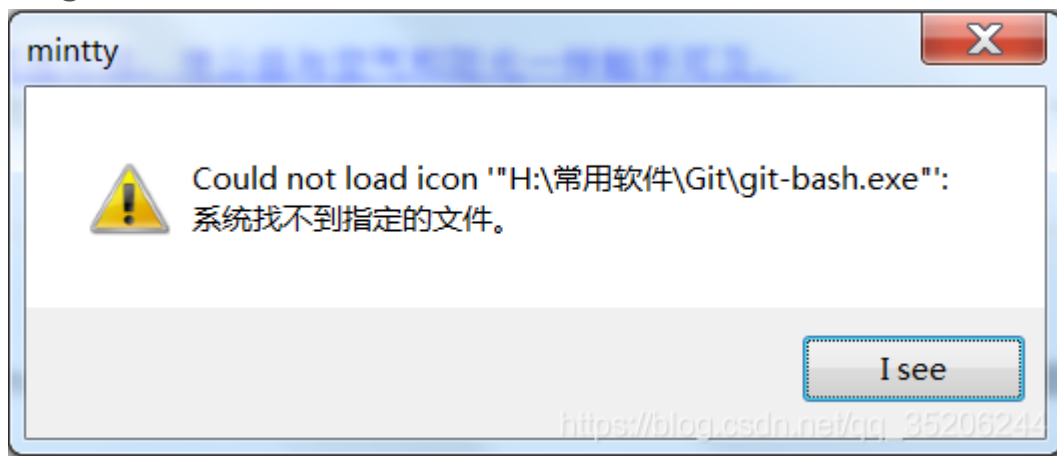


git学习

带着以下问题阅读本文，相信会事半功倍。

- 什么是git？为什么用git而不是其他的版本控制工具，如svn？
- 什么是仓库？本地仓库与远程仓库，常用的远程仓库有哪些，如何建立？
- 怎么用git？如何用git管理文件，实现版本控制？
- git安装尽量避免中文路径，一般会因路径含中文出现以下问题：



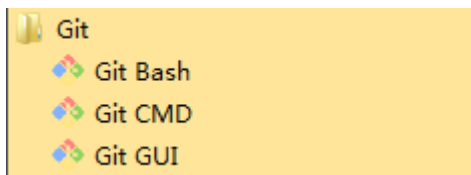
git是个分布式版本管理工具，与集中式版本管理工具svn相反。

Git是一个开源的分布式版本控制系统，可以有效、高速的处理从很小到非常大的项目版本管理，是目前使用范围最广的版本管理工具。

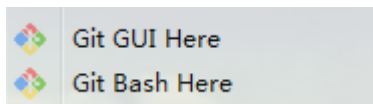
一、Windows上安装Git

一般我们工作的电脑都是Windows系统，要使用git首先要进行安装。从软件管家或者其他平台找到git的安装包，下载后默认安装即可。

安装成功之后，可以在开始菜单里面找到git：



或者在桌面右键，也可以看到：



其中GUI为用户界面模式，Bash为命令行模式，这里就以Bash为例子介绍git的基本使用方法（其实相比于GUI，个人觉得Bash更容易学习理解）

二、设置

由于git是分布式管理工具，需要输入用户名和邮箱以作为标识，因此，在命令行输入下列的命令：

```
1 > git config --global user.name "wangsong"
2 > git config --global user.email "15975301843@163.com"
3 #检查信息是否写入成功
4 git config --list
```

PS：注意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，根据个人情况设置。

查看自己的用户名和邮箱地址：

```
$ git config user.name
```

```
$ git config user.email
```

```
13291@DESKTOP-CUJ4I6P MINGW64 ~  
$ git config user.name  
wyh  
  
13291@DESKTOP-CUJ4I6P MINGW64 ~  
$ git config user.email  
wyhlightstar@163.com
```

修改自己的用户名和邮箱地址:

```
$ git config --global user.name "xxx"
```

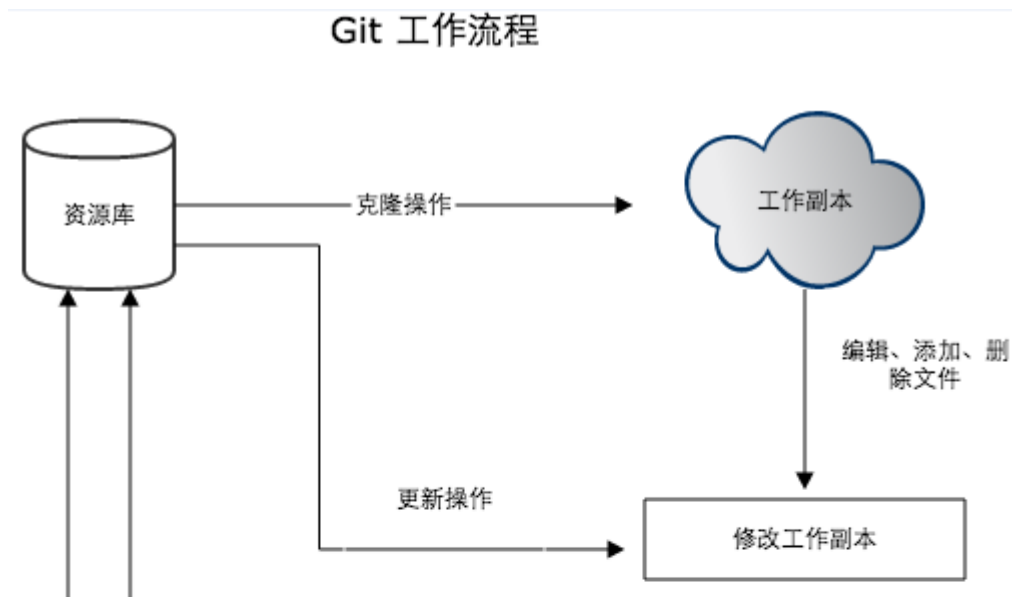
```
$ git config --global user.email "xxx"
```

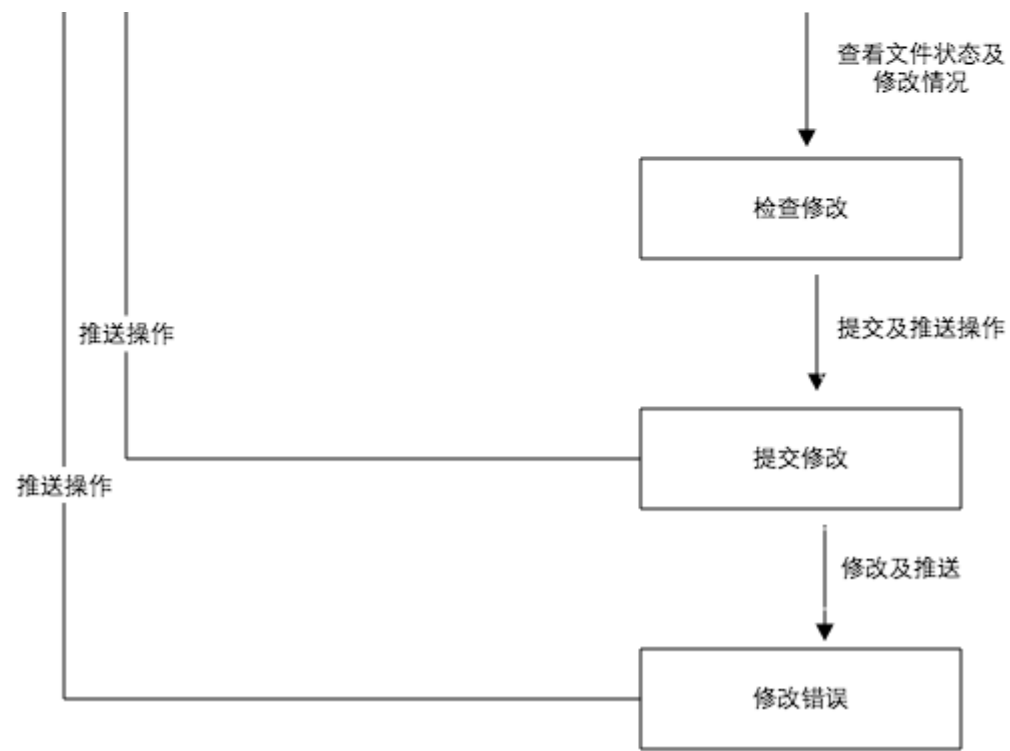
```
13291@DESKTOP-CUJ4I6P MINGW64 ~  
$ git config --global user.name "wyh"  
  
13291@DESKTOP-CUJ4I6P MINGW64 ~  
$ git config --global user.email "1329128091@qq.com"
```

三、工作原理

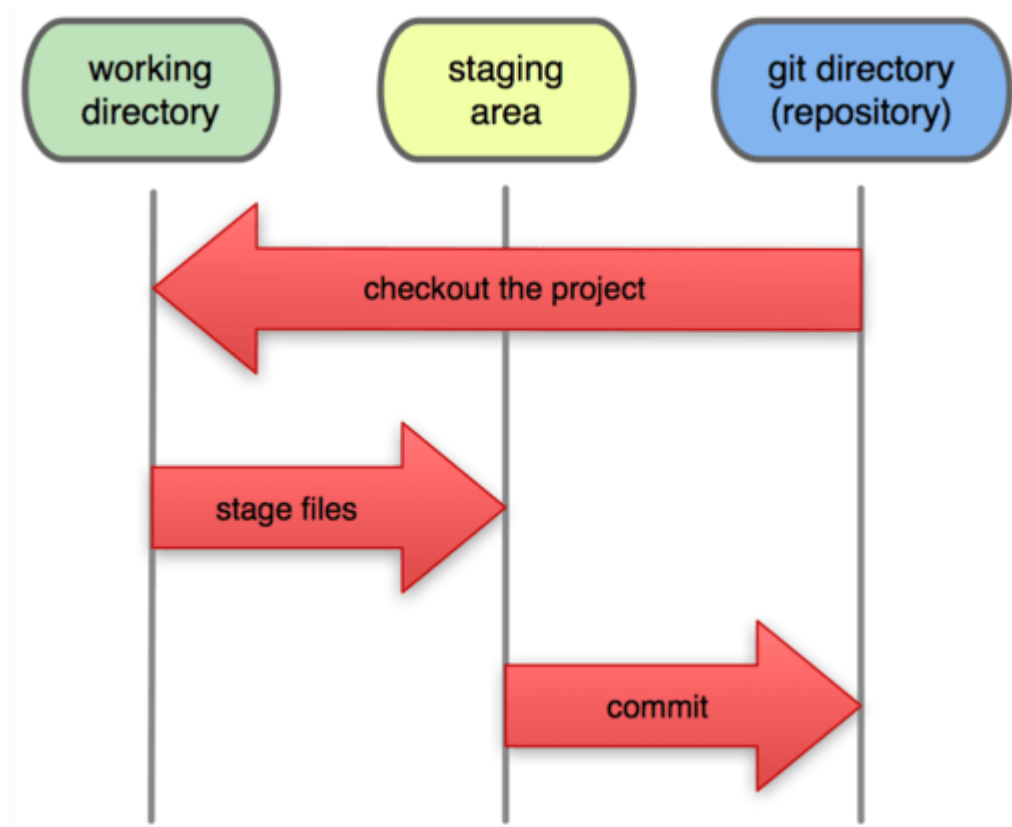
安装好之后, 在使用前先来了解一下Git的工作原理, 是很有必要的一件事, 下面是Git的工作流程和简化原理图:

1、Git工作流程



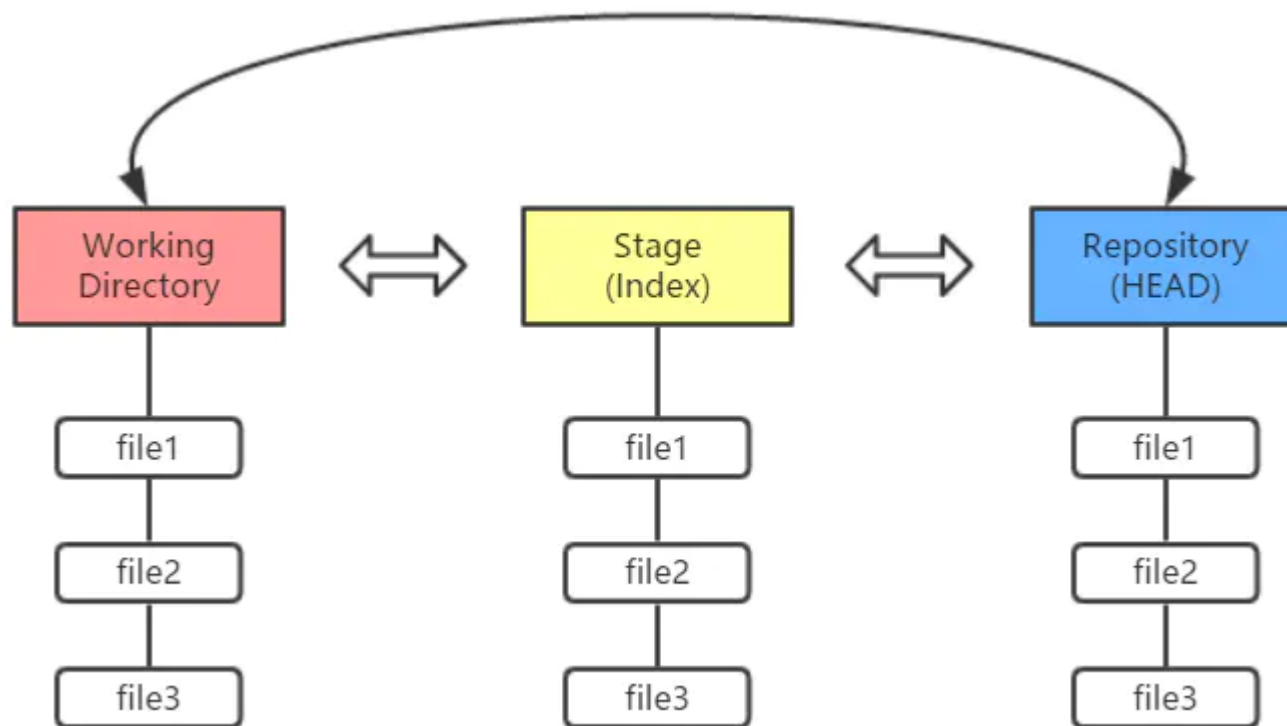


2、Git简化原理图



三棵树

你的本地仓库有 Git 维护的三棵“树”组成，这是 Git 的核心框架。这三棵树分别是：**工作区域、暂存区域和 Git 仓库**



工作区域 (Working Directory) 就是你平时存放项目代码的地方。

暂存区域 (Stage) 用于临时存放你的改动，事实上它只是一个文件，保存即将提交的文件列表信息。

Git 仓库 (Repository) 就是安全存放数据的位置，这里边有你提交的所有版本的数据。其中，HEAD 指向最新放入仓库的版本（这第三棵树，确切的说，应该是 Git 仓库中 HEAD 指向的版本）。

1. 在工作目录中添加、修改文件；
2. 将需要进行版本管理的文件放入暂存区域；
3. 将暂存区域的文件提交到 Git 仓库。

因此，Git 管理的文件有三种状态：已修改 (modified)、已暂存 (staged) 和已提交 (committed)，依次对应上边的每一个流程。

作者: spectre_hola

链接: <https://www.jianshu.com/p/e57a4a2cf077> <<https://www.jianshu.com/p/e57a4a2cf077>>

来源: 简书

著作权归作者所有。商业转载请联系作者获得授权, 非商业转载请注明出处。

四、基本用法

1、创建版本库

版本库就是我们所说的“仓库”, 英文名repository, 你可以理解为一个目录, 这个目录里面的所有文件都可以被Git管理, 文件的修改, 删除Git都能跟踪, 以便任何时刻都可以追踪历史, 或者在将来某个时刻还可以将文件”还原”。

下面是在我的电脑→D盘→TEST文件下, 创建一个名为lianxi的版本库:

```
zhangweigong@temp1 MINGW64 ~ (master)
$ cd D:

zhangweigong@temp1 MINGW64 /d
$ cd test

zhangweigong@temp1 MINGW64 /d/test
$ mkdir lianxi

zhangweigong@temp1 MINGW64 /d/test
$ cd lianxi

zhangweigong@temp1 MINGW64 /d/test/lianxi
$ pwd
/d/test/lianxi
```

命令解析:

cd: 进入某个目录

mkdir: 创建一个文件夹

pwd: 显示当前的目录路径

2、添加文件到版本库

要添加文件到版本库，首先需要将这个目录变为git可以管理的仓库，命令如下：

```
zhangweigong@temp1 MINGW64 /d/test/lianxi
$ git init
Initialized empty Git repository in D:/TEST/lianxi/.git/
```

然后，在lianxi目录下创建一个文件，这里我创建的文件为0409.txt，内容为123456

使用下列的命令，将创建的文件添加到暂存区，然后提交到仓库：

```
zhangweigong@temp1 MINGW64 /d/test/lianxi
$ git init
Initialized empty Git repository in D:/TEST/lianxi/.git/

zhangweigong@temp1 MINGW64 /d/test/lianxi (master)
$ git add 0409.txt

zhangweigong@temp1 MINGW64 /d/test/lianxi (master)
$ git commit -m '0409.txt文件提交'
[master (root-commit) 6573bcc] 0409.txt文件提交
1 file changed, 1 insertion(+)
create mode 100644 0409.txt
```

命令解析：

git add：将文件提交到暂存区

git commit -m：将暂存区文件提交到仓库（单引号内为注释）

3、检查是否有未提交的文件

通过下面的命令，检查该版本库是否有文件未提交：

```
zhangweigong@temp1 MINGW64 /d/test/lianxi (master)
$ git status
On branch master
nothing to commit, working tree clean
```

命令解析：

git status：检查当前文件状态

4、检查文件是否被修改

修改0409.txt的文件，然后重新检查状态：


```

zhangweigong@temp1 MINGW64 /d/test/lianxi (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   0409.txt

no changes added to commit (use "git add" and/or "git commit -a")

zhangweigong@temp1 MINGW64 /d/test/lianxi (master)
$ git diff 0409.txt
diff --git a/0409.txt b/0409.txt
index 4632e06..fc48423 100644
--- a/0409.txt
+++ b/0409.txt
@@ -1,2 @@
-123456
\ No newline at end of file
+123456
+666666
\ No newline at end of file

```

修改文件后，通过命令git status发现，文件已经被修改，但是未提交，如果要检查文件修改了什么内容，可以通过上图中的命令来查看，发现文件的第二行增加了666666的内容。

检查无误后，继续提交修改后的文件，提交命令和上面一样。

命令解析：

git diff：查看文件修改的内容

5、查看历史变更记录

再次修改文件内容，第三行增加233333的内容，然后保存提交：

```

zhangweigong@temp1 MINGW64 /d/test/lianxi (master)
$ git add 0409.txt

zhangweigong@temp1 MINGW64 /d/test/lianxi (master)
$ git commit -m "提交内容233333"
[master 4d3059e] 提交内容233333
1 file changed, 2 insertions(+), 1 deletion(-)

```

现在已经修改了2次文件，可以通过如下命令查看历史修改记录：

```

zhangweigong@temp1 MINGW64 /d/test/lianxi (master)
$ git log
commit 4d3059ece676cf16a8e691240e2216dae8586ffc (HEAD -> master)
Author: zhangweigong <zhangweigong@homedo.com>
Date: Tue Apr 10 10:45:57 2018 +0800
    提交内容233333

commit 59550e0aa6da1a68648418cf4825dbf237ab21b6
Author: zhangweigong <zhangweigong@homedo.com>
Date: Tue Apr 10 10:42:30 2018 +0800
    0409.txt

commit 6573bcc79c08f5d4becde10dc20ba748ebb1996e
Author: zhangweigong <zhangweigong@homedo.com>
Date: Tue Apr 10 10:30:46 2018 +0800
    0409.txt文件提交

zhangweigong@temp1 MINGW64 /d/test/lianxi (master)
$ git log --pretty=oneline
4d3059ece676cf16a8e691240e2216dae8586ffc (HEAD -> master) 提交内容233333
59550e0aa6da1a68648418cf4825dbf237ab21b6 0409.txt
6573bcc79c08f5d4becde10dc20ba748ebb1996e 0409.txt文件提交
  
```

如上图所示：每次提交都会有自己的版本号，当然，入过觉得这样看起来比较费事，可以使用命令，获得精简版本的日志记录。

命令解析：

git log：获得历史修改记录

git log --pretty=oneline：使记录只显示主要的内容，一行显示

6、版本回退

首先通过命令行查看当前的文件内容：

```

zhangweigong@temp1 MINGW64 /d/test/lianxi (master)
$ cat 0409.txt
123456
666666
233333
  
```

然后通过下列的命令，执行版本回退：

```

zhangweigong@temp1 MINGW64 /d/test/lianxi (master)
$ git reset --hard HEAD^
HEAD is now at 59550e0 0409.txt

zhangweigong@temp1 MINGW64 /d/test/lianxi (master)
$ cat 0409.txt
123456
666666

```

可以看到内容已经回退到上一个版本，通过git log查看修改记录，发现最近的一次233333内容的记录已经看不到了，如果想回到最新的版本，可以通过如下命令进行回退：

```

zhangweigong@temp1 MINGW64 /d/test/lianxi (master)
$ git reflog
59550e0 (HEAD -> master) HEAD@{0}: reset: moving to HEAD^
4d3059e HEAD@{1}: commit: 提交内容233333
59550e0 (HEAD -> master) HEAD@{2}: commit: 0409.txt
6573bcc HEAD@{3}: commit (initial): 0409.txt文件提交

zhangweigong@temp1 MINGW64 /d/test/lianxi (master)
$ git reset --hard 4d3059e
HEAD is now at 4d3059e 提交内容233333

zhangweigong@temp1 MINGW64 /d/test/lianxi (master)
$ cat 0409.txt
123456
666666
233333
zhangweigong@temp1 MINGW64 /d/test/lianxi (master)
$

```

从上图可以看到，文件版本又回退到了最新的状态。

命令解析：

cat：查看文件内容

git reset --hard HEAD^：回退到上一个版本

git reflog：获取历史版本号

git reset --hard 版本号：回退到该版本号对应的版本

PS：如果要回退到上上个版本，可以使用git reset --hard HEAD^^命令，但是这样稍显麻烦，如果回退到100个版本之前，只需要执行这个命令即可：git reset --hard HEAD~100；

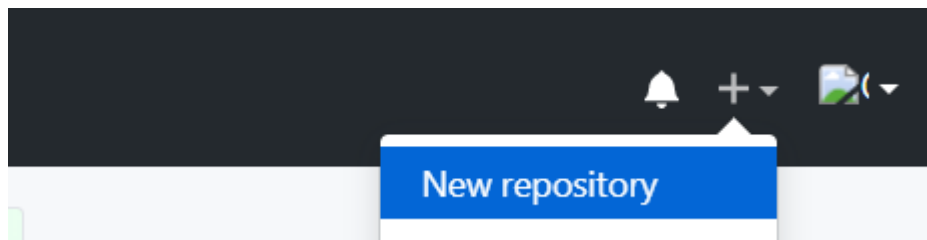
五、远程仓库配置

远程仓库有很多，比如github，国内的码云，局域网自建git服务器，托管在其他地方的服务器，本文以github为例

****网址：********<https://github.com>

4.1注册账户：类似普通的网站新用户注册，使用邮箱注册即可。邮箱需要验证

4.2新建仓库：点击右上角，加号，new repository



下一步

[/](#) [Pull requests](#) [Issues](#) [Marketplace](#) [Explore](#)

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Owner **账户名**

Repository name * **仓库名**

leezhou999 ▾ /

Great repository names are short and memorable. Need inspiration? How about **verbose-engine**?

Description (optional) **仓库描述**

☒ **Public**
Anyone can see this repository. You choose who can commit.

☐ **Private**
You choose who can see and commit to this repository.

Skip this step if you're importing an existing repository.

☐ **Initialize this repository with a README** **readme文件**
This will let you immediately clone the repository to your computer.

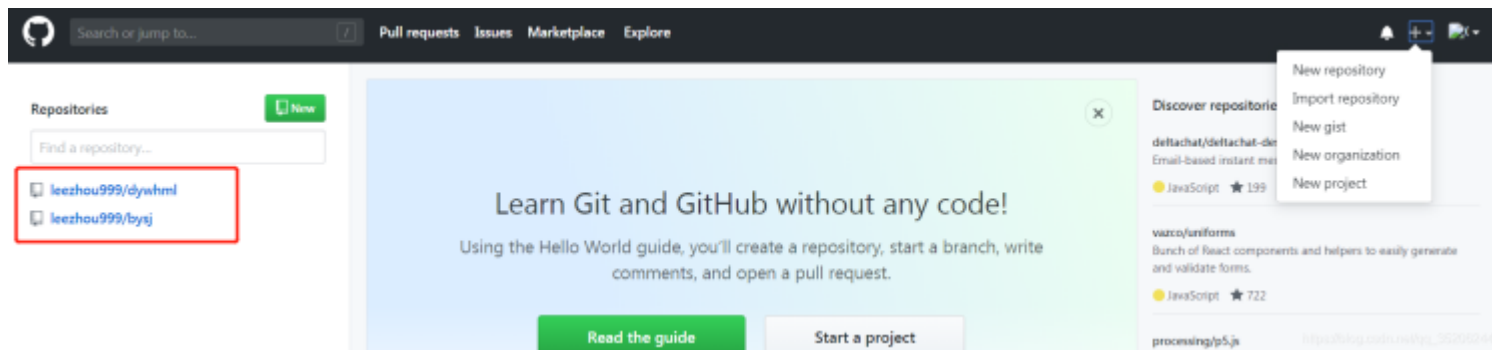
Add .gitignore: **None** ▾

Add a license: **None** ▾ [?](#)

Create repository

https://blog.csdn.net/qq_35206244

如，我已经建立好的仓库：



五、生成（配置）SSH

git客户端安装后，如何和远程仓库，如github连接呢？本文使用SSH。

5.1 用户名

```
1 git config --global user.name "注册名"
2 • 1
```

```
Administrator@USER-20190509ND MINGW64 /f/git/bysj (master)
$ git config --global user.name "leezhou999"
```

5.2 邮箱

```
1 git config --global user.email "注册邮箱"
2 • 1
```

```
Administrator@USER-20190509ND MINGW64 /f/git/bysj (master)
$ git config --global user.email "zhou1372@163.com"
```

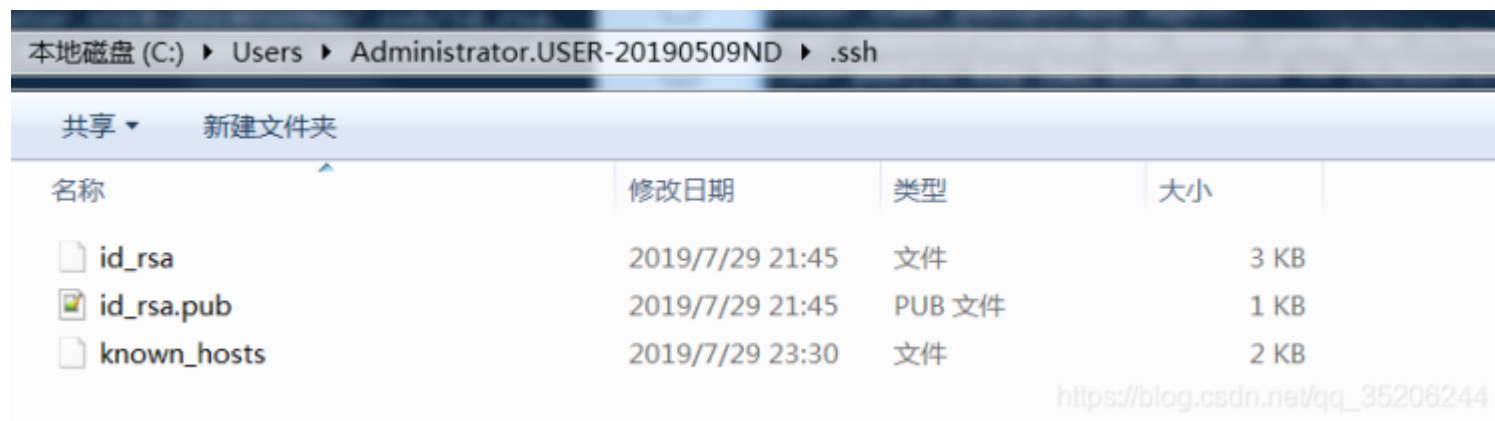
5.3 生成SSH（以有SSH可以跳过这一步）

```
1 ssh-keygen -t rsa -C "自己的邮箱"  
2 • 1
```

生成成功，如下图所示：

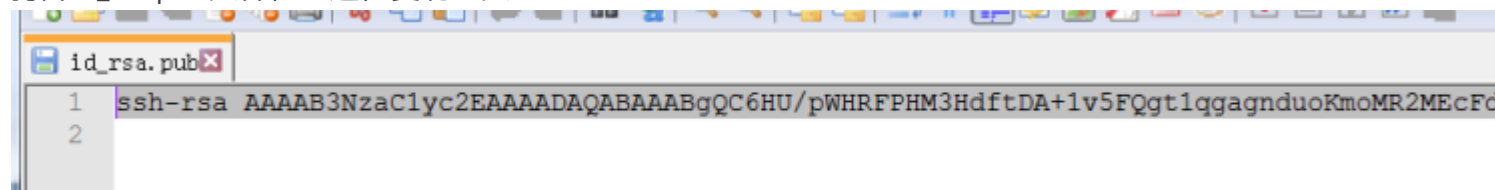
```
Administrator@USER-20190509ND MINGW64 ~  
$ ssh-keygen -t rsa -C "zhou1372@163.com"  
Generating public/private rsa key pair.  
Enter file in which to save the key (/c/Users/Administrator.USER-20190509ND/.ssh  
/id_rsa):  
Created directory '/c/Users/Administrator.USER-20190509ND/.ssh'.  
Enter passphrase (empty for no passphrase):  
Enter same passphrase again:  
Your identification has been saved in /c/Users/Administrator.USER-20190509ND/.ssh/id_rsa.  
Your public key has been saved in /c/Users/Administrator.USER-20190509ND/.ssh/id_rsa.pub.  
The key fingerprint is:  
SHA256:Q1FbtDwSbsk7gsUdFXYZ5MqR6NpBsBS6dVqyFPUgtG8 zhou1372@163.com  
The key's randomart image is:  
+---[RSA 3072]---+  
| .O+*o*o+o |  
| + @ % =. |  
| . X & B . |  
| B @ + + |  
| o S E o |  
| * o |  
| . . |  
+---[SHA256]---+  
  
https://blog.csdn.net/qq_35206244
```

SSH文件存放在C:/User/用户/.ssh下，id_rsa为私钥，id_rsa.pub为公钥。

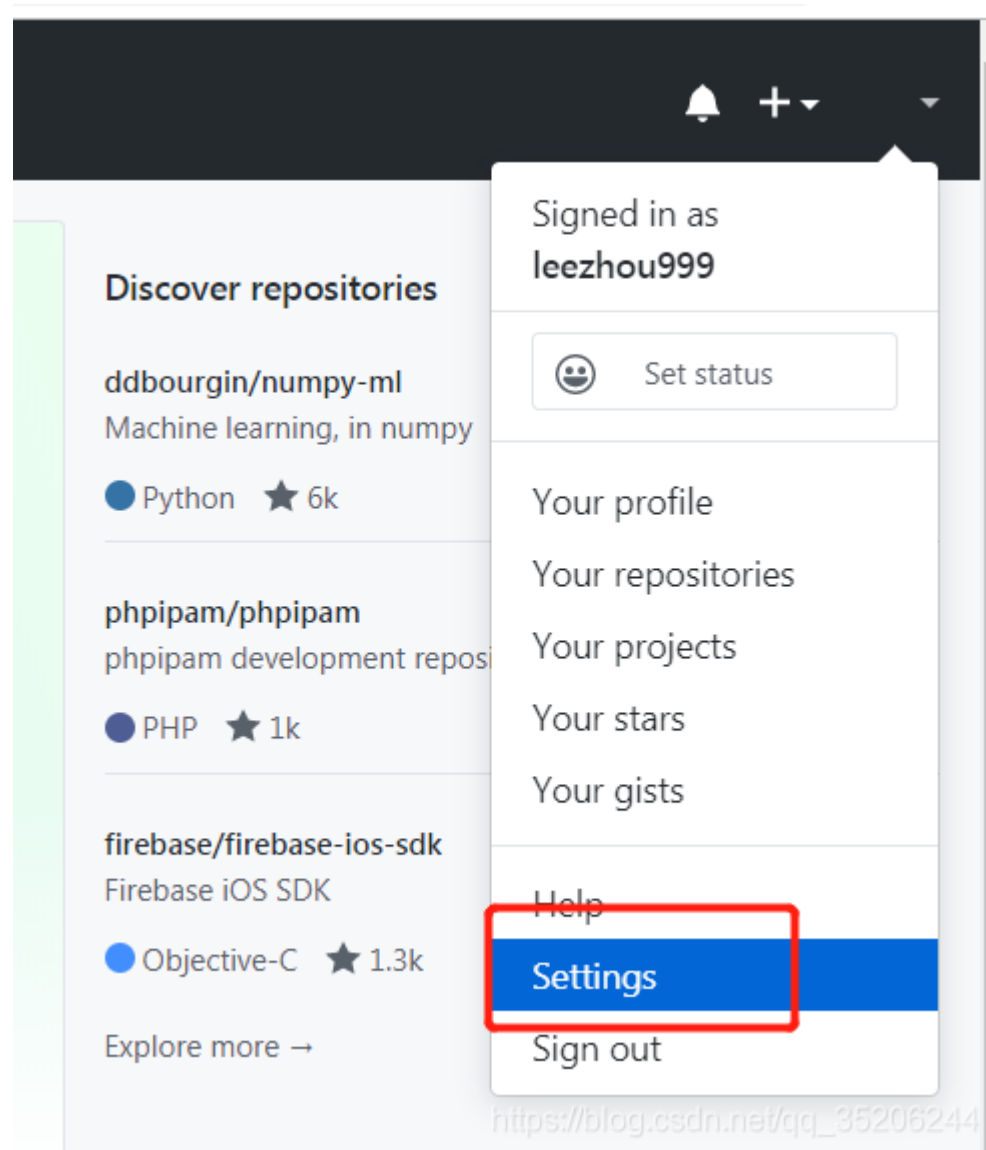


5.4 github配置SSH

打开id_rsa.pub文件，全选，复制全文



github->账户->setting



选择SSH and GPGkeys, New SSH key


The screenshot shows the GitHub 'Personal settings' page. On the left sidebar, the 'SSH and GPG keys' option is highlighted with a red box. The main content area is divided into two sections: 'SSH keys' and 'GPG keys'. In the 'SSH keys' section, there is a 'New SSH key' button (highlighted with a red box) and a list of existing keys. One key is shown with a green key icon, a title 'SSH keys', a long alphanumeric string, and a 'Delete' button. The 'GPG keys' section has a 'New GPG key' button.

Personal settings

- Profile
- Account
- Security
- Emails
- Notifications
- Billing
- SSH and GPG keys**
- Blocked users

SSH keys

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

 SSH keys

6c:25:dc:99:cb:c7:6d:ef:c7:c8:42:e7:d0:28:b8:4f

Added on 29 Jul 2019

Last used within the last week — Read/write

Delete

Check out our guide to [generating SSH keys](#) or [troubleshoot common SSH Problems](#).

GPG keys

New SSH key

New GPG key

自定义一个title，然后粘贴从公钥文件中拷贝的key

SSH keys / Add new

Title

Key

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

Add SSH key

https://blog.csdn.net/qq_35206244

5.5 测试SSH连接

```
1 ssh -T git@github.com
2 • 1
```

按照提示输入yes，回车，提示successfully之类的就说明SSH连接正常，github上的钥匙也会变成绿色

```
Administrator@USER-20190509ND MINGW64 ~  
$ ssh -T git@github.com  
The authenticity of host 'github.com (13.250.177.223)' can't be established.  
RSA key fingerprint is SHA256:nThbg6kXUpJWG17E1IGOCspRomTxdCARLviKw6E5Sj8.  
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes  
Warning: Permanently added 'github.com,13.250.177.223' (RSA) to the list of known hosts.  
Hi leezhou999! You've successfully authenticated, but GitHub does not provide shell access.
```



SSH keys

6c:25:dc:99:cb:c7:6d:ef:c7:c8:42:e7:d0:28:b8:4f

Added on 29 Jul 2019

Last used within the last week — Read/write

Delete

至此，本地git客户端和远程github建立了联系。

六、推送文件至远程仓库

在把文件推送到远程仓库之前，先要了解本地仓库这个概念，此外还有add，commit，push等概念，本文不再赘述。

基本流程:add->commit->push

6.1建立本地仓库

新建一个文件夹



Git

2019/7/29 23:30

文件夹

git bash中执行命令，将该文件夹初始化为一个仓库

```
1 git init  
2 • 1
```

```
Administrator@USER-20190509ND MINGW64 /f
$ cd Git

Administrator@USER-20190509ND MINGW64 /f/Git
$ git init
Initialized empty Git repository in F:/Git/.git/
```

结束以后在文件夹下面会出现一个隐藏的文件夹.git，没有的话，设置一下文件夹选项，显示隐藏文件

名称	修改日期	类型	大小
.git	2019/7/29 23:07	文件夹	
bysj	2019/7/29 23:31	文件夹	
dywhml	2019/7/29 23:17	文件夹	

6.2 推送文件至远程

本文建议，在远程建立好仓库，本地进行clone，然后再添加新文件，最后推送至远程。这样的步骤对新手比较友好。

如：事先在github建立了仓库，bysj，并新建了README文件，此时远程仓库中只有这一个文件。

一、clone远程至本地

```
1 git clone git@github.com:用户名/仓库名.git
2 • 1
```

将远程的bysj仓库及其中的README文件clone至本地

```
Administrator@USER-20190509ND MINGW64 /f/git (master)
$ git clone git@github.com:leezhou999/bysj.git
Cloning into 'bysj'...
Warning: Permanently added the RSA host key for IP address '13.229.188.59' to the list of known hosts.
remote: Enumerating objects: 9, done.
remote: Counting objects: 100% (9/9), done.
remote: Compressing objects: 100% (6/6), done.
remote: Total 9 (delta 2), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (9/9), done.
Resolving deltas: 100% (2/2), done.
```

https://blog.csdn.net/qq_35206244

资料 (F:) > Git > bysj >

共享 ▾ 新建文件夹

名称	修改日期	类型	大小
.git	2019/7/29 23:33	文件夹	
doc	2019/7/29 23:31	文件夹	
src	2019/7/29 23:31	文件夹	
WebContent	2019/7/29 23:31	文件夹	
README.md	2019/7/29 23:30	MD 文件	1 KB

https://blog.csdn.net/qq_35206244

然后，向bysj文件夹中手动添加了doc，src，WebContent三个文件夹及其中的子文件夹和文件等。

可以用ls命令查看bysj文件夹下的文件，可以看到有一个文件，三个文件夹，**使用命令进行add**

二、add

```
1 git add 文件夹1/ 文件夹2/
2 • 1
```

注意:add有多种形式,可以add某个文件,某个文件夹,或直接add当前仓库下所有文件

- 1 git add 单个文件
- 2 git add 文件夹1/ 文件夹2/多个文件夹之间空格隔开
- 3 git add .
- 4 • 1
- 5 • 2
- 6 • 3

```
Administrator@USER-20190509ND MINGW64 /f/git/bysj (master)
$ ls
doc/  README.md  src/  WebContent/

Administrator@USER-20190509ND MINGW64 /f/git/bysj (master)
$ git add doc/ src/ WebContent/
warning: LF will be replaced by CRLF in WebContent/WEB-INF/js/jquery-1.8.3.min.js.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in WebContent/WEB-INF/js/jquery-3.2.1.min.js.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in WebContent/WEB-INF/js/jquery-easyui-1.5.2/easyloader.js.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in WebContent/WEB-INF/js/jquery-easyui-1.5.2/jquery.easyui.min.js.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in WebContent/WEB-INF/js/jquery-easyui-1.5.2/jquery.easyui.mobile.js.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in WebContent/WEB-INF/js/jquery-easyui-1.5.2/jquery.min.js.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in WebContent/WEB-INF/js/jquery-easyui-1.5.2/locale/ea
```

三、commit

- 1 git commit -m “注释”
- 2 • 1

```
Administrator@USER-20190509ND MINGW64 /f/git/bysj (master)
$ git commit -m "bysj first commit"
[master 5b225b0] bysj first commit
1065 files changed, 292572 insertions(+)
create mode 100644 WebContent/META-INF/MANIFEST.MF
create mode 100644 WebContent/WEB-INF/css/Main.css
create mode 100644 WebContent/WEB-INF/css/flexslider.css
create mode 100644 WebContent/WEB-INF/css/reset.css
create mode 100644 WebContent/WEB-INF/css/responsive.css
create mode 100644 WebContent/WEB-INF/css/thems.css
create mode 100644 WebContent/WEB-INF/images/logo4.png
create mode 100644 WebContent/WEB-INF/images/nav1.jpg
create mode 100644 WebContent/WEB-INF/images/nav2.jpg
create mode 100644 WebContent/WEB-INF/images/nav3.jpg
create mode 100644 WebContent/WEB-INF/images/nav4.jpg
create mode 100644 WebContent/WEB-INF/js/Office/Office1.js
create mode 100644 WebContent/WEB-INF/js/Office/OfficeDealDoc.js
create mode 100644 WebContent/WEB-INF/js/Office/Oinfo.js
create mode 100644 WebContent/WEB-INF/js/admin/allDSTGroupInfo.js
create mode 100644 WebContent/WEB-INF/js/admin/allDScore.js
create mode 100644 WebContent/WEB-INF/js/admin/allDSpeechGroup.js
create mode 100644 WebContent/WEB-INF/js/admin/allDTG.js
create mode 100644 WebContent/WEB-INF/js/admin/allDTopic.js
https://blog.csdn.net/qq_3520624
```

四、push

```
1 git push -u origin master
2 • 1
```

```
Administrator@USER-20190509ND MINGW64 /f/git/bysj (master)
$ git push -u origin master
Enumerating objects: 1087, done.
Counting objects: 100% (1087/1087), done.
Delta compression using up to 4 threads
Compressing objects: 100% (976/976), done
Writing objects: 92% (1009/1086), 8.81 MiB | 598.00 KiB/s
```



```
Administrator@USER-20190509ND MINGW64 /f/git/bysj (master)
$ git push -u origin master
Enumerating objects: 1087, done.
Counting objects: 100% (1087/1087), done.
Delta compression using up to 4 threads
Compressing objects: 100% (976/976), done.
Writing objects: 100% (1086/1086), 13.74 MiB | 494.00 KiB/s, done.
Total 1086 (delta 385), reused 0 (delta 0)
remote: Resolving deltas: 100% (385/385), done.
To github.com:leezhou999/bysj.git
   b4e0f2e..5b225b0  master -> master
Branch 'master' set up to track remote branch 'master' from 'origin'
```

成功推送至远程仓库bysj，三个文件夹，一个README文件。

The screenshot shows a GitHub repository page for the repository 'bysj' owned by 'leezhou999'. The repository name 'bysj' is highlighted with a red box. The repository description is 'Leezhou's graduate project who is majored in software engineering.' Below the description, there are statistics: 4 commits, 1 branch, 0 releases, and 1 contributor. A progress bar is shown below these statistics. The repository is on the 'master' branch. There are buttons for 'New pull request', 'Create new file', 'Upload files', 'Find File', and 'Clone or download'. Below these buttons, a table lists the files in the repository:

File	Commit	Time
WebContent	bysj first commit	1 hour ago
doc	bysj first commit	1 hour ago
src	bysj first commit	1 hour ago
README.md	Update README.md	1 hour ago

The files 'WebContent', 'doc', 'src', and 'README.md' are highlighted with a red box. Below the table, the 'README.md' file is selected, showing its content: 'bysj' and 'Leezhou's graduate project who is majored in software engineering.' with a link to 'https://blog.csdn.net/qq_35208244'.

总结：如何推送文件至远程？

- ①建立本地仓库
- ②与远程建立连接，测试

```
1 ssh -T git@github.com
2 • 1
```

③init命令初始化仓库

```
1 git init
2 • 1
```

④手动拷贝文件，并执行add命令

```
1 git add 文件夹1/ 文件夹2/
2 • 1
```

⑤commit命令

```
1 git commit -m “注释”
2 • 1
```

⑥push命令

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