ECS 98F - Introduction to the Command Line

Grant Gilson & Rebekah Grace

UCDAVIS
COMPUTER SCIENCE

Agenda

Today's Lecture

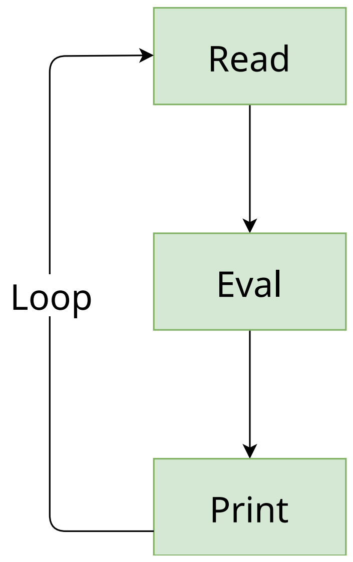
- What the CLI can do
- Introduction to Bash and the filesystem
- Where to find help on the CLI

Defining the CLI

An interface to the system

What is a shell:

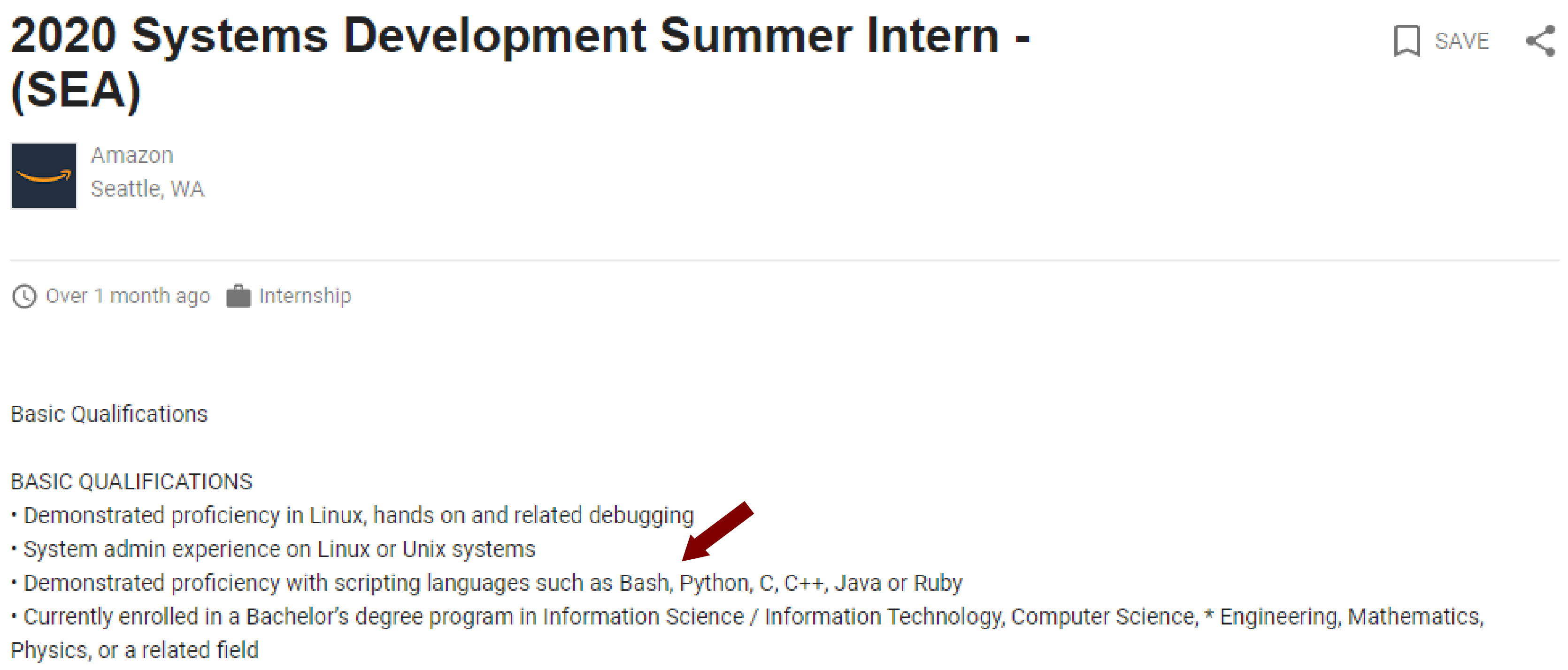
A program that accepts commands and returns the results.



Examples of shells

- Windows command prompt
- Python
- Bash
- zsh

Why learn the CLI



2020 Systems Development Summer Intern - (SEA)

Amazon
Seattle, WA

Over 1 month ago
Internship

Basic Qualifications

BASIC QUALIFICATIONS
• Demonstrated proficiency in Linux, hands on and related debugging
• System admin experience on Linux or Unix systems
• Demonstrated proficiency with scripting languages such as Bash, Python, C, C++, Java or Ruby
• Currently enrolled in a Bachelor's degree program in Information Science / Information Technology, Computer Science, * Engineering, Mathematics, Physics, or a related field

Why learn the CLI

Scenario:

- You are working on a company laptop/computer that you cannot install your favorite editor on
- You are working with shared computer without a GUI
- You are working with limited internet resources and need to lookup documentation

Pros:

- Finer control of the operating system
- Finer control over program behaviors
- No layers of GUI's to traverse through

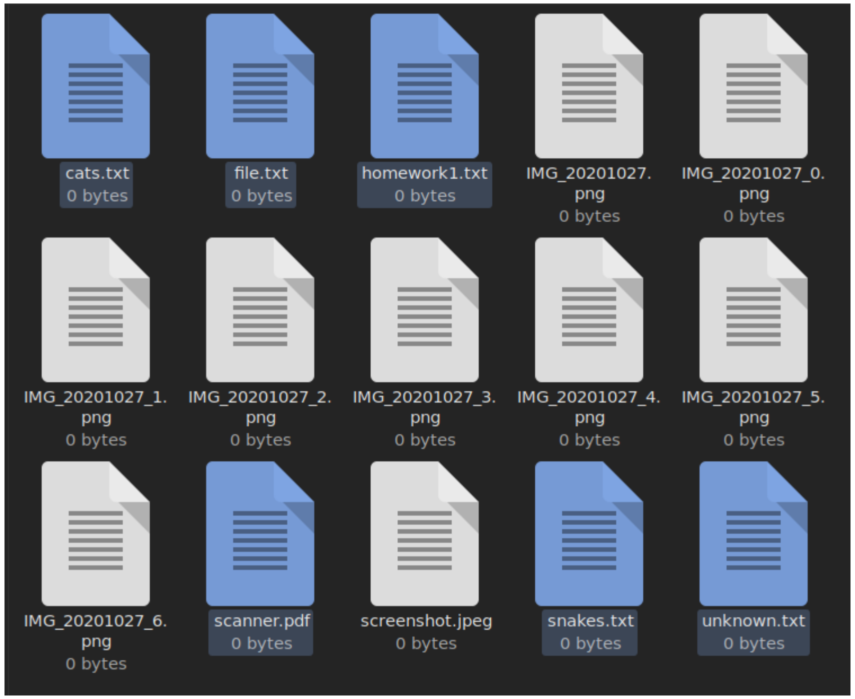
Cons:

- Does not protect you from yourself

Why learn the CLI

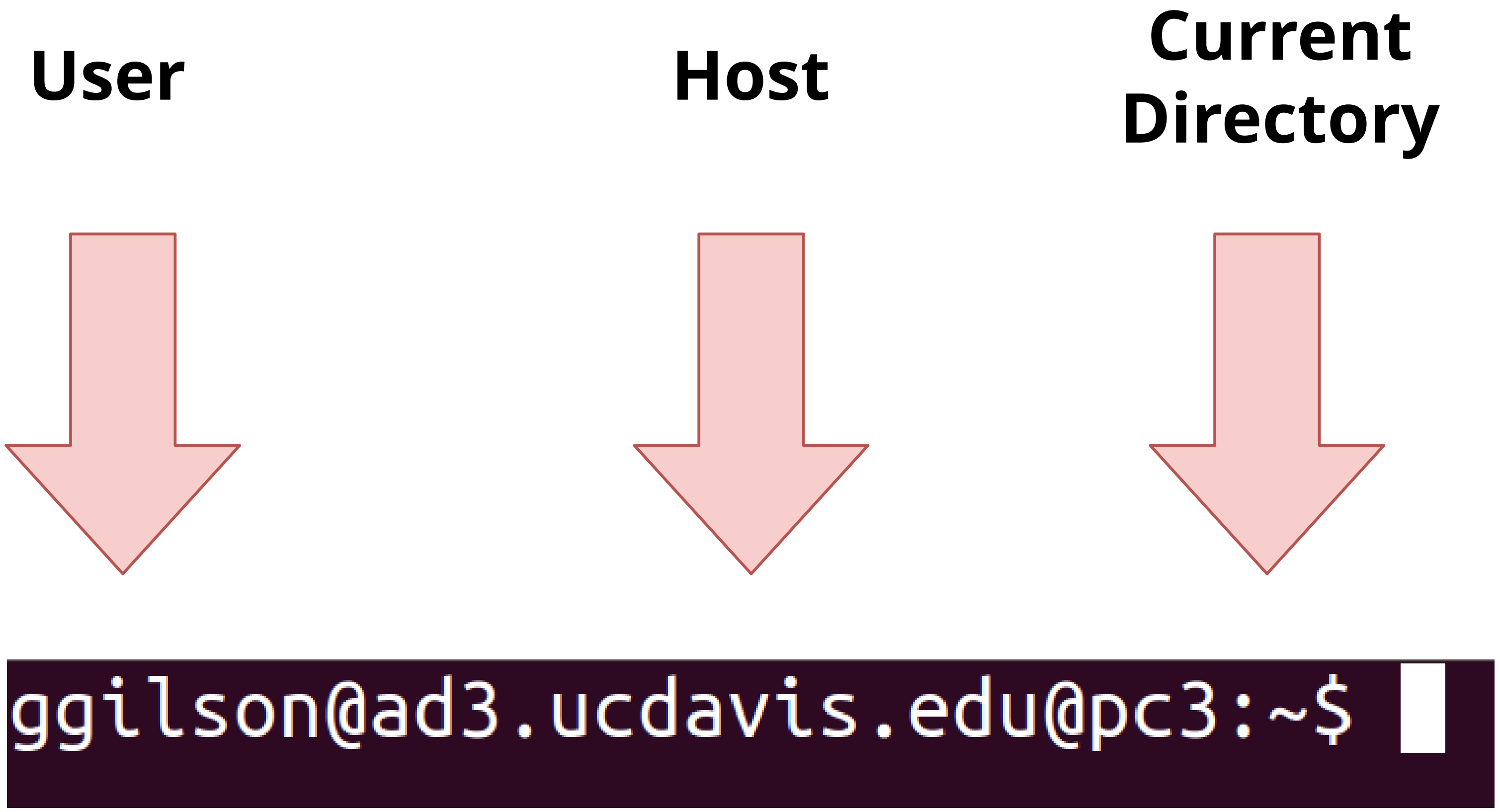
Substituting with the CLI

- Text Editor or IDE
  - Read or write files
  - Code compilation or execution
  - Find and replace
- File explorer
  - Creating files
  - Organizing your project



Introduction to Bash

Discecting the prompt



```
ggilson@ad3.ucdavis.edu@pc3:~$
```

The `~` character is an alias for your home directory

Filesystem overview

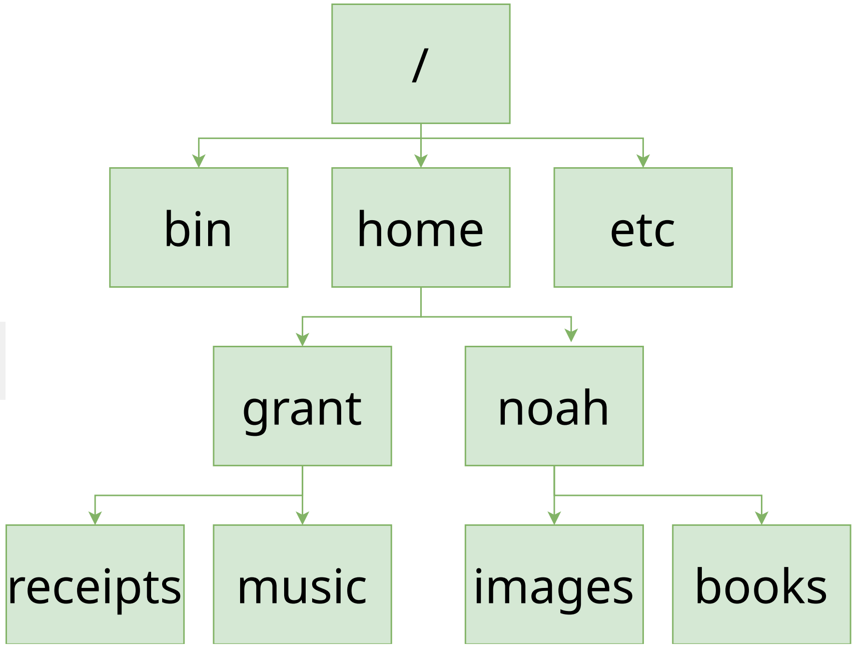
Tree structure

Command

```
grant@pc3.cs.ucdavis.edu:~$ pwd
```

Output

```
/home/grant
```



Valid path or not?

- `/home/noah/books`
- `/home/grant/noah`

Filesystem overview

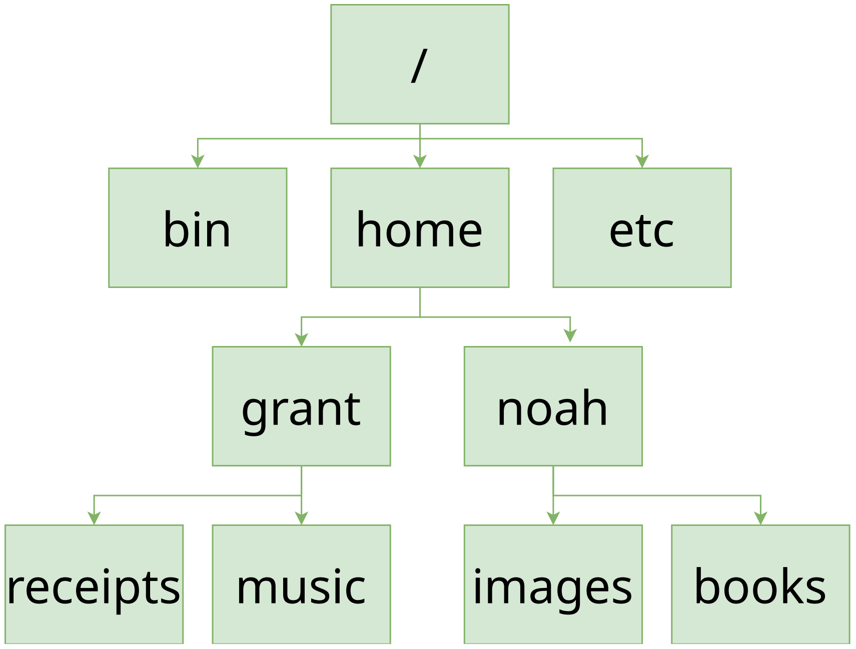
Tree

With `tree` we can see how our child files are organized from our current directory

```
grant@ubuntu:/home$ tree <path>
```

```
.
├── grant
│   └── receipts
|   └── music
├── noah
|   └── books
|   └── images

4 directories,
```



Types of filepaths

Absolute

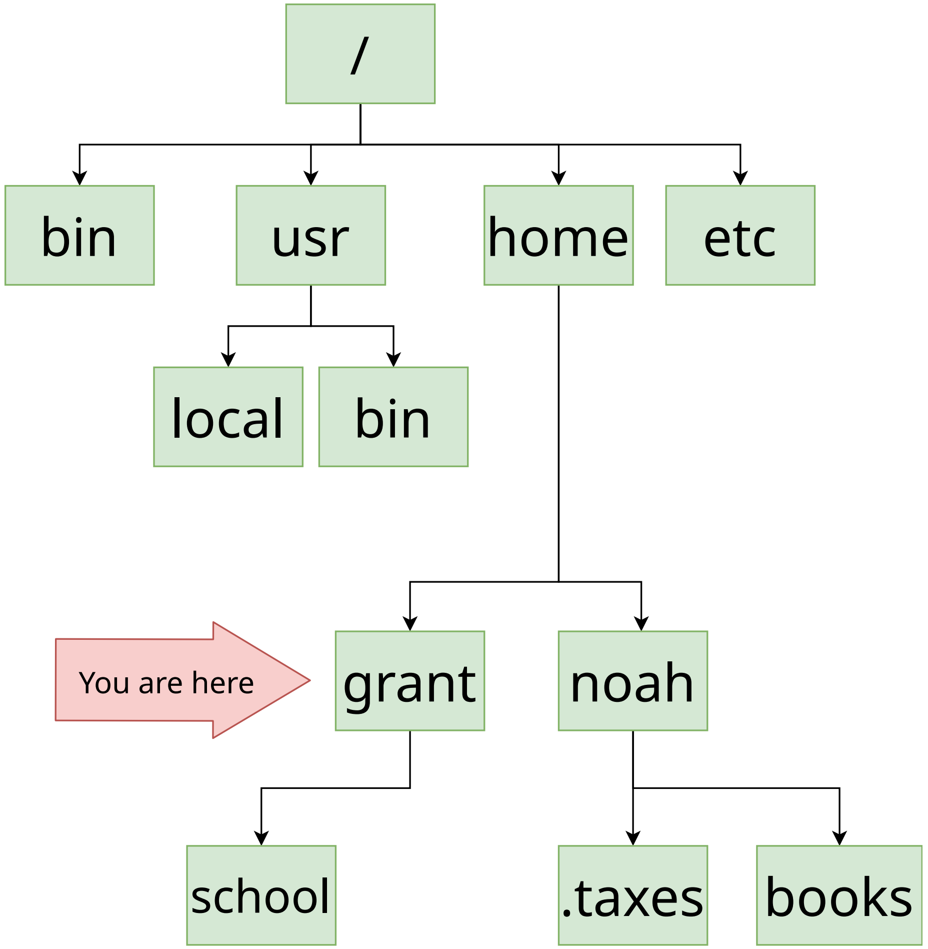
- `/home/grant/school`
- `/home/noah/books`

Relative

- `./school`
- `../noah/.taxes`
- `../../etc`

Special Entries

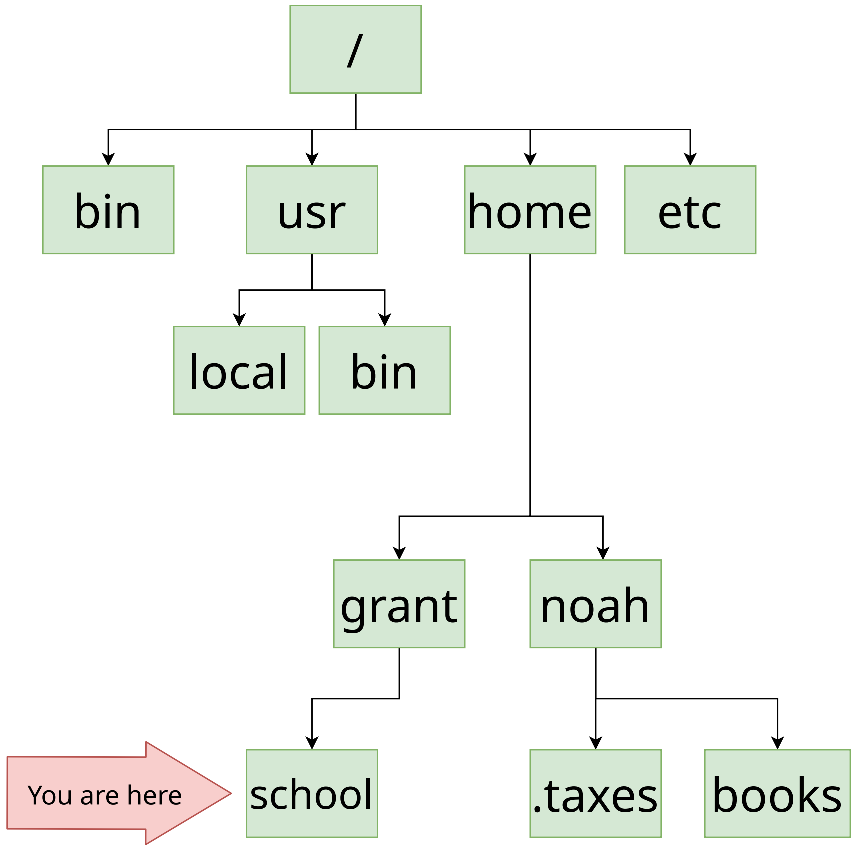
- `./` refers to your current location
- `../` refers to one directory above the current location



Types of filepaths

More filepath examples

How do we get to `local`?



Types of filepaths

More filepath examples

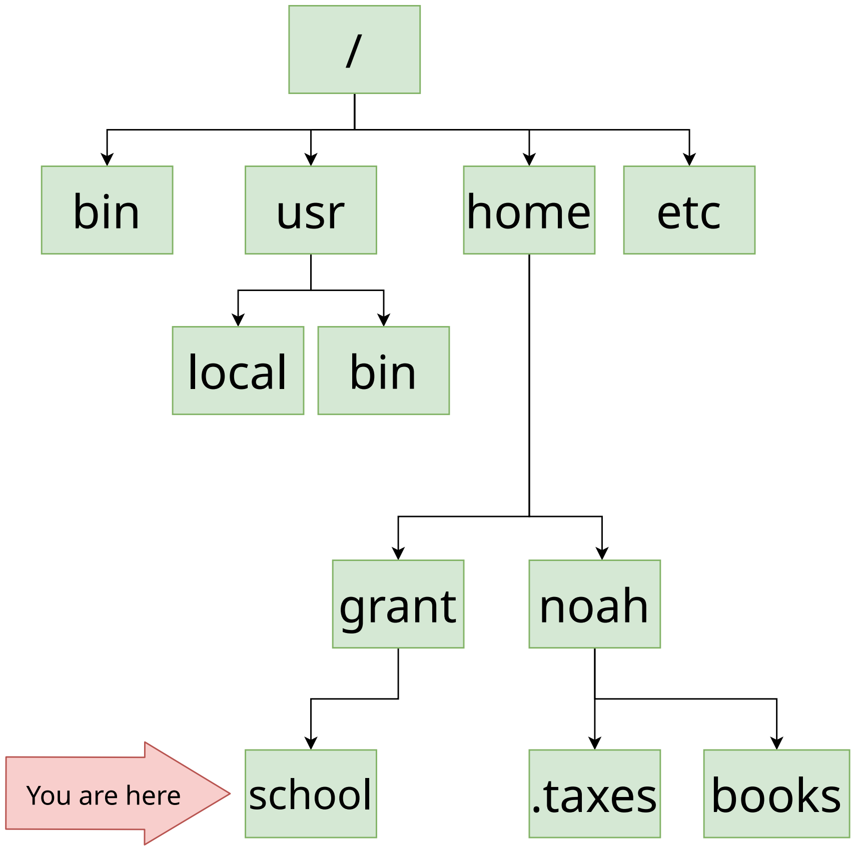
How do we get to `local`?

Absolute:

- `/usr/local`

Relative:

- `../../../usr/local`



Bash basics

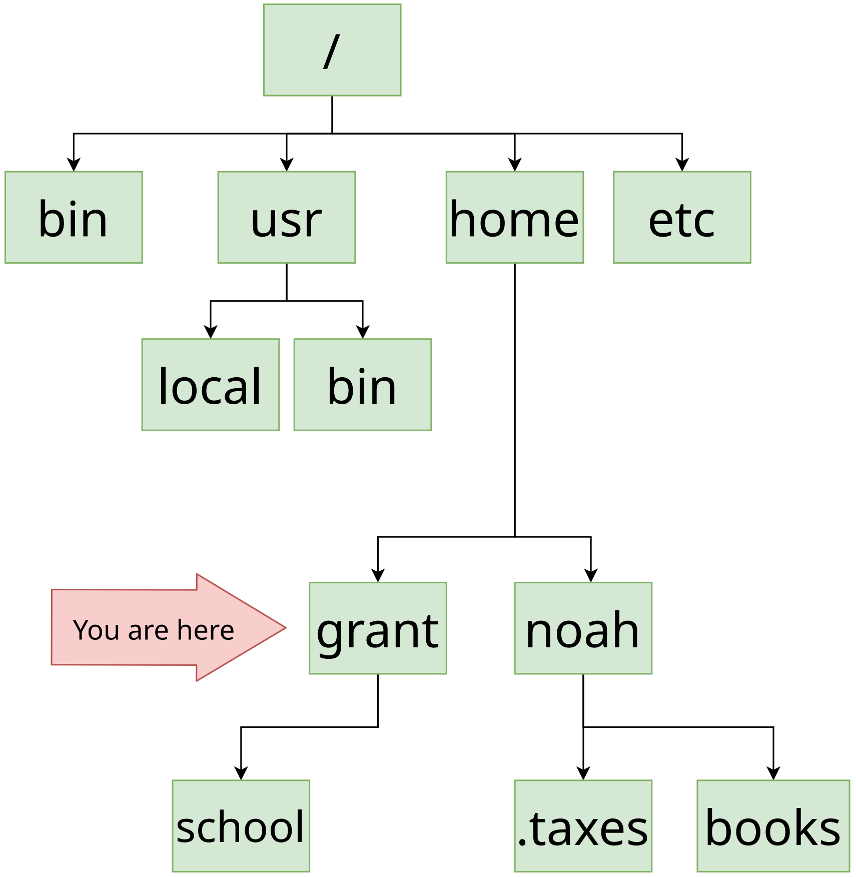
Traversing the filesystem

Command

```
$ cd <path>
```

Example

```
$ pwd
/home/grant
$ cd ../noah/.taxes
$ pwd
/home/noah/.taxes
$ cd /bin
$ pwd
/bin
```



Bash basics

Listing files

Command

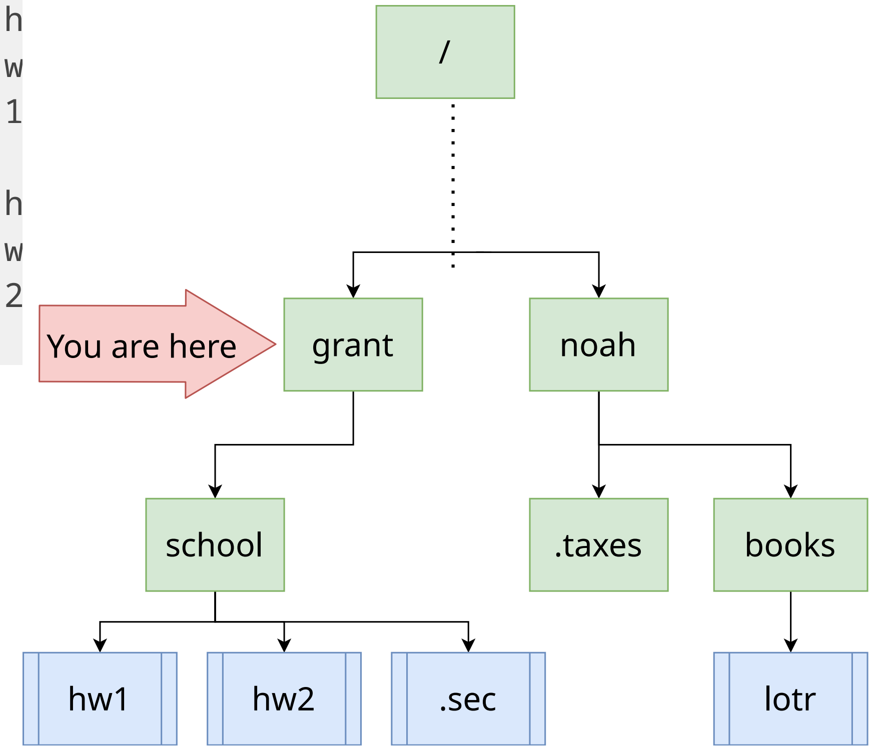
```
$ ls <path>
# defaults to working directory
```

Example

```
$ ls school
```

```
h
w
1

h
w
2
```



Bash basics

Listing hidden files

Command

```
$ ls -a <path>
```

```
. .. HW1 HW2 .sec
```

Command

```
$ tree -a <path>
```

```
grant
└── school
    ├── hw1
    ├── hw2
    └── .sec

1 directory, 3 files
```

Bash basics

Listing hidden files

Command

```
$ ls -la ./School
total 8
drwxrwxr-x 2 grant grant 4096 Oct 22 00:03 .
drwxrwxr-x 3 grant grant 4096 Oct 21 22:39 ..
-rw-rw-r-- 1 grant grant    0 Oct 21 23:30 HW1
-rw-rw-r-- 1 grant grant    0 Oct 21 23:30 HW2
-rw-rw-r-- 1 grant grant    0 Oct 22 00:03 .sec
```

- Flags are combinational
- Flag order does not matter

Bash basics

Managing files

Command

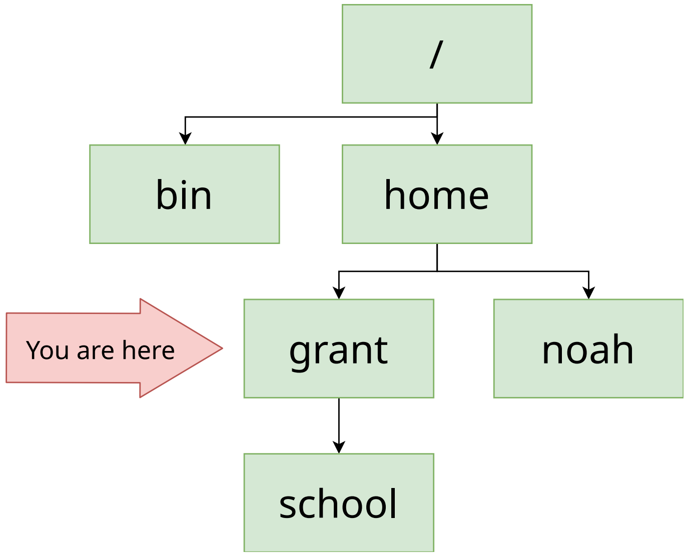
```
$ mkdir <flags> <path(s)>
```

Behavior

- Make directory(s)

Example

```
$ mkdir /home/grant/school/ecs120
$ mkdir ~/school/ecs50
```



Bash basics

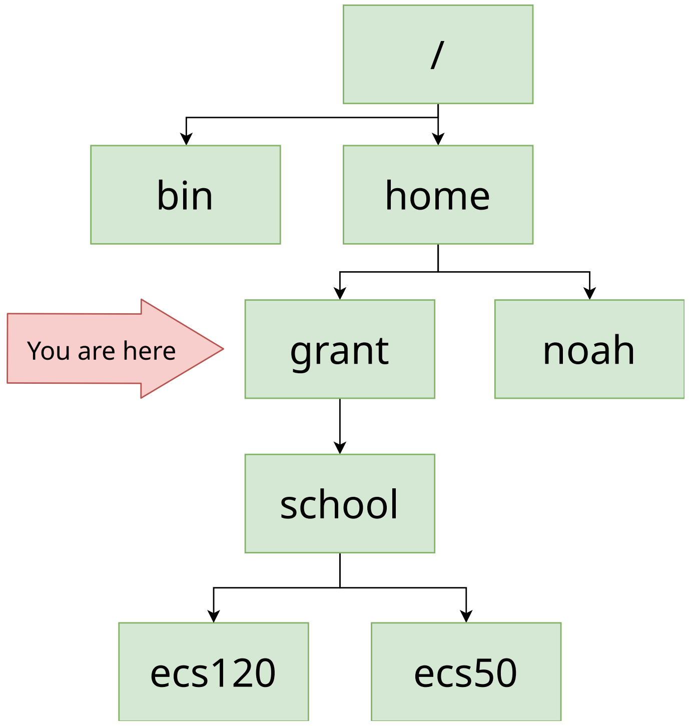
Managing files

Command

```
$ mkdir <flags> <path(s)>
```

Example

```
$ mkdir /home/grant/school/ecs120
$ mkdir ~/school/ecs50
```



Bash basics

Managing files

Command

```
$ mv <flags> <sourcePath(s)> <destinationPath>
```

Behavior

- Move files from source to destination
- Rename files
- Can overwrite existing files

Moving

```
$ ls
HW1 HW2 homework/
$ mv HW1 HW2 homework
      |   |          |
#   {sourceFiles} {destination}
$ tree
.
├── homework
│   ├── HW1
│   └── HW2
1 directory, 2 files
```

Rename

```
$ ls -a
HW1 HW2 .secret
$ mv .secret public.txt
$ ls
HW1 HW2 public.txt
```

Bash basics

Managing files

Command

```
$ cp <flags> <sourcePath(s)> <destinationPath>
```

Behavior

- Copy files/directories
- Can overwrite existing files

Example

```
$ cp .secret public
# copy in-place with new name
```

```
$ cp public /home/noah/
# copy file to directory
```

Bash basics

Managing files

Command

```
$ rm <flags> <filePath(s)>
```

Behavior

- Remove each specified file **permanently**
- Does not remove directories by default

Important Flags

- `-r`, remove directories and their contents recursively
- `-i`, prompt before every removal

Aliases

Bash shortcuts

Command

```
$ alias <aliasName>='<commandToRun>'
```

Behavior

- substitutes the alias name with the command to be run

Usage

- create shortcuts for tedious commands
- extend the default behavior of commands

```
$ alias rm='rm -i'          # confirm to delete file
$ alias rm='mv -t ~/.trash' # move file to a trash directory
$ alias cp='cp -b'          # make backup of destinationFile
$ alias mv='mv -u'          # move only if sourceFile is newer
$ alias hello="echo 'cow power'"
```

Reading files

Commands

```
$ cat <flags> <file>
$ less <flags> <file>
```

Behavior

- print contents of file to screen

Example

```
$ cat ~/.bashrc
```

Text Editors

Nano

Command

```
$ nano <flags> <file>
```

Behavior

- opens simple CLI text editor

Vim

Command

```
$ vim <flags> <file>
```

Behavior

- another text editor at your disposal

Finding more

Man pages

Command

```
$ man <command>
```

- system reference manuals
- contains details of all command behavior and flags
- documentation of c libraries

Example

```
$ man mkdir
$ man stdio
```

Finding more

Tldr pages

Command

```
$ tldr <flags> <command>
```

- list the typical uses of a command

Installation

```
$ sudo apt install tldr
```

Example

```
$ tldr vim
  Open a file:

      vim path/to/file

  Save and Quit:

      :wq<Enter>
```

Conclusion

- The CLI is a viable replacement to the GUI
- Tldr and Man are your friend