Persistence

Last Time: I/O devices + magnetic disk

today: interface to FS (files, directories)

Fri: FS implement

Basic Abstraction

Files

- array of bytes
- has a low-level name (inode number)
- persistent
- Create, read, write, grow, shrink, delete
  stat

Directory

- array of record
- maps human-readable name to low-level number
- create, delete, read, ~~write~~ not directly

/a/b → 513279
/home/djw/hw.txt

"everything is a file"

./ ../

File

Creating a file: creat/open(O_CREAT)
↳ fd: file descriptor

- open files tracked in table
  (not shared across processes in general)

0 → stdout
1 → stderr

echo foo 2>&1

Reading: read()

Some thing keeps of **offset** in file

lseek: adjust offset

whence SEEK_SET
SEEK_CUR
SEEK_END

lseek(fd, off, whence);

grow:

truncate:

delete: unlink???

rename:

stat: what should FS store about file?

meta data { - size, - read/write, - last time modified, - inode # [index node] } - pointers to blocks of data

per-file structure called **inode**

Directories

- just a special type of file
- map user-level name → inode number

create: mkdir

rm: rmdir

read: readdir

. current directory

.. parent directory

/ ROOT directory its parent is itself

inode # is well known (2)

Path names: /x/y/z Absolute

y/z Relative

CWD (current working directory) per-process

Open a file

fd = open("/a/b/c/file1", O_RDONLY);

- start root (inode is known (2)) [cwd]

read dir - look for next inode #

traversal

Hard Links: not a special type of file (file)
another name for same ~~inode~~

Symbolic Link: special type of file
(soft) own inode #

dangling pointer

why: multiple FS (different inode structures) in same tree [mount]