1. 更新代码

更新代码是将远程代码库上最新的代码更新至本地的操作

1. 本地无代码修改的更新

如果本地无代码修改就可以直接用git pull命令进行代码更新。

1. 本地有代码修改的更新

如果本地有代码修改需要先将代码commit后，并用git pull --rebase命令进行 代码更新。

注意：代码没有commit完整，git会阻止git pull --rebase的执行并返回错误 信息。 git status 命令可以检查本地代码库是否存在修改和commit个数。

1. git pull 和git pull --rebase的区别

两者的区别可以用下面的关系式来表示：

git pull = git fetch + git merge

git pull --rebase = git fetch + git rebase

详见：<https://www.cnblogs.com/kevingrace/p/5896706.html>

1. 缓存代码

缓存代码是将本地修改的代码缓存至本地代码库中的操作，为提交操作的前置步骤。

1. 缓存所有代码

git add --all/.

1. 缓存部分代码

git add pathName

1. 提交代码

合并提交：<https://www.cnblogs.com/tocy/p/git-rebase-merge-commit.html>

1. 推送代码
2. 合并代码
3. 提交从一个分支拷贝到另外一个分支
4. a分支更新代码git pull --rebase
5. 切换b分支git checkout b
6. b分支更新代码git pull --rebase
7. 拷贝a分支提交到b分支git cherry-pick a分支提交编号
8. 两个分支合并代码
9. a分支更新代码git pull --rebase
10. 切换b分支git checkout b分支名
11. b分支更新代码git pull --rebase
12. b分支合并a分支git merge --no-ff a分支名
13. 冲突解决

冲突是只合并代码时，git无法判断如何合并，需要人工进行处理的操作。

1. git pull --rebase中的冲突
2. git merge中的冲突
3. git am中的冲突
4. 分支操作
5. 分支查询

git branch -a ：显示所有分支

git branch ：显示本地分支

1. 创建分支

git branch branch-name：本地创建分支

git checkout branch-name ：切换分支

git checkout -b branch-name：新建分支并切换到该分之下

git push origin branch-name：把本地分支推到远程端

1. 删除分支

git branch -d/D branch-name：删除本地分支

git branch -r -d origin/branch-name：删除远程分支

git push origin :branch-name：推送删除分支到远端

1. 文件操作
2. 文件删除

git rm 文件 ：删除指定文件

git rm -r 文件/ ：删除执行文件夹及其子目录

git clean 参数(-n[显示要删除的文件和目录]，-f[删除文件]，-df[删除文件和 目录])：删除没有加入缓存的所有新建文件

1. git reset --hard origin/branch-name 同步远端