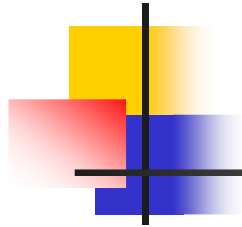


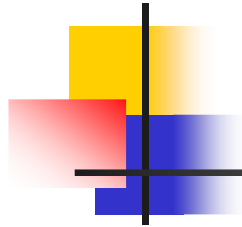
Gestion de fichiers

- Stockage de données (sur disques, disquettes, bandes..)
- Les fichiers ne sont pas des objets du langage C
- Utilisation de fonctions de la librairie standard
- 2 types de fichiers : binaires ou textes
 - (différenciés sous les systèmes Dos et Windows)
- Fichiers prédéfinis :
 - stdin : entrée standard (clavier)
 - stdout : sortie standard (écran)
 - stderr : fichiers d'erreurs (écran aussi)



Accès aux fichiers

- **Accès séquentiel** : lecture de tous les caractères depuis le premier jusqu'au caractère cherché.
- **Accès direct** : on se place directement sur l'information souhaitée.



Descripteur de fichier

- Un descripteur de fichier contient toutes les informations sur le fichier
 - Nom du fichier
 - Emplacement
 - Position courante
- Déclaration d'un pointeur :
`FILE *fichier ;`



Ouverture d'un fichier

```
FILE *fopen (char *nomfichier, char *mode) ;  
    Mode : r (lecture) w (écriture) a (ajouter)  
           b : mode binaire sur les PCs  
    Résultat : pointeur sur descripteur de fichier  
              NULL si ouverture impossible
```

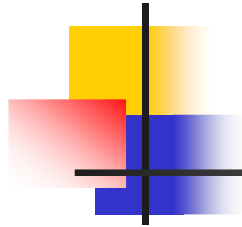
Exemple :

```
FILE *fichierEleve ;  
fichierEleve = fopen ("listeEleve.txt", "r");  
if (fichierEleve != NULL) {  
    ..  
}
```



Autres modes d'ouverture

- r : lecture
 - Le fichier doit exister
- r+ :
 - lecture et écriture (modification)
 - Le fichier doit exister
- w : écriture
 - Création du fichier s'il n'existe pas
 - Ecrasement du fichier s'il existe
- w+ : lecture et écriture
 - Création du fichier s'il n'existe pas
 - Ecrasement du fichier s'il existe
- a : écriture en fin de fichier
 - Le fichier est créé s'il n'existe pas
- a+ : écriture en fin de fichier et lecture du fichier
 - Le fichier est créé s'il n'existe pas



Fermeture de fichier

```
fclose (FILE *fichier);
```

- Fermer le fichier après utilisation

Exemple :

```
FILE *fich ;  
fich = fopen ("essai.txt", "w");  
if (fich != NULL) {  
    // Traitement du fichier  
    fclose (fich);  
}
```



Fin de fichier

- `int feof(FILE *descr) ;`
- Teste si la position courante est en fin de fichier.
- Résultat : 1 (VRAI) si c'est la fin de fichier et 0 (FAUX) dans le cas contraire.

```
while ( !feof (fich)) {  
    ..  
}
```



Lecture de chaîne de caractères

```
char *fgets (char *s, int max,  
             FILE *descr) ;
```

■ ***Exemple :***

```
char ligne [MAXLIGNE] ;  
FILE *fich;  
fich = fopen ("toto.txt", "r");  
/*Lecture de 1 ligne */  
if (fich != NULL) {  
    fgets (ligne, MAXLIGNE, fich);  
    ..  
}
```




Lecture de chaîne de caractères

```
int fscanf (FILE *descr,  
            char *format,  
            listeadresses,.) ;
```

- Même fonction que scanf mais les données sont lues dans le fichier à la place du clavier

```
fscanf (fich, "%d ", &valeur);
```



Ecriture avec fputs

```
int fputs (char *s, FILE *descr);
```

- Ecrit la chaîne s dans le fichier sans ajouter de \n
- Résultat : -1 si erreur

```
char message[] = "une ligne de texte ";  
fputs (message, fich);  
fputs ("fin ", fich);
```



Ecriture formatée

```
int fprintf (FILE *descr,  
             char *format,  
             listearguments) ;
```

- Résultat : nombre d'octets écrits, ou EOF si erreur.

```
fprintf (fich, " total %8.2f ", total);
```



Exemple – lecture de fichier

```
#define LGL 81          /* Longueur max d'une ligne de texte */
void main () {
    char ligne [LGL];
    FILE *fich;
    /* Ouverture du fichier a lire*/
    fich = fopen ("toto.txt", "r");
    if ( fich == NULL ) {
        printf (Ouverture impossible fichier %s\n", nomFichier);
    } else {
        printf ("Fichier\n",);
        fgets (ligne, LGL, fich); // Lecture 1ère ligne
        while (!feof(fich)) {      // Tant que non fin de fichier
            printf ("%s", ligne);  // Imprimer ligne à l'écran
            fgets (ligne, LGL, fich); // Lecture ligne suivante
        }
        fclose (fich); // fermeture du fichier
    }
}
```



Lecture et écriture de caractère

- Lecture du caractère courant dans le fichier.
`char fgetc(FILE *descr) ;`
- Ecriture d'un caractère dans le fichier.
`char fputc(char c, FILE *descr) ;`
- *Exemples :*

```
char c ;  
FILE *fich;  
fich = fopen ("toto.txt", "r");  
c = fgetc (fich) ;  
printf ("Caractère lu : %c\n" , c); ;
```



Exemple Copie de fichier

```
char c ;
FILE *entree, *sortie ;
entree = fopen ("fichier1.txt", "r");
sortie = fopen ("fichier2.txt", "w");
if (entree != NULL && sortie != NULL) {
    c = fgetc (entree) ;
    while ( !feof (entree)) {
        fputc (c, sortie)
        c = fgetc (entree)
    }
    fclose (entree) ;
    fclose (sortie) ;
}
```



Ecriture binaire

```
int fwrite (void *bloc, int taille,  
            int nb, FILE *descr) ;
```

Exemple :

```
FILE *fich ;  
int valeur ;  
....  
/*Ecriture de 1 entier */  
fwrite (&valeur, sizeof(int), 1, fich) ;
```

(pour les fichiers ouverts en mode w, a, r+, w+, a+)



Lecture binaire

```
int fread(void *bloc, int taille, int nb, FILE *descr) ;
```

Exemple :

```
FILE *fich ;
```

```
int valeur ;
```


```
....
```

```
/*Lecture de 1 entier */
```

```
fread (&valeur, sizeof(int), 1, fich) ;
```

(pour les fichiers ouverts en mode r,r+,w+,a+)

Traitement de fichier texte



```
#include <stdio.h>
#include <stdlib.h>
#define LGL 81      /* Longueur max d'une ligne de texte */
void main () {
    char ligne [LGL];
    char nomFichier [LGL];
    FILE *fich;
    /* Ouverture du fichier a lire*/
    printf("Nom du fichier a lire: ");
    fgets (ligne, LGL, stdin);
    sscanf (ligne, "%s", nomFichier);

    fich = fopen (nomFichier, "r");
    if ( fich == NULL ) {
        printf ("ERREUR d'ouverture du fichier %s\n", nomFichier);
    } else {
        printf ("Fichier %s \n", nomFichier);
        fgets (ligne, LGL, fich);      // Lecture 1ère ligne
        while (!feof(fich)) {         // Tant que non fin de fichier
            printf ("%s", ligne);
            fgets (ligne, LGL, fich); // Lecture ligne suivante
        }
        fclose (fich);
    }
}
```



Accès direct

```
int fseek(FILE *id, long nb, int mode) ;
```

Déplace le pointeur de nb octets,

Mode :

SEEK_SET : depuis le début du fichier

SEEK_CUR : depuis la position courante

SEEK_END : en arrière depuis la fin de
fichier.