Cours Python Système : Gestion des fichiers et dossiers + Perspectives avancées

Date : 05-05-2025

1. Structure du projet

```
Système-Project/
│
├── system/
│   ├── __init__.py
│   ├── file_operations.py
│   └── main.py
│
├── data/
│
├── main.py
└── README.md
```

2. Contenu complet des fichiers

`system/file_operations.py`

```
import os
import shutil

def creer_dossier():
    nom = input("Nom du dossier à créer : ").strip()
    if not os.path.exists(nom):
        os.makedirs(nom)
        print(f"✅ Le dossier '{nom}' a été créé avec succès.")
    else:
        print(f"⚠️ Le dossier '{nom}' existe déjà.")
```

```
def lister_fichiers():
    dossier = input("Entrez le nom du dossier pour lister les fichiers : ").strip()
    if os.path.exists(dossier):
        fichiers = os.listdir(dossier)
        if fichiers:
            print(f"📂 Contenu du dossier '{dossier}':")
            for f in fichiers:
                print(f"- {f}")
        else:
            print("📂 Le dossier est vide.")
    else:
        print(f"⚠️ Le dossier '{dossier}' n'existe pas.")

def supprimer_fichier():
    fichier = input("Entrez le nom du fichier à supprimer : ").strip()
    if os.path.exists(fichier):
        os.remove(fichier)
        print(f"🗑️ Le fichier '{fichier}' a été supprimé.")
    else:
        print(f"⚠️ Le fichier '{fichier}' n'existe pas.")
```

```python
def supprimer_dossier():
    dossier = input("Entrez le nom du dossier à supprimer : ").strip()
    if os.path.exists(dossier):
        shutil.rmtree(dossier)
        print(f"🗑️ Le dossier '{dossier}' et son contenu ont été supprimés.")
    else:
        print(f"⚠️ Le dossier '{dossier}' n'existe pas.")

def copier_fichier():
    source = input("Entrez le chemin du fichier source à copier : ").strip()
    destination = input("Entrez le chemin de destination : ").strip()
    if os.path.exists(source):
        shutil.copy(source, destination)
        print(f"📂 Fichier copié de '{source}' à '{destination}'.")
    else:
        print(f"⚠️ Fichier source '{source}' n'existe pas.")

def deplacer_fichier():
    source = input("Entrez le chemin du fichier à déplacer : ").strip()
    destination = input("Entrez le chemin de destination : ").strip()
    if os.path.exists(source):
        shutil.move(source, destination)
        print(f"📂 Fichier déplacé de '{source}' à '{destination}'.")
    else:
        print(f"⚠️ Fichier source '{source}' n'existe pas.")
```

`system/main.py`

```python
from .file_operations import *

def menu():
    while True:
        print("\n📁 Menu de Gestion des Fichiers et Dossiers")
        print("1. Créer un dossier")
        print("2. Lister les fichiers dans un dossier")
        print("3. Supprimer un fichier")
        print("4. Supprimer un dossier")
        print("5. Copier un fichier")
        print("6. Déplacer un fichier")
        print("7. Quitter")

        choix = input("Choisissez une option (1-7): ")

        if choix == "1":
            creer_dossier()
        elif choix == "2":
            lister_fichiers()
        elif choix == "3":
            supprimer_fichier()
        elif choix == "4":
            supprimer_dossier()
        elif choix == "5":
            copier_fichier()
        elif choix == "6":
            deplacer_fichier()
        elif choix == "7":
            print("👋 À la prochaine!")
            break
        else:
            print("❌ Option invalide, réessayez.")
```

main.py

```
from system.main import menu

if __name__ == "__main__":
    menu()
```

README.md

```
# Gestionnaire de Fichiers et Dossiers en Python

Ce projet permet de gérer facilement la création, la suppression, la copie, et le déplacement de fichiers et dossiers via un menu interactif en Python.

## Fonctionnalités :
- Créer un nouveau dossier
- Lister les fichiers d'un dossier
- Supprimer un fichier ou un dossier
- Copier ou déplacer un fichier

## Usage :
1. Cloner ou télécharger le projet ou suivre les instructions dans ce support
2. Ouvrir un terminal dans le dossier
3. Lancer : `python main.py`
4. Suivez le menu
```

Futures explorations (recherches à faire) :

- `socket` : Communication réseau (serveurs/clients), applications distribuées.
- `ipaddress` : Manipulation avancée d'adresses IP, sous-réseaux, réseaux.
- `datetime` : Gestion des dates/horaires, calculs temporels, formats.
- `calendar` : Générer des calendriers, vérifier des jours spéciaux, gérer des événements.
- **Autres notions avancées:** Multithreading, gestion d'erreurs, interfaces graphiques(Tkinter, PyQt, Kivy, Flet, ...).

Motivation :

"Explorer ces modules vous permettra de créer des applications complètes et robustes."
"Apprendre ces notions vous donne la maîtrise pour des projets complexes."
"Ne cessez jamais de pousser vos limites, l'avenir appartient à ceux qui innovent."

3. Conseils pour aller plus loin

- **Recherchez** comment utiliser chaque module : leur documentation officielle est une excellente ressource.
- **Testez** en créant de petits scripts pour manipuler IP, dates, ou communiquer en réseau.
- **Combinez** ces modules pour réaliser des projets avancés comme un calendrier interactif, un détecteur d'adresse IP, ou un serveur simple.