**Day 12 of 90daysofdevops cheat-sheet in Linux, Git - GitHub**

**Finally!! 🎉**

We have completed the Linux & Git-GitHub hands-on and learned something interesting from it.

Now its time to make interesting assignment, which not only will help you for the future but also for the DevOps Community!

So here articulated and documented **"cheat-sheet"** with all the commands learned so far in Linux, Git-GitHub and brief info about its usage.

**Linux & Git Cheat-Sheet**

**Linux Cheat-Sheet:**

**1. File and Directory Operations Commands**

* cd [directory]: Change your current directory to the specified directory.
* ls: Display a list of files and directories in the current directory.
* pwd: Print the absolute path of the current working directory.
* mkdir [directory]: Create a new directory with the specified name.
* rm [file]: Delete the specified file.
* rm -r [directory]: Remove the specified directory and its contents recursively.
* cp [source] [destination]: Copy the file from the source location to the destination location.
* mv [source] [destination]: Move or rename the file from the source location to the destination location.
* touch [file]: Create a new file with the specified name.
* cat [file]: Display the contents of the specified file.
* head file.txt : Display the first 10 lines of a file.
* tail file.txt : Shows the last 10 lines of the file “file.txt”.
* ln : Create links between files.
* find /path/to/search -name “\*.txt” : Searches for all files with the extension “.txt” in the specified directory.

**2. File Permission Commands**

* chmod [permissions] [file]: Change the permissions of the specified file.
* chown user file.txt : Change file ownership.
* chgrp group file.txt : Changes the group ownership of “file.txt” to the specified group.
* umask 022 :Sets the default file permissions to read and write for the owner, and read-only for group and others.

**3. File Compression and Archiving Commands**

* tar -czvf archive.tar.gz files/ : Creates a compressed tar archive named “archive.tar.gz” containing the files in the “files/” directory.
* gzip file.txt : Compresses the file “file.txt” and renames it as “file.txt.gz”.
* zip archive.zip file1.txt file2.txt : Creates a zip archive named “archive.zip” containing “file1.txt” and “file2.txt”.

**4. User Management Commands**

* sudo [command]: Execute the specified command with superuser privileges.
* useradd [username]: Create a new user with the specified username.
* passwd [username]: Set a password for the specified user.
* su [username]: Switch to the specified user.
* userdel [username]: Delete the specified user.
* who : Show who is currently logged in.
* finger : Display information about all the users currently logged into the system, including their usernames, login time, and terminal.
* usermod -a -G GROUPNAME USERNAME : Add an existing user to the specified group. The user is added to the group without removing them from their current groups.
* last**:** Show the recent login history of users.

**5. Process Management:**

* ps: Display a snapshot of the current running processes.
* top: Monitor the real-time system resources and running processes.
* kill [pid]: Terminate a process with the specified process ID.
* pkill process\_name : Terminates all processes with the specified name.
* pgrep process\_name : Lists all processes with the specified name.
* grep : Used to search for specific patterns or regular expressions in text files or streams and display matching lines.

**6. System Information Commands**

* uname -a : Displays all system information.
* whoami : Shows the current username.
* df -h : Displays disk space usage in a human-readable format.
* du -sh directory/ : Provides the total size of the specified directory.
* free -h : Displays memory usage in a human-readable format.
* uptime : Shows the current system uptime.

**7. Networking:**

* ping [host]: Send ICMP Echo Request packets to the specified host to check network connectivity.
* ifconfig: Display information about the network interfaces on your system.
* netstat : Display network connections and statistics.
* ssh : Securely connect to a remote server.
* scp : Securely copy files between hosts.
* wget : Download files from the web.
* curl : Transfer data to or from a server.

**Git-GitHub Cheat-Sheet:**

**Installation**

GitHub for Windows htps://[windows.github.com](http://windows.github.com)

GitHub for Mac htps://[mac.github.com](http://mac.github.com)

For Linux and Solaris platforms, the latest release is available on the official Git web site.

Git for All Platforms htp://[git-scm.com](http://git-scm.com)

**SETUP**

Configuring user information used across all local repositories.

1. git config --global[user.name](http://user.name)“[firstname lastname]” : set a name that is

identifiable for credit when review version history.

1. git config --global[user.email](http://user.email)“[valid-email]” set an email address that will be associated with each history marker.
2. git config --global color.ui auto set automatic command line coloring for Git for easy reviewing.

**Repository Management:**

* git init: Initialize a new Git repository in the current directory.
* git clone [repository]: Clone the specified repository from GitHub to your local machine.
* git add [file]: Add the specified file to the staging area for the next commit.
* git reset [file] : Unstage a file while retaining the changes in working directory.
* git diff : Diff of what is changed but not staged.
* git diff --staged : Diff of what is staged but not yet committed.
* git status : Show modified files in working directory, staged for your next commit.
* git commit -m "[message]": Commit the staged changes with a descriptive message.
* git reset --hard [commit] : Clear staging area, rewrite working tree from specified commit.
* git stash : Save modified and staged changes.
* git stash list : List stack-order of stashed file changes.
* git stash pop : Write working from top of stash stack.
* git stash drop : Discard the changes from top of stash stack.
* git rm [file] : Delete the file from project and stage the removal for commit.
* git mv [existing-path] [new-path] : Change an existing file path and stage the move.
* git log : Show the commit history for the currently active branch.
* git log --stat -M : Show all commit logs with indication of any paths that moved.
* .gitignore : Specifies intentionally untracked files to ignore.
* git cherrypick : Powerful tool for selectively applying specific commits between branches.

**Collaboration:**

* git branch: List all the branches in the repository.
* git branch [branch-name]: Create a new branch with the specified name.
* git checkout [branch]: Switch to the specified branch.
* git checkout -b [branch]: Create a new branch with the specified name.
* git merge [branch]: Merge the changes from the specified branch into the current branch.
* git rebase [branch] : Apply any commits of current branch ahead of specified one.
* git remote add [name] [url]: Add a remote repository with the specified name and URL.
* git fetch [remote]: Fetch the latest changes from the specified remote repository.
* git pull [remote] [branch]: Pull the latest changes from the remote branch and merge them into the current branch.
* git push [remote] [branch]: Push the local commits to the remote branch.

As we wrap up the Linux and Git/GitHub section, it's time to gear up for the next assignment, Get ready for new learnings, challenges, and discoveries. Stay tuned!

Thanks for reading...!!!

Happy Learning...!!!!