Les fichiers

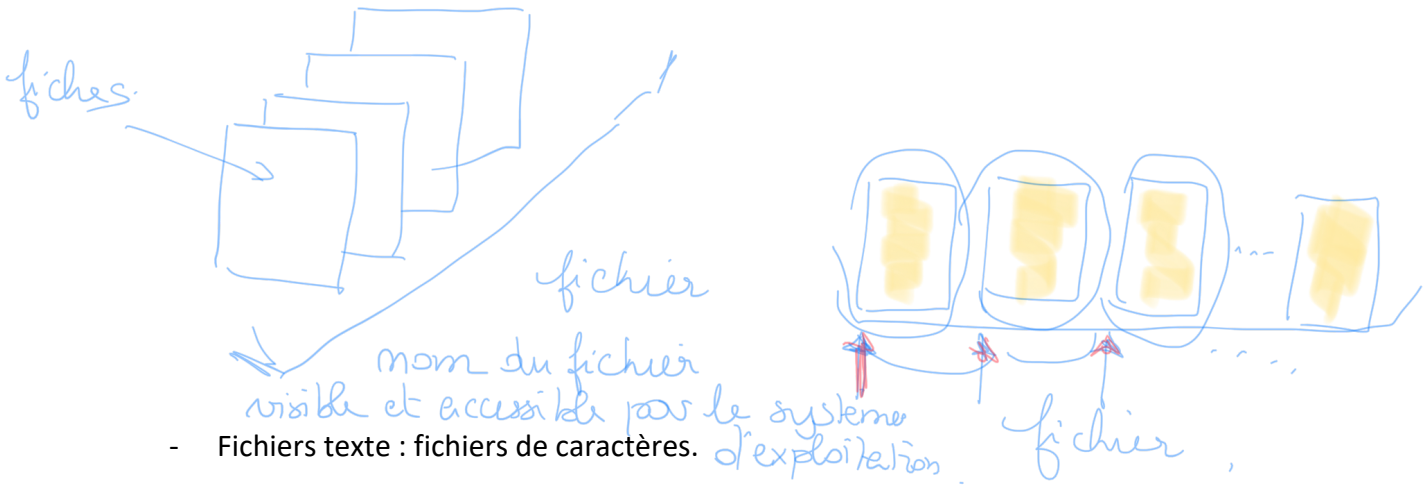
1. Définitions

Un fichier est une collection d'informations stockées sur un support physique (disque dur, bandes, disquettes, CD-ROM, DVD, ….).
C'est une suite d'informations binaires.

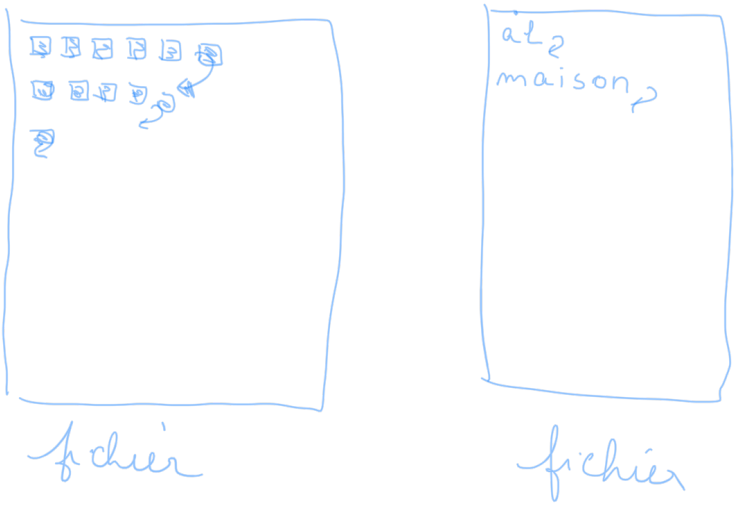
Il existe toutes sortes de fichiers selon les données qu'ils contiennent et le format dans lequel sont organisées ces données.
* des bases de données (sécu, gestion, finances, …)
* du texte (programme C, ..)
* des images (format gif, jpeg …..)
* du son
* des pages web (.htm, .html)
* des films

En LO01, nous traiterons 2 types de fichiers :

- Fichiers binaires structurés : ces fichiers contiennent le code binaire d'informations généralement de même type. On les appelle les enregistrements (ou fiches). Un enregistrement peut-être un ensemble de valeurs qui caractérisent un objet.



- Fichiers texte : fichiers de caractères.



Organisation des fichiers

L'organisation d'un fichier est la manière dont sont rangés physiquement les enregistrements du fichier. L'organisation est choisie à la création du fichier. Le choix d'une organisation correspond à un compromis entre rapidité d'accès, complexité de la mise en œuvre et type de stockage.

Organisation séquentielle :
On accède aux enregistrements les uns à la suite des autres en commençant par le début.
=> pour lire le 3ème enregistrement, il faut lire les deux précédents
=> l'insertion s'effectue en fin de fichier
=> peu exploitable pour les très gros fichiers

Organisation relative, accès direct :
Chaque enregistrement possède un numéro, on accède à la fiche par son numéro.
=> accès plus rapide à l'information.

Organisation indexée, accès séquentiel indexé :
Parcours séquentiel d'un index pour rechercher une clé.
=> obtention de l'adresse exacte de l'information recherchée
=> intéressant pour les fichiers importants en volume (bases de données)

En LO01 nous traiterons les fichiers à accès séquentiel

nom du fichier (système d'exploitation)
flot (pointeur) du fichier (programme)

2. Fonctions d'accès aux fichiers

FILE *fopen(char *nom, char *mode)

Ouverture : fopen()
`FILE* fopen(nom, mode)`
chaine de caractères

le flot est de type FILE *
↳ tête de lecture

La fonction retourne un élément de type `FILE*` (appelé stream ou flot ou pointeur de fichier).
Ce type est prédéfini dans stdio.h.
C'est l'élément qui est manipulé dans un programme.
Le nom est la chaine de caractères qui désigne le fichier (et son chemin d'accès) sur l'ordinateur.
Les modes :

- "r" : read => le fichier est existant et sera lu uniquement sinon retourne NULL.
- "w" : write => le fichier sera accédé en écriture, si le fichier existe déjà, il sera détruit.
- "a" : append => ajoute à la fin du fichier, s'il n'existe pas il sera créé.

Consultation
Création
ajout

Si une erreur à l'ouverture la fonction fopen() retourne NULL .
Il existe d'autres mode (r+, w+, a+…)
Dans les versions récentes du langage C, l'ouverture de fichiers binaires s'écrit avec les options "rb" , "wb" , "ab"

la fonction fopen permet d'établir le lien entre le fichier désigné par son nom et le pointeur du fichier manipulé par le programme.

par la suite, seul le pointeur de fichier sera utilisé
"r", "w", "a" → fichiers texte.
"rb", "wb", "ab" → fichiers structurés binaires

Exemple :

```
int main()
{ FILE *fp1, *fp2 ;/* fp1 et fp2 sont des pointeurs de fichiers */
fp1 = fopen ("notesLO01.dat", "r"); /* "rb" */
fp2 = fopen ("moyenne.txt", "w");
if (fp1 == NULL)
     printf ("le fichier notesLO01.dat n'existe pas");
if (fp2 == NULL)
     printf ("impossible de créer le fichier moyenne.txt ");
```

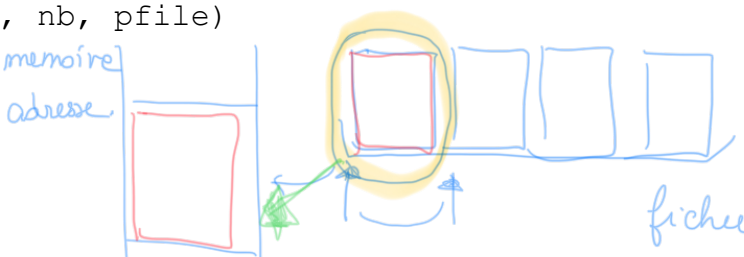
en mode "r" ou "rb", le pointeur est placé en début de fichier.

ATTENTION :
On ne peut pas ouvrir un fichier déjà ouvert.

<u>Lecture : fread(), fscanf()</u>

fread() permet de lire dans un fichier binaire structuré. Permet de lire en une seule lecture des données structurées.

```
fread( adresse, taille, nb, pfile)
```



adresse de la mémoire où l'élément lu dans le fichier sera copié
taille de l'élément (en octets)
sizeof(type) → donne le nombre d'octets occupés par un élément de type
nb : nb d'éléments (cas d'un tableau)
pfile : pointeur du fichier (tête de lecture)

fscanf() permet de lire dans un fichier texte

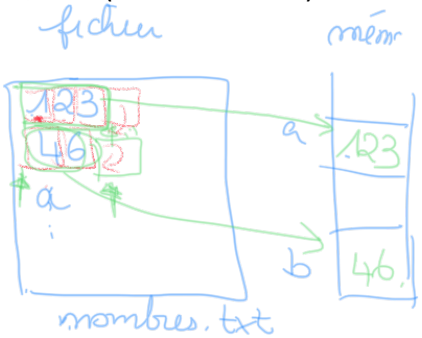
```
fscanf (pfile, "format", adresse des variables).
```

convertit les caractères lus dans un fichier texte (pfile) dans le format décrit par "format")
%d %c %s %f ;
le résultat est stocké à l'adresse donnée.
ex : int a, b;
fscanf (fpt, "%d %d", &a, &b);

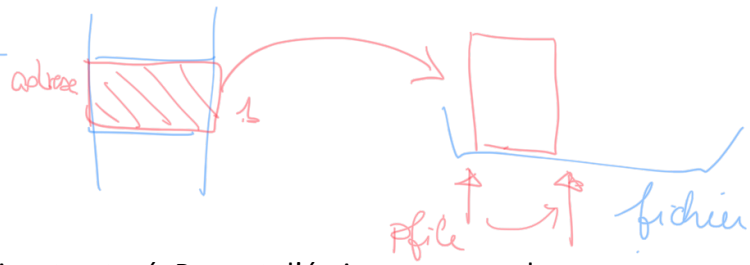
- se comporte comme un scanf
- utilisé uniquement sur un fichier ouvert en lecture

Ces fonctions lisent toujours par rapport à la position du pointeur du fichier (tête de lecture) et avancent la tête de lecture automatiquement

```
int a,b;
FILE *fpt;
fpt = fopen("nombres.txt","r");
fscanf(fpt,"%d %d",&a,&b);
```



écriture



Ecriture fwrite(fprintf()

fwrite() permet d'écrire dans un fichier binaire structuré. Permet d'écrire en une seule instruction des données structurées.

```
fwrite( adresse, taille, nb, pfile)
```

adresse de l'élément à écrire dans le fichier.
taille " " " " " " " "
nb (cas du tableau)
pfile : pointeur du fichier dans lequel on va copier (écrire) l'élément
(ne utilisation que fread)

fprintf() permet d'écrire dans un fichier texte

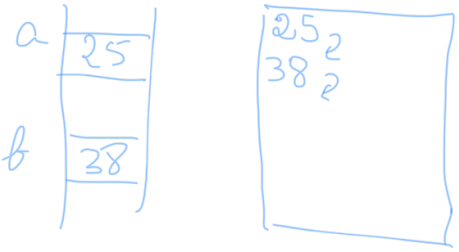
```
fprintf (pfile, "format", variables)
```

- se comporte comme un printf
- utilisé uniquement sur un fichier ouvert en ecriture

=> écrit toujours par rapport à la position du pointeur

Convertit grâce au format les contenus des variables en une suite de caractères copiée ensuite dans le fichier

```
int a=25, b=38;
FILE * fpt;
fpt = fopen ("fichier.txt", "w");
fprintf (fpt, "%d\n%d\n", a, b);
```



Ces fonctions ecrivent toujours à la position du pointeur du fichier (tête de lecture) et avancent la tête de lecture automatiquement

Les fichiers doivent être ouverts en mode écriture

Fermeture : fclose()

fclose (FILE*)

Fermeture du fichier => permet de terminer les accès à un fichier.
Le pointeur fp peut être réutilisé pour un autre fichier.
Le fichier peut-être ré-ouvert dans un autre mode.

fclose (pfile)

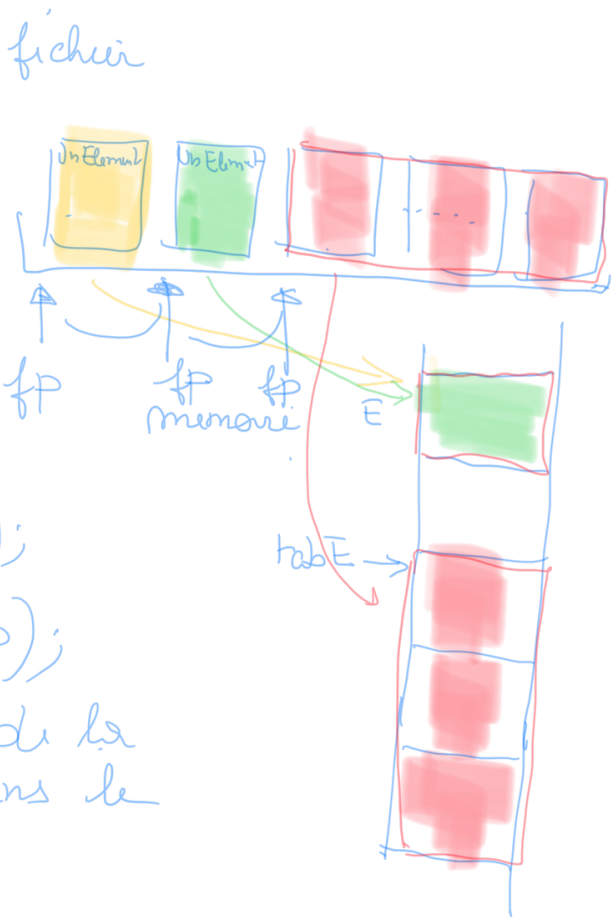
Test fin fichier : feof()

Cette fonction retourne 0 s'il n'est pas en fin de fichier, et une valeur non nulle sinon.

```
int feof(FILE*)
```

exemple d'utilisation du fread.

```
unElement E;
unElement tabE[3];
FILE *fp;
fp = fopen("------", "rb");
```




```
fread(&E, sizeof(unElement), 1, fp);
fread(&E, sizeof(unElement), 1, fp);
fread(tabE, sizeof(unElement), 3, fp);
```

le programmeur doit s'assurer de la compatibilité entre les infos dans le fichier et la mémoire.

```
fread(tabE, sizeof(unElement)*3, 1, fp)
```
(equivalent)

la fonction fread retourne le nombre d'octets lus si elle retourne 0, cela signifie qu'il n'y a plus d'elements à lire dans le fichier (fin de fichier)

```
if (fread(------) == 0)
    printf("fin de fichier");
```