The File System

Max Carter-Brown
Anglia Ruskin University, Wellcome Sanger Institute
max.carter-brown@aru.ac.uk

Contents

1. Outcomes

By the end of this document, you should be able to:

1. Create your own file systems using the in built commands, `mkdir`, `touch`
2. Modify your file system by renaming files, moving them, and copying them using the commands: `cp`, `mv`
3. Delete parts of your file system using `rm`

2. Introduction

The file system is the layout of where all your files live in your computer. I have a made up file system (Figure 1) which includes a root directory and a home directory. The root is denoted in the filesystem as `/`, and usually has no files or directories above it. The home directory is usually denoted as `~` and is further in to your file system and is where all of your familiar directories and files should live (Documents, Pictures, Downloads, etc).

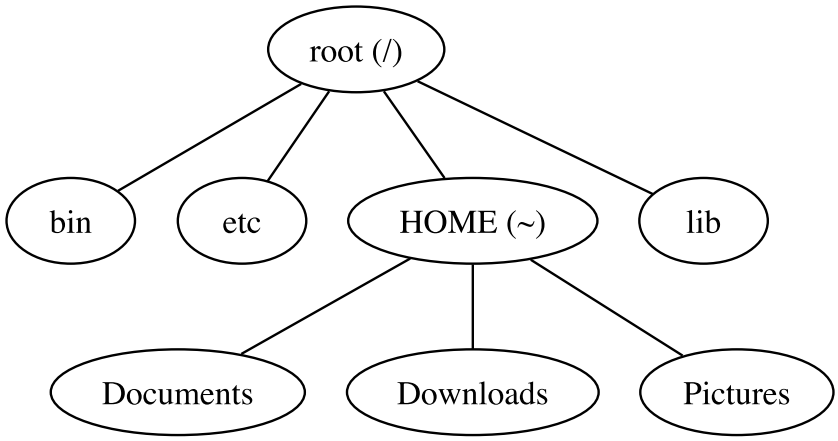

Figure 1: A made up file system with a root (`/`) and a home (`~`).

We were able to explore this file system using the `cd` and `ls` commands before. This is useful for exploring. But we have a powerful toolkit for creating, moving, renaming, and destroying files and directories too. The commands we will look look at are `mkdir` for making directories, `touch` for creating files, `cp` for copying files or directories, `mv` for moving files or directories, and `rm` for removing files or directories.

3. Making the file system

First we will make a directory. For this, it does not matter where you are in the file system, we will clean it up later.

Shell 1: Making directories using `mkdir` and going in to the directory you just made using `cd`.

```
# make the new directory
mkdir new_directory
# prove you made it
cd new_directory
# you should now be in your new directory
```

Make files using `touch`. `touch` is not the only way to create files, but it's a convenient one. There are other ways to make files which we will explore later.

Shell 2: Creating a file using `touch` and checking it's there using `ls`.

```
# You can create any file. I advise creating text based files
# (i.e. not binary files), so they can be manipulated further later.
touch file.txt
# check you made the file
ls -l
```

This file is empty, so it is not very helpful at the moment. There are a couple of things we can illustrate using this example though. What if we wanted to make a copy of this file? We use the command `cp` to copy files or directories.

Shell 3: Copying a file using `cp` and checking it's there using `ls`. We then copy another file into a subdirectory `copies`.

```
# copying files is called like this:
# cp <file> <copied file location>
cp file.txt file_copy.txt
# check you made the file
ls -l
# make another directory
mkdir copies
# and copy a file into it
cp file.txt ./copies/file_copy.txt
```

Lastly, we can move files around. `mv` is also used frequently for renaming files. We should now be in a directory called `new_directory` and there should be two files inside, `file.txt` and `file_copy.txt`, with a directory called `copies` with `file_copy.txt` inside. We made a whole file system in a couple of lines of code!

We can rename files easily.

Shell 4: Renaming a file from `file.txt` to `wow.txt`

```
mv file.txt wow.txt
```

And move them around just as easily.

Shell 5: `mv` can move and rename files at the same time.

```
# Moving a file from the `copies` directory
# into the current one
mv copies/file_copy.txt file.txt
```

We should have ended up with `wow.txt` and `file.txt` in the `new_directory` directory. There should be nothing in the `copies` directory.

4. Destroying the file system

Destroy files with `rm`. Destroy directories with `rm -r` (`-r` is recursive). **Use `rm` with extreme caution**. Once you delete a file or directory **it is gone forever**. There is no bin or trash with this command.

Shell 6: Destroying our little file system.

```
# delete everything we made
# delete the copies directory
rm -r copies

# delete the files in this current directory
# rm can take multiple arguments
rm file.txt wow.txt

# go up
cd ..
# and remove our directory
rm -r new_directory
```

And with that, we removed everything!