Manipulation de fichiers

Informatique pour tous

Arborescence des fichiers sous Linux

Les dossiers et fichiers de Linux ont une structure **arborescente** : chaque dossier (sauf la **racine** /) a un dossier **père** dans lequel il est inclus et contient éventuellement des dossiers **fils**.

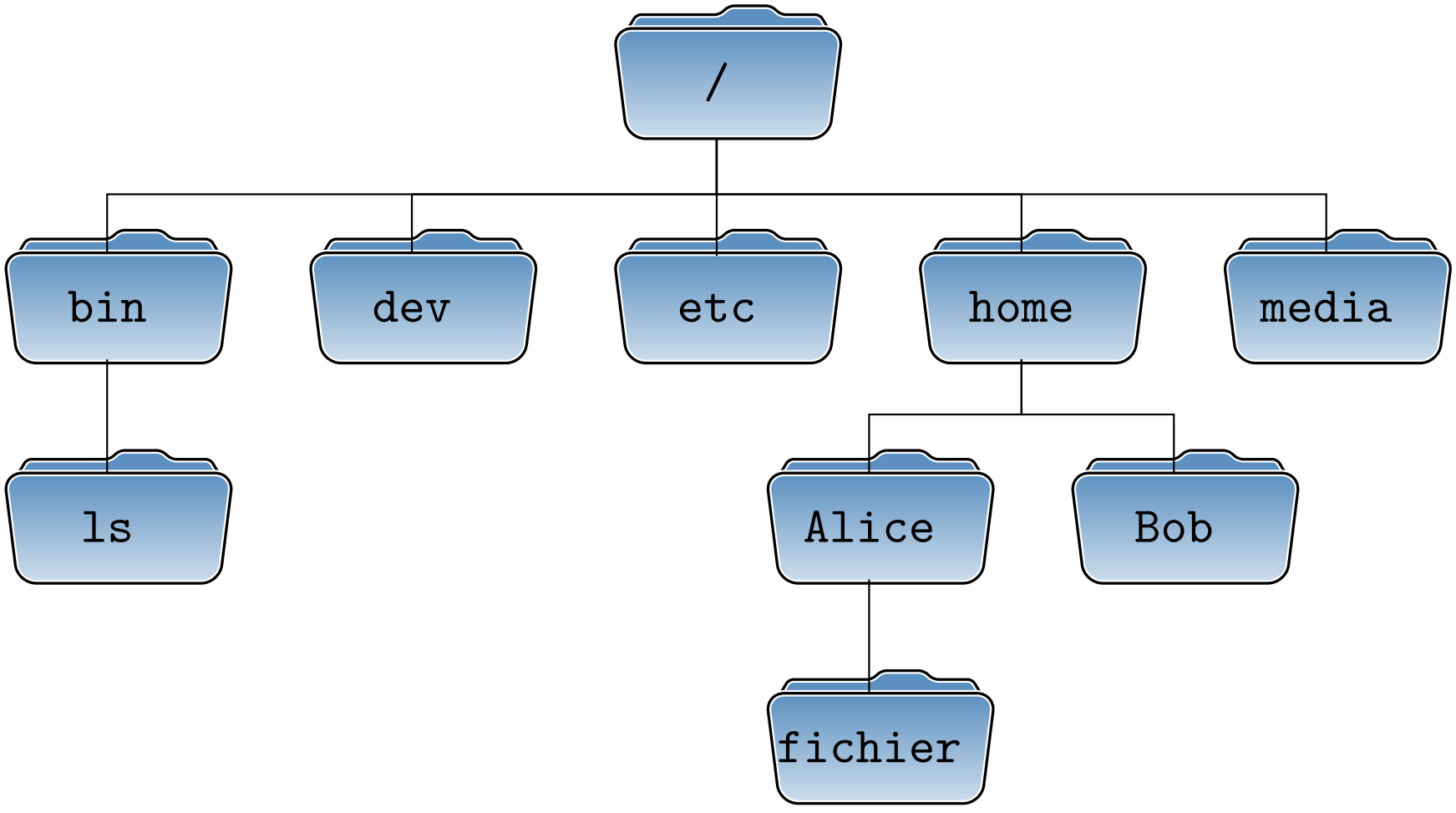
Arborescence des fichiers sous Linux

Les dossiers et fichiers de Linux ont une structure **arborescente** : chaque dossier (sauf la **racine** /) a un dossier **père** dans lequel il est inclus et contient éventuellement des dossiers **fils**.

Le **chemin** (**absolu**) d'un fichier `file` est la suite des fils qu'il faut parcourir depuis la racine / jusqu'à `file`.

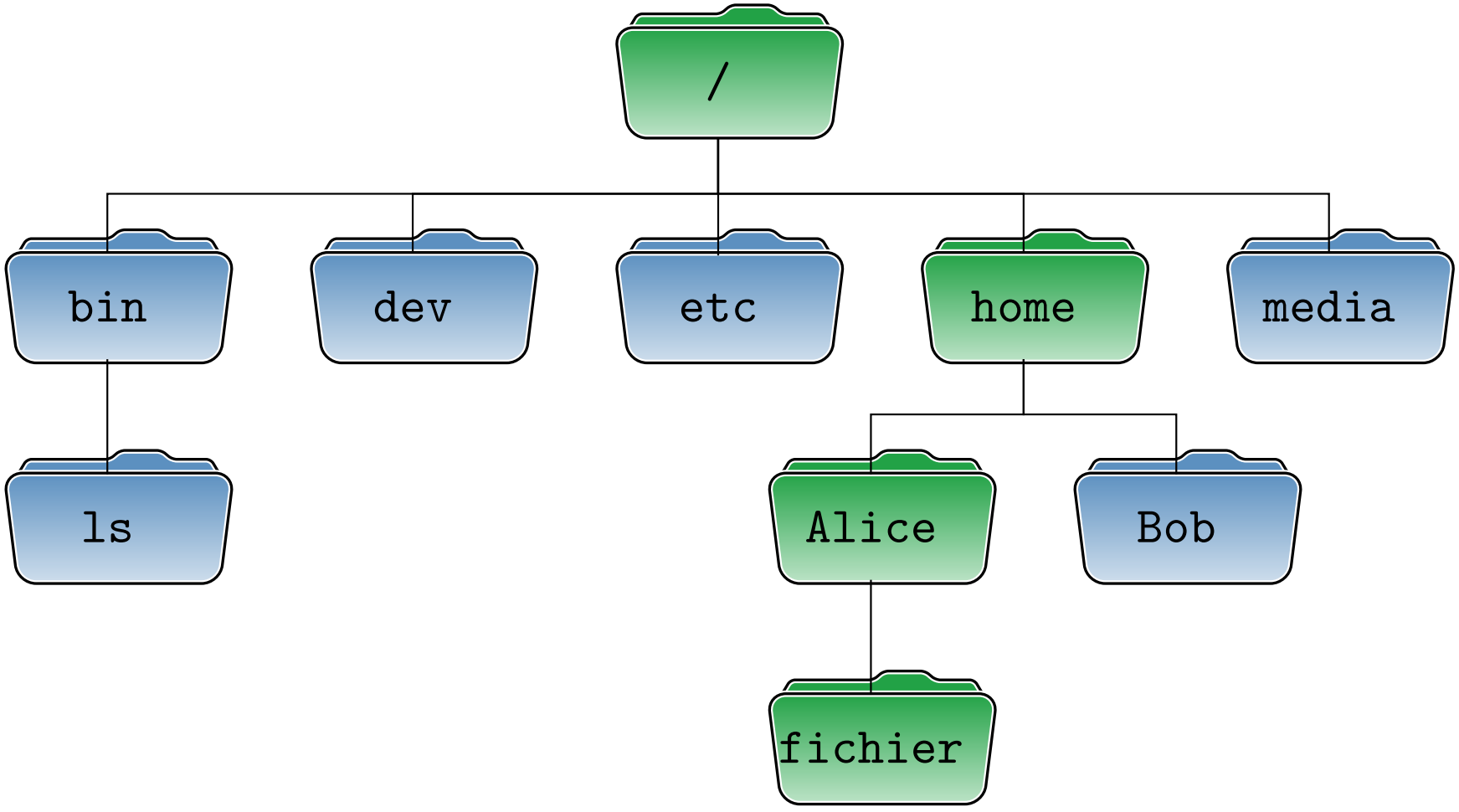
On peut aussi considérer un chemin **relatif** d'un fichier, depuis un autre répertoire que la racine.

Arborescence des fichiers sous Linux



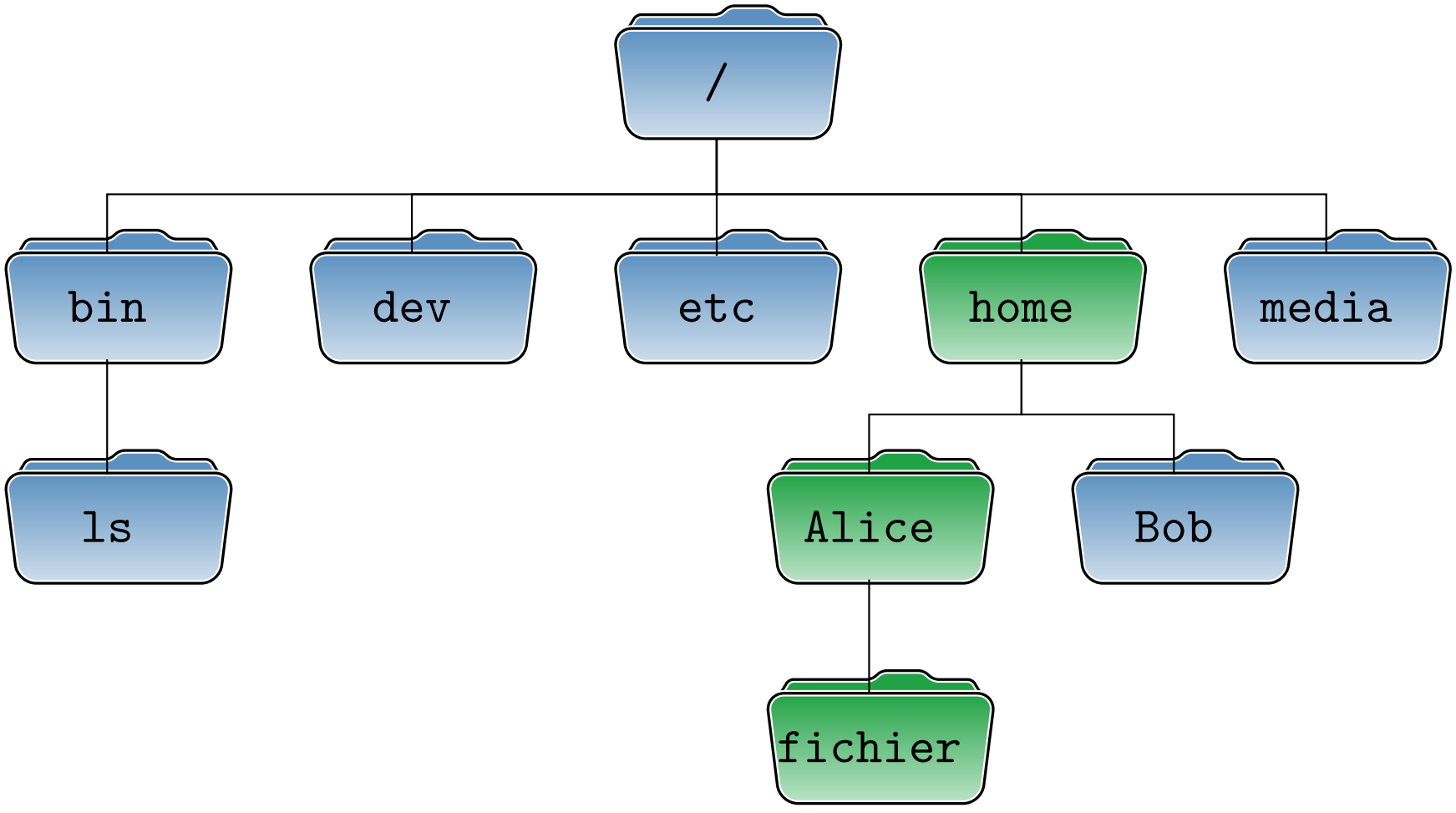
Exemple de système de fichiers sous Linux.

Arborescence des fichiers sous Linux



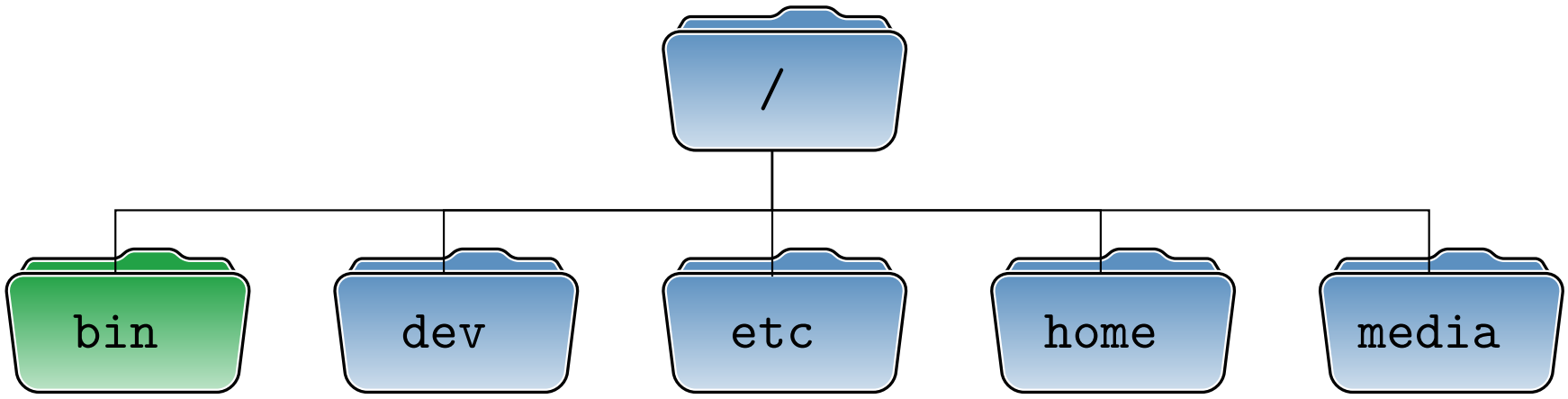
`fichier` a pour chemin **absolu** : `home/Alice/fichier`

Arborescence des fichiers sous Linux



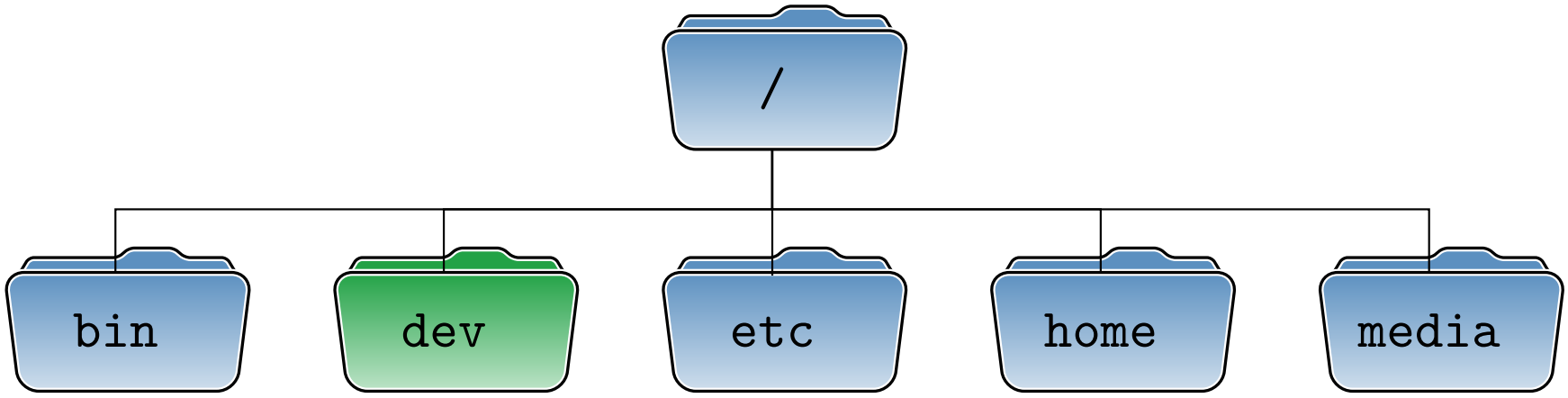
`fichier` a pour chemin **relatif** depuis `home` : `Alice/fichier`

Arborescence des fichiers sous Linux



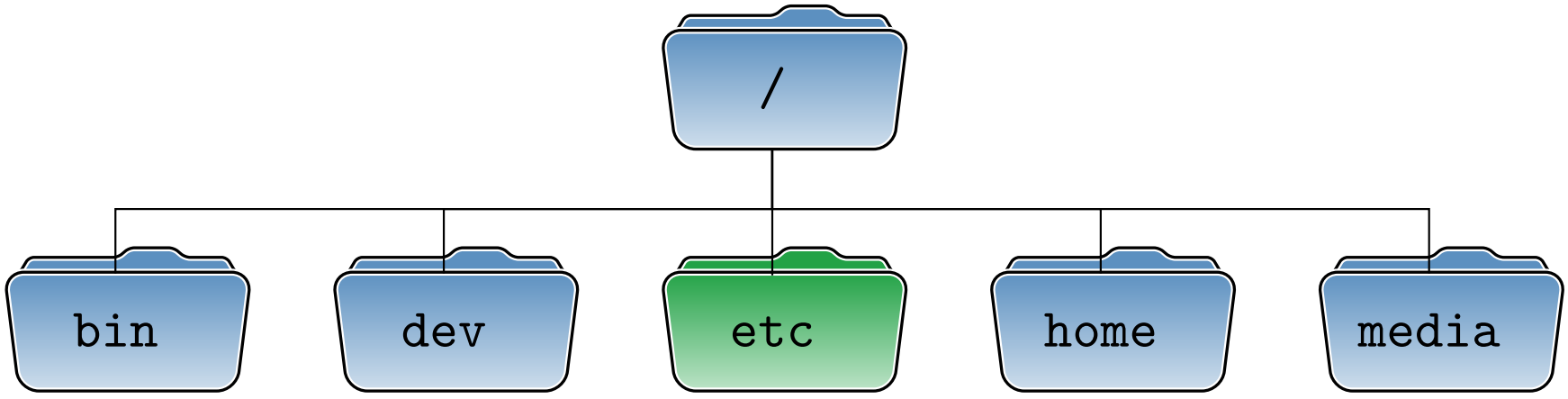
`bin` contient les programmes **exécutables** pour démarrer le système ainsi que les commandes de la console : ls, cd, mv...

Arborescence des fichiers sous Linux



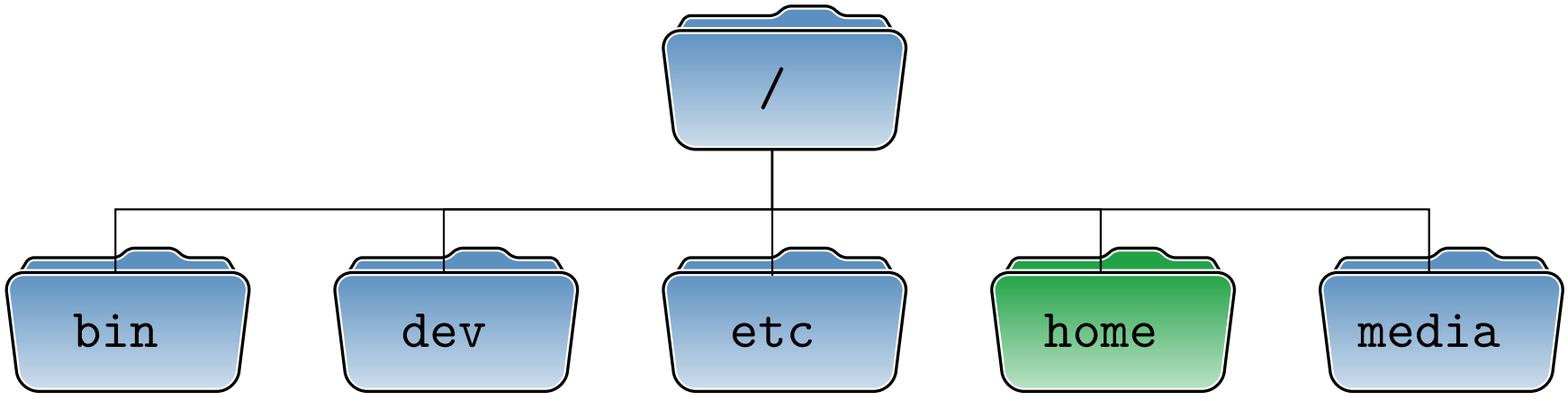
`dev` contient des interfaces vers des périphériques comme les disques durs, lecteur CD...

Arborescence des fichiers sous Linux



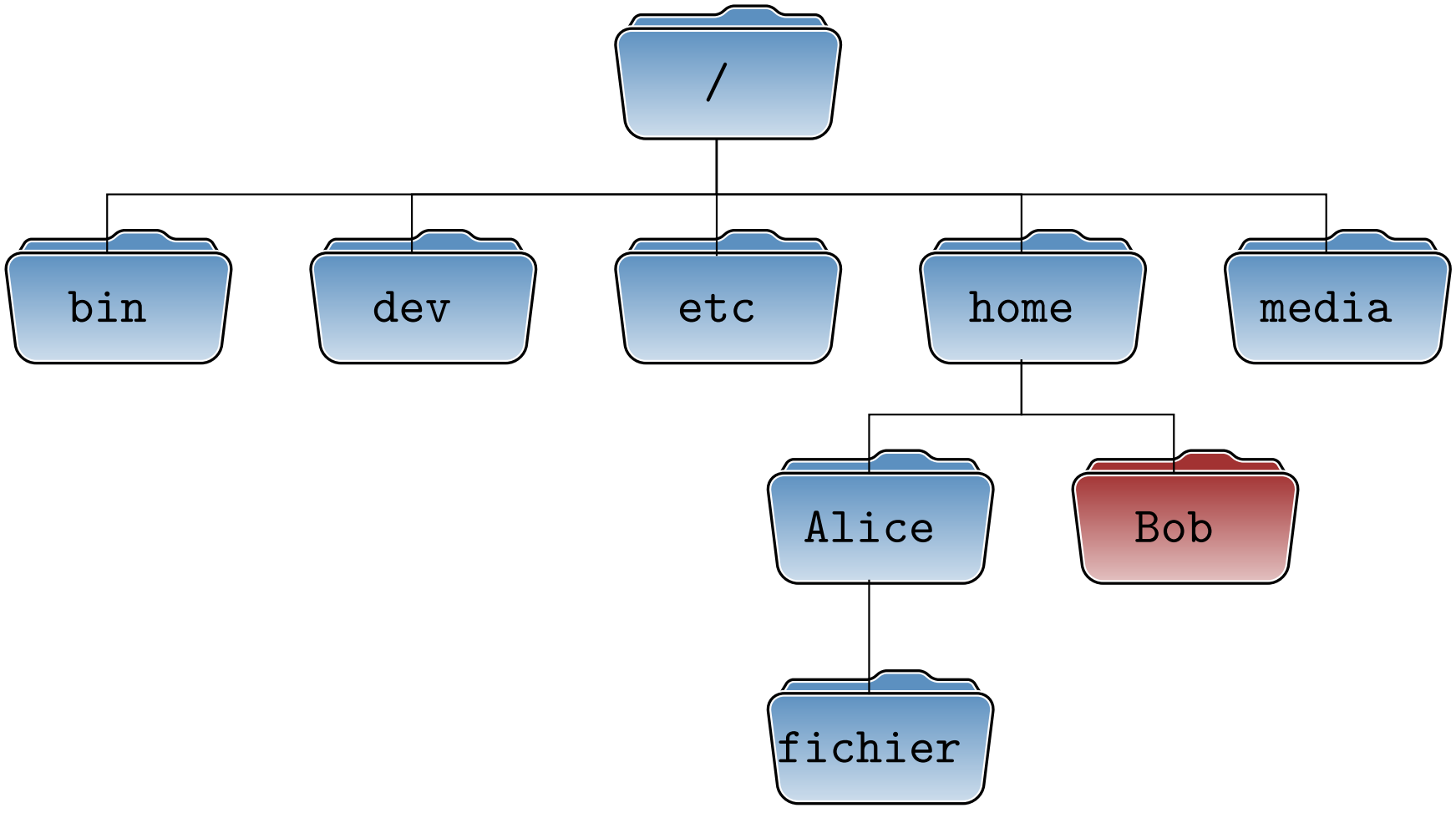
`etc` contient les fichiers de configuration du système ainsi qu'un fichier `/etc/shadow` contenant les mots de passes (cryptés).

Arborescence des fichiers sous Linux



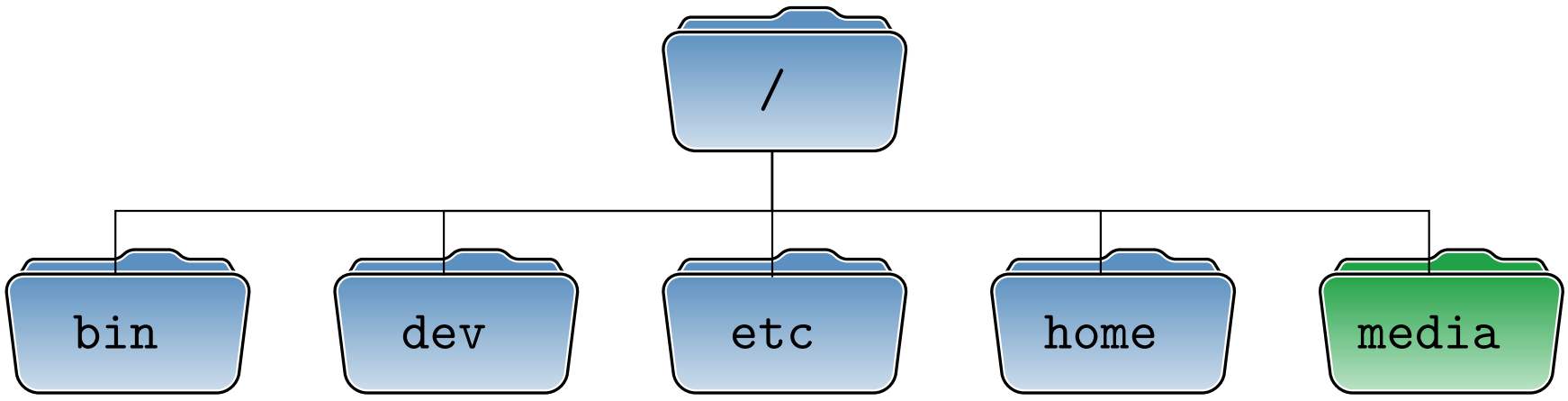
`home` contient les dossiers personnels des utilisateurs.

Arborescence des fichiers sous Linux



Certains fichiers ne peuvent être accédés que par certains utilisateurs :
Alice ne peut pas utiliser les fichiers de Bob.

Arborescence des fichiers sous Linux



Une clé USB est montée sous forme de dossier dans `media`.

Manipulation des fichiers

Pour manipuler (lire, écrire, renommer, supprimer...) des fichiers, il y a plusieurs solutions :

1. Interface graphique : explorateur sous Linux ou Windows.

Manipulation des fichiers

Pour manipuler (lire, écrire, renommer, supprimer...) des fichiers, il y a plusieurs solutions :

1. Interface graphique : explorateur sous Linux ou Windows.
2. Ligne de commande : Bash sous Linux, Powershell sous Windows.

Manipulation des fichiers

Pour manipuler (lire, écrire, renommer, supprimer...) des fichiers, il y a plusieurs solutions :

1. Interface graphique : explorateur sous Linux ou Windows.
2. Ligne de commande : Bash sous Linux, Powershell sous Windows.
3. Python : avec le module `os`, compatible sous tout OS.

Manipulation des fichiers

Question

Pourquoi utiliser autre chose que l'interface graphique ?

Manipulation des fichiers

Question

Pourquoi utiliser autre chose que l'interface graphique ?

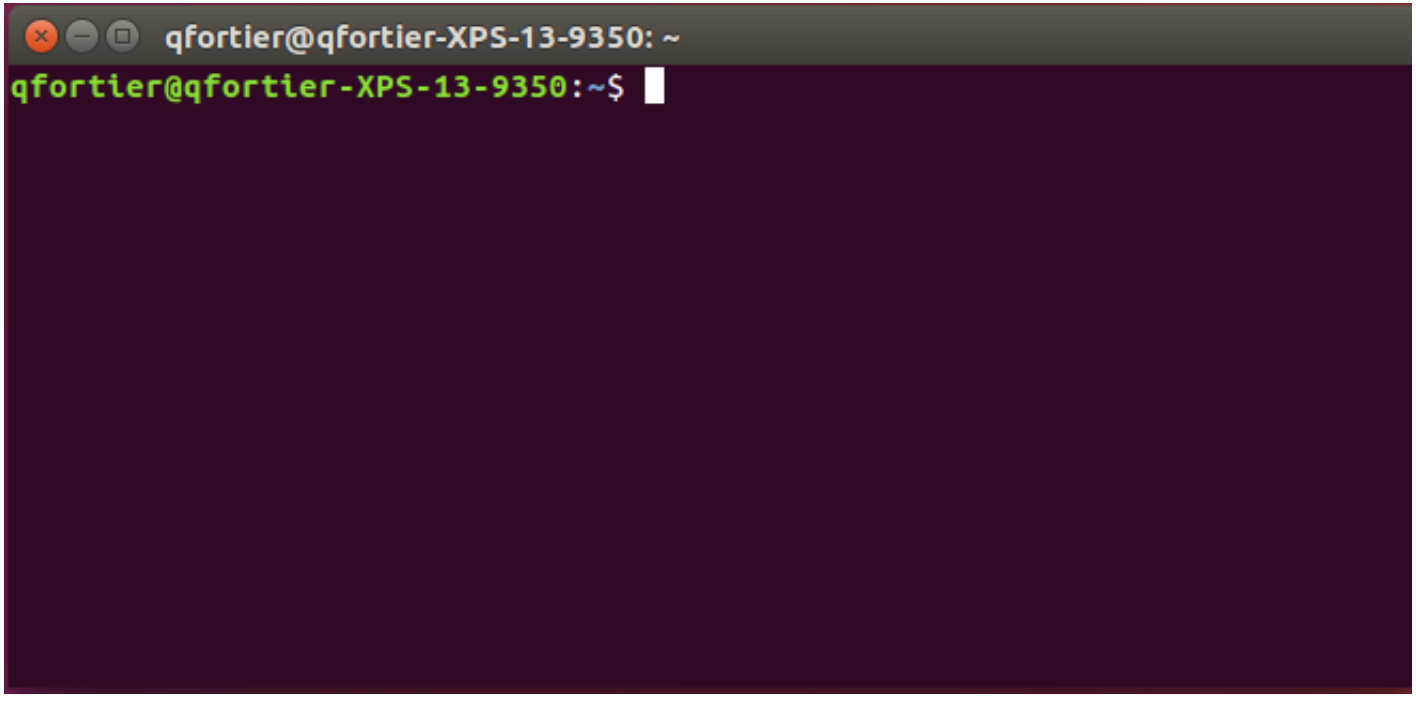
1. Pour faire des tâches répétitives : renommer 100 images .jpg en .png, supprimer tous les fichiers temporaires...
2. Pour accéder à distance à un ordinateur : très simple avec SSH
3. ...

Bash

Pour ouvrir une console : Ctrl Alt T

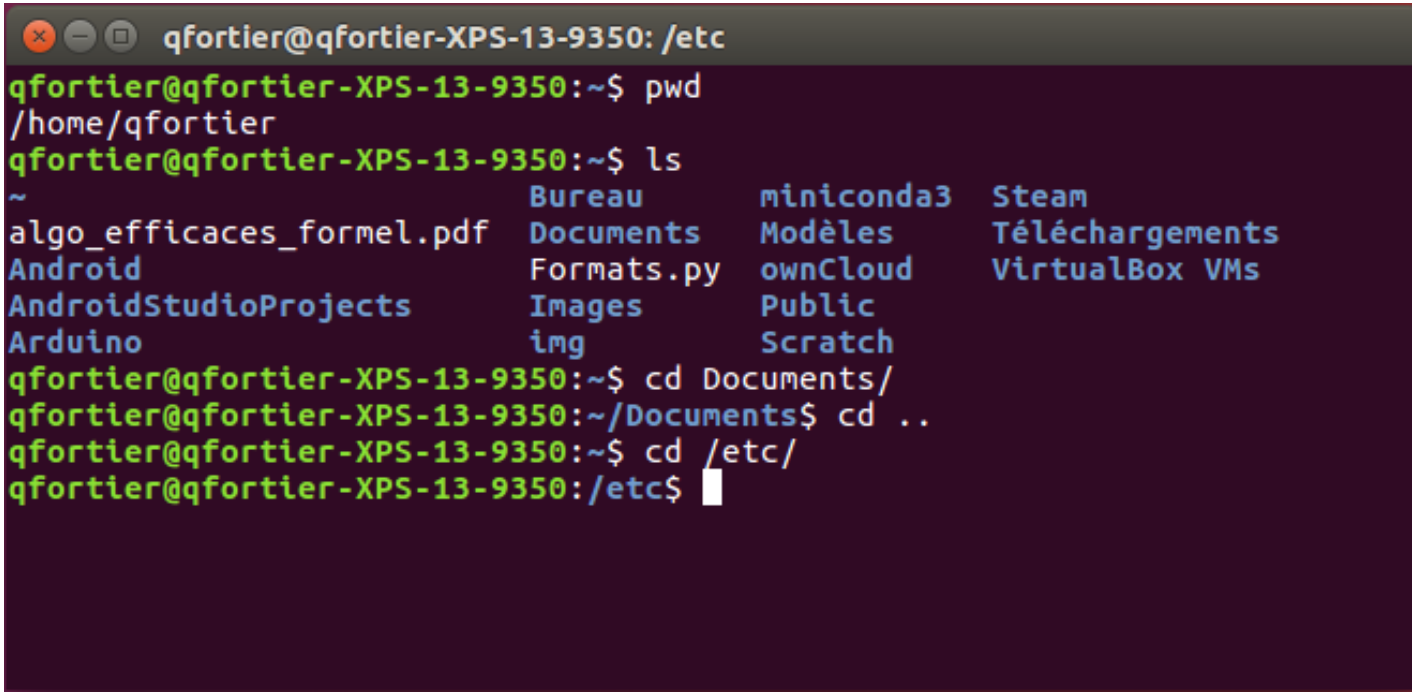
Bash

Pour ouvrir une console : Ctrl Alt T



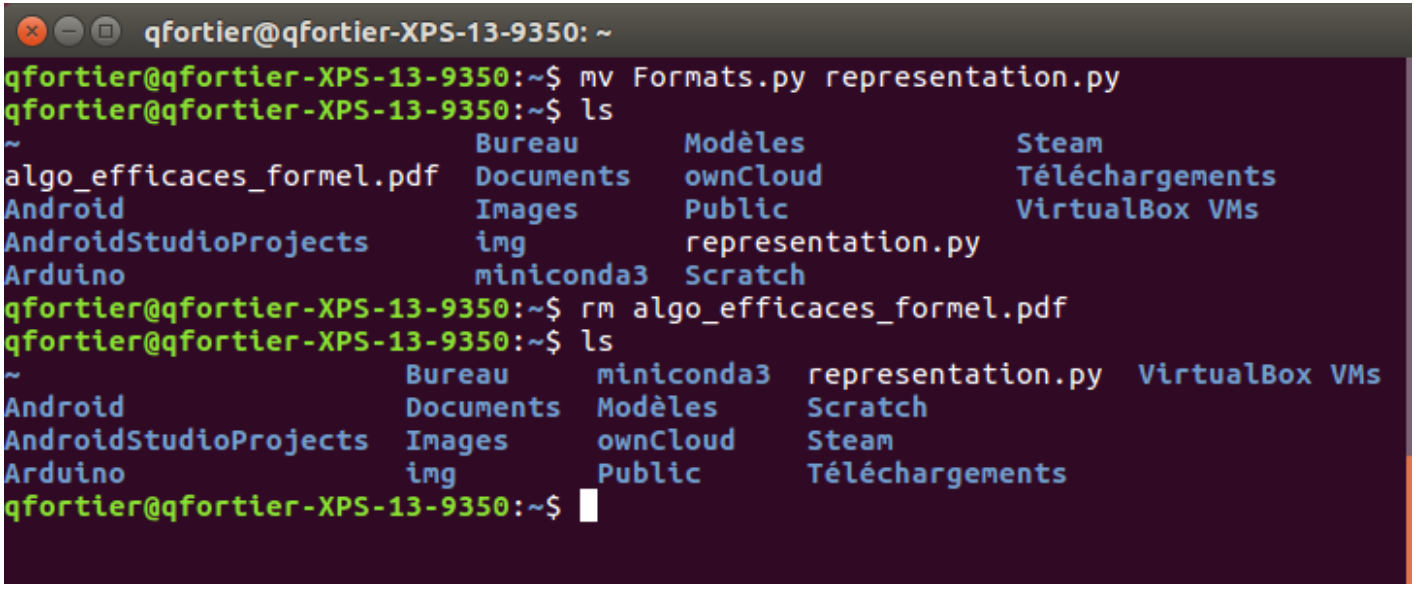
Bash

`pwd` (**p**ath **w**orking **d**irectory) : chemin du répertoire courant.
`ls` : **l**i**s**te les fichiers du répertoire courant.
`cd <dir>` (**c**hange **d**irectory) : le répertoire courant devient `<dir>`.



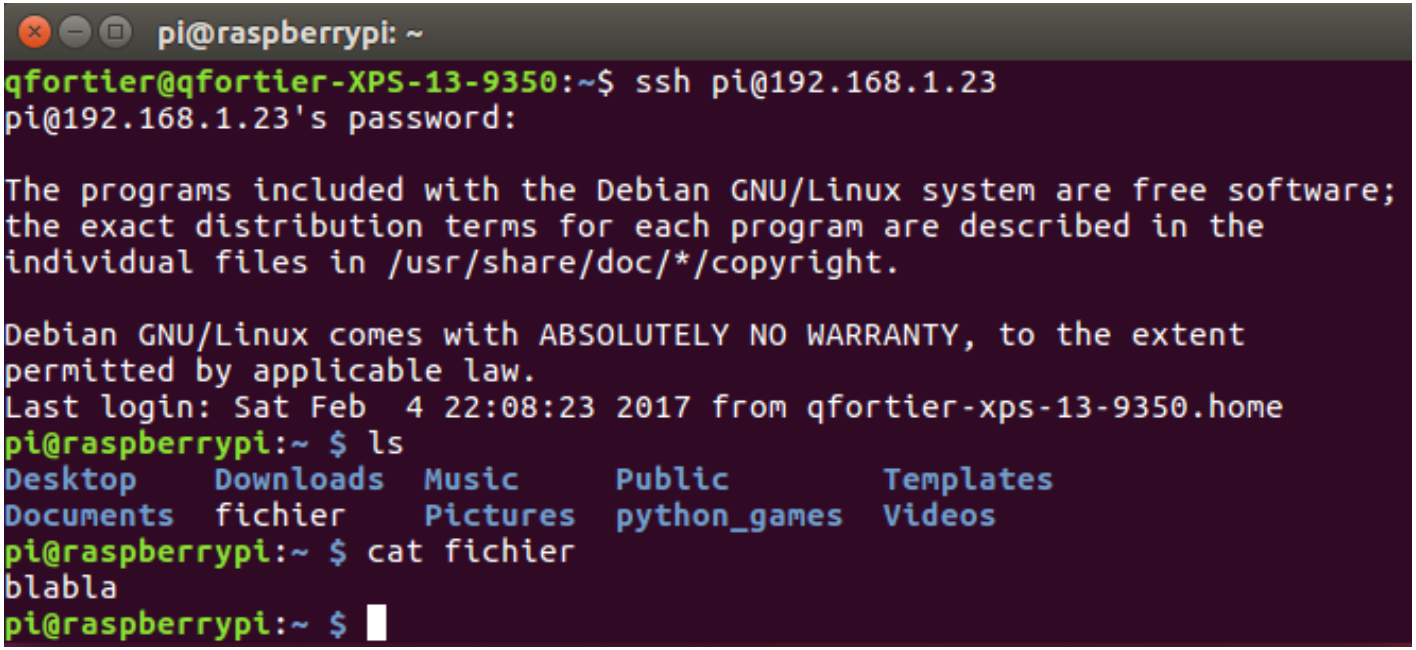
Bash

`cat <file>` (con**cat**enate) : affiche le contenu de `<file>`.
`cp <file1> <file2>` (**co**p**y**) : copie `<file1>` vers `<file2>`.
`mv <file1> <file2>` (**m**o**v**e) : déplace `<file1>` vers `<file2>`.
`rm <file>` (**rm**ove) : supprime `<file>`.



Bash

`ssh` : accéder à un ordinateur distant.



Le module `os` contient des fonctions de manipulation de fichiers :

	Bash	Python
Répertoire courant	`pwd`	`os.getcwd`
Changer de répertoire	`cd`	`os.chdir`
Lister les fichiers	`ls`	`os.listdir`
Déplacer un fichier	`mv`	`os.rename`
Supprimer un fichier	`rm`	`os.remove`

Python

```
In [15]: import os

In [16]: os.getcwd()
Out[16]: '/home/qfortier'

In [17]: os.chdir('Images')

In [18]: os.listdir()
Out[18]:
['image6.jpg',
 'image7.jpg',
 'image0.jpg',
 'image2.jpg',
 'image1.jpg',
 'image3.jpg',
 'image4.jpg',
 'image5.jpg',
 'image8.jpg',
 'image9.jpg']
```

Python

Question

Comment renommer toutes les images .jpg en .png ?

Question

Comment renommer toutes les images .jpg en .png ?

`os.listdir()` renvoie une liste de noms de fichiers.
`os.rename(<file1>, <file2>)` renomme `<file1>` en `<file2>`.

Question

Comment renommer toutes les images .jpg en .png ?

`os.listdir()` renvoie une liste de noms de fichiers.
`os.rename(<file1>, <file2>)` renomme `<file1>` en `<file2>`.

```
for filename in os.listdir():
    os.rename(filename, filename[:-3] + "png")
```

Python

```
for filename in os.listdir():
    os.rename(filename, filename[:-3] + "png")
```

```
In [24]: os.listdir()
Out[24]:
['image2.png',
 'image0.png',
 'image8.png',
 'image1.png',
 'image3.png',
 'image5.png',
 'image6.png',
 'image4.png',
 'image7.png',
 'image9.png']
```

Écriture et lecture

La fonction `open` permet d'ouvrir un fichier, elle prend 2 arguments :

1. Le chemin du fichier.
2. Le mode d'ouverture : `"r"` (lecture), `"w"` (écriture, en écrasant le fichier s'il existe déjà) ou `"a"` (écriture à la fin du fichier).

Écriture et lecture

La fonction `open` permet d'ouvrir un fichier, elle prend 2 arguments :

1. Le chemin du fichier.
2. Le mode d'ouverture : `"r"` (lecture), `"w"` (écriture, en écrasant le fichier s'il existe déjà) ou `"a"` (écriture à la fin du fichier).

Le résultat de `open` doit être récupéré dans une variable.

Écriture et lecture

La fonction `open` permet d'ouvrir un fichier, elle prend 2 arguments :

1. Le chemin du fichier.
2. Le mode d'ouverture : `"r"` (lecture), `"w"` (écriture, en écrasant le fichier s'il existe déjà) ou `"a"` (écriture à la fin du fichier).

Le résultat de `open` doit être récupéré dans une variable.

Une fois le traitement fini, il faut fermer le fichier avec `close`.

Lecture de fichier

On peut lire ligne par ligne un fichier avec `readline()` :

```
In [28]: miserables = open("miserables.txt", "r")

In [29]: miserables.readline()
Out[29]: 'En 1815, M. Charles-François-Bienvenu Myriel était évêqu
e de Digne. \n'

In [30]: miserables.readline()
Out[30]: 'C'était un vieillard d'environ soixante-quinze ans ; il
occupait le siége de Digne depuis 1806.\n'

In [31]: miserables.close()
```

Lecture de fichier

On peut lire toutes les lignes d'un coup avec `readlines()` (qui renvoie une liste des lignes) :

```
In [35]: miserables = open("miserables.txt", "r")

In [36]: lines = miserables.readlines()

In [37]: lines[100]
Out[37]: 'Pour libérer des pères de famille prisonniers pour dette
s : \tmille livres.\n'

In [38]: miserables.close()
```

Lecture de fichier

`read()` renvoie le fichier entier sous forme de chaîne de caractères :

```
In [39]: miserables = open("miserables.txt", "r")

In [40]: texte = miserables.read()

In [41]: len(texte)
Out[41]: 130760

In [42]: miserables.close()
```

Écriture dans un fichier

`write(<text>)` écrit `<text>` dans le fichier (à la place du texte existant si ouvert avec `"w"`, à la fin si ouvert avec `"a"`).

```
In [43]: miserables = open("miserables.txt", "a")

In [44]: miserables.write("Point final")
Out[44]: 11

In [45]: miserables.close()
```

Écriture dans un fichier

`write(<text>)` écrit `<text>` dans le fichier (à la place du texte existant si ouvert avec `"w"`, à la fin si ouvert avec `"a"`).

```
In [43]: miserables = open("miserables.txt", "a")

In [44]: miserables.write("Point final")
Out[44]: 11

In [45]: miserables.close()
```

```
In [50]: miserables = open("miserables.txt", "r")

In [51]: miserables.readlines()[-1]
Out[51]: 'Point final'
```

Écriture dans un fichier

`write(<text>)` écrit `<text>` dans le fichier (à la place du texte existant si ouvert avec `"w"`, à la fin si ouvert avec `"a"`).

```
In [52]: miserables = open("miserables.txt", "w")

In [53]: miserables.write("Oups")
Out[53]: 4

In [54]: miserables.close()

In [55]: miserables = open("miserables.txt", "r")

In [56]: miserables.read()
Out[56]: 'Oups'
```

Encodage

Les données d'un fichier doivent être stockées comme une suite de 0 et de 1.

Encodage

Les données d'un fichier doivent être stockées comme une suite de 0 et de 1.
Il faut donc être capable d'associer à chaque caractère une suite de 0 et de 1 : c'est ce qu'on appelle l'**encodage**.

Encodage

Quelques encodages de caractères :

1. ASCII : code les caractères latins et anglais, sur 7 bits. Codage par défaut pour Python 2.

Encodage

Quelques encodages de caractères :

1. ASCII : code les caractères latins et anglais, sur 7 bits. Codage par défaut pour Python 2.
2. UTF-8 : code les caractères de presque toutes les langues, sur 16 bits. Codage par défaut pour Python 3 et Linux.

Encodage

Quelques encodages de caractères :

1. ASCII : code les caractères latins et anglais, sur 7 bits. Codage par défaut pour Python 2.
2. UTF-8 : code les caractères de presque toutes les langues, sur 16 bits. Codage par défaut pour Python 3 et Linux.
3. CP1252 : code de nombreux caractères, sur 8 bits. Codage par défaut pour Windows.

Encodage

Quelques encodages de caractères :

1. ASCII : code les caractères latins et anglais, sur 7 bits. Codage par défaut pour Python 2.
2. UTF-8 : code les caractères de presque toutes les langues, sur 16 bits. Codage par défaut pour Python 3 et Linux.
3. CP1252 : code de nombreux caractères, sur 8 bits. Codage par défaut pour Windows.

Si on ouvre un fichier avec le mauvais encodage, des caractÃ"res mal dÃľcodÃľs vont apparaÃtre !

Encodage

`open` a une option pour préciser l'encodage utilisé :

Encodage

`open` a une option pour préciser l'encodage utilisé :

```
f = open("miserables.txt", "w", encoding="cp1252")
f.write("En 1815, M. Charles-François-Bienvenu Myriel était
évêque de Digne.")
f.close()
```

Encodage

`open` a une option pour préciser l'encodage utilisé :

```
f = open("miserables.txt", "w", encoding="cp1252")
f.write("En 1815, M. Charles-François-Bienvenu Myriel était
évêque de Digne.")
f.close()
```

```
In [16]: f = open("miserables.txt", "r")

In [17]: f.read()
---------------------------------------------
---------------
UnicodeDecodeError
```

Encodage

`open` a une option pour préciser l'encodage utilisé :

```
f = open("miserables.txt", "w", encoding="cp1252")
f.write("En 1815, M. Charles-François-Bienvenu Myriel était
évêque de Digne.")
f.close()
```

```
In [16]: f = open("miserables.txt", "r")

In [17]: f.read()
---------------------------------------------
---------------
UnicodeDecodeError
```

Par défaut, Python 3 ouvre les fichiers en UTF-8 !

Encodage

Il faut préciser le bon encodage :

```
In [18]: f = open("miserables.txt", "r", encoding="cp1252")

In [19]: f.read()
Out[19]: 'En 1815, M. Charles-François-Bienvenu Myriel était
évêque de Digne.'
```