git工作原理：

1.分布式，每个克隆或更新远程仓库的用户都拥有一份最新的完整的仓库。
2.分支结构，不要直接操作remote的远程分支，应该首先checkout一份到本地分支。远程分支是指与远程仓库上同步的分支。而你所有的操作应该是在你的本地分支上操作。
3.时刻记得你在那个分支下工作，因为你切换到任何分支都不会改变本地目录
4.“origin”代表远程仓库，为默认的远程仓库别名
5.git的分支仅仅是一个提交的标记，而不是目录，与svn的目录式分支结构不同
6.仓库的意义是整个代码仓库，本地分支/远程分支，都是你本地仓库中的分支

git常用操作命令及解释

注：以下所有以$开头的，为命令行操作，$之后的是命令, #号之后是注释

1.克隆一个仓库(以5+3新后台为例)
$ git clone git@v.htp.me:55tuan.git #相当于svn checkout
然后进入仓库目录，运行如下命令来忽略git对于文件权限的检查
$ git config core.filemode false

2.抽取一个分支以及切换到一个分支（切记，要得到一个分支，应该先抽取它，把它抽取到本地，生成一个本地分支）
命令格式 git checkout 本地分支名 origin/远程分支名，通常，本地分支名和远程分支名一样，当然，也可以不一样。
$ git checkout release/0.2 origin/release/0.2
$ git checkout master origin/master
$ git checkout master #切换分支也是使用checkout，只不过不再输入远程分支信息

3.查看分支，注意，*号在那个分支名前面，就说明你目前处在那个分支下。
$ git branch #查看本地分支
$ git branch -a #查看所有分支，包括你仓库中的远程分支
$ git remote prune origin #更新你本地仓库的所有分支和远程仓库的分支保持一致，主要是删掉本地仓库中与远程仓库相比较，已经不存在的分支

4.本地仓库分支的创建和删除
$ git branch 分支名 #创建一个本地分支

$ git branch -d 分支名 #删除一个本地分支，参数 -D 强制删除，删除前要先切换到其它分支

5.更新仓库但不将修改合并分支的修改

$ git fetch

6.更新仓库且将修改合并到本地分支，这个最常用，相当于svn up

$ git pull

7.查看状态，是否有提交，删除，更新等信息，如果看到大批文件权限有更新，请使用“git config core.filemode false” 来忽略权限检查

$ git status

注意：后面的操作测试和配管请勿使用，仅仅做为简介。

8.提交，注意：提交仅仅是将你的修改提交到你本地的仓库，而非远程

$ git commit -m “提交注释”

如果之前未使用 git add 将修改的文件加入暂存区，可再加一个参数 -a 来将所有未暂存的修改提交

9.推送到远程仓库，只有将你的修改或分支推送到远程仓库，其它人才能从该远程仓库获取

$ git push origin #推送当前分支到远程仓库，前提是远程仓库有这个分支

命令格式 git push origin 本地分支名:远程分支名，本地和远程分支名可以不一样，远程分支名是推送到远程仓库的分支名

10.删除远程仓库中的分支

$ git push origin :远程分支名

工作流程

1.生成你的公钥和私钥，将公钥发给git仓库管理员
2.克隆一个仓库
3.检出一个远程分支到本地分支
4.在本地分支上进行开发工作并提交
5.将你的提交推送到远程仓库
6.要经常更新你的本地分支 git pull