Python Auto File Sorter

Moves files from the 'downloads' folder into extension-based subfolders

Purpose & Scope

Purpose: Scan the 'downloads' folder next to the script and auto-sort files.

Scope: Works with any extension (TXT, JPG, PDF, PNG, etc.).

Result: Creates one folder per extension (e.g., TXT/, JPG/, PDF/) and moves files.

How It Works (Flow)

1) Determine the script directory (base_folder).

2) Point to downloads/ (create if it doesn't exist).

3) List all files inside downloads/.

4) Get each file's extension (e.g., .jpg → JPG).

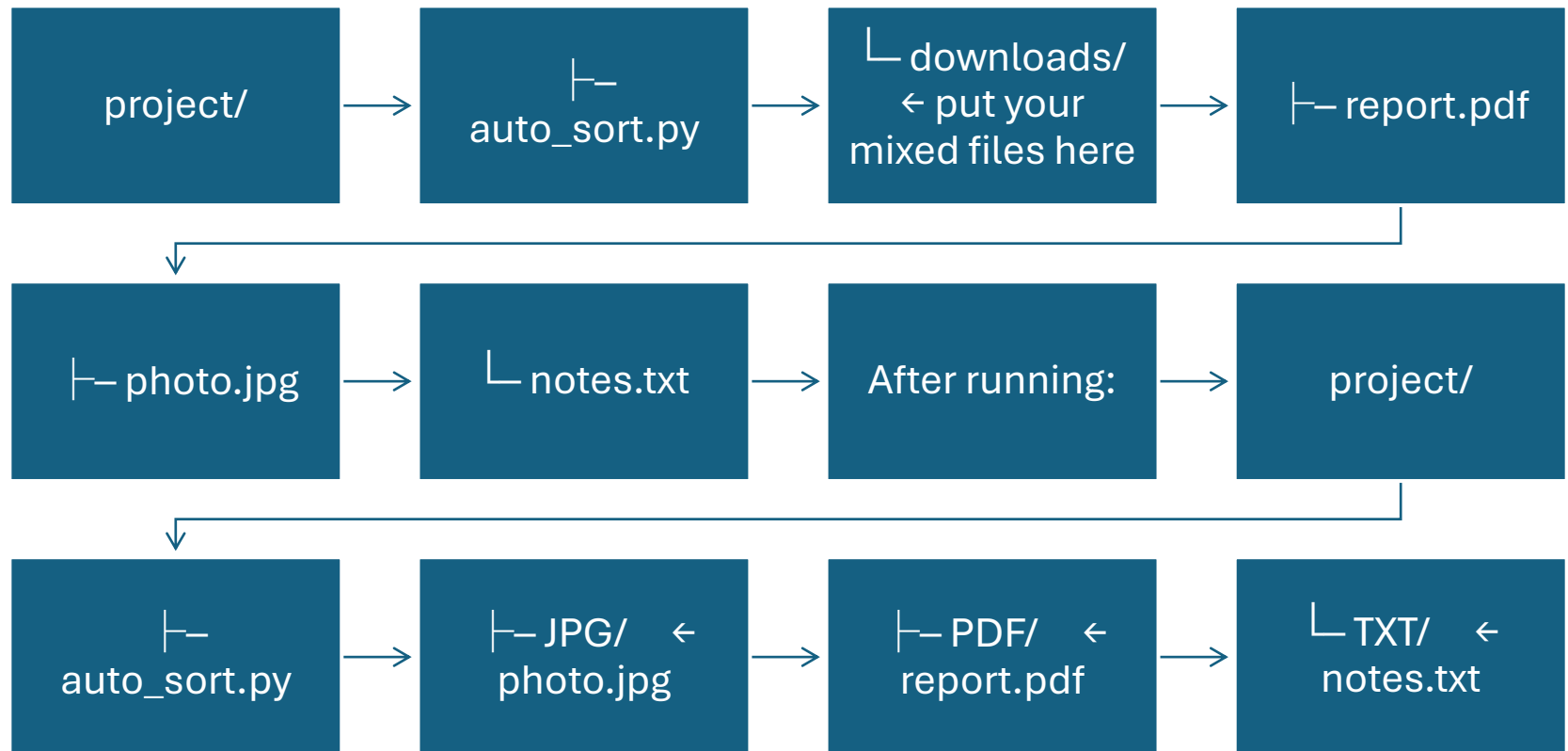5) Create the target folder if missing (JPG/, TXT/, etc.).

6) Move the file into the corresponding folder.

7) Print progress and final summary.

Key Features

- **Automatic:** No manual file selection required.

- **Safe defaults:** Creates folders when missing.

- **Minimal dependencies:** Uses only os and shutil (standard library).

- **Console feedback:** Prints each move and a final summary.

- **Extensible:** Can add dry-run, logging, conflict handling, etc.

Suggested Folder Layout

```
project/
├─ auto_sort.py
└─ downloads/        ← put your mixed files here
   ├─ report.pdf
   ├─ photo.jpg
   └─ notes.txt

After running:

project/
├─ auto_sort.py
├─ JPG/        ← photo.jpg
├─ PDF/        ← report.pdf
└─ TXT/        ← notes.txt
```

Code Highlights

- base_folder = os.path.dirname(os.path.abspath(__file__))
- source_folder = os.path.join(base_folder, 'downloads')
- os.makedirs(source_folder, exist_ok=True)
- for file_name in os.listdir(source_folder):
- ext = os.path.splitext(file_name)[1].lower().replace('.', '')
- folder_name = ext.upper() if ext else 'Unknowns'
- destination_folder = os.path.join(base_folder, folder_name)
- os.makedirs(destination_folder, exist_ok=True)
- shutil.move(source_path, destination_path)

How to Run

1) Place auto_sort.py in your project folder.

2) Create a 'downloads/' folder next to the script.

3) Put your files into 'downloads/'.

4) Run in terminal: python auto_sort.py

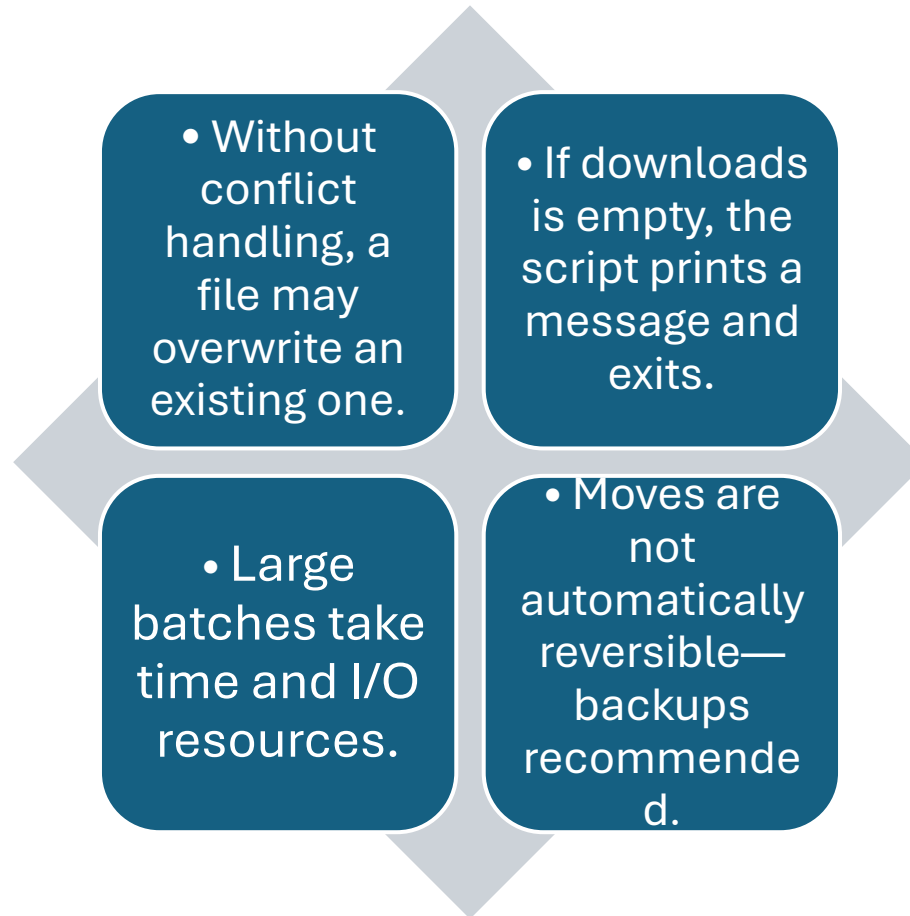5) Observe console output for moved files.

Safety & Robustness

- • Add name conflict handling to prevent overwrites (e.g., 'file (1).jpg').

- • Wrap moves in try/except to catch locked/unreadable files.

- • Optionally log actions to a file for auditing.

- • Use 'Unknowns' for files with no extension.

Customization Ideas

- Group images under IMAGES/ (JPG, PNG, GIF, etc.).
- Provide copy vs. move mode (shutil.copy vs. shutil.move).
- Add a dry-run flag to preview actions.
- Organize by date (YYYY-MM).
- Add CLI options: --source, --dest, --simulate.

Limitations & Notes

- Without conflict handling, a file may overwrite an existing one.

- If downloads is empty, the script prints a message and exits.

- Large batches take time and I/O resources.

- Moves are not automatically reversible—backups recommended.

Demo Scenario

- downloads/: report.pdf, photo.jpg, notes.txt, slides.PDF

- Run → PDF/: report.pdf, slides.PDF; JPG/: photo.jpg; TXT/: notes.txt

- Console prints lines like: 'report.pdf -> PDF moved to folder.'

Takeaways

- Automates repetitive file sorting tasks and saves time.

- Simple, portable, and easy to extend.

- A solid base for a more professional archiving tool.