Logging SenseHat Data to Thingspeak
The Internet of Things

The Raspberry Pi SenseHat is a sensor packed hardware board that sits on top of a Pi. Hat means Hardware Attached on Top. The Sense Hat was developed to take the Raspberry Pi into space aboard the International Space Station. Two of them are still there running code developed by school children in the UK and Europe.

The SenseHat can measure temperature, pressure, humidity, acceleration in three axis and also has a compass. There are 64 red, green, blue LED on the front with a joystick control too. We will be using some of these in this exercise.

The Internet of Things from WikiPedia is defined as:

"The Internet of Things is the network of physical devices, vehicles, home appliances and other items embedded with electronics, software, sensors, actuators, and connectivity which enables these objects to connect and exchange data"

We will be making the Raspberry Pi with the SenseHat one of these. Logging data to a website called Thingspeak. This data can then be viewed by anyone on the internet, if you wish to, to do things with. It could also be used by you to control other devices such as turning on the air conditioning should it get too warm or warn you if the air pressure drops suddenly that there is a storm coming.

Installing the SenseHat

Only place the SenseHat onto to a Raspberry Pi with the power turned off. The Hat is pressed into place onto the pins on the side of the Pi (or in the case of the Pi-Top screens onto the pins in the middle). Onto securely attached you may power up the Pi.

Reading the Instructions

To help you along these instructions are formatted a certain way. Anything that we know tends not to be read but is important to do is highlighted in bold text. A lot of questions concerning difficulties with the programming are from not reading the instructions properly.

The code that is to be typed into the Python window will be in a different font and blue in colour. **These two lines are not code so please don't ask "Do I type this in?"**

```
# A hash at the front is a comment ignored by Python. It is a human message!
This line shows what code looks like. This is not actual code. So don't type it in!
```

Programming

The programming will be done in the Python IDE found under the main Raspberry menu under programming.
Raspberry > Programming > Python3 IDE
From the Python Shell that opens, open a new file from the file menu.
File > New File
and save this file in Documents as SenseHatThingspeak.py.

Some First Exercises

To get you started this code will read the temperature and write it to the LED on the SenseHat.

```
from sense_hat import SenseHat

sense = SenseHat()
```

This first code sets up the libraries for the SenseHat and sets a variable called sense so that every time sense is used it connects with the SenseHat. Sense could be called something else such as hat or even dog it would still work. As long as that name was used when needed. But sense makes more sense in this case. It makes the variable relevant to what it is we are doing.

The next line sets a variable as temperature and asks the SenseHat for the temperature and making it a string of text rather than a number as it is captured. The SenseHat will not scroll a number only letters. So the number has to be changed into a string of characters.

```
temperature = str(sense.get_temperature())
```

This next line then scrolls the temperature across the LED.

```
sense.show_message("Temperature is " + temperature)
```

Save the file with Ctrl + S or File > Save. Run the file with Run > Run Module or F5. The temperature should scroll across the screen once.

Didn’t Work

Check your spelling. Capital letters are important so sensehat is not the same as SenseHat. And don’t forget the brackets ().

Expanding the Exercise

For this data logging to work we will need it to take the readings every 15 seconds.

Below the first line add this import:

```
from time import sleep
```

This as you can probably tell allows us to use time and put pauses (sleeps) into the code.

Now we need the code to repeat itself again and again until the code is stopped.

Above the "`temperature =`" line add this:

```
while True:
```

and then indent the two remaining lines with the Tab key so that **the final code should look like this**.

```
while True:
    First line of code
    Second line of code
    sleep(15)
```

Then **add the sleep for 15 seconds line at the bottom** as above.

What happens here is that the code indented in belongs to the while true statement. As long as it is true it runs again and again. It says "while True" which means it is always true, so it always runs.

Save and run the code as before. The temperature should scroll across the screen every 15 seconds. To stop it use Ctrl + C.

Didn't Work

Did you remember to indent the code after the while? Is the colon missing after True? Is it true or True?

Rounding the Temperature Down

Currently the temperature is read with far too many decimal places. For the logging two decimal places will be sufficient. **Change** the "`temperature =`" line to the following:

```
temperature = str(round(sense.get_temperature(), 2))
```

The 2 is the number of decimal places. Making it a zero would have none. Save and run again.

Didn't Work

Check the brackets again and for a missing comma.

LED Colours

Now let's add some colour to the readings. There is a simple addition to the scroll message you can add to change the colour. But first we need to explain how the colour is produced.

The LED on the SenseHat are called RGB LED short for red, green, blue. Each LED is actually three LED in one. Each is controlled individually and by changing the amounts the colour changes. The amounts are values between zero (off) and 255 (full brightness).

One way of changing the text colour in the code is to do **change** the line to this:

```
sense.show_message("Temperature is " + temperature, text_colour=[255,0,0])
```

This will produce red text as the red value is at 255 and the others for green and blue are at zero. A value of 150 would make a darker red.

Here are some other colours for you to try.

A better way to change the colours is to set a variable to the colour at the start and then just use that in the code when needed. **Under** the "`sense = Sensehat()`" line add these:

```
red = [150, 0, 0]
blue = [0, 0, 100]     # Those are square brackets not round ones.
```

or any colour you like in the same format. Then **change the other code** to read like this:

```
sense.show_message("Temperature is " + temperature, text_colour=red)
```

Save again and try it out. The background colour can also be changed using `back_colour` separated with a comma.

Didn't Work

Here is the completed code as it stands at the moment. Look out for missing commas and brackets.

```
from sense_hat import SenseHat
from time import sleep

sense = SenseHat()

red = [150, 0, 0]
blue = [0, 0, 255]

while True:
        temperature = str(round(sense.get_temperature(), 2))
        sense.show_message("Temperature is " + temperature, text_colour=red)
        sleep(15)
```

Functions

Once programs start to get longer to write it becomes neater and easier to uses functions to control the workflow. The complexity of the code is written once in a function and then just called when needed. Similar to the import already used.

A simple function we can use for our code is to define (def) the function as follows. The name used is a name like a variable used above for the LED colours. Make it a sensible name that means something to you to read. Python doesn't mind that much what you call it.

Copy the paste from that you have already used, no need to type it all again.

```
def temperature_reading():
        temperature = str(round(sense.get_temperature(), 2))
        sense.show_message("Temperature is " + temperature, text_colour=red)
        sleep(15)
```

And then under the while True we only have to call the function:

```
while True:
        temperature_reading()    # call the function to work and send 15 as a value to pause
```

Try the code. Save and run.

You now all the basics to get the Internet of things up and running except for a few commands that send the data.

Challenge
Now try and get new readings for pressure and humidity as well as temperature. The code is the same just replace temperature with pressure or humidity.

Thingspeak

The website being used here provides free access to users to log data. Go to the website thingspeak.com and take a look. Open the Chromium browser either from the menu or from the icon on the task bar.

In the Channels, Public Channels tab search by tag area look up sensehat. The s0ftwarec0rnwall user should show up. Click on the link and take a look. You will see some logged data already. This is where your data will appear shortly.

The code for just sending the temperature is below.
Add in the extra pieces as necessary to the code you already have. They're marked in bold.

```
from sense_hat import SenseHat
import requests
from time import sleep

sense = SenseHat()

red = [150,0,0]
green = [0,150,0]
blue = [0,0,150]

# Thingspeak address to send data to.
thingspeak_address = "https://api.thingspeak.com/update?api_key="

# The access key to Software Cornwall account
api_key = "7WFRBE3A9W6GRHYG"   # Series 1 readings
#   or "XKAZEDWIDGFYEQ4D"      # Series 2 readings
#   or "D27851HSL3NTTMRD"      # Series 3 readings

def send_temperature():
    temperature = str(round(sense.get_temperature(), 2))
    sense.show_message("Temperature is " + temperature, text_colour=red)
    response=requests.post(thingspeak_address+api_key + "&field1=" + temperature)
    print(temperature, response)
    sleep(15)

while True:
    send_temperature()
```

There are a couple of extra lines of code. Two are commented about the Thingspeak address and access code. These are dropped into the line starting "response =". The field number refers to the graphs on the webpage. A maximum of 8 are allowed per page numbered 1 through to 8.

The print lines will print the various values to the Shell to see that something is happening as well as a response value from Thingspeak. It should be 200 if all is well.

Test this code and check all works as it should.

If all is working change the code to send different data using these **extra lines** to log the pressure and humidity to the same field number you used.
Remember to copy and paste, it is quicker.

```
def send_pressure():
    pressure = str(round(sense.get_pressure(), 2))
    sense.show_message("Pressure is " + pressure, text_colour=green)
    response= requests.post(thingspeak_address + api_key + "&field1=" + pressure)
    print(pressure, response)
    sleep(15)

def send_humidity():
    humidity = str(round(sense.get_humidity(), 2))
    sense.show_message("Humidity is " + humidity, text_colour=blue)
    response= requests.post(thingspeak_address + api_key + "&field1=" + humidity)
    print(humidity, response)
    sleep(15)

while True:
    send_pressure()
    send_humidity()
```