READING DATA FROM SENSORS

Edmond Lascaris and Anne Jessup - Creative Commons – 11 May 2021

OVERVIEW

In this lesson we are going to write code that will allow us to download data from an atmospheric sensor. The sensor will give us data on temperature, atmospheric pressure, battery voltage and signal strength. We will use the python language to get this data. Once we have data we can do other amazing things such as save the data and even graph the data. Environmental data is very important because it allows us to monitor the environment we live in more closely.

LEARNING OBJECTIVES

- Learn how to use the requests library in python to get data from a URL
- Learn how to interpret data presented in the JSON format
- Learn how to process data in python so that it can be saved in variables

REQUESTING DATA FROM A URL IN PYTHON

Sensors are now commonly connected to the internet. This has given rise to the term – The Internet of Things. If a sensor is connected to the internet then it will often have a URL (Universal Resource Locator) to allow us to interact with the sensor. One common interaction is to be able to see or download data collected by the sensor. To get data from a sensor we need to import the requests library in python. In this lesson we will download some data from an atmospheric sensor. The sensor data includes temperature, air pressure, sensor battery voltage and signal strength.

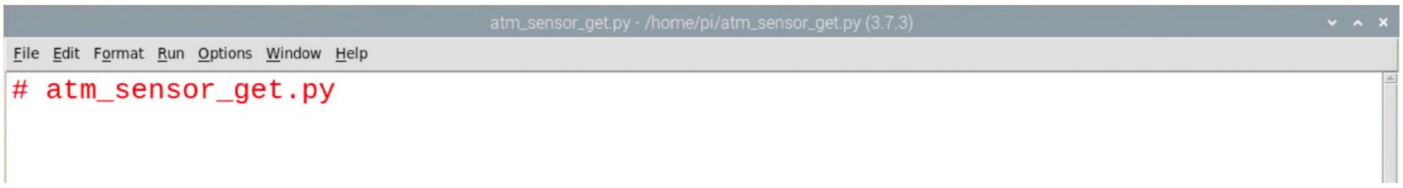
REQUESTING NEW DATA FROM A SENSOR URL

In this example we write a python3 program to download sensor data using the requests library. The data will be presented in the python Shell.

1. **Create a new file in python named atm_sensor_get.py**

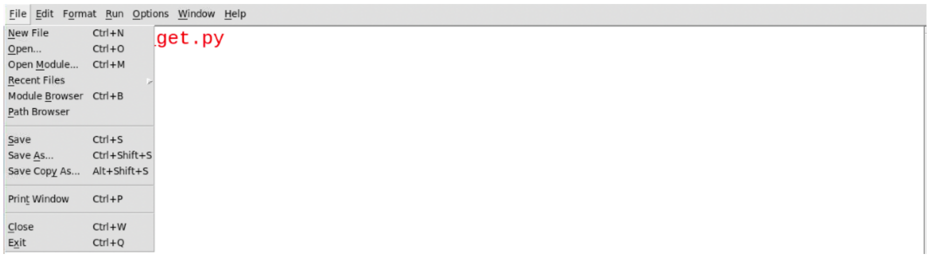
- From the Raspberry Pi main menu drop down select Programming > Python3 (IDLE).
- This will open up the Python Shell.
- From the File drop down menu select New File. This will open up a new file Editor Window.
- We are going to name the file **atm_sensor_get.py**

- As a placeholder we can add this file name within our new file. We will however add a # (hash) in front of the name so that we **comment it out**. When we **comment out** text in a file the computer will ignore it. The line will only be useful for humans. To signal that the text has been commented out, the text colour will change to red.


atm_sensor_get.py - /home/pi/atm_sensor_get.py (3.7.3)
File Edit Format Run Options Window Help

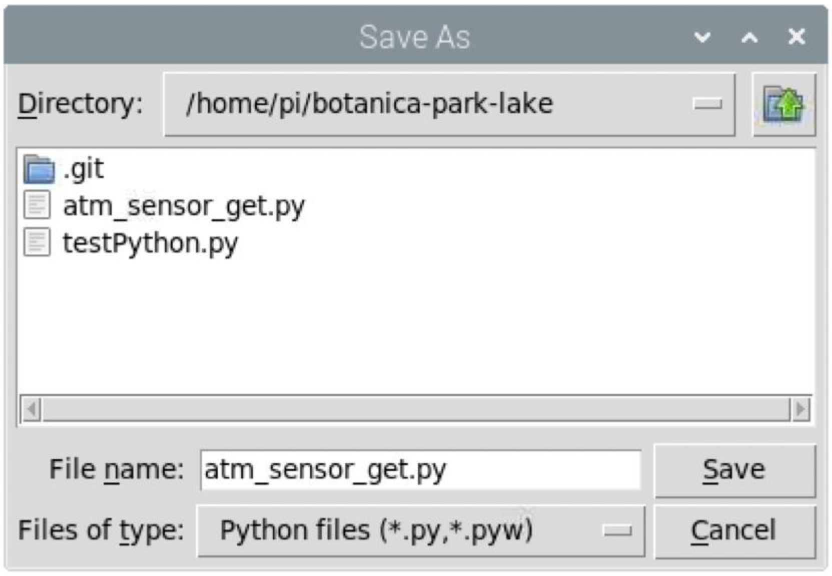
```
# atm_sensor_get.py
```

- To save the new file, select File > Save As



- Navigate to your working directory and then save the file as **atm_sensor_get.py**

- In Linux we never use names that include spaces. Replace spaces with either the _ (underline) symbol or the **–** (hyphen) symbol.



Directory: /home/pi/botanica-park-lake

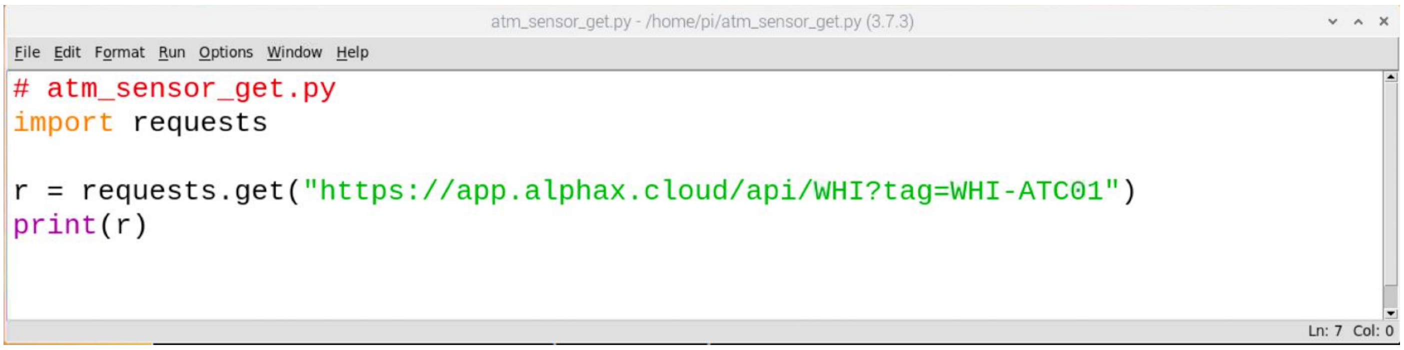
.git
atm_sensor_get.py
testPython.py

File name: atm_sensor_get.py
Files of type: Python files (*.py,*.pyw)

2. Request data from sensor using python

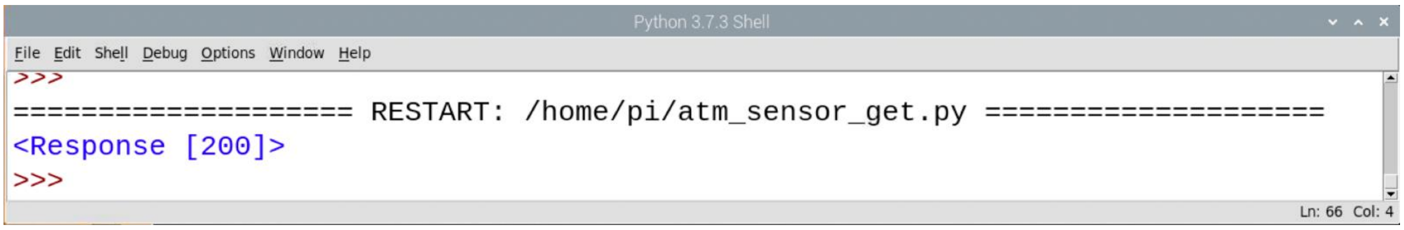
- Enter the following lines of request to request data from the atmospheric sensor.

- The key items in the code are described:
    - **import requests** – this statement imports a python library so that the python program can get data from the internet.
    - **r = requests.get(URL)** – this statement gets the data from a specified URL and stores the result in the variable object named **r**. **URL** is short for Uniform Resource Location, also known as a web address.
    - **print(r)** – this statement prints a code that determines if the request was successful or not.
- Save and the **Run** the program.


atm_sensor_get.py - /home/pi/atm_sensor_get.py (3.7.3)
File Edit Format Run Options Window Help
Ln: 7 Col: 0

```
# atm_sensor_get.py
import requests

r = requests.get("https://app.alphax.cloud/api/WHI?tag=WHI-ATC01")
print(r)
```

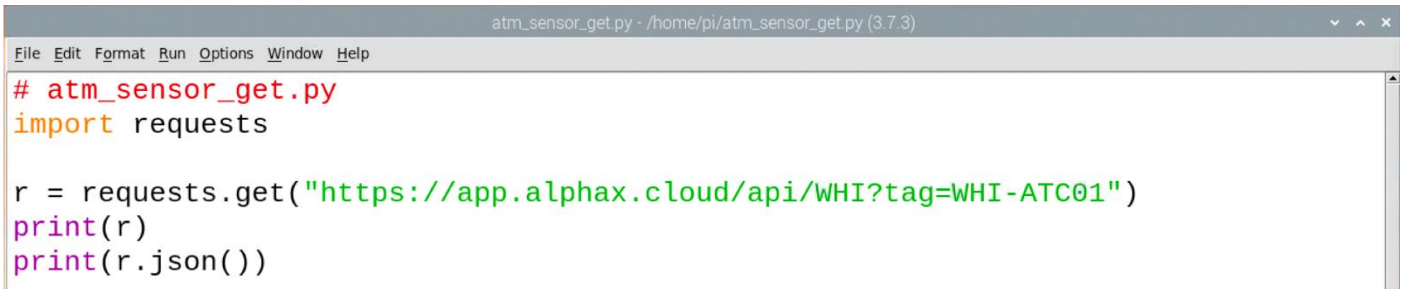
- The response in the Python Shell should be [**200**]. A value of [200] means that the request was successful.
- Other numbers or errors will indicate that there is a problem either with the Python code or with communication with the sensor.



```
>>>
==================== RESTART: /home/pi/atm_sensor_get.py ====================
<Response [200]>
>>>
```

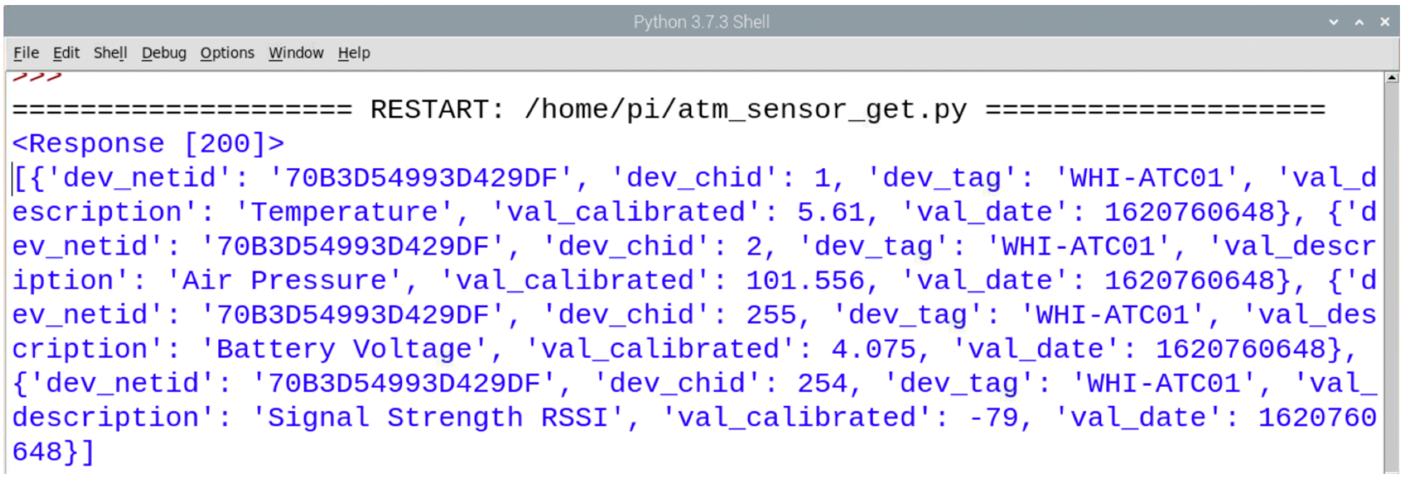
3. View sensor data in JSON format

- We can now add some more code so that we can look at the actual data.
- Add the statement **print(r.json())** and run the code.
- The method **json()** means that the data will be presented in **JSON** format.
- JSON stands for JavaScript Object Notation. It is an open standard to help format data for data exchange between computers and it uses human-readable text. The data is stored in **attribute-value pairs** similar to a **dictionary.**


atm_sensor_get.py - /home/pi/atm_sensor_get.py (3.7.3)
File Edit Format Run Options Window Help

```python
# atm_sensor_get.py
import requests

r = requests.get("https://app.alphax.cloud/api/WHI?tag=WHI-ATC01")
print(r)
print(r.json())
```

- You can see the data output in the python Shell Window.



```
>>>
==================== RESTART: /home/pi/atm_sensor_get.py ====================
<Response [200]>
[{'dev_netid': '70B3D54993D429DF', 'dev_chid': 1, 'dev_tag': 'WHI-ATC01', 'val_d
escription': 'Temperature', 'val_calibrated': 5.61, 'val_date': 1620760648}, {'d
ev_netid': '70B3D54993D429DF', 'dev_chid': 2, 'dev_tag': 'WHI-ATC01', 'val_descr
iption': 'Air Pressure', 'val_calibrated': 101.556, 'val_date': 1620760648}, {'d
ev_netid': '70B3D54993D429DF', 'dev_chid': 255, 'dev_tag': 'WHI-ATC01', 'val_des
cription': 'Battery Voltage', 'val_calibrated': 4.075, 'val_date': 1620760648},
{'dev_netid': '70B3D54993D429DF', 'dev_chid': 254, 'dev_tag': 'WHI-ATC01', 'val_
description': 'Signal Strength RSSI', 'val_calibrated': -79, 'val_date': 1620760
648}]
```

- Unfortunately, presenting the data this way is not very readable.

- In the next section of this lesson we will learn how to view and interpret data stored in the JSON format.

SMART CITIES

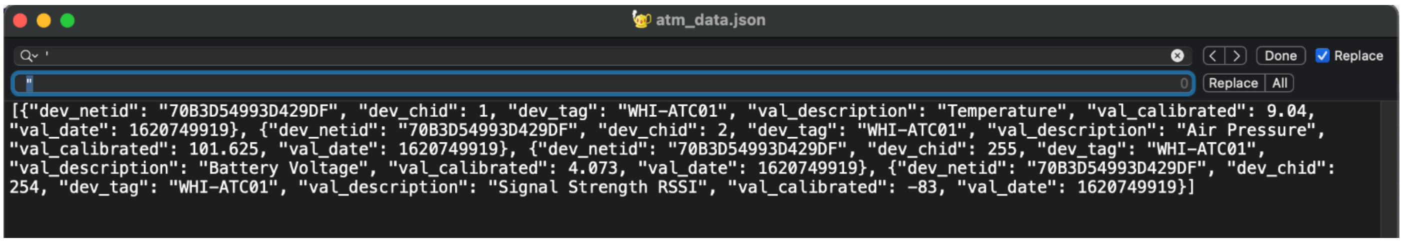
INTERPRETING DATA IN JSON FORMAT

As previously mentioned, **JSON** is an open standard that allows data to be exchanged more easily between computers. JSON data is stored as text, and this makes it human-readable however you need a special application to see this data more clearly. In this lesson we will use the web browser **Firefox** to visualise the data so we can see the arrangement of **attribute-value pairs.**

4. Copying, editing and saving JSON formatted data

One way to see the data more easily is to view it in the web browser Firefox. To do this we need to **Copy** the JSON data out-put from the Raspberry Pi and **Paste** it in a file on your PC. We also need to replace all **‘** characters with **“** otherwise Firefox wont show the JSON data correctly.

- Copy the data above from the Python Shell. To copy, **right-mouse button click** and select **Copy**.
- Paste it into a **text editor** on your PC. To paste, **right-mouse button click** and select **Paste**.
- Replace all **‘** characters with **“** using the in-build **Find/Replace** function in your **Text Editor.**
- Then **Save** the data as a file with the file extension **.json** on your PC. In this example the file name is **atm_data.json**



```
[{"dev_netid": "70B3D54993D429DF", "dev_chid": 1, "dev_tag": "WHI-ATC01", "val_description": "Temperature", "val_calibrated": 9.04,
"val_date": 1620749919}, {"dev_netid": "70B3D54993D429DF", "dev_chid": 2, "dev_tag": "WHI-ATC01", "val_description": "Air Pressure",
"val_calibrated": 101.625, "val_date": 1620749919}, {"dev_netid": "70B3D54993D429DF", "dev_chid": 255, "dev_tag": "WHI-ATC01",
"val_description": "Battery Voltage", "val_calibrated": 4.073, "val_date": 1620749919}, {"dev_netid": "70B3D54993D429DF", "dev_chid":
254, "dev_tag": "WHI-ATC01", "val_description": "Signal Strength RSSI", "val_calibrated": -83, "val_date": 1620749919}]
```

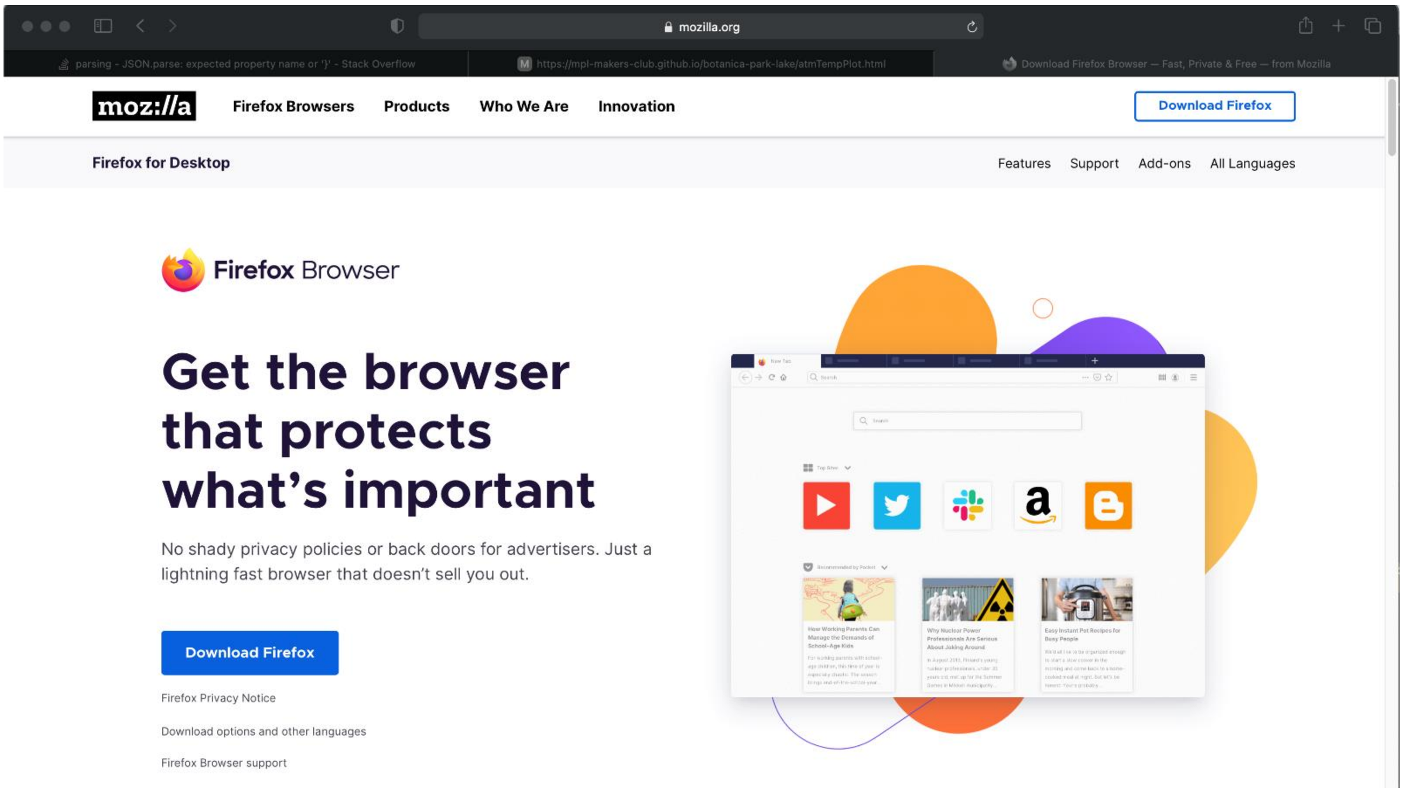
5. Download and install Firefox web browser

- On your PC, download and install the Firefox Browser.
- You can use the following link: https://www.mozilla.org/en-US/firefox/new/

City of Whittlesea
www.whittlesea.vic.gov.au

BANYULE NILLUMBIK TECH SCHOOL

WHITTLESEA TECH SCHOOL


mozilla.org
parsing - JSON.parse: expected property name or '}' - Stack Overflow
https://mpl-makers-club.github.io/botanica-park-lake/atmTempPlot.html
Download Firefox Browser — Fast, Private & Free — from Mozilla
moz://a
Firefox Browsers
Products
Who We Are
Innovation
Download Firefox
Firefox for Desktop
Features
Support
Add-ons
All Languages
Firefox Browser

Get the browser that protects what's important

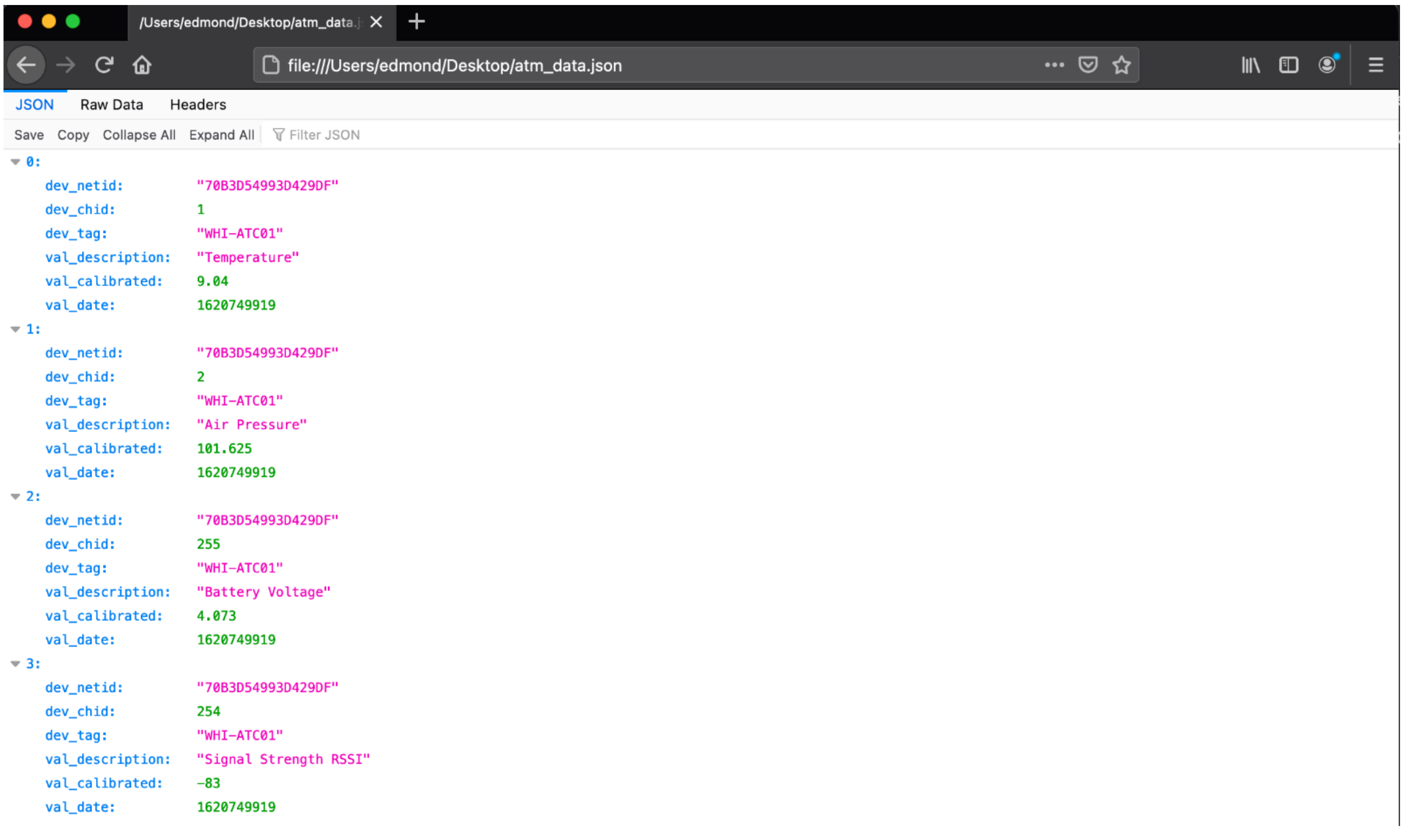
No shady privacy policies or back doors for advertisers. Just a lightning fast browser that doesn't sell you out.

Download Firefox
Firefox Privacy Notice
Download options and other languages
Firefox Browser support

6. Visualising JSON formatted data in Firefox

- In the Firefox Browser **Open** the new file you created named **atm_data.json**
- You can do this by selecting **File > Open File**... from the top Drop Down Menu.
- You should see something like the following displayed. See image below.
- Some key points:
  - Sensor data is divided into four sections labelled **[0, 1, 2, 3]**
  - Each section is for different data set:
    - **Temperature**
    - **Air Pressure**
    - **Battery Voltage**
    - **Signal Strength (RSSI)**
  - The attribute-value pair we are interested in (for each data set) is the attribute named "**val_calibrated**".

- For each data set [**0, 1, 2, 3**] the “**val-calibrated” attribute** will match with either **[temperature, pressure, battery voltage, signal strength]**
- For example, for data set **[0],** the “val_calibrated” value is **9.04** – which is the **Temperature**.

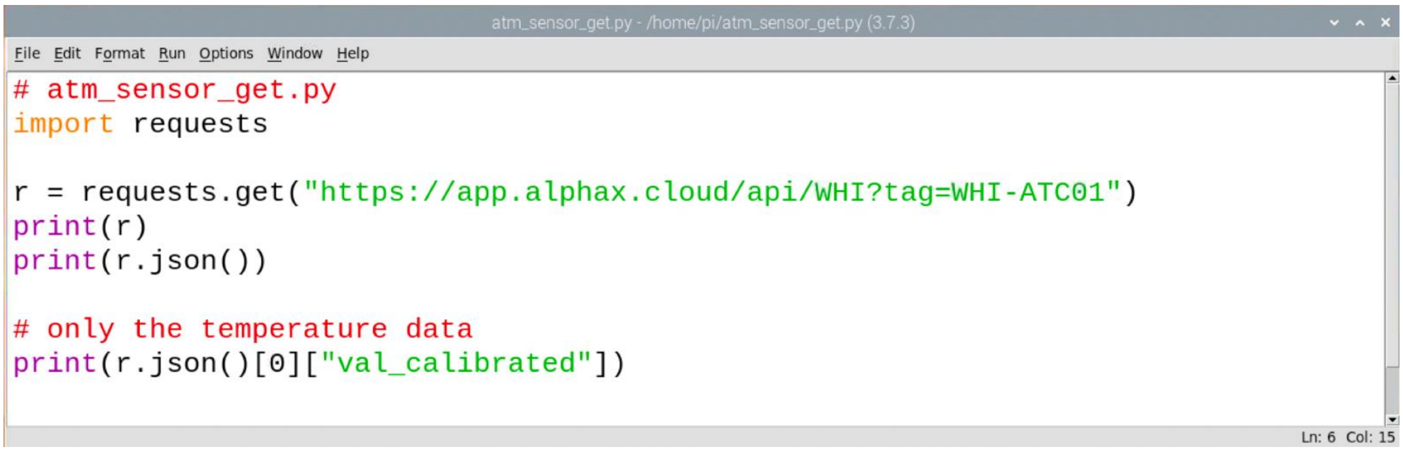


/Users/edmond/Desktop/atm_data.j
file:///Users/edmond/Desktop/atm_data.json
JSON
Raw Data
Headers
Save
Copy
Collapse All
Expand All
Filter JSON

```
▼ 0:
    dev_netid:          "70B3D54993D429DF"
    dev_chid:           1
    dev_tag:            "WHI-ATC01"
    val_description:    "Temperature"
    val_calibrated:     9.04
    val_date:           1620749919
▼ 1:
    dev_netid:          "70B3D54993D429DF"
    dev_chid:           2
    dev_tag:            "WHI-ATC01"
    val_description:    "Air Pressure"
    val_calibrated:     101.625
    val_date:           1620749919
▼ 2:
    dev_netid:          "70B3D54993D429DF"
    dev_chid:           255
    dev_tag:            "WHI-ATC01"
    val_description:    "Battery Voltage"
    val_calibrated:     4.073
    val_date:           1620749919
▼ 3:
    dev_netid:          "70B3D54993D429DF"
    dev_chid:           254
    dev_tag:            "WHI-ATC01"
    val_description:    "Signal Strength RSSI"
    val_calibrated:     -83
    val_date:           1620749919
```

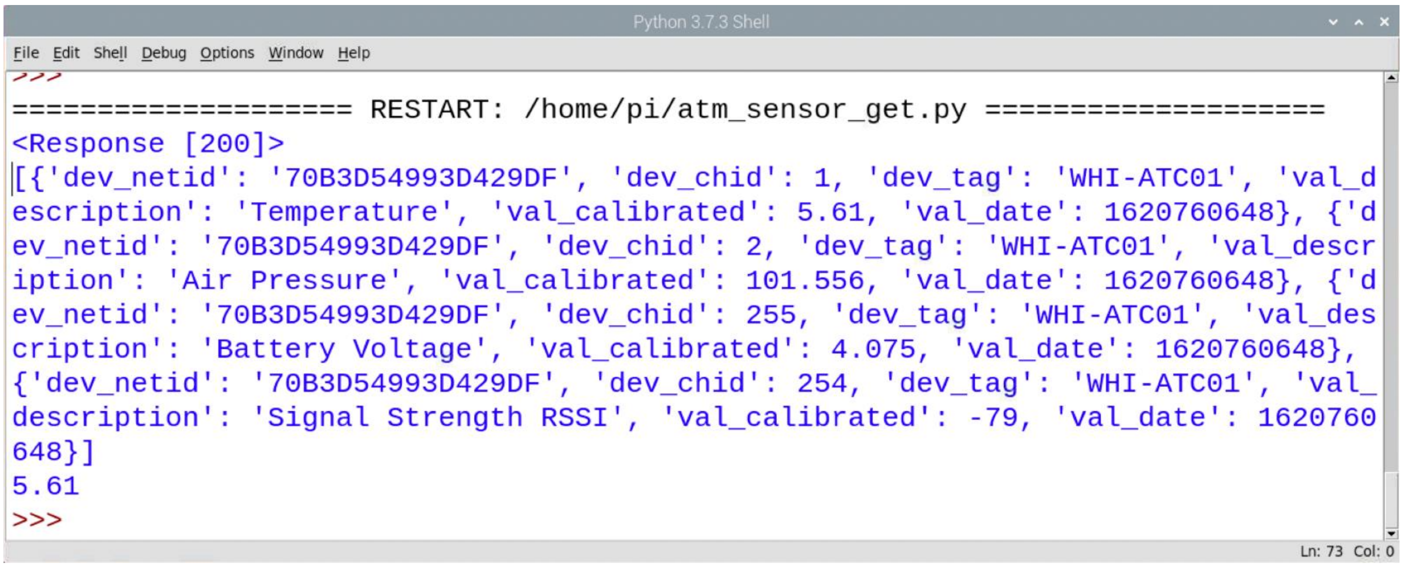
7. Learning how to extract Temperature data from JSON formatted data

- To extract the relevant data from our JSON data we enter the following code **print(r.json[0][“val_calibrated”])**
  - **[0]** – represents the first data set (containing Temperature data)
  - **[“val_calibrated”]** – is the attribute in the attribute-value pair that pairs with the temperature value
  - **print()** – will print out the temperature data to the Python Shell


atm_sensor_get.py - /home/pi/atm_sensor_get.py (3.7.3)
File Edit Format Run Options Window Help
Ln: 6 Col: 15

```
# atm_sensor_get.py
import requests

r = requests.get("https://app.alphax.cloud/api/WHI?tag=WHI-ATC01")
print(r)
print(r.json())

# only the temperature data
print(r.json()[0]["val_calibrated"])
```

- The result can be seen on the last line of the Python Shell.
- In this case the temperature is **5.61** (degrees Centigrade).

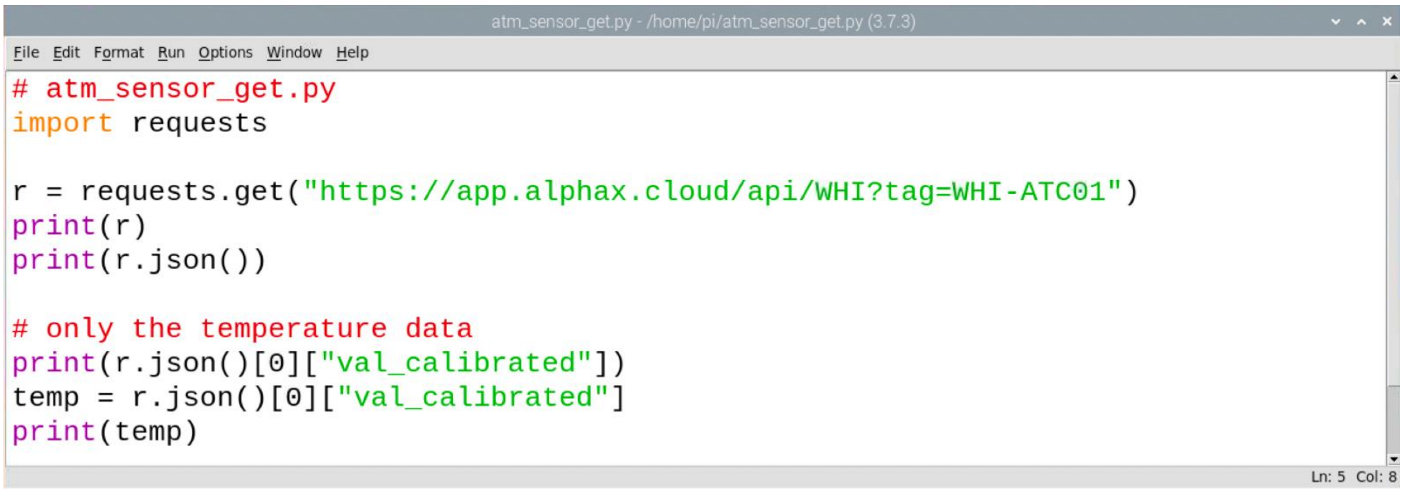


```
>>>
==================== RESTART: /home/pi/atm_sensor_get.py ====================
<Response [200]>
[{'dev_netid': '70B3D54993D429DF', 'dev_chid': 1, 'dev_tag': 'WHI-ATC01', 'val_d
escription': 'Temperature', 'val_calibrated': 5.61, 'val_date': 1620760648}, {'d
ev_netid': '70B3D54993D429DF', 'dev_chid': 2, 'dev_tag': 'WHI-ATC01', 'val_descr
iption': 'Air Pressure', 'val_calibrated': 101.556, 'val_date': 1620760648}, {'d
ev_netid': '70B3D54993D429DF', 'dev_chid': 255, 'dev_tag': 'WHI-ATC01', 'val_des
cription': 'Battery Voltage', 'val_calibrated': 4.075, 'val_date': 1620760648},
{'dev_netid': '70B3D54993D429DF', 'dev_chid': 254, 'dev_tag': 'WHI-ATC01', 'val_
description': 'Signal Strength RSSI', 'val_calibrated': -79, 'val_date': 1620760
648}]
5.61
>>>
```

8. Assigning temperature data into a variable

- Now that we know how to find and extract temperature data we can also pass on this value to a **variable**.
- Putting the temperature data in a variable allows us to save or manipulate the data more easily, just like in algebra.
- Add the following code:
    - **temp = r.json()[0][“val_calibrated”]** – assign temperature data to variable named **temp**
    - **print(temp)** – display the value held in **temp** in the Python Shell

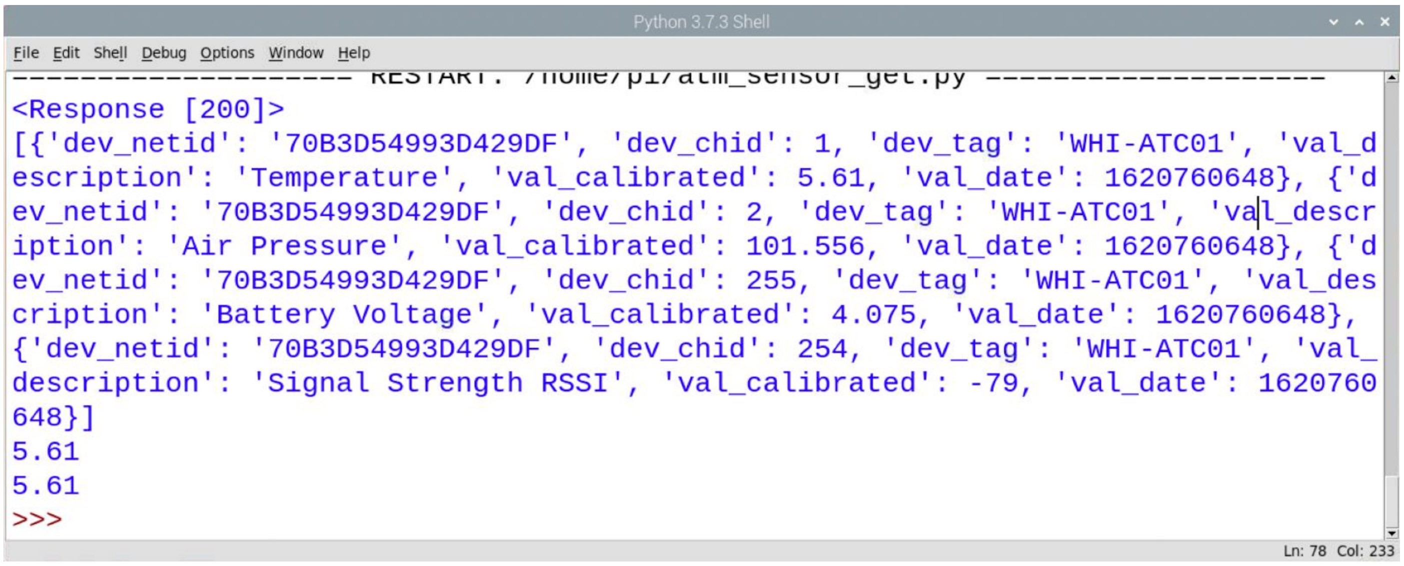
- Save and then Run this program.


atm_sensor_get.py - /home/pi/atm_sensor_get.py (3.7.3)

```
# atm_sensor_get.py
import requests

r = requests.get("https://app.alphax.cloud/api/WHI?tag=WHI-ATC01")
print(r)
print(r.json())

# only the temperature data
print(r.json()[0]["val_calibrated"])
temp = r.json()[0]["val_calibrated"]
print(temp)
```

Ln: 5 Col: 8

- The output of the program is shown below in the Python Shell window.
- The very last line of output (5.61) shows the output of the **print(temp)** statement.

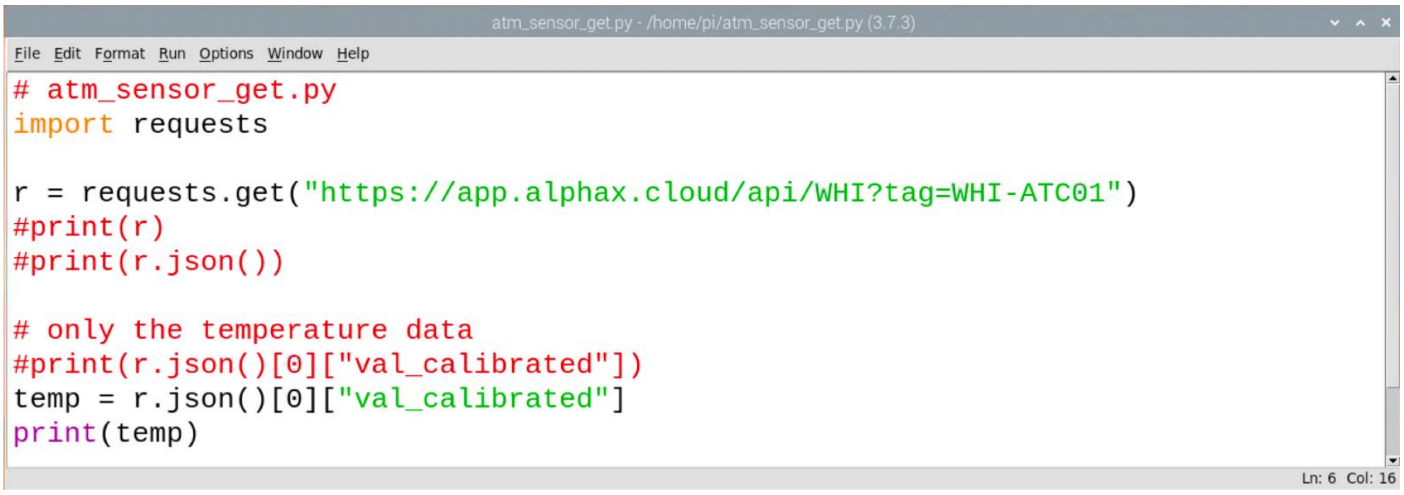

Python 3.7.3 Shell

```
==================== RESTART: /home/pi/atm_sensor_get.py ====================
<Response [200]>
[{'dev_netid': '70B3D54993D429DF', 'dev_chid': 1, 'dev_tag': 'WHI-ATC01', 'val_d
escription': 'Temperature', 'val_calibrated': 5.61, 'val_date': 1620760648}, {'d
ev_netid': '70B3D54993D429DF', 'dev_chid': 2, 'dev_tag': 'WHI-ATC01', 'val_descr
iption': 'Air Pressure', 'val_calibrated': 101.556, 'val_date': 1620760648}, {'d
ev_netid': '70B3D54993D429DF', 'dev_chid': 255, 'dev_tag': 'WHI-ATC01', 'val_des
cription': 'Battery Voltage', 'val_calibrated': 4.075, 'val_date': 1620760648},
{'dev_netid': '70B3D54993D429DF', 'dev_chid': 254, 'dev_tag': 'WHI-ATC01', 'val_
description': 'Signal Strength RSSI', 'val_calibrated': -79, 'val_date': 1620760
648}]
5.61
5.61
>>>
```

Ln: 78 Col: 233

- The output of the program has a lot of information we no longer need.
- To reduce the clutter from the output we can **comment out** some of the statements in our program using the # (hash) symbol. Note that when we comment out statement the text will change to a red colour.
- If we **comment out** statements the computer will no longer execute (run) them.
- Follow the example below and comment out three of the **print()** statements.

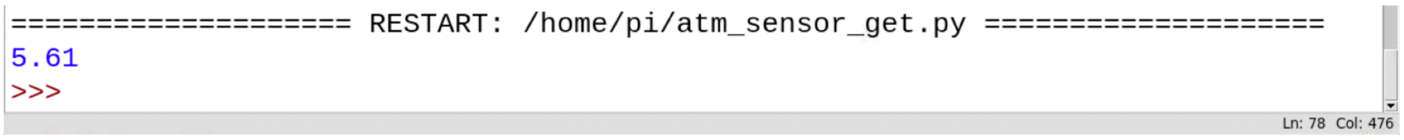
- Save and Run the program.


atm_sensor_get.py - /home/pi/atm_sensor_get.py (3.7.3)
File Edit Format Run Options Window Help
Ln: 6 Col: 16

```
# atm_sensor_get.py
import requests

r = requests.get("https://app.alphax.cloud/api/WHI?tag=WHI-ATC01")
#print(r)
#print(r.json())

# only the temperature data
#print(r.json()[0]["val_calibrated"])
temp = r.json()[0]["val_calibrated"]
print(temp)
```

- You can see that the output of the program has been simplified.

- We can always uncomment these print() statements in the future.



```
==================== RESTART: /home/pi/atm_sensor_get.py ====================
5.61
>>>
```

COLLECTING MORE DATA FROM THE ATMOSPHERIC SENSOR

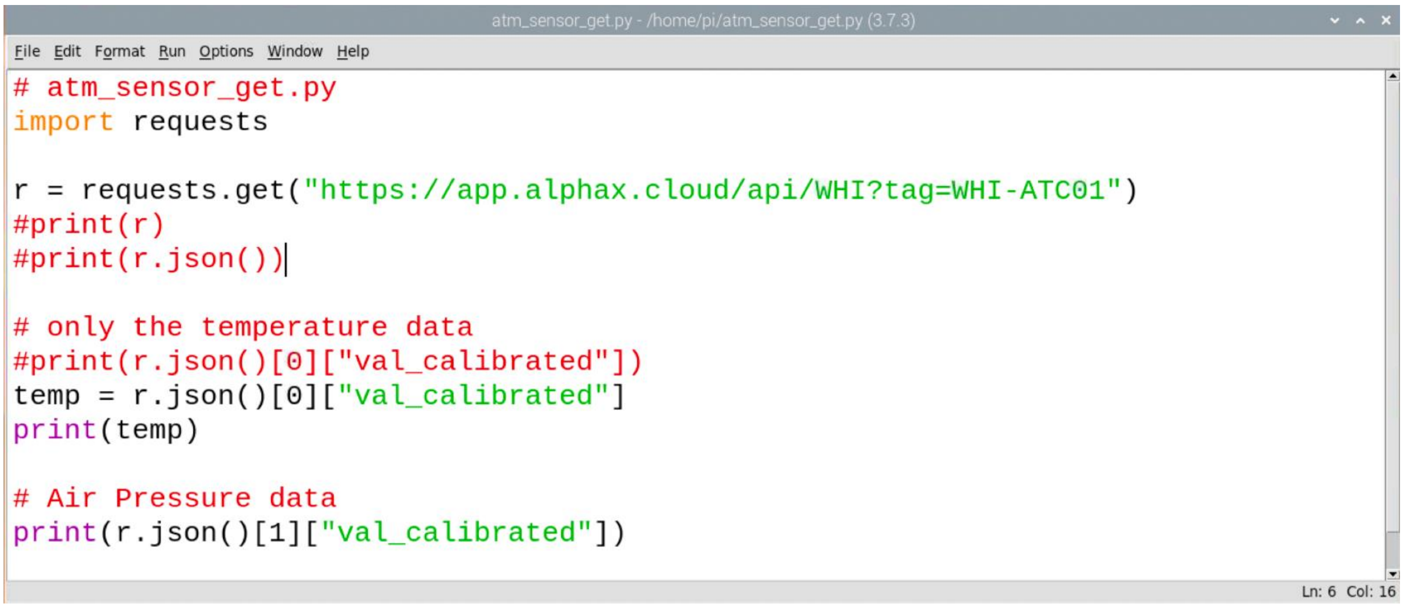
The atmospheric sensor collects other data in addition to temperature. The key data we are interested in is:

- **temperature** – measured in degrees Celsius
- **air pressure** – measured in kilopascals. 101kPa normal. 105kPa high pressure (fine weather).
- **battery voltage** – charge level of battery powering the sensor. Max voltage 4.20V. Min voltage 3.60V
- **signal strength** (RSSI) – a record of how powerful this signal is at the receiving station (LoRa Gateway)

In this example, we will capture data on the **air pressure** and the **battery voltage**.

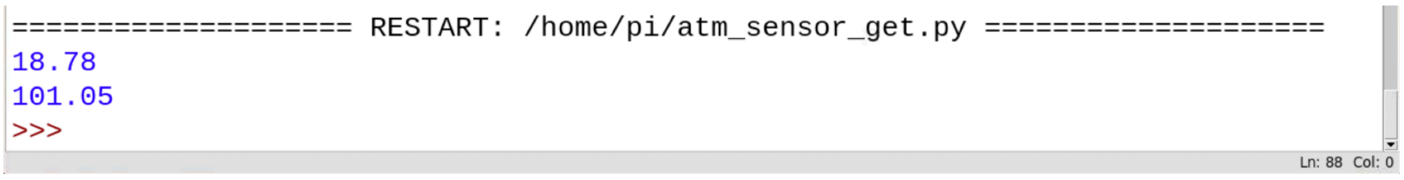
9. Extracting Air Pressure from JSON formatted data

- If we refer to the data presented in the **Firefox** web browser, we can see the data is divided into sections.
- Temperature data was in a section prefaced by **[0].**
- **Air Pressure** data is a section prefaced by **[1].**
- Using this information, we can extract the **Air Pressure** data using the following code.
    - **print(r.json()[1][“val_calibrated”])**
- Add this additional statement, then Save and Run the program.


atm_sensor_get.py - /home/pi/atm_sensor_get.py (3.7.3)
File Edit Format Run Options Window Help
Ln: 6 Col: 16

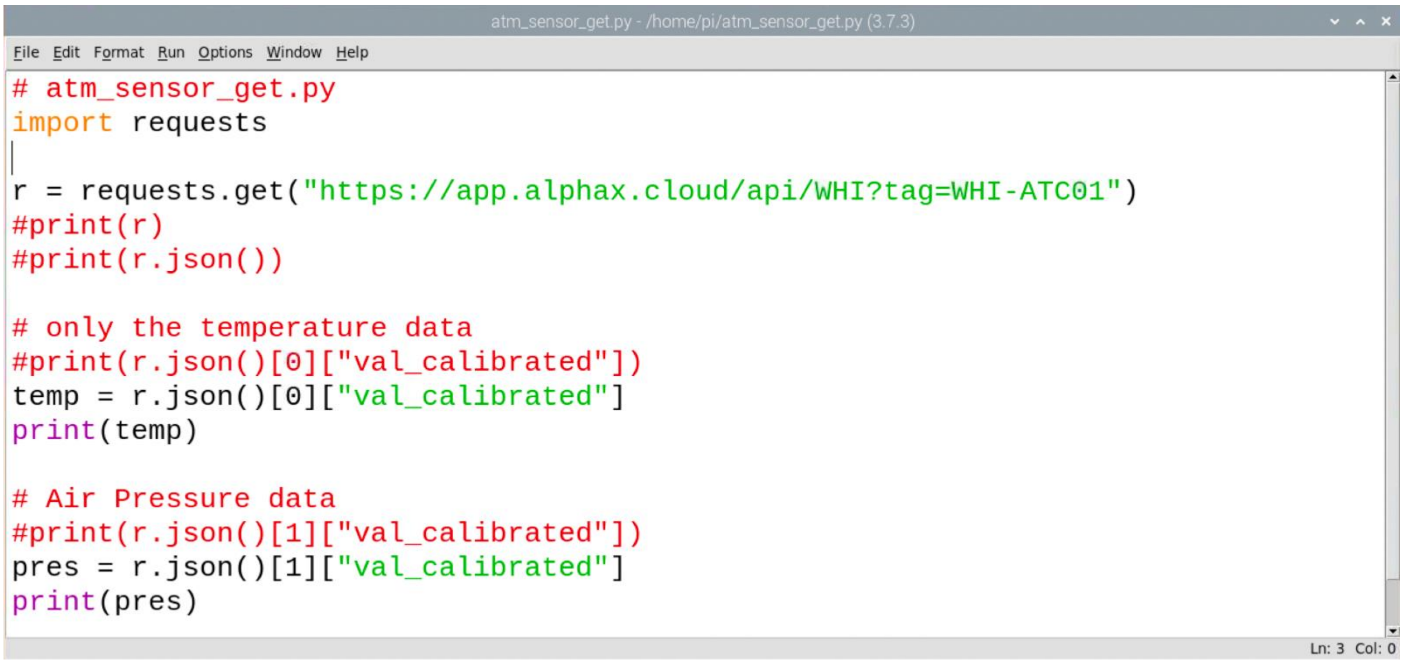
```
# atm_sensor_get.py
import requests

r = requests.get("https://app.alphax.cloud/api/WHI?tag=WHI-ATC01")
#print(r)
#print(r.json())

# only the temperature data
#print(r.json()[0]["val_calibrated"])
temp = r.json()[0]["val_calibrated"]
print(temp)

# Air Pressure data
print(r.json()[1]["val_calibrated"])
```

- This is the output in the python Shell Window.
- Both **Temperature** (18.78) and **Air Pressure** (101.05) data are now displayed.



```
==================== RESTART: /home/pi/atm_sensor_get.py ====================
18.78
101.05
>>>
```

- As for the temperature data, we can store the Air Pressure data in a variable.
- For **Air Pressure** we have created a new variable named **pres**
- Enter the following code, and also **comment out** the preceding print() statement.
  - **pres = r.json()[1][“val_calibrated”]**
  - **print(pres)**



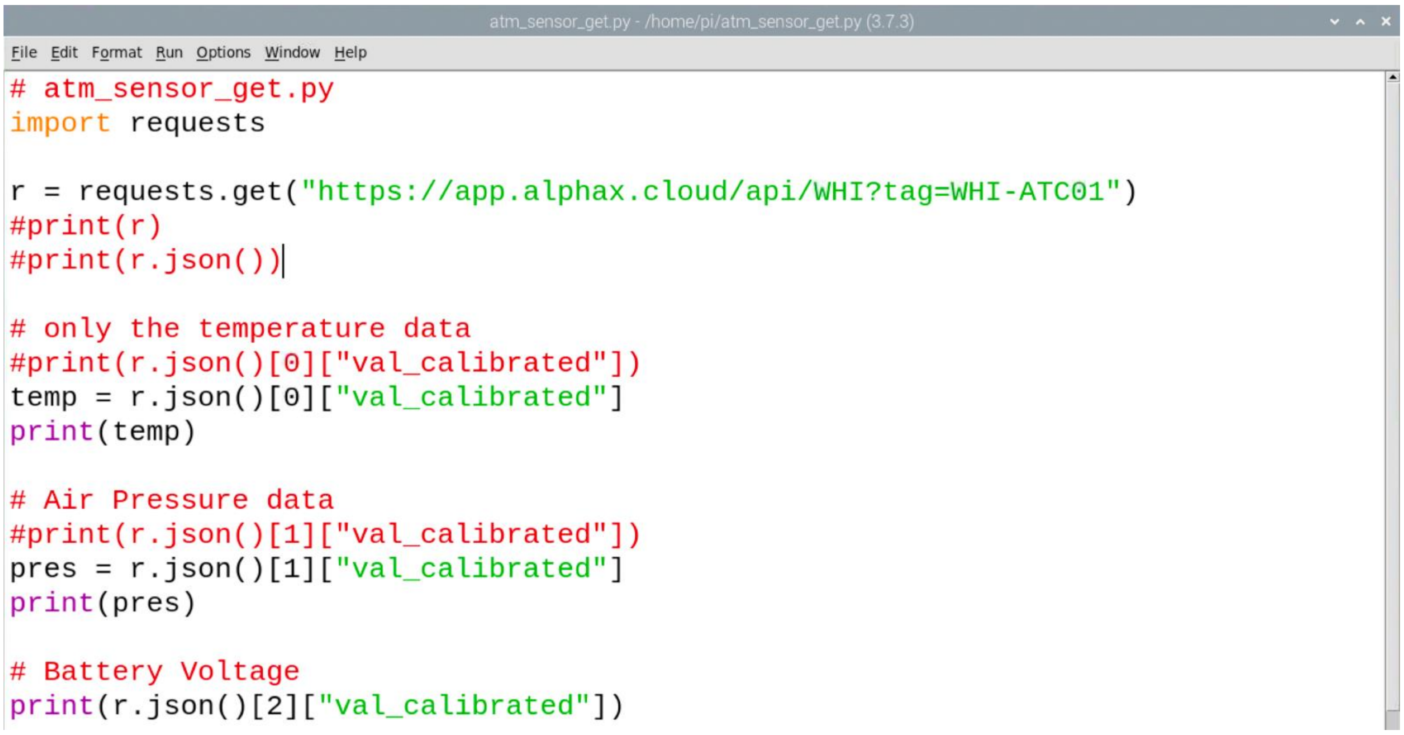
atm_sensor_get.py - /home/pi/atm_sensor_get.py (3.7.3)
File Edit Format Run Options Window Help
Ln: 3 Col: 0

```
# atm_sensor_get.py
import requests

r = requests.get("https://app.alphax.cloud/api/WHI?tag=WHI-ATC01")
#print(r)
#print(r.json())

# only the temperature data
#print(r.json()[0]["val_calibrated"])
temp = r.json()[0]["val_calibrated"]
print(temp)

# Air Pressure data
#print(r.json()[1]["val_calibrated"])
pres = r.json()[1]["val_calibrated"]
print(pres)
```

10. Extracting Battery Voltage from JSON formatted data

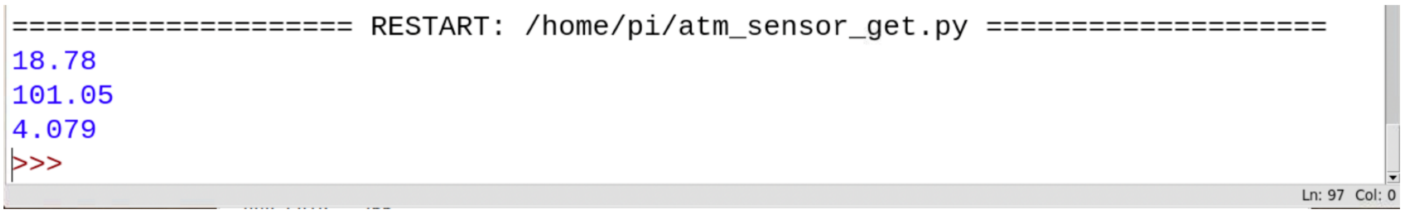
- In this last example we will also request Battery Voltage.

- Add the following code to extract the Battery Voltage data:
    - **print(r.json()[2][“val_calibrated”]**
- Save and Run the program.


atm_sensor_get.py - /home/pi/atm_sensor_get.py (3.7.3)
File Edit Format Run Options Window Help

```
# atm_sensor_get.py
import requests

r = requests.get("https://app.alphax.cloud/api/WHI?tag=WHI-ATC01")
#print(r)
#print(r.json())

# only the temperature data
#print(r.json()[0]["val_calibrated"])
temp = r.json()[0]["val_calibrated"]
print(temp)

# Air Pressure data
#print(r.json()[1]["val_calibrated"])
pres = r.json()[1]["val_calibrated"]
print(pres)

# Battery Voltage
print(r.json()[2]["val_calibrated"])
```

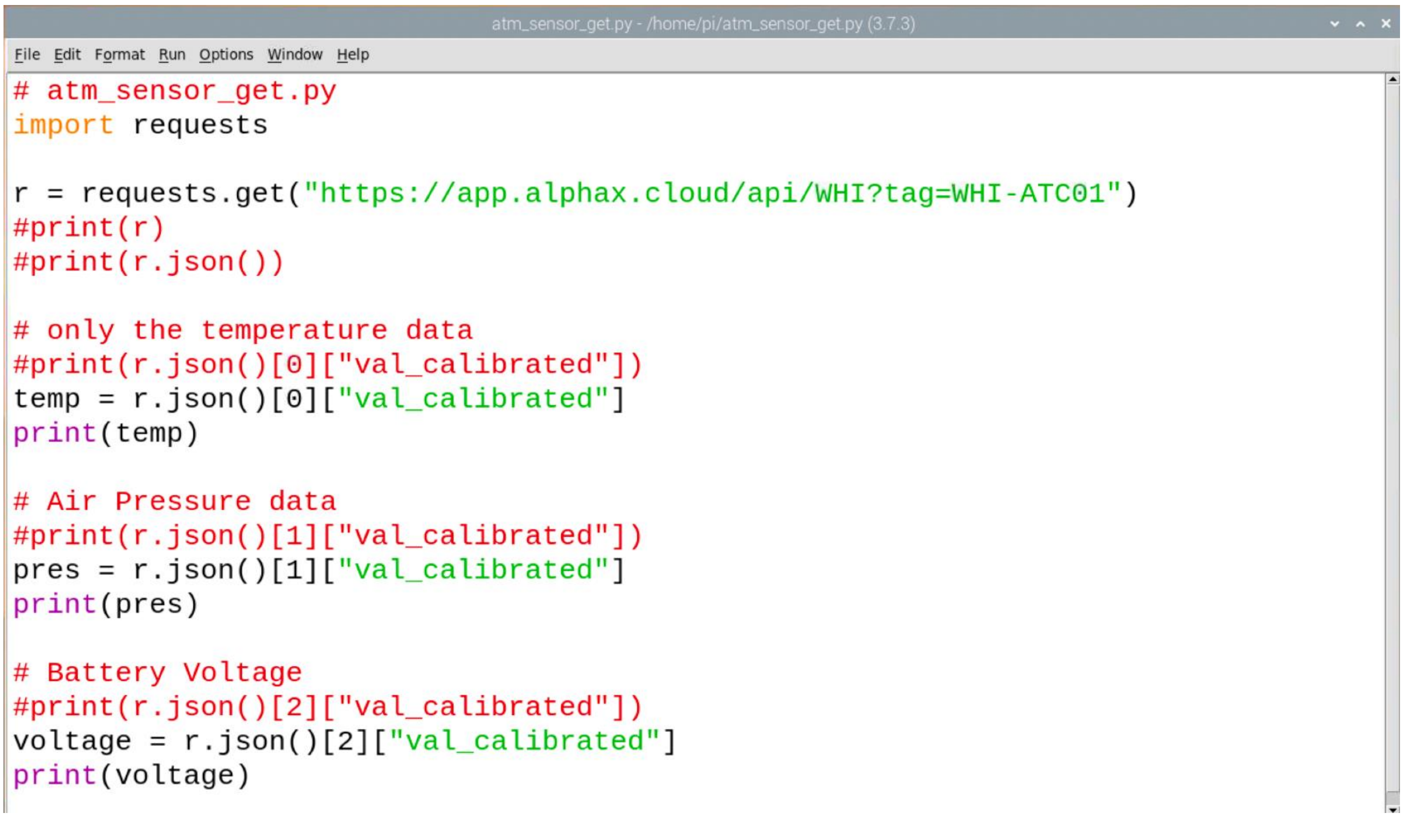
- The output in the python Shell Window will now show all three data values including Battery Voltage.
- If the battery voltage drops below 3.70 Volts, then the battery will need to be recharged.



```
==================== RESTART: /home/pi/atm_sensor_get.py ====================
18.78
101.05
4.079
>>>
```

- As a final step we will store the Battery Voltage data in a variable named **voltage**.
- Enter the following code and remember to comment out the preceding print() statement.
    - **voltage = r.json()[2][“val_calibrated”]**
    - **print(voltage)**

- Save and Run the program.


atm_sensor_get.py - /home/pi/atm_sensor_get.py (3.7.3)

```python
# atm_sensor_get.py
import requests

r = requests.get("https://app.alphax.cloud/api/WHI?tag=WHI-ATC01")
#print(r)
#print(r.json())

# only the temperature data
#print(r.json()[0]["val_calibrated"])
temp = r.json()[0]["val_calibrated"]
print(temp)

# Air Pressure data
#print(r.json()[1]["val_calibrated"])
pres = r.json()[1]["val_calibrated"]
print(pres)

# Battery Voltage
#print(r.json()[2]["val_calibrated"])
voltage = r.json()[2]["val_calibrated"]
print(voltage)
```

- Congratulations on completing this lesson. In the next lesson we will learn to save all this data in a file and also add a date/time stamp so that we have a record of when the data was collected.