

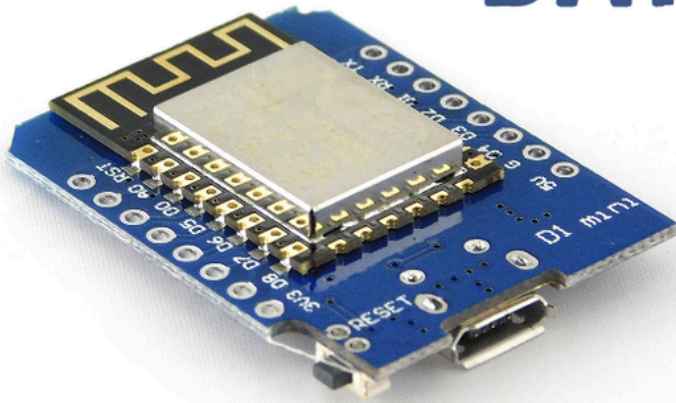
**AUTODESK**
Instructables

DIY: Monitor Your Car Battery: Code & Setup

By [MrDIYLab](#) in [CircuitsSensors](#)

Introduction: DIY: Monitor Your Car Battery: Code & Setup

MONITOR YOUR CAR BATTERY

MrDIY**WIFI**
MQTT

Having the ability to monitor your car battery can prevent some unpleasant surprises. I will show you how I assembled the hardware, loaded the software and installed the monitor in my car. I will be using the ESP8266 Board called Wemos D1 Mini.

New to ESP8266? Watch my [Introduction to ESP8266 video](#) first.

Step 1: Watch the Video

The video has step by step instructions that will guide you through the process. Feel free to add your questions in the comment section of the YouTube video if you need any further assistance.

Step 2: Order the Components

Buy on Amazon.com

- Wemos D1 Mini - <https://amzn.to/308Sxoy>
- Power Shield - <https://amzn.to/3gbfqxq>
- Assorted Resistors - <https://amzn.to/2PagR34>

Buy on AliExpress:

- Wemos d1 mini - https://s.click.aliexpress.com/e/_AEZGUP
- Power Shield - https://s.click.aliexpress.com/e/_AIE2SF
- Assorted Resistors - https://s.click.aliexpress.com/e/_AAfyJV

Buy on Amazon.ca

- Wemos D1 Mini - <https://amzn.to/3cpOxDe>
- Power Shield - <https://amzn.to/2XuolxD>
- Assorted Resistors - <https://amzn.to/2AtfU1P>

Step 3: Hardware

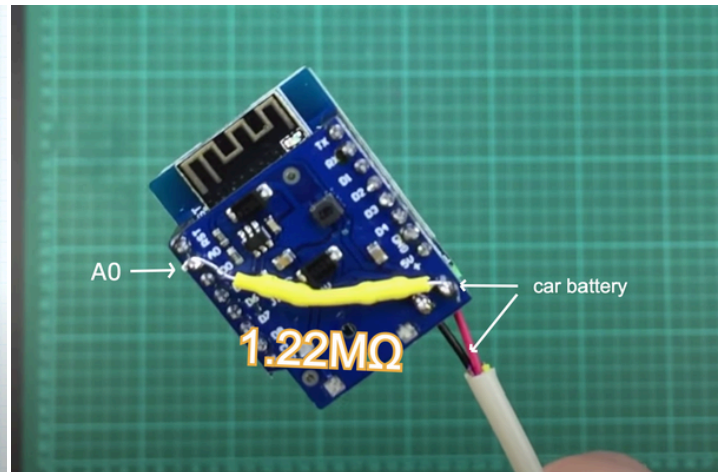
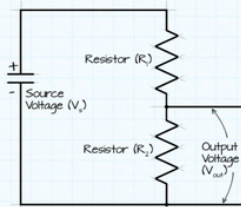
A voltage divider circuit is a very common circuit that takes a voltage and converts it to a lower one by using a pair of resistors. The Formula For calculating the output voltage is based on Ohm's law and is shown below:

$$V_{out} = \frac{V_s \times R_2}{R_1 + R_2}$$

V_s is the source voltage, measured in volts (V).
 R_1 is the resistance of the 1st resistor, measured in Ohms (Ω).
 R_2 is the resistance of the 2nd resistor, measured in Ohms (Ω).
 V_{out} is the output voltage, measured in volts (V).

Enter three known values and press 'Calculate' to solve for the other:

Source (V_s)	<input type="text" value="16"/>	Volts (V)
Resistance 1 (R_1)	<input type="text" value="1440"/>	<input type="text" value="k10ohms (kΩ)"/>
Resistance 2 (R_2)	<input type="text" value="100"/>	<input type="text" value="k10ohms (kΩ)"/>
Voltage (V_{out})	<input type="text" value="1.039"/>	Volts (V)
<input type="button" value="Calculate"/>		Click 'Calculate' to update the field with orange border.

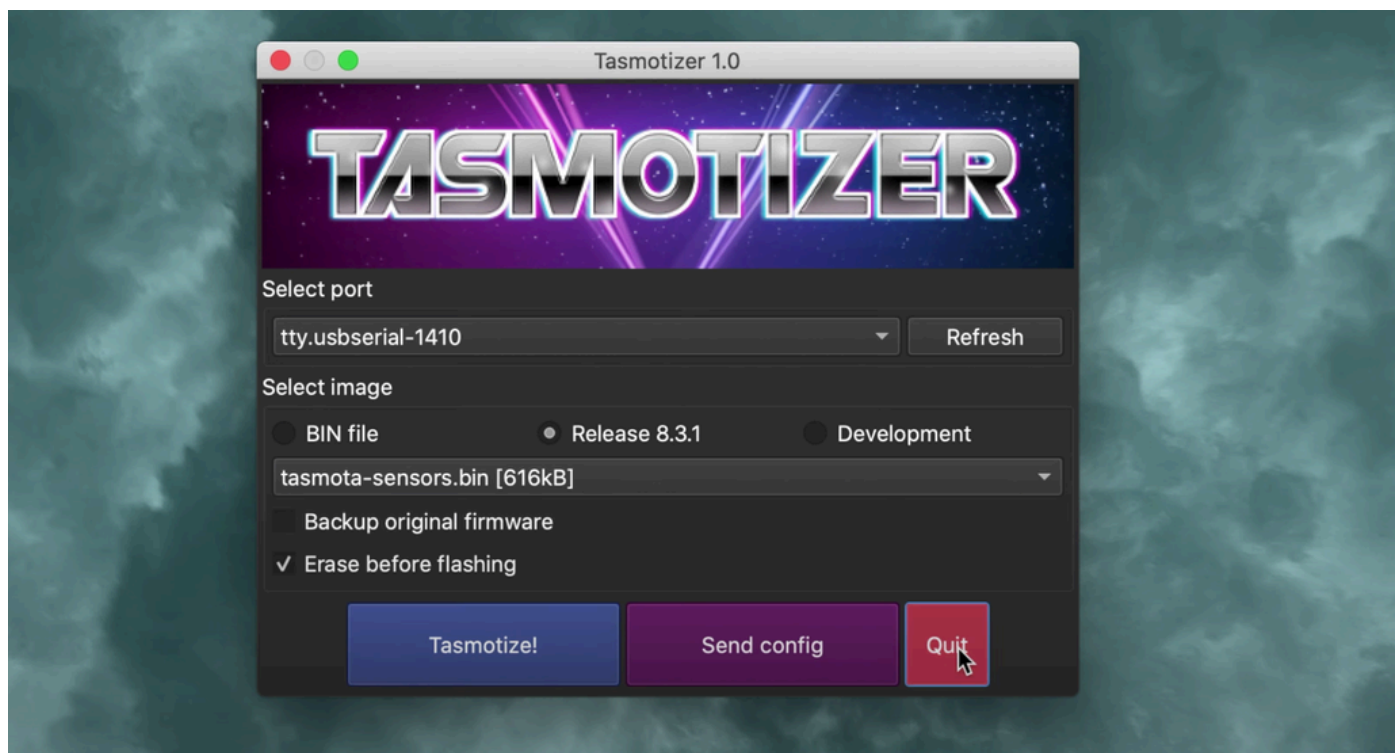


You will need a wemos d1 mini, a power shield and some resistors. First, I started by removing the power plug and installed the smaller connector to make the hardware more compact.

The D1 mini can measure external voltage up to 3.3v by using a voltage divider using $R_1=220K\Omega$ & $R_2=100K\Omega$. This the voltage within the 0-1 Volt that the ADC can tolerate. To increase the 3.3v to 16v needed for the car battery, we need to increase R_1 to 1.44M Ω . To do that we can add another 1.22M Ω in series to get the total 1.44M Ω . I did this by soldering a 1M Ω resistor to a 220K Ω resistor as shown here.

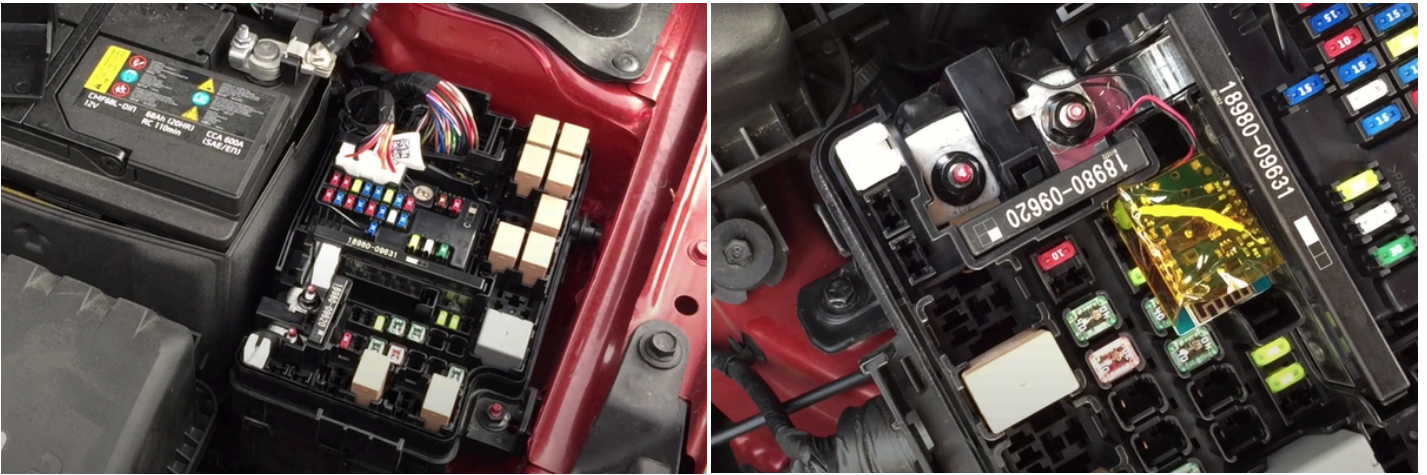
I attached a long wire to the power input terminals to be able to connect them to the car battery.

Step 4: Software



I then connected the D1 mini to my laptop and load the software. Make sure you select the sensor.bin version for the Analog input functionality. I then resumed with a typical Tasmota configuration.

Step 5: Car



At the car, I opened the hood and located the fuse box. I found the fuse box to be a safe and secure place to install my device.

I first wrapped the device in heat resistant tape to cover any exported pin to withstand the engine heat. Since the entire car chase is ground, I found the nearest screw and connected my ground to it. Next, I located the nearest connection to the battery positive rail and connected my positive power input to it.

Step 6: Calibrate

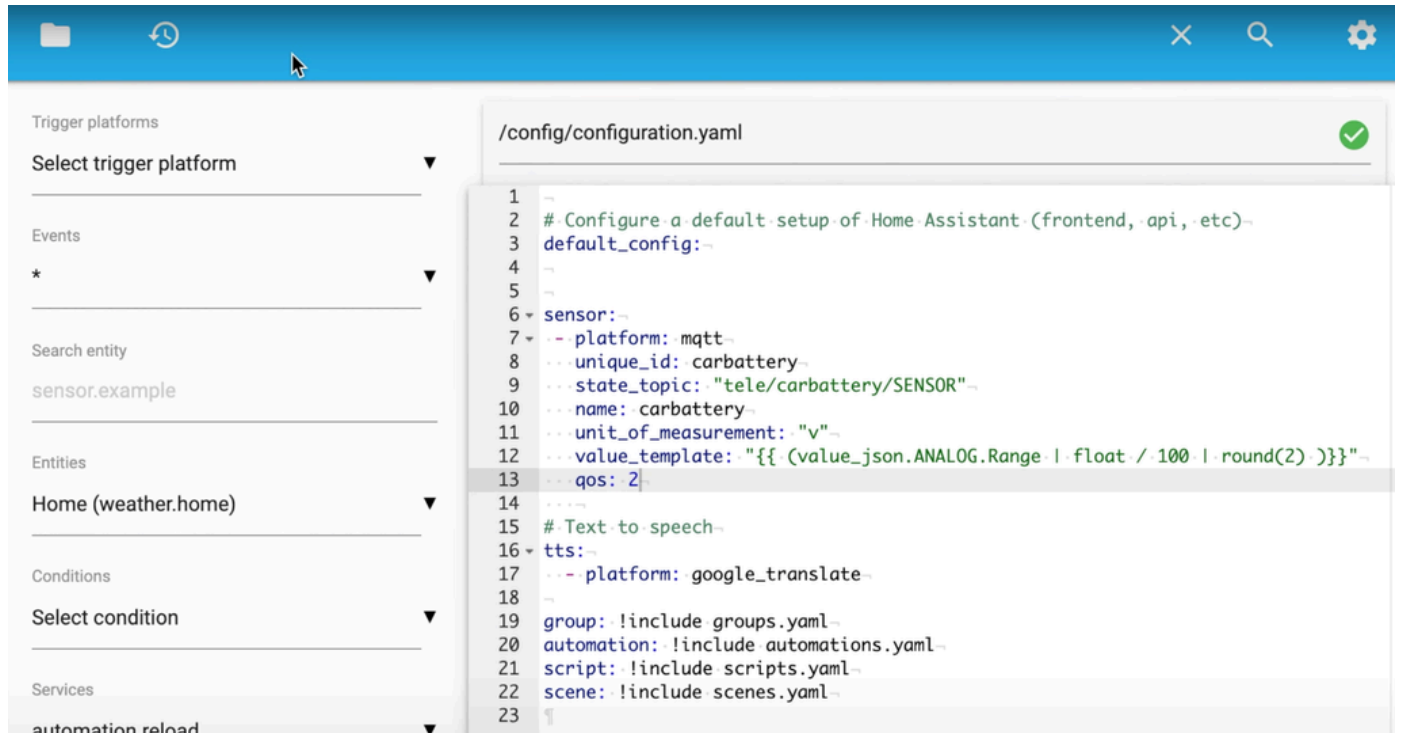
```
00:28:28 MQT: stat/carbattery/RESULT = {"AdcParam":[6,0,1023,0,1600]}
00:28:32 CMD: AdcParam 6, 0, 1023, 0, 1600
00:28:32 MQT: stat/carbattery/RESULT = {"AdcParam":[6,0,1023,0,1600]}
00:28:35 CMD: AdcParam 6, 0, 1023, 0, 1700
00:28:35 MQT: stat/carbattery/RESULT = {"AdcParam":[6,0,1023,0,1700]}
00:28:43 CMD: AdcParam 6, 0, 1023, 0, 1730
00:28:43 MQT: stat/carbattery/RESULT = {"AdcParam":[6,0,1023,0,1730]}
00:28:49 CMD: AdcParam 6, 0, 1023, 0, 1750
00:28:49 MQT: stat/carbattery/RESULT = {"AdcParam":[6,0,1023,0,1750]}
00:28:54 CMD: AdcParam 6, 0, 1023, 0, 1755
00:28:54 MQT: stat/carbattery/RESULT = {"AdcParam":[6,0,1023,0,1755]}
00:29:00 CMD: AdcParam 6, 0, 1023, 0, 1760
00:29:00 MQT: stat/carbattery/RESULT = {"AdcParam":[6,0,1023,0,1760]}
00:29:06 CMD: AdcParam 6, 0, 1023, 0, 1770
00:29:06 MQT: stat/carbattery/RESULT = {"AdcParam":[6,0,1023,0,1770]}
```

AdcParam 6, 0, 1023, 0, 170|

Main Menu

I then proceeded with calibrating the range of the analog input. I did this by connecting a multimeter to the battery and read the current voltage (in my case it was 12.73 volts). I then did some trial and error until my reading was 1273 as the analog reading. The last step was closing the fuse box and the car hood.

Step 7: Home Assistant



Back in Home Assistant, I opened the **configuration** file and added a new MQTT sensor - the code is attached above.

After I saved, I restarted home assistant for it to take effect. When it came back online, I added the new sensor to the dashboard.

Step 8: Done

The integration is now complete. You can now use this sensor to trigger alerts!

If you found this useful, please consider subscribing to [my YouTube channel](#) - It helps me a lot.

If you are interested in supporting my work, you can check [my Patreon page](#).

Much of the information contained is based on personal knowledge and experience. It is the responsibility of the viewer to independently verify all information.