Linux上的拷贝快捷键：ctrl+insert

粘贴快捷键：shift+insert

怎么使用git：右键

Git bash ——git命令行窗口界面

Git GUI——git图形化窗口界面

使用git之前，应配置用户名与邮箱。

配置用户名与邮箱：git config --global user.name “your\_name”

git config --global user.eamil your\_email.163.com

查看所有配置： git config --list

Git文件的三种状态：

1. 已提交（committed）:已提交表示数据已安全保存在本地数据库中
2. 已修改(modified): 已修改表示修改了文件，但还没保存到数据库中
3. 已暂存(staged):已暂存表示对一个已修改文件的当前版本做了标记，使之包含在下次提交的快照中。

对应的有三个工作区域：

1. 工作区：简单的理解为电脑里能看到的目录，如自己创建的本地项目目录
2. 暂存区：Git的版本库里存了很多东西，其中最重要的是称为stage（或者叫index）的暂存区，还有GIT自动创建的第一个分支master，一级指向master的一个指针叫HEAD
3. Git仓库：工作区有一个隐藏目录.git，这个不算工作区，而是Git的版本库

初始化git本地仓库：

Git init 创建一个空仓库

将文件放到暂存区：

Git add

查看暂存区状态：

Git status

提交文件：

Git commit -m

查看提交日志： git log

文件修改与提交：

修改当前目录下所有文件：

Git add . .表示当前路径。或者git add 绝对路径（从C盘开始写）

查看本地文件与 本地仓库中的文件的区别：

Git diff HEAD – git01.txt

暂存区文件撤销：

Git restore –staged git02.txt 可选择取消撤销的文件

或者 git reset HEAD git02.txt 取消上一次的操作。如果上一步操作为git commit ，那么此操作即为取消上传

直接提交当前目录下所有修改的文件：

Git commit -am ‘第四次提交’ a即为add的缩写，此命令将目录下所有文件直接上传至本地仓库

版本回退/版本前进：

Git log -–pretty=oneline 一行显示一条log记录

回退版本：

Git reset -–hard HEAD^ 一个^表示回退一个版本，两个^表示两个版本，同理

Git reset --hard HEAD~2 ~后面的数字表示在当前版本的基础上回退几个版本

Gti reset --hard b10679e2 跳转至指定版本，版本id不用全部抄完，抄个8位以上即可

如果退回到之前版本，关了git bash窗口，再打开时查看log，将不会看到该版本以后的log。此时可通过 git reflog 查看所有log日志，然后通过唯一标识跳转到指定版本。

删除文件：

方法1：在当前文件夹中删除，然后在git add ，git commit。 新增，修改，删除在git中均视为一个更新，所以操作与新增一样。

方法2： git rm git02.txt 删除本地仓库中文件目录同时也会把当前文件夹中文件删除。

Git ls-files 查看本地仓库中的文件目录

Git checkout git02.txt 拉新的版本到当前文件夹

Github：

Github与码云一般都是存放开源代码，gitlab一般为企业在服务器上自行搭建的远程仓库

下载项目的方式：

1.直接下载为压缩包；

2.git bash中输入git clone +下载地址

3.使用SSH的方式下载。优点：更安全，传输效率更高。

生成public key：ssh-keygen -t rsa -C [757289860@qq.com](mailto:757289860@qq.com)

然后在指定目录找到key。 主要要打开.pub的文件，复制粘贴。因为这个是公钥，另一个是私钥。然后按照网页指示添加。

最后，检查测试链接，执行命令：

通过SSH下载，命令与上述一样，即：git clone +SSH地址

远程仓库推送

创建空白的仓库：在github头像旁+内选择create repository

如何连接远程：

图形用户界面, 文本, 应用程序, 电子邮件

描述已自动生成

git branch -M master 切换到master目录

git remote add origin <https://github.com/princewu/test.git> 添加远程仓库

git push -u origin master 推送

后续修改后提交到远程仓库，直接先add commit提交到本地仓库，然后直接git push即可传到远程仓库，不用重新绑定远程仓库

Git分支操作：（在本地仓库中的操作）

Git checkout branch 切换分支

Git checkout -b new\_branch 新建分支并切换到新建分支。新建分支时，此时主干的内容即为分支的内容。

Git branch -d branch 删除指定分支

Gitbranch -D branch 强制删除分支

Git branch 查看所有分支，并且\*为当前所在分支

Git merge branch 合并分支。必须先切换到主干上再去合并分支，千万不能再分支上合并主干

Git branch -m | -M oldbranch newbranch 重命名分支，-m为正常的重命名操作，如果newbranch名已经存在 则需要用-M强制重命名

远程分支操作：

Git branch -a 查看本地与远程的分支

Git push origin branch\_name 推送本地分支到远程

Git push origin :remote\_branch 删除远程分支（本地分支保留）。删除与上一行推送仅仅差一个：。

Git fetch 获取远程仓库最新状态

Git checkout -b local\_branch origin/remote\_branch 拉取远程指定分支并在本地创建分支。首先应先获取远程仓库最新状态，然后在git branch -a中查看到远程仓库分支是否已经更新，然后再输入上述命令拉下来

分支冲突解决：

如何解决：随机应变，看实际工作需求。一般合并时如有分支冲突会提示冲突文件，同时在文件内冲突的地方会有标志。解决好冲突问题后，需重新add commit，此时才为分支merge完成。

冲突是如何产生的？同一个文件同一行的内容不同

多人协同开发操作冲突：

无论本地分支与多人协同，出现冲突的条件都是一样的，即同一个文件，同一行出现差异。

出现冲突潜在的解决方式：谁发现冲突谁去解决。

如何避免：推送之前先拉取最新代码，解决完冲突后再去提交。 Git pull

标签管理（即为版本管理）

Git tag 查看本地所有标签

Git tag tag\_name 创建标签，默认是按照最后一次提交记录来的

对制定版本打标签 git tag tag\_name +id 通过git log –pretty=oneline 查看版本id，然后在tag 命令后面写上id表示对指定版本打标签。Eg： git tag v0.9 22fdsfds32

或者： git tag -a v0.99 -m ‘先行版’ 3f33ces 这种也是对某个版本打标签，不过他加了标签注释

将一个标签推到远程： git push origin v1.0 将标签V1.0推送到远程

将标签推送到远程后，可以在右侧的release中查看标签信息，点进去后，可以通过压缩包查看代码。

将所有（未被推过的）标签推到远程： git push origin --tags

删除远程标签： git push origin:refs/tags/v1.0 仅删除远程标签，本地仍在。

删除本地标签： git tag -d v1.0

Idea集成git，如何从git上拉项目

如何从idea向远程仓库推项目

新项目：选择 VCS- create git repository 新建本地仓库

左侧项目菜单上，右键git-add 。 选择一个文件即上传单个文件，选择文件夹即文件夹整体上传

然后在同样的位置右键git-commit，推送到本地仓库。注意推送文件不用全选，然后记得写注释

然后在github上创建仓库（repository），复制ssh

然后右键git-manage remote 粘贴ssh，即与github连接

然后右键git-push，注意切换分支即可

Idea下分支操作

新建分支：1.命令行；2. 右下角图形化界面操作

Idea提交时，可以直接commit，因为idea会自动add修改后的文件到暂存区

Fetch与pull区别：fetch只会更新状态，pull把代码拉下来

\_Gitgnore插件使用：

上传项目时，.idea与.mi文件不用传到github上

推荐插件.ignore，可以在git上传时自动忽略文件