git使用指导

由于项目代码即将放在公司内部的gitlab上托管，根据最近使用经验，介绍一下在Windows上如何安装git客户端和使用gitlab。

介绍分2大部分来进行。之前有过使用SVN经验的同学可以直接查看1-4节关于TortoiseGit的使用说明，与SVN操作基本类似。如果首次接触版本控制和团队协作的工具可以直接从第五节开始学习使用命令行来进行git操作。

# 客户端介绍

要使用git，首先要在本地安装客户端。这里介绍两个客户端。

## Msysgit

一个包含命令行和简易图形界面的客户端。这里我们只需要用它为TortoiseGit提供命令行支持。

* 官方网站 <http://msysgit.github.io/>
* 下载地址 <https://github.com/msysgit/msysgit/releases/download/Git-1.9.4-preview20140815/Git-1.9.4-preview20140815.exe>

## TortoiseGit

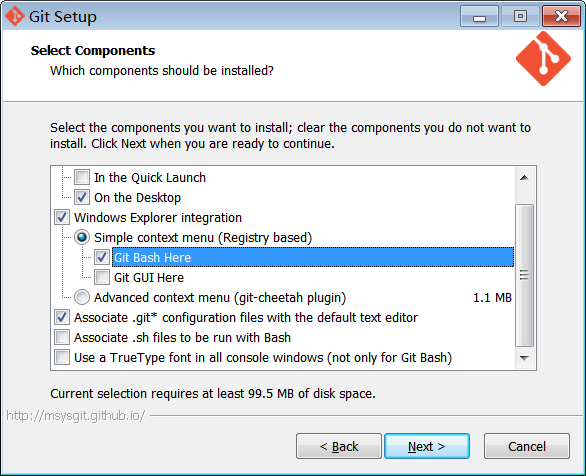
类似于TortoiseSVN的图形界面客户端。

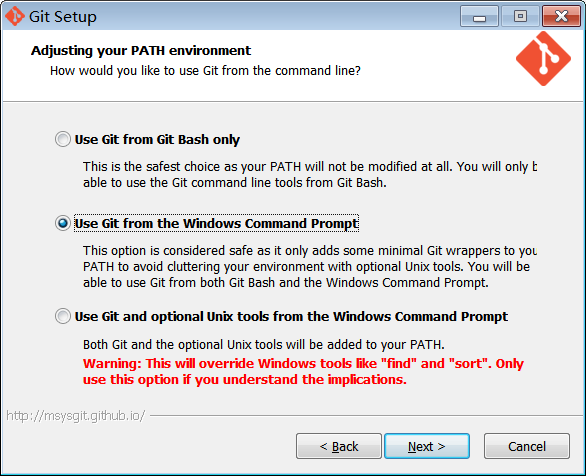
* 官方网站 <https://code.google.com/p/tortoisegit/>
* 下载地址 <https://code.google.com/p/tortoisegit/wiki/Download>

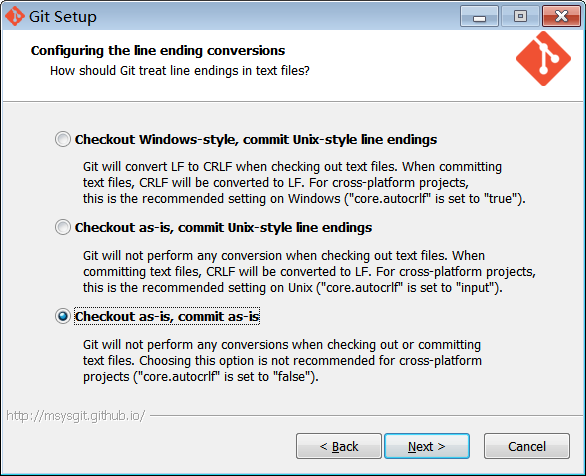
# 客户端安装

## Msysgit

Msysgit的安装过程，按照以下截图设置







## TortoiseGit

TortoiseGit的安装过程，默认设置就可以。

同时安装中文语言包，安装之后在设置中选择中文（开始🡪所有程序🡪TortoiseGit🡪Settings）。

# 客户端设置

由于github采用rsa加密认证，所以这里主要介绍在如何为客户端和自己的github账号配置认证信息。

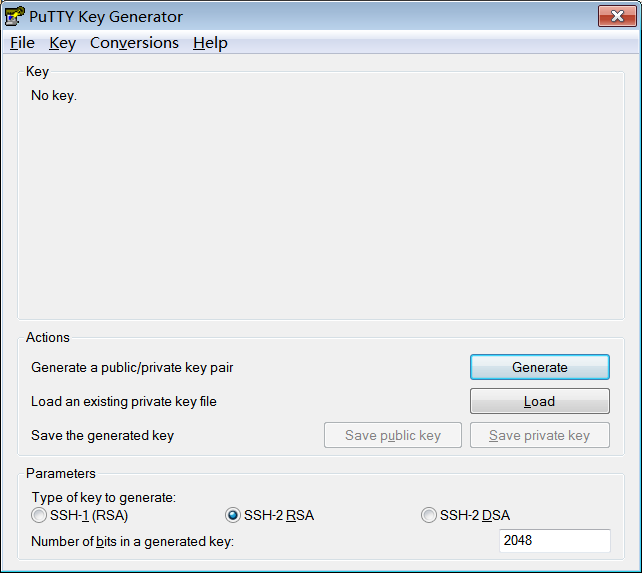
## Msysgit

由于我们只是用msysgit为TortoiseGit提供命令行支持，所以不需要在msysgit里设置认证信息。

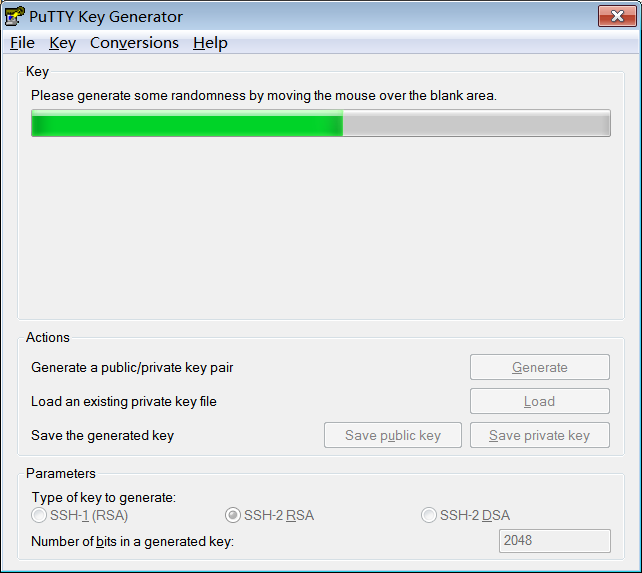
## TortoiseGit

由于TortoiseGit使用putty来通信，所以TortoiseGit使用的rsa key不是用ssh-keygen生成，而是用其自带的Puttygen工具生成。生成和设置rsa key的步骤如下。

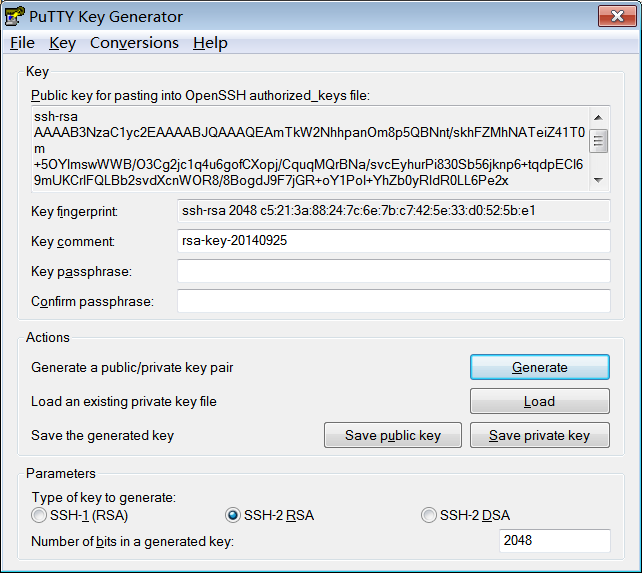
1. 运行TortoiseGit开始菜单中的Puttygen程序。



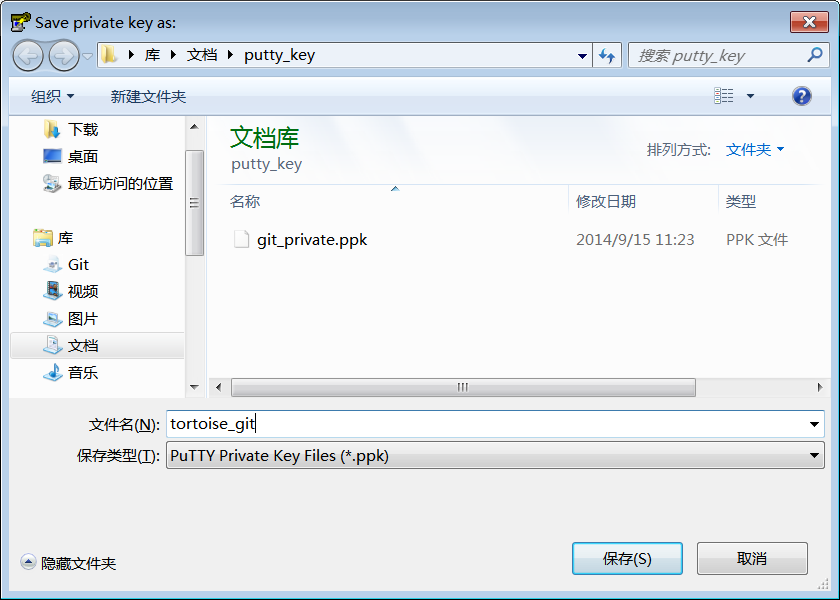
1. 点击[Generate]按钮，根据提示，随意移动鼠标，直到进度条走完，产生一个随机key。



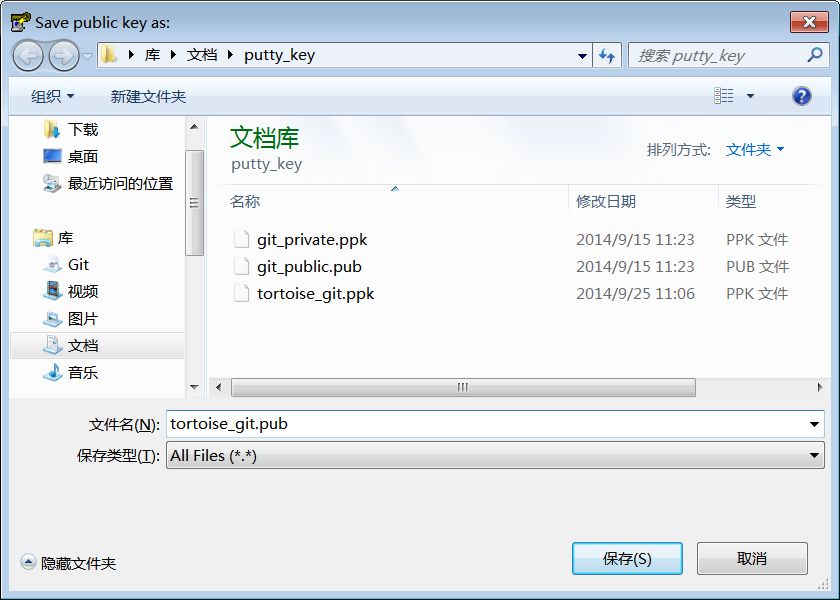
1. 在[Key passphrase]和[Confirm passphrase]里输入密码。



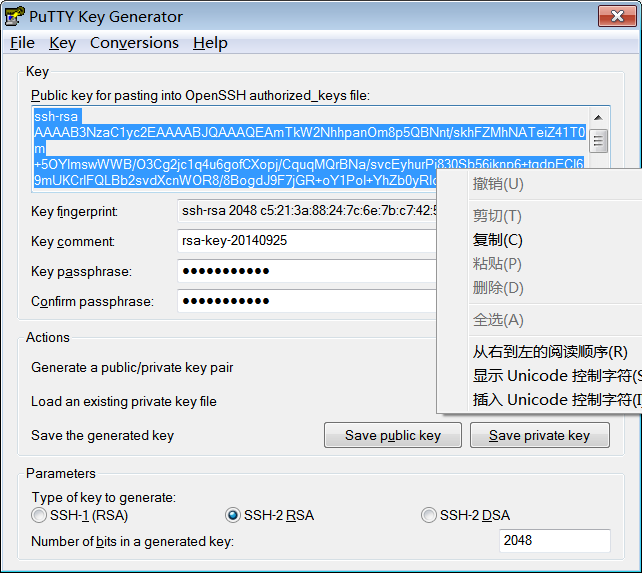
1. 然后点击[Save private key]按钮，把rsa私钥保存到ppk文件中。



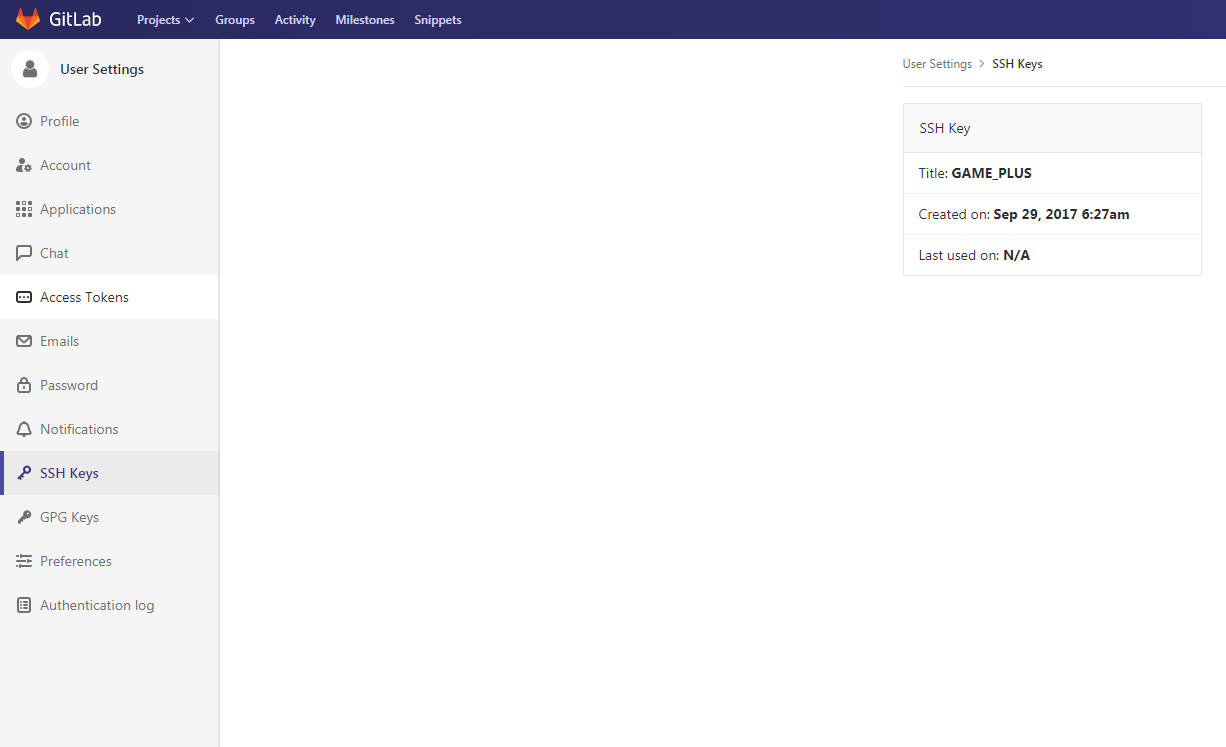
1. 点击[Save public key]按钮，把rsa公钥保存到pub文件中。

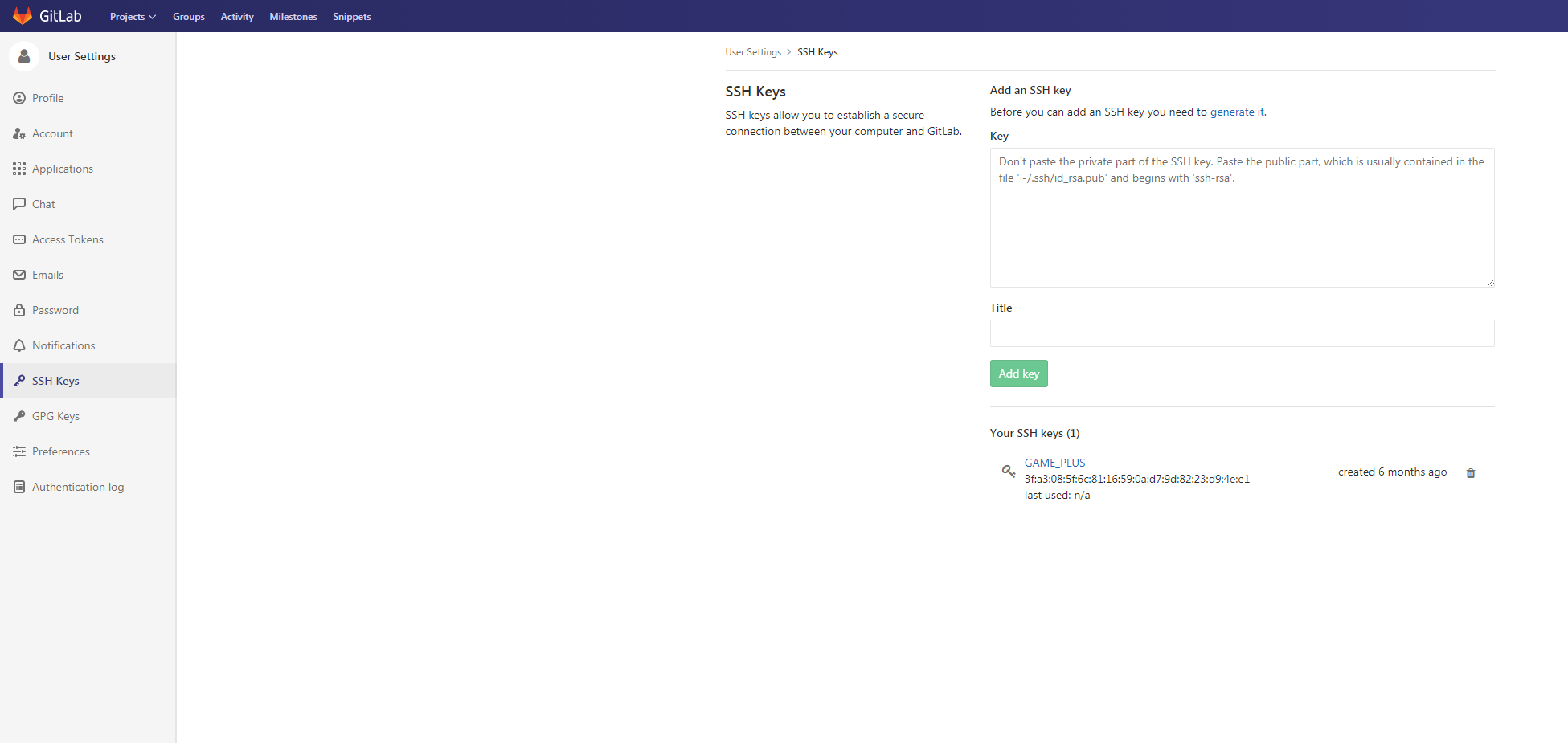


1. 复制下面整个蓝色高亮内容，即公钥。

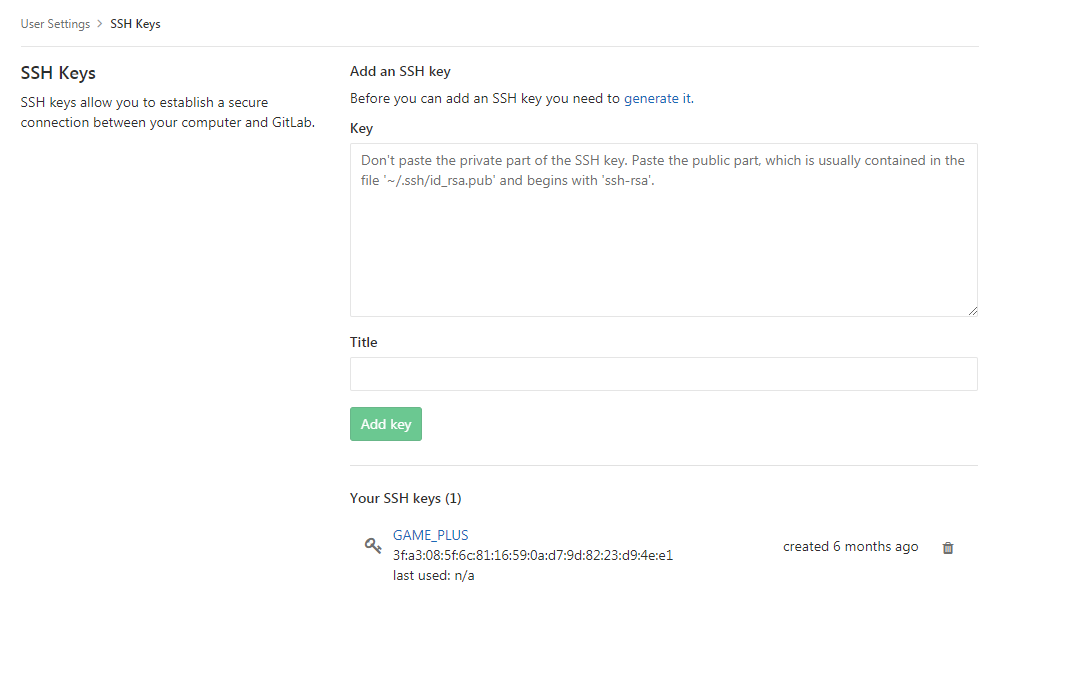


1. 打开gitlab，点右上角的[Settings]，然后点左边导航栏的[SSH keys]

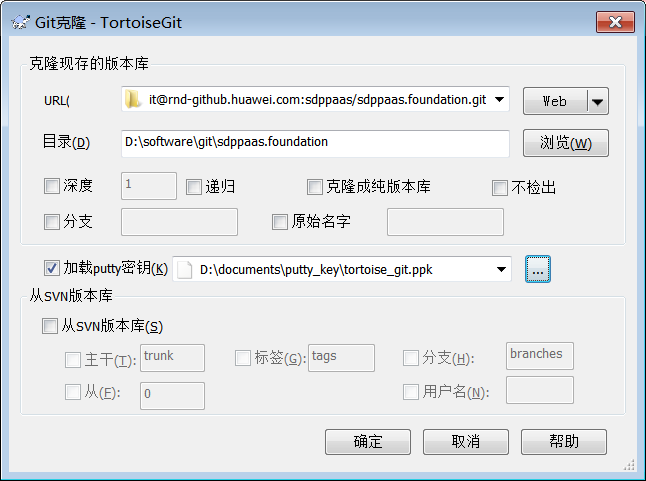


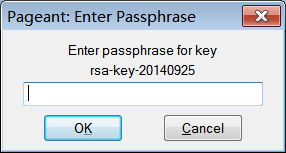


1. 点击[Add key]按钮，进入下方配置界面。为这个key随便起个名字填入Name，把刚才在Puttygen界面复制的key填入Public Key，点击[Save]按钮，key配置成功。完成后在SSH Public 界面下可以看到刚刚添加的公钥。



1. 使用TortoiseGit时，在桌面上任意位置右击，选择git克隆。选定[加载putty密钥]，并选择刚才生成的ppk文件。第一次使用会要求输入生成key时输入的passphrase。

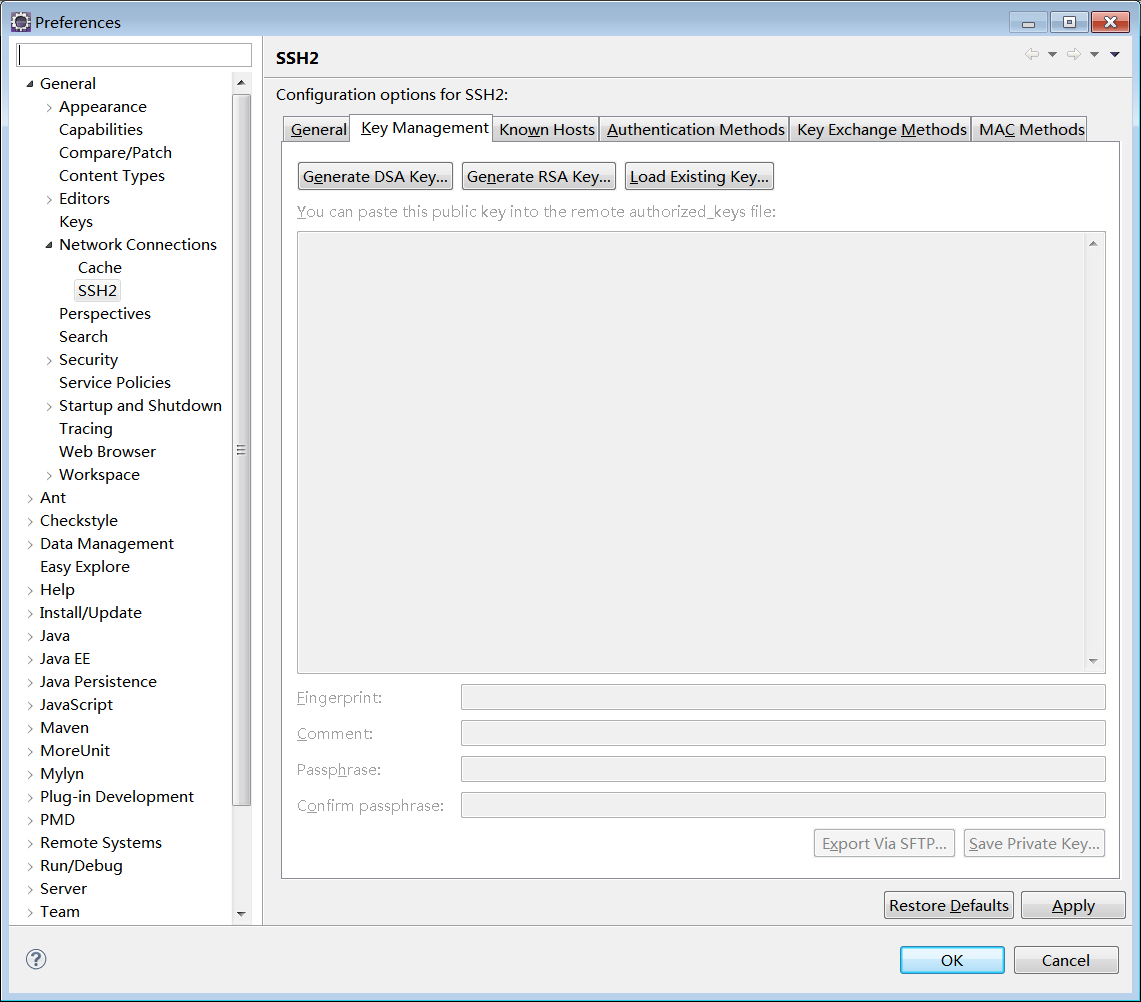




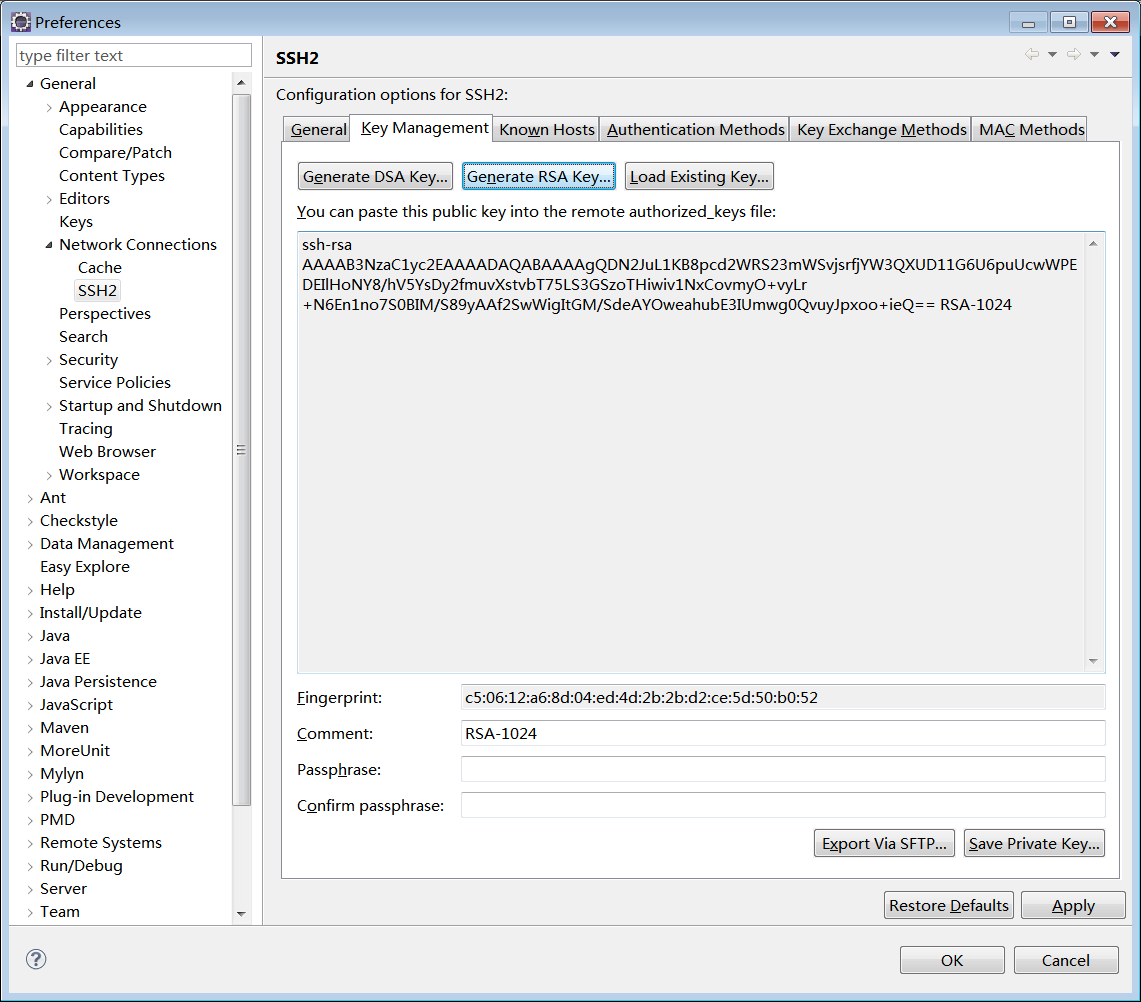
## Eclipse

从版本Juno (4.2)开始，eclipse集成了git客户端。配置认证信息以后，就可以在eclipse里像使用SVN一样使用git了。Eclipse设置认证信息的步骤如下。

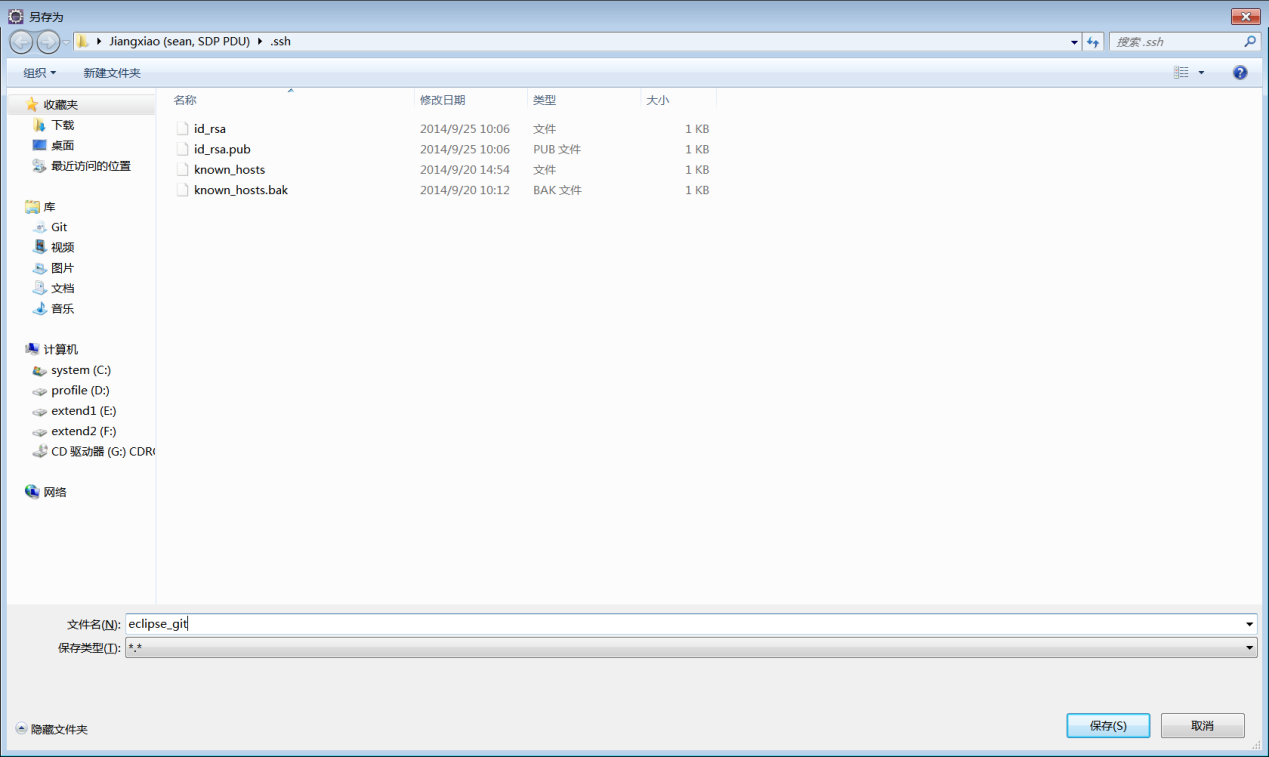
1. 打开eclipse的Preferences，打开[General]\[Network Connections]\[SSH2]，选中[Key Management]选项卡。

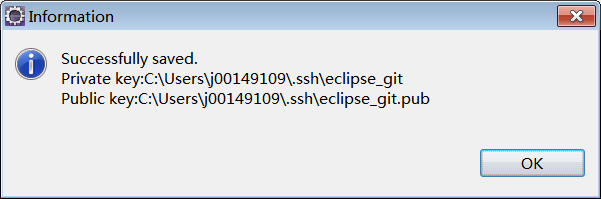


1. 点击[Generate RSA Key…]按钮，会随机生成一个key。

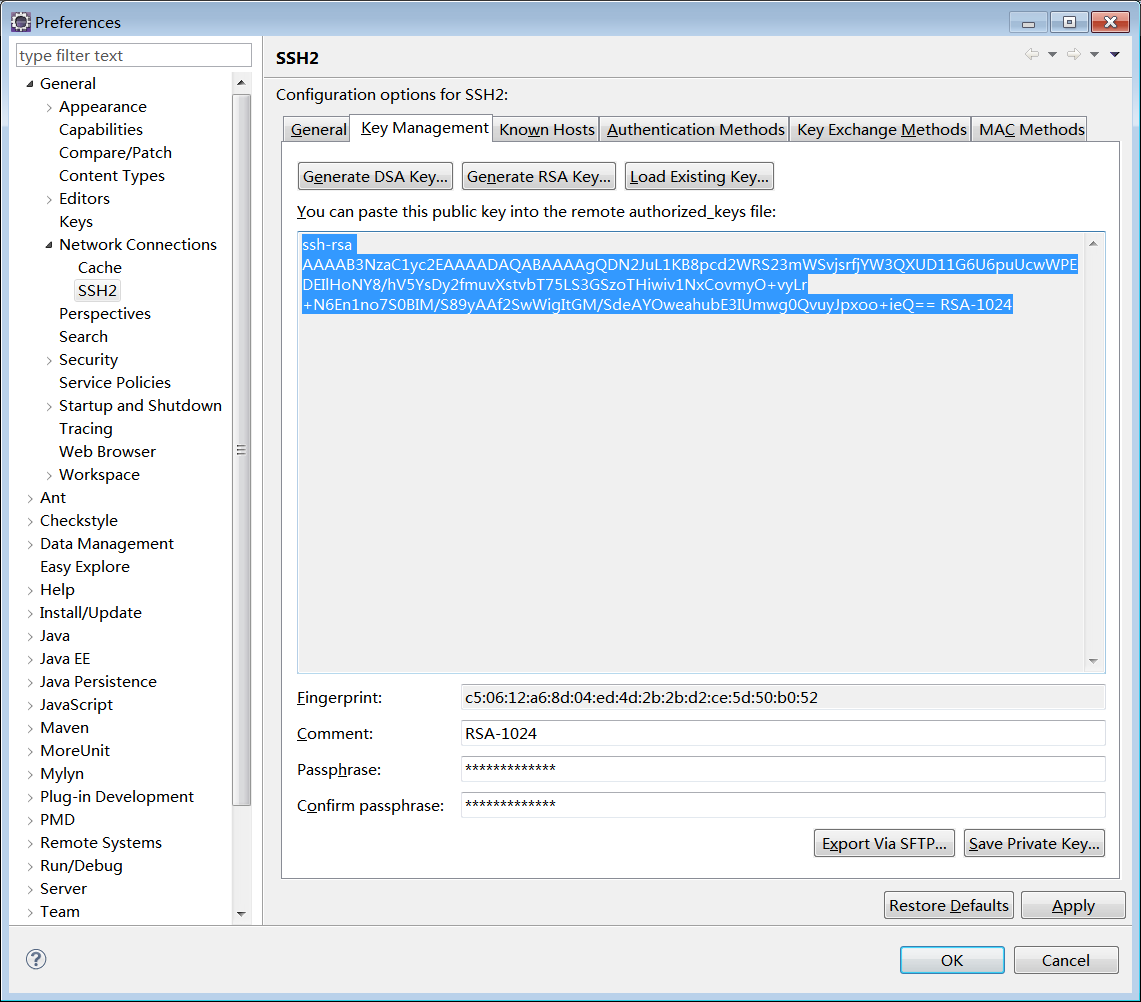


1. 在[Passphrase]和[Confirm passphrase]输入密码，点击[Save Private Key…]按钮，保存私钥。





1. 复制蓝色高亮内容，即公钥。接下来按照3.2节第7步开始，把这个key加入到isourceb中。



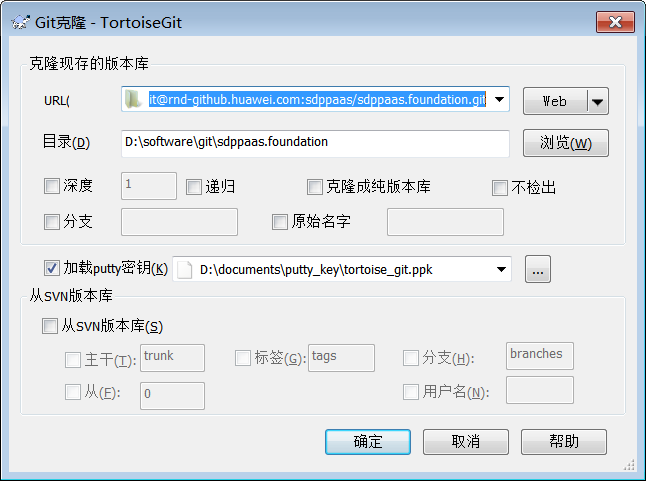
# 客户端使用

首先介绍一下git的提交(commit)机制。Git的提交不是直接提交到服务器，而是先提交到本地缓存。当提交完成后，要再推送(push)到服务器，服务器上才能有自己提交的内容。

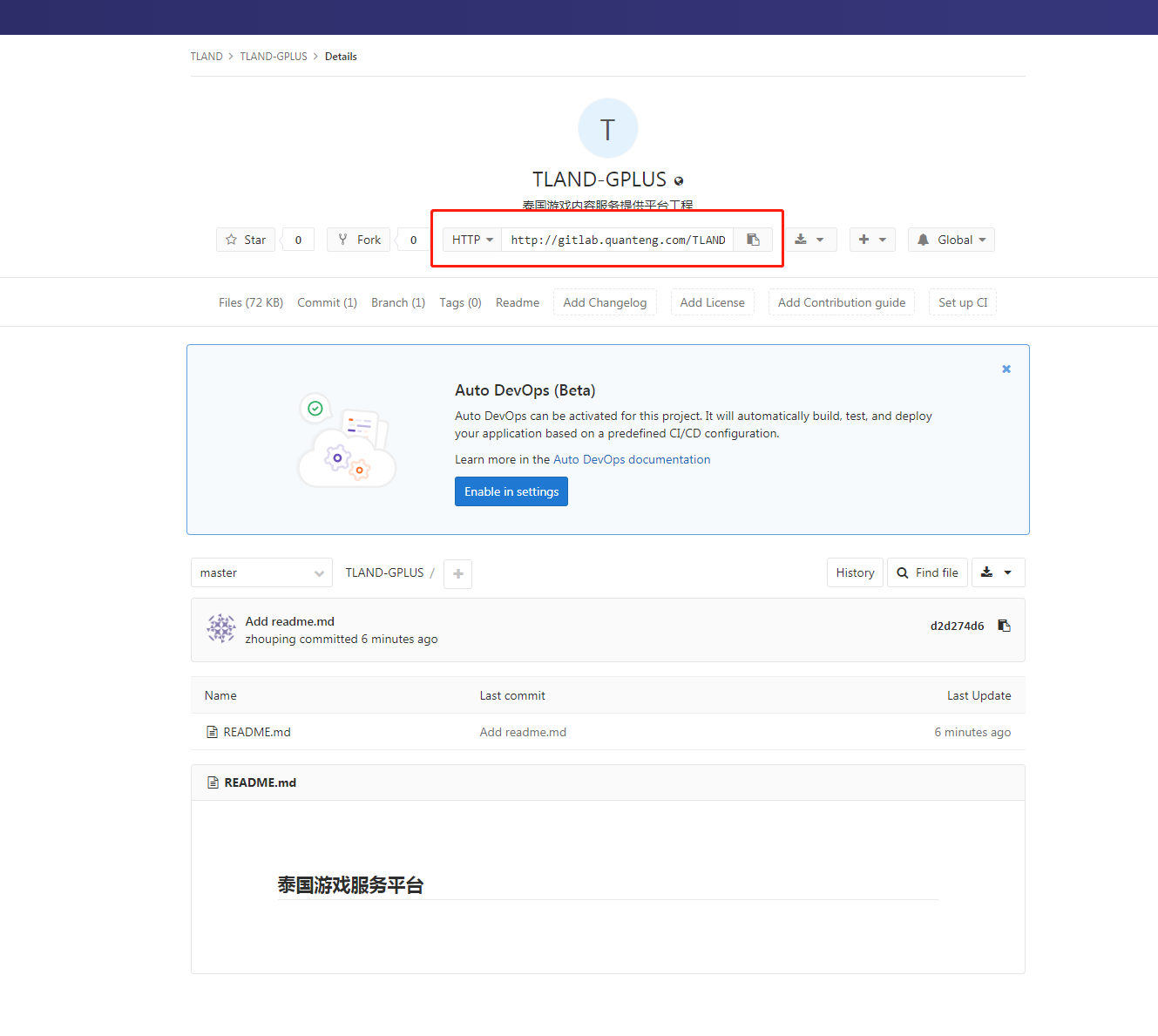
接下来就以TortoiseGit为例简单介绍一下各个基本操作，eclipse操作类似。

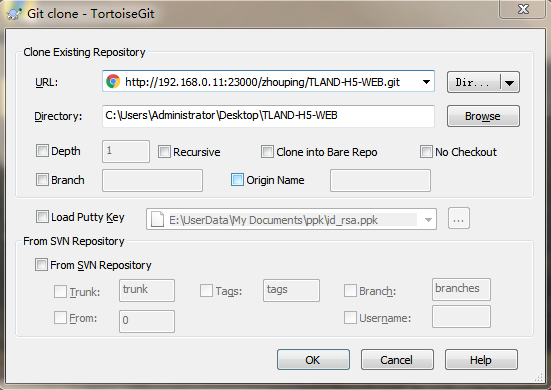
## 克隆(clone)

这个功能和SVN的检出(checkout)类似，从github上检出一个项目。在工作目录下右击鼠标，选择git克隆。



[URL]从gitlab上获取，注意要用http地址,并修改http地址中的gitlab.quanteng.com为192.168.0.11:23000。

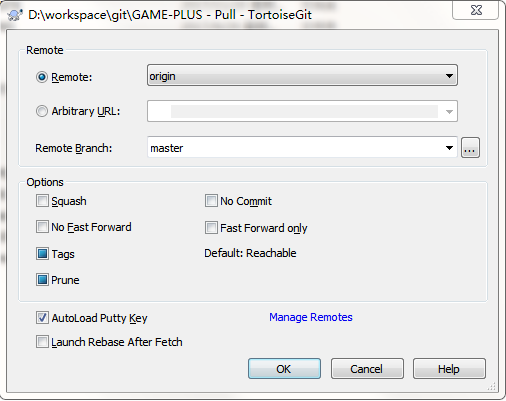




[加载putty密钥]选择前面用Puttygen生成的ppk文件。

## 拉取(pull)

这个功能和SVN更新(update)类似。克隆后再拉取，[远端]默认为origin，可以通过点击下面[远端管理]链接进行管理。[远端分支]默认为master，不用修改。勾选[自动加载putty密钥]。

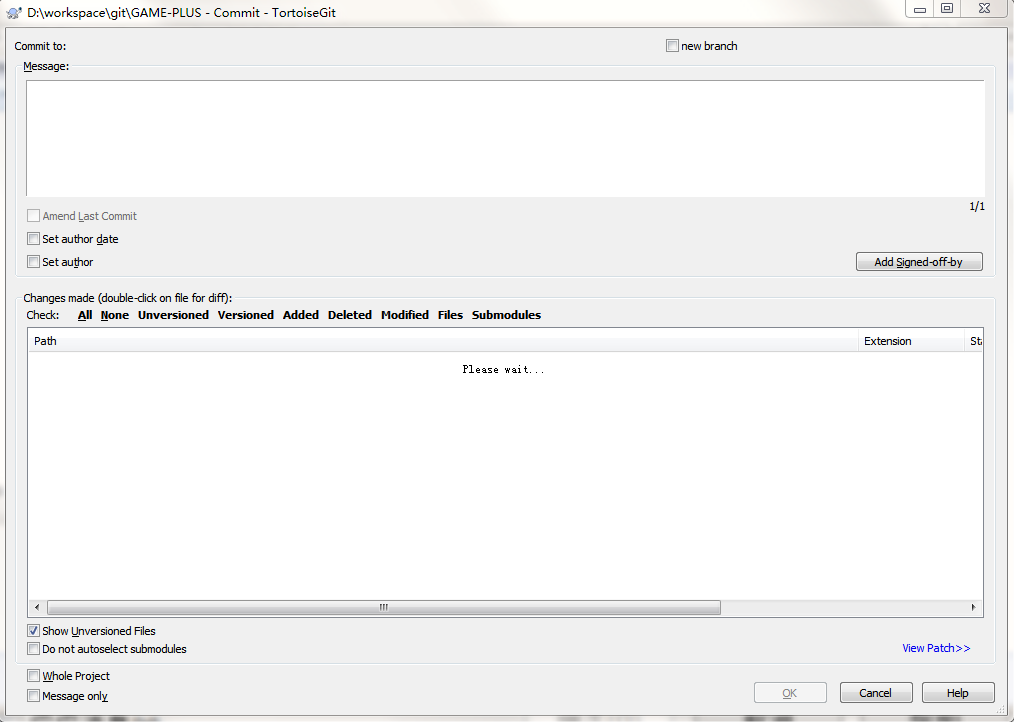


## 提交(commit)

这个功能和推送(push)组合起来类似SVN的提交(commit)，与SVN的区别是，它是向本地提交，而不会影响服务器上的内容。

右击想要提交的工程目录，选择“git提交”

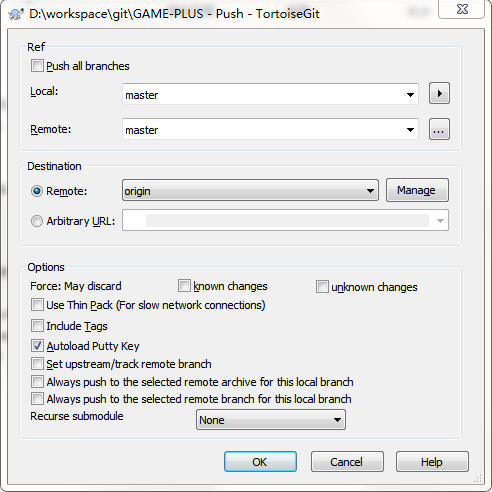
提交界面和SVN类似，注意勾选[设置作者日期]和[设置作者]。如果想要将当前目录下所有



## 推送(push)

把提交(commit)过的内容推送到github服务器。默认情况下，[本地]和[远端]都是master分支。注意勾选[自动加载putty密钥]。

右击想要提交的工程目录，选择 TortoiseGit->推送。界面如下所示。



# 命令行使用

## Git Bash安装

GitBash是一个Git的命令行终端，在内部我们可以使用Linux系统中的相关命令。熟悉Linux系统命令的可以很方便的使用。GitBash的安装只需要全部默认，一直点下一步就可以。

## Git 命令行说明

### 配置全局用户名和电子邮件地址 git config --global user.name "Your Name" git config --global user.email " your email address "

### 为特定的版本库配置用户名和电子邮件地址你可以为每个版本库单独设置用户名和邮件地址。这使得用户可在不同项目中使用不同的用户名和/或不同的邮件地址 cd /path/to/repo git config user.name "Your Name" git config user.email “your email address”

### 初始化新版本库: git init

### 将制定文件加入git控制： git add <file name> (git add .. 将当前目录下所有文件都加入控制。经过此步骤后，文件的改变记录只是保存到了git的暂存区，并没有真正的存储的本地的版本库，通过git commit命令来将暂存区的内容存储到版本库)

### 提交当前版本库到本地： git commit –m “ version name” （其中version name用来描述更改内容。例如：加入init.java文件）

### 查看当前暂存区状态： git status

### 查看操作记录 ： git log 该操作只是查看当前终端下的操作记录，使用git reflog可以查看所有操作记录。

### 恢复操作：

git checkout filename #还原状态到repository状态

 git checkout HEAD . #将所有文件都checkout出来

 git checkout xxxx . #xxxx是commit的编号的前四位，将xxxx编号的版本checkout出来

 git checkout – \* #恢复上一次commit的状态

### 图形化显示操作记录： gitk

### 显示当前工作区的内容与暂存区内容或者版本库内容的区别： git diff

### 生成public key：ssh-keygen -t rsa -C "youremail@example.com"

根据提示输入三个Enter(表示确认) 会在用户目录下生成.ssh文件夹

复制里面的.pub中的内容加入github publish key列表。这样此后的操作你才能拥有权限。

### 本地仓库与github仓库的关联操作：

* + - 1. git remote add <server-name> <server-url> 把在isourceb 上的远程仓库加入到remote列表中。Server name 一般取为 origin
      2. git remote 查看目前远端的仓库名称
      3. git –set upstream oirigin master 将本地的上传流操作与远端的master分支关联起来。第一次进行push或pull操作时需要设置
      4. git push 将本地的版本库上传到远端的对应版本库
      5. git pull 将远端的版本库拉取到本地，并且与本地的版本库进行合并。默认将进行快速合并，如果冲突不能解决会提示进行手动修复，修复完成后进行提交即可。
      6. git clone < 远端仓库地址> 将远端的仓库克隆到本地当前目录下。