

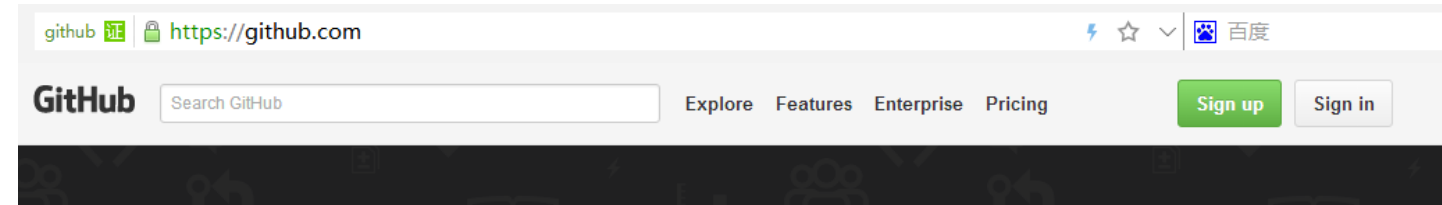
github入门到上传本地项目

- 一、创建github repository(仓库)
- 二、安装git客户端
- 三、为Github账户设置SSH key
- 四、上传本地项目到github

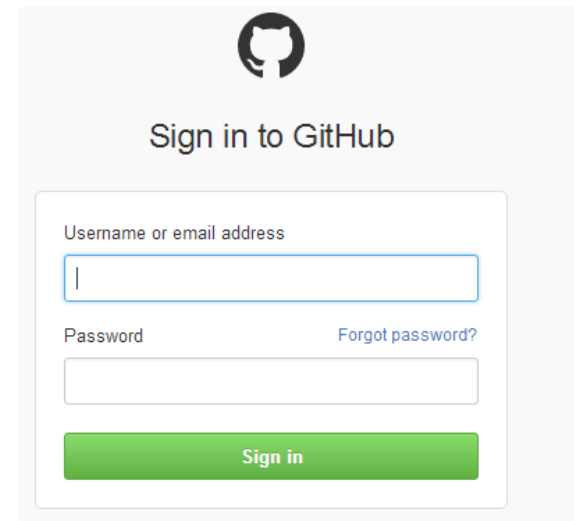
一、创建github repository(仓库)

1-1 登录github

github的官方网址：https://github.com ，如果没有账号，赶紧注册一个。



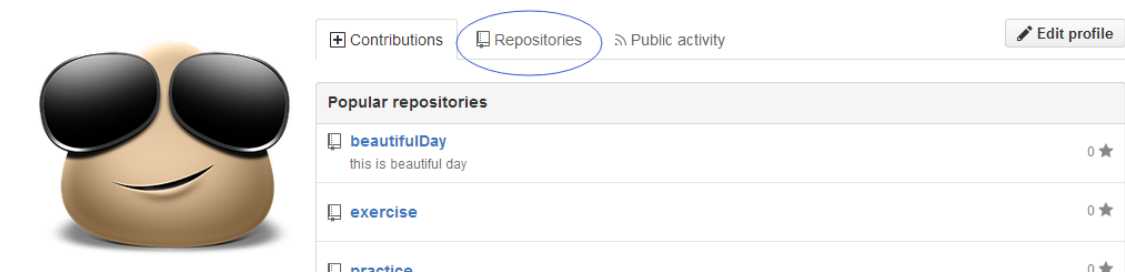
点击Sign in进入登录界面，输入账号和密码登录github。



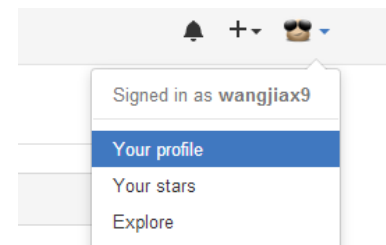
1-2 创建repository(仓库)

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，瞬间觉得这冰冰冷的代码充满了查克拉。

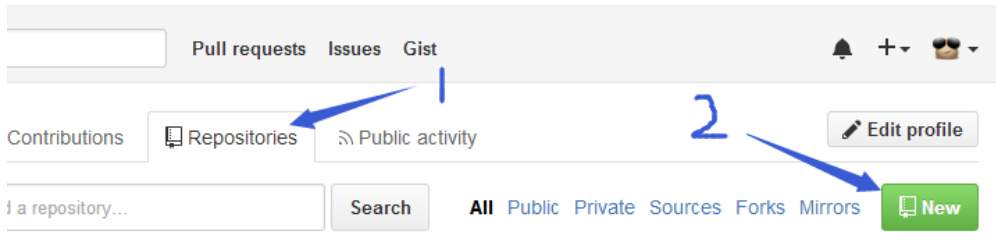
扯远了，来看怎么创建仓库，登录后可以看到有repository选项卡



如果没在这个页面也没关系，点击右上角的头像旁边的小三角，展开后可以看到Your profile，点击进入后也能看到repository



切换到repository选项卡，可以看到很醒目的new按钮。不用犹豫，点击它，开始创建自己的粮仓了。




下面是创建仓库信息，只有名字是必填项，现在我创建了一个仓库叫：beautifulDay


Create a new repository

A repository contains all the files for your project, including the revision history.

Owner

 wangjiax9

Repository name

beautifulDay 

Great repository names are short and memorable. Need inspiration? How about **urban-garbanzo**.

Description (optional)

this is beautiful day

☒ Public

Anyone can see this repository. You choose who can commit.

☐ Private

You choose who can see and commit to this repository.

☐ Initialize this repository with a README

This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: None

Add a license: None



Create repository

创建成功后，可以看到自己的仓库地址，如此，我的远程免费的仓库就创建了。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。

```
git init //把这个目录变成Git可以管理的仓库
git add README.md //文件添加到仓库
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
```

wangjiax9 / beautifulDay

Unwatch

1

Star

0

Fork

0

Code

Issues 0

Pull requests 0

Wiki

Pulse

Graphs

Settings

Quick setup — if you've done this kind of thing before

Set up in Desktop

 or

HTTPS

SSH

git@github.com:wangjiax9/beautifulDay.git

We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

...or create a new repository on the command line

```
echo "# beautifulDay" >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin git@github.com:wangjiax9/beautifulDay.git
git push -u origin master
```

...or push an existing repository from the command line

```
git remote add origin git@github.com:wangjiax9/beautifulDay.git
git push -u origin master
```

...or import code from another repository

You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

Import code

二、安装git客户端

Git是目前世界上最先进的分布式版本控制系统，[git与svn的五个基本区别](#)。它有以下特点：

分布式：Git版本控制系统是一个分布式的系统, 是用来保存工程源代码历史状态的命令行工具;

保存点：Git的保存点可以追踪源码中的文件, 并能得到某一个时间点上的整个工程项目额状态; 可以在该保存点将多人提交的源码合并, 也可以会退到某一个保存点上;

Git离线操作性：Git可以离线进行代码提交, 因此它称得上是完全的分布式处理, Git所有的操作不需要在线进行; 这意味着Git的速度要比SVN等工具快得多, 因为SVN等工具需要在线时才能操作, 如果网络环境不好, 提交代码会变得非常缓慢;

Git基于快照：SVN等老式版本控制工具是将提交点保存成补丁文件, Git提交是将提交点指向提交时的项目快照, 提交的东西包含一些元数据(作者, 日期, GPG等);

Git的分支和合并：分支模型是Git最显著的特点, 因为这改变了开发者的开发模式, SVN等版本控制工具将每个分支都要放在不同的目录中, Git可以在同一个目录中切换不同的分支；

分支即时性：创建和切换分支几乎是同时进行的, 用户可以上传一部分分支, 另外一部分分支可以隐藏在本地, 不必将所有的分支都上传到GitHub中去;

分支灵活性：用户可以随时 创建 合并 删除分支, 多人实现不同的功能, 可以创建多个分支进行开发, 之后进行分支合并, 这种方式使开发变得快速, 简单, 安全。

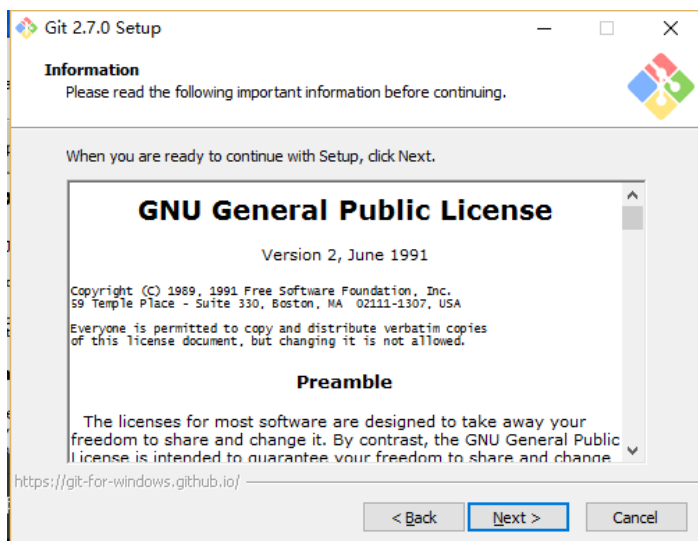
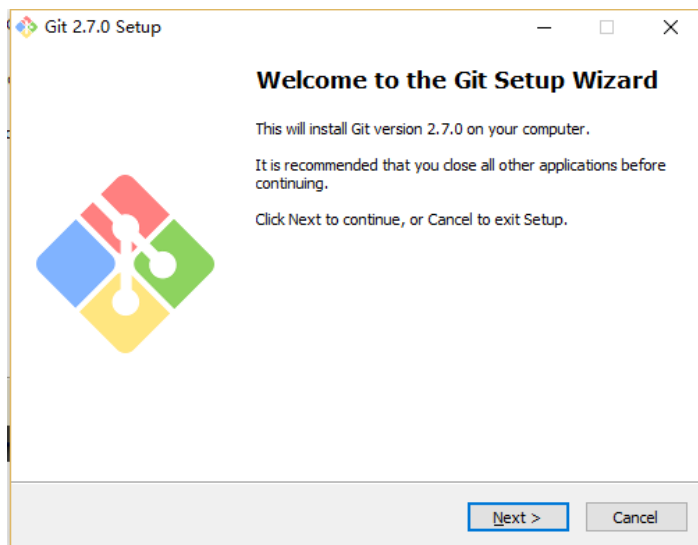
2-1 下载git客户端

官方下载地址：http://git-scm.com/download/ 根据你自己的系统 下载对应版本，没想到它知道我是Windows ^_^

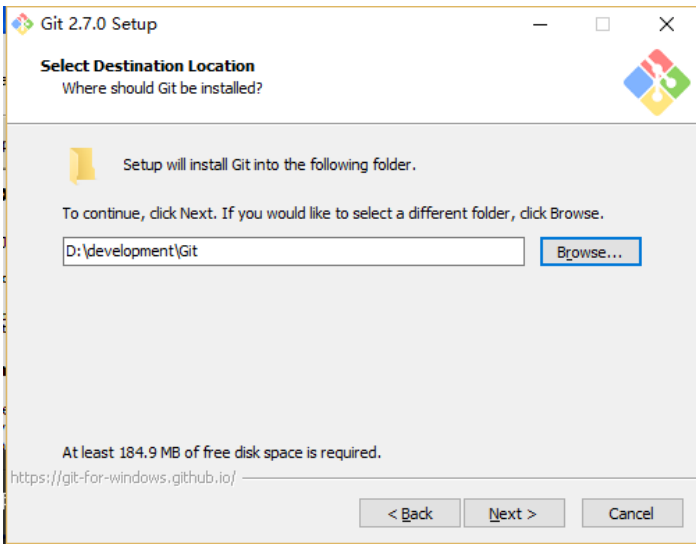


2-2 安装客户端

下载好之后咱们开始安装吧，欢迎界面，下一步。

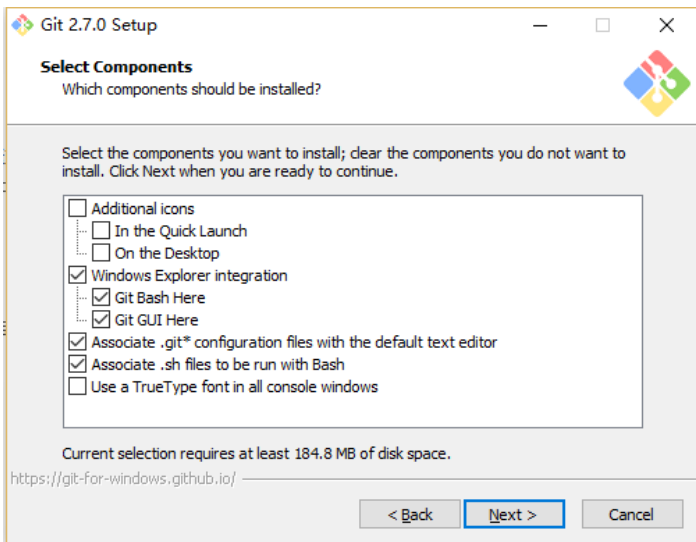


选择安装路径，千万别选带中文的路径，有时候会引起不必要的误会。

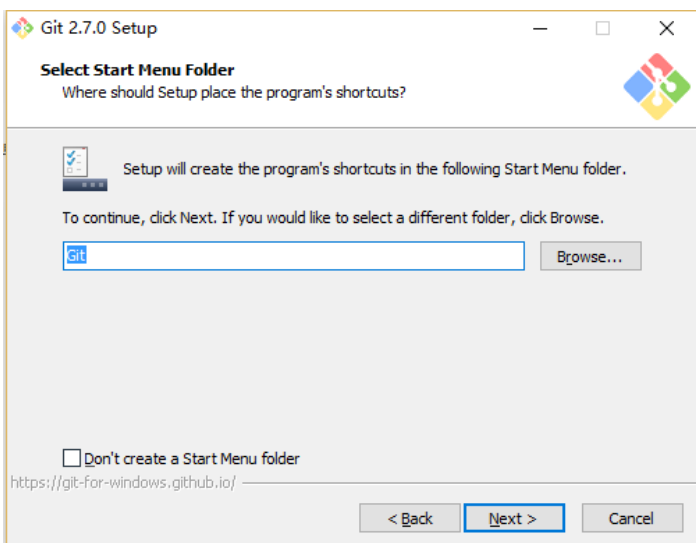


选择安装组件，按默认的来就好了。

- 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TrueType编码, 该编码是微软和苹果公司制定的通用编码;

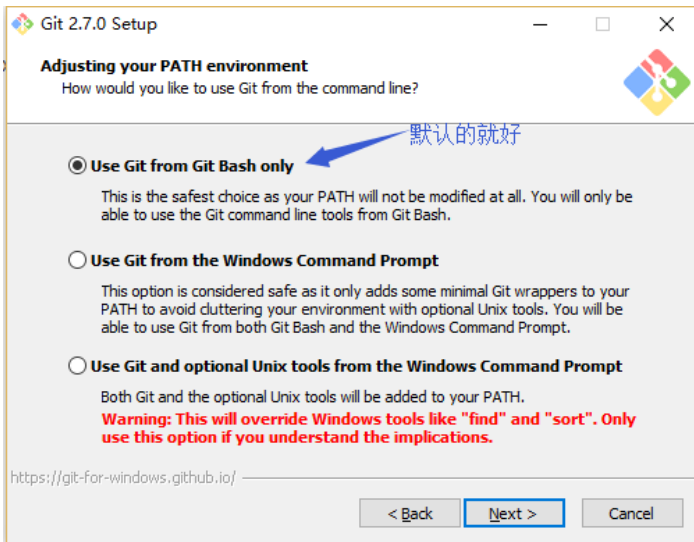


设置开始菜单中快捷方式的目录名称，默认就好，下一步吧



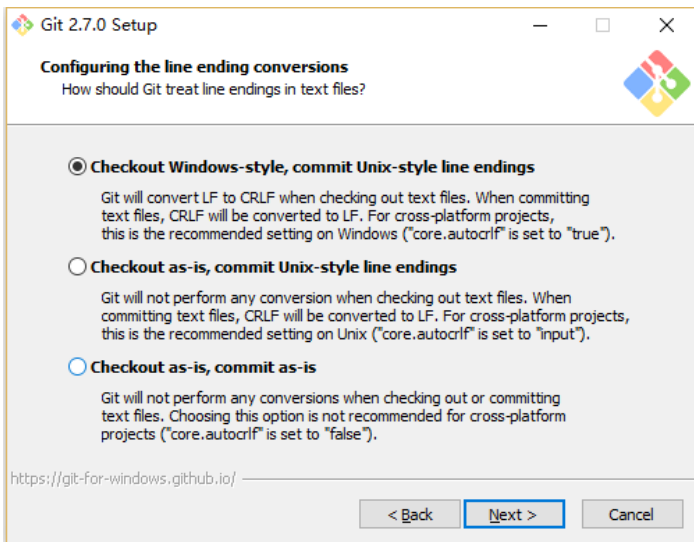
设置环境变量：选择使用什么样的命令行工具，一般情况下我们默认使用Git Bash即可，默认选择;

- 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;



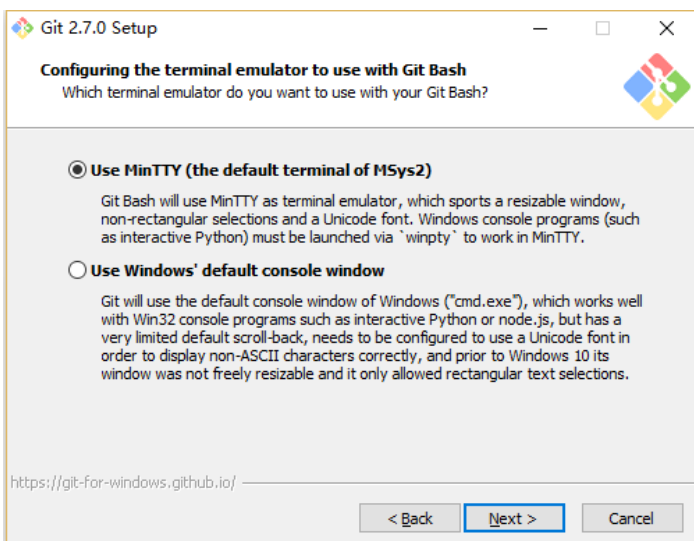
选择换行格式，依然是默认就好。

- 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;

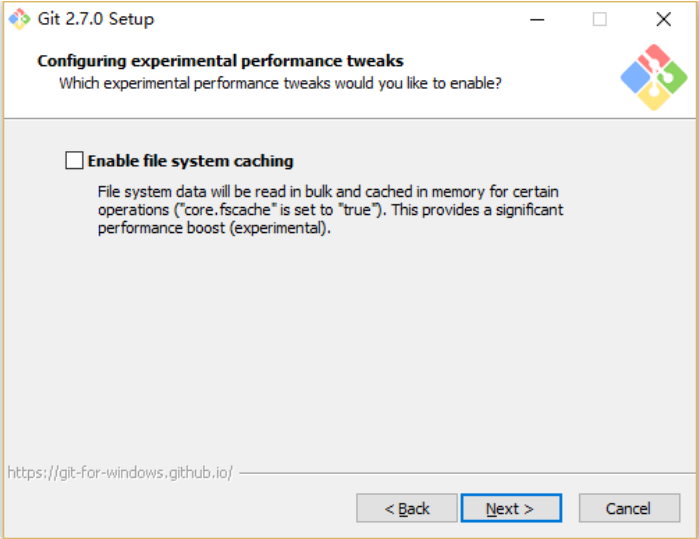


选择终端模拟器，依然默认就好

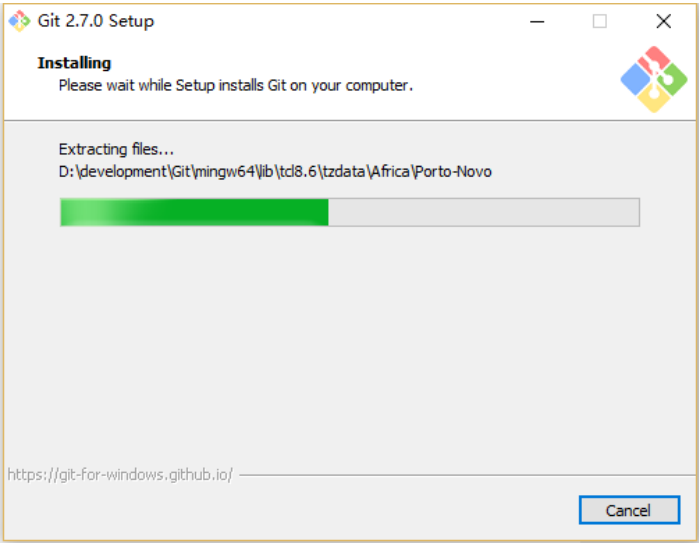
- 1) 使用MinTTY，就是在Windows开了一个简单模拟Linux命令环境的窗口Git Bash
- 2) 使用windows的系统的命令程序cmd.exe



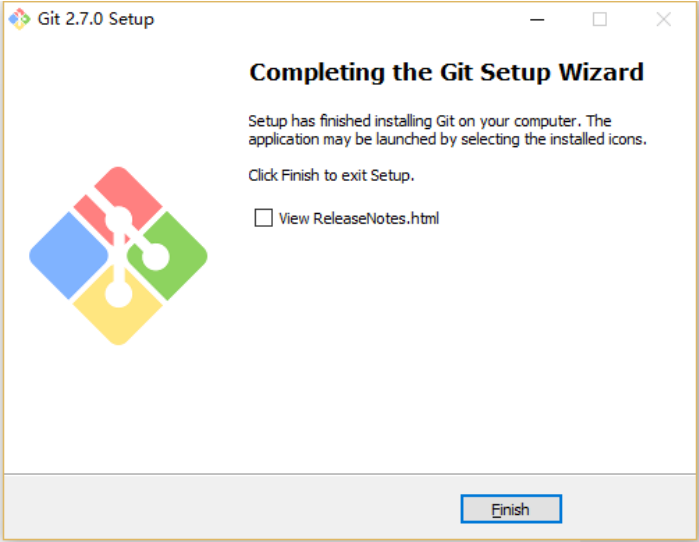
选择默认就好，不用文件系统缓存



安装中.....



git终于安装成功咯。



2-3 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

MINGW64:/

```
specter@X9 MINGW64 /
$ git config --global user.name "wangjiax9"
specter@X9 MINGW64 /
$ git config --global user.email "6215048wjl@163.com"
specter@X9 MINGW64 /
$
```

ps : git config -global 参数,有了这个参数,表示你这台机器上所有的Git仓库都会使用这个配置,当然你也可以对某个仓库指定的不同的用户名和邮箱。

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

3-1 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

```
specter@X9 MINGW64 /
$ cd ~/.ssh
specter@X9 MINGW64 ~/.ssh
$ ls
id_rsa id_rsa.pub known_hosts
```

也可以打开我的电脑C:\Users\specter\.ssh 里面找到

此电脑 > SYSTEM (C:) > 用户 > specter > .ssh	
名称	修改
id_rsa	2016/3/18 11:50
id_rsa.pub	2016/3/18 11:50
known_hosts	2016/3/18 11:50

如果没有生成,那么通过\$ ssh-keygen -t rsa -C "6215048wjl@163.com" 来生成。

- 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

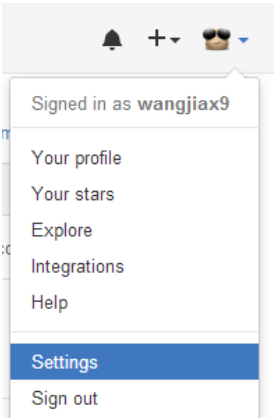
```
specter@X9 MINGW64 ~
$ ssh-keygen -t rsa -C "6215048wjl@163.com"
Generating public/private rsa key pair.
Enter file in which to save the key (/c/Users/specter/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /c/Users/specter/.ssh/id_rsa.
Your public key has been saved in /c/Users/specter/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:G600vs2veJ8YmgBgaKinb35rxTvD15ijqvo4Y4ZrC8 6215048wjl@163.com
The key's randomart image is:
+---[RSA 2048]-----+
|+
|+o
|o..
|. +.
|= . S .
+ .... +
|. oo.+++
|E++.=o=Ooo .
|BB+oo=++*++
+---[SHA256]-----+
```

生成成功后,去对应目录C:\Users\specter\.ssh里(specter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


```
id_rsa.pub - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAQDmSe3J6reoTp81JehpUPT1BnKQNey7Wk1BIcsu4/
17D/EshJup3TR
+FuIxB2U0afCXYmR5VQbKXtgChDtnJUIvx5jwyvZM10fe0/UyBEHRRfDRGpi
+QyeKH2Xm0zbM35ojJRBHAR4RiIVunFm4o
+kRzsACntMvm/MGbUIZq3LerDqtJSkKjaDDSH6gDJVQSmKWQqgtM8aINPFsH/3X26PaqB
DMnYf+e61Lo+SffR9v50TqsDVc/VKYP9x6NK3Se/X
+LqAhPB1MhKzd78VrY5X3QsdE/xtCMFbd4L7mAYKjF
+cvBjntSpfQkSD0paTPo1QIkvL57vAj4y34g8mDvUb 6215048wj1@163.com
```

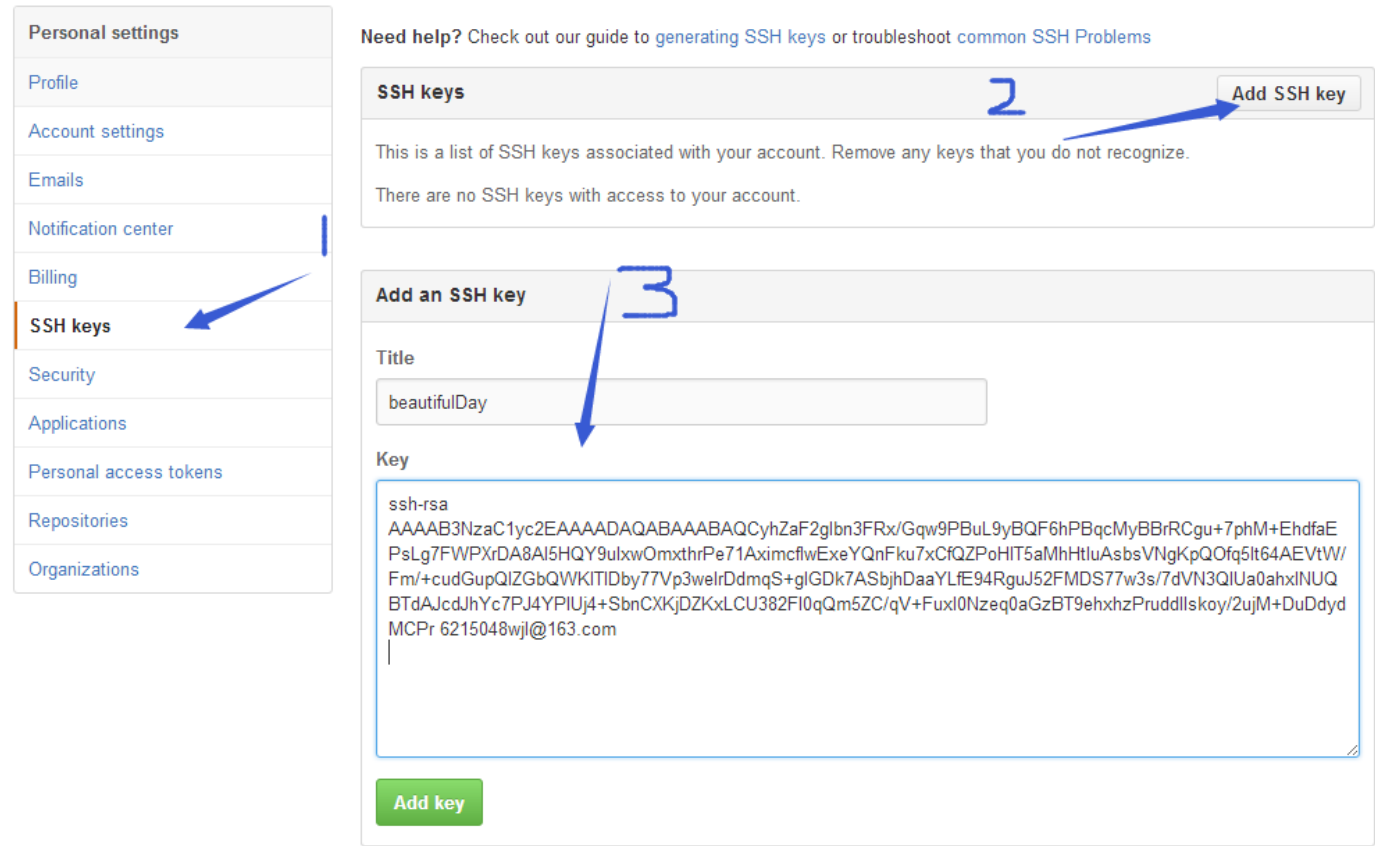
3-2 为github账号配置ssh key

切换到github，展开个人头像的小三角，点击settings



然后打开SSH keys菜单，点击Add SSH key新增密钥，填上标题，跟仓库保持一致吧，好区分。

接着将id_rsa.pub文件中key粘贴到此，最后Add key生成密钥吧。



如此，github账号的SSH keys配置完成。

SSH keys

Add SSH key

This is a list of SSH keys associated with your account. Remove any keys that you do not recognize.

beautifulDay

9f:62:37:ef:58:5d:bb:9e:c6:14:b5:d7:1c:98:49:72

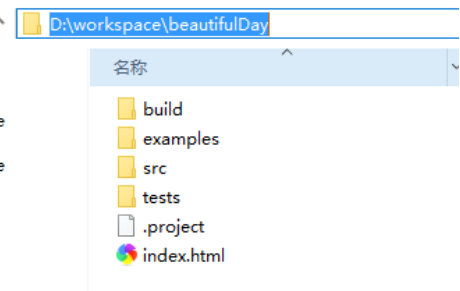
Added on 2 Feb 2016 — Never used

Delete

四、上传本地项目到github

4-1 创建一个本地项目

我这创建了几个空文件夹和一个文件及一个项目配置文件，好多前端项目都这样搭架构，我也追随潮流哈。



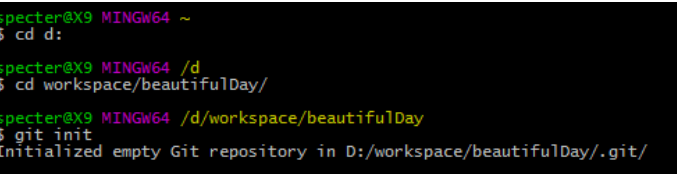
4-2 建立本地仓库

再来复习一下创建新仓库的指令：

```
git init //把这个目录变成Git可以管理的仓库
git add README.md //文件添加到仓库
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
```

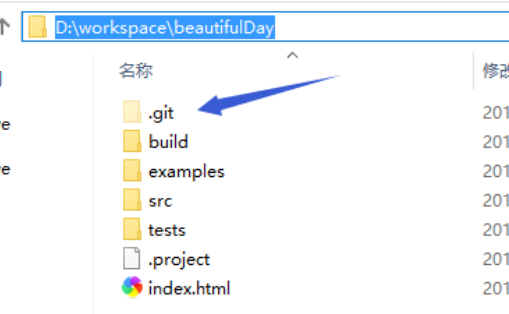
首先，进入到beautifulDay项目目录，还记得创建仓库成功后的那个页面吧，指令都在呢。

然后执行指令：`git init`



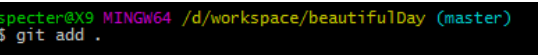
初始化成功后你会发现项目里多了一个隐藏文件夹.git

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。

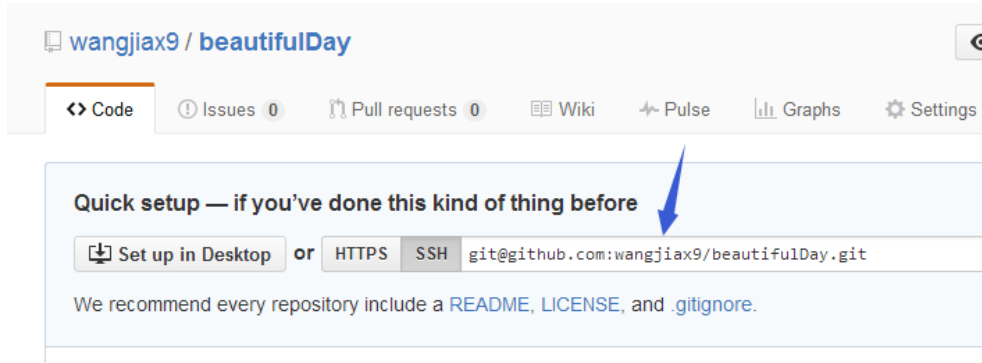
执行指令：`git commit -m "提交文件"`

```
specter@X9 MINGW64 /d/workspace/beautifulDay (master)
$ git commit -m "提交文件"
[master (root-commit) 61c1615] 提交文件
2 files changed, 53 insertions(+)
create mode 100644 .project
create mode 100644 index.html
```

如此本地仓库建立好了。

4-3 关联github仓库

到github beautifulDay仓库复制仓库地址



然后执行指令：`git remote add origin git@github.com:wangjiax9/beautifulDay.git`

```
specter@X9 MINGW64 /d/workspace/beautifulDay (master)
$ git remote add origin git@github.com:wangjiax9/beautifulDay.git
```

4-4 上传本地代码

执行指令：`git push -u origin master`

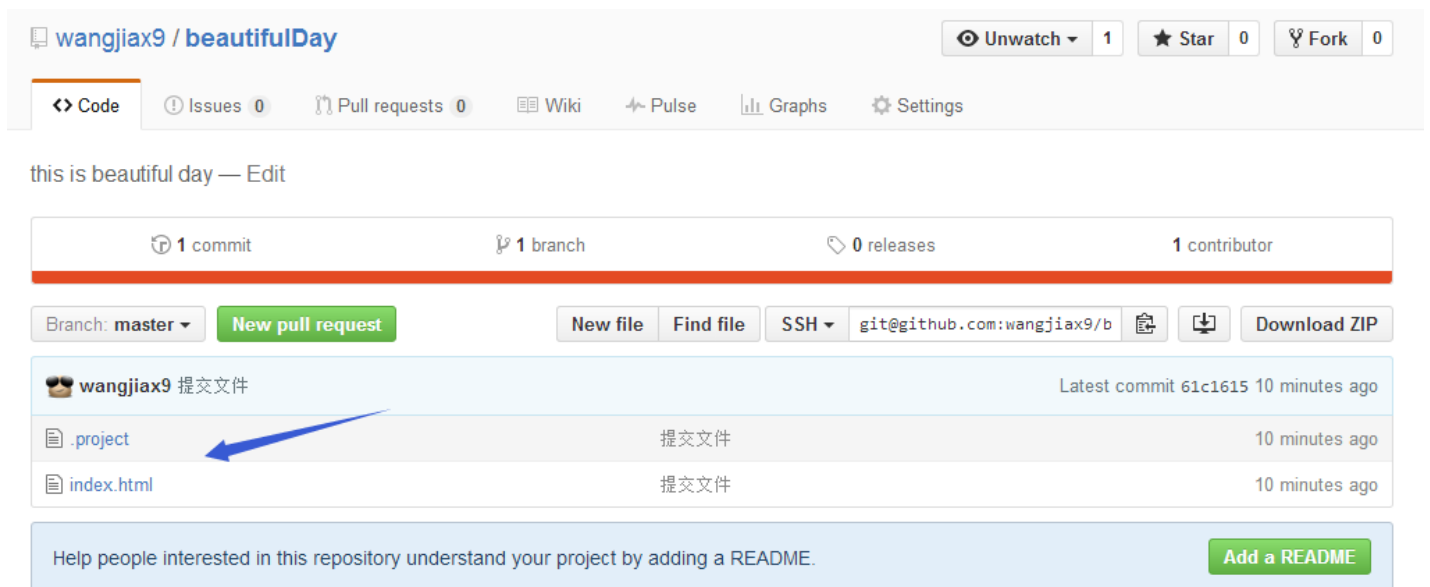
1) 敲一个：yes，然后回车

```
specter@X9 MINGW64 /d/workspace/beautifulDay (master)
$ git push -u origin master
The authenticity of host 'github.com (192.30.252.128)' can't be established.
RSA key fingerprint is SHA256:nThbg6kXUpJWC17E1IGOCspRomTxdCARLviKw6E5SY8.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'github.com,192.30.252.128' (RSA) to the list of known hosts.
Counting objects: 4, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 734 bytes | 0 bytes/s, done.
Total 4 (delta 0), reused 0 (delta 0)
To git@github.com:wangjiax9/beautifulDay.git
 * [new branch]      master -> master
Branch master set up to track remote branch master from origin.
```

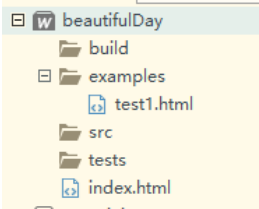
到此，本地代码已经推送到github仓库了，我们现在去github仓库看看。

咦！奇怪了，我的目录呢？这个坑突然冒出来是不是印象很深刻呢~

注意咯：`git`是不能管理空的文件夹的，文件夹里必须有文件才能add

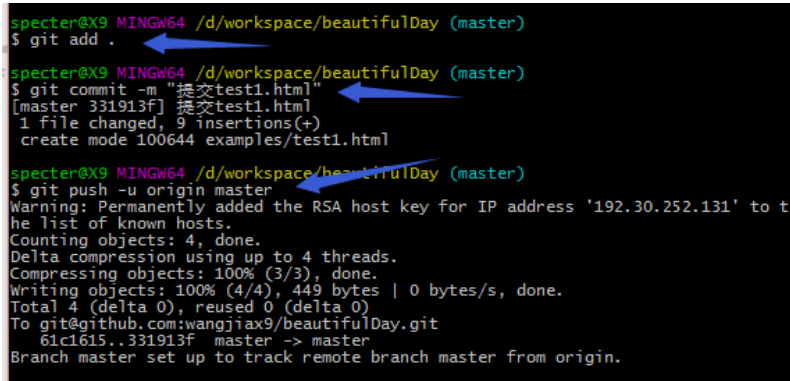


好，我们来试一下，我在examples里新建了一个test1.html文件

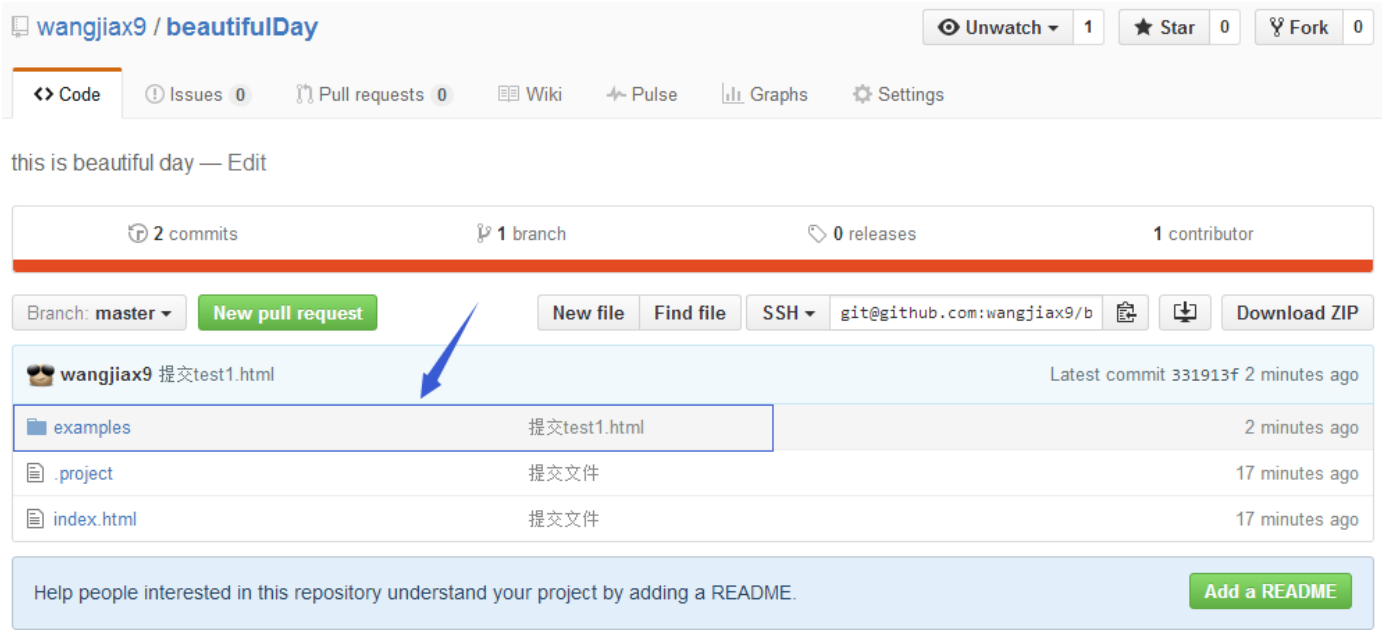


执行指令添加文件->提交文件->推送文件

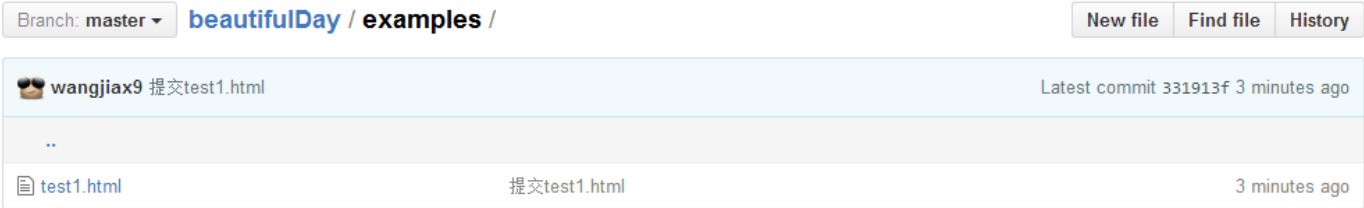
```
git add .
git commit -m "提交test1.html"
git push -u origin master
```



然后刷新一个github，你会看到，examples文件夹出来了。



打开examples文件夹，test1.html也在里面。



THE END