# TP CSE : Entrées et Sorties Bufferisées

Generated by Doxygen 1.8.11

# Contents

# Chapter 1

# Lectures et Écriture bufferisées

Barona Stephanie, Grand Maxence

### Compilation

```
1 make all
```

### Exécution

#### main

Ce programme lit un fichier donné comme premier argument, et écrit le contenu lu dans un autre fichier donné comme deuxième argument. Ce programme est compilé normalement.

```
1 ./main <filename1> <filename2>
```

- filename1 : Le fichier à lire
- filename2 : Le fichier à écrire

#### main_static

Ce programme lit un fichier donné comme premier argument, et écrit le contenu lu dans un autre fichier donné comme deuxième argument. Ce programme est compilé avec une bibliothèque statique.

```
1 ./main_static <filename1> <filename2>
```

- filename1 : Le fichier à lire
- filename2 : Le fichier à écrire

#### main_dyn

Ce programme lit un fichier donné comme premier argument, et écrit le contenu lu dans un autre fichier donné comme deuxième argument. . Ce programme est compilé avec une bibliothèque dynamique.

```
1 ./main_dyn <filename1> <filename2>
```

- filename1 : Le fichier à lire
- filename2 : Le fichier à écrire

#### generator

Ce programme écrit un fichier au contenu et à la taille aléatoire.

```
1 ./generator <filename>
```

- filename : Le fichier cible

#### test_format

Ce programme écrit une chaîne formatée dans un fichier, puis la relis et verifie que les valeurs lues sont toujours les mêmes.

```
1 ./test_format <filename1>
```

- filename1 : Le fichier où sera écrite puis lue la chaîne formatée

## Tests

```
1 make tests
```

Le tests sont effectuées par le script script_test.sh :

- Test les fonctions d'écritures et lectures bufférisées avec le programme test_format
- Génère des fichiers au contenu et à la taille aléatoire, donne ce fichier en entré du programme main (resp main_static main_dyn) et vérifie que le fichier <filename2> est identique au fichier <filename1>

# Chapter 2

# Data Structure Index

## 2.1 Data Structures

Here are the data structures with brief descriptions:

# Chapter 3

# File Index

## 3.1 File List

Here is a list of all documented files with brief descriptions:

# Chapter 4

# Data Structure Documentation

## 4.1 bfile Struct Reference

### Data Fields

- FILE * **f**
- char **mode**
- char * **buffer**
- unsigned **file_seek**
- unsigned **buffer_seek**
- size_t **size_buffer**
- unsigned **eof**

### 4.1.1 Detailed Description

pour les écritures et lectures bufferisées

**Parameters**

| | |
|---|---|
| *f* | : le fichier |
| *mode* | : Mode de lecteur, E pour écriture, L pour lecture |
| *buffer* | : Notre tampon |
| *file_seek* | : Curseur de f |
| *file_seek* | : Curseur de buffer |
| *size_buffer* | : taille du buffer |
| *eof* | : booléen indiquant si il reste des choses 'a lire dans le fichier |

The documentation for this struct was generated from the following file:

- bfile.c

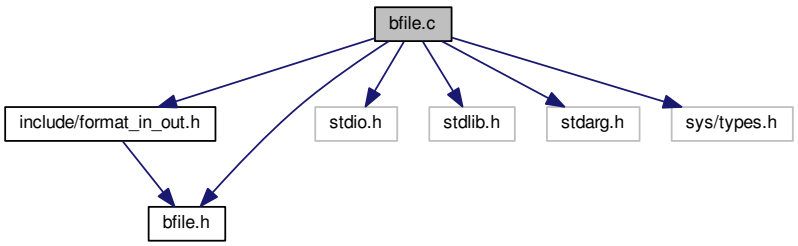
# Chapter 5

# File Documentation

## 5.1 bfile.c File Reference

```
#include "include/format_in_out.h"
#include "include/bfile.h"
#include <stdio.h>
#include <stdlib.h>
#include <stdarg.h>
#include <sys/types.h>
```

Include dependency graph for bfile.c:




### Data Structures

- struct bfile

### Functions

- bfile * bOpen (const char *path, char mode)

  *Le but de la fonction est d'ouvrir le fichier path avec le mode d'ouverture en lecture 'L' ou en écriture 'E' et d'allouer la structure de bfile avec son tampon et son fichier. La fonction renvoie NULL si le fichier ne peut pas être ouvert, un bfile sinon.*
- int bClose (bfile *bf)

  *Le but de la fonction est de fermer et libérée le fichier pointé sur la structure de donnée de type bfile.*

- int bWrite (void *p, int size, int nb_element, bfile *bf)

  *Le but de la fonction est d'écrire nb_element de size octets stockés à l'emplacement mémoire pointé par p, dans le tampon contenu dans bf.*
- int bRead (void *p, int size, int nb_element, bfile *bf)

  *Le but de la fonction est de lire nb_element de size octets dans le tampon contenu dans bf et y stocker à l'emplacement mémoire pointé par p/.*
- int bFlush (bfile *bf)

  *Le but de la fonction est xde vider le tampon dans le fichier f.*
- int bFill (bfile *bf)

  *Le but de la fonction est de remplir le tampon avec le fichier f.*

### 5.1.1 Detailed Description

émentation des écritures et lectures bufférisées

### 5.1.2 Function Documentation

#### 5.1.2.1 int bClose ( bfile * *bf* )

Le but de la fonction est de fermer et libérée le fichier pointé sur la structure de donnée de type bfile.

**Parameters**

| | |
|---|---|
| *bf* | : la structure à ferme |

**Returns**

Un entier retourné par la fonction fclose

#### 5.1.2.2 int bFill ( bfile * *bf* )

Le but de la fonction est de remplir le tampon avec le fichier f.

**Parameters**

| | |
|---|---|
| *bf* | la structure bfile |

**Returns**

Le nombre d'octet lu dans le fichier

#### 5.1.2.3 int bFlush ( bfile * *bf* )

Le but de la fonction est xde vider le tampon dans le fichier f.

**Parameters**

| | |
|---|---|
| *bf* | la structure bfile |

**Returns**

Le nombre d'octet écrit dans le fichier

#### 5.1.2.4 bfile * bOpen ( const char * *path*, char *mode* )

Le but de la fonction est d'ouvrir le fichier path avec le mode d'ouverture en lecture 'L' ou en écriture 'E' et d'allouer la structure de bfile avec son tampon et son fichier. La fonction renvoie NULL si le fichier ne peut pas être ouvert, un bfile sinon.

**Parameters**

| | |
|---|---|
| *path* | : le nom du fichier |
| *mode* | : le mode d'ouverture |

**Returns**

le bfile créé

#### 5.1.2.5 int bRead ( void * *p*, int *size*, int *nb_element*, bfile * *bf* )

Le but de la fonction est de lire nb_element de size octets dans le tampon contenu dans bf et y stocker à l'emplacement mémoire pointé par p/.

**Parameters**

| | |
|---|---|
| *p* | : le pointeur à remplir |
| *size* | : la taille des données à lrie |
| *nb_element* | : le nombre de fois que nous lisons. |
| *bf* | la structure bfile |

**Returns**

Le nombre d'octets lus.

#### 5.1.2.6 int bWrite ( void * *p*, int *size*, int *nb_element*, bfile * *bf* )

Le but de la fonction est d'écrire nb_element de size octets stockés à l'emplacement mémoire pointé par p, dans le tampon contenu dans bf.

**Parameters**

| | |
|---|---|
| *p* | : le pointeur à écrire |
| *size* | : la taille des données à écrire |
| *nb_element* | : le nombre de fois où on écrit. |
| *bf* | la structure bfile |

**Returns**

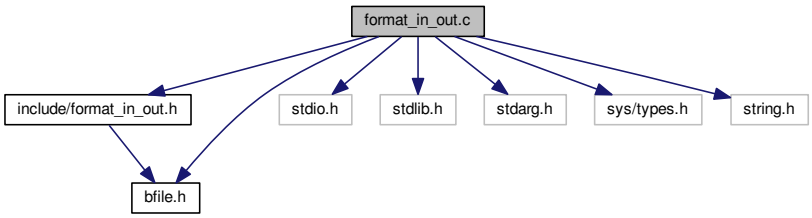
le nombre d'éléments écrits

## 5.2 format_in_out.c File Reference

Implémentation des écritures et lectures formatées.

```
#include "include/format_in_out.h"
#include "include/bfile.h"
#include <stdio.h>
#include <stdlib.h>
#include <stdarg.h>
#include <sys/types.h>
#include <string.h>
```

Include dependency graph for format_in_out.c:




### Functions

- char * convert_int_to_str (int x)

  *Le but de la fonction est de convertir un entier en chaîne.*
- unsigned is_separator (char c)

  *Le but de la fonction est de vérifier qu'un caractère est un séparateur ou non.*
- int fbWrite (bfile *bf, char *format,...)

  *Le but de la fonction est d'écrire la chaîne format dans bf->f en remplaçant les % par les valeurs des variables.*
- int **fbRead** (bfile *bf, char *format,...)

### 5.2.1 Detailed Description

Implémentation des écritures et lectures formatées.

### 5.2.2 Function Documentation

#### 5.2.2.1 char * convert_int_to_str ( int *x* )

Le but de la fonction est de convertir un entier en chaîne.

**Parameters**

| | |
|---|---|
| *x* | : l'entier à converir en chaîne |

**Returns**

Une chaîne de caractère

#### 5.2.2.2 int fbWrite ( bfile * *bf*, char * *format*, ... )

Le but de la fonction est d'écrire la chaîne format dans bf->f en remplaçant les % par les valeurs des variables.

**Parameters**

| | |
|---|---|
| *bf* | : la structure du fichier bufférisé |
| *format* | : La chaîne formatée |
| ... | : Les différentes variables à écrire |

**Returns**

Le nombre d'octet écrit

#### 5.2.2.3 unsigned is_separator ( char *c* )

Le but de la fonction est de vérifier qu'un caractère est un séparateur ou non.

**Parameters**

| | |
|---|---|
| *c* | : le caractère à tester |

**Returns**

Un booléen, true si c est séparateur, false sinon

# Index