**Windows环境下Github+TortoiseGet使用配置说明**

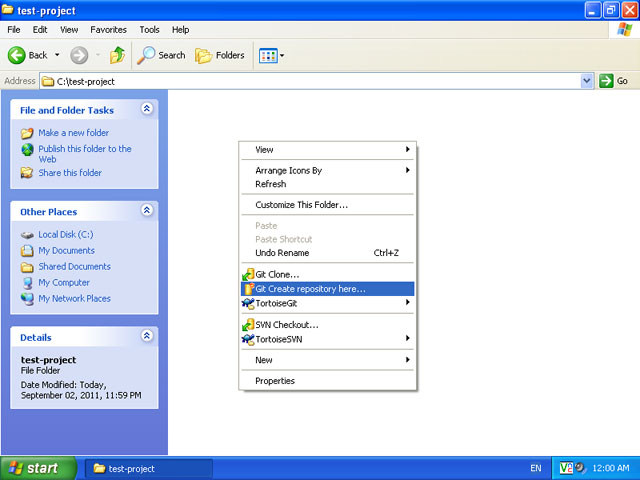
**git简介**  
  
git是由著名Linux内核(Kernel) 开发者Linus Torvalds为了便利维护Linux而开发的；

git有以下几项特色：  
  
属于分散式版本控制系统(Distributed Version Control System)速度非常快，而且在传输档案时使用gzip来减小档案大小有非常多指令可以使用，甚至能与非git系统的版本控制系统互连细谈这些特色就属于中阶git教学的内容了，简介方面我先就此打住。

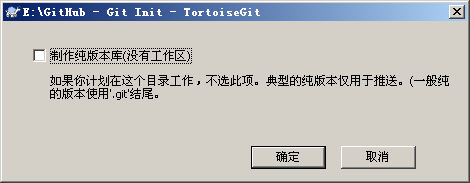
**GitHub简介**  
  
GitHub是个提供在线git空间的平台，能让多名开发人员很方便地进行协力软件开发。提供的方案包含免费和付费的账号，而其中的差异在于能否使用私人空间。虽然免费的账号只能使用公开的空间并且需要公布所有原始码，但对许多项目来说不是个问题。除此之外，免费的账号没有其他限制。  
  
使用GitHub的优点包含：  
  
拥有在线原始码和修改历史浏览，能让开发人员便利地分享原始码有人创立分支(Fork)时能够追踪所有的分支越来越多的知名企业在征才时会要求申请者提供GitHub账号，以用来观察在开源软件社群中的参与，还有程序写作风格和技巧等等 ；  
所以，要参与开源软件开发或只是要单纯备份/集中整理自己的原始码时，GitHub是个非常好的平台。  
  
  
**TortoiseGit**  
  
虽然说要用git指令才能完整地体验git的强大功能，TortoiseGit对于指令界面感到陌生的人可能会比较友善，也能够在功能和易用性上有个好的平衡。  
1.安装msysgit： <http://code.google.com/p/msysgit/downloads/list>（现在能下载到1.8.3）,下载后安装（win xp系统需要升级到SP3）。装了这个以后TortoiseGit才能正常使用。

2.TortoiseGit可到此下载： <http://code.google.com/p/tortoisegit/downloads/list>（现在能下载到1.8.4）,根据系统的类型，选择下载64bit或是32bit的版本；下载好以后安装然后重开机。  
  
**一、初始化本地Git项目环境**

1.建立git项目(Init)并且储存修改(Commit) ：

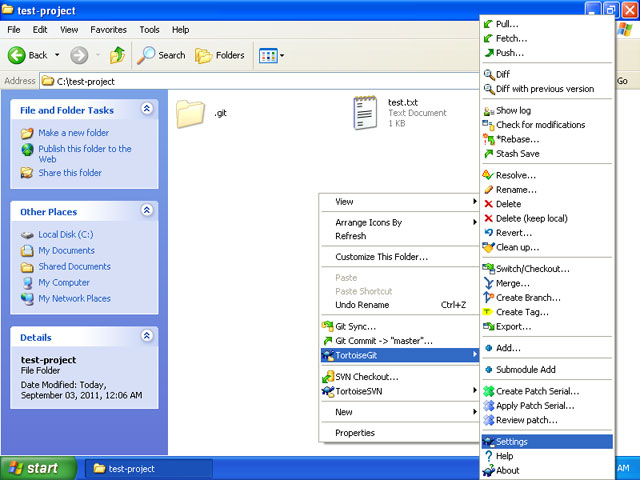


2.首先，选择一个已经存在的文件夹（英文名字），点右键并选择Git Create repository here…，此时会弹出一个对话框：

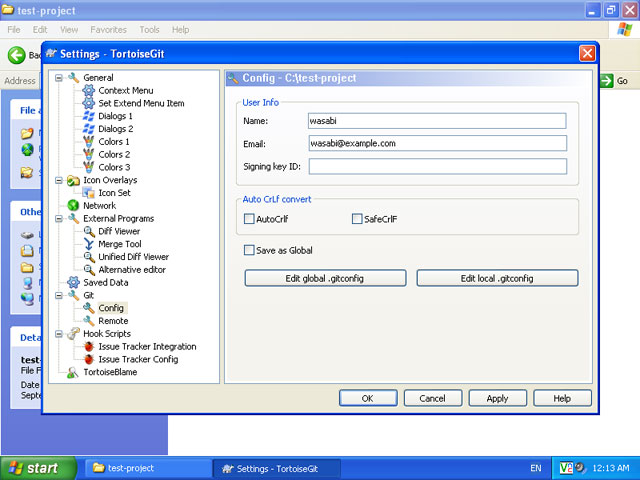


直接点击确认，即会创建一个可以在这个目前工作的本地库（打钩则会创建一个纯版本库，不能支持开发）；好了以后，如果有开啓显示隐藏档案，便可以看到一个.git的隐藏资料夹。

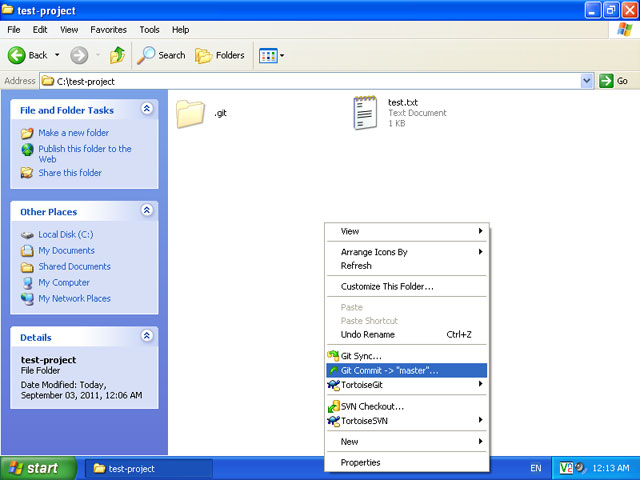
2.设置账户信息：



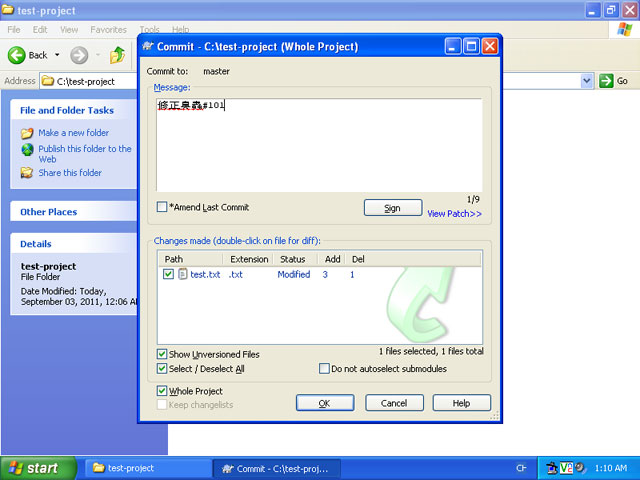
 选择config项：



建立好git数据库以后，要设定作者信息。在同个资料夹内按右键，并选择TortoiseGit -> Settings。到Git -> Config页面，在Name输入你的名字，在Email输入你的E-mail。好了以后按OK 关闭。

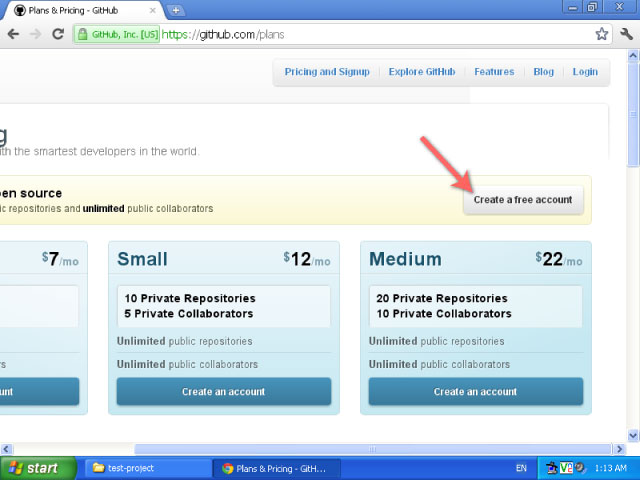


有原始码可以储存以后，在资料夹内按右键，并选择Git Commit -> "master"。master指的是当前的分支；

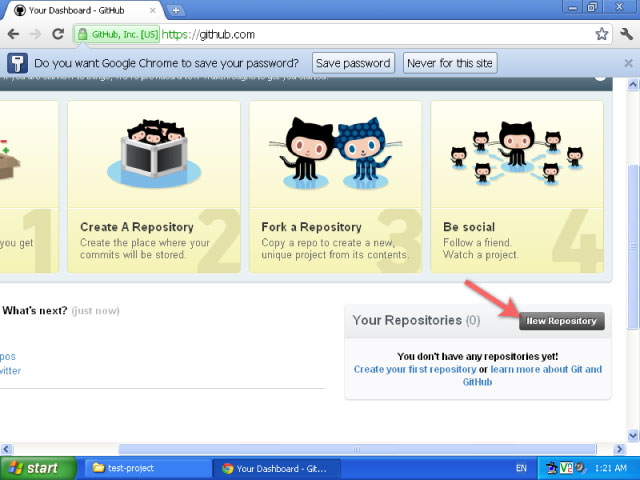


出现Commit视窗时，在Message中输入要储存变更的注释。好的开发者通常会在此大致叙述此次修改有新增或异动的内容。输入讯息后，勾取要储存的档案。要储存所有档案中的变更，勾Select / Deselect All。好了以后，点OK即可。等储存完成后，点Close来关闭视窗。

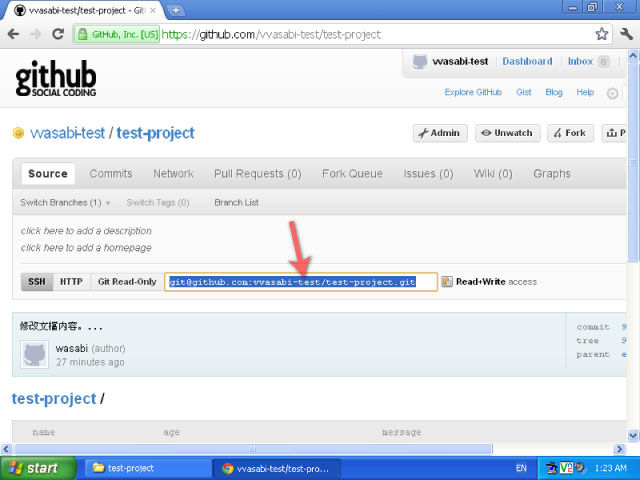
**二、建立GitHub账号 （如果是开发者、只是使用github，本节可以跳过）**



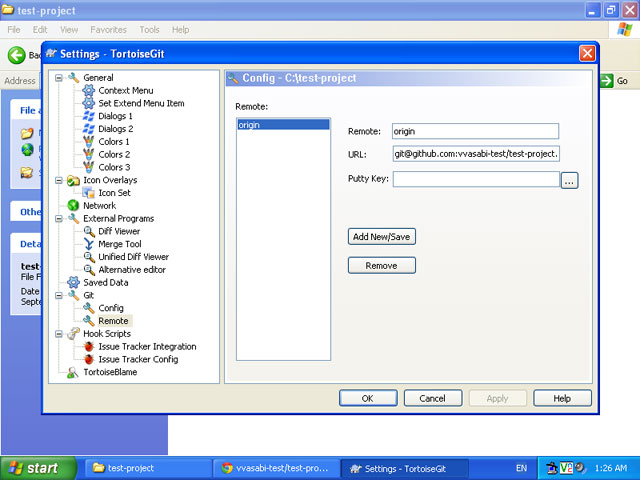
下一步骤便是建立一个GitHub账号： <https://github.com/plans>。如果要用免费方案，点Create a free account。输入使用者信息，并Create an account。



账号建立好以后，点New Repository以建立一个源码库。输入项目名称后按Create Repository即可。

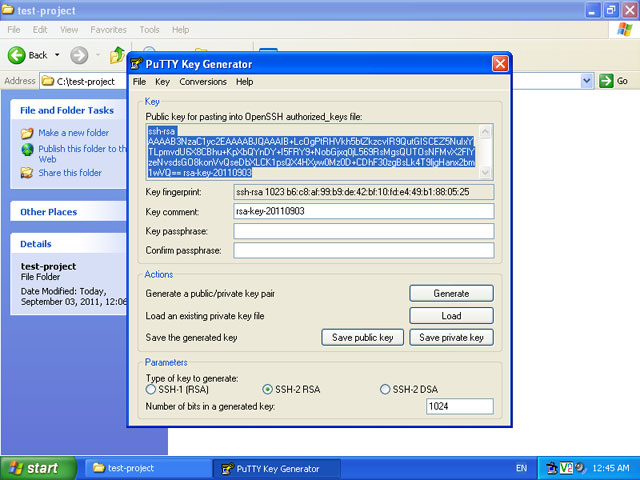


项目建立好以后，复制源码库的连结，如git@github.com:vvasabi-test/test-project.git。回到项目资料夹，点右键并选择TortoiseGit -> Settings。

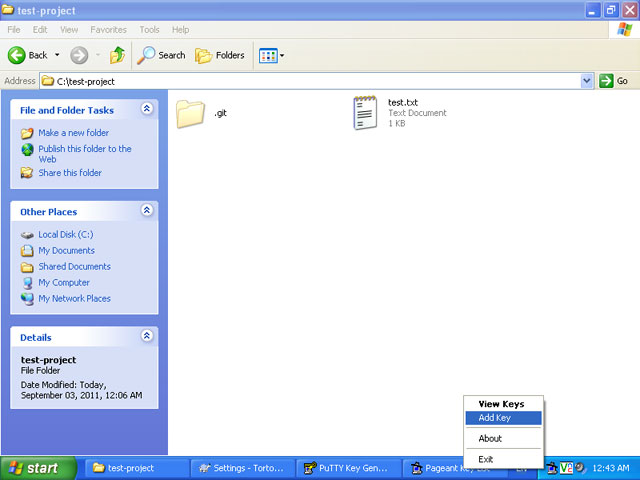


在这个视窗中，选择Git -> Remote，在Remote中输入origin，Url中贴上刚刚复制的连结然后点Add New/Save。好了以后按OK。  
这些步骤每个项目只需要做一次。

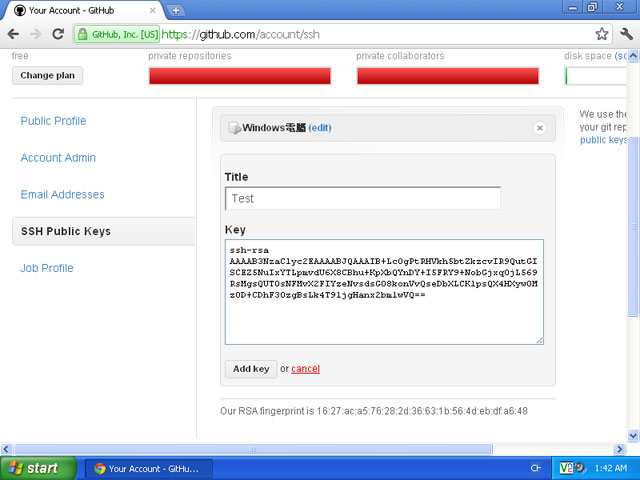
**三、上传修改内容(Push)**



现在要把修改的内容上传到GitHub。首先要建立认证密钥，开启开始选单 -> 所有程序 -> TortoiseGit -> Puttygen。点Generate，等待程序跑完，然后点Save private key，将此密钥存到任意处。如果需要保障密钥不会被盗用，可在储存前输入Key passphrase来用密码保护密钥(同样的密码需在Confirm passphrase内输入一次)。如果没有输入密码，出现PuTTYgen Warning时点Yes即可。

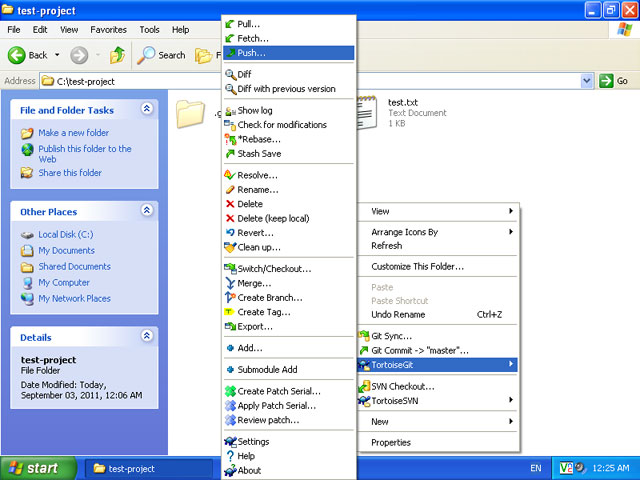


密钥储存完以后，回到开始选单，开启所有程序 -> TortoiseGit -> Pageant。这时状态列会跑出一个新图示，在这图示上点右键并选择Add Key，然后选择刚刚储存的密钥。



回到PuTTY Key Generator的视窗，并且选取和复制Public Key。开启GitHub网页，并点取Add your public key。点Add another public key，在Title栏输入随意名称，在Key栏贴上密钥，好了以后点Add key。

设定密钥的步骤虽然很麻烦，好消息是一台电脑只要设定一次。



回到项目资料夹，并按一右键，选择TortoiseGit -> Push…。勾Push all branches，然后点OK，出现PuTTY Security Alert时点Yes(这只会出现一次)。如果没有出现错误警告，这就代表项目已成功地上传到GitHub了。

**四、日常工作流程**  
  
开始使用GitHub时配置起来可能会比较复杂，但是大部分的配置只需要一次即可，日常使用github的步骤如下：   
1.提交（Commit）代码到本地git库：平常修改程序码时，在修改完以后需要储存变更(Commit)，在项目资料夹内点右键并点Git Commit -> "master"…；输入修改的讯息，选择要储存的档案并按OK；  
2.推送（Push）本地版本库的内容到github库：点TortoiseGit -> Push…来将修改上传到GitHub；勾Push all branches，然后点OK；

注意事项：如果Push出现错误，请确定：

1. Pageant是否已经打开？
2. Pageant是否导入Private key？
3. Github账户是否添加了SSH Public key（要与本地的private key是配对的）？