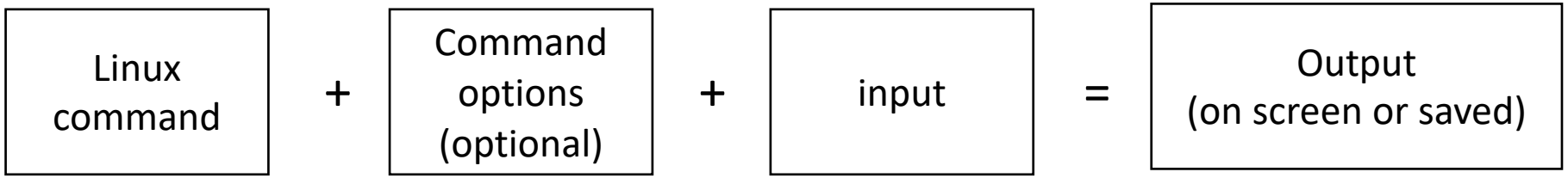
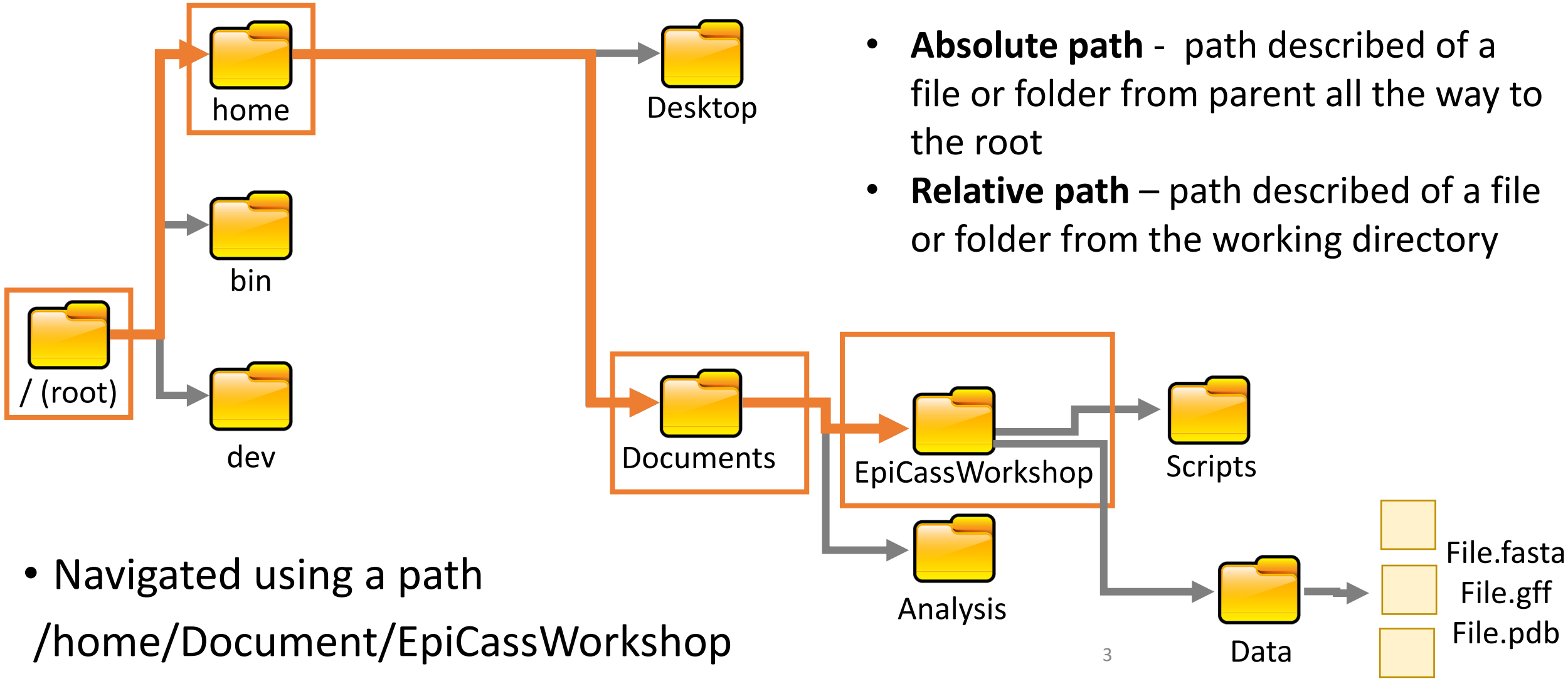
Introduction to Linux – Basic commands

Linux commands structure



Files in Linux - system hierarchy



- **Absolute path** - path described of a file or folder from parent all the way to the root
- **Relative path** – path described of a file or folder from the working directory

- Navigated using a path

/home/Document/EpiCassWorkshop

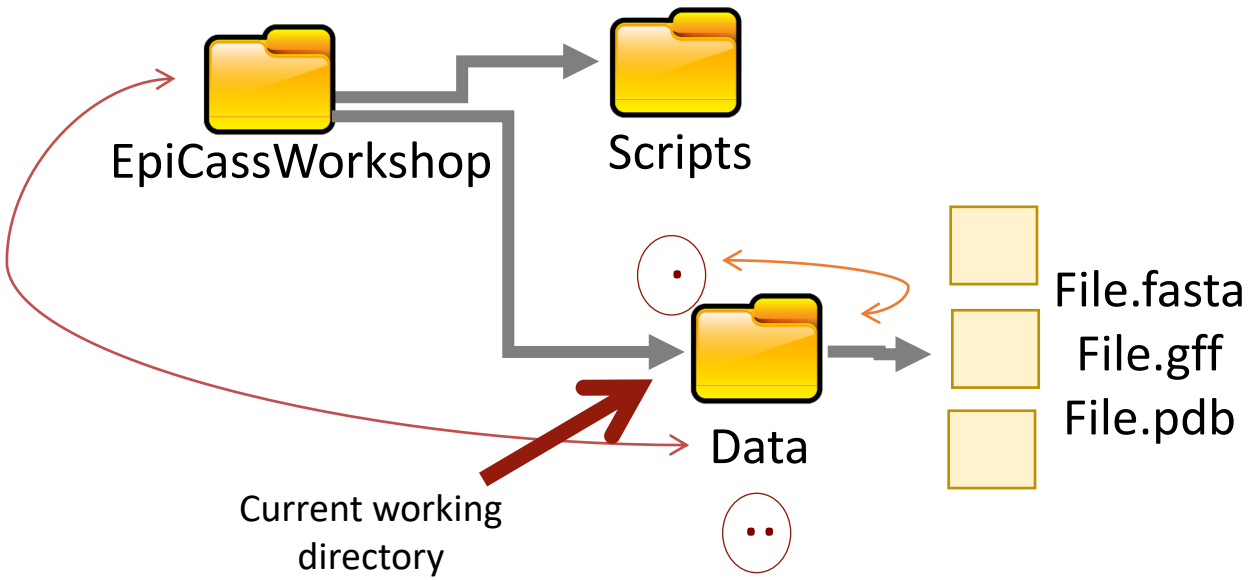
Working directories

- The first instance you log into a Linux server, the working directory is your home directory by default
- **~** or **cd** commands refers to the home directory
- The tilde **~** character can be used to specify paths starting at your home directory
- The command **pwd** gives the absolute path

Parent and working directories

./ (dot): the current working directory

../ (dot-dot): the parent directory



Creating directories and navigating through the file structure

Key commands for handling directories

- **mkdir** – creates new directory
- **rm / rmdir (empty dir) / rm -r** – removes a file or directory
- **pwd** – displays the absolute path of current working directory
- **cd** – change directory
- **ls** – lists the content of the current working directory

ls [option] command

- Lists the contect of the current working directory
- Some useful options:
    - **-l:** shows sizes, modified date and time, file or folder name and owner of file and permissions
    - **-a:** List all files including hidden file starting with ‘.’
    - **-lh:** shows sizes in easier readable format
    - **-R:** recursively lists sub-directories
    - **-lS:** sorting by file sizes
- Note: there are more options for ls command. Check out the manual – man ls

File/directory naming – what's important

- No two files in the same directory can have the same name
- Files in different directories can have the same name
- Linux is case-sensitive
- No space when naming files: TitleCasing, Under_Score, Full.Stop .. etc
- In most cases, file extensions are optional

Basic manipulating file commands

Displaying content of a file or parts of it

- **cat**: view the content of a short file. Not recommended for long files

  cat <filename>
- **more**: view the content of a long file and navigate through it. Type **q** to exit

  more <filename>
- **less**: view the content of a long file, by portions. Type **q** to exit

  less <filename>
- **head**: view the first lines of a long file – by default first 10 lines, use **–n** to change

  head <filename>
- **tail**: view the last lines of a long file – by default last 10 lines , use **–n** to change

  tail <filename>

Copy, move and remove

- **cp:** copy files and directories

  cp <pathfrom> <path to>

- **mv:** move or rename files and directories

  mv <pathfrom> <path to>

- **rm:** remove files and directories

  rm pathname

Warning: Linux does not have an undelete command! You can inflict terrific damage on your system with rm if you are not careful, particularly with wildcards. Try list command before using rm

File conents: wc command

- **wc** prints newline, word, and byte counts for each file

  wc <options> <filename>

Some useful options:

- **-c**: print the byte counts
- **-m:** print the character counts
- **-l:** print the newline counts (mostly used)
- For more info about the different commands, remember to use **man commandname**

Extracting content from a file: cut & grep command

- **grep**: to search for the occurrence of a specific pattern (regular expression using the wildcards…) in a file

  grep <pattern> <filename>

grep ***sequence*** *<filename>*

grep ***–v sequence*** *<filename>*

- **cut**: is used to extract specific fields from a file

  cut <options> <filename>

- Important options are

  -d (field delimiter)

  -f (field specifier)

Text editors

- **nano:** a simple and easy-to-use text editor
- Is installed by default in many other Linux distributions
- **gedit** is also very easy to use
- vim, emacs, geany, visual studio code: excellent programs but some do require some learning

Operations on files, using wildcards and combining commands

Basic operation on files: Sorting your data

- **sort** outputs a sorted order of the file content based on a specified sort key (default: takes entire input)

  sort <options> <filename>

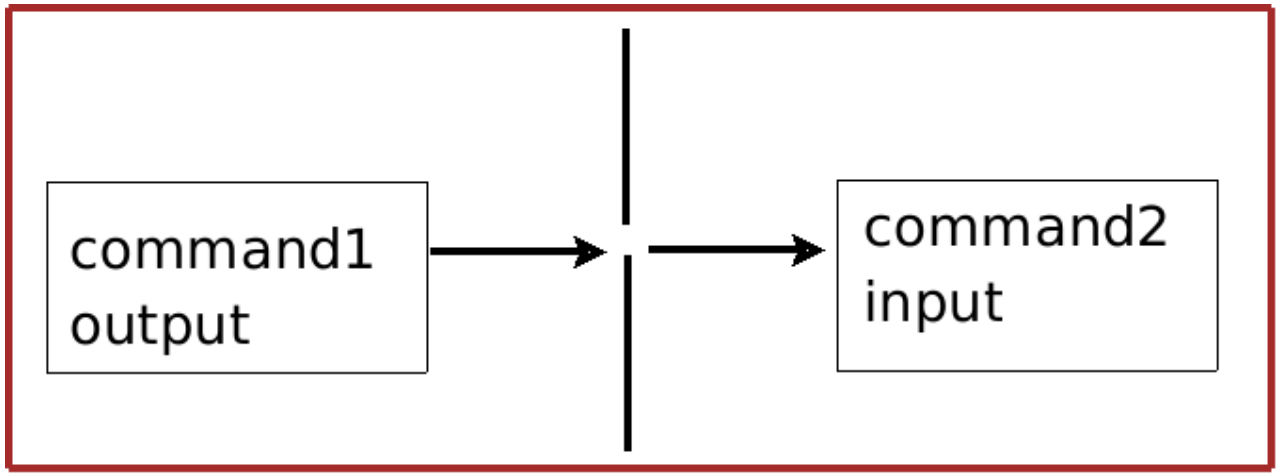
- Sort alphabetically (default option): **sort <filename>**
- Sort numerically: **sort -n <filename>**
- Sort on a specific column (n°4): **sort –k 4 <filename>**
- **uniq** outputs a file with no duplicated lines **uniq <options> <filename**
- Useful while using uniq command: option **–c** is to output each line with its number of repeats

Redirecting and appending outputs

- By **default**, the **standard output** of any command will appear to the **terminal screen**
- Redirection of the output result to a file is possible: filename.**in >** filename.**out**
- This is particularly useful for big files
- If the file does not exist, it will be automatically created and the result redirected to it.
- After redirecting an output into a file, it is possible to append new output to the same file: filename.**in >>** filename.**out**

Pipes : combining commands

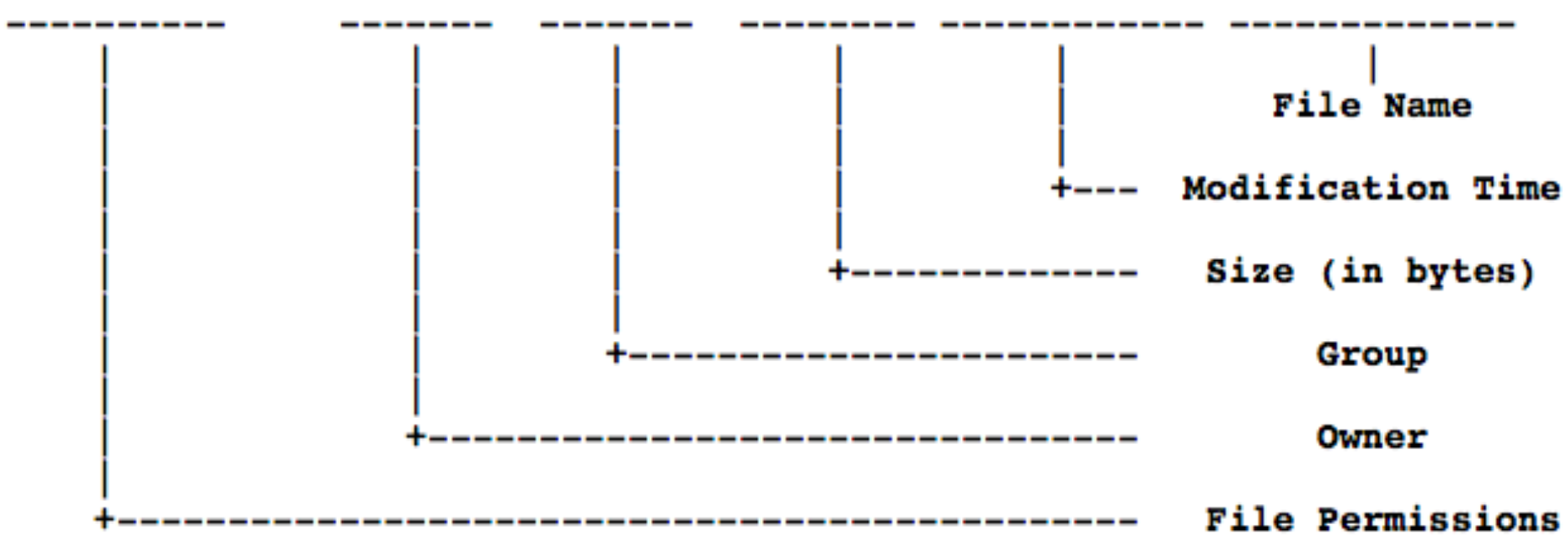
- As seen previously mentioned, outputs are printed in the screen or redirected to a file
- However, the output result of a command can also be redirected to another command
- This is particularly useful when several operations are needed for a file, with no need to store the intermediate outputs
- Combining several commands is done thanks to the use of a “|” character



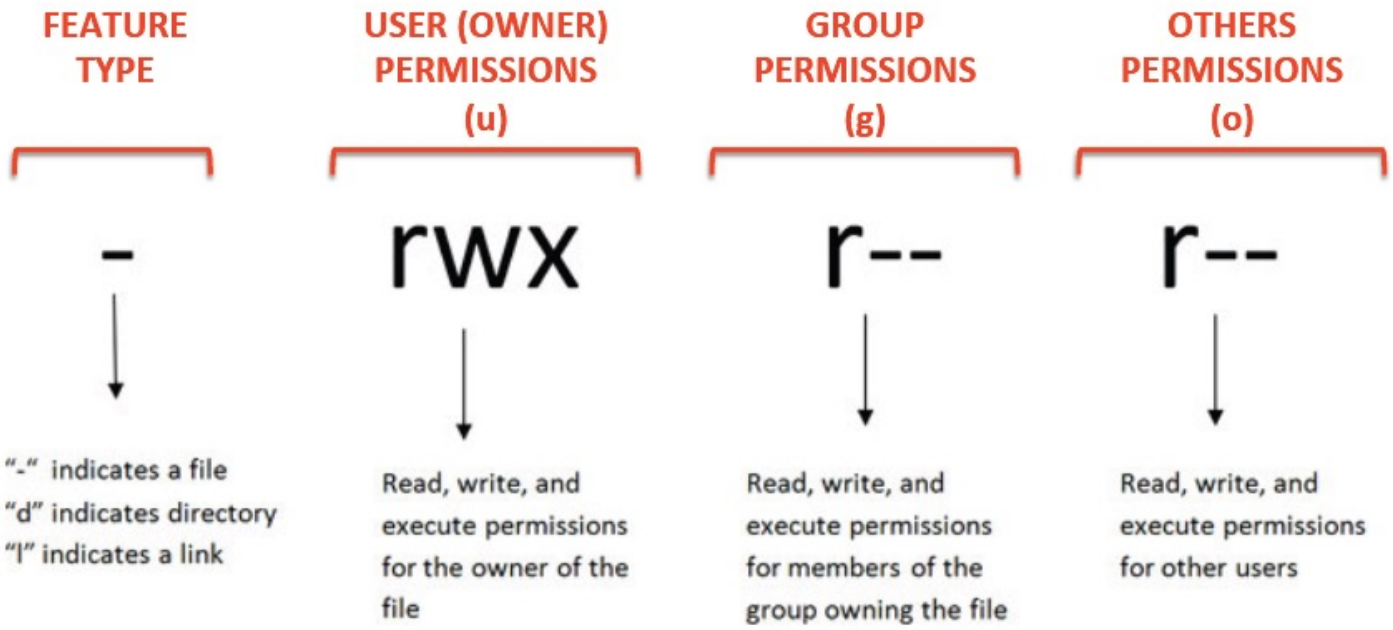
Files permissions

Remember ls –l command?

```
drwxr-xr-x 22 michaellandi      704 Dec 15 07:01 DNA-Methylation-workshop
drwxr-xr-x 11 michaellandi      352 Apr  5 17:39 EpiCass-Workshop
```



Files permissions: read, write and excute



- **r** indicates read permission
- **w** indicates write permission
- **x** indicates execution permission

Source: www.pluralsight.com

Chmod command

- Used to change the permissions of a file or a directory
- **chmod <options permissions> filename**
- Only the owner of the file can use chmod to change the permissions
- Permissions define permissions for the owner, the group of users and anyone else (others)
- There are two ways to specify the permissions:
    I. Using symbols (alphanumeric characters)
    II. Octal notation (digits - 0 to 7)

Using chmod command

Using symbols:

- user/owner **(u)**, group **(g)**, and others **(o)**
- To set a file, so it is public for reading, writing, and executing, the command is: **chmod u=rwx,g=rwx,o=rwx filename**

Using octal format:

r has the value of **4**

w has the value of **2**

x has the value of **1**

no permission has the value of **0**

e.g. what permission will 750 and 644 give?

Wildcards

- A group of special characters are called wildcards allow selecting filenames based on pattern of characters. Example of some of wildcards

***** - matches any characters

A*.fasta - all filenames that begin with A and end with .fasta

? - matches any single character

????.vcf - any filenames that contain exactly 4 characters and end with .vcf

[characters] - matches any character that is a member of the set characters

[abc]* - any filename that begins with "a" or "b" or "c" followed by any other characters

While doing your practical session, few tips

- Use tab completion - it will save you time!
- Build commands slowly!
- **man <commandname>** often gives you help
- Always have a quick look at files with less or head to double check their format
- Regular expressions are wierd, build them up slowly bit by bit
- If you did something smart but can’t remember what it was, try typing **history**
- Google should be your friend! (prioritise stackoverflow.com – results NOT questions)


BREAK
TIME !!