

Git - Config

add user name

```
git config [--global] user.name "[name]"
```

add user email

```
git config [--global] user.email "[email address]"
```

show the git config info

```
git config --list
```

Git - Create a new repository

Initialize the current dir to be repository

```
cd <project_dir>
```

```
git init
```

```
touch README
```

Cloned from remote repository

```
git clone <url>
```

Git - Add and Remove file

Show the current status in the workspace and index

```
git status
```

Add file from workspace into index

```
git add <file1> <file2> ...
```

Add all files from current dir into index

```
git add .
```

Add all tracked file and untracked file from the workspace into index

```
git add -A .
```

Delete tracked file in the index

```
git rm <file1> <file2> ...
```

Move tracked file in the index into workspace and untrack it

```
git rm --cached <file1> <file2> ...
```

Git - Commit

Commit all files in the index into local repository

```
git commit -m "commit message"
```

Recommit the previous commit with new changed file and commit message

```
git commit --amend <file1> <file2> ... -m "new message"
```

Git - Tag

Show all tags

```
git tag
```

Create a new tag at current commit object

```
git tag -a <tag_flag> -m "tag comment message"
```

Create a new tag at specified commit object

```
git tag -a <tag_flag> <commit_id> -m "tag comment message"
```

Show specified tag info in detail

```
git show <tag_flag>
```

Git - Undo modifies

Discard tracked files's modification in the workspace

```
git checkout -- <file> ...
```

Move the added file from index into workspace

```
git reset HEAD <file>...
```

Reset code with --soft para, changes in the index and workspace are reserved

```
git reset --soft <commit-id>
```

Reset code with --mixed para, changes in the index are discarded, however changes in the workspace are reserved

```
git reset --mixed <commit-id>
```

Reset code with --hard para, the changes in the index and workspace are discarded

```
git reset --hard <commit-id>
```

Git - Undo modifies (cont)

Revert code and create a new commit object for this operation. In this way, it will not overlap the commit history.

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

Undo the previous commit changes and create a new commit object for this operation

```
git revert HEAD~1
```

[Stackoverflow-What's the difference between Git Revert, Checkout and Reset?](#)

Git - Branch

List all branches in the local repository

```
git branch
```

List all branches in the local repository and remote repository

```
git branch -a
```

List all local branch info in detail

```
git branch -vv
```

Create a new branch

```
git branch <branch-name>
```

Create a new branch with specified commit

```
git branch <branch-name> <commit-id>
```

Checkout to another branch

```
git checkout <branch-name>
```

Delete branch in local repository

```
git branch -d <branch-name>
```

Build tracking relationship between local-branch and remote-branch

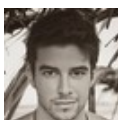
```
git branch --set-upstream-to=origin/<remote-branch-name> <local-branch-name>
```

Cancel the tracking relationship between local-branch and remote-branch

```
git branch --unset-upstream <local-branch-name>
```

List the changed files between two branches

```
git diff <branch1> <branch2> --stat
```



By [woshijpf \(woshijpf\)](#)
cheatography.com/woshijpf/
woshijpf.github.io/

Published 17th October, 2016.
Last updated 17th October, 2016.
Page 1 of 2.

Sponsored by [Readable.com](#)
Measure your website readability!
<https://readable.com>

Git - History

Show all commit log

```
git log
```

Show all commit log in shot format

```
git log --pretty=short
```

One line show one commit log

```
git log --pretty=oneline
```

Show all changed files in every commit log

```
git log --stat
```

Show the change history of specified file

```
git log --follow <file-name>
```

Show all diff info between two adjacent commit

```
git log -p
```

Show who changed file

```
git blame <file-name>
```

Show the recently commit log and branch checkout log

```
git reflog
```

Git - Remote Push And Update

List all remote repo info

```
git remote -v
```

Show specified remote repo info

```
git remote show <remote>
```

Add remote repo and named

```
git remote add <remote> <url>
```

Fetch all changed in the remote repo

```
git fetch <remote>
```

Fetch + Merge

```
git pull <remote> [remote-branch]:[local-branch]
```

Push local repo changes to remote repo

```
git push <remote> [local-branch]:[remote-branch]
```

Delete remote branch in remote repo

```
git push <remote> :[remote-branch]
```

Set -u parameter and build the tracking relationship between local repo and remote repo

Git - Remote Push And Update (cont)

```
git push -u <remote> [local-branch]:[remote-branch]
```

Git - Merge/Rebase

Merge branch into current branch

```
git merge <branch-name>
```

Rebase into branch

```
git rebase <branch-name>
```

Rebase Frequently Usage

```
git checkout <feature-branch>
```

```
git rebase master
```

```
git checkout master
```

```
git merge <feature-branch>
```

Abort Rebase

```
git rebase --abort
```

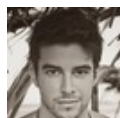
The Golden Rule of Rebasing

Don't rebasing master branch into your personal feature branch

Conflicts against base file

```
git diff --base <file-name>
```

What's the difference between 'git merge' and 'git rebase'?



By [woshijpf \(woshijpf\)](#)
cheatography.com/woshijpf/
woshijpf.github.io/

Published 17th October, 2016.
Last updated 17th October, 2016.
Page 2 of 2.

Sponsored by [Readable.com](#)
Measure your website readability!
<https://readable.com>