

# Git

`mkdir` 创建版本库（文件夹）

`pwd` 显示当前目录

`git init` 把当前目录变成git可以管理的仓库

`ls -ah` 显示当前目录中的隐藏文件

Git 添加文件需要两步：

(1) `git add` 把某文件添加到仓库

(2) `git commit -m "本次提交说明"` 把该文件提交到仓库

(可以多次add不同的文件，用一个commit来描述这次提交)

`git status` 显示仓库的当前状态

`git diff + 文件名` 查看修改的具体内容

`git log` 查看提交记录 + (`--pretty=oneline`) 精简输出信息为一行；以便确定要回退到哪个版本

`git reflog` 查看命令历史，以便确定要回到未来的哪个版本。

git 中 HEAD 表示当前版本：

`git reset --hard HEAD^` 回退到上一个版本；

(`HEAD^^`表示回退到上上个版本；`HEAD~100`回退到前100个版本)

`cat + 文件名` 显示当前文件的内容

`git reset --hard + 版本id的前几位数字` 重新回到未来的某个版本

工作区：电脑上能看到的文件目录，如 `Learn_git`

版本库：工作区的隐藏目录 `.git`，其中包括最重要的暂存区stage和自动创建的master分支

`git add` 命令实际上就是把要提交的所有修改放到暂存区（Stage），

然后，执行 `git commit` 就可以一次性把暂存区的所有修改提交到分支。

1.若修改还未添加到暂存区：

命令 `git checkout -- readme.txt` 就是，把 `readme.txt` 文件在工作区的修改全部撤销，有两种情况：

一种是 `readme.txt` 自修改后还没有被放到暂存区，现在，撤销修改就回到和版本库一模一样的状态；

一种是 `readme.txt` 添加到暂存区后，又作了修改，现在，撤销修改就回到添加到暂存区后的状态。

总之，就是让这个文件回到最近一次 `git commit` 或 `git add` 时的状态。

2.若只是添加到了暂存区还没有提交：

`git reset HEAD + 文件名` 可以把暂存区的修改撤销掉（unstage），重新放回工作区

3.若修改已经提交到了版本库：

采用版本回退命令：`git reset --hard HEAD^` 回退到上一个版本；

`rm test.txt` 在文件管理器中删除文件：

(1) 确实从版本库中删除文件：`git rm test.txt` 然后 `git commit -m`

(2) 删错了，但版本库中还有，把误删恢复：`git checkout -- test.txt`

`git checkout` 其实是用版本库里的版本替换工作区的版本，无论工作区是修改or删除，都可以还原。

添加远程库：

(1) 创建SSH key 命令：`ssh-keygen -t rsa -C "youremail@example.com"`

(2) 登陆GitHub，打开“Account settings”，“SSH Keys”页面：

(3) 点“Add SSH Key”，填上任意Title，在Key文本框里粘贴用户主目录中.ssh中 `id_rsa.pub` 的内容：

`git remote add origin git@github.com:wangjinxile/Learn_git.git` 关联一个远程库

`git push -u origin master` 把本地库上的内容推送到远程库上

`git push origin master` 之后需要提交至远程的命令

从远程库克隆：

`git clone git@github.com:xxxxx.git`

查看分支：`git branch`

创建分支：`git branch <name>`

切换分支: `git checkout <name>`

创建+切换分支: `git checkout -b <name>`

合并某分支到当前分支: `git merge <name>`

删除分支: `git branch -d <name>`

分支变化情况的直观展示: `git log --graph --pretty=oneline --abbrev-commit`

`git merge --no-ff -m "merge with no-ff" dev`禁用fast-forward分支合并形式, 删除分支后仍能看出合并

当手头工作没有完成时, `git stash` 存储工作现场,

`git stash list` 查看工作现场列表, `git stash apply stash@{0}` 指定恢复列表中的哪一个现场

恢复现场时:

一是用`git stash apply`恢复, 但是恢复后, `stash`内容并不删除, 你需要用`git stash drop`来删除;

另一种方式是用`git stash pop`, 恢复的同时把`stash`内容也删了:

`git branch -D feature-vulcan` 强行删除没有合并的分支

`git remote (-v)` 查看远程仓库(详细)信息

`git push origin master` 推送合并本地同名的`master`分支(主分支需要及时合并)

`git` 的标签是指向某个`commit ID`的指针:

(1) 切换到需要打标签的分支

(2) `git tag v1.0` 打标签 + ID号 对指定ID的分支打标签

(3) `git tag` 查看标签 -d + 标签名 删除

`git show v0.9` 显示标签信息

`$ git tag -a v0.1 -m "version 0.1 released" 3628164 -a` 跟版本号, -m跟注释说明 再跟ID

`git push origin <tagname>` 推送某标签到远程

`git push origin --tags` 推送所有为上传至远程的标签

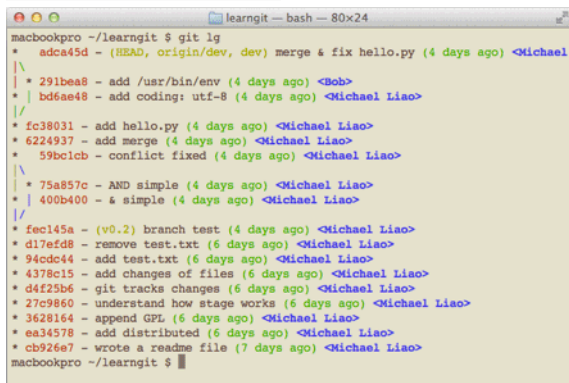
删除远程标签:

(1) 先删除本地: `git tag -d + 标签名`

(2) 再删除远程: `git push origin refs/tags/标签名`

忽略特殊文件是的查看`git status`时不会每次提醒: 在本地创建`.gitignore`文件, 把要忽略的文件名填进去

配置别名: `git config --global alias.lg "log --color --graph --pretty=format:%Cred%d%Creset -%C(yellow)%d%Creset %s %Cgreen(%cr) %C(bold blue)%C(%an)%Creset" --abbrev-commit"`



```
macbookpro ~/learnit $ git lg
* adca45d - (HEAD, origin/dev, dev) merge & fix hello.py (4 days ago) <Michael Liao>
|
| * 291bea8 - add /usr/bin/env (4 days ago) <Bob>
| * | bd6ae48 - add coding: utf-8 (4 days ago) <Michael Liao>
|/
* fc39031 - add hello.py (4 days ago) <Michael Liao>
* 6224937 - add merge (4 days ago) <Michael Liao>
* 59bcbcb - conflict fixed (4 days ago) <Michael Liao>
|
| * 75a857c - AND simple (4 days ago) <Michael Liao>
| * | 400b400 - & simple (4 days ago) <Michael Liao>
|/
* fec145a - (v0.2) branch test (4 days ago) <Michael Liao>
* d17efd8 - remove test.txt (6 days ago) <Michael Liao>
* 94cdc44 - add test.txt (6 days ago) <Michael Liao>
* 4378c15 - add changes of files (6 days ago) <Michael Liao>
* d4f25b6 - git tracks changes (6 days ago) <Michael Liao>
* 27c9860 - understand how stage works (6 days ago) <Michael Liao>
* 3628164 - append GPL (6 days ago) <Michael Liao>
* ea34578 - add distributed (6 days ago) <Michael Liao>
* cb926e7 - wrote a readme file (7 days ago) <Michael Liao>
macbookpro ~/learnit $
```