BasicOS — File system

Handed on friday 2nd of December 2022

Group :
Ahmed Ghaleb — ahmed.ghaleb@eurecom.fr
Guillaume Ung — guillaume.ung@eurecom.fr
William Chieu — william.chieu@eurecom.fr

Table des matières

1 Introduction **2**

2 General approach **2**

3 Utility functions **3**

4 Main commands **3**

4.1 create 3

4.2 write 3

4.3 ls 3

4.4 read 3

4.5 size 3

4.6 remove 3

5 Conclusion **4**

1 Introduction

In the context of the BasicOS course taught by Pr. Ludovic Apvrille at EURECOM we had to design a basic file system [1]. The feature of this file system is that it handles a no space left on device error.
What is a file system ? A file system is a method and data structure that the operating system uses to control how data is stored and retrieved.

Our program has been written in C and has a set of commands to manipulate files and the file system (`create`, `write`, `read`, `ls`, `size`, `remove`). All the data is stored in a file on the host computer.

The usage instructions are in written in the file `README.md`.

2 General approach

To design our file system we have decided to have one **superblock** and a fixed number of **inodes**. We thought that it would make it simpler for us.

TABLE 1 – Structure

Superblock	**Inodes**	**Datablocks**
Fixed #	Fixed #	Variable #
32B	10 000 * 12B	x * 512B

The superblock is a C structure containing fields like ; **`int size, int dbcount`**.
The inodes are C structures containing information about the file like ; **`char filename, int inodenumber, timestamps`**.
The datablocks are placed after the inodes in the file system, and we decided for the sake of simplicity that we would not have fragmentation. All the datablocks linked to a file are contiguous.
Functions for each command are written in separate files. **`src/myfs.c`** was mainly used for argument parsing and function calling.
We also had to code a set of utility functions to facilitate and avoid code duplication.

Function repartition
- Guillaume : **`read, remove`**
- William : **`create, ls`**
- Ahmed : **`write, size`**

1. cf. `https://perso.telecom-paristech.fr/apvrille/BasicOS/project.html`

3 Utility functions

Here is a list of the functions which were used to make the program.

```
inode_t get_inode(); // given a filename, returns the corresponding
    inode
int get_free_inode(); // given a table, returns index of free inode
int get_free_db(); // returns ptr to a given # of contiguous datablocks
int load_inodes(); // loads inodes of the FS in a data structure
int myfs_load(); // loads the FS into data structures
int myfs_init(); // budget version of create, used for tests
```

Utility functions

4 Main commands

4.1 create

4.2 write

4.3 ls

4.4 read

4.5 size

4.6 remove

5 Conclusion