3. Git与TortoiseGit基本操作

原创 2014年12月01日 15:27:59

* 标签：
* [Git](http://so.csdn.net/so/search/s.do?q=Git&t=blog) /
* [TortoiseGit](http://so.csdn.net/so/search/s.do?q=TortoiseGit&t=blog) /
* [基本操作](http://so.csdn.net/so/search/s.do?q=基本操作&t=blog) /
* [使用](http://so.csdn.net/so/search/s.do?q=使用&t=blog) /
* [Windows](http://so.csdn.net/so/search/s.do?q=Windows&t=blog) /
* 116560
* [编辑](http://write.blog.csdn.net/postedit/41647973)
* 删除

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.

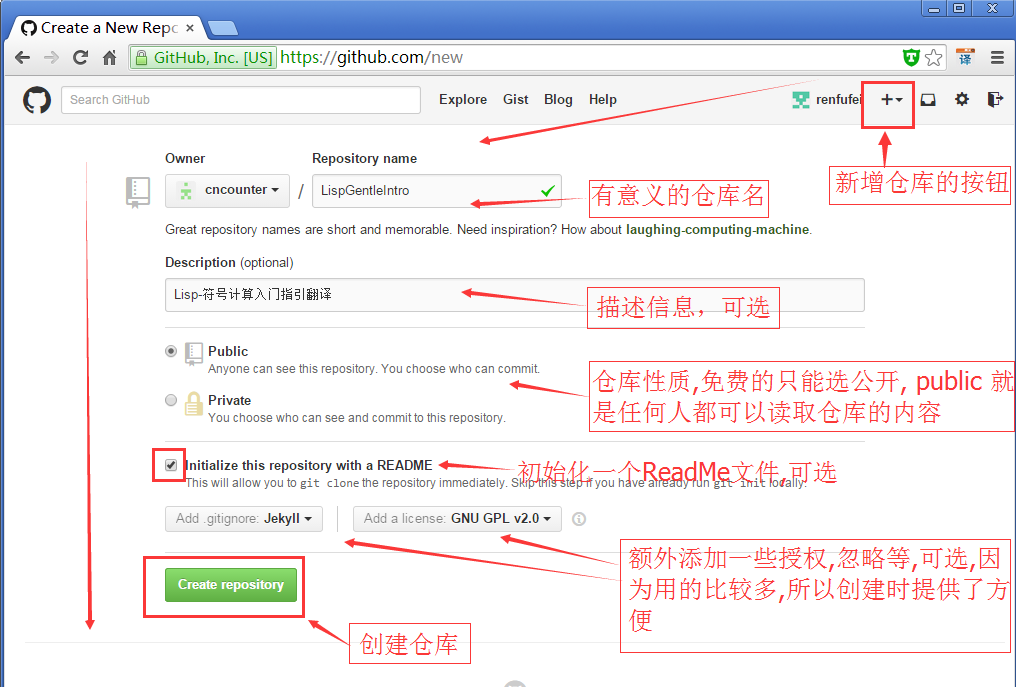
先看看SVN的操作吧, 最常见的是 检出(Check out ...), 更新 (Update ...), 以及 提交(Commit ...); 相当于有一个服务器端, 可以多个客户端进行访问和操作.

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.

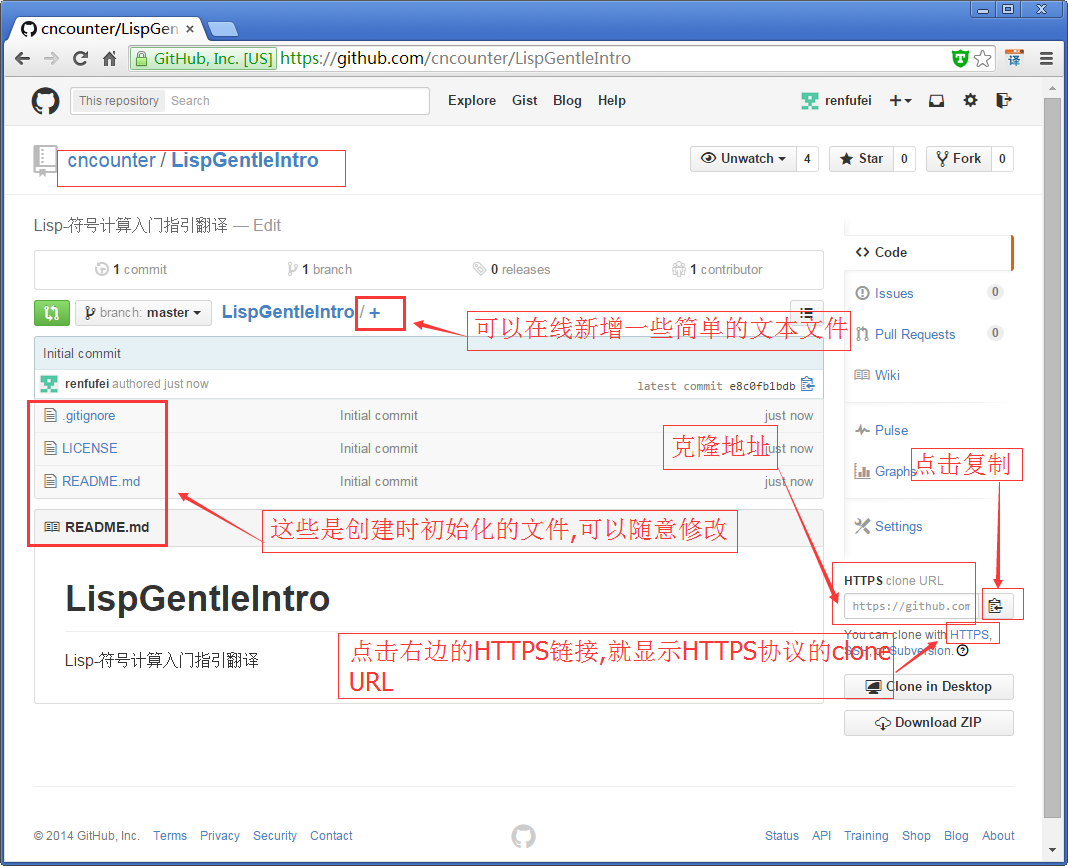
要创建仓库,必须得先**注册账号**. 请参考: [创建GitHub技术博客全攻略: http://blog.csdn.net/renfufei/article/details/37725057](http://blog.csdn.net/renfufei/article/details/37725057)

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 ...)

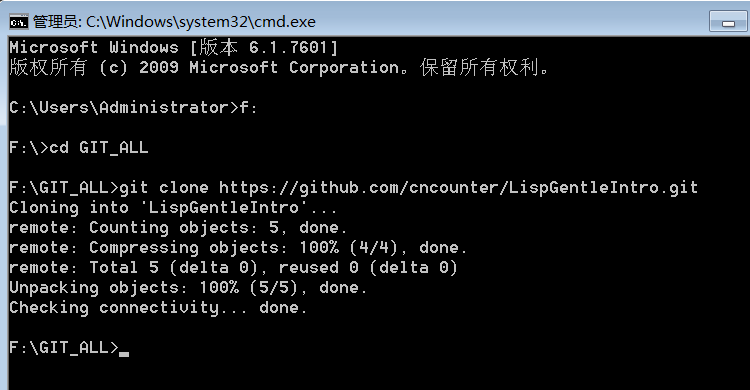
Win+R 快捷键打开运行对话框,输入 cmd 回车, 在 cmd 中(其实在 GitBash中也可以), cd 切换到存放git项目的目录, 如:

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 ...)

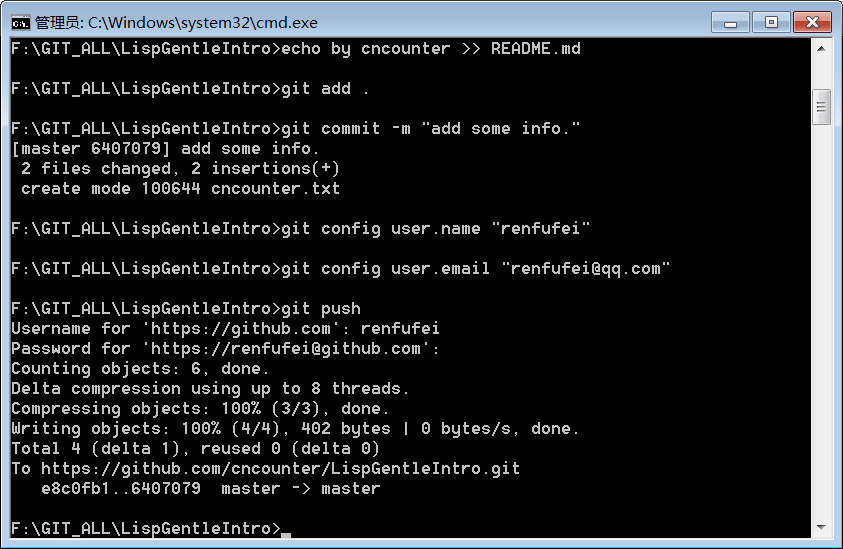
如果完成到一定程度,那么可以推送到远端在线仓库. 推送之前,请确保你已经设置了全局的 user.name 和 user.email, 如果没有设置,或者不想使用全局的信息,可以在当前项目下设置:

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.

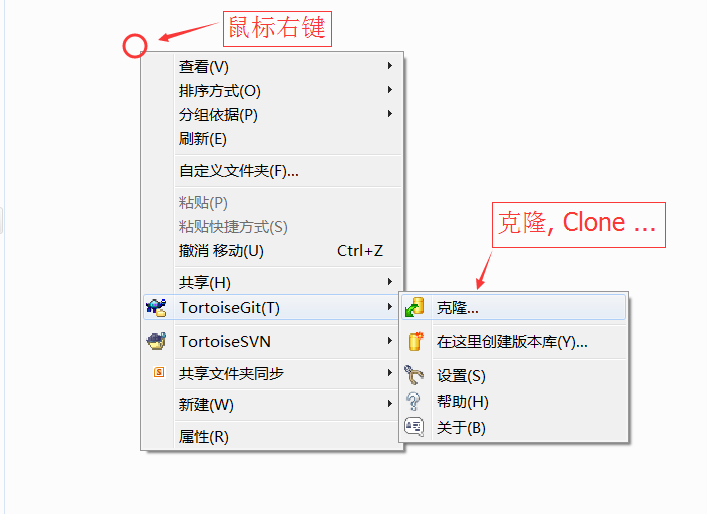
3. TortoiseGit使用与操作

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!!!)

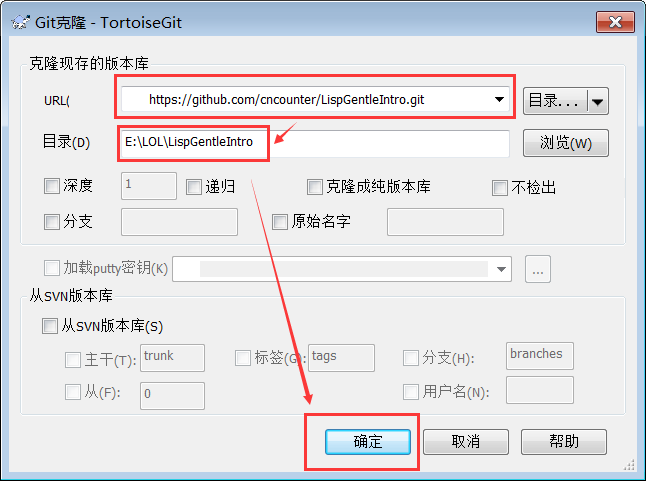
关于 TortoiseGit 的安装请参考 [前一小节](http://blog.csdn.net/renfufei/article/details/41647937).

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)

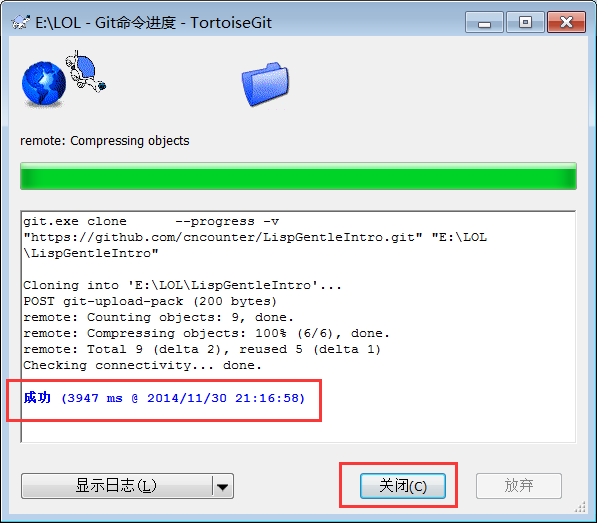
弹出克隆项目对话框:

[](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)

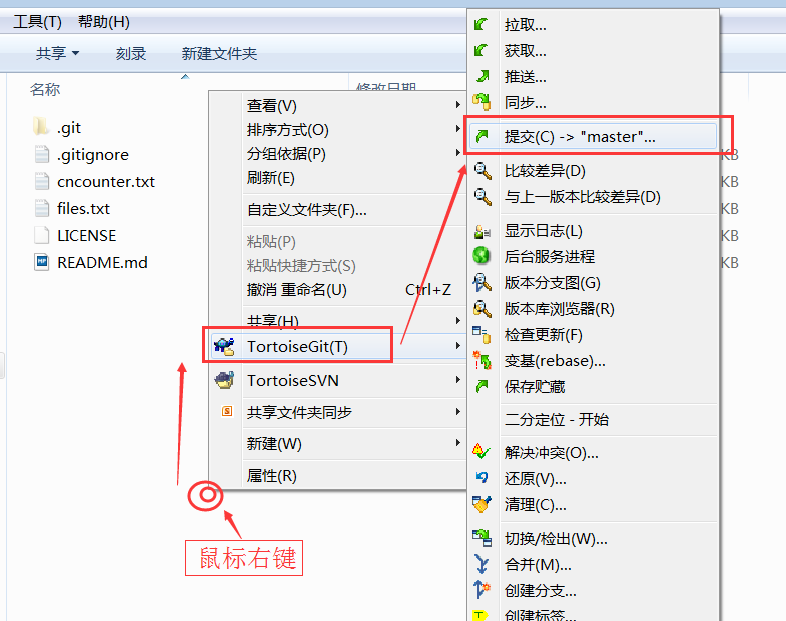
3.2 修改提交项目

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:

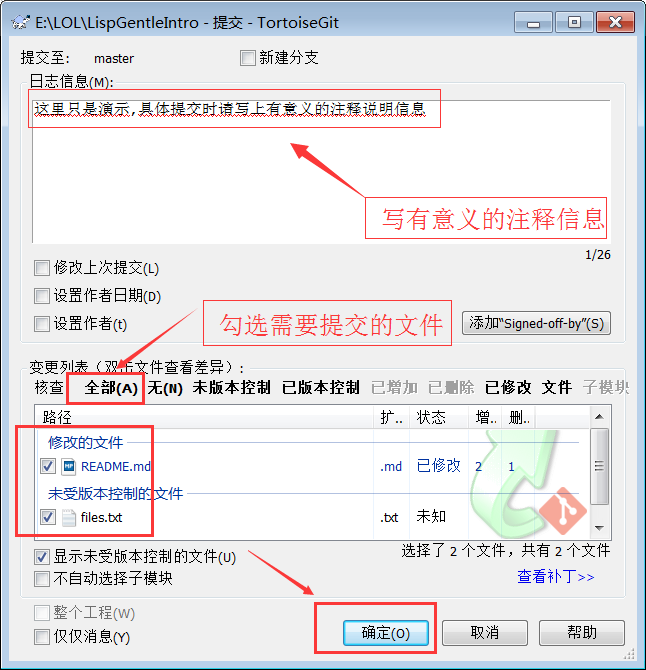
LispGentleIntro =============== Lisp-符号计算入门指引翻译 <a href="http://cncounter.duapp.com/">天朝计数器</a>

还可以试试增加一些文件. 如 files.txt .

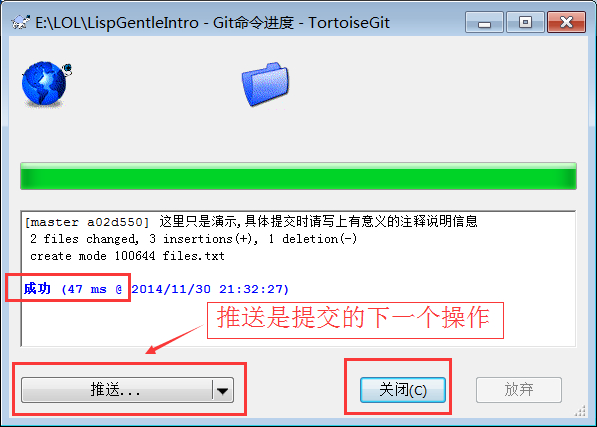
然后在本地项目的空白处点击鼠标右键, 选择 **TortoiseGit --> 提交(C) -> "master"...** 或: Commit -> "master"....

[](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)

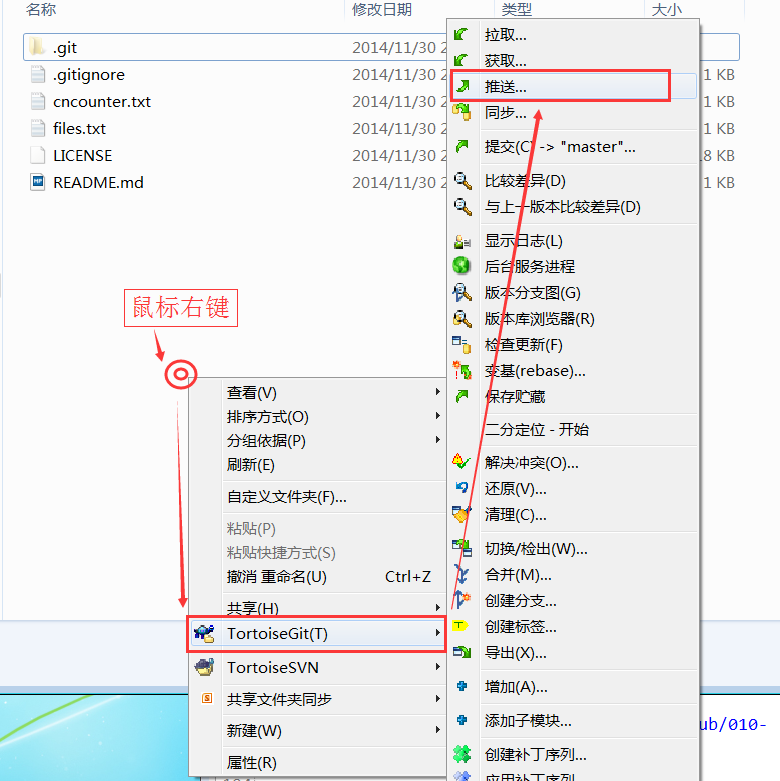
作为好习惯,填写提交日志,勾选需要提交的文件,然后点击 "确定" 按钮, 即提交到本地仓库.

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

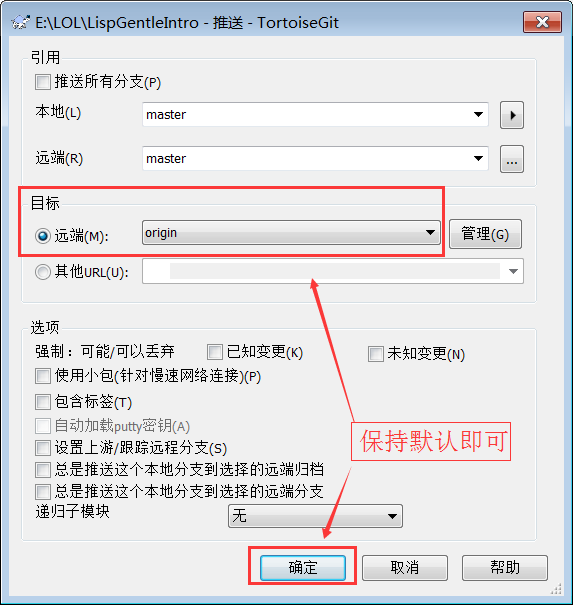
3.3 将提交到本地的项目推送到在线仓库

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....

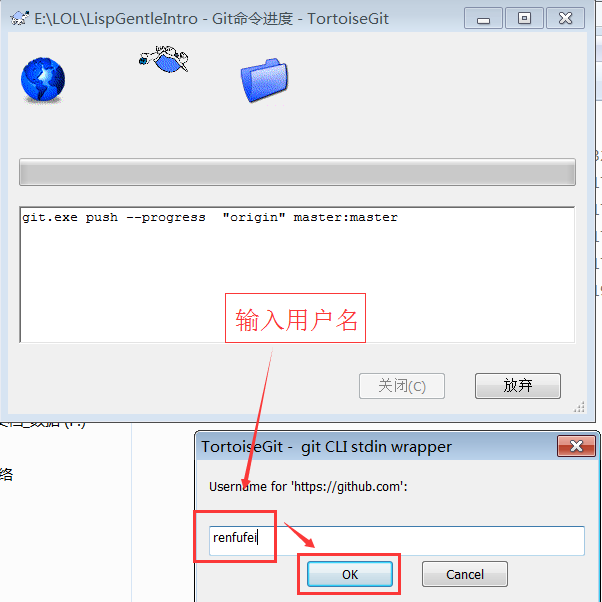
[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/407_tgit_rclick_push.png)

弹出推送(push)对话框:

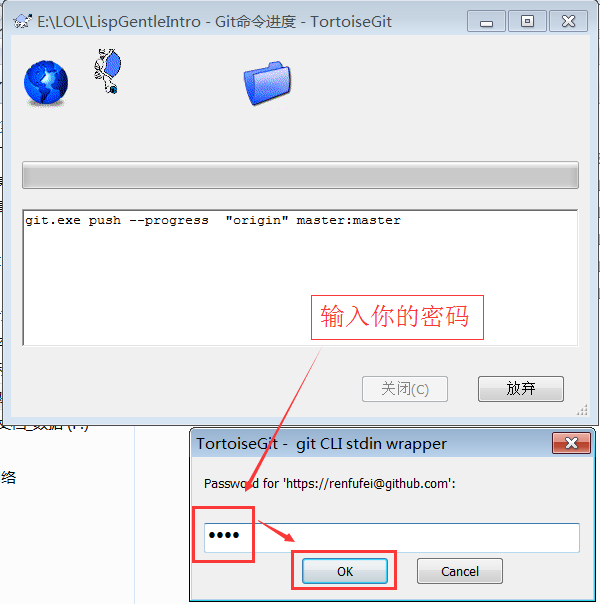
[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/408_tgit_push.png)

一般保持默认,点击 “**确定**” 按钮.

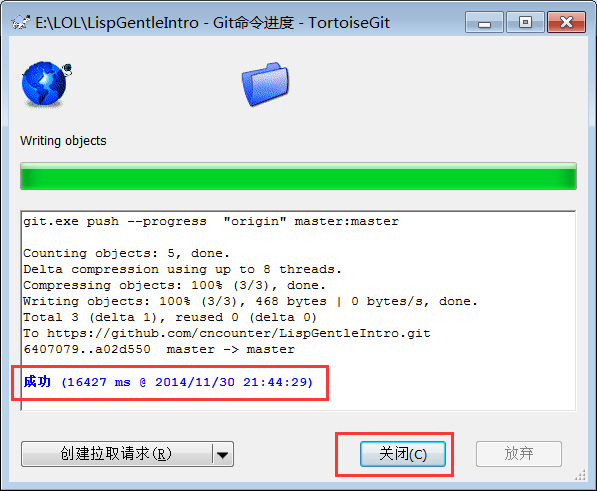
然后弹出推送进度界面, 可能要求你输入用户名:

[](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:

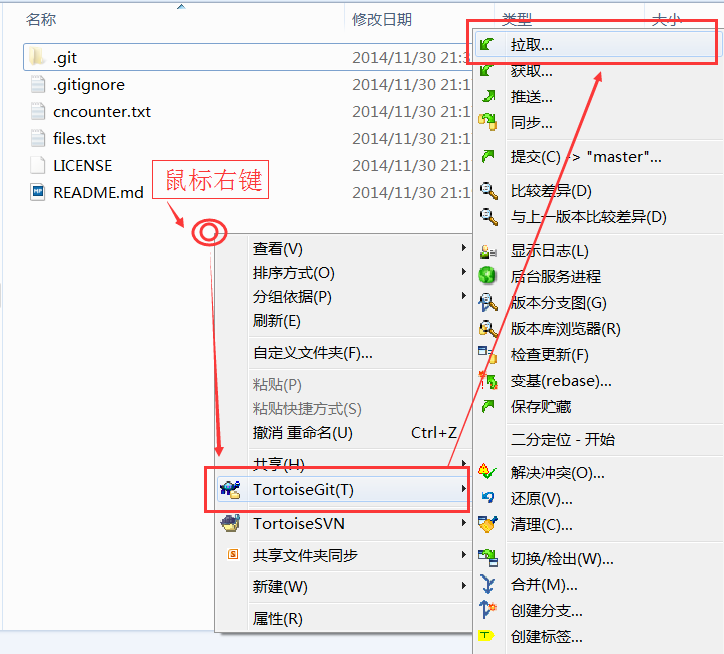
[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/411_pushok.png)

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.

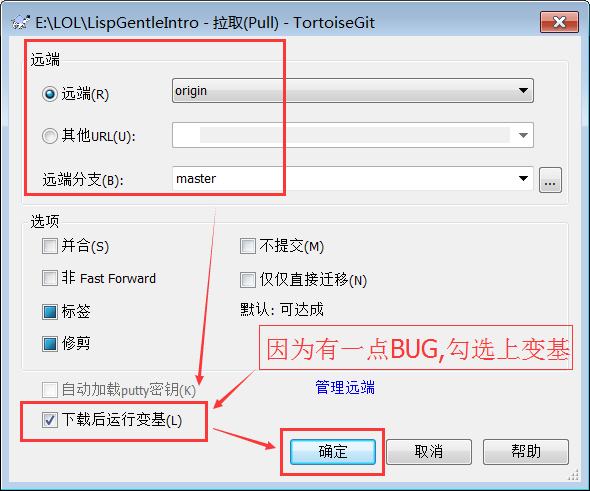
3.4 拉取项目(pull ...)

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 ...).

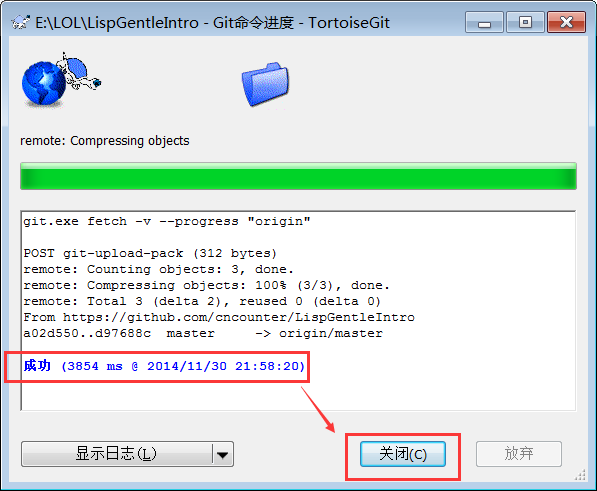
在本地项目的空白处点击鼠标右键, 选择 **TortoiseGit --> 拉取...** 或: Pull....

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

弹出拉取(pull)对话框:

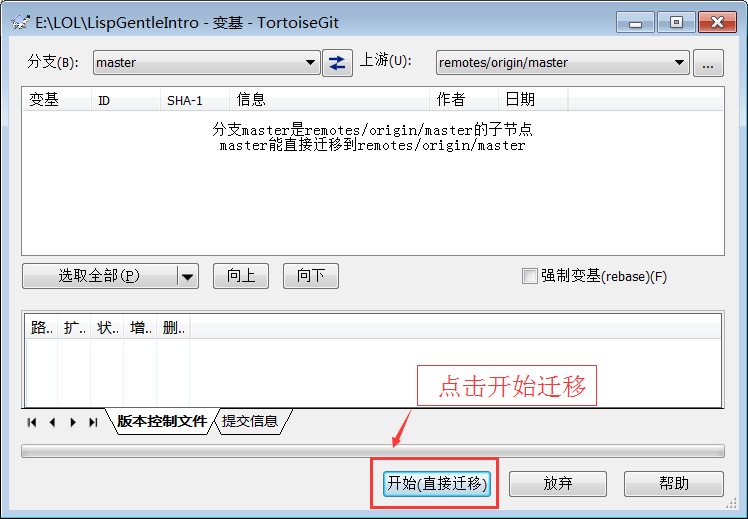
[](https://github.com/cncounter/translation/blob/master/tiemao_2014/GitHelp/413_pull_dialog.png)

【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)

然后执行变基:

【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).

4. 更复杂的操作

当然,多人协作过程中,避免不了会有各种意外情况需要处理,比如冲突,合并,变基等等, 关于这些复杂的操作,请参考: [GotGitHub系列: http://www.worldhello.net/gotgithub/01-explore-github/010-what-is-github.html](http://www.worldhello.net/gotgithub/01-explore-github/010-what-is-github.html)

5. 相关文章

1. [**目录**](http://blog.csdn.net/renfufei/article/details/41647773)
2. [**安装及配置Git**](http://blog.csdn.net/renfufei/article/details/41647875)
3. [**安装及配置TortoiseGit**](http://blog.csdn.net/renfufei/article/details/41647937)
4. [**Git与TortoiseGit基本操作**](http://blog.csdn.net/renfufei/article/details/41647973)
5. [**MarkDown示例**](http://blog.csdn.net/renfufei/article/details/41648021)
6. [**解决 TortoiseGit 诡异的 Bad file number 问题**](http://blog.csdn.net/renfufei/article/details/41648061)
7. [**加入QQ群GitHub家园: 225932282**](http://jq.qq.com/?_wv=1027&k=WHbwkD)