Git Tips

Tutorials

- 50+ Useful Git Commands for Everyone

?????

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

# 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 init
git config --global user.name "<user-name>"
git config --global user.email "<your-email-addr>"
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 ???

```
git mv <old-folder> <new-folder>
```

Commit ??

??? commit ??

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

? add ?? commit (?? edited & deleted)

```
git commit -am "<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

?? Comments

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

?????? Comment

```
git commit -v --amend
```

Clone for specified branch version

```
# Specified version
git clone -b 8.3.0 https://github.com/OpenSIPS/opensips-cp.git /var/www/opensips-cp
```

SSH Key Authentication

```
# 建立 ssh-key
# 預設路徑: ~/.ssh/id_rsa (private key) , ~/.ssh/id_rsa.pub (public-key)
ssh-keygen -t rsa -b 4096 -C "alang@my-linux-desktop"

# Gitlab 網站匯入 public ssh-key
# 測試 ssh key authentication
# 如果輸出 Welcome to GitLab, @alang! 表示連線認證成功
ssh -T git@gitlab.shurafom.eu

# 下載專案
# NOTE: 位址必須是 git@XXX.xxx.xxx 開頭。
# 如果使用 https:// 則必須改用 Personal Token 認證方式
git clone git@gitlab.shurafom.eu:myproject/myprog.git
```

diff

```
# 還沒執行 git add 前, 比對目前檔案與最近一次 commit 版本的內容差異
git diff file

# 比對兩個檔案的內容差異
```

```
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

Local unstaged changes *(??? add)*

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

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
```

Git Rebase

- ? PR ???? Git rebase ????? commit ??

Revision #26
Created 19 October 2022 13:23:00 by Admin
Updated 1 May 2024 17:28:20 by Admin