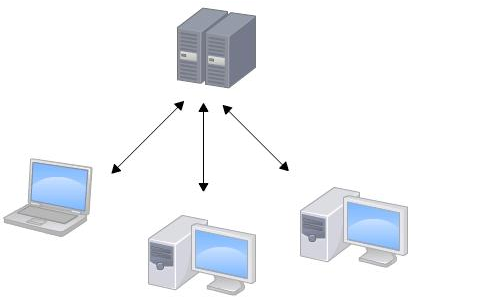
1. 版本控制软件

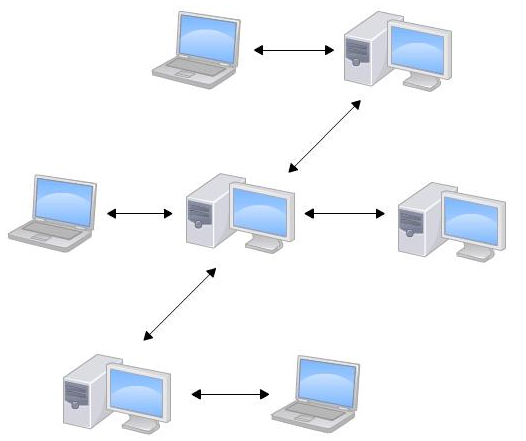
版本控制软件现在分成两种：

集中式：代表软件SVN



最大的优点是对不同的的用户使用不同的限制进行控制，不同的用户能够得到的代码量是不一样的。所以很多大的非开源商业项目都会使用这种方式进行代码的管理。

分布式：代表软件Git



最大的优点，是所有人都相当于是一台服务器，不会因为一台电脑（服务）不在组中就无法更新与管理代码。在开源项目中这样的设定是非常的方便的。

版本管理软件作用：

1. 自我代码版本的管理，可以对代码进行历史回滚等操作。
2. 团队协作，方便代码的整合。
3. 方便分享项目。等等

本次内容主要是以现在用得更多的Git来进行。

1. 安装Git

Git最早的时候只是在Linux等系统上使用，所以按道理说Git最好的使用环境是在Linux等命令行系统上。

但随着Git的发展Git已经开发了在Windows跟苹果电脑系统上的软件。在Windows上也有了图形化但并不建议大家使用图形化的，因为一些问题。。。（不要问我怎么知道的）图形化的使用是有些问题。

首先先在官网上下载安装包https://git-scm.com/downloads

但由于是外网的关系，下载速度是一个“快”，所以也可以从“国内镜像”下载。https://pan.baidu.com/s/1kU5OCOB#list/path=%2Fpub%2Fgit

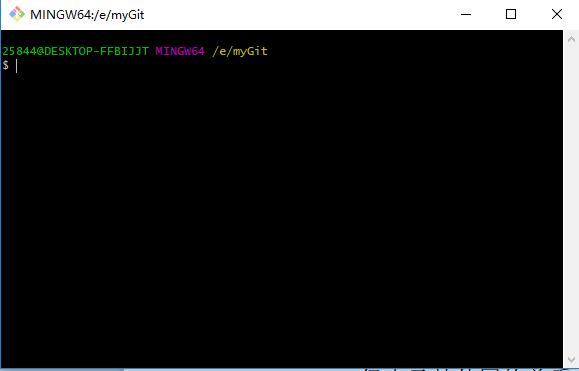
下载后，直接运行安装包，直接一路下点就可以了。

1. 使用Git
   1. 本地使用：
      1. 建立本地的仓库

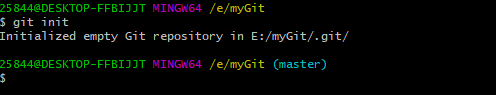
在需要建立仓库的文件夹（一般为项目文件夹）直接右键选择GIt Bash here。就会使用Git Bash 直接打开到该文件夹中。

由于Bash是Linux等系统的移植过来的命令行操作系统跟windows上的MSDOS的命令是有很多不同的，需要的同学请到相关的网站上查看Linux的命令。

在这里我们就应该看到这样的一个画面



接一来就直接在上面使用以下命令

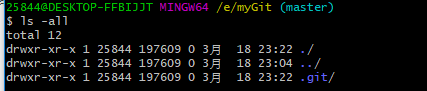


git init

该命令就是在本目录初始化git也就是建立git仓库。也就是在本目录中建立了一个.git的隐藏文件夹，不会看的同学也可以直接在Bash上输入

ls -all

来看当前目录中的所有文件。如图



现在还需要设置一下本仓库中的一个信息。

输入：

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

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

* + 1. 添加/更新仓库

现在的我们可以直接在本文件夹中建立一个测试用的文件，当然如果你放到项目中也就可以不用管这一步了，你直接添加项目中的所有文件就是了。

我们在这里建立可以新建 一个叫test.html的文件。也在里写点什么就是了。

然后就使用： git add 文件名

来把文件添加到更新的缓存区如图：

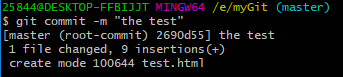


PS：Linux的操作提示遵从外国人对消息的原则，就是 No news is news.也就是一般来讲操作成功是不会有提示说明的（你妹啊。。坑爹啊。。。）

好了，上面这一步并还没有直接提交到仓库中，接下来还需要输入这个命令：

git commit -m”本次更新的注释”

如图：



看到这样的话就说明已经提交到仓库中去了

现在也可以使用查看状态命令来看看现在更新的结果：

git status

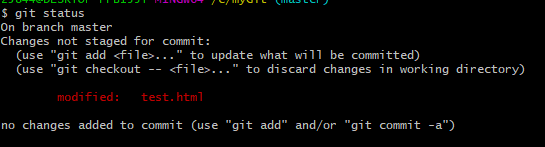
PS:这个命令会经常使用。

* + 1. 版本回滚

现在我们需要先了解git在单人使用上最为常用的一个功能就是版本回滚。也就是说，通Git提交的文件都有不同的版本，我们可通过Git来回滚到项目文件里的不同时期的状态。

首先，把刚刚的文件进行一些修改再进行提交。

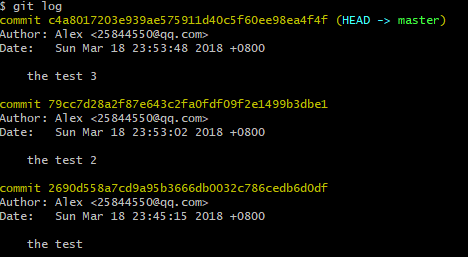
修后先别提交使用status命令可以看到修改的文件已经跟仓库里的文件不一样了。有了一些新的提示



这里我们可以多次修改与提交。

然后我们使用命令可以看到我们的一些操作日志与文件的操作日志

使用命令： git log

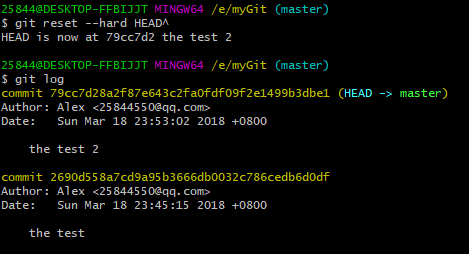


接一下来我们使用回滚命令：

git reset --hard HEAD^

这里HEAD^ 一个^是回滚上一个版本，两个就是回滚上上个版本。当然如果你要回滚到很前面的版本。。。比方说要回滚前69个版本可以数着打69个^。当然也可以使用HEAD~69来做到这一点。

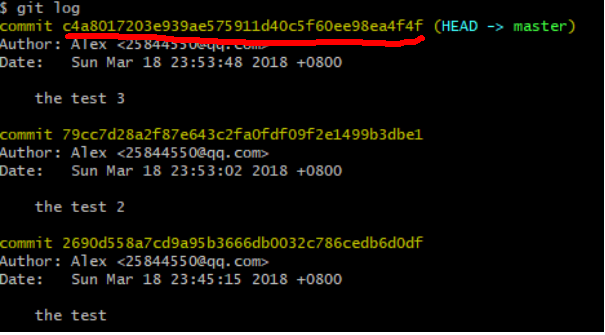
我们回滚上一个版本



可以看到我们回到了上一个版本，查看文件也变成了那一个版本的内容了。

当然如果想要回之前的那个版本的话。。。（妈蛋，你要你现在的版本还回滚个屁啊！！！！）我们也还是有办法的。

因为我们可以使用commit ID 来直接跳到对应的历史版本中去



红色的那部分就是commit ID 了。接下来使用这个命令：

git reset --hard commitID

比方说上面的例子就是： git reset --hard c4a801 (后面的不用写了系统会补的)

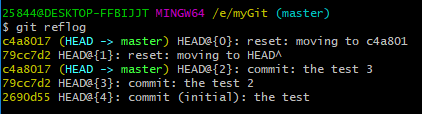


这样就跳回到就最开始的版本了。

当然如果早就忘记了commitID 也没有办法使用log看到话。可以使用命令：

git reflog

这个命令可以看到你所有操作的ID如图



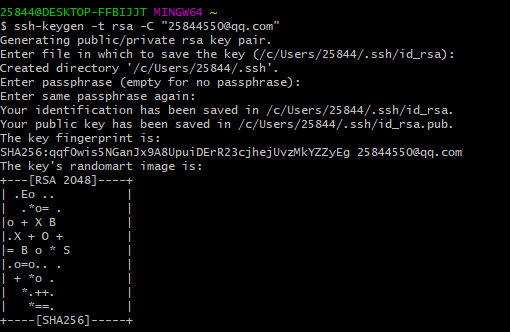
好了，到现在为止基本的本地单人版的版本控制已经完成。

* 1. 远程使用：
     1. 在GitHub上使用Git

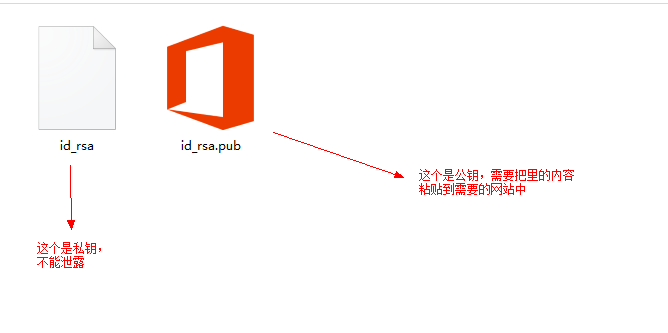
首先创建SSH Key。在用户主目录下，看看有没有.ssh目录，如果有，再看看这个目录下有没有id\_rsa和id\_rsa.pub这两个文件，如果已经有了，可直接跳到下一步。如果没有，打开Shell（Windows下打开Git Bash），创建SSH Key：

ssh-keygen -t rsa -C ["youremail@example.com"](mailto:\"youremail@example.com\")

如图



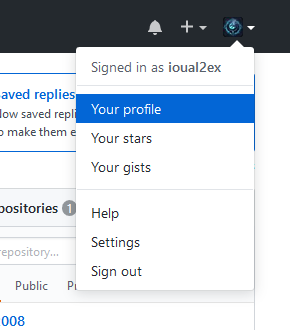
完成后在对应用户的目录中会多出.ssh文件夹，打开它里面有两个文件

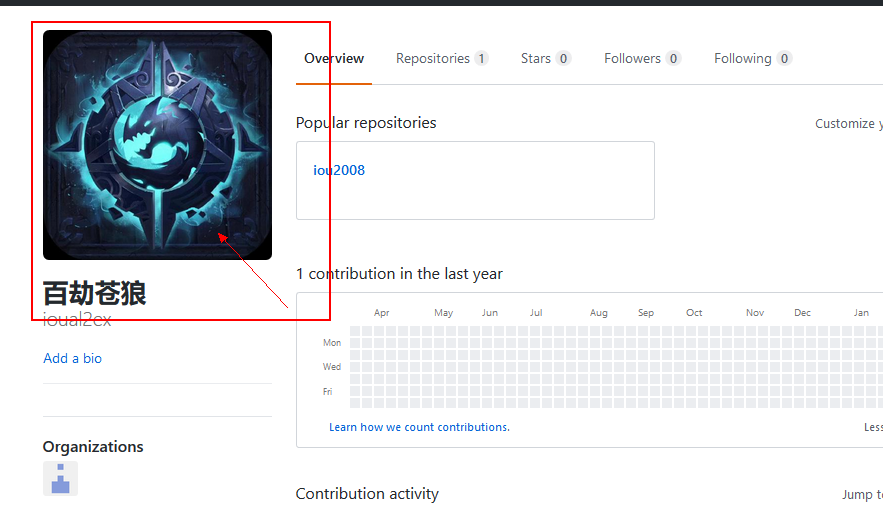


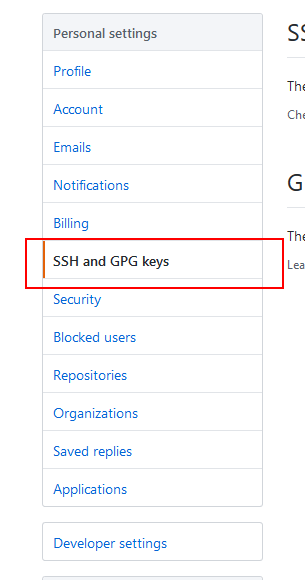
然后就是先去注册GitHub了地址：<https://github.com/>

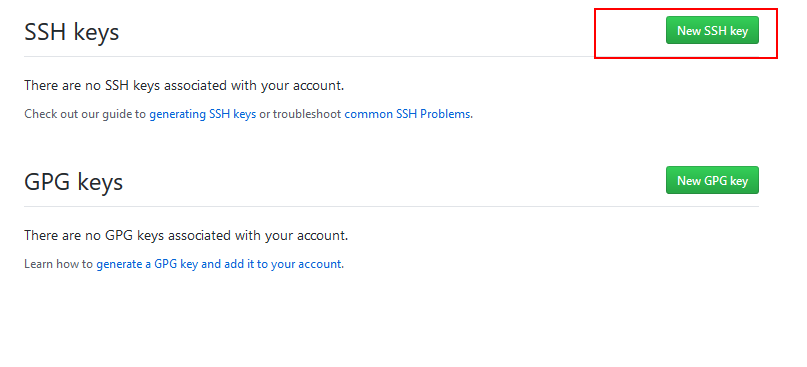
过程我就不说了，看到英文的同学，可以使用浏览器的翻译功能。

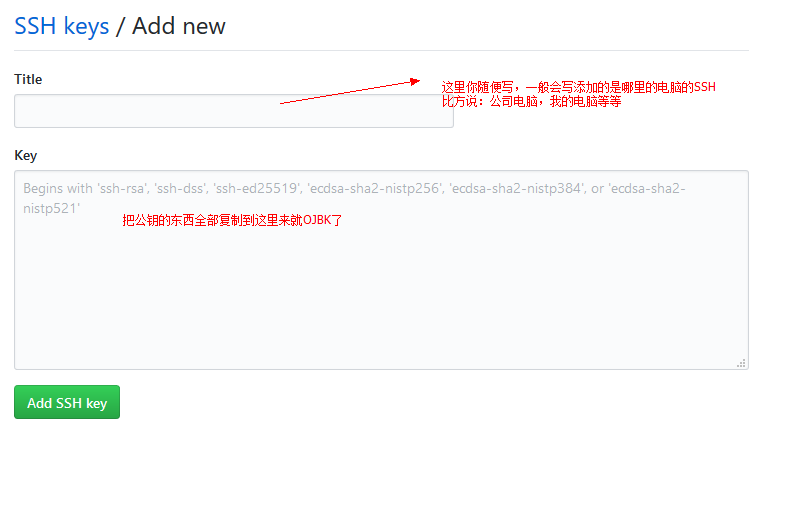
然后跟着下在图把SSH的公钥设置到GitHub中去。（由于网站会更新所以具体的操作有可能不一样。但基本上都是进去个人设置中去找对应的设置来设置的）

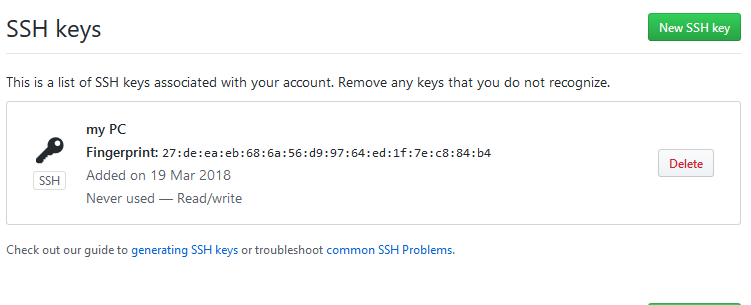






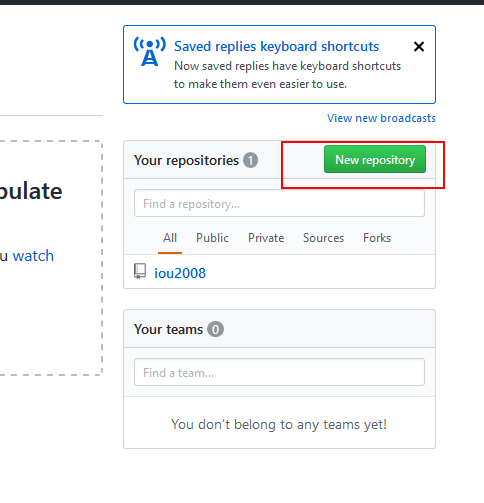


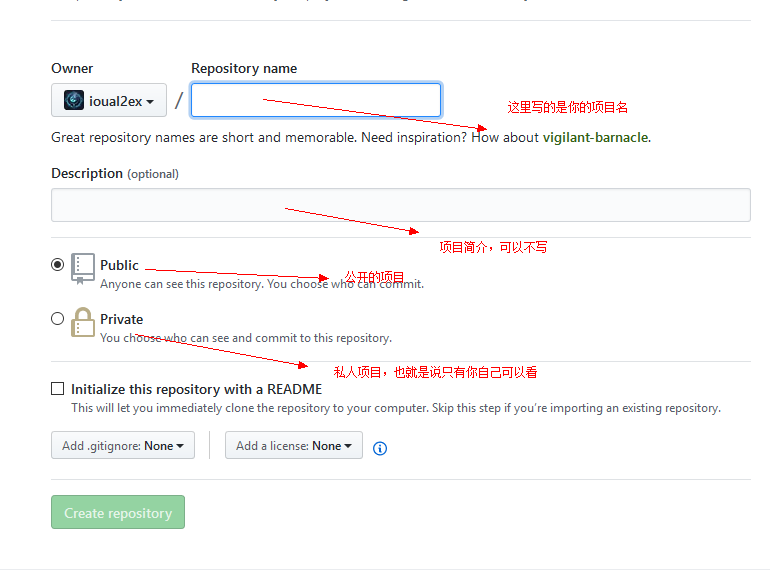


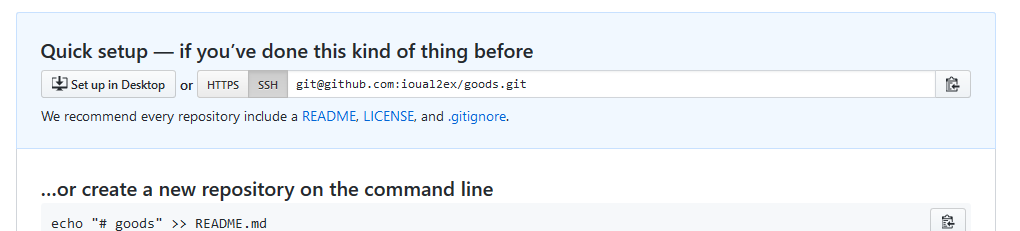
完成之后就会多一个SSH keys了当然也可以删除。

完成SSH key的添加我们就可以在Git上面添加我们的远程仓库了。

同样照着图来做就可以了。







GitHub两种连接方式，一种是HTTPS一种是SSH。点击对应的选择可以得到不同的地址（如果使用的是HTTPS是可以不用添加SSH Key的）。

接下来就是让本地仓库连接上远程仓库了。使用下面的命令：

git remote add origin [git@github.com:ioual2ex/goods.git](mailto:git@github.com:ioual2ex/goods.git)

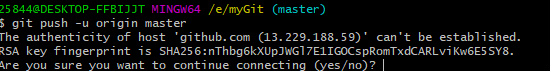
其中origin是远程仓库的名字，可以自己修改。但我们一般都使用这个名字就是了。



接下来就是把本地仓库的内容推送到远程仓库。使用下面的命令：

git push -u origin master

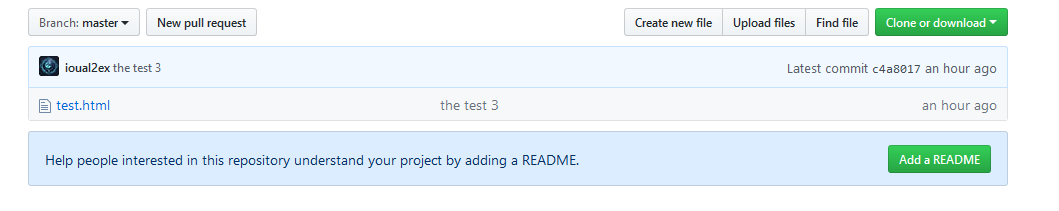
第一次使用这个命令的话会出现下面的对话



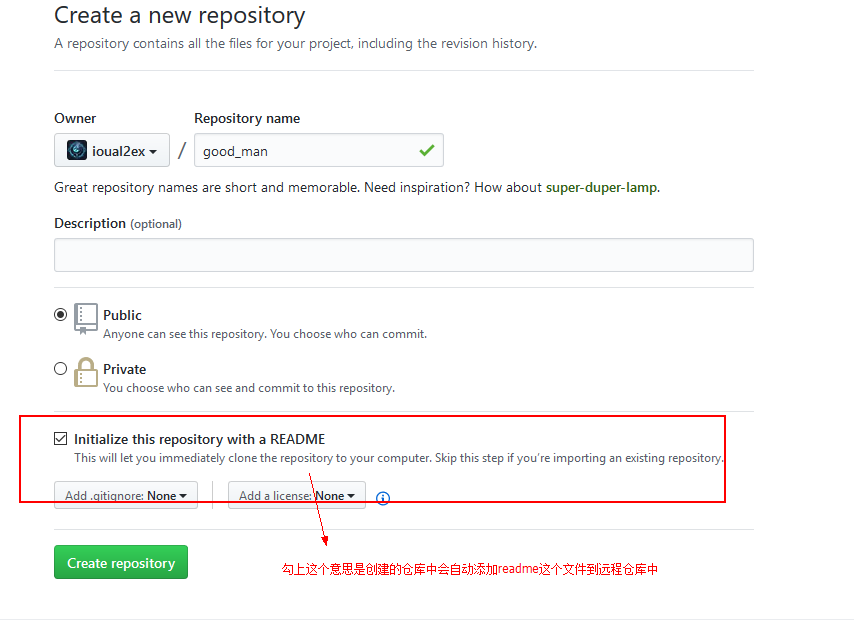
直接输入yes就可以了

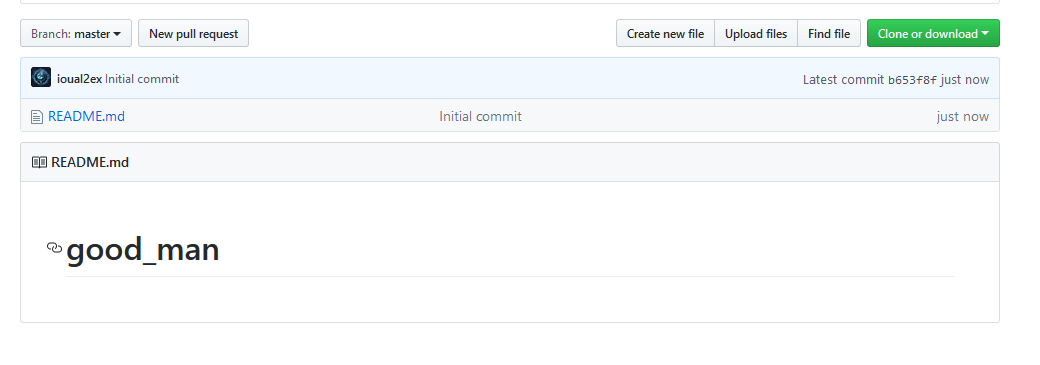
第一次需要写入-u参数推送master分支的所有内容，以后就可以直接使用简化命令 git push origin master只推荐修改或者添加的内容了。

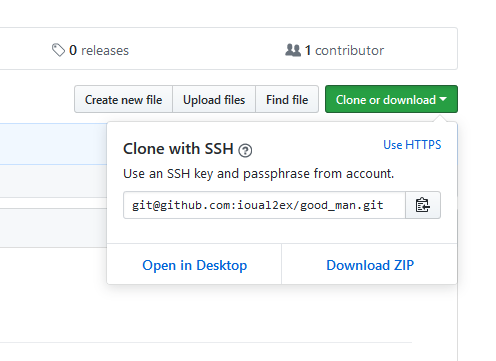
完成后可以看到GitHub上对应的项目已经同步成本地仓库了



如果需要从远程仓库克隆到本地仓库的话：如图



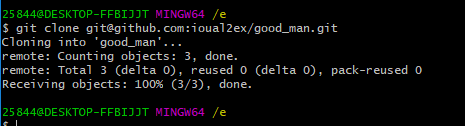




接着在需要的目录上使用Bash输入以下命令：

git clone Git项目的地址

完成后如下图



之后就可以看到目录里已经创建了刚刚克隆过来看目录跟文件了。

* + 1. 在码云上使用Git

基本上与GitHub一样操作，我这里就不详细来说明了。主要还是因为码云是国内的，都是中文了。可以直接认为就是中国的GitHub就是了。当然优点也非常的明显就是速度要比GitHub快上不少。不过当然有不好的地方就是上传的东西一般都会需要审查的，你们不要上传上不好的东西上去就没有问题。

* + 1. 多人协作

（由于电脑只有一台，这边只能做大概流程的介绍，中间有问题可以自行百度了）

首先协作的人同时需要先克隆同一个主仓库。

然后我们有两个小伙伴A跟B

比方说A跟B同进对这个项目进行修改。

A先进行push。

那项目就会更新。这个时候，由于仓库中的项目并没有修改过所以直接push就可以了。

B接着进行push。这个时候就会出现提示，说仓库已经更新过来需要本地仓库也要更新。这个时候B需要先使用pull把项目仓库中的更新内容先更新到本地来。

然后这个时候B再进行push。这个时候有两种情况。1、直接更新成功（完结散花）；2、项目文件发生了冲突，提交不了。这个时候可以使用bash来解决冲突，但是我个人是不推荐的。因为很多人不太会用vim(我也太久没有用了，有些命令已经忘记)。这个时候使用status命令可以让我们知道哪个文件出现了冲突。然后在自己的电脑上打这个文件，文件中git已经标记好了冲突的地方，手动把冲突的地方地进行修改再提交也就是 add 跟 commit

再进行push，这个时候就更新到仓库中去了。