Learning Objectives

Learners will be able to…
1. Predict the location of installed files and directories
1. Know which basic utilities they can always find on a Linux system
1. Understand what a symbolic link is
1. Edit a configuration file

Introduction to FHS

The Filesystem Hierarchy Standard ensures that software packages running on a Linux system will know where to find essential files and directories.

The current working directory for the terminal on the left is your workspace.

To view the directories off the root type in:

```
cd /
ls
```

The `cd /` switches you to the root directory, and the `ls` command lists the contents of that directory.

Name	Function
`/bin`	Binaries or executables that are essential for functionality
`/boot`	Files needed to boot the system such as the Linux kernel
`/dev`	Device files - interface with hardware drivers
`/etc`	Host-specific system configuration - editable text
`/home`	User directories live under here
`/lib`	Common libraries
`/lib64`	Common 64-bit libraries
`/media`	Mount point for removable media
`/mnt`	Mount point for mounting a filesystem temporarily
`/opt`	Optional add on software
`/proc`	Keeps track of running processes
`/root`	Home directory for root user
`/run`	Data relevant to running processes
`/sbin`	System binaries or executables that are essential for functionality
`/srv`	Data for services provided by this system

`/sys`	A symbolic link to the kernel source tree
`/tmp`	Temporary files that won't be persistent between reboots
`/var`	Variable files - things that will change as the operating system is being run such as logs and cache files

The /bin directory

The `/bin` directory contains commands you are familiar with from the last module and more.

The following commands must be available in the `/bin` directory:
Command	Description
———	———–
cat	Concatenate files to standard output
chgrp	Change file group ownership
chmod	Change file access permissions
chown	Change file owner and group
cp	Copy files and directories
date	Print or set the system date and time
dd	Convert and copy a file
df	Report filesystem disk space usage
dmesg	Print or control the kernel message buffer
echo	Display a line of text
false	Do nothing, unsuccessfully
hostname	Show or set the system's host name
kill	Send signals to processes
ln	Make links between files
login	Begin a session on the system
ls	List directory contents
mkdir	Make directories
mknod	Make block or character special files
more	Page through text
mount	Mount a filesystem
mv	Move/rename files
ps	Report process status
pwd	Print name of the current working directory
rm	Remove files or directories

info

Reminder - the `man` command

You can get more information about any of the commands above by typing `man` and then the command name at the terminal prompt.

The `more` command

The `more` command is similar to `cat` in that it will list out the contents of a file but it will display only a screen full at a time.

Try it out:

```
more -d prideandprejudice.txt
```

The `-d` parameter instructs the `more` command to put a prompt at the bottom telling you to press `space` for more text or `q` to quit.

The /boot and /dev directories

The `/boot` directory contains everything required for the boot process.

The exceptions are configuration files not needed at boot time and the map installer. The operating system kernel must be located in `/` or `/boot`.

Take a look at the boot directory:

BOOT DOESN'T HAVE ANYTHING IN IT RIGHT NOW

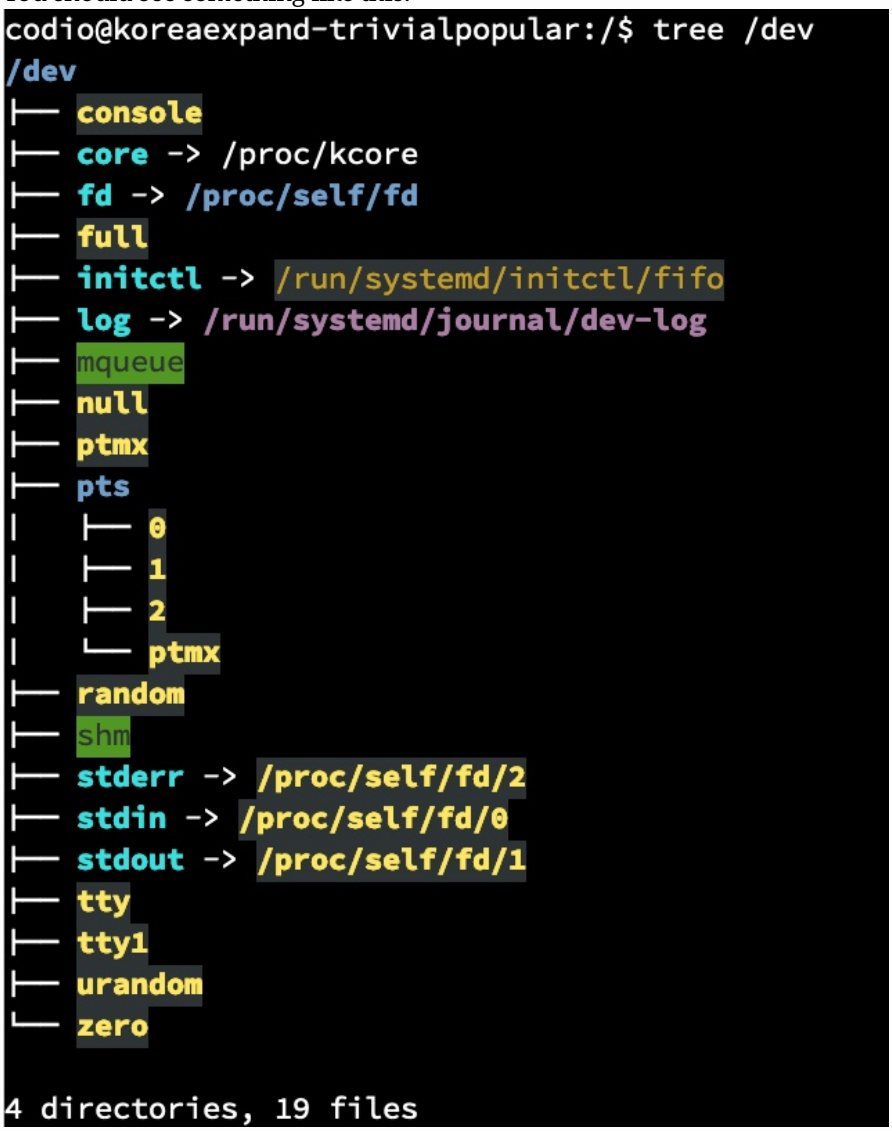
```
ls /boot
```

The /dev directory contains special or device files.

You will recognize some of the names such as console and stdout.

```
tree /dev
```

You should see something like this:



Notice that some files have a directory path listed to the right of them, these are called symbolic links (symlink). A symlink is a type of file in Linux that points to a different file or folder. Symlinks allow multiple access points to a file without needing multiple copies.

The /etc directory

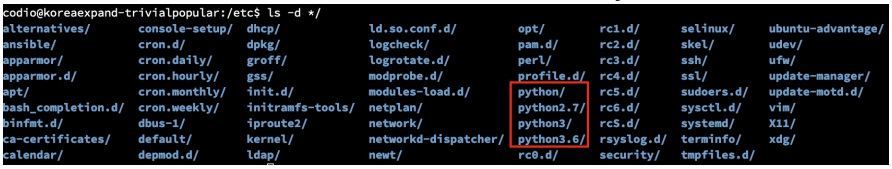
The `/etc` directory contains all system related configuration files.

Configuration files are editable text files, executable files should not be placed in this directory. The configuration files should be placed in subdirectories of the `/etc` folder grouped by the application they serve.

Let's first look at a list of directories that live under the `/etc` directory:

```
cd /etc
ls -d */
```

You should see directories for different versions of Python:



Next we'll look at the configuration file for the vim editor which we covered in the previous module. It is named `vimrc` and located in the `vim` directory.

```
cd vim
more -d vimrc
```

Changing the configuration file for vim will change its behavior

We will add a line to this file to tell it to display line numbers.

First let's take a look at the `prideandprejudice.txt` file before we make the change to see that vim does not display line numbers by default.

```
vim /home/codio/workspace/prideandprejudice.txt
```

important

To exit vim type the following sequence:

1. `esc` key
2. `:wq` to save changes, `:q` if you haven't made any changes, or `:q!` if you don't want to save changes.
3. `return` or `enter` depending on your system

By default these system files are read only. You have "super user" capabilities so you can modify the permissions for the file using the `sudo` - "super user do" combined with the `chmod` command which allows you to change file permissions.

The first command below will add write permission for all users for the file `vimrc`. The second command will append the line "set number" to the file, this tells `vim` to display line numbers.

```
sudo chmod ugo+w vimrc
echo "set number" >> vimrc
```

Now let's look at the file again and see that we have changed the behavior for `vim` and now we see line numbers.

```
vim /home/codio/workspace/prideandprejudice.txt
```

The /home and /lib directories

The `/home` directory contains user directories

Your user name is `codio` and you are the only user set up in the Linux system you access from the terminal on the left.

Your terminal window defaults to opening up with your workspace as the current working directory so when you type the commands below you'll see the full name of your working directory and the things you see in the file tree.

```
pwd
ls
```

The `/lib` directory contains essential shared libraries and kernel modules.

The `/lib` directory contains shared library images needed to boot the system and run the commands in `/bin` and `/sbin`.

```
cd /lib
ls
```

You should see a listing similar to the one in the image below. The `x86_64-linux-gnu` is circled to show that the contents of this directory are for the version of GNU/Linux running in the terminal.

A listing of the contents of the /lib directory with the x86_64-linux-gnu directory circled

The x86 refers to the type of processor and the 64 means that it is a 64-bit system (as opposed to a 32-bit).

The /media and /mnt directories

You will notice that these directories are empty. There are no devices mounted to your Linux instance in the terminal.

```
cd /media
ls -a
cd /mnt
ls -a
```

The `/media` directory is used for removable media such as USB drives and CD ROMS. This is typically used by the system. The `/mnt` directory is used for temporarily mounted file systems, mostly for user-mounted items.

All files that are accessible in a Linux system are arranged in one hierarchical tree that starts at the root `/`. The files accessible through the root can be spread out over multiple devices. This differs from what you see in a file system such as Windows where you will see a separate tree for each device. For example in Windows, c: is the internal hard drive and if you have partitioned your hard drive you would access the other partition through d:. If you insert a USB stick you might get another drive f:.

In Linux to make external devices accessible you need to attach them to your file tree. External devices are “mounted” to the Linux files system at `/media` or `/mnt`. More information about mounting file systems here: https://man7.org/linux/man-pages/man8/mount.8.html

You will learn more about mounting external devices in the **Listing and Mounting Hardware** assignment in the **Managing Devices** module.

The /opt, /proc and /root directories

The `/opt` directory is reserved for the installation of add-on application software packages. There is nothing installed in that directory in the system running in your terminal.

The `/proc` directory or more often referred to as filesystem is built every time the system starts and it contains information about currently running processes, hardware and memory management. It represents the current state of the kernel.

```
cd /proc
ls -a
```

You can learn more about processes in the Processes module.

The `/root` directory is the root user's home directory. As you will see if you try it out below, you do not have permission to access the `/root` directory.

The /run directory

The `/run` directory contains system information about the system since it was booted. This directory must be cleared at the beginning of the boot process.

Take a look at the contents of the `/run` directory.

```
cd /run
ls -a
```

You will see files with the extension `.pid`. These are **Processor identifier** files (PID). A PID file consists of a process identifier in ASCII-encoded decimal, followed by a newline character. The directories you see may also contain PID files and must also be cleared during the boot process.

Try this out:

Enter the `top` (table of processes) command. It will show you the list of running processes. Type `q` to exit the display.

```
top
```

Now you have a list of the processes and if you list out one of the `.pid` files you will see that the number matches the value in the table.

```
cat crond.pid
```

The cron daemon (crond) is a system-managed executable that runs in memory and is used to schedule tasks.

The /sbin directory

The `/sbin` directory is for system binaries. Binary files are executable programs and may also be referred to as commands. These binary files are essential for booting, restoring, recovering, and repairing the system. The `/sbin` directory must not contain any subdirectories. At the very least the `/sbin` directory must contain the `shutdown` command.

Take a look at the contents of the `/sbin` directory.

```
cd /sbin
ls -a
```

You can find more information about some of the other files you might typically see in the `/sbin` directory here.

What is the difference between `/bin` and `/sbin`?

The `/bin` directory contains binaries (commands) that are for users as well as items needed to bring the system up or repair it. The `/sbin` directory contains binaries that the system uses for booting up. These are generally not run by users, you need `sudo` privileges to be able to run them.

The /srv, /sys and /tmp directories

The `/srv` directory is used for data for services provided by the system. It is empty in the system in your terminal.

The `/sys` directory is a virtual file system where you can find information about devices, drivers, and other kernel components.

Take a look at the contents of the `/sys` directory.

```
cd /sys
ls -a
```

The `/tmp` directory may be used by applications to store temporary files, files that an application does not expect to remain after it stops running. It is recommended (but not required) that the files in the `/tmp` are deleted whenever the system is rebooted.

The /usr directory

The data in the /usr directory is read-only. It is for user-runnable programs and user-accessible data is located.

Required directories, or symbolic links to directories in `/usr`.

Directory	Description
`bin`	Most user commands
`lib`	Libraries
`local`	Local hierarchy (empty after the initial installation)
`sbin`	Non-vital system binaries
`share`	Architecture-independent

Optional directories, or symbolic links to directories in `/usr`.

Directory	Description
`games`	Games and educational binaries
`include`	Header files included by C programs
`libexec`	Binaries run by other programs
`lib` <qual>	Alternate Format Libraries
`src`	Source code

More information about the subdirectories of the /usr directory may be found in the FHS 3.0 document from the Linux Foundation.

What is the difference between `/bin` and `/usr/bin`?

The `/bin` directory contains executable commands that are required by the system and `/usr/bin` contains executable files that are not required.

info

How color is used in a Linux directory listing

Color	Meaning
White	Most files
Green	Executable
Blue	Directory
Cyan	Symbolic link file
Yellow with black background	Device
Magenta	Graphic image file
Red	Archive file
Red with black background	Broken link

Take a look at the contents of `/usr/bin`, you can use the entries in the table above to determine the types of files.

```
ls /usr/bin
```

One of the useful commands in the `/usr/bin` directory is **whereis**. Try it out:

```
whereis python3
```

Running this shows you all the locations of python3 related files.

What is the difference between `/sbin` and `/usr/sbin`?

The `/sbin` directory holds commands needed to boot the system. The `/usr/sbin` direcory contains program binaries for system administration which are not essential for the boot process.

Take a look at the contents of `/usr/sbin`, you can use the entries in the table above to determine the types of files.

```
ls /usr/sbin
```

If you are curious about any of these executables, you can type in `man` followed by the file name. As a reminder, use the letter `q` to exit the `man` command.

The `/usr/local` directory is for use by the system administrator when installing software locally.

The following directories, or symbolic links to directories, must be in `/usr/local`

Directory	Description
`bin`	Local binaries
`etc`	Host-specific system configuration for local binaries
`games`	Local game binaries
`include`	Local C header files
`lib`	Local libraries
`man`	Local online manuals
`sbin`	Local system binaries
`share`	Local architecture-independent hierarchy
`src`	Local source code

The /var directory

The `/var` (short for variable data) hierarchy contains files to which the system writes data during the course of its operation.

The following directories, or symbolic links to directories, are required in `/var`:
Directory	Description
———	———–
cache	Application cache data
lib	Variable state information
local	Variable data for /usr/local
lock	Lock files
log	Log files and directories
opt	Variable data for /opt
run	Data relevant to running processes
spool	Application spool data
tmp	Temporary files preserved between system reboots