Get to Git Cheatsheet

OSC Linux Committee

Bash

Why Bash?

Bash is the default shell used by most unix-based operating systems including Linux and OS X making it the norm between developers when it comes to picking a shell.

Bash basics

Bash has a bundle of built-in commands that makes interacting with your system easier.

List of Must Know bash commands to survive a terminal session

- pwd: Print name of current working directory. (pwd → print working directory)
- cd: Change the working directory. (cd → change directory)
- ls: List directory content. (ls → list)
- touch: Create a new file.
- cat: Display content of files.
- mkdir: Create a new directory. (mkdir → make directories)

- rm: Remove files. (rm → remove)
- cp: Copy files. (cp → copy)
- mv: Move files. (mv → move)

Git

Git is a fast, scalable, distributed revision control system with an unusually rich command set that provides both high-level operations and full access to internals.

How to write Git commands?

Git is a command-line tool that does not have GUI (Graphical User Interface) by default, so developers have to know how to interact with it. Git commands are usually written in the following format

```
$ git <command> [<args>]
```

Git Commands

- init: Create an empty Git repository or reinitialize an existing one.

```
$ git init
```

- status: Show the working directory status.

```
$ git status
```

- add: Add file contents to the Staging.

```
$ git add [file/s]
```

- commit: Record changes to the repository.

```
$ git commit -m "Type your commit message here"
```

- log: Show commit logs.

```
$ git log
```

- clone: Clone a repository into a new directory.

```
$ git clone [URL]
```

- pull: Fetch from and integrate with another repository or a local branch.

```
$ git pull
```

- push: Update remote refs along with associated objects.

```
$ git push
```