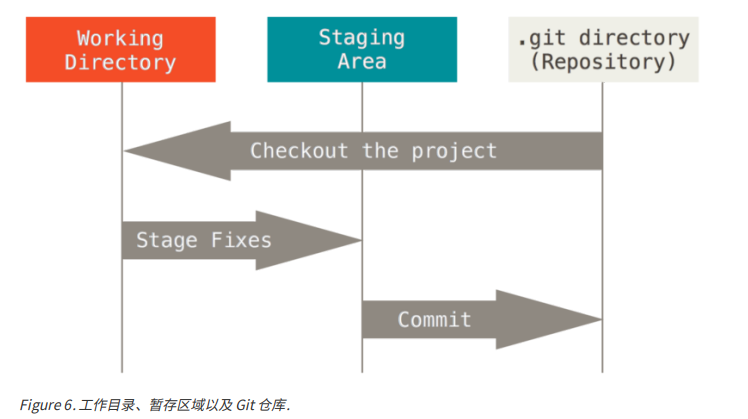
git:分布式版本控制系统

集中式版本控制系统：有一个单一的集中管理的服务器，保存所有文件的修订版本，而协同工作的人们都通过客户端连到这台服务器，取出最新的文件或者提交更新。

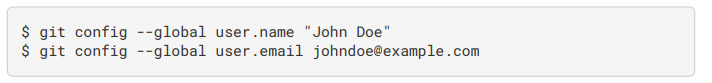
缺点：

中心服务器的单点故障会影响工作效率，在中央服务器故障时，所有人都无法协同工作，无法提交更新。



Git配置及常用命令：

用户信息：



git config --list：检查当前的配置信息

git init 将当前目录初始化为一个git仓库

git remote add origin git://127.0.0.1/abc.git 或

git remote set-url origin https://github.com/zbg-hub/xxx.git

这样就关联了远程仓库abc（新的远程仓库需要从github或者gitlab上建立），origin是远端仓库的本地名称

git add file 将文件添加进暂存区

git commit -m "提交的说明" 将暂存区的文件提交到仓库里

git remote remove origin移除本地的远端仓库（指的是解除本地仓库与远程仓库的关联）

git pull 拉取最新代码

git push -u origin master 推送到远端仓库abc(本地名叫origin) -u表示默认的推送的远端仓库，即下次使用git push会默认推送到abc

git branch branchName 建立一个新分支

git checkout branchName 切换到另一个分支

git checkout -b branchName 建立一个新分支并切换到该分支

检查是否有ssh key

cd ~/.ssh

ls

如果没有的话生成密钥对：

ssh-keygen -t rsa -C "zbg@163.com"

最后得到了两个文件：id\_rsa和id\_rsa.pub

然后添加密钥到ssh-agent

eval "$(ssh-agent -s)"

ssh-add ~/.ssh/id\_rsa

然后将id\_rsa.pub复制到gitlab中的ssh key中

如果之前使用https进行推拉，还要将仓库的url改成ssh的url，使用

git remote set-url origin git@github.com:zbg-hub/xxx.git

建立一个远程分支：

git checkout -b my-test //在当前分支下创建my-test的本地分支分支

git push origin my-test //将my-test分支推送到远程

git branch --set-upstream-to=origin/my-test //将本地分支my-test关联到远程分支my-test上

git branch -a //查看远程分支

git branch -d <BranchName> 删除本地分支

git push origin --delete <BranchName> 删除远端分支