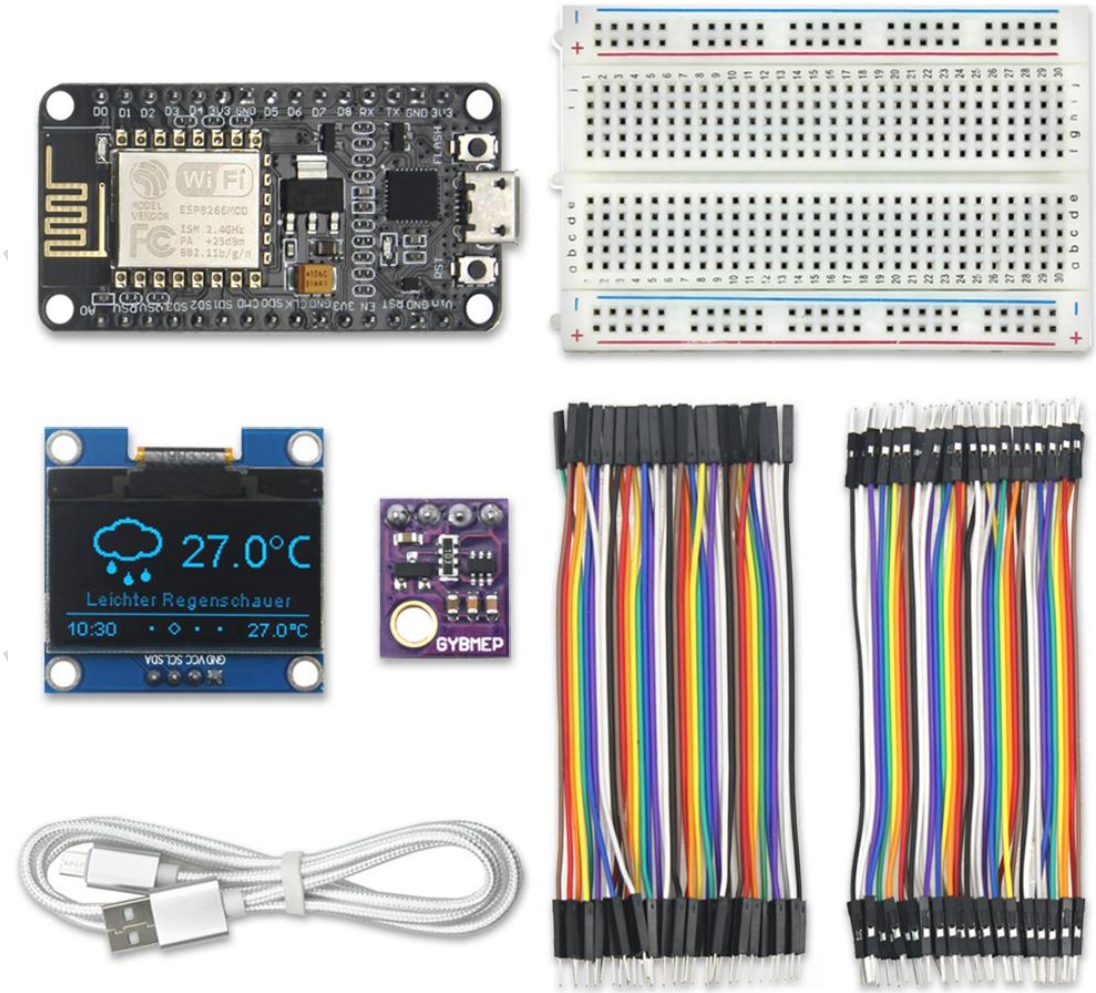
Preface

About Our Company

Located in Shenzhen, the Silicon Valley of China, KeeYees Technology Inc. is a big & professional Electronic Products Manufacturer and Seller, dedicated to open-source hardware research & development, production and marketing. All of our products comply with International Quality Standards and are very popular in a variety of different markets throughout of the world. KeeYees is your best choice in various electronic modules & components designed for customers of any level to learn Arduino and Raspberry Pi knowledge. In addition, we also sell products like 3D printer accessories, connectors and terminals kits, DIY parts and tools to support your work and design challenges from Home, School to Industrial applications!

US Amazon Store Homepage:
https://www.amazon.com/shops/A2K4DGCC72N9AG
UK Amazon Store Homepage:
https://www.amazon.co.uk/shops/A1F4U6XVWUBG1U
DE Amazon Store Homepage:
https://www.amazon.de/shops/A1F4U6XVWUBG1U
FR Amazon Store Homepage:
https://www.amazon.fr/shops/A1F4U6XVWUBG1U
IT Amazon Store Homepage:
https://www.amazon.it/shops/A1F4U6XVWUBG1U
ES Amazon Store Homepage:
https://www.amazon.es/shops/A1F4U6XVWUBG1U
JP Amazon Store Homepage:
https://www.amazon.co.jp/shops/A7NY3JX21TGU2

KeeYees 1.3" OLED display + ESP8266 NodeMCU + BME280 Weather Station Tutorial



Overview

This tutorial can realize the real-time update of weather data and time of the city via the wireless network. The newest BME280 module replaced the DHT11 and DHT22 module, can monitor not only the environment temperature and humidity, but also the air pressure accurately. Moreover, the 1.3" large OLED IIC display module replaced the small 0.96"screen can give all the data a clearer and larger display. This kit can not only monitor weather conditions of your local city, but also its surrounding environment. Combined with the ESP8266 NodeMCU and bme280 module, it can obtain the weather data whether you access the network or not. All in all, you can achieve the purpose whatever you want.

Part 1: Pin Connection

1. NodeMCU ESP8266<------->OLED

3.3V---VCC

GND---GND

D1---SCL

D2---SDA

2. NodeMCU ESP8266<------->BME280

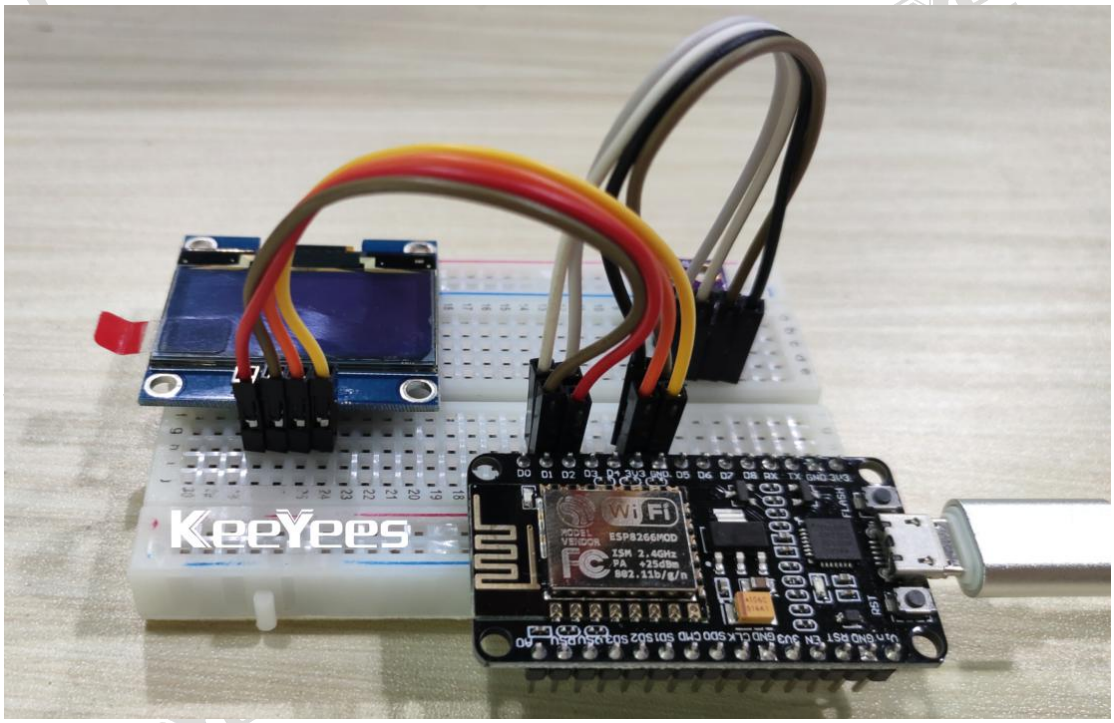
3.3V---VCC

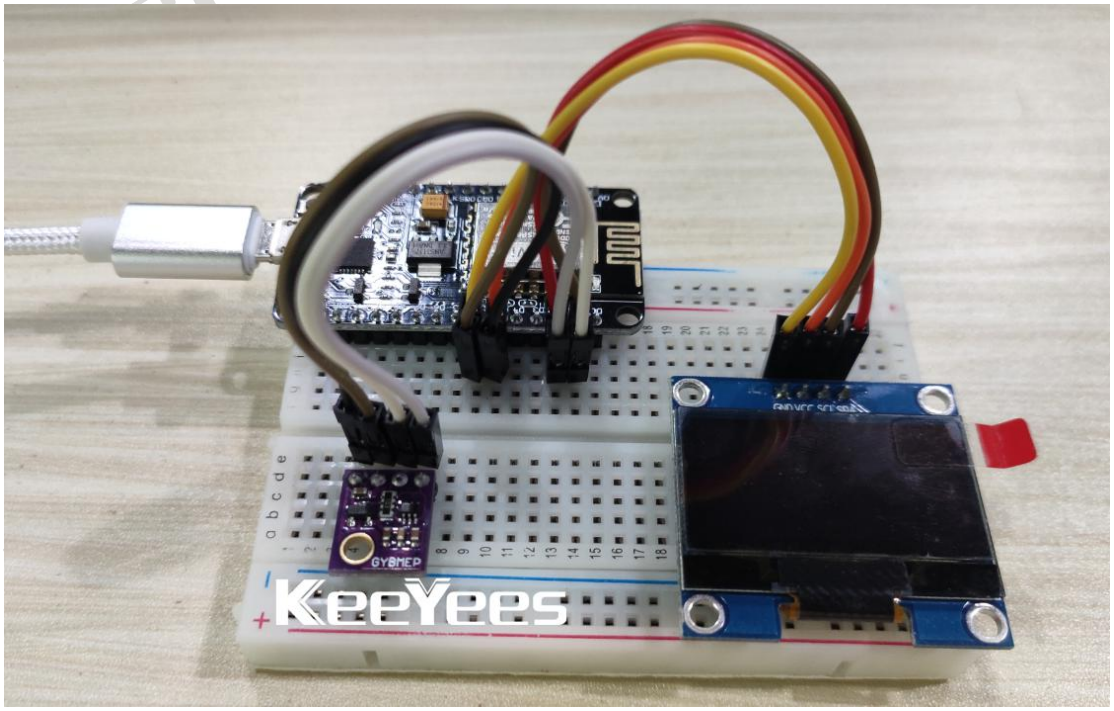
GND---GND

D1---SCL

D2---SDA

Connection Diagram





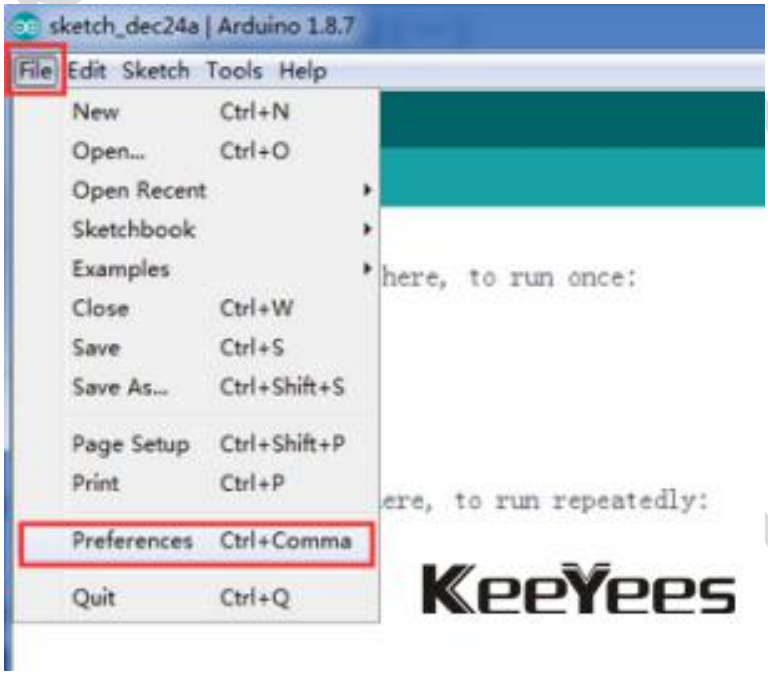
Part 2: Set up Development Environment

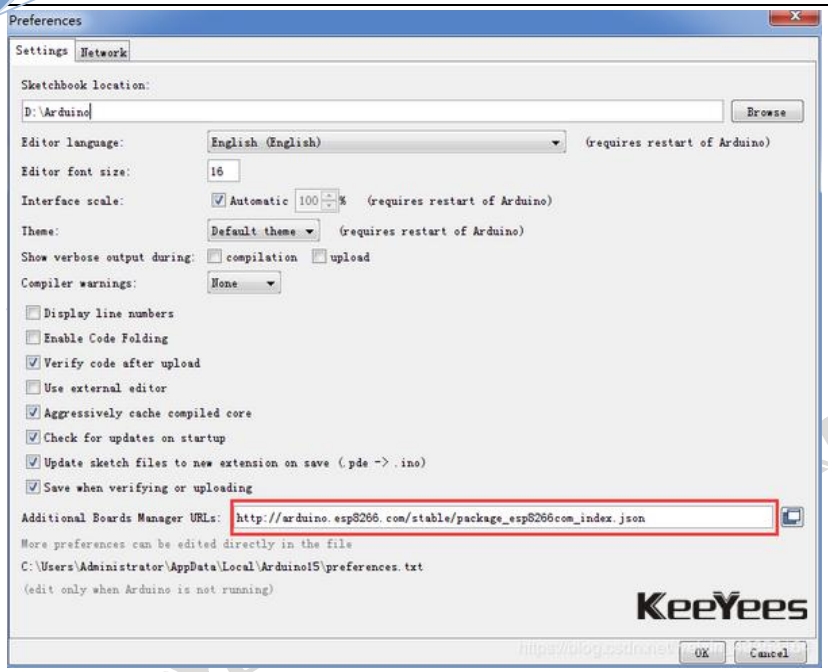
1. Download Arduino IDE 1.8.7

https://www.arduino.cc/en/Main/Software

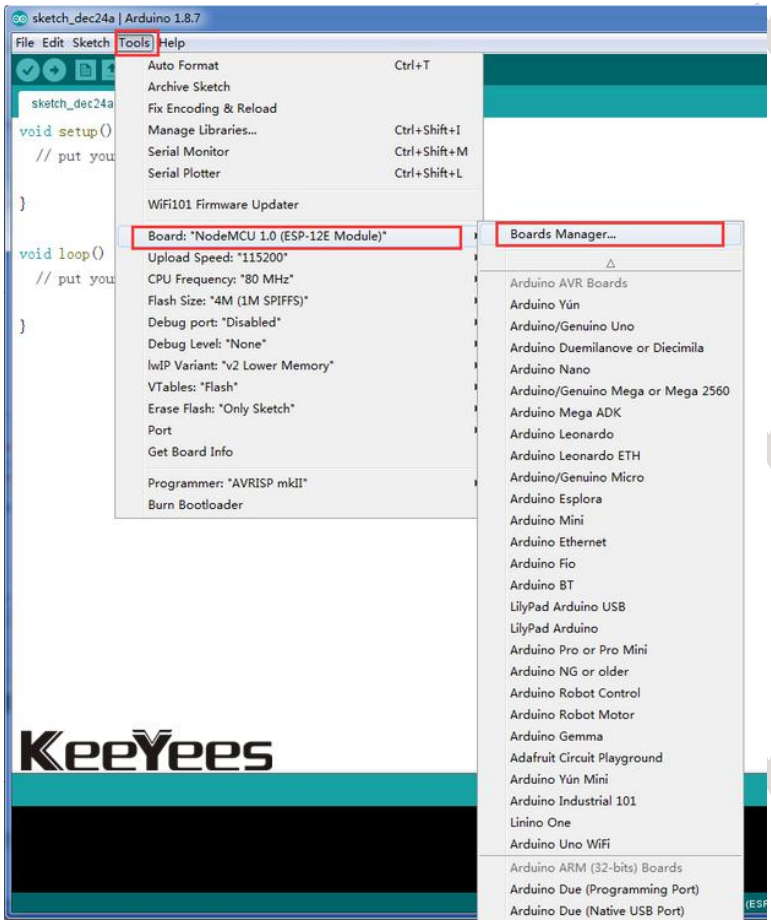
2. Add ESP8266 Development Board and Driver File

Step 1: Open Arduino IDE, click file->Preferences, in the pop-up window " Additional Boards Manager URLs " input: http://arduino.esp8266.com/stable/package_esp8266com_index.json , then click OK.



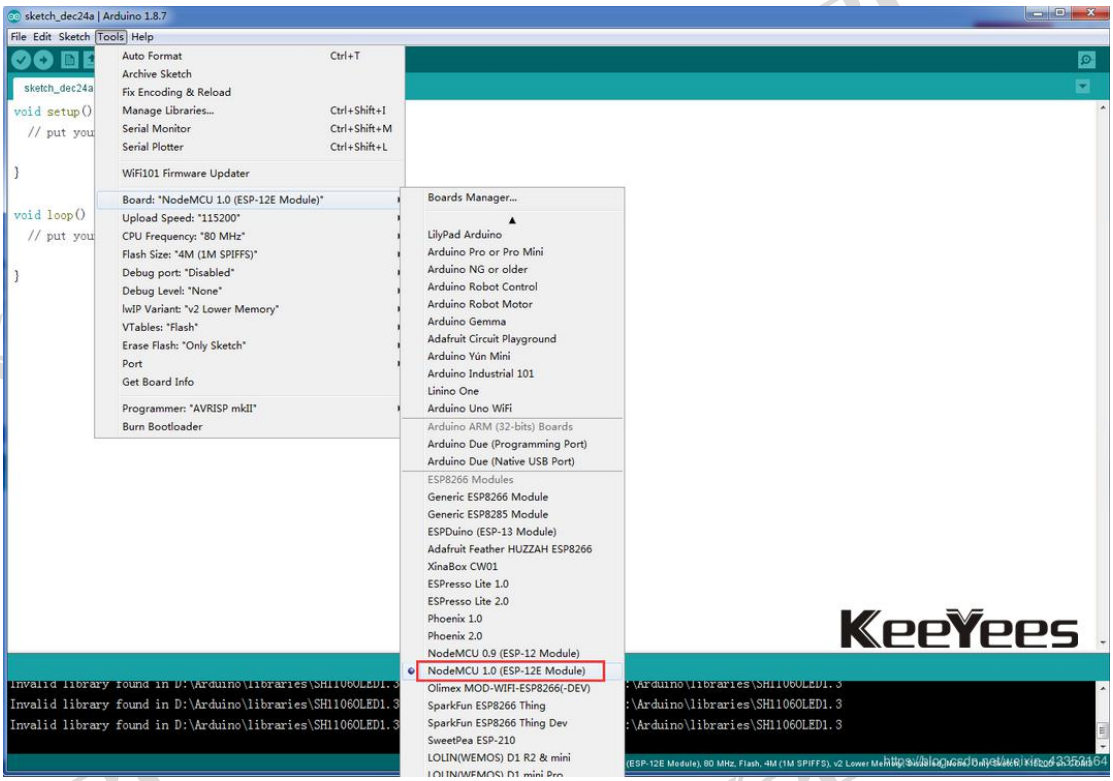


Step 2: Download ESP8266 development board

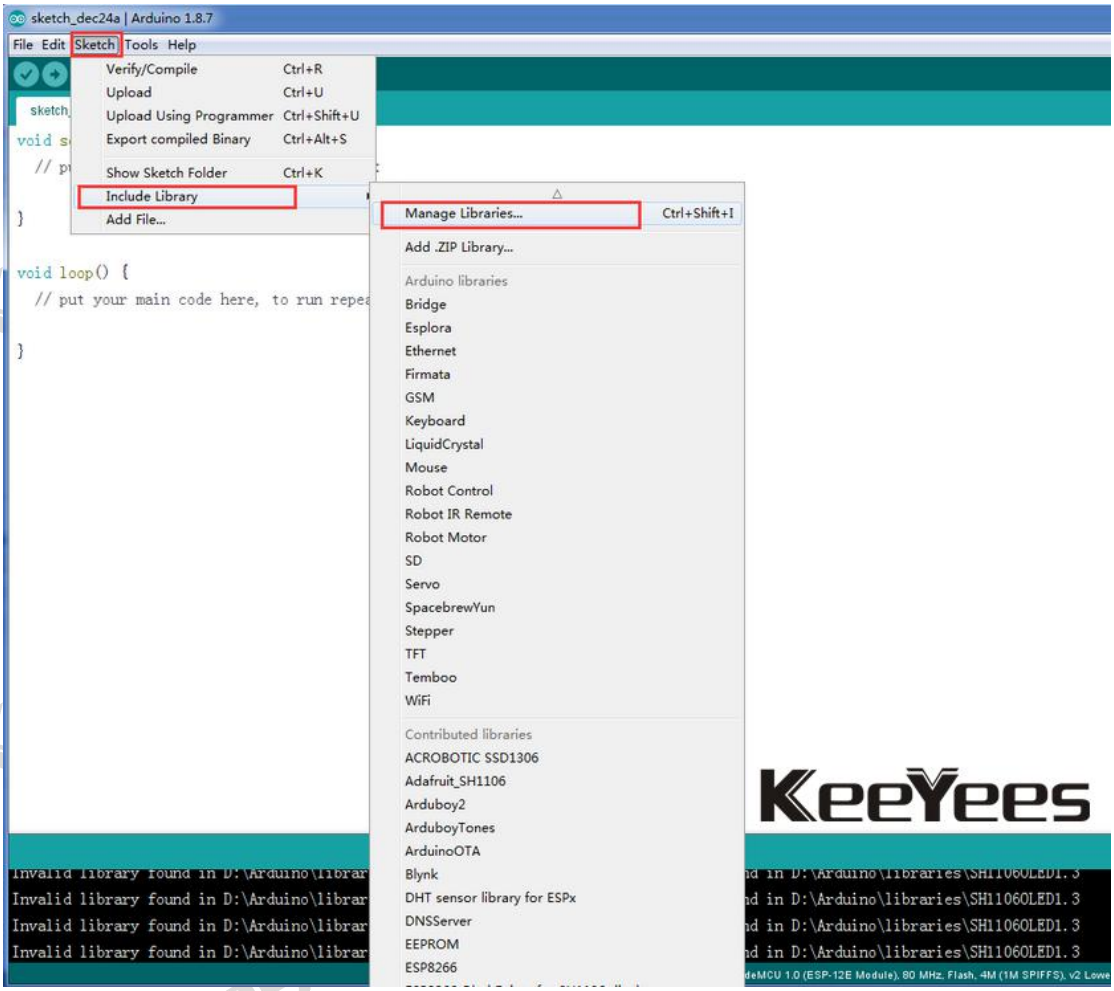


Step 3: Search for “esp8266” in the pop-up window and click “Install”.

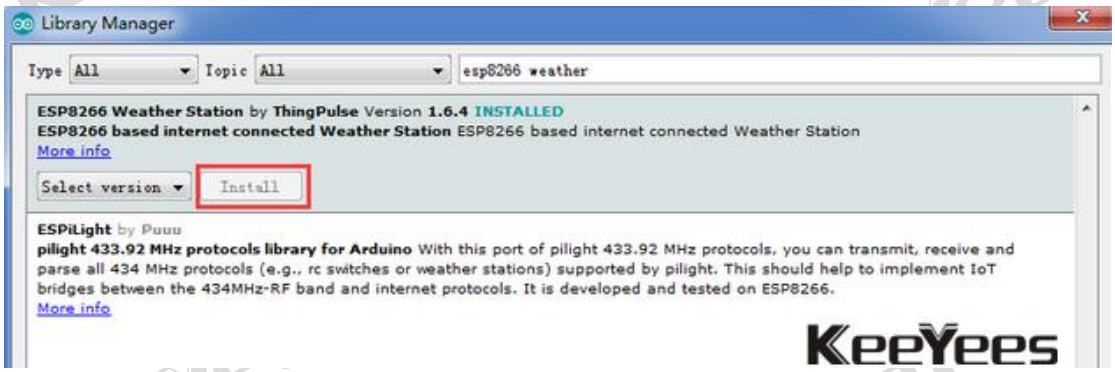
Step 4: After downloading, choose the correct module. If the list as shown in the figure below does not appear, it means that the download fails, so download again.



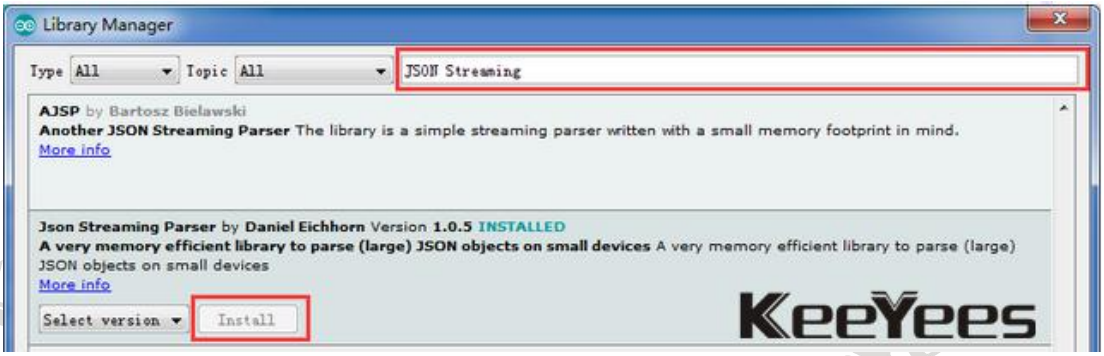
Step 5: To download the library file, click the options shown in the figure below.



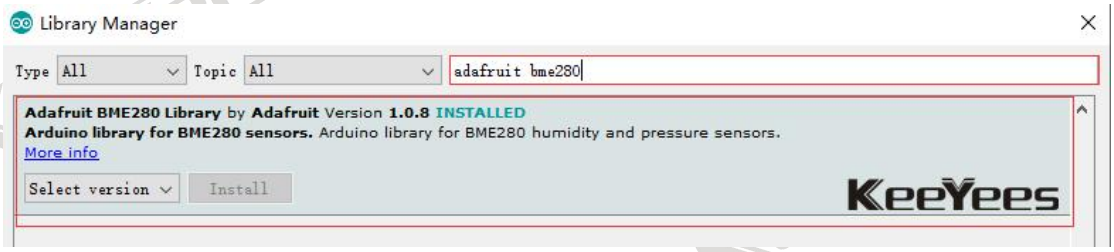
Step 6: Search for "esp8266 weather" and click "Insall".



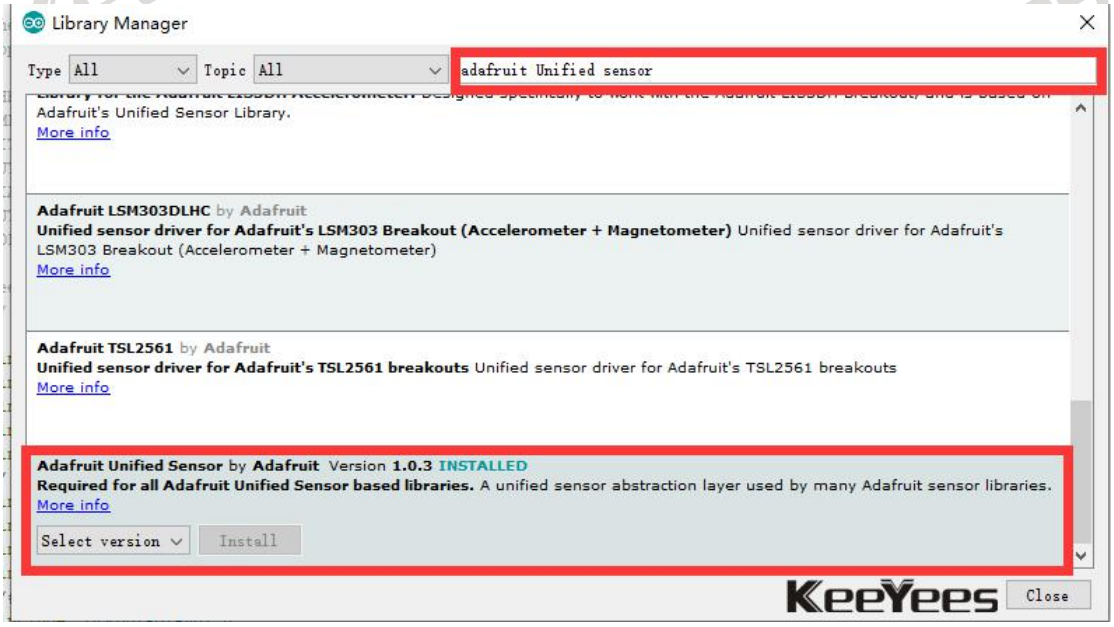
Step 7: Search for "JSON Streaming" and click "Insall".



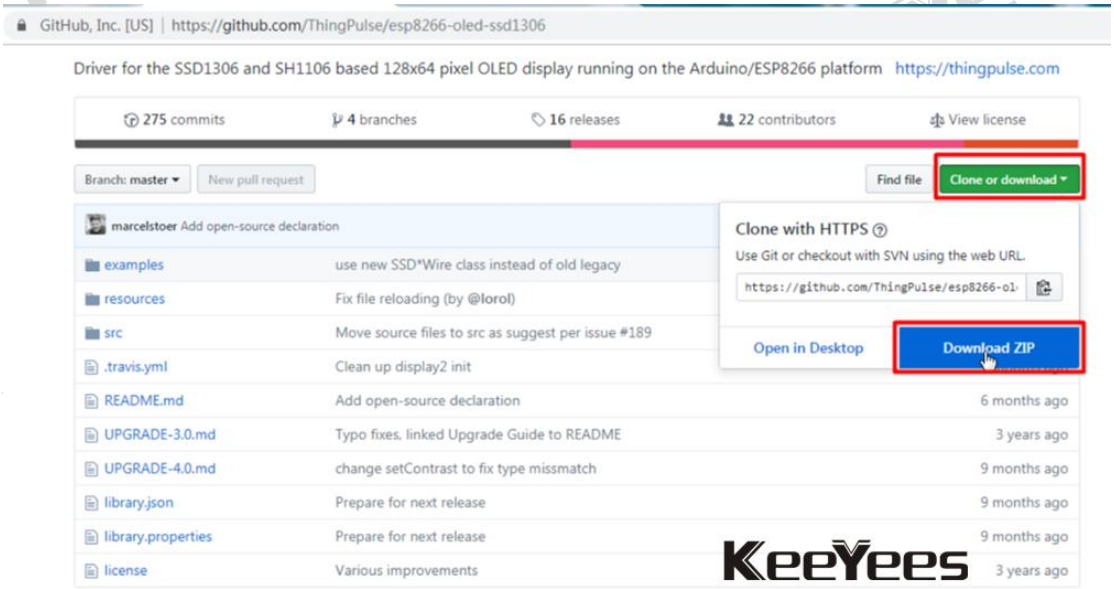
Step 8: Search for "adafruit bme280" to add the bme280 driver file and click "Insall".



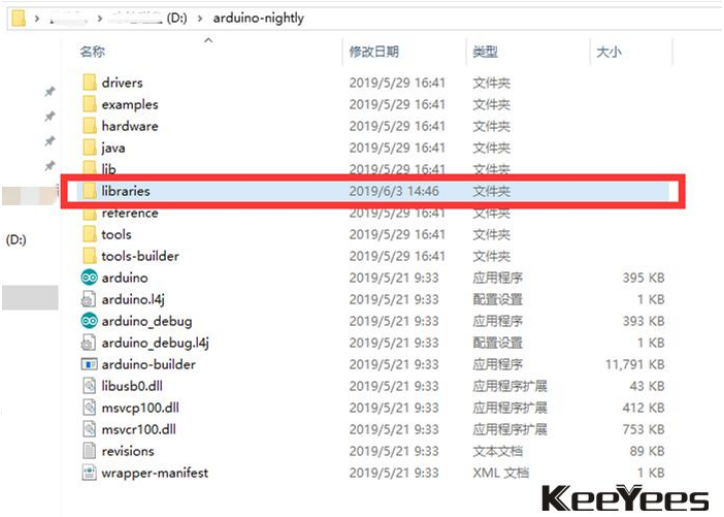
Step 9: Search for "adafruit Unified sensor" to add the adafruit sensor file and click "Insall".

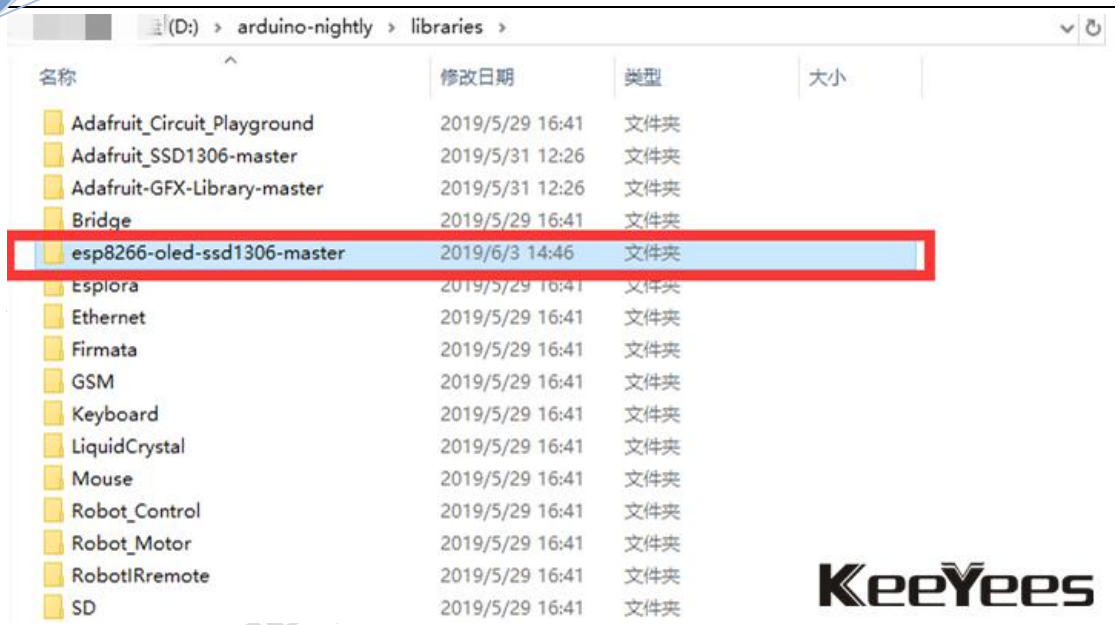


Step 10: Add the OLED driver file, go to the following URL https://github.com/ThingPulse/esp8266-oled-ssd1306, and then click Download ZIP.



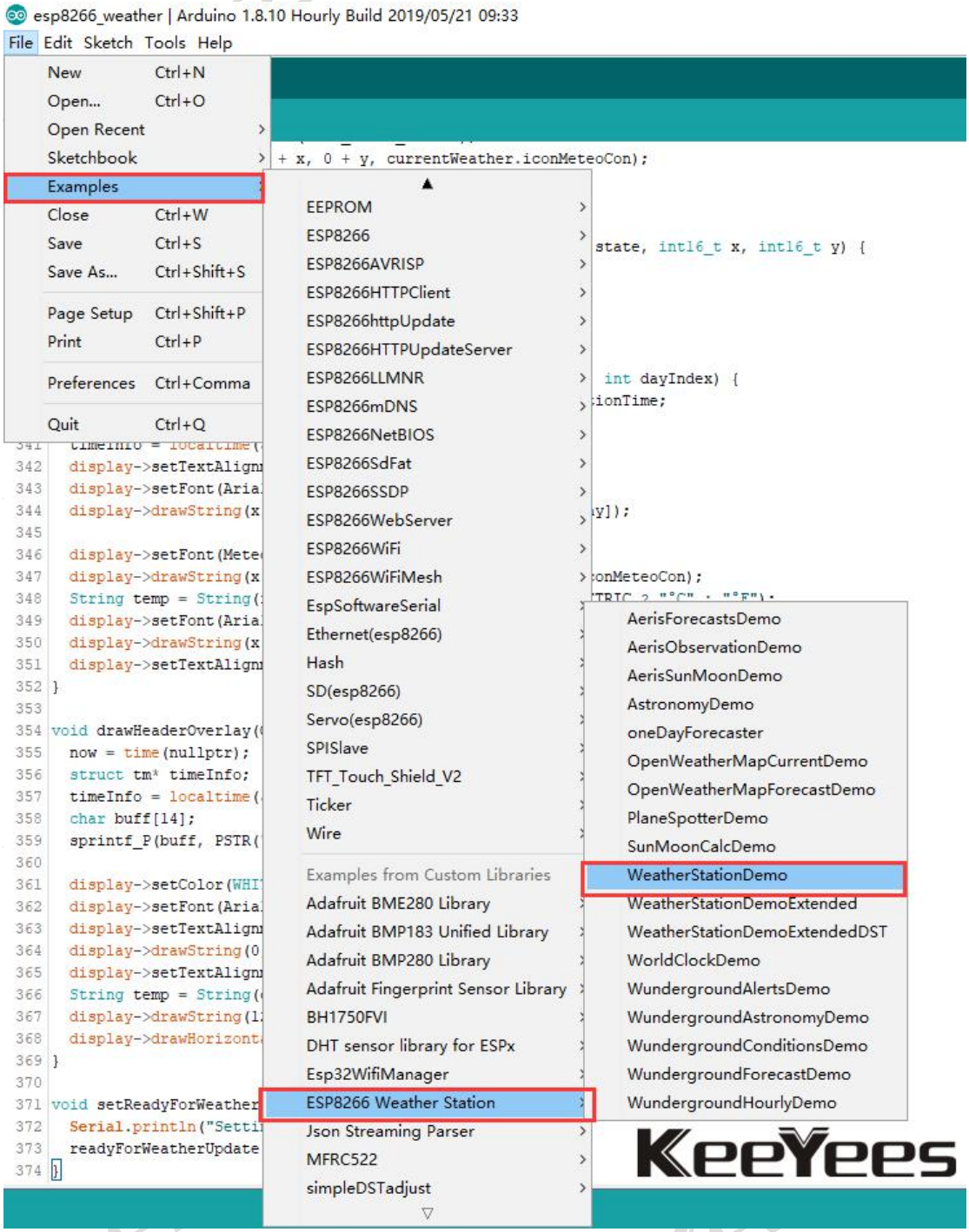
Step 11: Unzip the downloaded files. Then copy the extracted files into the libraries folder under the Arduino IDE installation path.




(D:) › arduino-nightly › libraries ›
名称
修改日期
类型
大小
Adafruit_Circuit_Playground 2019/5/29 16:41 文件夹
Adafruit_SSD1306-master 2019/5/31 12:26 文件夹
Adafruit-GFX-Library-master 2019/5/31 12:26 文件夹
Bridge 2019/5/29 16:41 文件夹
esp8266-oled-ssd1306-master 2019/6/3 14:46 文件夹
Esplora 2019/5/29 16:41 文件夹
Ethernet 2019/5/29 16:41 文件夹
Firmata 2019/5/29 16:41 文件夹
GSM 2019/5/29 16:41 文件夹
Keyboard 2019/5/29 16:41 文件夹
LiquidCrystal 2019/5/29 16:41 文件夹
Mouse 2019/5/29 16:41 文件夹
Robot_Control 2019/5/29 16:41 文件夹
Robot_Motor 2019/5/29 16:41 文件夹
RobotIRremote 2019/5/29 16:41 文件夹
SD 2019/5/29 16:41 文件夹
KeeYees

Part 3: Add and Modify Code

1. Click the options shown in the figure below successively.



2. Replace all code in the WeatherStationDemo with the following code:

```
/**The MIT License (MIT)

Copyright (c) 2018 by Daniel Eichhorn - ThingPulse

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.

See more at https://thingpulse.com
*/
#include <ESPWiFi.h>
#include <ESPHTTPClient.h>
#include <JsonListener.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_BME280.h>
// time
#include <time.h>                    // time() ctime()
#include <sys/time.h>                // struct timeval
#include <coredecls.h>               // settimeofday_cb()
//#include "SSD1306Wire.h"
```

```
#include "SH1106Wire.h"
#include "OLEDDisplayUi.h"
#include <Wire.h>
#include "OpenWeatherMapCurrent.h"
#include "OpenWeatherMapForecast.h"
#include "WeatherStationFonts.h"
#include "WeatherStationImages.h"


// Create the Lightsensor instance
#define BME_SCK 13
#define BME_MISO 12
#define BME_MOSI 11
#define BME_CS 10
#define SEALEVELPRESSURE_HPA (1013.25)
Adafruit_BME280 bme; // I2C
//DHTesp dht;
/**************************
 * Begin Settings
 **************************/

// WIFI
const char* WIFI_SSID = "BAN";
const char* WIFI_PWD = "chenyiwei";

String humi1;
String temp1;
#define TZ              7       // (utc+) TZ in hours
#define DST_MN          60      // use 60mn for summer time in some
countries

// Setup
const int UPDATE_INTERVAL_SECS = 10 * 60; // Update every 20 minutes
unsigned long delayTime;
// Display Settings
const int I2C_DISPLAY_ADDRESS = 0x3c;
#if defined(ESP8266)
const int SDA_PIN = D2;
const int SDC_PIN = D1;
const int DH1=D5;
#else
const int SDA_PIN = 4; //D3;
const int SDC_PIN = 5; //D4;
```

```
const int DH1=14;
#endif
// OpenWeatherMap Settings
// Sign up here to get an API key:
// https://docs.thingpulse.com/how-tos/openweathermap-key/
String OPEN_WEATHER_MAP_APP_ID =
"02a19f4506b3008018c8f690e62db526";
/*
Go to https://openweathermap.org/find?q= and search for a location. Go through the
result set and select the entry closest to the actual location you want to display
data for. It'll be a URL like https://openweathermap.org/city/2657896. The number
at the end is what you assign to the constant below.
  */
String OPEN_WEATHER_MAP_LOCATION_ID = "1795565";

// Pick a language code from this list:
// Arabic - ar, Bulgarian - bg, Catalan - ca, Czech - cz, German - de, Greek - el,
// English - en, Persian (Farsi) - fa, Finnish - fi, French - fr, Galician - gl,
// Croatian - hr, Hungarian - hu, Italian - it, Japanese - ja, Korean - kr,
// Latvian - la, Lithuanian - lt, Macedonian - mk, Dutch - nl, Polish - pl,
// Portuguese - pt, Romanian - ro, Russian - ru, Swedish - se, Slovak - sk,
// Slovenian - sl, Spanish - es, Turkish - tr, Ukrainian - ua, Vietnamese - vi,
// Chinese Simplified - zh_cn, Chinese Traditional - zh_tw.
String OPEN_WEATHER_MAP_LANGUAGE = "de";
const uint8_t MAX_FORECASTS = 4;

const boolean IS_METRIC = true;

// Adjust according to your language
const String WDAY_NAMES[] = {"SUN", "MON", "TUE", "WED", "THU", "FRI",
"SAT"};
const String MONTH_NAMES[] = {"JAN", "FEB", "MAR", "APR", "MAY", "JUN",
"JUL", "AUG", "SEP", "OCT", "NOV", "DEC"};

/***************************
  * End Settings
  **************************/
  // Initialize the oled display for address 0x3c
  // sda-pin=14 and sdc-pin=12
  //SSD1306Wire display(I2C_DISPLAY_ADDRESS, SDA_PIN, SDC_PIN);
SH1106Wire display(I2C_DISPLAY_ADDRESS, SDA_PIN, SDC_PIN);
  OLEDDisplayUi ui( &display );
```

```
OpenWeatherMapCurrentData currentWeather;
OpenWeatherMapCurrent currentWeatherClient;

OpenWeatherMapForecastData forecasts[MAX_FORECASTS];
OpenWeatherMapForecast forecastClient;

#define TZ_MN              ((TZ)*60)
#define TZ_SEC             ((TZ)*3600)
#define DST_SEC            ((DST_MN)*60)
time_t now;

// flag changed in the ticker function every 10 minutes
bool readyForWeatherUpdate = false;

String lastUpdate = "--";

long timeSinceLastWUpdate = 0;

//declaring prototypes
void drawProgress(OLEDDisplay *display, int percentage, String label);
void updateData(OLEDDisplay *display);
void drawBME(OLEDDisplay *display,OLEDDisplayUiState* state,int16_t x,
int16_t y);
void drawDateTime(OLEDDisplay *display, OLEDDisplayUiState* state, int16_t x,
int16_t y);
void drawCurrentWeather(OLEDDisplay *display, OLEDDisplayUiState* state,
int16_t x, int16_t y);
void drawForecast(OLEDDisplay *display, OLEDDisplayUiState* state, int16_t x,
int16_t y);
void drawForecastDetails(OLEDDisplay *display, int x, int y, int dayIndex);
void drawHeaderOverlay(OLEDDisplay *display, OLEDDisplayUiState* state);
void setReadyForWeatherUpdate();


// Add frames
// this array keeps function pointers to all frames
// frames are the single views that slide from right to left
FrameCallback frames[] = { drawDateTime, drawCurrentWeather, drawForecast,
drawBME};
int numberOfFrames = 4;

OverlayCallback overlays[] = { drawHeaderOverlay };
int numberOfOverlays = 1;
```

```
void setup() {
  Serial.begin(115200);
  Serial.println();
  Serial.println(F("BME280 test"));
  bool status;
  status = bme.begin(0x76);
    if (!status) {
          Serial.println("Could not find a valid BME280 sensor, check wiring!");
          while (1);
          Serial.println("-- Default Test --");
    delayTime = 1000;

    Serial.println();
    }
  // initialize dispaly
  display.init();
  display.clear();
  display.display();

  //display.flipScreenVertically();
  display.setFont(ArialMT_Plain_10);
  display.setTextAlignment(TEXT_ALIGN_CENTER);
  display.setContrast(255);
  WiFi.begin(WIFI_SSID, WIFI_PWD);
  int counter = 0;
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
    display.clear();
    display.drawString(64, 10, "Connecting to WiFi");
    display.drawXbm(40, 30, 8, 8, counter % 3 == 0 ? activeSymbole :
inactiveSymbole);
    display.drawXbm(54, 30, 8, 8, counter % 3 == 1 ? activeSymbole :
inactiveSymbole);
    display.drawXbm(68, 30, 8, 8, counter % 3 == 2 ? activeSymbole :
inactiveSymbole);
    display.display();

    counter++;
  }
  // Get time from network time service
  configTime(TZ_SEC, DST_SEC, "pool.ntp.org");
```

```
  ui.setTargetFPS(30);

  ui.setActiveSymbol(activeSymbole);
  ui.setInactiveSymbol(inactiveSymbole);

  // You can change this to
  // TOP, LEFT, BOTTOM, RIGHT
  ui.setIndicatorPosition(BOTTOM);

  // Defines where the first frame is located in the bar.
  ui.setIndicatorDirection(LEFT_RIGHT);

  // You can change the transition that is used
  // SLIDE_LEFT, SLIDE_RIGHT, SLIDE_TOP, SLIDE_DOWN
  ui.setFrameAnimation(SLIDE_LEFT);

  ui.setFrames(frames, numberOfFrames);

  ui.setOverlays(overlays, numberOfOverlays);

  // Inital UI takes care of initalising the display too.
  ui.init();

  Serial.println("");

  updateData(&display);

}

void loop() {
  if (millis() - timeSinceLastWUpdate > (1000L*UPDATE_INTERVAL_SECS)) {
    setReadyForWeatherUpdate();
    timeSinceLastWUpdate = millis();
  }

  if (readyForWeatherUpdate && ui.getUiState()->frameState == FIXED) {
    updateData(&display);
  }

  int remainingTimeBudget = ui.update();

  if (remainingTimeBudget > 0) {
```

```
        // You can do some work here
        // Don't do stuff if you are below your
        // time budget.
        delay(remainingTimeBudget);
    }

}

void drawProgress(OLEDDisplay *display, int percentage, String label) {
    display->clear();
    display->setTextAlignment(TEXT_ALIGN_CENTER);
    display->setFont(ArialMT_Plain_10);
    display->drawString(64, 10, label);
    display->drawProgressBar(2, 28, 124, 10, percentage);
    display->display();
}

void updateData(OLEDDisplay *display) {
    drawProgress(display, 10, "Updating time...");
    drawProgress(display, 30, "Updating weather...");
    currentWeatherClient.setMetric(IS_METRIC);
    currentWeatherClient.setLanguage(OPEN_WEATHER_MAP_LANGUAGE);
    currentWeatherClient.updateCurrentById(&currentWeather,
OPEN_WEATHER_MAP_APP_ID, OPEN_WEATHER_MAP_LOCATION_ID);
    drawProgress(display, 50, "Updating forecasts...");
    forecastClient.setMetric(IS_METRIC);
    forecastClient.setLanguage(OPEN_WEATHER_MAP_LANGUAGE);
    uint8_t allowedHours[] = {12};
    forecastClient.setAllowedHours(allowedHours, sizeof(allowedHours));
    forecastClient.updateForecastsById(forecasts,  OPEN_WEATHER_MAP_APP_ID,
OPEN_WEATHER_MAP_LOCATION_ID, MAX_FORECASTS);

    readyForWeatherUpdate = false;
    drawProgress(display, 100, "Done...");
    delay(1000);
}

void    drawBME(OLEDDisplay    *display,OLEDDisplayUiState*    state,int16_t    x,
int16_t y){

  float temp1=bme.readTemperature();
    float pres1=bme.readPressure()/100.0F;
    float humi1=bme.readHumidity();
```

```
  delay(delayTime);
  display->setTextAlignment(TEXT_ALIGN_CENTER);
  display->setFont(ArialMT_Plain_16);
  String humi=(IS_METRIC ? "H:" : "H:")+String(humi1, 1)+(IS_METRIC ? "%" :
"%");
  display->drawString(64+x, y, humi);
  display->setTextAlignment(TEXT_ALIGN_CENTER);
  display->setFont(ArialMT_Plain_16);

  String temp=(IS_METRIC ? " T:" : "T:")+String(temp1, 1)+(IS_METRIC ? "°C" :

"°F");

  display->drawString(64+x, 15+y, temp);
  display->setTextAlignment(TEXT_ALIGN_CENTER);
  display->setFont(ArialMT_Plain_16);
  String pres=(IS_METRIC ? " P:" : "P:")+String(pres1, 1)+(IS_METRIC ? "hPa" :
"hPa");
  display->drawString(64+x, 30+y, pres);
  }

void drawDateTime(OLEDDisplay *display, OLEDDisplayUiState* state, int16_t x,
int16_t y) {
  now = time(nullptr);
  struct tm* timeInfo;
  timeInfo = localtime(&now);
  char buff[16];


  display->setTextAlignment(TEXT_ALIGN_CENTER);
  display->setFont(ArialMT_Plain_10);
  String date = WDAY_NAMES[timeInfo->tm_wday];

  sprintf_P(buff,  PSTR("%s,  %02d/%02d/%04d"),
WDAY_NAMES[timeInfo->tm_wday].c_str(),  timeInfo->tm_mday,
timeInfo->tm_mon+1, timeInfo->tm_year + 1900);
  display->drawString(64 + x, 5 + y, String(buff));
  display->setFont(ArialMT_Plain_24);

  sprintf_P(buff,  PSTR("%02d:%02d:%02d"),  timeInfo->tm_hour,
timeInfo->tm_min, timeInfo->tm_sec);
  display->drawString(64 + x, 15 + y, String(buff));
  display->setTextAlignment(TEXT_ALIGN_LEFT);
}
```

```
void drawCurrentWeather(OLEDDisplay *display, OLEDDisplayUiState* state, int16_t x, int16_t y) {
  display->setFont(ArialMT_Plain_10);
  display->setTextAlignment(TEXT_ALIGN_CENTER);
  display->drawString(64 + x, 38 + y, currentWeather.description);

  display->setFont(ArialMT_Plain_24);
  display->setTextAlignment(TEXT_ALIGN_LEFT);

  String temp = String(currentWeather.temp, 1) + (IS_METRIC ? "°C" : "°F");

  display->drawString(60 + x, 5 + y, temp);

  display->setFont(Meteocons_Plain_36);
  display->setTextAlignment(TEXT_ALIGN_CENTER);
  display->drawString(32 + x, 0 + y, currentWeather.iconMeteoCon);
}


void drawForecast(OLEDDisplay *display, OLEDDisplayUiState* state, int16_t x, int16_t y) {
  drawForecastDetails(display, x, y, 0);
  drawForecastDetails(display, x + 44, y, 1);
  drawForecastDetails(display, x + 88, y, 2);
}

void drawForecastDetails(OLEDDisplay *display, int x, int y, int dayIndex) {
  time_t observationTimestamp = forecasts[dayIndex].observationTime;
  struct tm* timeInfo;
  timeInfo = localtime(&observationTimestamp);
  display->setTextAlignment(TEXT_ALIGN_CENTER);
  display->setFont(ArialMT_Plain_10);
  display->drawString(x + 20, y, WDAY_NAMES[timeInfo->tm_wday]);

  display->setFont(Meteocons_Plain_21);
  display->drawString(x + 20, y + 12, forecasts[dayIndex].iconMeteoCon);

  String temp = String(forecasts[dayIndex].temp, 0) + (IS_METRIC ? "°C" : "°F");

  display->setFont(ArialMT_Plain_10);
  display->drawString(x + 20, y + 34, temp);
  display->setTextAlignment(TEXT_ALIGN_LEFT);
}
```

```
void drawHeaderOverlay(OLEDDisplay *display, OLEDDisplayUiState* state) {
  now = time(nullptr);
  struct tm* timeInfo;
  timeInfo = localtime(&now);
  char buff[14];
  sprintf_P(buff, PSTR("%02d:%02d"), timeInfo->tm_hour, timeInfo->tm_min);

  display->setColor(WHITE);
  display->setFont(ArialMT_Plain_10);
  display->setTextAlignment(TEXT_ALIGN_LEFT);
  display->drawString(0, 54, String(buff));
  display->setTextAlignment(TEXT_ALIGN_RIGHT);

  String temp = String(currentWeather.temp, 1) + (IS_METRIC ? "°C" : "°F");

  display->drawString(128, 54, temp);
  display->drawHorizontalLine(0, 52, 128);
}

void setReadyForWeatherUpdate() {
  Serial.println("Setting readyForUpdate to true");
  readyForWeatherUpdate = true;
}
```

3. Change the **** in the code to your wireless network name and password you want to connect to.

File Edit Sketch Tools Help

esp8266_weather §

```
#include <time.h>                    // time() ctime()
#include <sys/time.h>                // struct timeval
#include <coredecls.h>               // settimeofday_cb()
//#include "SSD1306Wire.h"
#include "SH1106Wire.h"
#include "OLEDDisplayUi.h"
#include <Wire.h>
#include "OpenWeatherMapCurrent.h"
#include "OpenWeatherMapForecast.h"
#include "WeatherStationFonts.h"
#include "WeatherStationImages.h"
#include "DHTesp.h"

#include <BH1750FVI.h>

// Create the Lightsensor instance
BH1750FVI LightSensor(BH1750FVI::k_DevModeContLowRes);
#define BME_SCK 13
#define BME_MISO 12
#define BME_MOSI 11
#define BME_CS 10
#define SEALEVELPRESSURE_HPA (1013.25)
Adafruit_BME280 bme; // I2C
DHTesp dht;
/***************************
 * Begin Settings
 ***************************/

// WIFI
const char* WIFI_SSID = "******";
const char* WIFI_PWD = "********";

#define TZ              7       // (utc+) TZ in hours
#define DST_MN          60      // use 60mn for summer time in some countries

// Setup
const int UPDATE_INTERVAL_SECS = 10 * 60; // Update every 20 minutes
unsigned long delayTime;
// Display Settings
const int I2C_DISPLAY_ADDRESS = 0x3c;
#if defined(ESP8266)
const int SDA_PIN = D2;
const int SDC_PIN = D1;
const int DH1=D5;
#else
const int SDA_PIN = 4; //D3;
const int SDC_PIN = 5; //D4;
```

4. To get the API, click the url in the red box below, enter the web page and register an account with email, you can get the API for free, paste the obtained API string into the double quotation marks in the red box below.

File Edit Sketch Tools Help

esp8266_weather §

```
#define DST_MN          60      // use 60mn for summer time in some countries

// Setup
const int UPDATE_INTERVAL_SECS = 10 * 60; // Update every 20 minutes
unsigned long delayTime;
// Display Settings
const int I2C_DISPLAY_ADDRESS = 0x3c;
#if defined(ESP8266)
const int SDA_PIN = D2;
const int SDC_PIN = D1;
const int DH1=D5;
#else
const int SDA_PIN = 4; //D3;
const int SDC_PIN = 5; //D4;
const int DH1=14;
#endif
// OpenWeatherMap Settings
// Sign up here to get an API key:
// https://docs.thingpulse.com/how-tos/openweathermap-key/
String OPEN_WEATHER_MAP_APP_ID = "XXX";
/*
Go to https://openweathermap.org/find?q= and search for a location. Go through the
result set and select the entry closest to the actual location you want to display
data for. It'll be a URL like https://openweathermap.org/city/2657896. The number
at the end is what you assign to the constant below.
 */
String OPEN_WEATHER_MAP_LOCATION_ID = "*******";

// Pick a language code from this list:
// Arabic - ar, Bulgarian - bg, Catalan - ca, Czech - cz, German - de, Greek - el,
// English - en, Persian (Farsi) - fa, Finnish - fi, French - fr, Galician - gl,
// Croatian - hr, Hungarian - hu, Italian - it, Japanese - ja, Korean - kr,
// Latvian - la, Lithuanian - lt, Macedonian - mk, Dutch - nl, Polish - pl,
// Portuguese - pt, Romanian - ro, Russian - ru, Swedish - se, Slovak - sk,
// Slovenian - sl, Spanish - es, Turkish - tr, Ukrainian - ua, Vietnamese - vi,
// Chinese Simplified - zh_cn, Chinese Traditional - zh_tw.
String OPEN_WEATHER_MAP_LANGUAGE = "de";
const uint8_t MAX_FORECASTS = 4;

const boolean IS_METRIC = true;
```

Home Guides How-tos Hardware specs

How-tos
- Create OpenWeatherMap API key
- Create Wunderground API key
- Install drivers for USB-to-Serial
- Prepare Arduino IDE for ESP8266

Create OpenWeatherMap API key

As your device will be displaying data from OpenWeatherMap you need an "API key" from them. It uniquely ties requests from your device(s) to your account and ensures that the number of requests remains within your allotted quota.

- Go to https://docs.thingpulse.com/go/create-openweathermap-key
- Take note of the features in the "Free" column. By using the free plan you are limited to 60 calls per minute.
- In the "Free" column click on "Get API key and Start".

OpenWeatherMap Weather Maps API Price Partners Stations Widgets News About

Price

Home / Price

Subscribe to current weather, forecasts, and historical data collections and enjoy our fast simple API!

Current weather and forecasts collection

	Free	Startup	Developer	Professional	Enterprise
Price Price is fixed, no other hidden costs.	Free	40 USD / month	180 USD / month	470 USD / month	2,000 USD / month

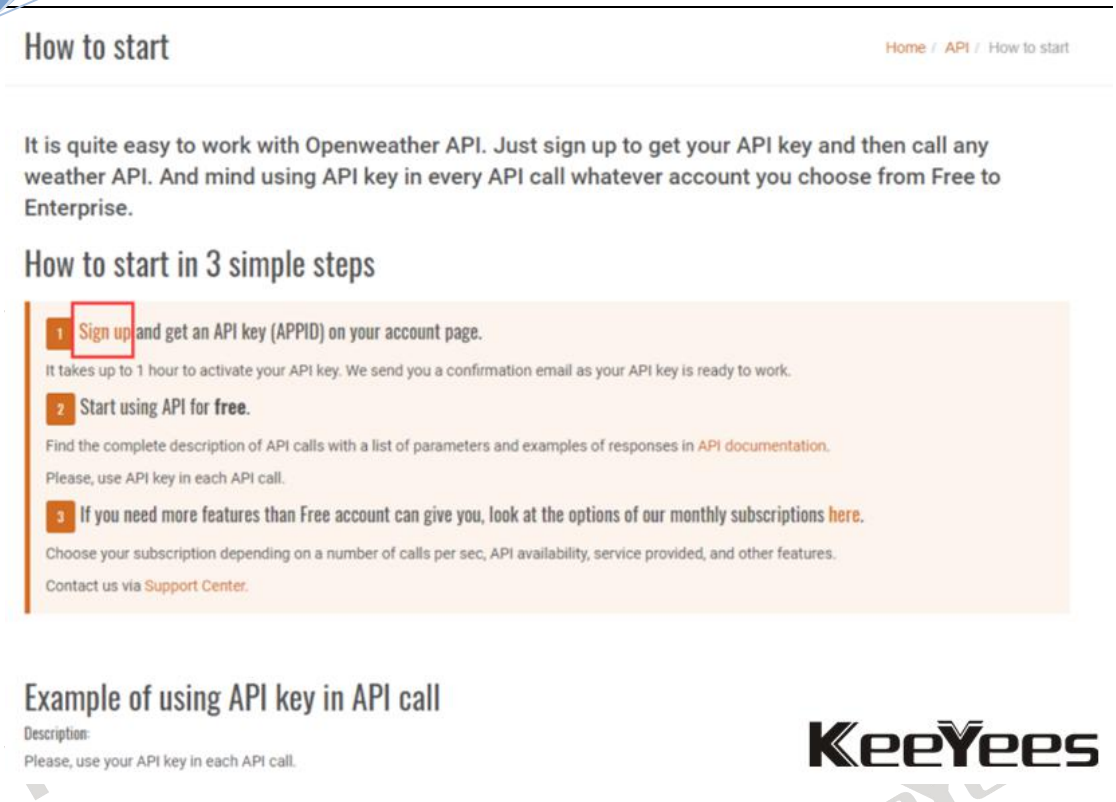
Subscribe to current weather, forecasts, and historical data collections and enjoy our fast simple API!
Please, read How to buy before you subscribe.

Please note that Current weather & forecasts collection and Historical weather collection are different products and have separate subscriptions.

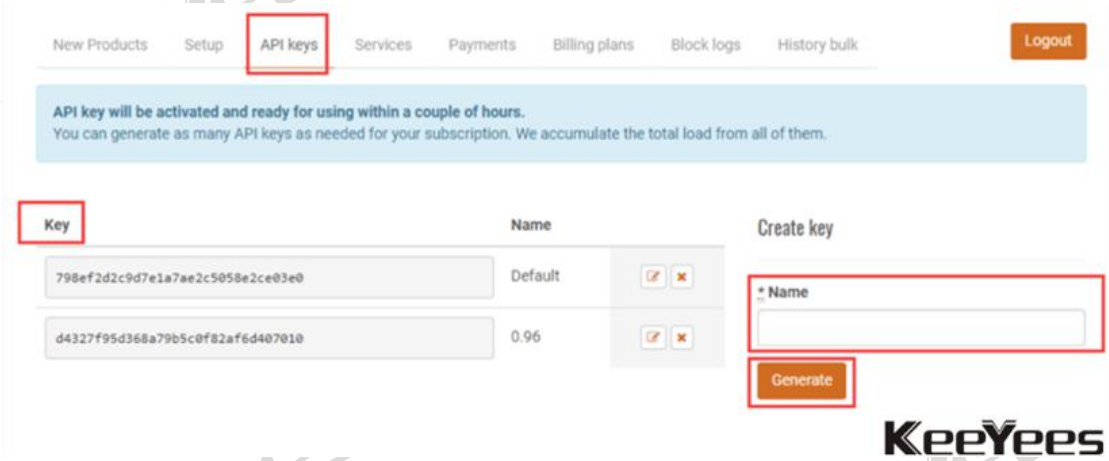
Current weather and forecasts collection

	Free	Startup	Developer	Professional	Enterprise
Price Price is fixed, no other hidden costs (VAT is not included)	Free	40 USD / month	180 USD / month	470 USD / month	2,000 USD / month
Subscribe	Get API key and Start	Subscribe	Subscribe	Subscribe	Subscribe
Calls per minute (no more than)	60	600	3,000	30,000	200,000
Current weather API	✓	✓	✓	✓	✓
5 days/3 hour forecast API	✓	✓	✓	✓	✓
16 days/daily forecast API	-	✓	✓	✓	✓
Weather maps 2.0: Current, Forecast, Historical NEW	-	-	✓	✓	✓
Relief maps NEW	-	-	✓	✓	✓
Weather maps 1.0	✓	✓	✓	✓	✓
Bulk download	-	-	-	✓	✓
UV index	✓	✓	✓	✓	✓
Weather alerts	✓	✓	✓	✓	✓

If you don't have an account, click Sign up to register.



5. Enter the following website, click API, enter a name in the red box on the right, and click "Generate" to generate an API KEY https://home.openweathermap.org/api_keys



6. Paste the generated Key into the code.

esp8266_weather §

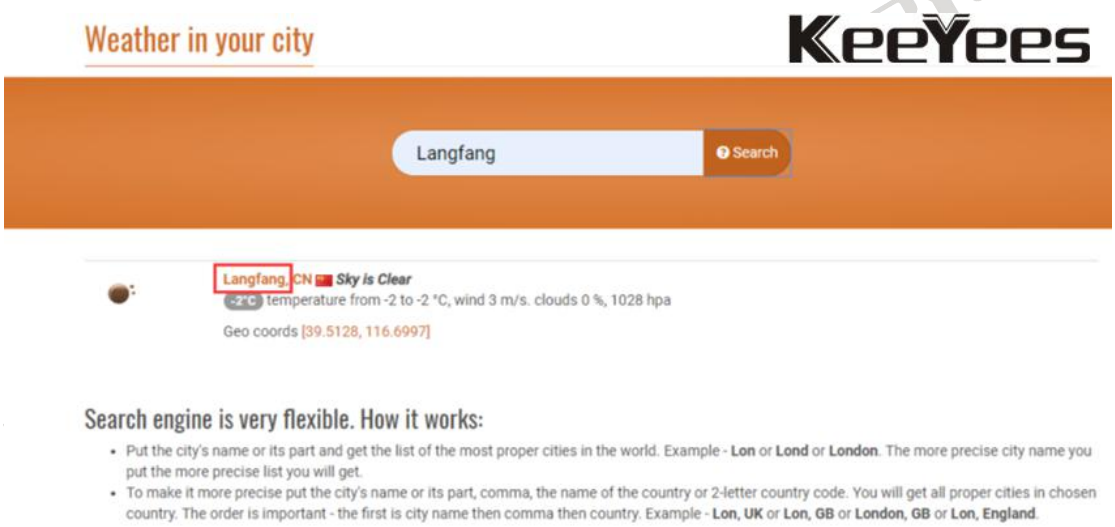
```
#define DST_MN          60      // use 60mn for summer time in some countries

// Setup
const int UPDATE_INTERVAL_SECS = 10 * 60; // Update every 20 minutes
unsigned long delayTime;
// Display Settings
const int I2C_DISPLAY_ADDRESS = 0x3c;
#if defined(ESP8266)
const int SDA_PIN = D2;
const int SDC_PIN = D1;
const int DH1=D5;
#else
const int SDA_PIN = 4; //D3;
const int SDC_PIN = 5; //D4;
const int DH1=14;
#endif
// OpenWeatherMap Settings
// Sign up here to get an API key:
// https://docs.thingpulse.com/how-tos/openweathermap-key/
String OPEN_WEATHER_MAP_APP_ID = "02a19f4506b3008018c8f690e62db526";

Go to https://openweathermap.org/find?q= and search for a location. Go through the
result set and select the entry closest to the actual location you want to display
data for. It'll be a URL like https://openweathermap.org/city/2657896. The number
at the end is what you assign to the constant below.
 */
```

7. Click the link in the first red box below and paste the obtained city code into the second red box.

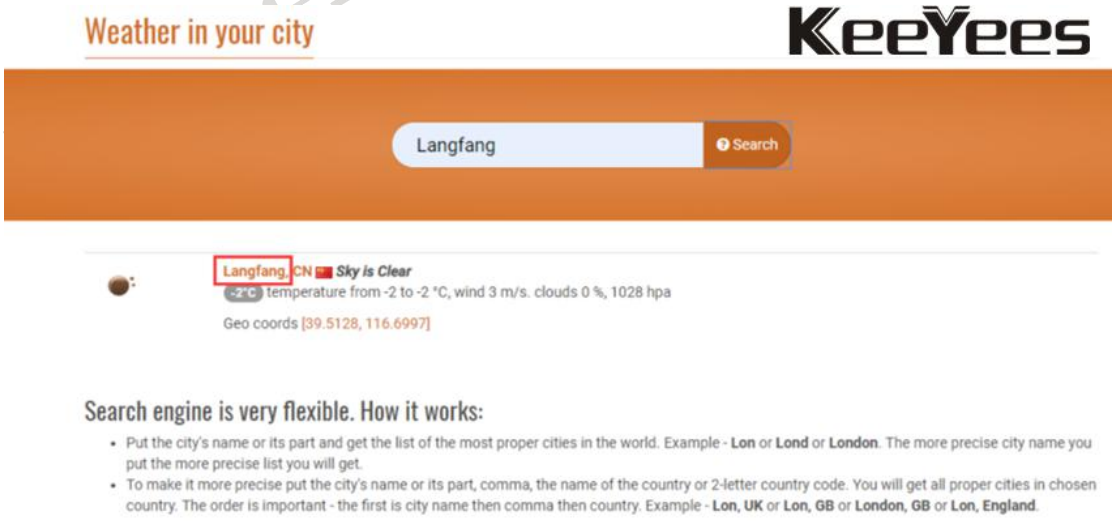
esp8266_weather §

```
#define DST_MN          60      // use 60mn for summer time in some countries

// Setup
const int UPDATE_INTERVAL_SECS = 10 * 60; // Update every 20 minutes
unsigned long delayTime;
// Display Settings
const int I2C_DISPLAY_ADDRESS = 0x3c;
#if defined(ESP8266)
const int SDA_PIN = D2;
const int SDC_PIN = D1;
const int DH1=D5;
#else
const int SDA_PIN = 4; //D3;
const int SDC_PIN = 5; //D4;
const int DH1=14;
#endif
// OpenWeatherMap Settings
// Sign up here to get an API key:
// https://docs.thingpulse.com/how-tos/openweathermap-key/
String OPEN_WEATHER_MAP_APP_ID = "02a19f4506b3008018c8f690e62db526";
/*
Go to https://openweathermap.org/find?q= and search for a location. Go through the
result set and select the entry closest to the actual location you want to display
data for. It'll be a URL like https://openweathermap.org/city/2657896. The number
at the end is what you assign to the constant below.
*/
String OPEN_WEATHER_MAP_LOCATION_ID = "1795565";

// Pick a language code from this list:
// Arabic - ar, Bulgarian - bg, Catalan - ca, Czech - cz, German - de, Greek - el,
// English - en, Persian (Farsi) - fa, Finnish - fi, French - fr, Galician - gl,
// Croatian - hr, Hungarian - hu, Italian - it, Japanese - ja, Korean - kr,
// Latvian - la, Lithuanian - lt, Macedonian - mk, Dutch - nl, Polish - pl,
// Portuguese - pt, Romanian - ro, Russian - ru, Swedish - se, Slovak - sk,
// Slovenian - sl, Spanish - es, Turkish - tr, Ukrainian - ua, Vietnamese - vi,
// Chinese Simplified - zh_cn, Chinese Traditional - zh_tw.
String OPEN_WEATHER_MAP_LANGUAGE = "de";
const uint8_t MAX_FORECASTS = 4;

const boolean IS_METRIC = true;
```

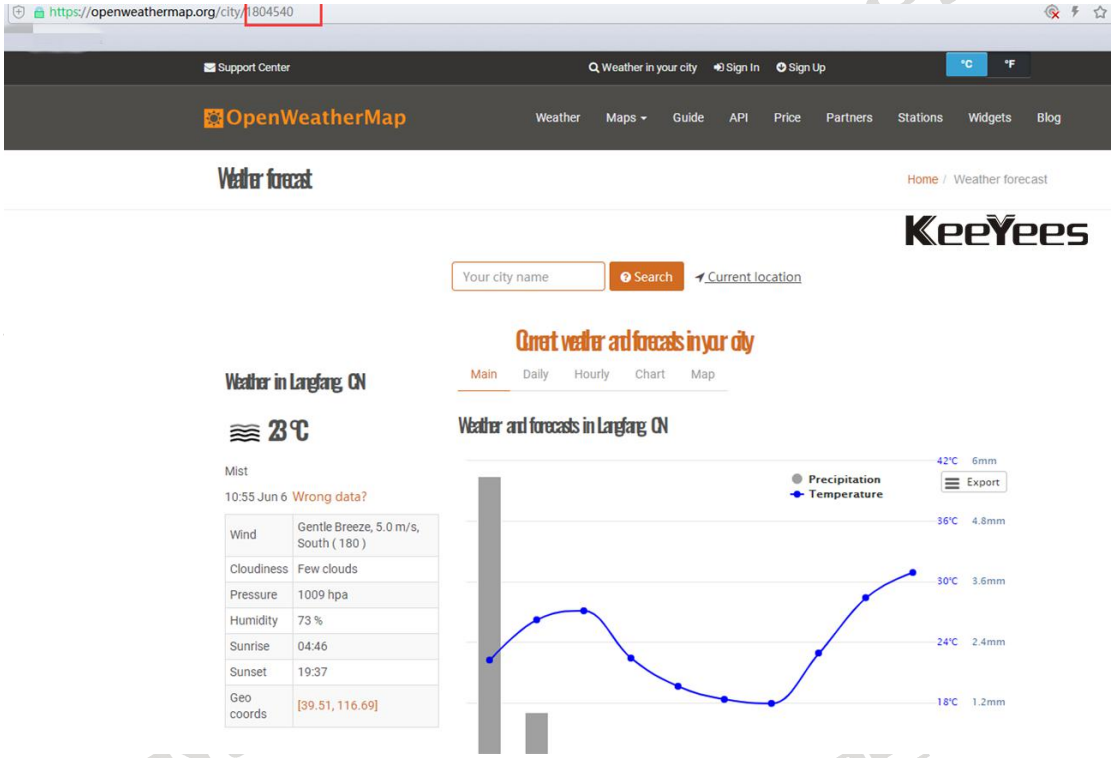
8. Search for the name of your city



9. Click the name of city.



10. The number after the url is the city code. Paste it into the code.

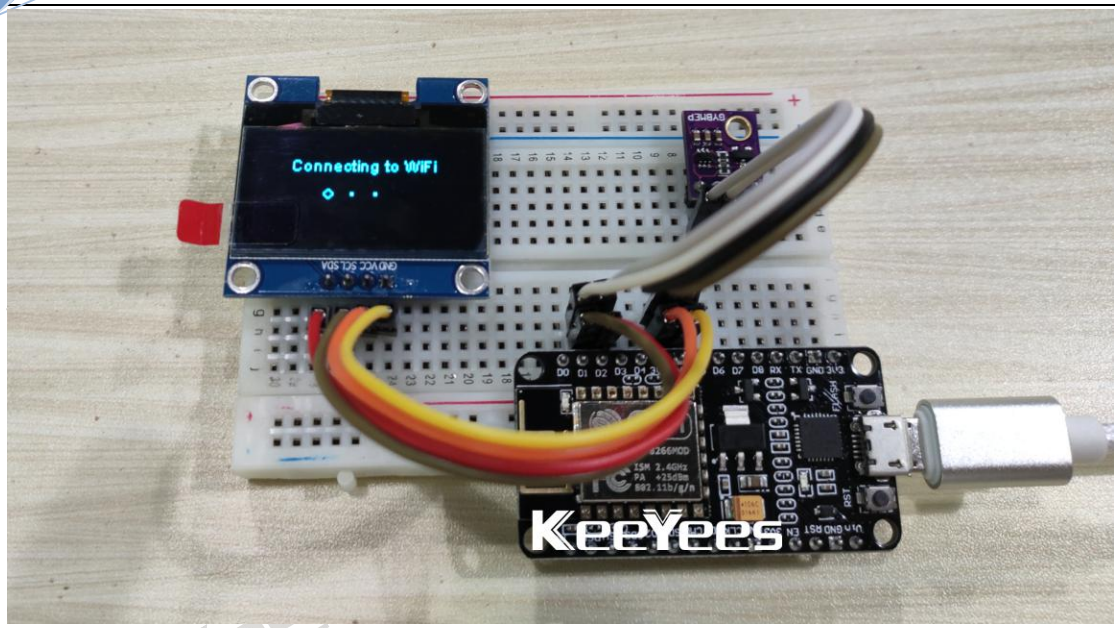


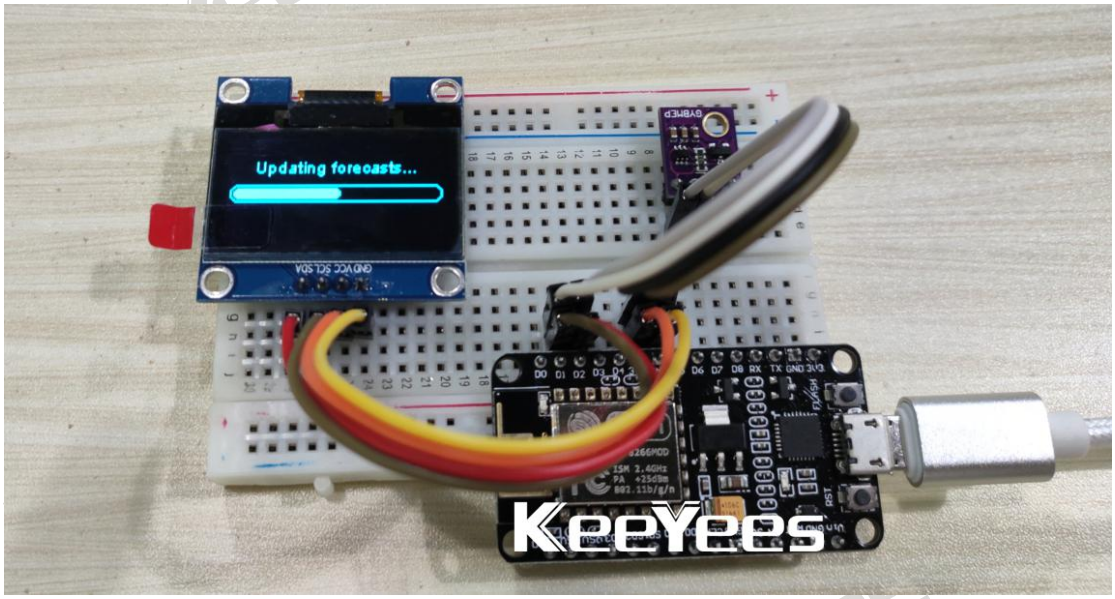
```
WeatherStationDemo §   WeatherStationFonts.h   WeatherStationImages.h
#else
const int SDA_PIN = 5; //D3;
const int SDC_PIN = 4; //D4;
#endif


// OpenWeatherMap Settings
// Sign up here to get an API key:
// https://docs.thingpulse.com/how-tos/openweathermap-key/
String OPEN_WEATHER_MAP_APP_ID = "d4327f95d368a79b5c0f82af6d407010";
/*
Go to https://openweathermap.org/find?q= and search for a location. Go through the
result set and select the entry closest to the actual location you want to display
data for. It'll be a URL like https://openweathermap.org/city/2657896. The number
at the end is what you assign to the constant below.
 */
String OPEN_WEATHER_MAP_LOCATION_ID = "1804540";

// Pick a language code from this list:
// Arabic - ar, Bulgarian - bg, Catalan - ca, Czech - cz, German - de, Greek - el,
// English - en, Persian (Farsi) - fa, Finnish - fi, French - fr, Galician - gl,
// Croatian - hr, Hungarian - hu, Italian - it, Japanese - ja, Korean - kr,
// Latvian - la, Lithuanian - lt, Macedonian - mk, Dutch - nl, Polish - pl,
// Portuguese - pt, Romanian - ro, Russian - ru, Swedish - se, Slovak - sk,
// Slovenian - sl, Spanish - es, Turkish - tr, Ukrainian - ua, Vietnamese - vi,
```

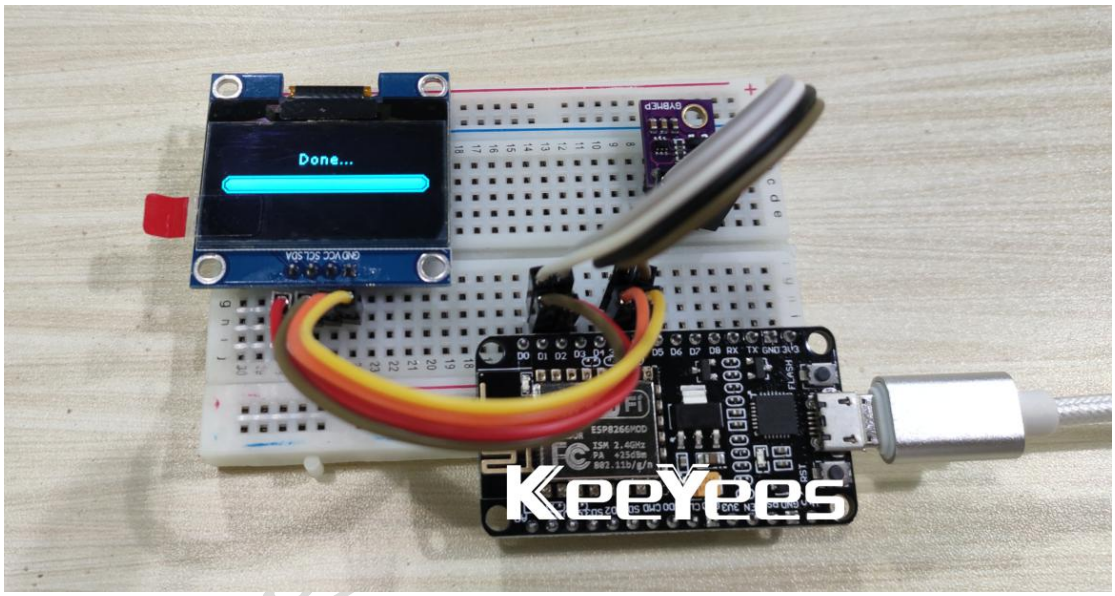
11. Finally, program the code to the development board.

Part 4: Display Effect Diagram

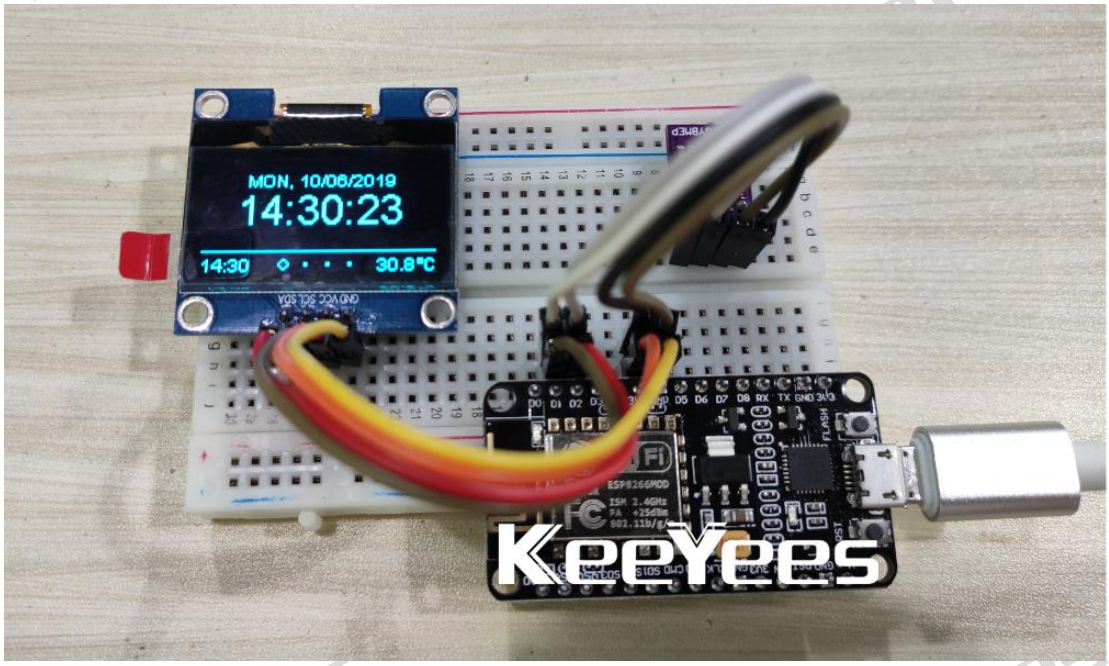
Connecting to WiFi


Connecting to WiFi
KeeYees

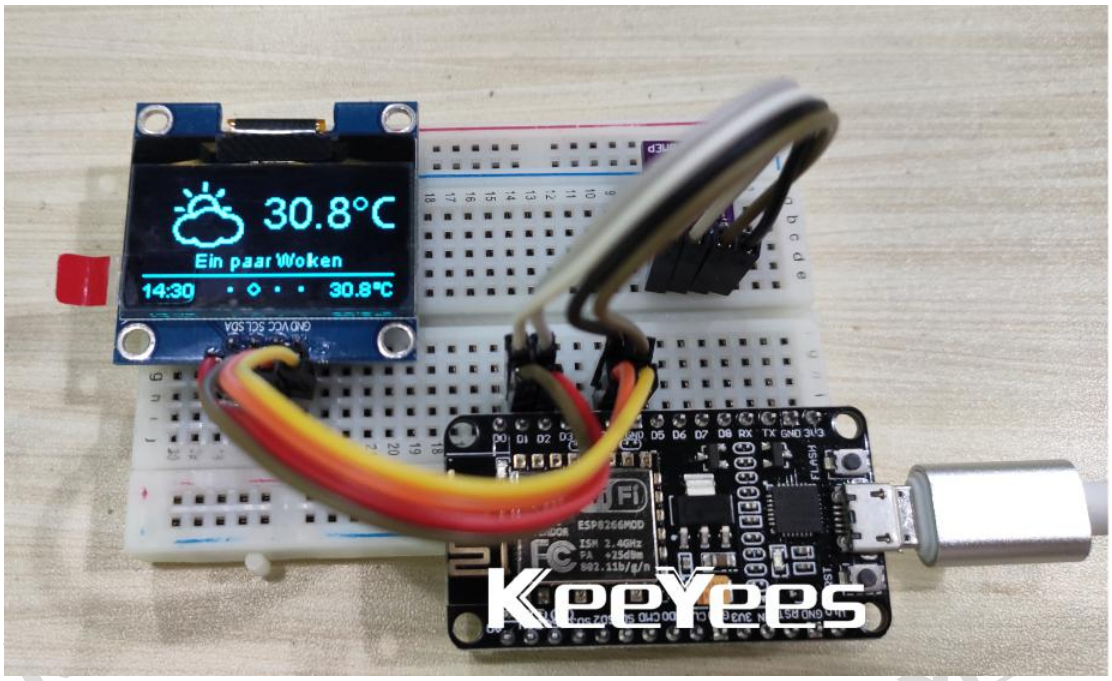

Updating forecasts...
KeeYees

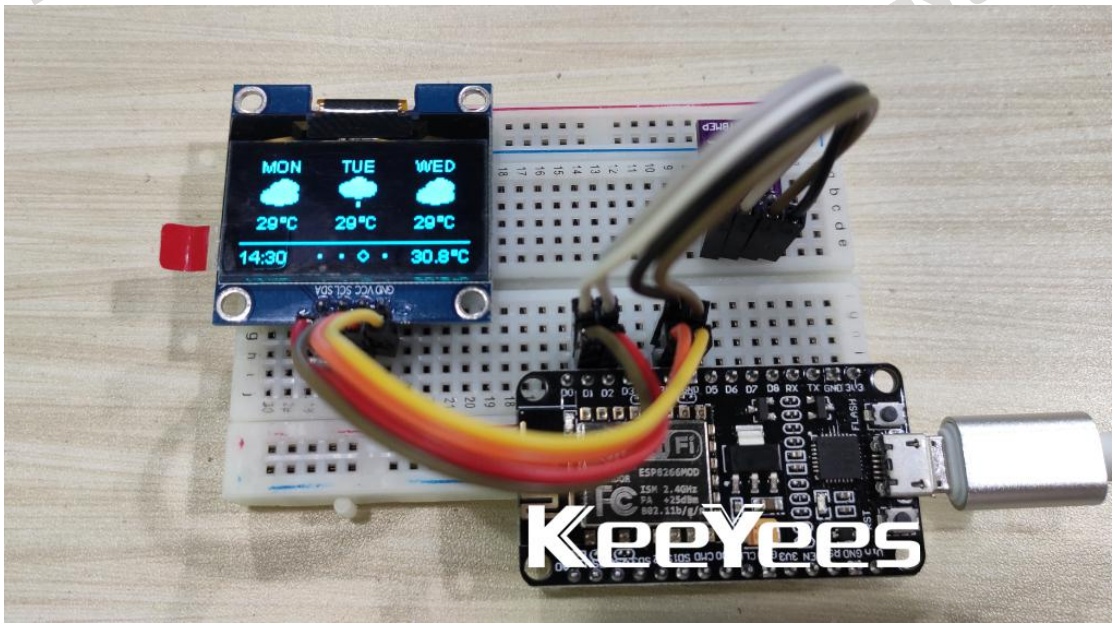


Display Date and Time

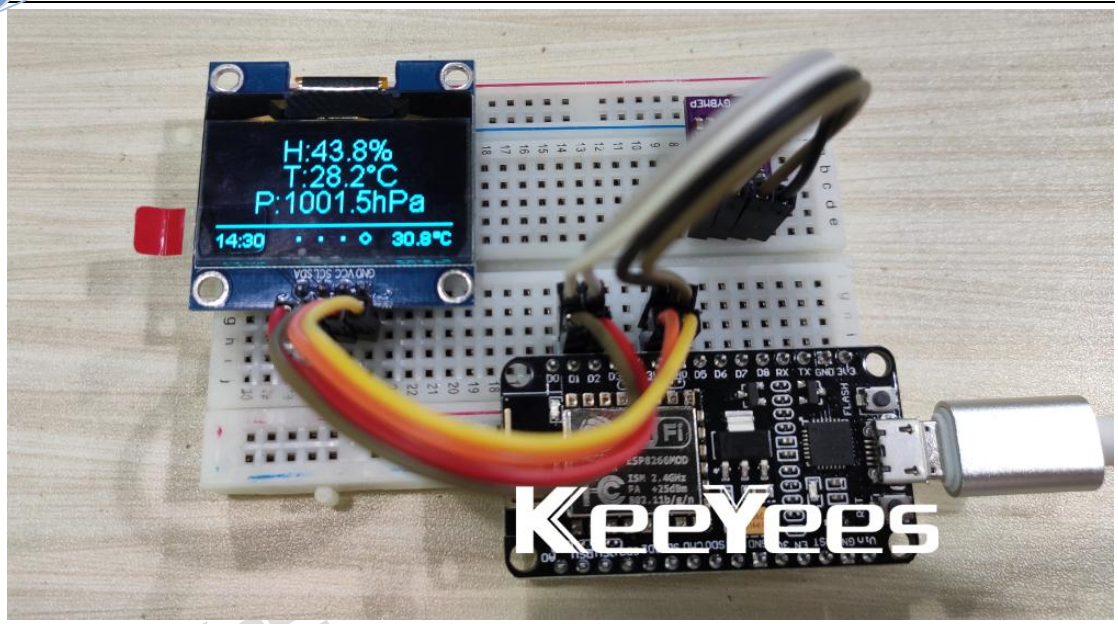


Display Weather and Temperature





Display Temperature Humidity and Atmospheric Pressure


H:43.8%
T:28.2°C
P:1001.5hPa
14:30
30.8°C
KeeYees