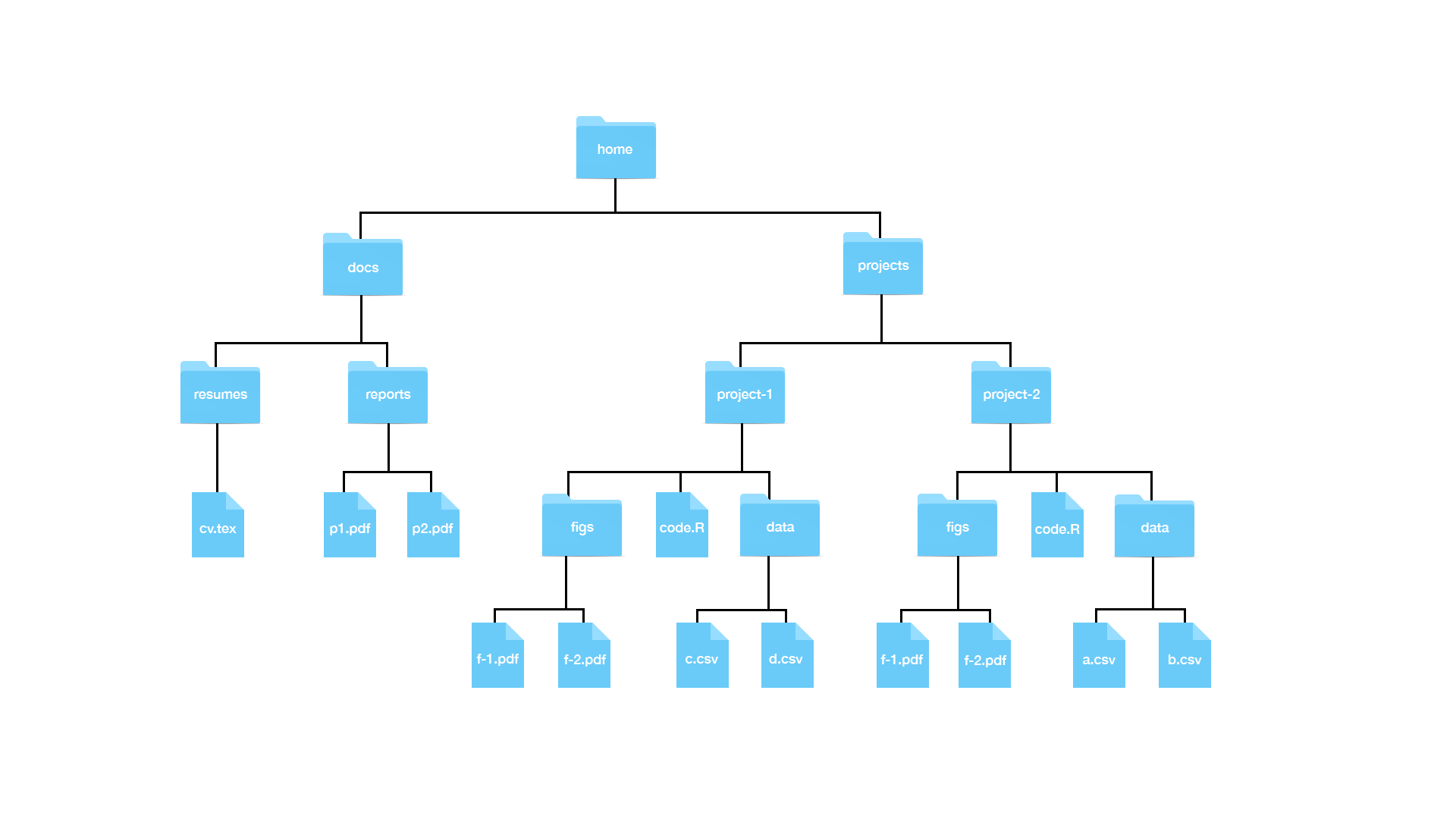
The Unix section discusses the basics of managing your filesystem from the terminal with Unix commands such as mv and rm.

There is a two-part graded comprehension check at the end of the section. Part 2 is only available to Verified learners.

If you get stuck, we encourage you to search the discussion boards for the answer to your issue or ask us for help!



**Useful Unix Commands**

| **Command** | **Description** | **Examples** |
| --- | --- | --- |
| ls | List directory content |  |
| mkdir *dir* | Make a directory | mkdir projects –make the directory projects  mkdir docs –make the directory docs  mkdir junk –make the directory junk |
| rmdir *dir* | Remove a directory (directory must be empty; otherwise use “rm”) | rmdir junk –remove the directory junk |
| cd *dir* | Change directory | cd /projects – move to the projects directory (an absolute path)  cd projects – move to the projects directory, assuming we are already in the home directory (a relative path) |
| cd .. | Go up one directory to the parent directory | cd ../.. – move up two parent directories from our current directory |
| cd ~ | Go to the home directory |  |
| cd - | Go to whatever directory you just left |  |
| pwd | Print the present working directory |  |
| Tab key | Autocomplete | cd d + tab – autocompletes to docs if it is the only directory that begins with d; or list the different options. |
| mv *file1 file2* | Move or rename files  *Warning –this is permanent, and you will not get a warning message if you are overwriting files.* | mv ~/docs/resumes/cv.tex ~/docs/reports/ –move the cv.tex file from the resume folder to the reports folder  mv cv.tex resume.tex – rename cv.tex to resume.tex  mv ~/docs/resumes ~ /docs/reports/ - move the resume folder into the reports folder |
| cp *file1 file2* | Copy file1 to file2 | cp ~ ~/docs/reports/ – make a copy of the cv.tex file from the resume folder in the reports folder |
| rm *file* | Delete file  *Warning – this is permanent! You cannot retrieve files from the recycling bin!* | rm ~/docs/resumes/cv.tex – delete the file cv.tex |
| less *file* | View file | less ~/docs/resumes/cv.tex –open cv.tex in the less text viewer |
| rm -r *dir* | Remove recursively all folders in directory *dir* and the directory itself. |  |
| ls -a | List all directory content, including hidden files |  |
| ls -l | List all directory content in long form (including permissions, size and date) |  |
| ls -t | List all directory content in chronological order | ls -lart – show more information for all files in reverse chronological order for your current directory |
| man *command* | Show the manual for the command. Note – this does not work for GitBash | man ls – show the manual instructions for the command ls. |
| help | Show the manual for the command in GitBash | ls --help – show help instructions for the command ls |
| *command1*| *command2* | Pipe the results of command 1 to command 2 | man ls | less – show the help instructions for the command ls in the less viewer |
| \* (wildcard) |  | ls \*.html –list all the files ending in html in your current directory  rm \*.html – remove all files ending in html in your current directory |
| ? (any character) |  | rm file.???.html – remove all files whose names follow the pattern; eg file-001.html, file-002.html etc.  rm file.???.\* – remove all files whose names follow the pattern regardless of their extension; eg file-001.html, file-002.csv, file-any.R, etc. |
| $*var* | >$ identifies a variable | echo $HOME – print your home directory  echo $SHELL – print your shell name |
| export *val=value* | Change the value of the variable *val*(Bash shell specific) |  |
| open *file*(mac)*file*(windows) | Opens a file or program | open Report.Rmd – open Report.Rmd in RStudio |

#### Absolute path vs. relative path

A full path specifies the location of a file from the root directory. It is independent of your present directory, and must begin with either a “/” or a “~”. In this example, the full path to our “project-1” file is:

/home/projects/project-1

A relative path is the path relative to your present working directory. If our present working directory is the “projects” folder, then the relative path to our “project-1” file is simply:

project-1

#### Path shortcuts

One period “.” is your current working directory

Two periods “..” is the parent directory (up one from your present working directory)

A tilde   “~” is your home directory.

#### More path examples

1.     Your current working directory is ~/projects and you want to move to the figs directory in the project-1 folder

* Solution 2: cd ~/projects/project-1/figs (absolute)
* Solution 2:  cd project-1/figs (relative)

2.     Your current working directory is ~/projects and you want to move to the reports folder in the docs directory

* Solution 1: cd ~/dos/reports (absolute)
* Solution 2: cd ../docs/reports (relative)

3.     Your current working directory is ~/projects/project-1/figs and you want to move to the project-2 folder in the projects directory.

* Solution 1: cd ~/projects/project-2 (absolute)
* Solution 2: cd ../../project-2 (relative)

### Key points

* The terminal helps to organize files in the system.
* On Mac, use utilities > terminal.
* On windows, use Git bash program.
* Use the keyboard to navigate the command line.

### Code:

echo "hello world"

The FileSystem

### Key points:

* We refer to all the files, folders, and programs (executables) on your computer as the **filesystem**.
* Your filesystem is organized as a series of nested folders each containing files, folders, and executables. (see the visualization in the textbook)
* In Unix, folders are referred to as directories and directories that are inside other directories are often referred to as subdirectories.
* The home directory is where all your stuff is kept.  There is a hierarchical nature to the file system.
* **Note for Windows Users:** The typical R installation will make your Documents directory your home directory in R. This will likely be different from your home directory in Git Bash. Generally, when we discuss home directories, we refer to the Unix home directory which for Windows, in this book, is the Git Bash Unix directory.

Working Directory

### Key points

* The working directory is the current location.
* Each terminal window has a working directory associated with it.
* The "pwd" command will display your working directory. The "/" symbol separates directories, while the first "/" at the beginning of the path stands for the root directory. When a path starts with "/", it is a "full path", which finds the current directory from the root directory. "Relative path" will be introduced soon.
* "~" means the home directory.

Unix Commands

### Key points

* navigate the file system with commands introduced in this video.
* Auto-complete paths, commands and file names with the "Tab" key.

### Code

ls #list dir content

mkdir folder\_name #create directory called "folder\_name"

rmdir folder\_name  #remove an empty directory as long as it is empty

rm -r folder\_name  #remove dir that is not empty, "r" stands for recursive

cd: change dir

../ # two dots represents parent dir

. # single dot represents current workingdir

cd ~/projects # concatenate with forward slashes

cd ../.. # change to two parent layer beyond

cd -  # whatever dir you were before

cd  # return to the home dir

Adding and Removing Files

### Key points

* The mv command moves files.
* **[warning]** mv will not ask you to confirm the move, and it could potentially overwrite a file.
* The rm command removes files.
* **[warning]** rm is permanent, which is different than throwing a folder in the trash.

### Code

mv path-to-file path-to-destination-directory

rm filename-1 filename-2 filename-3

Looking at a file

### Key points

* less allows you to quickly look at the content of a file
* Use q to exit the less page
* use the arrows to navigate in the less page

### Code

less cv.tex

Preparing for a science data project

### Key points

* Ideally, files (code, data, output) should be structured and self-contained
* In a project, we prefer using relative paths (path relative to the default working directory) instead of the full path so that code can run smoothly on other individual's computers.
* It is good practice to write a README.txt file to introduce the file structure to facilitate collaboration and for your future reference.

### Code

########### In terminal ###########

cd ~ # move to home directory

mkdir projects # make a new directory called projects

cd projects # move to ~/projects directory

mkdir murders # make new directory called murders inside of projects

cd murders # move to ~/projects/murders/

mkdir data rda # make two new directories, one is data the other is rda folder

ls # to check if we indeed have one data folder and one rda folder

pwd # check the current working directory

mkdir figs # make a directory called figs to store figures

########## In RStudio ########

# pick existing directory as new project

getwd() # to confirm current working directory

save() # save into .rda file, .RData is also fine but less preferred

ggsave("figs/barplot.png") # save a plot generated by ggplot2 to a dir called "figs"