# Teach Yourself to Fish

Knowing where executable commands live and the `man` command can take you a long way. You can teach yourself how to use Linux with this method, but it would be a long, slow process. More often than not, the `man` command will be used as a quick reference. It would be nearly impossible to memorize every option for every command and there is no need to do so when you have the `man` command at your fingertips.

To get help for the `man` command type the letter `h` while viewing a manual page. That will give you a list of commands you can use to navigate or search. Here is the concise version.

`Enter` - Move down one line.

`Space` - Move down one page.

`g` - Move to the top of the page.

`G` - Move to the bottom of the page.

`q` - Quit.

An environment variable is a storage location that has a name and a value. The one we are interested in at the moment is `PATH`. The `PATH` environment variable contains a list of directories that contain executable commands. You can determine the value of `PATH` by prepending it with a dollar sign (`$PATH`) and using the `echo` command to display its value to the screen.

```
$ echo $PATH
/bin:/usr/bin:/usr/sbin:/usr/local/bin
```

When you type in a command at the prompt and press `Enter`, that command will be searched for in the directories in your `$PATH`. In this example, `/bin` will be searched first. If the command is found it will be executed. If it is not found, then `/usr/bin` will be searched and so on. If no executable command is found that matches your request, you will be politely told that it cannot be found.

```
$ whatsupdoc
-bash: whatsupdoc: command not found
```

If you want to know exactly where a command is located you can use the `which` command. If the program `cat` is located in `/usr/bin` and in `/usr/local/bin`, the one which will get executed depends on your `$PATH`.

```
$ which cat
/bin/cat
$ which tac
/usr/bin/tac
```

Putting this all together, you can start looking at what is in each directory in your `$PATH` and use the `man` command to discover what each one of them does and how to use them. Remember, to exit the man command type the letter `q`.

```
$ echo $PATH
/bin:/usr/bin:/usr/sbin:/usr/local/bin
$ cd /bin
$ ls
awk diff cal cat cp date du echo grep groups less more
$ man diff
```

```
NAME
      diff - compare two files
...
$ cd /usr/bin
$ ls
clear crontab cut dos2unix find kill mv pstree pwd sed strings touch ...
$ man touch
```

Note that the output of the above `ls` commands was truncated. In reality there can be hundreds of commands in `/bin` and `/usr/bin`.

Many commands will provide hints for how to use them at the command line. Some commands will accept the `-h` flag, others will accept `--help`, and some will refuse to give you any help at all.

```
$ cal -h

Usage:
 cal [options] [[[day] month] year]

Options:
 -1, --one      show only current month (default)
 -3, --three    show previous, current and next month
 -s, --sunday   Sunday as first day of week
 -m, --monday   Monday as first day of week
 -j, --julian   output Julian dates
 -y, --year     show whole current year
 -V, --version  display version information and exit
 -h, --help     display this help text and exit
$ diff --help
Usage: diff [OPTION]... FILES
Compare files line by line.

  -i  --ignore-case  Ignore case differences in file contents.
  --ignore-file-name-case  Ignore case when comparing file names.
...
```

If you are not sure what command to use, you can search through the man pages with `man -k KEYWORD`. From there you can read the man page for the command or ask it for help with `-h` or `--help`.

```
$ man -k calendar
cal        (1)  - display a calendar
zshcalsys  (1)  - zsh calendar system
```

# Deep Dive

- ExplainShell - Type in a command-line to display help for each item.
- Getting Help From Linux - An article from the Linux Journal on using man pages.
- LinuxManPages.com - This website allows you to search man pages or browse a category of commands and man pages.
- Linux Commands Documentation - Linux commands broken down by category.