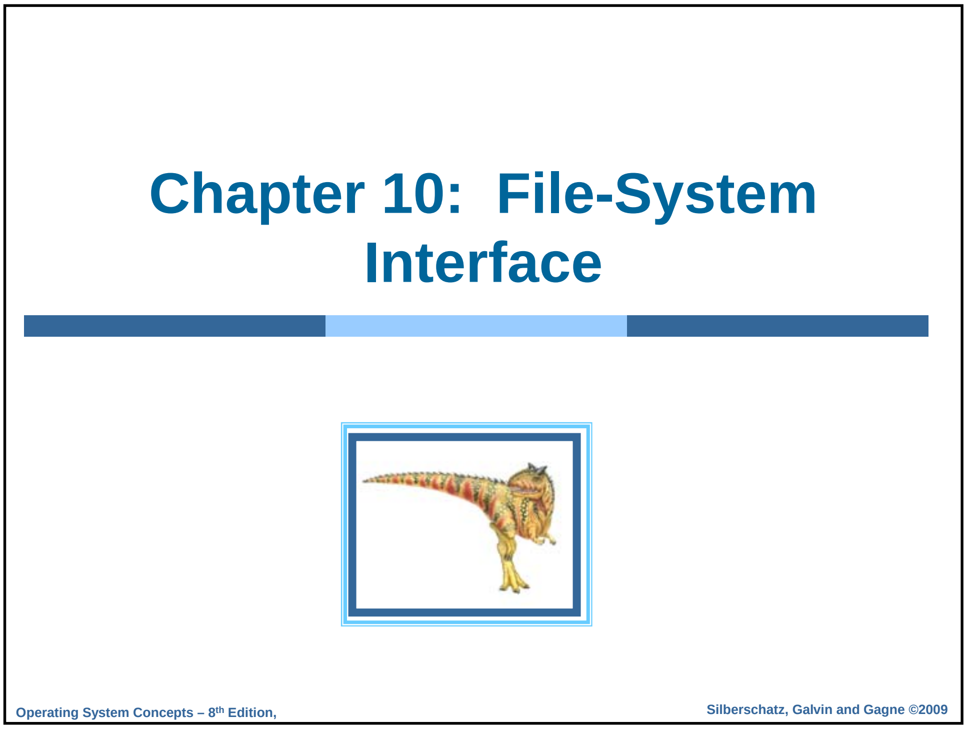
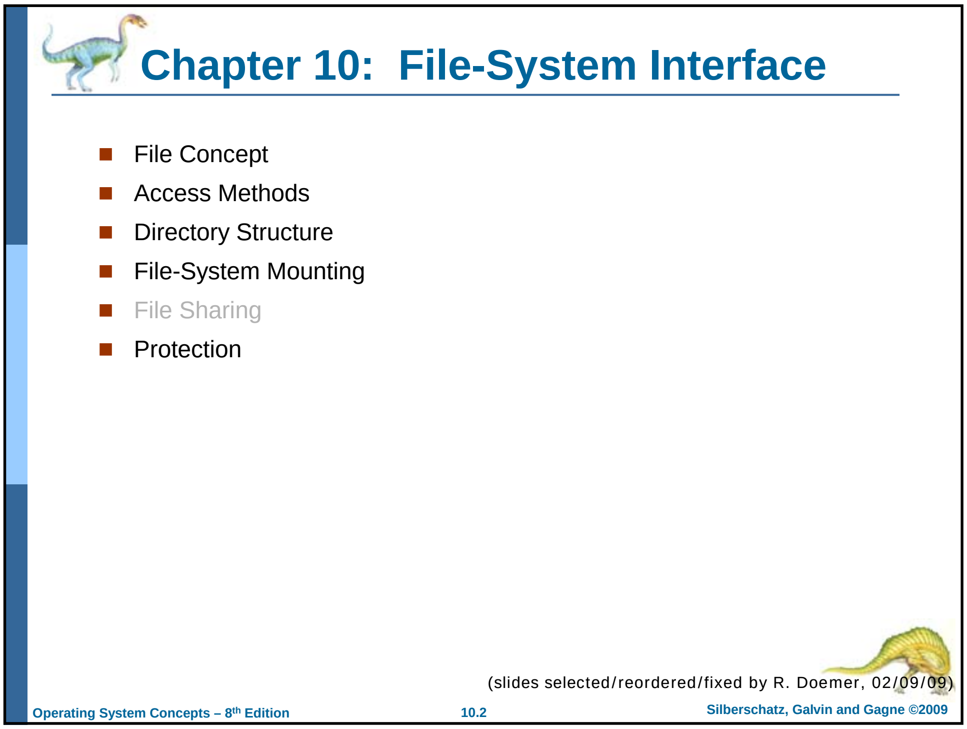


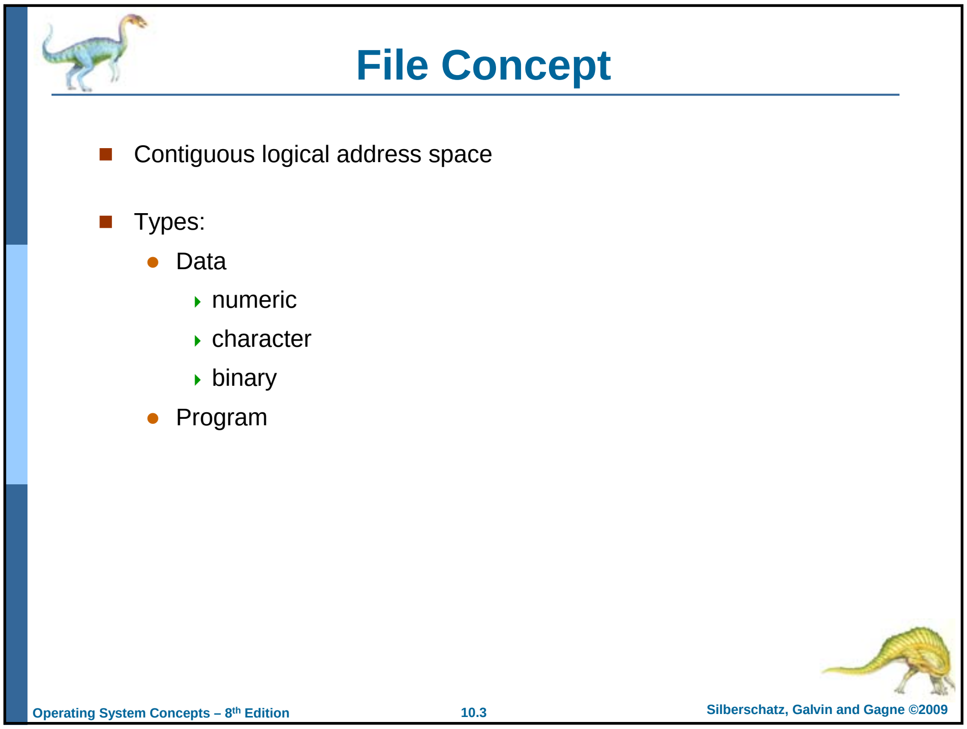
Chapter 10: File-System Interface



Chapter 10: File-System Interface

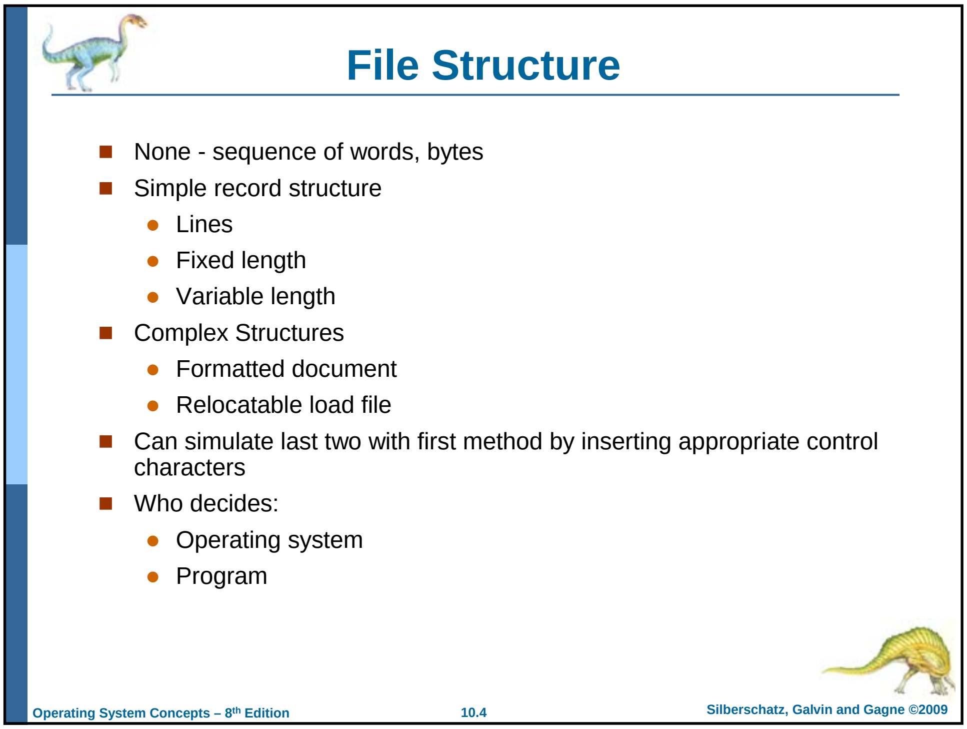
- File Concept
- Access Methods
- Directory Structure
- File-System Mounting
- File Sharing
- Protection

(slides selected/reordered/fixed by R. Doemer, 02/09/09)



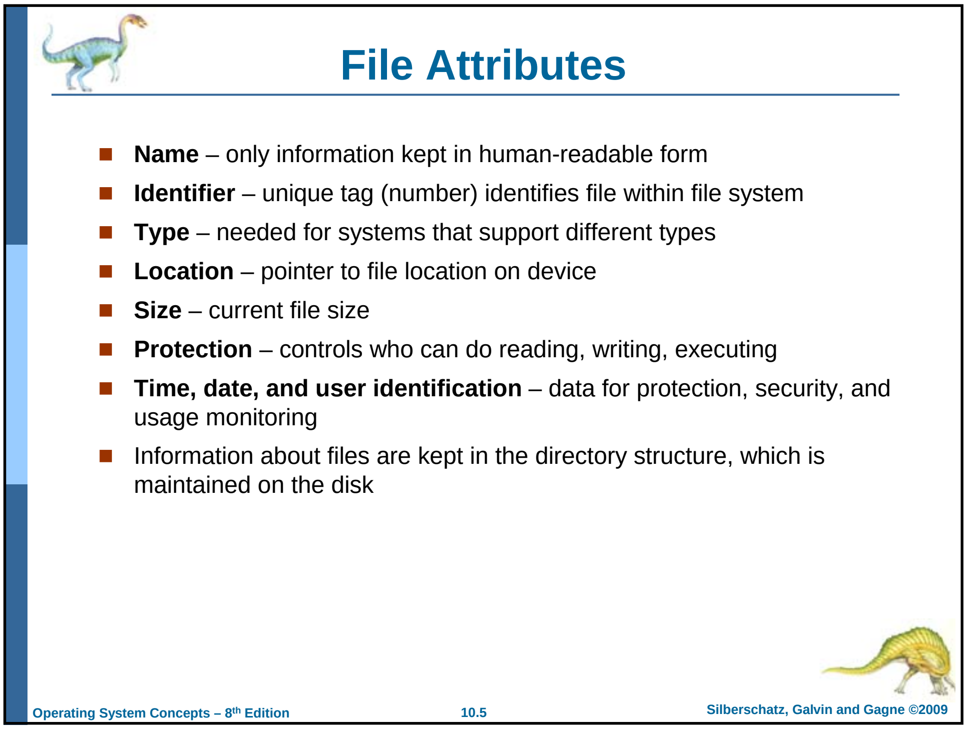
File Concept

- Contiguous logical address space
- Types:
  - Data
    - numeric
    - character
    - binary
  - Program



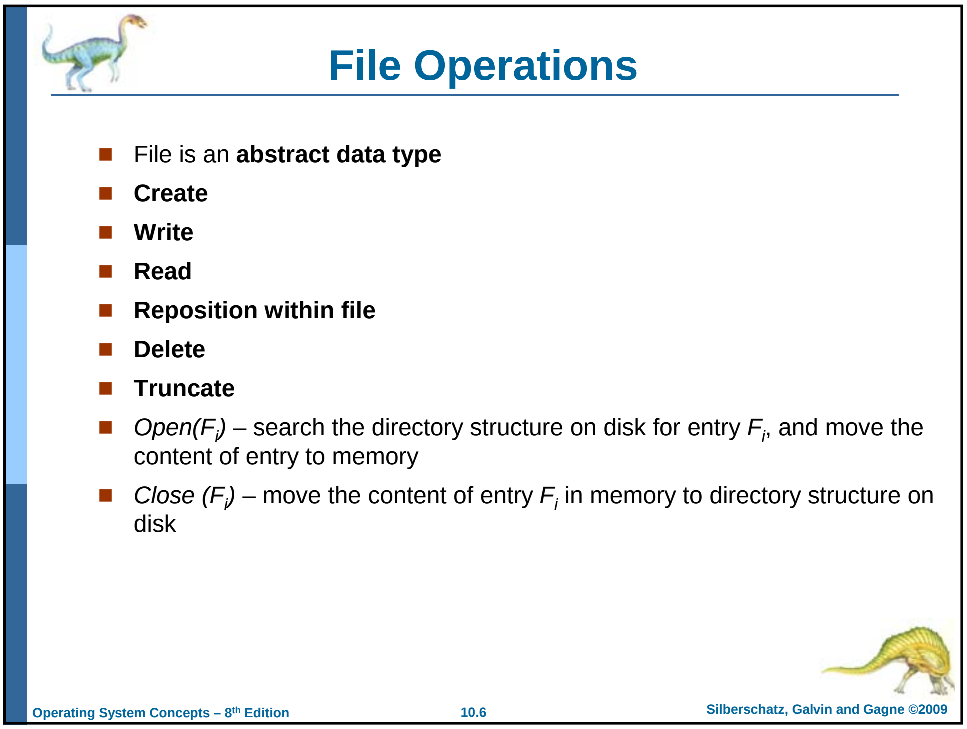
File Structure

- None - sequence of words, bytes
- Simple record structure
  - Lines
  - Fixed length
  - Variable length
- Complex Structures
  - Formatted document
  - Relocatable load file
- Can simulate last two with first method by inserting appropriate control characters
- Who decides:
  - Operating system
  - Program



File Attributes

- **Name** – only information kept in human-readable form
- **Identifier** – unique tag (number) identifies file within file system
- **Type** – needed for systems that support different types
- **Location** – pointer to file location on device
- **Size** – current file size
- **Protection** – controls who can do reading, writing, executing
- **Time, date, and user identification** – data for protection, security, and usage monitoring
- Information about files are kept in the directory structure, which is maintained on the disk



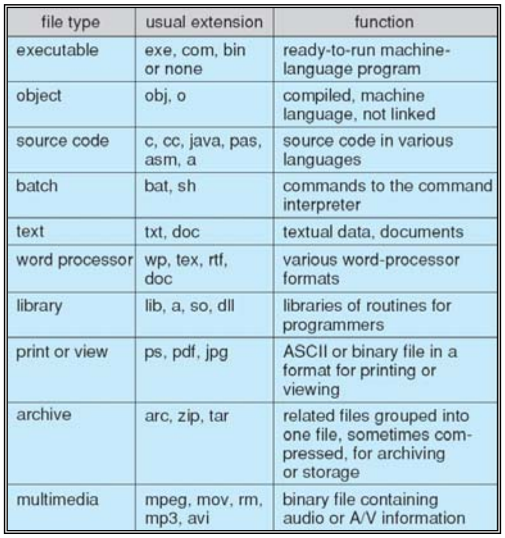
File Operations

- File is an **abstract data type**
- **Create**
- **Write**
- **Read**
- **Reposition within file**
- **Delete**
- **Truncate**
- *Open(F_i)* – search the directory structure on disk for entry F_i, and move the content of entry to memory
- *Close (F_i)* – move the content of entry F_i in memory to directory structure on disk

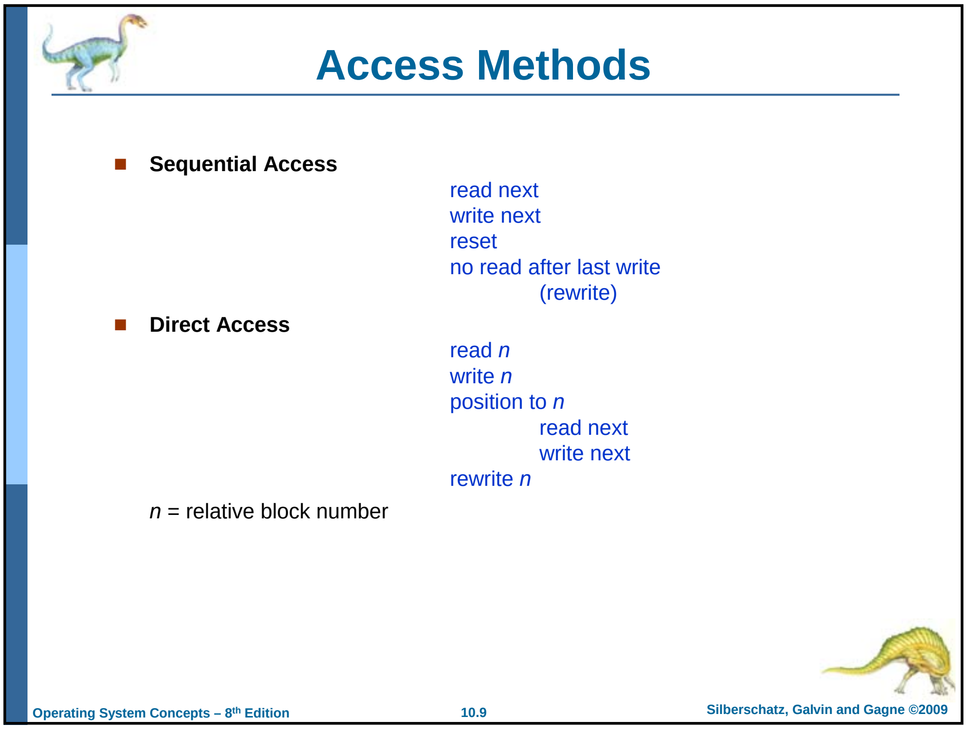
Open Files

- Several pieces of data are needed to manage open files:
  - File pointer:
    pointer to last read/write location, per process that has the file open
  - File-open count:
    counter of number of times a file is open – to allow removal of data from open-file table when last processes closes it
  - Disk location of the file:
    cache of data access information
  - Access rights:
    per-process access mode information

File Types – Name, Extension



file type	usual extension	function
executable	exe, com, bin or none	ready-to-run machine-language program
object	obj, o	compiled, machine language, not linked
source code	c, cc, java, pas, asm, a	source code in various languages
batch	bat, sh	commands to the command interpreter
text	txt, doc	textual data, documents
word processor	wp, tex, rtf, doc	various word-processor formats
library	lib, a, so, dll	libraries of routines for programmers
print or view	ps, pdf, jpg	ASCII or binary file in a format for printing or viewing
archive	arc, zip, tar	related files grouped into one file, sometimes compressed, for archiving or storage
multimedia	mpeg, mov, rm, mp3, avi	binary file containing audio or A/V information



Access Methods

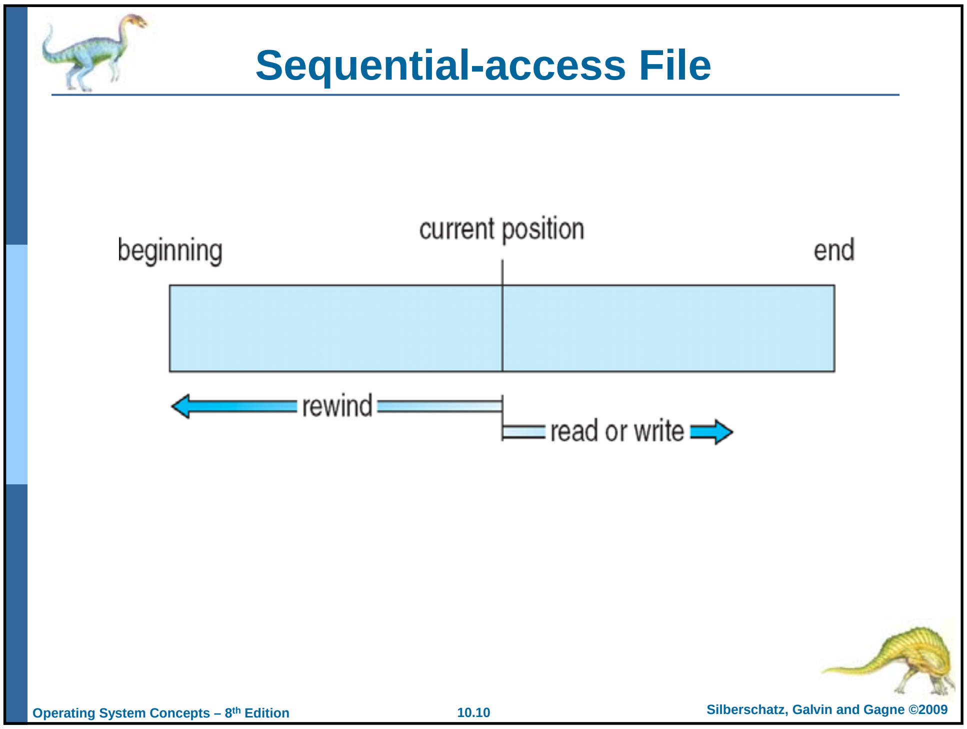
- **Sequential Access**

  read next
  write next
  reset
  no read after last write
  (rewrite)

- **Direct Access**

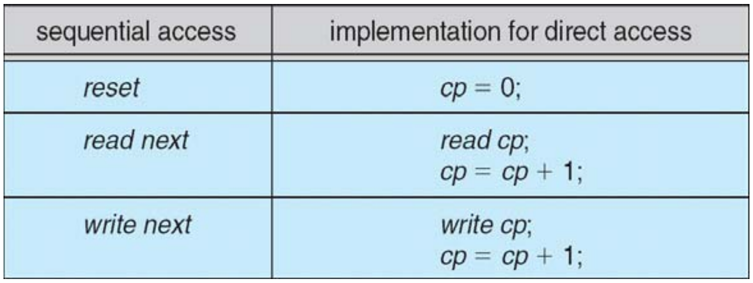
  read *n*
  write *n*
  position to *n*
  read next
  write next
  rewrite *n*

n = relative block number



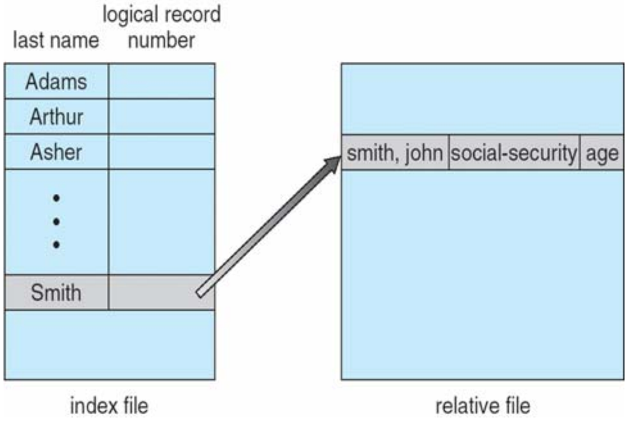
Sequential-access File

Simulation of Sequential Access on Direct-access File



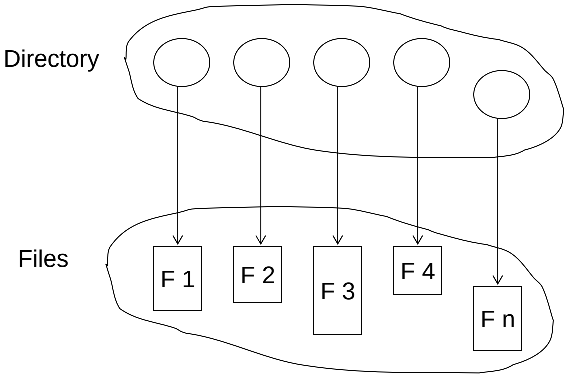
sequential access	implementation for direct access
reset	*cp* = 0;
read next	*read cp*; *cp* = *cp* + 1;
write next	*write cp*; *cp* = *cp* + 1;

Example of Index and Relative Files



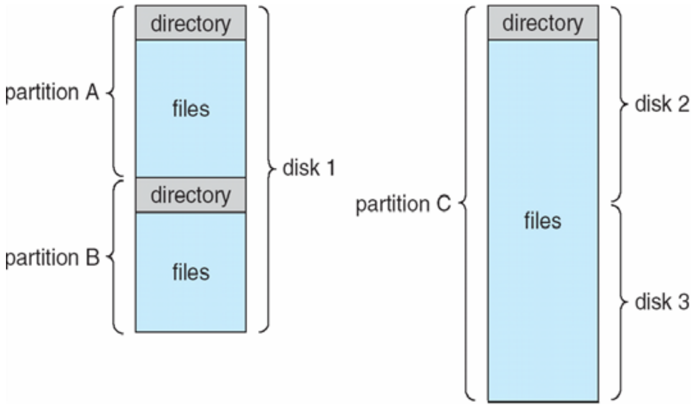
Directory Structure

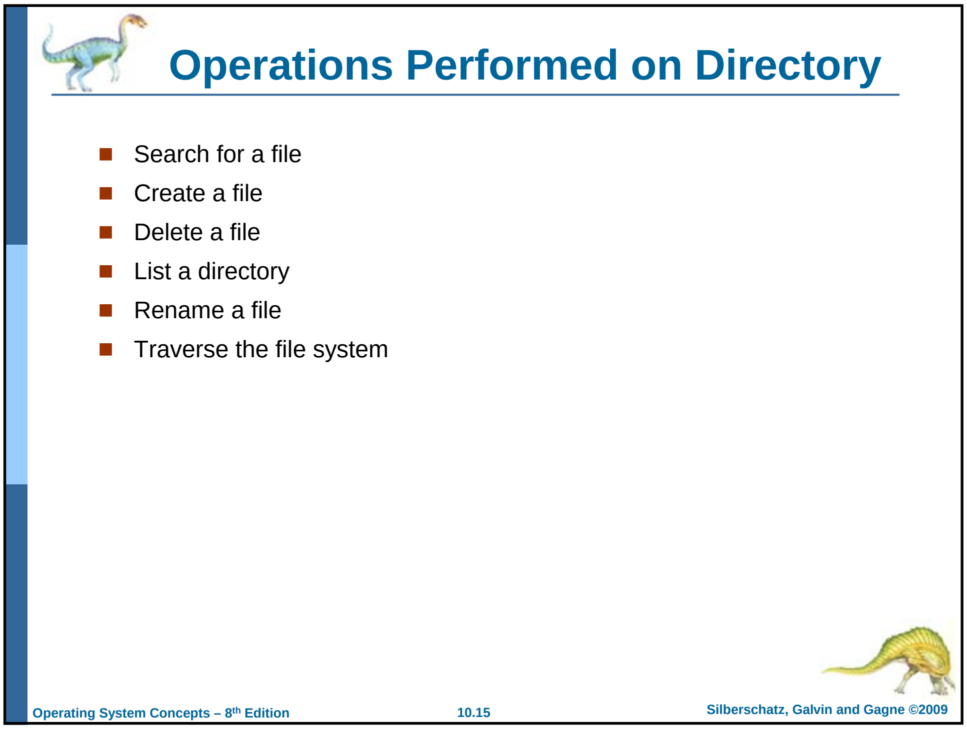
- A collection of nodes containing information about all files



Both the directory structure and the files reside on disk
Backups of these two structures are kept on tapes

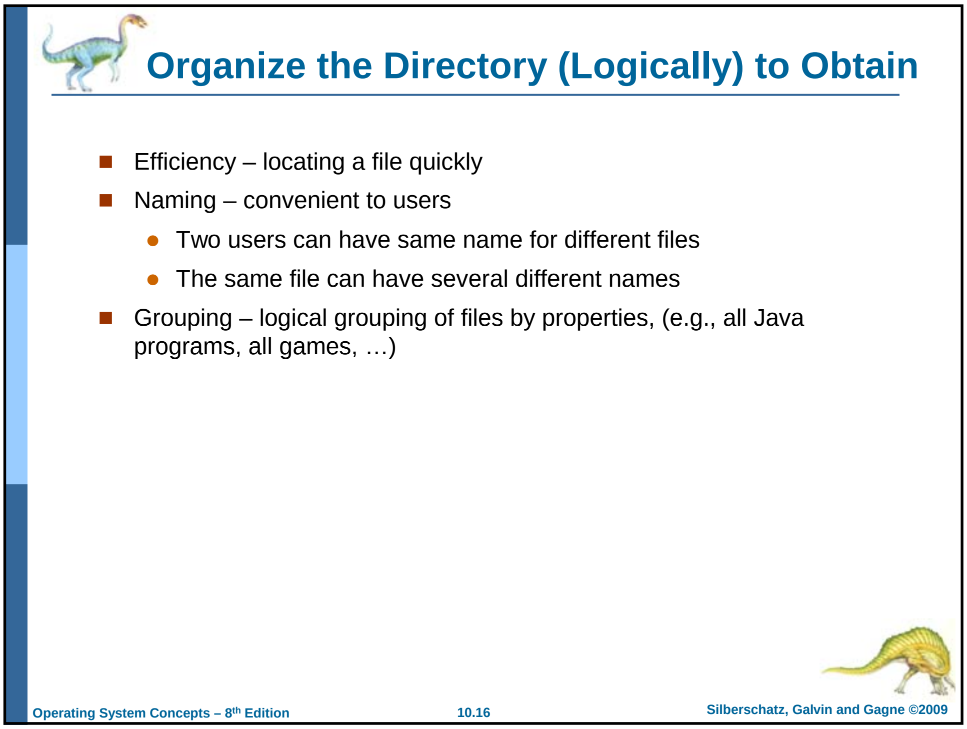
A Typical File-system Organization





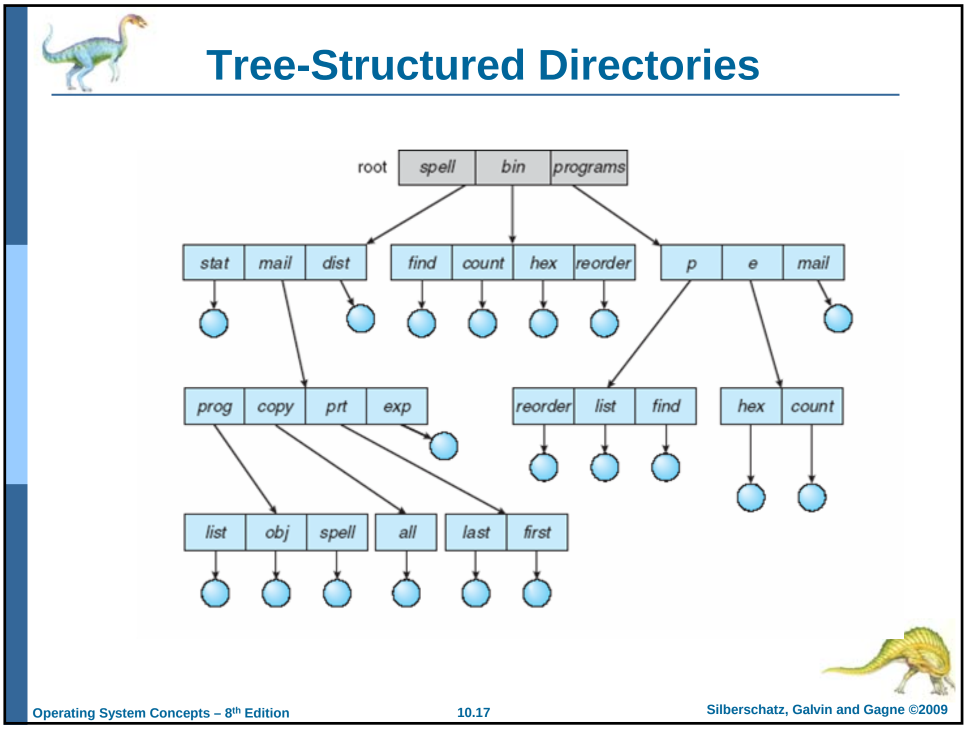
Operations Performed on Directory

- Search for a file
- Create a file
- Delete a file
- List a directory
- Rename a file
- Traverse the file system

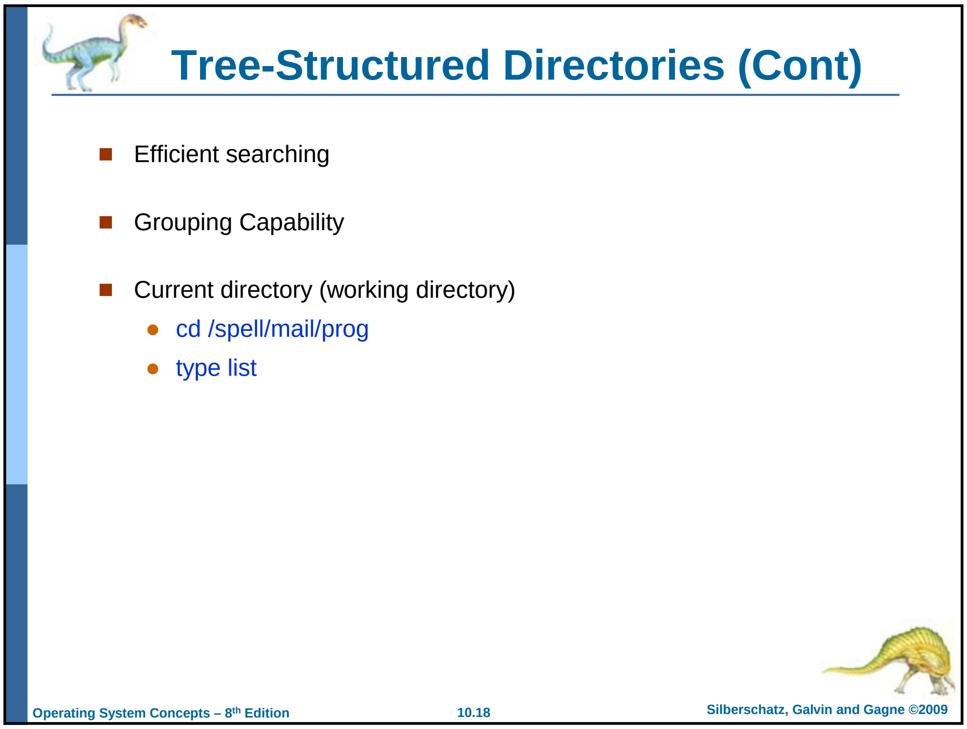


Organize the Directory (Logically) to Obtain

- Efficiency – locating a file quickly
- Naming – convenient to users
  - Two users can have same name for different files
  - The same file can have several different names
- Grouping – logical grouping of files by properties, (e.g., all Java programs, all games, …)



Tree-Structured Directories



Tree-Structured Directories (Cont)

- Efficient searching
- Grouping Capability
- Current directory (working directory)
  - cd /spell/mail/prog
  - type list

Tree-Structured Directories (Cont)

- **Absolute** or **relative** path name
- Creating a new file is done in current directory
- Delete a file

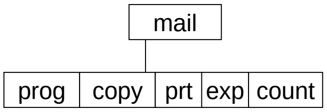
```
rm <file-name>
```

- Creating a new subdirectory is done in current directory

```
mkdir <dir-name>
```

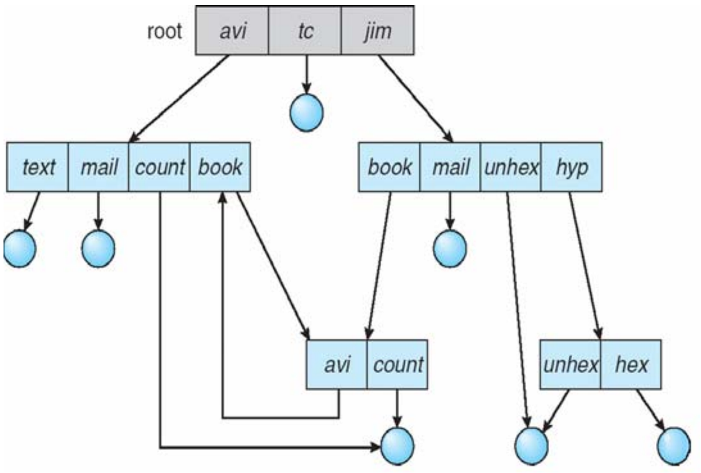
Example: if in current directory /mail

```
mkdir count
```



Deleting “mail” ⇒ deleting the entire subtree rooted by “mail”

General Graph Directory

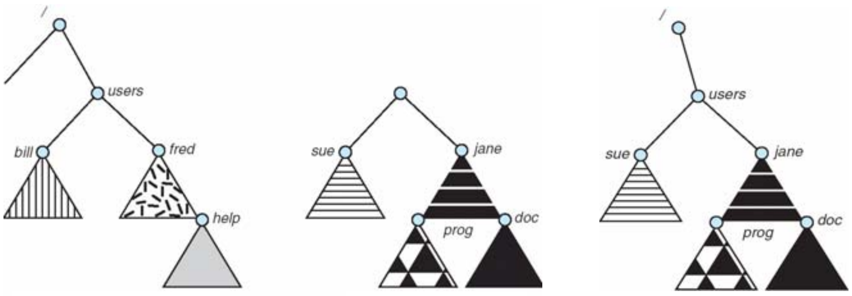


General Graph Directory (Cont.)

- How do we guarantee no cycles?
  - Allow only links to file not subdirectories
  - Garbage collection
  - Every time a new link is added use a cycle detection algorithm to determine whether it is OK

File System Mounting

- A file system must be **mounted** before it can be accessed
- A unmounted file system is mounted at a **mount point**
- In Unix/Linux, any **directory** can serve as a mount point
  - If not empty, previous contents are not accessable after mounting

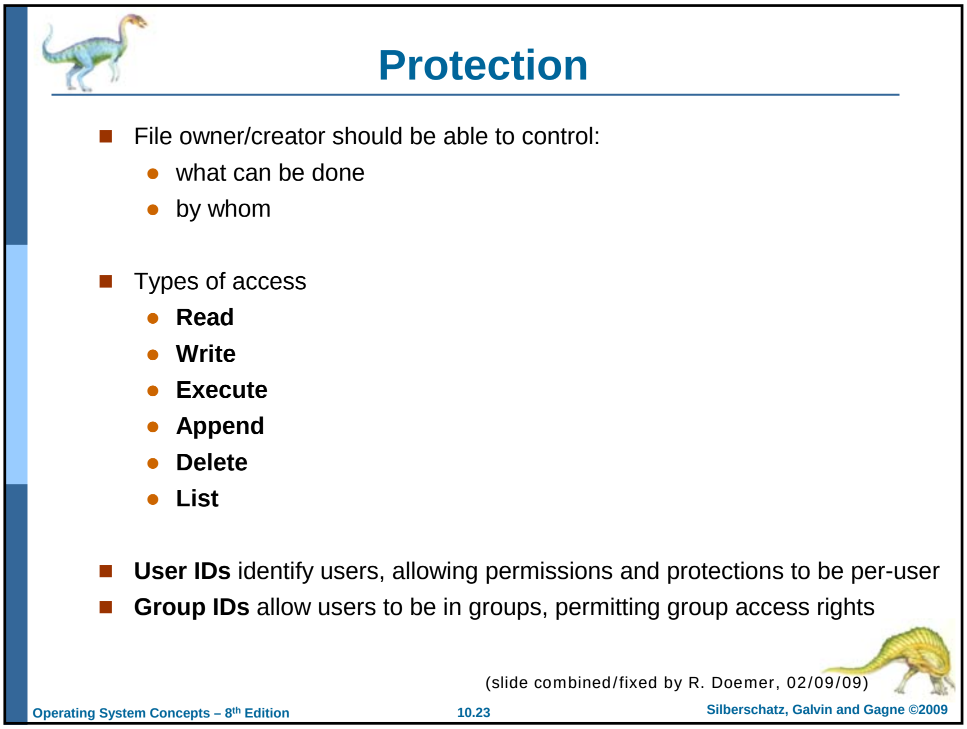


Initial directory tree

Other file system

Other file system mounted at **/users**

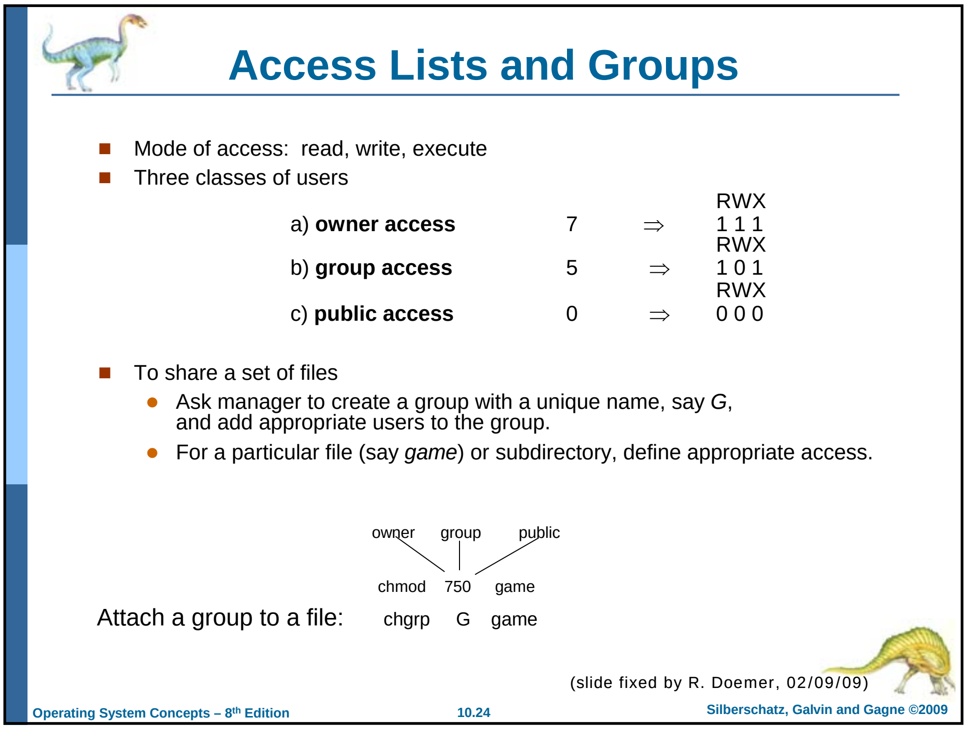
(slides combined/fixed by R. Doemer, 02/09/09)



Protection

- File owner/creator should be able to control:
  - what can be done
  - by whom
- Types of access
  - **Read**
  - **Write**
  - **Execute**
  - **Append**
  - **Delete**
  - **List**
- **User IDs** identify users, allowing permissions and protections to be per-user
- **Group IDs** allow users to be in groups, permitting group access rights

(slide combined/fixed by R. Doemer, 02/09/09)



Access Lists and Groups

- Mode of access: read, write, execute
- Three classes of users

			RWX
a) **owner access**	7	⇒	1 1 1
			RWX
b) **group access**	5	⇒	1 0 1
			RWX
c) **public access**	0	⇒	0 0 0

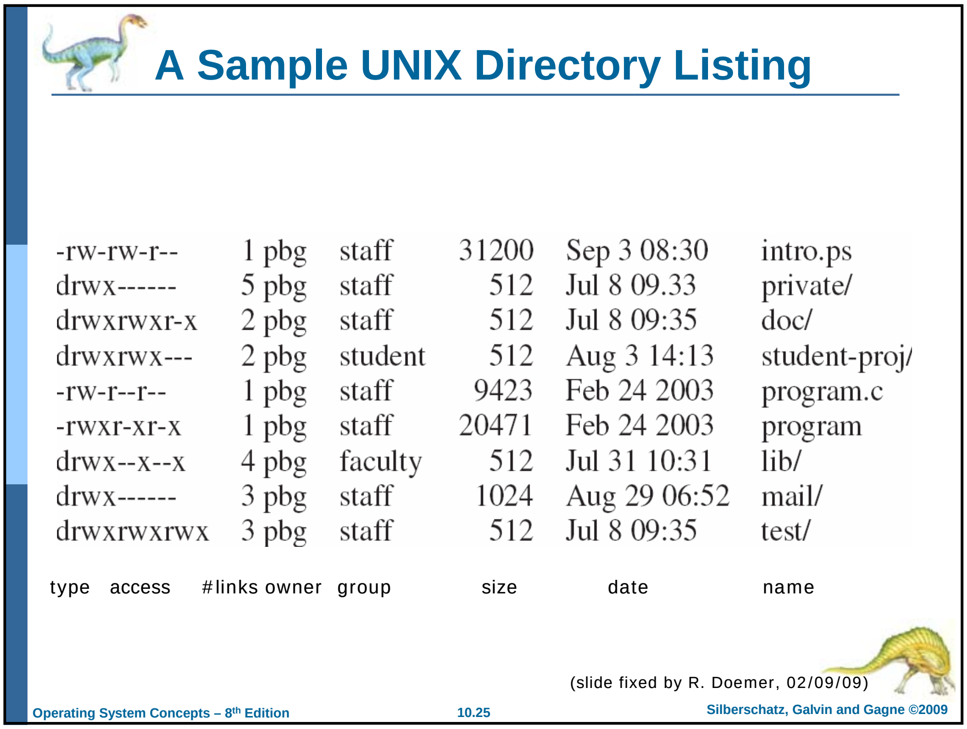
- To share a set of files
  - Ask manager to create a group with a unique name, say *G*, and add appropriate users to the group.
  - For a particular file (say *game*) or subdirectory, define appropriate access.

owner group public

chmod 750 game

Attach a group to a file: chgrp G game

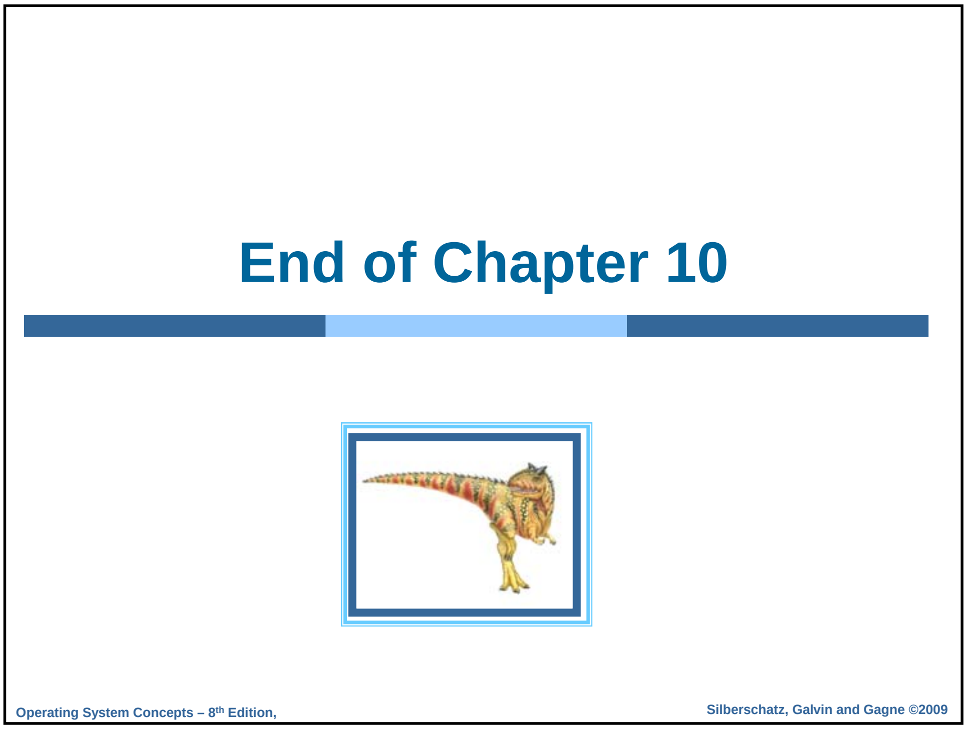
(slide fixed by R. Doemer, 02/09/09)



A Sample UNIX Directory Listing

```
-rw-rw-r--     1 pbg   staff      31200   Sep 3 08:30    intro.ps
drwx------     5 pbg   staff        512   Jul 8 09.33    private/
drwxrwxr-x     2 pbg   staff        512   Jul 8 09:35    doc/
drwxrwx---     2 pbg   student      512   Aug 3 14:13    student-proj/
-rw-r--r--     1 pbg   staff       9423   Feb 24 2003    program.c
-rwxr-xr-x     1 pbg   staff      20471   Feb 24 2003    program
drwx--x--x     4 pbg   faculty      512   Jul 31 10:31   lib/
drwx------     3 pbg   staff       1024   Aug 29 06:52   mail/
drwxrwxrwx     3 pbg   staff        512   Jul 8 09:35    test/
type access  #links owner  group    size      date          name
```

(slide fixed by R. Doemer, 02/09/09)



End of Chapter 10