

## **User Guide**

Make for Change

**A Make It Yourself Monitoring Platform for Air Quality Data  
Collection**

## Setup Raspbian

Download. Burn to a SD Card. Enable Wi-Fi. Connect. Change Password.

As operating for the Raspberry Pi we're going to use Raspbian Lite. Download the Raspbian Lite image from <https://www.raspberrypi.org/downloads/raspbian/> and Etcher from <https://etcher.io/>.

Once both are finished downloading, install and start Etcher. Click on "Select Image", find the Raspbian Lite image you've downloaded, then use "Select Drive" and pick your SD card as target device. The next step is to click on "Flash!" and wait.

## Wi-Fi Connection

In order to communicate with the Pi to configure it, we'll need a connection. Because the Pi Zero doesn't have an Ethernet connection by default, we will use Wi-Fi instead.

Press **Windows Key + R**. Type in **explorer.exe** and press **Enter**.

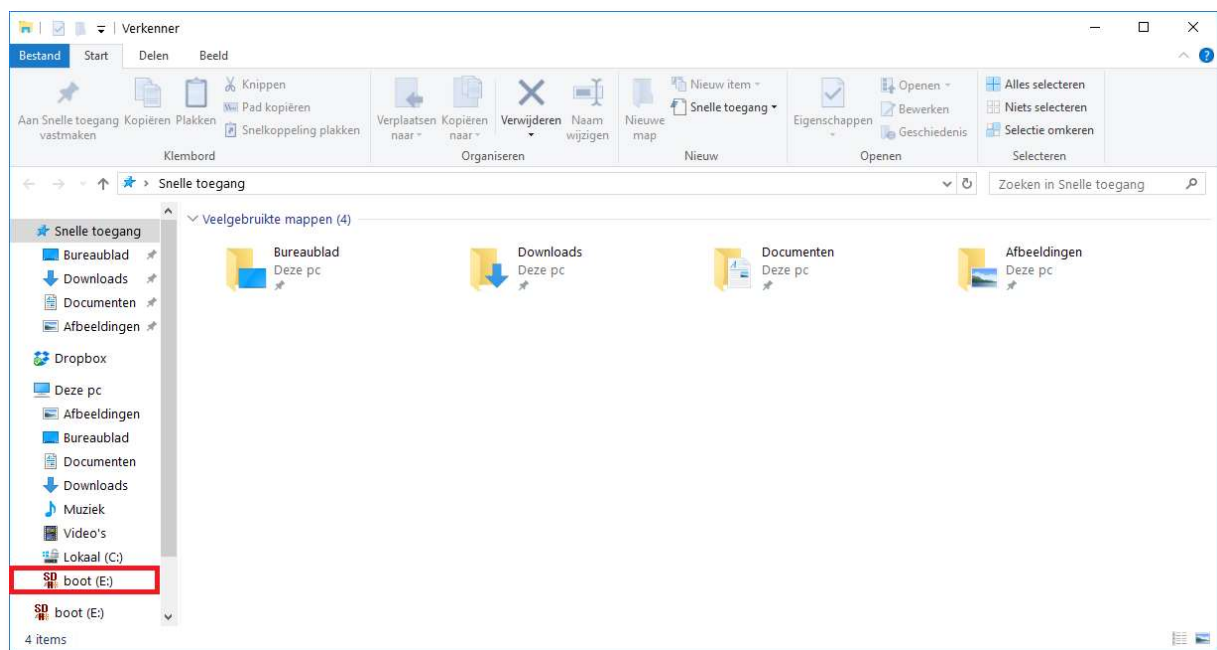


Figure 1 Windows Explorer

Click on the SD card and create a file named **wpa\_supplicant.conf**. Once the Pi is booted, it will copy the content of that file to the appropriate location. Raspbian will then try to connect to that Wi-Fi network.

```
network={
    ssid="name.of.the.network"
    psk="password.to.authenticate"
}
```

**[ Windows Users: make sure Windows didn't add the .txt extension. ]**

## Enabling SSH

Now that our Pi will connect to our Wi-Fi network, one possibility to communicate with it is through SSH. By default SSH isn't enabled on Raspbian since November 2016. In the same directory as you have configured the Wi-Fi connection, add a file named **ssh**. This will tell Raspbian to turn on the SSH service.

[ **Windows Users:** make sure Windows didn't add the **.txt** extension. ]

[ If don't see a file extension on Windows? How can I verify whether the **.txt** extension was added or not? ]

Open the **Explorer** screen. Click on **View** and make sure the **File Extensions** box is ticked, as shown on the picture below.

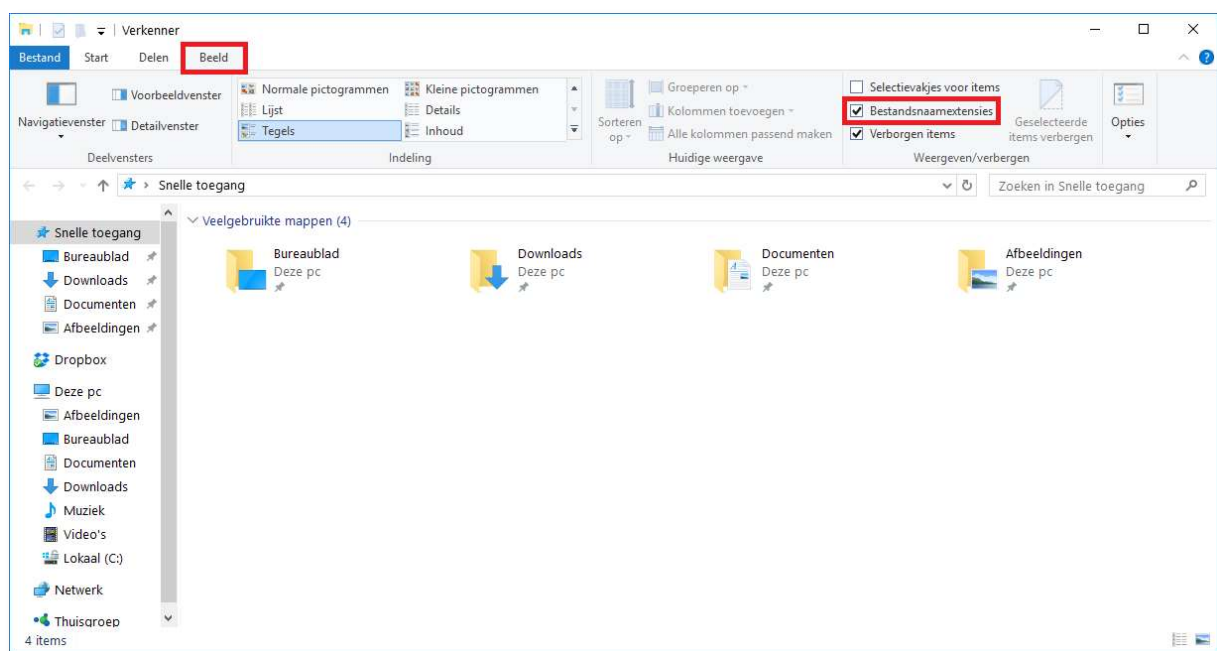


Figure 2 Make the File Extensions Visible on Windows

## Find the IP Address of the Pi

Now that we've configured the Pi, it's time to connect. Insert the SD card into the Pi and turn it on by connecting the power adapter.

The first step is determining your IP address.

- **Windows:** Open a Command Prompt by using **Windows Key + CTRL R**, type in **cmd.exe** and press **Enter**. Once the Command Prompt is visible, type in **ipconfig** and press **Enter**.
- **Linux or Mac:** Open a Terminal. Once the Terminal is open, type in **ip addr** and press **Enter**.

On Windows, the output should be almost equal to:

```
Selecteren C:\WINDOWS\system32\cmd.exe
Connection-specific DNS Suffix . :
Wireless LAN adapter Wi-Fi:
Connection-specific DNS Suffix . :
Link-local IPv6 Address . . . . . : fe80::9498:f96b:14b8:694e%11
IPv4 Address. . . . . : 192.168.1.3
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . : 192.168.1.1
Tunnel adapter LAN-verbinding* 13:
Media State . . . . . : Media disconnected
Connection-specific DNS Suffix . :
Tunnel adapter Reusable ISATAP Interface {AACAFEEC-A6BF-49C5-9579-89773AA3C613}:
Media State . . . . . : Media disconnected
Connection-specific DNS Suffix . :
```

Figure 3 Output of Using the "ipconfig" Command

What we are interested in, is the IPv4 Address and Subnet Mask. Each node in the network is assign a unique address which it can use to communicate. This is referring to the IPv4 Address. The Subnet Mask tells us more about the number of IPv4 Addresses available on the network. The Raspberry Pi has one of them. Go to <http://www.subnet-calculator.com/>. Use the IPv4 Address and Subnet Mask discovered through the previous step and fill them in.

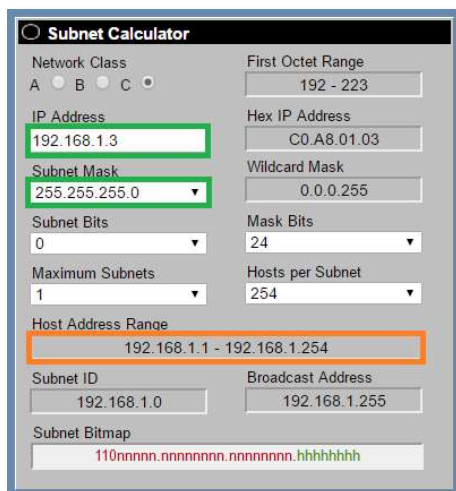


Figure 4 <http://www.subnet-calculator.com/>

What we are interested in is the Host Address Range. This information can be used in combination with the Advanced IP Scanner or LanScan.

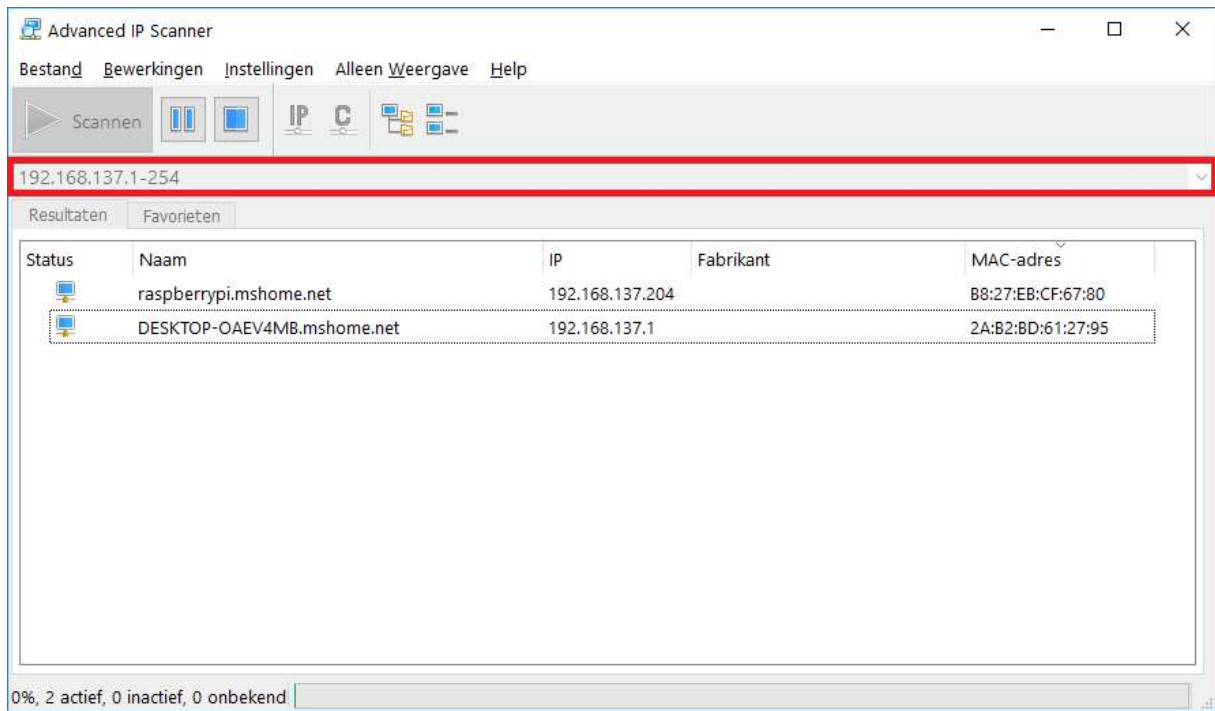


Figure 5 Output from Advanced IP Scanner

On Linux it possible to achieve the same through a package called **nmap**. The **nmap** Package can be installed through the Terminal with the following command: **sudo apt-get install nmap** for Debian based systems and **yum install nmap** for systems based on Red Hat. Based on the output from **ip addr** and the Host Address Range returned by the Subnet Calculator you can use **nmap 192.168.0.\*** to scan your network for the Raspberry Pi.

By default the MAC Address of the Raspberry Pi will start with B8:27:EB. Search for such device in the output from Advanced IP Scanner / LanScan or **nmap** and write down the IP Address.

## Connect to the Pi through SSH

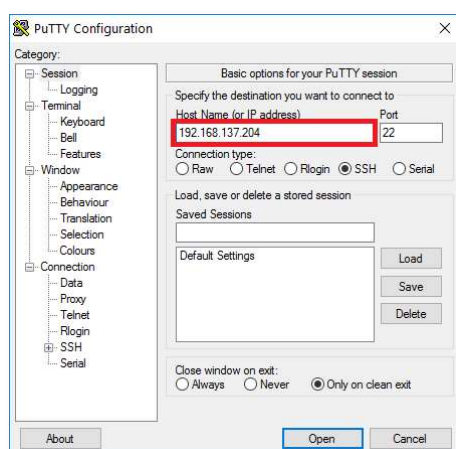


Figure 6 Putty on Windows

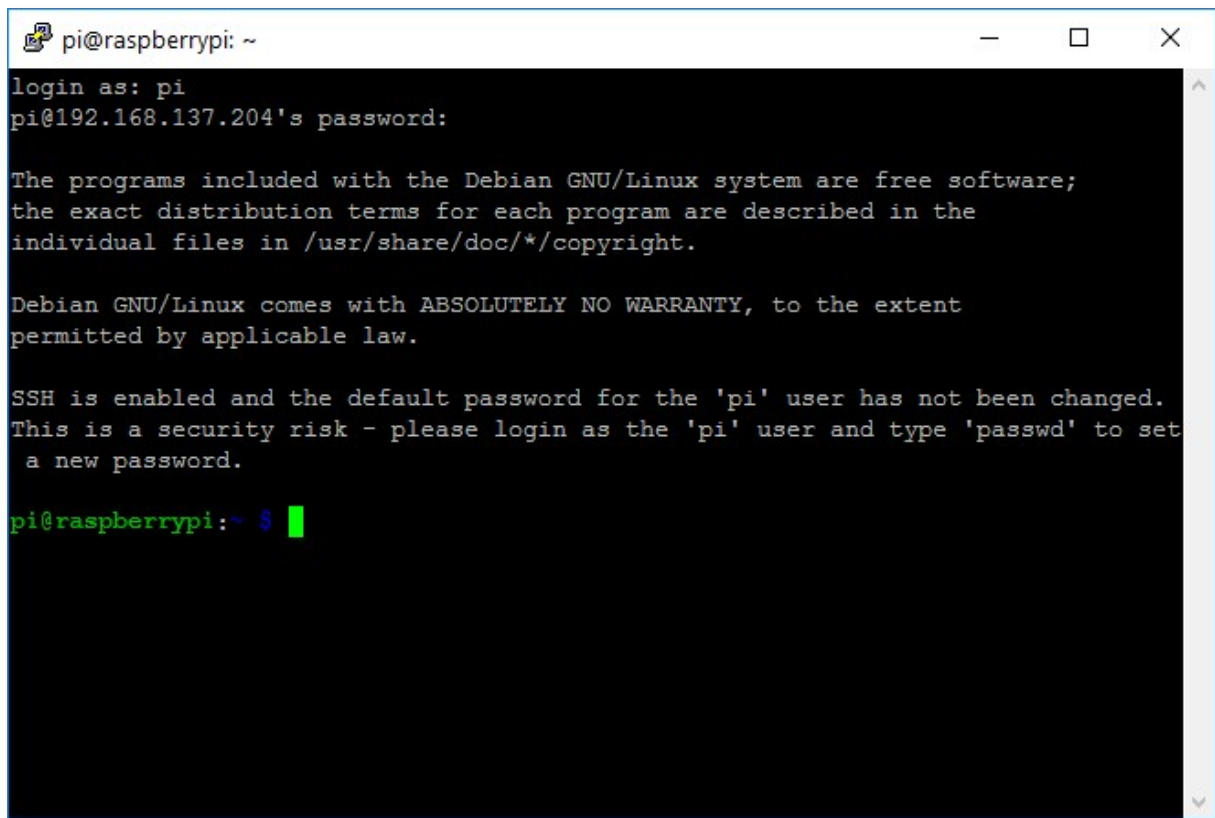
At this point you should have been able to find the IP Address of the Pi. On Mac and Linux, open a Terminal and type **ssh** followed by the IP Address of the Pi Zero and press **Enter**. If you are using Windows, you'll need to install a program such as Putty. Fill in the IP Address and click on **Open**, as shown on the left.

In the following step it will probably ask you if you want to trust the SSH Key used for the connection. Press or type in Yes.

The following screen should present a "Login as: " sentence. Type in **pi** and press **Enter**. Then use

**raspberry** as password.

You're now successfully connected to the Pi!



```
pi@raspberrypi: ~
login as: pi
pi@192.168.137.204's password:

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.

SSH is enabled and the default password for the 'pi' user has not been changed.
This is a security risk - please login as the 'pi' user and type 'passwd' to set
a new password.

pi@raspberrypi:~ $
```

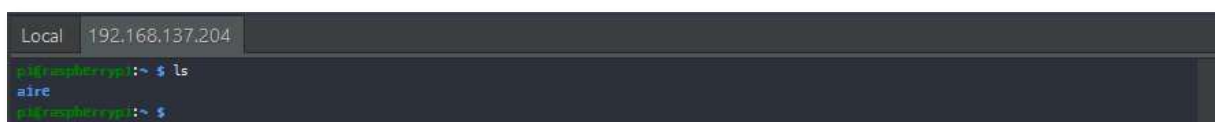
Figure 7 Successfully Connected to the Pi through Putty

## Password

You are now connected to the Pi by using the default user and password. It's best practice to at least configure another password. Type in **passwd** and press **Enter**. Go through the procedure to change the default password.

## Setup the Software

The entire project is located on GitHub and can be viewed via <https://github.com/del-aire/aire>. Before being able to download the source code, you'll need to install Git. Login to the Pi through SSH. Update the package list with **sudo apt-get update** and then run **sudo apt-get install git**. Now run **cd ~ && git clone https://github.com/del-aire/aire.git**, this will install the required software to collect data from the sensor(s) connected to the Pi. Once finished, use the **ls** command and check whether the **aire** directory is listed.



```
Local 192.168.137.204
pi@raspberrypi:~ $ ls
aire
pi@raspberrypi:~ $
```

Figure 8 Output from the "ls" Command

Each child directory of **aire** has a specific purpose, namely:

App	A Server that is accessible through the Browser to see Real-Time information.
Service	The actual Collection Service that will collect data from each sensor and store it in a file. It is using a Rolling File mechanism to ensure that the SD Card isn't filled with data. The oldest data is deleted whenever deemed necessary.
WiFi	The possibility to start a Hotspot on the Pi so that everyone can access the App to a specific network. *The people connected to the Hotspot will be on the same network which is transmitted by the Pi.

The bare required minimum is to enable the Collection Service, otherwise there isn't a real point in using Aire.

## Installation of Node.js & Node-Gyp

Go to your Home Directory through using `cd ~` and then navigate to the directory where we've cloned the Git repository with `cd aire`. Use the `chmod +x ./install.Node.sh` to ensure we can execute that Shell Script. Execute it with `sudo ./install.Node.sh` in order to install Node and npm. To update npm to the latest version use `npm install -g npm@latest` and `sudo npm install -g node-gyp` to install Node-Gyp.

[ There is an issue with running `npm install -g npm(at)latest` and `sudo npm install -g node-gyp` through a Shell Script. Therefore, at the moment you have to install these manually. ]

## Collection Service

### Introduction

The Collection Service is used to collect data from the actual sensor(s) connected to the Pi. How to connect and configure each sensor is described later on in this document under the section "Connect Each Sensor"

### Installation

Go to `~/aire/service` and run `sudo apt-get install wiringpi`. This is a library used by the C++ Addon to control the DHT Humidity and Temperature Sensor. Then execute the following command: `/opt/nodejs/bin/node-gyp configure && /opt/nodejs/bin/node-gyp build` to build the C++ Addon. Now run `sudo npm install` to install the required dependencies and devDependencies by the Collection Service.

[ If you encounter an error while installing `serialport`, try again by running `sudo npm install -g --unsafe-perm serialport` first followed by `sudo npm install`. ]

### Enable on Start-up

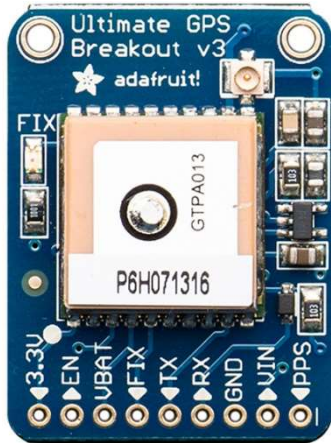
If you want the Collection Service to collect data whenever the Pi is up and running, it is highly recommended to install a Daemon. Go to `~/aire/service` and execute `chmod +x`



`./install.and.enable.Service.sh`. Then run `sudo ./install.and.enable.Service.sh` to enable the Service to run whenever the Pi is started.

## Connect Each Sensor

### Adafruit Ultimate GPS Breakout



#### Introduction

The Adafruit Ultimate GPS Breakout can be used to track your position around the globe. Due to being able to do 10 location updates a second, it is ideal for high speed, high sensitivity tracking.

#### Wiring

Connecting the Adafruit Ultimate GPS Breakout to the Raspberry Pi will be done through using the UART pins on the Pi.

It is also possible to connect the GPS Breakout through using a USB to TTL serial cable, but because the Raspberry Pi Zero is our main target device, due to its limited USB connectivity and considering the fact that it would require an additional cable, using UART seems to be a good option.

[ Insert Image ]

It should be noted that it is required to connect the RX pin on the Adafruit Ultimate GPS Breakout to the TXD pin on the Raspberry Pi and the TX pin on the GPS Breakout to the RXD pin on the Raspberry Pi.

The cross connection between the TXD and RX or the RXD and TX pin is required because the TX / TXD pin is used for transmission while the RX / RXD pin is used for receiving information.

#### Software Setup

Open `/boot/cmdline.txt` with `sudo nano /boot/cmdline.txt` and remove the following structure: `console=serial0,115200`. This will disable the possibility to access the console over the TX and RX pin on the Pi. Save the file by using **CTRL + O** and quit with **CTRL + X**. Then disable the TTY Service with `sudo systemctl stop serial-getty@ttyS0.service` and `sudo systemctl disable serial-getty@ttyS0.service`. For the Pi 3 there is also an additional step required. Run `sudo nano /boot/config.txt` and add `enable_uart=1` at the bottom of the file. Save with **CTRL + O** and exit through using **CTRL + X**. Reboot the Pi with **sudo reboot now!**

To test whether you successfully configured on the Adafruit Ultimate GPS Breakout you can use `cat /dev/ttyS0`.

Ex. `$GPRMC,223835.000,A,5345.6248,N,00242.4512,W,1.04,299.82,240517,.,,A*70`



[ In case you're using a TTL Cable you can configure the right device in the `~/aire/service/config.Sensor.json` configuration file. Search for a key called **sensorName** with value **AdafruitUltimateGps.Gpgga** and change the value of **serialInterface**. ]

## National Marine Electronics Association

The hardware interface of the Adafruit Ultimate GPS Breakout is designed to output data according to the NMEA standard. The NMEA standard consists of multiple predefined sentences, each of them contain specific information.

For determining the position around the globe the \$GPGGA sentence contains a lot of interesting data. The \$GPGGA NMEA sentence is being parsed by the air quality monitor to determine its location.

### *\$GPGGA Sentence*

Global Positioning System Fix Data

Name	Example	Description
Identifier	\$GPGGA	
Time	151012	15:10:12 Z
Latitude	5345.6802	53° 45' 40.8179" N
North or South	N	
Longitude	00242.4430	2° 42' 26.5828" W
East or West	W	
Quality  0 = Invalid 1 = GPS Fix 2 = DGPS Fix	1	
Satellite count	05	
Horizontal Dilution of Precision	1.5	
Altitude	26.0, M	Altitude above or below mean sea level.
Height of geoid above WGS84 ellipsoid	52.17, M	

Time since last DGPS update	Blank	
DGPS reference station identifier	Blank	
Checksum	*{Number}	Can be used to verify that the transmission was successful and doesn't contain any errors.

Translating this table into a National Marine Electronic Association sentence would result in: \$GPGGA,151012,5345.6802,N,00242.4430,W,1,05,1.5,26.0,M,52.171,M,,,\*{Number}

Source: <http://aprs.gids.nl/nmea/>

## DHT11 / DHT22 Temperature and Humidity Sensor

### Introduction

Both the DHT11 and DHT22 Sensor can be used to measure Humidity and Temperature. The DHT22 has better accuracy and a wider measuring range and therefore has the preference above the DHT11.



Figure 9 Picture of the DHT22

### Wiring

[ Insert Image ]

### Software

If you have used another GPIO pin or the DHT11 Sensor instead of the DHT22 open the `~/aire/service/config.Sensor.json` file and search for `"sensorName": "Dht"`. Change the `sensorType` to 11 in case of the DHT11 and configure the `gpioPin` if you are using another pin.