Utilisation du module csv dans Python

Langages et programmation

Révisions de 1ère

En utilisant ce module avec Python, nous pouvons facilement lire et écrire dans des fichiers CSV. Chaque ligne du fichier csv est une ligne d'un tableau.[1]
Afin d'extraire les données d'un fichier CSV, nous devons parcourir les lignes à l'aide d'une boucle et utiliser des méthodes de fractionnement pour extraire les données.
Les fichiers CSV sont utilisés par des programmes qui gèrent de grandes quantités de données. Ils servent à exporter les données à partir de feuilles de calcul et de bases de données, pour les utiliser dans d'autres programmes.

I. Utiliser le module CSV pour lire un ficher CSV

Pour lire à un fichier csv, nous devons **créer un objet** avec la méthode reader().
Nous devons appeler la méthode csv.reader() sur un objet fichier déjà ouvert, en utilisant la fonction open() en mode lecture 'r'. Votre fichier csv doit être dans le même répertoire que l'interpréteur Python.

```
import csv

# Ouvrir le fichier csv
with open('fichier.csv', 'r') as f:
    # Créer un objet csv à partir du fichier
    obj = csv.reader(f)
```

Ensuite, nous allons itérer les lignes de notre objet fichier, où, chaque ligne sera une liste Python. Par conséquent, nous utiliseront la boucle for pour afficher les lignes du fichier.

```
import csv

# Ouvrir le fichier csv
with open('fichier.csv', 'r') as f:
    # Créer un objet csv à partir du fichier
    obj = csv.reader(f)

    for ligne in obj:
        print(ligne)
```

Notez que le bloc de la boucle for est à l'intérieur de l'instruction with, sinon vous aurez une erreur, puisque à l'extérieur de with le fichier est fermé.

Nous allons créer un fichier fichier.csv contenant des champs séparés par une virgule.

Langage, Créateur, Année, Extension
Python, Guido van Rossum, 1991, .py
Java, James Gosling, 1995, .java
C++, Bjarne Stroustrup,1983,.cpp

Si nous appliquons le code précédent sur le fichier qu'on a créé, nous aurons le résultat suivant :

```
['Langage', ' Créateur', ' Année', ' Extension']
['Python', ' Guido van Rossum', ' 1991', ' .py']
['Java', ' James Gosling', ' 1995', ' .java']
['C++', ' Bjarne Stroustrup', '1983', '.cpp']
```

Dans cet exemple, nous avons ouvert fichier.csv en mode lecture à l'aide de open(). Puis nous avons lu sont contenu csv.reader(f).
Puis on a utilisé la boucle for pour itérer sur les lignes du fichier.

II. Écrire dans un fichier en utilisant la méthode *csv.writer()*

On utilise csv.writer() pour écrire les données dans un fichier. De plus on peut également l'utiliser pour créer un nouveau fichier et l'ouvrir en mode écriture w.

[1] Issu de https://pythonforge.com/module-csv-lire-ecrire/

```
import csv

print("Un programme qui utilise csv.writer() pour écrire dans un fichier")
print("\n")
# Les données que nous allons écrire
data =[('Nom','Prénom','Age'),('Dubois','Marie', 40),('Bernard', 'Sarah',
32),('Thomas','Henri', 38)]
# Ouvrir le fichier en mode écriture
fichier = open('noms.csv','w')
print("Nous avons ouvert le fichier noms, s'il n'existe pas il sera créé ")
# Créer l'objet fichier
obj = csv.writer(fichier)
# Chaque élément de data correspond à une ligne
for element in data:
    obj.writerow(element)
fichier.close()
```

Dans ce code, nous avons ouvert le fichier noms.csv en mode écriture `w`, qui crée le fichier s'il n'existe pas. Notez que CSV utilise Excel par défaut pour afficher le contenu.

III. Le module CSV et la classe *DictReader()*

Nous allons à présent utiliser `csv.DictReader()` pour lire un fichier CSV sous forme d'un dictionnaire.

Nous allons utiliser le fichier noms.csv qu'on a créé dans l'exemple précédent.

```
import csv

with open("noms.csv", 'r') as f:
    obj = csv.DictReader(f)

    for i in obj:
        print(i)
```

L'exécution du code.

```
{'Nom': 'Dubois', 'Prénom': 'Marie', 'Age': '40'}
{'Nom': 'Bernard', 'Prénom': 'Sarah', 'Age': '32'}
{'Nom': 'Thomas', 'Prénom': 'Henri', 'Age': '38'}
```

Si vous utilisez une version Python inférieur à 3.8, vous devez utiliser `print(dict(i))` au lieu de `print(i)` dans la boucle `for`.

IV. Écrire dans un fichier avec la classe *DictWriter()*

Il est également possible d'utiliser la classe `DictWriter()` pour écrire dans un fichier CSV à partir d'un dictionnaire Python.

```
import csv

print("Un programme qui utilise csv.DictWriter() pour écrire dans un fichier")
print("\n")
nom_colonnes =['Nom','Prénom','Age']
fichier = open('personnel.csv','w')
with fichier:
    obj = csv.DictWriter(fichier, fieldnames=nom_colonnes)
    obj.writeheader()
    obj.writerow({'Nom': 'Dubois','Prénom': 'Marie','Age': 40})
    obj.writerow({'Nom': 'Bernard','Prénom': 'Sarah','Age': 32})
    obj.writerow({'Nom': 'Thomas','Prénom': 'Henri','Age': 38})
```

Nous avons créé un nouveau fichier personnel.csv en mode écriture `w` en utilisant la fonction `open()`. En fait, nous avons utilisé `DictWriter()` pour écrire les données dans le fichier en spécifiant les noms de colonnes `fieldnames` comme étant les noms des champs qui correspondent aux clés du dictionnaire créé.
Ouvrir le fichier personnel.csv pour voir à quoi il ressemble.