# Git 常用命令

* 1、基础git config --global user.name "Your Name"设置你的仓库用户名（用于标识提交者）git config --global user.email "email@example.com"设置你的仓库邮箱（用于标识提交者）git init 初始化一个git仓库git add --all 添加所有更改的文件git add filename1 当然可以指定添加filename1git commit -m "commit message" 添加更改的信息，必须要有，不然报错，不建议不加。git status 查看git当前状态git diff filename1 查看filename1到底修改了哪些内容git log 查看最近的提交日志git log --pretty=oneline 单行显示提交日志git reset --hard commitID 利用git log得到的commitID返回版本git reset --hard HEAD^回到上一个版本git reflog 查看命令的历史，可以找到git log看不到的commitID，因为git log只显示当前的提交日志，如果你提交了一次，退回版本后又后悔了，就能查看上次提交的commitIDgit checkout -- filename1 利用版本库中的版本替换工作区中的文件。功能有2：撤销文件修改，分两种情况：撤销工作区中的修改（没有使用git add命令添加到暂存区）撤销暂存区中的修改（添加到了暂存区又做了修改）找回删除的文件工作区中文件误删了，可以通过此命令从版本库中找回git reset HEAD filename1 撤销add，回到工作区git rm filename1 删除文件git remote add origin https://github.com/pengloo53/learngit.git 将本地库关联到github远程库上git push -u origin master 第一次推送的时候要加上-u参数，可以将本地库的master分支与远程库的master分支关联起来；下次提交就不需要加-u了。git clone https://github.com/pengloo53/learngit.git 克隆远程库到本地2、分支管理git checkout -b dev创建dev分支并切换到dev。相当于git branch dev、git checkout dev两条命令。git branch查看当前分支git merge dev合并指定分支到当前分支，如，你现在master分支，那么执行命令就将dev分支合并到了master分支上。git branch -d dev删除dev分支git log --graph --pretty=oneline --abbrev-commit查看分支合并图git merge --no-ff -m "merge with no-ff" dev禁用「Fast forward」，也就是保留分支的相关信息。git stash 将工作区现场储藏起来，等以后恢复后继续工作。通常用于处理更为着急的任务时，例如：bug。git stash list 查看保存的工作现场git stash apply恢复工作现场git stash drop 删除stash内容git stash pop 恢复的同时直接删除stash内容git stash apply stash@{0} 恢复指定的工作现场，当你保存了不只一份工作现场时。git branch -D feature-vulcan 强行删除分支。用于不需要合并，就地删除的情况。git remote 查看远程库的信息，一般返回origin。git remote -v 查看远程库的详细信息。git push origin master 将本地master分支推送到远程master分支。master分支为主分支，因此要时刻与远程同步；dev分支为开发分支，团队成员都需要在上面工作，所以也需要与远程同步；bug分支只用于在本地修复bug，没有必要推送到远程；feature新功能分支是否推送到远程，取决于你是否和其他人合作在上面开发。git clone https://github.com/pengloo53/learngit.git 将远程库克隆到本地，默认只能看到master分支。git checkout -b dev origin/dev 创建远程dev分支到本地git pull 将远程分支的最新内容抓取下来。git branch --set-upstream dev origin/dev将本地dev分支与远程dev分支之间建立链接。多人协作工作模式首先，可以试图用git push origin branch-name推送自己的修改；如果推送失败，则因为远程分支比你的本地更新，需要先用git pull试图合并（如果git pull提示“no tracking information”，则说明本地分支和远程分支的链接关系没有创建，用命令git branch --set-upstream branch-name origin/branch-name）;如果合并有冲突，则解决冲突，并在本地提交；没有冲突或者解决掉冲突后，再用git push origin branch-name推送就能成功！3、标签管理git tag v1.0 给当前分支打上标签git tag 查看所有的标签，按时间顺序列出。git log --pretty=oneline --abbrev-commit缩略commitID并单行显示提交信息git tag v0.9 commitID通过上一条命令查看commitID，然后打上标签。用于忘记打标签的情况，因为标签其实就是只想某个commitID的指针，默认情况下，标签打在最新的提交上。git show v0.9 查看标签信息。git tag -a v0.1 -m "version 0.1 released" commitID创建带有说明的标签，-a指定标签名，-m指定说明文字。git tag -d v0.1 删除标签v0.1git push origin v1.0 推送标签1.0到远程git push origin --tags 推送所有的标签到远程git push origin :refs/tags/v0.9 删除远程标签，但是前提是要先在本地删除对应标签。4、自定义Gitgit config --global color.ui true 让Git显示颜色.gitignore在这个文件里编辑你要忽略的文件，并提交到Git中，就可以忽略特殊文件的检查。如将\*.db写入.gitignore文件中，将忽略所有db文件。可以参考github收集的所有.gitignoregit config --global alias.st status将status的别名设置成st，那么git st=git status。git config --global alias.unstage 'reset HEAD' 那么git reset HEAD filename=git unstage filenamegit config --global alias.last 'log -1' 敲git last就显示最后一次提交了。5、搭建Git服务器sudo apt-get install git 安装Git；sudo adduser git 添加Git用户；sudo git init --bare sample.git 初始化git仓库；sudo chown -R git:git sample.git修改仓库的所属用户为git；将git用户的信息git:x:1001:1001:,,,:/home/git:/bin/bash改成git:x:1001:1001:,,,:/home/git:/bin/git-shell，为了禁用shell登录。git clone git@server:/director/sample.git 克隆Git服务器上的仓库