File Organizer

Tags

File Organizer

Introduction

We often encounter problems organizing our files on our computer. It becomes really hectic when you download a file and want to store it somewhere, but you cannot find the file. Employees spend from 1 hour to 3 hours searching for a file on their system every day. So, a file organizer is really important.

Modules used

- OS Module
- Shutil Module

By the OS module we can change directories, change path, add path, check if the file or folder is available and more.

The Shutil module helps to move files to one folder to another folder and it has many more features.

Code

```
import os
import shutil
from extention_map import extension_map  # Ensure this is a dict


def organize_files(path):
    # Change the working directory to the provided path
    try:
        os.chdir(path)
    except Exception as e:
        print(f"Error changing directory: {e}")
        return

    # Create folders based on the extension_map if they don't ex
    for folder_name in extension_map.values():
        folder_path = os.path.join(path, folder_name)
        if not os.path.exists(folder_path):
            os.makedirs(folder_path)

    # List all files in the directory
    try:
        check = os.listdir(path)
    except Exception as e:
        print(f"Error listing directory contents: {e}")
        return

    # Iterate over each file in the directory
    for file_name in check:
        file_path = os.path.join(path, file_name)

        # Skip directories
        if os.path.isdir(file_path):
            continue
```

```
        # Get the file extension
        basename, extension = os.path.splitext(file_name)

        # Check if the file's extension is in the extension_map
        if extension in extension_map:
            folder_name = extension_map[extension]
            target_folder_path = os.path.join(path, folder_name
            try:
                shutil.move(file_path, os.path.join(target_folde
                print(f"Moved: {file_name} -> {folder_name}")
            except Exception as e:
                print(f"Error moving file {file_name}: {e}")

    print("Files have been organized.")


inp = input(
    "Enter your path name where all your files are present: (Ex
    "username]\\OneDrive\\Desktop\\Python Projects\\FileOrganize
organize_files(inp)
```

This program handles many error exceptions that helps the user to understand whats the error that is happening.

Conclusion

By using this program in the program in your system you can save a lot of time handling files.

The file organization project is a practical example of how Python can be used to automate tedious tasks, such as sorting and organizing files based on their extensions. By leveraging Python's powerful standard libraries like `os` and `shutil`, we were able to create a script that:

1. **Interacts with the File System:**
   - Changes the working directory to the user-specified path.
   - Lists all files and directories in the given path.
2. **Categorizes Files:**
   - Creates destination folders based on a predefined mapping of file extensions to folder names.
   - Processes each file in the directory, checks its extension, and moves it to the appropriate folder.
3. **Prompts User Input:**
   - Takes a directory path from the user, making the script versatile and adaptable to different file structures.

How to clone

```
To clone this repository to your local machine, follow these
steps:

1. **Ensure Git is Installed:**
   - Before you begin, make sure you have Git installed on yo
ur computer. You can download Git from [git-scm.com](https://
git-scm.com/).

2. **Clone the Repository:**
   - Open your terminal (macOS, Linux) or Command Prompt / Gi
t Bash (Windows).
   - Use the `git clone` command followed by the URL of this
repository.

    ```
```
```

````
    git clone https://github.com/your-username/your-repository.git
    ```

3. **Navigate to the Cloned Repository:**
   - Change into the directory of the cloned repository.

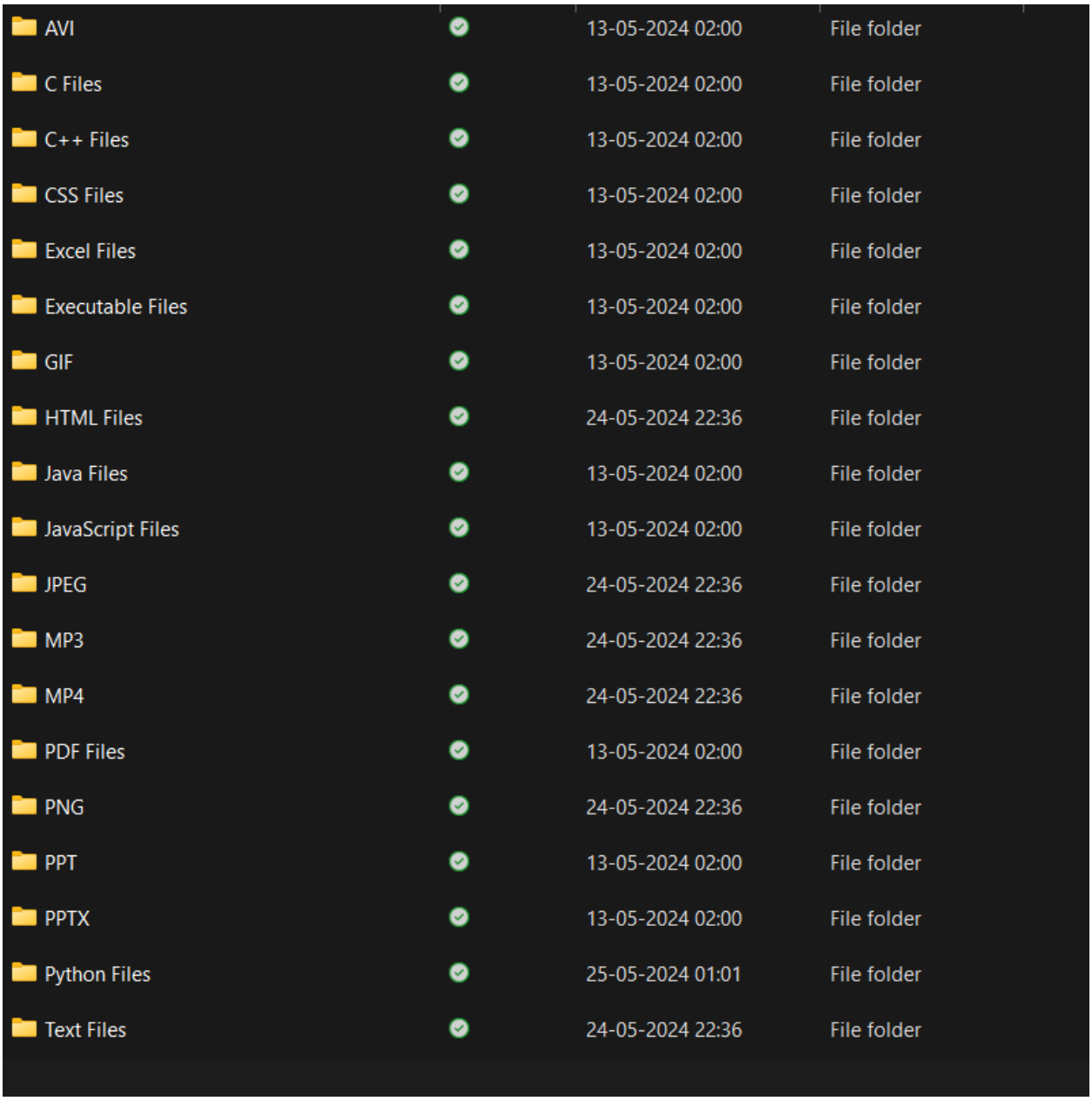
    ```
    cd your-repository
    ```

4. **Verify the Clone:**
   - You can verify that the repository has been cloned by listing the files in the directory.

    ```
    ls  # For macOS/Linux
    dir # For Windows
    ```
````

How to use

After you have cloned the git repository you will export the python files present in it and paste it in the folder you want your files to be organized. It will create various files like :
````



As per the files you have in your folder the files will be shifted to these folders. If you don't want so many folders to be created you can go in `extention_map.py` and remove the content of the dictionary and then run the code.

Notes

- You need both the files `main.py` and `extention_map.py` to run the code otherwise the program will show an error.
