intro-shell-cmd-line

Clara Jégousse

3/1/23

Table of contents

Preface

Why learn how to use the shell

There are many reasons to learn about the shell.

- For most bioinformatics tools, you have to use the shell. There is no graphical interface. If you want to work in metagenomics or genomics you're going to need to use the shell.
- The shell gives you *power*. The command line gives you the power to do your work more efficiently and more quickly. When you need to do things tens to hundreds of times, knowing how to use the shell is transformative.
- To use remote computers or cloud computing, you need to use the shell.
- We're going to use it in this class, for all of the reasons above.

1 Introduction

Questions

- How do you **access** the shell?
- How do you **use** the shell?

1.1 How to access the shell

You access the shell using the “terminal”. The terminal is a window into which we will type commands in. The terminal is already available on Mac and Linux. For windows you must download a separate program.

1.2 Starting with the shell

We will spend most of our time learning about the basics of the shell by manipulating some experimental data from a hearing test.

1.2.1 Make a new directory

First, we’re going to create a new **directory** to work in using the command `mkdir` which stands for “**m**a**k**e **dir**ectory”.

Note

The word “directory” a location for storing files on a computer and it is synonym to “folder”.

To create the directory, we must type the following command in the terminal and hit return to execute the command.

```
$ mkdir training
```

1.2.2 Listings

Now let's see if our new directory was created using the command `ls` that stands for "list" or "listing".

```
$ ls
```

You should see your new directory called "training" in the list displayed in the terminal.

1.2.3 Navigating the directories

Now we're going to go into that directory using the command `cd` which stands for "**c**hange **d**irectory".

```
$ cd training
```

Now let's look at see what's in here. There should be nothing right now, because we just made it, but let's check using the command `ls` again.

Now we are going to create a text file using the command `touch`

```
$ touch file.txt
```

We can now check with if the file was created using the `ls` command again.

Now we're going to download the data for this tutorial. For this you'll need internet access, because you're going to get it off the web. Enter the command:

```
$ git clone https://github.com/clarajegousse/training-data
```

This command will grab all of the data needed for this workshop from the internet.

1.3 Listings

Now let's check to see that we got the data.

```
$ ls
```

In there, all mixed up together are files and directories/folders. If we want to know which is which, we can type: `ls -F`

```
$ ls -F
```

Anything with a `/` after it is a directory. Things with a `*` after them are programs. If there's after the name, then it's a file.

You can also use the command `ls -l` to see whether items in a directory are files or directories. `ls -l` gives a lot more information too, such as the size of the file.

```
$ ls -l
```

So, we can see that we have several files, directories and a program. Great!

1.4 Manuals

To know in more detail how to use the command `ls`, we can consult the manual using the following:

```
$ man ls
```

To exit the manual, hit the `q` key to "**q**uit".

Every single command has its own manual. Earlier, we used the command `cd` and we can also consult the manual for `cd`:

```
$ man cd
```

2 The directory file structure

As you've already just seen, you can move around in different directories or folders at the command line. Why would you want to do this, rather than just navigating around the normal way.

When you're working with programs, you're working with your data and it's key to be able to have that data in the right place and make sure the program has access to the data. Many of the problems people run in to with command line bioinformatics programs is not having the data in the place the program expects it to be.

Now let's see where we are by using the command `pwd` which stands for "print working directory":

```
$ pwd
```

The output should look like `/Users/[username]/training` (`/home/[username]/training`) but with you username. This is called a hierarchical file system structure, like an upside down tree with root (`/`) at the base.

Tip

To display your username, you can use the command `whoami`:

```
$ whoami
```

When you are working at your computer (or log in to a remote computer), you are on one of the branches of that tree, your home directory (`/home/username`).

2.1 Moving around the file system

Let's practice moving around a bit.

We're going to work in that data directory we just downloaded.

This is called a hierarchical file system structure, like an upside down tree with root (`/`) at the base.

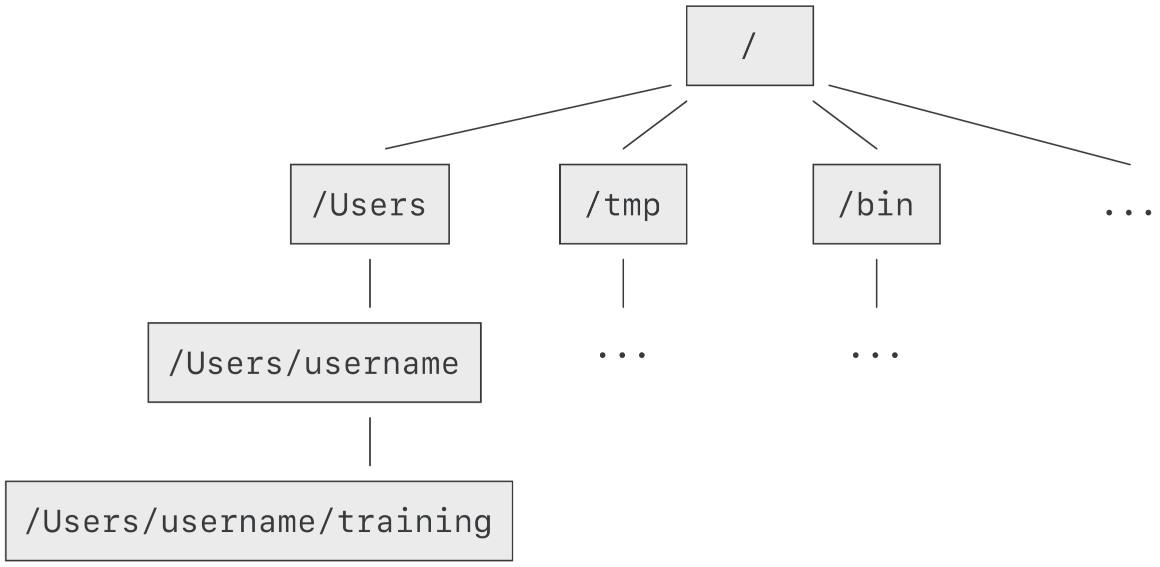

Figure 2.1: Hierarchical file systen structure

When you are working at your computer (or log in to a remote computer), you are on one of the branches of that tree, your home directory (`/home/username`)

Now let's go do that same navigation at the command line.

Type cd This put's you in your home directory. This folder here.

Now using `cd` and `ls`, go in to the 'data' directory and list its contents.

Let's also check to see where we are. Sometimes when we're wandering around in the file system, it's easy to lose track of where we are and get lost.

If you want to know what directory you're currently in, type pwd This stands for 'print working directory'. The directory you're currently working in.

What if we want to move back up and out of the 'data' directory? Can we just type 'edamame'? Try it and see what happens.

To go 'back up a level' we need to use ..

Type cd ..

Now do `ls` and `pwd`. See now that we went back up in to the 'edamame' directory. .. just means go back up a level.

3 03-tips-tricks.qmd

4 04-files

References