Le langage C - Les fichiers

ASR2 - Système

Semestre 2, année 2012-2013

Mars 2013

Haut niveau / Bas niveau

Deux possibilités pour manipuler les fichiers en C :

Descripteurs de fichiers (bas niveau)

- La référence est un entier (`int monfichier`)
- `open()`, `close()`, `read()`, `write()`,*etc.*
- Correspondent à des appels systèmes

Pointeurs de flux (haut niveau, équiv. stream C++)

- La référence est un `FILE *`
- `fopen()`, `fclose()`, `fgetc()`, `fputc()`,*etc.*
- Principaux avantages : portable (norme C ANSI) et optimisé (p.ex. tampons d'écriture)

$\rightarrow$ Aujourd'hui : pointeurs de flux (on dit aussi pointeurs de fichiers).

Ouverture/fermeture

Pour lire ou écrire dans un fichier, on commence donc par créer le pointeur de flux. L'ouverture d'un fichier renvoie le pointeur correspondant, qui est ensuite passé aux différentes fonctions appelées.

```c
#include <stdio.h>

FILE *fopen(const char *path, const char *mode);
int fclose(FILE *fp);
```

fopen

Ouvre un flux vers le fichier passé en paramètre (chemin). Le second paramètre *(mode) peut être par exemple "r" (read), "w" (write), ou "a" (append) selon ce que l'on souhaite faire.*

fclose

Ferme le flux passé en paramètre.

Lecture/écriture

Lire et écrire dans un fichier est aussi simple que de lire sur l'entrée standard (clavier) et d'écrire sur la sortie standard (écran).

```c
#include <stdio.h>

int fprintf(FILE *stream, const char *format, ...);
int fscanf(FILE *stream, const char *format, ...);

int fgetc(FILE *stream);
int fputc(int c, FILE *stream);
```

Entrée/sortie standards

Par défaut, le système vous permet d'utiliser des pointeurs de flux prédéfinis qui représentent l'entrée standard (*stdin*), la sortie standards (*stdout*) et la sortie d'erreur (*stderr*).

P. ex : `fgetc(stdin)` renvoie un caractère lu depuis le terminal.

Exemple

En fait, ces 2 versions sont identiques.

```
char *filename = malloc(256*sizeof(char));
```

```
printf("File name : ");
scanf("%s", filename);
```

```
fprintf(stdout, "File name : ");
fscanf(stdin, "%s", filename);
```

```
FILE *stream = fopen(filename, "r");
if (stream == NULL){
   fprintf(stderr, "Cannot open %s\n", filename);
   free(filename);
   return EXIT_FAILURE;
}

for(char c = fgetc(stream) ; c != EOF ; c = fgetc(stream))
   printf("%c", c);

fclose(stream);
free(filename);
return EXIT_SUCCESS;
```

Exemple

En fait, ces 2 versions sont identiques.

```
char *filename = malloc(256*sizeof(char));
```

```
printf("File name : ");
scanf("%s", filename);
```

```
fprintf(stdout, "File name : ");
fscanf(stdin, "%s", filename);
```

```
FILE *stream = fopen(filename, "r");
if (stream == NULL){
   fprintf(stderr, "Cannot open %s\n", filename);
   free(filename);
   return EXIT_FAILURE;
}

for(char c = fgetc(stream) ; c != EOF ; c = fgetc(stream))
   printf("%c", c);

fclose(stream);
free(filename);
return EXIT_SUCCESS;
```

The Mission !


NATIONAL SECURITY AGENCY
UNITED STATES OF AMERICA


OIPC ICPO
INTERPOL


DIRECTION CENTRALE DU RENSEIGNEMENT INTERIEUR
DCRI


DEPARTMENT OF JUSTICE