Définition de CSV

CSV signifie "Comma Separated Values" (Valeurs Séparées par des Virgules). Il s'agit d'un format de fichier couramment utilisé pour stocker des données tabulaires, telles que des feuilles de calcul.

Il existe 2 manières principales de travailler avec grâce au `module csv`. Soit avec `reader` et `writer` qui ressemble aux fichiers basiques, soit avec `DictReader` et `DictWriter`, mais on a obligatoirement besoin d'une en-tête.

L'utilisation de `DictWriter` et `DictReader` simplifie la manipulation de fichiers CSV en vous permettant d'accéder aux données à l'aide de noms de colonnes au lieu d'indices. Les noms de colonnes sont calqués sur la première ligne du csv qui représente l'en-tête. Ceci a pour but d'améliorer la lisibilité et la maintenabilité de votre code.

Comment ouvrir un fichier CSV

Pour lire un fichier CSV en Python, il faut tout d'abord que vous ouvriez le fichier en mode de lecture.

> avec Reader

```
import csv

with open('donnees.csv', mode='r') as fichier_csv:
    lecteur_csv = csv.reader(fichier_csv)
```

> avec DictReader

```
import csv

with open('donnees.csv', mode='r') as fichier_csv:
    lecteur_dict = csv.DictReader(fichier_csv)
```

Comment lire les données d'un fichier CSV

Après avoir ouvert le fichier CSV, vous pouvez lire les données en utilisant la boucle `for`.

> avec reader

```
import csv
```

```
with open('donnees.csv', mode='r') as fichier_csv:
    lecteur_csv = csv.reader(fichier_csv)

    next(lecteur_csv) # skip une ligne, l'en-tête

    for ligne in lecteur_csv:
        nom = ligne[0]
        age = ligne[1]
        print(f"Nom : {nom}, Age : {age}")
```

> avec DictReader

```
import csv

with open('donnees.csv', mode='r') as fichier_csv:
    lecteur_dict = csv.DictReader(fichier_csv)

    # la première ligne est utilisé comme en-tête

    for ligne in lecteur_dict:
        nom = ligne['Nom']
        age = ligne['Age']
        print(f"Nom : {nom}, Age : {age}")
```

Comment écrire des données dans un fichier CSV

Après avoir ouvert le fichier CSV en mode d'écriture, vous pouvez écrire des données dans le fichier.

Si le fichier dans lequel vous essayez d'écrire n'existe pas, alors il sera créé à la volet.

> avec writer

```
import csv

with open('nouveau_fichier.csv', mode='w', newline='') as fichier_csv:
    writer_csv = csv.writer(fichier_csv)

    writer_csv.writerow(['Nom', 'Age'])
    writer_csv.writerow(['Alice', 25])
    writer_csv.writerow(['Bob', 30])
```

> avec DictWriter

```

```

```python
import csv

with open('nouveau_fichier_dict.csv', mode='w', newline='') as fichier_csv:
    writer_csv_dict = csv.DictWriter(fichier_csv, fieldnames=['Nom', 'Age'])

    writer_csv_dict.writeheader()  # Écrit l'en-tête avec les noms des colonnes
    writer_csv_dict.writerow({'Nom': 'Alice', 'Age': 25})
    writer_csv_dict.writerow({'Nom': 'Bob', 'Age': 30})
```

Bibliothèques CSV populaires

Outre la bibliothèque `csv` de base, Python offre des bibliothèques populaires pour travailler avec des fichiers CSV, telles que `pandas` et `openpyxl`. Ces bibliothèques offrent des fonctionnalités avancées pour la manipulation de données tabulaires.

- **Pandas (Optionnel)** : Pandas est une bibliothèque de manipulation de données très puissante. Elle permet de lire et d'écrire des fichiers CSV, ainsi que de réaliser des opérations avancées sur les données.
- **Openpyxl (Optionnel)** : Openpyxl est une bibliothèque pour lire et écrire des fichiers Excel, mais elle peut également être utilisée pour lire et écrire des fichiers CSV.