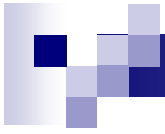




Cours de Langage C

Lecture & écriture

dans des fichiers



Les fichiers

- Un programme a en général besoin :
 - De lire des données (texte, nombres, images, sons, mesures, ...)
 - De sauvegarder des résultats (texte, nombres, images, sons, signaux générés, ...)

Cela se fait en lisant et en écrivant dans des fichiers

- Pour manipuler un fichier, on utilise un **pointeur** sur une donnée spécifique dont le type est **FILE** (structure prédéfinie que nous n'avons pas besoin de connaître précisément) :

```
FILE *fichier
```

- La variable **fichier** contiendra l'adresse en mémoire du début du fichier

Ouverture-fermeture de fichiers

- **Ouverture** d'un fichier à l'aide de la fonction **fopen** :

```
fichier = fopen("C:/Data/fichier1.txt", "r");
```

Cette fonction renvoie un pointeur sur le fichier ouvert.

fopen est définie dans le fichier stdio.h par :

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

• **nom** est une chaîne de caractères (tableau de caractères) contenant le nom du fichier ou bien un flot de données standard (stdin, ...)

• **mode** désigne le type de traitement des données

- **"r"** (read) : lecture (si le fichier existe)
- **"w"** (write) : écriture (le fichier est écrasé s'il existe et s'il n'existe pas, il est créé)
- **"a"** (append) : écriture à la fin d'un fichier existant

Mode
d'ouverture



Ouverture-fermeture de fichiers

Où se trouve le fichier ouvert ?

→ *Dans le répertoire de travail (plus exactement là où est le fichier exécutable .exe)*

Comment travailler sur un fichier situé ailleurs ?

→ *Le chemin absolu d'accès au fichier peut être donné in extenso (attention les \ de windows deviennent des / en C)*

```
fichier = fopen("C:/Data/fichier1.txt", "r");
```

```
fichier = fopen("fichier1", "r");
```

A quoi sert l'extension .txt ?

→ *A rien pour le Langage C.*

→ *Elle permet en revanche à l'OS de l'ordinateur de sélectionner le programme permettant d'ouvrir le fichier.*



Ouverture-fermeture de fichiers

- **Fermeture** d'un fichier à l'aide de la fonction **fclose** :

Important : ***Il faut toujours fermer un fichier après l'avoir utilisé
→ Afin de libérer la mémoire***

```
fclose(fichier) ;
```

fclose est définie dans le fichier stdio.h par :

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



Écriture-lecture de fichiers textes

• **Écriture :** `fprintf(FILE *fichier, char *proto, ...);`

Exemple : `double a;`
`fprintf(fichier, «%lf», a);`

→ *Presque même syntaxe que printf !*

`printf("%lf", a);`

• **Lecture :** `fscanf(FILE *fichier, char *proto, ...);`

Exemple : `double a;`
`fscanf(fichier, "%lf", &a);`

→ *Presque même syntaxe que scanf !*

`scanf("%lf", &a);`



Exemple : écriture dans un fichier

```
include <stdio.h>
void main()
{
    double a=1.5, b=2.5;
    FILE    *fichier;

    // Ouverture du fichier en écriture grâce à "w"
    fichier = fopen("essai.txt", "w");

    // Verifier que le fichier a bien été ouvert
    if (fichier != NULL)
    {
        // Ecriture
        fprintf(fichier, "%lf\n", a);
        fprintf(fichier, "%lf\n", b);
        // Fermeture du fichier
        fclose(fichier);
    }
}
```



Exemple : lecture à partir d'un fichier

```
#include <stdio.h>
void main()
{
    int      i;
    double   tab[2];
    FILE     *fichier;

    // Ouverture du fichier en lecture grâce à "r"
    fichier = fopen("essai.txt", "r");
    if (fichier != NULL)
    {
        for(i=0; i<2; i++)
            fscanf(fichier, "%lf\n", tab+i);
        fclose(fichier);
    }
    for(i=0; i<2; i++) printf("%lf\n", tab[i]);
}
```




Format d'un fichier

- Les fichiers précédents ont été écrits au format « **texte** »
 - On a utilisé jusqu'ici les fonctions : `fprintf` et `fscanf`
 - Tout se passe comme si l'affichage de la fenêtre console était envoyé dans le fichier.
 - Ou bien comme si le fichier se comportait comme les saisies clavier pour envoyer des données au programme.
 - Les caractères sont stockés avec le code ASCII : *American Standard Code for Information Interchange*
 - Avec un langage évolué tel que le C :
 - Nous n'avons pas à connaître le code ASCII
 - Nous n'avons pas à connaître la représentation des nombres dans les mémoires



Format d'un fichier

- Lorsqu'on a des **données numériques**, il est souvent plus efficace de recopier directement le contenu de la mémoire donc de les écrire sous forme « **binaire** ».

- Soit un nombre **1.345643355454E-18**
 - Au format **double**, il occupe 8 octets
 - Au format **ASCII**, il occupe 20 octets (1 octet par caractère)

- On a 2 types de fichiers : fichiers « **texte** » et « **binaire** ».
- Le format binaire va occuper beaucoup moins de place en mémoire
- Comme il s'agit d'une copie directe de la mémoire, on n'a pas à savoir comment est codé chaque nombre

- **Modes d'ouverture de fichiers binaires :**

- "**rb**" (read) : lecture
- "**wb**" (write) : écriture (le fichier est écrasé s'il existe)
- "**ab**" (append) : écriture à la fin d'un fichier existant

Écriture-lecture de fichiers binaires

• **Ecriture** d'un **bloc** de données en binaire

int **fwrite**(void *source, int taille_type, int nombre, FILE *flot)

fwrite :

- Écrit tout un bloc de données en un seul appel
- Retourne un entier = nombre d'éléments effectivement écrits

Exemple :

fwrite(**tableau**, **sizeof(float)**, **DIM**, **fichier**) ;

Pointeur sur les données

• **Lecture** d'un **bloc** de données en binaire

int **fread**(void *destination, int taille_type, int nombre, FILE *flot)

fread :

- Lit un bloc de données en un seul appel
- Retourne un entier = nombre d'éléments effectivement lus

Exemple :

fread(**tableau**, **sizeof(float)**, **DIM**, **fichier**) ;

Pointeur sur les données

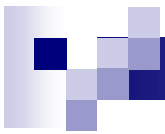
Pointeur sur le fichier

Pointeur sur
le fichier



Exemple : fichiers binaires

```
#define DIM 10000
void main()
{
    int    i;
    double sum, tab1[DIM], tab2[DIM];
    FILE   *fichier;
    // Remplissage du tableau
    for(i=0; i<DIM; i++)
        tab1[i] = i * atan(1);
    // Ecriture du fichier au format binaire
    fichier = fopen("essai2.bin", "wb");
    if (fichier != NULL)
    {
        fwrite(tab1, sizeof(double), DIM, fichier);
        fclose(fichier);
    }
    // Lecture du fichier
    fichier = fopen("essai2.bin", "rb");
    if (fichier != NULL)
    {
        fread(tab2, sizeof(double), DIM, fichier);
        fclose(fichier);
    }
}
```



Comparaison : fichiers binaires et fichiers textes

	Fichiers textes	Fichiers binaires
Taille	Peu compacte	compacte
Lisible avec un programme courant	oui	non
Lisible avec un programme spécifique	oui	oui
Écriture/Lecture par blocs	non	oui
Fonctions	fopen(mode r,w ou a) fclose() fprintf() fscanf()	fopen(mode rb, wb ou ab) fclose() fwrite() fread()