MiniFS Introduction

Introduction

In this assignment, we need to implement:

- A mini file system
- Operations in this file system

The structure of MiniFS

A file system requires at least:

1. **Virutal Disk Image**: The raw strorage medium (a file, `disk.img`).
2. **VCB(Volume Control Block)**
3. **FCB(File Control Block)**
4. **File and Directory**

0. Partition and Volume

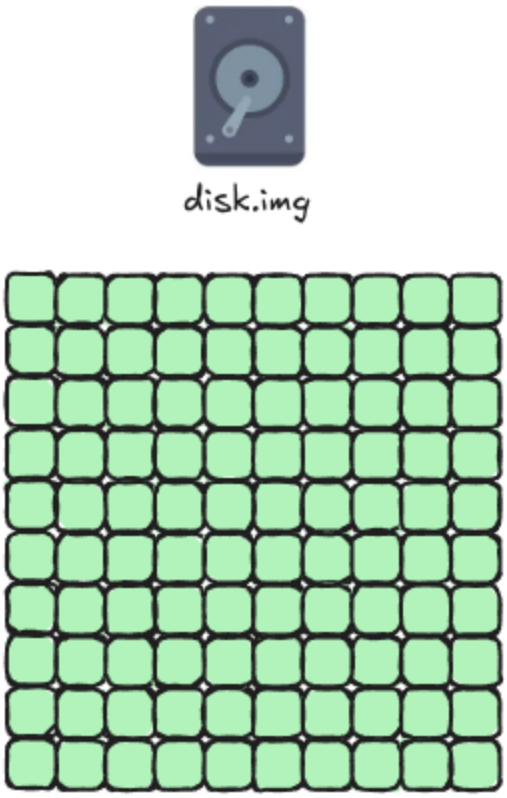
Take an example:

1. We buy a disk (HDD/SSD)
2. Make partitions (System/Data)
3. Format the partitions with specific File System (NTFS/ExFAT)
4. After formatting, we got volumes.

1. Virtual Disk Image

- We have a file disk.img
- Assume the virtual disk has 100Kb (Fixed size).

- There's 100 blocks in disk image, and each of they is 1Kb



2. VCB

- a VCB contains infomation of the volume, such as:

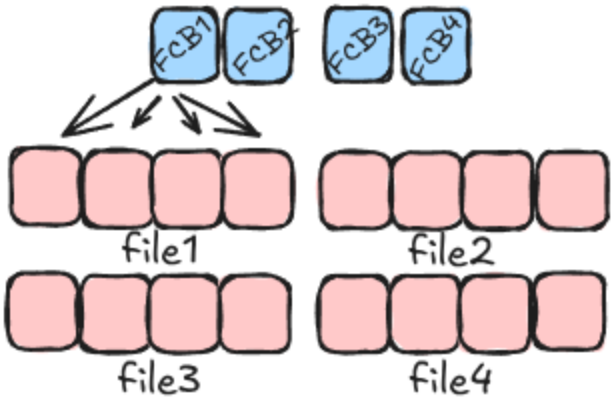
Field	Value/Description (Example)
Total Block Count	100
Block Size	1 KB (1024 bytes)
Free Block Map Location	Block 1 (Pointer to the Free Space Bitmap)
FCB Table Start Address	Block 2
Max FCB Count	The maximum number of files that can be created.
Free FCB Count	Current number of available FCBs.

- It is called superblock in UFS, while master file table in NTFS.

3. FCB(inode)

- a FCB contains infomation of the file, such as:

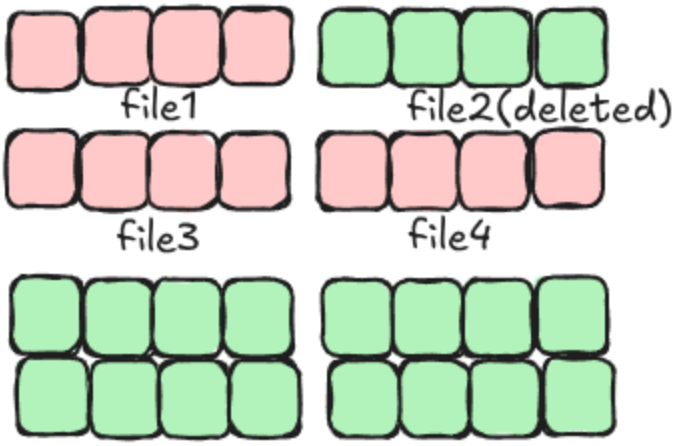
Field	Value/Description (Example)
Identifier Number	0-Max_N
File Size	0-1 KB (1024 bytes)
4 direct block pointer	Block n1,n2,n3,n4



4. File and Directory

In UNIX, a directory is treads as a special type of file, both of they have their own inode.

- File Structure (the contain of file blocks)
  - File Type: Image, document and executable file.
  - We just implement plain text file.
- Directory Structure
  - Dentry: Records file name and number of inode.
  - Tree-Structured Directory

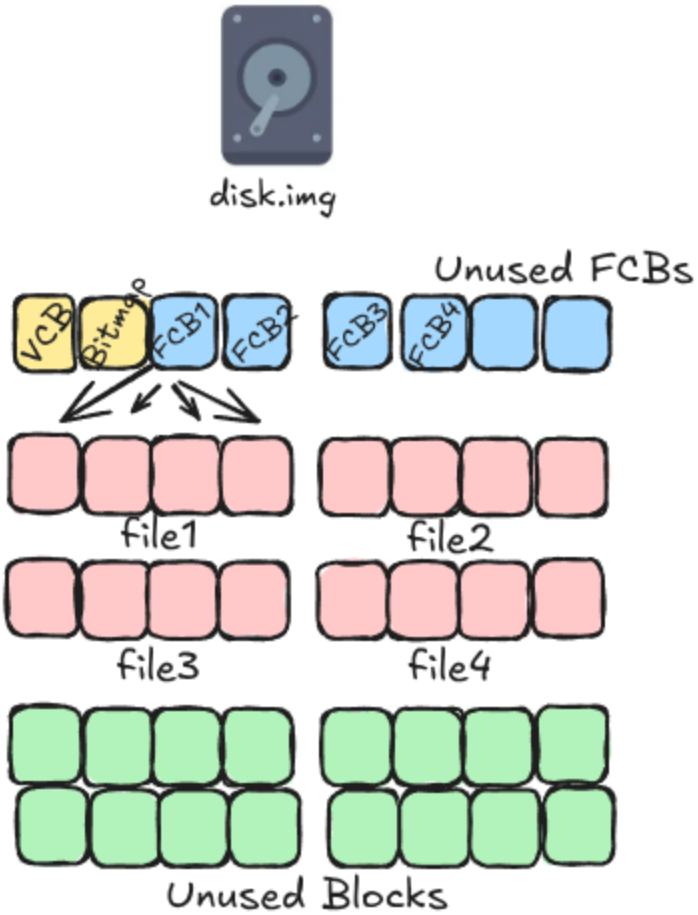


5. Bitmap

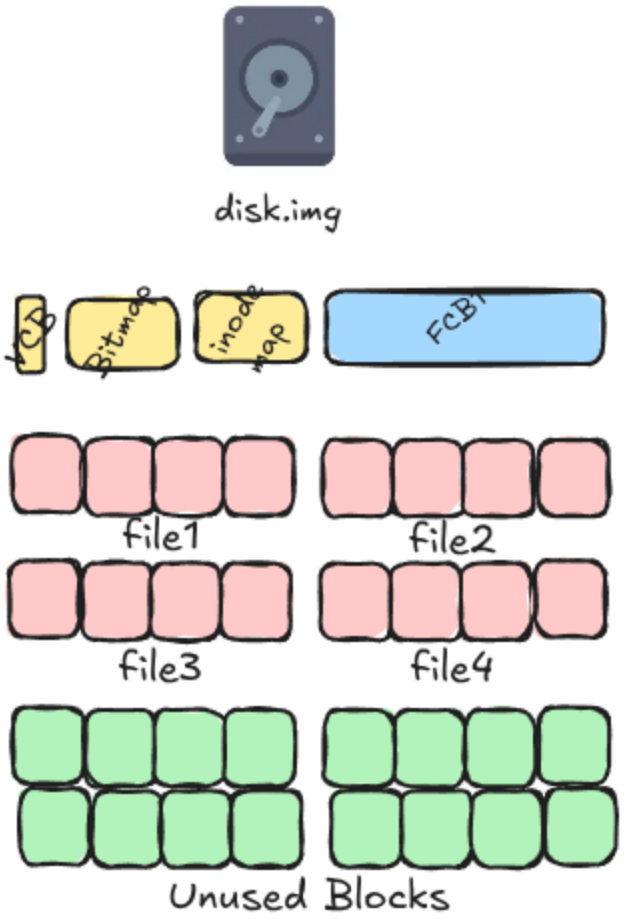
Stores the usage of blocks.

Summary: Disk Layout

Block(s)	Type	Rationale
Block 0	Superblock (VCB)	Volume-wide metadata.
Block 1	Free Space Bitmap Free inode	Tracks usage of the 100 blocks. Tracks usage of the 100 inodes.
Block 2 to 9	**FCB / Inode Table**	**One block stores MANY FCBs.** (e.g., 8 blocks × 3 FCBs/block = 24 files max).
Block 10 to 99	Data Blocks	Stores the file and directory contents. (90/4=22 files)



Another structure design:



File Operation in File System

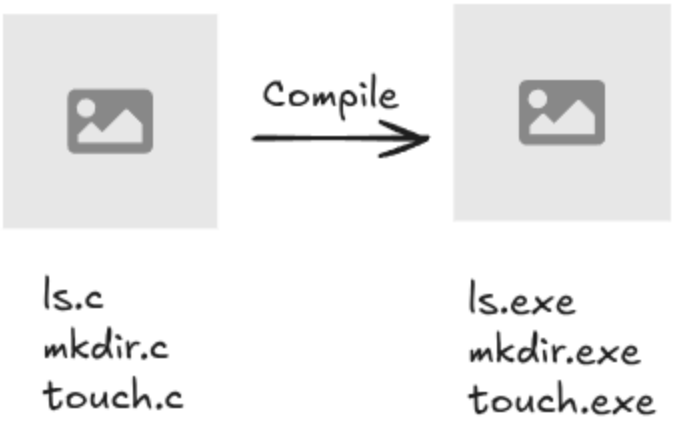
Disk Utility

In every OS, we use disk utility to make a partition for a disk, and format a partition.
In our project, we use `mkfs` to:

- Write the VCB (Superblock) to Block 0 with total block counts, etc.
- Initialize the Bitmap (setting all data blocks to 'free').
- Initialize the FCB Table (setting all FCB slots to 'free' or 'unused').
- Create the root directory by allocating the first FCB and its first data block.

Custom Program

In every OS, we can create and access file via shell.
In this case, we use a C program to emulate a shell. Just use system API.



mkfs

touch

append

cat

Contirbutions

1. 虛擬磁碟結構設計，完成 mkfs (✅:完成 ☑️:分配中 ❌:未完成)

Items	黃婧絜	鄒昱宸	蕭宏均	陳裕勛	李冠緯	Description
Design Disk Structure			✅			
mkfs mkVCB()	✅	✅				
mkfs mkBitmap()	✅	✅				
mkfs mkFCB()			✅			
mkfs mkRoot()					✅	Create first FCB to root dirctory.
mkfs.exe			✅			Merge these functions to an executable file for initializing the file system.

2. 實作 inode 與檔案讀寫，完成 touch / append / cat

Items	黃婧絜	鄒昱宸	蕭宏均	陳裕勛	李冠緯	Description
shell touch()			☑️			1. id = mkFCB() 2. add dentry(file name and id) to root directory(block10)
shell append()	☑️	☑️				Append text to the file. 1. find the file name in the directory and get its inode id 2. get the DBPs from the inode 3. write content to these blocks
shell cat()	☑️	☑️				Show the content of the file.

3. 實作目錄與路徑解析，完成 mkdir / ls / stat / rm / rmdir

Items	黃婧絜	鄒昱宸	蕭宏均	陳裕勛	李冠緯	Description
shell mkdir()					☑️	
ls()						
stat()						
rm()						
rmdir()						

3. 整合成 shell，設計測試流程，繳交程式與說明文件

Items	黃婧絜	鄒昱宸	蕭宏均	陳裕勛	李冠緯	Description
shell.exe			☑️			