**GitHub for Windows 使用入门**

1. **注册账户以及创建仓库**

打开github网站注册github账号，并创建仓库（免费用户只能建公共仓库）。

Create a New Repository，填好名称后Create，之后会出现一些仓库的配置信息。

1. **安装客户端msysgit**

github是服务端，要想在自己电脑上使用git我们还需要一个git客户端，我这里选用msysgit，这个只是提供了git的核心功能，而且是基于命令行的。如果想要图形界面的话只要在msysgit的基础上安装TortoiseGit即可。

装完msysgit后右键鼠标会多出一些选项来。

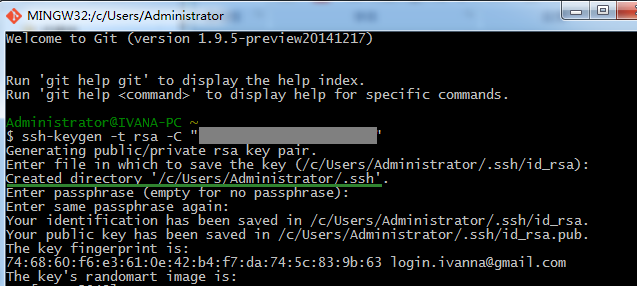


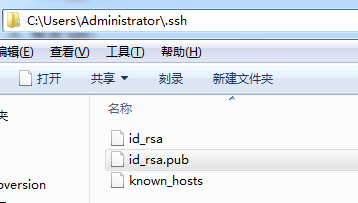
在本地仓库里右键选择Git Init Here，会多出来一个.git文件夹，这就表示本地git创建成功。右键Git Bash进入git命令行，为了把本地的仓库传到github，还需要配置ssh key。

1. **配置SSH Key**

首先在本地创建ssh key。点击打开Git Bash窗口，输入如下命令:

*ssh-keygen -t rsa -C* [*your\_email@youremail.com*](mailto:your_email@youremail.com)

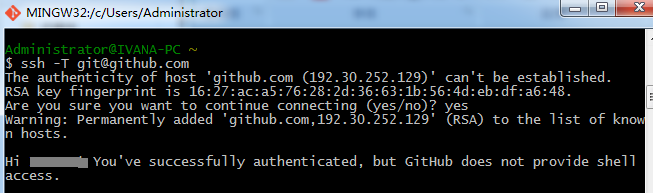
后面的[your\_email@youremail.com](mailto:your_email@youremail.com)改为你的邮箱，之后会要求确认路径和输入密码，我们这使用默认的一路回车就行。成功的话会在~/下生成.ssh文件夹。**

**

回到github，进入Account Setting - SSH Keys - Add SSH Key, 填入任意title，将id\_rsa.pub里内容粘贴到key中，并保存。为了验证是否成功，在git bash下输入：

*$ ssh -T* [*git@github.com*](mailto:git@github.com)

如果是第一次的会提示是否continue，输入yes就会看到：You’ve successfully authenticated, but GitHub does not provide shell access 。这就表示已成功连上github。



1. **Git 配置**

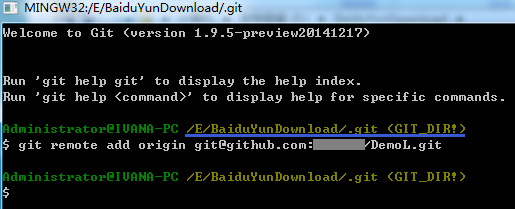
本地设置username和email，因为github每次commit都会记录他们。在Git Bash中输入如下命令：

*$ git config --global user.name "your name"*

*$ git config --global user.email* [*your\_email@youremail.com*](mailto:your_email@youremail.com)

点击要上传的仓库中的.git文件夹，右键git bash，添加远程地址：

*$ git remote add origin* [*git@github.com:yourName/yourRepo.git*](mailto:git@github.com:yourName/yourRepo.git)

**

后面的yourName和yourRepo表示你在github的用户名和刚才新建的仓库，加完之后进入.git，打开config文件，这里会多出一个remote “origin”内容，这就是刚才添加的远程地址，也可以直接修改config来配置远程地址。

1. **提交、上传**

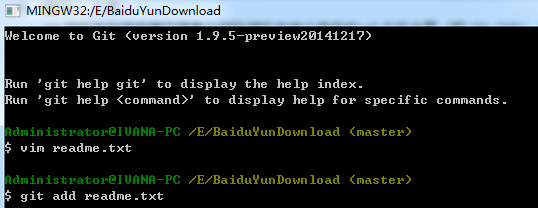
Git把所有的历史提交信息全部存储在本地仓库中的.git文件夹里（即Git\_DIR），它就是一个Git项目的版本仓库；你对本地的源代码进行编辑修改后创建的提交也都会先保存在这里面，然后再推送到远端的服务器。当我们我把项目目录和“Git目录”一起拷到其它电脑里，它能马上正常的工作（所有的提交信息全都保存在Git目录里）。

接下来在本地仓库里添加一些文件，比如readme.txt

*$ vim readme.txt*

然后将该文件放入版本库（即.git文件夹）中：

*$ git add readme.txt*

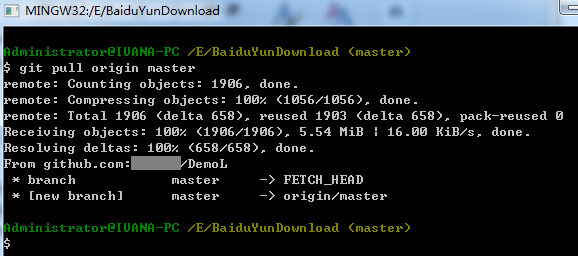


此时需要把本机缓存中的内容提交到本机的 HEAD 里面，使用命令：

*$git commit -m "注释"*

上传到github：

*$ git push origin master*



git push命令会将本地仓库推送到远程服务器。

git pull命令则相反。

1. **gitignore文件**

.gitignore顾名思义就是告诉git需要忽略的文件，.gitignore 配置文件用于配置不需要加入版本管理的文件，配置好该文件可以为我们的版本管理带来很大的便利。

配置语法：

以斜杠“/”开头表示目录；

以星号“\*”通配多个字符；

以问号“?”通配单个字符

以方括号“[]”包含单个字符的匹配列表；

以叹号“!”表示不忽略(跟踪)匹配到的文件或目录；

示例如下：

1. 规则：fd1/\*

忽略目录 fd1 下的全部内容；注意，不管是根目录下的 /fd1/ 目录，还是某个子目录 /child/fd1/ 目录，都会被忽略；

1. 规则：/fd1/\*

忽略根目录下的 /fd1/ 目录的全部内容；

附：

**常见问题：**

1. git操作提示*fatal:*This operation must be run in a work tree

当在.git文件夹下进行git操作时，会得到该错误。

此时回到仓库目录下（一般为.git文件夹的上一层文件夹）进行操作，该问题即可避免

1. push时发生*error:*srcrefspecmasterdoesnotmatchany
2. 当本地版本库，即.git文件夹为空时，进行提交将会得到该错误

向版本库中运行add命令后再进行push，即可避免该问题

1. 不是当前文件夹下没有文件，而是没有把文件添加到git项目中，

通过先add文件再commit 文件到项目中，然后就可以把项目中的文件，push到服务器上了

1. push时发生*error: insufficient permission for adding an object to repository database ./objects*

服务端没有可写目录的权限，.git文件夹中objects文件夹没有group写权限

给予对应权限即可修复该问题

**后续阅读推荐:**

**git clone :**[**http://blog.csdn.net/hudashi/article/details/7664396/**](http://blog.csdn.net/hudashi/article/details/7664396/)