



Basic Weather station

In this worksheet, you are going to explore the other functions of the SenseHAT. It has several sensors on-board which can be used to build a basic weather station.

This weather station tutorial will show you how to setup the sense HAT software itself and how to retrieve the data from its three primary sensors, those being the temperature, humidity and pressure sensors. We will also briefly touch on how to write text to the LED Matrix as we will use this as a way of displaying your sensor data.

1. We can now get writing our python weather station script. We will start with a basic script that continually reads and displays all the data from the Sense HAT's sensors.

To begin writing our new python script, open up the IDLE editor, create a new file and save it as:

```
weather_script.py
```

2. Now enter the following lines of code into the script, we will explain each block of code as we go.

```
from sense_hat import SenseHat
import time
import sys
```

First we need to import all the libraries that we plan to utilize in our script. In our case, we will be using the following libraries.

sense_hat Library

This is the library that we utilize to interact with the Sense Hat itself, without this we wouldn't be able to read any of the sensor data or interact with the LED matrix.

time Library

This allows us to do a large variety of different time stuff, but for our simple script we will be just using its sleep functionality. This allows us to suspend the current thread for a small period.

sys Library

This library provides us access to some variables and functions that are managed by the interpreter itself. In the case of our script we will be using this to terminate the script if we ever need to do so.

```
sense = SenseHat()
sense.clear()
```

The first line creates a link to the Sense HAT library and initializes itself so we can start making calls to it.

The second line tells the library to clear the LED Matrix, by default this means switching off all the LED's. It's always a good idea to do this when dealing with the Sense HAT as it ensures nothing is being displayed already.

```
try:
    while True:
```

We setup our first try statement, we need to do this so we can break out of our while loop by pressing "Ctrl + C" make sure you keep the indentation for the while True. This is because Python is sensitive to indentation. The next few lines of code will need a four-space indentation.

```
temp = sense.get_temperature()
```

Getting the temperature is extremely simple thanks to the Sense HAT Library. All we have to do is make a call to the library for it to retrieve the temperature from the sensor. The output that this will give us will be in Celsius, it also provides a larger decimal number, but we will deal with that on our next line of code.

```
temp = round(temp, 1)
print("Temperature C", temp)
```

This bit of code is extremely basic and just prints the temperature to the terminal.

```
humidity = sense.get_humidity()
humidity = round(humidity, 1)
print("Humidity :", humidity)

pressure = sense.get_pressure()
pressure = round(pressure, 1)
print("Pressure:", pressure)
```

Both the humidity and pressure sensors can be read just like the temperature sensor. Luckily for us the Sense HAT library makes this incredibly simple. Their values also come back with as a large decimal number, so we will again round them then simply display the values to terminal. There isn't much extra to say about these two code blocks as they operate just like the temperature code.

```
time.sleep(1)
```

This is a simple call to the time library that puts the script to sleep for a period of approximately 1 second. This is basically to reduce the rate at which the data is read and outputted.

You can speed up the read rate by decreasing this number or removing the line completely. You can also slow it down further by increasing the number. The number should approximately be the number of seconds you want it to wait between reads.

```
except KeyboardInterrupt:
    pass
```

This code makes the try look for a KeyboardInterrupt exception. When it is triggered we ignore the exception so we can have the script leave the while loop cleanly. We do this by calling pass.

3. With all the code entered into our python file you should end up with something that looks like below.

```
#!/usr/bin/python

from sense_hat import SenseHat
import time
import sys

sense = SenseHat()
sense.clear()

try:
    while True:
        temp = sense.get_temperature()
        temp = round(temp, 1)
        print("Temperature C",temp)

        humidity = sense.get_humidity()
        humidity = round(humidity, 1)
        print("Humidity :",humidity)

        pressure = sense.get_pressure()
        pressure = round(pressure, 1)
        print("Pressure:",pressure)

        time.sleep(1)

except KeyboardInterrupt:
    pass
```

Once your code looks something like the one displayed above and you are certain you have correctly indented your code, you can run your code by saving it and pressing F5.

4. You should now start to see text similar to the following appear in your terminal if everything is working exactly as it should be.

```
('Temperature C', 30.0)
('Humidity :', 39.8)
('Pressure:', 1025.7)
```

Once you're happy with the data that is being displayed you can stop the script by pressing *Ctrl + C*. This will terminate the script from running. Of course, you probably don't want to have to be looking at your Raspberry Pi's terminal to be able to get the current data from it. Instead we are going to show you a method for displaying data: display the data to the LED matrix.

Improving your weather station - Utilizing the LED Matrix

1. Changing our weather python script to start displaying its data to the LED matrix is relatively easy. It will basically involve us concatenating all our prints together into a single string, then issuing the `show_message` command to the Sense HAT library.

2. We can begin making some changes to our weather station script. Above the `time.sleep` line we need to add an extra line of code. This basically handles everything in one simple line.

Above

```
time.sleep(1)
```

Add

```
sense.show_message("Temperature C" + str(temp) + "Humidity:" +
str(humidity) + "Pressure:" + str(pressure), scroll_speed=(0.08),
back_colour= [0,0,200])
```

Make sure you have typed out this line of code all onto a singular line. This will make the temperature, humidity, and pressure scroll across the Sense HAT's LED matrix.

We will explain a bit of what we are doing here. We firstly concatenate the temperature onto the end of "Temperature C", however since the temp variable is a number we need to wrap it in str() to convert it to a string.

Not converting the string will cause issues due to the way Python interprets adding a number to a string. We do this same process for the humidity and pressure.

We also decrease the scroll speed to 0.08, we do that with the following part of the code

```
scroll_speed=(0.08)
```

you can increase or decrease this number to either speed up or decrease the speed the text scrolls.

The last bit of code we have on this line is

```
back_colour= [0,0,200]
```

This part of the code basically sets the background colour of the text to blue.

You can also change the colour of the text itself. We can do this with our example of setting the colour of the text to a pinky colour by adding the following: ,

```
text_color=[200,0,200]
```

after

```
back_color[0,0,200].
```

3. With that line added there is one last thing we will want to add, that being a call to

```
sense.clear().
```

At the bottom of your script after the exception handling and pass, add the following line of code.

```
sense.clear()
```

This code will basically ensure that the LED matrix is completely cleared if we kill the script for any reason. It also stops the annoyance of having to deal with a partially turned on LED matrix and has the added benefit of saving a miniscule amount of power, but there is no point on leaving something switched on when you are not utilizing it.

It is also good coding practice to make sure you clean-up everything when you stop a script from running. It helps stop things like memory leaks from occurring.

4. With those changes made your code should look similar to what is displayed below. Remember that the *sense.show_message* code should be contained within a single line.

```
#!/usr/bin/python

from sense_hat import SenseHat
import time
import sys

sense = SenseHat()
sense.clear()

try:
    while True:
        temp = sense.get_temperature()
        temp = round(temp, 1)
        print("Temperature C",temp)

        humidity = sense.get_humidity()
        humidity = round(humidity, 1)
        print("Humidity :",humidity)

        pressure = sense.get_pressure()
        pressure = round(pressure, 1)
        print("Pressure:",pressure)
```

```
    sense.show_message("Temperature C" + str(temp) + "Humidity:" +
str(humidity) + "Pressure:" + str(pressure), scroll_speed=(0.08),
back_colour= [0,0,200])
    time.sleep(1)
except KeyboardInterrupt:
    pass
sense.clear()
```

Once your code looks something like the one displayed above and you are certain you have correctly indented your code you can save your code and run it using F5.

The text should now begin to scroll across the Sense HAT's LED matrix. Of course, as you will quickly notice, this isn't the easiest way of viewing all the sensors data.

This is mainly due to the small nature of the LED matrix making the text more difficult to read and only being able to display a single letter at one time. Of course, if you have an actual LED screen and not a light matrix, it will be a much better option for displaying your data on your Pi.