Git、GitHub、GitLab的联系与区别

Git：是一个开源的分布式版本控制系统，可以有效、高速的处理项目版本管理。它是一个命令，是一种工具。

GitHub：是一个开源项目托管平台，提供公共仓库和私有仓库，使用私有仓库需要付费。

GitLab：使用MIT许可证的基于网络的Git仓库管理工具，且具有wiki和issue跟踪功能。可以创建私人的免费仓库，使用与GitHub类似。

**联系：**

GitHub和GitLab都是基于Git实现的在线代码仓库，主要用来存放代码和管理代码的两个网站。使用方法二者相似，都提供了分享开源项目的平台，为开发团队提供了存储、分享、发布和合作开发项目的中心化云存储的场所。

**区别：**

1、GitHub使用私有仓库，需要付费；

GitLab可以创建私人的免费仓库，且可以部署在自己的服务器上。

2、GitLab让开发团队对他们的代码仓库拥有更多的控制，相比较GitHub，GitLab拥有更多的特色：

1. 允许免费设置仓库权限
2. 允许用户选择分享一个project的部分代码
3. 允许用户设置project的获取权限，进一步提升案例性
4. 可以设置获取到团队整体的改进进度
5. 通过innersourcing让不在权限范围内的人访问不到该资源

总结：从代码的私有性来看，GitLab是一个更好的选择；

但相对开源项目而言，GitHub是代码托管的首选。

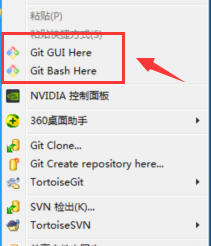
Git的安装和使用

## 一、Git安装

1、下载并安装Git-2.26.2-64-bit.exe



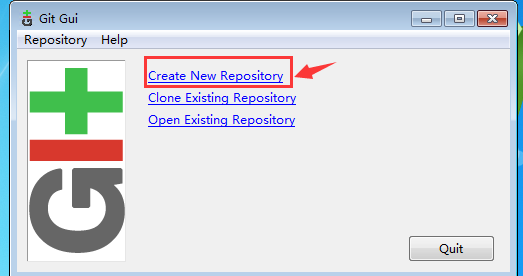
2、安装完毕后，鼠标右键快捷菜单中多了Git GUI Here 和Git Bash Here，分别为Git的GUI和Git的命令行工具



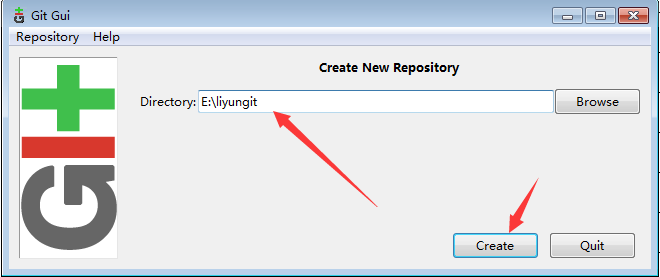
## 二、Git使用

### （一）创建Git本地仓库

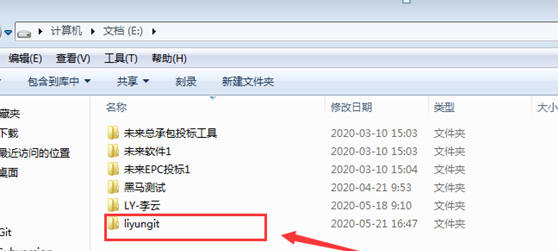
1、鼠标右键选择Git GUI Here打开，在弹出的Git Gui窗口中选择Create New Repository，创建一个仓库



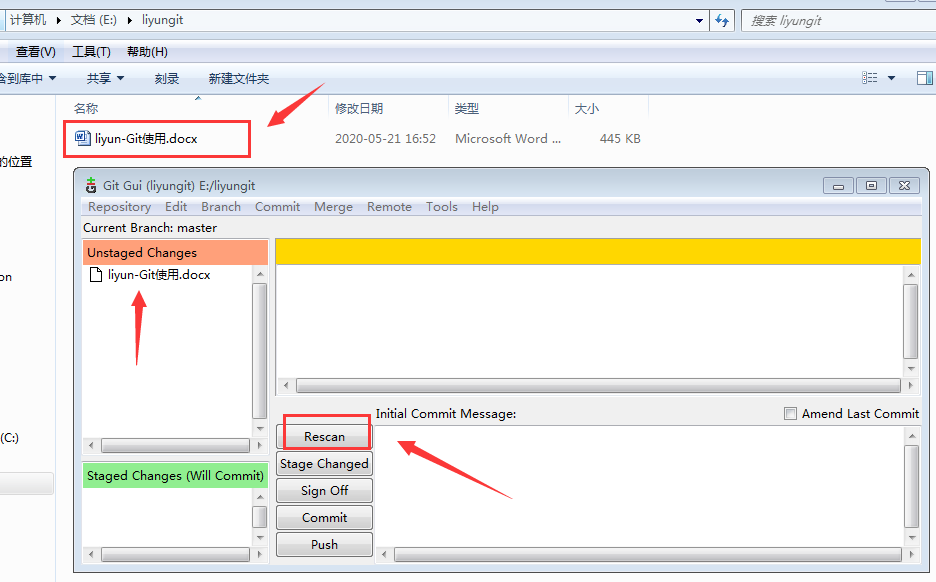
2、输入或选择一个路径作为仓库，点击Create进行创建



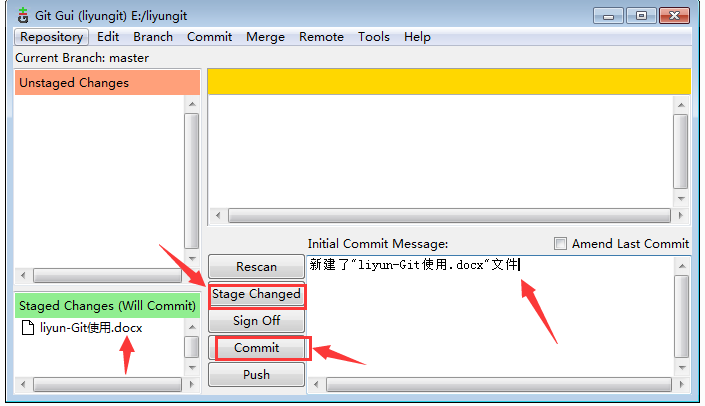
3、创建完毕后，在刚才的路径下多了一个liyungit的空的本地仓库



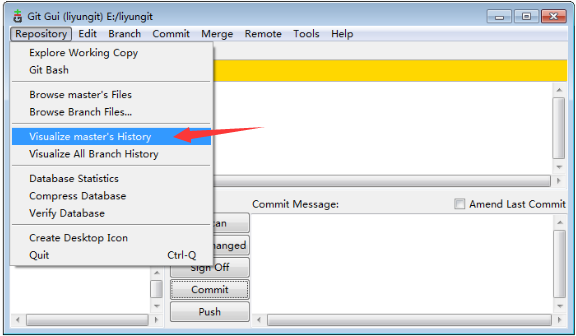
4、在空仓库中新建或放入需要被管理的文件，并在Git GUI中点击Rescan进行刷新，此时可以看到Unstaged Changes区域下显示了在本地库中新建的文件“liyun-Git使用.docx”

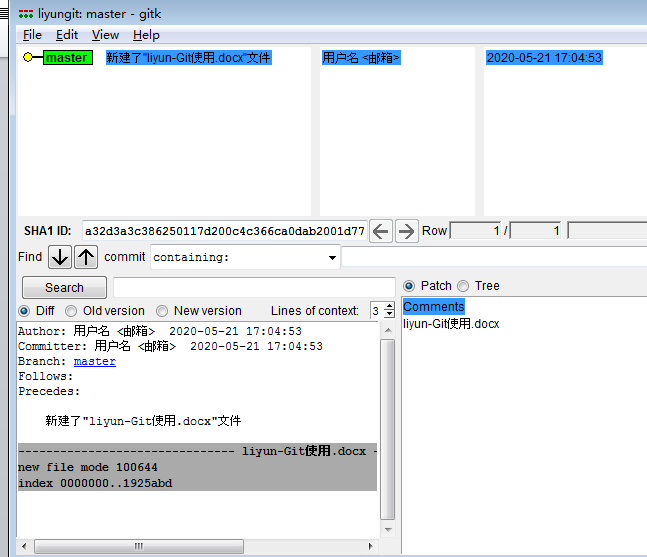


5、点击Stage Changed按钮将文件添加到Staged Changes(Will Commit)区域中，接着在提交信息输入框中输入本次提交的说明，最后点击Commit按钮进行提交



6、提交成功后，可以通过Repository菜单下的Visualize master’s History按钮查看到提交的历史记录 （注：本地库提交不用Push）

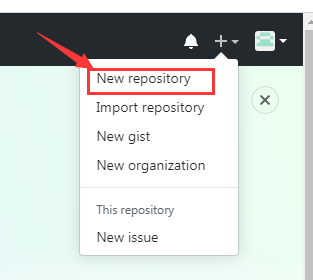


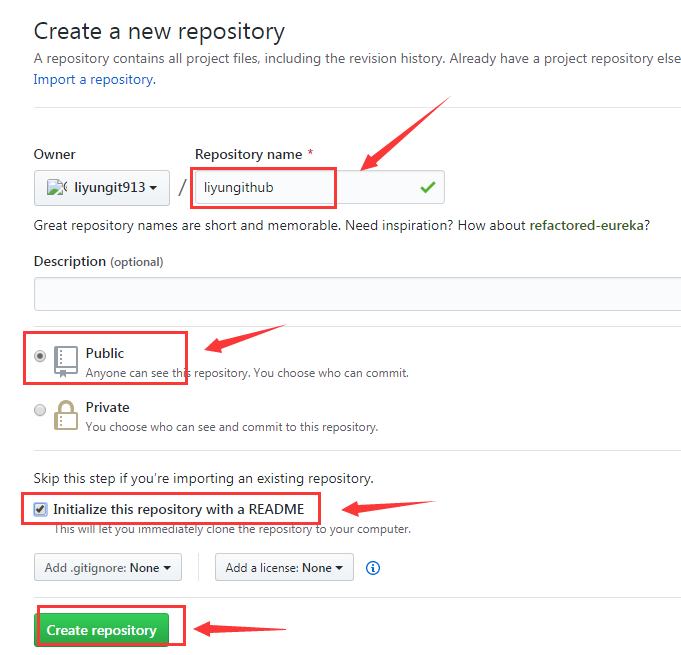


### （二）创建Git Hub远程仓库并与本地进行连接

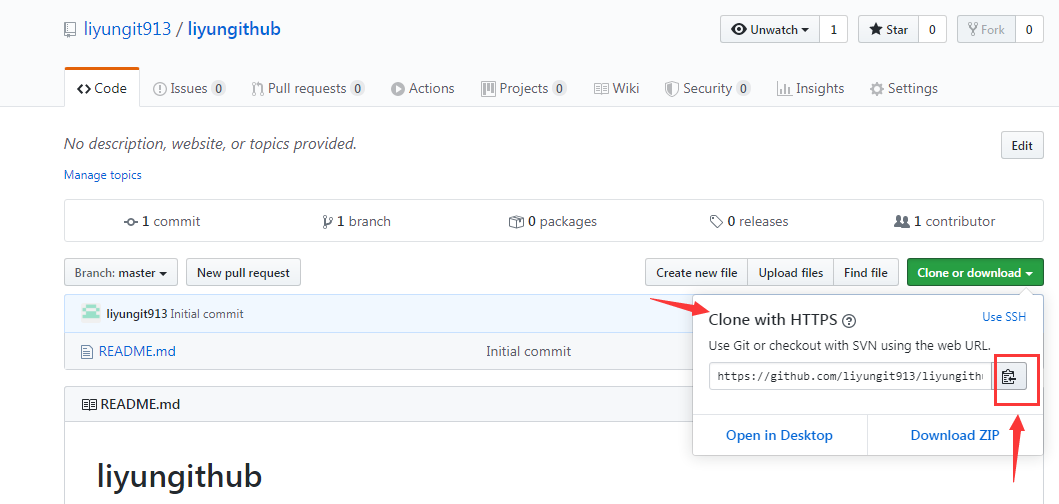
1、登录Git Hub官网，注册一个账号并登录账号

2、点击create a new repository按钮，创建一个远程仓库“liyungithub”，选择public公用的，并勾选使用自述文件初始化此存储库的选项，最后点击Create repository按钮进行创建

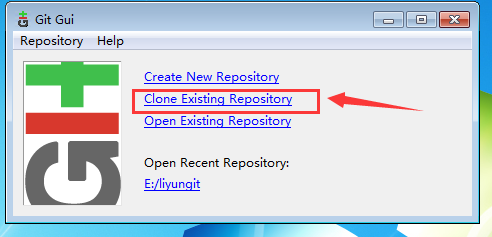




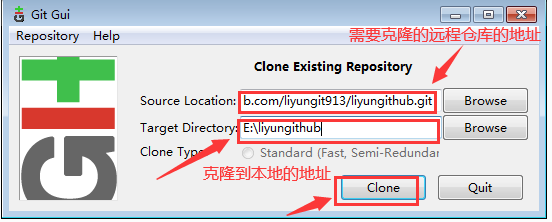
3、创建成功后，点击Clone or download按钮复制远程仓库地址(克隆方式https和ssh均可)



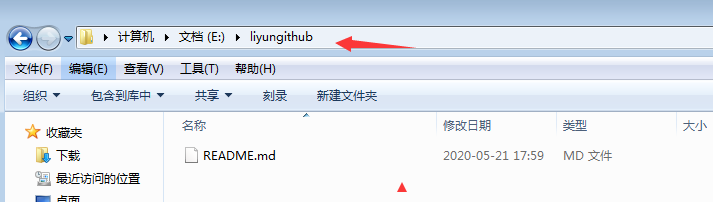
4、打开Git Gui工具，选择Clone Existing Repository克隆远程仓库



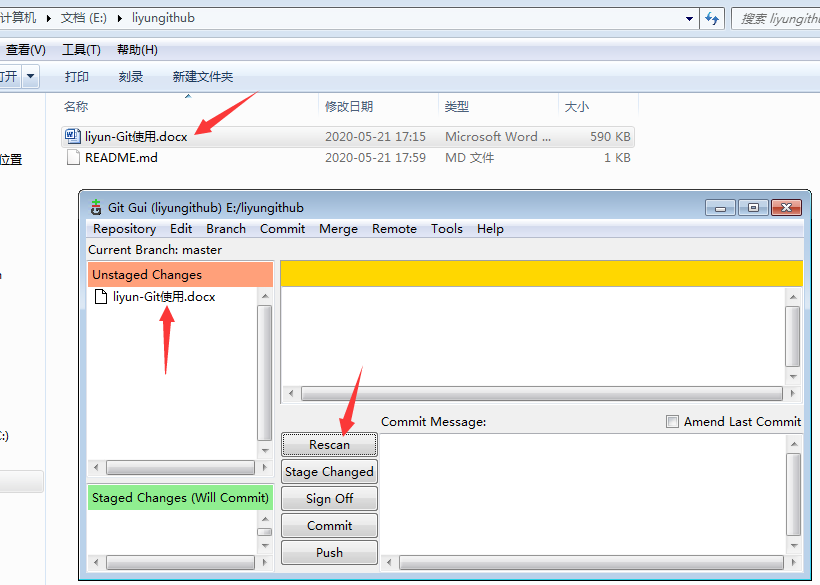
5、将从Git Hub上复制的远程仓库地址粘贴到Source Location栏，并在Target Directory地址栏中输入或选择本次需要克隆到本地的路径，最后点击Clone按钮进行克隆远程仓库到本地（注：本地的文件夹liyungithub仓库不能提前新建，否则克隆会报错）



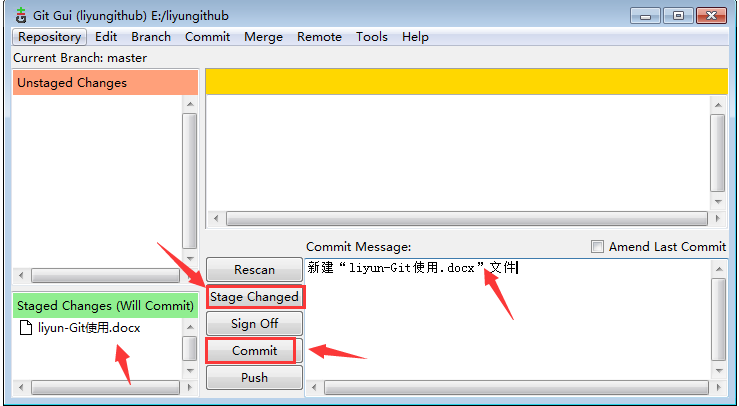
6、克隆成功后，在本地liyungithub仓库下会显示克隆仓库的内容（README.md为创建远程仓库自带的文件）



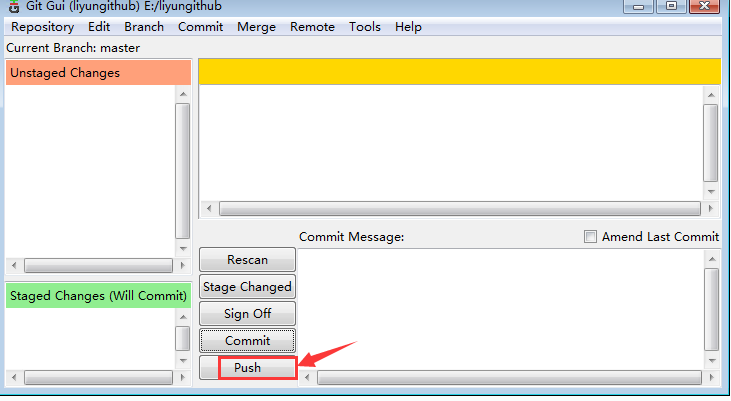
7、在仓库中放入需要管理的文件“liyun-Git使用.docx”，并在Git GUI中点击Rescan进行刷新，此时可以看到Unstaged Changes区域下显示了新建的文件

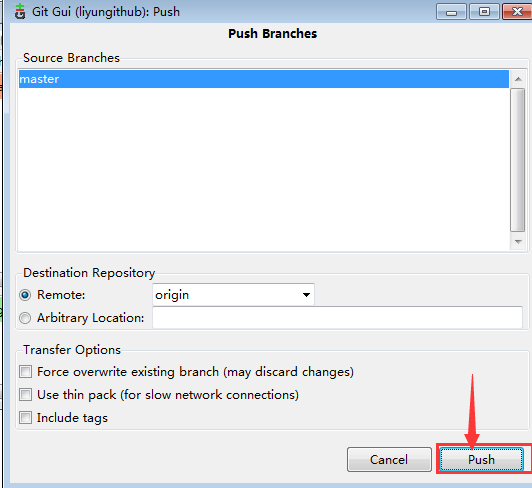


8、点击Stage Changed按钮将文件添加到Staged Changes(Will Commit)区域中，并在提交信息输入框中输入本次提交的说明，最后点击Commit按钮进行提交

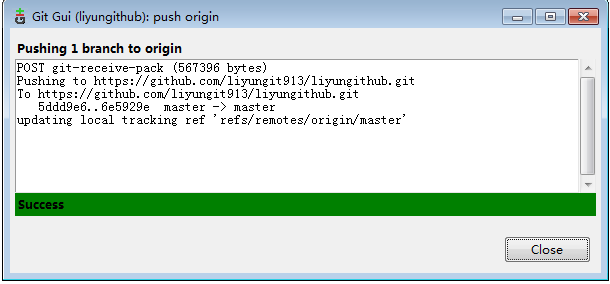


9、提交成功后，点击Push按钮，在弹出的窗口中继续点击Push，将数据推送到远程仓库

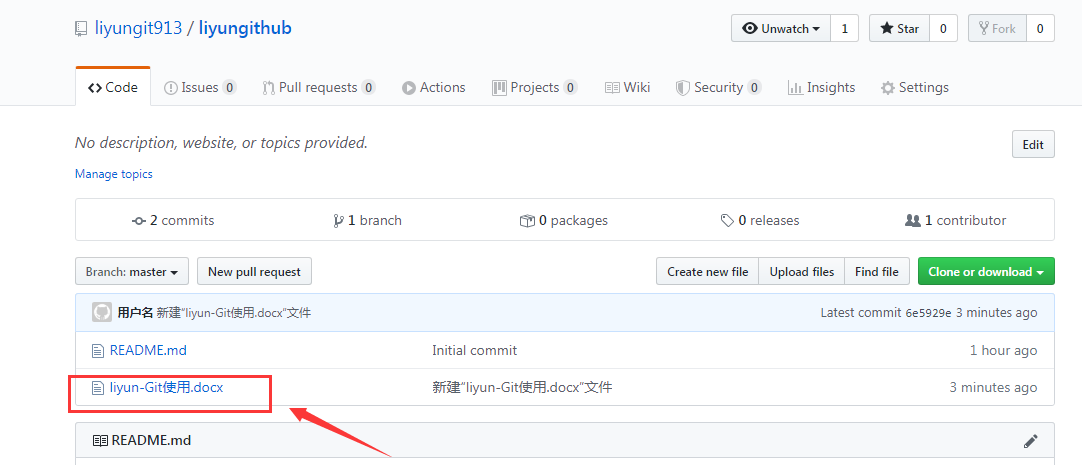




10、推送成功后，点击Close按钮关闭



11、最后登录Git Hub账号，刷新网页进行查看，此时已能看到文件被推送成功了



12、同样的，可以通过Git Gui工具中的Repository菜单下的Visualize master’s History按钮查看到提交的历史记录

