**基于Git bush的本地仓库和GitHub远程库管理的学习笔记**

1. 安装git
2. 了解git分布式版本控制的概念，理解git的三个工作区域和文件的三种状态

//<https://www.bilibili.com/video/BV1XP4y147v1?p=4&vd_source=7b9a6a0efb76da176ba8bd43c2559556>

1. 配置用户信息

git config --global user.email “you@example.com”

git config --global user.name “Your Name”

git config --global init.defaultbranch main //将之后新建仓库默认分支改为main（用户级）

//[git config配置 - fireporsche - 博客园 (cnblogs.com)](https://www.cnblogs.com/fireporsche/p/9359130.html)

1. Win10环境下在工作区右键git bush here将git bush的目录定位到工作区

Win11可以用以下命令完成

cd C:/Users/Lee/… //定位到目标地址

cd ~ //定位到用户根目录

cd .. //返回上一级目录

1. 新建远程仓库
2. github新建仓库
3. 获得密钥

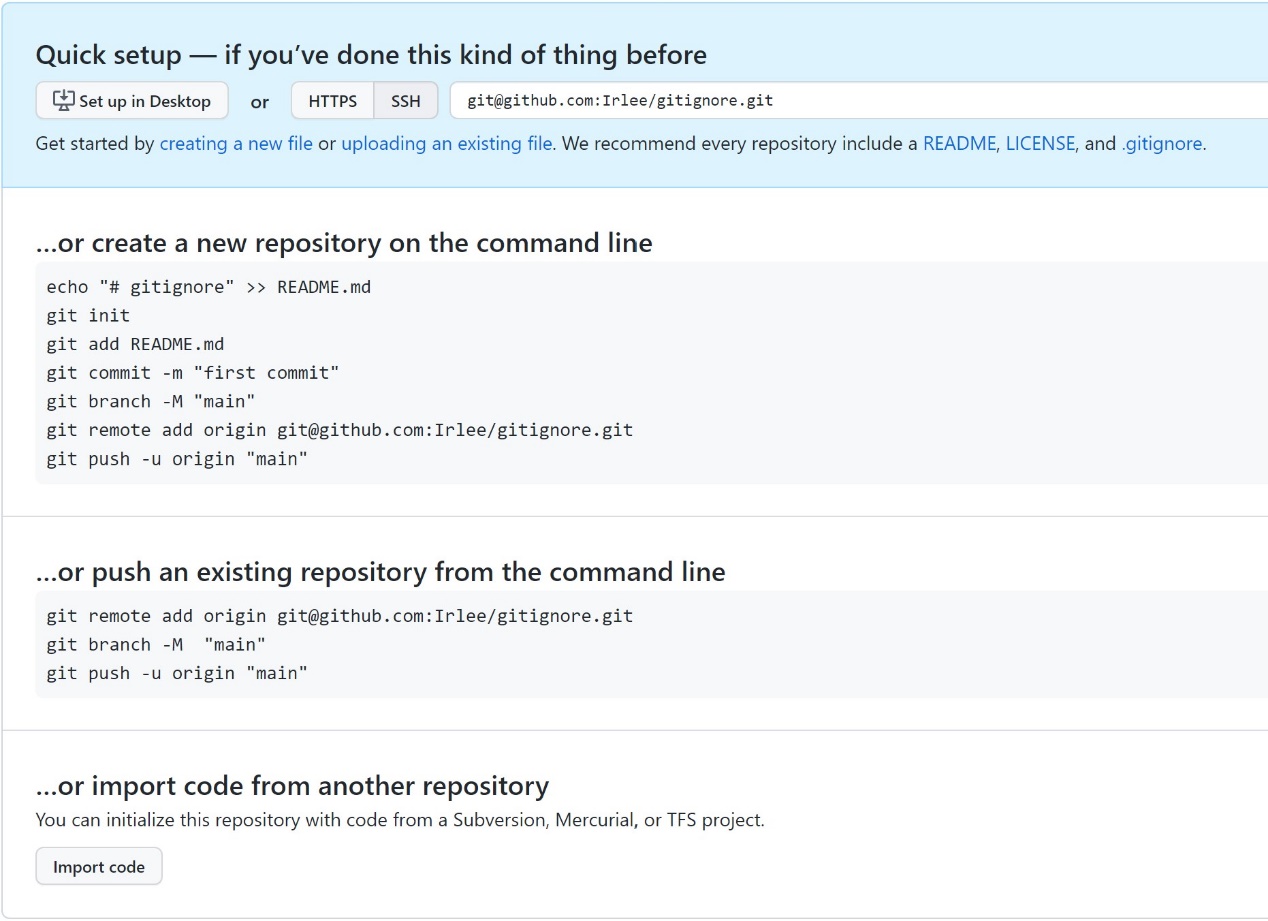
ssh-keygen -t rsa -C " you@example.com " //生成ssh key

ssh-keygen -t rsa -b 4096 -C " you@example.com " //生成4096位的ssh key

cat ~/.ssh/id\_rsa.pub //打印公钥

1. 在github关联公钥
2. 本地验证是否关联成功

ssh -T git@github.com //检查验证是否成功



1. 创建本地仓库并提交

git init //在git bush显示的目录下新建 Git 仓库

git status //查看仓库当前的状态，显示有变更的文件。

git add <file/path> //添加文件到暂存区

git restore <file> //discard workspace changes, used before add, NoRevert

git restore --staged <file> //discard staged changes, used after add, can be undone

git commit -m ‘备注’ //将暂存区内容添加到仓库中

git commit -a //修改文件后不需要执行 git add 命令，直接来提交

//该命令后不能加备注,回车后添加本次commit备注

//添加完成后输入:wq返回git bush

git commit -am ‘备注’ //无需add, 添加注释直接提交，相当于合并add和commit

git commit -amend //修复旧提交，实质相当于完全用一个新的提交替换旧的提交

git diff //显示已写入暂存区和已经被修改但尚未写入暂存区文件的区别

git diff --staged //显示暂存区和上一次提交(commit)的差异

git diff HEAD //显示工作版本(Working tree)和HEAD的差别

git rm <file> //将文件从暂存区和工作区中删除，工作区文件无法恢复

git rm -f <file> //删除之前修改过或已经放到暂存区的文件，则必须使用

//强制删除选项 -f

git rm --cached <file> //将文件从 Git 仓库中删除（亦即从暂存区域移除）

//但仍然希望保留在当前工作目录中。

clear //清屏

reset //这个命令将重新启动git bush，但整个命令过程速度有点慢

git log //查看历史提交记录

git log [-num] --pretty=oneline //查看之前num次历史提交记录，每次提交显示一行

git log [-num] --oneline //同上句

git log --graph --pretty=online //清晰显示分支操作的历史提交记录

git reflog //显示可引用的历史版本记录

git shortlog //只显示提交备注

1. 版本回退

git reset [--soft | --mixed | --hard] [HEAD]

git reset --soft HEAD~3 //回退上上上一个版本，只改变HEAD指向

git reset --hard HEAD~3 //回退上上上一个版本，HEAD指向，Index，working tree

git reset --hard bae128f //回退到某个版本回退点之前的所有信息。

git reset --hard origin/main //将本地的状态回退到和远程的一样

--hard 标记是 reset 命令唯一的危险用法，它也是 Git 会真正地销毁数据的仅有的几个操作之一。其他任何形式的 reset 调用都可以轻松撤消，但是 --hard 选项不能，因为它强制覆盖了工作目录中的文件。在这种特殊情况下，我们的 Git 数据库中的一个提交内还留有该文件的 v3 版本， 我们可以通过reflog 来找回它。但是若该文件还未提交，Git 仍会覆盖它从而导致无法恢复。

1. 分支操作

git checkout branch1 //切换到分支branch1

git checkout -b branch2 //新建并切换到新分支branch2

git branch //查看当前仓库所有的分支，\*标记当前分支

git branch -a //查看本地和远程仓库的所有分支

git branch -d branch1 //删除分支branch1

git merge branch1 //将分支branch1合并到主分支main

git branch -m | -M oldbranch newbranch //重命名分支

1. 远程仓库操作

git clone [url] //克隆远程仓库到本地

git remote add origin [url] //添加远程版本库

git push origin branch //将本地分支branch推送到远程仓库

git push origin --delete branch //删除远程origin主机仓库的branch分支

git push origin :branch //等同于上一句，慎用！

git checkout -b <本地分支名> <远程分支名>/<远程分支名>

//拉取远程分支并在本地创建分支

git fetch //将某个远程主机的更新，全部取回本地

git fetch origin //从远程refs/heads/命名空间复制所有分支，并将存储到

//本地的refs/remotes/origin/命名空间中

git merge origin/master //在本地分支上合并远程分支

git rebase origin/master //同上一句

git remote -v //显示所有远程仓库

git remote show [remote] //显示某个远程仓库的信息

git remote add [shortname] [url] //添加远程版本库

git remote rm name //删除远程仓库

git remote rename old\_name new\_name //修改远程仓库名

git pull <远程主机名> <远程分支名>:<本地分支名>

git pull  //命令用于从远程获取代码并合并本地的版本。

//相当于git fetch 和 git merge FETCH\_HEAD 的简写。

1. 文件删除、写入、查看、查找、重命名以及用户配置

rm <file.后缀> //删除工作区文件，文件无法找回

rm -r <file> //删除文件夹及其子文件,恢复方法未知，慎用！

rm -rf <path> //有时候-r执行后无法删除，-f强制，-rf一定可以！

echo ‘text’ > <file> //将内容text写入file，如果没有则新建

echo ‘text’ >> <file> //在file中添加内容，上一句会覆盖原内容

echo “text” >> <file> //text中可包含单引号

vim <file> //可以在vim编辑器里直接编辑

window系统下可以将文本编辑器设置为vscode或者notepad++，并且通过其进行diff和merge

ls //显示路径下的文件和目录列表

cat <file.后缀> //打印该文件内容

find //打印当前directory下所有文件名

find \*.txt //查找后缀为.txt的所有文件

find -name \*关键字\* //递归查找包含“关键字”的文件名

mv <file1.后缀> <file2.后缀> //将工作区file1重命名为file2

//但git记录file2被删除，file2 untracked

git mv -v <file1.后缀> <file2.后缀> //只能修改已经追踪的文件和文件夹

//修改之后,相当于执行了add,可直接 commit

git config --system --list //查询系统级配置，修改需要管理员权限，一般只需修改global

git config --global --list //查询用户级配置

git config --local --list //查询仓库级配置

git config --local -e //编辑仓库级别配置文件

git config --global -e //编辑用户级别配置文件

git config --global user.email “you@example.com”

git config --global user.name “Your Name”

git config --global init.defaultbranch main //将之后新建仓库默认分支改为main

//与github同步，作用域当前系统账户

1. 更改远程default branch

从main改为master，需要在本地进行下面的操作

git branch -m main master

git fetch origin

git branch -u origin/master master

git remote set-head origin -a

1. 标签管理

git tag //查看所有标签

git tag name //新建标签

git tag -a name -m ‘xxx’ //添加标签，并指定标签描述信息

git tag -d name //删除一个本地标签

git push origin name //推送tag到远程库

git push origin --tags //推送全部未推送的标签到远程

git push origin :refs/tag/name //删除一个远程标签

1. 忽略文件
2. 在working tree新建.gitignore文件

echo “# those files modified will not be tracked” >> .gitignore

1. 忽略规则如下

• Blank lines or lines starting with # are ignored.

• You can negate a pattern by starting it with an exclamation point (!).

1. 一个.gitignore例子

# ignore all .a files

\*.a

# but do track lib.a, even though you're ignoring .a files above

!lib.a

# only ignore the TODO file in the current directory, not subdir/TODO

/TODO

# ignore all files in any directory named build

build/

# ignore doc/notes.txt, but not doc/server/arch.txt

doc/\*.txt

# ignore all .pdf files in the doc/ directory and any of its subdirectories

doc/\*\*/\*.pdf

【附录】linux下快捷键

1. 在终端下：

          复制命令：Ctrl + Shift + C  组合键

          粘贴命令：Ctrl + Shift + V  组合键

1. 在控制台下：

          复制命令：Ctrl + [Insert](https://so.csdn.net/so/search?q=Insert&spm=1001.2101.3001.7020)  组合键

          粘贴命令：Shift + Insert  组合键

Ctrl+a 光标移动到开始位置   
Ctrl+e 光标移动到最末尾   
Ctrl+k 删除此处至末尾的所有内容   
Ctrl+u 删除此处至开始的所有内容   
Ctrl+d 删除光标后的一个字符   
Ctrl+h 删除光标前一个字符   
Ctrl+w 删除此处到左边的单词   
Ctrl+y 粘贴由 Ctrl+u ， Ctrl+d ， Ctrl+w 删除的单词   
Ctrl+l 翻页   
Ctrl+r 查找历史命令   
Ctrl+b 向回移动光标   
Ctrl+f 向前移动光标

CTRL+C 终止当前操作

1. Vim编辑器下：

命令行

i a o进入编辑模式

q 退出编辑器

qa! 强制退出

w 保存

wq 保存并退出

e!放弃修改，回到文档打开的状态

u 撤销

cc 删除并进入编辑模式

x 删除字符

dw 删除单词

dd 删除整行包括回车，停留在命令模式

y 复制

yy 复制整行

p 粘贴