git clone repo //第一次拉代码

git init //在空目录先创建空仓库，添加remote后给git pull

git reset --hard HEAD^//清除修改，完全回退到上一个提交

git reset --soft HEAD^//清除修改，保留add之后，commit之前。用于重置暂存区的文件与上一次的提交(commit)保持一致，工作区文件内容保持不变。//把上一次的提交放到或者回退到暂存区

git pull --rebase //在本地的已有的仓库中更新代码，使用rebase不会额外生成一个commit

git pull //在本地的已有的仓库中更新代码，如有合并则额外生成一个commit

git log

git branch//查看本地分支

git branch -m demo1 demo2//修改分支名

git checkout//切换分支

git checkout demo //创建新分支

git checkout -b demo //创建并切换到新分支

git checkout -b demo remote/orign/分支名 //创建并切换到新分支并关联到远程版本仓库上的分支

git tag -d <tag name> //删除本地tag

本地操作的前提得有一个分支

git checkout <tag name>

git checkout -b <tag name>

git commit --amend //把改动追加到上一次的提交

git diff //本地与暂存区的差别

git diff --cache //暂存区与本地git仓库最新提交之间的差异

git rm //从暂存区中删除文件

git status //查看工作区的状态

推送到reviewer

git push -u origin main:refs/for/main

git push -u 远程版本控制仓库地址的别名 本地分支名:refs/for/远程分支名

https://blog.csdn.net/zebghfv/article/details/120782767?spm=1001.2101.3001.6661.1&utm\_medium=distribute.pc\_relevant\_t0.none-task-blog-2%7Edefault%7ECTRLIST%7Edefault-1-120782767-blog-110457931.pc\_relevant\_multi\_platform\_featuressortv2dupreplace&depth\_1-utm\_source=distribute.pc\_relevant\_t0.none-task-blog-2%7Edefault%7ECTRLIST%7Edefault-1-120782767-blog-110457931.pc\_relevant\_multi\_platform\_featuressortv2dupreplace&utm\_relevant\_index=1

ssh -keygen -t rsa -c "username"

~/.ssh目录

拷贝id\_rsa.pub文件内容

在gitlab中配置pubkey

<https://xiaomi.f.mioffice.cn/drive/folder/fldk4pHZJ5suLGewMM3VCtFs0Hh>

拉取代码的一般步骤：

创建目录，命令和结果如图所示：$ mkdir vld$ cd vld初始化repo：$ repo init -u ssh://gerrit-mirror.x-ringtek1.com:29418/vldprojects/manifests -b dev同步代码：$ repo sync注意：由于整个代码包含编译工具链，同步的时间可能比较长。同步完成后，代码目录结构：

编译：

4.4.1 编译代码拉下来后，进入testplat的目录：cd testplat其中，build.sh就是Linux平台编译的脚本，通过build.sh编译所需要项目的的测试case。编译测试工程的命令和Windows差不多：./build.sh [project\_path\_cfg] [module\_name]例如，编译m1shuttle ap的uart测试case：./build.sh projects/m1shuttle/ap uart最后生成物在vld/out目录下。