Git 基本常用操作

**多人协作：**

**当你从远程仓库克隆时，实际上Git自动把本地的master分支和远程的master分支对应起来了，并且，远程仓库的默认名称是origin**

git remote

origin

git remote -v显示更详细的信息

git remote -v

origin ssh://kezunren@192.168.31.65:29418/source-mc (fetch)

origin ssh://kezunren@192.168.31.65:29418/source-mc (push)

**抓取分支：抓取ssh地址**

git clone ssh://kezunren@192.168.31.65:29418/source-mc && scp -p -P 29418 kezunren@192.168.31.65:hooks/commit-msg source-mc/.git/hooks/

**查看分支：**

git branch -- 查看的是本地的分支

\* master

dev

git branch –a -- 查看的是服务器的分支

\* master

dev

test

…

**在本地 新建/切换/删除 分支：**

git checkout -b dev origin/dev -- 在本地新建dev分支

git checkout dev -- 切换到本地的dev分支

git checkout master -- 切换到本地的master分支

git branch –d dev -- 删除本地的dev分支，前提是当前分支不在dev中，需要切换到其他分支才能删除dev分支，以此类推等

**一般是在本地建立个人分支：**

git reflog 查看最近使用的命令

git checkout -b kzr （本地个人分支kzr）

**每当修改为文件后**

git add \* (或者添加修改的文件) (kzr)

**然后在个人分支上commit**

git commit -m “desc” (kzr)

**然后切换到dev分支（dev分支对应git服务器的dev分支）**

**先前就要建立好连接**

（git checkout dev origin/dev）

（git branch --set-upstream-to=origin/dev）

git checkout dev (dev)

dev分支先pull服务器的代码下来，防止出现冲突

git pull (dev)

然后合并本地的分支到dev

git merge kzr (dev)

**然后提交最后代码**

git push (dev)

**然后可以再切回本地分支（这时要merge dev，保持和dev一致）**

git merge dev (kzr)

**查看修改：**

git status -- 查看本地有没有修改文件

git log -- 查看远程仓库修改提交记录

**添加本地修改到分支：**

git add \* -- 添加所有修改文件

git add txt.txt -- 添加txt.txt文件

**确认修改分支：**

git commit –m ‘修改的内容’

**推送分支：**

**推送分支，就是把该分支上的所有本地提交推送到远程库。推送时，要指定本地分支，这样，Git就会把该分支推送到远程库对应的远程分支上：**

**master** 分支是主分支，因此要时刻与远程同步

**dev** 分支是开发分支，团队所有成员都需要在上面工作，也需要与远程同步

**bug** 分支只用于在本地修复bug，就没必要推到远程了，除非老板要看看你每周到底修复了几个bug

**feature** 分支是否推到远程，取决于你是否和你的小伙伴合作在上面开发

提交到master分支

git push origin master

提交到dev分支

git push origin dev

目前在荔枝网络这边提交push是：（后面是需要提交到的分支master或dev）

git push origin HEAD:refs/for/dev