Slide 1

Administrivia

- Project proposals due today. Accepted without penalty through Wednesday.

Slide 2

Suggestions for Project Ideas

- Explore something about the lab machines' configuration that interests you.
- Example: The locally-written script(s) called from the default `.bash_profile`. Purpose was originally to allow same configuration / home directory to work well on different systems. Still a need for that (Linux and OS X machines). How does the current approach work? Is it okay (flexible, robust, reasonably amenable to being changed by users), or can you propose something better?

Slide 3

Suggestions for Project Ideas, Continued

- Think of something you do often, or would like to do, that seems amenable to scripting / automation.
- Example: Do performance testing of a program, varying some parameter of interest (e.g., for a sort program, size of input, or for a parallel program, number of threads or processes), and automatically produce a graph of the results. (How could you do this?)
- (Example from my files, so to speak: Directory with many EPS files, used as figures in a big LaTeX document. Want to print each of them on a separate page, with its filename and the figure number used to reference it in the document. How to do this?)

Slide 4

Installing and Updating Software — Package Managers

- "Modern" way to package software for installation depends on "package manager" — something that keeps track of what's installed, what depends on what, etc. (Examples — Fedora Core has `yum`, Debian has `apt-get`.) Software packaged as, e.g., `.rpm` or `.deb` files.
- If installing in "normal" system directories, and as root, probably best to take this approach.
- If you want to install in other directories (e.g., your home directory), or you don't have root access, however . . .

Slide 5

Installing and Updating Software — “Tarballs”

- Traditionally, Unix software distributed in the form of a “tarball” (archive created by `tar`, possibly compressed, usually containing source). Still often available and useful — e.g., to install in your home directory.
- What do you do with a tarball? Typical installation goes like this:
  - “Untar” the file (`tar xf`). Usually creates a directory, often containing `README` and/or `INSTALL` files — which you should review.
  - Run `configure` script to set system-specific options. Usually figures most things out for itself, but may need/allow user input, either via command-line options or standard input.
  - Run `make` to compile, etc. Normally puts created files in the same directory.
  - Run `make install` to move/copy executables, etc., to system directories. Notice that this is the only step that requires root privileges —

Slide 6

and only if installing in system directories.

Slide 7

Minute Essay

- Do you have a Unix or Unix-like system you manage?
- If so, what do you use to install software on it?