//////////////////////////////////git介绍//////////////////////////////////

git是分布式版本控制系统 不需要中央服务器

svn是集中式版本控制系统 需要重要服务器

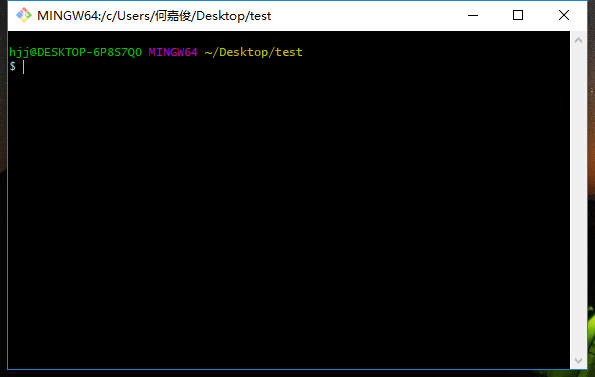
///////////////////////////////////安装git////////////////////////////////////////

下载地址：<https://git-scm.com/downloads>

点其中一个即可



安装完成后，右键任意文件，会有一个Git Bash指令，打开后出现下图：



Desktop/test 指的是当前文件的路径

////////////////////////////////////创建远程仓库///////////////////////////////////////////

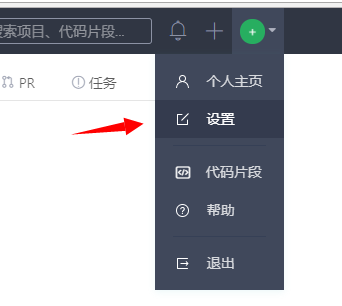
1，选择一个远程仓库:码云或者github

2，注册账号

3，创建SSH Key 密钥。由于本地仓库和远程仓库需要通过SSH来加密链接，所以需要创建密钥。密钥有两个，1个是公共密钥，1个是私有密钥。

步骤：以码云为例子

1）点击右上角设置按钮



2）点击SSH公钥



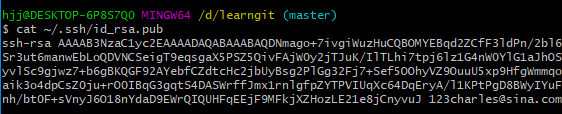
1. 点击怎么样生成公钥



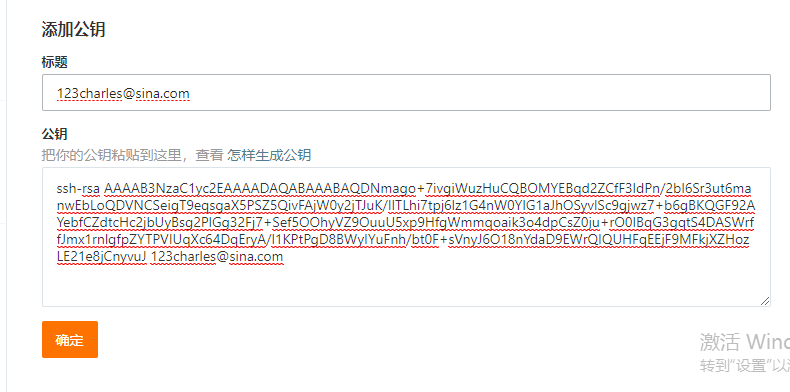
4）复制该行代码到git上，把邮箱地址修改成任意自己的邮箱地址，回车三次即可生成公钥



生成成功

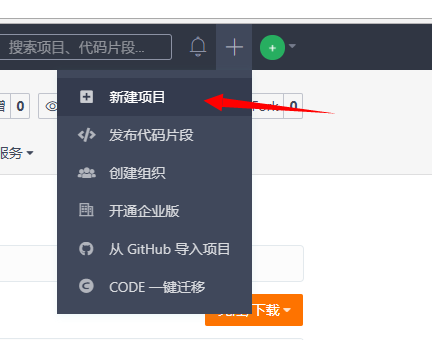


5）把生成的公钥添加到下面这里，标题随意，点确定即可



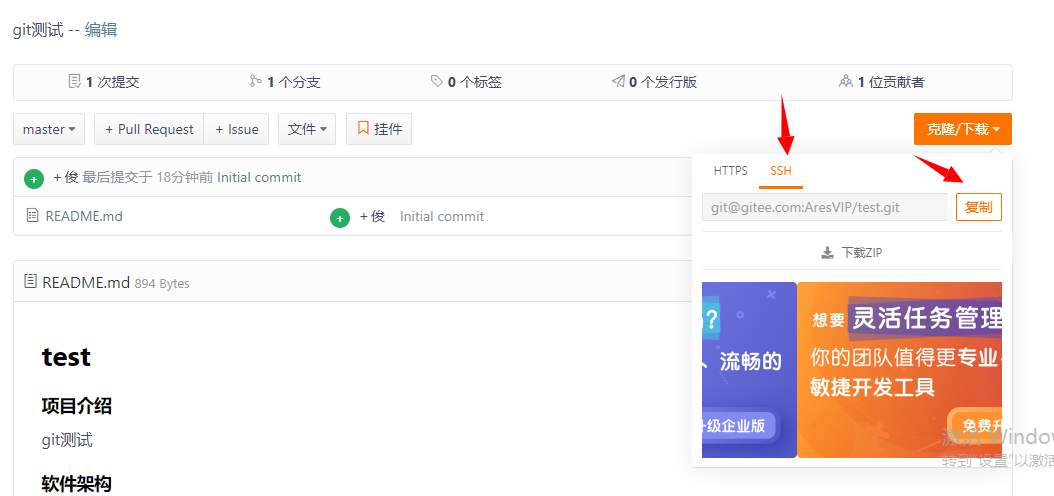
////////////////////////////远程仓库与本地仓库链接////////////////////////////////////

1，新建项目



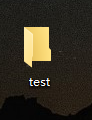
1. 创建成功后，克隆远程仓库到本地

1）点击远程仓库克隆/下载，点击SSH，点击复制



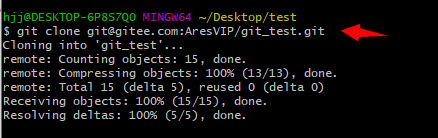
1. 在本地创建一个目录，对着目录右键git bash here，打开命令行工具

如图：

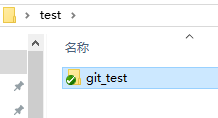


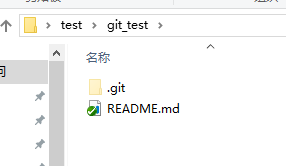
1. 输入指令 git clone git@gitee.com:AresVIP/git\_test.git

如图：

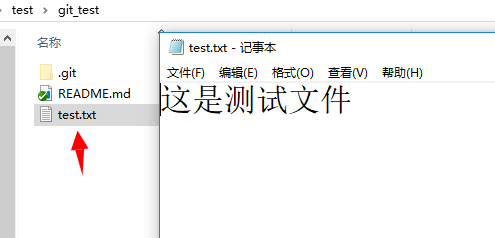


把远程的仓库拉取到本地上来



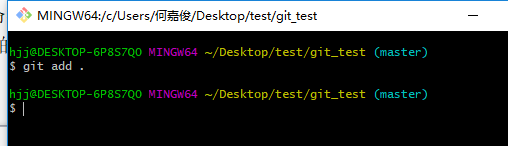


1. 提交本地文件到远程仓库
2. 在本地新增一个文件



1. 在git\_test下打开命令提示行，在命令行输入git add .

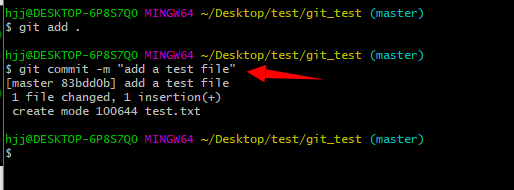
解释：这个指令是把所有的文件都添加到缓冲区 .的意思是所有文件，这里也可以写单个文件名



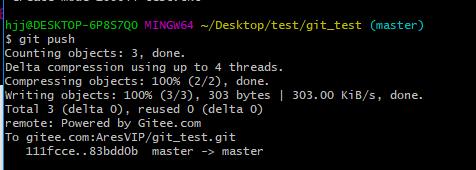
注意，是在git\_test目录打开，而非原来的test目录

1. 在命令行中输入git commit -m “add a test file”，把缓冲区的文件提交到版本控制区

解释：commit -m是把缓冲区的文件提交到版本控制区 ，双引号里面的内容可以是任意的，在这里只是一个说明，但是要注意这里必须要填内容



4）最后输入git push 把版本控制区的文件都提交到远程仓库里面



远程仓库里面就有新增加的文件了



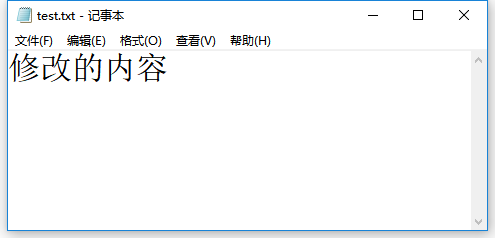
/////////////////////////////把修改后的文件提交到版本控制器里////////////////////////////

$ git status //查看仓库状态

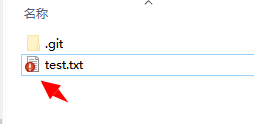
$ git diff //查看具体修改的内容

例：修改一个文件后，检查该仓库状态，再把该文件添加到缓冲区，然后提交到版本控制器

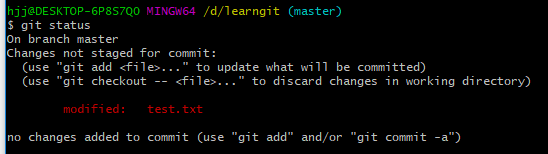
1. 修改test.txt文件



修改后变成红色叹号



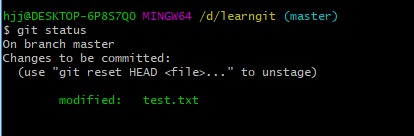
1. 使用git status 查看仓库状态，发现test.txt文件被修改了，但是并没有提交到缓冲区



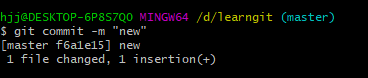
3，把修改后的test.txt文件提交到缓冲区



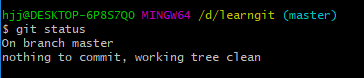
4，查看当前状态，添加成功



5，把在缓冲区的test.txt文件添加到版本控制器



1. 检查当前状态，没有东西需要提交，工作目录是干净的



/////////////////////////////////////版本回退/////////////////////////////////////////////

$ git log //查看的历史记录

$ git log --pretty=oneline //简洁模式

Commit id 版本号

3628164fb26d48395383f8f31179f24e0882e1e0 append GPL

ea34578d5496d7dd233c827ed32a8cd576c5ee85 add distributed

cb926e7ea50ad11b8f9e909c05226233bf755030 wrote a readme file

$ git reset --hard HEAD^ //返回上一版本号

$ git reset --hard HEAD^^ //返回上上版本号

$ git reset --hard HEAD~100 //返回上100个版本号

$ git reset --hard 3628164 //返回到指定的本号

$ git reflog //记录你的每一次命令和commit id

////////////////////////////////撤销修改和删除文件/////////////////////////////////////////

$ git checkout -- file //撤回工作区的修改或删除

$ git reset HEAD file //撤回缓存区的修改

$ git rm fileName.txt //删除文件

///////////////////////////////创建分支与合并分支//////////////////////////////////////////////

1. 在远程仓库创建分支

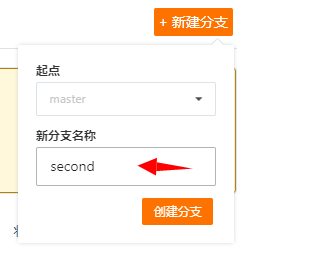
1）



2）



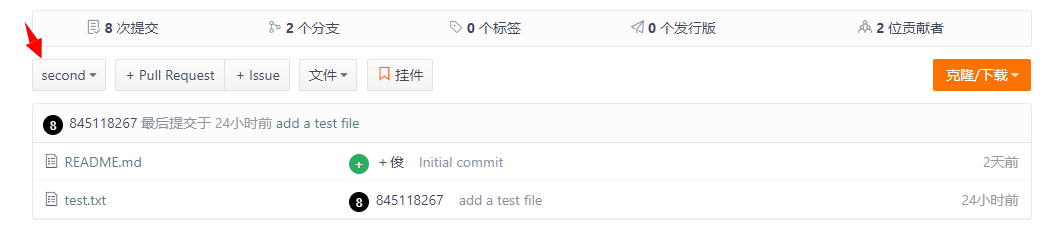
3）



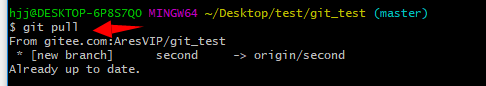
4）



5）

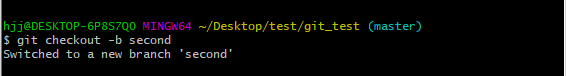


1. 在本地拉取远程仓库分支 $ git pull

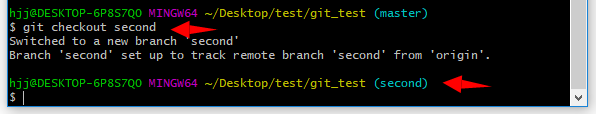


new branch 新分支

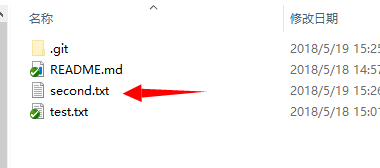
注意这里除了可以在远程仓库拉取下来也可以在本地新建分支 $git checkout -b second



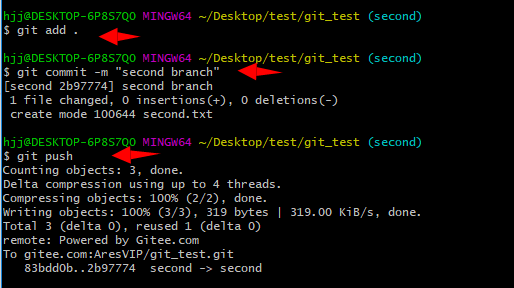
1. 切换分支 $ git checkout second 切换到second分支



1. 在本地创建一个second.txt

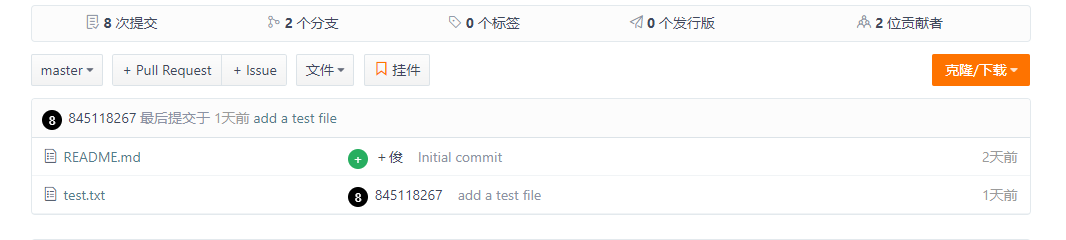


1. 把该文件推送到远程仓库的second分支上



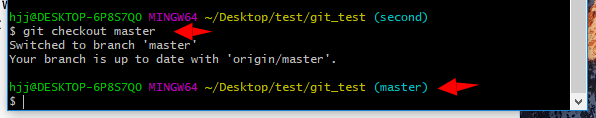
1. 查看远程仓库，在second分支上确实有second这个文件，但是master分支上并没有



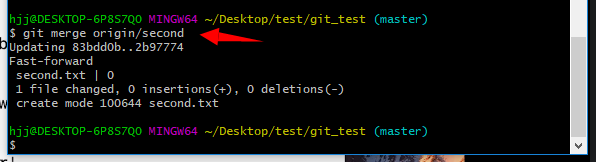


合并分支

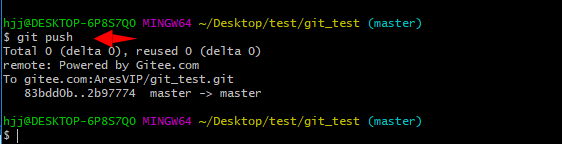
1. 切换回mater分支



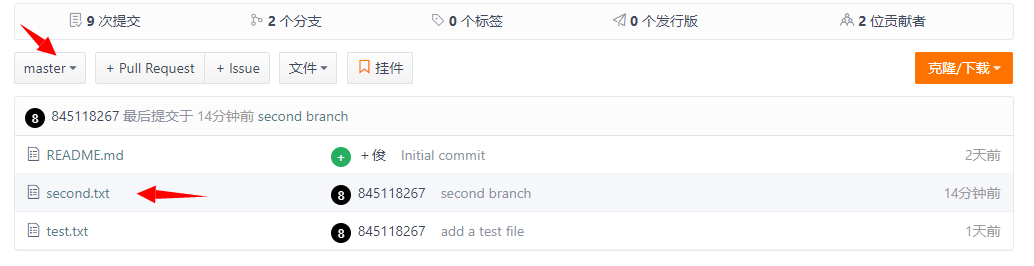
8，合并远程仓库的second分支到本地的master分支



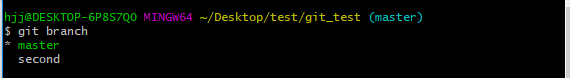
9，把合并后的master分支推送到远程仓库的master分支



1. 合并后的远程仓库master分支就有second这个文件了



11，git branch 检查本地仓库有多少分支，这里显示有master和second两个分支，\*表示目前正处于master分支上



12，如果不需要second分支，可以使用git branch -d second即可删除该分支

