Module of the Month: Python CSV module

Inland Empire Python Users Group
May 20, 2014

John Sheehan

What is “CSV”?

In the strictest sense, CSV is a text based data format where value are separated by commas - in other words: Comma Separated Values. Data values are generally comma delimited and data records are separated by line breaks.

```
John Smith,123 Main Street,951-555-1212,24
Jane Doe,99 Ownlywon Way,714-867-5309,32
```

What is “CSV”?

In the strictest sense, CSV is a text based data format where value are separated by commas - in other words: Comma Separated Values. Data values are generally comma delimited and data records are separated by line breaks.

```
John Smith,123 Main Street,951-555-1212,24
Jane Doe,99 Ownlywon Way,714-867-5309,32
```

In practice however, CSV has come to casually also encompass variations on this format:

```
"John Smith","123 Main Street","951-555-1212",24
Jane Doe|99 Ownlywon Way|714-867-5309|32
"Onik Chormanskivich"~"2 Crown Drive"~"909-747-2112"~"67"
```

What is “CSV”?

- Note that the CSV data format is generally treated as a variable length field format as opposed to a fixed length format (even if the data fields happen to be of fixed length). In other words, one of the key factors is that the data record format uses delimiters to identify where one field ends and the next one starts.

What is “CSV”?

- Note that the CSV data format is generally treated as a variable length field format as opposed to a fixed length format (even if the data fields happen to be of fixed length). In other words, one of the key factors is that the data record format uses delimiters to identify where one field ends and the next one starts.

- Usually you would be working with CSV files, however the format could also be applied to streaming formats or in-memory data structures as well.

What is “CSV”?

- CSV is a very common data interchange format.

What is “CSV”?

- CSV is a very common data interchange format.
- CSV is a record based format designed to hold “table” type data as opposed to nested (object) data.*

*Nested data can be “faked” with additional program logic

Python CSV Module

- Allows you to easily read and create CSV formatted data

Python CSV Module

- Allows you to easily read and create CSV formatted data
- Is one of Python's "included batteries"

Python CSV Module

- Allows you to easily read and create CSV formatted data
- Is one of Python's "included batteries"
- Simple, but does one thing well

Using the Python CSV module:

```
import csv
```

Using the Python CSV module:

```
import csv

f = open('~/sample.csv')
try:
    reader = csv.reader(f)
    for row in reader:
        print row
finally:
    f.close()
```

Using the Python CSV module:

- The csv.reader method returns an iterator where every call to next() returns the next record in the data set.

Using the Python CSV module:

- The csv.reader method returns an iterator where every call to next() returns the next record in the data set.

- The csv.reader method can also take additional parameters that specify a particular CSV format dialect or information about specific nuances of the format (i.e. what delimiter is being used or what character is being used for quoting).

Using the Python CSV module:

- The csv.reader method returns an iterator where every call to next() returns the next record in the data set.
- The csv.reader method can also take additional parameters that specify a particular CSV format dialect or information about specific nuances of the format (i.e. what delimiter is being used or what character is being used for quoting).
- Each CSV record is returned as a Python list.

```
['John Smith', '123 Main Street', '951-555-1212', '24']
```

Writing with the Python CSV module:

```
import csv

f_in = open('~/sample.csv')
f_out = open('~/output.csv', 'w')
try:
    reader = csv.reader(f_in)
    writer = csv.writer(f_out, delimiter='|', quotechar='$', quoting=csv.QUOTE_NONNUMERIC)
    for row in reader:
        writer.writerow( (row[0], int(row[3])) )
finally:
    f_out.close()
    f_in.close()

print open('~/output.csv').read()
```

CSV Conventions:

Delimiting (Most commonly used delimiters)

- Comma (,)
- Pipe (|)
- Tab ( )
- Tilde (~)

CSV Conventions:

Quoting (Usually " is used)

- csv.QUOTE_ALL
  Quote ALL fields

- csv.QUOTE_MINIMAL
  Quote only fields with special characters

- csv.QUOTE_NONNUMERIC
  Quote all non-numeric fields.

- csv.QUOTE_NONE
  Don’t quote anything

Errata

- Microsoft Excel and other spreadsheets can read the basic CSV format natively.

Errata

- Microsoft Excel and other spreadsheets can read the basic CSV format natively.
- By reading in a CSV file and then writing to a new CSV file, the Python CSV module can be used to easily convert from one CSV dialect to another.

Errata

- Microsoft Excel and other spreadsheets can read the basic CSV format natively.
- By reading in a CSV file and then writing to a new CSV file, the Python CSV module can be used to easily convert from one CSV dialect to another.
- No (generally used) formal standard exists for the CSV format.

Errata

- Microsoft Excel and other spreadsheets can read the basic CSV format natively.
- By reading in a CSV file and then writing to a new CSV file, the Python CSV module can be used to easily convert from one CSV dialect to another.
- No (generally used) formal standard exists for the CSV format.
- Fun fact: The CSV format was defined as a method of list-directed I/O for Fortran 77

Errata

- Microsoft Excel and other spreadsheets can read the basic CSV format natively.
- By reading in a CSV file and then writing to a new CSV file, the Python CSV module can be used to easily convert from one CSV dialect to another.
- No (generally used) formal standard exists for the CSV format.
- Fun fact: The CSV format was defined as a method of list-directed I/O for Fortran 77

Python docs:

https://docs.python.org/2/library/csv.html

Discussion