Working with CSV Files

Welcome back! Let's discuss how to work with CSV files in Python. A file with the CSV file extension is a Comma Separated Values file. All CSV files are plain text, contain alphanumeric characters, and structure the data contained within them in a tabular form. Don't confuse Excel Files with csv files, while csv files are formatted very similarly to excel files, they don't have data types for their values, they are all strings with no font or color. They also don't have worksheets the way an excel file does. Python does have several libraries for working with Excel files, you can check them out here and here.

Files in the CSV format are generally used to exchange data, usually when there's a large amount, between different applications. Database programs, analytical software, and other applications that store massive amounts of information (like contacts and customer data), will usually support the CSV format.

Let's explore how we can open a csv file with Python's built-in csv library.

Notebook Location.

Run **pwd** inside a notebook cell to find out where your notebook is located

In [1]:
```
pwd
```

Out[1]:
```
'C:\\Users\\Marcial\\Pierian-Data-Courses\\Complete-Python-3-Bootcamp\\15-PDFs-and-Spreadsheets'
```

Reading CSV Files

In [2]:
```
import csv
```

When passing in the file path, make sure to include the extension if it has one, you should be able to Tab Autocomplete the file name. If you can't Tab autocomplete, that is a good indicator your file is not in the same location as your notebook. You can always type in the entire file path (it will look similar in formatting to the output of **pwd**.

In [3]:
```
data = open('example.csv')
```

In [4]:
```
data
```

Out[4]:
```
<_io.TextIOWrapper name='example.csv' mode='r' encoding='cp1252'>
```

Encoding

Often csv files may contain characters that you can't interpret with standard python, this could be something like an @ symbol, or even foreign characters. Let's view an example of this sort of error (its pretty common, so its important to go over).

In [5]:
```
csv_data = csv.reader(data)
```

Cast to a list will give an error, note the **can't decode** line in the error, this is a giveaway that we have an encoding problem!

In [6]:
```
data_lines = list(csv_data)
```

```
---------------------------------------------------------------------------
UnicodeDecodeError                        Traceback (most recent call last)
<ipython-input-6-1f501450909b> in <module>
----> 1 data_lines = list(csv_data)

C:\ProgramData\Anaconda3\lib\encodings\cp1252.py in decode(self, input, final)
     21 class IncrementalDecoder(codecs.IncrementalDecoder):
     22     def decode(self, input, final=False):
---> 23         return codecs.charmap_decode(input,self.errors,decoding_table)[0]
     24
     25 class StreamWriter(Codec,codecs.StreamWriter):

UnicodeDecodeError: 'charmap' codec can't decode byte 0x8d in position 1835: character maps t
o <undefined>
```

Let's not try reading it with a "utf-8" encoding.

In [7]:
```
data = open('example.csv',encoding="utf-8")
csv_data = csv.reader(data)
data_lines = list(csv_data)
```

In [8]:
```
# Looks like it worked!
data_lines[:3]
```

Out[8]:
```
[['id', 'first_name', 'last_name', 'email', 'gender', 'ip_address', 'city'],
 ['1',
  'Joseph',
  'Zaniolini',
  'jzaniolini0@simplemachines.org',
  'Male',
  '163.168.68.132',
  'Pedro Leopoldo'],
 ['2',
  'Freida',
  'Drillingcourt',
  'fdrillingcourt1@umich.edu',
  'Female',
  '97.212.102.79',
  'Buri']]
```

Note the first item in the list is the header line, this contains the information about what each column represents. Let's format our printing just a bit:

In [9]:
```
for line in data_lines[:5]:
```

```
    print(line)
```

```
['id', 'first_name', 'last_name', 'email', 'gender', 'ip_address', 'city']
['1', 'Joseph', 'Zaniolini', 'jzaniolini0@simplemachines.org', 'Male', '163.168.68.132', 'Pedro Leopoldo']
['2', 'Freida', 'Drillingcourt', 'fdrillingcourt1@umich.edu', 'Female', '97.212.102.79', 'Buri']
['3', 'Nanni', 'Herity', 'nherity2@statcounter.com', 'Female', '145.151.178.98', 'Claver']
['4', 'Orazio', 'Frayling', 'ofrayling3@economist.com', 'Male', '25.199.143.143', 'Kungur']
```

Let's imagine we wanted a list of all the emails. For demonstration, since there are 1000 items plus the header, we will only do a few rows.

In [10]:
```
len(data_lines)
```

Out[10]:
```
1001
```

In [11]:
```
all_emails = []
for line in data_lines[1:15]:
    all_emails.append(line[3])
```

In [12]:
```
print(all_emails)
```

```
['jzaniolini0@simplemachines.org', 'fdrillingcourt1@umich.edu', 'nherity2@statcounter.com', 'ofrayling3@economist.com', 'jmurrison4@cbslocal.com', 'lgamet5@list-manage.com', 'dhowatt6@amazon.com', 'kherion7@amazon.com', 'chedworth8@china.com.cn', 'hgasquoine9@google.ru', 'ftarra@shareasale.com', 'abathb@umn.edu', 'lchastangc@goo.gl', 'cceried@yale.edu']
```

What if we wanted a list of full names?

In [13]:
```
full_names = []

for line in data_lines[1:15]:
    full_names.append(line[1]+' '+line[2])
```

In [14]:
```
full_names
```

Out[14]:
```
['Joseph Zaniolini',
 'Freida Drillingcourt',
 'Nanni Herity',
 'Orazio Frayling',
 'Julianne Murrison',
 'Lucy Gamet',
 'Dyana Howatt',
 'Kassey Herion',
 'Chrissy Hedworth',
 'Hyatt Gasquoine',
 'Felicdad Tarr',
 'Andrew Bath',
 'Lucais Chastang',
 'Car Cerie']
```

Writing to CSV Files

We can also write csv files, either new ones or add on to existing ones.

New File

This will also overwrite any exisiting file with the same name, so be careful with this!

In [15]:
```
# newline controls how universal newlines works (it only applies to text
# mode). It can be None, '', '\n', '\r', and '\r\n'.
file_to_output = open('to_save_file.csv','w',newline='')
```

In [16]:
```
csv_writer = csv.writer(file_to_output,delimiter=',')
```

In [17]:
```
csv_writer.writerow(['a','b','c'])
```

Out[17]:
```
7
```

In [18]:
```
csv_writer.writerows([['1','2','3'],['4','5','6']])
```

In [19]:
```
file_to_output.close()
```

Existing File

In [20]:
```
f = open('to_save_file.csv','a',newline='')
```

In [21]:
```
csv_writer = csv.writer(f)
```

In [22]:
```
csv_writer.writerow(['new','new','new'])
```

Out[22]:
```
13
```

In [23]:
```
f.close()
```

That is all for the basics! If you believe you will be working with CSV files often, you may want to check out the powerful pandas library.