# GitHub 新手使用手册

V1.0 20200201 源自互联网，引自简书【[小贤笔记](https://www.jianshu.com/u/195a91a868f5" \t "https://www.jianshu.com/p/_blank)】的文章。

V1.1 20200203 根据这几天的实践，加入自己的修改，更详细一些；增加了克隆Notepad++代码库的操作；

说明1：学习最好的方式是学习学习的方法，对于github，如果是完全从0开始，建议先看一到两个github的视频教程，了解一点基本的概念，然后找到最好最合适的最详细的一遍跟着操作就能成功的帖子，对于初学者的自信心是至关重要的，也会让大家有兴趣继续。这个文章很长，当时很多标绿色的都可以跳过，祝好。不想看这个手册的话，可以看廖雪峰的博客，很棒、很赞。

<https://www.liaoxuefeng.com/wiki/896043488029600/896827951938304>

说明2：本文摘抄自简书“小贤笔记”的文章，中间添加了一些修改和完善，主要目的是记录了自己学习的过程，更详细一些，原文网址如下：<https://www.jianshu.com/p/bf95d885978e>。

GitHub是什么？可以百度，也可以看下面绿色的部分，也可以直接跳过这个绿色的部分。学习就是需要取舍，亦如做人，不是吗？

**基于git实现的代码托管。git是目前最好用的版本控制系统了，非常受欢迎，比之svn更好。GitHub可以免费使用，并且快速稳定。即使是付费帐户，每个月不超过10美刀的费用也非常便宜。利用GitHub，你可以将项目存档，与其他人分享交流，并让其他开发者帮助你一起完成这个项目。优点在于，他支持多人共同完成一个项目，因此你们可以在同一页面对话交流。创建自己的项目，并备份，代码不需要保存在本地或者服务器，GitHub做得非常理想。**

**学习Git也有很多好处。他被视为一个预先维护过程，你可以按自己的需要恢复、提交出现问题,或者您需要恢复任何形式的代码，可以避免很多麻烦。Git最好的特性之一是能够跟踪错误，这让使用Github变得更加简单。Bugs可以公开，你可以通过Github评论，提交错误。**

**在GitHub页面，你可以直接开始，而不需要设置主机或者DNS。对于我来说，我可以在公司和家编辑同样的一套代码了，用U盘或者网盘都好麻烦的说理论说太多容易忘，来实践实践吧。**

**准备工作：打开github网站，注册一个账户，注意牢记用户名和注册邮箱，注意牢记用户名和注册邮箱，注意牢记用户名和注册邮箱。**

## 一. 创建 github repository(仓库)

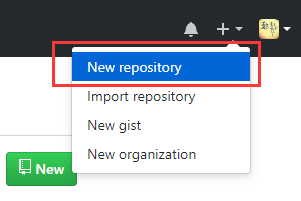
#### 1. 用注册号的用户名密码登录 [github](https://links.jianshu.com/go?to=https://github.com/" \t "https://www.jianshu.com/p/_blank)

#### 2. 创建 repository(仓库)

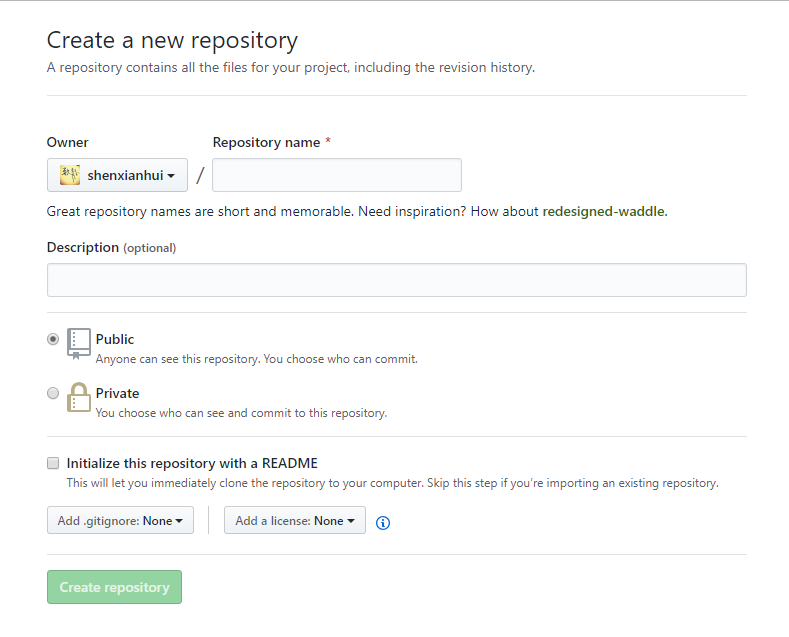
为啥要叫 repository(仓库)？我起初也纳闷，叫代码库不更简单明了么？ 但仔细一琢磨，仓库一般都是放粮食的吧，这是把代码当作饱腹之物，多有爱，瞬间觉得这冰冷冷的代码充满了查克拉。

扯远了，来看怎么创建仓库

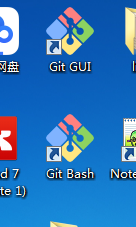
看到右上角 "+" 没? 点击选择 "New repository"



下面是创建仓库信息，只有名字是必填项



创建成功后，可以看到自己的仓库地址。它还介绍了github仓库的常用指令, 这个指令需要在本地安装 git 客户端后再客户端里执行，**就是这两个货，详见第二章。**

****

常用指令如下：

git init // 把这个目录变成Git可以管理的仓库

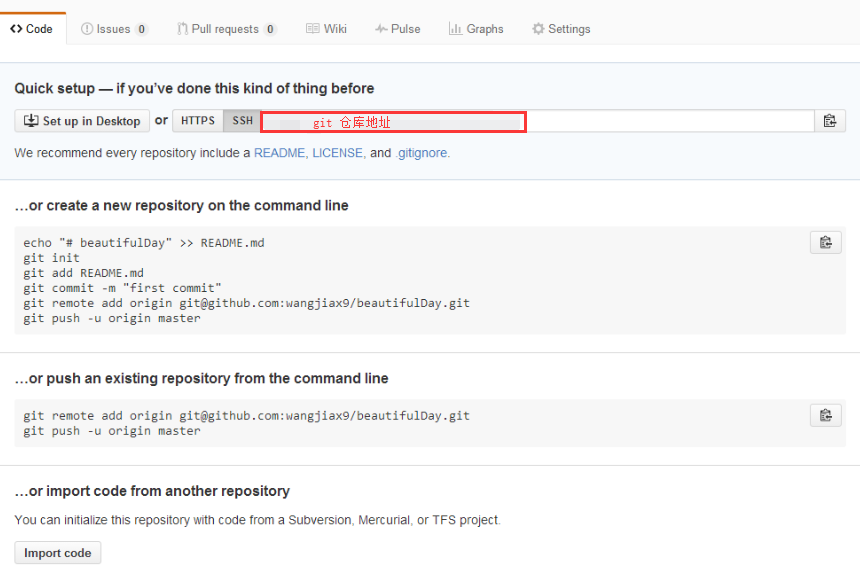
　　git add README.md // 将某个文件README.md添加到本地仓库

　　git add . // 不但可以跟单一文件，还可以跟通配符，更可以跟目录。一个点就把当前目录下所有未追踪的文件全部add了

　　git commit -m "first commit" // 把文件提交到仓库

　　git remote add origin git@github.com:wangjiax9/practice.git // 关联远程仓库

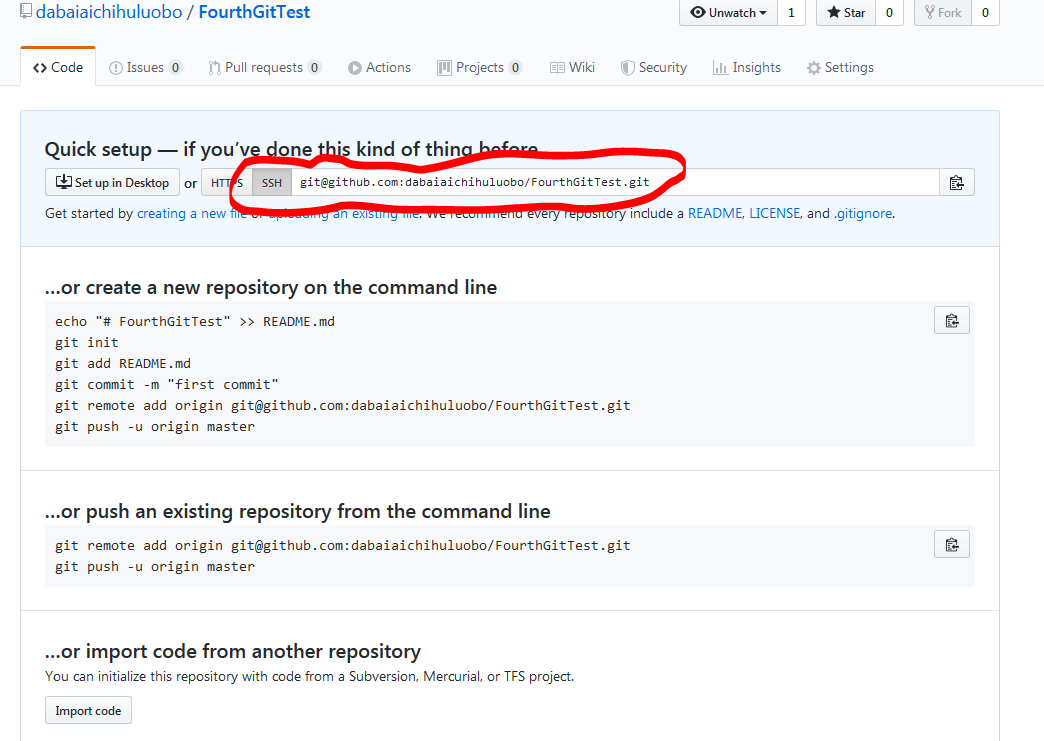
　　git push -u origin master // 把本地库的所有内容推送到远程库上



**注意GitHub有两种方式分别为HTTPS和SSH两种方式，如下图**

****

**系统默认是HTTPS模式，我是使用SSH模式，需要点击切换一下，也有网友说可以使用HTTPS模式的仓库地址，这个没有试过，不过Linus老人家应该做的不会那么死，理论上肯定是兼容的，想尝试的网友可以尝试一下**。

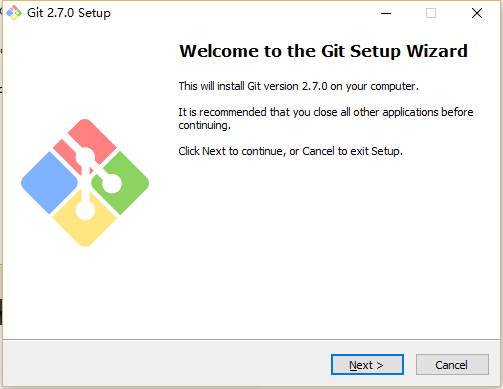
本文新建git仓库FourGitTest地址如下：【原作者有一个beautifulDay的地址，这里为了学习，我自己重新新建了一个】，**注意这个仓库需要后面与本地仓库进行关联。**

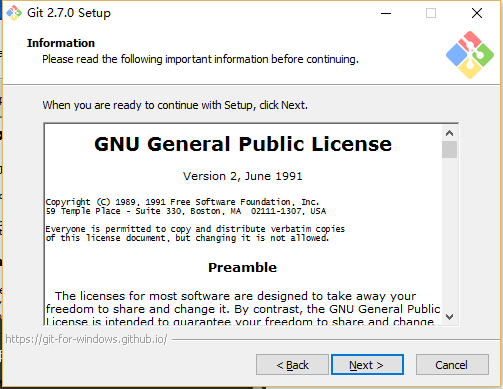
## 二. 安装 git 客户端

#### 1. 下载 [git](https://links.jianshu.com/go?to=https://git-scm.com/" \t "https://www.jianshu.com/p/_blank)

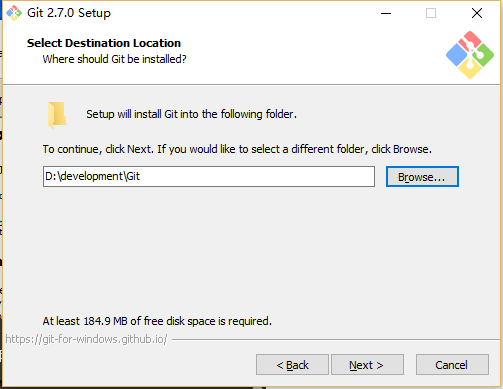


#### 2. 安装客户端



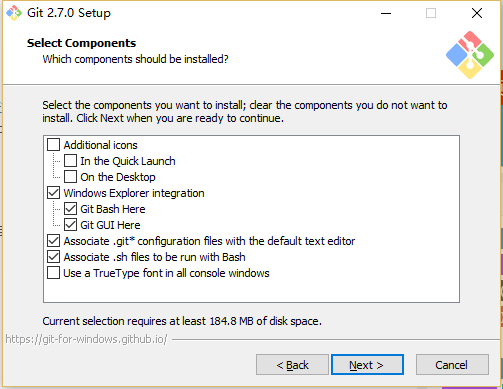


选择安装路径，千万别选带中文的路径，有时候会引起不必要的误会

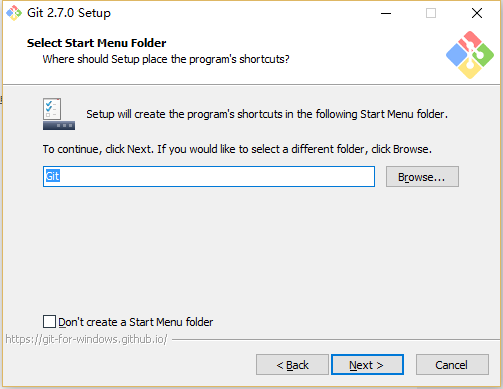


选择安装组件，按默认的来就好了。**理论上下面的这段绿色的全部可以不用看，直接跳转到第三节。如果想详细了解这一步步安装都是什么含义，可以看下这部分绿色的内容。总之，这部分，一路默认安装就可以。**

**1）图标组件(Addition icons) : 选择是否创建快速启动栏图标 或者 是否创建桌面快捷方式;  
2）桌面浏览(Windows Explorer integration) : 浏览源码的方法, 单独的上下文浏览 只使用bash 或者 只用Git GUI工具; 高级的上下文浏览方法 使用git-cheetah plugin插件;  
3）关联配置文件(Associate .git\*) : 是否关联git配置文件, 该配置文件主要显示文本编辑器的样式;  
4）关联shell脚本文件(Associate .sh) : 是否关联Bash命令行执行的脚本文件;  
5）使用TrueType编码 : 在命令行中是否使用TruthType编码, 该编码是微软和苹果公司制定的通用编码;**

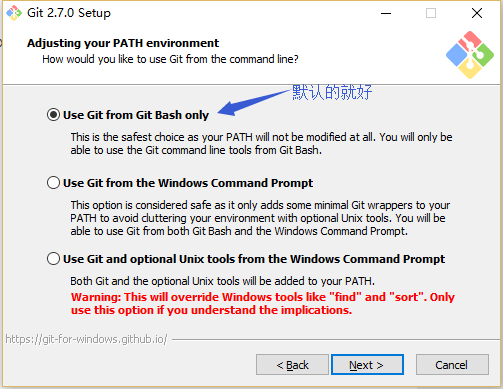
****

**设置开始菜单中快捷方式的目录名称，默认就好，下一步吧**

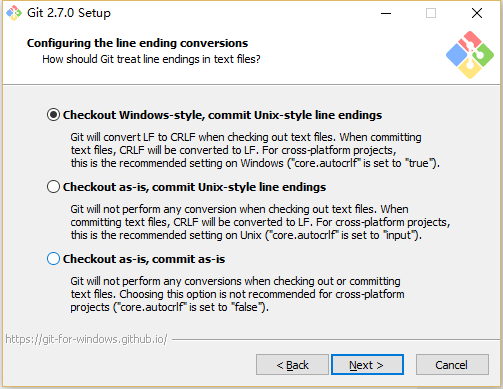
****

**设置环境变量 : 选择使用什么样的命令行工具, 一般情况下我们默认使用Git Bash即可, 默认选择;  
1）Git自带 : 使用Git自带的Git Bash命令行工具;  
2）系统自带CMD : 使用Windows系统的命令行工具;**

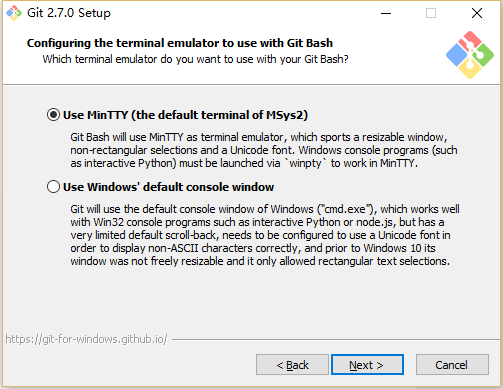
**3） 二者都有 : 上面二者同时配置, 但是注意, 这样会将windows中的find.exe 和 sort.exe工具覆盖, 如果不懂这些尽量不要选择;**

****

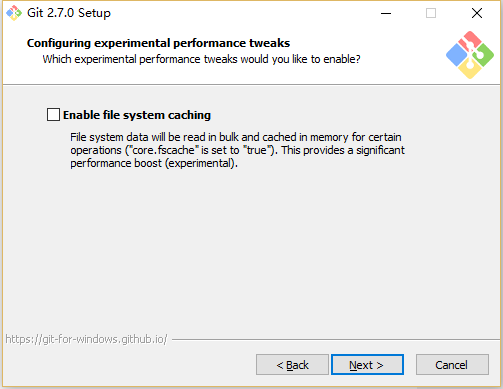
**选择换行格式 ，依然是默认就好。  
1）检查出windows格式转换为unix格式 : 将windows格式的换行转为unix格式的换行在进行提交;  
2）检查出原来格式转为unix格式 : 不管什么格式的, 一律转为unix格式的换行在进行提交;  
3）不进行格式转换 : 不进行转换, 检查出什么, 就提交什么;**

****

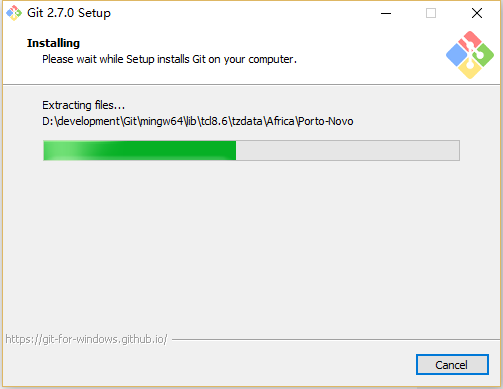
**选择终端模拟器，依然默认就好  
1）使用MinTTY，就是在Windows开了一个简单模拟Linux命令环境的窗口Git Bash  
2）使用windows的系统的命令行程序cmd.exe**

****

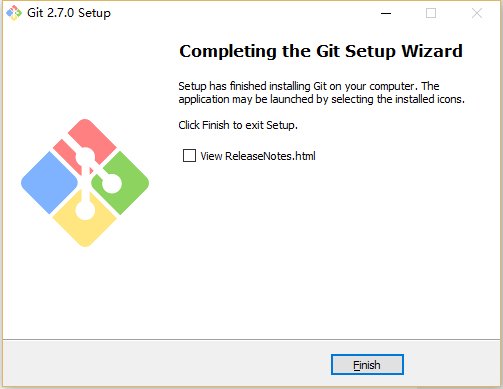
**选择默认就好，不用文件系统缓存**

****

**安装ing...**

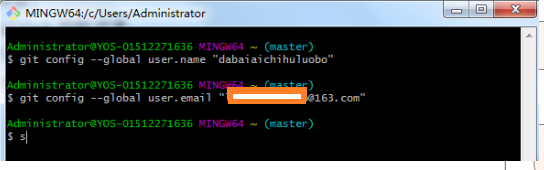
****

**安装成功**

****

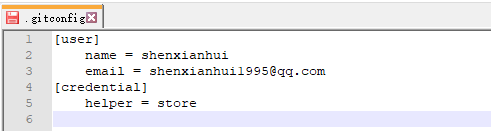
#### 3. 绑定用户

打开git-bash.exe，在桌面快捷方式/开始菜单/安装目录中  
因为Git是分布式版本控制系统，所以需要填写用户名和邮箱作为一个标识，用户和邮箱为**你github注册的账号和邮箱**。在程序中找到并打开git-bash.exe，依次输入如下指令：（输入后，没有任何消息，这就是最好的消息）  
git config --global user.name "Your Name"  
git config --global user.email "your\_email@example.com"  
注: [your\_email@example.com](https://links.jianshu.com/go?to=mailto:your_email@example.com" \t "https://www.jianshu.com/p/_blank) 的部分请改成你在创建账户时用的邮箱地址；**Your Name是你注册Github的账号，不是你电脑的名字。如果你所有的仓库都是用同一个账号密码，那么这个操作其实只要操作一次就可以，在安装好这个git-bash软件后，配置一次就可以。**



以下绿色部分可以跳过。

**这个命令会在 C:\Users\Administator （Administator为电脑用户名，每个人不同）目录下生成 .gitconfig 文件**

****

**PS: 下面代码是为了记住用户名和密码, 不用每次都输入, 需要的直接添加保存即可**

**[credential]**

**helper = store**

**当然, 还有其他方法保存, 可以直接在git bash 中执行命令：**

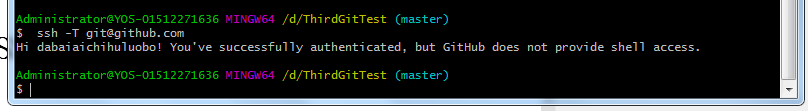
* **记住密码十五分钟  
  git config --global credential.helper cache**
* **自定义记住的时间 (单位: 秒)  
  git config credential.helper 'cache --timeout=3600'**
* **永久记住  
  git config --global credential.helper store**

**最后再输入一次账号密码就可以保存了**

测试是否成功？有时邮箱输错一个字母，而我们忽略了这一步测试一下，最后会造成后面remote远程连接时乱七八糟的错误。

在命令行输入：ssh -T git@github.com 如果用户名和邮箱都注册成功，会显示欢迎字符。

如下：



## 三. 为 Github 账户设置 SSH key

众所周知 ssh key 是加密传输。

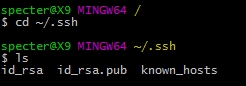
加密传输的算法有好多，git 使用 rsa，rsa 要解决的一个核心问题是，如何使用一对特定的数字，使其中一个数字可以用来加密，而另外一个数字可以用来解密。这两个数字就是你在使用 git 和 github 的时候所遇到的 public key 也就是公钥以及 private key 私钥。

其中，公钥就是那个用来加密的数字，这也就是为什么你在本机生成了公钥之后，要上传到 github 的原因。从 github 发回来的，用那公钥加密过的数据，可以用你本地的私钥来还原。

如果你的 key 丢失了，不管是公钥还是私钥，丢失一个都不能用了，解决方法也很简单，重新再生成一次，然后在 github.com 里再设置一次就行

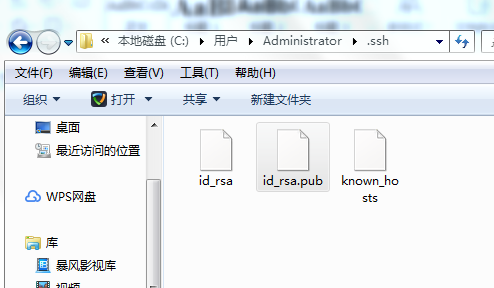
#### 1. 生成 SSH Key

首先检查是否已生成密钥 cd ~/.ssh  
ls 如果有3个文件，则密钥已经生成，id\_rsa.pub 就是公钥

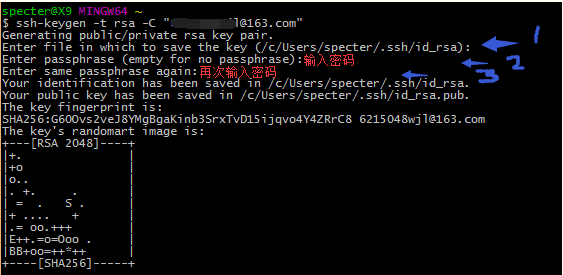


也可以打开我的电脑 C:\Users\Administator\.ssh 里面找到

C:\Users\Administrator\.ssh



如果没有生成，那么通过 $ ssh-keygen -t rsa -C "your\_email@example.com" 来生成。注: [your\_email@example.com](https://links.jianshu.com/go?to=mailto:your_email@example.com" \t "https://www.jianshu.com/p/_blank) 的部分请改成你在创建账户时用的邮箱地址。**一般第一次安装应该没有生成秘钥，需要我们自己生成秘钥。如下图所示，输入生成秘钥指令后提示如下：**



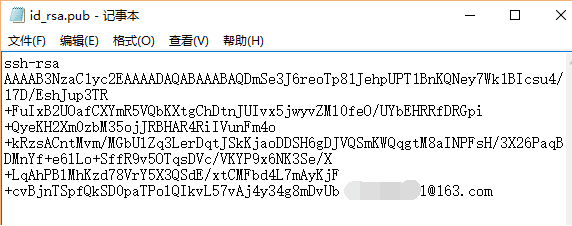
此时，注意上图中的1、2、3处含义分别如下：

1. 是路径确认，直接按回车存默认路径即可；  
   2）直接回车键，这里我们不使用密码进行登录, 用密码太麻烦;  
   3）直接回车键。

我用自己的电脑生成的秘钥是这样的：



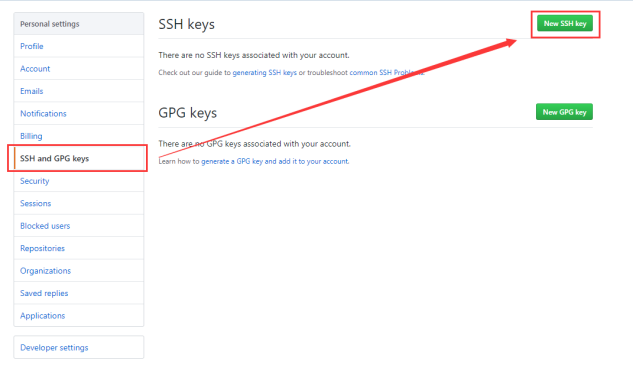
生成成功后，去对应目录 C:\Users\Administator\.ssh （Administator为电脑用户名，每个人不同）用记事本打开 id\_rsa.pub，得到 ssh key公钥



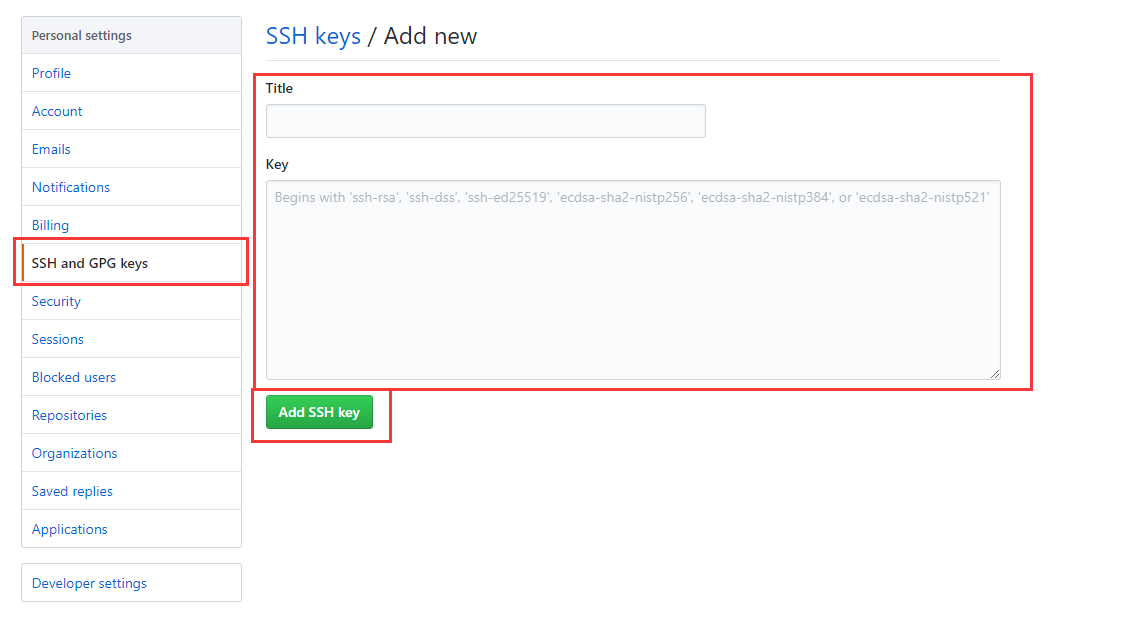
#### 2. 为 github 账号配置 ssh key

切换到 github，展开个人头像的小三角，点击settings

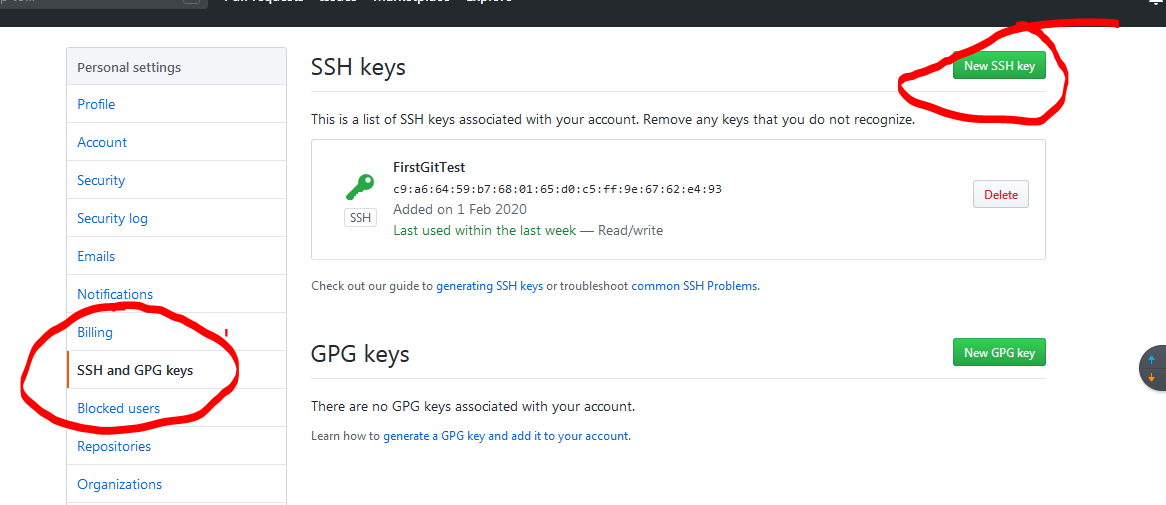
然后打开 SSH and GPG keys 菜单， 点击 New SSH key 新增密钥



标题title建议和仓库名一致, 便于区分. 接着将 id\_rsa.pub 文件中 key 粘贴到key处，最后 Add SSH key 生成密钥



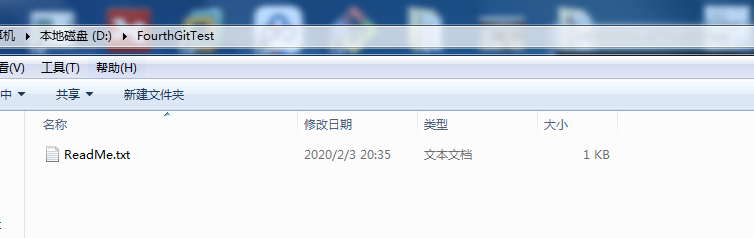
如此，github账号的SSH keys配置完成



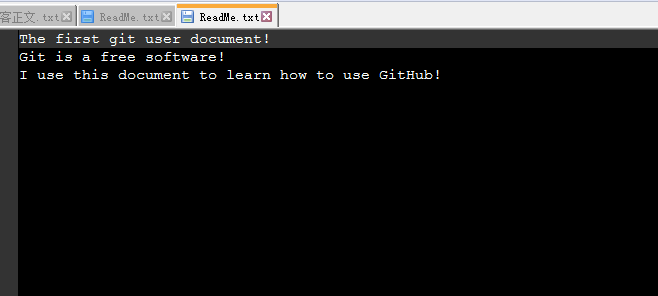
## 四. 上传本地项目到 github

#### 1.创建一个本地仓库

即新建一个文件夹，里面放入一个文本文档模拟代码。

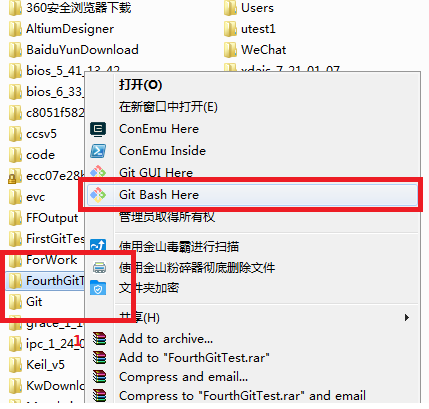


ReadMe.txt内容如下：

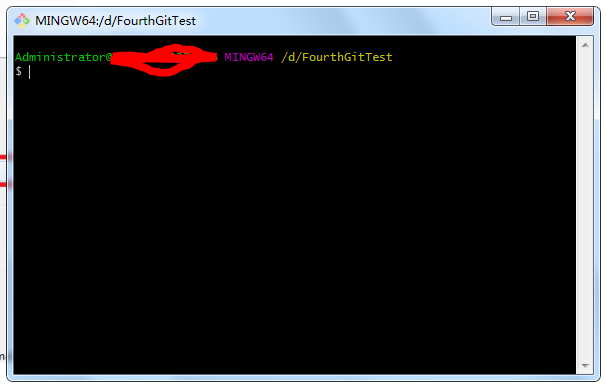


#### 2.建立本地仓库

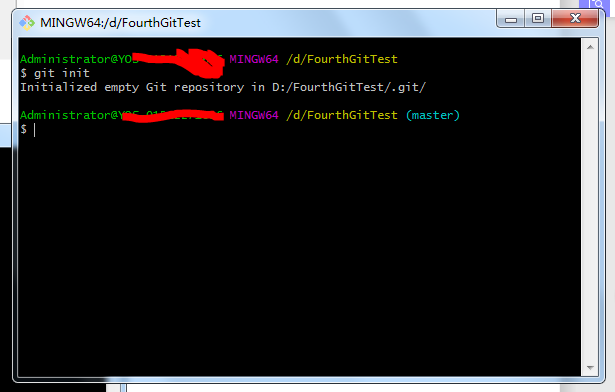
首先，选中文件夹，右键选择git bash here



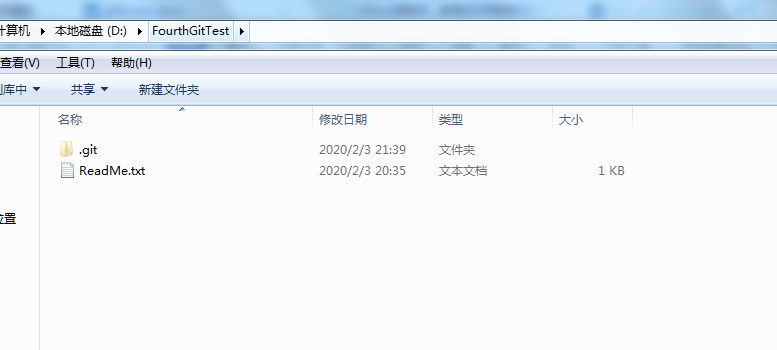
弹出如下界面：



输入git init

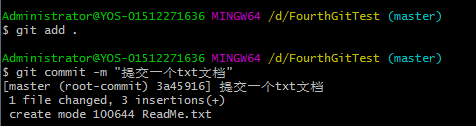


初始化成功后你会发现项目里多了一个隐藏文件夹.git  
这个目录是Git用来跟踪管理版本库的，没事千万不要手动修改这个目录里面的文件，不然改乱了，就把Git仓库给破坏了。



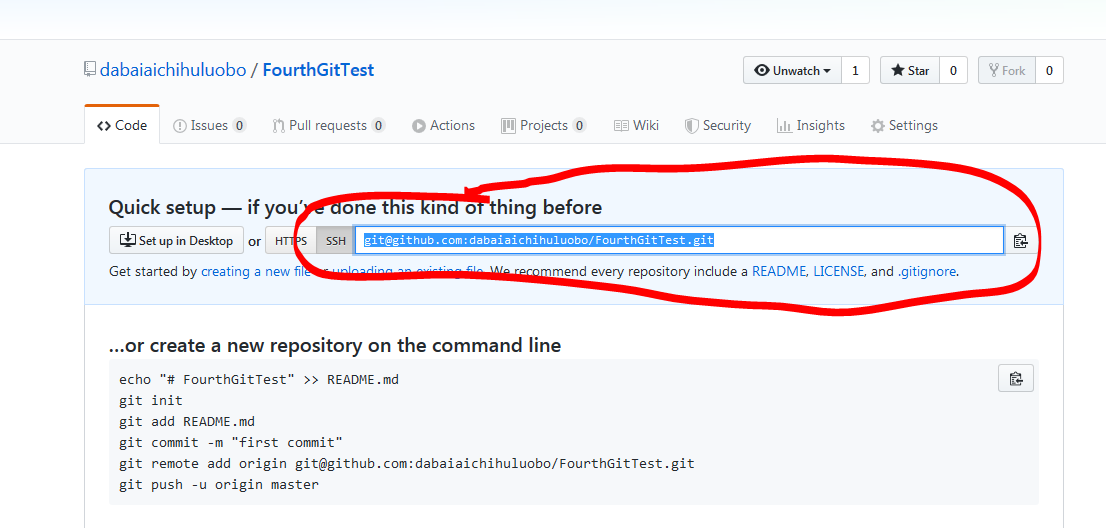
接着，将所有文件添加到仓库  
执行指令：git add .

然后把文件提交到仓库，双引号内是提交注释  
执行指令：git commit -m "提交文件"，这两步的结果如下图所示：

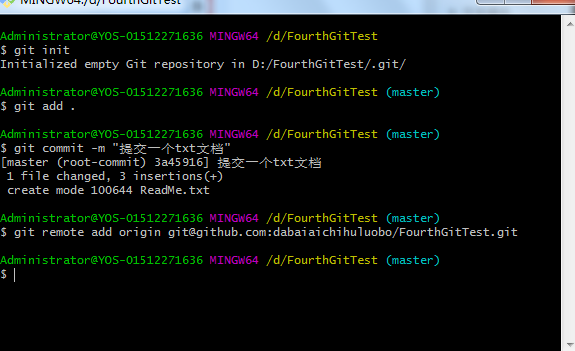
  
如此本地仓库建立好了。

#### 3. 关联 github 仓库

到 github 仓库复制仓库地址

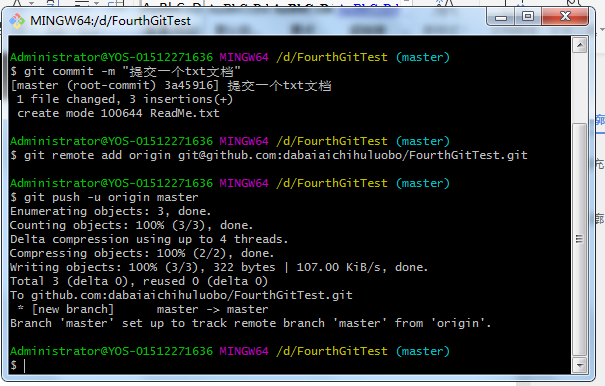


然后执行指令：git remote add origin git@github.com:dabaiaichihuluobo/FourthGitTest.git，执行完毕后没有任何消息，这就是最好的消息。



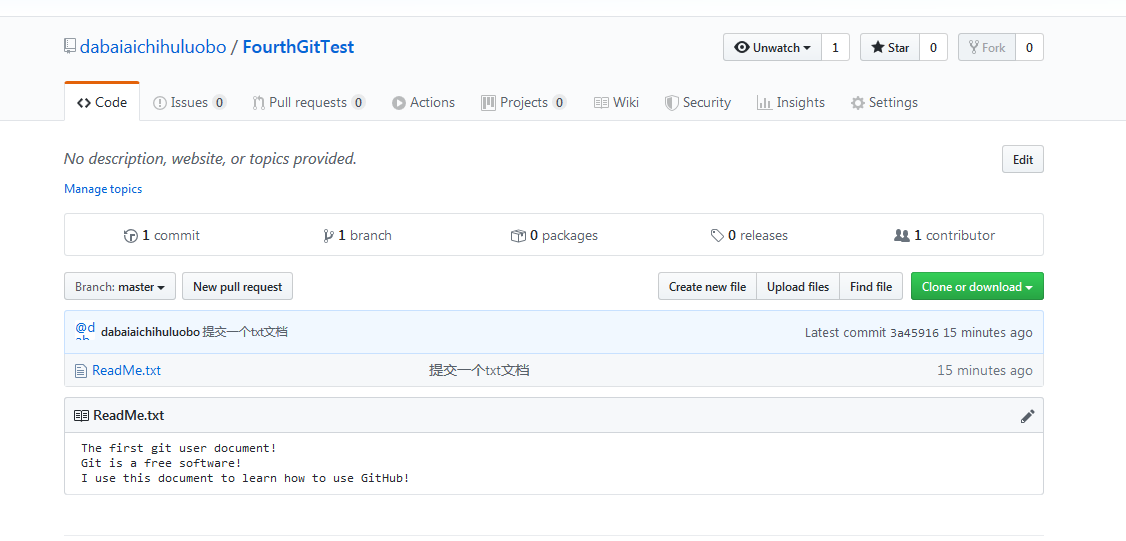
#### 4. 上传本地代码

执行指令：git push -u origin master，等待大概5秒钟左右，有回令了，毕竟我们往远程上传东西了，如下：



到此，本地代码已经推送到github仓库了，我们现在去 github 仓库看看。

当当当当，多了一个东西，如下：



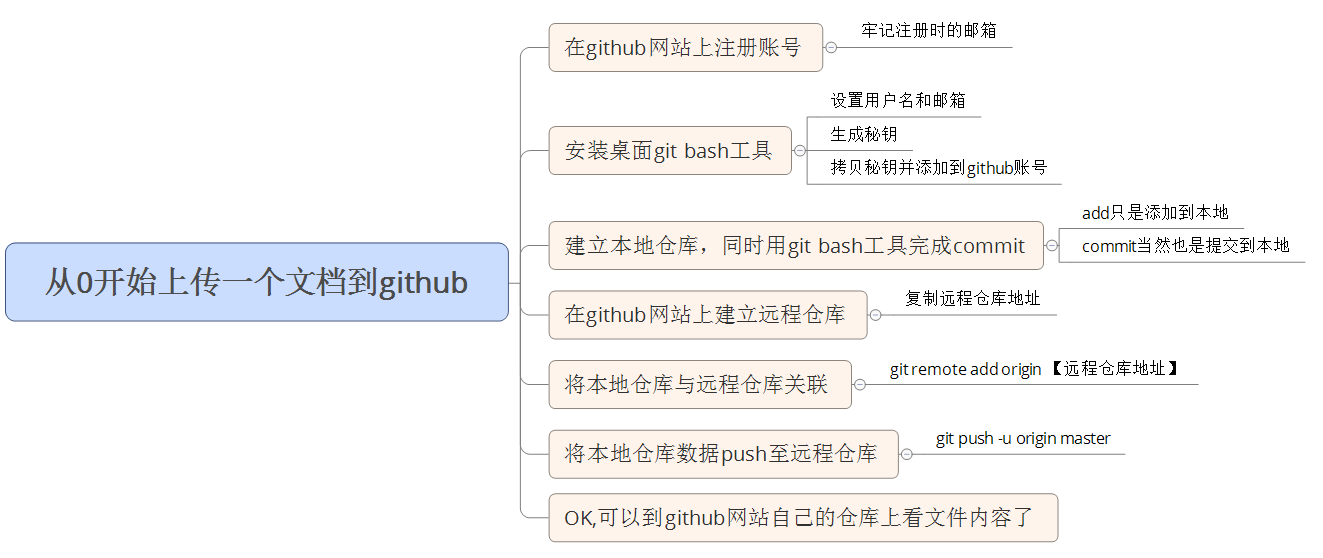
我们完成了文件的上传，是的，上传一个txt文档成功了。

一般情况下，你也不知道自己的文件发生了多少变化，你可以输入git status，会告诉你有哪些变化，然后你输入git commit -a -m "your message“， 把所有变化提交到本地库，继续git push origin master，就更新到远程库了。

注意：一般的变化都不要push，只有最终感觉差不多了，再把文件push上去。

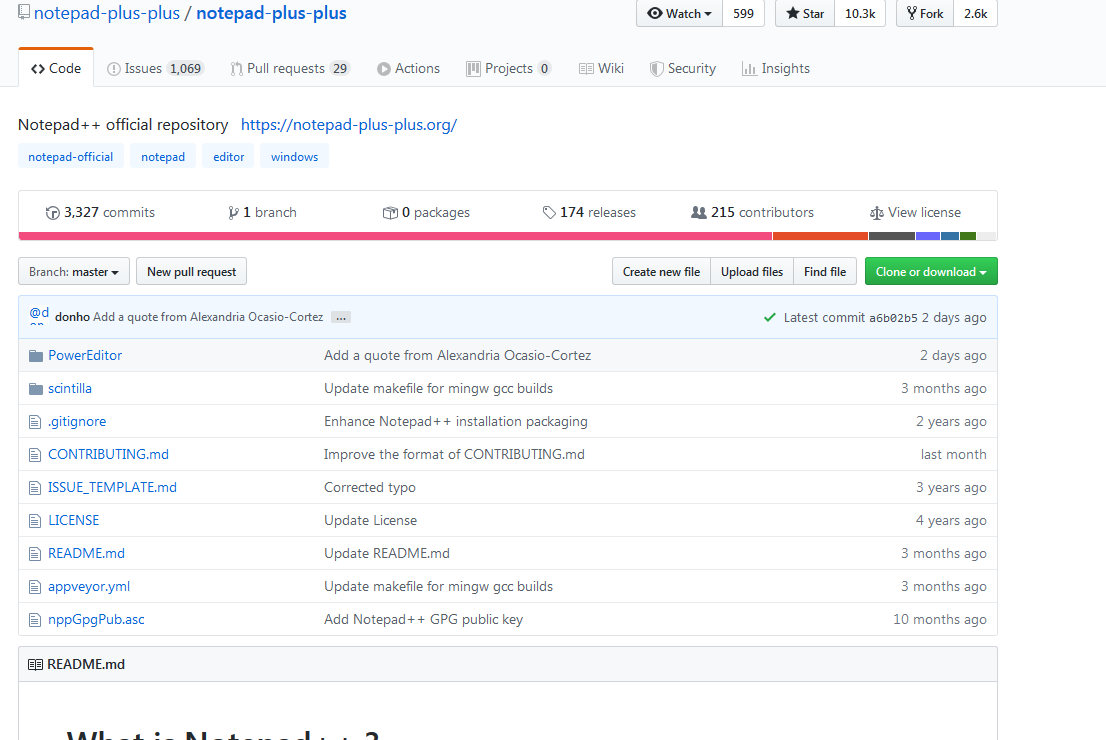
原作者说git是不能管理空的文件夹的，文件夹里必须有文件才能add。具体不细说了，可以参见原文。

电影看到末尾才有彩蛋，这个彩蛋都是留给有耐心的人，恩，把这个阶段的学习的过程梳理一下贴在下面：说来惭愧，第一次用xmind，本想画一个一步一步的流程图，没有找到，就用这个代替吧。从上到下，是操作的顺序。

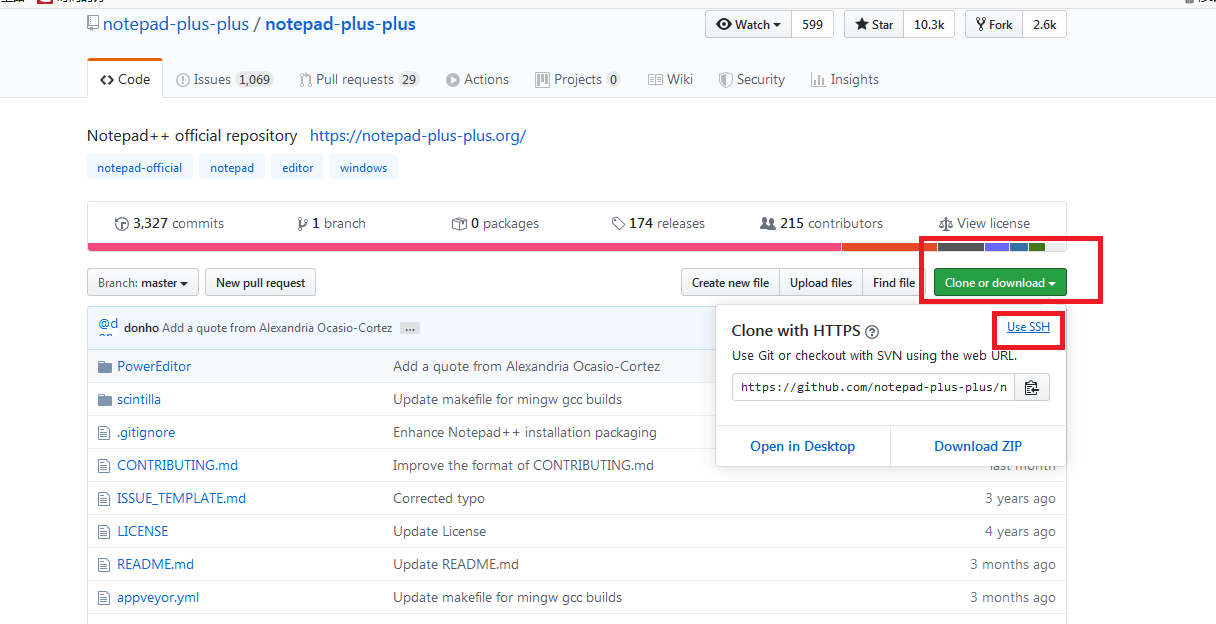


## 克隆其他库

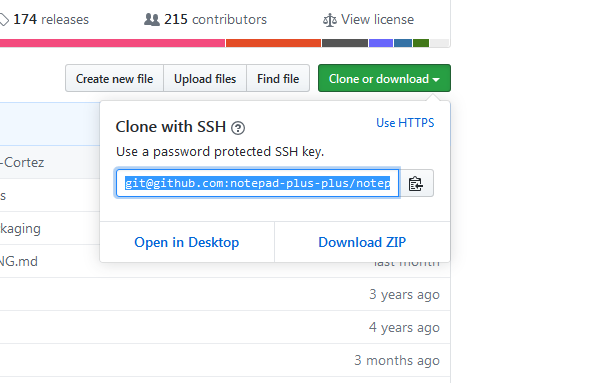
以大名鼎鼎的notepad++为例，其源代码在这：



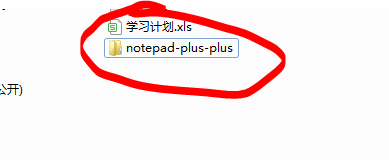
右上角附近点击clone or download，选择Use SSH



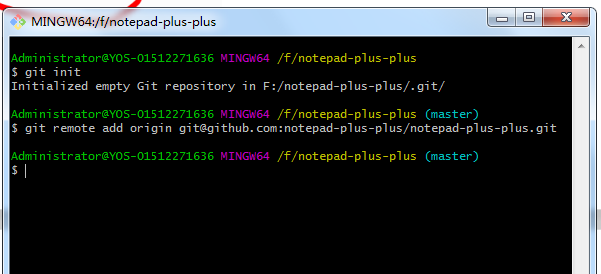
然后把这个仓库地址复制



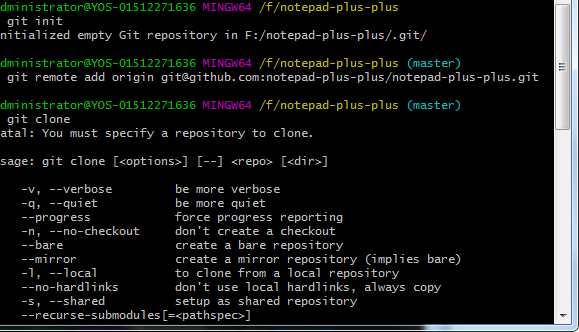
在你的电脑上新建一个与人家库名相同的文件夹，



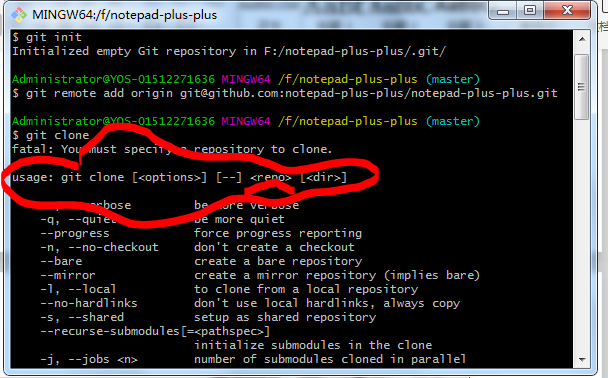
然后在文件夹上右击，在弹出菜单上选git bash，打开命令行，输入git init，初始化库，然后输入git remote add origin ,右击命令行窗口标题栏，在弹出菜单上选“编辑”->"粘贴"，回车，就建立了联系。



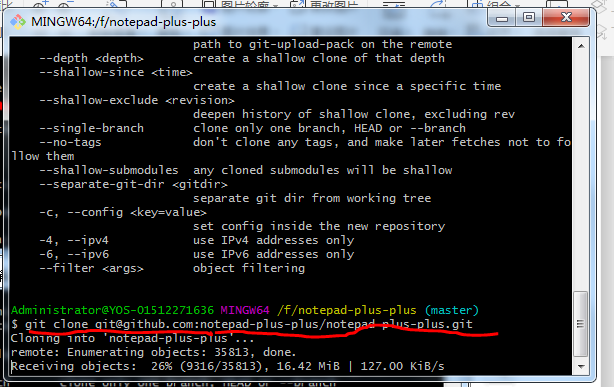
第三步，按照某位大神的教程，输入git clone。什么？竟然又报错了？

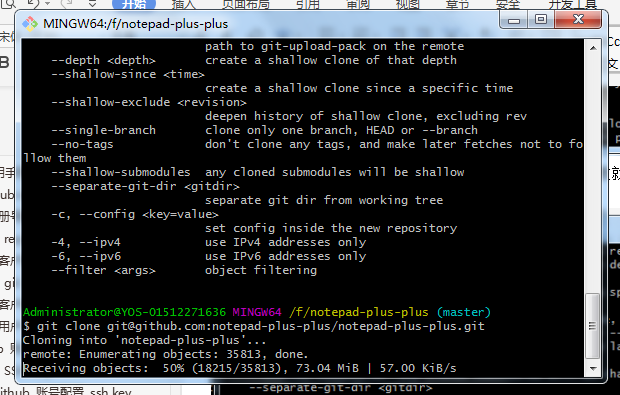


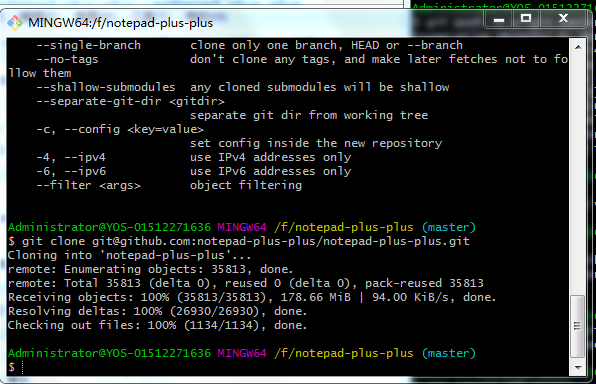
仔细看了下指令帮助中的clone指令的参数，



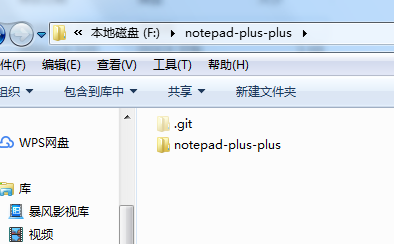
于是，灵机一动，clone指令可能得需要带个参数，这个参数就是远程网上仓库的地址，试试，果然，开始下载了，只是，这个速度也太慢了。

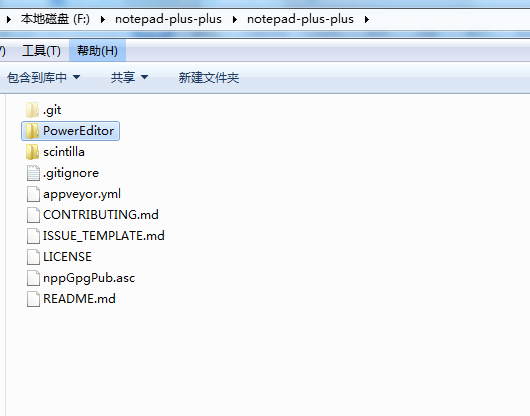


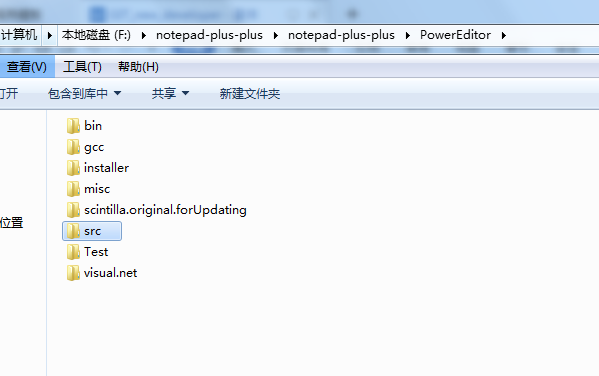


终于，

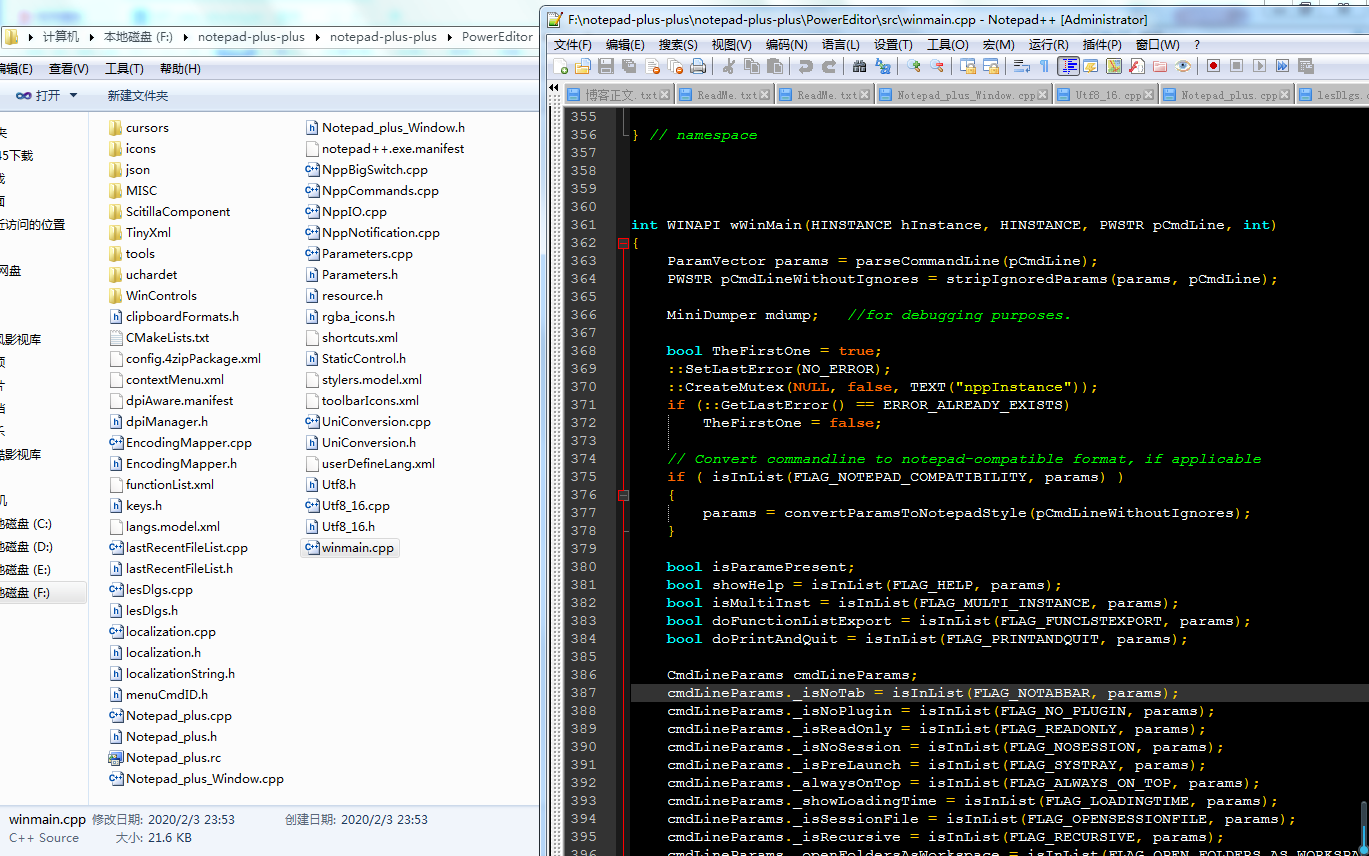
打开自己本地的仓库







终于看到代码了，看到代码的那一刻，顿时觉得自己会的那点MFC简直就是小儿科。



至此，克隆其他库完成。