



# UNIX filesystem

---

CS 2204

Class meeting 2



# UNIX filesystem

---

- The filesystem is your interface to
  - physical storage (disks) on your machine
  - storage on other machines
  - output devices
  - etc.
- *Everything* in UNIX is a file (programs, text, peripheral devices, terminals, ...)
- There are no drive letters in UNIX! The filesystem provides a *logical* view of the storage devices



# Working directory

---

- The current directory in which you are working
- `pwd` command: outputs the absolute path (more on this later) of your working directory
- Unless you specify another directory, commands will assume you want to operate on the working directory



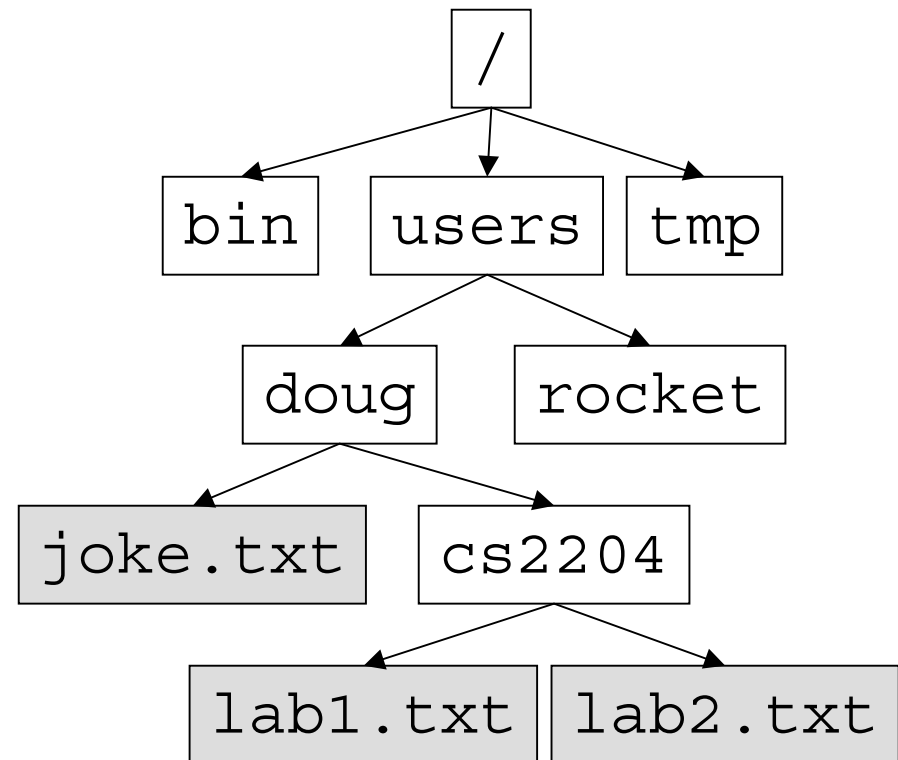
# Home directory

---

- A special place for each user to store personal files
- When you log in, your working directory will be set to your home directory
- Your home directory is represented by the symbol ~ (tilde)
- The home directory of “user1” is represented by `~user1`

# UNIX file hierarchy

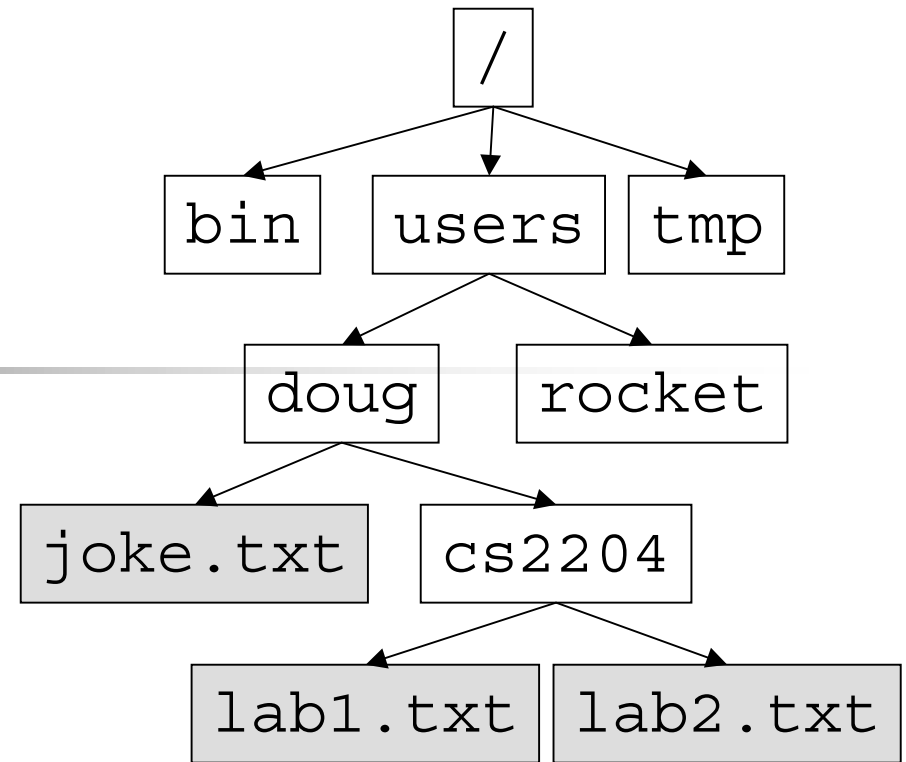
- Directories may contain plain files or other directories
- Leads to a tree structure for the filesystem
- Root directory: /



# Path names

- Separate directories by /
- Absolute path
  - start at root and follow the tree
  - e.g. `/users/doug/joke.txt`
- Relative path

`../joke.txt`    `~/joke.txt`    `~doug/joke.txt`





# Changing directories

---

- Change the working directory with the `cd` command
  - `cd <dir_name>`
  - Use absolute or relative path names
  - `cd` by itself equivalent to `cd ~`
- Let's try putting it all together...



# Output of `ls -lF`

---

```
total 4
```

```
lrwxr-xr-x 1 bowman user 18 Aug 28 13:41 home ->
  /usr/people/bowman/
```

```
-rw-r--r-- 1 bowman user 94 Aug 28 13:42 nothing.txt
```

```
drwxr-xr-x 2 bowman user  9 Aug 28 13:40 test_dir/
```

↑           ↑           ↑           ↑           ↑           ↑  
Permissions Owner Group           Modify date File name  
File type



# Types of files

---

- Plain (-)
  - Most files
  - Includes binary and text files
- Directory (d)
  - A directory is actually a file
  - Points to another set of files
- Link (l): A pointer to another file or directory
- Special: e.g. peripheral devices



# Creating links

---

- `ln -s <existing_file>  
<link_name>`
- This command creates a symbolic link
- The file “link\_name” will be a pointer to the “existing\_file” which may be in another directory or even on another physical machine



# File permissions

---

- Permissions used to allow/disallow access to file/directory contents
- Read (r), write (w), and execute (x)
- For owner, group, and world (everyone)
- `chmod <mode> <file(s)>`
  - `chmod 700 file.txt`
  - `chmod g+rw file.txt`
- Let's try some examples



# File ownership

---

- Each file has a single owner
- `chown` command can be used to change the owner (usually only root user can use this command)
- There are also various *groups* to which users can belong
- Groups may have different permissions than everyone else



# File modification date

---

- Last time the file was changed
- Useful information when
  - There are many copies of a file
  - Many users are working on a file
- `touch` command can be used to update the modification date to the current date, or to create a file if it doesn't exist



# Looking at file contents

---

- `cat <filename(s)>`
  - “concatenate”
  - output the contents of the file all at once
- `more <filename(s)>`
  - Output the contents of a file one screen at a time
  - Allows forward and backward scroll and search



# Moving, renaming, copying, and removing files

---

- `mv <file1> <file2>` (rename)
- `mv <file1> <dir>` (move)
- `mv <file1> <dir/file2>` (move & rename)
- `cp <file1>`  
    [`<file2>` | `<dir>` | `<dir/file2>`] (copy)
- `rm [-i] <file(s)>` (remove)
- Let's try some examples...



# Creating and removing directories

---

- `mkdir <dir_name>`
  - Create a subdirectory of the current directory
- `rmdir <dir_name>`
  - Remove a directory (only works for empty directories)
- `rm -r <dir_name>`
  - Remove a directory and all of its contents, including subdirectories



# Wildcards in file names

---

- All of the commands covered here that take file names as arguments can also use wildcards
  - \* for any string, e.g. \*.txt, obj\*, a\*.\*
  - ? for any character, e.g. doc?
  - [] around a range of characters, e.g. [a-c]\*



# Getting help on UNIX commands

---

- These notes only give you the tip of the iceberg for these basic commands
- `man <command_name>` shows you all the documentation for a command
- `apropos <keyword>` shows you all the commands with the keyword in their description