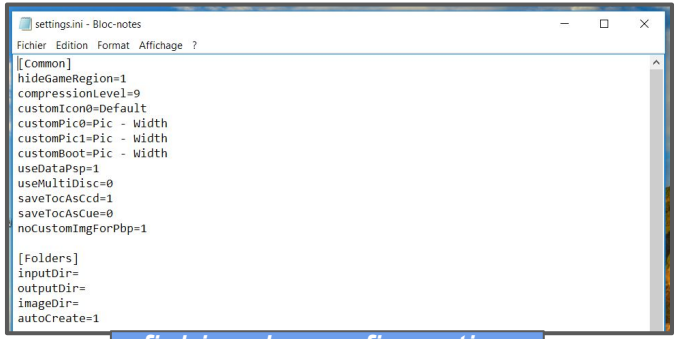
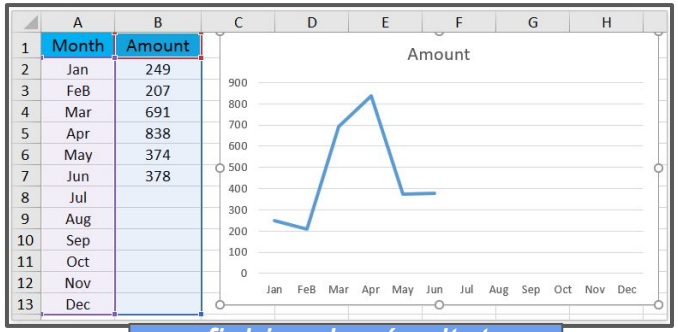
Programmation C

17 - Gestion des fichiers

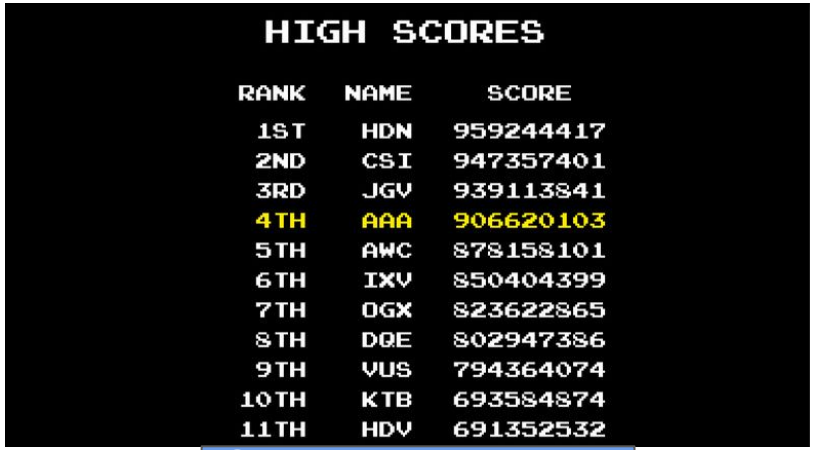
Utilisation des fichiers



fichier de configuration



fichier de résultat



Sauvegarde des scores

stdio.h

Ouvrir un fichier

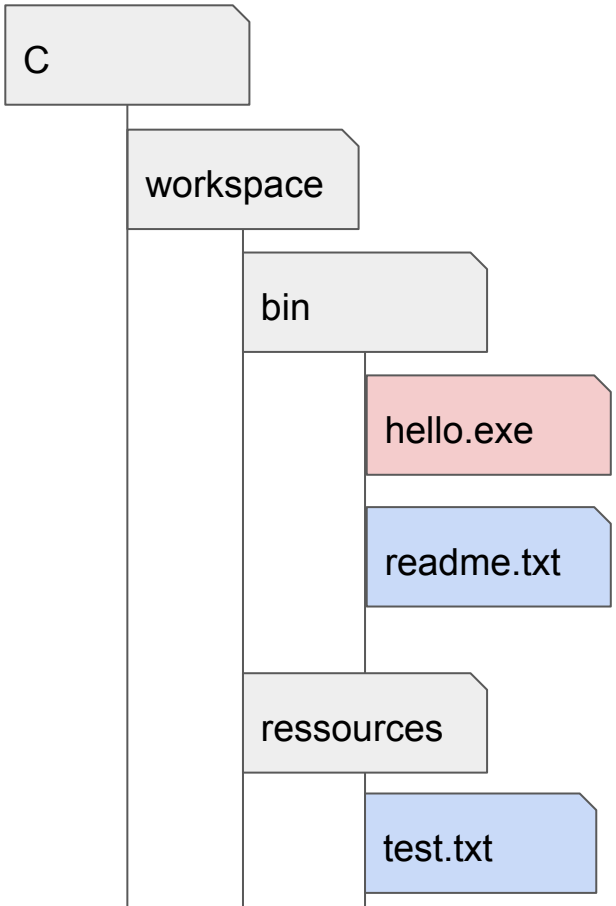
FILE* fichier **fopen(** chemin du fichier **,** mode **)**

mode	nom	description
“r”	Lecture seule	Lecture possible mais pas l’écriture. Si le fichier n’existe pas, retourne NULL.
“w”	Écriture seule	Ecriture possible mais pas la lecture. Si le fichier n’existe pas, il sera créé automatiquement.
“a”	Ajout	Ajoute du contenue a la fin du fichier. Si le fichier n’existe pas, il sera créé automatiquement.
“r+”	Lecture et Écriture	Vous pourrez lire et écrire dans le fichier. Si le fichier n’existe pas, retourne NULL.
“w+”	Effacement + Lecture et Écriture	Comme pour l’option “w” mais ici, en plus, le fichier est vidée de son contenue à l’ouverture.
“a+”	Ajout en Lecture et Écriture	Comme pour l’option “a” mais avec en plus la possibilitée de lire le contenue.

Retourne NULL en cas d’échec

Ajouter ‘b’ à ces modes pour traiter le fichier en binaire (ex “br”)

Chemin Absolu et relatif



Chemin Absolu:
chemins complet depuis la racine du volume (disque).

Chemin Relatif:
chemin relatif à la position courante lorsque le binaire est exécuté.

Exemple:

readme.txt:
- absolu: c:/workspace/bin/readme.txt
- relatif: ./readme.txt ou readme.txt

test.txt
- absolu: c:/workspace/ressources/test.txt
- relatif: ../ressources/test.txt

stdio.h

Fermer un fichier

int result **fclose(** FILE* fichier **)**

```
FILE* pt_fichier= fopen("readme.txt", "r");

if (pt_fichier == NULL)
{
   printf("Impossible d'ouvrir le fichier readme.txt");
}
else
{
   int resultat = fclose(pt_fichier);

   if(resultat  == EOF)
      printf(“Erreur lors de la fermeture du fichier readme.txt”);
}
```

Exemple

- fclose, retour 0 si tout est ok sinon EOF

- EOF est un define de stdio.h qui signifie soit erreur soit fin de fichier)

- Dans le cas de fclose, EOF signifie erreur

stdio.h

Ouvrir et fermer un fichier

FILE* fichier **fopen(** chemin du fichier **,** mode **)**

mode	nom	description
“r”	Lecture seule	Lecture possible mais pas l’écriture. Si le fichier n’existe pas, retourne NULL.
“w”	Écriture seule	Ecriture possible mais pas la lecture. Si le fichier n’existe pas, il sera créé automatiquement.
“a”	Ajout	Ajoute du contenue a la fin du fichier. Si le fichier n’existe pas, il sera créé automatiquement.
“r+”	Lecture et Écriture	Vous pourrez lire et écrire dans le fichier. Si le fichier n’existe pas, retourne NULL.
“w+”	Effacement + Lecture et Écriture	Comme pour l’option “w” mais ici, en plus, le fichier est vidée de son contenue à l’ouverture.
“a+”	Ajout en Lecture et Écriture	Comme pour l’option “a” mais avec en plus la possibilitée de lire le contenue.

int result **fclose(** FILE* fichier **)**

stdio.h

Ecrire un caractère dans un fichier

int result **fputc(** char caractère **,** FILE* fichier **)**

```
FILE* pt_fichier= fopen("readme.txt", "w");

if( fputc('X', pt_fichier) != 0 )
{
        printf("Erreur lors de l'écriture");
}
```

Exemple

- Retournes 0 si tout est ok sinon EOF
- EOF est un define de stdio.h qui signifie soit erreur soit fin de fichier)

stdio.h

Ecrire une chaîne dans un fichier

int result **fputs(** char* chaîne **,** FILE* fichier **)**

```
FILE* pt_fichier= fopen("readme.txt", "w");

if( fputs(“Salut”, pt_fichier) != 0 )
{
        printf(“Erreur lors de l’écriture”);
}
```

Exemple

- Retournes 0 si tout est ok sinon EOF
- EOF est un define de stdio.h qui signifie soit erreur soit fin de fichier)

stdio.h

Ecrire une chaîne formatée dans un fichier

int result **fprintf(** FILE* fichier **,** chaîne formatée **,** liste params **)**

```
FILE* pt_fichier= fopen("readme.txt", "w");

int high_score = 12354;
int resultat = fprintf(pt_fichier, "Mon score: %d points\n", high_score);

if( resultat  != 0 )
{
        printf(“Erreur lors de l’écriture”);
}
```

Exemple

- Retournes 0 si tout est ok sinon EOF
- EOF est un define de stdio.h qui signifie soit erreur soit fin de fichier)

stdio.h

Ecrire dans un fichier

int result **fputc(** char caractère **,** FILE* fichier **)**

int result **fputs(** char* chaîne **,** FILE* fichier **)**

int result **fprintf(** FILE* fichier **,** chaîne formatée **,** liste params **)**

- Retournes 0 si tout est ok sinon EOF
- EOF est un define de stdio.h qui signifie soit erreur soit fin de fichier)

stdio.h

Lire un caractère dans un fichier

int result **fgetc(** FILE* fichier **)**

```
FILE* pt_fichier= fopen("readme.txt", "r");

int resultat = fgetc(pt_fichier);
if( resultat  == EOF )
        printf(“Erreur lors de la lecture ou fin de fichier”);
else
        printf(“mon char = %c”, (char)resultat );
```

Exemple

Retourne EOF en cas de fin de fichier, sinon le résultat peut être casté en char pour afficher la lettre lue.

Déplacement du curseur de lecture

Salut les codeurs,

bienvenue dans cette formation sur le C

Fichier

- Lorsque le curseur de lecture arrive en fin de fichier, il retourne EOF

stdio.h

Lire une chaîne dans un fichier

char* result **fgets(** char* chaîne **,** int nb_char **,** FILE* fichier **)**

```
FILE* pt_fichier= fopen("readme.txt", "r");

char tab[100];
char* resultat = fgets(tab, 50, pt_fichier);

if( resultat  == NULL)
        printf(“Erreur lors de la lecture ou fin de fichier”);
else
        printf(“mon string = %s”, tab);
```

Exemple

Retourne NULL si il y a une erreur

stdio.h

Ecrire une chaîne formatée dans un fichier

int result **fscanf(** FILE* fichier **,** chaîne formatée **,** liste pointeurs **)**

```
FILE* pt_fichier= fopen("readme.txt", "r");

int prix= 0;
int resultat = fscanf(pt_fichier, “%d”, &prix);

if( resultat  == EOF )
        printf(“Erreur lors de la lecture ou fin de fichier”);
else
        printf(“Prix de l’article = %d”, prix);
```

Exemple

Si aucune donnée ne peut être extraite, alors la valeur EOF vous sera retournée.
Sinon, le nombre de paramètres correctement extraits vous sera renvoyé

les espaces et retours à la ligne sont considérés comme des séparateurs

stdio.h

Lire dans un fichier

int result **fgetc(** FILE* fichier **)**

char* result **fgets(** char* chaîne **,** int nb_char **,** FILE* fichier **)**

int result **fscanf(** FILE* fichier **,** chaîne formatée **,** liste pointeurs **)**

stdio.h

Position du curseur de fichier

long result **ftell(** FILE* fichier **)**

```
FILE* pt_fichier= fopen("readme.txt", "r");

printf("Position = %ld", ftell(pt_fichier ) );

fgetc(pt_fichier);
fgetc(pt_fichier);
printf("Position = %ld", ftell(pt_fichier ) );
```

Exemple

```
Position = 0
Position = 2
```

Retourne la position du pointeur ou -1 en cas d'erreur

stdio.h

Positionner le curseur en début de fichier

void **rewind(** FILE* fichier **)**

```
FILE* pt_fichier= fopen("readme.txt", "r");

printf("Position = %d", ftell(pt_fichier ) );

fgetc(pt_fichier);
fgetc(pt_fichier);

rewind(pt_fichier);
printf("Position = %d", ftell(pt_fichier ) );
```

Exemple

```
Position = 0
Position = 0
```

stdio.h

Déplacer le curseur dans le fichier

int result **fseek(** FILE* fichier , long déplacement , position_origine **)**

position_origine	description
SEEK_SET	Début du fichier
SEEK_CUR	Position courante du curseur.
SEEK_END	Fin du fichier

```
FILE* pt_fichier= fopen("readme.txt", "r");

printf("Position = %d", ftell(pt_fichier ) );

fgetc(pt_fichier);
fgetc(pt_fichier);

fseek(pt_fichier, 0, SEEK_SET);
printf("Position = %d", ftell(pt_fichier ) );
```

Exemple

```
Position = 0
Position = 0
```

stdio.h

Se déplacer dans un fichier

long result **ftell(** FILE* fichier **)**

void **rewind(** FILE* fichier **)**

int result **fseek(** FILE* fichier **,** long déplacement **,** int position_origine **)**

stdio.h

Renommer/déplacer un fichier

int result **rename(** char* ancien_chemin **,** char* nouveau_chemin **)**

```
int resultat = rename("mon_fichier.txt", "fichier.txt");

if(resultat != 0)
   printf("Erreur lors du renommage !" );
```

Renommer

```
int resultat = rename("fichier.txt", "doc/fichier.txt");

if(resultat != 0)
   printf("Erreur lors du déplacement !" );
```

Déplacer

stdio.h

Supprimer un fichier

int result **remove(** char* chemin **)**

```
int resultat = remove("mon_fichier.txt");

if(resultat != 0)
   printf("Erreur lors de la suppression!" );
```

Exemple

Attention: le fichier est définitivement supprimé.