

Session: File System and Commands
Topic: Files and Directories

Daniel Chang

Files

- A collection of information stored in a computer
- Analogous to books in a library

Ordinary

- Text - Something you can read
- Binary - Something computer can read

Special

- Include things like "links"

Directory

- Contains other files (like a bound volume or shelf)
- Windows uses the concept of "Folders"

Pathnames

Must be able to name location of files

- "Root" is starting point, named "/"
- Sub-directories identified by name, separated by "/"
- Absolute pathnames always start with "root"
- Relative pathnames are from current directory

Referencing higher directories

- "." refers to directory above (room outside)
- "." refers to current directory

Referencing user directories

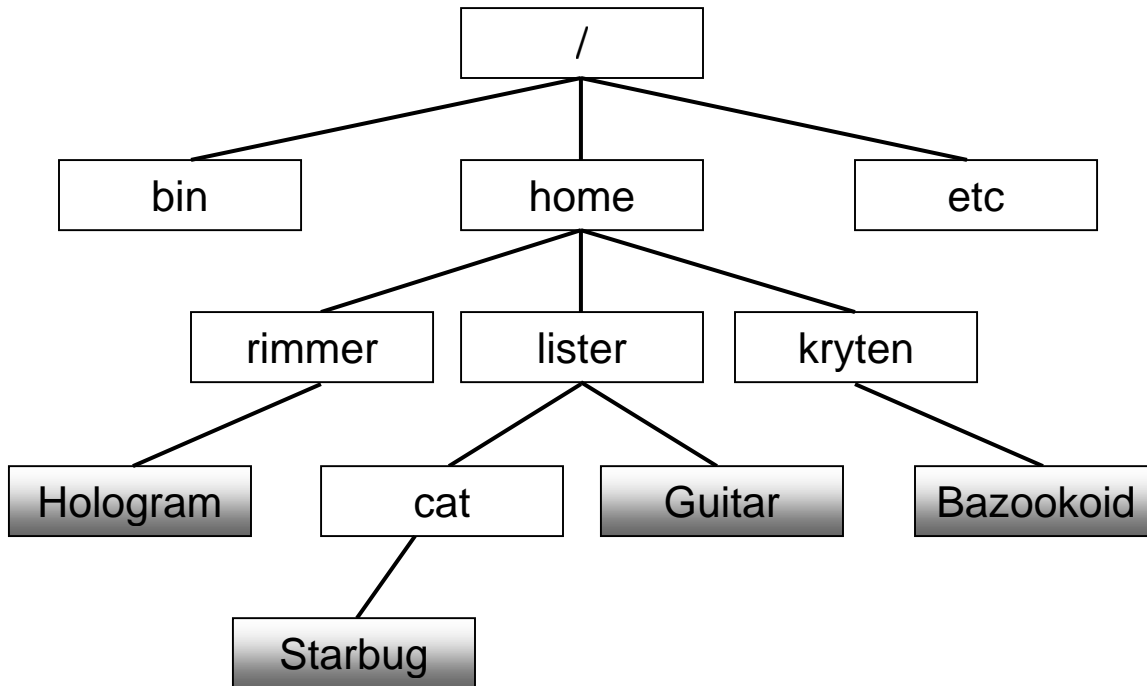
- "~[*username*]" represents the absolute path to a user directory
- "~/ " represents the absolute path to your user directory

Note

- "Links" (created with "ln") totally mess this structure up

Directory Structure

- Lower files are "contained" in the upper ones (directories)
- Starting container is called the "root", symbolized by "/"
- In example below ordinary files are capitalized



Pathnames

- Your username is "lister"
- Your current directory is "cat"

Absolute pathnames

- bin
- Guitar

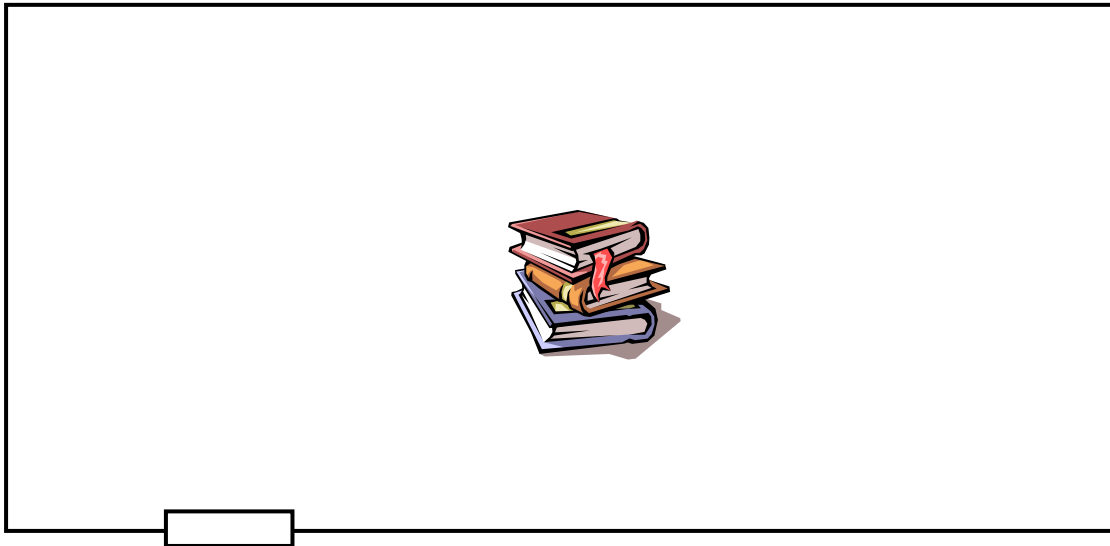
Relative pathnames

- Starbug
- home
- Guitar
- rimmer

Where could you use user directory references? (~)

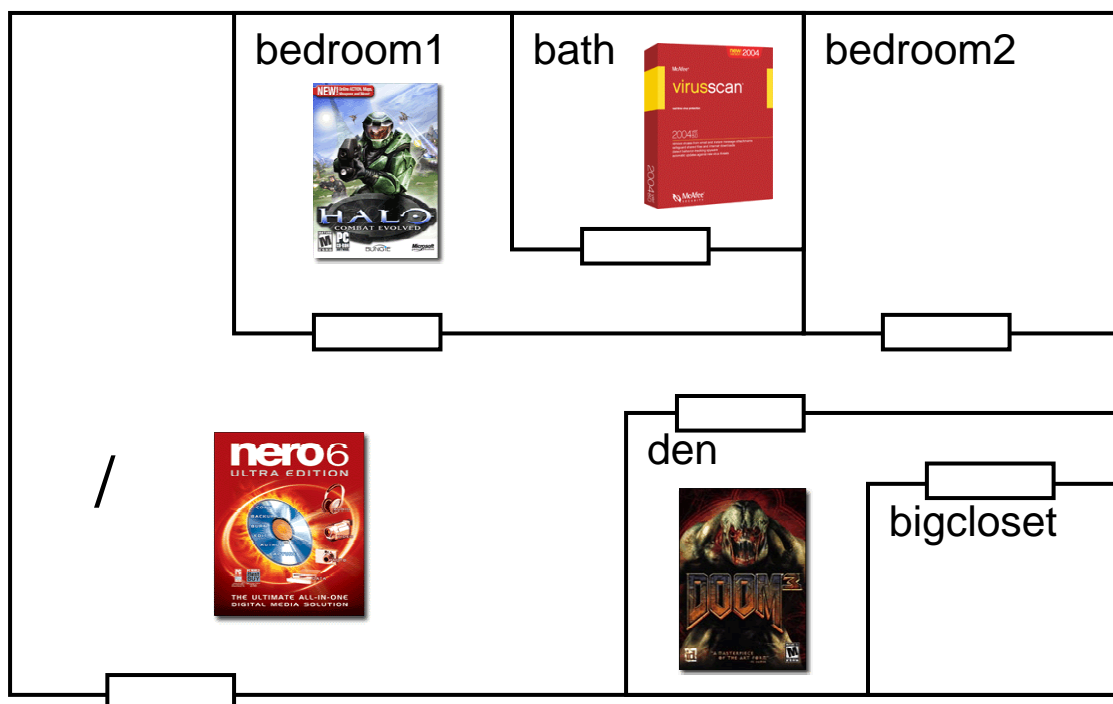
Directory Structure (Alternative)

- Think of "root" as a house (named "/") without the rooms



Directory Structure

- Think of directories as rooms within the house (each with a name)
- Some rooms may have other rooms within them (such as a bathroom inside a bedroom)
- Then files are simply objects (like books) lying on the floor in a room



Absolute Pathnames

- Simply give directions on how to get to a room, assuming you were standing outside the house
- Do not forget the separators
- Go to the house ("/"), then go to "bedroom1", then ("/") go to "bath" to get the program "virusscan"

Relative Pathnames

- Give directions on how to get to a room, assuming you are already standing inside a room in the house
- If you are in "bedroom2"
- Go out of the room (".."), then go to "den" to get the program "Doom3"

Absolute Pathnames

nero6:

virusscan:

doom3:

Relative Pathnames - From "bedroom1"

virusscan:

nero6:

doom3:

Commands

- Name typically represents a UNIX program (file) located somewhere (ex. `"/usr/bin"`)
- Typical command structure:

```
commandname [flags] [parameters]
```

Flags

- Commands often accept one or more flags after command name
- Each flag starts with "-", and is separated from other flags by spaces
- Individual flags may be combined with a single "-"

```
ls -l -a  
ls -la
```

Parameters

- Commands often accept one or more parameters after command name
- Parameters are typically pathnames representing files affected by command
- Parameters are separated by spaces (otherwise the command name would be `"cp@&a0had"` not `"cp"`)

```
cp /home/spike/original.txt copy.txt
```

↑
There is a space
here, eh?

↑
Here too, knob

Some File Commands

ls	list files
cat	view file contents
more	view file contents (pause each screen)
touch	creates file / updates time stamp
cp	copy file to a new file
mv	move file to a new directory, rename file
rm	delete file

Some Directory Commands

pwd	display absolute pathname to current directory
mkdir	create directory
rmdir	remove directory
cd	navigate directories

Other commands

man	displays "manual" page on commands
ln	links a filename to an actual file in a different location

Command Reference

```
ls [-a][-l][-p][-r][-R][-x] [pathname]
```

Description: Lists the files in a directory.

Options:

- [-a] Display all files
- [-l] Displays all information
- [-r] Reverses order
- [-R] Includes sub-directories

Examples:

```
ls  
ls -al  
ls -al *.exe
```

touch filename

Description: Immediately creates an empty file, or updates the time stamp on an existing file.

```
cp [-i][-R][-r] srcfile directory[/newfile]
```

Description: Copies the contents of a file or directory to another file or directory.

Options:

- [-i] Ask before you replace
- [-R] Copy directories and contents
- [-r] Same as [-R]
- *srcfile* - File you want to copy
- *directory* - New location of file
- *newfile* - New name of file

Example:

```
cp .plan .plan.backup  
cp -r ~/public_html/* /temp/
```

```
mv [-i] oldfilename newfilename  
mv [-i] oldfilename directory[/newfilename]
```

Description: Renames or moves a file from one directory to another, either with the same name or a different one. Note the original file (and name) will no longer exist. This is not a copy.

Options:

- [-i] Prompt you before replacing a file
- *oldfilename* - existing file.
- *newfilename* - new name of the file
- *directory* - location of new file

Example:

```
mv this.out that.in  
mv this.out PhDResearch\  
mv this.out PhDResearch\that.in
```

rm [-i][-r] *filename*

Description: Deletes files permanently. There is no recovery or undo command for this deletion.

Options:

- [-i] Prompt you before replacing a file
- [-r] Recursive, deletes an entire directory and all contents and subdirectories. Very serious.
- *filename* - the name of the file that you want to delete. Wildcards are allowed.

Example:

```
rm this.out
rm -i this.*
rm *.*
```

more [-s][-u][*filename*]

Description: Displays the contents of a file to standard output. The output is paused after each full screen.

Options:

- [-s] Squeezes out extra blanks
- [-u] Ignores "_" or backspace

Example:

```
more .tcshrc
```

less [*filename*]

Description: Less is more.

wc [-c][-l][-w][*filename*]

Description: Counts characters, lines, or words in a file.

Options:

- [-c] Number of characters
- [-l] Number of lines
- [-w] Number of words

pwd

Description: Displays the current directory location.

cd [*directory*]

Description: Changes current working directory.

Examples:

```
cd classes
cd ~dchang
cd /usr/include/java/classes
cd ../public_html/classes
```

mkdir *directory*

Description: Creates a new directory. Use absolute or relative pathnames.

Examples:

```
mkdir classes
mkdir ~dchang/classes/cop3502
```

rmdir *directory*

Description: Removes the specified directory. Use absolute or relative pathnames.

Examples:

```
rmdir classes
rmdir ~dchang/classes/cop3502
```

man *command*

Description: Displays manual page for a command.

```
ln [-s] [target] [link_name]
```

Description: Creates a "link" to a file or directory in a different locatin, that can be accessed using a link name. By default "hard" links are created, but the majority of the time a "symbolic" link would be desired.

Deleting a symbolic link will (should) not delete the target.

Options:

- [-s] Create symbolic instead of hard link. This is what you want.
- [*target*] Pathname to the file or directory to link to
- [*link_name*] The filename to be created in the current directory that will link to "target"

Example:

```
mkdir realdir
touch realdir/realfile
ln -s realdir ldir
ln -s realdir/realfile lfile
```

```
pico lfile (would actually edit "realdir/realfile")
cd ldir (would actually change current dir to "realdir/")
```