SAVING SENSOR DATA TO A FILE

Edmond Lascaris and Anne Jessup - Creative Commons – 8 June 2021

OVERVIEW

In this lesson we are going to learn how to present sensor data in a simple web page. The web page will be **dynamic** because we will update the web page with new data each time we run our **atm_sensor_get.py** program.

Presenting **dynamic** data in a web page makes it much easier to monitor changes. Normally a **dynamic** web page is hosted on a web server. If data is hosted on a web server then you can view your sensor data from anywhere in the world. This very useful when you need to routinely monitor data coming from an important project.

Today we will only be looking at simple weather sensor data: temperature, atmospheric pressure and battery voltage. We can also use sensors to monitor other important projects. We use water level sensors to monitor the health of waterbodies. Some waterbodies are home to the threatened Growling Grass Frog. They need water to be present all year round (especially in summer) so that they can breed successfully. We will develop and deploy these types of sensors soon!

LEARNING OBJECTIVES

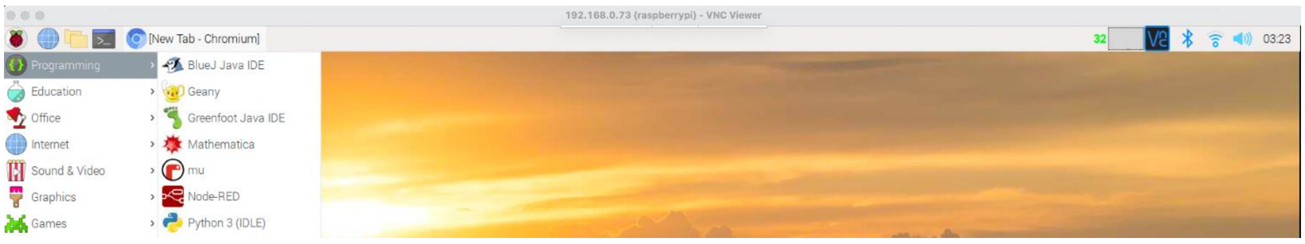
- Learn how create a dynamic html file using python
- Learn how to display the html file using a web browser
- Learn how to troubleshoot using the Terminal

CREATE A DYNAMIC HTML FILE USING PYTHON

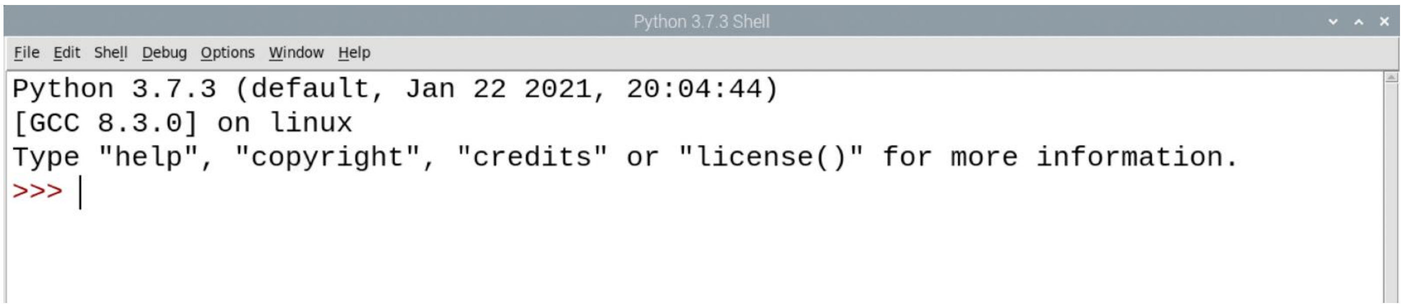
In this example we will experiment with a new python library named **html** that will help us create **dynamic** html pages. We will demonstrate how we can include new sensor data in a html file (web page).

1. **Simplify existing atm_sensor_get.py file**

- From the Raspberry Pi main menu drop down, select **Programming** > **Python3 (IDLE).**

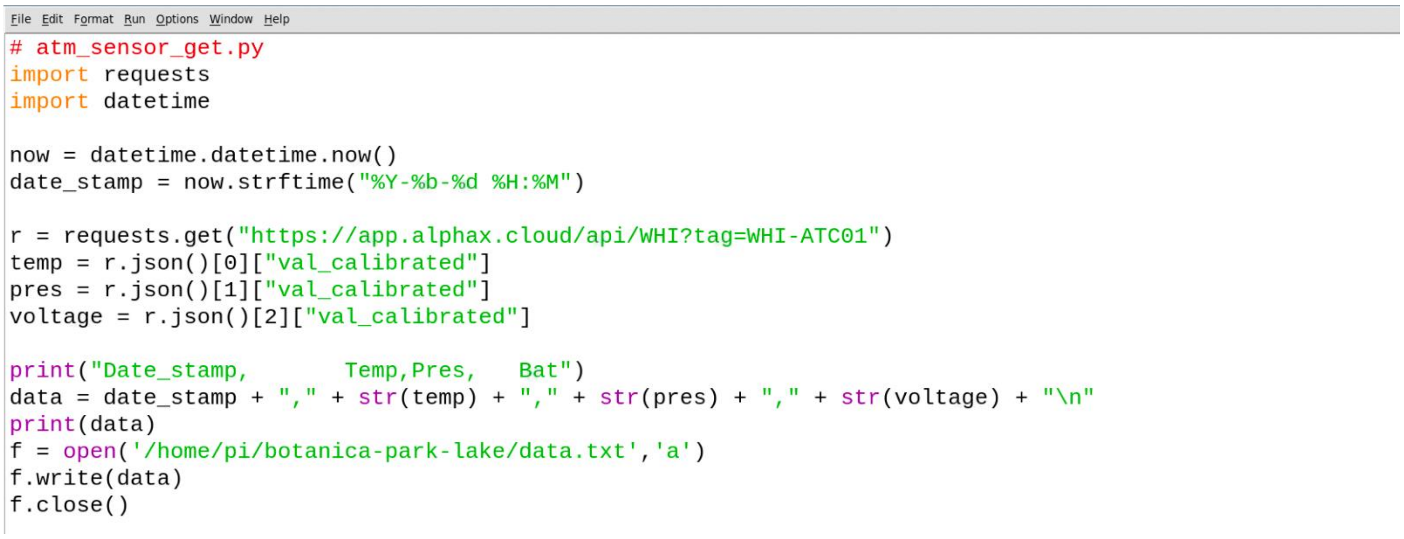


- This will open the Python Shell.



```
Python 3.7.3 (default, Jan 22 2021, 20:04:44)
[GCC 8.3.0] on linux
Type "help", "copyright", "credits" or "license()" for more information.
>>> 
```

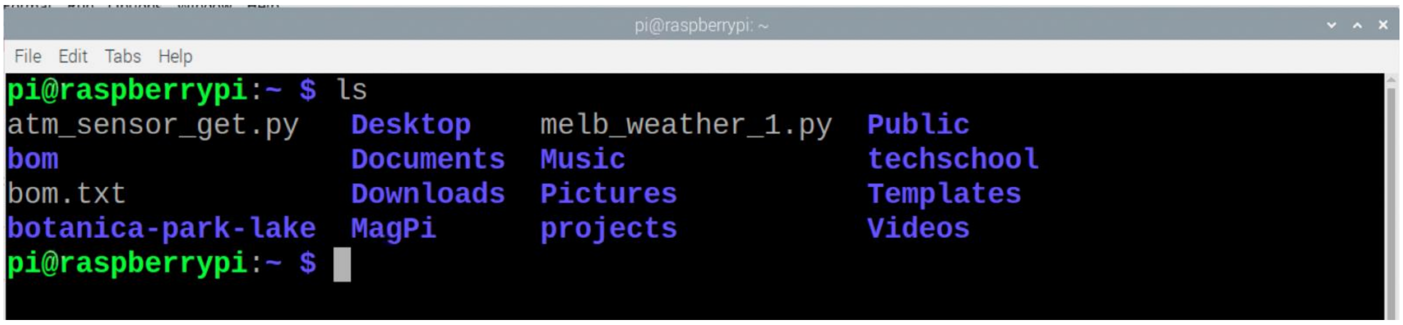
- From the **File** drop down menu select **Recent Files** and open **atm_sensor_get.py.**
- Edit the file to **remove** some of the following:
    1. Comments (**comments start with a #)**
    2. **print()** statements.
    3. White spaces between code statements.
- Your code should look like this when editing is complete. **Save** the changes.



```python
# atm_sensor_get.py
import requests
import datetime

now = datetime.datetime.now()
date_stamp = now.strftime("%Y-%b-%d %H:%M")

r = requests.get("https://app.alphax.cloud/api/WHI?tag=WHI-ATC01")
temp = r.json()[0]["val_calibrated"]
pres = r.json()[1]["val_calibrated"]
voltage = r.json()[2]["val_calibrated"]

print("Date_stamp,        Temp,Pres,   Bat")
data = date_stamp + "," + str(temp) + "," + str(pres) + "," + str(voltage) + "\n"
print(data)
f = open('/home/pi/botanica-park-lake/data.txt','a')
f.write(data)
f.close()
```

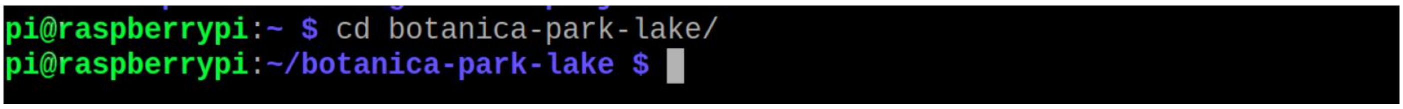
2. Create an empty file named atm.html using the Terminal

- Open the **Terminal**.
- Enter the command **ls** to list the contents in the **/home/pi** directory (home directory path shown as **~**)



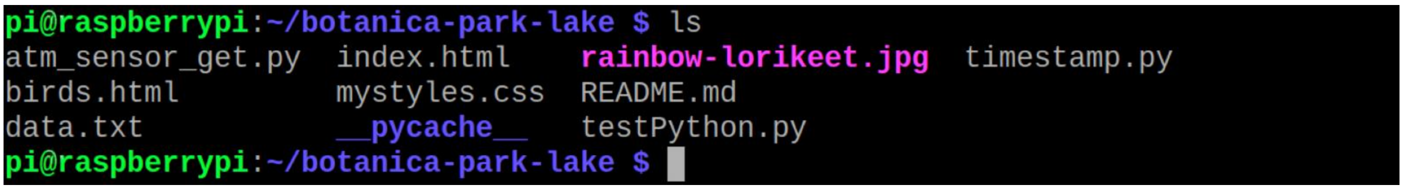
```
pi@raspberrypi:~ $ ls
atm_sensor_get.py    Desktop     melb_weather_1.py  Public
bom                  Documents   Music              techschool
bom.txt              Downloads   Pictures           Templates
botanica-park-lake   MagPi       projects           Videos
pi@raspberrypi:~ $
```

- Navigate to your project directory (e.g. **botanica-park-lake)** using the command **cd botanica-park-lake**
- As a short-cut, you can press the **Tab key** to help complete a statement.
- Try it for yourself. Enter **cd bot** – then enter the **Tab** key to complete it.



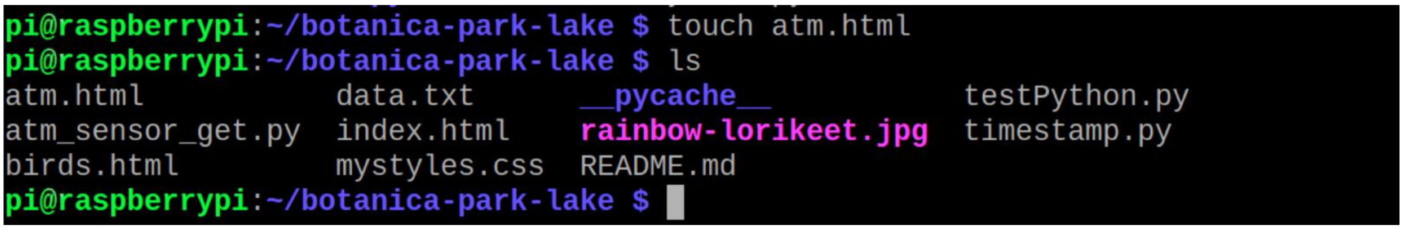
```
pi@raspberrypi:~ $ cd botanica-park-lake/
pi@raspberrypi:~/botanica-park-lake $
```

- Enter **ls** to view the contents of the directory.



```
pi@raspberrypi:~/botanica-park-lake $ ls
atm_sensor_get.py  index.html    rainbow-lorikeet.jpg  timestamp.py
birds.html         mystyles.css  README.md
data.txt           __pycache__   testPython.py
pi@raspberrypi:~/botanica-park-lake $
```

- To create a new file named **atm.html** enter the command **touch atm.html**
- Verify that the new file was created with the command **ls**
- This file will be our **dynamic web page** to display current atmospheric sensor data.



```
pi@raspberrypi:~/botanica-park-lake $ touch atm.html
pi@raspberrypi:~/botanica-park-lake $ ls
atm.html           data.txt      __pycache__           testPython.py
atm_sensor_get.py  index.html    rainbow-lorikeet.jpg  timestamp.py
birds.html         mystyles.css  README.md
pi@raspberrypi:~/botanica-park-lake $
```

3. String variable over several lines

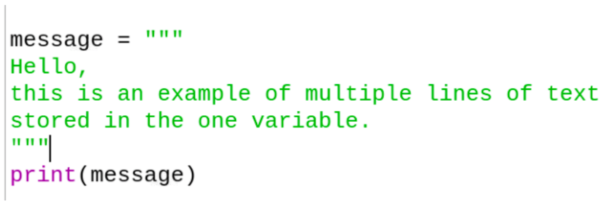
- Normally when we create a String it is normally contained within a single line of code as in the **message = "Hello"** example below.

- A string must be contained within quotation marks.

```
message = "Hello"
print(message)
```

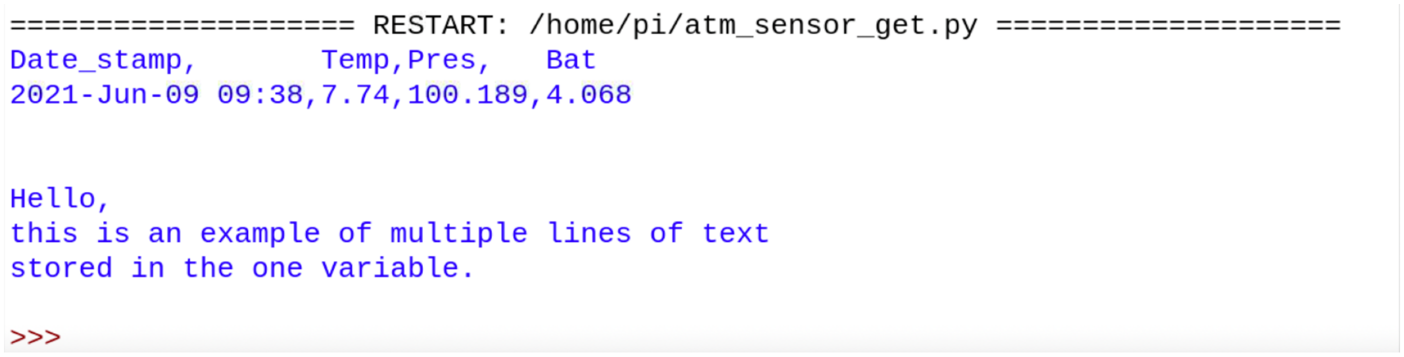
- If a String covers multiple lines, then we enclose the String with three quotation marks at the start (“””) and three quotation marks at the end (“””).

- The code below was added to the end of the atm_sensor.get.py section of code.



```
message = """
Hello,
this is an example of multiple lines of text
stored in the one variable.
"""
print(message)
```

- **Save** and **Run** this program. The output of the program in the Python Shell is shown below.



```
==================== RESTART: /home/pi/atm_sensor_get.py ====================
Date_stamp,        Temp,Pres,    Bat
2021-Jun-09 09:38,7.74,100.189,4.068


Hello,
this is an example of multiple lines of text
stored in the one variable.

>>>
```

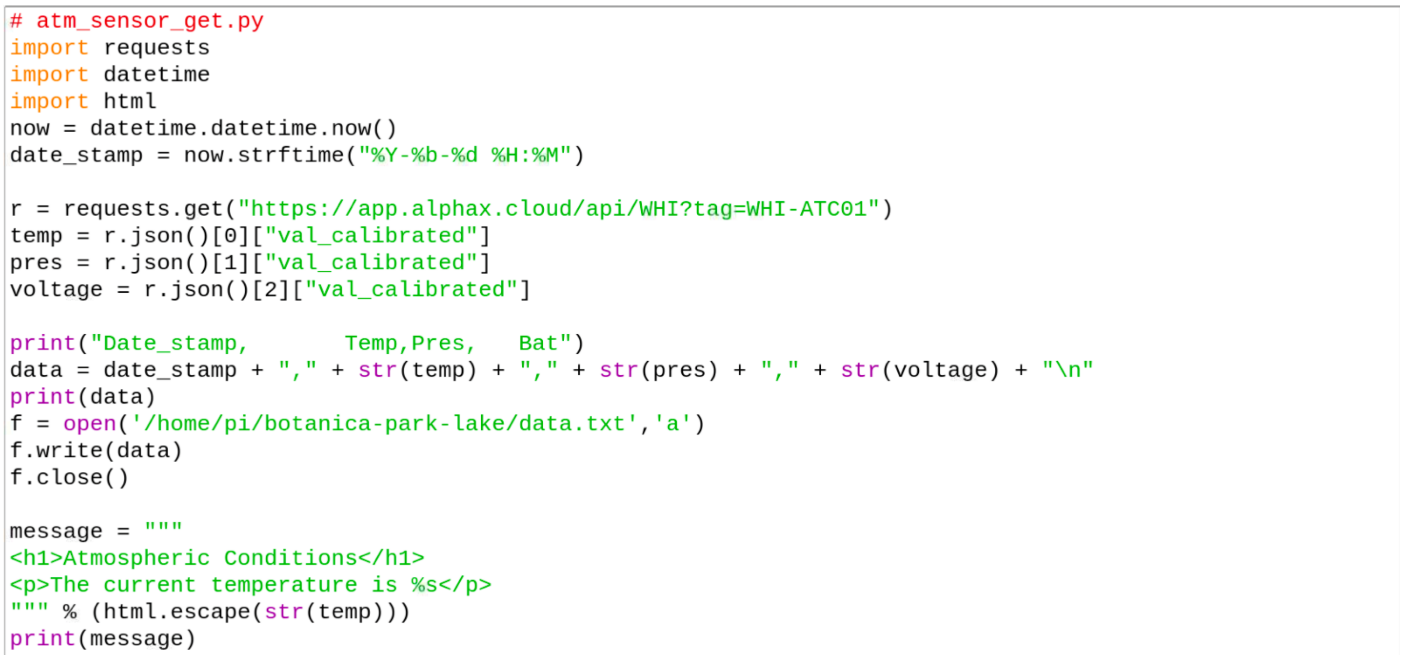
- Now we can write several lines of HTML within the multi-line String.

  1. Follow the example of the multi-line message String – **message = “””**

  2. Don’t forget to add an extra import statement at the top of the code – **import html**

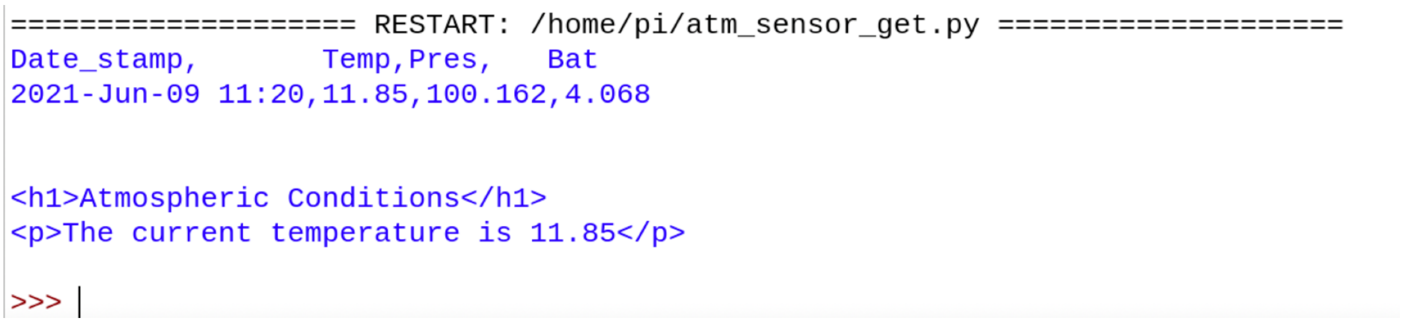
  3. Add a final **print()** statement so that you can see the message.

  4. We also need to add the **str()** method to convert numbers to String.



```python
# atm_sensor_get.py
import requests
import datetime
import html
now = datetime.datetime.now()
date_stamp = now.strftime("%Y-%b-%d %H:%M")

r = requests.get("https://app.alphax.cloud/api/WHI?tag=WHI-ATC01")
temp = r.json()[0]["val_calibrated"]
pres = r.json()[1]["val_calibrated"]
voltage = r.json()[2]["val_calibrated"]

print("Date_stamp,       Temp,Pres,   Bat")
data = date_stamp + "," + str(temp) + "," + str(pres) + "," + str(voltage) + "\n"
print(data)
f = open('/home/pi/botanica-park-lake/data.txt','a')
f.write(data)
f.close()

message = """
<h1>Atmospheric Conditions</h1>
<p>The current temperature is %s</p>
""" % (html.escape(str(temp)))
print(message)
```

- You should see the following output when you run the code.

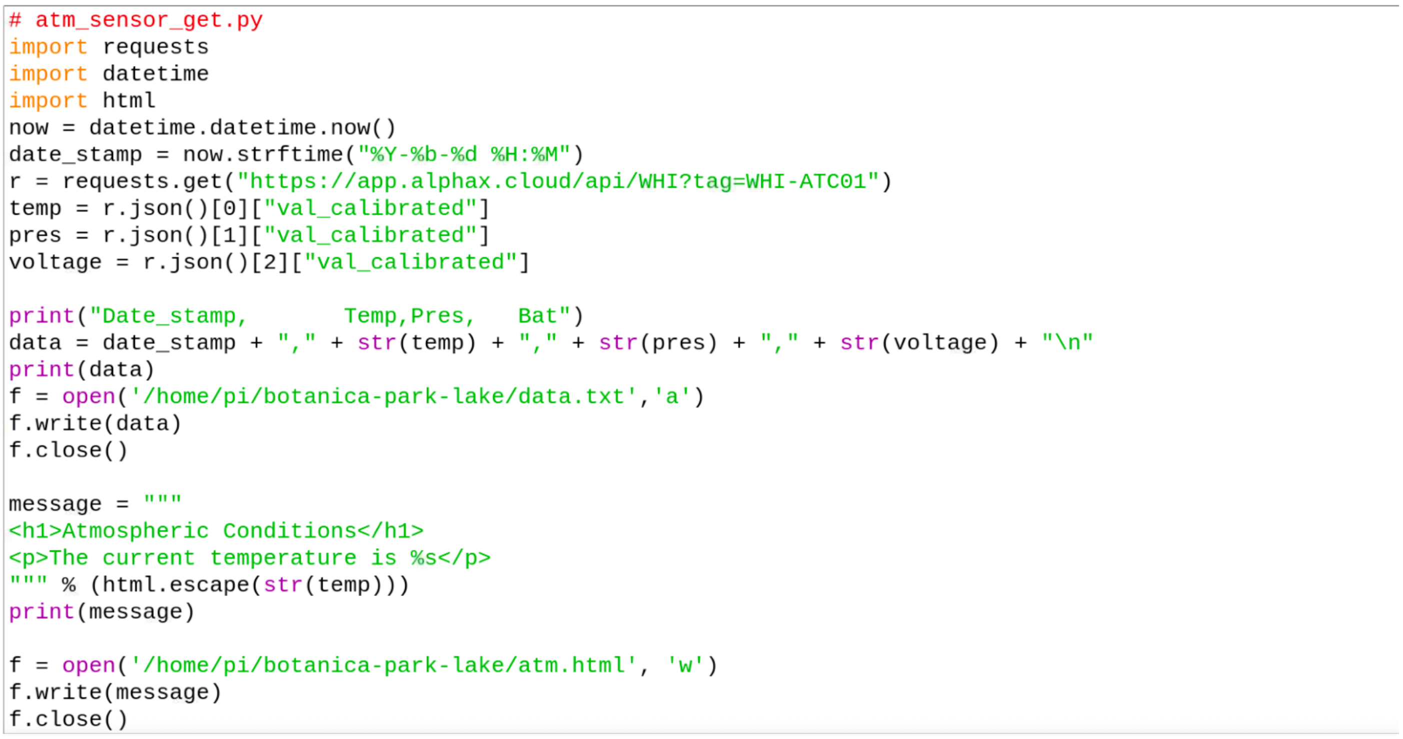


```
==================== RESTART: /home/pi/atm_sensor_get.py ====================
Date_stamp,       Temp,Pres,   Bat
2021-Jun-09 11:20,11.85,100.162,4.068


<h1>Atmospheric Conditions</h1>
<p>The current temperature is 11.85</p>

>>> |
```

4. Saving the message String to the atm.html file

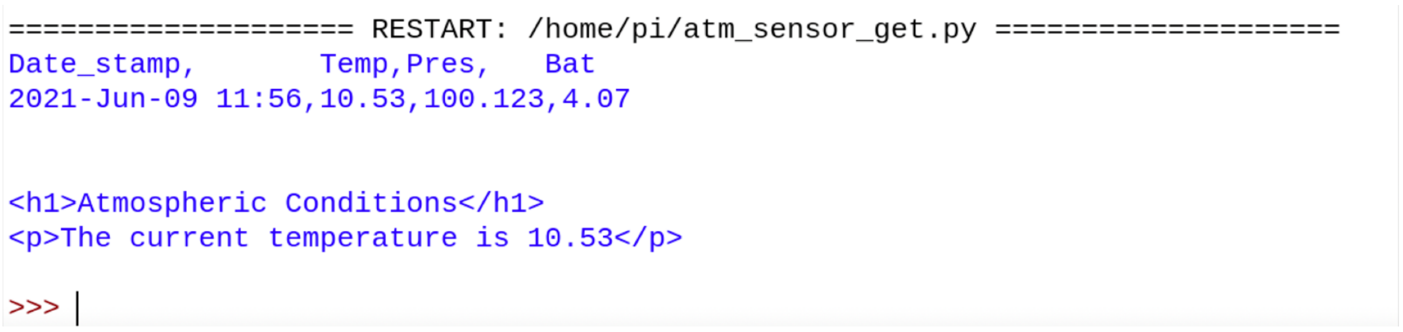
- Now that we have prepared our message String, we need to save it to our atm.html file.
- Add the following lines to the end of our program as per the example below.
  - **f = open('/home/pi/botanica-park-lake/atm.html', 'w')**
  - **f.write(message)**
  - **f.close()**
- In the first statement we used the **'w'** option, which is short for **write**.

- This means that when we **write** to the **atm.html** file - all existing contents will be **erased** and written over.
- If the option was **'a'** (short for **append**) – then we would add new content to the end of the file, preserving existing data in the file.



```python
# atm_sensor_get.py
import requests
import datetime
import html
now = datetime.datetime.now()
date_stamp = now.strftime("%Y-%b-%d %H:%M")
r = requests.get("https://app.alphax.cloud/api/WHI?tag=WHI-ATC01")
temp = r.json()[0]["val_calibrated"]
pres = r.json()[1]["val_calibrated"]
voltage = r.json()[2]["val_calibrated"]

print("Date_stamp,       Temp,Pres,   Bat")
data = date_stamp + "," + str(temp) + "," + str(pres) + "," + str(voltage) + "\n"
print(data)
f = open('/home/pi/botanica-park-lake/data.txt','a')
f.write(data)
f.close()

message = """
<h1>Atmospheric Conditions</h1>
<p>The current temperature is %s</p>
""" % (html.escape(str(temp)))
print(message)

f = open('/home/pi/botanica-park-lake/atm.html', 'w')
f.write(message)
f.close()
```

- The output will look like the following.
- Behind the scenes, our message String will have been written to the file atm.html



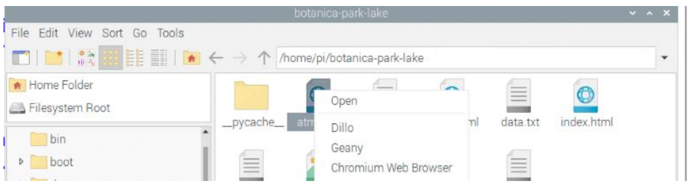
```
==================== RESTART: /home/pi/atm_sensor_get.py ====================
Date_stamp,       Temp,Pres,   Bat
2021-Jun-09 11:56,10.53,100.123,4.07



<h1>Atmospheric Conditions</h1>
<p>The current temperature is 10.53</p>

>>> 
```

DISPLAYING AN HTML FILE USING A WEB BROWSER

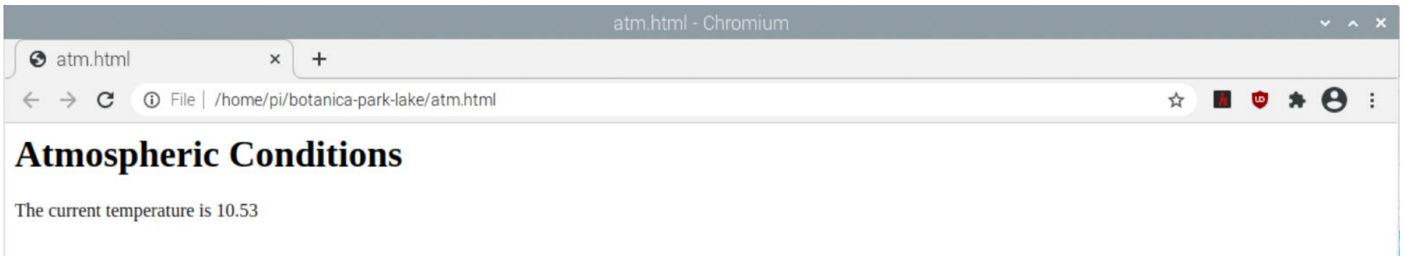
When developing an HTML file we want to be able to quickly view it without having to load it onto a live web server. One easy way to view an HTML file is to open it up in a Web Browser. The following example will show you how to use the Chromium Web Browser on the Raspberry Pi to view HTML files.

5. Viewing HTML files using a Web Browser

- To open the atm.html file open the **File Manager**.
- Find the file **atm.html** and **right-mouse button click**.
- Select the **Chromium Web Browser** option.



- This should open up the Chromium Web browser and show the following output.



Atmospheric Conditions

The current temperature is 10.53

6. Adding more sensor data to the HTML file

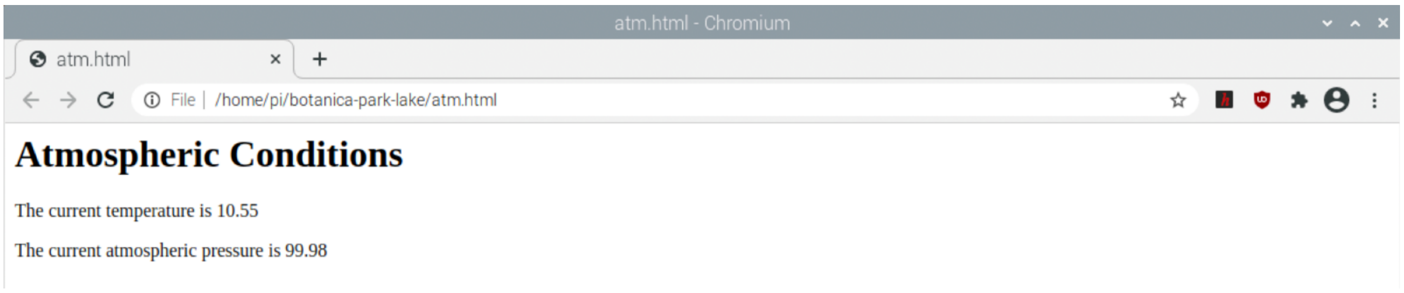
- Now that we know how to view our dynamic web page, we can start adding more sensor data.
- Modify the multi-line **message variable** to include atmospheric pressure data.
- Follow the example below.

```
message = """
<h1>Atmospheric Conditions</h1>
<p>The current temperature is %s</p>
<p>The current atmospheric pressure is %s</p>
""" % (html.escape(str(temp)), html.escape(str(pres)))
print(message)
```

- When you Run the program, you should see the following output in the Python Shell.

```
==================== RESTART: /home/pi/atm_sensor_get.py ====================
Date_stamp,        Temp,Pres,   Bat
2021-Jun-09 12:45,10.55,99.98,4.062


<h1>Atmospheric Conditions</h1>
<p>The current temperature is 10.55</p>
<p>The current atmospheric pressure is 99.98</p>

>>>
```

- You can also see the changes in the Web Browser.
- Don’t forget to click on the **Reload Page** button, so that the changes to the atm.html file are loaded into the Browser.



Atmospheric Conditions

The current temperature is 10.55

The current atmospheric pressure is 99.98

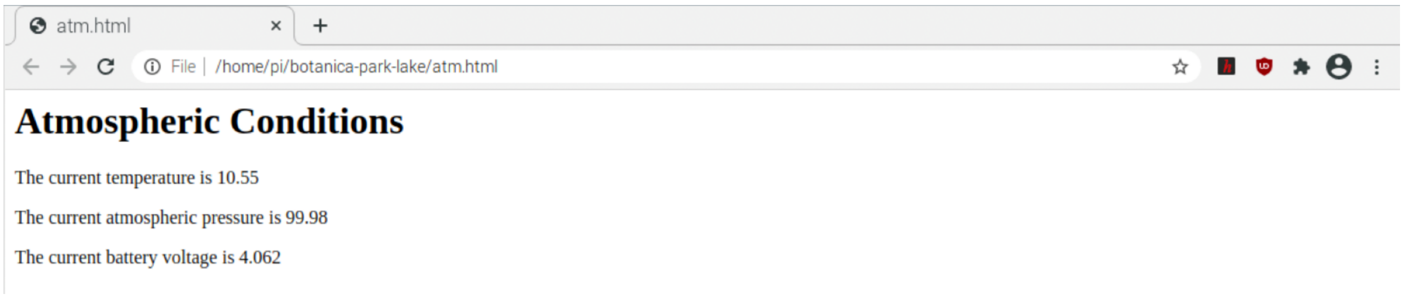
- Lastly, we can also include sensor data for battery voltage.

```
message = """
<h1>Atmospheric Conditions</h1>
<p>The current temperature is %s</p>
<p>The current atmospheric pressure is %s</p>
<p>The current battery voltage is %s</p>
""" % (html.escape(str(temp)), html.escape(str(pres)), html.escape(str(voltage)))
print(message)
```

- The output in the Python Shells looks like this.

```
==================== RESTART: /home/pi/atm_sensor_get.py ====================
Date_stamp,        Temp,Pres,   Bat
2021-Jun-09 12:51,10.55,99.98,4.062


<h1>Atmospheric Conditions</h1>
<p>The current temperature is 10.55</p>
<p>The current atmospheric pressure is 99.98</p>
<p>The current battery voltage is 4.062</p>
```

- The output in the web browser will look like the following.
- Don’t forget to click the **Reload this page** button.



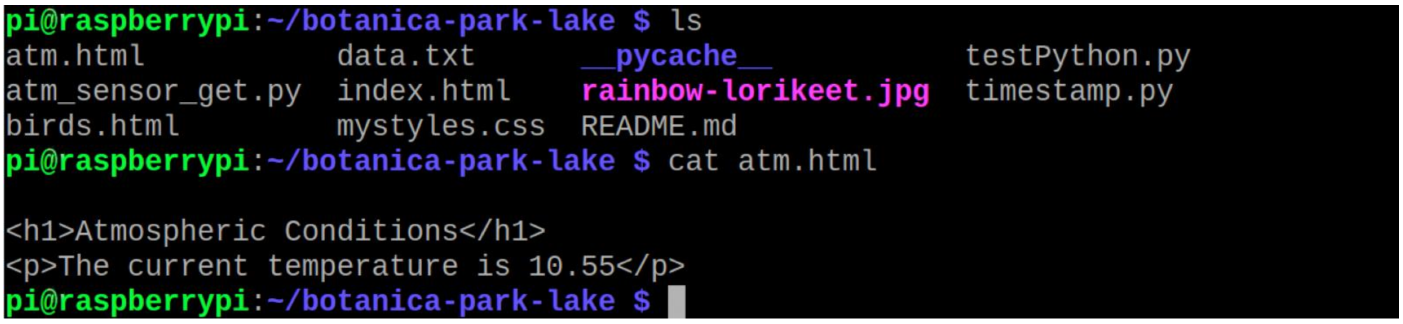
```
Atmospheric Conditions

The current temperature is 10.55

The current atmospheric pressure is 99.98

The current battery voltage is 4.062
```

TROUBLESHOOTING USING THE TERMINAL

If your web page doesn't display correctly you may have missed some code or received a syntax error. A syntax error is the equivalent of a grammatical error or spelling mistake in English. In this example we will use the Terminal to see if changes have been written to the atm.html file.

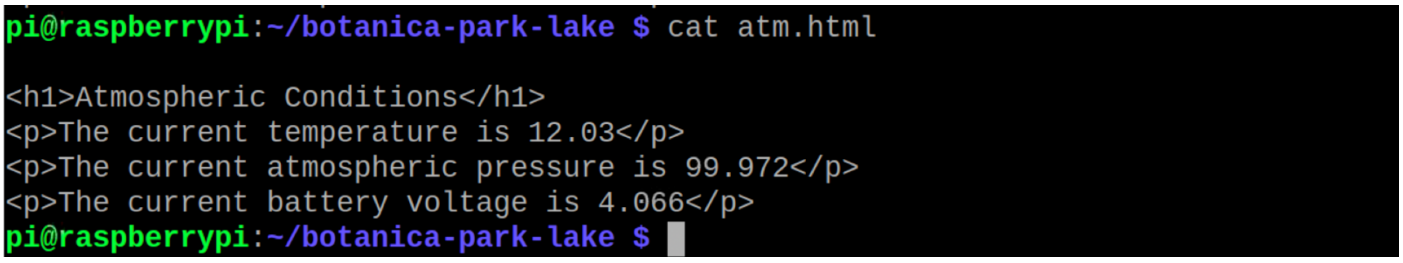
7. **Checking for file changes using the Terminal command cat**

- Open the **Terminal** and navigate to your project directory.
- List the files in the project directory with the command **ls**
- You should see your atm.html file.
- To inspect the contents of the atm.html file, enter the command **cat atm.html**
- This will produce a listing of the file contents.



```
pi@raspberrypi:~/botanica-park-lake $ ls
atm.html            data.txt      __pycache__          testPython.py
atm_sensor_get.py   index.html    rainbow-lorikeet.jpg timestamp.py
birds.html          mystyles.css  README.md
pi@raspberrypi:~/botanica-park-lake $ cat atm.html

<h1>Atmospheric Conditions</h1>
<p>The current temperature is 10.55</p>
pi@raspberrypi:~/botanica-park-lake $
```

- In the example above, some of the html in the file appears to be missing. Perhaps there was an error in my program.
- After fixing the error and running the program again, this is the output I get.
- Using the **command cat** is an easy way to inspect your files.



```
pi@raspberrypi:~/botanica-park-lake $ cat atm.html

<h1>Atmospheric Conditions</h1>
<p>The current temperature is 12.03</p>
<p>The current atmospheric pressure is 99.972</p>
<p>The current battery voltage is 4.066</p>
pi@raspberrypi:~/botanica-park-lake $
```

- Congratulations! Now you can make you're own dynamic web pages.

- In our next lesson we will try to automate the creation of these dynamic web pages.