Comprendre les Paths en Python

1. Introduction aux Paths en Python

Un **path** (ou chemin) est l'adresse d'un fichier ou d'un dossier dans le système de fichiers. Python offre plusieurs façons de manipuler les chemins, notamment avec le module `os` et le module moderne `pathlib`.

2. Types de Paths

2.1 Paths Absolus vs Relatifs

- **Path absolu** : Fournit l'emplacement complet d'un fichier ou dossier.
  - Exemple : `/home/user/documents/file.txt` (Linux/macOS) ou `C:\Users\User\Documents\file.txt` (Windows)
- **Path relatif** : Spécifie un chemin à partir du répertoire courant.
  - Exemple : `documents/file.txt`

2.2 Utilisation de `os.path`

Le module `os` permet de manipuler les chemins de manière portable.

```
import os

# Obtenir le chemin absolu d'un fichier
file_path = os.path.abspath("file.txt")
print(file_path)

# Vérifier si un fichier existe
exists = os.path.exists("file.txt")
print("Existe ?", exists)

# Récupérer le répertoire d'un fichier
directory = os.path.dirname(file_path)
print("Répertoire :", directory)
```

3. Utilisation de `pathlib` (Recommandé)

Le module `pathlib` offre une approche orientée objet plus intuitive.

3.1 Création et Manipulation de Paths

```
from pathlib import Path

# Définir un path
path = Path("documents/file.txt")
```

```
# Obtenir le chemin absolu
print(path.resolve())

# Vérifier l'existence du fichier
print(path.exists())

# Obtenir le dossier parent
print(path.parent)
```

3.2 Lecture d'un Fichier avec `pathlib`

```
from pathlib import Path

file_path = Path("example.txt")

# Lire le contenu du fichier ligne par ligne
if file_path.exists():
    with file_path.open(mode='r', encoding='utf-8') as file:
        for line in file:
            print(line.strip())  # Affiche chaque ligne sans les sauts de ligne
```

4. Conclusion

- Utilisez `pathlib` pour une gestion moderne et intuitive des chemins.
- `os.path` reste utile pour la compatibilité avec d'anciens scripts.
- Vérifiez toujours l'existence d'un fichier avant de tenter de le lire.

Python simplifie la manipulation des fichiers et des chemins, rendant l'accès aux données flexible et robuste ! 🚀