# What is the Git?

A tool for version control, recording the path of your file revisions, and for collaborated work.

## Start A Git:

* Step 1:

Git Bash at an empty directory, input “**git init**”, and u get a .git file.

* Step 2:

Input “**git add .**” to add all files into local repo, or git add <filename.format>

* Step 3:

Input “**git commit -m <message>**”, the “message” denotes the description of your commitment. (or includes step2 and 3 with “**git commit -am <...>**”)

Then, u can type “**git status**” to checkout the file in your local repo.

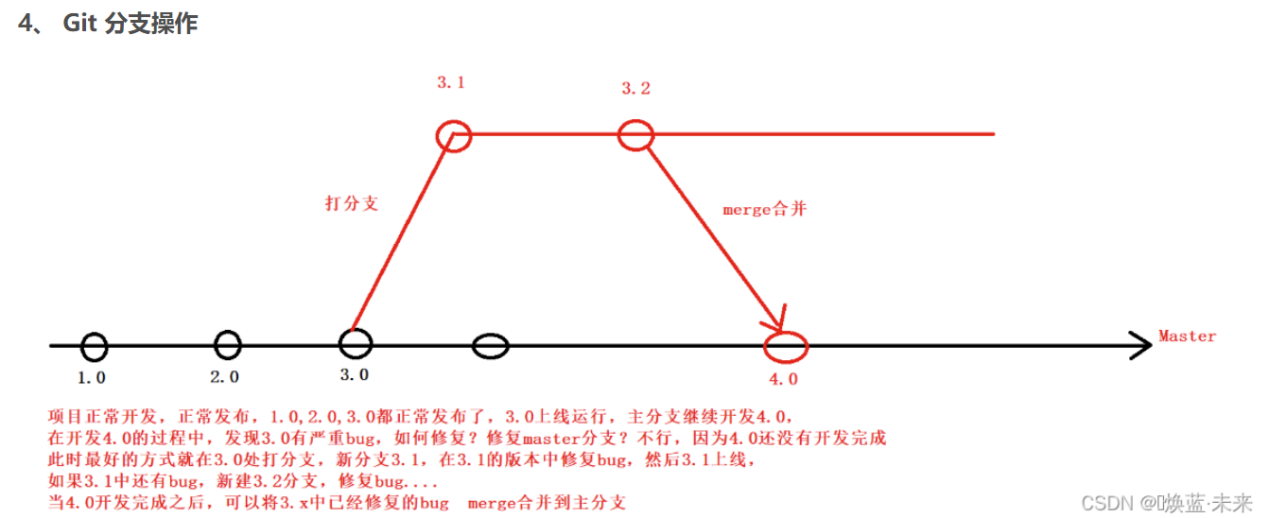
## Back To A former version:

Type “**git reflog**” u can see the version number at the beginning of each sentence, “**git log**” to know the commit history.

Print “**git reset --hard** **<version number>**” to get back!

Thus, u can go to anywhere u have worked, forward or backward.

## Branch:



Branches like a copy of your current work. The main complete work won’t changed if u revised it at the branch, thus entails a perfect platform for minor revision.



It can also use “**git checkout -b <name of branch>**” to quickly build a new branch and select it.

A more detailed lists:

**git branch --merged**  # 显示所有已合并到当前分支的分支

**git branch --no-merged**  # 显示所有未合并到当前分支的分支

**git branch -m master master\_copy** # 本地分支改名

(git branch -m <name> 将当前分支改名为<name>)

**git checkout -b master\_copy**  # 从当前分支创建新分支master\_copy并检出

**git checkout -b master master\_copy**  # 上面的完整版

**git checkout features/performance** # 检出已存在的features/performance分支

**git checkout --track hotfixes/BJVEP933** # 检出远程分支hotfixes/BJVEP933并创建本地跟踪分支

**git checkout v2.0** # 检出版本v2.0

**git checkout -b devel origin/develop** # 从远程分支develop创建新本地分支devel并检出

**git checkout -- README**  # 检出head版本的README文件（可用于修改错误回退）

**git merge origin/master** # 合并远程master分支至当前分支

**git cherry-pick ff44785404a8e** # 合并提交ff44785404a8e的修改

**git push origin master** # 将当前分支push到远程master分支

**git push origin :hotfixes/BJVEP933**  # 删除远程仓库的hotfixes/BJVEP933分支

Git branch -d <branch name> # delete branch

## Remote repo interaction:

All of your revisions or your work will finally connect with other people work, so its essential to commit your work in the remote server.

* Step one: type “**ssh-keygen -t rsa -C ["youremail@example.com](mailto:\"youremail@example.com\")**["](mailto:\"youremail@example.com\") ,to generate ssh key for identification.

The **["youremail@example.com](mailto:\"youremail@example.com\")**["](mailto:\"youremail@example.com\") is the register email in your Github.

* Then u got key, open id\_rsa.pub to copy the key, into your github, account/SSH AND GPG KEYS, paste it.
* Then u got a connection, check it with “**ssh -T [git@github.com](mailto:git@github.com\”)”**

**----------------------------------------------------**

git remote -v ： show all remote u have, with the corresponding URL

Git remote show: show all remote repo

Git remote rename old-name new-name: rename the repo

Git remote remove name: delete remote repo

**-------------------------------------------**

And u will get: 

Otherwise goto github :

<https://docs.github.com/en/authentication/connecting-to-github-with-ssh>

for help

* Then type “**git remote add origin <repo:ssh>**” to create a remote cache for your code. You can use “**git push -u origin master**” to update your own **master branch** into remote (github repo) **master branch**



