# 证书与key

<https://www.cnblogs.com/einstein-2014731/p/11492637.html>{

[Gitee使用方法](https://www.cnblogs.com/einstein-2014731/p/11492637.html)

原文链接：<https://www.cnblogs.com/yiven/p/8465054.html>

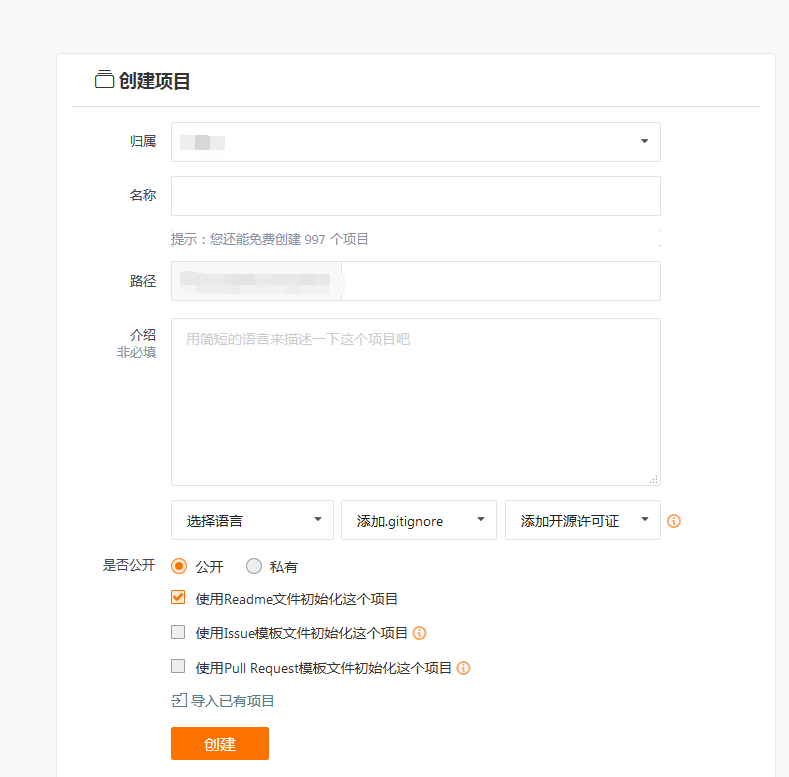
**1.创建仓库：**

**a.创建远程仓库**

登入Gitee后，点击头像旁边的"+"加号-->新建项目



填写项目基本信息后直接点击创建，这样一个远程仓库就已经创建完成了

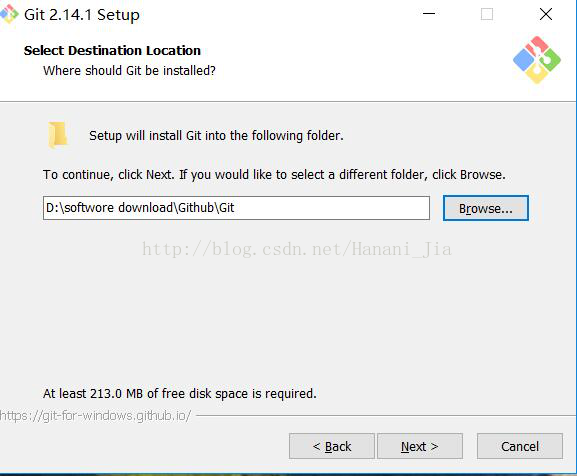


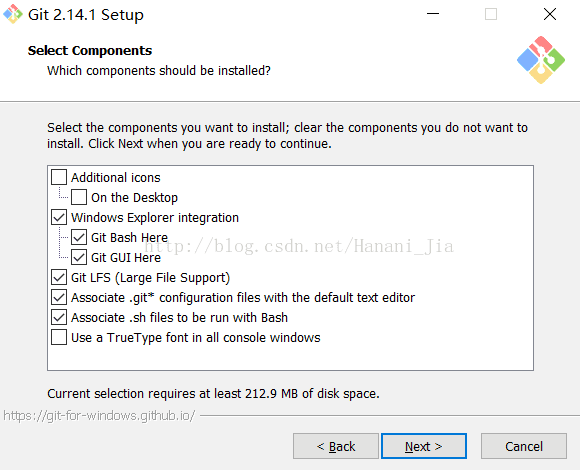
**b.初始化本地仓库**

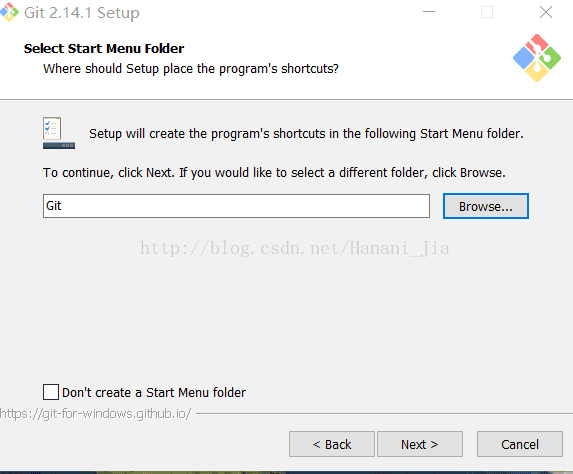
**1)Git Bash安装**

本地仓库的初始化需要借助软件Git Bash 来完成。[点击前往Git官网](https://git-scm.com/" \t "_blank)

下载适用您的电脑的Git版本，下载完成后运行安装程序，过程中使用默认选项即可，若须修改安装路径，须保证安装路径为全英文。







完成安装后运行Git Bash，出现下面这样的命令行窗口，

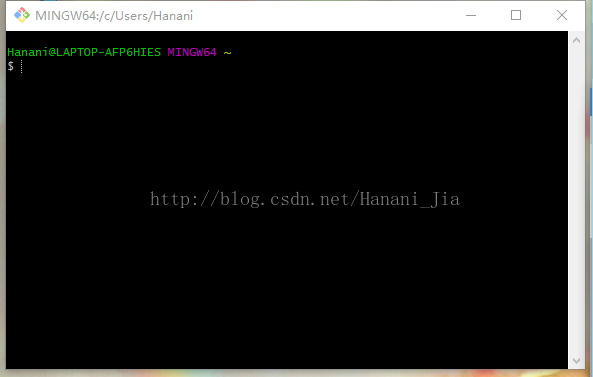
Hanani@LAPTOP-AFP6HIES MINGW64 ~ 是什么意思，

@前是计算机用户名，

@后是计算机名，

MINGW64表示正在运行的Git Bash的信息

~表示根目录，默认情况下是C:\Users\Administrator，这里是C:\Users\Hanani



**2)获取SSHKey**

首先要在本地创建一个ssh key 这个的目的就是你现在需要在你电脑上获得一个密匙。

按如下命令来生成sshkey:

$ ssh-keygen -t rsa -C "youremail@youremail.com"

# Generating public/private rsa key pair...

# 三次回车即可生成 ssh key

查看你的 public key

$ cat ~/.ssh/id\_rsa.pub

# ssh-rsa AAAAB3NzaC1yc2E... youremail@youremail.com

并把他添加到Gitee（gitee.com [SSHKey添加地址](http://git.oschina.net/profile/sshkeys)）



添加后，在终端中输入

#Gitee

$ ssh -T git@gitee.com

#GitHub

$ ssh -T git@github.com

第一次绑定的时候输入上边的代码之后会提示是否continue,输入yes后程序会自动连接，如果要求登录，直接输入登录信息即可。

再次执行上面的命令，检查是否成功连接，如果返回一下信息，则表示添加成功

#Gitee

Welcome to Gitee.com, YourName!

#GitHub

You've successfully authenticated, but GitHub does not provide shell access.

**3) 设置基本信息**

 接下来还需要简单的设置一些东西。

$ git config --global user.name "yourname"

$ git config --global user.email "youremail@youremail.com"

name尽量和码云或GitHub保持一致，但email必须是码云或GitHub注册时使用的邮箱。命令不分前后，没有顺序。

**4)初始化本地库**

然后就是将你的远程仓库克隆到本地，或者你可以在本地初始化一个项目后再进行云端绑定。

克隆

[复制代码](javascript:void(0);)

#Gitee

$ git clone https://gitee.com/yourname/repository

#Github

$ git clone https://github.com/yourname/repository.git

#yourname 您在码云或github注册的用户名

#repository 您创建的远程仓库名称

[复制代码](javascript:void(0);)

本地初始化

[复制代码](javascript:void(0);)

#Gitee

$ cd d:/test //首先在文件系统中创建一个项目文件夹，然后在Git中 cd 到这个工程目录

$ git init //初始化本地项目

$ git remote add origin <远程仓库地址> //绑定远程仓库

#注:地址形式为 https://gitee.com/yourname/test.git 或 git@gitee.com:yourname/test.git

#Github

$ cd d:/test

$ git init

$ git remote add origin <远程仓库地址>

#注：地址形式为 https://github.com/yourname/test.git

复制代码

[复制代码](javascript:void(0);)

**c.更新到远程仓库**

在本地进行编辑后，更新到远程仓库

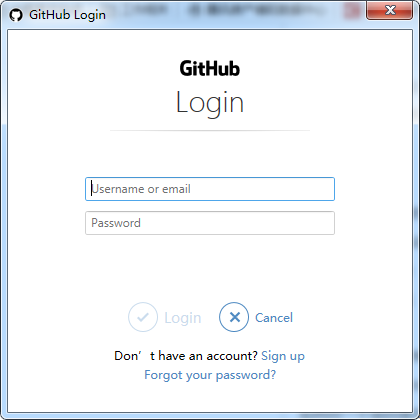
git add . //指定更新内容 . 表示全部更新，test.txt 表示更新指定文件

git commit -m "一些注释说明" //添加更新说明

git push origin master //执行更新操作

在执行更新操作时，由于需要验证远程仓库的用户信息，程序反应较慢，不要以为Git Bash挂掉了哦

如果是第一次更新，那么可能会跳出下面这样的窗口，输入您的用户名和密码，点击登录即可。



如果后续进行更新操作时出现这种情况，那么表示登录失败了，需要重新记录您的用户信息，再次输入用户名、密码登录即可

Logon failed, use ctrl+c to cancel basic credential prompt.

Username for 'https://githun.com'

如何从远程仓库同步最新版本到本地

$ cd d:/test

$ git pull origin master

如何清屏

$ clear

} <https://www.cnblogs.com/einstein-2014731/p/11492637.html> 完

自己实操：

$ pkg install git

$ ssh-keygen -t rsa -C "gitee@注册邮箱.com" #就揿3次enter

Generating public/private rsa key pair.

Enter file in which to save the key (/data/data/com.termux/files/home/.ssh/id\_rsa):

Enter passphrase (empty for no passphrase):

Enter same passphrase again:

Your identification has been saved in /data/data/com.termux/files/home/.ssh/id\_rsa.

Your public key has been saved in /data/data/com.termux/files/home/.ssh/id\_rsa.pub.

The key fingerprint is:

SHA256:lShAH脱敏省略e9Nmc gitee@注册邮箱.com

$ cat ~/.ssh/id\_rsa.pub

ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQCvwLZ/脱敏省略+Ss7q9k= [gitee@注册邮箱.com](mailto:gitee@注册邮箱.com)

#将上述字符串作为ssh key导入gitee.com，参见<https://www.cnblogs.com/einstein-2014731/p/11492637.html>

$ ssh -T git@gitee.com

Hi gitee用户名! You've successfully authenticated, but GITEE.COM does not provide shell access.

#but GITEE.COM does not provide shell access属于正常现象

$ cd /data/data/com.termux/files/home/2011081040gitee

$ git config --global user.name "gitee用户名"

$ git config --global user.email "gitee@注册邮箱.com"

$ git init

2011081225删除文件{

<https://zhuanlan.zhihu.com/p/95508176>

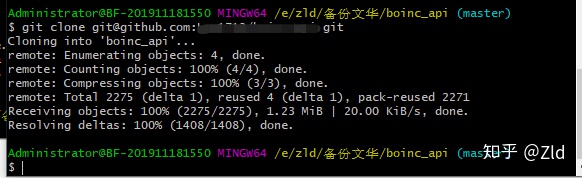
# 如何删除github上的文件

[](https://www.zhihu.com/people/zld-91-16)

[左手牵右手](https://www.zhihu.com/people/zld-91-16)

程序员

1.首先先把文件拉去下来 git clone <file> 看到done基本就成功了。

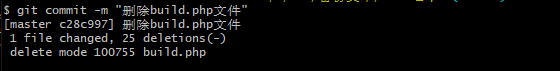


2.首先删除本地仓库文件 git rm <file>

https://pic3.zhimg.com/80/v2-2c35476e6ca064ef1da4114f88fac4e2_720w.png

此时本地文件没有了，github仓库上还有此文件。

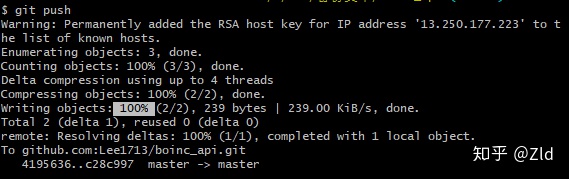
3.提交并添加备注 git commit -m "备注"



4.把删除的文件恢复到最新版本 git checkout

https://pic1.zhimg.com/80/v2-cad55875eb5dabec25ca929ae9afa0e4_720w.png

5. 将本地内容推送到远程仓库



6.刷新github仓库 看到选定删除的文件没有了。

有问题可以私下沟通。

发布于 2019-12-05

自己实操：

$ pwd

/data/data/com.termux/files/home/2011081040gitee

$ git clone git@gitee.com:gitee用户名/it #这是网上你账户里已有的仓库

Cloning into 'it'...

remote: Enumerating objects: 47, done.

remote: Counting objects: 100% (47/47), done.

remote: Compressing objects: 100% (29/29), done.

remote: Total 47 (delta 15), reused 47 (delta 15), pack-reused 0

Receiving objects: 100% (47/47), 22.86 MiB | 3.18 MiB/s, done.

Resolving deltas: 100% (15/15), done.

$ cd it

$ ls

各种文件……

1.docx

1.pdf

$ git rm 1.docx

rm '1.docx'

$ git commit -m "rm\_w"

[master b9ca203] rm\_w

1 file changed, 0 insertions(+), 0 deletions(-)

delete mode 100644 1.docx

$ git checkout

Your branch is ahead of 'origin/master' by 1 commit.

(use "git push" to publish your local commits)

$ git push

Enumerating objects: 3, done.

Counting objects: 100% (3/3), done.

Delta compression using up to 8 threads

Compressing objects: 100% (2/2), done.

Writing objects: 100% (2/2), 214 bytes | 107.00 KiB/s, done.

Total 2 (delta 1), reused 0 (delta 0), pack-reused 0

remote: Powered by GITEE.COM [GNK-5.0]

To gitee.com:gitee用户名/it

116f621..b9ca203 master -> master

上www.gitee.com/gitee用户名/it/看，网上的1.docx也没了

$ pwd

/data/data/com.termux/files/home/2011081040gitee/it

$ ls

各种文件……

1.pdf

$ cp 1.pdf ../..

$ git rm 1.pdf

rm '1.pdf'

$ git commit -m "1"

[master e81b05e] 1

1 file changed, 0 insertions(+), 0 deletions(-)

delete mode 100644 1.pdf

$ git checkout

Your branch is ahead of 'origin/master' by 1 commit.

(use "git push" to publish your local commits)

$ git push

Enumerating objects: 3, done.

Counting objects: 100% (3/3), done.

Delta compression using up to 8 threads

Compressing objects: 100% (2/2), done.

Writing objects: 100% (2/2), 223 bytes | 223.00 KiB/s, done.

Total 2 (delta 1), reused 0 (delta 0), pack-reused 0

remote: Powered by GITEE.COM [GNK-5.0]

To gitee.com:gitee用户名/it

b9ca203..e81b05e master -> master

上www.gitee.com/gitee用户名/it/看，1.pdf也没了

}2011081225删除文件 完

# 上传文件

$ pwd ; ls

/data/data/com.termux/files/home/2011081040gitee/it

各种文件……

$ touch test.txt #可以是创建，也可以是从别处传过来

$ git add test.txt

$ git commit -m "add\_test"

[master d10c482] add\_test

1 file changed, 0 insertions(+), 0 deletions(-)

create mode 100644 test.txt

$ git checkout

Your branch is ahead of 'origin/master' by 1 commit.

(use "git push" to publish your local commits)

$ git push

Enumerating objects: 4, done.

Counting objects: 100% (4/4), done.

Delta compression using up to 8 threads

Compressing objects: 100% (2/2), done.

Writing objects: 100% (3/3), 263 bytes | 263.00 KiB/s, done.

Total 3 (delta 1), reused 0 (delta 0), pack-reused 0

remote: Powered by GITEE.COM [GNK-5.0]

To gitee.com:gitee用户名/it

e81b05e..d10c482 master -> master

上www.gitee.com/gitee用户名/it/ 看，也出现了test.txt

对于gitee、介于10与100MB之间的文件，可以用此方法

# 新建文件夹

似乎不能上传空文件夹

$ pwd ; ls

各种文件……

$ mkdir testDir

$ cd testDir/ ; touch desktop.ini

$ cd ..

$ git add testDir #不可以带/

$ git commit -m "1" ; git checkout ; git push

上www.gitee.com/gitee用户名/it/看，也出现了testDir

# 删除刚刚创建的文件夹

$ git rm -r testDir/ ; git commit -m "1" ; git checkout ; git push

rm 'testDir/desktop.ini'

[master 06c5b2e] 1

1 file changed, 0 insertions(+), 0 deletions(-)

delete mode 100644 testDir/desktop.ini

Your branch is ahead of 'origin/master' by 1 commit.

(use "git push" to publish your local commits)

Enumerating objects: 3, done.

Counting objects: 100% (3/3), done.

Delta compression using up to 8 threads

Compressing objects: 100% (2/2), done.

Writing objects: 100% (2/2), 207 bytes | 207.00 KiB/s, done.

Total 2 (delta 1), reused 0 (delta 0), pack-reused 0

remote: Powered by GITEE.COM [GNK-5.0]

To gitee.com:gitee用户名/it

ba7ca11..06c5b2e master -> master

上www.gitee.com/gitee用户名/it/看，testDir没了

# termux/linux/unix 同一用户切换gitee、github{

参考<https://blog.csdn.net/weixin_30399155/article/details/96124793>{

**git中的ssh和https方式的使用（gitee为例）**



[weixin\_30399155](https://me.csdn.net/weixin_30399155) 2018-12-16 18:25:00 https://csdnimg.cn/release/blogv2/dist/pc/img/articleReadEyes.png1101 https://csdnimg.cn/release/blogv2/dist/pc/img/tobarCollect.png收藏

文章标签： [运维](https://www.csdn.net/gather_29/MtzaAg4sOTUzMC1ibG9n.html) [git](https://www.csdn.net/gather_27/MtzaYgwsMzEzMy1ibG9n.html) [开发工具](https://www.csdn.net/gather_21/MtTaEg0sMjc4NDUtYmxvZwO0O0OO0O0O.html)

版权

在使用git管理代码，或者使用github，国内的码云(gitee)的时候，有两种方式可以使用，分别是https和ssh，以下均使用gitee为例。

* 1. **ssh方式**

   配置ssh，如果不配置ssh的话，clone项目的时候会报错，原因是没有配置公钥。

1 Permission denied (publickey)

2 Could not read from remote repository.

                   首先，进入用户目录，查看本地是否已经存在公钥文件。

1 cd ~/.ssh

2 ls

                  如果显示文件夹不存在，则本地还没有生成密钥文件，则使用git提供的密钥生成工具进行生成。

1 ssh-keygen -t rsa -C "xxxxx@xxxxx.com"

                 注意，回车后有几次需要回车，第二部需要你设置密码，如果不设置直接回车，则以后在使用ssh方式拉取提交代码的时候不需要再输入密码。

                 完成后，用户文件夹下多出了一个.ssh文件夹，里面有两个文件，一个id\_rsa 和id\_rsa.pub文件，使用命令打开id\_rsa.pub文件，复制内容

1 cat ~/.ssh/id\_rsa.pub

　　　　  然后粘贴到github或gitee上的公钥中，可以给项目添加公钥，也可以给自己的账号添加公钥，二者的区别官方的解释是：

　　　　  项目的 sshkey 只针对项目，且我们仅对项目提供了部署公钥，即项目下的公钥仅能拉取项目，这通常用于生产服务器拉取仓库的代码。 而用户的 key 则是针对用户的，用户添加了 key 就对用户名下的项目和用户参加了的项目具有权限，一般而言， 用户的key具有推送和拉取的权限，而项目的 key 则只具有拉取权限。



 　　　　　　　　　　 输入命令进行测试：

1 ssh -T git@gitee.com

                                      如果返回 ”Hi gitee用户名xxxxxx......“ 说明配置成功了。

                                      设置好了之后，然后拉取代码，第一次拉取的时候，会询问是否信任连接，yes就行了，就可以正常拉取，提交代码了，也不需要设置每次都输入账号和密码了。

                        注意，信任完gitee的连接后，.ssh文件夹里多出了一个known\_hosts文件，这里就记录了gitee的相关连接信息。这里会有一个小问题，那假如我有多个git账号，比如：1，gitee账号，用于自己个人 2，gihub账号，用于公司开发，那么怎么配置多个密钥呢？

                       解决办法：1，生成多个密钥文件

1 ssh-keygen -t rsa -C 'xxxxx@company.com' -f ~/.ssh/gitee\_id\_rsa

2 ssh-keygen -t rsa -C 'xxxxx@qq.com' -f ~/.ssh/github\_id\_rsa

                                          2，在 ~/.ssh 目录下新建一个config文件，添加如下内容（其中Host和HostName填写git服务器的域名，IdentityFile指定私钥的路径）

1 # gitee

2 Host gitee.com

3 HostName gitee.com

4 PreferredAuthentications publickey

5 IdentityFile ~/.ssh/gitee\_id\_rsa

6 # github

7 Host github.com

8 HostName github.com

9 PreferredAuthentications publickey

10 IdentityFile ~/.ssh/github\_id\_rsa

                                         3，用ssh命令分别测试

1 $ ssh -T git@gitee.com

2 $ ssh -T git@github.com

* 1. **https方式**

配置credential.helper，首先简单介绍一下credential.helper这个配置项的含义：这个配置项允许用户自行指定git所使用的凭据管理工具。SSH协议并不采用这里讨论的凭证存储。

首先，检查credential.helper配置，输入命令

1 git config -l|grep credential.helper

没有配置的话，显示为空，windows安装如果安装的是GitGUI的时候，默认是勾选manager的，credential.helper=manager，如果没有配置的话，每次拉取，提交都需要密码。使用以下命令配置：

1 git config credential.helper manager

再次尝试pull代码的时候会弹出窗口要求输入用户名密码（只需要输入这一次就ok了），以后就不需要再输入密码了，windows而言，这个凭据放在windows的凭据管理器中

                        整个过程可以这样描述：当配置credential.helper后，git在需要用户名密码时会首先向指定的凭据管理器查找凭据，如果凭据不存在（对应第一次pull），则弹窗提示用户输入用户名密码，然后凭据管理器会记录这个凭据；如果凭据存在（对应第二次pull），则直接使用该凭据进行对应的git操作。

                       注意：当你需要在同一台机器上使用多个git账号这么搞就不行了，因为2个账号必定使用的不同的凭据。

转载于:https://www.cnblogs.com/rucker/p/10127058.html

}参考<https://blog.csdn.net/weixin_30399155/article/details/96124793> 完

自己实操：

$ mkdir 2011081400github

$ cd 2011081400github/

$ ssh-keygen -t rsa -C "github@注册邮箱.com"

Generating public/private rsa key pair.

Enter file in which to save the key (/data/data/com.termux/files/home/.ssh/id\_rsa): githubKey

Enter passphrase (empty for no passphrase):

Enter same passphrase again:

Your identification has been saved in githubKey.

Your public key has been saved in githubKey.pub.

The key fingerprint is:

SHA256:+rC9IIz脱敏省略3o7/Yw github@注册邮箱.com

The key's randomart image is:

+---[RSA 3072]----+

| o\* . .o oo |

| .+ . oo o.o+|

| .. .o o o \*|

| . o + o .o|

| =.S. . .. |

| + + \*o . . |

| o B =o.. . o|

| . = \*o =.|

| . . +. E =|

+----[SHA256]-----+

$ ls -lh

total 16K

-rw------- 1 u0\_a282 u0\_a282 2.5K Nov 8 14:10 githubKey

-rw------- 1 u0\_a282 u0\_a282 566 Nov 8 14:10 githubKey.pub

$ cd ../.ssh/;ls

authorized\_keys id\_rsa id\_rsa.pub known\_hosts

$ pwd

/data/data/com.termux/files/home/.ssh

$ cd -

/data/data/com.termux/files/home/2011081400github

$ cat githubKey.pub

ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQDaHHQ1aBVRCYmPyCPo8chbm0Cx3ZEwLAM6mtFU9s0Y+脱敏省略sxX7cg3ObhBu94vFvQfXDuw6ifBKwli9U4uExBK9A13n8pHw/OoSA7Iw2uD8YbU= github@注册邮箱.com

#将其存入github.com的ssh key

$ mv githubKey\* ../.ssh/

$ cd -

/data/data/com.termux/files/home/.ssh

$ ls -a

. .. authorized\_keys githubKey githubKey.pub id\_rsa id\_rsa.pub known\_hosts

$ touch config #新建一个文件，以使ssh区分不同网站用的不同key

$ nano config

$ cat config

# gitee

Host gitee.com

HostName gitee.com

PreferredAuthentications publickey

IdentityFile ~/.ssh/id\_rsa

# github

Host github.com

HostName github.com

PreferredAuthentications publickey

IdentityFile ~/.ssh/githubKey

$ cd -

/data/data/com.termux/files/home/2011081400github

$ ssh -T git@github.com

The authenticity of host 'github.com (192.30.255.112)' can't be established.

RSA key fingerprint is SHA256:nThbg6kXUpJWGl7E1IGOCspRomTxdCARLviKw6E5SY8.

Are you sure you want to continue connecting (yes/no/[fingerprint])? yes

Warning: Permanently added 'github.com,192.30.255.112' (RSA) to the list of known hosts.

Hi Github用户名! You've successfully authenticated, but GitHub does not provide shell access.

$ git config --global user.name "Github用户名"

$ git config --global user.email [github@注册邮箱.com](mailto:github@注册邮箱.com)

$ rm -rf .git/

$ git init

Initialized empty Git repository in /data/data/com.termux/files/home/2011081400github/.git/

$ git clone git@github.com:Github用户名/it

……这就可以了

} termux/linux/unix 同一用户切换gitee、github 完

# Windows

windows类似。可在清华源下载。安装完git for windows后，将上述的.ssh文件夹放在windows上git里的~/中即可

如果要生成key，也跟上述步骤差不多

# 其他

<https://www.cnblogs.com/jsdy/p/12322172.html> git add . 是添加所有当前目录的所有文件