

Git Tips

Tutorials

- [50+ Useful Git Commands for Everyone](#)

?????

```
# Using git to edit the configuration
git config global --edit

# List the global configurations
git config --global --list

# Using vi/cat to edit the configuration
vi ~/.gitconfig

# Set the author's email address and name
git config --global user.email "alang.hsu@gmail.com"
git config --global user.name "Alang Hsu"

# Set default editor
git config --global core.editor "vi"

# Set default branch name
git config --global init.defaultBranch "main"
```

.gitconfig

```
[user]
email = alang.hsu@gmail.com
name = Alang Hsu
[core]
editor = vi
[init]
defaultBranch = main
```

?????

```
mkdir test-git-push
cd test-git-push
git config --global user.name "<user-name>"
git config --global user.email "<your-email-addr>"
git init
echo "Test Git Push only" > README.md
git add .
git commit -m "Initial commit"
git remote add origin https://<user-name>@github.com/a-lang/test-git-push.git
git remote -v
git push -u origin master
```

Git commit

??? commit ??

```
git commit -m "Fixed a typo in somewhere"
```

? add ?? commit (-a)

- ?? edited & deleted
- ?????? commit ??????????????
- ????????????

```
git commit -a -m "<commit-message>"
```

????????

```
# [] Vim []
git config --global core.editor "vim"

# [] -m []
git commit
```

Commit ??????

1. ??????????????
2. ?????? 50 ??????????????
3. ????????
4. ??????????

5. ????????
6. ??????? 72 ??????????????????
7. ????? what ?? why vs. how

??(??)???? Commit `--amend`

```
git commit --amend

# □ add □□□ commit
git commit -a --amend
```

`--amend` ?????commit ????? commit ????

???????????????????????????????? commit ??????? commit ????????????

????? push ? commit ????? --amend????????????????????

??? HEAD

HEAD ?????? checked-out ????????????????????

Git diff

```
# □□□ git add □□□□□□□□□□ commit □□□□□
git diff file

# □□□□□□□□□□
git diff --word-diff file1 file2

# □□□ Commit □□□□□
git log --graph --pretty=format:'%Cred%h%Creset -%C(yellow)%d%Creset %s %Cgreen(%cr) %C(bold
blue)<%an>%Creset' --abbrev-commit --date=relative
git diff <old-commit-id> <new-commit-id>
```

Undo in Git

????????????????

“ git restore -p : ??????????????????????????????

Local unstaged changes (??? add)

```
git status
git restore <filename>
```

????? staging ?????? checkout ???

checkout ????????? committed ? staged ???

```
git checkout <file-name>
```

Local staged changes (?? add ??? commit)

```
git status
git restore --staged <filename>
git restore <filename>
```

Local committed changes (?? commit ??? push)

```
git status
git log
git reset --soft HEAD~
git log
```

“ **NOTE:** ?????????? commit????????? commit?????? git reset <commit-id> ??
commit-id ??? git log --oneline ???

????? commit ?? push ?????????? git revert ?

Public committed changes (?? push)

```
git log --oneline
git revert <last-commit-id> --no-edit
git push
git log
```

NOTE: ??? `git revert` ??log ???? Revert "XXXX" ? commit ?????? commit
???????

Git Prompt with bash

.bashrc:

```
# Kali-like Prompt with Git status

git_stats() {
    local STATUS=$(git status -s 2> /dev/null)
    local UNTRACKED=$(echo "$STATUS" | grep '^??' | wc -l)
    local STAGED=$((echo "$STATUS" | grep '^M ' | wc -l) + $(echo "$STATUS" | grep '^D ' | wc -l) + $(echo
"$STATUS" | grep '^R ' | wc -l) + $(echo "$STATUS" | grep '^C ' | wc -l) + $(echo "$STATUS" | grep '^A ' | wc -l))
    local DRC=$((echo "$STATUS" | grep '^ D' | wc -l) + $(echo "$STATUS" | grep '^ R' | wc -l) + $(echo
"$STATUS" | grep '^ C' | wc -l))
    local MODIFIED=$(echo "$STATUS" | grep '^ M' | wc -l)
    local STATS=""
    if [ $UNTRACKED != 0 ]; then
        STATS="\e[43m untr: $UNTRACKED "
    fi
    if [ $MODIFIED != 0 ]; then
        STATS="$STATS\e[43m mod: $MODIFIED "
    fi
    if [ $DRC != 0 ]; then
        STATS="$STATS\e[43m drc: $DRC "
    fi
    if [ $STAGED != 0 ]; then
        STATS="$STATS \e[42m staged: $STAGED "
    fi
    if [ ! -z "$STATS" ]; then
        echo -e "\e[30m $STATS\e[0m"
    fi
}

function origin_dist {
    local STATUS=$(git status 2> /dev/null)
    local DIST_STRING=""
    local IS_AHEAD=$(echo -n "$STATUS" | grep "ahead")
}
```

```

local IS_BEHIND=$(echo -n "$STATUS" | grep "behind")
if [ ! -z "$IS_AHEAD" ]; then
    local DIST_VAL=$(echo "$IS_AHEAD" | sed 's/^[^0-9]*//g')
    DIST_STRING="$DIST_VAL AHEAD"
elif [ ! -z "$IS_BEHIND" ]; then
    local DIST_VAL=$(echo "$IS_BEHIND" | sed 's/^[^0-9]*//g')
    DIST_STRING="BEHIND $DIST_VAL"
fi
if [ ! -z "$DIST_STRING" ]; then
    echo -en "\e[97;45m $DIST_STRING"
fi
}

__PS1_GIT_BRANCH='`__git_ps1` '
__PS1_GIT_DIST='`origin_dist` '
__PS1_GIT_STATS='`git_stats` '

if $__git_ps1 2>/dev/null;then

PS1="\[\033[38;5;209m\] ──[\[\033[38;5;141m\]\u\[\033[38;5;209m\]@\[\033[38;5;105m\]\h\[\033[38;5;231m\]:\w\[\033[38;5;209m\]]\[\033[33m\]${__PS1_GIT_BRANCH}${__PS1_GIT_DIST}${__PS1_GIT_STATS}\[\033[00m\]\n\[\033[38;5;209m\] └─\[\033[38;5;209m\]\\$\[\033[37m\] "
else
    source /usr/share/git-core/contrib/completion/git-prompt.sh

PS1="\[\033[38;5;209m\] ──[\[\033[38;5;141m\]\u\[\033[38;5;209m\]@\[\033[38;5;105m\]\h\[\033[38;5;231m\]:\w\[\033[38;5;209m\]]\[\033[33m\]${__PS1_GIT_BRANCH}${__PS1_GIT_DIST}${__PS1_GIT_STATS}\[\033[00m\]\n\[\033[38;5;209m\] └─\[\033[38;5;209m\]\\$\[\033[37m\] "
fi

```

Rename & Delete files

```

# Deleting
git rm my.sh

# Renaming
git mv old.sh new.sh

```

Git Alias

- [10 levels of Git aliases: Beginner to intermediate concepts](#)

```
git config --global alias.st status
git config --global alias.c commit
git config --global alias.br branch
```

```
~/gitconfig
```

```
[alias]
  st = status
  c = commit
  loo = log --oneline
  # [commit] commit [comment]
  onemore = commit -a --amend --no-edit
  # [commit] commit, [commit]
  undo = reset --soft HEAD^
  # [commit] commit, [commit]
  cancel = reset --hard HEAD^
```

Gitignore

Git ???

????????????????:

0. ?????????? .o .so ?
1. ??? Node.js ?????? node_modules ???
2. ? Log ?????????????????? .log ?
3. ?????????????? .env ?
4. ??? (IDE) ??????????

```
.gitignore
```

```
# ignore all directories with the name test
test/

.env
log/
node_modules/
```

```
make/*.o
make/*.so
```

```
# ignores all .md files
.md
```

```
# does not ignore the README.md file
!README.md
```

Git log

- ?? commit-id ??????? 4 - 8 ???????

```
# [4] commit [2]
```

```
git log
```

```
git log --oneline
```

```
# -p: patch, [4] commit [4]
```

```
git log -p
```

```
git log -p -2 # [4] commit [4]
```

```
# [4] commit [4]
```

```
git show <commit-id>
```

```
# [2] commit [2], [8]
```

```
git --stat
```

```
#
```

```
git log --graph --pretty=format:'%Cred%h%Creset -%C(yellow)%d%Creset %s %Cgreen(%cr) %C(bold blue)<%an>%Creset' --abbrev-commit --date=relative
```

```
* 957ae62 - (HEAD -> master, origin/master, origin/HEAD) Fixed the bug if the file config doesn't exist (4 7 [4]) <A-Lang>
```

```
* 2b64ce6 - Just fixed some no-wrap (4 8 [4]) <A-Lang>
```

```
* 1eab2be - Added the title in the window (4 8 [4]) <A-Lang>
```

```
* 5f139b3 - Added the function contents_edit (4 8 [4]) <A-Lang>
```

```
* 727a72b - Removed the directory sshto-1.0 (4 8 [4]) <A-Lang>
```

```
* 764c2a4 - fix download with changed user (4 9 [4]) <Ivan Marov>
```

```
* b8b718f - make pause if error occurred in go_to_target (4 11 [4]) <Vaniac>
```

```
* cf20d21 - new screenshot and readme (5 [4]) <Vaniac>
```



```
* 1bcbb6a - ref (5) <Vaniac>
* 48df1bd - new screenshot and readme (5) <Vaniac>
```

Git revert (Rollback)

?? `git revert` ????? rollback ??? commit ?????????????????? Revert ? commit ??????
commit ?????????? (?? `git reset` ??)

Rollback ??? commit ?????

```
# Rollback the latest commit
git revert HEAD
```

Rollback ?? commit ?????

```
“ ????? revert ? commit ?????????????????????????? git merge
????????????????? revert????? git revert --abort
```

```
git revert <commit-id>
```