Notes_convert_txt_json

June 28, 2021

tetxtfile to json (June 26, 2021), easy to complex TRYING Example 1

[238]:
```
import json
#filename = '/home/hdbhatt/Ana/txt_to_json/data.txt'
filename = '/home/hdbhatt/Ana/txt_to_json/runs_info.txt'
```

[239]:
```
#The textfile was this, where first word is key and the second one is the value␣
 ↪of the key
#name Lisa
#designation Programmer
#age 34
#salary 54000
```

dict where the lines from txt file will be stored

[240]:
```
print (filename)
dict1 ={}
```

```
/home/hdbhatt/Ana/txt_to_json/runs_info.txt
```

creating dictionary

[241]:
```
with open(filename) as fh:
    for line in fh:
        command, description = line.strip().split(None, 1)
        dict1[command] = description.strip()
```

reads each line and trims of extra spaces and gives only valid words creating json file named test1.json

[242]:
```
#out_file = open("test1.json","w")
out_file = open("eff_runs.json","w")
json.dump(dict1, out_file,indent=4,sort_keys = False)
out_file.close()
```

A json file exits in the directory named test1.json. Note the line of code: command, description = line.strip().split(None, 1). Here split(None, 1) is used to trim off all excess spaces between a key-value pair and '1' denotes split only once in a line. This ensures in a key-value pair, the

spaces in the value are not removed and those words are not split. Only the key is separated from its value.

''' **Now How DO I Make a JSON FILE with my own namings** '''

Now How DO I Make a JSON FILE with my own namings

Let us take real example of my efficiency in table form, where the first row is the the defination of efficiency, so, I need to change it to json file system, with under the first column

[243]:
```
import numpy as np
import json
file_eff = open('/home/hdbhatt/Ana/txt_to_json/eff.txt','r')
read_line = file_eff.readlines()
data_eff =np.loadtxt('/home/hdbhatt/Ana/txt_to_json/eff.txt',skiprows=1)
#data_eff # This is array (npdataarray i.e numpy.ndarray), so data_eff.head()␣
 ↪does not work,
# so use data_eff[:5], to print the first 5 rows
```

[244]:
```
data_eff.shape # it will give row vs column # dont do shape()
```

[244]:
```
(514, 11)
```

[245]:
```
#data_eff #print txtfile
```

[246]:
```
#data_eff[2:]#print all
```

[247]:
```
#data_eff[:2] #only 2
```

[248]:
```
#make json object to save numbers in txt
json_file = {}
```

[249]:
```
#build the json file
for n in range(len(data_eff[:,0])):
    json_file[str(int(data_eff[n,0]))] = {"charge":data_eff[n,3],"TESHMS":
 ↪data_eff[n,4],"TEHMS":data_eff[n,5],"HCAL_EFF":data_eff[n,6],"PCAL_EFF":
 ↪data_eff[n,7],"PAERO_EFF":data_eff[n,8],"TLT":data_eff[n,9]}

#here The first column [n,0] i.e run number is the head of the json element,␣
 ↪which has to be string,
#and is a integer in txtfile,
```

[250]:
```
[n,0] #[row,column]
```

[250]:
```
[513, 0]
```

[251]:
```
#check number of runs (or rows)
#len(data_eff[:,0])# works good but not len(data_eff[:0])
len(data_eff[:])
```

[251]: 514

[252]:
```
#save the json file as
with open('eff_json.json','w') as outfile:
    json.dump(json_file,outfile,ensure_ascii=False,indent=4)
```

[253]:
```
ls
```

```
charge_compare.ipynb  eff_runs.json     json_file_all.json  test1.json
datat.txt             eff.txt           notes               trial.txt
data.txt              Hem_all_eff.json  runs_info.json      txt_to_json.ipynb
eff_json.json         Hem_all_eff.txt   runs_info.txt
```

[254]:
```
#json_file #prints the json file
```

Now I am converting whole eff.txt file into json file, earlier, I have only printed required one, excluding charge, rate, and product of eff. First, make a json file name, say json_file_all = {}, then from the textfile, grab the required columns from data_eff.txt file as[n,0] or [n,10] and so on

[255]:
```
#make json object to save whole json file
json_file_all = {}
```

[256]:
```
for n in range(len(data_eff[:,0])):
    json_file_all[str(int(data_eff[n,0]))] = {"rate":data_eff[n,1],"curr":
 ↪data_eff[n,2],"charge":data_eff[n,3],"TESHMS":data_eff[n,4],"TEHMS":
 ↪data_eff[n,5],"HCAL_EFF":data_eff[n,6],"PCAL_EFF":data_eff[n,7],"PAERO_EFF":
 ↪data_eff[n,8],"TLT":data_eff[n,9],"EFF_PROD":data_eff[n,10]}
```

- now dump the so created json file into the name json_file_all.json

[257]:
```
with open('json_file_all.json','w') as outfile:
    json.dump(json_file_all,outfile,ensure_ascii=False,indent=4)
```

[258]:
```
ls
```

```
charge_compare.ipynb  eff_runs.json     json_file_all.json  test1.json
datat.txt             eff.txt           notes               trial.txt
data.txt              Hem_all_eff.json  runs_info.json      txt_to_json.ipynb
eff_json.json         Hem_all_eff.txt   runs_info.txt
```

[262]:
```
#json_file_all #prints the json file here
```

[260]:
```
#ls -ltrh
```

You ARE DONE Converting TXT file to JSON FILE

***** Now compare two columns and print the key, value pairs i.e Run Num and the diff: from any two json files *****

[263]:
```
#read the first json file as
with open('eff_json.json') as first:
    data_first = json.load(first)
```

[267]:
```
#read the second json file as # but I am opening same json file
with open('eff_json.json') as second:
    data_second = json.load(second)
```

[328]:
```
#Let's compare TLT and "PAERO_EFF"

for key, value in data_first.items():
    RunNumber = key
    Total_LT = value["TLT"]

    if key in data_second:
        Aero_eff = data_second[str(RunNumber)]["PAERO_EFF"]
        if(abs(Total_LT-Aero_eff)>0.06):
            #print(RunNumber,"  "  "%0.4f"  % abs(Total_LT-Aero_eff))
            print(RunNumber, "     "  "%0.4f     %0.4f     %0.4f"     %␣
 ↪(Total_LT, Aero_eff, Total_LT-Aero_eff))
#space or "" or "\t" should not be defined in between strings :"%0.4f     %0.4f ␣
 ↪   %0.4f" :instead give the tab
```

```
6133      0.9979     0.9345     0.0635
6346      0.9903     0.9123     0.0780
6407      0.9900     0.9086     0.0815
6410      0.9820     0.9174     0.0646
6411      0.9837     0.9221     0.0616
6415      0.9837     0.9199     0.0638
6513      0.9950     0.9281     0.0670
6515      0.9828     0.9227     0.0601
7620      0.9971     0.9364     0.0607
7622      0.9964     0.9333     0.0632
7659      0.9941     0.9213     0.0728
7660      0.9910     0.9303     0.0607
```

This Printed only those Run Numbers and their values, where the difference bewteen the TLT and aero eff > 0.06

[ ]:

[ ]:

[ ]:

[ ]:

[ ]: