

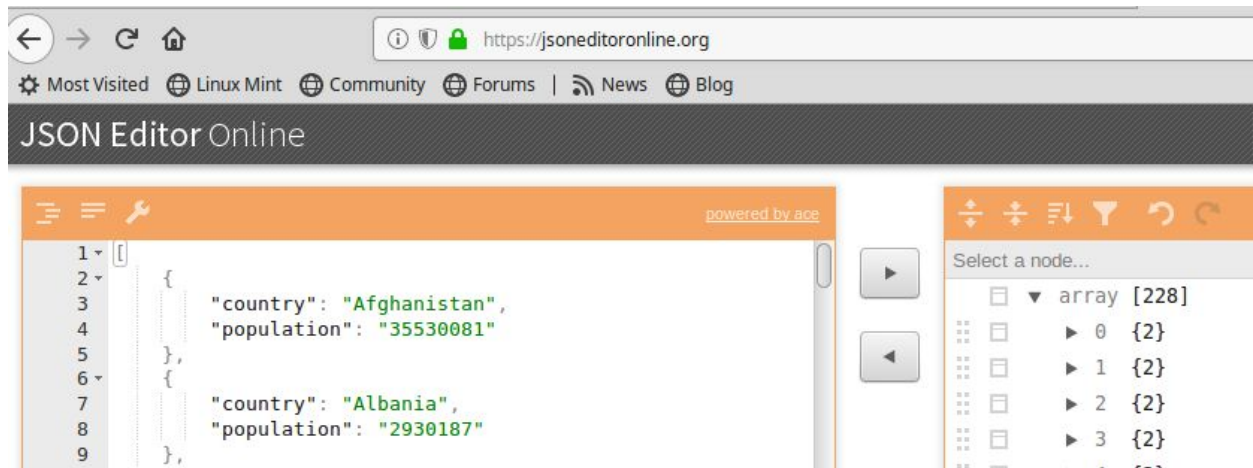
# Python writes HTML

FACT 1: Python is a great language for reading and writing data in files, but it is not so great with graphics or sharing output.

FACT 2: HTML browsers with CSS are great at graphics and sharing output on the WWW, but Javascript does not read and write files.

So ... let's use Python to read in a data file (Figure 1) and write out an HTML (Figure 2)!

**Figure 1. Raw data in <https://jsononlineeditor.org>**



**Figure 2. Processed (sorted) data in an HTML browser, ready to share with the world**

A screenshot of a web browser displaying a table of country names and their populations. The address bar shows 'file:///home/purple\_admin/Desktop/test.html'. The table lists China, India, United States, Indonesia, Brazil, and Pakistan with their respective population values.

China	1409517397
India	1339180127
United States	324459463
Indonesia	263991379
Brazil	209288278
Pakistan	197015955

Step 1: Open Idle and create a new File

Idle is a simple Integrated Development Environment (IDE) for Python that will help you be a more productive programmer.

## Step 2: Download countries.json to your work folder

## Step 3: Code the load!

To read JavaScript Object Notation (JSON) files in Python, use the very handy Python json module.

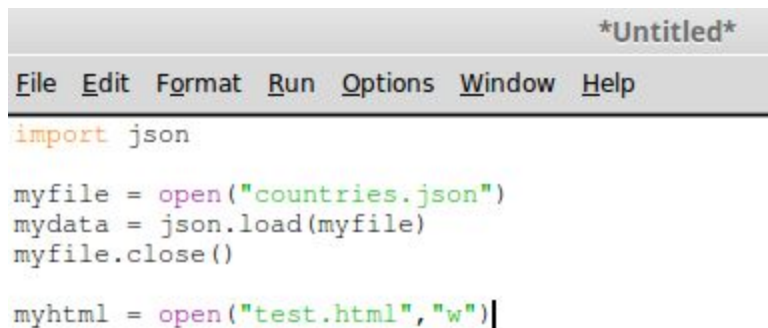


```
*Untitled*
File Edit Format Run Options Window Help
import json
|
myfile = open("countries.json")
mydata = json.load(myfile)
myfile.close()
```

In the code above we import the **json** module, **open** countries.json into a variable called **myfile** and then **load** it as a json object in the variable called **mydata**. Once loaded, you can safely **close** myfile.

## Step 4: Create the HTML file

Use the “w” or write in the open command to write a new file to your work folder.



```
*Untitled*
File Edit Format Run Options Window Help
import json

myfile = open("countries.json")
mydata = json.load(myfile)
myfile.close()

myhtml = open("test.html", "w")|
```

## Step 5: Sort the data on the population attribute

The sorted command allows you to sort data and store it in a variable. In this case we overwrite the existing variable with the sorted data. Notice that we had to type cast population to an integer in order for the numeric sort to work as the population data are strings by default.

```
*Untitled*
File Edit Format Run Options Window Help

import json

myfile = open("countries.json")
mydata = json.load(myfile)
myfile.close()

myhtml = open("test.html", "w")

mydata = sorted(mydata, key=lambda x: int(x['population']), reverse=True)|
```

Step 6: Loop through data and write out each country / population pair

The **.write()** function allows you to write to a file object. In this case we are writing an HTML table with each table row <tr> being a different country and its population. Make sure to use the **.close()** function to close the file object, otherwise the write will not complete.

```
*Untitled*
File Edit Format Run Options Window Help

import json

myfile = open("countries.json")
mydata = json.load(myfile)
myfile.close()

myhtml = open("test.html", "w")

mydata = sorted(mydata, key=lambda x: int(x['population']), reverse=True)

myhtml.write("<table>")
for country in mydata:
    myhtml.write("<tr>")
    myhtml.write("<td>" + country["country"] + "</td>")
    myhtml.write("<td>" + str(country["population"]) + "</td>")
    myhtml.write("<tr>")
myhtml.write("</table>")
myhtml.close()|
```

Step 7: Save the file to work, as countries.py and run the program

## CONGRATULATIONS!

# Enrichment 1# - Automatically open file

Wouldn't it be nice for your new HTML to pop open immediately after it is created? Try these 4 lines of code:

Code to add

```
Import os
Import webbrowser
filename = 'file:///'+os.getcwd()+ '/' + 'test.html'
webbrowser.open_new_tab(filename)
```

Usually it is a good idea to group your import statements together for good organization of your code. Finally, as code runs sequentially (one line after another), place the webbrowser code at the bottom of the file.



```
writeit.py - /home/purple_admin/Desktop/writeit.py (3.4.3)
File Edit Format Run Options Window Help

import os
import json
import webbrowser

# open data for reading and load into a variable, then close
myfile = open("countries.json")
mydata = json.load(myfile)
myfile.close()

# open a html file for writing
myhtml = open("test.html", "w")

# sort data
mydata = sorted(mydata, key=lambda x: int(x['population']), reverse=True)

# loop over data and write it to a table, then close to finish write
myhtml.write("<table>")
for country in mydata:
    myhtml.write("<tr>")
    myhtml.write("<td>" + country)
    myhtml.write("<td>" + str(country))
    myhtml.write("<tr>")
myhtml.write("</table>")
myhtml.close()

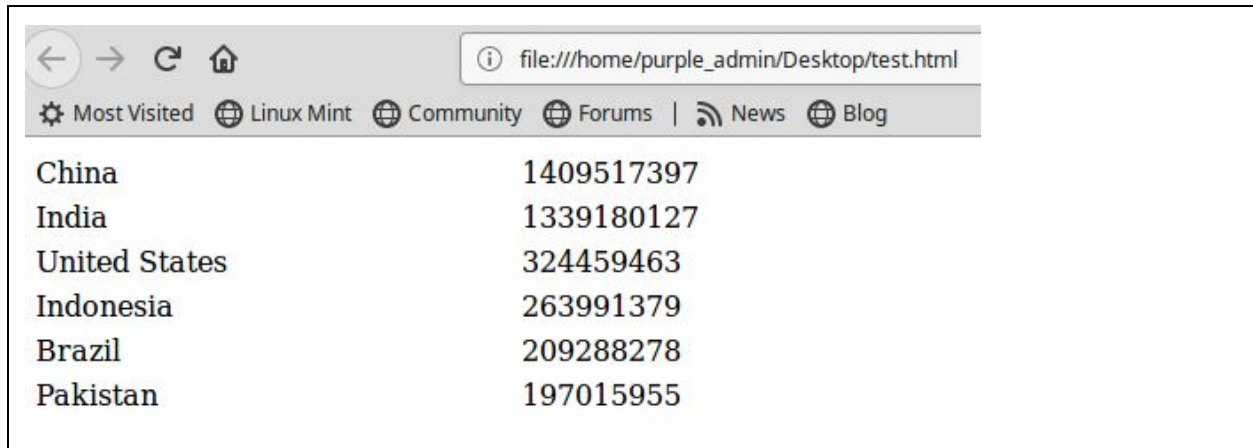
# open in a new browser tab
filename = 'file:///'+os.getcwd()+ '/' + 'test.html'
webbrowser.open_new_tab(filename)
```

**Import os and webbrowser modules**

**Create variable with "full path" and Open it up with webbrowser!**

# Enrichment 2# - Add CSS

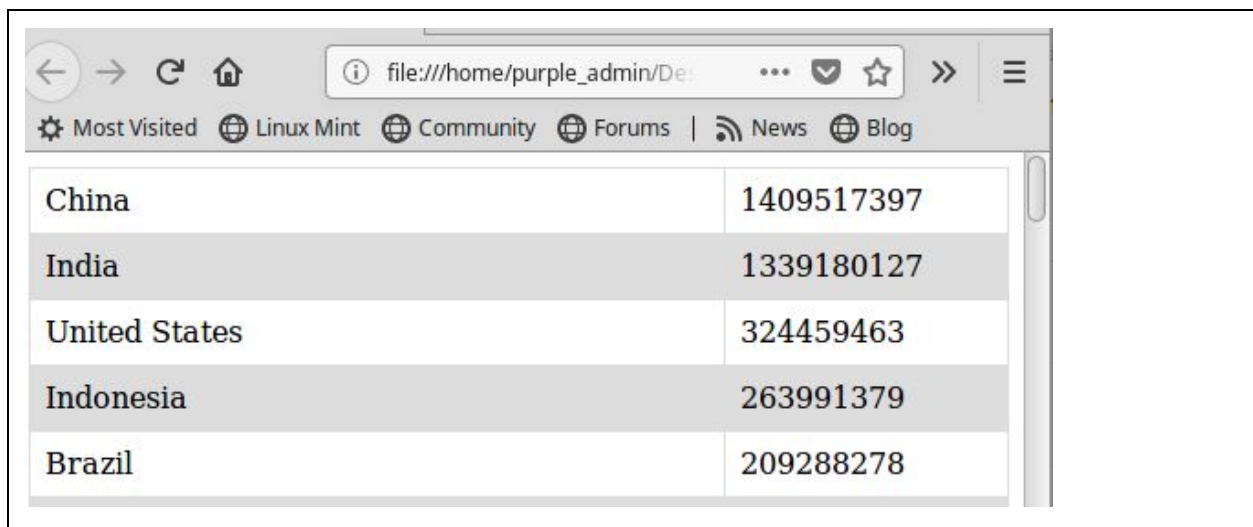
I think we can agree that the table is quite ugly and plain as it is.



A screenshot of a web browser window displaying a plain HTML table. The browser's address bar shows the file path: file:///home/purple\_admin/Desktop/test.html. The table lists countries and their corresponding population values.

China	1409517397
India	1339180127
United States	324459463
Indonesia	263991379
Brazil	209288278
Pakistan	197015955

Why not add some custom CSS to make it more interesting, like the image below.



A screenshot of a web browser window displaying the same table as before, but with custom CSS styling. The table has a border, and the rows are styled with alternating background colors (white and light gray). The text is left-aligned with padding.

China	1409517397
India	1339180127
United States	324459463
Indonesia	263991379
Brazil	209288278

Five lines of code to add some CSS to the <table>, <td>, <th> and <tr> tags. Voila!

```
mydata = sorted(mydata, key=lambda x: int(x['population']), reverse=True)

# Write out CSS to file
myhtml.write("<style>")
myhtml.write("table { border-collapse: collapse; width: 100%; }")
myhtml.write("td, th { border: 1px solid #dddddd; text-align: left; padding: 8px;}")
myhtml.write("tr:nth-child(even) { background-color: #dddddd;}")
myhtml.write("</style>")

myhtml.write("<table>")
for country in mydata:
    myhtml.write("<tr>")
```