聊一聊TortoiseGit

学习TortoiseGit之前，我们需要先明白一个概念

**版本控制！**

什么是版本控制

版本控制（Revision control）也就是版本迭代。

* 实现跨区域多人协同开发
* 追踪和记载一个或者多个文件的历史记录
* 组织和保护你的源代码和文档
* 统计工作量
* 并行开发、提高开发效率
* 跟踪记录整个软件的开发过程
* 减轻开发人员的负担，节省时间，同时降低人为错误

简单说就是用于管理多人协同开发项目的技术。

无论是工作还是学习，或者是自己做笔记，都经历过这样一个阶段！我们就迫切需要一个版本控制工具！

常见的版本控制工具

我们学习的东西，一定是当下最流行的！

主流的版本控制器有如下这些：

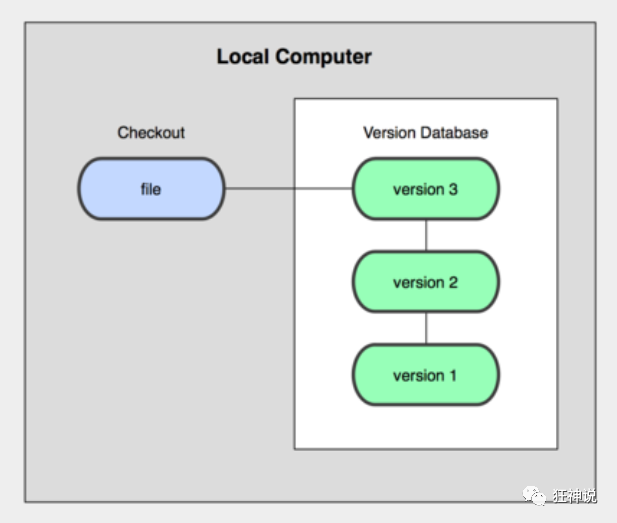
* **Git**
* **SVN**（Subversion）
* **CVS**（Concurrent Versions System）
* **VSS**（Micorosoft Visual SourceSafe）
* **TFS**（Team Foundation Server）
* Visual Studio Online

版本控制产品非常的多，现在影响力最大且使用最广泛的是Git与SVN

版本控制分类

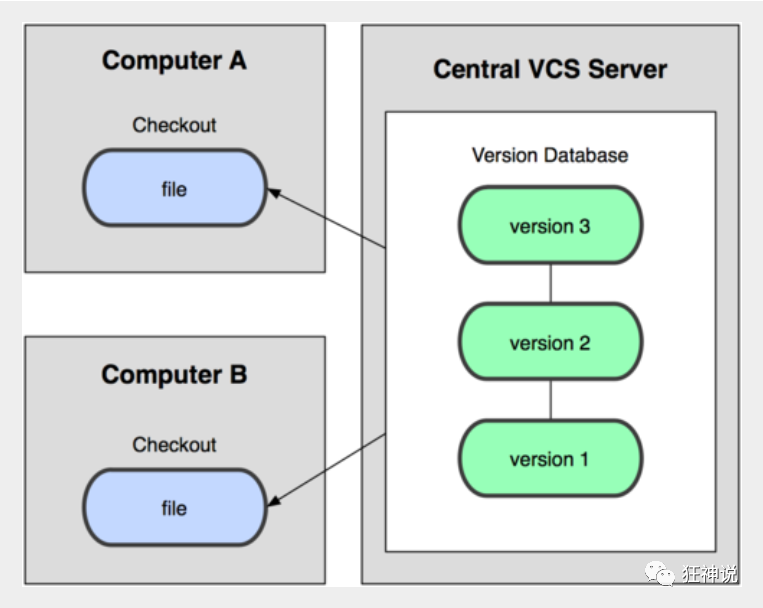
**1、本地版本控制**

记录文件每次的更新，可以对每个版本记录补丁文件（比如通过日期进行分割），适合个人用，如RCS。

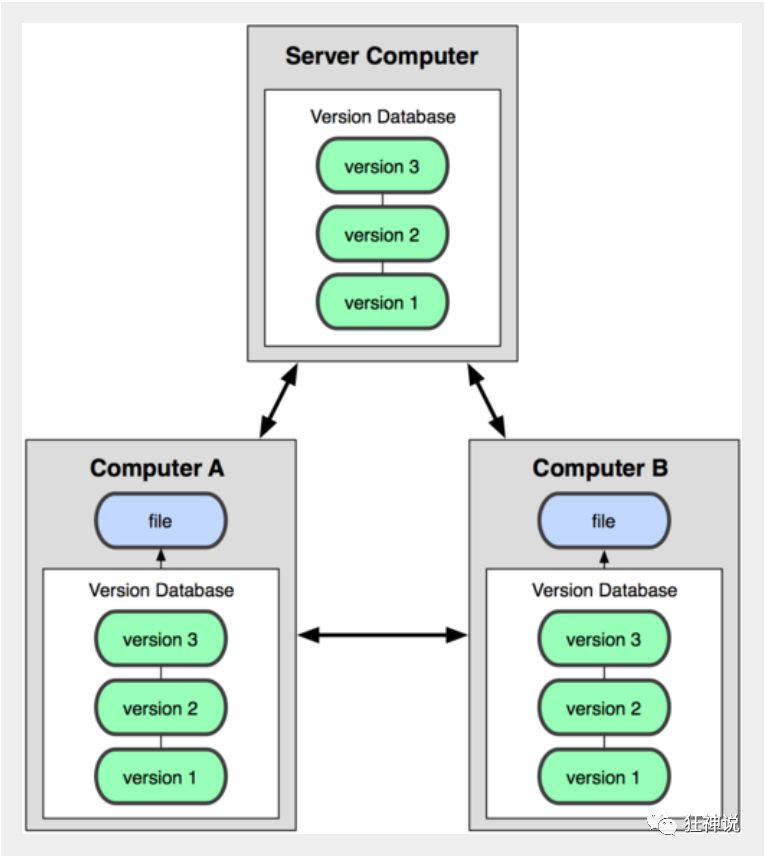


**2、集中版本控制  SVN**

所有的版本数据都保存在服务器上，协同开发者从服务器上同步更新或上传自己的修改（如果断网就无法同步历史数据）

**3、分布式版本控制 Git**

每个人都拥有全部的代码！安全隐患！



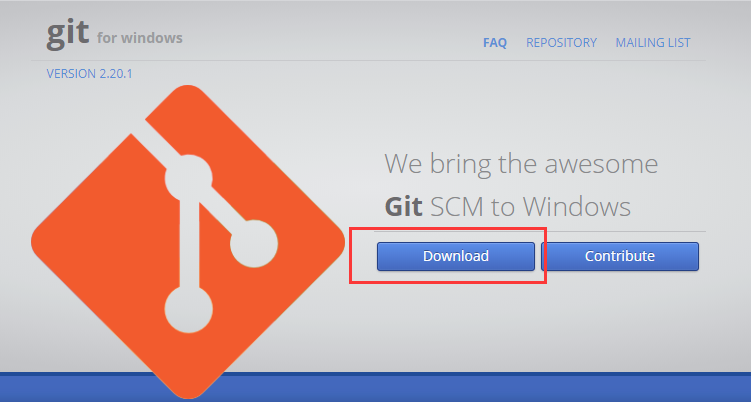
所有版本信息仓库全部同步到本地的每个用户，这样就可以在本地查看所有版本历史（可以离线提交）

Git与TortoiseGit的关系

**其实TorToiseGit就是Git的一个Gui包装**， Git作为一个复杂的版本控制系统，命令之多，即使经常使用，一些命令也记不住，一般只记住几个常用的命令，不是所有使用Git命令行都是高效的。要借助一个图形化的软件，TortoiseGit来操作git。用GUI（图形界面）再谈CLI（命令行），我相信这会更容易让人接受。

下载Git

第1步：下载Git，下载地址：<https://git-for-windows.github.io/>，一般官网下载慢，所以采用镜向下载http://npm.taobao.org/mirrors/git-for-windows/



**第2步：**下载安装Tortoisegit

<https://tortoisegit.org/download/>

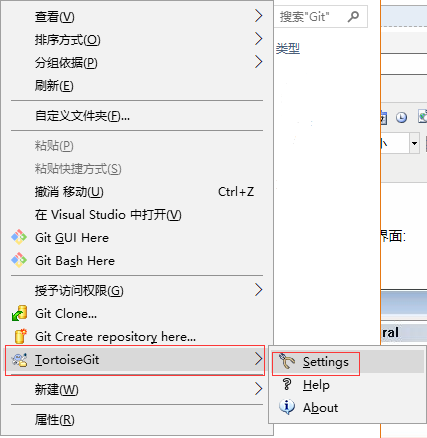


**Tortoisegit 配置**

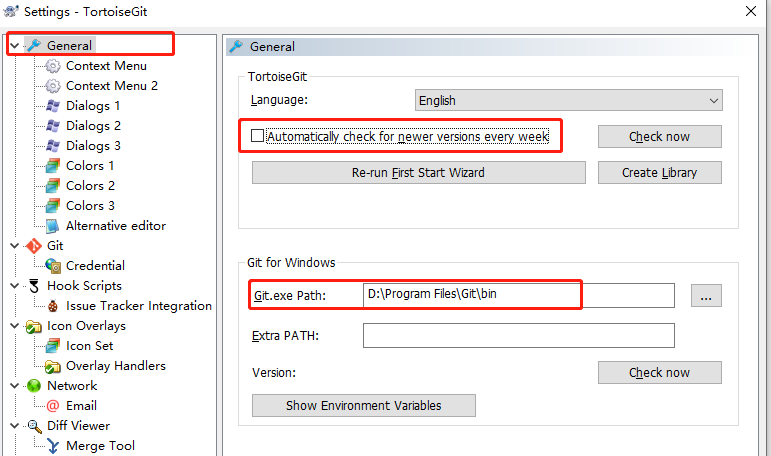
**第1步：**首先，请选定一个存放Git项目的目录，这样方便管理. 如: D:\Git，然后在资源管理器中打开：



**第2步：**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：

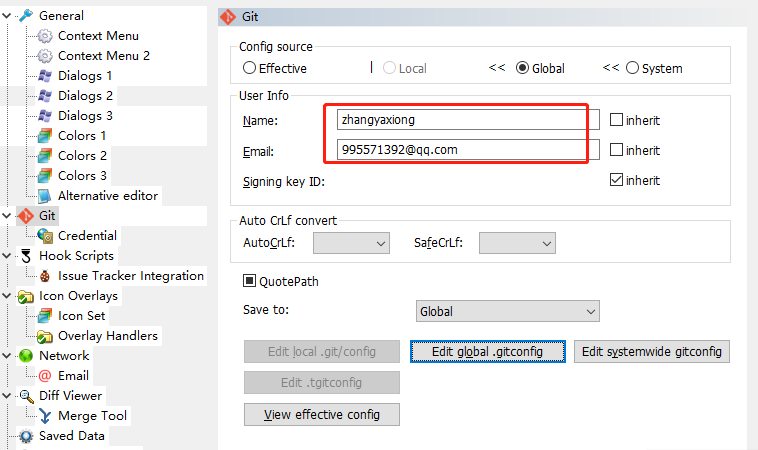


**第3步：**选中General，在右边的 Language中选择简体中文(当然,你也可以继续使用英文)。取消勾选升级检查的复选框，可能还需要指定 git.exe 文件的路径，如 "D:\Program Files\Git\bin"。完成后，点击应用，确定关闭对话框:



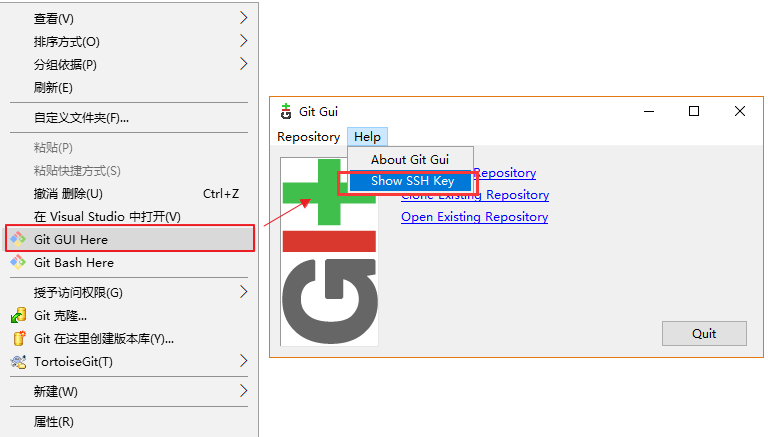
**第4步：**配置用户，用户作为你操作git的个人标识，进入设置，点选左边的Git标签，可以发现,右边可以配置用户的名字与Email信息(可以通过命令去添加相应的信息，比如记住密码等credential.helper=store

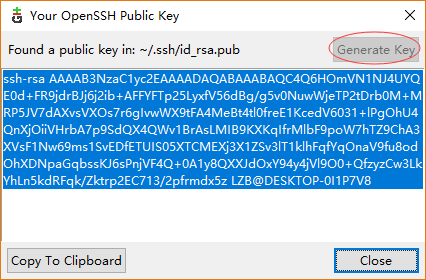
). 如下图所示：



**Tortoisegit 使用示例**

**第1步：**通过**SSH URL方式传输**，使用之前需要先将本地的 SSH 公钥配置到 Git 服务器上。

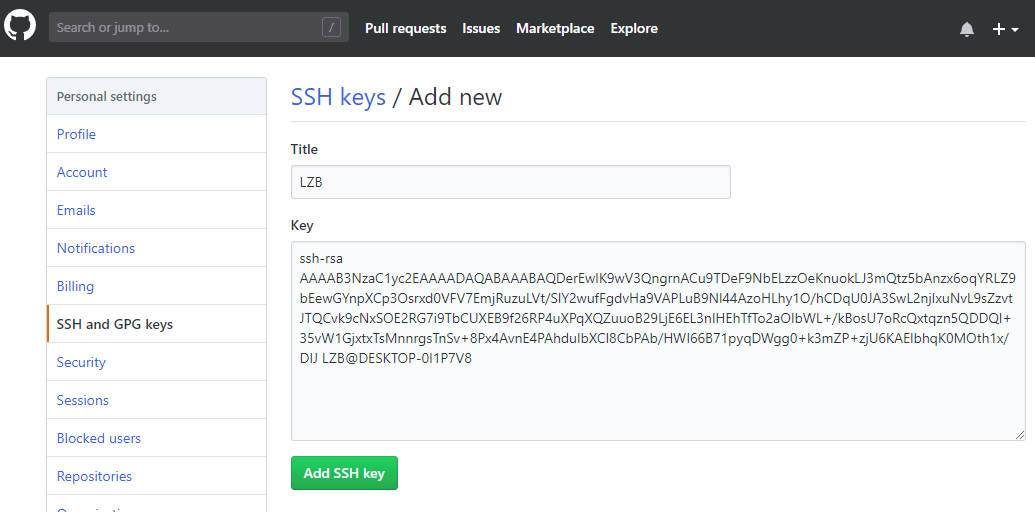
* 需要创建一个SSH Key，在任意文件夹下点击右键，选择 Git GUI Here。在弹出的程序中选择主菜单的【Help】→【Show SSH Key】：
* 如果没有Key，则点击“Generate Key”生成一个SSH Key：



把生成的全部内容拷出来。

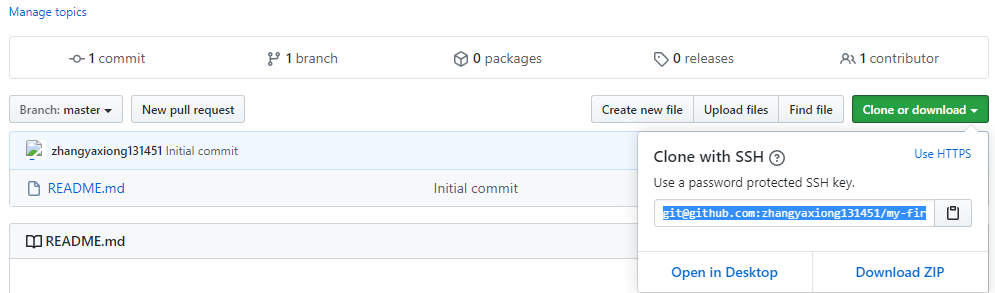
**第2步：**把 SSH Key 填到Git 服务器的配置中(zhangyaxiong131451/14789632@zyx)。

**GitHub服务器：**登陆GitHub，【头像】→【Settings】→【SSH and GPG Keys】→【New SSH Key】，填入上一步的复制的SSHKey，然后【Add SSH Key】：

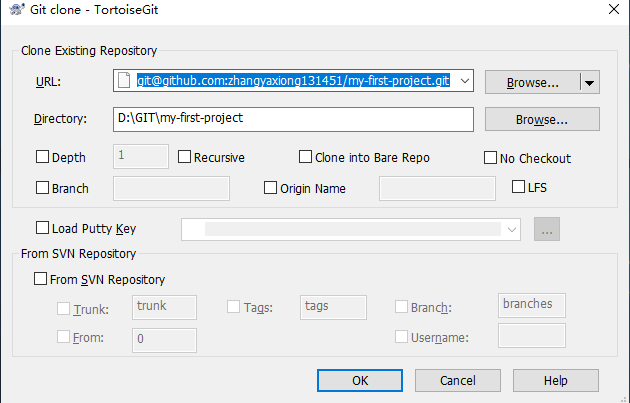


**第3步**：要克隆 GitHub 或 Gitblit 上的项目，先要获取对应远程存储库的路径（注意：此处使用 SSH URL 方式）：

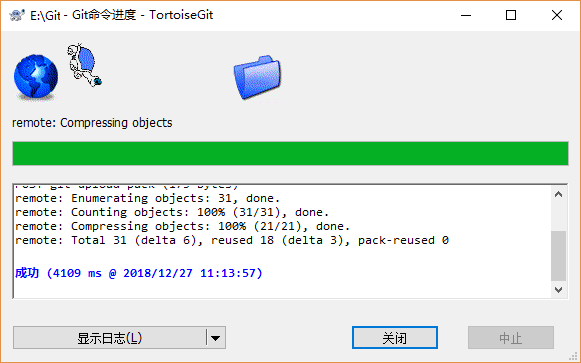
　　GitHub仓库：从 GitHub项目页面上得到对应存储库的 giturl 路径：



**第4步：**在本地文件夹的空白位置处，右击鼠标，在菜单中选择【Git克隆】：



　　把git 路径填写到URL，并选择本地文件目录，点击【确定】，弹出下载进度窗口。（服务器端至少要有一个文件，否则 pull的时候会报错 fatal: Couldn't find remote ref HEAD）



克隆成功，则在文件夹上就可以看到角标标识：



* Git文件上角标符号说明：

文件上的图标，可以反映出当前文件或者文件夹的状态：

　　1、正常的：绿色的对号 https://img2018.cnblogs.com/blog/1545277/201812/1545277-20181226130633376-1375659427.png

　　2、被修改过的：红色感叹号  https://img2018.cnblogs.com/blog/1545277/201812/1545277-20181226130659002-1901548592.png

　　3、新添加的：蓝色的加号https://img2018.cnblogs.com/blog/1545277/201812/1545277-20181226130708046-702059723.png

　　4、未受控的（无版本控制的）：蓝色的问号https://img2018.cnblogs.com/blog/1545277/201812/1545277-20181226130715848-1344413806.png

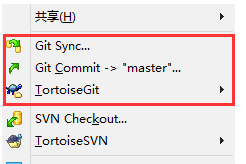
　　5、忽略不受控的：灰色的减号https://img2018.cnblogs.com/blog/1545277/201812/1545277-20181226130727649-1356151074.png

　　6、删除的：红色的x号 https://img2018.cnblogs.com/blog/1545277/201812/1545277-20181226130737919-811538319.png

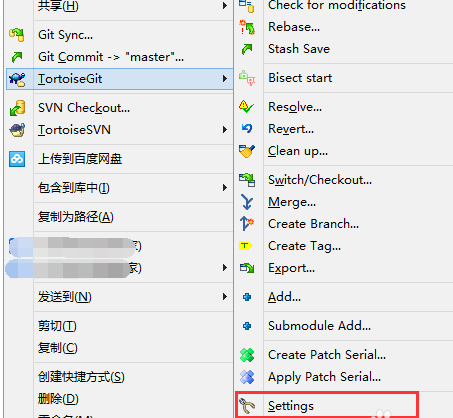
7、有冲突的：黄色的感叹号 

将TortoiseGit**管理右键菜单添加相应的git操作（比如pull，showlog等等）**

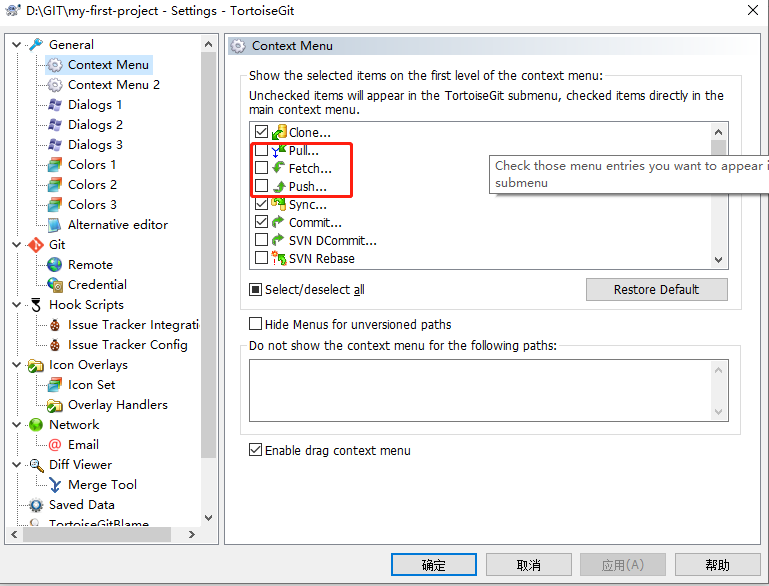
1点击TortoiseGit下载的代码或文件目录上右键鼠标，则会弹出如图所示的右键菜单列表，从右键菜单中可以看到默认Git只有三个功能菜单，很多常用的菜单都没有出现，需要点击TortoiseGit打开二级菜单才可以，下图：



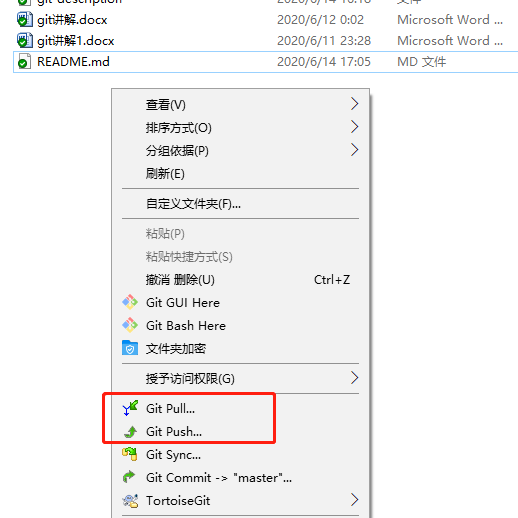
2 我们将pull、show log、push这些常用的git菜单从二级添加到一级，此处选择TortoiseGit然后在二级菜单中选项“Settings”选项



3 进入到设置对话框页面中，可以看到关于功能设置的都在这个页面，这里我们只需要看General》Context Menu



4勾选上“pull或者是push等”功能菜单，然后点击确定保存即可。，那么二级菜单就会被添加到一级菜单。或者把之前已经勾选的去掉，则原来在一级右键菜单会移动到二级菜单中了



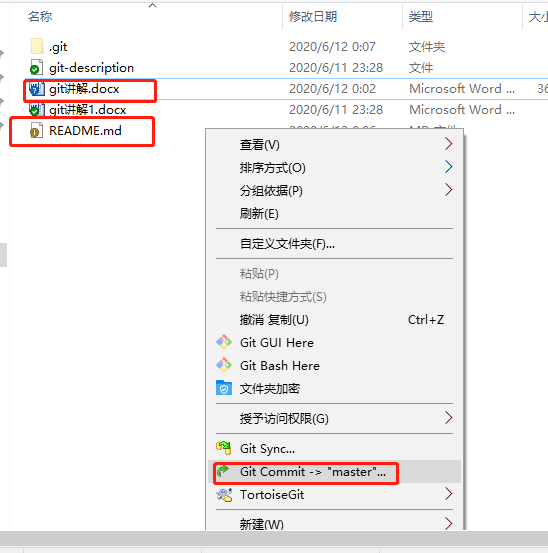
将代码提交到服务器

Git的使用惯例的流程：

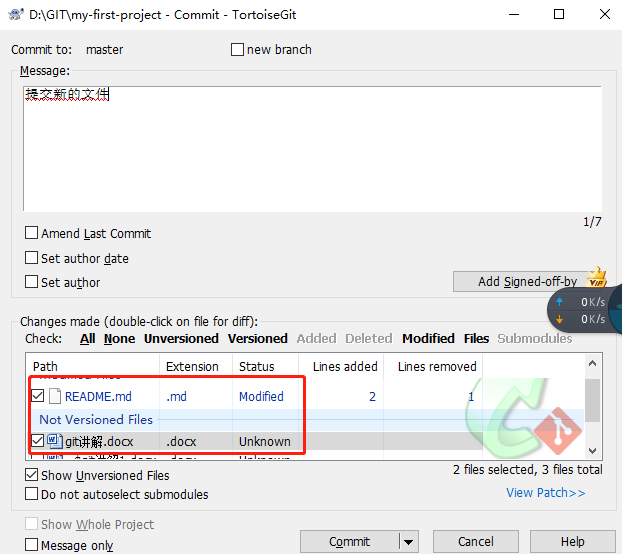
**https://img2018.cnblogs.com/blog/1545277/201812/1545277-20181227115208257-1683338978.png**

**第一步：**改动，修改本地项目中的某些文件，如修改 README.md 内容，还可以增加一些文件， 如git讲解.docx。

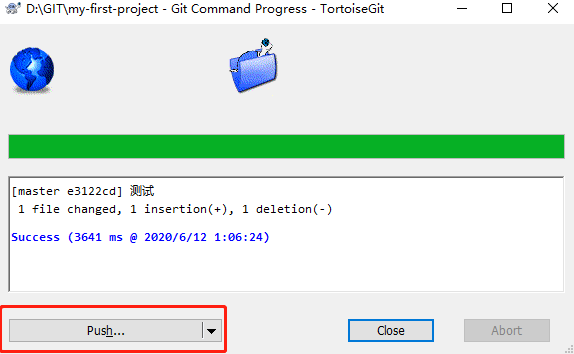
**第二步：**提交本地，在本地项目的空白处点击鼠标右键，选择 【Git Commit -> "master"...】



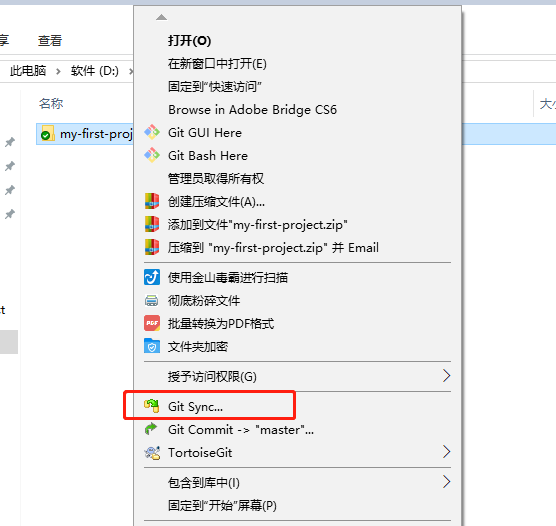
在弹出提交（Commit）对话框中完成提交说明信息，和选择需要提交的文件，可根据需要新建分支，然后点击 【Commit】 按钮，将修改提交到**本地仓库：**



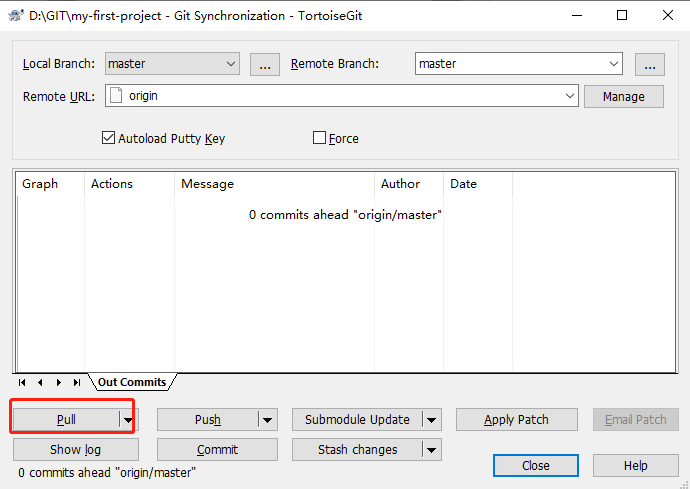
　　弹出提交进度窗口，提交成功后还需要“push”将本地仓库的修改推送到**远程仓库**。

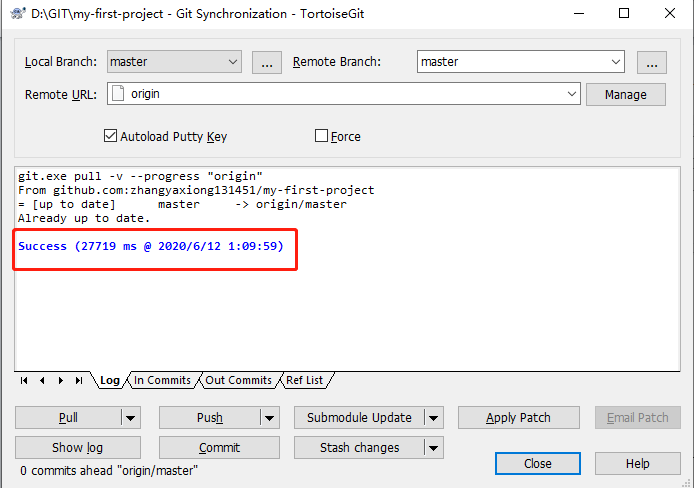
第三步：同步拉取，在实际工作中，如果多人协作或者多个客户端进行修改，那么我们还要拉取别人推送到在线仓库的内容，所以在推送之前需要先执行同步拉取(Pull ...)操作。

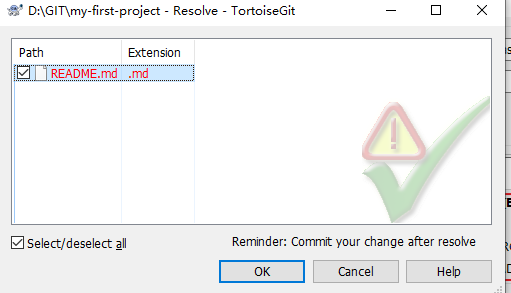
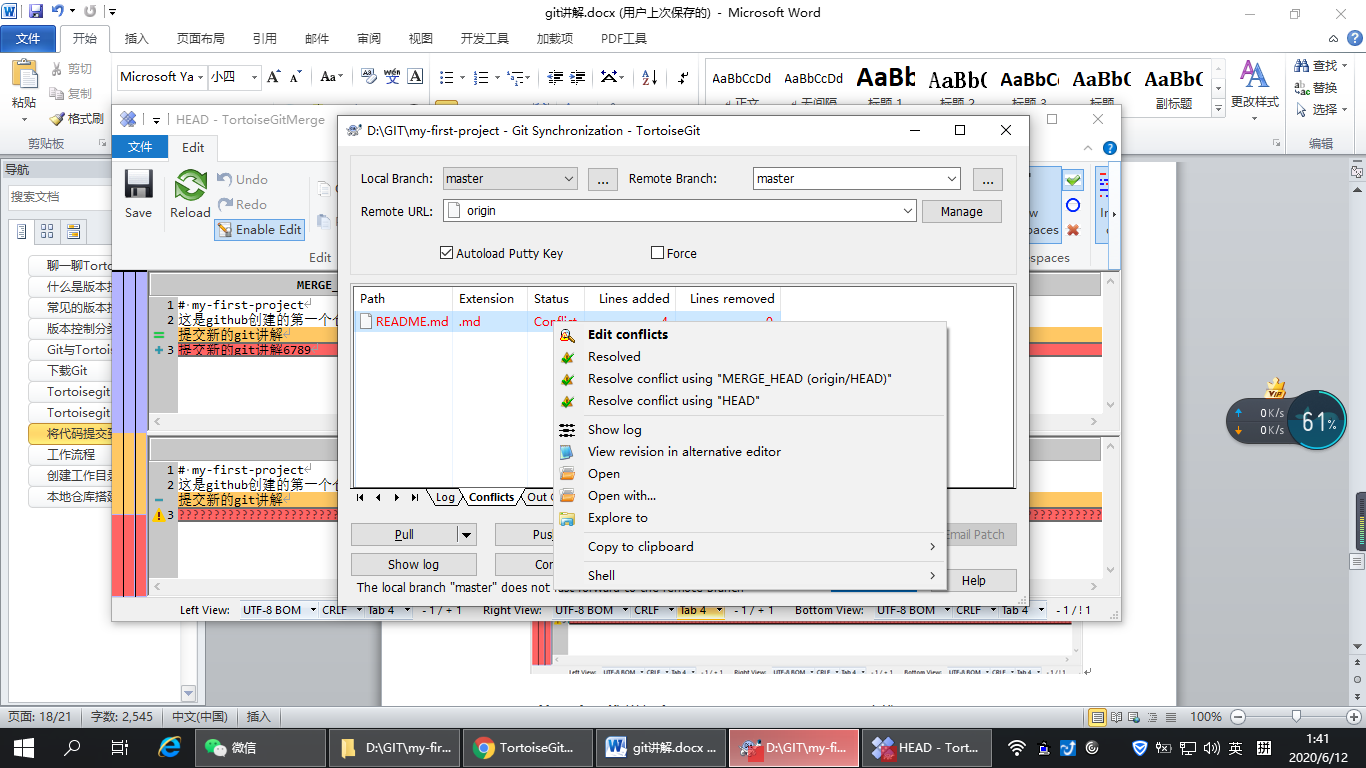
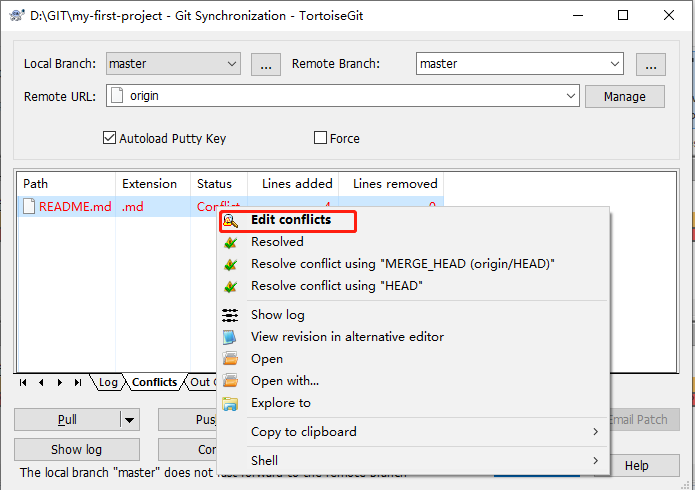
　　在本地仓库文件夹上【右击鼠标】→【Git Sync】：



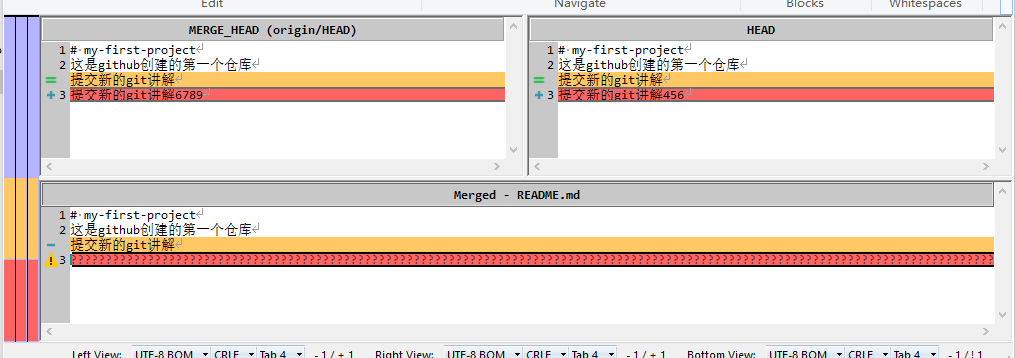
　　打开Git同步窗口（包括常规操作及日志，同右击菜单快捷操作一样），点击【拉取(P)】，将远程分支拉取到本地：



　　如果服务器上的文件没有被修改过，就会直接提示已经更新到最新，那你就可以直接进行下一步“push”操作了：　　反之，如果服务器上的文件被修改过了（本地文件修改前不是最新版本），就会提示冲突。先要解决冲突，然后再提交结果：

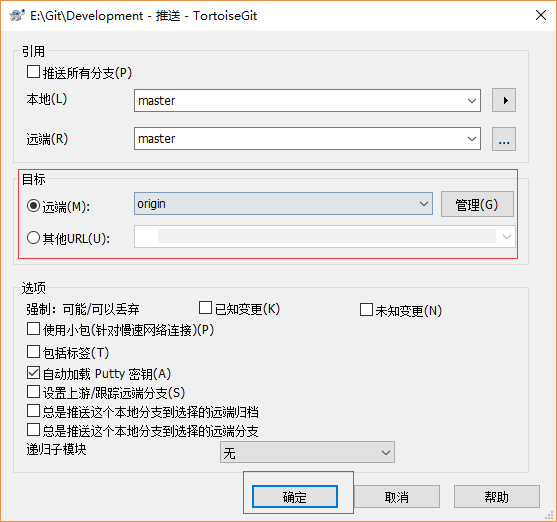


需要注意的是，和使用TFS、SVN的习惯一样，你在修改本地内容之前，最好先 拉取（pull）一下，减少冲突的可能。



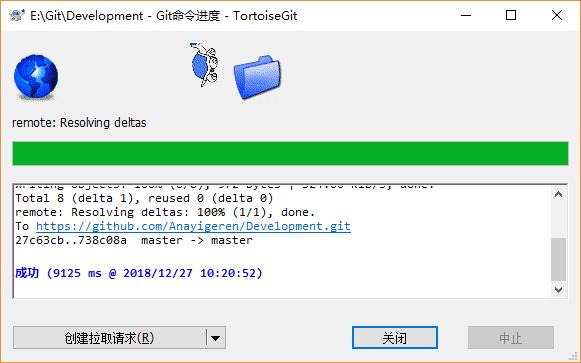
**第四步：推送远程**，将提交到本地仓库的修改推送到远程仓库，可以直接在提交成功后的提示窗口上点击【Push】，或者在Git同步窗口点击【push.】，鼠标右击的菜单上也有相应的快捷操作：选择 【TortoiseGit(T)】→【push】

　　弹出推送(push)对话框：



　　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，点击【确定】，然后要求输入密码，密码输入正确后，显示推送成功界面：



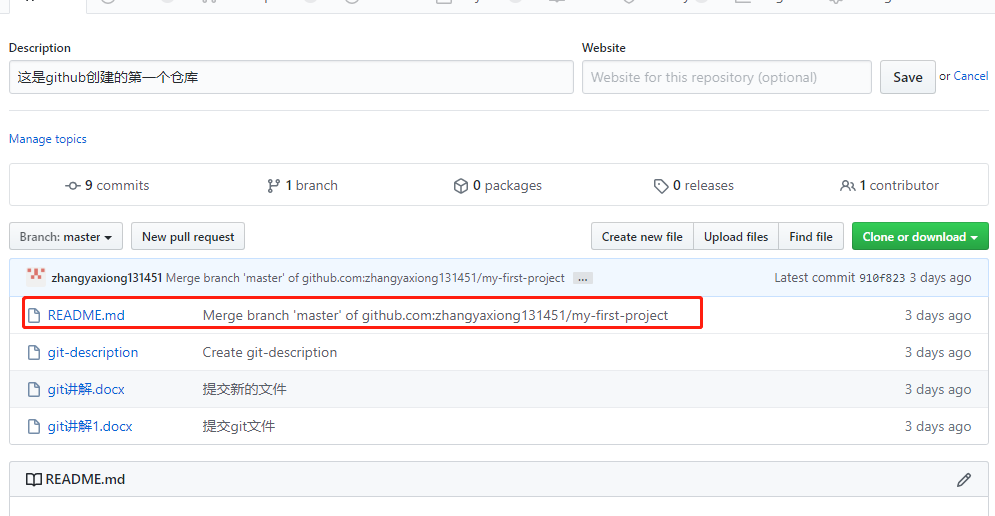
　　如果你按照上一小节**Tortoisegit 配置**的设置操作，则输入密码以后会记住密码。密码会明文保存在C:\Users\用户名\.git-credentials 这个文件中，请小心保存。

Git本地有三个工作区域：工作目录（Working Directory）、暂存区(Stage/Index)、资源库(Repository或Git Directory)。如果在加上远程的git仓库(Remote Directory)就可以分为四个工作区域。文件在这四个区域之间的转换关系如下：

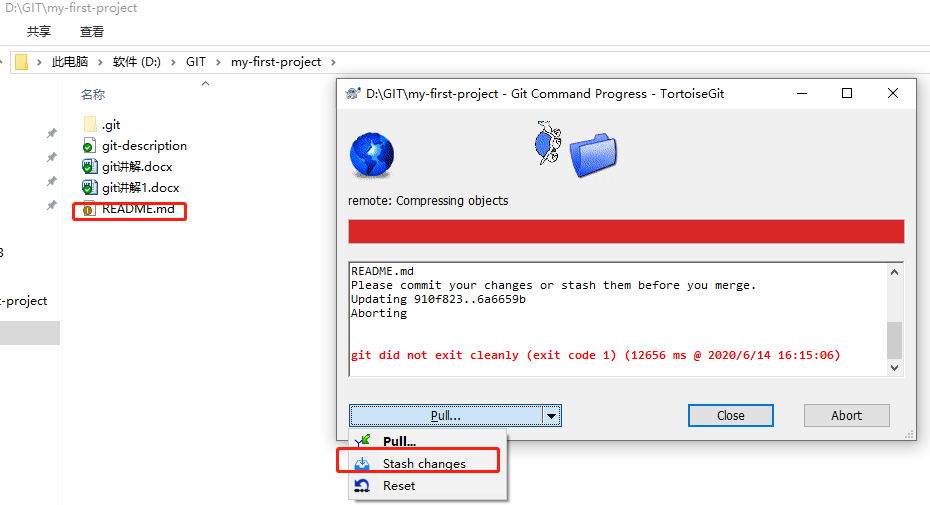
[TortoiseGit处理代码冲突](https://www.cnblogs.com/jason-beijing/p/5718190.html)

一般是两种场景：

场景1：服务器某个文件（如README.md）有他人修改，更新到了最新，

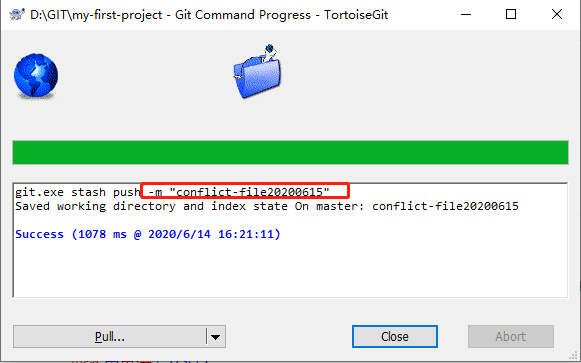


但是本地没有pull -> 写新代码 -> pull -> 提示有冲突



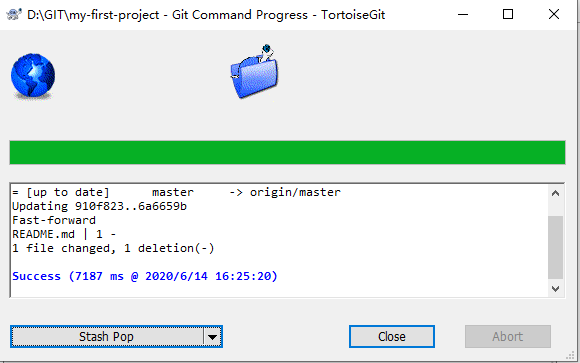
解决的办法：

1 点击stash changes(把自己的代码隐藏存起来)，并备注好填写的隐藏信息标示



2 重新pull

3 点击stash pop(把存起来的隐藏的代码取回来 )



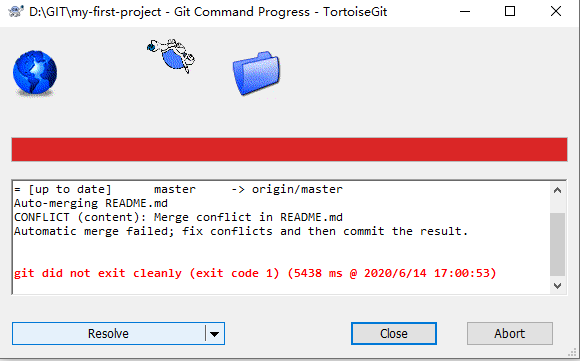
4 右键选择edit conficts，解决后点击编辑页面的 mark as resolved

5 commit&push

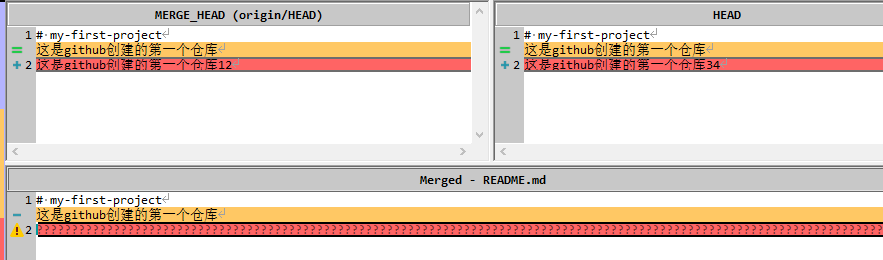
场景2： 没有pull，但是写新代码直接commit&pus，最后提示有冲突

解决办法：

1 更新pull，提示如下图所示：



2 点击Resolve，进入编辑冲突文件界面



3 根据自己的选择，保存或者更新相应的记录，然后标示为已解决冲突



4 commit&push