TP n°7 : Les fonctions de lecture du SGF

Les TP n°7 ET n°8 sont à rendre ensemble sur AMETICE avant le mercredi 11 décembre 2024 à 18h15.

1 Programmation d'un mini S.G.F.

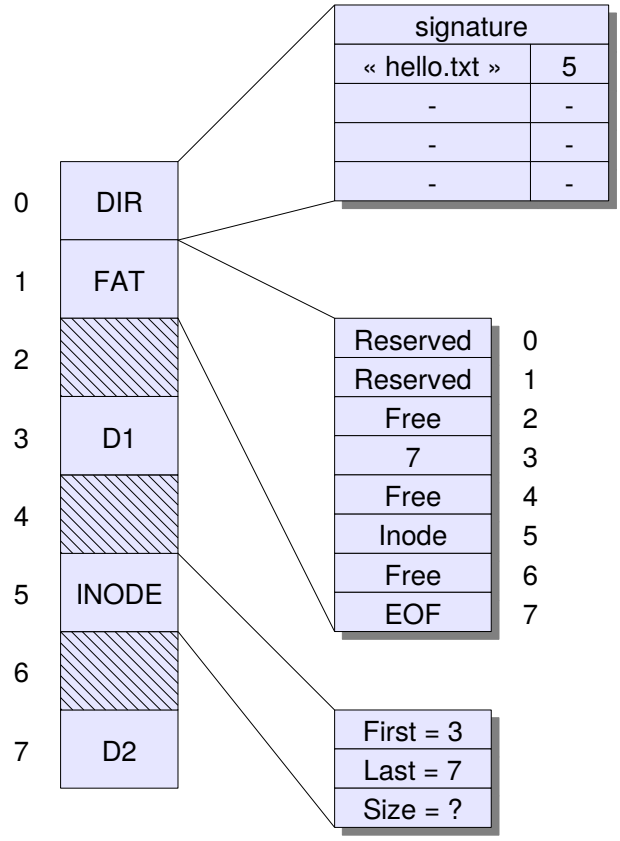
Nous allons, pendant deux séances de travaux pratiques, reconstruire les fonctions d'un S.G.F. très simplifié.

- Commencez par récupérer les sources de ce mini SGF.
- Le projet est organisé de la manière suivante :
  - ▷ `sgf-header.h` : structures de données et prototypes des fonctions du SGF (**complet**).
  - ▷ `sgf-disk.c` : entrée/sortie de blocs vers le disque simulé par le fichier `disk0` (**complet**).
  - ▷ `sgf-fat.c` : gestion de la FAT en mémoire et sur disque (**complet**).
  - ▷ `sgf-dir.c` : lecture/écriture des couples (nom, adr-i-node) dans le répertoire (**incomplet**).
  - ▷ `sgf-io.c` : opérations d'ouverture/lecture/écriture/fermeture des fichiers (**incomplet**).
  - ▷ `sgf-impl.o` : Implantation des fonctions à terminer (**code C non fourni**).

2 Explication du codage choisi

Commencez par étudier le fichier `sgf-header.h`. Il regroupe toutes les définitions des structures et des fonctions. Il permet de comprendre la structure choisie pour coder le répertoire, le chaînage des blocs et les fichiers sur notre disque simulé.

Le schéma ci-dessous détaille un disque de huit blocs (0 à 7) qui comporte un fichier (`hello.txt`) composé de deux blocs (`D1` et `D2`) rangés respectivement dans le bloc physique 3 et le bloc physique 7. Le descripteur de ce fichier se trouve dans le bloc physique 5. On trouve l'adresse du descripteur dans le répertoire. Le répertoire est toujours dans le bloc 0 suivi de la FAT.



La FAT : Le chaînage des blocs qui composent un fichier est placé dans une table unique appelée la FAT (File Allocation Table). Cette table comporte autant d'entrées que de blocs sur le disque. Elle est stockée sur disque à partir du bloc 1. La valeur de chaque entrée est interpretée comme suit :

```
FAT[n] = FAT_RESERVED le bloc n est réservé (et donc occupé)
FAT[n] = FAT_FREE     le bloc n est libre
FAT[n] = FAT_INODE    le bloc n contient un i-node (il est occupé)
FAT[n] = FAT_EOF      le bloc n est le dernier du fichier
FAT[n] = m            le bloc m est le suivant du bloc n dans le fichier
                      auquel ils appartiennent tous les deux.

Pour un fichier constitué des blocs 3 et 7, nous avons

    inode.first == 3          FAT[3] == 7
    inode.last == 7           FAT[7] == FAT_EOF
```

3 Lister le répertoire

Programmez la fonction `list_directory` du fichier `sgf-dir.c` sans faire appel à la version `_impl` (des explications sont disponibles dans le commentaire).

Vous pouvez utiliser la commande `format` livrée avec le projet pour fabriquer un disque vierge avec plusieurs fichiers de test placés à l'intérieur.

4 Lecture dans un fichier

La lecture se déroule en trois étapes :

1. ouverture du fichier en lecture (étudiez la fonction `sgf_open` du fichier `sgf-io.c`)
2. lecture d'un caractère dans le buffer (fonction `sgf_getc`)
3. lecture d'un bloc pour remplir le buffer (fonction `sgf_read_block`)

Commencez par programmer la fonction `sgf_getc` disponible dans le fichier `sgf-io.c`. Des explications sont disponibles dans le commentaire. Vérifiez le bon fonctionnement avant de passer à la fonction suivante.

Continuez avec la fonction `sgf_read_block` et testez son bon fonctionnement.

5 Accès direct en lecture

On vous demande de réaliser la fonction `sgf_seek` qui réalise le déplacement du pointeur `ptr` en lecture. Pour ce faire, suivez les étapes ci-dessous :

1. Ajoutez l'implantation de la fonction `sgf_seek` dans le fichier `sgf-io.c`. Cette fonction va déplacer le pointeur associé au fichier ouvert (`ptr` donc) en utilisant le deuxième paramètre. Si la position demandée n'est pas un multiple de la taille des blocs, alors le bloc en question devra être chargé dans le tampon (fonction `sgf_read_block`).

   Attention : prenez soin de limiter la lecture de bloc notamment quand la nouvelle position concerne le même bloc que la position précédente. Cette fonction renvoie 0 en cas de succès et -1 sinon.

2. Testez cette fonction en lisant, dans un fichier, les caractères dont la position est multiple de 1.

3. Testez à nouveau avec une position multiple de 8.