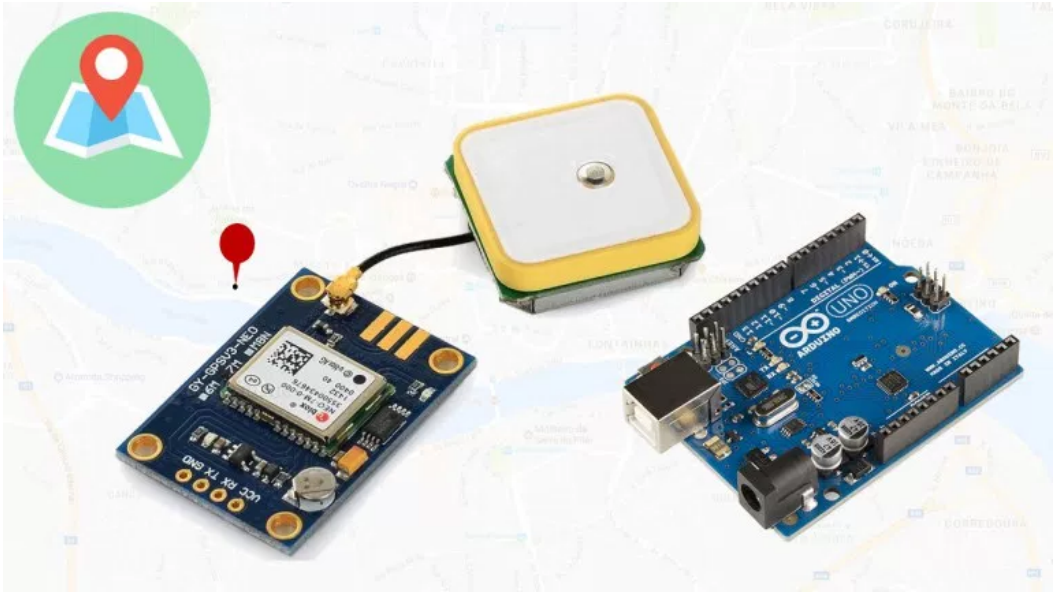
Guide to NEO-6M GPS Module with Arduino

This guide shows how to use the NEO-6M GPS module with the Arduino to get GPS data. GPS stands for *Global Positioning System* and can be used to determine position, time, and speed if you're travelling.



You'll learn how to:

» Wire the NEO-6M GPS module to the Arduino UNO

» Get location

Introducing the NEO-6M GPS Module

The NEO-6M GPS module is shown in the figure below. It comes with an external antenna, and does't come with header pins. So, you'll need to get and solder some.

- This module has an external antenna and built-in EEPROM.
- Interface: RS232 TTL
- Power supply: 3V to 5V
- Default baudrate: 9600 bps
- Works with standard NMEA sentences

ENROLL IN: [Arduino Step-by-step Projects Course - Build 25 Projects]

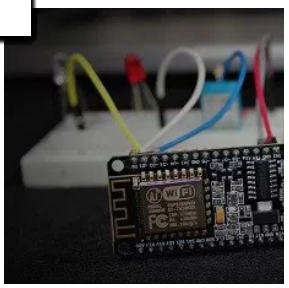
The NEO-6M GPS module is also compatible with other microcontroller boards. To learn how to use the NEO-6M GPS module with the Raspberry Pi, you can read: Email Alert System on Location Change with Raspberry Pi and GPS Module.

Where to buy?

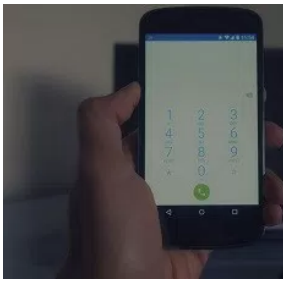
You can get the NEO-6M GPS module for a price between $5 to $20. We recommend checking the NEO-6M GPS module page on Maker Advisor to compare the price in

The NEO-6M GPS module has four pins: VCC, RX, TX, and GND. The module communicates with the Arduino via serial communication using the TX and RX pins, so the wiring couldn't be simpler:

NEO-6M GPS Module	Wiring to Arduino UNO
VCC	5V
RX	TX pin defined in the software serial
TX	RX pin defined in the software serial



Home Automation usir
Build IoT projects and h
gadgets with the ESP82



Build a Home Automat
Learn how to build a ho
system using open-sour
software from scratch.

GND GND

Getting GPS Raw Data

To get raw GPS data you just need to start a serial communication with the GPS module using Software Serial. Continue reading to see how to do that.

Parts Required

For testing this example you'll need the following parts:

» Arduino – read Best Arduino Starter Kits

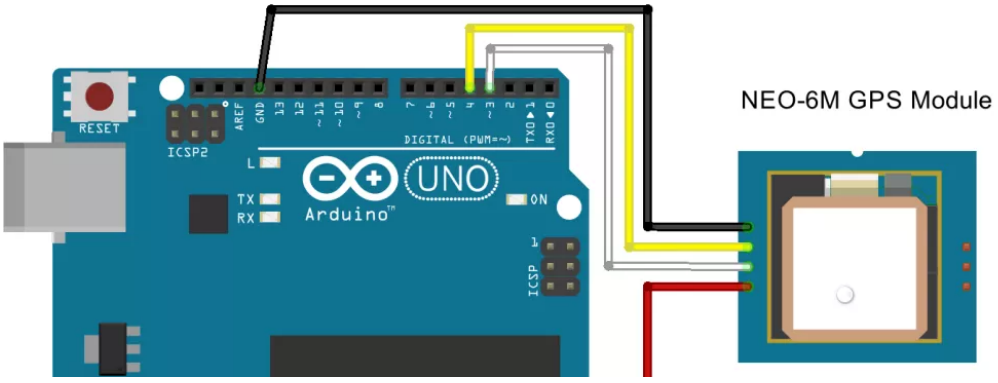
» NEO-6M GPS module

» Jumper wires

You can use the preceding links or go directly to MakerAdvisor.com/tools to find all the parts for your projects at the best price!

Schematics

Wire the NEO-6M GPS module to your Arduino by following the schematic below.



fritzing

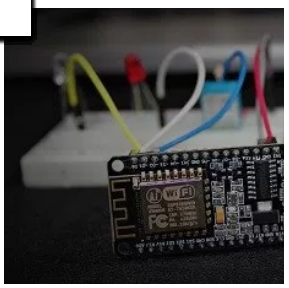
» The module GND pin is connected to Arduino GND pin

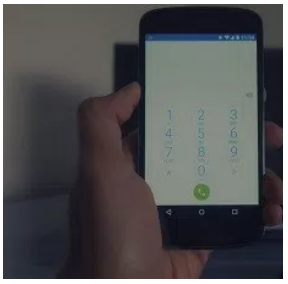
» The module RX pin is connected to Arduino pin 3

» The module TX pin is connected to Arduino pin 4

» The module VCC pin is connected to Arduino 5V pin

Code





Copy the following code to your Arduino IDE and upload it to your Arduino


```
/*
 * Rui Santos
 * Complete Project Details http://randomnerdtutorials.com
 */

#include <SoftwareSerial.h>

// The serial connection to the GPS module
SoftwareSerial ss(4, 3);

void setup(){
  Serial.begin(9600);
  ss.begin(9600);
}

void loop(){
  while (ss.available() > 0){
    // get the byte data from the GPS
    byte gpsData = ss.read();
    Serial.write(gpsData);
  }
}
```

Projects/Arduino-GPS/GPS_Raw_Data.ino view raw

This sketch assumes you are using pins 4 and 3 as RX and TX serial pins to establish serial communication with the GPS module. If you're using other pins you should edit that on the following line:

```
SoftwareSerial ss(4, 3);
```

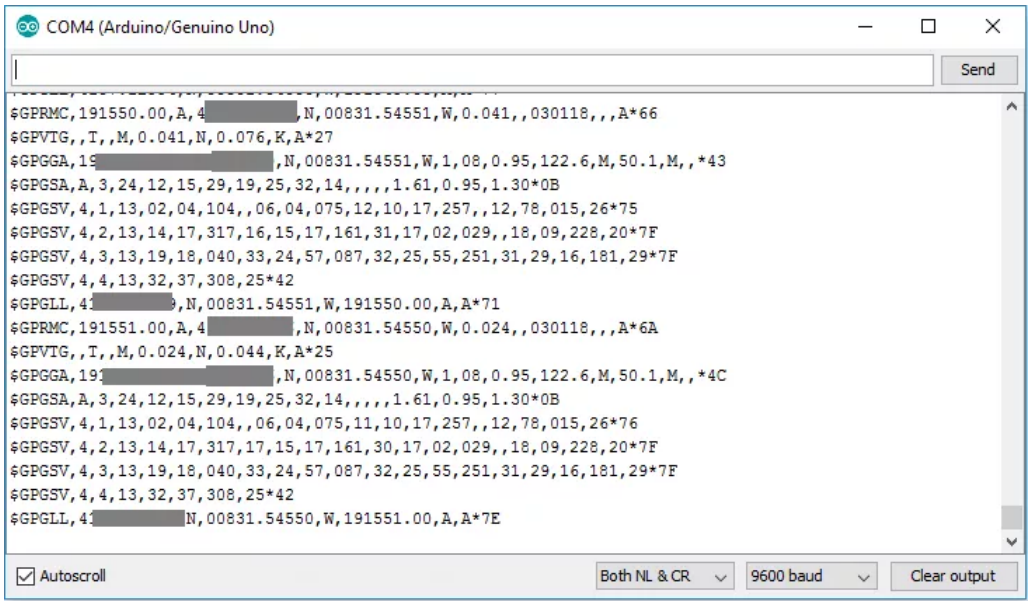


```
ss.begin(9600);
```

This sketch listen to the GPS serial port, and when data is received from the module, it is sent to the serial monitor.

```
while (ss.available() > 0){
  // get the byte data from the GPS
  byte gpsData = ss.read();
  Serial.write(gpsData);
}
```

Open the Serial Monitor at a baud rate of 9600.



You should get a bunch of information in the GPS standard language, NMEA. Each line you get int the serial monitor is an NMEA sentence.

NMEA stands for National Marine Electronics Association, and in the world of GPS, it is a standard data format supported by GPS manufacturers.

Understanding NMEA Sentences

NMEA sentences start with the $ character, and each data field is separated by a comma.

```
$GPGGA,110617.00,41XX.XXXXX,N,00831.54761,W,1,05,2.68,129.0,M,50.1,M,,*42
$GPGSA,A,3,06,09,30,07,23,,,,,,,,4.43,2.68,3.53*02
$GPGSV,3,1,11,02,48,298,24,03,05,101,24,05,17,292,20,06,71,227,30*7C
$GPGSV,3,2,11,07,47,138,33,09,64,044,28,17,01,199,,19,13,214,*7C
$GPGSV,3,3,11,23,29,054,29,29,01,335,,30,29,167,33*4E
$GPGLL,41XX.XXXXX,N,00831.54761,W,110617.00,A,A*70
```


There are different types of NMEA sentences. The type of message is indicated by the characters before the first comma.

The GP after the $ indicates it is a GPS position. The $GPGGA is the basic GPS NMEA message, that provides 3D location and accuracy data. In the following sentence:

```
$GPGGA,110617.00,41XX.XXXXX,N,00831.54761,W,1,05,2.68,129.0,M,50.1,M,,*42
```

- **110617** – represents the time at which the fix location was taken, 11:06:17 UTC
- **41XX.XXXXX,N** – latitude 41 deg XX.XXXXX' N
- **00831.54761,W** – Longitude 008 deg 31.54761' W

» **1** – fix quality (0 = invalid; 1= GPS fix; 2 = DGPS fix; 3 = PPS fix; 4 = Real Ti
5 = Float RTK; 6 = estimated (dead reckoning); 7 = Manual input mode; 8 =
mode)

» **05** – number of satellites being tracked

» **2.68** – Horizontal dilution of position

» **129.0, M** – Altitude, in meters above the sea level

» **50.1, M** – Height of geoid (mean sea level) above WGS84 ellipsoid

» empty field – time in seconds since last DGPS update

» empty field – DGPS station ID number

» ***42** – the checksum data, always begins with *

The other NMEA sentences provide additional information:

» **$GPGSA** – GPS DOP and active satellites

» **$GPGSV** – Detailed GPS satellite information

» **$GPGLL** – Geographic Latitude and Longitude

» **$GPRMC** – Essential GPS pvt (position, velocity, time) data

» **$GPVTG** – Velocity made good

To know what each data field means in each of these sentences, you can consult NMEA data here.

Parsing NMEA Sentences with TinyGPS++ Library

You can work with the raw data from the GPS, or you can convert those NMEA messages into a readable and useful format, by saving the characters sequences into variables. To do that, we're going to use the TinyGPS++ library.

Library.

Installing the TinyGPS++ Library

Follow the next steps to install the TinyGPS++ library in your Arduino IDE:

1. Click here to download the TinyGPSPlus library. You should have a .zip folder in your Downloads folder
2. Unzip the .zip folder and you should get TinyGPSPlus-master folder
3. Rename your folder from ~~TinyGPSPlus-master~~ to TinyGPSPlus
4. Move the TinyGPSPlus folder to your Arduino IDE installation libraries folder
5. Finally, re-open your Arduino IDE

The library provides several examples on how to use it. In your Arduino IDE need to go to **File** > **Examples** > **TinyGPS++**, and choose from the example

Note: the examples provided in the library assume a baud rate of 4800 for the GPS module. You need to change that to 9600 if you're using the NEO-6M GPS module.

Getting Location Using the NEO-6M GPS Module and the TinyGPS++ Library

You can get the location in a format that is convenient and useful by using the TinyGPS++ library. Below, we provide a code to get the location from the GPS. This is a simplified version of one of the library examples.

```
/*
 * Rui Santos
 * Complete Project Details http://randomnerdtutorials.com
 */

#include <TinyGPS++.h>
#include <SoftwareSerial.h>

static const int RXPin = 4, TXPin = 3;
static const uint32_t GPSBaud = 9600;

// The TinyGPS++ object
TinyGPSPlus gps;

// The serial connection to the GPS device
SoftwareSerial ss(RXPin, TXPin);

void setup(){
  Serial.begin(9600);
  ss.begin(GPSBaud);
}
```


Projects/Arduino-GPS/NEO_6M_get_location.ino view raw

You start by importing the needed libraries: TinyGPSPlus and SoftwareSerial:

```
#include <TinyGPS++.h>
#include <SoftwareSerial.h>
```

Then, you define the software serial RX and TX pins, as well as the GPS baud rate. If you are using other pins for software serial you need to change that here. Also, if your GPS module uses a different default baud rate, you should also modify that.

```
static const int RXPin = 4, TXPin = 3;
static const uint32_t GPSBaud = 9600;
```

Then, you create a TinyGPS++ object:

```
TinyGPSPlus gps;
```

And start a serial connection on the pins you've defined earlier

```
SoftwareSerial ss(RXPin, TXPin);
```

In the setup(), you initialize serial communication, both to see the readings on the serial monitor and to communicate with the GPS module.

```
void setup() {
  Serial.begin(9600);
  ss.begin(GPSBaud);
}
```

In the loop is where you request the information. To get TinyGPS++ to work, you have to repeatedly funnel the characters to it from the GPS module using the encode() method.

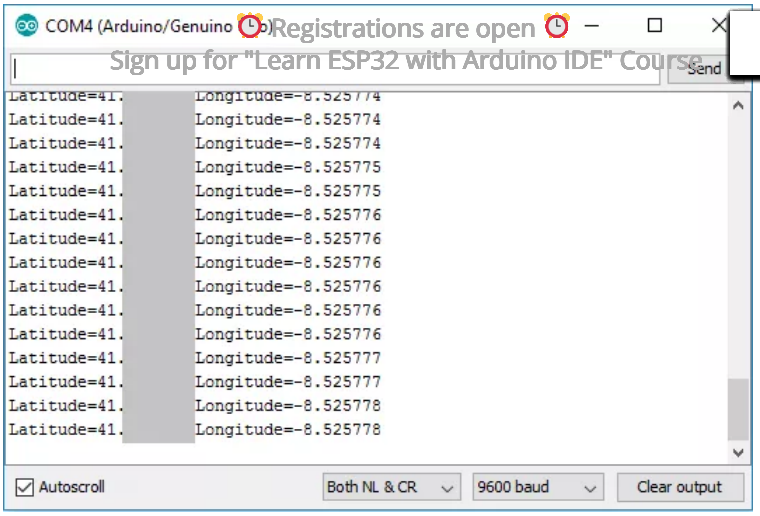
```
while (ss.available() > 0){
 gps.encode(ss.read());
```

Then, you can query the gps object to see if any data fields have been updated:


```
}
```

Getting the latitude and longitude is has simple has using **gps.location.lat()**, and **gps.location.lng()**, respectively.

Upload the code to your Arduino, and you should see the location displayed on the serial monitor. After uploading the code, wait a few minutes while the module adjusts the position to get a more accurate data.



Getting More GPS Information Using the TinyGPS++ Library

The TinyGPS++ library allows you to get way more information than just the location, and in a simple way. Besides the location, you can get:

- date
- time
- speed
- course
- altitude
- satellites
- hdop

The code below exemplifies how you can get all that information in a simple way.

```
/*
 * Rui Santos
 * Complete Project Details http://randomnerdtutorials.com
```



```
 */

#include <TinyGPS++.h>
#include <SoftwareSerial.h>

static const int RXPin = 4, TXPin = 3;
static const uint32_t GPSBaud = 9600;

// The TinyGPS++ object
TinyGPSPlus gps;

// The serial connection to the GPS device
SoftwareSerial ss(RXPin, TXPin);
```

```
void setup(){
  Serial.begin(9600);
  ss.begin(GPSBaud);
}
```

Projects/Arduino-GPS/NEO_6M_get_all_GPS_data.ino view raw

Note: the TinyGPS++ library is well commented on how to use all its functionalities.

Wrapping Up

We hope you've found this guide useful. We intend to make a GPS data logger with the NEO-6M GPS module and the SD card module, so stay tuned.

If you liked this project you may also like:

» Guide to SIM900 GSM GPRS Shield with Arduino

» Email Alert System on Location Change with Raspberry Pi and GPS Module

» Arduino Step-by-Step Projects

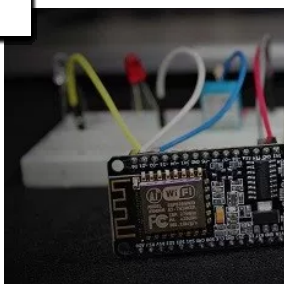
Thanks for reading.

What to Read Next...

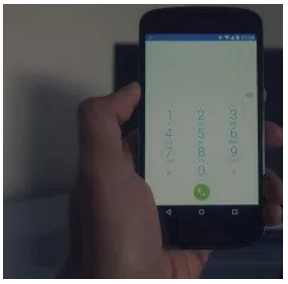
01 Jul, 2018
Complete Guide for Ultrasonic Sensor HC-SR04 with Arduino

01 Jul, 2018
Complete Guide for RF 433MHz Transmitter/Receiver Module With Arduino

14 Mar, 2018
Guide to SIM900 GSM GPRS Shield with Arduino



Home Automation usir
Build IoT projects and h
gadgets with the ESP82



Build a Home Automat
Learn how to build a ho
system using open-sour
software from scratch.

Previous Post
Year in Review 2017 – Random Nerd Tutorials

Next Post
Getting Started with Raspberry Pi 3

Leave a Comment:

Name *

E-Mail *

Website

Save my name, email, and website in this browser for the next time I comment.

Comment

Post Comment

I haven't researched it yet, but it would be convenient to get m/ft relative to an initial point.

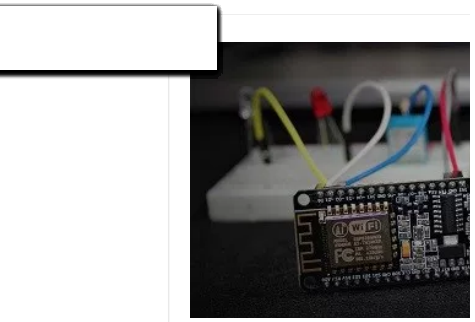
Reply

Sara Santos says
January 5, 2018

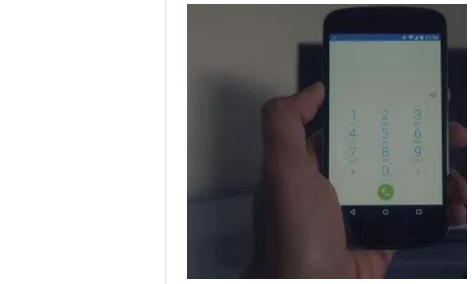
Hi John.
Yes, getting the distance to an initial point can be convenient.
You can do that with the library.

Check the example called “Distance and Course” at the documentation here: http://arduiniana.org/libraries/tin
They give an example on how to do that.
I hope this helps.
Thanks 🙂

Reply



Home Automation usir
Build IoT projects and h
gadgets with the ESP826



Build a Home Automat
Learn how to build a ho
system using open-sour
software from scratch.

Bertil says
January 4, 2018

1:
why this void setup() {
Serial.begin(115200);
ss.begin(GPSBaud);
}
erlier you said 9600

2:
Serial.print(gps.location.lat(), 6);

what makes the number 6 ?

3:
Howe can I write to get the readings in “Degrees minutes seconds formats (DDD MM SS)”

Regards

Bertil

Reply

Hi Bertil.

1. It is 9600. I’ve already corrected that. Thanks for noticing.
2. The 6 specifies the number of decimal places.
3. You can get the location in that format in the $GPGLL NMEA sentence, when you get the raw data. To do that with the library, check the “Custom NMEA Sentence Extraction” in the library documentation here http://arduiniana.org/libraries/tinygpsplus/

Thanks.
Sara

Reply

LanceArn says
January 27, 2018

Thank you very much for this Guide. I have had this module for several weeks now and followed many guides but I could not get it to receive GPS DATA, yet alone to display it in the Serial Monitor. The I found your Guide to the NEO-6M GPS Module.
When I first ran the “Getting GPS Raw Data” code, I was excited that it worked first time.
I then tried the “Getting Location ” code. After 5 seconds it started to display the Lat & Long location. Two successes.
Thirdly, I had a go at the “Getting More GPS Information” code. After several tries I was disappointed that I was seeing no GPS DATA on the monitor. I spent the next 5 minutes looking over the code, trying to find out where I had gone wrong. Then suddenly, the monitor came to life and there was the GPS DATA being displayed and updating. I was impressed. You had done what many other codes could not do, you enabled me to receive the GPS DATA.

Next project – Add a 0.96” OLED display and make the PGS portable.

Kudos for this guide..

Reply

Sara Santos says
January 29, 2018

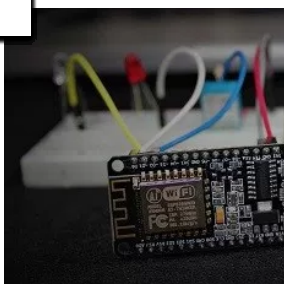
Hi Lance!
We always do our best to make our guides easy to follow, so that anyone is able to follow along.
We’re happy to hear that you’ve found this guide useful, and that your GPS module is working perfectly.
We also intend to make a new GPS project – a GPS datalogger – in a

Reply

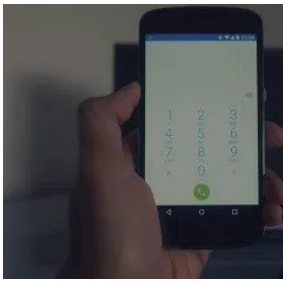
Aivars says
February 4, 2018

Hello!
Thank you for the tutorial!
Can I get a signal to synchronize time with my DS3231? Are new sentences sent in 00 seconds?

Reply



Home Automation usir
Build IoT projects and h
gadgets with the ESP82



Build a Home Automat
Learn how to build a ho
system using open-sour
software from scratch.

Sara Santos says
February 5, 2018

Hi.
What do you mean with synchronize with DS3231?
The GPS module allows you to get information about the time too.

Reply

Aivars says
February 5, 2018

Hello! I need a clock that contains the RTC block DS3231 to be precisely tuned to the other, which is several kilometers away. In my opinion, the only way through the GPS. How do i do this?

Reply

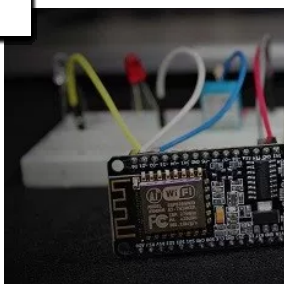
Sara Santos says
February 6, 2018

Hi again.
We don't have any tutorial on that subject.
Maybe you need to read the time through the GPS module, and then, parse that time to the DS3231,
In this post you can see how to get GPS data. An on the following link, you can see how to set the time for the DS3231 real time clock:
https://randomnerdtutorials.com/guide-for-real-time-clock-rtc-

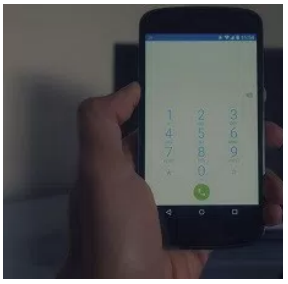
Mark says
February 11, 2018

Hi how can i stop that infinite loop? I just need to scan data once. Any adice? thanks!!

Reply



Home Automation usir
Build IoT projects and h
gadgets with the ESP82



Build a Home Automat
Learn how to build a ho
system using open-sour
software from scratch.

Rui Santos says
February 12, 2018

You can create a separate function that is called once in the setup function, for example

Reply

David says
May 22, 2018

Hi, I have a NEO6M GPS board from Wish, following this guide does not give me the NMEA sentences, all I get on the serial monitor is UBX-G60xx 00040007 FF47FFFFp)Z♀b*ROM CORE 7.03 (45969) Mar 1 repeating over and over again. My baud rate is set at 57600. My board is connected to the arduino 3.3V out and I have a voltage divider between Arduino TX and the module RX. The red light on the board is always on, and it does feel hot to the touch.

Any ideas?

Reply

Rui Santos says
June 11, 2018

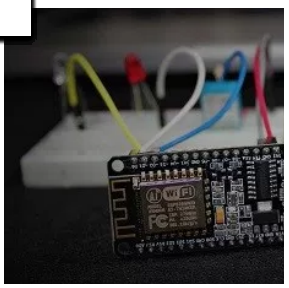
Hello David,
I've never encountered that error before, so I don't know how to fix it. Sorry about that.
Regards,
Rui

dcp_david says
October 13, 2018

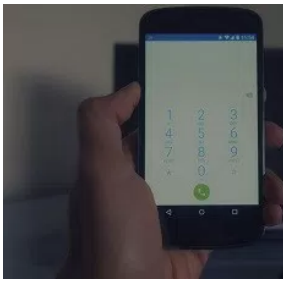
it is restarting because it is powered with 5V.

Reply

Stewart Lindenberger says
May 27, 2018



Home Automation usir
Build IoT projects and h
gadgets with the ESP82



Build a Home Automat
Learn how to build a ho
system using open-sour
software from scratch.

Yet another well presented tutorial – Thank You!

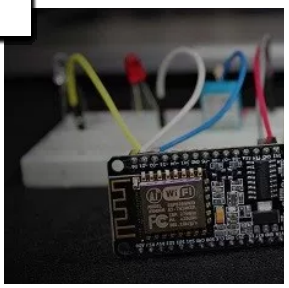
Questions

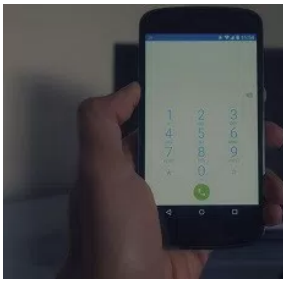
Why is some data reported as zero: altitude = 0.00, number of satellites = 0?

Is there an equivalent SoftwareSerial library for the ESP32?

Why do several of the items have “double” in the comments in the code? Is the data presented half of the real value or twice the real value?

Reply





Rui Santos says
June 11, 2018

Hi Stew! Thanks for your kind words. The variable declared as double simply means “Double variable = Double precision floating point number. On the Uno and other ATMEGA based boards, this occupies 4 bytes. That is, the double implementation is exactly the same as the float, with no gain in precision.” More information here: arduino.cc/reference/en/language/variables/data-types/double/

I guess that happens (altitude = 0) when the requests fails to retrieve the data properly…

Reply

Faruq Sahir says
June 2, 2018

how gps module cab fetch location without internet???

Reply

Rui Santos says
June 11, 2018

Hi Faruq, the GPS module makes a request to satellites that provide the current location. No need for an Internet connection.

Reply

Asad Ali says
June 17, 2018

Hey! Thanks for such a useful tutorial. I was wondering if there is a way to reduce the time taken by the GPS module before it starts displaying data?

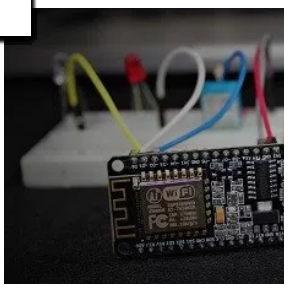
Reply

Rui Santos says
June 23, 2018

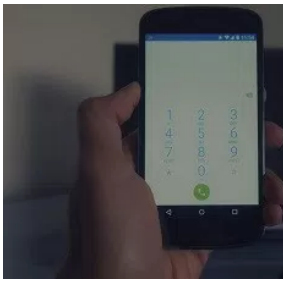
I think, you would need to optimize the library for a faster GPS readings... Or use a more powerful board with the module.

Reply

Add Your Reply



Home Automation usir
Build IoT projects and h
gadgets with the ESP82



Build a Home Automat
Learn how to build a ho
system using open-sour
software from scratch.