Slide 1

Administrivia

- Reminder: Homework 5 due today.

Slide 2

Memory Management — A Few More Things

- (Review slides about segmentation from last time?)
- Difference between “physical and virtual memory address spaces”? Address space is an abstraction, one per process, addresses usually range from smallest to largest possible. Physical memory is, well, physical?
- Differences between Windows and Linux memory management? as far as I can tell, similar ideas. Pluses/minuses of different page replacement algorithms? refer to text (sorry!).
- How does “the system” know something is in (physical) memory as opposed to on disk? Page table (accessed by MMU) says what’s in memory. If it’s not, o/s maintains separate data structure(s) that say what’s on disk.

Slide 3

Files and Filesystems — Overview

- Very abstract view — requirements for long-term information storage are:
  - Store large amounts of information.
  - Have information survive past end of creating process.
  - Allow concurrent access by multiple processes.
- Usual solution — "files" on disk and other external media, organized into "file systems".
- In terms of the two views of an o/s:
  - "Virtual machine" view — filesystem is important abstraction.
  - "Resource manager" view — filesystem manages disk (and other device) resources.
- We'll look first at the user view, then at implementation.

Slide 4

File Abstraction

- Many, many aspects of "file abstraction" — name, type, ownership, etc., etc. Most involve choices/tradeoffs.
- In the following slides, a quick tour of some of the major ones, with some of the possible variations.

Slide 5

File Abstraction, Continued

- File names — always “text string”, but some choices: maximum length? case-sensitive? ASCII or Unicode? “extension” required?
- File structure — how file appears to application program:
  - Unstructured sequence of bytes — maximum flexibility, but maybe more work for application.
  - Sequence of fixed-length records — widely used in older systems, not much any more.
  - Tree (or other) structure supporting access by key.

Slide 6

File Abstraction, Continued

- File types — include “regular files”, also directories and (in some systems, such as UNIX) “special files”. Regular files subdivide into:
  - ASCII files — sequences of ASCII characters, generally separated into lines by line-end character(s).
  - Binary files — everything else, including executables, various archives, MS Word format, etc., etc. Most have some structure, defined by the expectations of the program(s) that work with them — applications for some types, operating system for executables.
- File access — sequential versus random-access.
- File attributes — “other stuff” associated with file (owner, protection info, time of creation / last use, etc.)

Slide 7

File Abstraction, Continued

- File operations (things one can do to a file) include create, delete, open, close, read, write, get attributes, set attributes. Example program using low-level wrappers for system calls on p. 266.
- Many systems also support operations for "memory-mapped files" (read whole file into memory, process there, write back out — as mentioned in previous discussion of memory management).

Slide 8

Directory/Folder Abstraction

- Basic idea — way of grouping / keeping track of files. Can be
  - Single-level (simple but restrictive).
  - Two-level (almost as simple, better if multiple users, but also restrictive).
  - Hierarchical.
- Implies need for path names, which can be absolute or relative (to "working directory").
- Operations on directories include create, delete, open, close, read, add entry, remove entry.

Slide 9

Minute Essay

- If you have a system that supports multiple different file systems (such as Linux with Samba to access Windows files), what issues might arise in copying files between different file systems?

  (We had an interesting problem with backing up `/users` to an OS X machine because the default for OS X is case-insensitive.)

Slide 10

Minute Essay Answer

- Case sensitivity is one source of potential problems. Other potential problems include restrictions on what characters can appear in filenames and what notion of file ownership and permissions is supported.