https://github.com/githubken0426/DemoPro.git

**0 、执行git初始化仓库**

git init

2.配置git远程仓库的连接

git remote add origin https://github.com/account\_name/test.git

3.拉取远程分支代码

git pull origin master(如果是初次pull,需要输入git账号，密码)

3.1 设置Git全局账号密码

git config --global user.name "用户名"

git config --global user.email "用户邮箱"

4.添加文件目录到git仓库

git add .(小数点是添加当前目录下的所有文件)

5.上传文件到git仓库

git commit -m "注释内容"

6.推送到分支

git push -u origin master

git status

cat .git/HEAD

设置用户名和邮箱

git config --global user.name “你的用户名”

git config --global user.email “你的邮箱”

git config --list

**一、Git 创建仓库**

1.初始化一个Git仓库\

git init

git init newrepo

2.配置git远程仓库的连接

git remote add origin https://github.com/account\_name/test.git

3.添加文件目录到git仓库

git add .(小数点是添加当前目录下的所有文件)

4.推送到远程分支

git push -u origin master（The -u tells Git to remember the parameters, so that next time we can simply run git push and Git will know what to do.）

**二、git clone**

git clone <repo>

git clone <repo> <directory> //克隆到指定的目录

git clone -b <branch> <repo> //克隆远程某个分支

http://git.newit.club/acts2/acts-sample-mobile-2.0.git

http://git.newit.club/aduck-ta/hybriddevtrain.git

**三、git log**

git log

git log --pretty=oneline --abbrev-commit //历史提交的commit

git log --graph//查看分支合并图

git reflog //查看命令历史

git status // 查看冲突提交状态

1、git log -- filename(git log filename) 可以看到该文件相关的commit记录

2、git log -p filename 可以显示该文件每次提交的diff

3、git show comit\_id filename

可以查看某次提交中的某个文件变化

4、git show commit\_id

查看某次提交

5、gitk --follow filename

以图形化界面的方式显示修改列表

**四、git tag**

git tag //默认为HEAD

git tag <tagname> <commitid> //根据commit id标记tag

git tag -a <tagname> -m <标签备注>

git show <tagname> //查看标签信息

标签和commit挂钩。如果这个commit既出现在master分支，又出现在dev分支，那么在这两个分支上都可以看到这个标签。

git push origin <tagname> //推送一个本地标签；

git push origin --tags //推送全部未推送过的本地标签；

git tag -d <tagname> //删除一个本地标签；

git push origin :refs/tags/<tagname> //删除一个远程标签。

**五、git branch**

git branch //查看分支

git branch <name> //创建分支

git checkout <name>或者git switch <name> //切换分支

git checkout -b <name>或者git switch -c <name> //创建+切换分支

git branch -d <name> //删除分支

**六、git conflict**

//Creating a new branch is quick & simple.

**七、Git commit**

git commit -m “message”

git commit -a -m “massage”

-a参数可以将所有已跟踪文件中的修改、删除操作都提交到本地仓库，即使它们没有经过git add添加到暂存区。

注意，新加的文件（即没有被git系统管理的文件）是不能被提交到本地仓库的。

建议一般不要使用-a参数，正常的提交还是使用git add先将要改动的文件添加到暂存区，再用git commit 提交到本地版本库。

**八、git merge**

git merge <name> //合并某分支到当前分支，默认是Fast forward模式，这种模式下，删除分支后，会丢掉分支信息。

git merge --no-ff -m "<comments>" <name>// 禁用Fast forward模式，Git会在merge时生成一个新的commit，这样，从分支历史上就可以看出分支信息

merge master时候，会提示Please enter a commit message to explain why this merge is necessary.

1.按键盘字母 i 进入insert模式

2.修改最上面那行黄色合并信息,可以不修改

3.按键盘左上角"Esc"

4.输入":wq",回车键即可

**九、git pull**

git pull <remote branch>:<local branch>

Pull:先拉取远程分支代码，然后和本地代码merge，会产生一个merge commit。

Pull(Rebase): 不会产生merge commit

**十、git push**

git push origin <local branch>:<remote branch> // 提交本地test分支作为远程的master分支,远程的会自动创建一个test分支,:前后不能有空格

git push origin <local branch>:<remote branch> // 提交本地test分支作为远程的test分支

git push origin :<remote branch> // 刚提交到远程的test将被删除，但是本地还会保存的，不用担心

**十一、git stash**

git stash //把当前工作现场“储藏”起来，等以后恢复现场后继续工作

git stash list

git stash apply //恢复stash,不会删除stash

git stash drop //删除stash

git stash pop //恢复的同时也删除stash

**十二、git fetch**

git fetch 命令：

$ git fetch <远程主机名> //这个命令将某个远程主机的更新全部取回本地

如果只想取回特定分支的更新，可以指定分支名：

$ git fetch <远程主机名> <分支名> //注意之间有空格

最常见的命令如取回origin 主机的master 分支：

$ git fetch origin master

取回更新后，会返回一个FETCH\_HEAD ，指的是某个branch在服务器上的最新状态，我们可以在本地通过它查看刚取回的更新信息：

$ git log -p FETCH\_HEAD

一、未git add 缓存代码,git checkout -- filepathname放弃所有的文件修改可以使用 git checkout .

二、已经 git add 缓存了代码,git reset HEAD filepathname （比如： git reset HEAD readme.md）,放弃所以的缓存, git reset HEAD^,

三、已经git commit,git reset --hard HEAD^ 来回退到上一次commit。回退到任意版本：git reset --hard commitid

git remote add origin git@gitee.com:longdcheng/cheng\_pro.git

将代码推送到远程仓库

git push origin master

# 注意，如果出现，推送失败可能就需要运行强推(因为此时代码可能不是基于仓库中代码进行的修改)

git push -f origin master