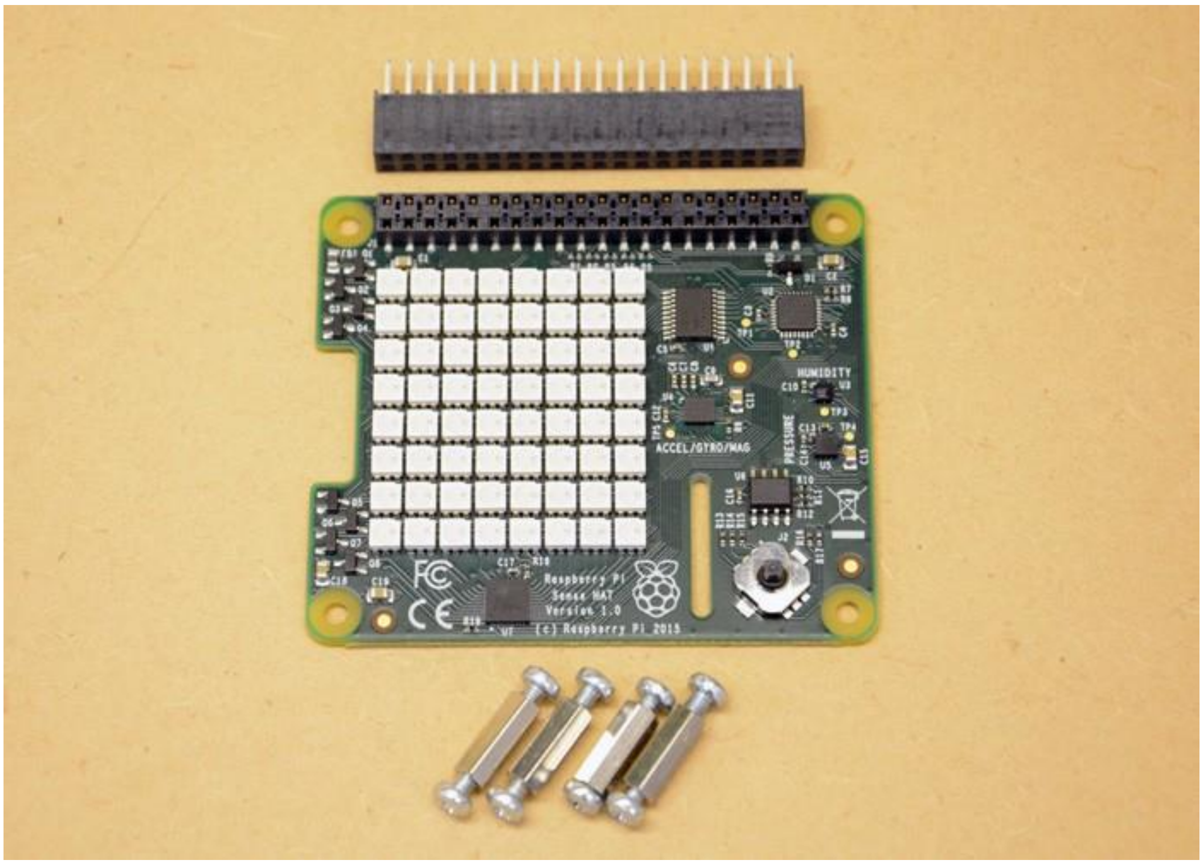
Plug and Play IoT with the Pi Sense HAT

Posted by **stuartChilds** on Tue, Sep 15 2015



Creating connected sensors and displays with MQTT and Node-RED.
Packed with on-board features including a gyroscope, humidity and pressure sensors, not to mention an 8x8 RGB LED matrix, the Sense HAT for the Raspberry Pi makes interfacing with the outside world easier than ever.

Destined for Space

Originally designed for Astro Pi – a project that will see two Raspberry Pis sent into space – these boards have been in the hands of UK schoolchildren and teachers since the launch of the project in January 2015. This means that there are already comprehensive learning resources and code examples for the Sense HAT available online.

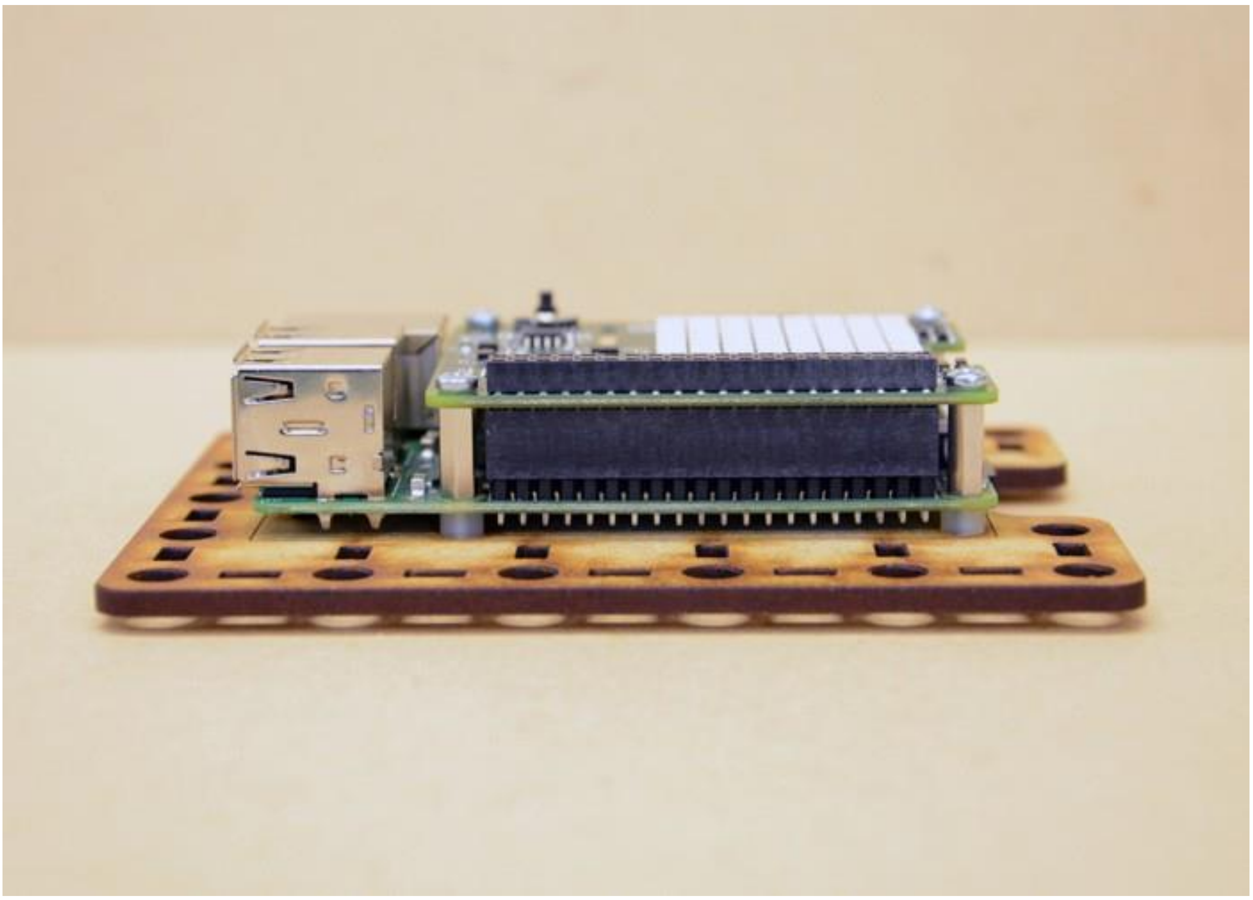
Students were encouraged to submit project proposals in a bid to get their code executed in space, on the two very special Pis being sent up to the International Space Station towards the end of 2015.

General availability

Sense HATs have now gone into full production and are an official Raspberry Pi product, available to buy and finding their way into the hands of hobbyists and the wider community.

In this post we will set up a Raspberry Pi 2 with a Sense HAT, before using it to build a remote sensor and display node connected to a Node-RED system (that we built in aprevious post).

Installing the Sense HAT



First, ensure the Pi is turned off and power lead disconnected. Mount the Sense HAT following the instructions here. Included are 4 spacers to hold the HAT securely in place, a nice touch and something often overlooked with add-on boards.

Power up the Pi and you should be treated to a colourful illumination of the RGB LED matrix. If you experience problems with booting now, you may need to check that your power supply can handle the extra current needed to power the Sense HAT along with the Pi (though it is only an extra 40mA or so).

Next ensure Raspbian is up to date:

$ sudo apt-get update

$ sudo apt-get upgrade

Install the Sense HAT software package:

$ sudo apt-get install sense-hat

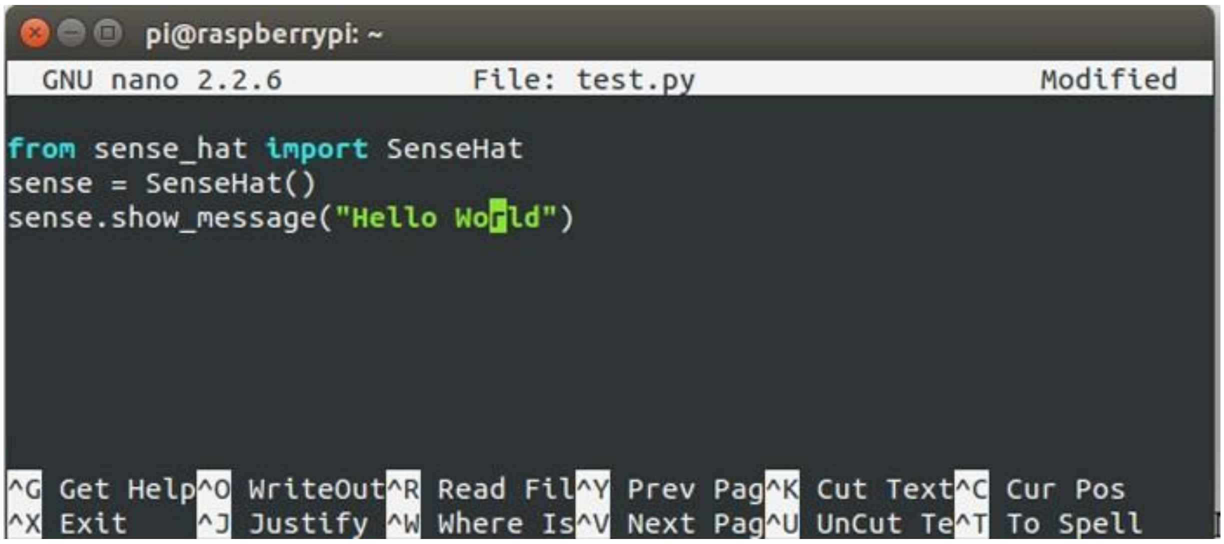
And Pillow, to ensure compatibility with Python 3:

$ sudo pip-3.2 install pillow

Then reboot the Pi to complete the installation:

$ sudo reboot

Testing the Sense HAT



```
pi@raspberrypi: ~
  GNU nano 2.2.6             File: test.py                        Modified

from sense_hat import SenseHat
sense = SenseHat()
sense.show_message("Hello World")









^G Get Help^O WriteOut^R Read Fil^Y Prev Pag^K Cut Text^C Cur Pos
^X Exit    ^J Justify ^W Where Is^V Next Pag^U UnCut Te^T To Spell
```

First, let's check that the LED matrix is working correctly by using a simple Python program. Use a text editor to create a file (the *nano* editor is used in this example as it is simple to use):

$ nano test.py

Once in the editor, add the following 3 lines:

from sense_hat import SenseHat

sense = SenseHat()

sense.show_message(“Hello World”)

Save the file and quit the editor.

Now execute the file and watch the text scroll on the LED display on the HAT. Note that we will use Python 3 throughout this post:

$ python3 test.py

If this works it indicates that the HAT is functioning correctly, but we might want to also check that we can get data from one of the sensors, just to be sure. For example, we can get temperature data from the combined humidity and temperature sensor by creating a new file (e.g. *temp.py*) with the following lines:

from sense_hat import SenseHat

sense = SenseHat()

while True:

temp = sense.get_temperature()

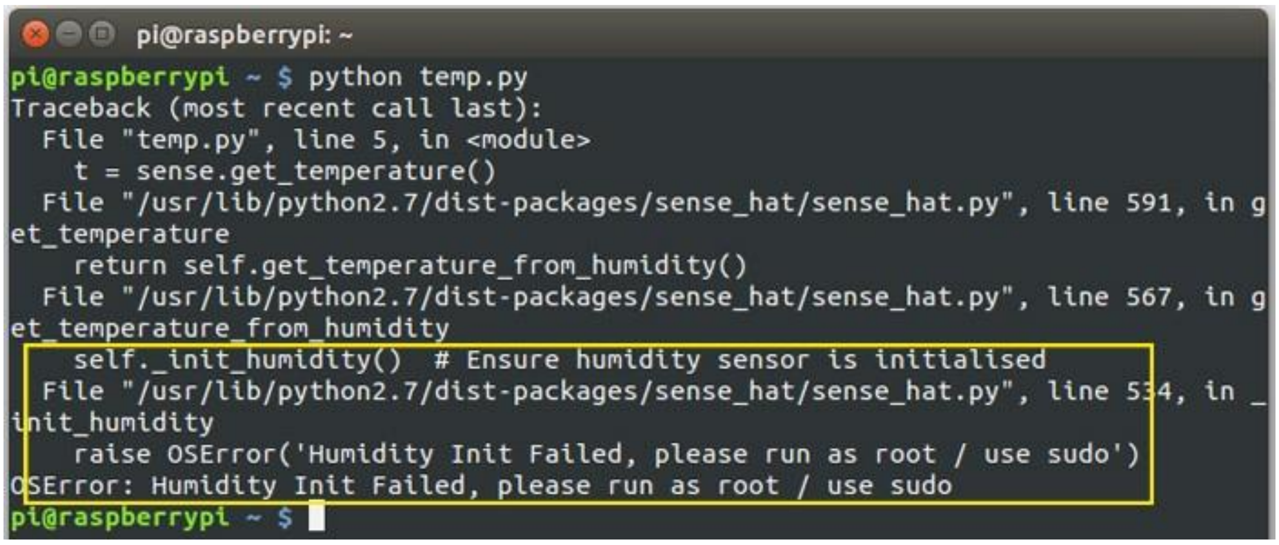
temp = round(temp, 2)

msg = "Temperature = %s" % (temp)

sense.show_message(msg)

Save the file as before and attempt to execute it with the following command:

$ python3 temp.py



```
pi@raspberrypi: ~
pi@raspberrypi ~ $ python temp.py
Traceback (most recent call last):
  File "temp.py", line 5, in <module>
    t = sense.get_temperature()
  File "/usr/lib/python2.7/dist-packages/sense_hat/sense_hat.py", line 591, in g
et_temperature
    return self.get_temperature_from_humidity()
  File "/usr/lib/python2.7/dist-packages/sense_hat/sense_hat.py", line 567, in g
et_temperature_from_humidity
    self._init_humidity()  # Ensure humidity sensor is initialised
  File "/usr/lib/python2.7/dist-packages/sense_hat/sense_hat.py", line 534, in _
init_humidity
    raise OSError('Humidity Init Failed, please run as root / use sudo')
OSError: Humidity Init Failed, please run as root / use sudo
pi@raspberrypi ~ $
```

This gives an error and it appears that the humidity sensor can only be initialised by root:

$ sudo python3 temp.py

Alternatively, we can get a temperature reading from the pressure sensor by changing the following line:

temp = sense.get_temperature()

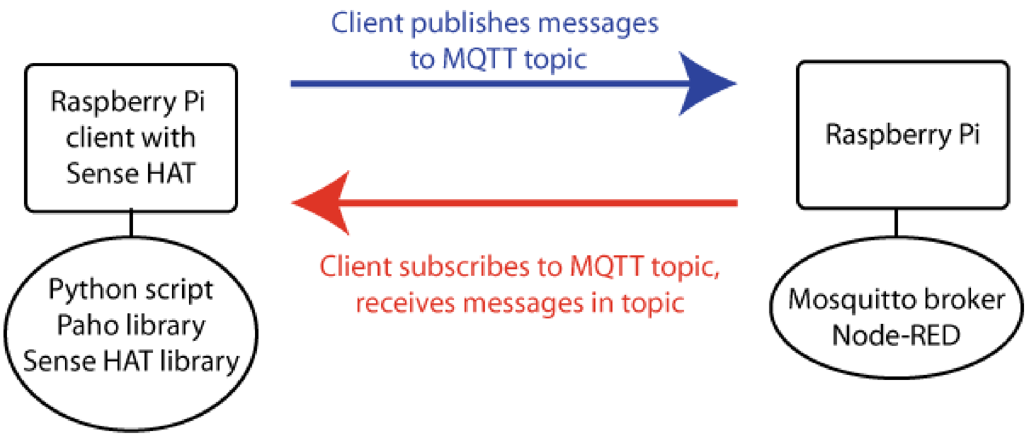
to:

temp = sense.get_temperature_from_pressure()

Due to the proximity of these sensors to the Raspberry Pi, it is worth noting that temperature readings are affected by the heat it generates.

Note how we can get usable data from the on-board sensors very easily, using just a few lines of code thanks to the libraries we installed earlier. For those wishing to look at the libraries in more detail, the source code is up on GitHub.

Simple IoT messaging with MQTT


Client publishes messages to MQTT topic
Raspberry Pi client with Sense HAT
Raspberry Pi
Python script Paho library Sense HAT library
Client subscribes to MQTT topic, receives messages in topic
Mosquitto broker Node-RED

MQTT is a popular Machine-to-Machine (M2M) or Internet of Things (IoT) messaging protocol designed to be lightweight and reliable, suitable for use with microcontrollers and low bandwith links.

It uses a central 'broker' to which clients connect, and can publish and/or subscribe to one or more 'topics', enabling messages to be exchanged between different systems.

Node-RED allows for rapid prototyping of IoT applications, amongst many other uses. It is browser based and enables you to wire a whole host of different inputs and outputs together, with functions and other application logic, including built-in support for MQTT and various sensors.

In a previous post we used the MQTT protocol to send messages between differentplatforms, using a Raspberry Pi running the Mosquitto MQTT broker and Node-RED.

We will now build on this by setting up a new MQTT client with the Pi and Sense HAT that can publish and subscribe to topics on our existing broker.

If you do not already have your own broker running you may wish to set one up. Alternatively there are several public online brokers available, but note that these are used at your own risk!

Publishing to a topic

We will use Paho, a Python MQTT client library, with our script.

Begin by installing the latest version of Paho on the Pi:

```
$ sudo pip-3.2 install paho-mqtt
```

Now we can write a short python script that publishes sensor data to MQTT topics. Create a new file:

```
$ nano clientPublish.py
```

Next paste in the code below:

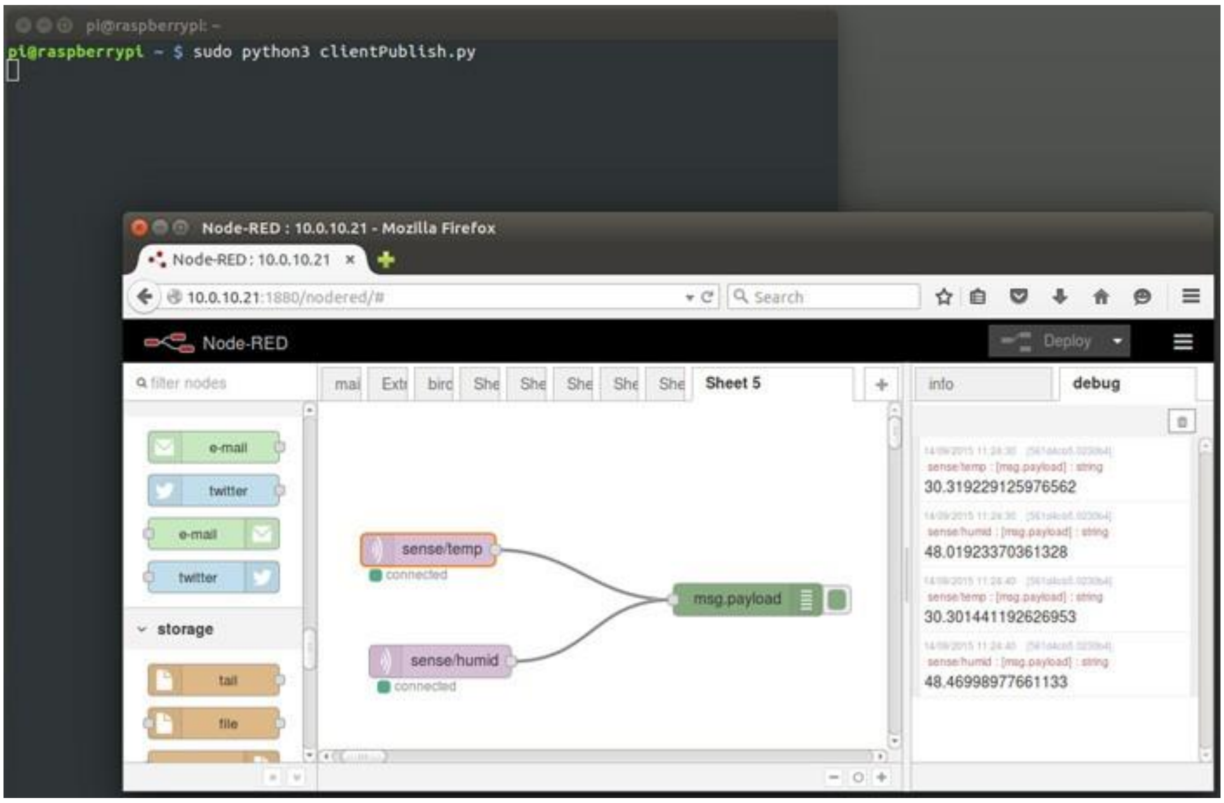
```
import paho.mqtt.client as mqtt

import time

from sense_hat import SenseHat

sense = SenseHat()

# set up mqtt client

client = mqtt.Client("python_pub")

#set mqtt username/pw

client.username_pw_set(username="pi", password="PASS")

#set server to publish to

client.connect("192.168.0.21", 1883)

client.loop_start()

try:

while True:

#publish temp to topic

client.publish("sense/temp", sense.get_temperature())

#publish humidity

client.publish("sense/humid", sense.get_humidity())
```

```
#pause for 10 seconds

time.sleep(10)

#deal nicely with ^C

except KeyboardInterrupt:

print("interrupted!")

client.loop_stop()
```

Ensure that you modify the details for username, password and IP address as necessary. Also note that you will need clients subscribed to the *'sense/temp'* and*'sense/humid'* topics. In our example, we have two '*MQTT in'* nodes connected to the debug console in Node-RED (see screen below).

Save the file and exit the editor. Now execute the script:

```
$ sudo python3 clientPublish.py
```



```
pi@raspberrypi ~ $ sudo python3 clientPublish.py
```

```
14/09/2015 11:24:30  [illegible]
sense/temp : [msg.payload] : string
30.319229125976562
14/09/2015 11:24:30  [illegible]
sense/humid : [msg.payload] : string
48.01923370361328
14/09/2015 11:24:40  [illegible]
sense/temp : [msg.payload] : string
30.301441192626953
14/09/2015 11:24:40  [illegible]
sense/humid : [msg.payload] : string
48.46998977661133
```

This shows we can publish sensor data from the Sense HAT to our MQTT topics and subsequently get that data into Node-RED for further processing as required.

Subscribing to a topic

In our existing Node-RED setup we already had a DS18B20 temperature sensor connected, with readings being published to an MQTT topic, and used this for convenience. You could use any existing topic or create a new one and publish test messages to it using an MQTT client of choice.

On the Sense HAT Pi, create a new file:

```
$ nano clientSubscribe.py
```

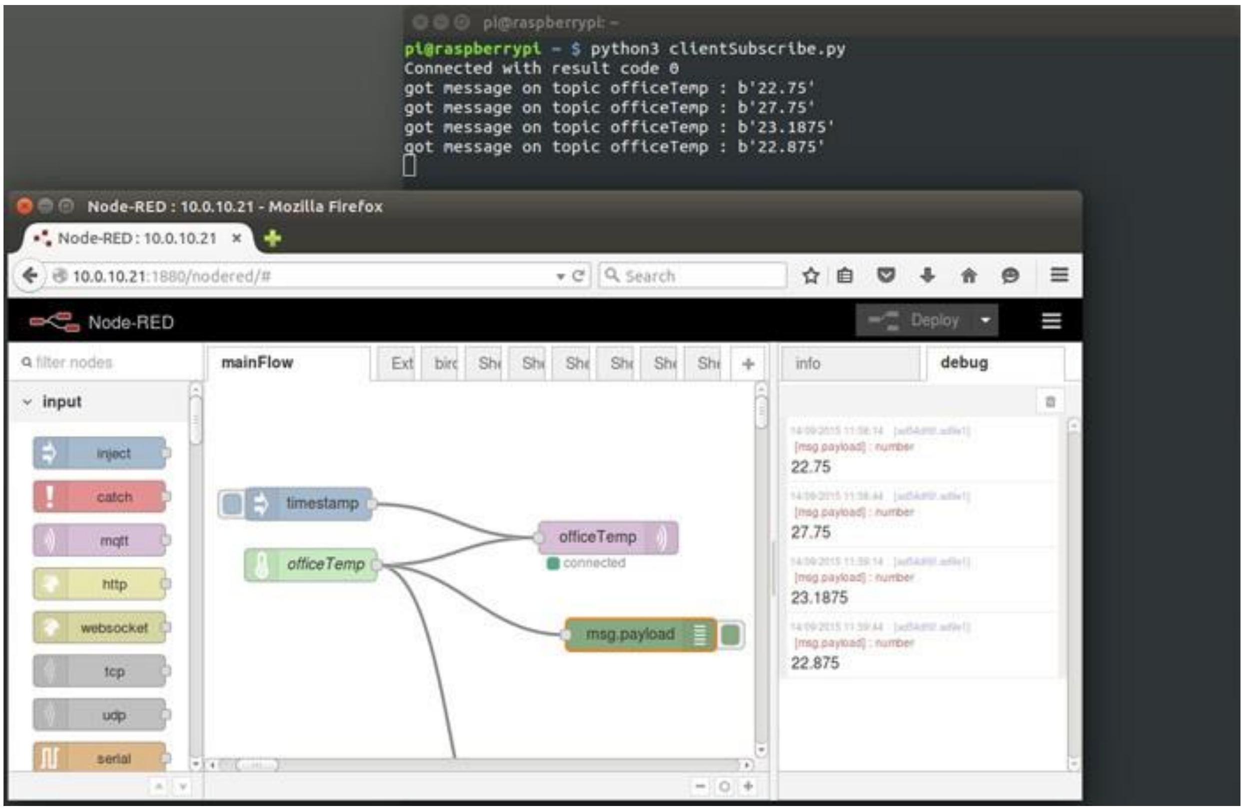
Paste in the code below:

```
import paho.mqtt.client as mqtt

from sense_hat import SenseHat

sense = SenseHat()

# The callback for when the client receives a CONNACK response from the server.

def on_connect(client, userdata, flags, rc):

print("Connected with result code "+str(rc))

# Subscribing in on_connect() means that if we lose the connection and

# reconnect then subscriptions will be renewed.

client.subscribe("officeTemp")

# The callback for when a PUBLISH message is received from the server.

def on_message(client, userdata, msg):

print("got message on topic %s : %s" % (msg.topic, msg.payload))

sense.show_message("MQTT Temp = %.2f" % (float(msg.payload)))

client = mqtt.Client()

client.username_pw_set(username="pi", password="PASS")

client.on_connect = on_connect

client.on_message = on_message

client.connect("192.168.0.21", 1883, 60)

# Blocking call that processes network traffic, dispatches callbacks and

# handles reconnecting.

# Other loop*() functions are available that give a threaded interface and a

# manual interface.
```

```
try:
client.loop_forever()
# deal nicely with ^C
except KeyboardInterrupt:
print("interrupted!")
```

As before, save the file and exit the editor and execute the script:

```
$ python3 clientSubscribe.py
```



```
pi@raspberrypi ~ $ python3 clientSubscribe.py
Connected with result code 0
got message on topic officeTemp : b'22.75'
got message on topic officeTemp : b'27.75'
got message on topic officeTemp : b'23.1875'
got message on topic officeTemp : b'22.875'
```

Keep an eye on the shell window with the Python script running and the window with your MQTT client (in this case Node-RED in a browser). When messages are published to the relevant topic, the Python script will display the content of the messages in the shell window. The Sense HAT will also scroll the message across the LED matrix display.

An alternative to scrolling the message content – in this case temperature readings – would be to map values to different colours and simply update the colour of the LEDs accordingly.

Summary

The Sense HAT integrates plenty of really useful sensors with a versatile LED matrix display at an affordable price. With no soldering required and great software libraries provided, one can be getting readings from sensors and displaying messages within minutes. Used together with Node-RED and MQTT it makes it possible to rapidly prototype IoT projects on the Pi with minimum outlay.

Read this Blog Post on DesignSpark

More about Raspberry Pi in the DesignSpark Raspberry Pi Design Centre