launchcode

**John Woolbright, Technical Curriculum Developer**

June, 15, 2022

# Bash Introduction

# Bash

- Bash is the GNU Project shell.
- Bash stands for Bourne Again Shell.
- Bash is a text based shell.

# GNU

- GNU stands for GNU is Not Unix. A confusing, but informative recursive acronym.
- GNU contains many programs that are included with the Linux kernel, one of which is the default shell for Ubuntu: Bash.

# Shell

- A **shell** is an interface between a user and the kernel.
- There are two different types of shells: text based and graphical.
- The Ubuntu graphical windows management shell is called GNOME. The Ubuntu text based shell is called **Bash**.

# Terminal Emulator

- In order to access the Bash shell, which is text based in nature, we must open up an application that can display output and receive our Bash commands.
- The program that manages the input, output, window, minimize window, maximize window, close window, keyboard press events, and mouse events is called a terminal emulator.
- The Bash shell we will be using throughout this class will be presented to us in a terminal emulator.

# Terminal

In the version of Ubuntu we are using the terminal emulator is simply named **terminal**. In other Linux distributions the terminal emulator will likely be named something else.

Terminal emulators are named such because they are emulating the text-based terminals which used to be the **only way to interface with a computer**.

# Bash Shell

The Bash shell is extraordinarily powerful. Almost everything that can be accomplished using a Linux distribution can be completed using the Bash shell.

# Bash Commands

Being a text based shell, Bash is expecting us to invoke various commands.

These commands may include **arguments** and **options**.

# General Bash Command Structure

```
command --option(s) argument
```

The structure is consistent across the commands and programs we will be invoking.

# Example

```
ls -la /home/student
```

The command “ls” is being invoked with the options -l and -a upon the argument /home/student.

# Bash Arguments

A Bash command may require no arguments, one argument only, or many arguments.

# No Argument Example

**“pwd”**

The print working directory command takes no arguments

# Default Argument Example

“ls”

The list contents command will automatically use the current working directory if no arguments are provided. This is known as a default argument.

# One Argument example

```
ls /home/student
```

The ls command has been provided a specific directory in which we want the contents of listed. Instead of using the current working directory, ls will use the provided directory instead.

# Multiple Argument Example

```
rename /home/student/example.file new-name.file
```

Two arguments were provided to the **rename** command. The first argument is the existing file we want to modify. The second argument is the new name we want the file to have.

# Bash Options

A Bash command may be presented with any number of options. The options may modify the command, provide additional information to the command, or change the output of the command.

Options are indicated by using double hyphens (--option-name) or single short-hand hyphens (-o).

# Bash Double Hyphen Option Example

```
ls --all
```

Print all contents of directory, including any hidden files or directories.

# Bash Single Hyphen example

ls -a

-a is the shorthand version of the --all option for the ls command.

Not all commands have both a short and full version. You can learn about the various options by viewing the Reference Manual for a given command.

# Bash Combining Options

```
ls -la
```

You can oftentimes combine multiple options together. The -a flag will list all files/directories including hidden ones.

The -l option displays the output in a long format providing more information about the files and directories.

The shorthand options can be combined together behind a single hyphen.

# Bash Shell Variables

The Bash shell has numerous variables that contain information useful to Bash. They are indicated by the following pattern:

$VARIABLE_NAME

# $BASH

The $BASH shell variable contains the absolute path to the shell this session is using.

# $HOME

The $HOME shell variable contains the absolute path to the home directory of the user that initiated the Bash shell.

# $PATH

The $PATH shell variable contains a collection of all the tools currently accessible to this current Bash Shell session.

# Bash Command Reference Manuals

To learn more about any Bash command you read its Reference Manual.

You can access the Reference Manual by entering man command-name in your Bash Shell.

This will open up the Manual in your terminal. You can explore the provided information with the directional keys on your keyboard. You can exit the Manual by pressing the q key.

# Walkthrough

The walkthrough will take us through some of the basic Bash commands you will be using regularly.

Throughout this class you will continue to learn more about Bash as it is a key tool in most Linux distributions.

launch_code

launchcode.org