

# Le terminal

## Manipulations basiques

La majorité des interactions avec les systèmes Linux se font à travers des terminaux. Ce premier exercice a pour objectif de vous familiariser avec le terminal, et les capacités de celui-ci.

1. Ouvrir un terminal
2. Faire afficher « *bonjour !* » au terminal
3. Lister les fichiers du répertoire courant
4. Créer un répertoire « *test\_dir* »
5. En une commande unique, créer un fichier nommé « *test\_file* » dans le répertoire précédemment créé
6. Lister les fichiers contenus dans le répertoire « *test\_dir* »
7. Se positionner dans le répertoire « *test\_dir* »
8. Modifier le fichier « *test\_file* » afin d'y ajouter du texte à votre discrétion
9. Afficher dans le terminal le contenu du fichier « *test\_file* »
10. Renommer « *test\_file* » en « *test\_file\_v2* »
11. Copier « *test\_file\_v2* » dans un fichier « *test\_file\_v2\_2* »
12. Supprimer « *test\_file\_v2* »
13. Revenir dans le répertoire parent
14. Supprimer le répertoire « *test\_dir* »

## Scripts bash

Linux permet la création de scripts bash (fichiers en *.sh*). Ceux-ci permettent l'automatisation de différentes tâches et activités. Amusons-nous par exemple à automatiser l'exercice précédent.

L'objectif ici est de pouvoir, avec une commande simple, rejouer le premier exercice de ce TD.

1. Ouvrir un terminal
2. Créer un répertoire bac à sable « *bac\_a\_sable* »
3. Se positionner dans le répertoire « *bac\_a\_sable* »
4. Créer un fichier « *test\_script.sh* »
5. Dans le fichier « *test\_script.sh* » ajouter la ligne :

```
#!/bin/bash
```

Celle-ci donne un contexte au script, elle permet de définir ce qui va exécuter le code. Ici le programme « *bash* » donc

6. Sous cette ligne, ajouter :

- Une ligne de commande permettant la création d'un répertoire « *test\_dir* »
- Une ligne de commande permettant, en une seule commande, la création et l'écriture de « *toto* » dans un fichier « *test\_file.txt* »
- Une ligne de commande permettant d'afficher dans le terminal le contenu de « *test\_file.txt* »
- Une ligne supprimant le répertoire « *test\_dir* » et son contenu
- Faire afficher « *0* » au terminal

*Nota : Il peut s'avérer utile de tester vos commandes avant de les ajouter à votre script...*

7. Exécuter le script à l'aide de la commande :

```
./test_script.sh
```

8. L'exécution du script se solde par un échec :

```
./test_script.sh  
-bash: ./test_script.sh: Permission denied
```

Par défaut, un script bash est créé sans les droits d'exécution. Modifier les droits liés à ce script afin de le rendre exécutable par n'importe qui

9. Lancer le script « *test\_script.sh* » :

```
$ ./test_script.sh  
toto  
0
```

10. Pousser plus loin les capacités du script :

- Permettez de donner du texte en argument au script. Celui-ci écrira ce texte dans le fichier « *test\_file.txt* ».
- Ajouter une condition au script : si on n'a pas donnée de texte en argument du script, le texte est « *toto* » par défaut et le script retourne « *-1* » au lieu de « *0* »

# Exercices avancés

## Exercice 1 : redirection et lecture

Créer un script qui va devoir exploiter les données d'un fichier de notes que vous allez créer au préalable dans le même répertoire que le script.

Ce fichier sera appelé `FichierNote.txt` et devra se présenter comme ceci :

```
Martin Eric 12
Petit Julie 8
Richard Diane 14
```

Le script devra afficher les lignes dans lesquelles la note est supérieure ou égale à 10.

Exemple :

```
[ ~ ] ./NomDuScript
Martin Eric 12
Richard Diane 14
```

## Exercice 2 : écriture, redirection et calcul

Vous avez deux scripts bash et un fichier texte :

- `script1.sh` doit générer des nombres aléatoires et les écrire dans un fichier nommé `nombres.txt`. Chaque nombre doit être sur une nouvelle ligne.
- `script2.sh` doit lire les nombres à partir de `nombres.txt`, trouver leur somme et écrire la somme dans un fichier nommé `somme.txt`.
- Le fichier `nombres.txt` doit contenir au moins 10 nombres entiers aléatoires compris entre 1 et 100.

Utilisez des redirections pour réaliser la tâche.

# Annexe A : Commandes et redirection

Les commandes Linux acceptent de nombreuses options dont on peut consulter la documentation en tapant `man ls` par exemple pour la commande `ls`. Celle-ci comporte des options pour afficher les fichiers cachés `ls -a` ou encore pour afficher les détails et permissions d'un fichier `ls -l`.

Commande	Description
cd	<i>Change Directory</i> Se déplacer dans l'arborescence
ls	<i>Lister</i> le contenu du répertoire courant
mkdir	Permet de créer un répertoire ( <i>make dir</i> )
pwd	<i>Affiche le dossier courant</i> (Print Workgin Directory)
cp	<i>Copier</i> des fichiers ou des répertoires
mv	<i>Déplacer (move)</i> ou renommer des fichiers ou des répertoires
rm	<i>Effacer (remove)</i> des fichiers ou des répertoires
cat	<i>Visualiser (concaténer)</i> le contenu d'un fichier
echo	<i>Afficher</i> un message ou le contenu d'une variable
touch	<i>Créer</i> un fichier vide ou réinitialiser le <i>timestamp</i> d'un fichier
chmod	Modifier les autorisations d'accès à un fichier
nano	<i>Éditeur de texte</i> en ligne de commande. Il y en a beaucoup
ps	Afficher les informations des <i>processus</i> en cours
top	<i>Gestionnaire de ressource</i> en TUI
kill	Envoyer un <i>signal</i> au processus, généralement pour l'arrêter
grep	<i>Filtrer</i> une sortie en ne gardant que les lignes contenant un terme

Linux permet la redirection des sorties des commandes.

Une commande `echo 'bonjour'` aura pour sortie « *bonjour* ». Cette sortie peut être redirigée afin de ne pas être affichée du tout, ou au contraire pour être dirigée vers un fichier texte, ou de log par exemple :

```
$ echo 'bonjour'
bonjour
$ echo 'bonjour' > /dev/null
$ echo 'bonjour' > text.txt
$ cat text.txt
Bonjour
```

*Nota : rediriger une sortie vers un fichier inexistant aura pour effet de créer ce fichier.*