

# GESTION DES FICHIERS C/UNIX

## Introduction

### Deux modes d'appel

- Par la bibliothèque standard C (appel de haut niveau)
- Par appels système (bas niveau)

Nécessite les includes suivant :

```
<unistd.h>  
<sys/types.h>  
<sys/stat.h>  
<fcntl.h>
```

Les fichiers sont vus comme des blocs d'octets, d'où l'utilisation de zone tampon de caractères.

Les fichiers sont identifiés par un entier appelé « descripteur de fichier » qui est un type entier.

Le système maintient une table des descripteurs en mémoire.

Les numéros des descripteurs suivant sont réservés :

```
0 canal d'entrée standard <-  
1 canal de sortie standard ->  
2 canal de sortie d'erreur standard ->
```

Donc le prochain descripteur utilisable pour ouvrir un fichier est 3

## Gestion des fichiers :

### Modification du masque par défaut

```
int umask (int new_mask)
```

Pose un masque par défaut, le masque par défaut est normalement 022, il est possible de le modifier.

```
Umask (022);  
Creat("toto",0666); => rw- r-- r—
```

```
110 110 110 => 666
```

```
&    000 010 010  => 022
      =====
      110 100 100  => 644 =>rw- r-- r--
```

### Ouvrir un fichier

Int open (const char \* référence, int mode, mode\_t permission)

Int open (const char \* référence, int mode)

*Où mode peut prendre les valeurs suivantes :*

O\_RDONLY ouverture en lecture seule

O\_WRONLY ouverture en écriture seule

O\_RDWR ouverture en lecture et écriture

O\_APPEND positionne les données en fin de fichier (mode ajout)

O\_TRUNC Ramène la longueur du fichier à zéro en supprimant le contenu.

O\_CREAT Permet de créer un fichier vide

O\_EXCL combiné avec O\_CREAT permet de garantir l'exclusivité de l'utilisation de la fonction open.

Nb : Il est possible d'associer ces valeurs avec l'opérateur ou |

*Où permission peut prendre les valeurs suivantes :*

S\_IRUSR lecture seule pour l'utilisateur

S\_IWUSR écriture seule pour l'utilisateur

S\_IXUSR Rendre exécutable le fichier pour l'utilisateur

S\_IRGRP lecture pour le groupe

S\_IWGRP écriture pour le groupe

S\_IXGRP Exécutable pour le groupe

S\_IROTH lecture seule pour les autres

S\_IWOTH écriture pour les autres

S\_IXOTH Exécutable pour les autres

Nb : Il est possible d'associer ces valeurs avec l'opérateur ou |

Open renvoie le numéro du descripteur sinon -1

### Autre manière de créer un fichier

Size\_t creat(char \* référence, mode\_t permission)

Idem à open(...) pour les permissions.

### Lecture dans un fichier

Size\_t read (int descripteur, void \* buffer, size\_t nb\_car)

Renvoie le nombre de caractères effectivement lus.

### **Ecrire dans un fichier**

Size\_t write(int descripteur, const void \* buffer, size\_t nb\_car)

Renvoie le nombre de caractères effectivement écrits.

### **Déplacement dans un fichier**

Off\_t lseek (int descripteur, off\_t deplace, int base)

SEEK\_SET ou 0 : Déplacement effectué à partir du début

SEEK\_CUR ou 1 : Déplacement effectué à partir de la position courante

SEEK\_END ou 2 : Déplacement effectué à partir de la fin du fichier

#### Exemple d'utilisation :

Lseek(df,0,2) Calcul de la taille d'un fichier

Lseek(df,0,0) Retour au début du fichier

Lseek(df,0,1) Retour à la position courante de l'index du fichier.

### **Fermeture d'un fichier**

Int close (int descripteur)

Renvoi 0 si OK sinon -1

### **Consulter les droits d'accès d'un fichier**

Int access (char \* référence, int mode)

Permet de tester les droit d'un fichier passé en paramètre.

#### **Mode peut prendre des valeurs entre :**

R\_OK lecture, W\_OK écriture, X\_OK exécutable, F\_OK le fichier existe ?

Retourne 0 si OK sinon -1

### **Modification des droit d'accès d'un fichier**

Int chmod(char \* référence, int permission)

*Où permission peut prendre les valeurs suivantes :*

S\_IRUSR lecture seule pour l'utilisateur  
 S\_IWUSR écriture seule pour l'utilisateur  
 S\_IXUSR Rendre exécutable le fichier pour l'utilisateur  
 S\_IRGRP lecture pour le groupe  
 S\_IWGRP écriture pour le groupe  
 S\_IXGRP Exécutable pour le groupe  
 S\_IROTH lecture seule pour les autres  
 S\_IWOTH écriture pour les autres  
 S\_IXOTH Exécutable pour les autres

Nb : Il est possible d'associer ces valeurs avec l'opérateur ou |

Retourne 0 si OK sinon -1

## Gestion des répertoires :

### Créer un répertoire

Int mkdir (const char \* repertoire, mode\_t mode)

*Où mode peut prendre les valeurs suivantes :*

S\_IRUSR lecture seule pour l'utilisateur  
 S\_IWUSR écriture seule pour l'utilisateur  
 S\_IXUSR Rendre exécutable le fichier pour l'utilisateur  
 S\_IRGRP lecture pour le groupe  
 S\_IWGRP écriture pour le groupe  
 S\_IXGRP Exécutable pour le groupe  
 S\_IROTH lecture seule pour les autres  
 S\_IWOTH écriture pour les autres  
 S\_IXOTH Exécutable pour les autres

Nb : Il est possible d'associer ces valeurs avec l'opérateur ou |

### Supprimer un répertoire

Int rmdir (const char \* chemin)

Attention un répertoire ne peut être supprimé que si celui ci est vide.

### Changer de répertoire courant

Int chdir(const chr \* chemin)

### Connaître le répertoire courant

Char \* getcwd(char \* tampon, taille\_t taille)

Ecrit le nom du répertoire dans tampon, renvoie *null* si le nom du répertoire dépasse la taille du tampon inscrit dans taille.

## Duplication des descripteurs et redirection :

Int dup (int ancien\_desc) renvoie le nouveau descripteur sinon -1

Int dup2 (int ancien\_desc, int nouveau\_desc)

Permet de dupliquer un descripteur de fichier, en offrant un ou plusieurs descripteurs différents, ayant accès au même fichier.

L'utilité est de faire de la redirection :

Exemple :

```
Df=open(« licence.txt »,1) ; (1)
```

```
Close(0) ; (2)
```

```
Dfbis=dup(df) ; /* prend le plus petit numéro de descripteur disponible*/ (3)
```

```
Close (df) ; (4)
```

### Etape 1

0 (Stdin)
1 (Stdout)
2 (StdErr)
3 (licence.txt)

On obtient après ouverture du fichier licence.txt, le numéro de descripteur 3 affecté à celui ci.

### Etape 2

<del>0 (Stdin)</del>
1 (Stdout)
2 (StdErr)
3 (licence.txt)

L'instruction `close(0)` permet de fermer définitivement le descripteur 0 (entrée de données par défaut), donc laisse une place libre.

**Etape 3**

0 (licence.txt)
1 (Stdout)
2 (StdErr)
3 (licence.txt)

L'instruction dup(df) permet d'affecter ou dupliquer le descripteur numéro 3 dans la case vide la plus basse, donc la case qui à pour index 0, on affecte donc le descripteur numéro 0 au fichier licence.txt.

**Etape 4**

0 (licence.txt)
1 (stdout)
2 (StdErr)
<del>3 (licence.txt)</del>

Close (df) permet de fermer le descripteur 3 et de libérer celui ci.  
On a donc au final un canal direct entre le clavier et le fichier licence.txt  
Et tout ce que je tape au clavier est entré directement dans le fichier.