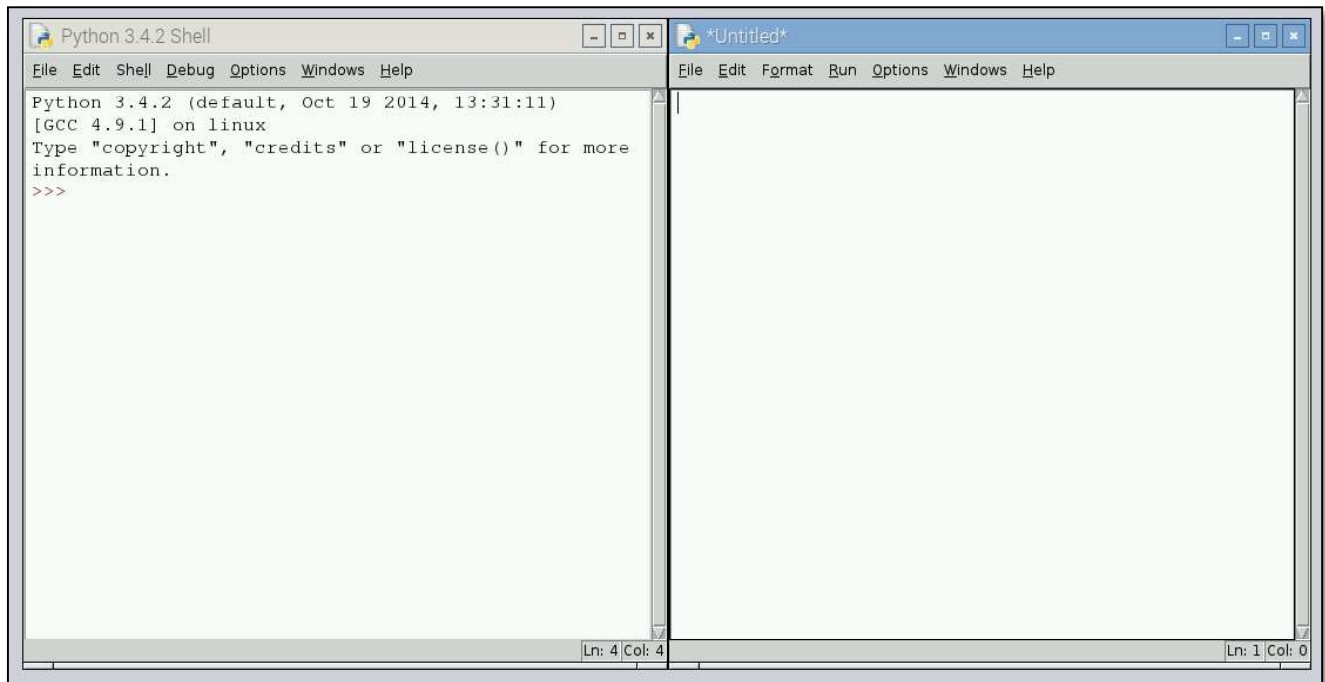


Approach 2: Storing readings in Device itself as csv file

In this approach, python script is used which takes the readings from device and stores it in comma separated values file. To begin your script you need to boot your Raspberry Pi into desktop mode and run Idle for Python 3 from the programming section of the menu. Once Idle has loaded you will need to select **File** and then **New File** which will load a separate window in which you can write your code.



In your right hand window, add the following lines of python code. The lines starting with a `#` symbol are **comments** and are ignored by the computer. We have used comments here to break the code into four sections, which will make it easier to build your program as it gets more complex.

```
File Edit Format Run Options Window Help
##### Libraries #####
from sense_hat import SenseHat
from datetime import datetime

##### Logging Settings #####

##### Functions #####

##### Main Program #####
```

- The first section, **Libraries**, is where you will import code that will give your program extra abilities. The line `from sense_hat import SenseHat` allows your program to use the Sense-HAT hardware. The line `from datetime import datetime` allows your program to use the time module.
- The section headed **Logging Settings** is where you will be able to control different features of your logger program.
- The third section, **Functions**, will contain short "chunks" of reusable code which do a specific job, such as writing the current data to a file.
- The final section, **Main Program**, is the part of your code which uses each of the functions in the right sequence to run the whole program.

In order to get data from the Sense HAT you will need to write a function called **get_sense_data** which will check each sensor in turn and store the sensor data in a list. The function should be added to the **Functions** section.

```
def get_sense_data():
    sense_data=[]

    sense_data.append(sense.get_temperature_from_humidity())
    sense_data.append(sense.get_temperature_from_pressure())
    sense_data.append(sense.get_humidity())
    sense_data.append(sense.get_pressure())
```

The first line defines your function name, and the second sets up an empty **list** structure into which you will add your collected data.

The next four lines get data from some of the sensors and adds (or appends) them to the `sense_data` list.

Next we'll need to add some lines to your **Main Program** Section, this will need to do two things:

- create a sense object, which represents the Sense HAT
- repeatedly **get_sense_data** from the sensors and display it

Add the following code to the **Main Program** section:

```
sense = SenseHat()

while True:

    sense_data = get_sense_data()

    print(sense_data)
```

3. Another function you will need is the `file_setup` function which will create a list of headings that will be written to the file before any data. The function is shown below and needs to be added to your Functions section.

```
def file_setup(filename):

    header = ["temp_h", "temp_p", "humidity", "pressure",

              "pitch", "roll", "yaw",

              "mag_x", "mag_y", "mag_z",

              "accel_x", "accel_y", "accel_z",

              "gyro_x", "gyro_y", "gyro_z",

              "timestamp"]

    with open(filename, "w") as f:

        f.write(",".join(str(value) for value in header)+ "\n")
```

This function is slightly different to the previous as it needs an input (or parameter) in order to work; in this case the input has been called filename. When the main program calls this function it must also give the function the name of the file to write to. If it were called like this `file_setup("output.csv")` the function would create output.csv

The function itself creates a list of header names called header. It then opens a file in write mode (which will overwrite any previous data) and refers to that file as f. whilst the file is open it joins all the list headings together using commas and writes that line to the file.

4. The two functions and the settings you added now need to be used in the main program.

Straight after the lines that read:

```
##### Main Program #####
```

```
sense = SenseHat()
```

add the following:

```
batch_data= []
```

```
if FILENAME == "":
```

```
    filename = "SenseLog-"+str(datetime.now())+".csv"
```

```
else:
```

```
    filename = FILENAME+"-"+str(datetime.now())+".csv"
```

```
file_setup(filename)
```

The first line here creates an empty list that the program will keep adding sense_data lines to until it reaches 50 (or whatever value is set by WRITE_FREQUENCY). The if/else block checks whether a FILENAME has been set, if it hasn't then the default of "SenseLog" is used. The current date and time is also added to the filename. Finally the file_setup functions is called and given the filename that was decided upon in the previous if / else block.

5. The last step is to change some of the logic inside the while True: loop.

You need to make it collect sense_data

Then use the `log_data` function to convert the `sense_data` into csv form and add the the current `batch_data`. Once the data is logged, the program checks whether the size of `batch_data` exceeds the `WRITE_FREQUENCY` setting, if so the data is written to a file and `batch_data` is reset.

Your while `True`: loop should be updated to look like this:

```
while True:
```

```
    sense_data = get_sense_data()
```

```
    log_data()
```

```
    if len(batch_data) >= WRITE_FREQUENCY:
```

```
        print("Writing to file..")
```

```
        with open(filename,"a") as f:
```

```
            for line in batch_data:
```

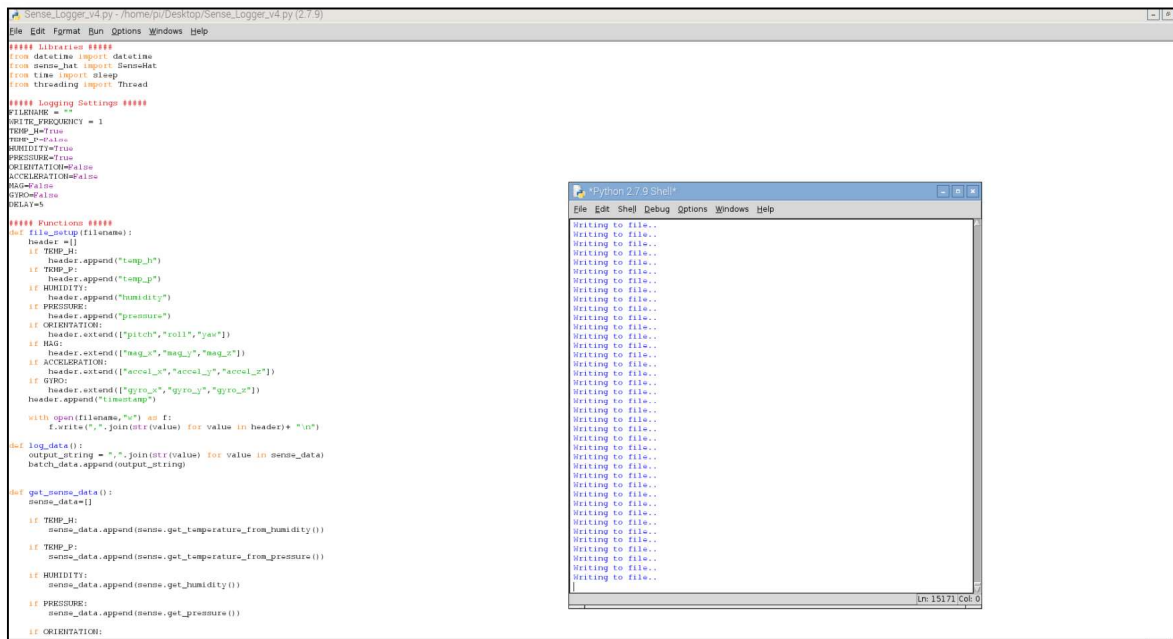
```
                f.write(line + "\n")
```

```
            batch_data = []
```

The line `print("Writing to file..")` is optional, but it will show whenever data is being written. The line `with open(filename,"a") as f:` opens the file in append mode which adds the data at the end point of the file rather than overwriting. You can check your code against a full code listing [here](#). When you running the program you should simply see the messages saying `Writing to file..` every so often.

Writing to a file

You can stop logging by pressing `Ctrl+C`



Code:

```
##### Libraries #####
```

```
from datetime import datetime
```

```
from sense_hat import SenseHat
```

```
from time import sleep
```

```
from threading import Thread
```

```
##### Logging Settings #####
```

```
FILENAME = ""
```

```
WRITE_FREQUENCY = 1
```

```
TEMP_H=True
```

```
TEMP_P=False
```

```
HUMIDITY=True
```

```
PRESSURE=True
```

```
ORIENTATION=False
```

```
ACCELERATION=False
```

MAG=False

GYRO=False

DELAY=5

Functions

def file_setup(filename):

 header =[]

 if TEMP_H:

 header.append("temp_h")

 if TEMP_P:

 header.append("temp_p")

 if HUMIDITY:

 header.append("humidity")

 if PRESSURE:

 header.append("pressure")

 if ORIENTATION:

 header.extend(["pitch","roll","yaw"])

 if MAG:

 header.extend(["mag_x","mag_y","mag_z"])

 if ACCELERATION:

 header.extend(["accel_x","accel_y","accel_z"])

 if GYRO:

 header.extend(["gyro_x","gyro_y","gyro_z"])

 header.append("timestamp")

```
with open(filename,"w") as f:
```

```
    f.write(",".join(str(value) for value in header)+ "\n")
```

```
def log_data():
```

```
    output_string = ",".join(str(value) for value in sense_data)
```

```
    batch_data.append(output_string)
```

```
def get_sense_data():
```

```
    sense_data=[]
```

```
    if TEMP_H:
```

```
        sense_data.append(sense.get_temperature_from_humidity())
```

```
    if TEMP_P:
```

```
        sense_data.append(sense.get_temperature_from_pressure())
```

```
    if HUMIDITY:
```

```
        sense_data.append(sense.get_humidity())
```

```
    if PRESSURE:
```

```
        sense_data.append(sense.get_pressure())
```

```
    if ORIENTATION:
```

```
        o = sense.get_orientation()
```

```
        yaw = o["yaw"]
```

```
        pitch = o["pitch"]
```

```
        roll = o["roll"]
```

```
        sense_data.extend([pitch,roll,yaw])
```


if MAG:

```
mag = sense.get_compass_raw()

mag_x = mag["x"]

mag_y = mag["y"]

mag_z = mag["z"]

sense_data.extend([mag_x,mag_y,mag_z])
```

if ACCELERATION:

```
acc = sense.get_accelerometer_raw()

x = acc["x"]

y = acc["y"]

z = acc["z"]

sense_data.extend([x,y,z])
```

if GYRO:

```
gyro = sense.get_gyroscope_raw()

gyro_x = ["x"]

gyro_y = ["y"]

gyro_z = ["z"]

sense_data.extend([gyro_x,gyro_y,gyro_z])

sense_data.append(datetime.now())

return sense_data
```

def timed_log():

while True:

```
log_data()
```

```
sleep(DELAY)
```

```
##### Main Program #####
```

```
sense = SenseHat()
```

```
batch_data= []
```

```
if FILENAME == "":
```

```
    filename = "SenseLog-"+str(datetime.now())+".csv"
```

```
else:
```

```
    filename = FILENAME+"-"+str(datetime.now())+".csv"
```

```
file_setup(filename)
```

```
if DELAY > 0:
```

```
    sense_data = get_sense_data()
```

```
    Thread(target= timed_log).start()
```

```
while True:
```

```
    sense_data = get_sense_data()
```

```
    if DELAY == 0:
```

```
        log_data()
```

```
    if len(batch_data) >= WRITE_FREQUENCY:
```

```
        print("Writing to file..")
```

```
        with open(filename,"a") as f:
```

```
            for line in batch_data:
```

```
                f.write(line + "\n")
```

```
            batch_data = []
```

OUTPUT CSV FORMAT:

Sensor-log-2016-10-17 19:16:44 379585.csv				
File Edit Search Options Help				
temp	h_humidity	pressure	timestamp	
38.5250015259	35.0395736694	950.701171875	2016-10-17 19:16:44.389857	
38.2704544007	34.1230479614	950.602617100	2016-10-17 19:16:49.421340	
38.4704551697	33.349697113	950.684814453	2016-10-17 19:16:54.445490	
38.5613632202	33.280418396	950.713134766	2016-10-17 19:17:00.276915	
38.3917279663	34.8799287742	950.69409625	2016-10-17 19:17:05.318029	
38.361366272	35.1299438477	950.679199219	2016-10-17 19:17:10.326827	
38.3068161011	32.949069768	950.665039062	2016-10-17 19:17:15.362825	
38.3977279663	34.8046188354	950.672851562	2016-10-17 19:17:22.337735	
38.5866206787	34.53652541	950.708466604	2016-10-17 19:17:27.746507	
38.5250015259	34.5817108154	950.693847656	2016-10-17 19:17:32.924526	
38.3977279663	34.4280853271	950.715332031	2016-10-17 19:17:38.106662	
38.4159088135	32.9099082947	950.707275391	2016-10-17 19:17:43.115218	
38.4885300108	35.6287649426	950.706654088	2016-10-17 19:17:48.170883	
38.4522705078	35.0034980774	950.720458984	2016-10-17 19:17:53.186773	
38.5868206787	35.2654953083	950.724609375	2016-10-17 19:17:58.318945	
38.4885300108	34.6244445881	950.717041816	2016-10-17 19:18:03.354986	
38.4704551697	34.5124282837	950.705078125	2016-10-17 19:18:08.646975	
38.5068206787	34.3845845030	950.705322266	2016-10-17 19:18:21.311706	
38.4159088135	33.5726851331	950.703857422	2016-10-17 19:18:27.343484	
38.4340896686	35.0907821655	950.750488281	2016-10-17 19:18:33.399980	
38.3977279663	33.7985220492	950.703613281	2016-10-17 19:18:38.407800	
38.4522705078	34.4762840271	950.749511719	2016-10-17 19:18:43.477217	
38.4885300108	33.0424449512	950.727783203	2016-10-17 19:18:48.532577	
38.4885300108	33.1826910504	950.754150391	2016-10-17 19:18:53.544898	
38.3431816101	34.6811141968	950.765869141	2016-10-17 19:18:59.328018	
38.4522705078	36.4523239136	950.726806641	2016-10-17 19:19:04.332704	
38.3795471191	34.7252992859	950.742675781	2016-10-17 19:19:09.345377	
38.3431816101	35.4823760986	950.751953125	2016-10-17 19:19:14.342471	
38.3250807629	34.5003814697	950.744628906	2016-10-17 19:19:19.418284	
38.2704544007	35.0834255981	950.754882812	2016-10-17 19:19:24.452617	
38.2159118652	35.2986237687	950.723388672	2016-10-17 19:19:29.468013	
38.3250807629	35.1050425903	950.756032930	2016-10-17 19:19:34.450357	
38.1795463562	35.2112731934	950.745685469	2016-10-17 19:19:39.528118	
38.2886352539	34.2925338745	950.778080594	2016-10-17 19:19:44.538475	
38.3068161011	35.1568881042	950.762451172	2016-10-17 19:19:49.541630	
38.361366272	35.9281921387	950.758732422	2016-10-17 19:19:54.578173	
38.3795471191	35.1028327942	950.743896484	2016-10-17 19:19:59.719187	
38.3250807629	35.6631126404	950.731281172	2016-10-17 19:20:04.791822	
38.2340927124	33.5063362122	950.738769531	2016-10-17 19:20:09.828887	
38.3795471191	35.0854248417	950.738097190	2016-10-17 19:20:15.900645	
38.1977272034	33.3255996784	950.732421875	2016-10-17 19:20:21.146385	
38.2522735596	34.884669804	950.759521484	2016-10-17 19:20:26.154673	
38.2159118652	34.2503623962	950.731933594	2016-10-17 19:20:31.307479	
38.2704544007	34.0033569336	950.721191406	2016-10-17 19:20:36.387361	
38.2886352539	34.9040222168	950.715376172	2016-10-17 19:20:41.624146	
38.3068161011	35.2383842468	950.733398438	2016-10-17 19:20:46.663514	
38.2340927124	34.9823417664	950.762207831	2016-10-17 19:20:51.679099	
38.2522735596	37.6210784912	950.723388672	2016-10-17 19:20:57.084212	
38.4159088135	35.6119041443	950.764092570	2016-10-17 19:21:02.088428	
38.2522735596	34.6871414185	950.766742188	2016-10-17 19:21:07.099152	
38.2522735596	35.4014313166	950.760366628	2016-10-17 19:21:12.100200	