Git Command cheat sheet by kang

ŀ축 명령어 설정

\$ git config --global alias.co checkout \$ git config --global alias.br branch \$ git config --global alias.ci commit \$ git config --global alias.cm "commit -m" \$ git config --global alias.cam "commit -am" 간편 파일 편집과 내용 보기 \$ git config --global alias.st status \$ echo "1st version" > file1 # 새 파일 or 수정 \$ git config --global alias.ss "status -s" \$ echo "Add a line" >> file1 # 파일에 추가 \$ git config --global alias.lg "log --oneline" 전체 설정: [설치폴더]/Git/etc/gitcol \$ cat > file2 # 여러 줄 입력(새 파일 or 수정) \$ git config --global alias.lgg "log -oneline -graph" \$ git config --system core.autocrlf true 1st version \$ git config --global alias.lga "log -oneline -graph --all" \$ git config --system core.safecrlf false Add a line [Enter] \$ git config --global alias.rs restore [ctrl + C] 사용자 설정: [사용자계정]/.gitcol \$ git config --global alias.di diff \$ touch file1 file2 # 빈 파일 생성 \$ git config --global alias.sh "show" \$ git config --global user.name hskang \$ cat file1 file2 # 파일 내용 보기 ✓ ☐ new-repo \$ git config --global alias.tg "tag" \$ git config --global user.email hskang@gmail.com 🗸 🥛 .git \$ git config --global alias.sts "stash" \$ git config --global init.defaultBranch main hooks \$ git config --global alias.rs "reset" 저장소 설정: [basic]/.git/con info \$ git config --global alias.rb "rebase" \$ git config --local core.editor 'code --wait' \$ git config --global alias.mg "merge" \$ git init 브랜치 이름 수정 \$ git config --global alias.fc "fetch" nack 저장소 생성 \$ git init . # 현재 폴더 \$ git config --global alias.rs "reset" refs \$ git branch -m | -M new-branch \$ git init new-repo \$ git config --global alias.ps "push" heads Initialized empty Git repository in C:/git-master/new-repo/.git/ __ tags \$ git config --global alias.pl "pull" \$ ls -a WD(Working Directory) \$ rm -rf .git # 저장소 기능 상실 \$ cat file \$ git init # 현재 폴더 저장소 생성 \$ rm file \$ mv fromfile tofile git add . git diff \$ git add * git restore file git add --all | -A git restore --worktree | -W file git add file \$ git rm file \$ git Is-files SA(Staging Area, Index) # file 해시값 조회 git restore --staged --worktree file \$ git Is-files -s file git restore -S -W file \$ git rm --cached file \$ git hash-object file # SA 모든 해시값 조회 git diff HEAD 🛱 git commit -m "message" \$ git Is-files -s sit restore --source=HEAD --worktree | -W file \$ git commit --am "messag<mark>e</mark>" \$ git restore -s HEAD --worktree | -W file # SA 파일 내용 조회 git restore --source=HEAD file \$ git cat-file -p <해시값> git restore -source HEAD file git diff --staged | --cached git restore -- staged | -S file git restore --source=HEAD --staged | -S file git restore -s HEAD --staged | -S file \$ git blame file \$ git log **GR(Git Local Repository)** \$ cat file \$ git log --oneline \$ git log --graph --all \$ git log --stat -n s git commit --amend HEAD \$ git log --patch | -p 이전 버전으로 재성 \$ git reset HEAD~ \$ g<mark>it</mark> checkout main diff HEAD~2 HEAD \$ git reset --mixed HEAD~ git checkout/ git restore --source=HEAD~ -W file # WD에 복구 \$ git reset --soft HEAD~ git switch main git restore -s HEAD~ -W file \$ git reset --hard HEAD~ g<mark>it switch -</mark> \$ git restore --source=HEAD~ -S file # SA에 복구 \$ git restore -s HEAD~ -S file \$ git show HEAD~ HEAD[^] \$ git restore --source=HEAD~ -W -S file # WD, SA에 복구 \$ git show --quiet | -q \$ git show -- oneline HEAD~ \$ git show --name-only \$ git restore -s HEAD~ -W -S file \$ git show -s HEAD~ \$ git show --name-status <u>브랜치 생성과 삭제, 목록 보기</u> git checkout HE \$ git branch new-branch git switch detach HEAD~2 \$ git checkout -b new-branch 버전 태그 \$ git switch -c new-branch \$ git tag v2.0.0 \$ git branch --delete | -d branch-name \$ git tag v1.0.0 HEAD~ \$ git branch -D branch-name \$ git tag -a v1.1.0 -m "1st version" 깃 저장소 \$ git branch \$ git tag -a v1.1.0 # 편집기에서 메시지 추가 \$ git branch --list \$ git tag -a v1.1.0 HEAD~ # 편집기에서 메시지 추가 \$ git branch -a \$ git tag -a v1.1.0 HEAD~ -m "1st version" \$ git branch -v RR(Remote Repository, Github) 저장소 복제 \$ git clone URL \$ git clone URL NEW-DIR \$ git clone URL.