Python CSV to JSON

Here's my code, really simple stuff...

```
import csv
import json

csvfile = open('file.csv', 'r')
jsonfile = open('file.json', 'w')

fieldnames = ("FirstName","LastName","IDNumber","Message")
reader = csv.DictReader( csvfile, fieldnames)
out = json.dumps( [ row for row in reader ] )
jsonfile.write(out)
```

Declare some field names, the reader uses CSV to read the file, and the filed names to dump the file to a JSON format. Here's the problem...

Each record in the CSV file is on a different row. I want the JSON output to be the same way. The problem is it dumps it all on one giant, long line.

I've tried using something like `for line in csvfile:` and then running my code below that with `reader = csv.DictReader( line, fieldnames)` which loops through each line, but it does the entire file on one line, then loops through the entire file on another line... continues until it runs out of lines.

Any suggestions for correcting this?

Edit: To clarify, currently I have: (every record on line 1)

```
[{"FirstName":"John","LastName":"Doe","IDNumber":"123","Message":"None"},
{"FirstName":"George","LastName":"Washington","IDNumber":"001","Message":"Something"}]
```

What I'm looking for: (2 records on 2 lines)

```
{"FirstName":"John","LastName":"Doe","IDNumber":"123","Message":"None"}
{"FirstName":"George","LastName":"Washington","IDNumber":"001","Message":"Something"}
```

Not each individual field indented/on a separate line, but each record on it's own line.

Some sample input.

```
"John","Doe","001","Message1"
"George","Washington","002","Message2"
```

python json csv

edited Oct 31 '13 at 12:46

asked Oct 31 '13 at 3:15
BeanBagKing
510 1 6 19

possible duplicate of Dumping multiple variables to disk in Json. One variable per line – beroe Oct 31 '13 at 4:48

i'm not sure your code does *exactly* what you say; it should produce `[{..row..},{..row..},...]` not `{..row..}{..row..}..` . That is to say, the output looks like it will be a json array of json objects, not a stream of unconnected json objects. – SingleNegationElimination Oct 31 '13 at 12:42

6 Answers

The problem with your desired output is that it is not valid json document,; it's a *stream of json documents*!

That's okay, if its what you need, but that means that for each document you want in your output, you'll have to call `json.dumps`.

Since the newline you want separating your documents is not contained in those documents, you're on the hook for supplying it yourself. So we just need to pull the loop out of the call to json.dump and interpose newlines for each document written.

```
import csv
import json

csvfile = open('file.csv', 'r')
jsonfile = open('file.json', 'w')

fieldnames = ("FirstName","LastName","IDNumber","Message")
reader = csv.DictReader( csvfile, fieldnames)
for row in reader:
    json.dump(row, jsonfile)
    jsonfile.write('\n')
```

answered Oct 31 '13 at 12:49
SingleNegationElimination
79k 9 141 203

Perfect! Sorry you had to do a bit of mind reading to get it, and thanks for the corrections/clarifications. This is exactly what I was looking for. – BeanBagKing Oct 31 '13 at 13:03

but problem is outfile is not a valid json – MONTYHS Mar 7 '14 at 14:48

@MONTYHS: The first sentance of this answer explains that outfile is not a json document; and what it is instead. Are you having a different problem from the person who asked this question? – SingleNegationElimination Mar 7 '14 at 16:44

You can try this

```
import csvmapper

# how does the object look
mapper = csvmapper.DictMapper([
  [
     { 'name' : 'FirstName'},
     { 'name' : 'LastName' },
     { 'name' : 'IDNumber', 'type':'int' },
     { 'name' : 'Messages' }
  ]
 ])

# parser instance
parser = csvmapper.CSVParser('sample.csv', mapper)
# conversion service
converter = csvmapper.JSONConverter(parser)

print converter.doConvert(pretty=True)
```

answered Feb 8 '15 at 15:20
Snork S
68 7

Add the `indent` parameter to `json.dumps`

```
 data = {'this': ['has', 'some', 'things'],
         'in': {'it': 'with', 'some': 'more'}}
 print(json.dumps(data, indent=4))
```

Also note that, you can simply use `json.dump` with the open `jsonfile`:

```
json.dump(data, jsonfile)
```

edited Oct 31 '13 at 3:19
satoru
9,405 8 42 81

answered Oct 31 '13 at 3:17
Wayne Werner
15.8k 6 65 124

Not quite what I'm looking for. I edited my original question to clarify and show the desired output. Thank you for the tip though, this may come in handy later. – BeanBagKing Oct 31 '13 at 12:43

```
import csv
import json
csvfile = csv.DictReader('filename.csv', 'r'))
output =[]
for each in csvfile:
    row ={}
    row['FirstName'] = each['FirstName']
    row['LastName']  = each['LastName']
    row['IDNumber']  = each ['IDNumber']
    row['Message']   = each['Message']
    output.append(row)
json.dump(output,open('filename.json','w'),indent=4,sort_keys=False)
```

answered Oct 31 '13 at 12:03
MONTYHS
352 2 17

When I try to use this I get "KeyError: 'FirstName'". It doesn't seem like the key is being added. I'm not sure exactly what you're trying to do here, but I don't think the output matches what I'm looking for since you use the same indent=4 as Wayne. What output should I expect? I edited my original post to clarify what I'm looking for. – BeanBagKing Oct 31 '13 at 12:41

The key error is most likely because this code does not pass a headers argument to `DictReader` , so it's guessing the field names from the first line of the input file: John, Doe, 5, "None" instead of "FirstName, lastname," and so on... – SingleNegationElimination Oct 31 '13 at 12:45

Better option, this one actually parses the CSV for the desired fields (not just in order, as in the marked answer) – GarciadelCastillo Mar 5 '14 at 19:41

I get an error saying `TypeError: expected string or buffer` – CodyBugstein Jun 25 '15 at 15:41

As slight improvement to @MONTYHS answer, iterating through a tup of fieldnames:

```
import csv
import json

csvfilename = 'filename.csv'
jsonfilename = csvfilename.split('.')[0] + '.json'
csvfile = open(csvfilename, 'r')
jsonfile = open(jsonfilename, 'w')
reader = csv.DictReader(csvfile)

fieldnames = ('FirstName', 'LastName', 'IDNumber', 'Message')

output = []

for each in reader:
  row = {}
  for field in fieldnames:
    row[field] = each[field]
output.append(row)

json.dump(output, jsonfile, indent=2, sort_keys=True)
```

answered Mar 5 '14 at 19:43
GarciadelCastillo
197 12

I took @SingleNegationElimination's response and simplified it into a three-liner that can be used in a pipeline:

```
import csv
import json
import sys

for row in csv.DictReader(sys.stdin):
    json.dump(row, sys.stdout)
    sys.stdout.write('\n')
```

answered Nov 25 '15 at 22:25
Lawrence I. Siden
2,260 6 24 35