Day 2-Linux Core Concepts Part 2

More About the Bash Shell

The Bash (Bourne Again) shell is an upgraded version of the Bourne shell. It provides many helpful features such as:

Tab Completion

Press the `Tab` key after typing a few characters of a command or file name to auto-complete it. For example:

```
cd Doc<Tab>  # Expands to Documents if it exists
```

Command History

View previously entered commands using the `history` command. For example:

```
history
```

Aliases

Create shortcuts for longer commands using aliases. For example:

```
alias up='uptime'
up  # Runs the uptime command
```

Environment Variables

Environment variables are dynamic values used by the shell and processes to store configuration settings and system information. For example, the **$SHELL** environment variable is used to show the default shell. To use environment variables use the **$** sign followed by the variable name.

Accessing Environment Variables

Use the `$` sign followed by the variable name to reference an environment variable. For example:

```
echo $SHELL  # Displays the current default shell
```

Viewing All Environment Variables

Use the `env` command to list all environment variables currently available in the shell.

```
env
```

Setting an Environment Variable

Use the `export` command to set an environment variable so it is inherited by child processes.

```
export EDITOR=nano
```

Temporary Variables Without Export

If you assign a variable without `export`, it is only available in the current shell and not passed to subprocesses.

```
MY_VAR=test
echo $MY_VAR  # Works in the current shell only
```

Making Variables Persistent

To retain environment variables across sessions or reboots, add them to the `~/.profile`, `~/.bashrc`, or `~/.bash_profile` file depending on the system.

```
export JAVA_HOME=/usr/lib/jvm/java-17-openjdk
```

Understanding the PATH Variable

When a user runs a command, the shell uses the `PATH` variable to locate executable files in listed directories.

```
echo $PATH
```

Finding the Location of a Command

Use the `which` command to see the path from which a command is being executed.

```
which python3
# Output: /usr/bin/python3
```

Adding a Directory to PATH

You can append a new directory to the `PATH` variable using `export`.

```
export PATH=$PATH:/opt/myapp/bin
```