git init # 初始化本地git仓库（创建新仓库）

git config --global user.name "xxx" # 配置用户名

git config --global user.email "xxx@xxx.com" # 配置邮件

git config --global color.ui true # git status等命令自动着色

git config --global color.status auto

git config --global color.diff auto

git config --global color.branch auto

git config --global color.interactive auto

git config --global --unset http.proxy # remove proxy configuration on git

git clone git+ssh://git@192.168.53.168/VT.git # clone远程仓库

git status # 查看当前版本状态（是否修改）

git add xyz # 添加xyz文件至index

git add . # 增加当前子目录下所有更改过的文件至index

git commit -m 'xxx' # 提交

git commit --amend -m 'xxx' # 合并上一次提交（用于反复修改）

git commit -am 'xxx' # 将add和commit合为一步

git rm xxx # 删除index中的文件

git rm -r \* # 递归删除

git log # 显示提交日志

git log -1 # 显示1行日志 -n为n行

git log -5

git log --stat # 显示提交日志及相关变动文件

git log -p -m

git show dfb02e6e4f2f7b573337763e5c0013802e392818 # 显示某个提交的详细内容

git show dfb02 # 可只用commitid的前几位

git show HEAD # 显示HEAD提交日志

git show HEAD^ # 显示HEAD的父（上一个版本）的提交日志 ^^为上两个版本 ^5为上5个版本

git tag # 显示已存在的tag

git tag -a v2.0 -m 'xxx' # 增加v2.0的tag

git show v2.0 # 显示v2.0的日志及详细内容

git log v2.0 # 显示v2.0的日志

git diff # 显示所有未添加至index的变更

git diff --cached # 显示所有已添加index但还未commit的变更

git diff HEAD^ # 比较与上一个版本的差异

git diff HEAD -- ./lib # 比较与HEAD版本lib目录的差异

git diff origin/master..master # 比较远程分支master上有本地分支master上没有的

git diff origin/master..master --stat # 只显示差异的文件，不显示具体内容

git remote add origin git+ssh://git@192.168.53.168/VT.git # 增加远程定义（用于push/pull/fetch）

git branch # 显示本地分支

git branch --contains 50089 # 显示包含提交50089的分支

git branch -a # 显示所有分支

git branch -r # 显示所有原创分支

git branch --merged # 显示所有已合并到当前分支的分支

git branch --no-merged # 显示所有未合并到当前分支的分支

git branch -m master master\_copy # 本地分支改名

git checkout -b master\_copy # 从当前分支创建新分支master\_copy并检出

git checkout -b master master\_copy # 上面的完整版

git checkout features/performance # 检出已存在的features/performance分支

git checkout --track hotfixes/BJVEP933 # 检出远程分支hotfixes/BJVEP933并创建本地跟踪分支

git checkout v2.0 # 检出版本v2.0

git checkout -b devel origin/develop # 从远程分支develop创建新本地分支devel并检出

git checkout -- README # 检出head版本的README文件（可用于修改错误回退）

git merge origin/master # 合并远程master分支至当前分支

git cherry-pick ff44785404a8e # 合并提交ff44785404a8e的修改

git push origin master # 将当前分支push到远程master分支

git push origin :hotfixes/BJVEP933 # 删除远程仓库的hotfixes/BJVEP933分支

git push --tags # 把所有tag推送到远程仓库

git fetch # 获取所有远程分支（不更新本地分支，另需merge）

git fetch --prune # 获取所有原创分支并清除服务器上已删掉的分支

git pull origin master # 获取远程分支master并merge到当前分支

git mv README README2 # 重命名文件README为README2

git reset --hard HEAD # 将当前版本重置为HEAD（通常用于merge失败回退）

git rebase

git branch -d hotfixes/BJVEP933 # 删除分支hotfixes/BJVEP933（本分支修改已合并到其他分支）

git branch -D hotfixes/BJVEP933 # 强制删除分支hotfixes/BJVEP933

git ls-files # 列出git index包含的文件

git show-branch # 图示当前分支历史

git show-branch --all # 图示所有分支历史

git whatchanged # 显示提交历史对应的文件修改

git revert dfb02e6e4f2f7b573337763e5c0013802e392818 # 撤销提交dfb02e6e4f2f7b573337763e5c0013802e392818

git ls-tree HEAD # 内部命令：显示某个git对象

git rev-parse v2.0 # 内部命令：显示某个ref对于的SHA1 HASH

git reflog # 显示所有提交，包括孤立节点

git show HEAD@{5}

git show master@{yesterday} # 显示master分支昨天的状态

git log --pretty=format:'%h %s' --graph # 图示提交日志

git show HEAD~3

git show -s --pretty=raw 2be7fcb476

git stash # 暂存当前修改，将所有至为HEAD状态

git stash list # 查看所有暂存

git stash show -p stash@{0} # 参考第一次暂存

git stash apply stash@{0} # 应用第一次暂存

git grep "delete from" # 文件中搜索文本“delete from”

git grep -e '#define' --and -e SORT\_DIRENT

git gc

git fsck