

FR2I

Formation en Réseaux Internationaux d'Ingénieurs

















LINUX – LE NOYAU ACCÈS AUX FICHIERS





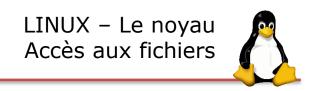












Descripteur de fichiers

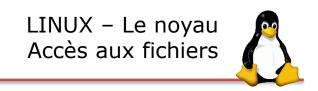
- Les entrées/sorties sont identifiés par un descripteur de fichier (nombre entier)
- Un programme hérite de trois fichiers ouverts dont les descripteurs sont :
 - 0 -> pour l'entrée standard (clavier) ou stdin
 - 1 -> pour la sortie standard (écran, terminal) ou stdout
 - 2 -> pour la sortie d'erreur standard (écran, terminal) ou stderr
- Quand le système ouvrira d'autres fichiers :
 - les descripteurs prendront pour valeur 3, 4...











Création d'un fichier

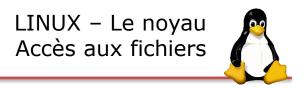
```
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
                             Nom du
#include inits.h>
                             fichier
                                                 Mode
main()
{
       int fd;
      fd = creat( "DATA", 0777);
       printf("File descriptor: %d\n", fd);
```











Les modes de création de fichiers

S_IRWXU|S_IRWXG|S_IRWXO

S_IRUSR

S_IRWXU

00700 user (file owner) has read, write and execute permission

S_IRUSR (S_IREAD)

00400 user has read permission

S_IWUSR (S_IWRITE)

00200 user has write permission

S_IXUSR (S_IEXEC)

00100 user has execute permission

S_IRWXG

00070 group has read, write and execute permission

S_IRGRP

00040 group has read permission

S IWGRP

00020 group has write permission

S IXGRP

00010 group has execute permission

S IRWXO

00007 others have read, write and execute permission

S IROTH

00004 others have read permission

S IWOTH

00002 others have write permission

S_IXOTH

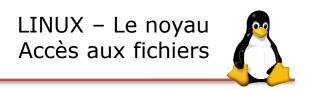
00001 others have execute permission











Retourne un descripteur de

Ouverture et fermeture d'un fichier

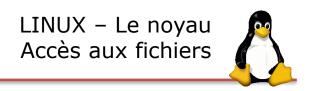
fichier (-1 si erreur) int open(path, flags) char *path; Ouvre un fichier selon le mode spécifié par int flags; flags: **O_RDONLY** Open for reading only **O_WRONLY** Open for writing only O_RDWROpen for reading and writing int open(path, flags, mode) char *path; **O_APPEND** Append on each write int flags; O CREAT Create file if it does not exist int mode; O_TRUNC Truncate size to 0 **O EXCL** Error if create and file exists **O_BLKINUSE** Block if file is in use close(fd) **O_BLKANDSET** Block if file is in use; int fd; then set in use **O_FSYNC** Do file writes synchronously



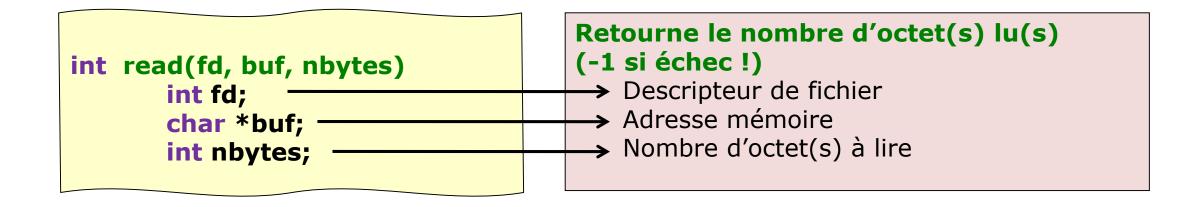


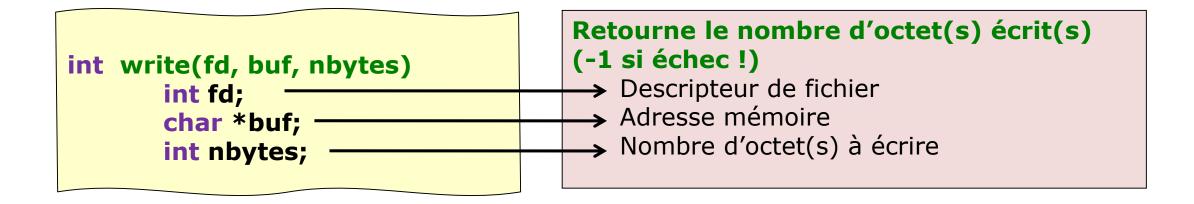






Lecture et écriture de fichier



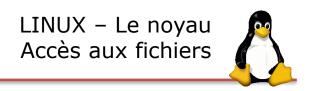












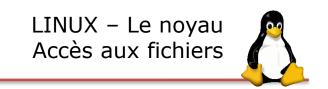
```
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <sys/uio.h>
#include <string.h>
#include <limits.h>
main()
    int fd;
    int cc;
    char *buf = "#file: DATA - date: Tue Sep 22 16:03:05 WET 1992 author:root\n";
    if ( ( fd = open( "DATA", O_CREAT|O_WRONLY, S_IRWXU ) )!= -1) {
         if ( ( cc = write( fd, buf, strlen(buf) ) != strlen(buf) ) {
                  printf("Erreur d'écriture dans le fichier\n");
         }
         printf("Nombre de caractères écrits : %d\n", cc);
         close(fd);
}
```









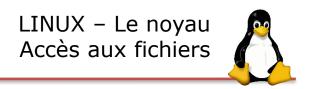












Positionnement dans un fichier

```
off_t lseek(fd, offset, whence)
int fd;
off_t offset;
int whence;
```

Positionner la « tête » de lecture/écriture dans un fichier

(retourne le nouvel emplacement, mesuré en octets depuis le début du fichier si succès, -1 si échec!)

Place la tête de lecture/écriture à la position **offset** dans le fichier associé au descripteur **fd** en suivant la directive **whence**

0 : début du fichier

1: position courante

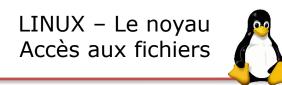
2 : fin du fichier

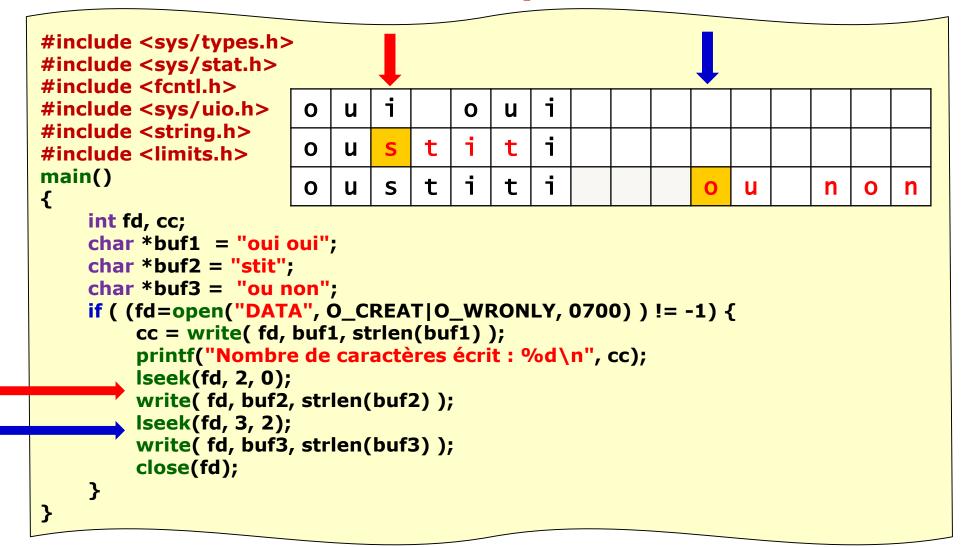










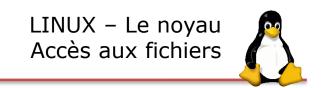










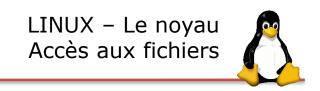












Autres fonctions...

```
chdir ( ... )
rmdir ( ... )
chmod ( ... )
chown ( ... )
chroot ( ... )
```