



Linux et Bash - Manipulation de fichier et dossier

🕒 45 minutes 📁 Normal



DataScientest • com

Introduction à Linux

Manipulation de fichier et dossier

Nous allons maintenant voir comment créer des documents et des dossiers.

Création de fichier et dossier

Pour créer un fichier vide, nous passons par la commande `touch` et nous spécifions un nom de fichier.

Exécutez cette commande pour créer un fichier `my_file`.

```
1 touch my_file
2
```

Si ce fichier existe déjà, cela ne fait que le mettre à jour sa dernière date de modification.

Dans le dossier maison (~), créez les fichiers `file1`, `file2`, `file3` et vérifiez leur existence en utilisant `ls`.

Pour créer un répertoire, on peut utiliser la commande `mkdir` (qui signifie `make directory`) en lui précisant un nom pour ce dossier :

Exécutez cette commande pour créer un dossier `my_directory`.

```
1 mkdir my_directory
2
```

Créez un dossier `directory1` dans le dossier maison.

Nous pouvons aussi créer des fichiers et des dossiers en dehors du dossier courant en utilisant le chemin absolu ou relatif : par exemple, nous pouvons exécuter la commande suivante :

Créez un fichier `file4` dans ce dossier en utilisant la commande suivante.

Machine status



Ubuntu
Server
18.04 LTS
SSD
Volume
Type
64-bit x86

Online

34.245.135.23



Connect

Reset



Stop



Vérifiez que le fichier a bien été créé en utilisant la commande `ls -R ~` qui permet de lister les fichiers de manière récursive.

Nous pouvons utiliser la commande `rm` qui signifie `remove` pour supprimer un fichier ou un dossier :

Supprimez le fichier `file3` avec cette commande :

```
1 rm file3
2
```

ou

```
1 rm ./file3
2
```

ou

```
1 rm /home/ubuntu/file3
2
```

Pour supprimer un dossier, il faut ajouter l'argument `-r` :

Supprimez le dossier créé précédemment en utilisant la commande :

```
1 rm -r /home/ubuntu/directory1
2
```

Créez à présent un dossier `directory2` dans le dossier `maison`. Nous pouvons bien évidemment copier/coller des fichiers/dossiers en utilisant `cp` ou les déplacer en utilisant `mv`.

Copiez le fichier `file1` et collez le dans le dossier `directory2` en utilisant la commande suivante :

```
1 cp ./file1 ./directory2/
2
```

Nous pouvons faire la même chose en donnant un nouveau nom au fichier de destination :

```
1 cp ./file1 ./directory2/new_file1
2
```

Déplacez le fichier `file2` dans le dossier `directory2` en le renommant `new_file2` avec la commande :

```
1 mv ./file2 ./directory2/new_file2
2
```

Nous pouvons utiliser `mv` pour renommer un fichier si le fichier de destination est dans le même répertoire que le fichier d'origine. Observez tous les changements avec la commande `ls -R`. Pour copier des dossiers, on peut utiliser l'argument `-r` comme nous pouvons le faire avec la commande `rm`.

Édition et lecture de fichier

Imprimer dans un fichier



```
1 echo hello world
2
```



Nous affichons `hello world` dans la sortie standard, c'est-à-dire ici le terminal.

Pour rediriger cette impression, nous passons par `>` ou `>>` avec le nom d'un fichier.

Par exemple, nous pouvons écrire `hello world` dans le fichier `file1` en utilisant cette commande :

Exécutez cette commande pour écrire une ligne dans le `file1`.

```
1 echo hello world > file1
2
```

La différence entre ces deux possibilités est que la première écrase le contenu du fichier avant d'écrire dedans alors que la seconde ajoute le contenu à la suite du fichier s'il existe déjà. Dans le cas où le fichier n'existe pas, les deux commandes sont équivalentes : nous créons le fichier et nous ajoutons le résultat de la commande `echo` dans le fichier. Pour afficher le contenu d'un fichier, nous pouvons utiliser la commande `cat`.

Essayez d'afficher le contenu du fichier `file1` :

```
1 cat file1
2
```

Cette commande devrait retourner `hello world`.

Exécutez les commandes suivantes :

```
1 echo hello world > file1
2 echo hello world > file1
3
4 echo hello world >> file2
5 echo hello world >> file2
6
```

Affichez ensuite le contenu de ces deux fichiers : le premier ne devrait contenir qu'une seule ligne alors que le second devrait en contenir deux. Nous pouvons utiliser ce système de redirection avec toute commande qui imprime sur la sortie standard.

Exécutez la commande suivante pour imprimer la liste du contenu de la racine / dans un fichier appelé `root_content`.

```
1 ls / > root_content
2
```

Affichez ensuite son contenu en utilisant la commande `cat`.

Dans certains cas (fichiers trop volumineux), nous ne voulons afficher que le début ou la fin du fichier. Dans ce cas, nous pouvons utiliser `head` ou `tail`. Ces deux commandes peuvent prendre l'argument `-n` qui introduit un nombre de lignes à afficher. Essayez ces deux commandes avec le fichier `root_content` :

Exécutez cette commande :

```
1 head -n 2 root_content
2 tail -n 3 root_content
3
```

Robin
BIRON

Dans les parties précédentes, nous avons vu comment ajouter du contenu à un fichier de manière automatique avec le résultat d'une commande. Nous pouvons vouloir utiliser un moyen plus simple pour modifier le contenu d'un fichier. Sachant que nous allons être amenés à créer des scripts, il nous faut un vrai éditeur de texte. Cet outil est `nano` qui est fourni par défaut avec Ubuntu.

Tapez `nano` pour ouvrir cet éditeur et ajoutez quelques lignes.

Pour fermer `nano`, il faut faire `ctrl+x`. Il est demandé alors si nous souhaitons sauvegarder les données modifiées. Ensuite, il nous est demandé de rentrer un nom de fichier. Nous pouvons alors valider le nom du fichier en tapant sur entrée. Notez enfin que pour modifier un fichier existant, nous pourrions faire : `nano nom_de_fichier`.

Validated