廖雪峰原教程地址:

<https://www.liaoxuefeng.com/wiki/0013739516305929606dd18361248578c67b8067c8c017b000>

Git是目前世界上最先进的分布式版本控制系统，分布式版本控制系统没有不依赖"中央服务器"，有没有无所谓，有也是方便多人协作时"交换"修改，每个人的电脑都是一个完整的版本库。Git跟踪的是修改，而非文件。

SVN是集中式版本控制系统，版本库放在中央服务器，服务器崩了就Over了。而且，版本控制系统必须联网才能使用，对带宽、速度要求高。

【安装】

----Linux----

git //查看系统有没有安装git

（1）sudo apt-get install git //在Debian或Ubuntu Linux中安装

（2）sudo apt-get install git-core //老一点的Debian或Ubuntu Linux中安装

（3）依次输入： ./config，make，sudo make install //如果是其他Linux版本，可以直接通过源码安装。Git官网下载源码，然后解压。

---Windows----

下载：<https://git-for-windows.github.io/> ;安装;打开 Git bash

【配置】

git config --global user.name "你的用户名"

//设置用户名

git config --global user.email "你的邮箱"

//设置邮箱

git config --global color.ui true

//让Git显示颜色

git config --global alias.st status

//设置别名 以后status就可以简写成st

git config --global alias.ci commit

//commit可以简写成ci

git config --global alias.unstage 'reset HEAD'

//reset HEAD 简写为unstage

eg: git ci -m "这是上传的信息"

//等价于: git commit -m "这是上传的信息"

--global 参数是全局参数，对当前用户起作用，也就是这些命令在这台电脑的所有Git仓库下都有用.如果不加，只对当前的仓库起作用。

每个仓库的配置文件都放在.git/config中，删除别名直接删除[alias]后对应的行即可。

当前用户的Git配置文件放在用户目录下(<\*1>如果改过gitbash的用户主目录，就在改过的主目录中)的隐藏文件.gitconfig中。

【命令】

[创建版本库]

mkdir demo

//创建demo文件夹

cd demo

//打开demo文件夹

pwd

//显示当前目录

git init

//将这个目录变成git可以管理的仓库-初始化一个git仓库

ls -ah

//显示当前目录下的文件 -ah可以显示隐藏的文件

[添加/推送到本地仓库]

git add test.txt

//将test.txt文件添加到仓库（要添加的文件要放在仓库的目录/子目录下)add可以反复多次使用

git commit -m "这是这次提交的介绍/说明"

//将文件提交到仓库 commit可以一次提交很多文件

[其他git命令]

git status

//查看仓库当前状态

git diff demo.txt

//查看修改内容

git log

//显示从最近到最远的提交日志,查看提交历史

git log --pretty=oneline

//显示从最近到最远的提交日志（简化版）

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

//设置了不同的颜色输出提交日志

git reflog

//可以查看所记录的你的每一条命令,查看命令历史

git diff HEAD -- demo.txt

//查看工作区和版本库里最新版本的区别

git config --global log -1

//显示最近一次的提交

git stash

//把当前工作区存储起来

git stash list

//查看存储过哪些工作现场

<1>git stash apply stash@{0}

//恢复某个工作现场，但恢复后，stash内容不删除

git stash drop stash@{0}

//删除stash内容

<2>git stash pop

//恢复工作现场并删除记录

[回退版本]

git reset --hard HEAD^

//回退到上一个版本 HEAD指向的版本就是当前版本

git reset --hard HEAD^^

//回退到上上一个版本,以此类推

git reset --hard HEAD~100

//回退到前一百个版本(数字随便写，回退到对应版本)

git resit --hard commitId(版本号)

//前进/回退到对应版本号的那个版本（版本号只需写前几个数字即可）

[删除仓库/远程仓库文件]

<1>git rm 文件

//删除本地中文件

git rm -r 文件夹

//删除文件夹

<2>git rm --cached 文件

//缓存区的对应文件被删除

git rm -r --cached 文件夹

//缓存区的对应文件夹被删除

git commit -m "提交信息"

//提交

git push

//推送到远程仓

[撤销修改]

git checkout -- demo.txt

//(只修改了工作区时)丢弃demo.txt文件在工作区的所有修改(回到了添加到暂存区git add/版本库 git commit时的状态)

<1>git reset HEAD demo.txt

//(还没commit时)把暂存区的修改撤销掉。HEAD表示最新的版本。**然后再执行上一步，丢弃工作区的修改。**

<2>git reset --hard HEAD

//(还没commit或已经commit但没提交到远程库时)直接回到上次提交的版本

[远程仓库]

git remote add origin(远程库的名字) 远程仓库项目的ssh

//在本地仓库中执行，关联本地仓库和一个远程库。如果同一个项目要推到码云和Github两个平台上，不要让远程库名字冲突

git push -u origin master

//将本地库的master的所有内容推送到远程库上(只有第一次推送需要加 -u)

git push 远程库名 远程库的某分支

//将当前推送到远程库的某个分支上

git clone 项目的SSH

//克隆一个本地库

git remote

//查看远程库的信息

git remote -v

//显示更详细的远程库信息(可以抓取和推送的远程库的地址)

[拉取远程库的最新提交]

git branch --set-upstream 分支名 远程库名/远程库分支名

//建立本地分支和远程分支的关联，指定本地分支与远程库的某分支的链接

git pull

//拉取远程库的最新提交

[关于分支]

<1>git branch 分支名

//创建分支

git checkout 分支名

//切换另一个分支

<2>git checkout -b 分支名

//创建并切换分支

git merge 分支名

//合并指定分支到当前分支

git branch -d 分支名

//删除某分支

git branch -D 分支名

//强行删除某分支（再放弃某个分支时使用）

git branch

//查看分支

合并分支时，Git会用[Fast forward]模式，但这种模式下，删除分支后，会丢掉分支信息。

git merge --no-ff -m "提交的说明" 分支名

//--no-ff 表示禁用Fast forward模式下合并某分支到当前分支

[冲突时]

git status //查看冲突的文件

手动解决冲突

再添加、提交

git log --graph //看到分支合并图

git log --graph --pretty=oneline --abbrev-commit //查看分支合并情况

[关于标签]

git tag 标签

//添加标签

git tag 标签 版本号

//给之前的提交打标签

git tag

//查看所有标签

git show 标签

//查看标签信息

git tag -a 标签 -m "说明信息" 版本号

//创建带有说明的标签

eg: git tag v1.0

git tag -s 标签 -m "说明信息" 版本号

//还可以通过-s用私钥签名一个标签。<\*3>签名采用PGP签名，必须首先安装gpg（GnuPG），若未找到gpg，或者没有gpg密钥对，就会报错。

git tag -d 标签

//删除标签

git push 远程库名 标签

//推送标签到远程库

git push 远程库名 --tags

//一次性推送所有标签到远程库

--删除远程库的标签--

git tag -d 标签

//删除本地标签

git push 远程库 :refs/tags/标签

//再从远程库删除

[Git版本控制的流程]

工作区-[add]->stage/index（暂存区）-[commit]->当前分支(master或其他)

[创建SSH]

ssh-keygen -t rsa -C "你的邮箱"

在github或码云中添加你的公钥

[忽略特殊文件]:不让某些文件上传到Git

在Git工作区的根目录下创建一个.gitignore文件

将要忽略的文件名/类型填进去，Git就会自动忽略这些文件

把文件推送到Git

git add -f 某文件

//强制添加某个被忽略的文件到暂存区

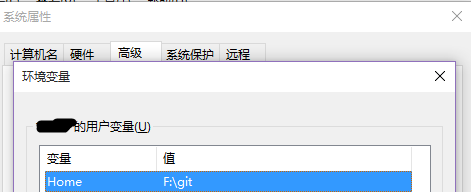
git check-ignore -v 某文件

//检查某个被忽略的文件的规则写在了.gitignore文件的哪行里

【附加】

<\*1>如何修改Git bash的用户主目录

windows安装gitbash默认的用户主目录是C:\Users\xxxx\下;在此电脑-高级系统设置-系统变量-用户变量中添加Home及你想设置的文件夹路径。将之前用户目录下已经生成的.ssh、.gnupg、.bash\_history、.gitconfig等文件，拷贝到你新设定的用户目录下。



<\*2>windows自带的记事本写的中文上传到远程仓库会乱码。

<\*3>还没研究过，研究完再补充。