Chapitre 2 : Programmation et automatisation

Ce chapitre présente les techniques permettant d'automatiser des tâches avec Python : gestion de fichiers, manipulation de données, génération de rapports, et interaction avec des API ou des pages web.

1. Principes d'automatisation de tâches

1.1. Description

L'automatisation consiste à écrire des programmes qui effectuent automatiquement des tâches répétitives :

- copier ou trier des fichiers,
- nettoyer et analyser des données,
- envoyer des emails,
- générer des rapports périodiques.

Python est très adapté à cela grâce à ses bibliothèques puissantes.

Exemple

```
import shutil, os
# Copier tous les fichiers texte vers un dossier "backup"
if not os.path.exists("backup"):
    os.mkdir("backup")
for fichier in os.listdir("."):
    if fichier.endswith(".txt"):
        shutil.copy(fichier, f"backup/{fichier}")
print("Tous les fichiers texte ont été copiés dans backup/")
```

2. Bibliothèques Python pour l'automatisation

a. Pandas et NumPy

Servent à **analyser et transformer des données numériques** (tableaux, statistiques, etc.)

Exemple

```
import pandas as pd
import numpy as np

# Création d'un DataFrame
data = {"Produit": ["A", "B", "C"], "Prix": [10, 15, 20]}
df = pd.DataFrame(data)
```

```
# Calcul de la moyenne
print("Prix moyen :", np.mean(df["Prix"]))
```

b. Os et Shutil

Servent à **gérer les fichiers et dossiers** : création, suppression, déplacement.

Exemple

```
import os, shutil

# Créer un dossier
os.mkdir("Documents")

# Vérifier s'il existe
if os.path.exists("Documents"):
    print("Le dossier existe.")

# Déplacer un fichier
shutil.move("data.csv", "Documents/data.csv")
```

c. Openpyxl ou Pandas pour Excel/CSV

Ces bibliothèques permettent de **lire, écrire et modifier des fichiers Excel ou CSV**.

Exemple

```
import pandas as pd

# Lecture d'un fichier CSV
df = pd.read_csv("ventes.csv")
print(df.head())

# Sauvegarde dans un nouveau fichier Excel
df.to_excel("ventes_modifiées.xlsx", index=False)
```

3. Définitions et exemples d'automatisation

Exemple 1 : Envoi automatique d'un e-mail

```
import smtplib
from email.mime.text import MIMEText

msg = MIMEText("Bonjour, ceci est un mail automatique.")
msg["Subject"] = "Notification Python"
```

```
msg["From"] = "expediteur@gmail.com"
msg["To"] = "destinataire@gmail.com"

with smtplib.SMTP_SSL("smtp.gmail.com", 465) as smtp:
    smtp.login("expediteur@gmail.com", "motdepasse")
    smtp.send_message(msg)

print("E-mail envoyé avec succès !")
```

4. Manipulation de fichiers avec Python

a. Parcourir un dossier

```
import os
for fichier in os.listdir("."):
    print(fichier)
```

b. Vérifier l'existence d'un fichier ou dossier

```
if os.path.exists("rapport.pdf"):
    print("Le fichier existe.")
```

c. Créer ou supprimer un dossier

```
os.mkdir("Nouveau_Dossier")
os.rmdir("Nouveau_Dossier")
```

5. Visualisation de données

Bibliothèques : Matplotlib, Seaborn, Plotly

Elles permettent de tracer des graphiques pour analyser les données.

Exemple

```
import pandas as pd
import matplotlib.pyplot as plt

df = pd.DataFrame({"Mois": ["Jan", "Fév", "Mar"], "Ventes": [120, 150, 100]})
plt.plot(df["Mois"], df["Ventes"], marker="o")
plt.title("Évolution des ventes")
plt.show()
```

6. Requests – Interaction avec une API

Description

requests permet de **récupérer ou envoyer des données sur Internet**, par exemple vers une API météo ou financière.

Exemple

```
import requests

url = "https://api.github.com"
response = requests.get(url)
print("Statut :", response.status_code)
print("Contenu :", response.json())
```

7. Beautiful Soup – Web scraping

Description

Permet d'extraire des informations d'une page web (titres, prix, liens…).

Exemple

```
import requests
from bs4 import BeautifulSoup

page = requests.get("https://news.ycombinator.com/")
soup = BeautifulSoup(page.text, "html.parser")

titres = [a.text for a in soup.select(".titleline a")]
print("Titres trouvés :", titres[:5])
```

8. Tkinter / PyQt – Interfaces graphiques

Description

Permettent de créer des **interfaces utilisateur (GUI)** pour interagir avec les programmes Python.

Exemple

```
import tkinter as tk

fenetre = tk.Tk()
fenetre.title("Ma première fenêtre")

label = tk.Label(fenetre, text="Bonjour Python !")
label.pack()

fenetre.mainloop()
```

9. Copier, déplacer, supprimer des fichiers (shutil)

Exemple

```
import shutil

# Copier un fichier
shutil.copy("data.txt", "backup/data.txt")

# Supprimer un fichier
os.remove("data.txt")
```

10. Sérialisation et désérialisation (Pickle)

Description

Sauvegarde d'objets Python dans des fichiers binaires pour les réutiliser plus tard.

Exemple

```
import pickle

# Sauvegarde (sérialisation)
d = {"nom": "Alice", "âge": 25}
with open("data.pkl", "wb") as f:
    pickle.dump(d, f)

# Lecture (désérialisation)
with open("data.pkl", "rb") as f:
    data = pickle.load(f)
print(data)
```

11. Traitement de fichiers volumineux (streaming)

Description

Le streaming permet de lire les fichiers **en morceaux** au lieu de tout charger en mémoire.

Exemple

```
with open("fichier_gros.txt", "r") as f:
    for ligne in f:
        print(ligne.strip())  # Traite chaque ligne sans tout charger
```

12. Mini-projet : Génération automatique d'un rapport

Objectif

Créer un script qui :

1. Lit un fichier Excel (ventes, notes, etc.)

2. Analyse les données
3. Génère un graphique
4. Enregistre un rapport PDF et Excel

Exemple résumé

```
import pandas as pd
import matplotlib.pyplot as plt
from reportlab.pdfgen import canvas

# Lecture des données
df = pd.read_excel("ventes.xlsx")

# Analyse
rapport = df.groupby("Produit")["Total"].sum()
rapport.plot(kind="bar", title="Ventes par produit")
plt.savefig("graphique.png")

# Génération PDF
pdf = canvas.Canvas("rapport.pdf")
pdf.drawString(100, 800, "Rapport automatique des ventes")
pdf.drawImage("graphique.png", 100, 500, width=400, height=250)
pdf.save()
```