创建

**ssh-keygen -t rsa -C "youremail"**//最好在主用户根目录下时输入此项指令，一路回车，另外-C要大写

之后**cat id\_rsa.pub**就会出现key，将key复制后在自己的GitHub中添加ssh（点击头像settings里有有一个ssh选项

然后直接添加就行）

**git init** //初始化版本库，没有为项目创建过版本库时，需要在目录下先创建版本库

**git add "文件名包括扩展名"** //添加文件到暂存库

**git commit -m "对文件的说明"** //提交暂存库中的文件到版本库

先添加再提交，可以添加多个文件后一起提交

**git status** //查看仓库当前状况，比如你对仓库里的一个文件只进行了修改后，Git会提醒你有一个文件被修改但是未被提交no changes added to commit (use "git add" and/or "git commit -a")

**git diff** 文件名包括扩展名 //查看你修改的文件的具体修改情况

版本回退

**git log** //显示从最近到最远的提交日志，加上**--pretty=oneline**后可以直接简单显示

前面一大串的字符是这个版本的commit id

**git reset --hard HEAD^** //HEAD表示版本，也表示当前版本，HEAD^即上一个版本，HEAD^^即上上个版本，往前100个版本则可以用HEAD~100

一旦还原当之前的版本，你再使用git log就会只显示从你还原的那个版本往前的版本

如果你想要还原到未来的版本，那么你就需要有那个版本的commit id（起码有好几位，太少也会找不到）

有的话就 **git reset --hard 00114ba** //00114ba是我的一个未来版本的commit id，各个版本的commit id 不同

Git版本的回退是内部有一个指针指向当前的HEAD，改变版本只是单纯地将指针指向你的改变版本，然后更新工作区的文件

如果没有commit id就  **git reflog** //查看操作日志

会显示

**c39a4f8 (HEAD -> master, origin/master) HEAD@{0}: reset: moving to HEAD~1**

**00114ba HEAD@{1}: commit: Second Test**

**c39a4f8 (HEAD -> master, origin/master) HEAD@{2}: commit (initial): first test**

可以看到commit id了

**git reset HEAD 文件名包括扩展名** //撤销掉暂存区中对文件的修改

忽略

**.gitignore**中可以放置不需要Git管理的文件（没有，需要自行创建）

**touch .gitignore**

**vi .gitignore** 之后在里面加入你想要忽略的文件，比如test1.txt

那么在工作区中的test1.txt文件将会被忽略，做任何的修改也不会被Git管理到

**echo "内容" > TEST.txt** //回写，**>**重定向

上传文件到GitHub

配置用户名和邮箱

**git config –global user.name “youruserid”**

**git config –global user.email “youremail”**

**创建公钥**

**ssh-keygen –t rsa –C “youremail”**

**cat ~/.ssh/id\_rsa.pub 复制密钥到GitHub中**

**克隆远程仓库**

**git clone “你的项目地址”**

**上传**

**git add 文件名.后缀**

**git commit –m “描述”**

**git push //上传**

一些错误

1. git push 时报错

git@github.com: Permission denied (publickey).

fatal: Could not read from remote repository.

Please make sure you have the correct access rights

and the repository exists.

[git@github.com](mailto:git@github.com):权限被拒绝

致命：无法从远处存储库读取

请确保你有正确的访问权限，并且存储库是存在的

**这种错误一般是缺少ssh key，需要创建key后放到GitHub中**