

# | Ebook Version Control System |

---

| VCS = Version Control System

- Suatu system yang menyimpan dan mengelola rekaman dari source code
- 

| Tipe Version Control

- **Local Version Control**

( Version control yang berjalan pada local komputer )

- **Centralized Version Control** (subversion)

( Version control yang berjalan pada local dan di server(database), di local hanya versi yang terbaru dan di server full versinya )

- **Distributed Version Control** (Git dll)

( alternatif dari CVCs, dalam DVCs tidak hanya versi terakhirnya saja namun semua riwayat di copy, sehingga jika server down maka masih biasa berkerja)

---

| Git = Software VCS

| <https://git-scm.com/>

| Windows | Mac OS | Linux/Unix

| Sejarah Git dan tentang Git

- latar belakang OpenSource projek Linux Kernel
- tahun 1991-2001 Linux Kernel hanya di develop hanya dengan memanfaatkan patch dan archived files
- 2002 Linux Kernel memulai dengan DVCs bernama bitkeeper
- 2005 hubungan dengan perusahaan bitkeeper kurang baik, sehingga pembuat linux, **Linus Torvalds** membuat DVCs sendiri
- Git di perkenalkan pertama kali pada tahun 2005
- Git menggunakan algoritma hashing SHA-1
- git lebih mengedepankan perubahan yang terjadi di file tersebut

---

| Istilah pada git

- Repo (repository, sebutan projek di Git, folder kosong yang akan kita jadikan git

repo)

- Branch (cabang bebas dari commit seperti master, main dll)
- Hash (penanda unik pada commit berupa angka dan huruf, Git menggunakan algoritma SHA-1, hash harus di jaga agar data tetap valid jika hash yang di belakang rusak maka data yang di depan tidak valid / rusak)
- Commit (rekaman / snapshot pada repo)
- Snapshot (berisikan semua perubahan yang terjadi pada semua file yang kita commit, dan setiap snapshot akan menghasilkan hash)
- Remote (sumber / sever yang memiliki repo github, gitlab)
- Checkout (berpindah ke sebuah commit)
- Clone (menggambil repo dari remote)
- Push (mengirim commit ke repo server)
- Pull (menggambil commit dari repo server)
- Merge (menggabungkan branch)
- HEAD (pointer menuju hash paling akhir / versi terbaru)
- .git (adalah isi dari repository kita)

---

## | 3 Area pada git / The Tree States

- **working directory (red)**  
(tempat file yang belum di track / new file)
- **staging index / staging area (green)**  
(tempat file yang sudah di track dan siap untuk commit)
- **commit (black)**  
(file sudah di commit, dan aman di repo)

---

## | Command git dan fungsinya

### | Instalasi

- > sudo/apt install git (debian / ubuntu)  
(menginstall git di terminal)
- > git  
(menampilkan git / seperti help)
- > git --version  
(menampilkan version git)

### | Configuration

> git config --global user.name "nama anda"  
(mengisi nama untuk git)

> git config --global user.email "email anda"  
(mengisi email untuk git)

> git config --list --show-origin

> git config --list

(melihat seluruh configuration)

> git config --global alias.<nama shortcut>  
comand-git

(shortcut untuk command git)

contoh: git config --global alias.ko commit

contoh: git config --global alias.logone "log --  
oneline"

> alias nama-shortcut="command git"

(membuat shortcut git secara sementara)

contoh: alias log="git log"

| konfigurasi text editor default VScode

```
> git config --global core.editor "code --wait"
> git config --global diff.tool "default-difftool"
> git config --global difftool.default
difftool.cmd "code --wait --diff \${LOCAL}
\${REMOTE}"
```

---

## | Command untuk mengoperasikan git

```
> ls -l
```

(untuk melihat isi file .git)

```
> git init {harus masuk ke dalam folder yang
akan di jadikan repo}
```

(untuk membuat folder menjadi repositori)

```
> git add name-file
```

(menambahkan file ke staging index)

```
> git checkout <branch>
```

(berpindah branch)

```
> git branch <nama branch>
```

(membuat branch baru)

> git status

(melihat status repositori)

> git commit -m "pesan"

(commit file ke repository)

> git log

(menampilkan hasil history commit)

> git log --oneline

(menampilkan hasil history commit dengan pendek)

> git merge name-branch

(menggabungkan branch)

> git diff

(melihat perubahan yang terjadi pada file yang kita ubah)

> git clean -f

(membatalkan penambahan file yang ada di working directory)



> git restore nama-file (**working directory**)  
• (membatalkan perubahan / penghapusan file yang ada di working directory)

> git restore --staged nama-file (**staging index**)  
(membatalkan penghapusan / perubahan yang ada di staged index)

> git log --oneline --graph  
(menampilkan histori commit dengan grafik secara sederhana)

> git log --all --decorate --oneline --graph  
(menampilkan histori commit dengan grafik yang saling terhubung)

> git show code-hash  
(melihat perubahan yang terjadi pada commit tertentu)

> git diff hash1 hash2  
(membandingkan commit1 dengan commit2, bukan membandingkan hasil dari commit1)



sampai ke commit2, namun hanya membandingkan hasil perubahan commit1 dan hasil akhir commit2)

> git difftool hash1 hash2

(melihat perbandingan menggunakan VScode)

> git reset --mode hash

(membatalkan perubahan di commit dengan cara reset commit dengan mode)

## mode

- --soft (memindahkan pointer HEAD namun tidak melakukan perubahan di staged index dan working directory)

- --mixed (default)

(memindahkan pinter HEAD dan mengubah staged index menjadi sama seperti repository namun tidak mengubah apapun di working directory)

- --hard (memindahkan pointer HEAD dan mengubah staged index dan working directory menjadi sama seperti repository)

> git commit --amend -m "pesan"

(amend commit, alternatif dari reset commit dan melakukan perubahan lalu commit ulang, hal ini bisa dilakukan tanpa manual, dengan amend commit, namun amend akan merubah data commit karena data bertambah)

> git checkout hash -- nama-file

(melihat versi file sebelumnya)

> git checkout branch/hash

(kembali ke versi yang kita inginkan)

> git blame nama-file

(melihat siapa yang mengubah file tersebut)

---

## | Note

setiap perubahan itu harus di commit entah itu menghapus, mengedit, merename dll

tidak ada cara untuk membatalkan perubahan yang sudah di commit namun ada cara yaitu Rollback commit / Revert commit

operasi rename file dalam git adalah yaitu operasi penggabungan menghapus file lalu membuat file baru dengan isi yang sama namun dengan name-file yang berbeda, sehingga terdeteksi sebagai rename file

.gitignore file yang tidak di track oleh git, dan juga isi file nya menuju pada file yang tidak kita inginkan

| Shortcut git

> git commit -am "pesan"

(cara cepat commit, hanya berlaku ketika file yang modified)

> git add .

(menambahkan semua file dari working directory ke staging index)

