How to use Git

# 写在最前面

这是一个关于如何使用Git的文档，包括使用Git所需要掌握的最基本的技能，你几乎可以参考这个文档傻瓜式地来使用Git。

各位工程师、设计师、PM、XX们~ 赶快投入Git的怀抱吧，你们会收获更多。

什么？你问我什么是Git？回去自己百度一下吧。

By 李灿 @ 2013.9.4

# 为什么要使用Git，而不是SVN

关于为什么要使用Git的文章，在Google搜索“Why Git”有41,500,000个结果，但是以下几点是我有切身体会的。

**没用中心版本库，避免了服务器硬盘坏的尴尬**

我们COC资源的SVN版本库，就因为某个版本对应的文件损坏了，结果这个版本所涉及到的所有相关文件和目录都取不出来了

**每个人都有一个版本库**

不仅速度会快很多，而且还可以离线查看历史版本

**一切皆分支**

分支可以很方便地同时开发多个功能

不像SVN，在Git里面创建分支没有任何拷贝目录开销

集中式 VS 分布式

# Git的一些概念

**Git和GitHub的关系：**

请自行类比SVN跟Google Code的关系。

Git ： 版本控制工具

GitHub ： 项目托管平台

**工作副本：**

等同于SVN的工作副本，也就是大家在自己电脑上所看到的并进行修改的各种文件和文件夹。

**Git分为本地版本库和远程版本库。**

**本地版本库：**

所有数据存放在工作副本的.git目录里面，跟SVN的.svn目录类似，但是.git里面存放了整个版本库，而不像SVN一样只有最新版本。

**远程版本库：**

除了本地版本库以外，同一个项目的其他所有版本库都叫远程版本库（包括服务器上的版本库和其他开发人员的版本库），平时我们需要打交道的远程版本库一般为服务器上的版本库。

# 客户端 TortoiseGit

## 下载

<https://code.google.com/p/tortoisegit/>

<ftp://192.168.1.99/software/Git/> (FTP)

## 安装



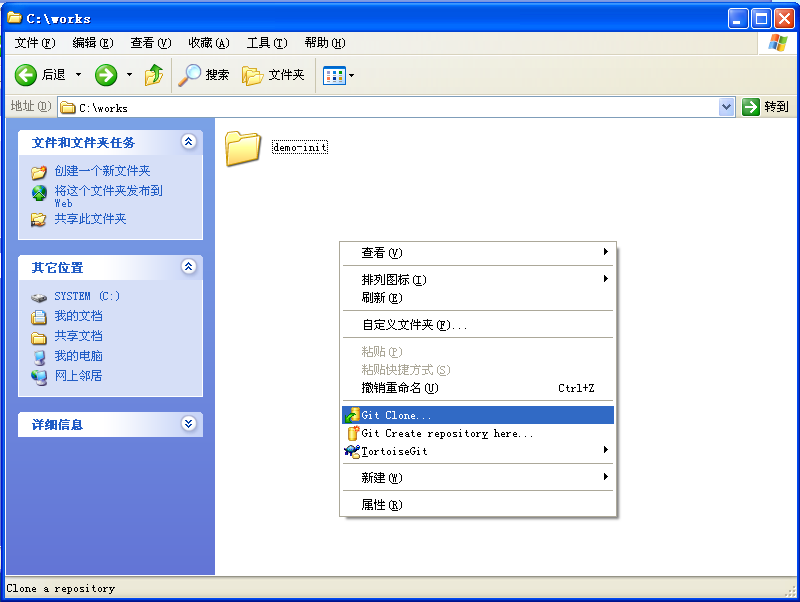
一路Next即可。

## 克隆版本库（Clone）

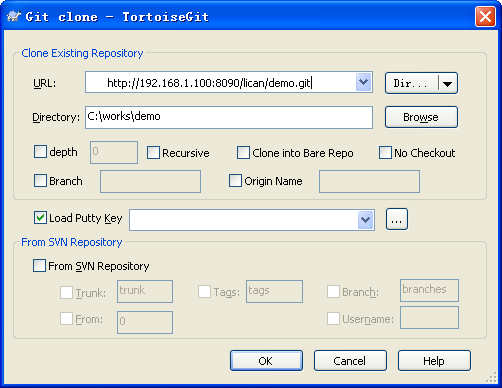
从**远程版本库**克隆出一个新的**本地版本库**，并检出**工作副本**。

相当于SVN的Checkout。

在空白处右击，选择Git Clone...



弹出Clone对话框：



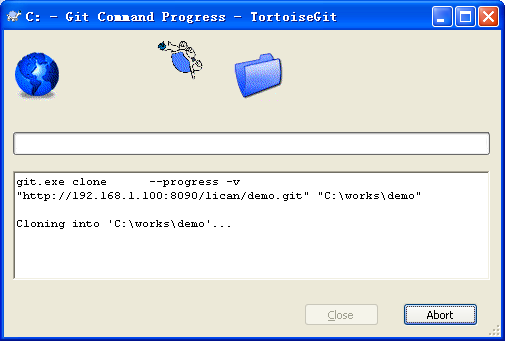
在URL处输入服务器版本库的地址，图中示例为http://192.168.1.100:8090/lican/demo.git

如果服务器上没有项目的**远程版本库**，需要先在服务器上创建一个项目，请略过本节内容直接参考“5服务器端 GitLab”。

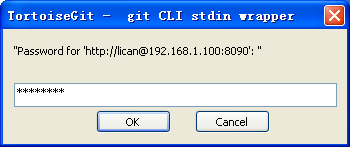
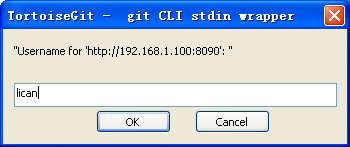
下方的Directory处会自动加上版本库的名字（这里从“C:\works”自动变成了“C:\works\demo”）

然后点击OK即可。

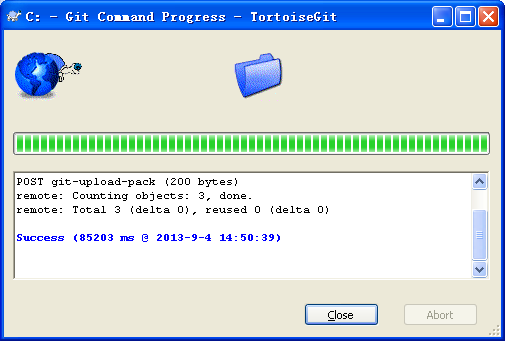
弹出Clone进度对话框



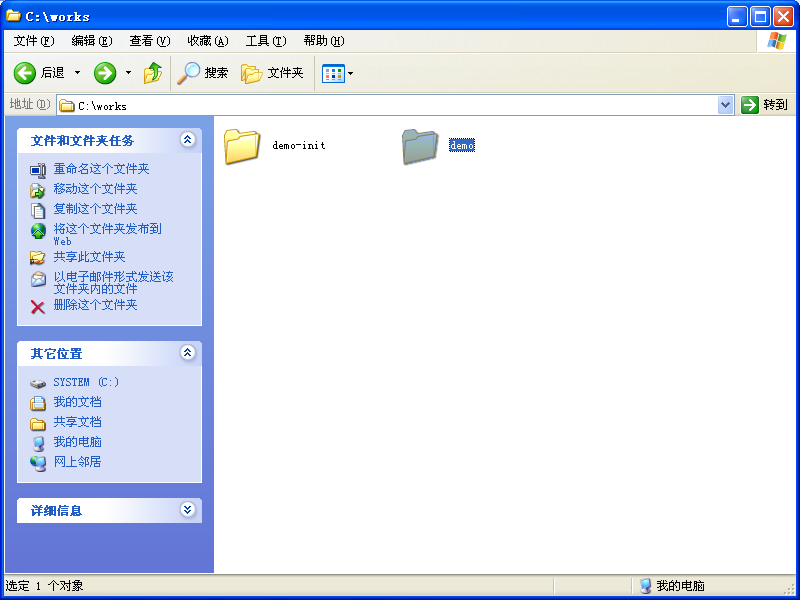
期间会弹出对话框提示输入用户名和密码



最后显示Clone成功。



然后刚刚操作的文件夹中出现了我们刚刚Clone出来的版本库/工作副本。



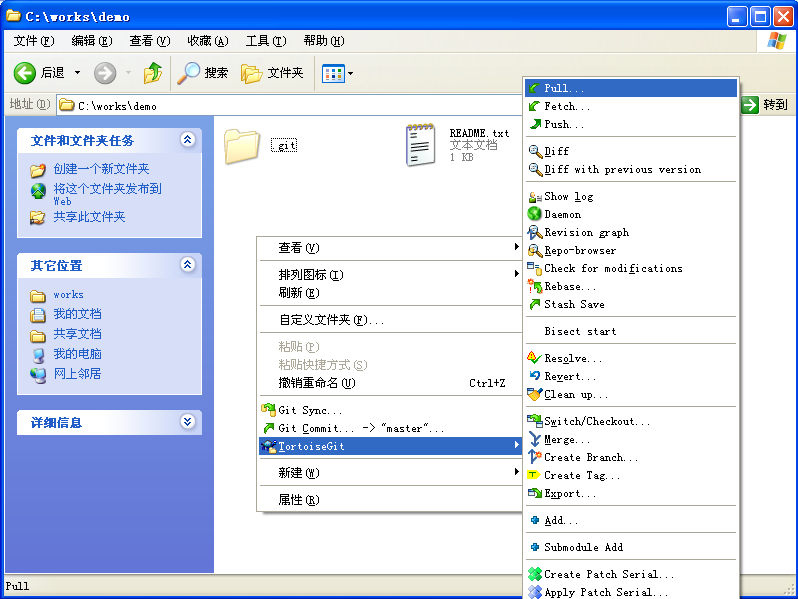
## 更新 （Pull）

从**远程版本库**获取内容到**本地版本库**和**工作副本**。

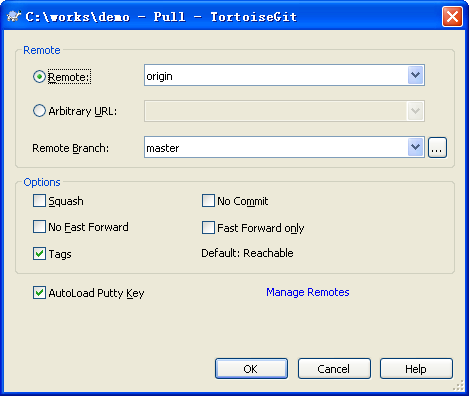
相当于SVN的Update。

进行此操作前，如果有修改过的文件，需要先进行Commit操作（见“提交到本地版本库（Commit）”）。

打开项目文件夹，在空白处右击，依次选择TortoiseGit 🡪 Pull...

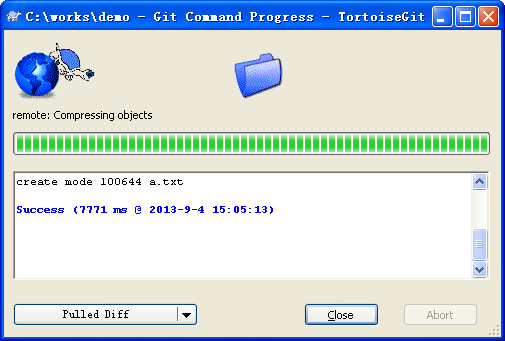


弹出Pull对话框：

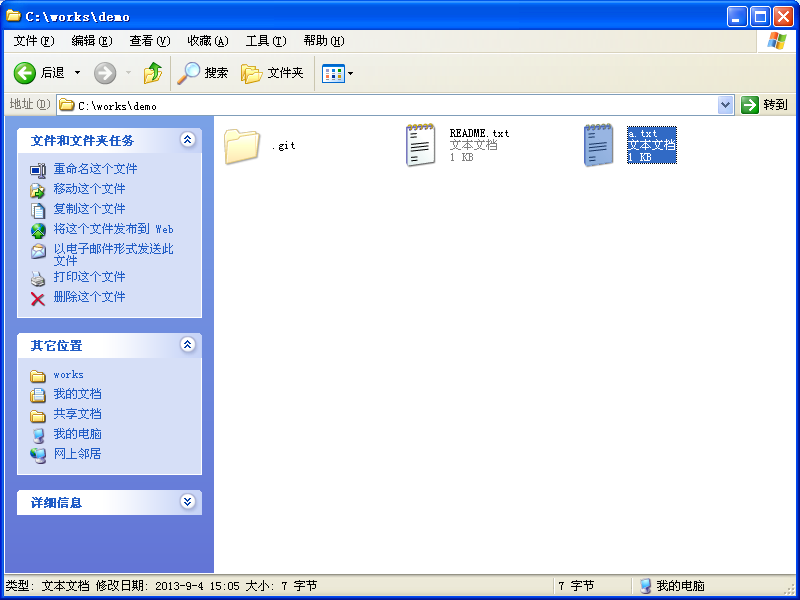


点击OK

然后会弹出Pull进度对话框，依次按提示输入用户名和密码即可，成功后显示如下。



然后文件夹中应该已经出现了其他人提交的文件（如果有的话）。



## 提交

提交分为两步：

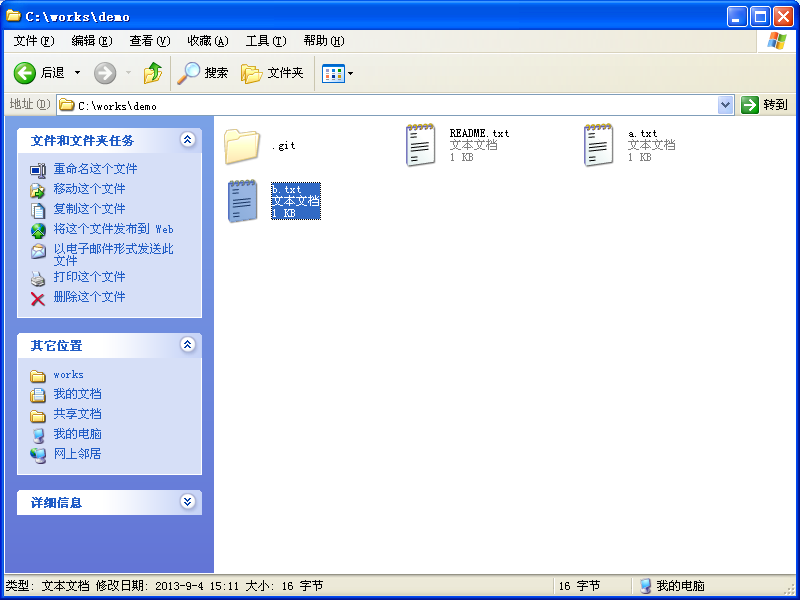
1. 提交到本地版本库（Commit）
2. 上传到服务器（Push）

两步加起来相当于SVN的Commit。

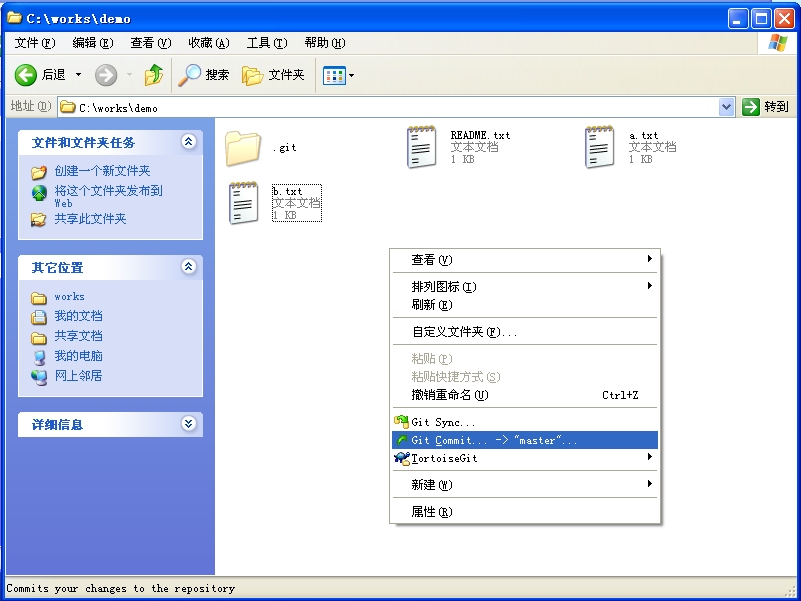
## 提交到本地版本库（Commit）

将**工作副本**的内容提交到**本地版本库**

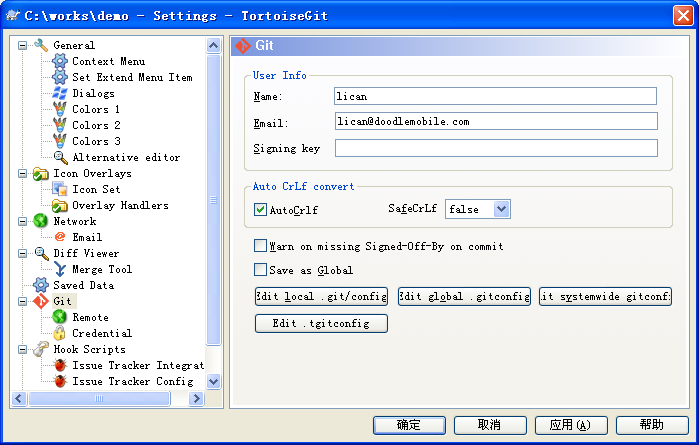
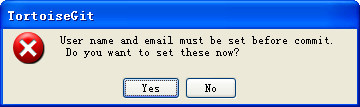
假设我们新建了一个文件“b.txt”



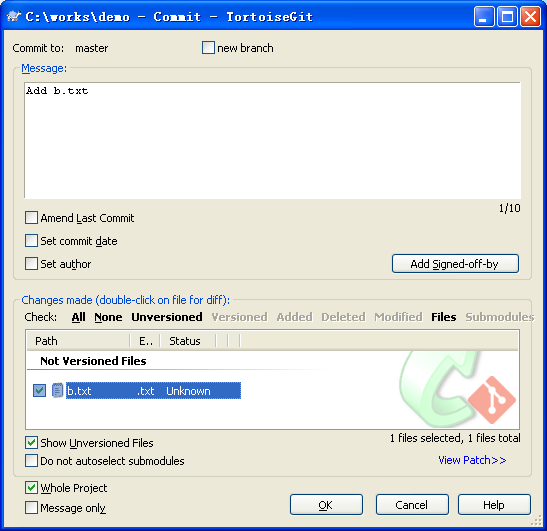
打开项目文件夹，在空白处右击，选择 Git Commit... -> “master”...



如果是第一次提交，会提示你输入名字的email，用来区分不同的提交者。

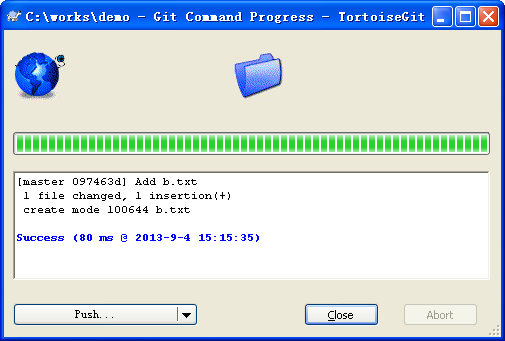


弹出Commit对话框：



输入提交信息（Message），然后在下方选择需要提交的文件，也可以点击“All”来选择所有文件。然后点击OK

提示Commit成功：

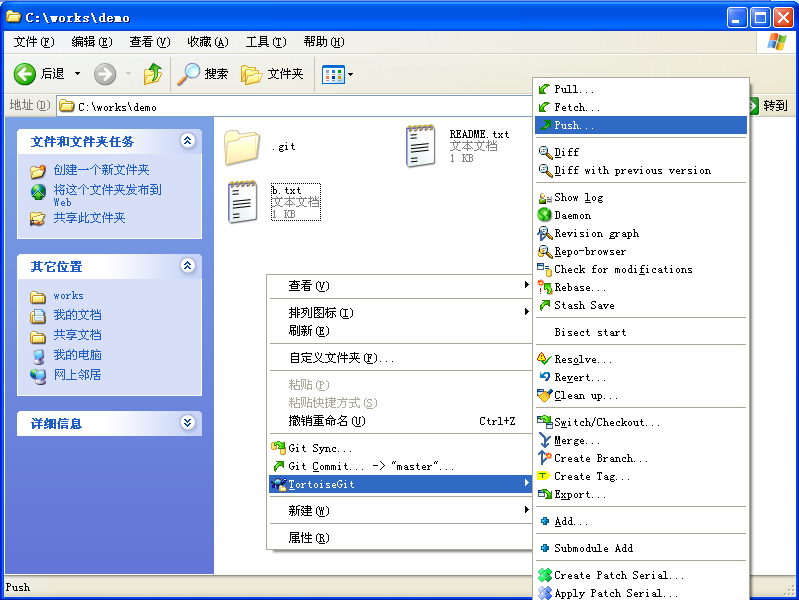


## 上传到服务器（Push）

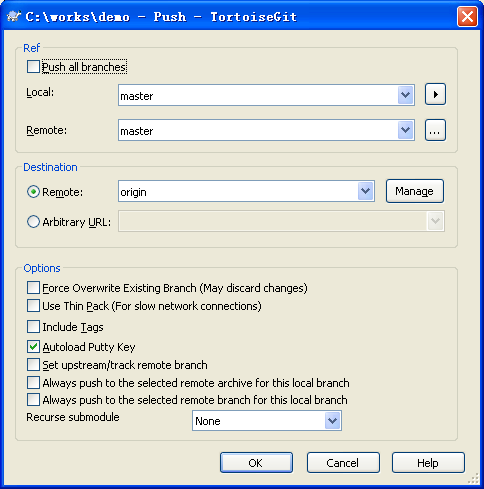
刚刚的Commit的操作还只是提交到**本地版本库**了，还没有上传到**远程版本库**（服务器），文件只是存在你自己的机器上，其他人还看不到你更改的文件。

一般来说，并不是每次Commit都要立即上传（Push）到服务器上，一般是隔一段时间Push一次。

如果要上传到服务器，在空白处右击，依次选择TortoiseGit 🡪 Push...

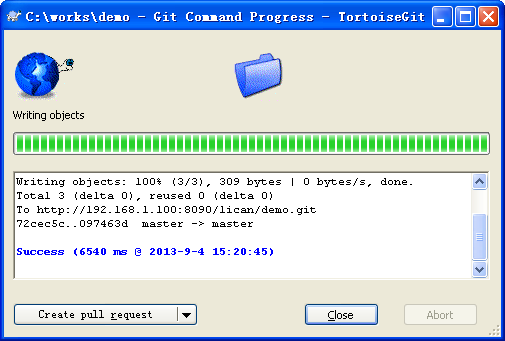


弹出Push对话框：



点击OK。

然后会弹出Push进度对话框，依次按提示输入用户名和密码即可，成功后显示如下。



这样，其他人（在Pull时）就可以看到你更改的文件了。

# 其他客户端

## msysgit

Git官方的Windows客户端，与TortoiseGit选用其中的一个即可。

<https://code.google.com/p/msysgit/> (貌似需要翻墙)

<ftp://192.168.1.99/software/Git/> (FTP)



## Eclipse插件：EGit（强烈推荐）

安装地址

eclipse 4.2/3.8: <http://download.eclipse.org/egit/updates>

eclipse 3.7: <http://download.eclipse.org/egit/updates-2.3>

eclipse 3.5/3.6: <http://download.eclipse.org/egit/updates-2.1>

关于支持的版本详细信息，参见

[http://wiki.eclipse.org/EGit/FAQ#What\_versions\_of\_Eclipse\_does\_EGit\_target.3F](http://wiki.eclipse.org/EGit/FAQ" \l "What_versions_of_Eclipse_does_EGit_target.3F)

# 服务器端 GitLab

公司所用的Git服务器是用的GitLab。

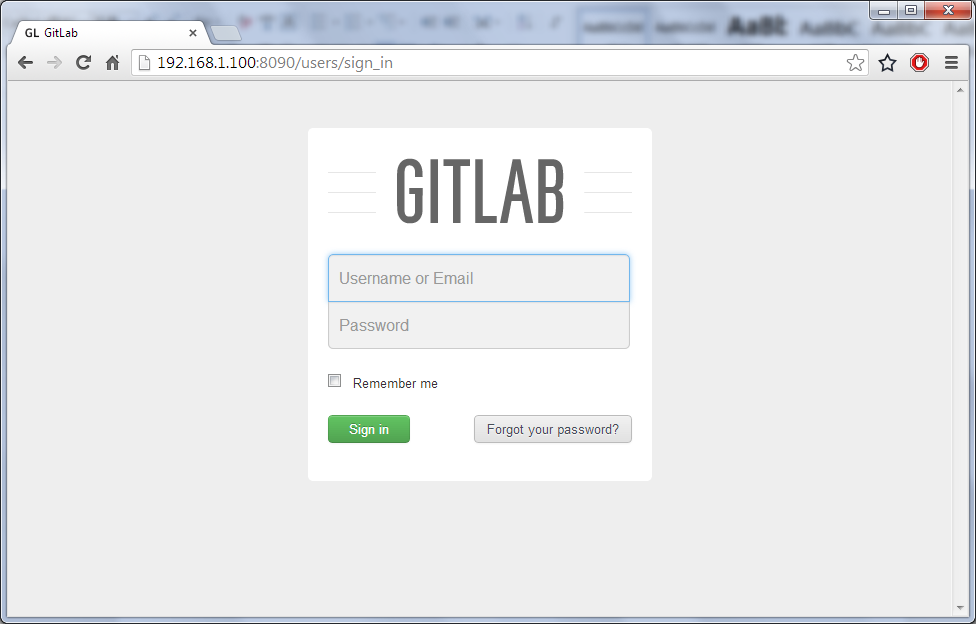
## 什么是GitLab

简而言之，GitLab就是GitHub的开源版本。

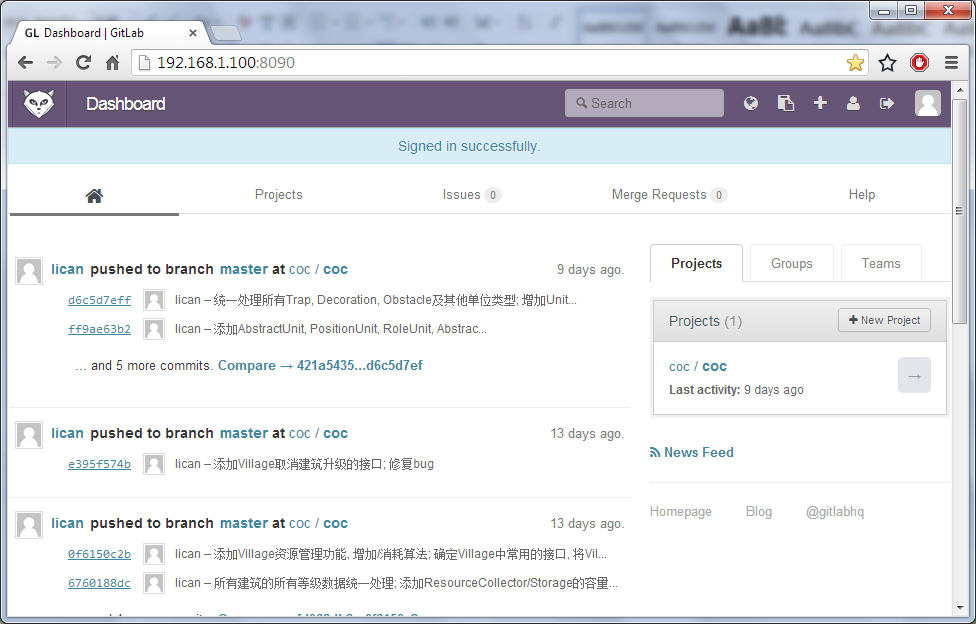
## 地址

[http://192.168.1.100:8090](http://192.168.1.100:8090/)

打开后登录界面如下：



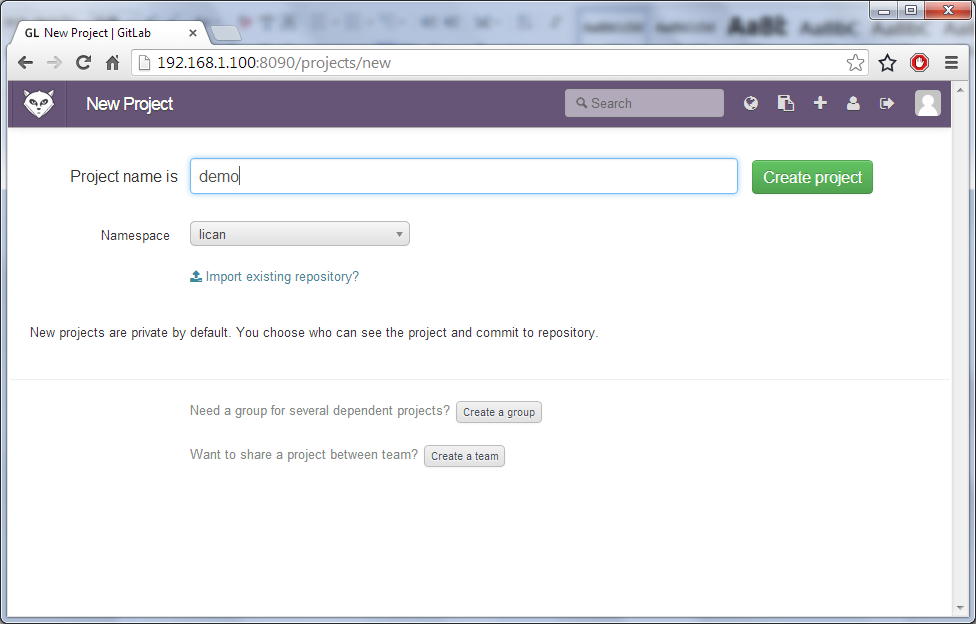
输入用户名密码登录后主界面如下：



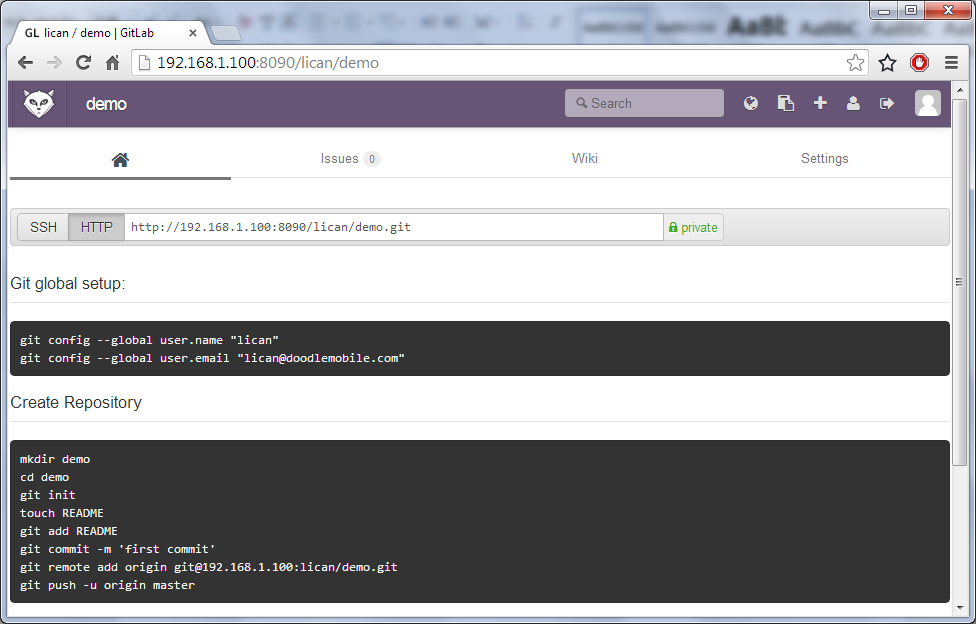
## 步骤1：在服务器上创建项目

点击右边按钮New Project

输入项目名字，然后点击Create project

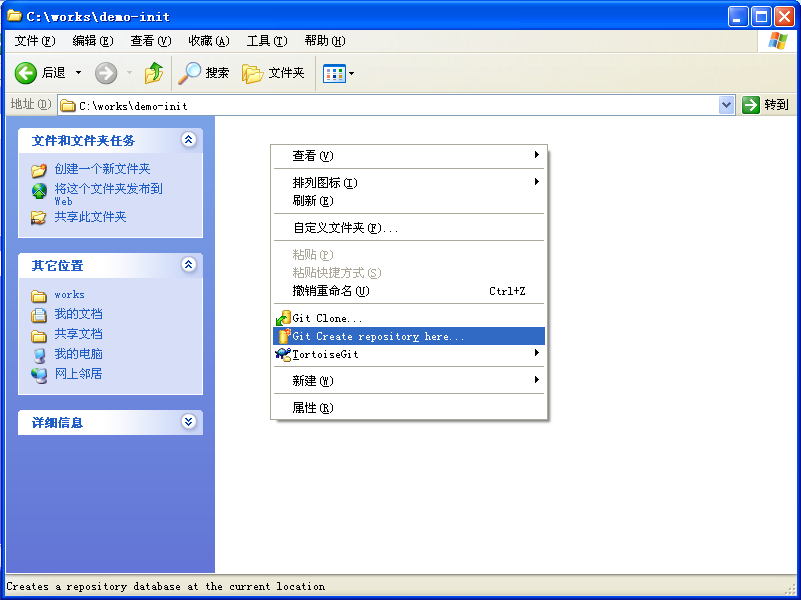


创建完成后如下：得到版本库的地址为http://192.168.1.100:8090/lican/demo.git



## 步骤2：初始化本地版本库

打开资源管理器，在**需要创建本地版本库的文件夹内**右击（假设已经正确安装了TortoiseGit），选择Git Create repository here…

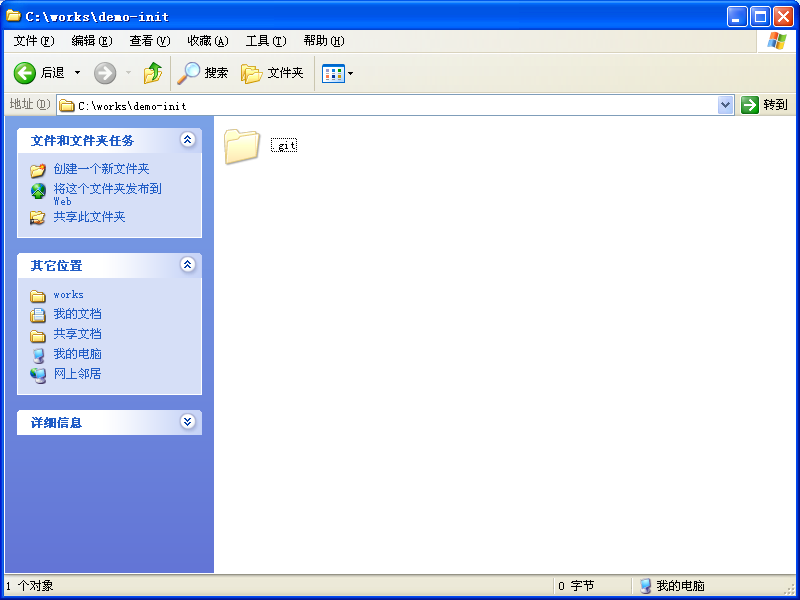


如果没有右击时没有选中任何文件夹，就是当前所打开的文件夹，否则就是所选中的文件夹，这里为C:\works\demo-init。这是我想要建立**本地版本库**和**工作副本**的地方。

在弹出的窗口直接点击OK

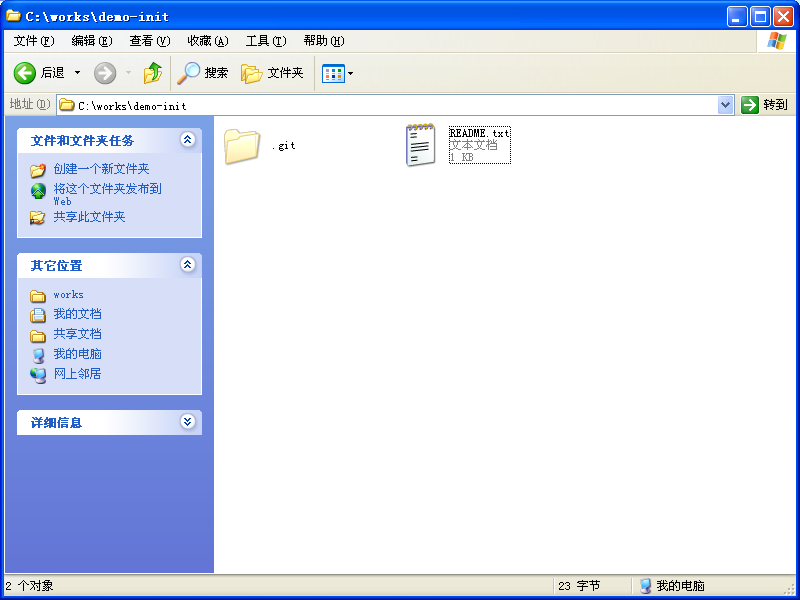


初始化完成后，文件夹内会出现一个.git的隐藏文件夹，这就是**本地版本库**存储的地方。

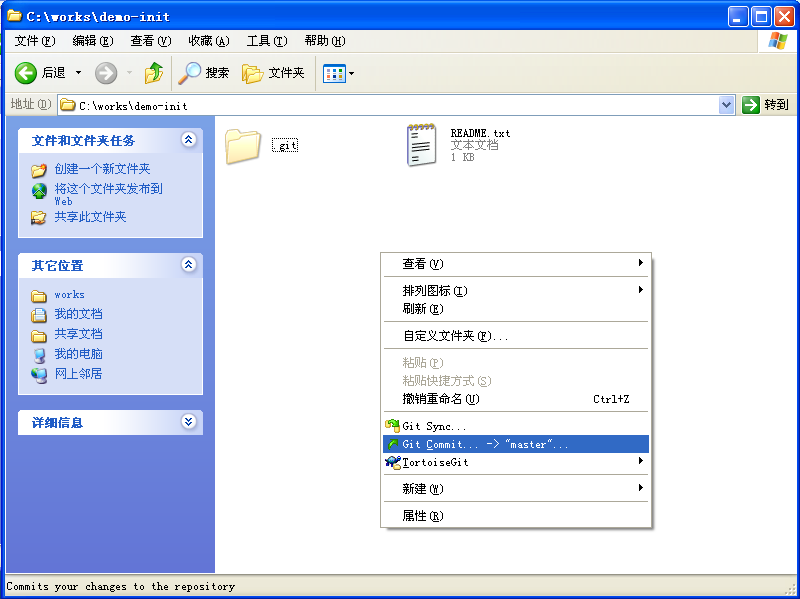


## 步骤2.5：创建一个README文件

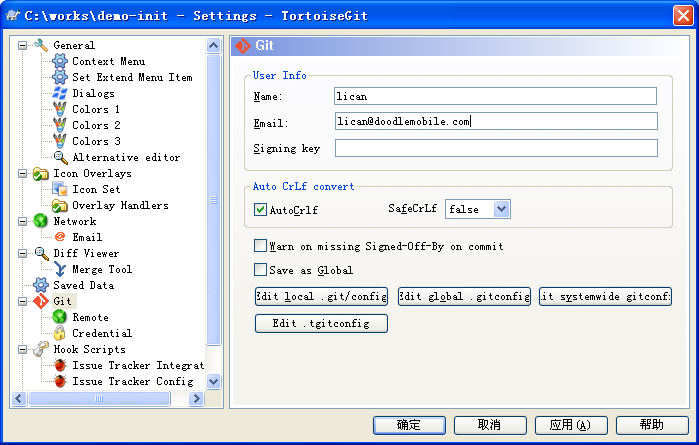
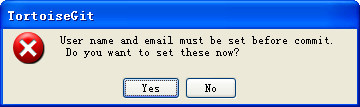
这一节不是必须的，这样做只是为了能够直观地看到效果，并对版本库的内容提供一个简要或者详细的说明



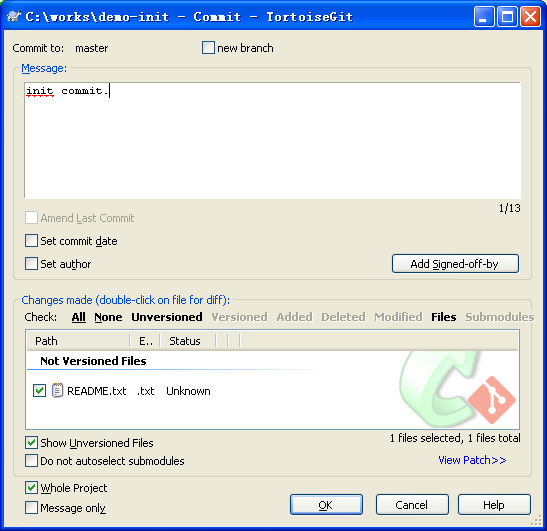
空白处右击，选择 Git Commit... -> “master”...



如果是第一次提交，会提示你输入名字的email，这个是用来区分是谁提交的。

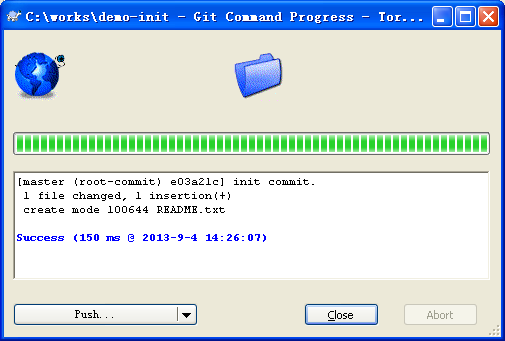


然后弹出Commit界面：



输入提交信息，并点击下方的All勾选上所有文件，最后点击OK。

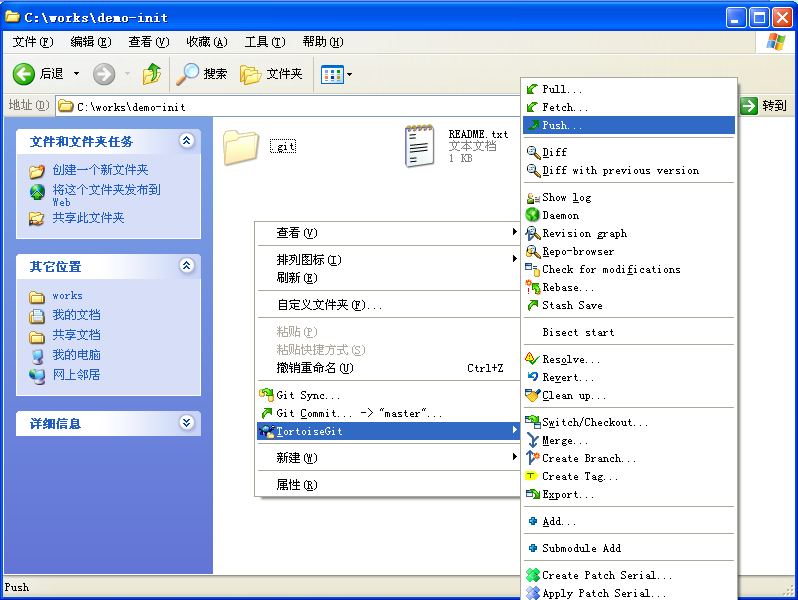
显示提交成功：



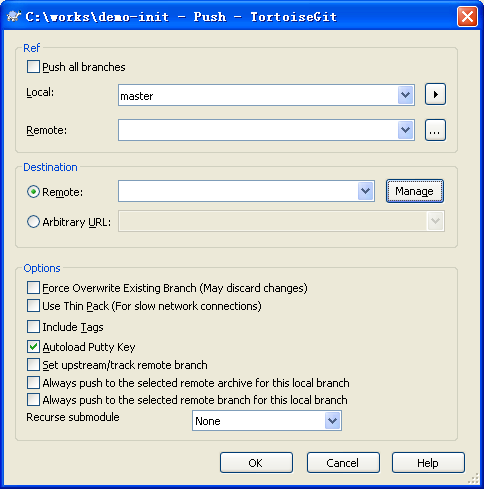
关闭。

## 步骤3：将本地版本库上传到服务器

在空白处右击，依次选择TortoiseGit 🡪 Push...

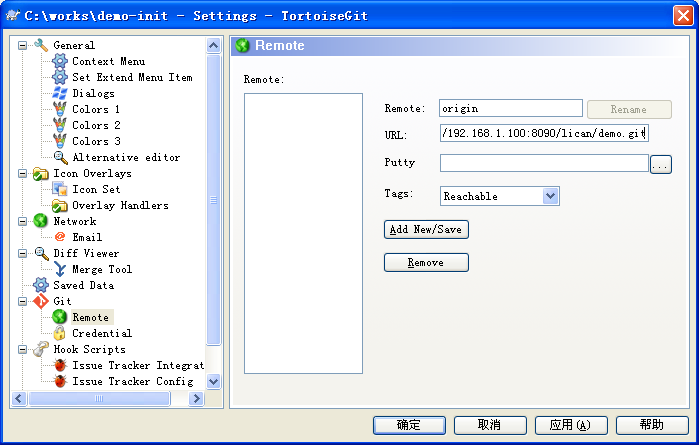


弹出Push对话框：



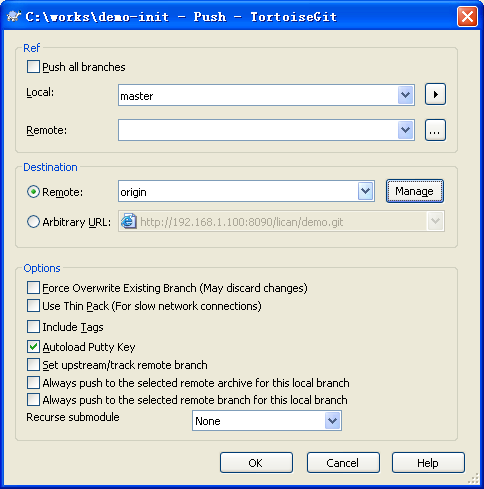
本地分支处选择master

目标处选择Remote，然后点击Manage



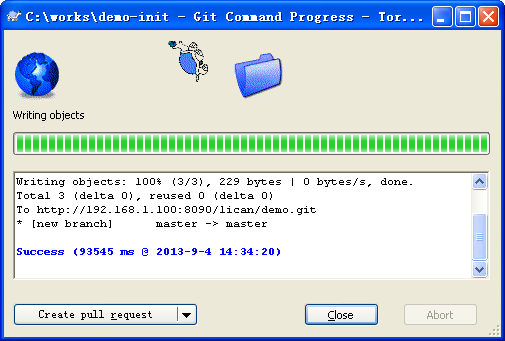
然后在弹出的对话框中URL处输入服务器版本库的地址，并点击确定

返回到Push对话框

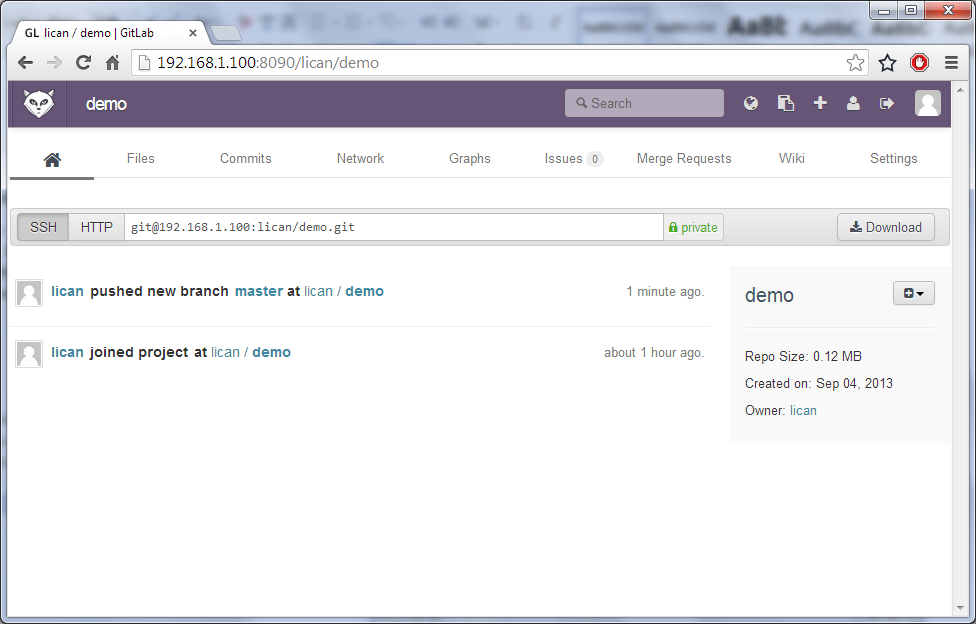


点击OK

然后会弹出Push进度对话框，依次按提示输入用户名和密码（即刚刚登录GitLab的用户名和密码）即可，成功后显示如下。



最后再回到浏览器，刷新：



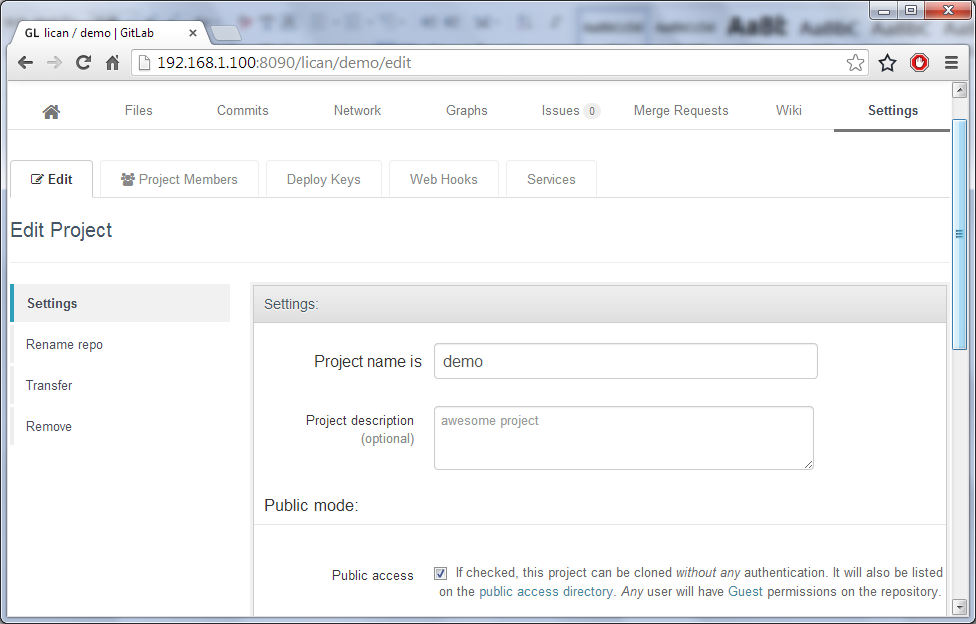
图中显示“... pushed new branch master ...”，表示服务器版本库上已经有了一个master分支。

到此，服务器上的这个版本库就可以正常用于开发使用了。

## 步骤3.5：将版本库设为公开访问

建议这么做，这样其他人可以在GitLab的Public projects中看到你的项目~~~

点击项目的Settings标签，然后勾选上Public access那个复选框，最后点击底下的Save即可。

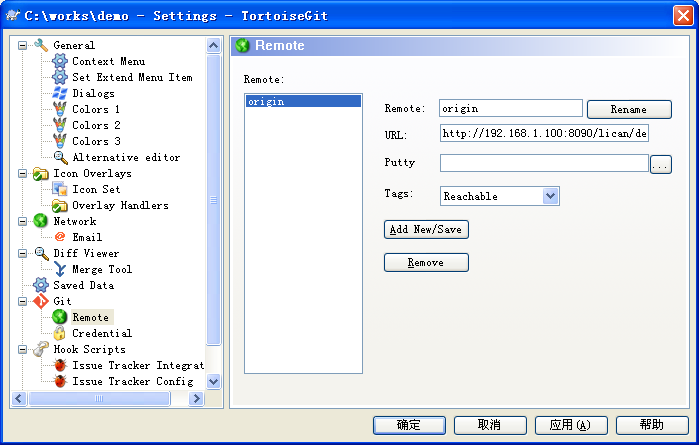


# 补充

## 不想每次输入用户名密码？

有一个简便的方法，在TortoiseGit的设置里面，把URL的http://后面加上“<username>:<password>@”。

如图



把URL从

http://192.168.1.100:8090/lican/demo.git

改成

http://lican:123@192.168.1.100:8090/lican/demo.git

# 工程师们的分享会

**@ 2013.9.5 4:00 PM**

**手把手教你用GitLab，TortoiseGit和EGit**

你懂的

**使用SSH**

每次输入用户名密码觉得麻烦？但又不想把密码写在URL里面？那就用SSH吧~

**解决冲突**

两个人修改同一个文件结果悲剧了吧？教你如何在Git中解决文件冲突！

**使用分支**

多功能同时开发什么的，分支才是王道啊

**以及更多……**