# 1 File System Operations

## 1.1 Directory Entries

- Has user-readable name and i-number
  - Map files to inodes

  **Example**

  ```
  ("file-name", i-number)
  ```

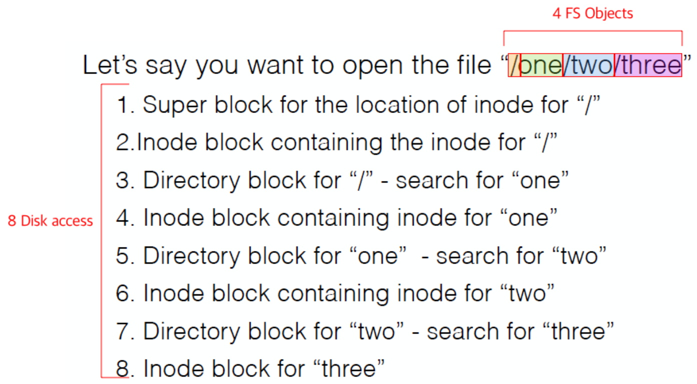
- Linux's ext2 has following attributes in its inode
  - Inode number
  - Directory entry length
  - Name length
  - File type
  - Name

## 1.2 How do we follow a path?

- Example
  - Read the first block of "/somefile"
    - The inode for the root directory "/" is known
    - Read the inode for "/" to file the data block(s) that hold the directory entires for the root directory
    - Read the first data block for the directory
    - Search the entries for "`somefile`"
    - The entry for "somefile" identifies the inode
    - Read the inode for "`somefile`"
    - The inode tells us the location of the first data block for the file
    - Read that block into memory access the data at the beginning of file

## 1.3 How many disks blocks are accessed?

- Example



4 FS Objects

Let’s say you want to open the file “/one/two/three”

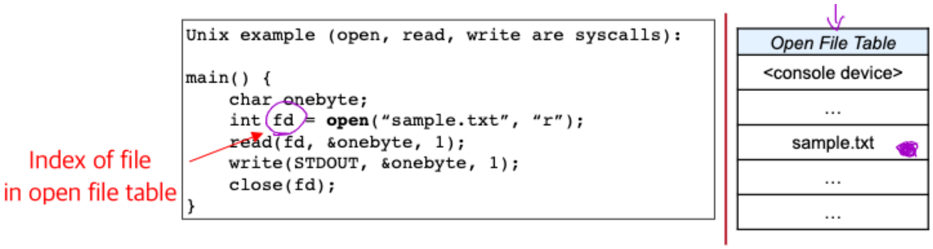
8 Disk access

1. Super block for the location of inode for “/”
2.Inode block containing the inode for “/”
3. Directory block for “/” - search for “one”
4. Inode block containing inode for “one”
5. Directory block for “one” - search for “two”
6. Inode block containing inode for “two”
7. Directory block for “two” - search for “three”
8. Inode block for “three”

- There are 8 disk accesses in total
- There are 4 file system objects

## 1.4 Handling File Operations

- Open file table
  - On open, file name, and it’s inode number is stored system-wide into array
  - Index to entry in open-file table is used in subsequent use, so no searching is required
  - This is what `fd` is for



```
Unix example (open, read, write are syscalls):

main() {
    char onebyte;
    int fd = open(“sample.txt”, “r”);
    read(fd, &onebyte, 1);
    write(STDOUT, &onebyte, 1);
    close(fd);
}
```

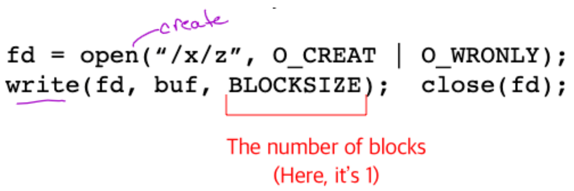
Index of file in open file table

| Open File Table |
|---|
| <console device> |
| … |
| sample.txt |
| … |
| … |

## 1.5 Possible Operations

- `mkdir("/x")`
  - Create a directory
- `creat("/x/y")`
  - Create an empty file
- `unlink("/x/y")`
  - Remove file or directory

- `fd = open("/x/z", O_CREAT | O_WRONLY);`
  `write (fd, buf, BLOCKSIZE);`
  `close(fd);`
  - Open a file for writing, and write one block to it
  - If it does not exist, create it



create

```
fd = open("/x/z", O_CREAT | O_WRONLY);
write(fd, buf, BLOCKSIZE);  close(fd);
```

The number of blocks
(Here, it's 1)