Jan. 18, 2013

CSV (/csv/) System & OS (/systems-programming/)

# Using the CSV module in Python

If you want to import or export spreadsheets and databases for use in the Python interpreter, you must rely on the CSV module, or Comma Separated Values format.

## What is a CSV File?

CSV files are used to store a large number of variables – or data. They are incredibly simplified spreadsheets – think Excel – only the content is stored in plaintext.

And the CSV module is a built-in function that allows Python to parse these types of files.

It's worth noting that when you work with a CSV file, you are dabbling in JSON development.

JSON – which stands for JavaScript Object Notation – is a format that is used to store information as JavaScript code in plaintext files. You don't need to know JavaScript to work with these files, nor is the practice confined to that language. Obviously, since we're working with Python here.

The text inside a CSV file is laid out in rows, and each of those has columns, all separated by commas. Every line in the file is a row in the spreadsheet, while the commas are used to define and separate cells.



## Working with the CSV Module

To pull information from CSV files you use loop and split methods to get the data from individual columns.

The CSV module explicitly exists to handle this task, making it much easier to deal with CSV formatted files. This becomes especially important when you are working with data that's been exported from actual spreadsheets and databases to text files. This information can be tough to read on its own.

Unfortunately, there is no standard so the CSV module uses "dialects" to support parsing using different parameters. Along with a generic reader and writer, the module includes a dialect for working with Microsoft Excel and related files.

## CSV Functions

The CSV module includes all the necessary functions built in. They are:

- csv.reader
- csv.writer
- csv.register_dialect
- csv.unregister_dialect
- csv.get_dialect
- csv.list_dialects
- csv.field_size_limit

In this guide we are only going to focus on the reader and writer functions which allow you to edit, modify, and manipulate the data stored in a CSV file.

## Reading CSV Files

To pull data from a CSV file, you must use the reader function to generate a reader object.

The reader function is designed to take each line of the file and make a list of all columns. Then, you just choose the column you want the variable data for.

It sounds a lot more complicated than it is. To prove it, let's take a look at an example.

```
import CSV
With open('some.csv', 'rb') as f:
reader = csv.reader(f)
for row in reader:
print row
```

Notice how the first command is used to import the CSV module?

Let's look at another example.

```
import csv
import sys

f = open(sys.argv[1], 'rb')
reader = csv.reader(f)
for row in reader
print row

f.close()
```

In the first two lines, we are importing the CSV and sys modules. Then, we open the CSV file we want to pull information from.

Next, we create the reader object, iterate the rows of the file, and then print them. Finally, we close out the operation.

## CSV Sample File

We're going to take a look at an example CSV file. Pay attention to how the information is stored and presented.

```
Title,Release Date,Director
And Now For Something Completely Different,1971,Ian MacNaughton
Monty Python And The Holy Grail,1975,Terry Gilliam and Terry Jones
Monty Python's Life Of Brian,1979,Terry Jones
Monty Python Live At The Hollywood Bowl,1982,Terry Hughes
Monty Python's The Meaning Of Life,1983,Terry Jones
```

## Reading CSV Files Example

We're going to start with a basic CSV file that has 3 columns, containing the variables "A", "B", "C", and "D".

```
$ cat test.csv
A,B,"C D"
1,2,"3 4"
5,6,7
```

Then, we'll use the following Python program to read and display the contents of the above CSV file.

```
import csv

ifile = open('test.csv', "rb")
reader = csv.reader(ifile)

rownum = 0
for row in reader:
# Save header row.
if rownum ==0:
header = row
else:
colnum = 0
for col in row:
print '%-8s: %s' % (header[colnum], col)
colnum + = 1

rownum + = 1

ifile.close()
```

When we execute this program in Python, the output will look like this:

```
$ python csv1.py
A       : 1
B       : 2
C D       : 3 4
A       : 5
B       : 6
C D       : 7
```

# Writing to CSV Files

When you have a set of data that you would like to store inside a CSV file, it's time to do the opposite and use the write function. Believe it or not, this is just as easy to accomplish as reading them.

The ***writer()*** function will create an object suitable for writing. To iterate the data over the rows, you will need to use the ***writerow()*** function.

Here's an example.

The following Python program converts a file called "test.csv" to a CSV file that uses tabs as a value separator with all values quoted. The delimiter character and the quote character, as well as how/when to quote, are specified when the writer is created. These same options are available when creating reader objects.

```
import csv

ifile  = open('test.csv', "rb")
reader = csv.reader(ifile)
ofile  = open('ttest.csv', "wb")
writer = csv.writer(ofile, delimiter='', quotechar='"', quoting=csv.QUOTE_ALL)

for row in reader:
    writer.writerow(row)

ifile.close()
ofile.close()
```

When you execute this program, the output will be:

```
$ python csv2.py
$ cat ttest.csv
"A"     "B"     "C D"
"1"     "2"     "3 4"
"5"     "6"     "7"
```

## Quoting CSV Files

With the CSV module, you can also perform a variety of quoting functions.

They are:

- ***csv.QUOTE_ALL*** - Quote everything, regardless of type.
- ***csv.QUOTE_MINIMAL*** - Quote fields with special characters
- ***csv.QUOTE_NONNUMERIC*** - Quote all fields that are not integers or floats
- ***csv.QUOTE_NONE*** - Do not quote anything on output

# More Python Reading and Resources

http://docs.python.org/2/library/csv.html (http://docs.python.org/2/library/csv.html)
http://www.doughellmann.com/PyMOTW/csv/ (http://www.doughellmann.com/PyMOTW/csv/)
http://effbot.org/librarybook/csv.htm (http://effbot.org/librarybook/csv.htm)
http://www.linuxjournal.com/content/handling-csv-files-python (http://www.linuxjournal.com/content/handling-csv-