这个教程较为简洁https://www.liaoxuefeng.com/wiki/896043488029600/896067074338496

开始：

从https://git-scm.com/downloads下载win客户端

安装，打开GIT-Bash：

**1、仓库配置**

$ git config --global user.name "Your Name"

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

配置用户名，邮箱。配置完之后可以通过git config --global -l看见配置的内容。

**2、使用github远程仓库**

1）创建SSH Key（如果有了请忽略，位置在用户主目录下.ssh下id\_rsa和id\_rsa.pub）：

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

2）登陆GitHub，打开“Account settings”，“SSH Keys”页面：

然后，点“Add SSH Key”，填上任意Title，在Key文本框里粘贴id\_rsa.pub文件的内容。

**3、对现有项目Fork**

如果我们想要修改他人github项目的话，我们直接git clone代码到本地是不能pull的，所以我们使用fork，先把代码复制到自己的github仓库，然后git clone到本地修改，然后在提交pull（这里的pull是pull到自己github仓库了，我们自己的github仓库中的代码是fork源的一个分支），这时候我们想要把修改的代码提交给他人的话，就可以在自己github上pull，等其他人看到后就可以把代码做一个合并

然后发现之前已经Fork过了，提示：

This branch is 2 commits behind ZeweiChu:master. ？！！

选择pull request：

左边选择自己的仓库名和分支，右边改成原来仓库的仓库名和分支

发现更新了2个readme.md --！

点击绿色的create pull request

提示输入标题和信息，继续点击。

此时，会提示 [jmy45](https://github.com/jmy45) wants to merge 2 commits into [jmy45:master](https://github.com/jmy45/PyTorch-Course) from [ZeweiChu:master](https://github.com/ZeweiChu/PyTorch-Course)

### This branch has no conflicts with the base branch

Merging can be performed automatically.

然后点击merge pull request->confirm merge

就成功了。

**4、远程库克隆**

git clone git@github.com:jmy45/PyTorch-Course.git 太慢了

git clone https://github.com.cnpmjs.org/jmy45/PyTorch-Course.git 这个速度还行

要查看远程库的信息，用git remote -v：

发现结果是 origin https://github.com.cnpmjs.org/jmy45/PyTorch-Course.git

然后已经修改了一些信息，但是没有git add：

于是用：

git checkout

git pull origin master更新至最新

修改一下git remote的地址：

git remote set-url origin git@github.com:jmy45/PyTorch-Course.git

**5、更新至远程库**

**把本地的修改推送至远程库**

git push -u origin master

提示成功。

**6、新建了这个Record记录仓库**

在github上新建了这个Record记录仓库，记录一些使用的问题和日常遇到的问题。

添加整个文件夹的文件：

git add Pytorch-Course/

关联一个远程库，使用命令git remote add origin git@server-name:path/repo-name.git

git push -u origin master第一次推送master分支的所有内容

git push origin master推送最新修改

新建版本库：

$ mkdir learngit

$ cd learngit

$ pwd

/Users/michael/learngit

$ git init

git init表示由Git管理

在新建版本库中添加文件，并输入以下：

$ git add readme.txt

$ git commit -m "wrote a readme file"

以下是用于修改后提交新文件:

$ git status

$ git diff readme.txt

$ git add readme.txt

$ git commit -m "add distributed"

git log --pretty=online查看提交历史

回到上一个版本：

$ git reset --hard HEAD^

$ git reset --hard 1094a（这是又回去了）

$ git reflog可以找到commit id

工作区：比方说就是learngit

版本库：分为暂存区stage和分支master以及指向master的HEAD

git add是把文件放入暂存区

git commit把暂存区内容提交到当前分支

用git diff HEAD -- readme.txt命令可以查看工作区和版本库里面最新版本的区别

如何撤销修改？

还未进行Git add：

$ git checkout -- readme.txt把readme.txt文件在工作区的修改全部撤销

1）readme.txt自修改后还没有被放到暂存区，撤销修改就回到和版本库一模一样的状态

2）readme.txt已经添加到暂存区后又作了修改，撤销修改就回到添加到暂存区后的状态

已经进行了Git add：

$ git reset HEAD readme.txt可以把暂存区的修改撤销掉（unstage），重新放回工作区

$ git checkout -- readme.txt

删除文件：

$ git rm test.txt