

# **Git-2.18.0-64-bit 安装教程及 Git Bash 新手使用步骤**

制作整理人：田一蒙

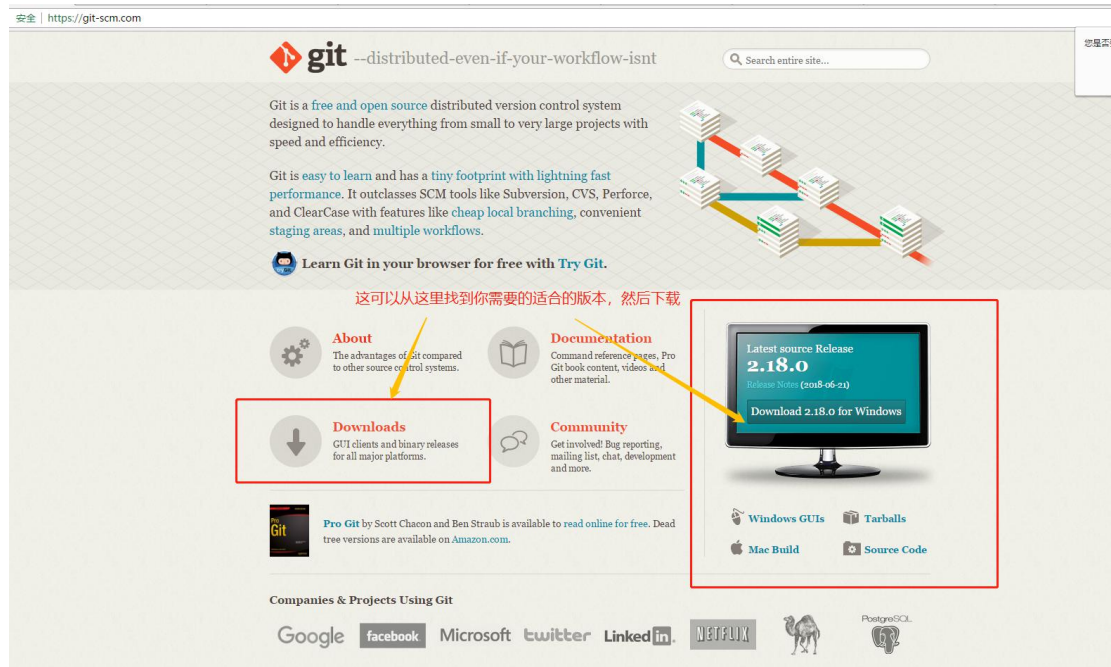
参考资料：网上 CSDN 各个关于 GIT 的文章。

整理时间：2018 年 8 月

## 目录

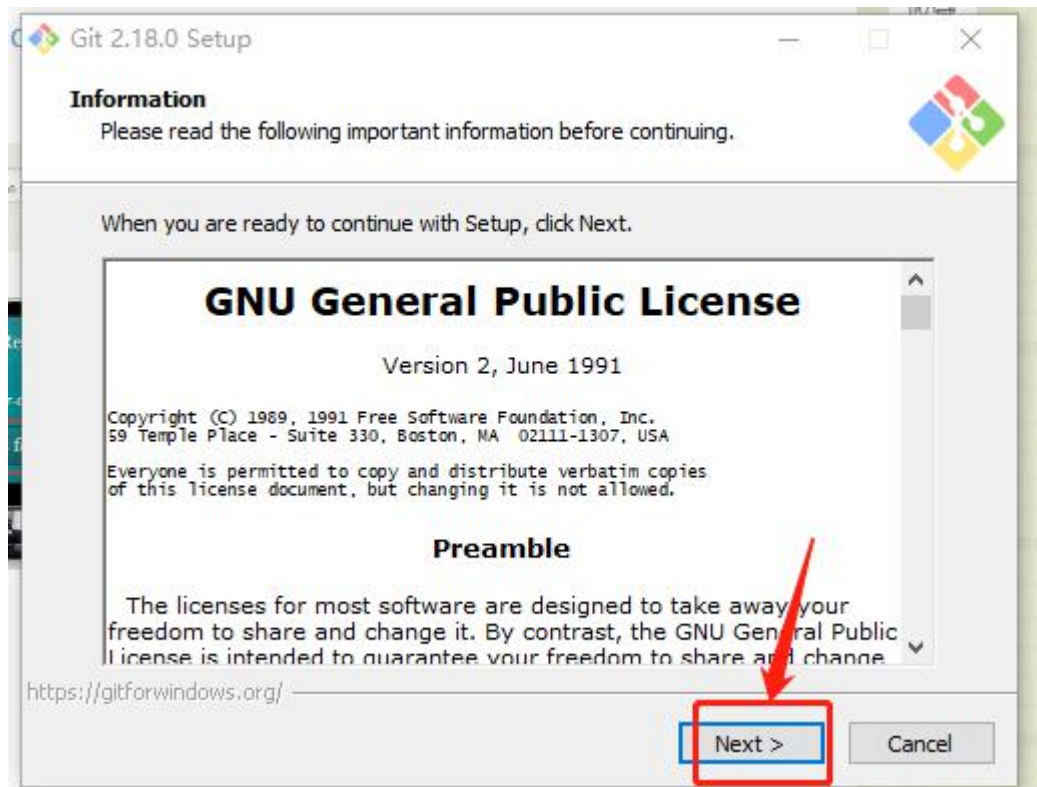
1. git-scm.com 首先进入 GitHub 官网，下载适合自己电脑的版本	3
2. 双击安装程序，进入欢迎界面点击，阅读相关协议，点击【Next >】	3
3. 选择安装位置，点击【Next >】	4
4. 选择安装组件：这里可以选项设置，设置完成后点击【Next >】	5
5. Git 2.18.0 设置选择 Git 所使用的默认编辑器哪个编辑器	6
6. 设置环境，选择使用什么样儿的命令行工具，一般情况我们使用默认配置，使用 Git Bash	7
7. 设置 HTTPS 传输加密方式	8
8. 选择换行格式，	9
9. 配置 Git bash 终端仿真器	10
10. 性能配置，是否启用文件系统缓存	11
11. 安装完成，点击【Finish】	12
12. 配置 github 的 ssh 密钥:	13
(1) 打开 Git Bash 查看电脑上是否已经存在 SSH 密钥:	13
(2) 创建新的 ssh key:	13
(3) 测试 ssh 链接 github:	15
(4) 设置 username 和 email	15
13. 克隆在 Github 上创建的库至本地电脑中，方便以后进行上传代码。	16
(1) Git bash 的定位	17
(2) 输入 git clone “你在 github 上创建成功的库网址”	17
(3) 打开之前定位的 D 盘，如下图可以看到已经有以我库名所创建的文件夹了。	18
(4) 打开该文件夹，创建任意一个格式的，任意名称的文件，本人在在这里创建的是一个.txt 格式的文件。	18
(5) 重新定位 Git bash 的位置，定位在我们库的文件夹，输入 ls 查看你目前所定位的文件夹中的文件，可发现创建的文件已存在。如下图:	19
(6) 输入 git add newStudent.txt 后回车	19
(7) 在输入 git commit -m “这是我写的备注” 引号内的内容可以随便写，目的是方便查找区分。如下图:	19
(8) 输入 git push origin master 后会出下面的窗口，在这里登陆你的 GitHub 账号之后点击 login	20
(9) 验证	21

## 1.git-scm.com 首先进入 GitHub 官网，下载适合自己电脑的版本

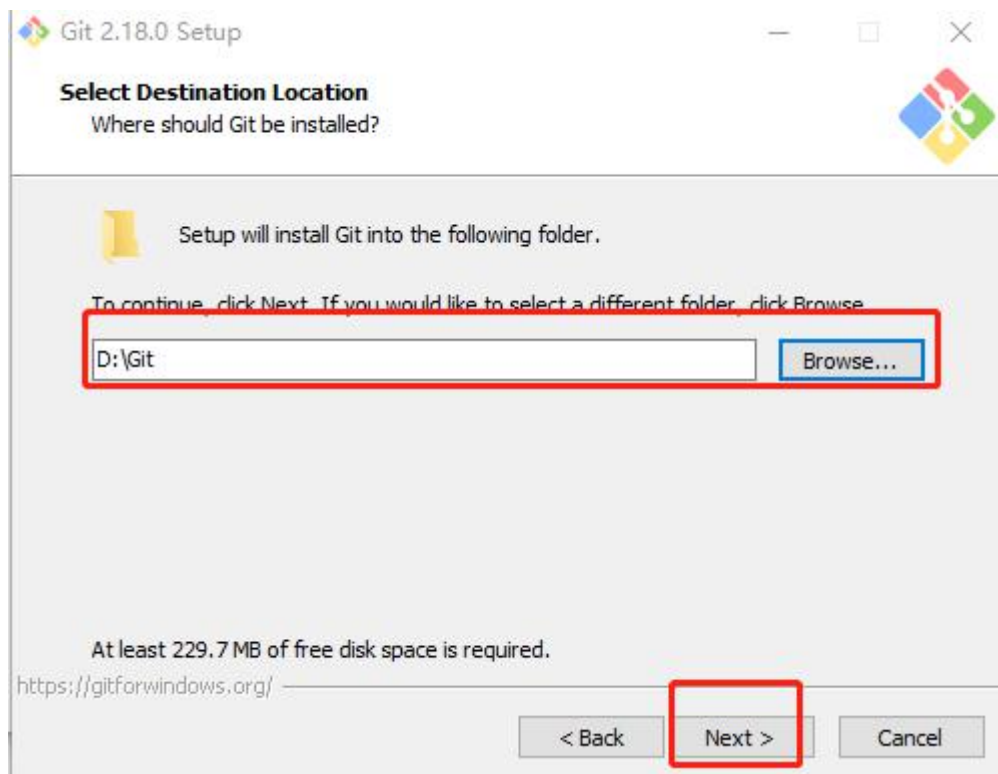


## 2.双击安装程序，进入欢迎界面点击，阅读相关协议 ，点击【Next >】





### 3.选择安装位置，点击【Next >】



## 4. 选择安装组件：这里可以选项设置，设置完成后点击【Next >】

图标组件（Additional icons）：选择是否创建桌面快捷方式

桌面浏览（Windows Explorer integration）

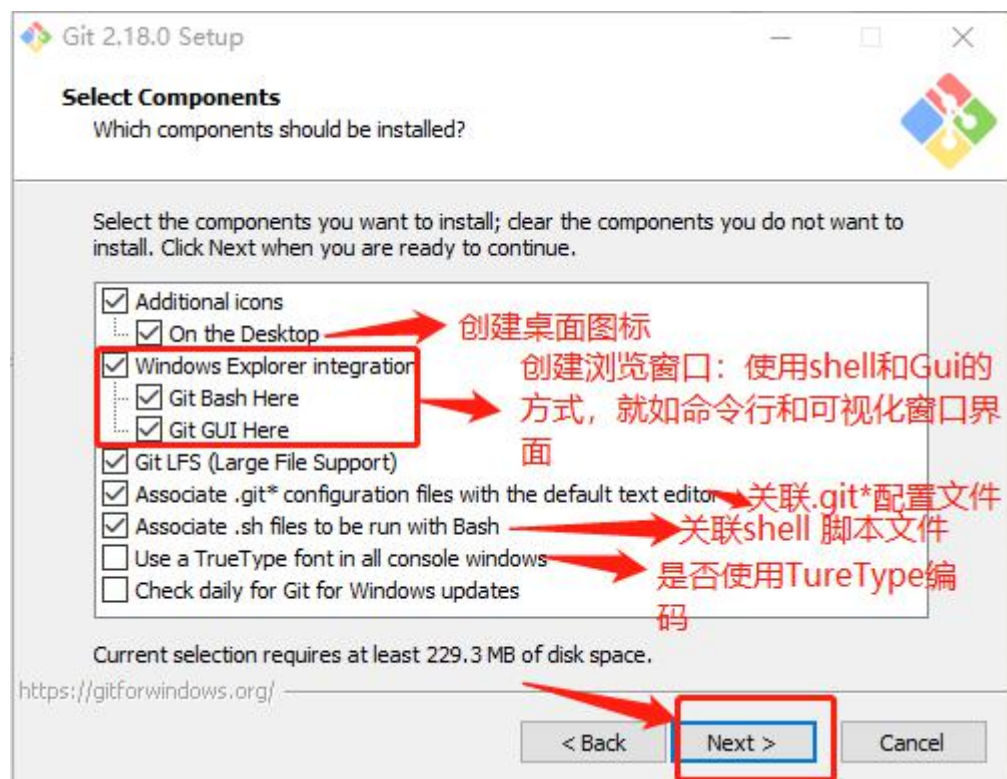
使用 Git Bash 方式，shell 方式

受用桌面程序方式

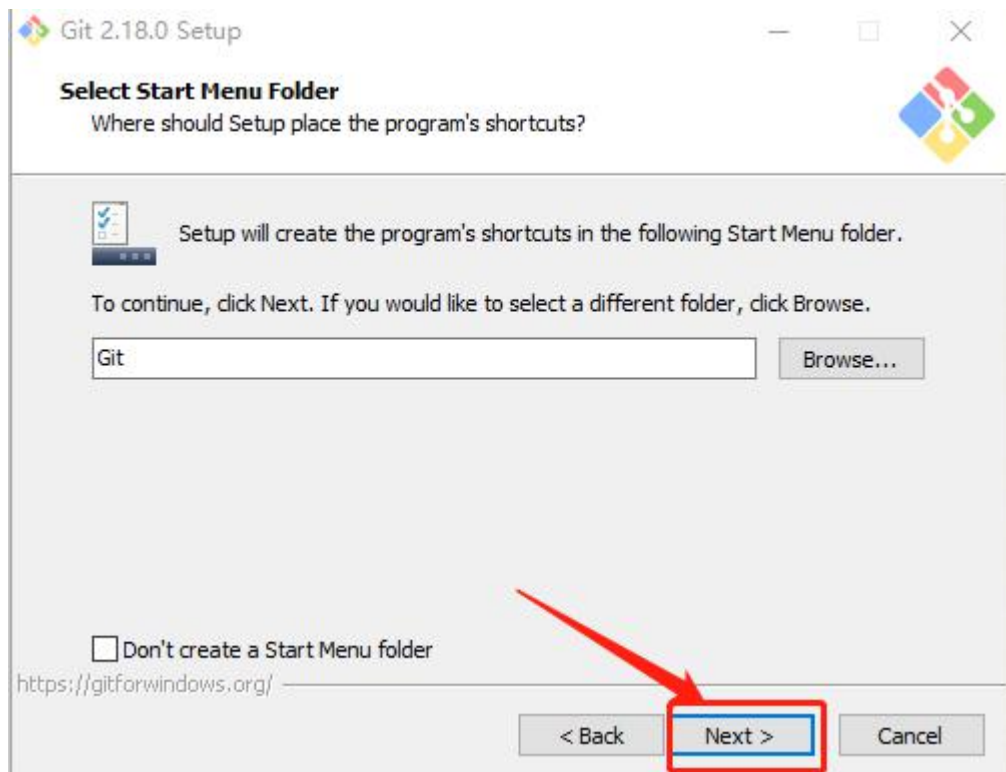
关联配置文件：是否关联 git 配置文件，该配置文件主要显示文本编辑器样式

关联 shell 脚本文件：是否关联 Bash 命令执行脚本文件

使用 TrueType 编码：在命令行中是否使用 TrueType 编码，该编码是微软和苹果公司制定的通用编码



是否创建开始菜单快捷方式目录，点击【Next >】

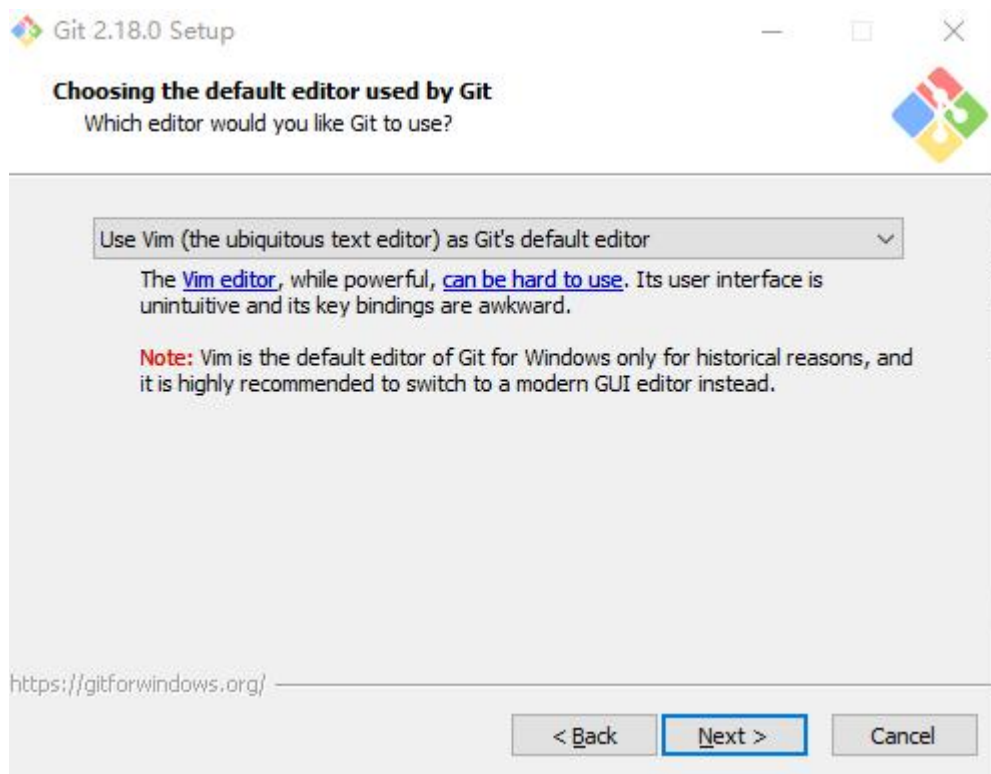


## 5. Git 2.18.0 设置选择 Git 所使用的默认编辑器哪个编辑器

使用 Vim（ubiautext 文本编辑器）作为 Git 的默认编辑器

Vim 编辑器虽然功能强大，但却很难使用。它的用户界面是 uhintuitive，它的关键 bindinas 很笨拙。

注意：出于历史原因，Vim 是 Windows Git 的默认编辑器，因此强烈建议切换到现代 GUI 编辑器。



本人在这里仍是选用默认的 Vim 编辑器。

点击【Next >】

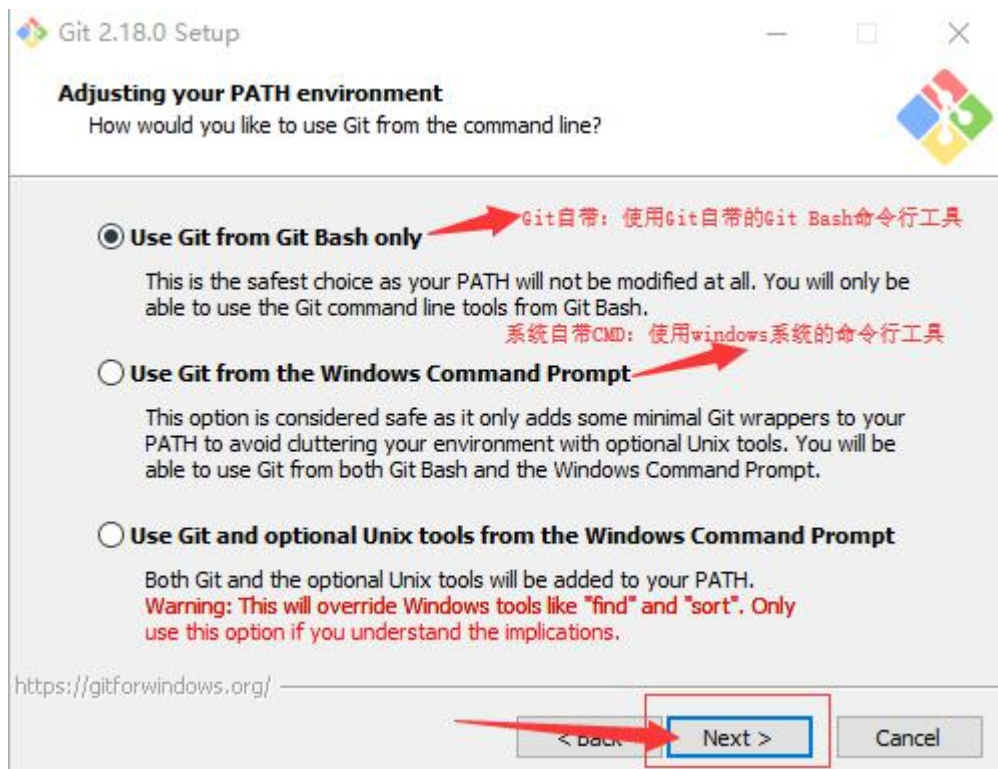
## 6.设置环境，选择使用什么样儿的命令行工具，一般情况我们使用默认配置，使用 Git Bash

Git 自带：使用 Git 自带的 Git Bash 命令行工具

系统自带 CMD：使用 windows 系统的命令行工具

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





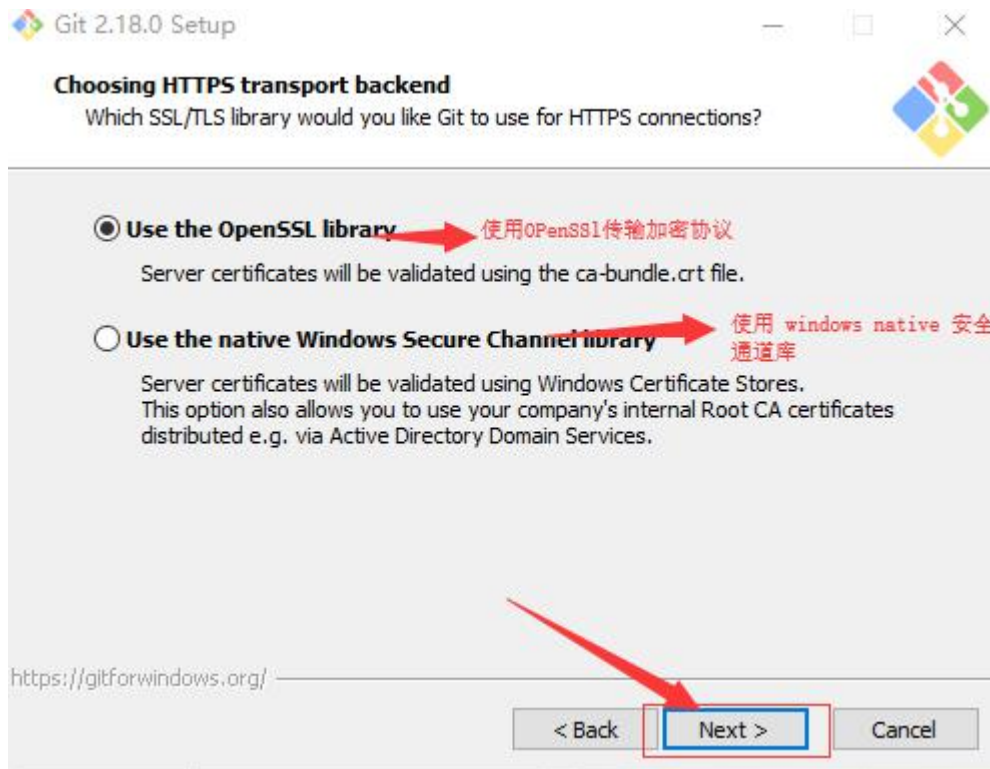
这里本人选择 Git 自带的命令行工具，选择后，点击【Next >】

## 7.设置 HTTPS 传输加密方式

使用 OpenSSL 库

使用本机 Windows 安全通道库





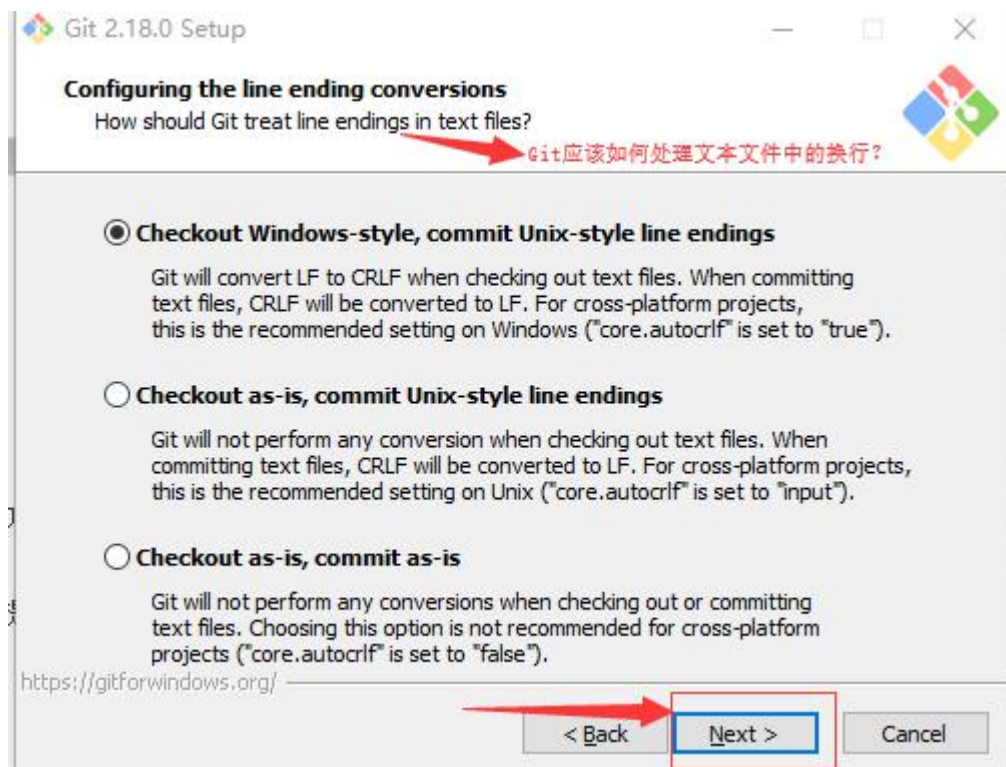
这里本人选择第一个，而不是本机 Windows ，选择后点击【Next >】

## 8.选择换行格式，

让 Git 能够自动转换文件中的换行符：签出到本地时转换为 Windows 下的换行符，提交到服务器时转换为 Unix 下的换行符

让 Git 在签出到本地时不做转换，保留原始文件的换行符；提交到服务器时转换为 Unix 下的换行符

让 Git 在签出到本地时和提交到服务器时都不做转换



本人这里选择第一个，选择后，点击【Next >】

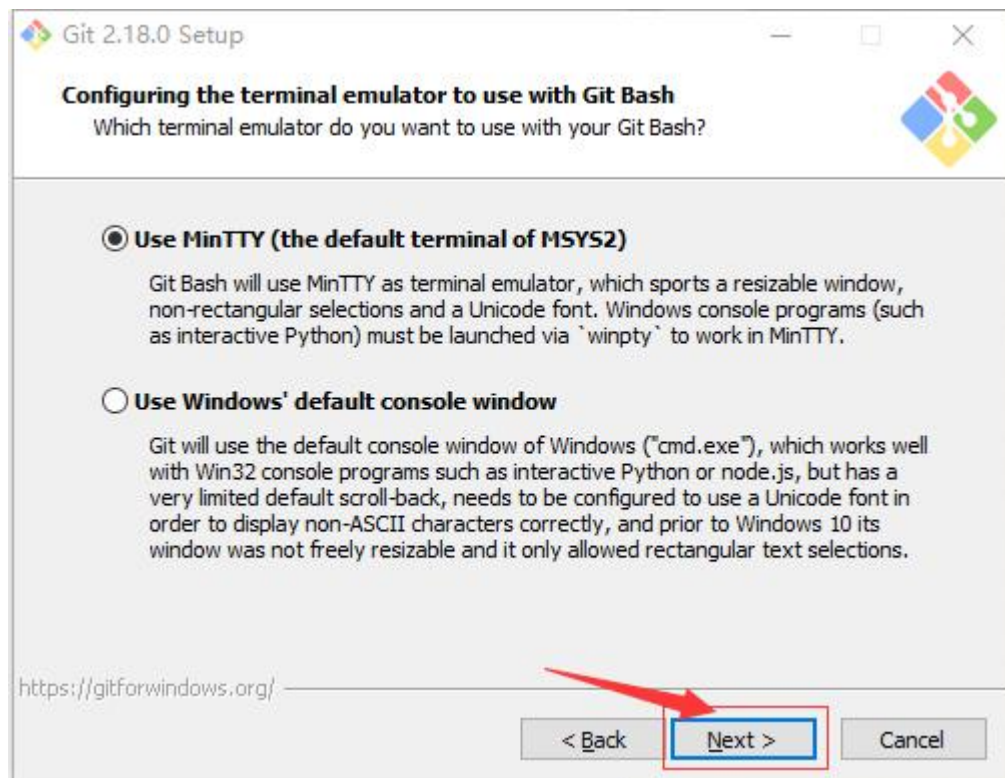
## 9.配置 Git bash 终端仿真器

使用 MinTTY 终端

使用 MinTTY（MSYS2 的默认终端）Git Bash 将使用 MinTTY 作为终端模拟器，它具有可调整的窗口。非 rectangular 选择和 Unicode 字体。Windows 控制台程序（如交互式 Python）必须通过“winpty”启动，以便在 MinTTY 中工作。

使用 windows 默认的命令

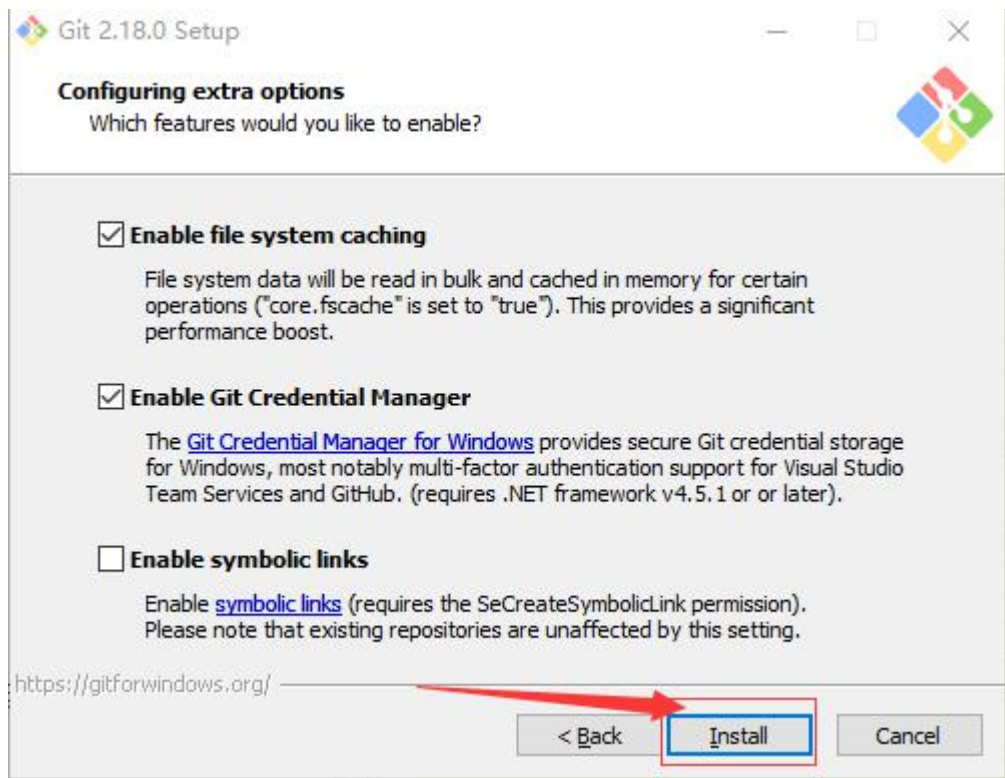
使用 Windows 的默认控制台窗口。Git 将使用 Windows 的默认控制台窗口（“cmd.exe”），它与诸如交互式 Python 或 node 之类的 Win32 控制台程序很好地工作。但是有一个非常有限的默认回滚，需要使用 Unicode 字体来显示非 ascii 字符，在 Windows 10 之前，它的窗口是不可自由调整的，它只允许使用 rectangular 文本选择。



本人这里默认选择第一个，选择后，点击【Next >】

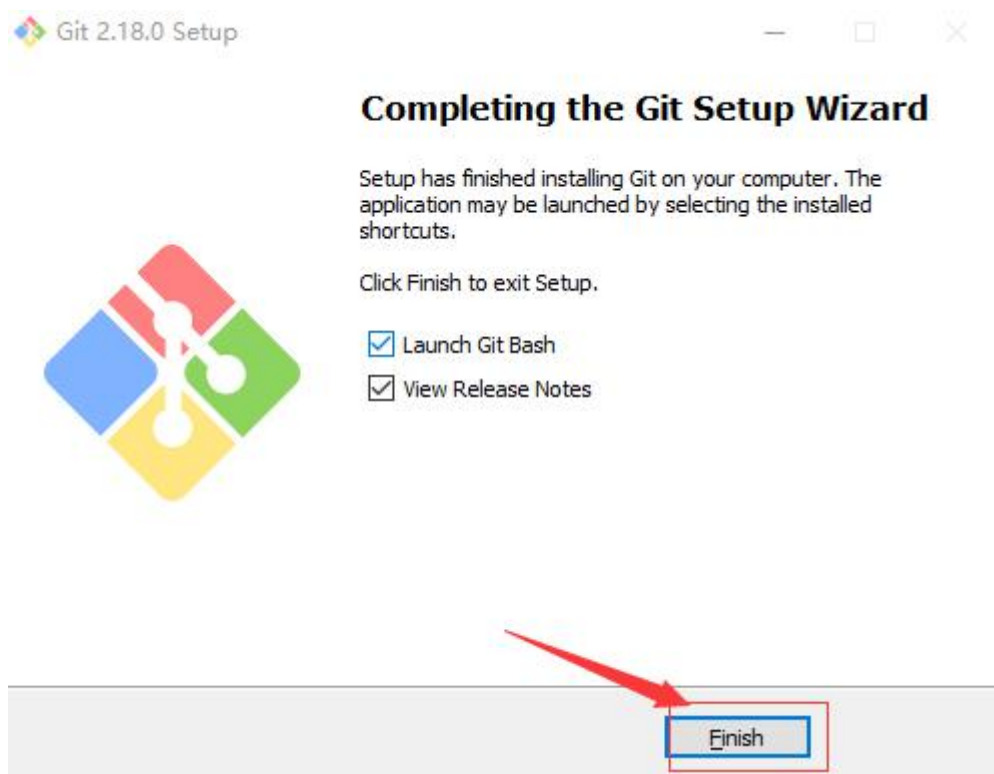
## 10.性能配置，是否启用文件系统缓存

- (1) 使文件系统缓存文件系统数据将被批量读取，并在内存中缓存在某些操作（“核心”。fscache“被设置为true”）。这提供了显著的性能提升。
- (2) 使用Git证书管理器 Windows的Git凭证管理器为Windows提供了安全的Git凭据存储。最明显的是对Visual Studio Team Services和GitHub的多因素身份验证支持。（要求，NET framework v4.5.1或更高版本）。
- (3) 支持符号链接启用符号链接（需要分泌的秘密通道）。请注意，现有存储库不受此设置的影响。



本人这里选择默认勾选的前两个，选择后，点击【Install >】

## 11.安装完成，点击【Finish】



环境变量配置

安装成功后需要配置 Git 环境变量「注意该步骤为 Git 在 windows cmd 命令中配置，如果不配置，直接使用 Git Bash 即可」

在 Path 变量中增加：D:\Git\cmd

验证是否配置成功，打开 windows 命令行，输入 git version 命令，出现下列信息表示配置成功。



```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows [版本 10.0.16299.547]
(c) 2017 Microsoft Corporation。保留所有权利。

C:\Users\guess>git version
git version 2.18.0.windows.1

C:\Users\guess>
```

## 12.配置 github 的 ssh 密钥:

### (1)打开 Git Bash 查看电脑上是否已经存在 SSH 密钥:

输入 `cd ~/.ssh`

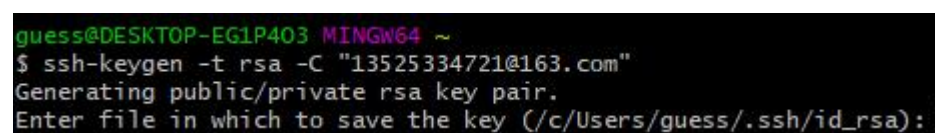


```
MINGW64:/c/Users/guess
guess@DESKTOP-EG1P403 MINGW64 ~
$ cd ~/.ssh
bash: cd ~/.ssh: No such file or directory
```

若如上图显示无法找到该文件则要创建新的 ssh key;

### (2)创建新的 ssh key:

输入 `ssh-keygen -t rsa -C "your_email@youremail.com"`



```
guess@DESKTOP-EG1P403 MINGW64 ~
$ ssh-keygen -t rsa -C "13525334721@163.com"
Generating public/private rsa key pair.
Enter file in which to save the key (/c/Users/guess/.ssh/id_rsa):
```

执行这条命令会如上图提示文件保存路径，可以直接按 Enter，  
然后提示输入 passphrase（密码），输入两次（可以不输直接两次 Enter），

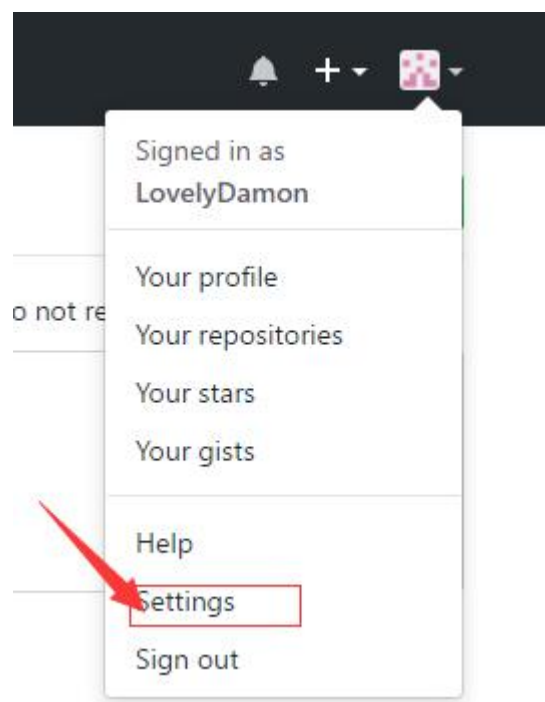


然后会在 .ssh 目录生产两个文件：id\_rsa 和 id\_rsa.pub

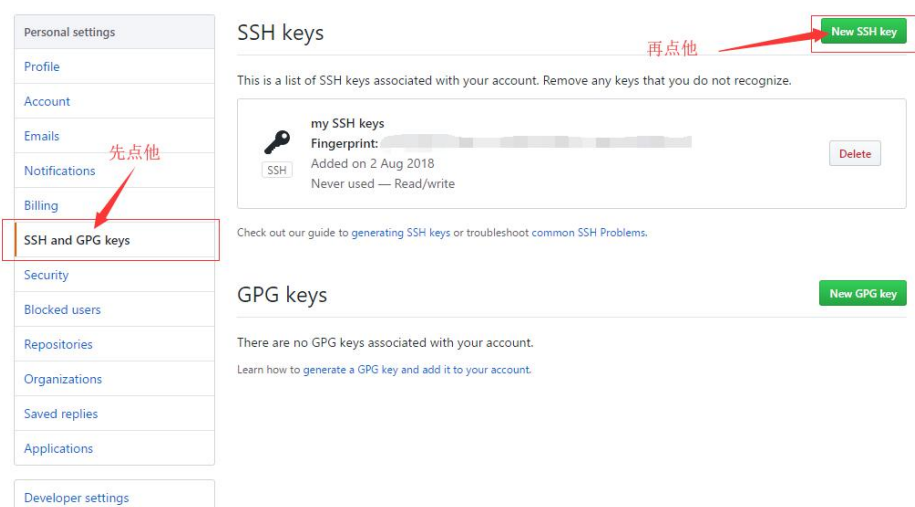
用记事本打开 .ssh 目录下的 id\_rsa.pub 文件，复制里面的内容；

复制 ssh key 到 github：

登录 Github, 找到右上角的图标，打开点进里面的 Settings



再选中里面的 SSH and GPG KEYS，点击右上角的 New SSH key，然后 Title 里面随便填，再把刚才 id\_rsa.pub 里面的内容复制到 Title 下面的 Key 内容框里面，最后点击 Add SSH key，这样就完成了 SSH Key 的加密。



## SSH keys / Add new

Title

标题自己看着随便写，也可以不填

Key

Begins with 'ssh-rsa', 'ssh-dss', 'ssh-ed25519', 'ecdsa-sha2-nistp256', 'ecdsa-sha2-nistp384', or 'ecdsa-sha2-nistp521'

把你复制的内容粘贴到这里

最后点击完成创建

Add SSH key

### (3)测试 ssh 链接 github:

输入 `ssh -T git@github.com`

```
guess@DESKTOP-EG1P403 MINGW64 ~  
$ ssh -T git@github.com  
The authenticity of host 'github.com (52.74.223.119)' can't be established.  
RSA key fingerprint is SHA256:nThbg6kXUpJWG17E1IGOCspRomTxdCARLviKw6E5SY8.  
Are you sure you want to continue connecting (yes/no)? yes  
Warning: Permanently added 'github.com,52.74.223.119' (RSA) to the list of known  
hosts.  
Hi LovelyDamon! You've successfully authenticated, but GitHub does not provide s  
hell access.
```

如果是第一次的会提示是否 continue，输入 yes，看到 successfully 就说明可以了。

### (4) 设置 username 和 email

输入

`git config --global user.name "Firstname Lastname"` (此处 name 可修改也不是用于登录 github 的登录名)

`git config --global user.email "your_email@youremail.com"`



```
guess@DESKTOP-EG1P403 MINGW64 ~  
$ git config --global user.name"newStudentTian"  
  
guess@DESKTOP-EG1P403 MINGW64 ~  
$ git config --global user.email"13525334721@163.com"  
  
guess@DESKTOP-EG1P403 MINGW64 ~  
$
```

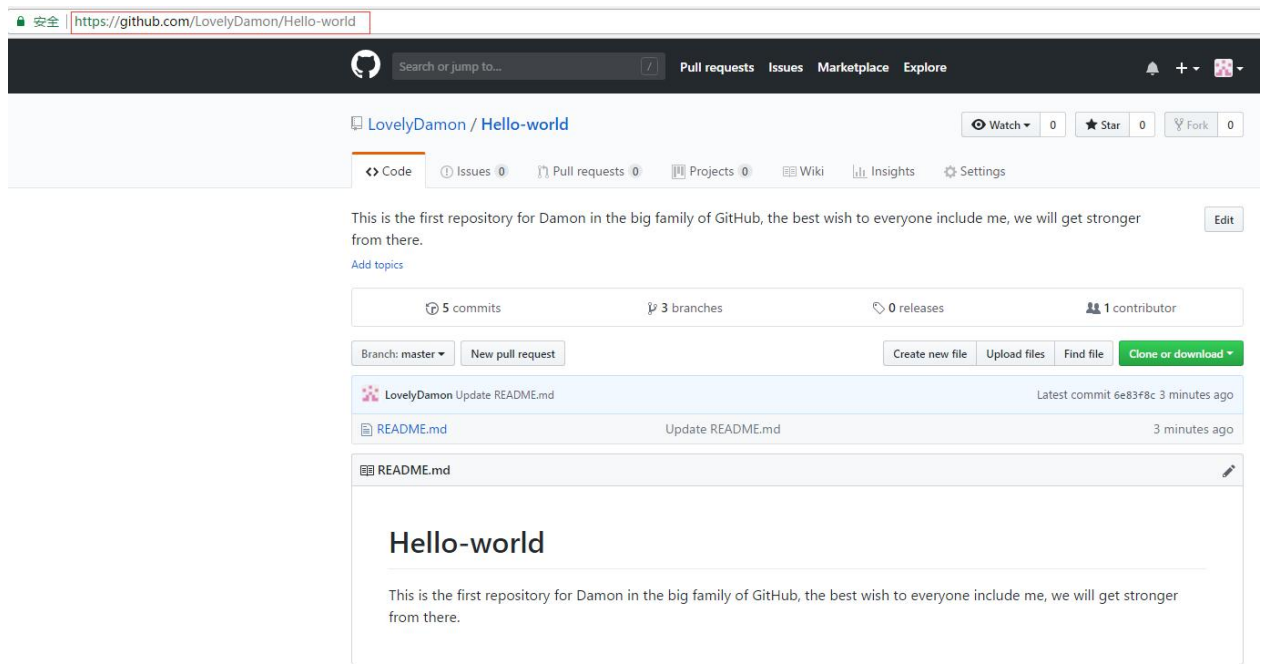
输入上边的代码，email 是一定要注册 GitHub 的那个邮箱地址

这两个的顺序可以颠倒，没有固定的顺序。

设置自己的 git 信息即完成安装和设置，可以输入 `git config --list` 查看自己的 git 信息。

```
guess@DESKTOP-EG1P403 MINGW64 ~  
$ git config --list  
core.symlinks=false  
core.autocrlf=true  
core.fscache=true  
color.diff=auto  
color.status=auto  
color.branch=auto  
color.interactive=true  
help.format=html  
rebase.autosquash=true  
http.sslcainfo=D:/Git/mingw64/ssl/certs/ca-bundle.crt  
http.sslbackend=openssl  
diff.astextplain.textconv=astextplain  
filter.lfs.clean=git-lfs clean -- %f  
filter.lfs.smudge=git-lfs smudge -- %f  
filter.lfs.process=git-lfs filter-process  
filter.lfs.required=true  
credential.helper=manager  
  
guess@DESKTOP-EG1P403 MINGW64 ~  
$
```

**13.克隆在 Github 上创建的库至本地电脑中，方便以后进行上传代码。**



## (1) Git bash 的定位

如果你不习惯将自己的文件储存在 C 盘中，你要先把 git bash 定位在你想要的储存的盘中。本人在这里将文件储存在 D 盘中。

```
guess@DESKTOP-EG1P403 MINGW64 /C/Users
$ cd /D

guess@DESKTOP-EG1P403 MINGW64 /D
```

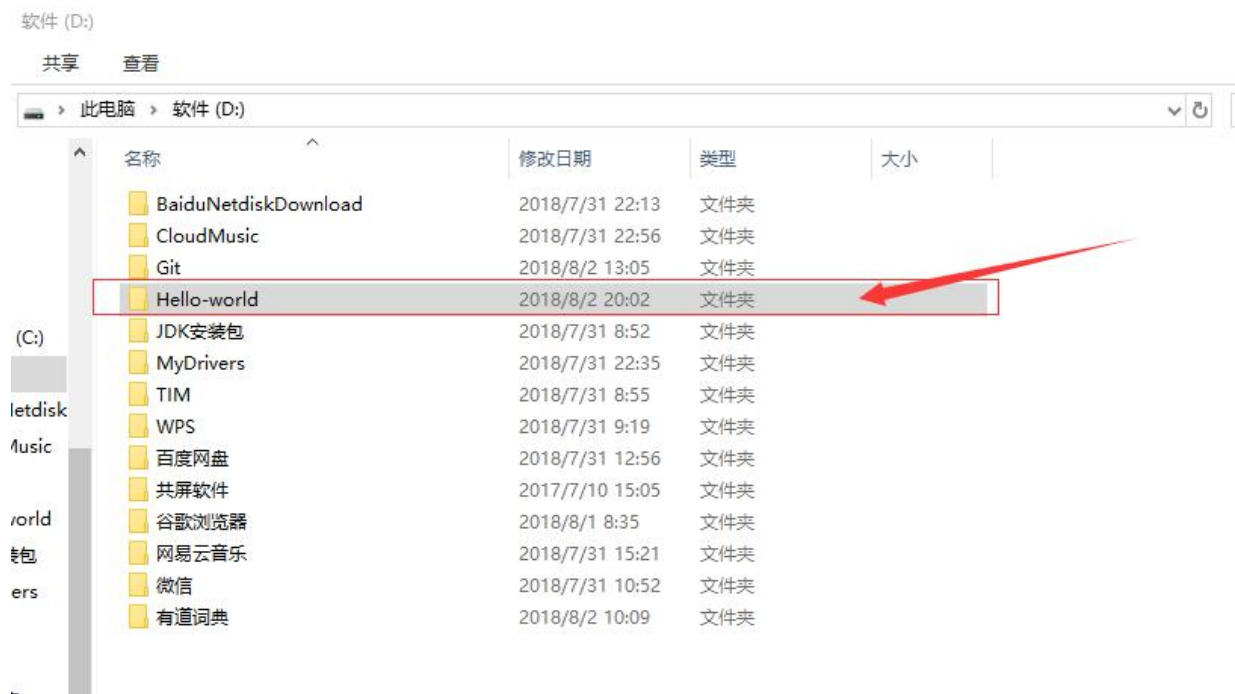
输入之后会出现/D 说明定位成功。

## (2) 输入 git clone “你在 github 上创建成功的库网址”

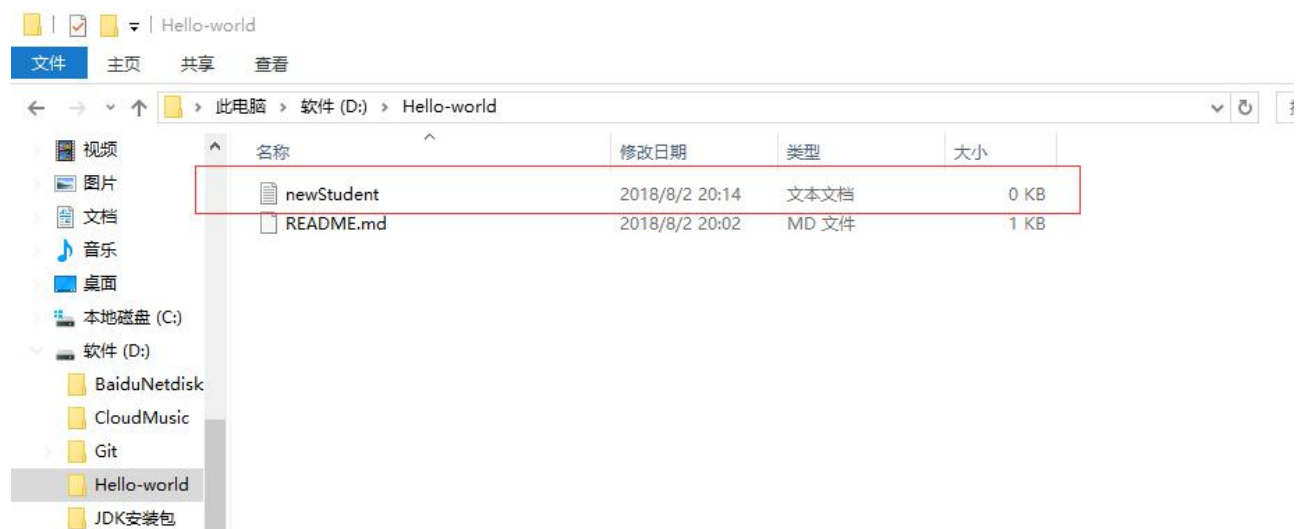
```
guess@DESKTOP-EG1P403 MINGW64 /D
$ git clone https://github.com/LovelyDamon/Hello-world
Cloning into 'Hello-world'...
remote: Counting objects: 16, done.
remote: Compressing objects: 100% (11/11), done.
remote: Total 16 (delta 4), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (16/16), done.

guess@DESKTOP-EG1P403 MINGW64 /D
$
```

(3) 打开之前定位的 D 盘，如下图可以看到已经有以我库名所创建的文件夹了。



(4) 打开该文件夹，创建任意一个格式的，任意名称的文件，本人在这里创建的是一个.txt 格式的文件。



(5)重新定位 **Git bash** 的位置，定位在我们库的文件夹，输入 **ls** 查看你目前所定位的文件夹中的文件，可发现创建的文件已存在。如下图：

```
guess@DESKTOP-EG1P403 MINGW64 /D
$ cd /D/Hello-world

guess@DESKTOP-EG1P403 MINGW64 /D/Hello-world (master)
$ ls
newStudent.txt  README.md

guess@DESKTOP-EG1P403 MINGW64 /D/Hello-world (master)
$
```

(6)输入 **git add newStudent.txt** 后回车

(7)在输入 **git commit -m “这是我写的备注”** 引号内的内容可以随便写，目的是方便查找区分。如下图：

```
guess@DESKTOP-EG1P403 MINGW64 /D/Hello-world (master)
$ git add newStudent.txt

guess@DESKTOP-EG1P403 MINGW64 /D/Hello-world (master)
$ git commit -m "新学生"
```

如果你完成上边操作出现下面所示的情况，不要慌，只需要将你之前设置的 **git** 信息再输入一遍即可，最后再输入一次 **git commit -m “这是我写的备注”**。

```
*** Please tell me who you are.

Run

  git config --global user.email "you@example.com"
  git config --global user.name "Your Name"

to set your account's default identity.
Omit --global to set the identity only in this repository.

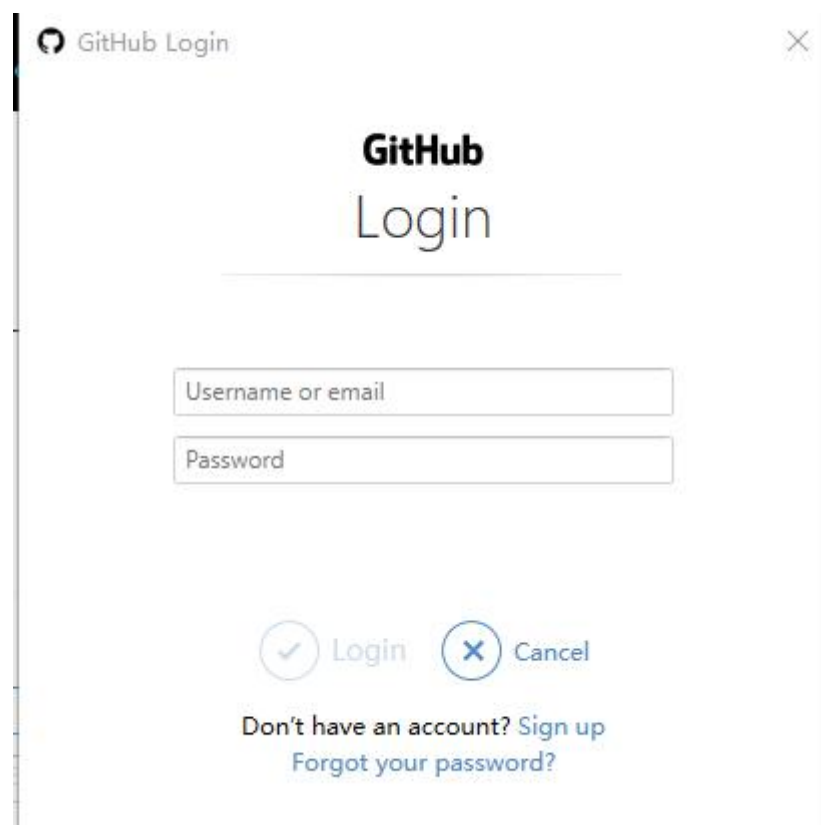
fatal: unable to auto-detect email address (got 'guess@DESKTOP-EG1P403.(none)')

guess@DESKTOP-EG1P403 MINGW64 /D/Hello-world (master)
$ git config --global user.email "13525334721@163.com"

guess@DESKTOP-EG1P403 MINGW64 /D/Hello-world (master)
$ git config --global user.name "newStudentTian"

guess@DESKTOP-EG1P403 MINGW64 /D/Hello-world (master)
$ git commit -m "新学生"
[master 9ac8c64] 新学生
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 newStudent.txt
```

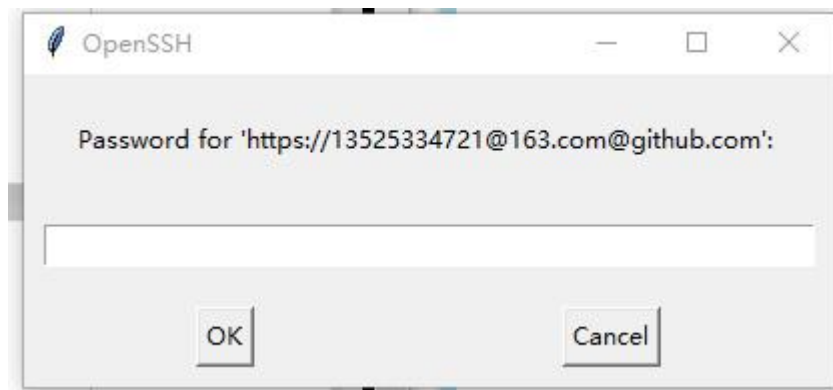
(8)输入 `git push origin master` 后会出下面的窗口，在这里登陆你的 **GitHub** 账号之后点击 **login**



The image shows a 'GitHub Login' window. At the top left is the GitHub logo and the text 'GitHub Login'. In the center, the words 'GitHub' and 'Login' are displayed in a large, bold font. Below this, there are two input fields: 'Username or email' and 'Password'. At the bottom, there are two buttons: 'Login' (with a checkmark icon) and 'Cancel' (with an 'X' icon). Below the buttons, there are two links: 'Don't have an account? Sign up' and 'Forgot your password?'.

```
guess@DESKTOP-EG1P403 MINGW64 /D/Hello-world (master)
$ git push origin master
Ligon failed, use ctrl+c to cancel basic credential prompt.
Username for 'https://github.com': 13525334721@163.com
```

如上所示若登录失败，根据提示重新输入一次用户名，会出现下面的界面，再次输入你的 Github 登陆密码



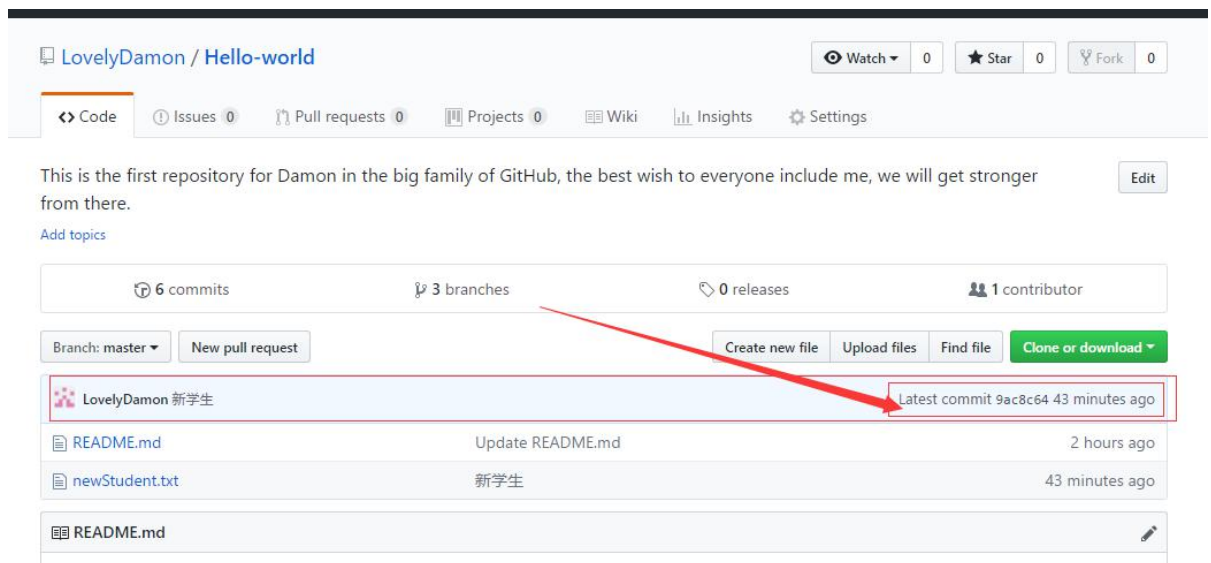
输入正确的密码后，若出现如图所示情况，说明你成功了！

```
guess@DESKTOP-EG1P403 MINGW64 /D/Hello-world (master)
$ git push origin master
Ligon failed, use ctrl+c to cancel basic credential prompt.
Username for 'https://github.com': 13525334721@163.com
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 2 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 293 bytes | 293.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/LovelyDamon/Hello-world
  6e83f8c..9ac8c64 master -> master

guess@DESKTOP-EG1P403 MINGW64 /D/Hello-world (master)
$
```

## (9)验证

打开你的 GitHub 网站，找到你创建的库。



可以看到你最新提交的记录。

在此之后,你只需要将你的代码,放到库的对应的文件夹中,然后使用, `git add` 、  
`git commit -m "备注"` ,最后 `git push origin master` ,将你的代码  
提交就可以了。