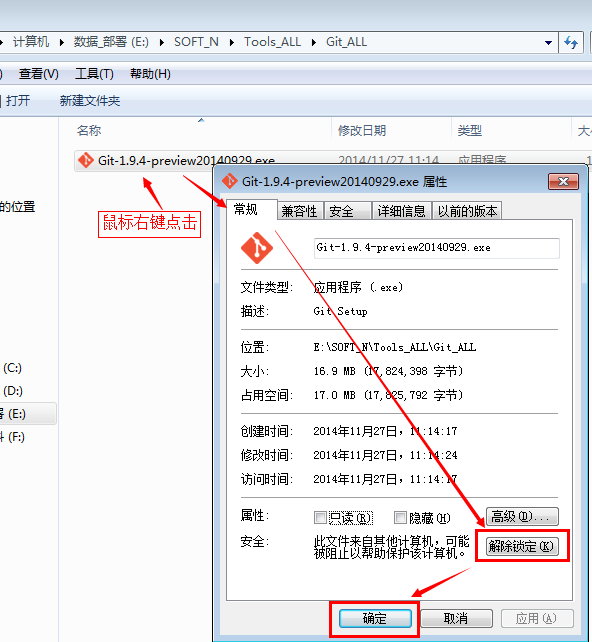
# [1. Git安装与配置](http://blog.csdn.net/renfufei/article/details/41647875) [下载Git安装包](http://blog.csdn.net/renfufei/article/details/41647875#t0)相关网站[相关文章](http://blog.csdn.net/renfufei/article/details/41647875#t2)

1·下载Git安装包

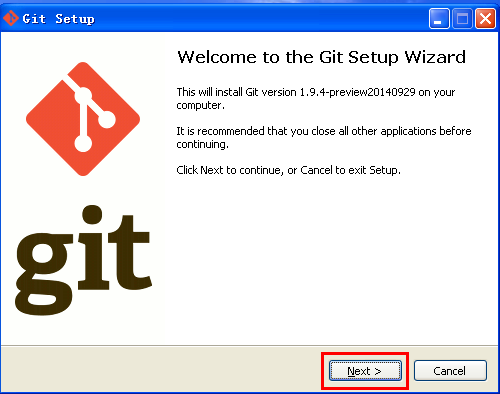
2·去除文件锁定(非必须)

右键点击下载的文件,选择属性,然后点击"解除锁定"按钮,确定即可. 如下图所示:

[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/01_UnlockFile.png)

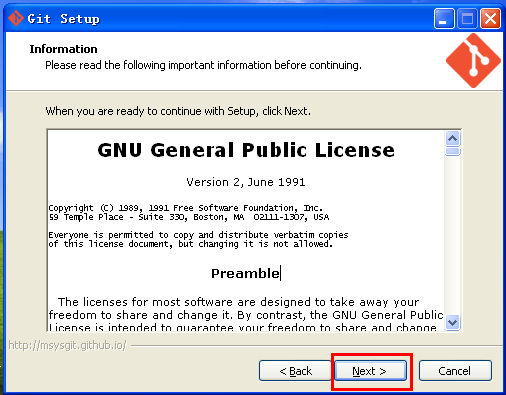
3·然后鼠标双击安装文件, 如果有Windows拦截警告，允许即可

4·然后出现安装向导界面,点击下一步(Next)即可:

[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/02_WizardNext.png)

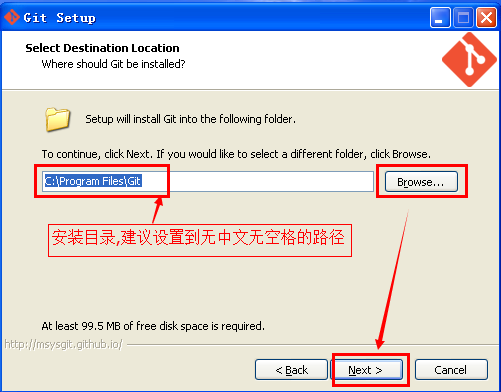
安装向导界面

5·接着出现授权信息界面， Next即可:

[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/03_LicenceNext.png)

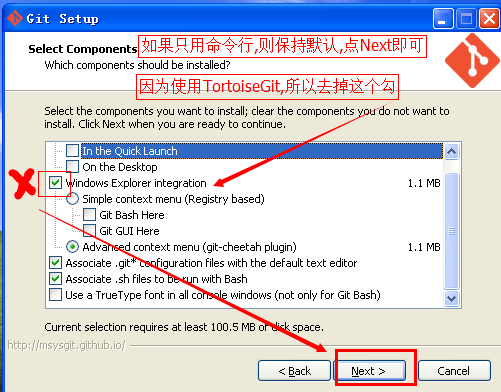
授权信息界面

6·选择安装路径

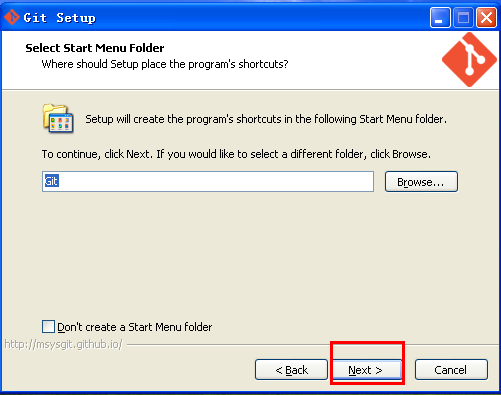
[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/04_InstallPath.png)

7·选择文件关联,如果你不清楚,直接默认,下一步即可.

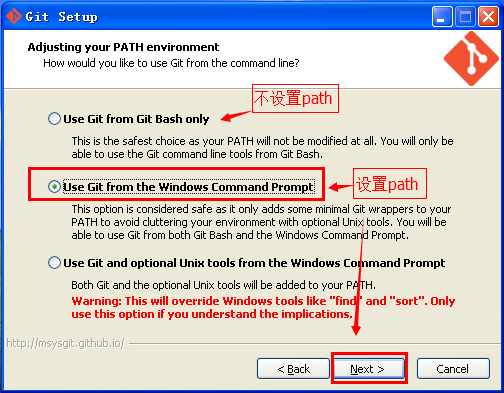
如果你想要右键菜单清爽一点,就把下图中的那个勾给去掉. 这里把所有勾去掉也是可以的,更清爽(我就是这么干的).

[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/05_Associate.png)

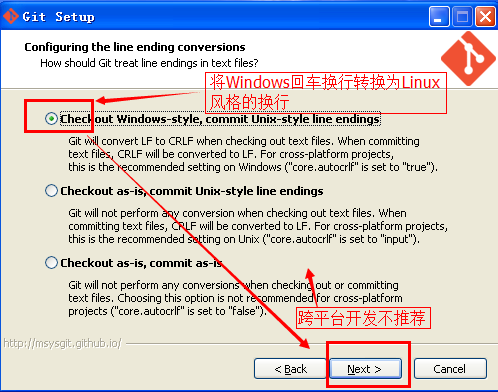
8·接着出现开始菜单文件夹,默认,下一步即可:

[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/06_StartMenu.png)

9·然后是是否配置Path的配置,选择中间一个,可以通过 Windows命令行 (CMD)调用 git 命令。 然后点击下一步.

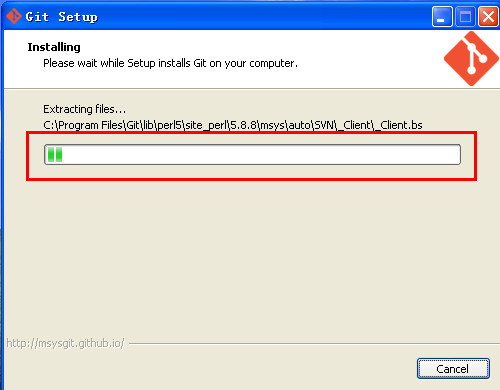
[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/07_GitPath.png)

10·选择回车换行的格式。默认即可.(检出时转换为Windows风格,提交时转换为Linux风格.)

[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/08_CRLF.png)

回车换行风格(CRLF-LF)

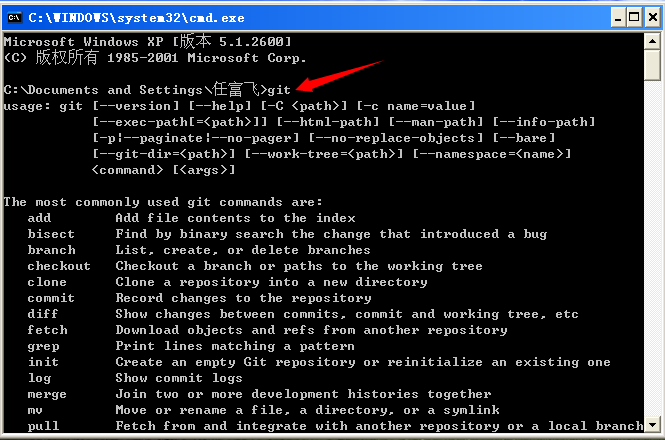
11·然后是安装进度界面

[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/09_Installing.png)

12·安装完成. 去掉那个查看版本说明的复选框,点击完成(Finish)按钮即可.

[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/10_Finish.png)

13·可以在cmd里面测试是否设置了Path,是否安装成功. 在CMD中输入 git 或者 git --version 试试:

[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/11_Git_CMD.png)

14·如果按照前面的步骤安装下来,那么 git 程序所在的路径已经添加到系统 PATH 中(path就相当于系统自动查找路径列表),所以可以直接在任意路径的 cmd 下执行 git 命令. 如果没有添加,则需要 cd 切换到Git所在的 bin 目录下,才能执行 git 命令.

15·设置你自己的昵称与email

设置本地机器默认commit的昵称与Email. 请使用有意义的名字与email.

git config --global user.name "tiemaocsdn"

git config --global user.email "tiemaocsdn@qq.com"

git config --global push.default simple

如果要使用git进行推送,则必须配置 push.default ,否则推送失败. 姓名与Email只用于日志标识.实际推送到GitHub等在线仓库时,要用有操作权限的账号登录.

查看git配置可以使用 -l 参数(l 就是 list 的首字母,L的小写):

git config -l

在某个项目根路径下面可以设置单独的Email与姓名.

git config user.name "tiemaocsdn"

git config user.email "tiemaocsdn@qq.com"

可以看到, 配置单个项目时,少了 --global 参数.

# [2. TortoiseGit安装与配置](http://blog.csdn.net/renfufei/article/details/41647937)

1. [下载](http://blog.csdn.net/renfufei/article/details/41647937#t0)
2. [安装](http://blog.csdn.net/renfufei/article/details/41647937#t1)
3. [配置](http://blog.csdn.net/renfufei/article/details/41647937#t2)
4. [示例](http://blog.csdn.net/renfufei/article/details/41647937#t3)
5. [相关文章](http://blog.csdn.net/renfufei/article/details/41647937#t4)

TortoiseGit 简称 tgit, 中文名海龟Git. 海龟Git只支持神器 Windows 系统, 有一个前辈海龟SVN, TortoiseSVN和TortoiseGit都是非常优秀的开源的版本库客户端. 分为32位版与64位版.并且支持各种语言,包括简体中文(Chinese, simplified; zh\_CN).

**注意:** XP系统已经不支持最新的安装包,请尽量使用 Win7 或更高版本的操作系统.

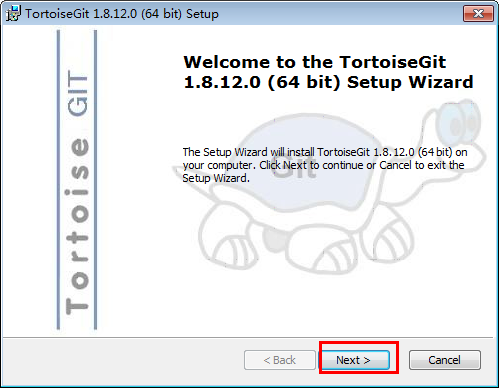
官方说明是: Windows XP 只兼容 1.8.16.0 及之前的版本: (last version which is compatible with Windows XP is [1.8.16.0](https://download.tortoisegit.org/tgit/1.8.16.0/);)

1·下载TortoiseGit安装包

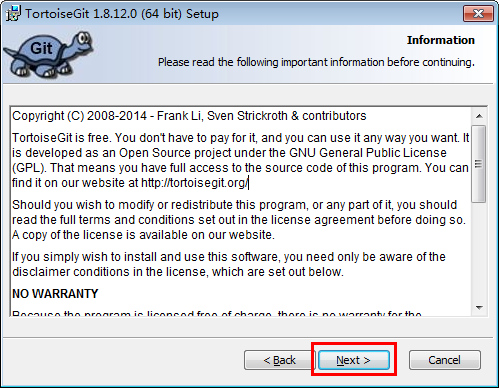
请下载对应的安装包和语言包,放到某个目录下. 然后像上一节所说的 Git 安装包一样, 查看下载完成的文件属性,解除锁定.

2· 我们需要先安装程序包,然后安装语言包(LanguagePack). 因为TortoiseGit 只是一个程序壳,必须依赖一个 Git Core,也就是上一节我们安装的 Git. 所以安装前请确定已完成[上一节的操作](http://blog.csdn.net/renfufei/article/details/41647875). 下面以64位版本为演示(64,32位除文件名不一样,其他的操作都一致)

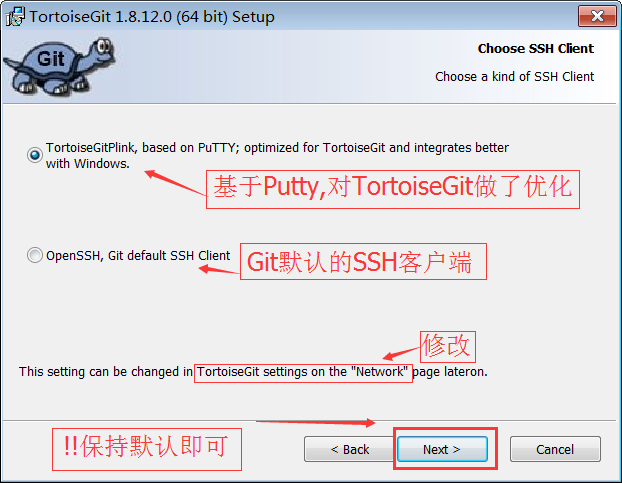
3·双击安装程序 TortoiseGit-2.3.0.0-64bit.msi. 弹出安装向导界面:

[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/203_tgitWizard.png)

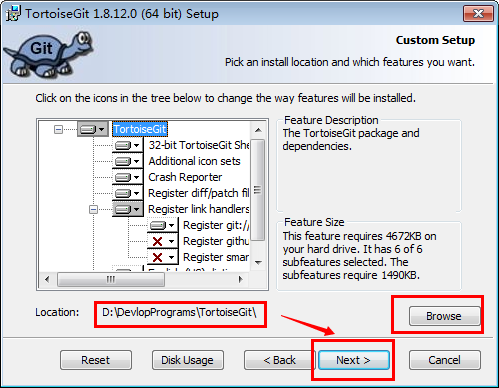
4·下一步,进入版权信息界面. 直接点击下一步(Next)即可.

[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/204_tgit_License.png)

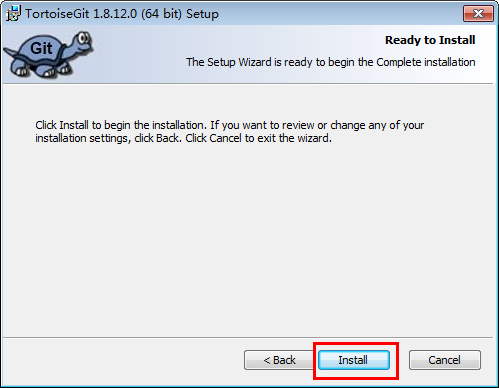
5·下一步,选择SSH客户端. 可以选择 TortoiseGitPlink(位于TortoiseGit安装目录/bin下), 也可以选择 Git 默认的SSH客户端,位于 Git安装目录/bin/ssh.exe (如果配置了Path,那直接是 ssh.exe)

[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/204_2_tgit_Network.png)

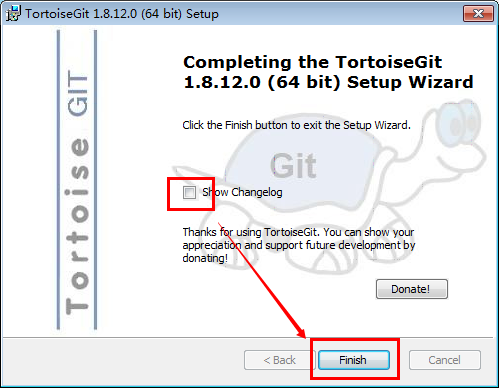
6·接着是选择安装目录,可以保持默认,或者安装到开发环境目录下,安装的程序组件保持默认即可：

[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/205_tgit_dir.png)

7·下一步到确认安装界面,点击 Install按钮安装即可,如下图所示:

[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/206_install_tgit.png)

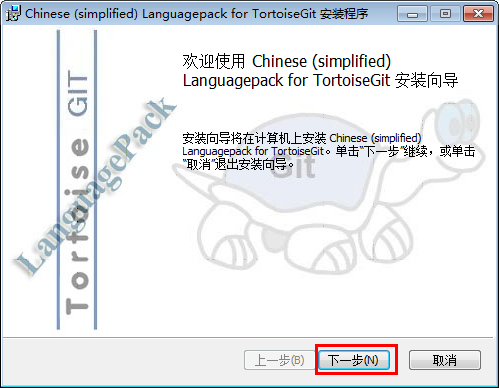
8·安装完成,点击 Finish 按钮即可:

[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/207_tgit_installed.png)

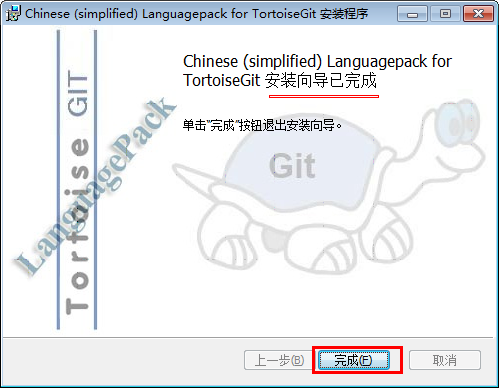
9·如果以前有老版本,则选择覆盖,关闭旧程序并尝试重启即可

10· 安装语言包

双击打开 TortoiseGit-LanguagePack-2.3.0.0-64bit-zh\_CN.msi ,则弹出语言包安装向导:

[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/208_LanguageWizard.png)

11· 点击下一步(Alt+N), 语言包会自动安装完成:

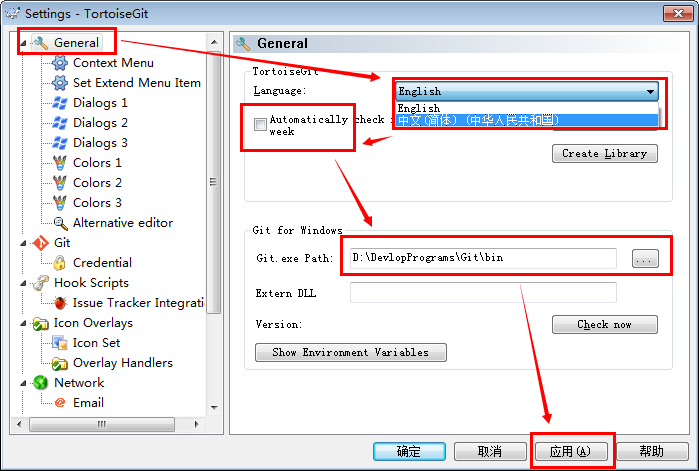
[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/209_LangPackFinished.png)

点击完成按钮即可

## 配置

**1.** 首先,请选定一个存放Git项目的目录,这样管理方便. 如: F:\STUDY\GIT\_STUDY , 然后在资源管理器中打开.

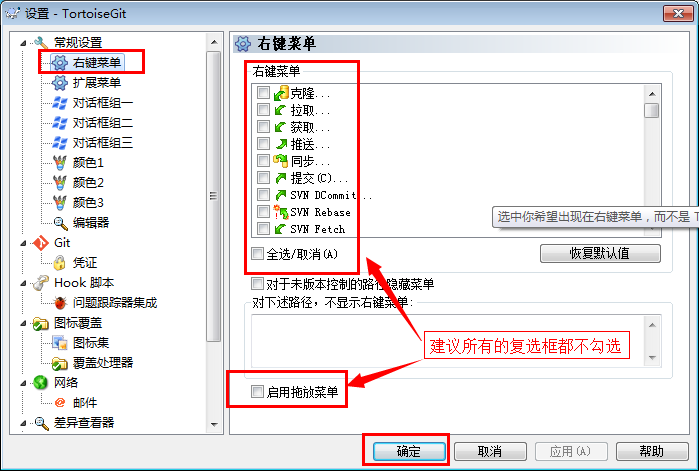
**2.** 在空白处点击鼠标右键, 选择 --> TortoiseGit --> Settings, 然后就可以看到配置界面:

[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/210_LanguageSel.png)

**3.** 选中General,在右边的 Language中选择中文. 不勾选自动升级的复选框,可能还需要指定 Git.exe 文件的路径,如 "D:\DevlopPrograms\Git\bin". 完成后,点击应用,确定关闭对话框.(当然,你也可以继续使用英文)

**4.** 再次点击鼠标右键,可以看到弹出菜单中已经变成中文. 原来的 **Settings** 变成 **设置**; **Clone** 变为 **克隆**.

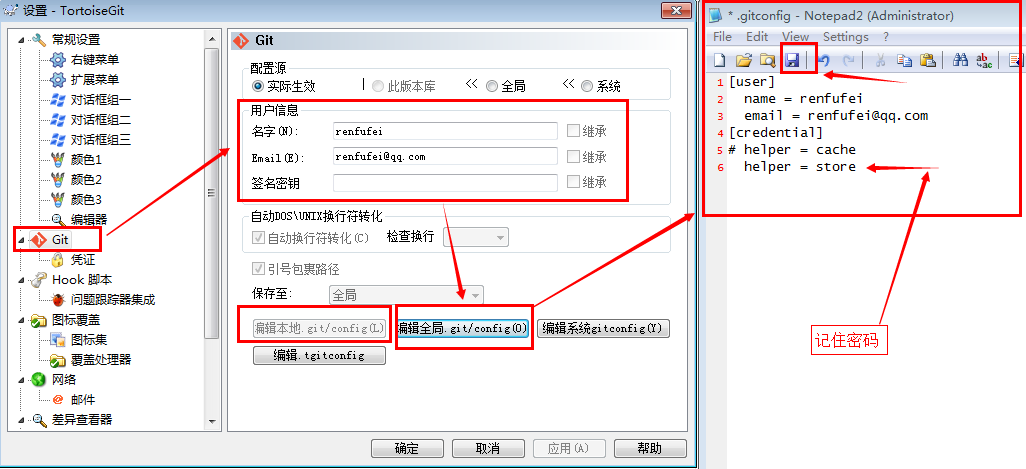
**5.** 配置右键菜单. 在设置对话框中,点选左边的"右键菜单",然后在右边将所有的复选框都去掉,这样右键菜单显得比较干净:

[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/211_contextMenu.png)

**6.**设置记住密码

**!!!!!** 密码会明文保存在 C:\Users\Administrator\.git-credentials 这种文件中, **请小心使用**.

进入设置, 点选左边的Git标签.可以发现,右边可以配置用户的名字与Email信息. 如下图所示:

[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/212_remberpass.png)

因为当前还没有本地项目,所以 “编辑本地 .git/config(L)” 按钮处于灰色不可用状态,如果在某个本地Git项目下打开配置对话框,那么这个按钮就可用,然后就可以编辑此项目的一些属性。

点击 “编辑全局 .git/config(O)”按钮,会使用记事本打开全局配置文件,在全局配置文件中,在后面加上下面的内容:

[credential]

helper = store

完成后保存,关闭记事本,确定即可。

则当你推送项目到GitHub等在线仓库时,会记住你输入的用户名和密码(这里不是用户的姓名和Email哦.)

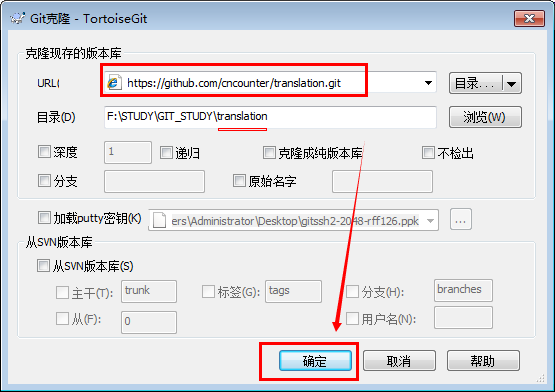
如果你编辑的是 本地 .git/config(L)，其实这个翻译为**本地**有点问题,应该叫**局部**,也就是在某个项目下面设置,只对此项目有效.配置是一样的.

* **用户名**: 就是你注册的账号,如: tiemaocsdn
* **密码**: 当然是注册时填写的密码: \*\*\*\*\*\*\*\*\*
* **Email**: 是你的联系邮箱,给别人联系你时使用
* **用户姓名/昵称**: 可以随便取,但最好有点意义

## 示例

克隆一个项目:

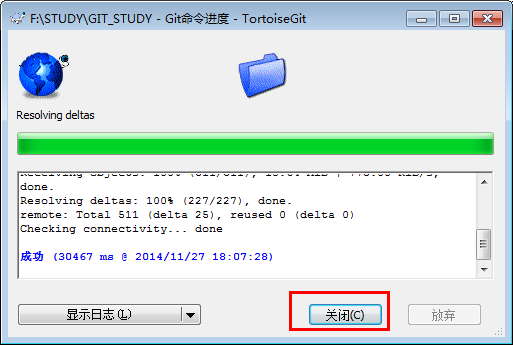
在工作目录下,如 “F:\STUDY\GIT\_STUDY”中,鼠标右键点击空白处,选择: TortoiseGit --> 克隆...(clone...) ；则会弹出克隆对话框,如下图所示:

[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/213_CloneRepository.png)

在URL中填写项目的访问地址,如:

https://github.com/cncounter/translation.git

本地目录,你也可以自己选择,一般默认即可.然后点击"确定"按钮,开始克隆项目. 根据项目大小,时间会不一样. 克隆完成后,如果没有错误,会给出提示:

[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/215_cloneok.png)

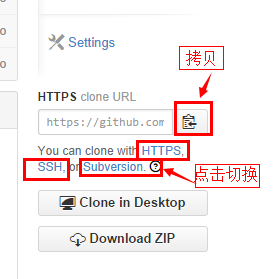
**克隆**: 就类似于拷贝,但是因为SVN,Git之类的版本库存储时都是一个(数据)库,所以叫克隆.

**Git项目的URL**: 可以在线访问Git项目页面,在右侧可以看到SSH, HTTPS, SVN的访问路径.

例如:

* 页面地址: <https://github.com/cncounter/translation>
* HTTPS : <https://github.com/cncounter/translation.git>
* SSH : git@github.com:cncounter/translation.git
* SVN : <https://github.com/cncounter/translation>

可以发现,大同小异. 如下图所示:

[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/214_cloneurl.png)

# [3. Git与TortoiseGit基本操作](http://blog.csdn.net/renfufei/article/details/41647973)

1. [GitHub操作](http://blog.csdn.net/renfufei/article/details/41647973#t0)
2. [Git使用与操作](http://blog.csdn.net/renfufei/article/details/41647973#t1)
   1. [1 克隆项目clone](http://blog.csdn.net/renfufei/article/details/41647973#t2)
   2. [2 提交项目commit](http://blog.csdn.net/renfufei/article/details/41647973#t3)
   3. [3 推送项目push](http://blog.csdn.net/renfufei/article/details/41647973#t4)
   4. [4 拉取项目pull](http://blog.csdn.net/renfufei/article/details/41647973#t5)
3. [TortoiseGit使用与操作](http://blog.csdn.net/renfufei/article/details/41647973#t6)
   1. [1 克隆项目](http://blog.csdn.net/renfufei/article/details/41647973#t7)
   2. [2 修改提交项目](http://blog.csdn.net/renfufei/article/details/41647973#t8)
   3. [3 将提交到本地的项目推送到在线仓库](http://blog.csdn.net/renfufei/article/details/41647973#t9)
   4. [4 拉取项目pull](http://blog.csdn.net/renfufei/article/details/41647973#t10)
4. [更复杂的操作](http://blog.csdn.net/renfufei/article/details/41647973#t11)
5. [相关文章](http://blog.csdn.net/renfufei/article/details/41647973#t12)

## 1. GitHub操作

本节先简单介绍 git 的使用与操作, 然后再介绍 TortoiseGit 的使用与操作.

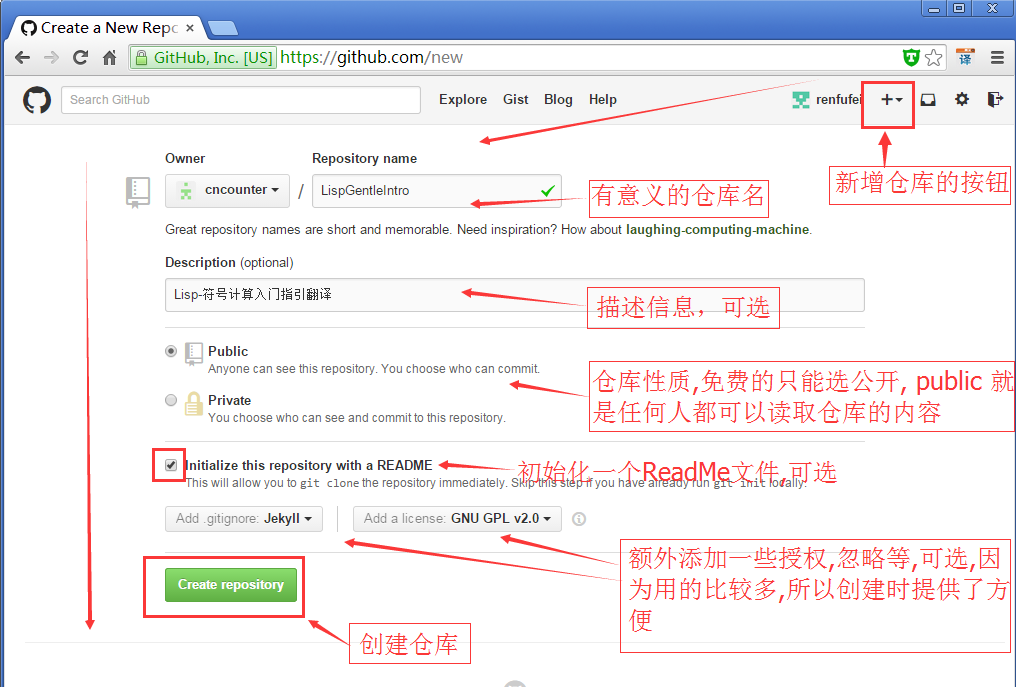
先看看SVN的操作吧, 最常见的是 检出(Check out ...), 更新 (Update ...), 以及 提交(Commit ...); 相当于有一个服务器端, 可以多个客户端进行访问和操作.

Git是SVN以后的新一代的版本控制系统/协议. 适合于大规模的开源软件多人协作的开发. 有一个总仓库(如GitHub,CSDN,OSchina等提供), 每个客户端都有一个本地仓库(这个本地仓库有点对应于SVN的服务端仓库), 以及一个可以看见的本地项目.

让我们想想,至少需要有哪些功能.以 GitHub为例,首先,必须得有一个线上仓库(Repository), 这首先得新建一个仓库.

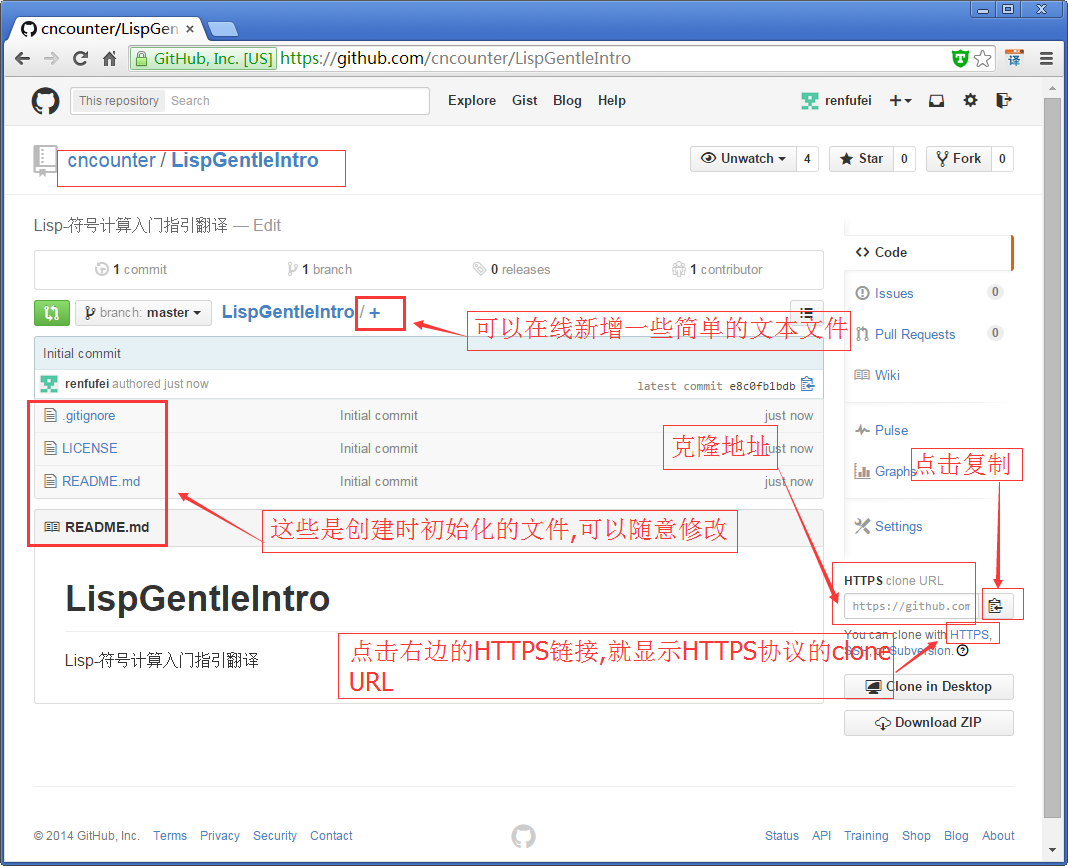
要创建仓库,必须得先**注册账号**. 请参考: [创建GitHub技术博客全攻略: http://blog.csdn.net/renfufei/article/details/37725057](http://blog.csdn.net/renfufei/article/details/37725057)

然后,登录 GitHub, 在右上角找到 New Repository 或者 加号下拉按钮(+),点击进入新建仓库页面: <https://github.com/new>, 如下图所示:

[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/301_new_repository.png)

依次填写仓库名,以及其他信息后,点击 "**Create repository**" 按钮,即可创建一个在线仓库. 因为这个仓库是挂在你的账号下的,所以可以是任意合法的字符,只要和你的其他仓库不冲突即可.

仓库创建成功后,就会进入仓库预览页面, 如下图所示:

[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/302_cloneURL.png)

然后，我们可以点击右边的 HTTPS 链接, 上方的文本框里面就显示了 HTTPS协议下的仓库访问地址, 可以点击进去全选,也可以点击右边的复制按钮复制到剪贴板. 例如,刚刚创建的项目访问路径是:

https://github.com/cncounter/LispGentleIntro.git

是一个以 https:// 开始,以 .git 结尾的 URL,根据提示,叫做 clone URL.

好了,仓库创建完成,可以进入下一步了.

## 2. Git使用与操作

如果你偏爱命令行,那么可以简单参考下这一小节. 否则,请往下翻,直接看 [**TortoiseGit使用与操作**](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/03_Usage.md#tgitoperate)

### 2.1 克隆项目(clone ...)

Win+R 快捷键打开运行对话框,输入 cmd 回车, 在 cmd 中(其实在 GitBash中也可以), cd 切换到存放git项目的目录, 如:

Microsoft Windows [版本 6.1.7601]

版权所有 (c) 2009 Microsoft Corporation。保留所有权利。

C:\Users\Administrator>F:

F:\>cd GIT\_ALL

然后执行命令: git clone https://github.com/cncounter/LispGentleIntro.git ,就会把项目克隆到当前目录下, 项目的目录名字默认为 **LispGentleIntro**.

F:\GIT\_ALL>git clone https://github.com/cncounter/LispGentleIntro.git

Cloning into 'LispGentleIntro'...

remote: Counting objects: 5, done.

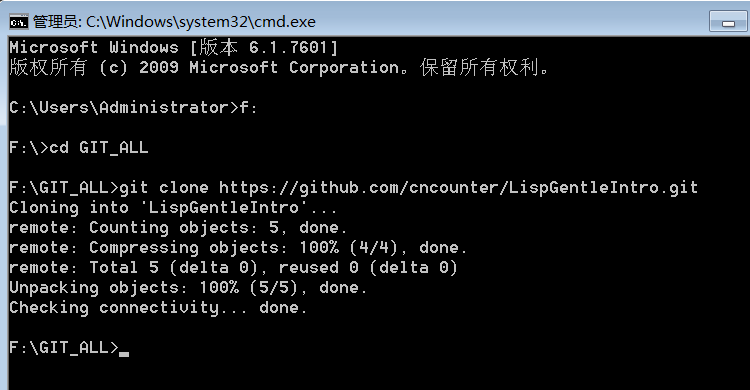
remote: Compressing objects: 100% (4/4), done.

remote: Total 5 (delta 0), reused 0 (delta 0)

Unpacking objects: 100% (5/5), done.

Checking connectivity... done.

命令行截图如下:

[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/303_gitclone.png)

### 2.2 提交项目(commit ...)

然后可以 cd 切换到 LispGentleIntro 目录, 新增或者修改某些文件.这里只是模拟一下操作. 实际情况可能是 使用 Eclipse 导入项目(不要拷贝,Copy...)进行一些编辑, 然后将当前目录下的所有有变动的文件告诉索引库,接着提交到本地仓库.

F:\GIT\_ALL>cd LispGentleIntro

F:\GIT\_ALL\LispGentleIntro>echo some content >> cncounter.txt

F:\GIT\_ALL\LispGentleIntro>echo by cncounter >> README.md

F:\GIT\_ALL\LispGentleIntro>git add .

F:\GIT\_ALL\LispGentleIntro>git commit -m "add some info."

[master 86d090f] add some info.

2 files changed, 4 insertions(+)

create mode 100644 cncounter.txt

提交到本地仓库有什么好处呢? 本地仓库就是一个版本管理库,我们在编写代码时,如果写错了,那么可能需要 Ctrl+Z 来撤销,撤销撤销,如果很多,而且跨文件,就很麻烦,可能需要Copy文件什么的。 这时候本地仓库就很有用了. 修改到某个阶段,然后就提交到本地仓库. 可以有效防止丢失,方便回退. 而且,提交到本地仓库不会影响别人看到的内容(只有本机可见).

### 2.3 推送项目(push ...)

如果完成到一定程度,那么可以推送到远端在线仓库. 推送之前,请确保你已经设置了全局的 user.name 和 user.email, 如果没有设置,或者不想使用全局的信息,可以在当前项目下设置:

F:\GIT\_ALL\LispGentleIntro>git config user.name "renfufei"

F:\GIT\_ALL\LispGentleIntro>git config user.email "renfufei@qq.com"

接着,可以执行推送(push), 推送就是将已经提交到本地仓库的那部分内容给推到远端在线仓库.(修改了,但没提交的那部分内容,不会被推送). 执行 git push 命令即可:

F:\GIT\_ALL\LispGentleIntro>git push

Username for 'https://github.com': renfufei

Password for 'https://renfufei@github.com':

Counting objects: 6, done.

Delta compression using up to 8 threads.

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

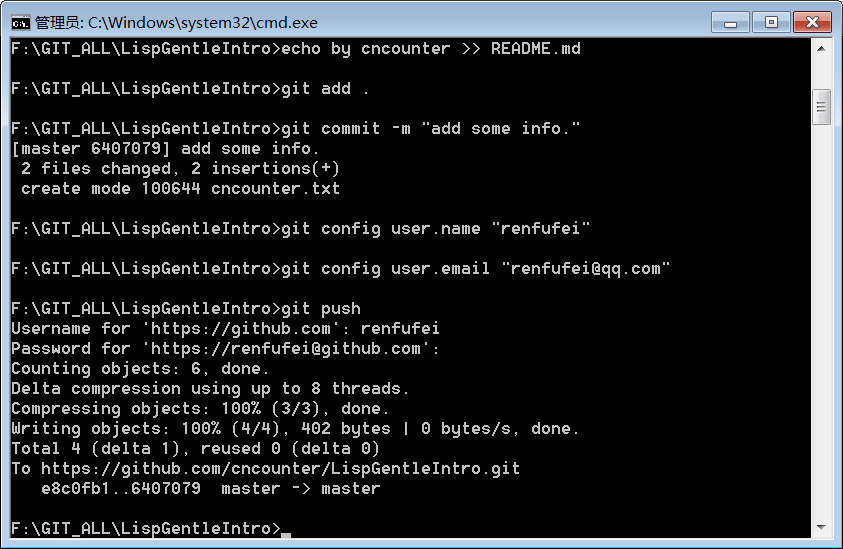
Writing objects: 100% (4/4), 402 bytes | 0 bytes/s, done.

Total 4 (delta 1), reused 0 (delta 0)

To https://github.com/cncounter/LispGentleIntro.git

e8c0fb1..6407079 master -> master

在连接上远端服务器之后,会要求你输入用户名与密码. 其中,用户名是你注册GitHub时的账号,不是邮箱. 接着是密码,注意密码是没有回显的,也就是为了安全,不告诉你具体输入了几个字符,输入完成回车即可. 截图如下:

[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/304_git_push.png)

好了,推送完成,这时候用浏览器打开在线仓库,看看内容是不是已经变了?

### 2.4 拉取项目(pull ...)

当然,如果多人协作, 或者多个客户端进行修改, 那么我们还要拉取(Pull ... )别人推送到在线仓库的内容下来。 大神们是不推荐使用 pull 命令进行拉取的, 因为封装了细节(git pull == git fetch + git merge). 对于这群更喜欢用命令行的神们来说, 一切在掌控之中是一种强迫症!!!(开个玩笑, 其实项目成员复杂,约定不好以后, pull 确实会有很多问题,会坑人.)

常规使用,我们执行 git pull 即可:

F:\GIT\_ALL\LispGentleIntro>git pull

Already up-to-date.

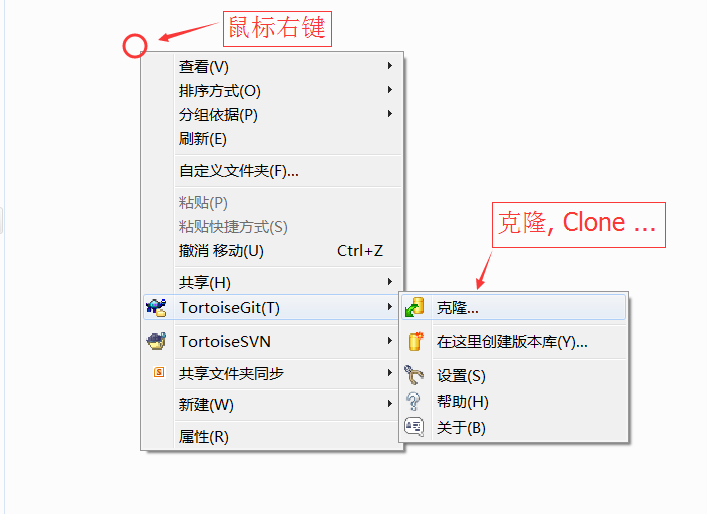
当然,因为没有其他文件被修改,所以直接提示 已经更新到最新. 常规操作就这些,需要注意的是,和使用SVN的好习惯一样,你在修改本地内容之前,最好先 pull 一下,减少冲突的可能.

## 3. TortoiseGit使用与操作

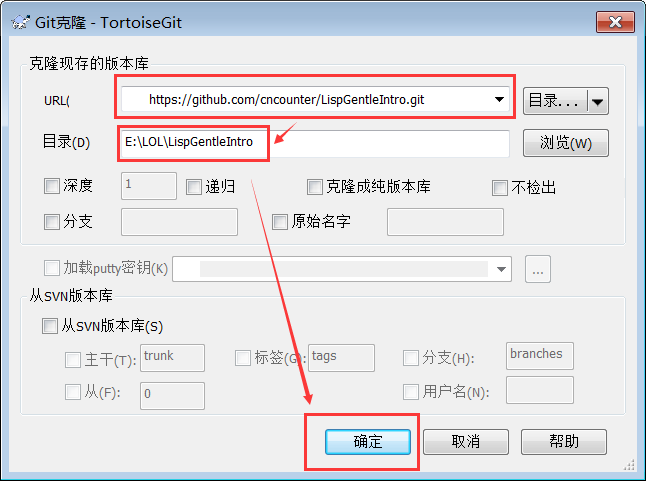
使用 Git命令有时候确实不怎么方便,特别是每次都要输入密码,如果配置 SSH 的方式,又实在是很麻烦.(当然,必须使用 Windows 神器才有方便友好的客户端图形界面啦!!!)

### 3.1 克隆项目

打开资源管理器(我的电脑/计算机), 进入规划好的某个目录中, 然后在空白处点击鼠标右键, 选择 TortoiseGit --> 克隆... (Clone...).

[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/401_rightclick.png)

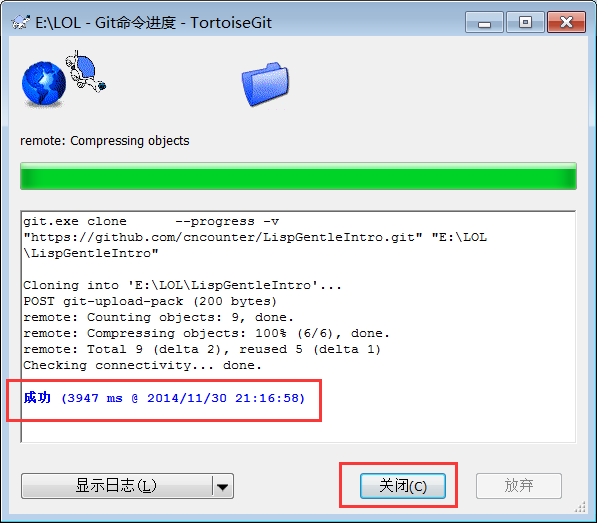
弹出克隆项目对话框:

[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/402_tgitclone.png)

在对话框的 URL中输入项目地址,如:

https://github.com/cncounter/LispGentleIntro.git

确定本地目录,然后点击 **确定** 按钮. 等待完成后,点击关闭按钮即可.

[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/403_tgitcloned.png)

### 3.2 修改提交项目

项目克隆完成后(可以将克隆 clone 理解为 下载, 检出 checkout 操作). 修改本地项目中的某些文件,如 将 README.md 修改为如下内容:

LispGentleIntro

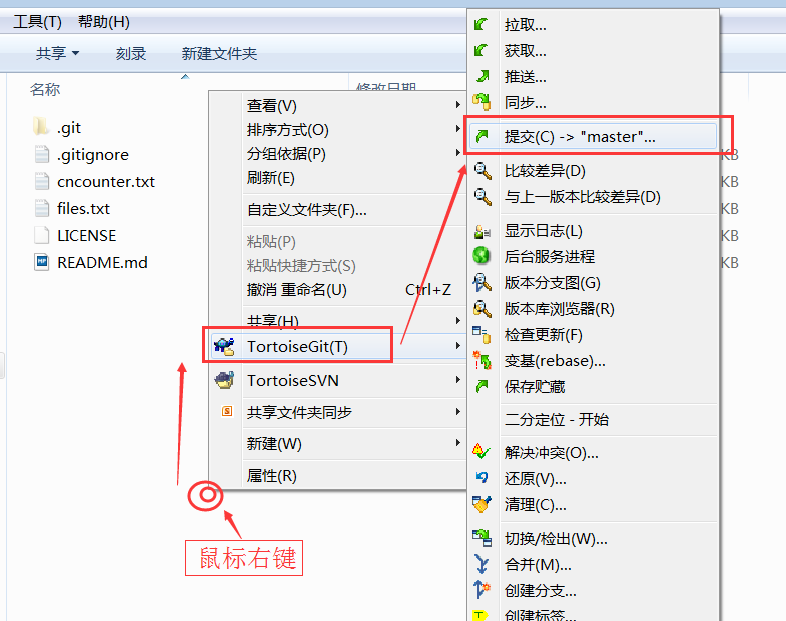
===============

Lisp-符号计算入门指引翻译

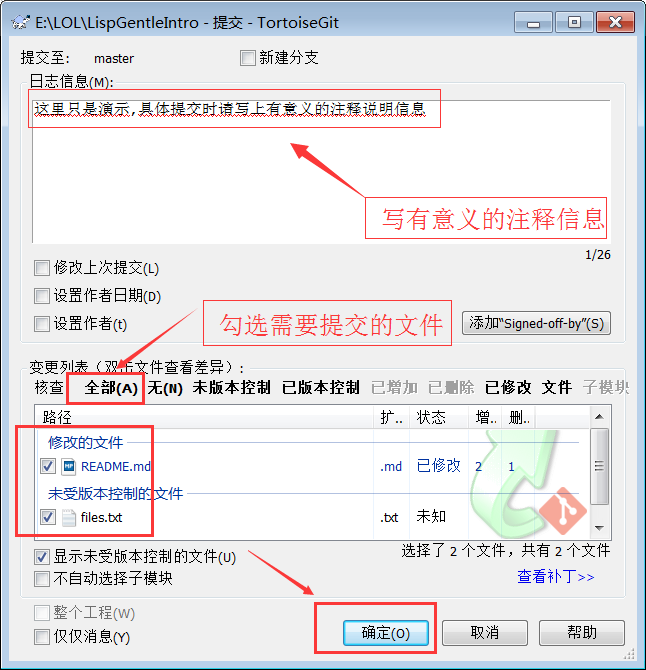
<a href="http://cncounter.duapp.com/">天朝计数器</a>

还可以试试增加一些文件. 如 files.txt .

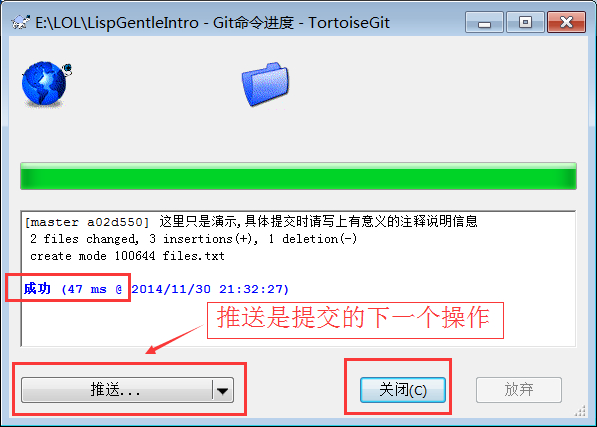
然后在本地项目的空白处点击鼠标右键, 选择 **TortoiseGit --> 提交(C) -> "master"...** 或: Commit -> "master"....

[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/404_rcick_commit.png)

弹出提交（Commit）对话框:

[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/405_commitlog.png)

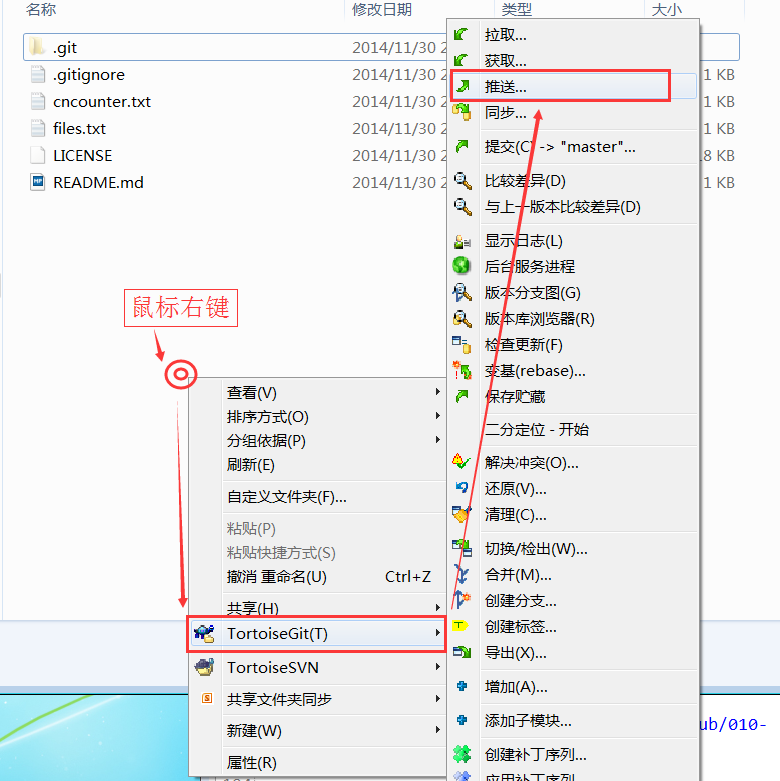
作为好习惯,填写提交日志,勾选需要提交的文件,然后点击 "确定" 按钮, 即提交到本地仓库.

[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/406_commit2push.png)

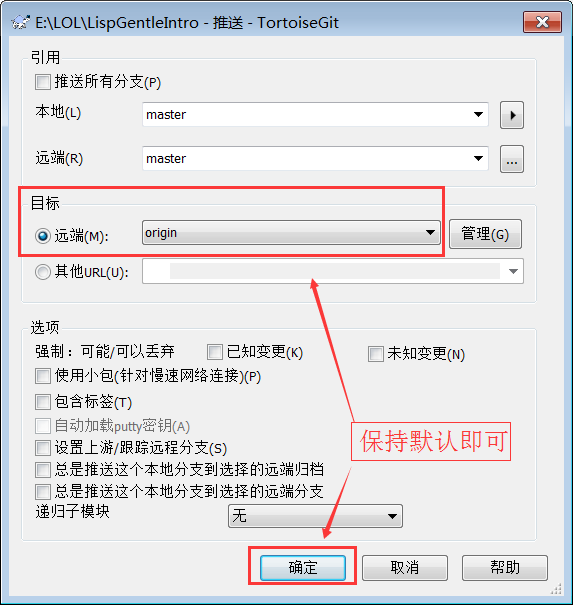
### 3.3 将提交到本地的项目推送到在线仓库

推送是提交的下一步操作.

在本地项目的空白处点击鼠标右键, 选择 **TortoiseGit --> 推送...** 或: Push....

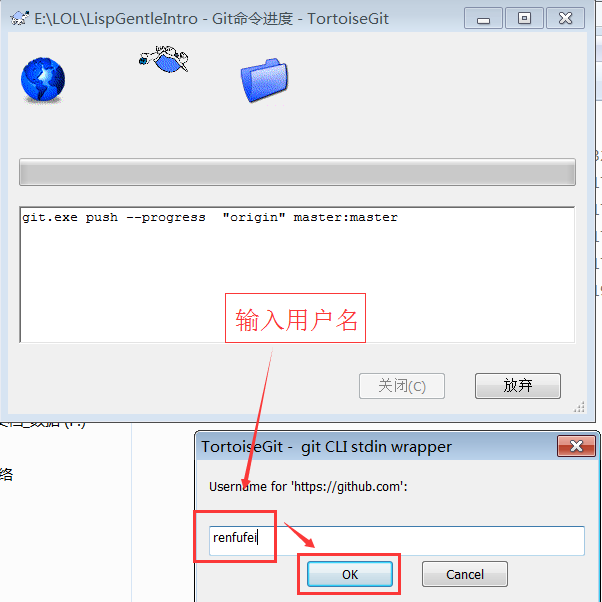
[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/407_tgit_rclick_push.png)

弹出推送(push)对话框:

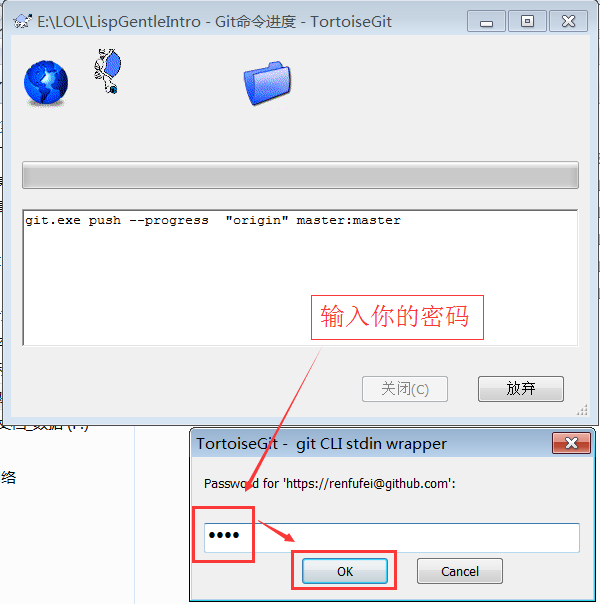
[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/408_tgit_push.png)

一般保持默认,点击 “**确定**” 按钮.

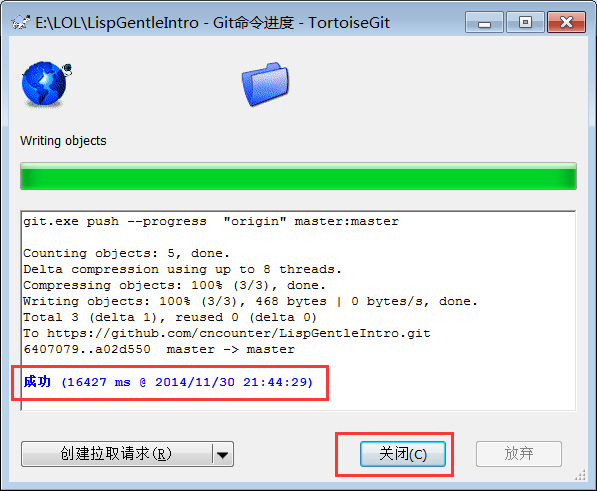
然后弹出推送进度界面, 可能要求你输入用户名:

[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/409_username.png)

确定OK，然后要求输入密码:

[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/410_pass.png)

密码输入正确后,OK，显示推送成功界面:

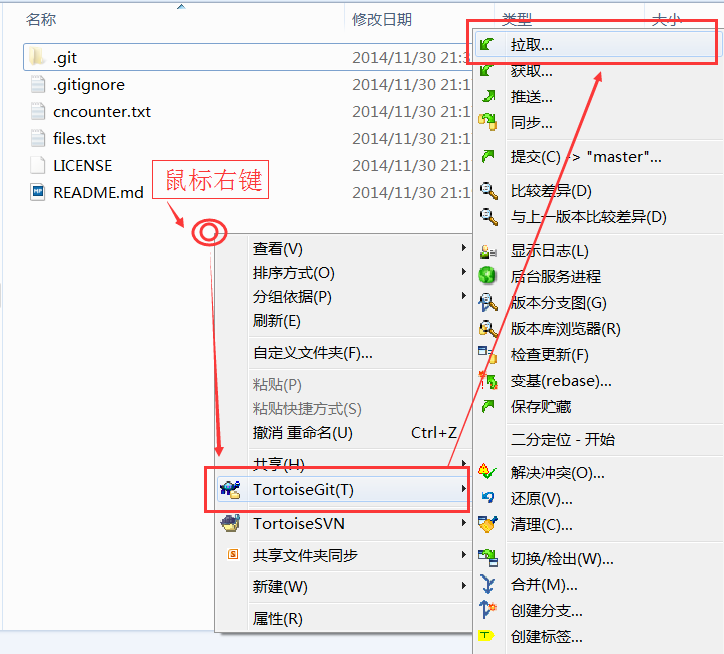
[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/411_pushok.png)

如果你按照上一小节的设置操作,则输入密码以后会记住密码. 密码会明文保存在 C:\Users\Administrator\.git-credentials 这种文件中, 请小心使用.

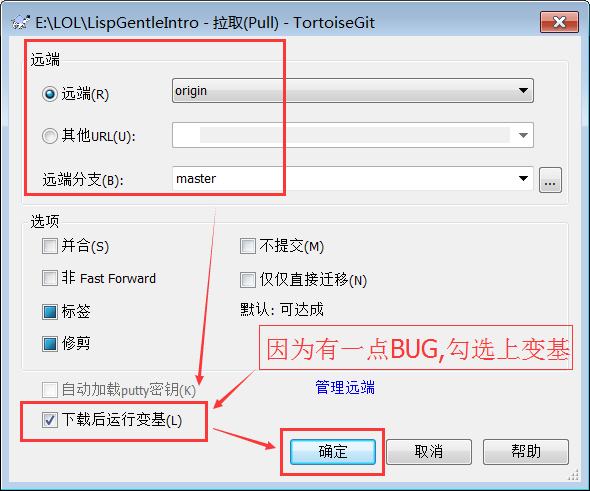
### 3.4 拉取项目(pull ...)

如果本地的项目没有在线仓库的新,则需要执行拉取操作(Pull ...).

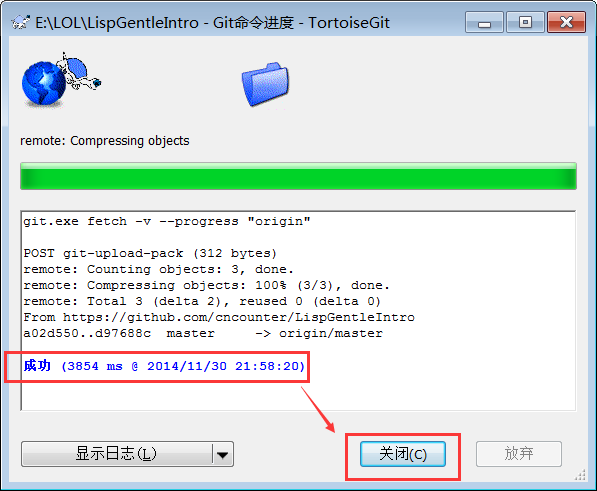
在本地项目的空白处点击鼠标右键, 选择 **TortoiseGit --> 拉取...** 或: Pull....

[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/412_rck_pull.png)

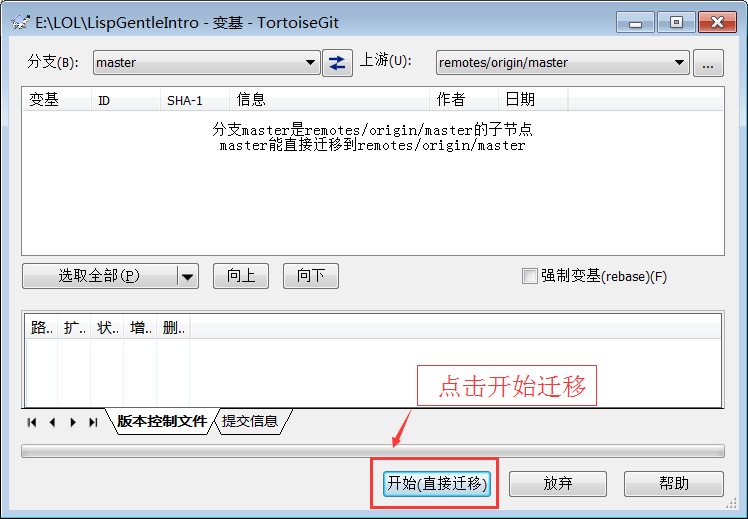
弹出拉取(pull)对话框:

[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/413_pull_dialog.png)

【如果拉取有BUG,请不要使用 rebase, 下载[最新的 TortoiseGit](http://download.csdn.net/detail/renfufei/9738152) 即可】, 进入 拉取进度界面:

[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/414_pulled.png)

【如果拉取有BUG,请不要使用 rebase, 下载[最新的 TortoiseGit](http://download.csdn.net/detail/renfufei/9738152) 即可】

[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/414_c_base.png)

然后完成即可.

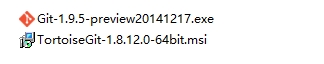
这应该是 TortoiseGit 的一个BUG, 要解决这个问题,请参考 [解决 TortoiseGit 诡异的 Bad file number 问题](http://blog.csdn.net/renfufei/article/details/41648061) (或者可以换回老版本. 如果你没有出这个问题,那么,可能是我的机器哪里损坏了).

# windows安装TortoiseGit详细使用教程【基础篇】

安装准备：

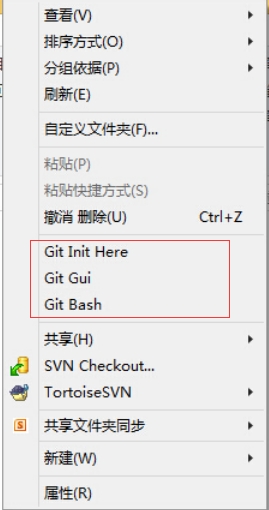
1. 首先你得安装windows下的git [msysgit1.9.5](https://github.com/msysgit/msysgit/releases/download/Git-1.9.5-preview20141217/Git-1.9.5-preview20141217.exe)
2. 安装版本控制器客户端tortoisegit  [tortoisegit1.8.12.0](http://download.tortoisegit.org/tgit/1.8.12.0/) [32和64别下载错，不习惯英文的朋友，也可以下个语言包]

一、安装图解：

[](http://s3.51cto.com/wyfs02/M01/58/A2/wKioL1S2cGWDgtw7AAAzV1JtLz4385.jpg)

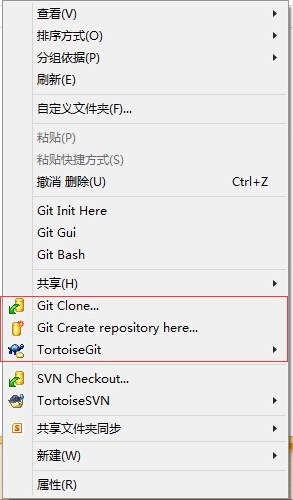
1. 先安装GIT【一路默认即可】

安装好git以后，右键，会发现菜单多了几项关于GIT的选项

[](http://s3.51cto.com/wyfs02/M00/58/A5/wKiom1S2caHx4weEAAEH0iQdstE374.jpg)

2.安装tortoisegit【一路默认即可】

安装好以后，右键，会发现菜单多了几项关于tortoisegit的选项

[](http://s3.51cto.com/wyfs02/M02/58/A5/wKiom1S2dyOg2eTzAAEqf6QkmoY252.jpg)

到此，安装算完成了，相比tortoiseSVN，安装完tortoiseGIT不需要重启电脑。

接下来是使用tortoiseGIT

二、图解使用tortoiseGIT

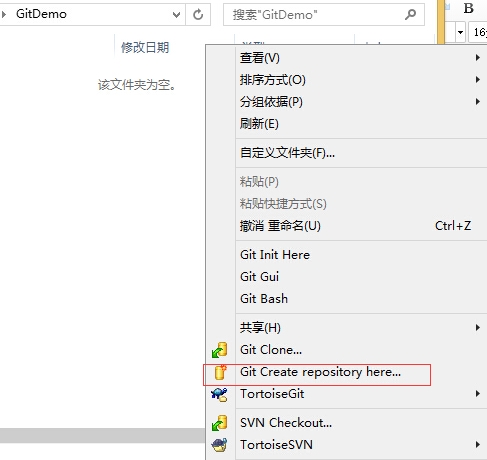
这里只是做一些最基本的功能的演示：

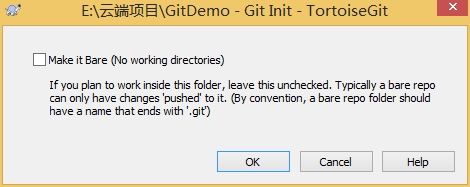
创建版本库，提交(commit)文件，推送(push)文件，更新文件，创建分支。

简介：git属于分布式版本控制器，其实每个人的电脑都是一个完整的版本库；比如两个人在局域网写一个项目，那么这两个人的电脑里面都是一个完整的项目，当有新的代码出现的时候，直接push给对方即可。当然人多的时候，又不在局域网里面，不可能push给每一个人，git的解决方案是用一台电脑当做集中的版本库，大家把新代码都push到上面去，要得时候在pull到自己电脑上。国外比较著名的就是github了,国内的oschina也广为人知。

1. 创建一个版本库

创建一个文件夹GitDemo，然后  右键--Git Create repository here...

[](http://s3.51cto.com/wyfs02/M00/58/A2/wKioL1S2fNPRoYPhAAEePBn7-gA930.jpg)

[](http://s3.51cto.com/wyfs02/M01/58/A5/wKiom1S2fOWy-rywAADGJqlins8081.jpg)

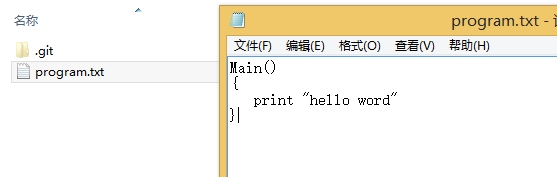
此处不用勾选Make is Bare

[](http://s3.51cto.com/wyfs02/M02/58/A2/wKioL1S2fbHz9Ry7AABS-_Bho2M886.jpg)

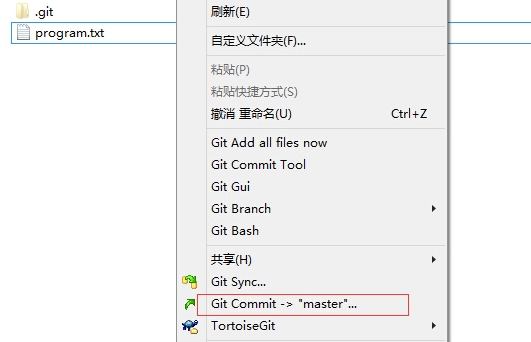
确定后，在GitDemo文件夹下回多一个.git文件夹【电脑要设置为显示隐藏文件夹】

此文件夹，尽量不要乱修改或删除。

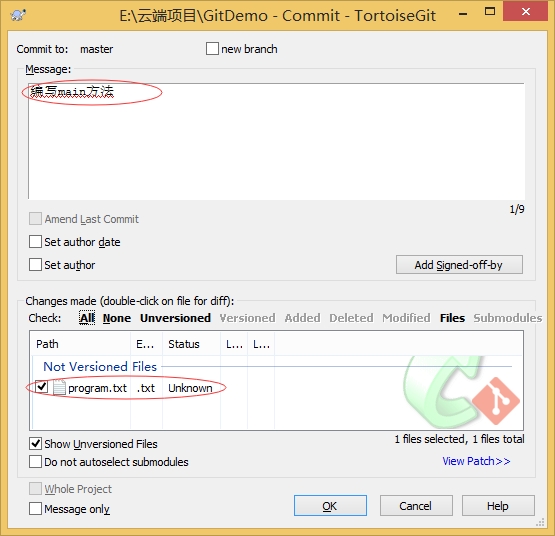
2.创建一个文件，然后提交(commit)到版本库[这个操作可以在离线状态操作]

[](http://s3.51cto.com/wyfs02/M02/58/A2/wKioL1S2geSRWeEWAACAZDlmXzw696.jpg)

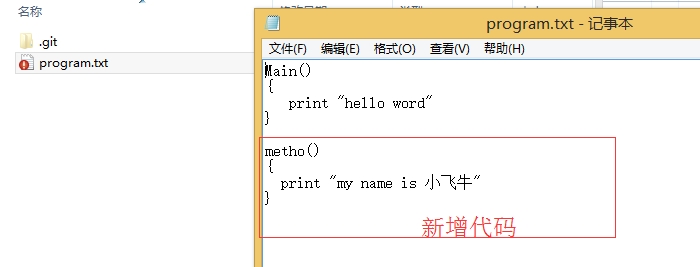
2.1选择Git Commit->"master"

[](http://s3.51cto.com/wyfs02/M01/58/A2/wKioL1S2gnfB9Wq_AADu--urKHw095.jpg)

2.2填写message，勾选要commit的文件，然后点击OK。会在弹出一个窗口点击close即可。

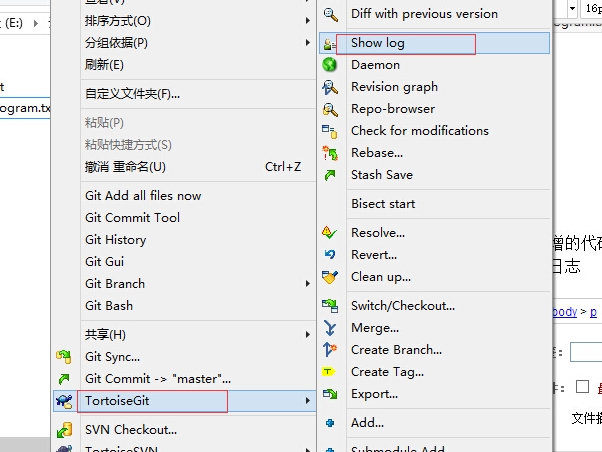
[](http://s3.51cto.com/wyfs02/M02/58/A2/wKioL1S2gtrQzKatAAIGrB0nv8w918.jpg)

3.然后在program中添加一个新的方法method。当文件代码改变，那么文件会显示为一个红色感叹号

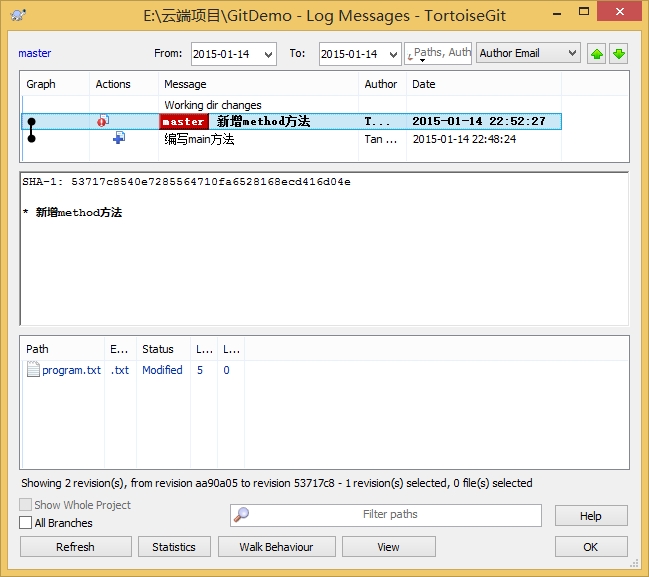
[](http://s3.51cto.com/wyfs02/M00/58/A2/wKioL1S2g5aRaP6GAADVFkMb_6M134.jpg)

4.将新增的代码，提交到版本库中，参考2.1和2.2步。

5.查看日志

[](http://s3.51cto.com/wyfs02/M00/58/A5/wKiom1S2hCbx0ucYAAJRb8cNTlM825.jpg)

在日志中，文件更改的信息，都很清楚。

[](http://s3.51cto.com/wyfs02/M02/58/A5/wKiom1S2hHLhOlA7AAJiRVFH9fY252.jpg)

6.将代码推送(push)到云端，例如：github、oschina上面

   6.1:在oschina上新建一个项目,进入oschina-->Git@OSC-->新建项目

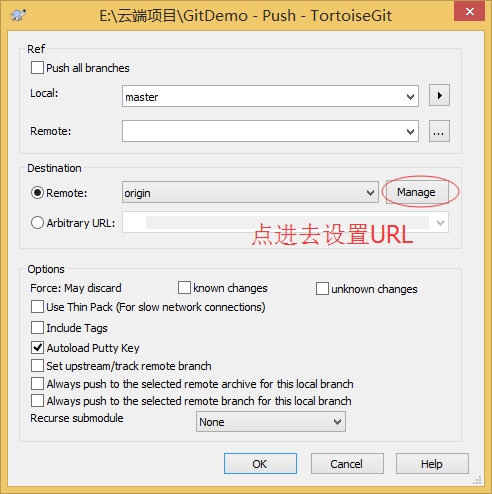
[](http://s3.51cto.com/wyfs02/M01/58/A2/wKioL1S2hligPtqIAAGekalEh3o374.jpg)

点击创建项目，下图的红框就是项目的URL了

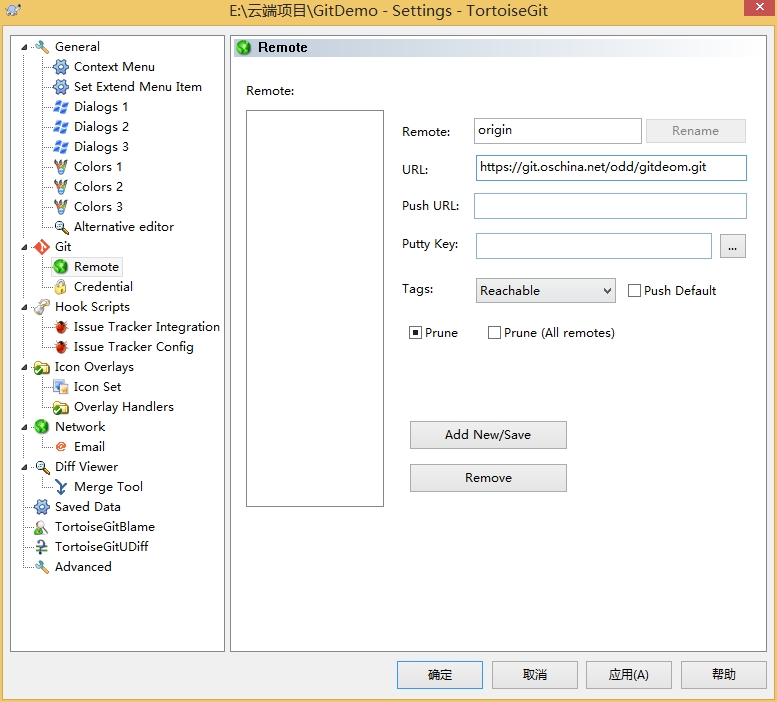
[](http://s3.51cto.com/wyfs02/M00/58/A2/wKioL1S2h27hXZlDAADMIxMjQPs030.jpg)

6.2将电脑得代码push到云端。

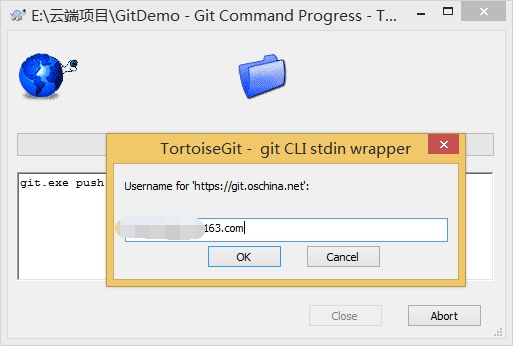
右键-->tortoiseGit-->push

[](http://s3.51cto.com/wyfs02/M00/58/A5/wKiom1S2iE2TKsACAAH33MtS4vo136.jpg)

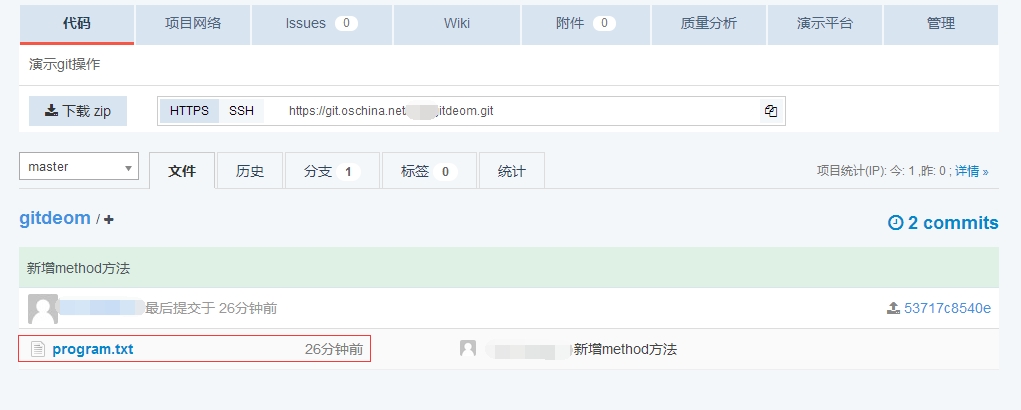
设置好URL，确定。然后开始push，输入用户名，输入密码就OK了

[](http://s3.51cto.com/wyfs02/M01/58/A2/wKioL1S2iUHzLNcQAAME87a87ME496.jpg)

输入用户名，后面还要输入密码，然后等待成功。

[](http://s3.51cto.com/wyfs02/M01/58/A5/wKiom1S2iLvDMZh5AAEP4gIvaL4995.jpg)

6.3成功后，我们去oschina上看看push的项目吧

[](http://s3.51cto.com/wyfs02/M02/58/A2/wKioL1S2igmATaqRAAHJqxGZciM938.jpg)

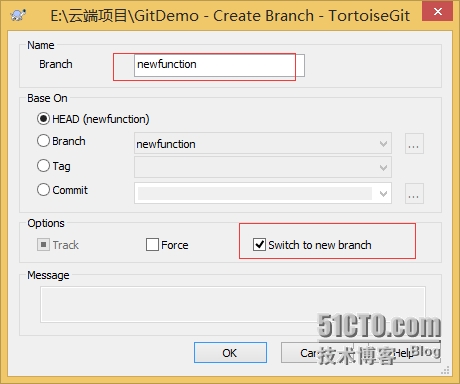
7.git还有个特别好用的功能，分支。

发现问题：你代码写了很多，运行OK；但是突然想加个新功能进去，这个功能你也不知道能否正常运行，而且修改过程中，除了新加代码和文件进去，还会修改以前的代码。要是万一失败，修改回来也是一种很麻烦的事情。这种时候很多人就用备份方式。来看看git是怎么优雅的处理这个问题的。

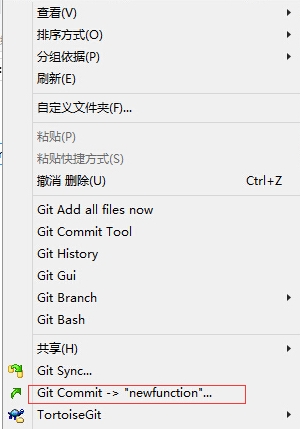
git的处理方式：当你想加一个新功能进去的时候，你可以新建一个分支，例如名字叫newfunction，然后在分支中把新功能加上去，如果OK，将代码合并到master分支上，如果新功能失败，切换回master分支上来，在newfunction写的代码，又全看不到了。

[ps:master是新建项目的主分支]

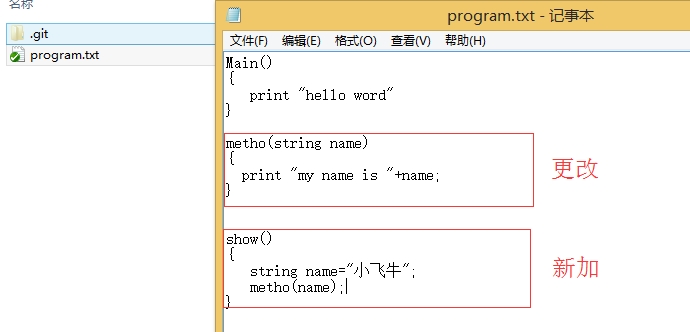
7.1新建一个分支，右键-->tortoiseGit-->Create Branch

[](http://s3.51cto.com/wyfs02/M02/58/A2/wKioL1S2jeWjAmwPAAE2UQMZ6i4217.jpg)

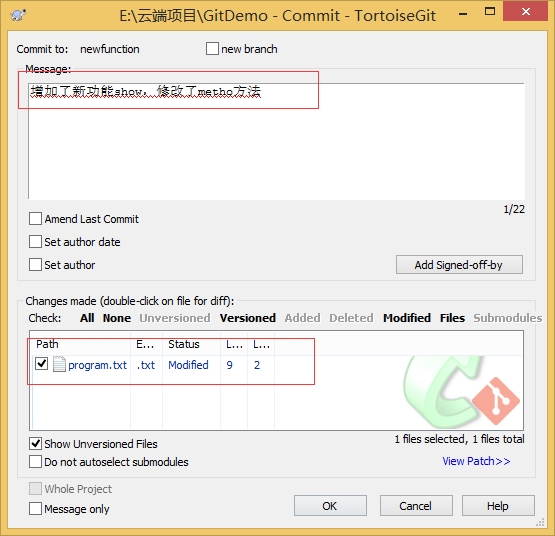
右键，你会发现当前的分支为你新建的newfunction分支了

[](http://s3.51cto.com/wyfs02/M02/58/A6/wKiom1S2jXjCcTaPAAD1j9rKeKM309.jpg)

7.2接下来，就开心新增加你的功能；我们再proprame.txt中不仅增加了show方法，还修改了metho方法

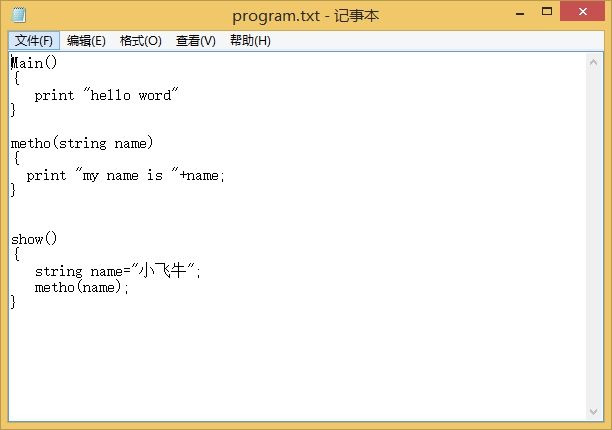
[](http://s3.51cto.com/wyfs02/M00/58/A2/wKioL1S2jmyj9-dnAAEU2pHaKUk307.jpg)

7.3提交我们的代码，此时我们发现这个新功能不行，存在很多漏洞等等问题。我们想回到之前的代码。怎么办呢？

[](http://s3.51cto.com/wyfs02/M02/58/A3/wKioL1S2j5GSSoE4AAIdIWi_tV8665.jpg)

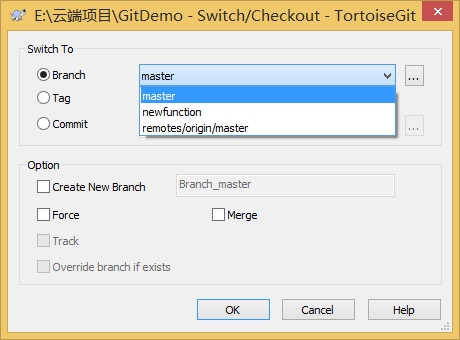
7.4为了回到之前的项目，我们直接切回master分支。

先来看看现在的文件内容，新加的和修改的都在。

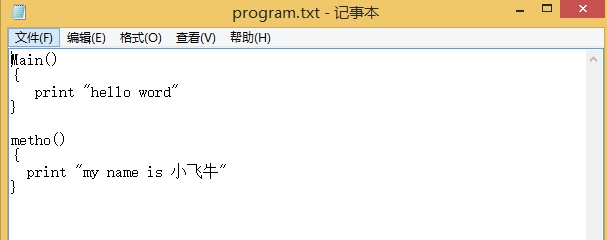
[](http://s3.51cto.com/wyfs02/M00/58/A6/wKiom1S2j--CQedUAAD3tf5SDWk378.jpg)

然后我们切换到master分支，右键-->tortoiseGit-->Switch/Checkout

选择master即可，然后确定

[](http://s3.51cto.com/wyfs02/M01/58/A6/wKiom1S2kIWR_elPAAEO8ggIGnY473.jpg)

再来看看program中的代码是否回到了从前呢？

[](http://s3.51cto.com/wyfs02/M02/58/A3/wKioL1S2kYbgz0kxAACfBVCWAH8487.jpg)

无论是修改，还是添加，都没影响到复原。这就是分支的作用。这个功能是不是让你省去了很多工作量呢。