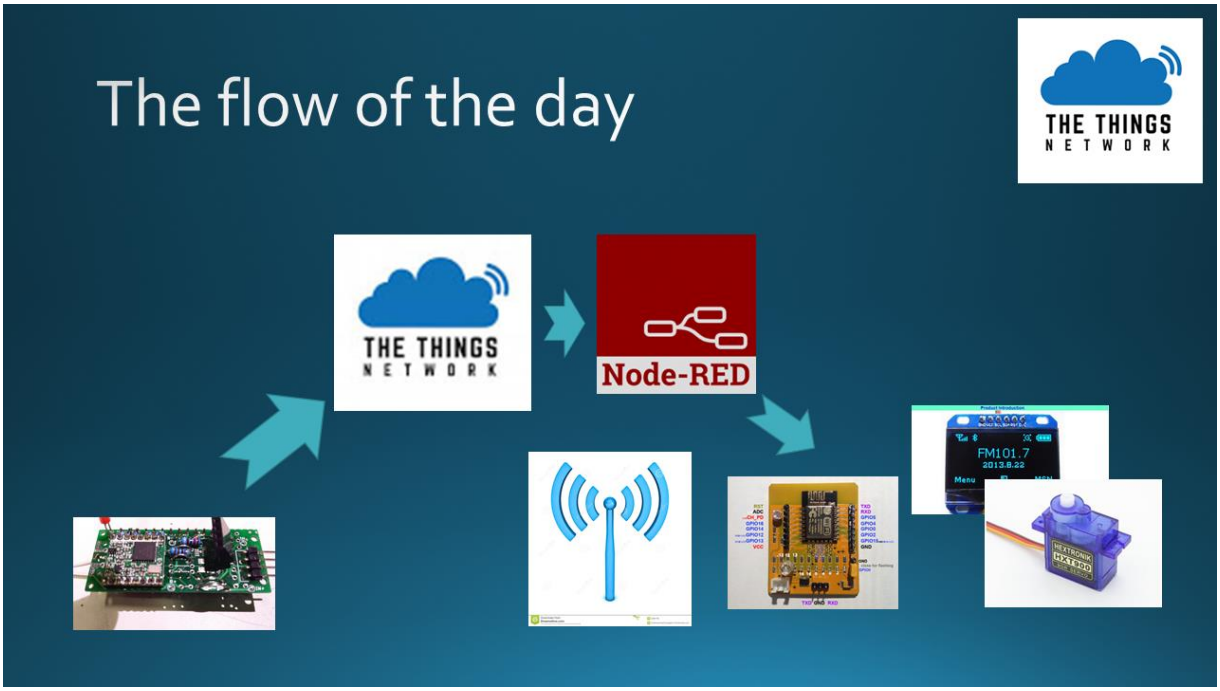
Workshop IoT integration

We will create an IoT application containing four main parts:

- A TTN Node
- The TTN back-end
- A Node-Red application on your PC
- A Wifi connected actuator based on an ESP8266

Building a TTN node was part of one of our previous workshops. We will use the temp_humidity example, posting temperature and humidity to TTN.

This work is licensed under the Creative Commons Attribution-NonCommercial 4.0 International License. To view a copy of this license, visit http://creativecommons.org/licenses/by-nc/4.0/ or send a letter to Creative Commons, PO Box 1866, Mountain View, CA 94042, USA.



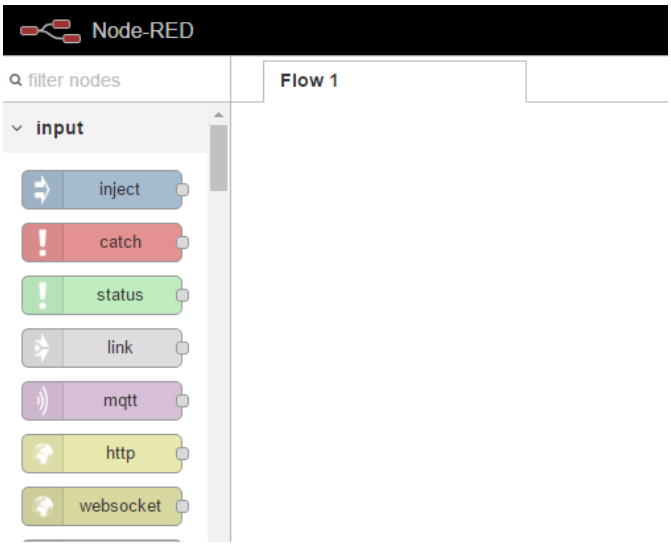
Installing NodeRed on a PC

Follow the instructions on https://nodered.org/docs/getting-started/installation to install NodeRed on your system.

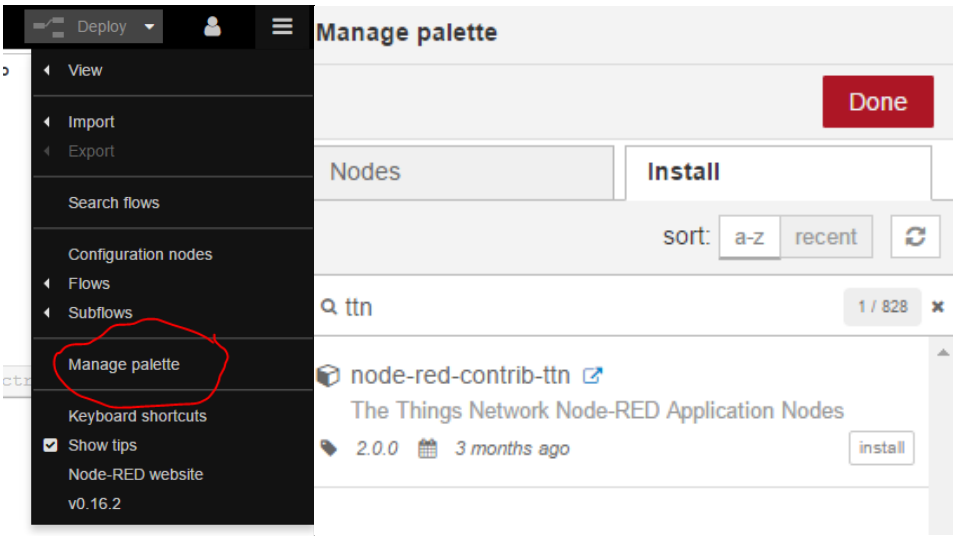
- On Windows it will be installed on your home directory.
- Enter the home directory in a command-line and start:

```
Node-red
```

- The log will show the address of the node-red web interface: http://127.0.0.1:1880/
- Open that address



- Now select the top right menu and select manage palette:



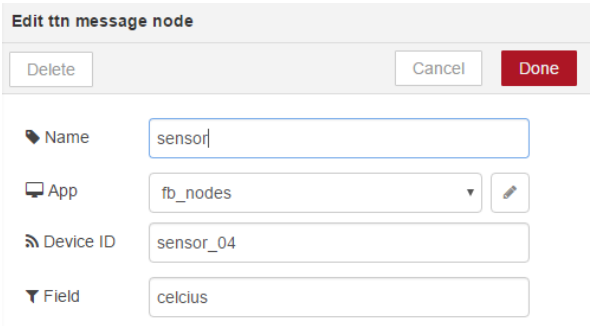
- Search for TTN in the Install tab and click install to install the TTN nodes.
- Click again install on the warning screen.
- 'Done' to exit the palette management.
- TTN nodes can now be added in your Node-Red flow.

This work is licensed under the Creative Commons Attribution-NonCommercial 4.0 International License.

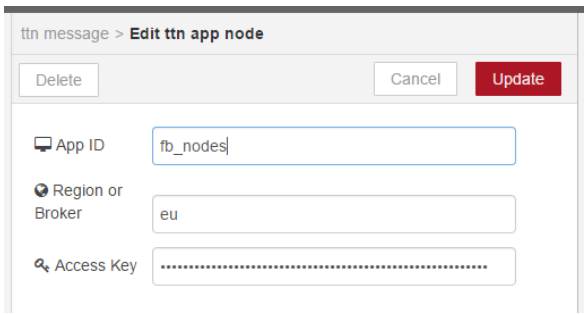
Node Red introduction

We start with a clean 'flow'. Remove any flows with the Menu->Flows->Delete flow.

- Add a TTN **message** node by drag and drop it on your flow screen.
- By double clicking on the TTN message node you can change the values:



The name can be any name, App refers to your TTN console. With the Pencil you can add your application here:



- App ID is the written name 'Application ID' from the TTN console
- Region is eu, it is already filled in grey, but you should type it in as well!
- Access Key is a copy of the access key of your application (access key of application, default key).
- Check Update and select your App in the previous screen.
- Device ID is the Device ID in TTN console

You can select celcius or humidity in 'field' if you use the temp_humidity sketch. If you leave 'field' empty all the fields will be shown. Beware that you use the same payload function in TTN as used in the previous nodes workshop!! If you do not have a payload function loaded, you can use an empty 'field', all the output will be shown. This is the payload function of the previous workshop which can be entered in the TTN Console (so not in Node-Red!):

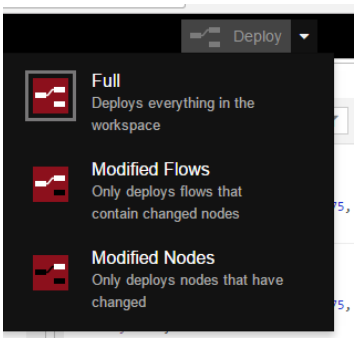
```
function Decoder(bytes, port) {

var humi = 20+5*((bytes[1] >> 2) & 0x0F);
var temperature = -2400+6.25*(((bytes[1] & 0x03) << 8) | bytes[0]);
return {
humidity: humi,
celcius: temperature / 100.0
};
}
```

This work is licensed under the Creative Commons Attribution-NonCommercial 4.0 International License.

- Now add a ‘Debug’ output node to your flow and connect the both with a wire:

- Select msg.payload by double clicking on the debug node and activate it by clicking on the button on the right.
- Now activate your flow with Deploy->Full:



You will see the results of your node appear in the right debug column.

This work is licensed under the Creative Commons Attribution-NonCommercial 4.0 International License.

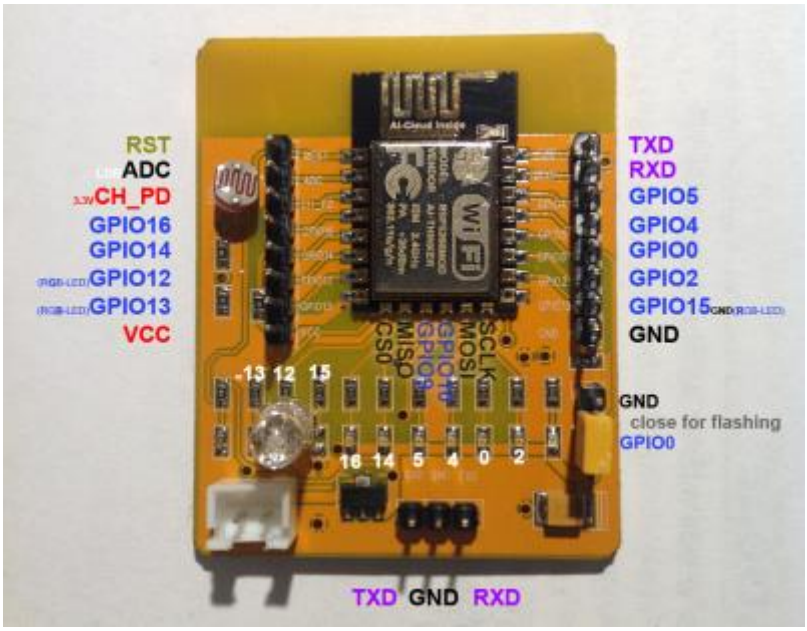
Install ESP Easy

Parts used here can be ordered at http://www.tinytronics.nl/shop/Things-Network-IOTSolution-Workshop

The set contains a ESP module with battery holder, a servo, a relay, an OLED display, a FTDI with 3.3v option and some jumper wires.

We will install an ESP control program to manage the ESP module. First connect the module to your PC with the FTDI adapter:

- Connect your FTDI adapter after selecting 3V on it with the jumper.
- The wiring is straight, so connect GND->GND (blue), RX->RX (brown) and TX->TX (green).
- For programming you need to mount the jumper on the two-pin connector (shown in the lower right corner on the picture below. You can also connect the two pins with a jumper wire.
- Now connect the FTDI adapter to your PC and put the batteries in the holder. The module is 'power hungry' (80mA average): use good batteries or results will be unpredictable.
- Do NOT power the module from the FTDI adapter! The used FTDI adapter is only offering 50mA max on 3.3V!



Follow the instructions on:

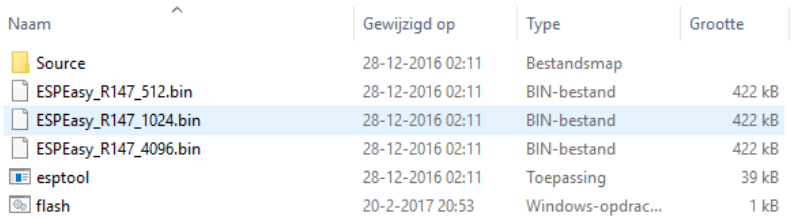
https://www.letscontrolit.com/wiki/index.php/Tutorial_ESPEasy_Firmware_Upload

Fast track, for Windows PC users:

Download and unpack http://www.letscontrolit.com/downloads/ESPEasy_R147_RC8.zip

In the download directory start the flash tool:

This work is licensed under the Creative Commons Attribution-NonCommercial 4.0 International License.



Naam	Gewijzigd op	Type	Grootte
Source	28-12-2016 02:11	Bestandsmap	
ESPEasy_R147_512.bin	28-12-2016 02:11	BIN-bestand	422 kB
ESPEasy_R147_1024.bin	28-12-2016 02:11	BIN-bestand	422 kB
ESPEasy_R147_4096.bin	28-12-2016 02:11	BIN-bestand	422 kB
esptool	28-12-2016 02:11	Toepassing	39 kB
flash	20-2-2017 20:53	Windows-opdrac...	1 kB

- Com port: you can check the COM port in the Arduino IDE or device management, fill in the right number
- Flash size is 4096
- Build version is 147
- Put the flash jumper on your module and start the upload. You may reset the board by connecting the RST pin as seen in the picture to GND.
- The flashing starts

```
Uploading 431376 bytes from ESPEasy_R147_4096.bin to flash at 0x00000000
        erasing flash
        size: 069510 address: 000000
        first_sector_index: 0
        total_sector_count: 106
        head_sector_count: 16
        adjusted_sector_count: 90
        adjusted_size: 05a000
        espcomm_send_command: sending command header
        espcomm_send_command: sending command payload
        setting serial port timeouts to 10000 ms
        setting serial port timeouts to 1000 ms
        espcomm_send_command: receiving 2 bytes of data
        writing flash
```

- ....

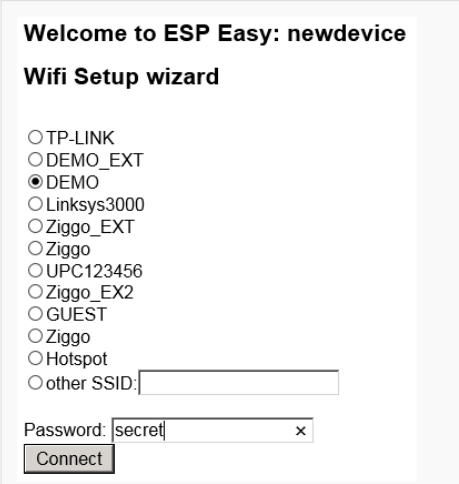
It is also possible to download the firmware through the Arduino IDE (for Linux or Mac users): https://www.letscontrolit.com/wiki/index.php/Tutorial_Arduino_Firmware_Upload. Installing the firmware is only done once, configuration or reset can be done through web or serial interface.

Applying wifi settings

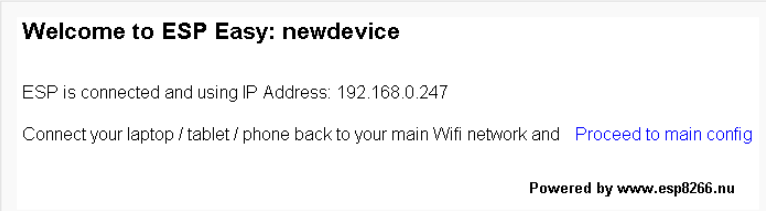
After installing the firmware, the module will reboot and start as an access point ‘ESP_0’

- We have to find a wifi channel ESP_0. At this moment you should sync with other workshop attendees and power off your node until a free slot is available, or use the serial commands (wifissid and wifikey) to enter the wifi credentials as shown below.
- Connect your system to the wifi SSID ESP_0 and go to your browser.
- Now enter the Wifi Credentials:

This work is licensed under the Creative Commons Attribution-NonCommercial 4.0 International License.



The module will reconnect, and from that moment it can work together with other nodes because it is getting its own IP address. Take note of that address!



Now you can reconnect to the 'normal' wifi and check 'Proceed to main config'

Your module is activated, and can be controlled by http requests:

http://<your ESP address>/control?cmd=gpio,12,1 will light the RGB LED

- Connect the servo by putting three male jumper wires into the servo's connector. The Brown connection is GND, connect it to the GND pin of the ESP module board. The Red connection goes to the spare 5V pin of the FTDI. The signal (orange) goes to GPIO13 on the ESP module board.
- http://<your ESP address/control?cmd=Servo,1,13,100 in your browser will move the servo. 100 is the position, you can change it from 0-180

Using the Serial Port to change configuration

You can use the serial interface to change the configuration as well. You can find the commands here. Some options:

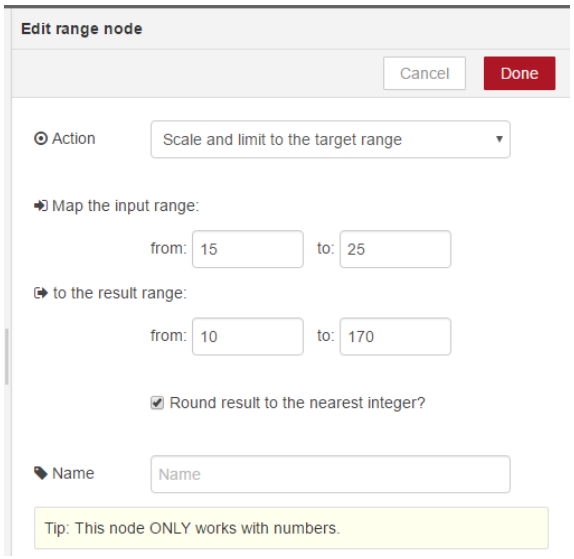
- Reset: will erase the flash memory (not the firmware, but the user settings). The module will reboot in 'ESP_0' access point mode and you have to restart the configuration
- WifiSSID <Your Wifi SSID>, wifissid WORKSHOP : set the SSID to connect
- Wifikey <your wifi password>, wifissid 1234567890: set the password of the wifi connection
- Wificonnect: connect with the given credentials
- Save: save the settings in flash
- Settings: show current settings, including the IP address
- ip <ip address>, ip 192.168.0.201: set a fixed IP address, beware of DHCP address ranges!

This work is licensed under the Creative Commons Attribution-NonCommercial 4.0 International License.

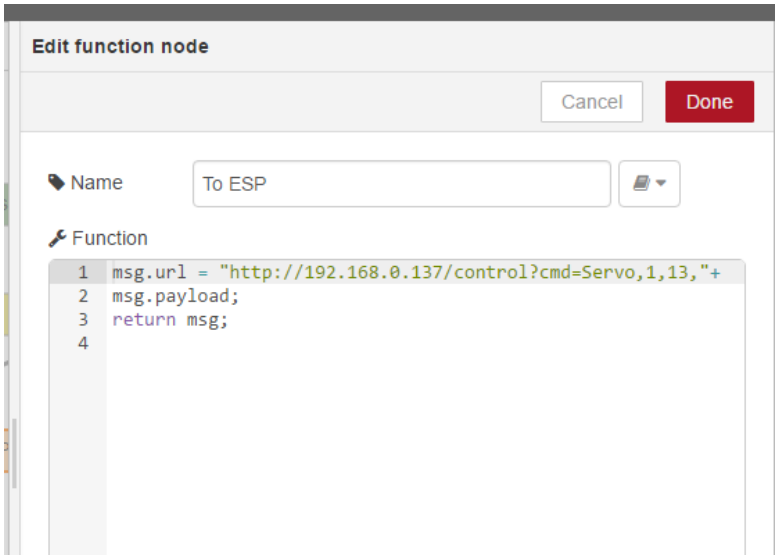
NodeRed Application

Now we are going to control the ESP according to the results of the TTN sensor.

- Add a range function and connect it to the sensor output.
- Double click on the ‘range’ node:

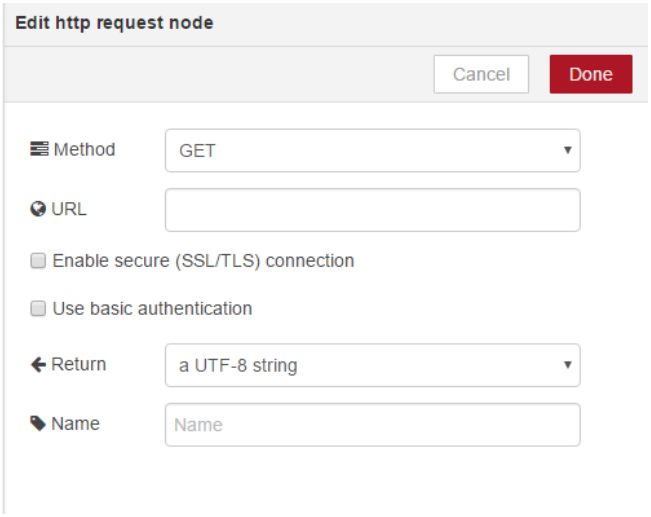


- We map the input results (Celsius temperature) from 15-25 to a servo position of 10 to 170 degrees, thereby creating an analogue gauge thermometer!
- Add a ‘function’
- Double click on the ‘function’ node:

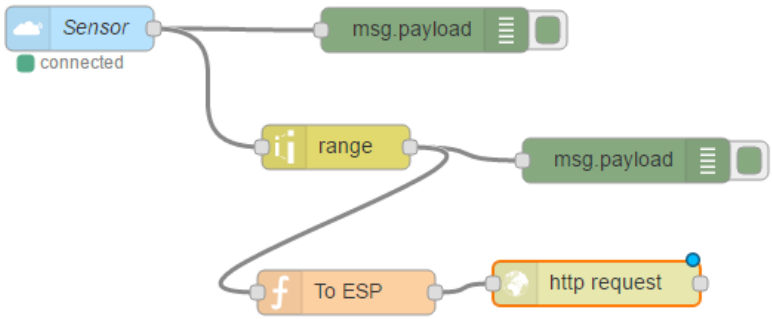


- Change the IP address according to your ESP.
- The last node we will add is a ‘HTTP Request’ to the ESP

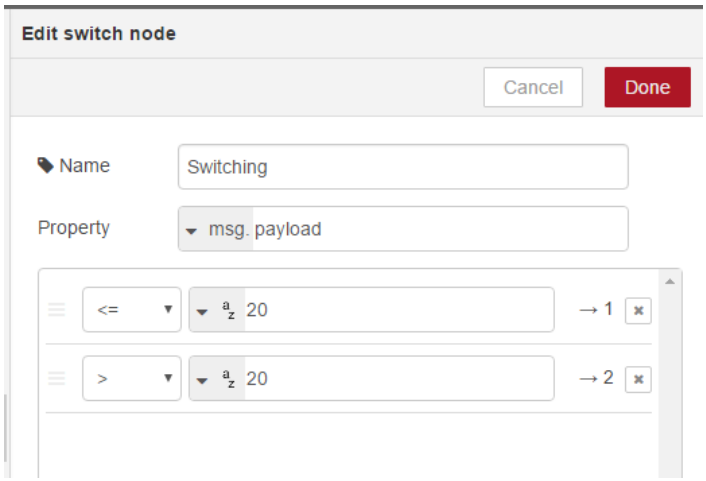
This work is licensed under the Creative Commons Attribution-NonCommercial 4.0 International License.



We constructed the URL already in the previous function and we connect the blocks. We can use ‘debug’ nodes to get some debug information:

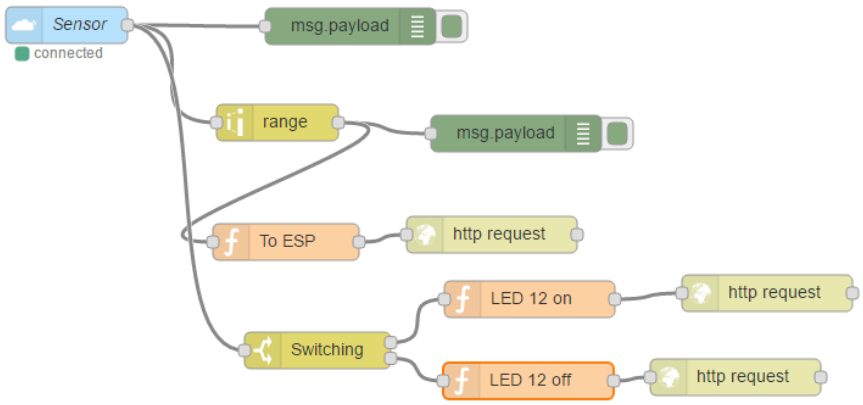


Now we add a temperature switch for a heater or blower. We add the ‘switch’ node:



- Add two switches, less than 20 and greater than 20 (this is the temperature we are looking at). The switch now has two ‘exits’. The upper one is getting active when the temperature is low. The lower one when the temperature is higher.
- Now you can copy the URL function two times, and the HTTP request two times. The flow will look like this:

This work is licensed under the Creative Commons Attribution-NonCommercial 4.0 International License.



- Change the LED pin functions:
  - msg.url = http://<YOUR ESP ADDRESS>/control?cmd=GPIO,12,1 to put the LED on
  - msg.url = http://<YOUR ESP ADDRESS>/control?cmd=GPIO,12,0 to put the LED off
- You may connect your relay to output 12, it will switch when the temperature changes.

Attach the display

In the interface of the ESP module add a display under 'Devices':

Task Settings	Value
Device:	Display - OLED SSD1306 ?
Name:	display
Delay:	60
IDX / Var:	1
I2C Address:	3C
Rotation:	Normal
Display Size:	128x64
Line 1:	%ip%
Line 2:	

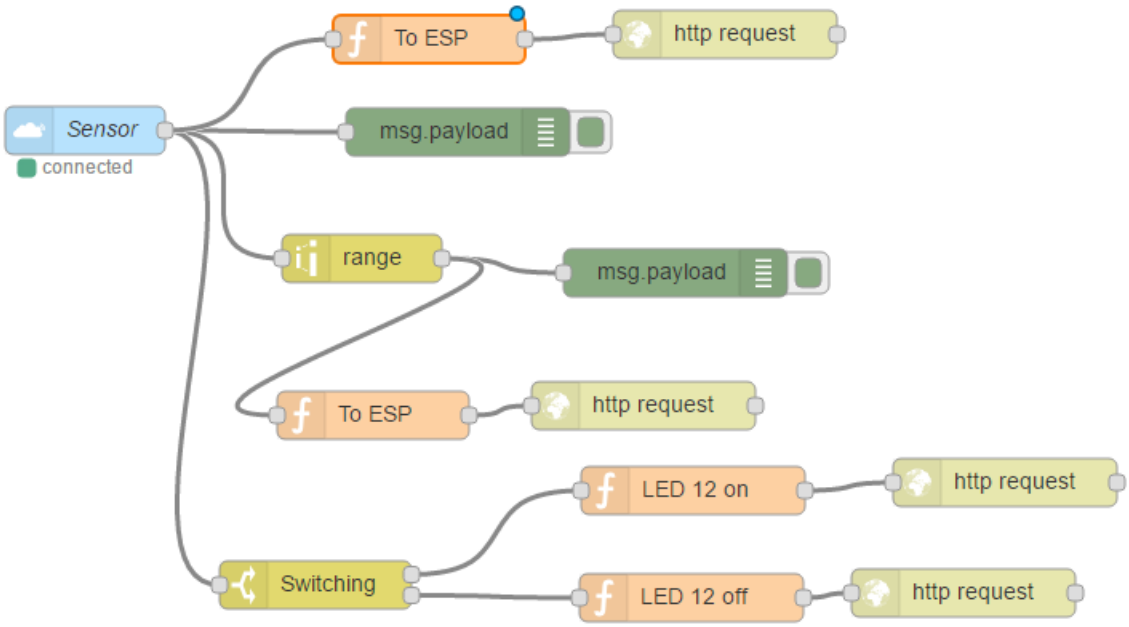
Connect:

- VCC->VCC
- GND->GND
- SDA->GPIO5
- SCL->GPIO4
- The display will show the IP address of your module.
- Now again copy the 'function' and 'HTTP request' node.
- Connect the 'function' node with your TTN sensor and the 'HTTP request' with the 'function' node.
- Change the 'function':

```
msg.url = "http://<YOUR ESP ADDRESS/control?cmd=oled,3,1,"+
msg.payload;
return msg;
```

This work is licensed under the Creative Commons Attribution-NonCommercial 4.0 International License.

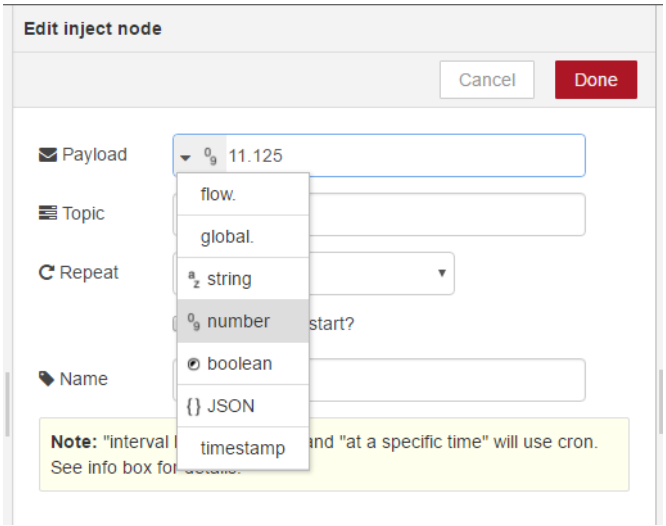
Now the measurement of your sensor will show up on the OLED display!



Simulating a TTN node

If you do not have a TTN node there are three possibilities:

- Buy or build one
- Use the TTN application ‘access key’ from someone who wants to share the results of his node. Beware of the payload functions, you may not be able to alter them.
- Use a ‘Inject’ node replacing the TTN node in your Node-Red flow. This is an input node object ‘inject’, and change Payload to ‘number’. You may enter the temperature value here. The object has a ‘button’, by pressing this button the object will insert the given value into your flow.



This work is licensed under the Creative Commons Attribution-NonCommercial 4.0 International License.

Reference

- Node-Red: https://node-red.org , http://noderedguide.com/
- Node-Red TTN docs: https://www.npmjs.com/package/node-red-contrib-ttn
- Node-Red and TTN: https://hansboksem.wordpress.com/2017/03/02/thethingsnetwork-weather-station-with-node-red-and-domoticz/
- ESP Easy: https://www.letscontrolit.com/wiki/index.php/ESPEasy
- 'free' cloud-based Node-Red: https://fred.sensetecnic.com/ (the examples in this tutorial connecting the local ESp8266 are currently only supported with local running Node-Red). This instance will run for max 24 hours.
- Change the default SPI connection of the OLED display to I2C: http://electronics.stackexchange.com/questions/164680/ssd1306-display-spi-connection-or-i2c-according-to-resistors

This work is licensed under the Creative Commons Attribution-NonCommercial 4.0 International License.