Skills Network Editor

author-ide.skills.network/render

Module 2 Cheat Sheet - Introduction to Linux Commands

Getting information

Return your user name:

```
1. whoami
```

Return your user and group id:

```
1. id
```

Return operating system name, username, and other info:

```
1. uname -a
```

Display reference manual for a command:

```
1. man top
```

List available `man` pages, including a brief description for each command:

```
1. man -k .
```

Get help on any command (for eg: `curl`):

```
1. curl --help
```

> This provides a brief overview of the `curl` command's usage and options.

Return the current date and time:

```
1. date
```

Navigating and working with directories

List files and directories by date, newest to last:

```
1. ls -lrt
```

Find files in directory tree that end in `.sh`:

```
1. find -name \'\*.sh\'
```

Return path to present working directory:

```
1. pwd
```

Make a new directory:

```
1. mkdir new_folder
```

Change the current directory:

> **Up one level:**

```
1. cd ../
```

> **To home:**

```
1. cd ~` or `cd
```

> **To some other directory:** `cd path_to_directory`

Remove directory verbosely:

```
1. rmdir temp_directory -v
```

Monitoring system performance and status

List selection of/all running processes and their PIDs:

```
1. ps
```

```
1. ps -e
```

Display resource usage:

```
1. top
```

List mounted file systems and usage:

```
1. df
```

Creating, copying, moving, and deleting files:

Create an empty file or update existing file's timestamp:

```
1. touch a_new_file.txt
```

Copy a file:

```
1. cp file.txt new_path/new_name.txt
```

Change file name or path:

```
1. mv this_file.txt that_path/that_file.txt
```

Remove a file verbosely:

```
1. rm this_old_file.txt -v
```

Working with file permissions

Change/modify file permissions to 'execute' for all users:

```
1. chmod  +x  my_script.sh
```

Change/modify file permissions to 'execute' only for you, the current user:

```
1. chmod u+x my_file.txt
```

Remove 'read' permissions from group and other users:

```
1. chmod go-r
```

Displaying file and string contents

Display file contents:

```
1. cat my_shell_script.sh
```

Display file contents page-by-page:

```
1. more ReadMe.txt
```

Display first 10 lines of file:

```
1. head -10 data_table.csv
```

Display last 10 lines of file:

```
1. tail -10 data_table.csv
```

Display string or variable value:

```
1. echo "I am not a robot"
2. echo "I am $USERNAME"
```

Basic text wrangling

Sorting lines and dropping duplicates:

Sort and display lines of file alphanumerically:

```
1. sort text_file.txt
```

> In reverse order:

```
1. sort -r text_file.txt
```

Drop consecutive duplicated lines and display result:

```
1. uniq list_with_duplicated_lines.txt
```

Displaying basic stats:

Display the count of lines, words, or characters in a file:

> **Lines:**

```
1. wc  -l table_of_data.csv
```

> **Words:**

```
1. wc  -w my_essay.txt
```

> **Characters:**

```
1. wc  -m some_document.txt
```

Extracting lines of text containing a pattern:

Some frequently used options for `grep`:

Option	Description
`-n`	Print line numbers along with matching lines
`-c`	Get the count of matching lines
`-i`	Ignore the case of the text while matching
`-v`	Print all lines which do not contain the pattern
`-w`	Match only if the pattern matches whole words

Extract lines containing the word "hello", case insensitive and whole words only:

```
1. grep  -iw hello  a_bunch_of_hellos.txt
```

Extract lines containing the pattern "hello" from all files in the current directory ending in `.txt`:

```
1. grep  -l hello  *.txt
```

Merge two or more files line-by-line, aligned as columns:

Suppose you have three files containing the first and last names of your customers, plus their phone numbers.

Use `paste` to align file contents into a Tab-delimited table, one row for each customer:

```
1. paste first_name.txt last_name.text phone_number.txt
```

Use a comma as a delimiter instead of the default Tab delimiter:

```
1. paste -d "," first_name.txt last_name.text phone_number.txt
```

Use the `cut` command to extract a column from a table-like file:

Suppose you have a text file whos rows consist of first and last names of customers, delimited by a comma.

Extract first names, line-by-line:

```
1. cut -d "," -f 1 names.csv
```

Extract the second to fifth characters (bytes) from each line of a file:

```
1. cut -b 2-5 my_text_file.txt
```

Extract the characters (bytes) from each line of a file, starting from the 10th byte to the end of the line:

```
1. cut -b 10- my_text_file.txt
```

Compression and archiving

Archive a set of files:

```
1. tar -cvf my_archive.tar.gz file1 file2 file3
```

Compress a set of files:

```
1. zip my_zipped_files.zip file1 file2

2. zip my_zipped_folders.zip directory1 directory2
```

Extract files from a compressed zip archive:

```
1. unzip my_zipped_file.zip

2. unzip my_zipped_file.zip -d extract_to_this_direcory
```

Working with networking commands

Print hostname:

```
1. hostname
```

Send packets to URL and print response:

```
1. ping  www.google.com
```

Display or configure system network interfaces:

```
1. ifconfig

2. ip
```

Display contents of file at a URL:

```
1. curl  <url>
```

Download file from a URL:

```
1. wget  <url>
```