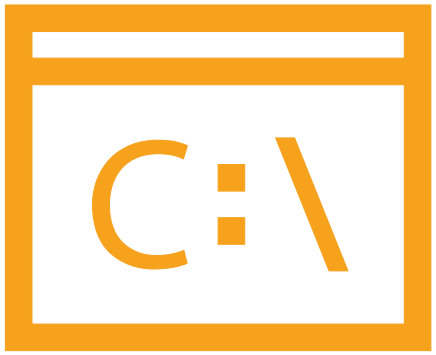


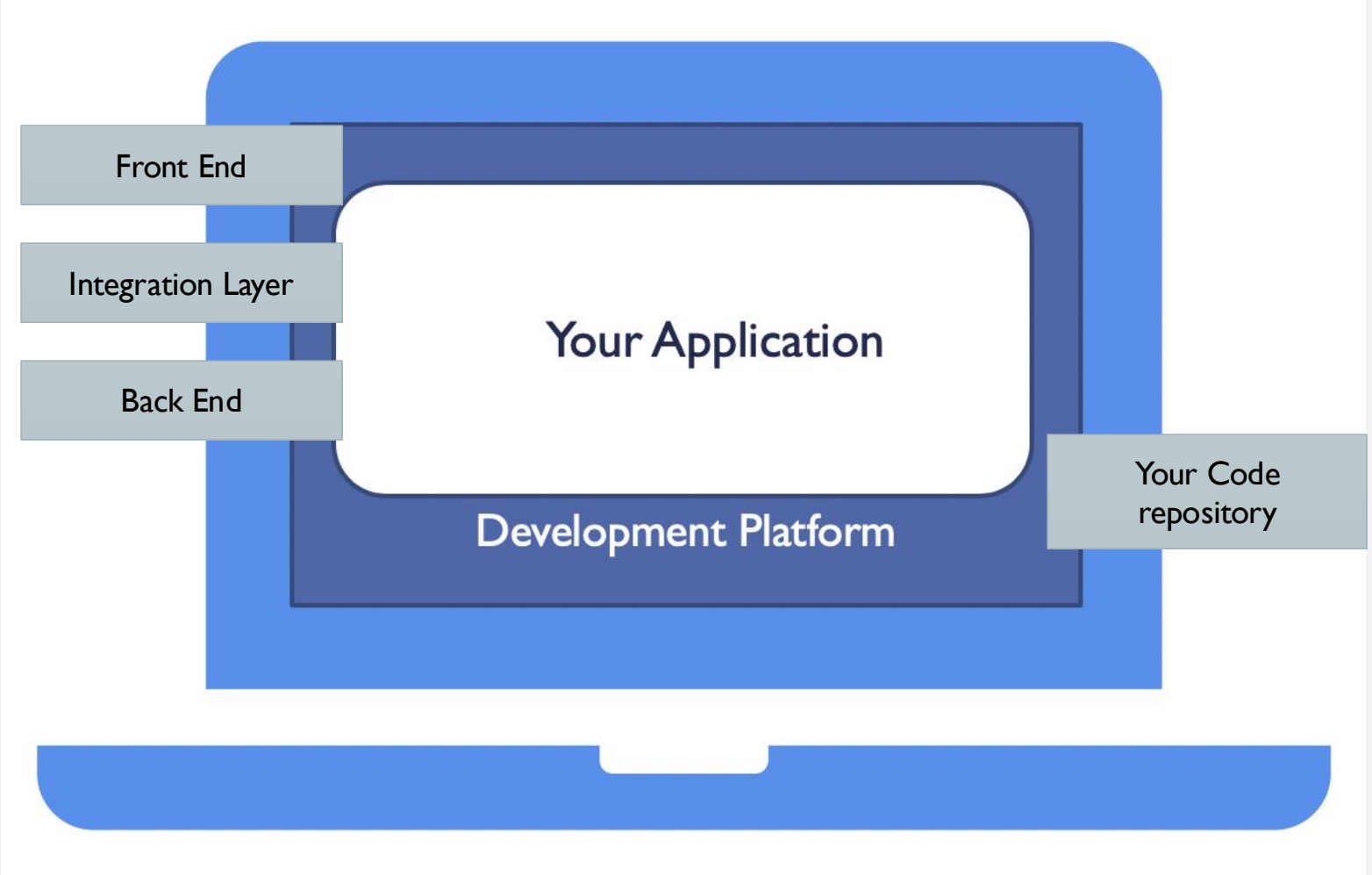
LINUX COMMAND SHELL, REGEX

Jan 15, 2025

AGENDA

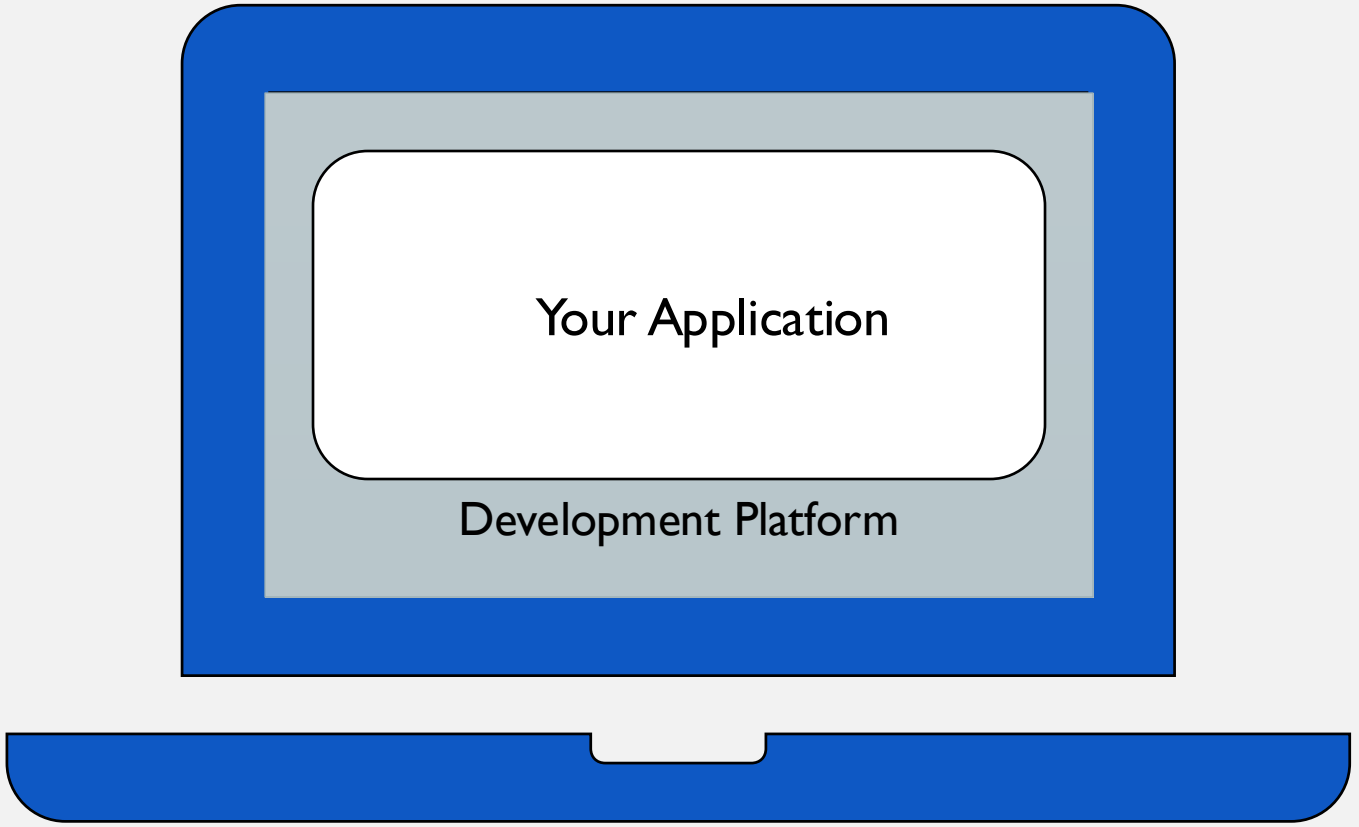
- Components and architectures for applications
- Operating Systems
- Terminal
- Shell commands
- Vim editor
- Regex

WHAT MAKES UP YOUR APPLICATION?

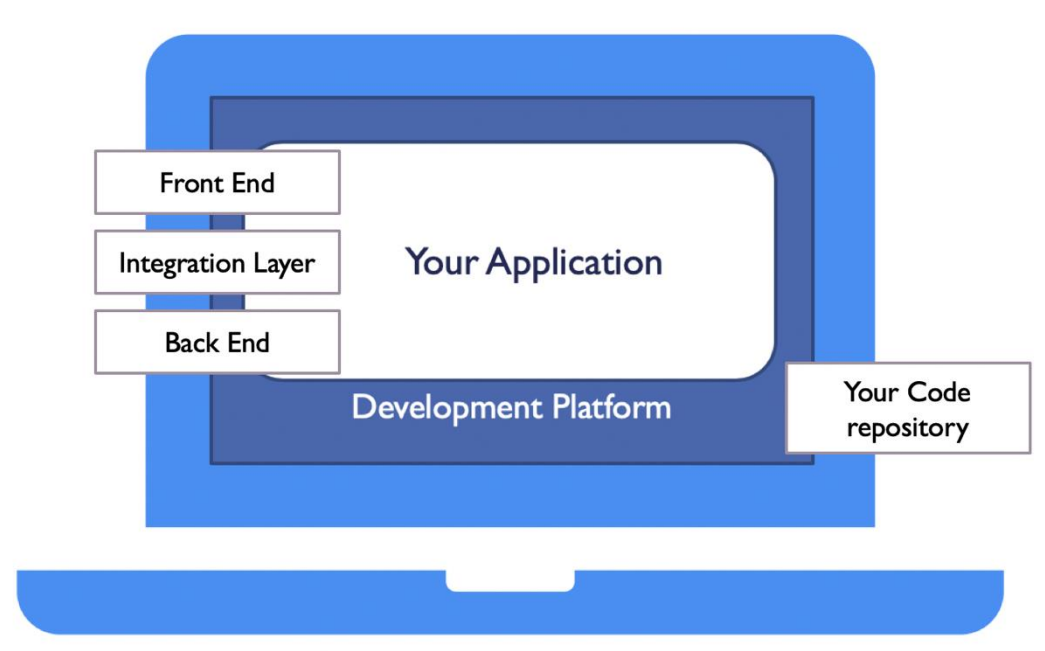


DIFFERENT ARCHITECTURES FOR HOSTING APPLICATION

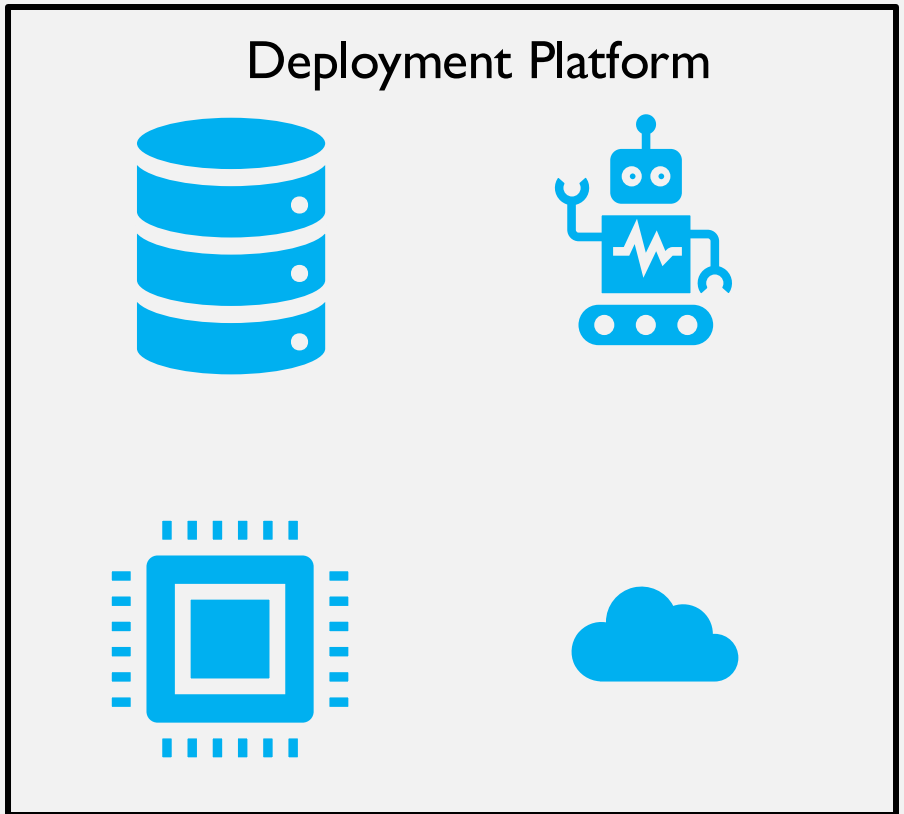
EVERYTHING RESIDES ON YOUR COMPUTER

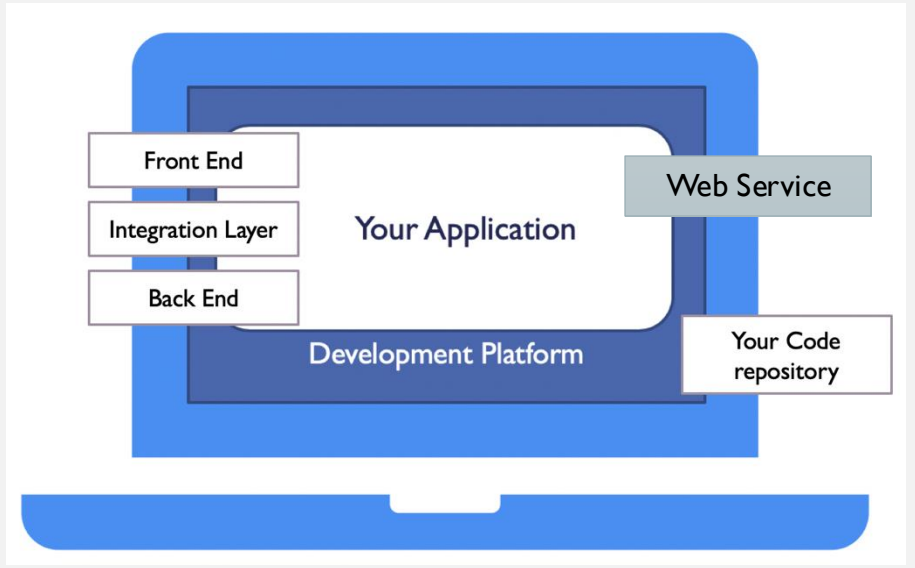


DEPLOY YOUR APPLICATION ON AN EXTERNAL RESOURCE



Your application build





Deployment Platform

Web Service

Your application build

BUILDING BLOCKS FOR SOFTWARE DEVELOPMENT

Operating system

A framework

Programming languages

Database

Repository

Web services

OPERATING SYSTEM

Hardware Components are only able to control themselves

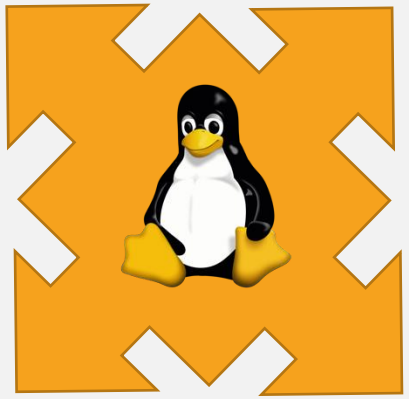
Components do not know how to interact with each other

How do we get the components to operate together?

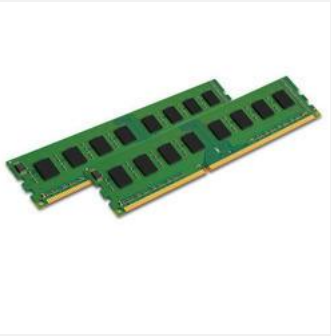
Make way, the Operating System is here!

COMPUTER ARCHITECTURE

CPU



RAM



I/O devices

Storage devices

HOW DO WE TALK TO THE OS?

SHELL!

HOW TO USE THE TERMINAL WINDOW

When you select the terminal icon on your desktop, you can create an interface with a shell (csh, sh, bash, zsh, ksh etc…)

Everything you type on the terminal window is sent back to the shell and the response from the OS is printed on your terminal window

Types of terminals include gnome-terminal, konsole, xterm, rxvt, kvt, nxterm, and eterm.

DEMO

GETTING HELP WITH THE COMMANDS

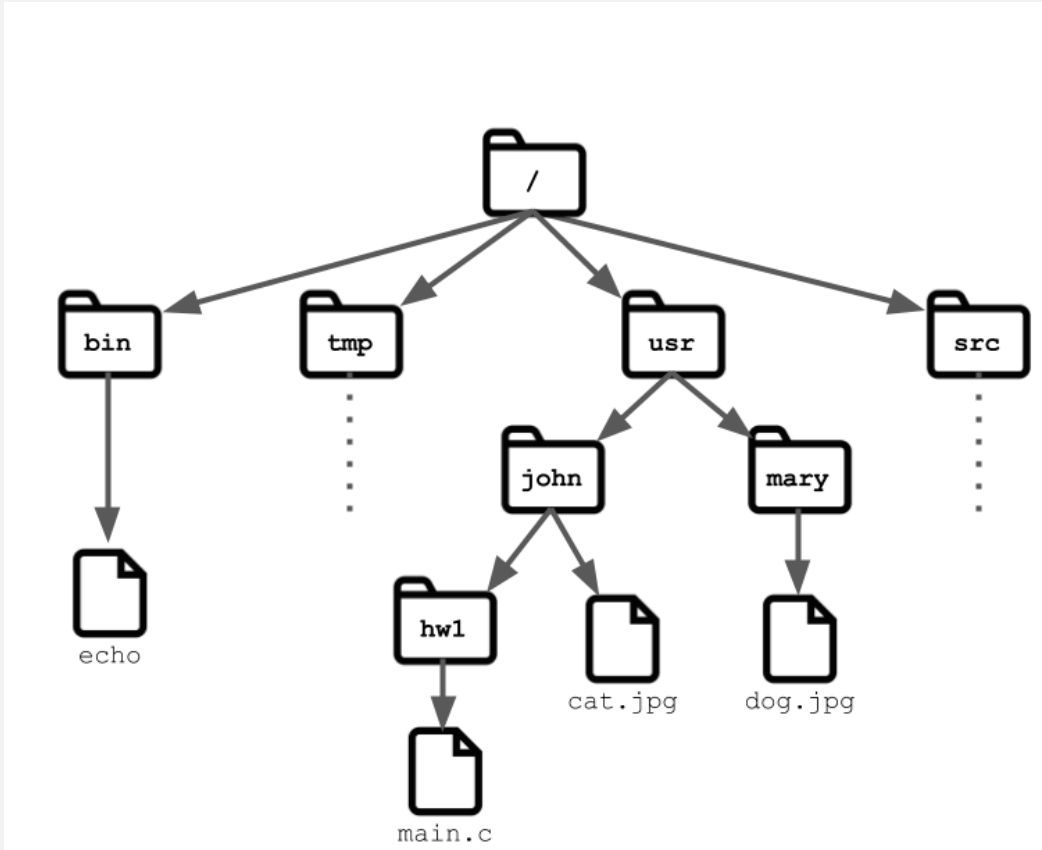
Most programs will understand the "--help" option and print information about the command's usage

The shell can also lookup the usage of a command by using the "man" command.

Let's see an example.

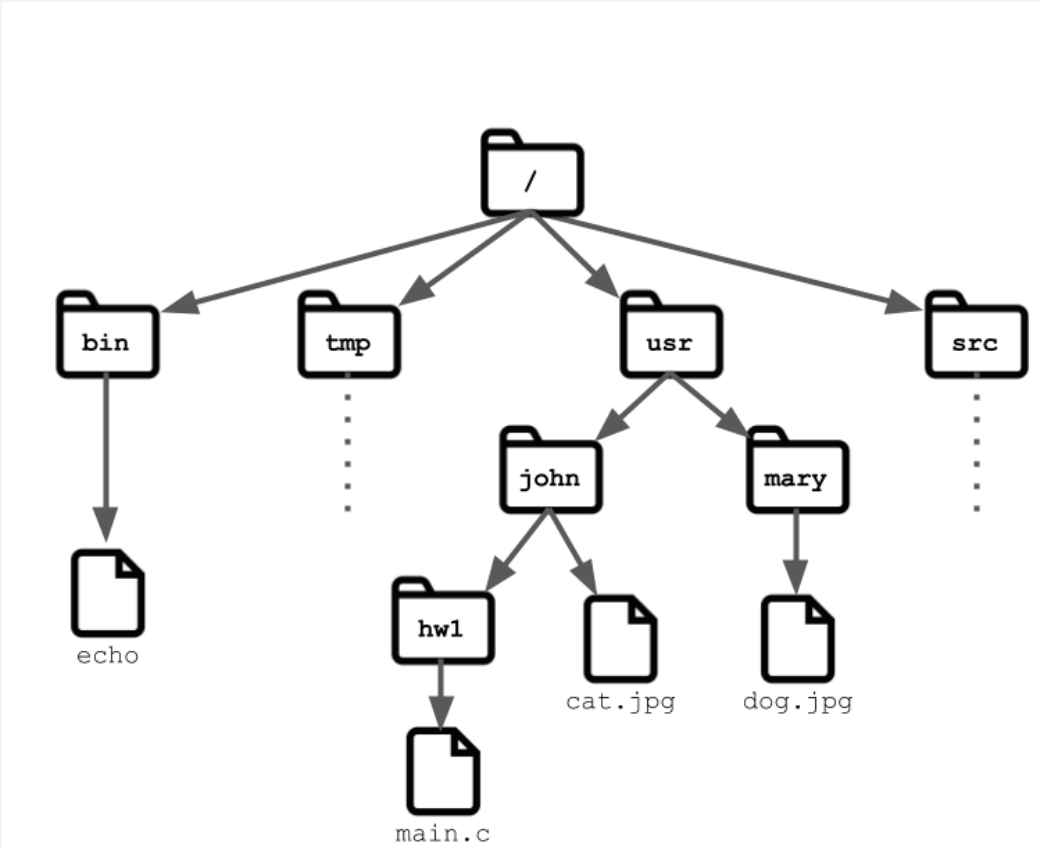
NAVIGATING THE FILE SYSTEM

- Current working directory
  - `pwd` – print working directory
  - `cd` – change directory
  - Relative path vs full path
- File System Commands
  - `ls` – list directory contents
  - `cp` – copy files
  - `rm` – remove files
  - `mv` – move files
  - `mkdir` – make directory
  - `rmdir` – remove directory



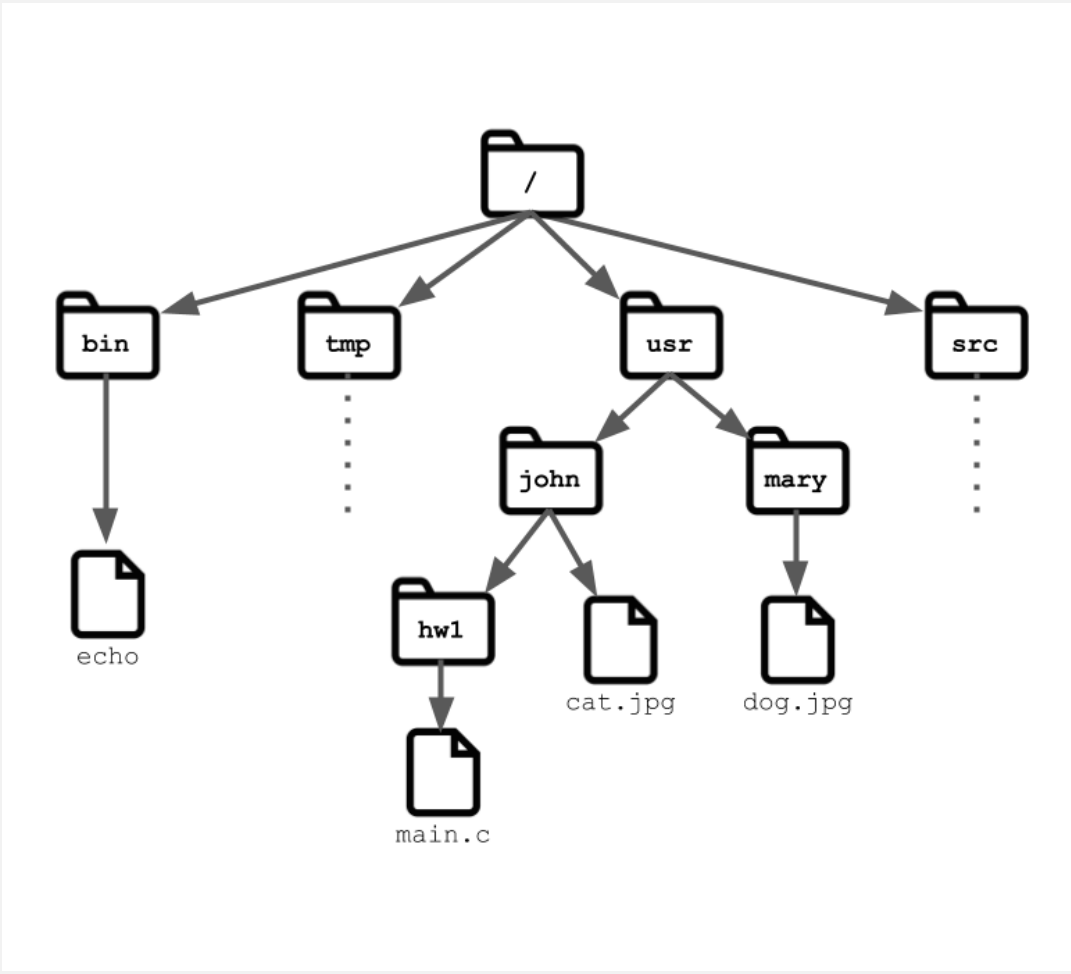
REFERRING TO FILES: ABSOLUTE PATHS

- List the directories on the path from the root (“/”)
- Separated by “/”



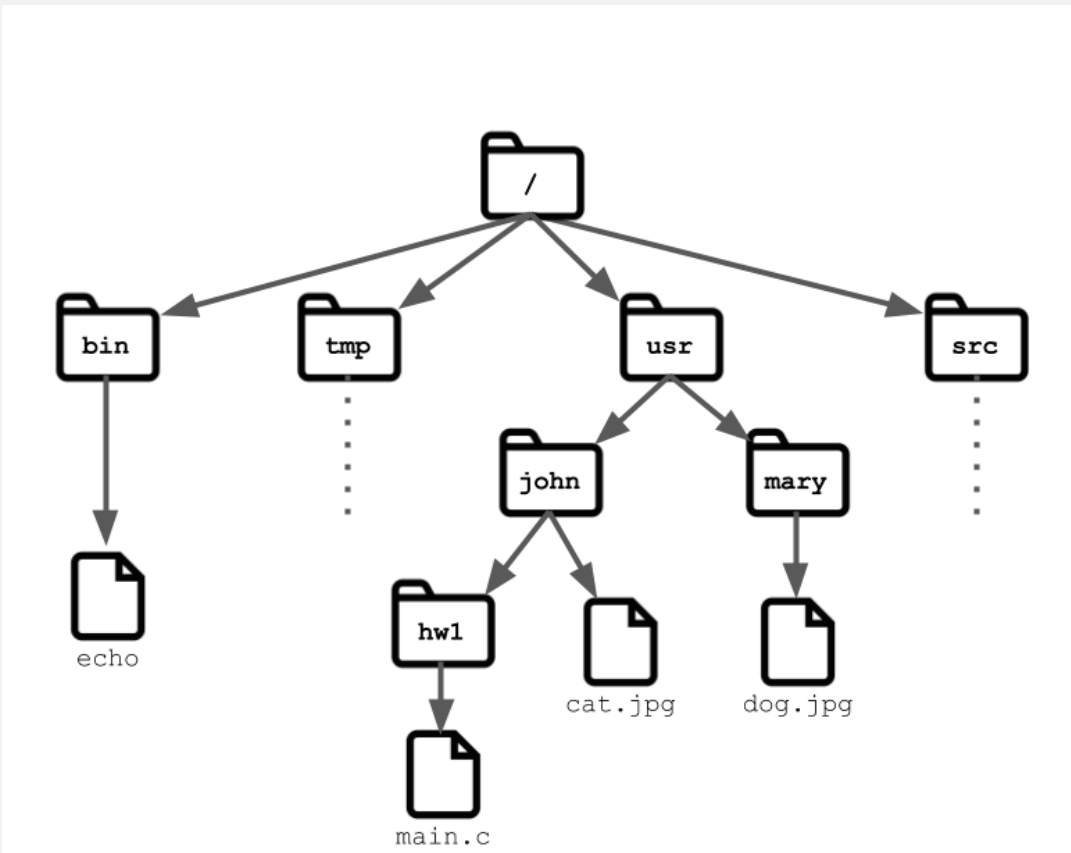
REFERRING TO FILES: RELATIVE PATHS

- Current directory (.)
- Parent directory (..)
- Relative path



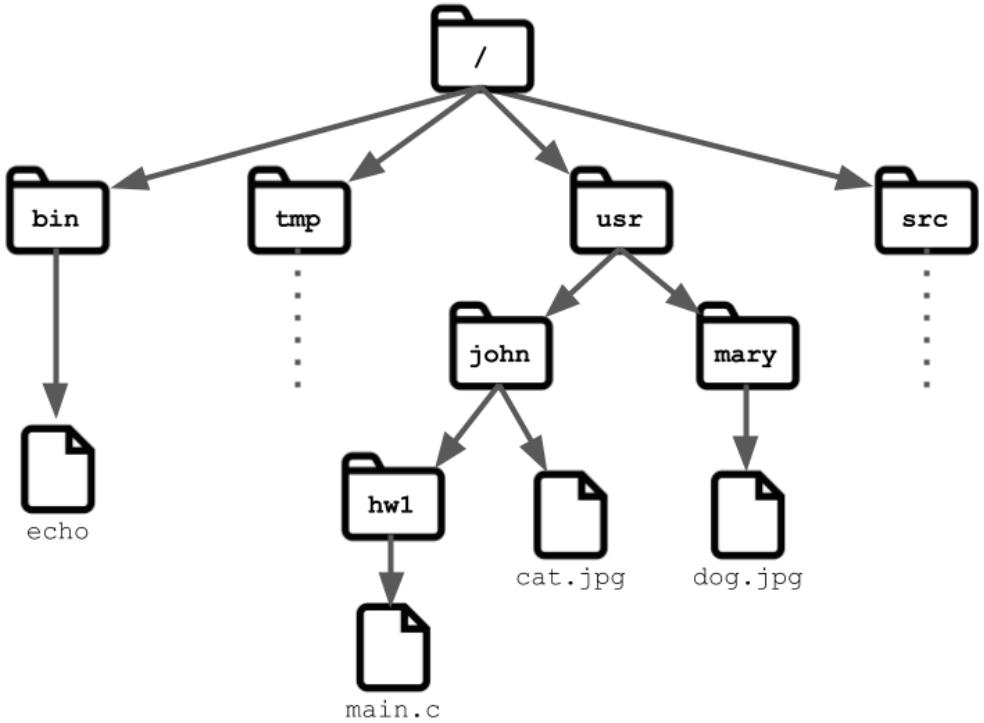
LET'S MAKE IT A BIT OF A CHALLENGE

- If you're in the "john" folder, what is the relative path to "echo" ?



LET'S MAKE IT A BIT OF A CHALLENGE

- If you're in the "john" folder, what is the relative path to "echo" ?



`../../bin/echo`

ONCE YOU KNOW HOW TO GET TO YOUR FILES, HOW DO YOU WORK WITH THEM?

- `cat` – copy the contents to the screen
- `more (less)` – display file contents in a user-friendly manner
- `head` - copy the first lines of a file to the screen
- `tail` – copy the last lines of a file to the screen
- `wc` – count the number of lines, words, characters in a file
- `grep` – globally search a regular expression and print

PIPING

- Usually, the shell takes an input from the terminal and prints the output back on the terminal
- There can be instances where we need to fetch information from one process and feed it into another process. This is called piping.
- For eg: Read some portions of a file and send the filtered text to another file.

USEFUL COMMANDS IN PIPING

- `wc` – count the number of lines, words, characters in a file
- `grep` – output lines matching a pattern
- `sort` – sort lines of text
- `uniq` – filter out repeated lines in a file
- `cut` – cut out selected portions of each line of a file
- `tee` – sends the data to both a file and stdout
  - receives information from stdin and sends that information to a file, as well as to stdout.

AUTO COMPLETION HACKS

- From command line,
  - Type part of name of a command or filename, press <tab> and it will auto complete the name for you
  - Arrow keys (up and down) to go back through history of commands you typed in.

VIM

- `vim` filename to edit a file, Vim starts out in command mode.
- To enter the insert mode, type `i` (for "insert")
  - To get out of insert mode, hit the Escape key.
  - Once you press Escape, you're in command mode again.
  - press ":" and Vim will switch to last-line mode. Enter a command like
    - `:w` to write the file, or
    - `:q` to exit the editor, or
    - `:q!` to quit without saving etc.

HELPFUL TIPS WHEN WORKING WITH VIM

h moves the cursor one character to the left
j moves the cursor down one line
k moves the cursor up one line
l moves the cursor one character to the right
0 moves the cursor to the beginning of the line
$ moves the cursor to the end of the line
w moves forward one word
b moves backward one word
G move to the end of the file
gg moves to the beginning of the file
dd deletes a record
x deletes a character

LET'S PRACTICE

- pwd
- ls
- ls –l
- cd (home)
- cd (down)
- mkdir
- rmdir
- rm
- touch
- cp (from) (to)
- mv (from) (to)
- echo "this" >> target
- cat
- vim

REGEX

- RegEx: A regular expression is a special text string for describing a search pattern.
- In Windows, a wildcard notation such as *.txt finds all text files in a directory.
- The regex equivalent is ^.*\.txt$.
- Why RegEx?

METACHARACTERS

Metacharacters are the building blocks of regular expressions.

Characters in RegEx are understood to be either a metacharacter with a special meaning or a regular character with a literal meaning.

CARDINALITY

Metacharacter	Usage	Example	Explanation	Matching strings
*	Zero or more of the preceding character	t*	Any word that has 0 or more occurrences of the letter t.	tttttttt, aaaaaa, tattaabbb
?	Zero or one of the preceding character(s)	t?	Any word that has 0 or 1 occurrence of the letter t.	tttttttt, aaaaaa, tattaabbb
+	One or more of the preceding character(s)	t+	Any word that has 1 or more occurrences of the letter t.	tttttttt, aaaaaat, tattaabbb
{m,n}	The preceding character appears **m** to **n** times	A{1,5}	Any string that has 1-5 occurrences of A.	Apple, AAAAAAAAAA, bAll

RANGE OF VALUES

Metacharacter	Usage	Example	Explanation	Matching strings
[]	Any enclosed character	[A-Z], [0-9], [ABC]	Any value in the range specified. [A-Z] denotes any capitalized letter. [0-9] denotes any digit. [ABC] denotes any one of the 3 letters	

* Regular expressions are case-sensitive

ANCHORS

Metacharacter	Usage	Example	Explanation	Matching strings
^	Anchor - The beginning of a string	^The	Any string that starts with The.	**The**re, **The**, **The**ory, **The**y're
$	Anchor – The end of the string	coffee$	a string that ends in with coffee.	I am going to get myself a cup of **coffee**
\b or \w	Anchor – The beginning/end of a word	\bThis\b	Any string that has the exact word This.	**This** is a book.

SEQUENCE OF CHARACTERS

Metacharacter	Usage	Example	Explanation	Matching strings
()	The matching sequence of characters	(ab), (wh)	Any string that has the specified sequence of characters will match	about, what, where, why

* Regular expressions are case-sensitive

WILDCARD CHARACTER

Metacharacter	Usage	Example	Explanation	Matching strings
.	Any one character	a.t	A word that has a 3-char substring a(any 1 character)t. Some would be	act, pants, react

Metacharacter	Usage	Example	Explanation	Matching strings
.	Any one character	a.t	A word that has a 3-char substring a(any 1 character)t. Some would be	act, pants, react
[]	Any enclosed character	[A-Z], [0-9], [ABC]	Any value in the range specified. [A-Z] denotes any capitalized letter. [0-9] denotes any digit. [ABC] denotes any one of the 3 letters	
*	Zero or more of the preceding character	t*	Any word that has 0 or more occurrences of the letter t.	ttttttt, aaaaaa, tattaabbb
?	Zero or one of the preceding character(s)	t?	Any word that has 0 or 1 occurrence of the letter t.	ttttttt, aaaaaa, tattaabbb
+	One or more of the preceding character(s)	t+	Any word that has 1 or more occurrences of the letter t.	ttttttt, aaaaat, tattaabbb
^	Anchor - The beginning of a string	^The	Any string that starts with The.	There, The, Theory, They're
$	Anchor – The end of the string	coffee$	a string that ends in with coffee.	I am going to get myself a cup of coffee
\	Escape character	[0-9\-]	Any string that has either a digit or a – . Since – is a metacharacter, we will use the \ so the – can be treated as itself.	
|	Boolean OR	[A|a]	Any string that has a capitalized or lower-case a.	Apple, tablet, A, an
{m,n}	The preceding character appears **m** to **n** times	A{1,5}	Any string that has 1-5 occurrences of A.	Apple, AAAAAAAAAA, bAll
\b	Anchor – The beginning/end of a word	\bThis\b	Any string that has the exact word This.	This is a book.

GREP WITH REGEX

- Literal matches – eg. `egrep -ivn ”unix” textfile.txt`
- Anchor matches – eg. `egrep "^unix” textfile.txt`
- Matching Any Character – eg. `egrep "..eat” textfile.txt`
- Bracket Expressions – eg. `egrep ”^[A-Z]" textfile.txt`
- Repeat Pattern Zero or More Times – eg. `egrep "[A-Za-z ]*” textfile.txt`

RESOURCES

Further Reading:

- https://regex101.com/
- https://regexone.com
- http://www.zytrax.com/tech/web/regex.htm
- https://docs.python.org/2/howto/regex.html
- https://regexr.com

QUESTIONS?