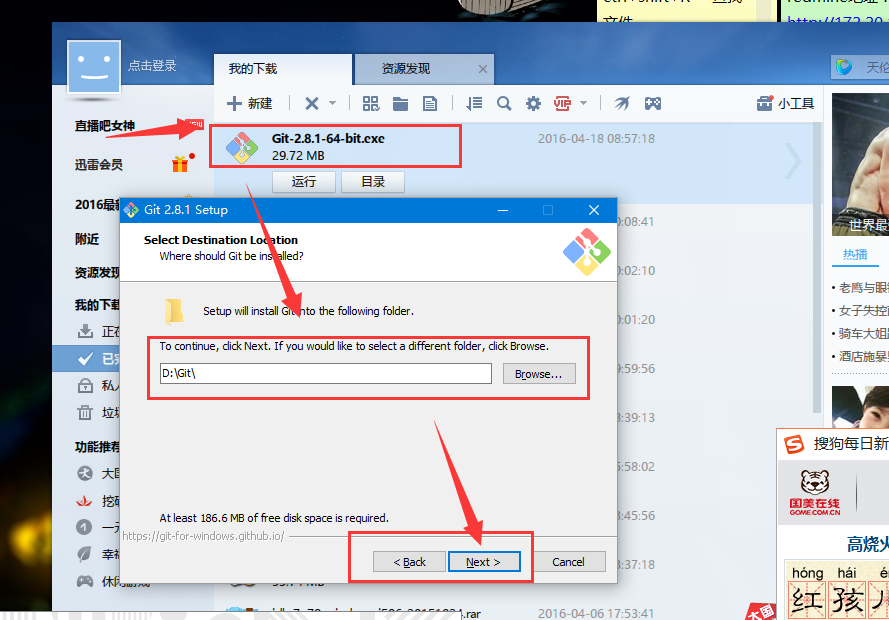
GIT从零开始

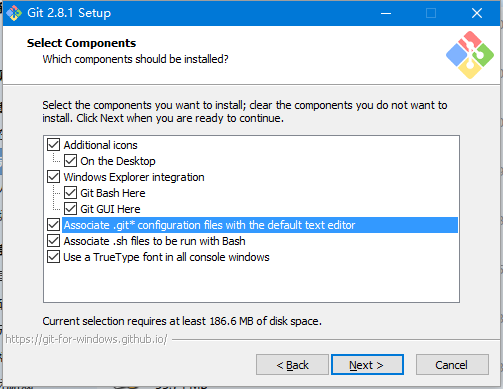
参考：

<http://www.liaoxuefeng.com/wiki/0013739516305929606dd18361248578c67b8067c8c017b000>

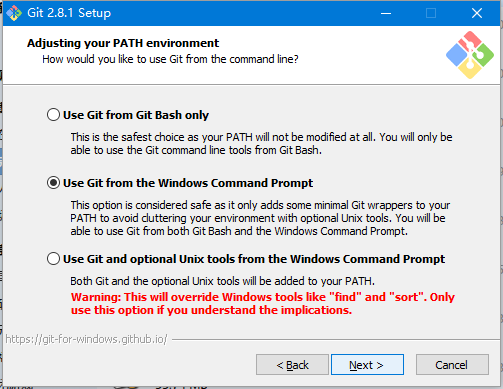
# 安装

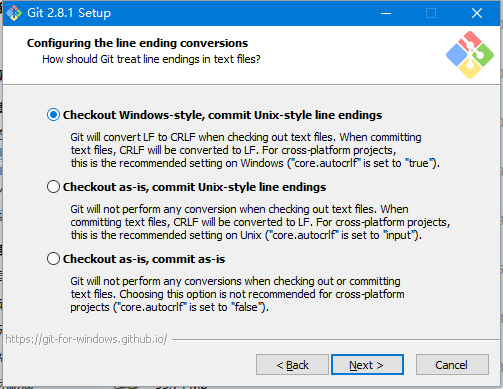


都安上算了

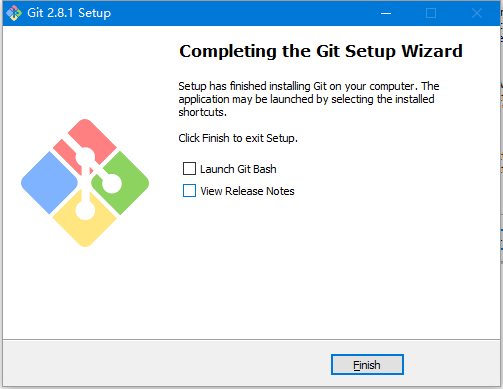


用默认的吧

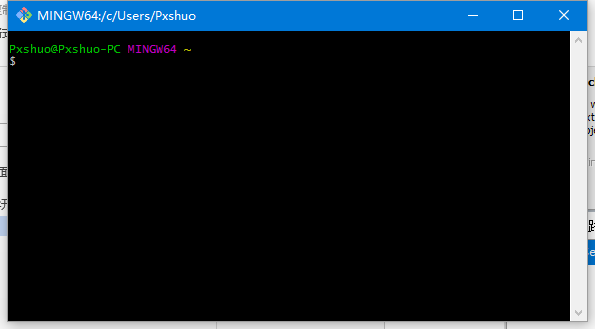




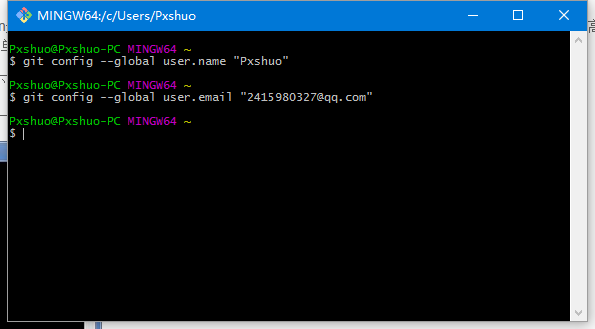
接下来就是一路默认直到安装



找到git->git bash,win10中直接在小娜中搜索git bash即可



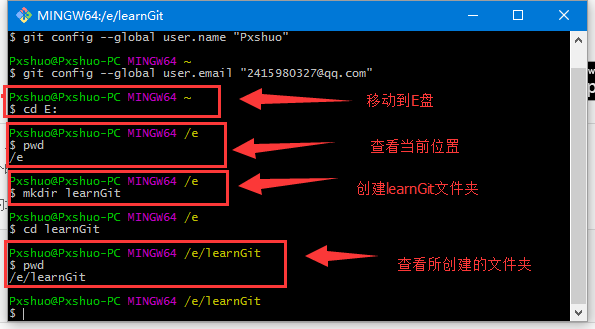
然后在这个命令窗口中进行设置

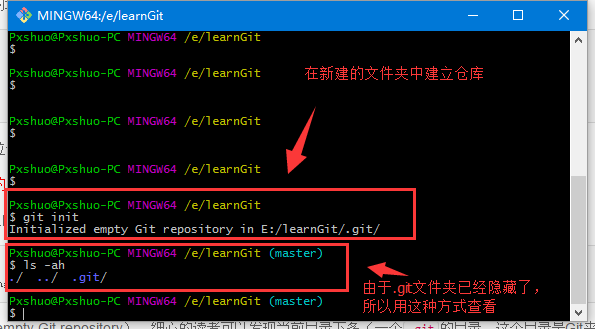


# 创建版本库

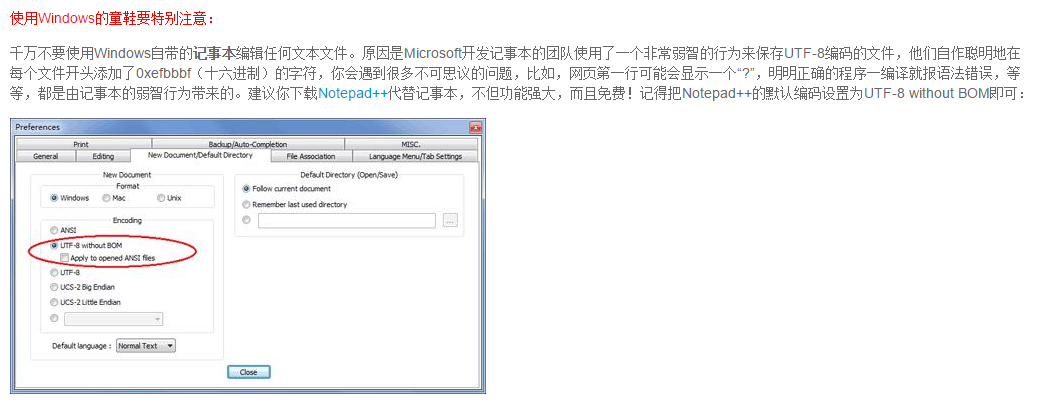
也叫仓库，英文为repository

1、新建一个文件夹，尽量不使用中文

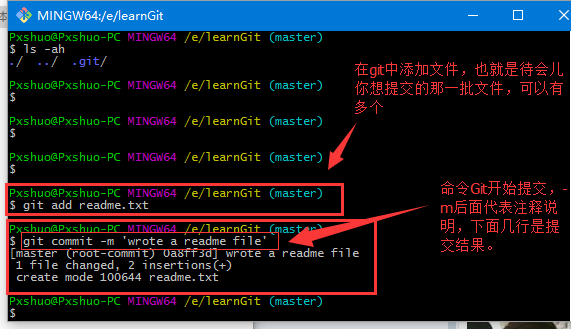
2、初始化仓库



Ps：虽然他这么说，但是我还没遇到过这种情况，有待验证

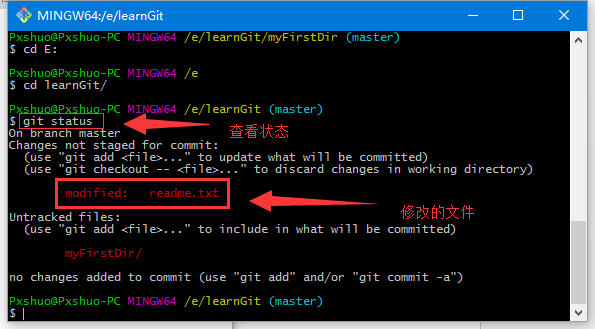


3、添加文件并提交文件与注释

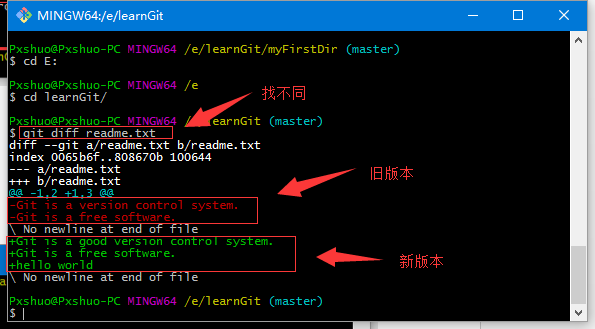


# 时光穿梭机

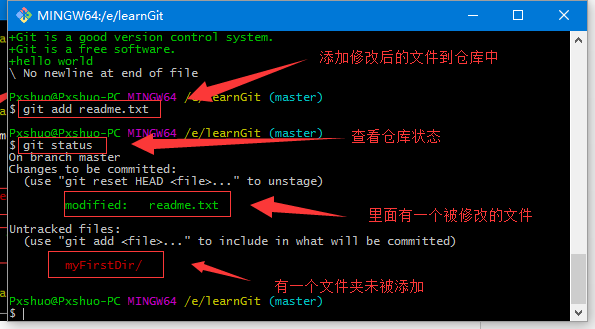
1. 修改文件，查看文件状态



1. 对比不同之处



1. 重新添加到仓库

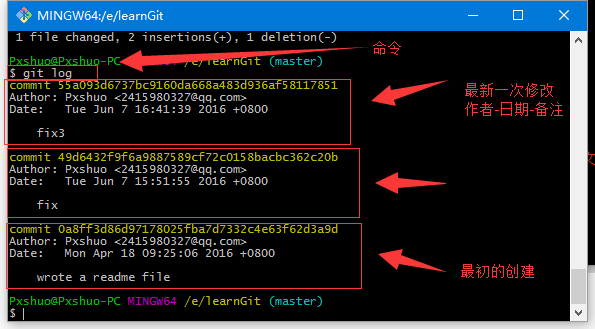


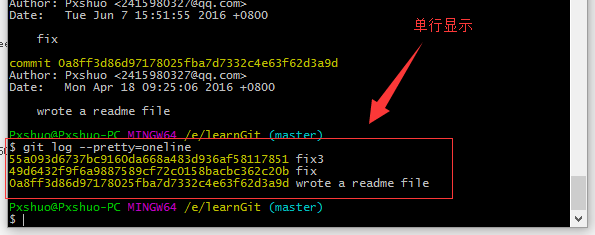
1. 总结

git status与git diff

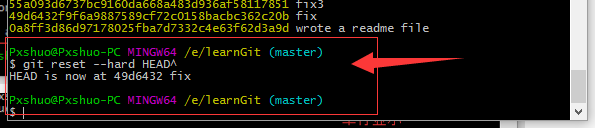
## 1、版本回退

1、历史记录git log

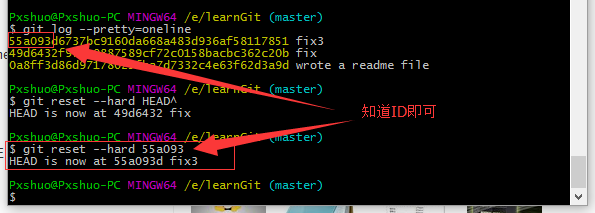




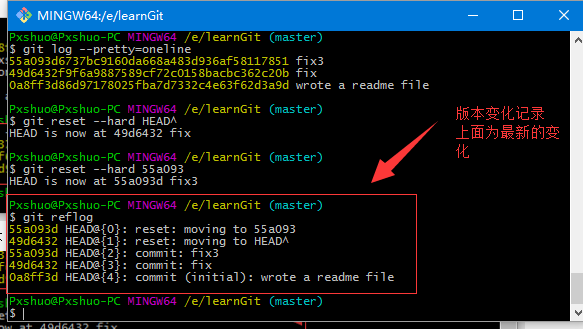
2、回退版本。当前版本为HEAD，前一版本HEAD^,再之前为HEAD^，前100版本为HEAD~100



3、现场恢复最新版本，需要知道最新版本id的前几位



1. 也可以查看以往的记录



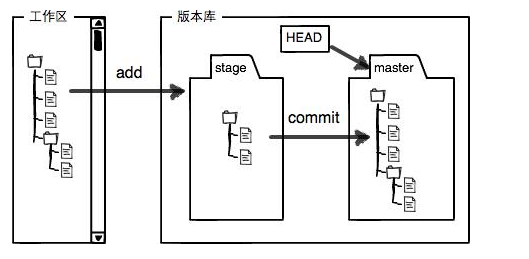
1. 总结

git log、git log –pretty=online

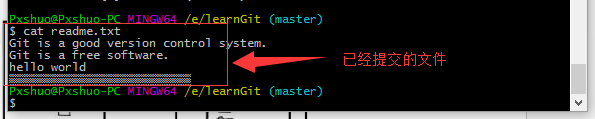
git reset –hard HEAD^^(或者HEAD~100,或者某一个提交ID)

git reflog

## 2、工作区与缓存区

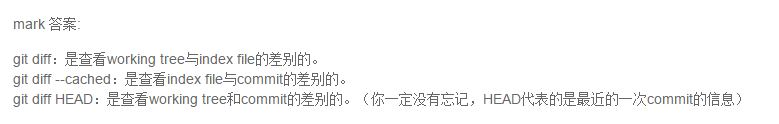


## 3、管理修改



* 工作区中存着你实时改正的文件
* 缓存区中存着你add的文件
* master分支中保存着已经上传的内容

分为以上三部分

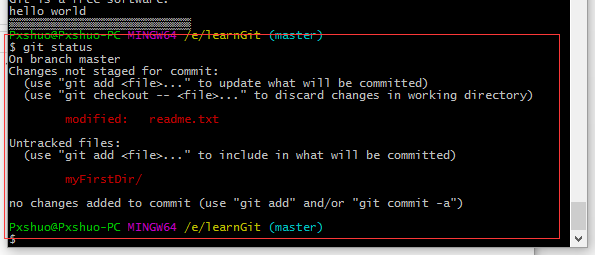


总结

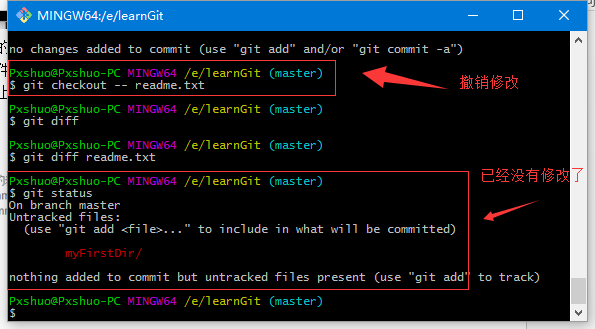
cat filename

## 4、撤销修改

1、只是在工作区中进行了修改，想取消修改，恢复到上次commit或者add的时候，使用git checkout file



2、使用回退



3、如果已经add过了，使用git reset HEAD file,取消add操作，再通过git checkout – file进行回滚



4、总结

git checkout –file 恢复到上一次add或者commit时文件的样子

git reset HEAD file 取消add操作

git reset HEAD^ file 恢复到commit的上一个版本

对于reset HEAD的猜测。在add的时候，将\_HEAD指向add的文件，commit的时候将HEAD指向\_HEAD,reset的时候将\_HEAD指向HEAD，就取消了add的操作

## 5、删除文件

1、删除文件 rm file



2、将删除的文件还原

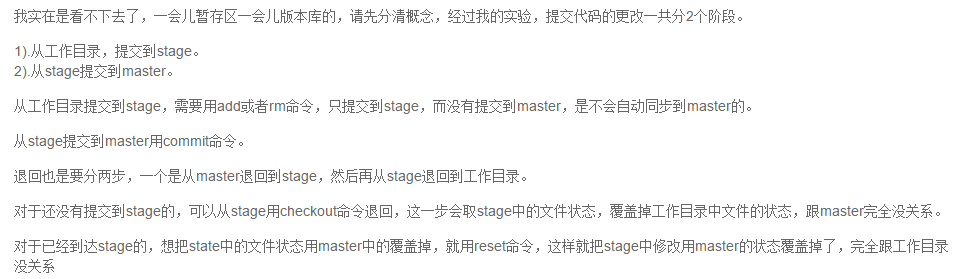




恢复版本库里的最新版本。包括add操作与commit操作。

如果工作区文件已经删除，如果之前add过，进行提交也同样会删除

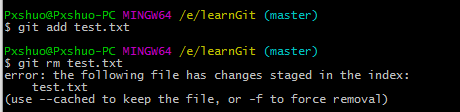
Reset HEAD 表示退回版本库的某一个版本，并不是退回一个文件的某个版本



3、总结

checkout 是检出add中的数据—在本地的时候—远程再进行测试

rm会删除文件，但不会把状态提交到stage中，但git rm会将状态提交到stage中。所以rm删除文件之后在之前有过add操作时可以通过checkout直接进行恢复，git rm需要通过reset HEAD – file 重设头文件之后进行回滚



修改之后add文件到stage，进行git rm 会因为stage中的修改没有更新而提示，使用-f可以强制删除，更新stage

关于checkout

1-准备实验数据

//首先新建文件test.txt，并写入数据new

git add test.txt//将数据提交到stage

git commit test.txt -m "new"//将数据由stage提交到master

//修改test.txt，添加数据add1

git add test.txt//将修改后的文件再次提交到stage

//再次修改test.txt，添加数据add2

现在工作区中数据为**newadd1add2**  
stage中的数据为**newadd1**  
master中的数据为**new**

2-开始测试

git checkout -- test.txt

工作区中的数据变为**newadd1**  
表示checkout恢复的为stage中的数据

对于刚刚执行过commit操作的情况，目前我学到的commit之前有两种操作，一个是git add file之后，一个是git rm file之后。

git commit test.txt -m "add1"//提交stage中的数据

//向test.txt中再次添加add2

git checkout -- test.txt

stage与master中数据都是**newadd1**  
工作区中的数据由**newadd1add2**变回**newadd1**  
add之后stage中的数据与master中的数据相同，并不能确定恢复的是谁的数据

git rm test.txt

删除文件之后，使用checkout就会提示错误，因为无法找到test.txt  
这个时候工作区与stage中的test.txt都被删除了，之后master中的数据没被删除，但是恢复并没有成功

综上所述，checkout恢复的确实是stage中的数据

再者，git reset --hard HEAD^ 是恢复至上一个版本，不是恢复到上一个文件。之前用单个文件进行说明一直以为是将文件恢复到上一个状态呢。其实如果修改了多个文件，这个操作会将所有文件都退回一个版本。  
目前接触的每个版本是依据commit操作或者reset操作来界定版本的。可以回滚到某一个commit的时刻，也可以回滚到某一个reset操作的时刻（通过ID来进行回滚）

# 远程仓库

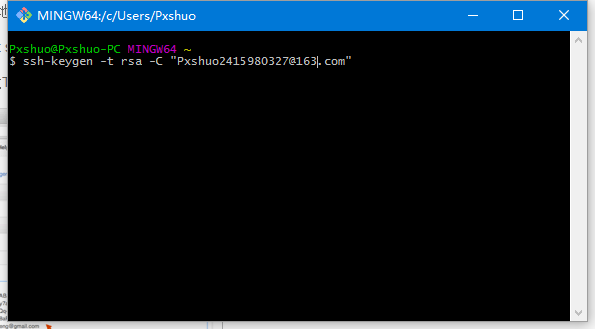
## 1、新建github账户

使用163邮箱进行注册。qq邮箱收不到邮件了。

Pxshuo163

Pxshuo2415980327@163.com

2、创建ssh密钥

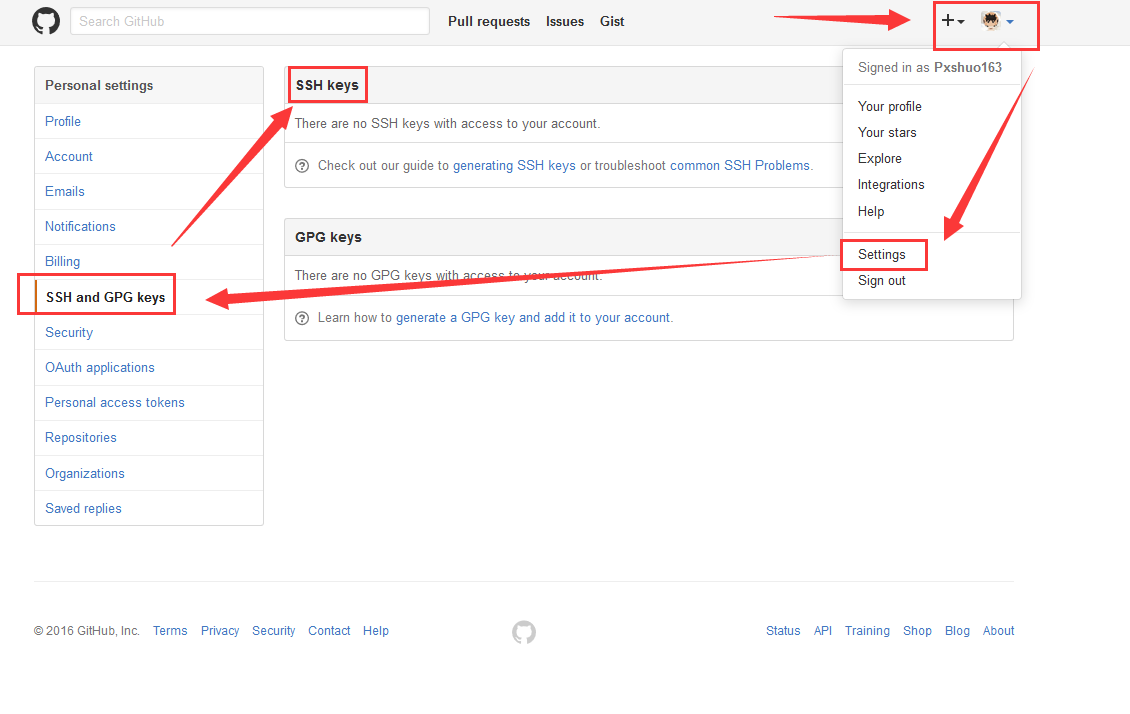


之后全部直接点确认即可

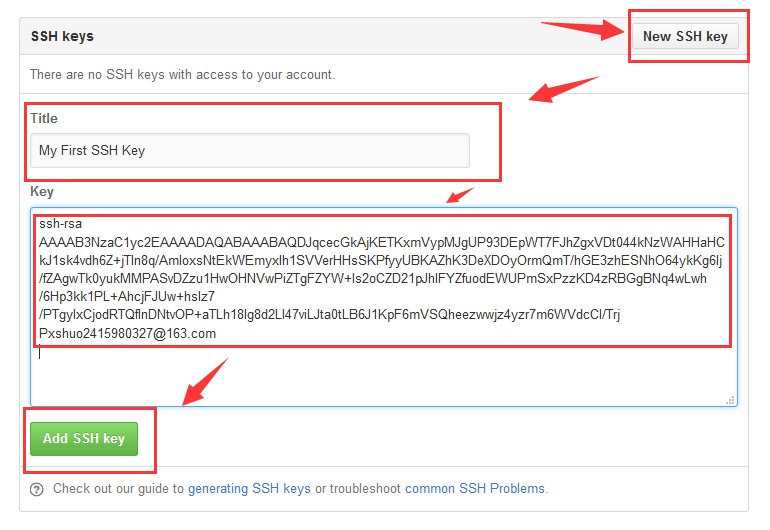


之后登陆github账号

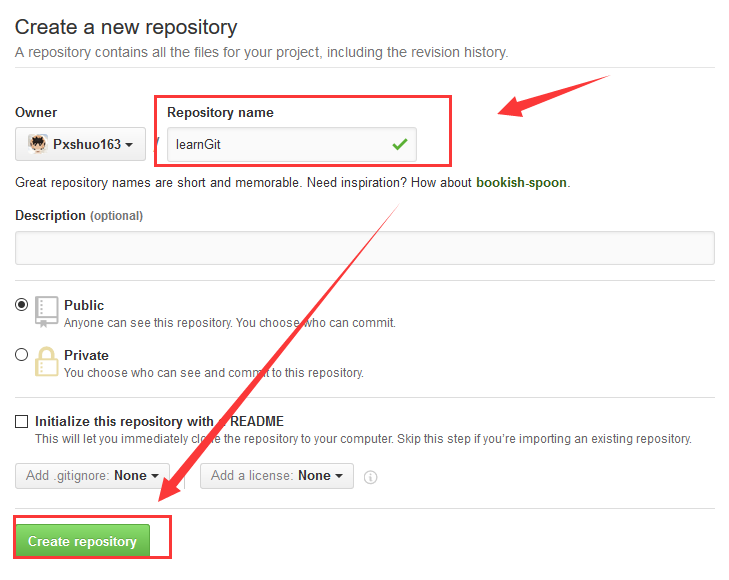
https://github.com/



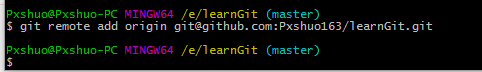
点击add SSH Key 将id\_rsa.pub中的内容添加到密钥中去



## 2、创建远程库



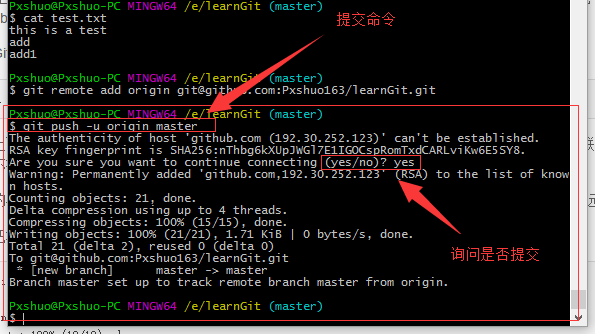
1. 连接到github中



git@github:github的用户名/所见的工程名.git



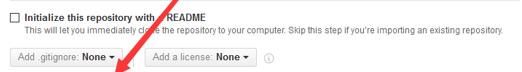
1. 提交到远程库



第一次提交的时候使用 git push –u origin master，通过-u将两个关联起来

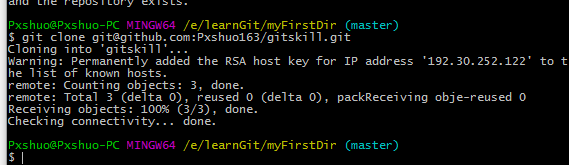
之后可以通过git push origin master 推送即可

1. 从远程库克隆



这个是自动创建readme文件用的

开始克隆：git clone [git@github.com:Pxshuo163/gitskill.git](mailto:git@github.com:Pxshuo163/gitskill.git)



可以使用\*匹配所有文件

## 3、尝试在另一台机器上进行连接库

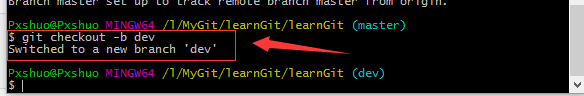
同样使用Pxshuo2415980327@163.com

# 分支管理

## 1、创建于合并分支

版本回退使用HEAD进行版本控制。在每一次reset或者commit的时候都会生成一个新的版本。HEAD指向当前的默认分支master，而master作为一个指针指向当前的版本。并随着版本的修正不断地改变。当创建分支的时候，比如创建一个dev，也就意味着创建了一个指针，指向master的当前指向，而head也改为指向dev。合并版本就是意味着将master指到dev所在的位置。而commit添加的注释应该是版本的注释

1. 创建分支，分支名称不用引号包含





1. 查看当前分支

