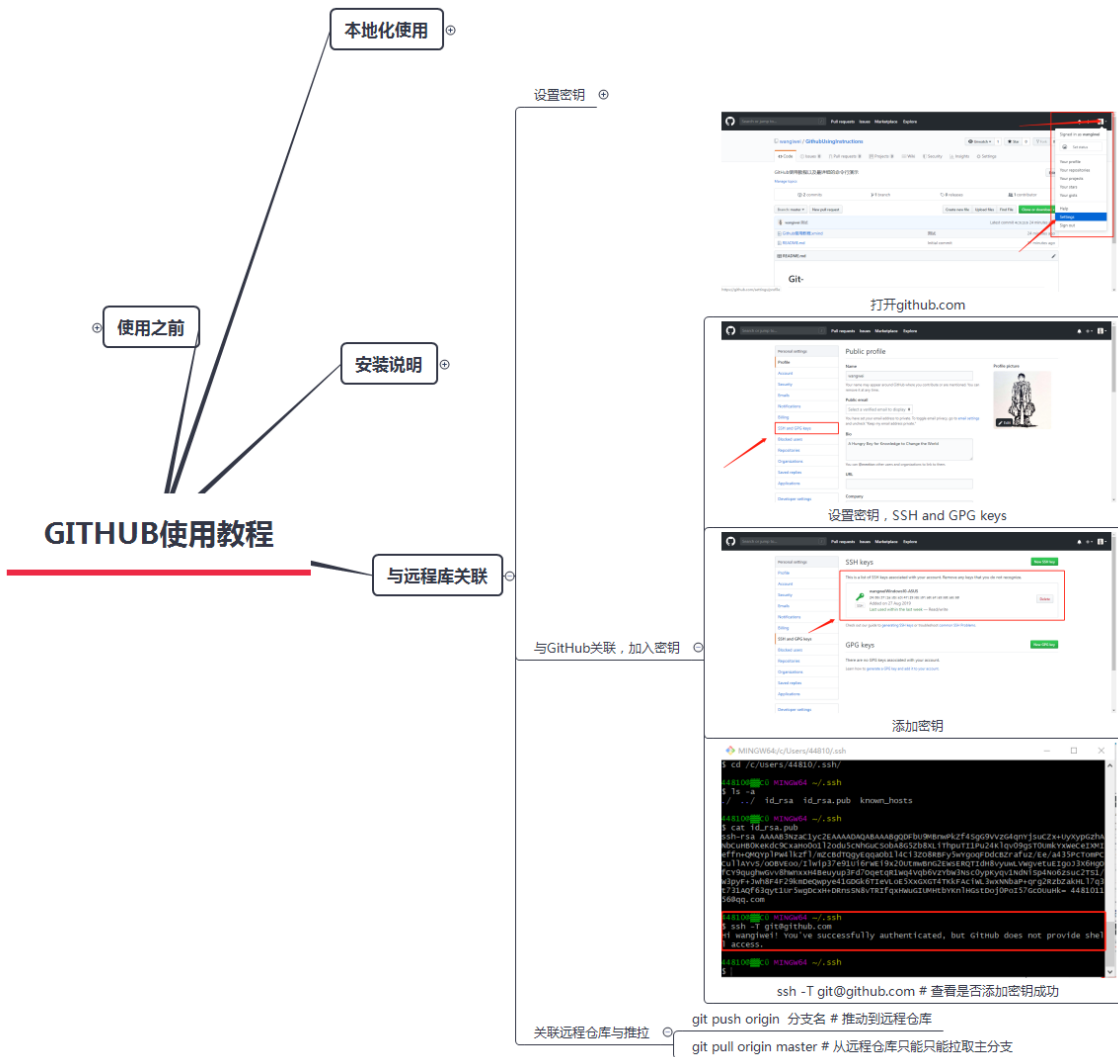


Github使用教程

Github使用教程	1
1. 安装说明	4
1.1. 下载地址.....	4
1.1.1. 国外官网： https://git-scm.com/downloads	4
1.1.2. 国内镜像： https://github.com/waylau/git-for-win	4
2. 本地化使用	4
2.1. 创建空项目	4
2.1.1. 第一种方法，从本地创建，不建议	5
2.1.1.1. cd 某个文件夹（拖取就好）	5
2.1.1.2. git clone 框内链接	5
2.1.2. 第二种方法，从云账户拉取	5
2.1.2.1. mkdir 文件夹 # 新建文件夹	5
2.1.2.2. git init # 初始化仓库，可以得到.git文件	5
2.2. 添加文件.....	6
2.2.1. git add 文件名 # 添加该目录下此文件	6
2.2.2. git add . # 添加该目录下所有文件.....	6
2.3. 提交文件修改.....	6
2.3.1. git commit -m "添加修改说明" # 提交上一步所添加（可多个）的文件并且标记修改说明	6
2.4. 查看输入日志.....	6
2.4.1. git log # 查看输入命令日志.....	6
2.4.2. git log --pretty=oneline # 查看输入命令日志—精简.....	7
2.4.3. git reflog # 保存输入命令日志	7
2.4.4. git diff # 查看版本差异，只是对代码等有效	8
2.5. 时光穿梭机.....	8
2.5.1. 版本回退	8
2.5.1.1. git reset --hard HEAD^ # 回退上一版本.....	8
2.5.1.2. git reset --hard HEAD^^ # 回退上上版本	9
2.5.1.3. git reset --hard HEAD~100 # 回退先前100个版本.....	9
2.5.1.4. git reset --hard 具体版本号 # 回退到具体版本，版本号已由 git log 查找	9
2.5.2. 撤销修改	9
2.5.2.1. git checkout -- 文件名 # 回退到某一版本，未使用 git add 文件名 时，回到没修改的版本	9
2.5.2.2. 只是返回到最后一个commit提交的地方.....	9

2.6.	新建文件夹与打开文件.....	9
2.6.1.	touch .gitignore #	
	创建文件, git识别此文件内容, 不传此文件所包含的文件名称	9
2.7.	本地分支管理.....	9
2.7.1.	分支管理	9
2.7.1.1.	创建分支	10
2.7.1.1.1.	git branch 分支名 # 创建一个分支	10
2.7.1.2.	查看所在分支	10
2.7.1.2.1.	git branch # 查看所在分支	10
2.7.1.3.	切换分支	10
2.7.1.3.1.	git checkout 分支名 # 转到分支	10
2.7.1.3.2.	git switch 分支名	10
2.7.1.4.	创建与切换同时进行	10
2.7.1.4.1.	git checkout -b 分支名 # 创建一个分支并转到该分支.....	10
2.7.1.5.	删除分支	10
2.7.1.5.1.	git branch -d 分支名	10
2.7.1.6.	删除远程分支	10
2.7.1.6.1.	git push origin :分支名	10
2.7.1.6.2.	git push origin --delete 分支名	10
2.7.2.	分支合并	10
2.7.2.1.	git merge 被合并的分支名 # 合并到分支.....	10
2.8.	远程分支管理	10
2.8.1.	创建远程分支	10
2.8.1.1.	git push --set-upstream origin 分支名 #	
	创建远程分支并且将现在本地分支直接推到远程仓库分支	11
2.8.2.	查看分支合并图	11
2.8.2.1.	git log --graph	11
2.9.	远程	11
2.9.1.	拉取分支	11
2.9.1.1.	git pull origin 分支名 # 将远程分支拉取到本地刚创建的分支	11
2.10.	标签管理	11
2.10.1.	git tag 标签名 # 打标签	11
2.10.2.	git tag 标签名 commitID # 制定commitId打标签	11
2.10.3.	git tag # 查看所有标签	11
2.10.4.	git tag -a 标签名 -m "标签信息" # 指定标签信息	11
2.10.5.	git show 标签名 # 查看标签名	11
2.10.6.	git push origin 标签名 # 推送标签到远程.....	11
2.10.7.	git tag -d 标签名 # 删除标签.....	11

2.10.8.	<code>git push origin --tags</code> # 一次性推送全部尚未推送到远程的本地标签	11
2.10.9.	<code>git checkout</code> 标签名 # 切换到指定标签	11
2.10.10.	删除远程标签	11
2.10.10.1.	从本地删除	11
2.10.10.1.1.	<code>git tag -d</code> 标签名 # 删除标签	12
2.10.10.2.	再从远程删除	12
2.10.10.2.1.	<code>git push origin :refs/tags/</code> 标签名	12
3.	与远程库关联	12
3.1.	设置密钥	12
3.1.1.	<code>ssh-keygen -t rsa -C</code> "注册时的邮箱帐号"	12
3.1.2.	<code>cd /c/Users/44810/.ssh/</code>	12
3.1.3.	<code>ls -a</code> # 查看所有文件信息, 包括隐藏文件	13
3.1.4.	<code>cat id_rsa.pub</code> # 打开密钥文件	13
3.2.	与GitHub关联, 加入密钥	14
3.2.1.	打开github.com	14
3.2.2.	设置密钥, SSH and GPG keys	14
3.2.3.	添加密钥	15
3.2.4.	<code>ssh -T git@github.com</code> # 查看是否添加密钥成功	15
3.3.	关联远程仓库与推拉	16
3.3.1.	<code>git push origin</code> 分支名 # 推动到远程仓库	16
3.3.2.	<code>git pull origin master</code> # 从远程仓库只能拉取主分支	16
4.	使用之前	16
4.1.	注册GitHub帐号	16
4.1.1.	前往github.com注册帐号, 记住用户名以及注册邮箱账号	16



1. 安装说明

1.1. 下载地址

1.1.1. 国外官网：<https://git-scm.com/downloads>

1.1.2. 国内镜像：<https://github.com/waylau/git-for-win>

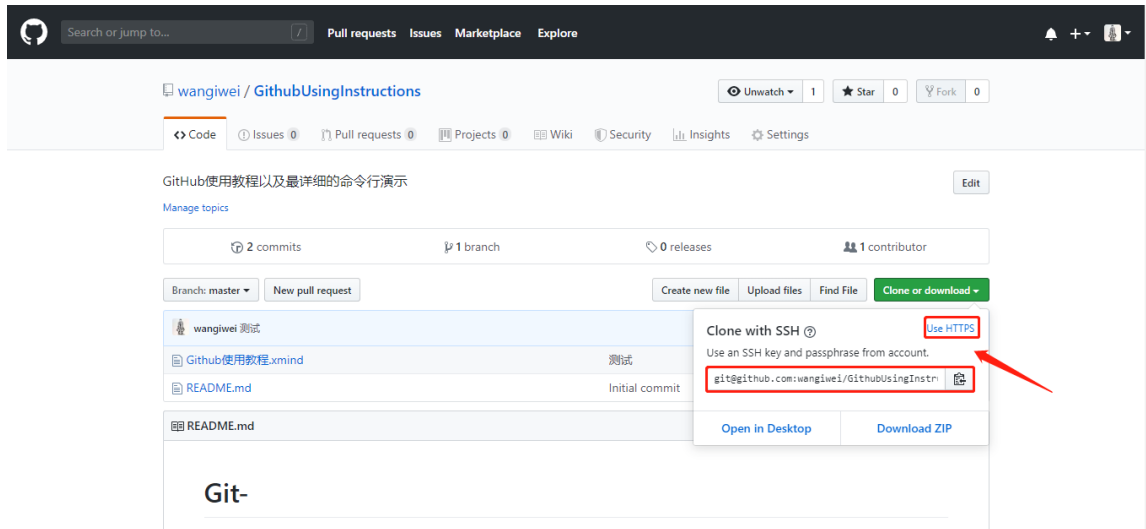
2. 本地化使用

2.1. 创建空项目

2.1.1. 第一种方法，从本地创建，不建议

2.1.1.1. cd 某个文件夹（拖取就好）

2.1.1.2. git clone 框内链接



2.1.2. 第二种方法，从云账户拉取

2.1.2.1. mkdir 文件夹 # 新建文件夹

2.1.2.2. git init # 初始化仓库，可以得到.git文件

```
MINGW64:/c/Users/44810/Desktop/GithubUsingInstructions
$ git status
On branch master
Your branch is ahead of 'origin/master' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean

44810@MINGW64 ~/Desktop/GithubUsingInstructions (master)
$ git push origin master
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 4 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 19.34 KiB | 2.76 MiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To github.com:wangiwei/GithubUsingInstructions.git
   e76392e..4c3c2c6  master -> master

44810@MINGW64 ~/Desktop/GithubUsingInstructions (master)
$ ls -a
./ ../ .git/ Github使用教程.xmind README.md

44810@MINGW64 ~/Desktop/GithubUsingInstructions (master)
$
```

2.2. 添加文件

2.2.1. `git add 文件名` # 添加该目录下此文件

2.2.2. `git add .` # 添加该目录下所有文件

2.3. 提交文件修改

2.3.1. `git commit -m "添加修改说明"` #

提交上一步所添加（可多个）的文件并且标记修改说明

2.4. 查看输入日志

2.4.1. `git log` # 查看输入命令日志

```
MINGW64:/c/Users/44810/Desktop/GithubUsingInstructions
44810@MINGW64 ~/ssh
$ git log
fatal: not a git repository (or any of the parent directories): .git

44810@MINGW64 ~/ssh
$ cd /c/Users/44810/Desktop/GithubUsingInstructions

44810@MINGW64 ~/Desktop/GithubUsingInstructions (master)
$ git log
commit 4c3c2c64e4cfd2eb2e3e7e54a262206f0b371afb (HEAD -> master, origin/master, origin/HEAD)
Author: wangwei <448101156@qq.com>
Date:   Wed Aug 28 10:03:31 2019 +0800

    测试

commit e76392e3cb6148e0d7a2250400259fa07eece98
Author: wangwei <43876952+wangiwei@users.noreply.github.com>
Date:   Wed Aug 28 09:51:01 2019 +0800

    Initial commit

44810@MINGW64 ~/Desktop/GithubUsingInstructions (master)
$
```

2.4.2. git log --pretty=oneline # 查看输入命令日志—精简

```
MINGW64:/c/Users/44810/Desktop/GithubUsingInstructions
Date:   Wed Aug 28 09:51:01 2019 +0800

    Initial commit

44810@MINGW64 ~/Desktop/GithubUsingInstructions (master)
$ git refolg
git: 'refolg' is not a git command. see 'git --help'.

The most similar command is
    reflog

44810@MINGW64 ~/Desktop/GithubUsingInstructions (master)
$ git reflog
4c3c2c6 (HEAD -> master, origin/master, origin/HEAD) HEAD@{0}: commit: 测试
e76392e HEAD@{1}: clone: from git@github.com:wangiwei/GithubUsingInstructions.git

44810@MINGW64 ~/Desktop/GithubUsingInstructions (master)
$ git log --pretty=oneline
4c3c2c64e4cfd2eb2e3e7e54a262206f0b371afb (HEAD -> master, origin/master, origin/HEAD) 测试
e76392e3cb6148e0d7a2250400259fa07eece98 Initial commit

44810@MINGW64 ~/Desktop/GithubUsingInstructions (master)
$
```

2.4.3. git reflog # 保存输入命令日志

```
MINGW64/c/Users/44810/Desktop/GithubUsingInstructions

测试

commit e76392e3cb6148e0d7a2250400259fa07eece98
Author: wangwei <43876952+wangiwei@users.noreply.github.com>
Date: Wed Aug 28 09:51:01 2019 +0800

Initial commit

44810@MINGW64 ~/Desktop/GithubUsingInstructions (master)
$ git refofg
git: 'refofg' is not a git command. see 'git --help'.

The most similar command is
  reflow

44810@MINGW64 ~/Desktop/GithubUsingInstructions (master)
$ git reflow
4c3c2c6 (HEAD -> master, origin/master, origin/HEAD) HEAD@{0}: commit: 测试
e76392e HEAD@{1}: clone: from git@github.com:wangiwei/GithubUsingInstructions.git

44810@MINGW64 ~/Desktop/GithubUsingInstructions (master)
$
```

2.4.4. git diff # 查看版本差异，只是对代码等有效

```
MINGW64/c/Users/44810/Desktop/GithubUsingInstructions

The most similar command is
  reflow

44810@MINGW64 ~/Desktop/GithubUsingInstructions (master)
$ git reflow
4c3c2c6 (HEAD -> master, origin/master, origin/HEAD) HEAD@{0}: commit: 测试
e76392e HEAD@{1}: clone: from git@github.com:wangiwei/GithubUsingInstructions.git

44810@MINGW64 ~/Desktop/GithubUsingInstructions (master)
$ git log --pretty=oneline
4c3c2c64e4cfd2eb2e3e7e54a262206f0b371afb (HEAD -> master, origin/master, origin/HEAD) 测试
e76392e3cb6148e0d7a2250400259fa07eece98 Initial commit

44810@MINGW64 ~/Desktop/GithubUsingInstructions (master)
$ git diff
diff --git "a/Github\344\275\277\347\224\250\346\225\231\347\250\213.xmind" "b/Github\344\275\277\347\224\250\346\225\231\347\250\213.xmind"
index fe673a1..e3ec0bd 100644
Binary files "a/Github\344\275\277\347\224\250\346\225\231\347\250\213.xmind" and "b/Github\344\275\277\347\224\250\346\225\231\347\250\213.xmind" differ

44810@MINGW64 ~/Desktop/GithubUsingInstructions (master)
$
```

2.5. 时光穿梭机

2.5.1. 版本回退

2.5.1.1. git reset --hard HEAD^ # 回退上一版本

2.5.1.2. `git reset --hard HEAD^^` # 回退上上版本

2.5.1.3. `git reset --hard HEAD~100` # 回退先前100个版本

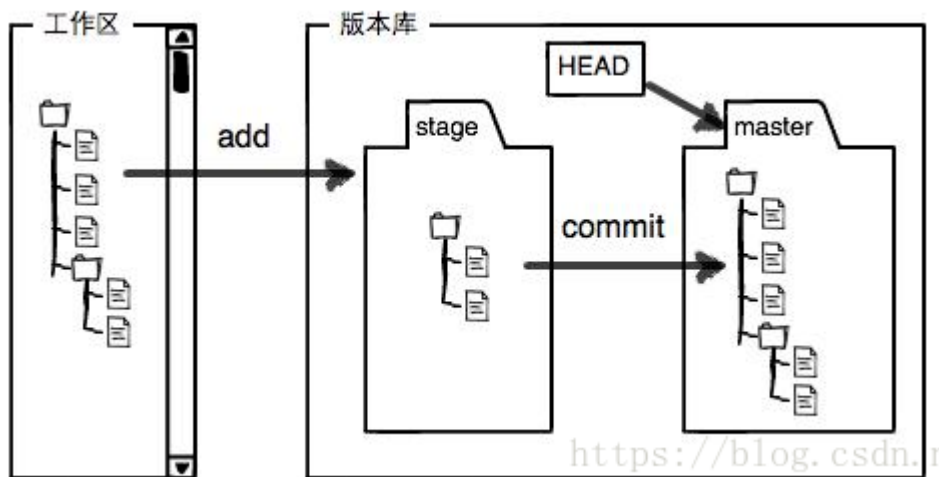
2.5.1.4. `git reset --hard` 具体版本号 # 回退到具体版本，版本号已由 `git log` 查找

2.5.2. 撤销修改

2.5.2.1. `git checkout --` 文件名 # 回退到某一版本，未使用 `git add` 文件名时，回到没修改的版本

2.5.2.2. 只是返回到最后一个commit提交的地方

2.5.2.3.



2.6. 新建文件夹与打开文件

2.6.1. `touch .gitignore` #

创建文件，`git`识别此文件内容，不传此文件所包含的文件名称

2.7. 本地分支管理

2.7.1. 分支管理

2.7.1.1. 创建分支

2.7.1.1.1. `git branch` 分支名 # 创建一个分支

2.7.1.2. 查看所在分支

2.7.1.2.1. `git branch` # 查看所在分支

2.7.1.3. 切换分支

2.7.1.3.1. `git checkout` 分支名 # 转到分支

2.7.1.3.2. `git switch` 分支名

2.7.1.4. 创建与切换同时进行

2.7.1.4.1. `git checkout -b` 分支名 # 创建一个分支并转到该分支

2.7.1.5. 删除分支

2.7.1.5.1. `git branch -d` 分支名

2.7.1.6. 删除远程分支

2.7.1.6.1. `git push origin :分支名`

2.7.1.6.2. `git push origin --delete` 分支名

2.7.2. 分支合并

2.7.2.1. `git merge` 被合并的分支名 # 合并到分支

2.8. 远程分支管理

2.8.1. 创建远程分支

2.8.1.1. git push --set-upstream origin 分支名 #

创建远程分支并且将现在本地分支直接推到远程仓库分支

2.8.2. 查看分支合并图

2.8.2.1. git log --graph

2.9. 远程

2.9.1. 拉取分支

2.9.1.1. git pull origin 分支名 # 将远程分支拉取到本地刚创建的分支

2.10. 标签管理

2.10.1. git tag 标签名 # 打标签

2.10.2. git tag 标签名 commitID # 制定commitId打标签

2.10.3. git tag # 查看所有标签

2.10.4. git tag -a 标签名 -m "标签信息" # 指定标签信息

2.10.5. git show 标签名 # 查看标签名

2.10.6. git push origin 标签名 # 推送标签到远程

2.10.7. git tag -d 标签名 # 删除标签

2.10.8. git push origin --tags # 一次性推送全部尚未推送到远程的本地标签

2.10.9. git checkout 标签名 # 切换到指定标签

2.10.10. 删除远程标签

2.10.10.1. 从本地删除

2.10.10.1.1.git tag -d 标签名 # 删除标签

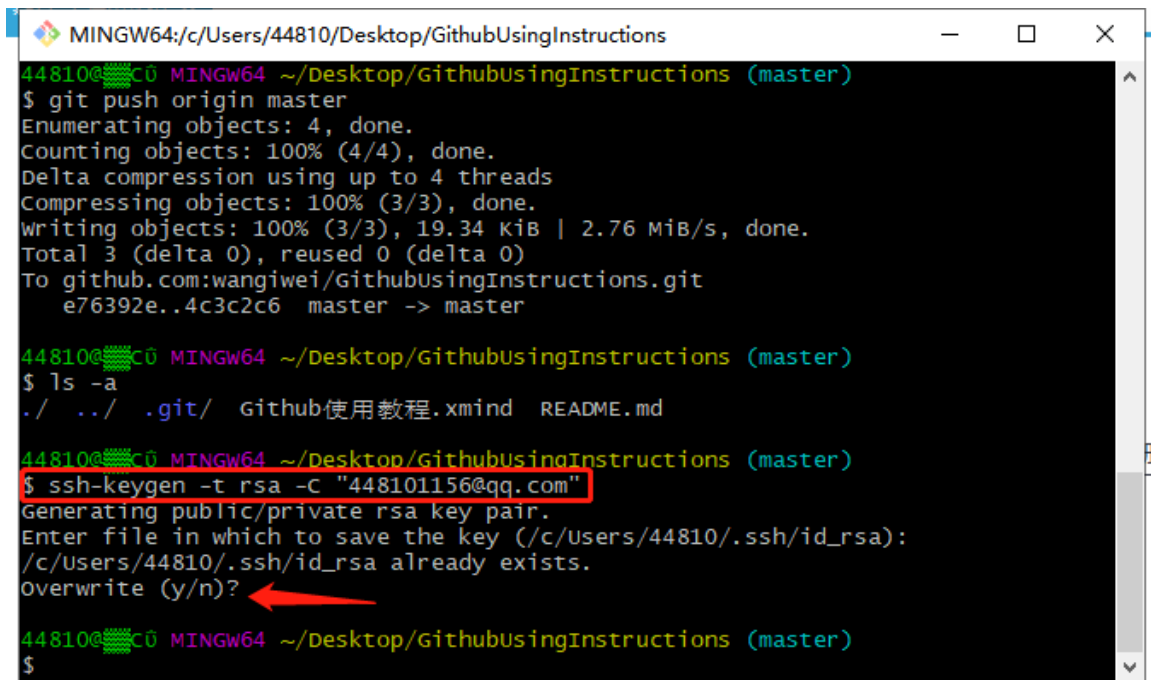
2.10.10.2.再从远程删除

2.10.10.2.1.git push origin :refs/tags/标签名

3. 与远程库关联

3.1. 设置密钥

3.1.1. ssh-keygen -t rsa -C "注册时的邮箱帐号"

A screenshot of a Windows terminal window titled 'MINGW64:/c/Users/44810/Desktop/GithubUsingInstructions'. The terminal shows the following commands and output:
1. `$ git push origin master`
Output: Enumerating objects: 4, done. Counting objects: 100% (4/4), done. Delta compression using up to 4 threads. Compressing objects: 100% (3/3), done. Writing objects: 100% (3/3), 19.34 KiB | 2.76 MiB/s, done. Total 3 (delta 0), reused 0 (delta 0). To github.com:wangiwei/GithubUsingInstructions.git e76392e..4c3c2c6 master -> master
2. `$ ls -a`
Output: ./ ../ .git/ Github使用教程.xmind README.md
3. `$ ssh-keygen -t rsa -C "448101156@qq.com"` (This line is highlighted with a red box)
Output: Generating public/private rsa key pair. Enter file in which to save the key (/c/Users/44810/.ssh/id_rsa): /c/Users/44810/.ssh/id_rsa already exists. overwrite (y/n)?
A red arrow points to the 'overwrite (y/n)?' prompt.
4. The prompt returns to `$`.

3.1.2. cd /c/Users/44810/.ssh/

```
MINGW64:/c/Users/44810/Desktop/GithubUsingInstructions
44810@C0 MINGW64 ~/Desktop/GithubUsingInstructions (master)
$ git push origin master
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 4 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 19.34 KiB | 2.76 MiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To github.com:wangiwei/GithubUsingInstructions.git
   e76392e..4c3c2c6  master -> master

44810@C0 MINGW64 ~/Desktop/GithubUsingInstructions (master)
$ ls -a
./ ../ .git/ Github使用教程.xmind README.md

44810@C0 MINGW64 ~/Desktop/GithubUsingInstructions (master)
$ ssh-keygen -t rsa -C "448101156@qq.com"
Generating public/private rsa key pair.
Enter file in which to save the key (/c/Users/44810/.ssh/id_rsa):
/c/Users/44810/.ssh/id_rsa already exists.
Overwrite (y/n)?

44810@C0 MINGW64 ~/Desktop/GithubUsingInstructions (master)
$
```

3.1.3. ls -a # 查看所有文件信息，包括隐藏文件

```
MINGW64:/c/Users/44810/.ssh
44810@C0 MINGW64 ~/Desktop/GithubUsingInstructions (master)
$ ssh-keygen -t rsa -C "448101156@qq.com"
Generating public/private rsa key pair.
Enter file in which to save the key (/c/Users/44810/.ssh/id_rsa):
/c/Users/44810/.ssh/id_rsa already exists.
Overwrite (y/n)?

44810@C0 MINGW64 ~/Desktop/GithubUsingInstructions (master)
$ ^C

44810@C0 MINGW64 ~/Desktop/GithubUsingInstructions (master)
$ ls -a
./ ../ .git/ Github使用教程.xmind README.md

44810@C0 MINGW64 ~/Desktop/GithubUsingInstructions (master)
$ cd /c/Users/44810/.ssh/

44810@C0 MINGW64 ~/.ssh
$ ls -a
./ ../ id_rsa id_rsa.pub known_hosts

44810@C0 MINGW64 ~/.ssh
$
```

3.1.4. cat id_rsa.pub # 打开密钥文件

```
MINGW64:/c/Users/44810/.ssh
44810@MINGW64 ~/Desktop/GithubUsingInstructions (master)
$ ls -a
./ ../ .git/ Github使用教程.xmind README.md

44810@MINGW64 ~/Desktop/GithubUsingInstructions (master)
$ cd /c/Users/44810/.ssh/

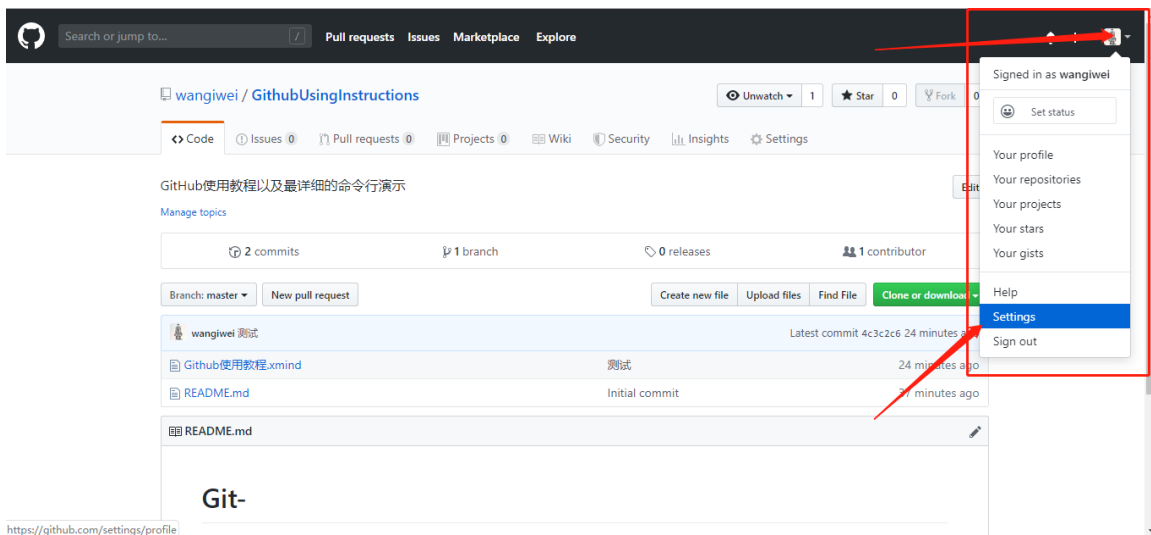
44810@MINGW64 ~/.ssh
$ ls -a
./ ../ id_rsa id_rsa.pub known_hosts

44810@MINGW64 ~/.ssh
$ cat id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQgQDFbU9MBnWPKZF4SgG9VVZG4qnYjsuCZx+uyXypGzha
NbcuH80kEkdC9CxaHo0o1l2odu5cNhGuCsoba8G5Zb8XLiThpuTI1Pu24Klqv09gsT0UmkyxweCeIXMI
effn+QMqYp1Pw41kzf1/mZcBdTQgyEqqaob1l4ci3ZO8RBFy5wYgoqFDdcBZr afuz/Ee/a435PcTomPC
Cu1lAYvS/oOBVEoo/I1wip37e91Ui6rWEi9x20UtmwBnG2EwsERQTidH8vyuWLVwgvetuEigoJ3X6HgO
fCY9qughwGvv8hwnxxH4Beuyup3Fd7OqetqR1Wq4Vqb6Vzybw3Nsc0ypkyqv1NdNiSp4No6zsuc2TS1/
w3pyF+Jwh8F4F29kmDeQwpye41GDGk6TIeVL0E5XxGXGT4TKkFACiWL3wxNNbaP+qrg2RzbZakHL17q3
t731AQf63qyt1Ur5wgDcxH+DRnsSN8VTRIfqxHWuGIUMhtbYKn1HGstDoj0PoI57GCOUuHk= 4481011
56@qq.com

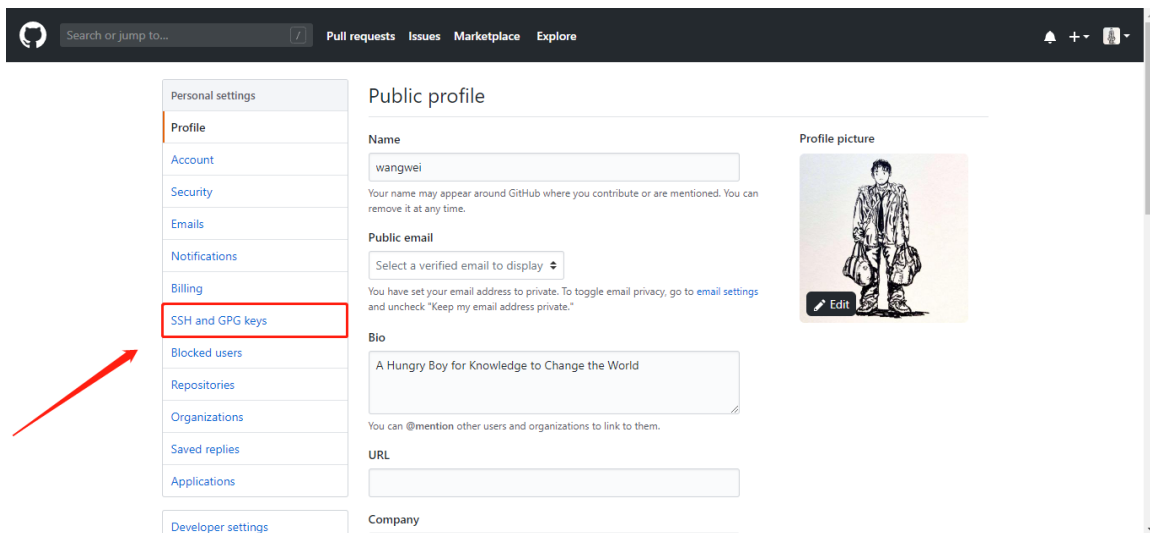
44810@MINGW64 ~/.ssh
$
```

3.2. 与GitHub关联，加入密钥

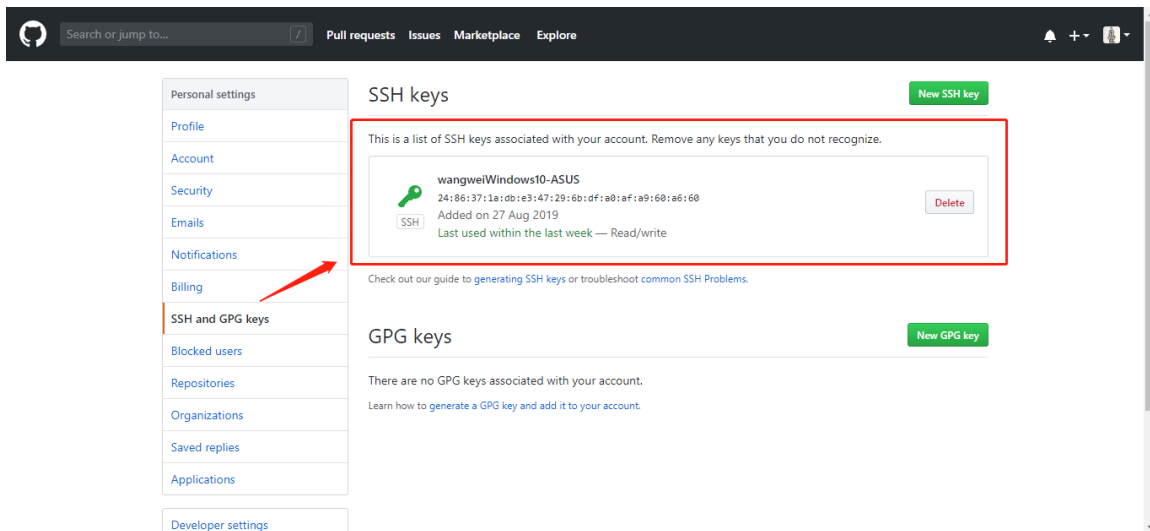
3.2.1. 打开github.com



3.2.2. 设置密钥，SSH and GPG keys



3.2.3. 添加密钥



3.2.4. ssh -T git@github.com # 查看是否添加密钥成功

```
MINGW64:/c/Users/44810/.ssh
$ cd /c/Users/44810/.ssh/

44810@MINGW64 ~/.ssh
$ ls -a
./ ../ id_rsa id_rsa.pub known_hosts

44810@MINGW64 ~/.ssh
$ cat id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQgQDFbU9MBnwPkZf4SgG9VVzG4qnYjsuCZx+UyXypGzha
NbCuHB0KeKdc9CxaHo0o1l2odu5cNhGuCSobA8G5Zb8XLiThpuTI1Pu24K1qv09gsT0UmkyxWeCeIXMI
effn+QMqYp1Pw41kzf1/mZcBdTQgyEqqaOb1l4Ci3Zo8RBFy5wygoqFDdcBZr afuz/Ee/a435PcTomPC
Cul1AYvs/oOBVEoo/iLwip37e91Ui6rWEi9x20utmWbNg2EwSERQTidH8vyuWLVwgvetuEIgoJ3X6HgO
fcY9qughwGvv8hwnxxH4Beuyup3Fd7OgetqR1wq4Vqb6VzYbw3Nsc0ypKyqv1NdNiSp4No6Zsuc2TS1/
w3pyF+Jwh8F4F29kmDeQwpye41GDGk6TieVLoE5XxGXGT4TKkFACiWL3wxNNbaP+qrg2RzbZakHL17q3
t731AQf63qyt1ur5wgDcxH+DRnsSN8vTRIfqxHwuGIUMhtBYKn1HGstDoj0PoI57GcoUuHk= 4481011
56@qq.com

44810@MINGW64 ~/.ssh
$ ssh -T git@github.com
Hi wangiwei! You've successfully authenticated, but GitHub does not provide shell
access.

44810@MINGW64 ~/.ssh
$
```

3.3. 关联远程仓库与推拉

3.3.1. `git push origin 分支名` # 推动到远程仓库

3.3.2. `git pull origin master` # 从远程仓库只能拉取主分支

4. 使用之前

4.1. 注册GitHub帐号

4.1.1. 前往github.com注册帐号，记住用户名以及注册邮箱账号