

```
1 学习自https://www.breakyizhan.com/git/32.html
```

Git是一个分布式版本控制的软件，它只是一个•管理代码版本的管理工具

Git的官网下载地址

```
1 http://git-scm.com/downloads
```

安装后有三个版本Git

```
1 Git GUI ///图形化大的界面
2 Git BASH ///在CMD基础上多加了一些比较方便的命令
3 Git CMD
```

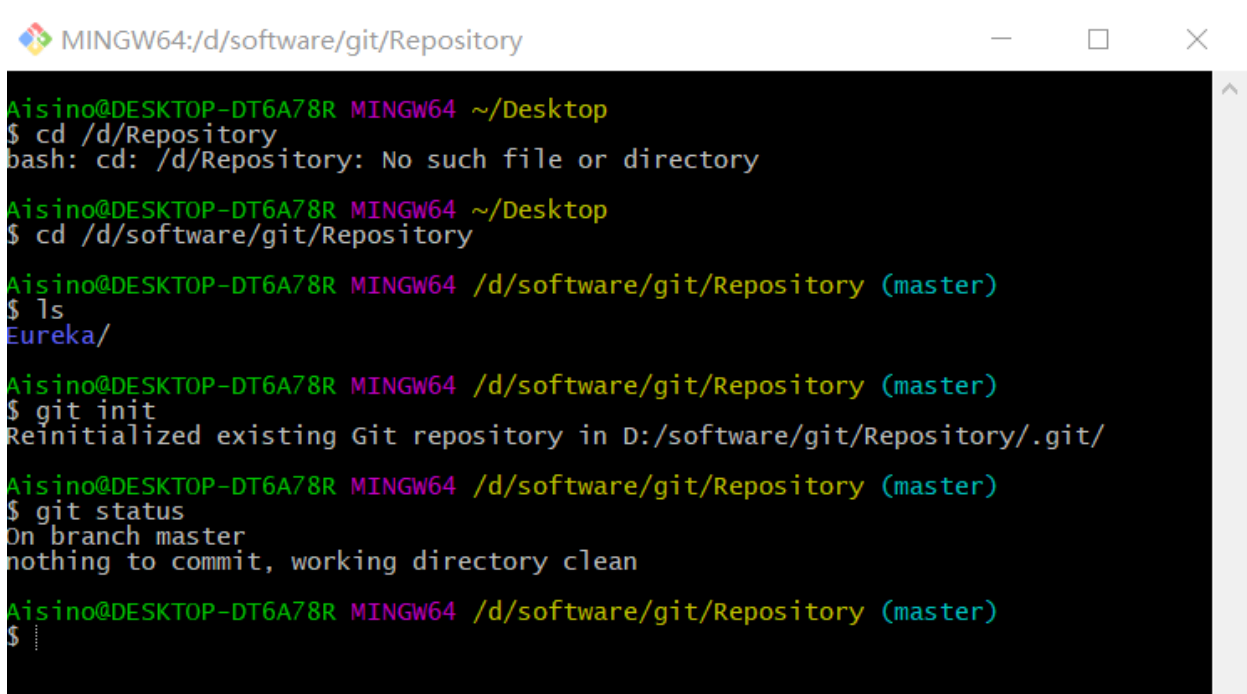
配置名字及邮箱

```
1 git config --global user.name "Your Name"
2 git config --global user.email "Your Email"
```

版本库：存放代码的地方

创建版本库

```
1 cd /d/software/git/Repository] cd(跳转)到程序的目录下
2 git init //创建版本库
3 git status //查看版本库当前状态
```

A screenshot of a Windows command prompt window titled 'MINGW64:/d/software/git/Repository'. The window shows a series of commands and their outputs. The user starts at the desktop, navigates to the repository directory, and initializes a new Git repository. The output shows the repository is reinitialized and the working directory is clean.

```
MINGW64:/d/software/git/Repository
Aisino@DESKTOP-DT6A78R MINGW64 ~/Desktop
$ cd /d/Repository
bash: cd: /d/Repository: No such file or directory
Aisino@DESKTOP-DT6A78R MINGW64 ~/Desktop
$ cd /d/software/git/Repository
Aisino@DESKTOP-DT6A78R MINGW64 /d/software/git/Repository (master)
$ ls
Eureka/
Aisino@DESKTOP-DT6A78R MINGW64 /d/software/git/Repository (master)
$ git init
Reinitialized existing Git repository in D:/software/git/Repository/.git/
Aisino@DESKTOP-DT6A78R MINGW64 /d/software/git/Repository (master)
$ git status
On branch master
nothing to commit, working directory clean
Aisino@DESKTOP-DT6A78R MINGW64 /d/software/git/Repository (master)
$
```

添加文件到暂存区

一般把文件添加到版本库都是没有信息提示的，这是因Git类似于Linux，nothing is good thing。

```
1 git add .classpath //添加文件.classpath到版本库
```

```
2 git status
3 git add . //添加所有文件到版本库
4 git status
```

```
Aisino@DESKTOP-DT6A78R MINGW64 /d/software/git/Repository (master)
$ git add 测试.txt

Aisino@DESKTOP-DT6A78R MINGW64 /d/software/git/Repository (master)
$ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        new file:   "\346\265\213\350\257\225.txt"
```

通过上面操作文件已经添加到了版本库的暂存区，添加进入暂存区的文件都会变成绿色，没有添加进去的都会是红色。

提交到版本库

```
1 git commit -m "Hello Git" ///把代码提交到版本库，“Hello Git”时描述这段代码的内容
```

```
Aisino@DESKTOP-DT6A78R MINGW64 /d/software/git/Repository (master)
$ git commit -m "Hello Git"
[master bb514d6] Hello Git
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 "\346\265\213\350\257\225.txt"

Aisino@DESKTOP-DT6A78R MINGW64 /d/software/git/Repository (master)
$ git status
On branch master
nothing to commit, working directory clean

Aisino@DESKTOP-DT6A78R MINGW64 /d/software/git/Repository (master)
$
```

-m 后的描述可以省略

最后一句话表示工作区已经干净了没有东西可以提交了，文件已经提交到版本库了。

Git的三个区

工作区

工作区就是我们写代码的地方，就是前面的/d/software/git/Repository]目录下的区域，不包括隐藏文件夹。

暂存区

暂存区一般存放在 “.git目录下” 的index文件中 (/ .git/index/) 。用git add 的文件都会暂时放在这里，git add 可以多次添加，然后一起提交到版本库。

版本库

版本库就是在隐藏的目录.git。

查看Git的版本库

```
1 git log ///查看git的版本库
```

```
Aisino@DESKTOP-DT6A78R MINGW64 /d/software/git/Repository (master)
$ git log
commit c7691b712e3734798c530305c8c4503bad8c8adb
Author: wangyufeng123 <wangyufeng@szhtxx.com>
Date: Mon Dec 17 14:12:12 2018 +0800

    Hello Git two

commit bb514d6f1a20fbe80c02aedb6ed129a3f54bdcdc
Author: wangyufeng123 <wangyufeng@szhtxx.com>
Date: Mon Dec 17 13:38:43 2018 +0800

    Hello Git

commit c308a168e2e5758639f2acc2cb5eed5ae4d34bde
Author: wangyufeng123 <wangyufeng@szhtxx.com>
Date: Fri Nov 23 13:35:30 2018 +0800

    Eureka

Aisino@DESKTOP-DT6A78R MINGW64 /d/software/git/Repository (master)
```

版本回退

```
1 git reset --hard c308a ///回退到Eureka的版本
```

```
Aisino@DESKTOP-DT6A78R MINGW64 /d/software/git/Repository (master)
$ git reset --hard bb51
HEAD is now at bb514d6 Hello Git

Aisino@DESKTOP-DT6A78R MINGW64 /d/software/git/Repository (master)
$ git log
commit bb514d6f1a20fbe80c02aedb6ed129a3f54bdcdc
Author: wangyufeng123 <wangyufeng@szhtxx.com>
Date: Mon Dec 17 13:38:43 2018 +0800

    Hello Git

commit c308a168e2e5758639f2acc2cb5eed5ae4d34bde
Author: wangyufeng123 <wangyufeng@szhtxx.com>
Date: Fri Nov 23 13:35:30 2018 +0800

    Eureka

Aisino@DESKTOP-DT6A78R MINGW64 /d/software/git/Repository (master)
$
```

版本恢复

```
1 git reflog ///查看每一次操作Git的记录
2 git reset --hard
3 git log ///查看Git版本库
```

```
MINGW64:/d/software/git/Repository

Aisino@DESKTOP-DT6A78R MINGW64 /d/software/git/Repository (master)
$ git reflog
bb514d6 HEAD@{0}: reset: moving to bb51
c7691b7 HEAD@{1}: commit: Hello Git two
bb514d6 HEAD@{2}: commit: Hello Git
c308a16 HEAD@{3}: commit (initial): Eureka

Aisino@DESKTOP-DT6A78R MINGW64 /d/software/git/Repository (master)
$ git reset --hard c7691
HEAD is now at c7691b7 Hello Git two

Aisino@DESKTOP-DT6A78R MINGW64 /d/software/git/Repository (master)
$ git log
commit c7691b712e3734798c530305c8c4503bad8c8adb
Author: wangyufeng123 <wangyufeng@szhtxx.com>
Date: Mon Dec 17 14:12:12 2018 +0800

    Hello Git two

commit bb514d6f1a20fbe80c02aedb6ed129a3f54bdcde
Author: wangyufeng123 <wangyufeng@szhtxx.com>
Date: Mon Dec 17 13:38:43 2018 +0800

    Hello Git

commit c308a168e2e5758639f2acc2cb5eed5ae4d34bde
Author: wangyufeng123 <wangyufeng@szhtxx.com>
Date: Fri Nov 23 13:35:30 2018 +0800

    Eureka

Aisino@DESKTOP-DT6A78R MINGW64 /d/software/git/Repository (master)
$ |
```

查看修改内容

```
1 git diff -- HEAD 测试.txt
```

撤销修改

```
1 git checkout -- 测试.txt ///撤销对测试.txt的修改
2 git checkout . ///撤销所有文件的修改
3 git reset --hard 测试.txt ///把测试.txt从暂存区放回工作区
4 git checkout -- 测试.txt
```

删除文件

```
1 rm 测试.txt
2 git checkout -- 测试.txt 回复删除的测试.txt文件
```

GIT远程仓库github的建立，生成public key和github的设置

生成SSH keys

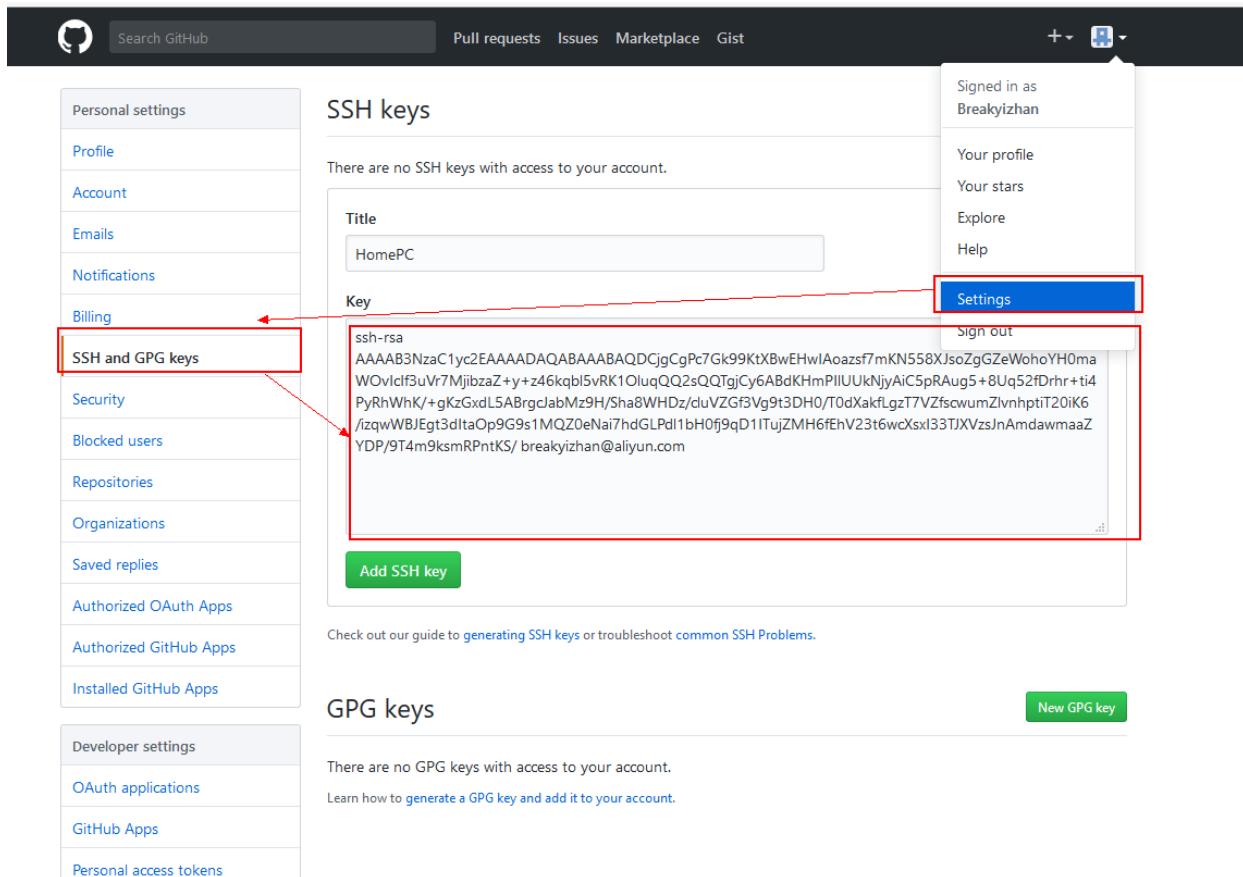
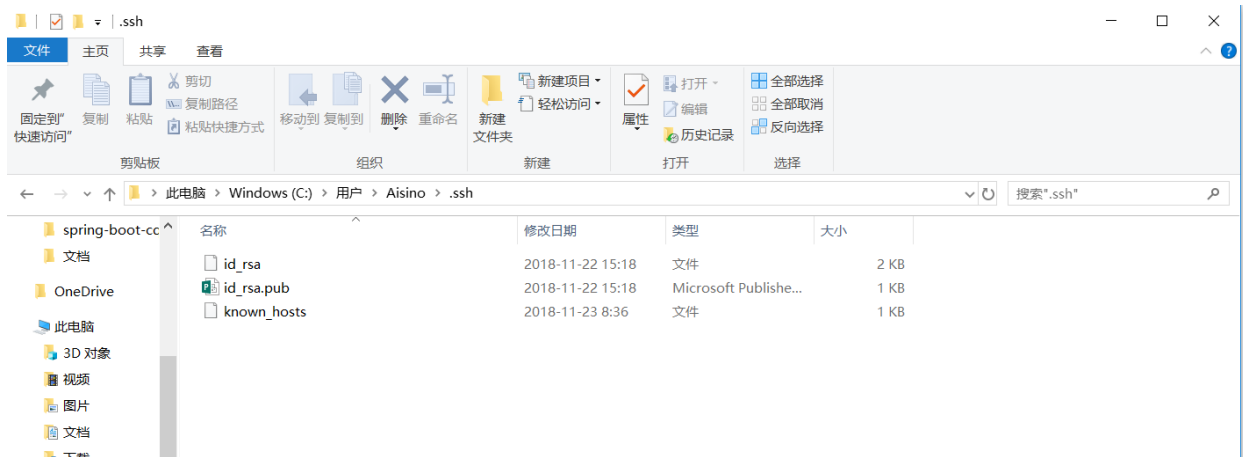
使用SSH keys的目的是为了识别这些代码时谁推送的，生成的SSH key 有两个，用git下面的命令就可以生成，一路回车使用默认值就可以，不用密码。

```
1 ssh-keygen -t rsa -C "Your Email"
```


然后找到生成pub key里面的id_rsa.pub文件，默认目录是

C:\Users\Administrator\.ssh，复制里面的内容到github上面去创建SSH keys

```
Aisino@DESKTOP-DT6A78R MINGW64 /d/software/git/Repository (master)
$ ssh-keygen -t rsa -C "wangyufeng@szhtxx.com"
Generating public/private rsa key pair.
Enter file in which to save the key (/c/Users/Aisino/.ssh/id_rsa):
/c/Users/Aisino/.ssh/id_rsa already exists.
Overwrite (y/n)?
Aisino@DESKTOP-DT6A78R MINGW64 /d/software/git/Repository (master)
$ |
```



创建Git仓库



[Pull requests](#) [Issues](#) [Marketplace](#) [Gist](#)

New repository

Import repository

New gist

New organization

Create a new repository

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

Owner

Breakyizhan

Repository name

HelloGIT

Great repository names are short and memorable. Need inspiration? How about **musical-winner**.

Description (optional)

HelloGIT

☒ 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

© 2017 GitHub, Inc. [Terms](#) [Privacy](#) [Security](#) [Status](#) [Help](#)



[Contact GitHub](#) [API](#) [Training](#) [Shop](#) [Blog](#) [About](#)

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

Set up in Desktop or **HTTPS** SSH `https://github.com/Breakyizhan/HelloGIT.git`

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

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

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

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

```
git remote add origin https://github.com/Breakyizhan/HelloGIT.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

ProTip! Use the URL for this page when adding GitHub as a remote.

关联本地仓库和github仓库

```
1 git remote add origin https://github.com/Breakyizhan/HelloGIT.git
```

测试仓库的连接

```
1 ssh -T git@github.com
```

```
Aisino@DESKTOP-DT6A78R MINGW64 /d/software/git/Repository (master)
$ ssh -T git@github.com
Hi peakbro! You've successfully authenticated, but GitHub does not provide shell
access.
Aisino@DESKTOP-DT6A78R MINGW64 /d/software/git/Repository (master)
```

设置两个仓库

1. 去到你自己的项目目录中，设置这个目录中项目对应的账号。

```
1 git config user.name "newname"
2 git config user.email "newemail"
```

现在git方面，不同的邮箱是需要不同的密钥的。

2. 生成新的密钥

```
1 ssh-keygen -t rsa -C "newemail"
```

记得不要一路回车，在第一个对话的时候输入id_rsa_ownone，这样子我们就在默认目录C:\Users\Administrator\.ssh找到生成pub key里面的id_rsa_ownone.pub文件。

3. 复制id_rsa_ownone.pub文件里面的内容到GitHub上面；

4. 尝试连接GitHub；

5. 重启一下ssh-agent，把key添加到里面去

Git从远程Github仓库克隆代码以及把代码推送到github 远程库

从远程库克隆

1. 在本地想要克隆的文件夹下面创建GIT版本库，以及[建立远程库的连接](#)。

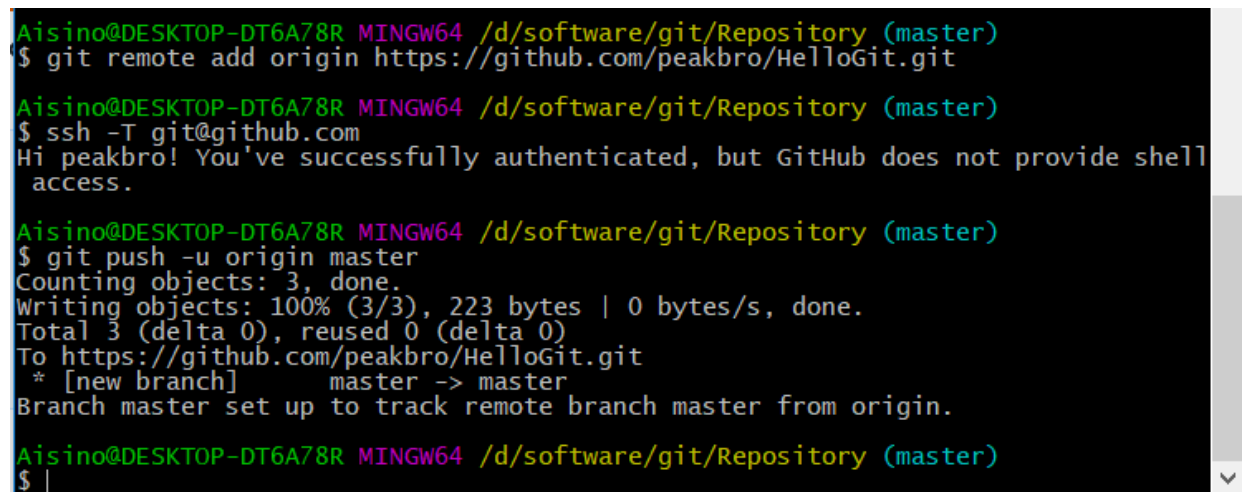
2. 用git clone克隆远程库所在项目的代码

```
1 git clone git@github.com:Breakyizhan/HelloGIT.git
```

把代码推送到远程库

这个推送是推送本地版本库里面的代码，一般是在commit的命令之后执行的

```
1 git push -u origin master
```



```
Aisino@DESKTOP-DT6A78R MINGW64 /d/software/git/Repository (master)
$ git remote add origin https://github.com/peakbro/HelloGit.git

Aisino@DESKTOP-DT6A78R MINGW64 /d/software/git/Repository (master)
$ ssh -T git@github.com
Hi peakbro! You've successfully authenticated, but GitHub does not provide shell
access.

Aisino@DESKTOP-DT6A78R MINGW64 /d/software/git/Repository (master)
$ git push -u origin master
Counting objects: 3, done.
Writing objects: 100% (3/3), 223 bytes | 0 bytes/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/peakbro/HelloGit.git
 * [new branch]      master -> master
Branch master set up to track remote branch master from origin.

Aisino@DESKTOP-DT6A78R MINGW64 /d/software/git/Repository (master)
$ |
```

把代码从远程库更新到本地

```
1 git pull
```

git分支的目的：并行开发

创建分支

```
1 git checkout -b dev ///创建并切换分支
2 相当于如下
```



```
3 git branch dev ///创建分支
4 git checkout dev ///切换分支
```

合并分支

```
1 git checkout master //切换回master
2 git merge dev //合并分支
3 git branch -d dev //删除分支，如果没有需要也可以不用删除
```

暂存区的使用

```
1 $ git add . //现在在dev分支，代码添加文件到暂存区，但是没有commit
2 $ git stash //把代码暂存起来
3 $ git checkout master //切回主线
4 $ git checkout -b EmergencyIssue //创建分支修改EmergencyIssue
5 //After one hour..... EmergencyIssue fixed.
6 $ git checkout master //修改好之后切回主线
7 $ git merge --no-ff -m "merged bug fix EmergencyIssue" iEmergencyIssue //分支合并
8 $ git branch -d EmergencyIssue //删除分支
9 $ git checkout dev //切回dev的分支
10 $ git status //查看状态，但是发现什么都没有
11 $ git stash list //可以看到暂存区存起来的list
12     stash@{0}: WIP on dev: 888888 add merge
13 $ git stash pop //恢复stash后并删除stash的内容
14 $ git stash list
```

GIT查看name和email

```
1 $ git config user.name
2 $ git config user.email
```

GIT修改name和email

```
1 $ git config user.name "username" //修改当前仓库的username
2 $ git config user.email "useremail" //修改当前仓库的用户email
3 $ git config --global user.name "username" //修改全部仓库的username
4 $ git config --global user.email "useremail" //修改全部仓库的用户email
```

查看所有GIT的config

```
1 $ git config --list
```