1.版本控制

如果在开发团队中没有使用版本控制，多个开发人员共同负责同一个软件或文档的开发，每个人在各自的机器上有整个软件文档的备份，并对之实施编程开发，在分别完成各自任务之后，再通过文本比对工具将各自机器上的不同版本的程序整合到一台机器上。没有进行版本控制或者版本控制本身缺乏正确的流程管理，在软件开发过程中将会引入很多问题，如软件代码的一致性、软件内容的冗余、软件过程的事务性、软件开发过程中的并发性、软件源代码的安全性，以及软件的整合等问题。

版本控制的目的是实现开发团队并行开发、提高开发效率的基础。其目的在于对软件开发进程中文件或目录的发展过程提供有效的追踪手段，保证在需要时可回到旧的版本，避免文件的丢失、修改的丢失和相互覆盖，通过对版本库的访问控制避免未经授权的访问和修改，达到有效保护企业软件资产和知识产权的目的。

版本控制的功能在于跟踪记录整个软件的开发过程，包括软件本身和相关文档，以便对不同阶段的软件及相关文档进行表示并进行差别分析，对软件代码进行可撤消的修改，便于汇总不同开发人员所做的修改，辅助协调和管理软件开发团队。

2.版本控制工具

2.1.Visual Source Safe(简称VSS)

VSS是美国微软公司的产品，目前常用的版本为6.0版。VSS是配置管理的一种很好的入门级的工具。

易学易用是VSS的强项，VSS采用标准的windows操作界面，只要对微软的产品熟悉，就能很快上手。VSS的安装和配置非常简单，对于该产品，不需要外部的培训（可以为公司省去一笔不菲的费用）。只要参考微软完备的随机文档，就可以很快的用到实际的工程当中。

VSS的配置管理的功能比较基本，提供文件的版本跟踪功能，对于build和基线的管理，VSS的打标签的功能可以提供支持。VSS提供share（共 享)、branch(分支）和合并（merge)的功能，对于团队的开发进行支持。VSS不提供对流程的管理功能，如对变更的流程进行控制。

VSS不能提供对异地团队开发的支持。此外VSS只能在windows平台上运行，不能运行在其他操作系统上。 有软件提供商提供VSS插件，可以同时解决VSS跨平台和远程连接两个问题，例如SourceAnywhere for VSS, SourceOffSite等。

VSS的安全性不高，对于VSS的用户，可以在文件夹上设置不可读，可读，可读/写,可完全控制四级权限。但由于VSS的文件夹是要完全共享给用户后，用户才能进入，所以用户对VSS的文件夹都可以删除。这一点也是VSS的一个比较大的缺点。

VSS没有采用对许可证进行收费的方式，只要安装了VSS，对用户的数目是没有限制的。因此使用VSS的费用是较低的。

微软不再对VSS提供技术支持。

2.2.Concurrent Version System(简称CVS)

CVS是开发源代码的配置管理工具，其源代码和安装文件都可以免费下载。

CVS是源于unix的版本控制工具，对于CVS的安装和使用最好对unix的系统有所了解能更容易学习，CVS的服务器管理需要进行各种命令行操作。目前，CVS的客户端有winCVS的图形化界面，服务器端也有CVSNT的版本，易用性正在提高。

CVS的功能除具备VSS的功能外，还具有：

它的客户机/服务器存取方法使得开发者可以从任何因特网的接入点存取最新的代码；它的无限制的版本管理检出(checkout)的模式避免了通常的 因为排它检出模式而引起的人工冲突；它的客户端工具可以在绝大多数的平台上使用。同样，CVS不提供对变更流程的自动管理功能。

一般来说，CVS的权限设置单一，通常只能通过CVSROOT/passwd, CVSROOT/readers, CVSROOT/writers文 件，同时还要设置CVS REPOS的物理目录权限来完成权限设置，无法完成复杂的权限控制；但是CVS通过CVS ROOT目录下的脚本，提供了相应功 能扩充的接口，不但可以完成精细的权限控制，还能完成更加个性化的功能。

CVS是开发源码软件，无需支付购买费用。

同样因为CVS是开发源码软件，没有生产厂家为其提供技术的支持。如发现问题，通常只能靠自己查找网上的资料进行解决。

2.3.SVN

SVN全名Subversion，即版本控制系统。

SVN与CVS一样，是一个跨平台的软件，支持大多数常见的操作系统。作为一个开源的版本控制系统,Subversion 管理着随时间改变的数据。 这些数据放置在一个中央资料档案库中。 这个档案库很像一个普通的文件服务器, 不过它会记住每一次文件的变动。 这样你就可以把档案恢复到旧的版本, 或是浏览文件的变动历史。Subversion 是一个通用的系统, 可用来管理任何类型的文件, 其中包括了程序源码。

SubVersion：实现服务系统的软件。

TortoiseSVN：是SVN客户端程序，为windows外壳程序集成到windows资源管理器和文件管理系统的Subversion客户端。

SVNService.exe：是专为 SubVersion 开发的一个用来作为 Win32 服务挂接的入口程序。

AnkhSVN：是一个专为Visual Studio提供SVN的插件。

2.4.Git

Git是一个开源的分布式版本控制系统，用以有效、高速的处理从很小到非常大的项目版本管理.

Git 是 Linus Torvalds 为了帮助管理 Linux 内核开发而开发的一个开放源码的版本控制软件。

Torvalds 开始着手开发 Git 是为了作为一种过渡方案来替代 BitKeeper，后者之前一直是 Linux 内核开发人员在全球使用的主要源代码工具。开放源码社区中的有些人觉得 BitKeeper 的许可证并不适合开放源码社区的工作，因此 Torvalds 决定着手研究许可证更为灵活的版本控制系统。尽管最初 Git 的开发是为了辅助 Linux 内核开发的过程，但是我们已经发现在很多其他自由软件项目中也使用了 Git。例如最近就迁移到 Git 上来了，很多 Freedesktop 的项目也迁移到了 Git 上。

3.Git的使用

Git 与 SVN 区别

- 1、Git 是分布式的，SVN 不是：

这是 Git 和其它非分布式的版本控制系统，例如 SVN，CVS 等，最核心的区别。

- 2、Git 把内容按元数据方式存储，而 SVN 是按文件：

  所有的资源控制系统都是把文件的元信息隐藏在一个类似 .svn、.git 等的文件夹里。

- 3、Git 分支和 SVN 的分支不同：

  分支在 SVN 中一点都不特别，其实它就是版本库中的另外一个目录。Git 分支是指针指向某次提交，而 SVN 分支是拷贝的目录。这个特性使 Git 的分支切换非常迅速，且创建成本非常低。

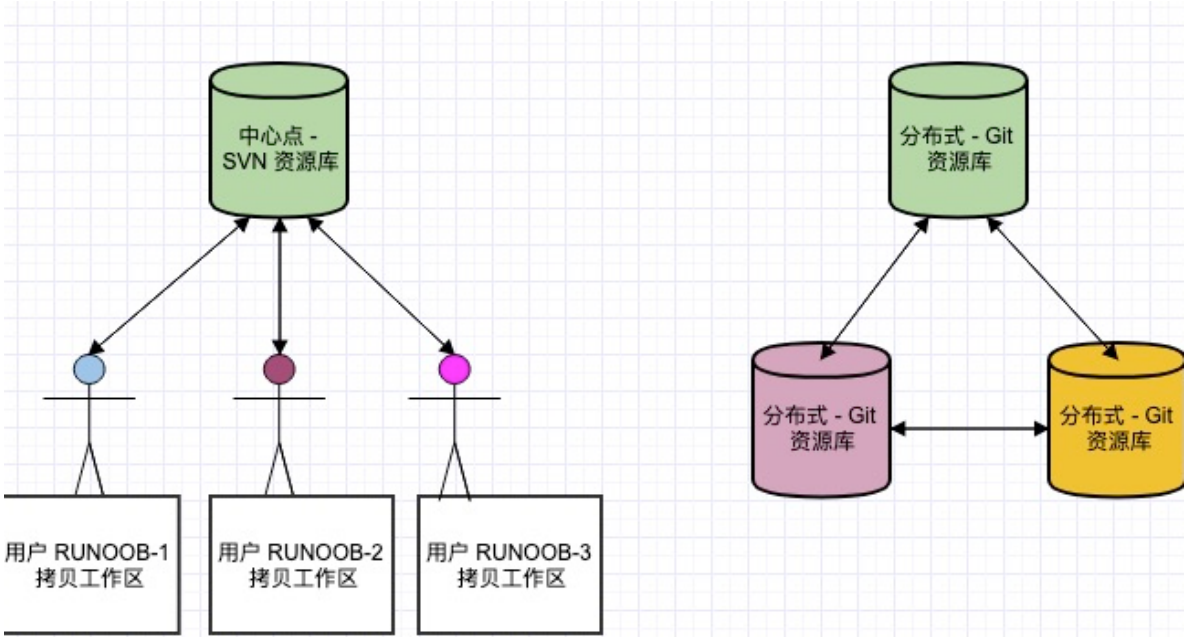
  Git 有本地分支，SVN 无本地分支。在实际开发过程中，经常会遇到有些代码没写完，但是需紧急处理其他问题，若我们使用 Git，便可以创建本地分支存储没写完的代码，待问题处理完后，再回到本地分支继续完成代码。

- 4、Git 没有一个全局的版本号，而 SVN 有：

  目前为止这是跟 SVN 相比 Git 缺少的最大的一个特征。

- 5、Git 的内容完整性要优于 SVN：

  Git 的内容存储使用的是 SHA-1 哈希算法。这能确保代码内容的完整性，确保在遇到磁盘故障和网络问题时降低对版本库的破坏。



3.1安装

下载地址:https://www.git-scm.com/download/win

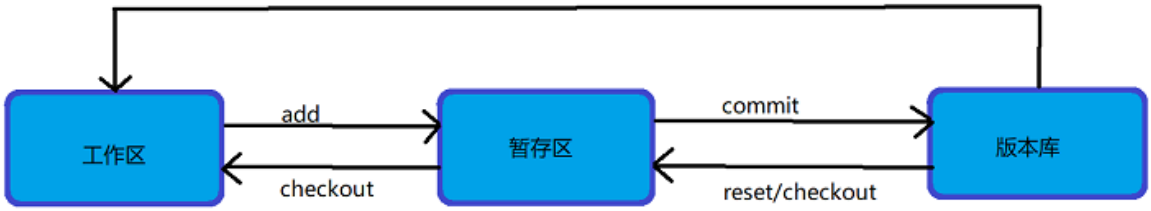
检验是否安装成功,桌面上鼠标右击后:

3.2 git的工作区域

(1)工作区:用来对代码进行修改

(2)暂存区

(3)git仓库区



工作区->暂存区 git add 文件名

暂存区->仓库 git status 先查看文件状态

git commit -m “提交描述”

3.3 初始化仓库

(1)新建文件夹，进入到该目录,右键打开git bash

(2)在文件夹内初始化git(创建git仓库)

命令:git init (会生成一个.git的隐藏文件)

(3)仓库中添加信息

git add 文件名-> 结果:new file：文件名 //工作区到暂存区

git add * 添加所有文件

git commit -m ‘描述信息’ //暂存区到仓库

(4)仓库中修改信息

修改完成后按照原来的程序再执行

(5)删除文件

git rm 文件名 如果想要删除文件夹，则添加参数 -r

git commit -m ‘提交描述’

(6)删除文件夹

当我们需要删除暂存区或分支上的文件, 但本地又需要使用, 只是不希望这个文件被版本控制, 可以使用

git rm -r --cached 文件夹名称

实例: git rm -r --cached target 删除target文件夹

git commit -m '删除了target' 提交,添加操作说明

4. Git远程服务器介绍

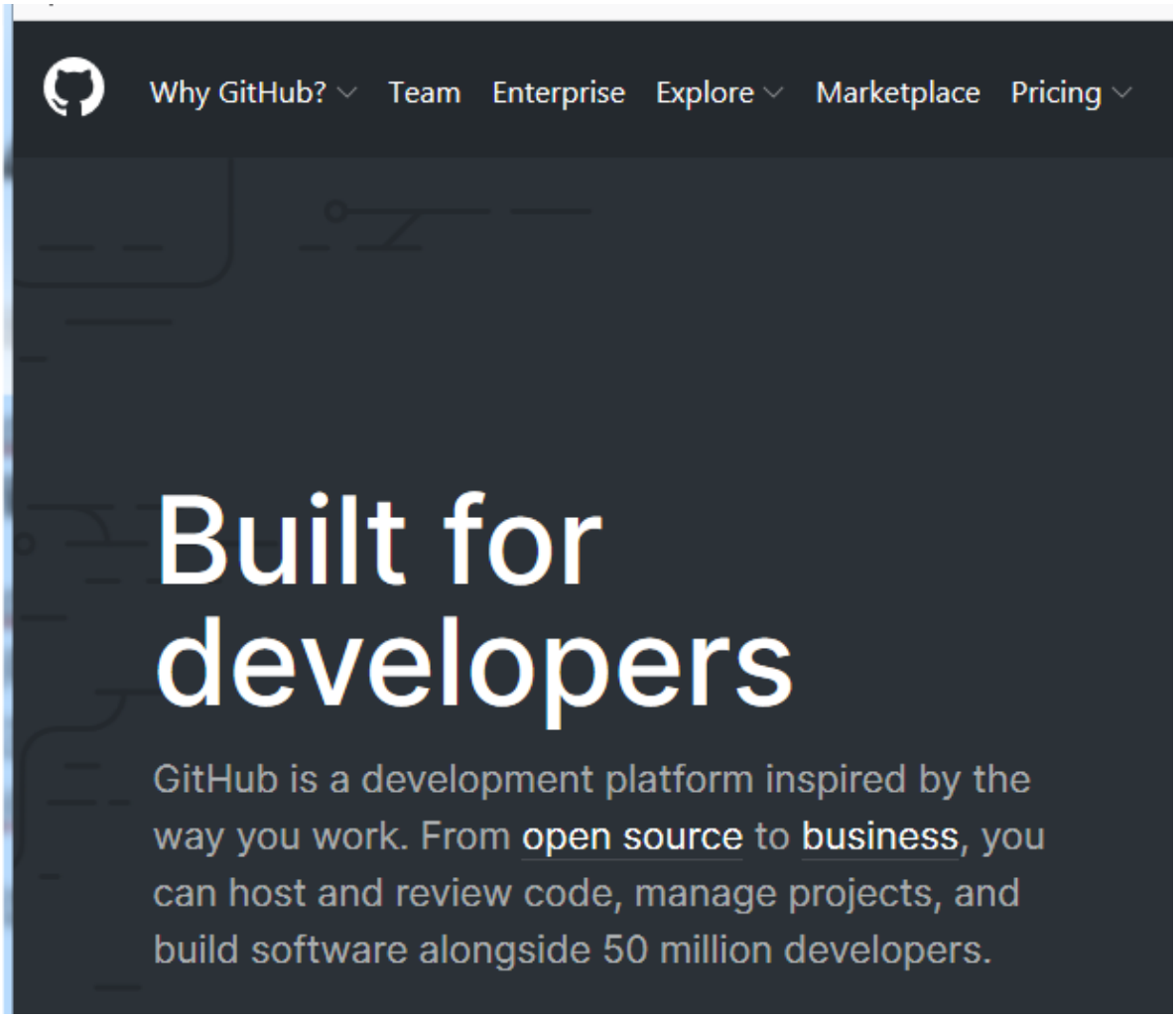
4.1 GitHub介绍

通过git管理github托管项目代码

gitHub是一个面向开源及私有软件项目的托管平台，因为只支持git 作为唯一的版本库格式进行托管，故名gitHub。

gitHub于2008年4月10日正式上线，除了git代码仓库托管及基本的 Web管理界面以外，还提供了订阅、讨论组、文本渲染、在线文件编辑器、协作图谱(报表)、代码片段分享(Gist)等功能。目前，其注册用户已经超过350万，托管版本数量也是非常之多，其中不乏知名开源项目 Ruby，on Rails、jQuery、python等。

访问地址:https://github.com/



4.2.GitLab

GitHub 和 GitLab 都是基于 web 的 Git 仓库，使用起来二者差不多，它们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。

GitHub 作为开源代码库，拥有超过 900 万的开发者用户，目前仍然是最火的开源项目托管平台，GitHub 同时提供公共仓库和私有仓库，但如果使用私有仓库，是需要付费的。

GitLab 解决了这个问题，你可以在上面创建私人的免费仓库。

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 , 它有不少特色：

(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，进一步提升安全性；

(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；

所以，从代码的私有性上来看，GitLab 是一个更好的选择。但是对于开源项目而言，GitHub 依然是代码托管的首选。

访问地址:https://git.lug.ustc.edu.cn/users/sign_in

logo:

4.3 gitee(码云)

码云（Gitee）是 OSCHINA 推出的代码托管协作开发平台，支持 Git 和 SVN,提供免费的私有仓库托管。2016 年推出企业版，提供企业级代码托管服务，成为开发领域领先的 SaaS 服务提供商。

使用 GitHub 时，国内的用户经常遇到的问题是访问速度太慢，有时候还会出现无法连接的情况。如果你希望体验 Git 飞一般的速度，可以使用国内的代码托管与开发协作平台 —— Gitee。除了访问速度更快以外，Gitee 还提供了免费的私有仓库供个人开发者使用。同时，Gitee 也有着国内数一数二的开源生态，这里有非常多的优秀开源项目和开发者，你可以在这里和他们无障碍地沟通交流，不管是找开源项目还是分享自己的开源项目，Gitee 都是极佳的选择。

作为国内代码托管平台的佼佼者，目前已经有超过 500 万名开发者在 Gitee 上托管了 1000 余万个代码仓库，而其提供了研发管理、代码托管、文档管理服务的企业版的服务客户也超过了 10 万家。

访问地址:https://gitee.com/

4.4 注册账号

地址:https://gitee.com/

4.5. 基本概念

(1) 仓库（Repository）

仓库即你的项目，你想在github上开源一个项目，那就必须要新建一个repository,如果你开源的项目多，那你拥有的仓库也就很多

(2) 收藏(star)

仓库主页的star按钮，意思是收藏项目的人数。

(3) 复制克隆项目(fork)

在原项目的基础上新增代码和结构，也可以理解成拿别人的代码进行二次加工。Fork后，会在自己账号下，生成自己的相同仓库

(4) 发起请求(pull request，简称PR)

这个是基于fork的，当其他人改进完代码后，想将这个项目合并到原项目，则这个时候会给你发起一个pull request。如果接受了请求，这个时候就可以拥有改进的项目了

(5) 关注(watch)

即观察，可以随时看到被关注项目的更新

(6) 事务卡片（Issue）

发现代码有bug,但是目前还没成型，需要讨论时使用

当别人发现你的问题时，会提个Issue

(7) Gitee主页

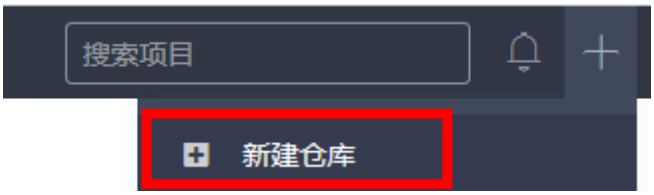
账号创建完后，点击导航栏gitee图标即可进入主页。左侧显示功能列表，右侧显示仓库动态

(8) 仓库主页

仓库主页主要显示项目的信息，如:代码，版本，收藏，关注，fork等

4.6.创建仓库

一个git库对应一个开源项目



仓库主页说明

4.7 仓库管理

新建文件

编辑/删除文件

被删除的文件如何查看信息:

上传文件(可以同时选择多文件):

下载项目:

Gitee issue(问题)

张三发现李四的开源git,提交issue,李四登录gitee后，和张三交流并关闭issue

4.8 基本概念实战

收藏，关注，复制克隆项目,发起请求，事务卡片

删除仓库:

4.9 初始化git的基本信息

设置登录的账户信息: 用户名和邮箱地址是本地git客户端的一个变量，每次commit都会用用户名和邮箱纪录。

设置用户名: git config --global user.name '用户名'

设置用户名邮箱: git config --global user.email '邮箱'

实例: git config --global user.email '123@qq.com'

命令执行位置:桌面右击->git base here

注意:该设置在gitee仓库主页显示谁提交的文件,如果想要修改用户信息，则将该命令再执行一次

查看设置: git config --list

4.10 git管理远程仓库

目的:备份，实现代码共享

实现过程:

客户端:

(1)将本地项目提交到git

(2)建立本地和远程仓库的关系

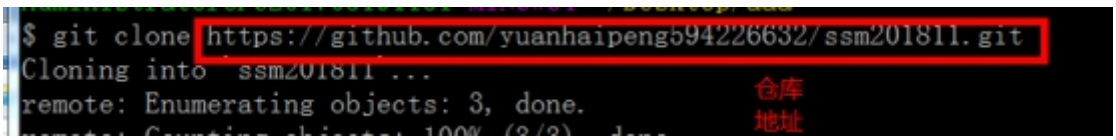
步骤1:

git克隆操作：将远程仓库的项目复制到本地

命令: git clone 仓库地址

仓库地址:

注意:初始化操作使用一次即可



步骤2:

git push 将本地仓库提交到远程（注意先提交到缓存区，再提交到仓库，最后提交远程）

步骤3：要更新你的本地仓库至最新改动，执行：

git pull

从非默认位置更新到指定的url。

git pull http://git.example.com/project.git

实例:

Cd 进入仓库

```
Administrator@PC201703191131 MINGW64 ~/Desktop/bb/rep190123 (master)
$ git add springbootest

Administrator@PC201703191131 MINGW64 ~/Desktop/bb/rep190123 (master)
$ git commit -m 'aaaa111'
On branch master
Your branch is ahead of 'origin/master' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean

Administrator@PC201703191131 MINGW64 ~/Desktop/bb/rep190123 (master)
$ git push
Username for 'https://github.com': yuanhaipeng594226632
Enumerating objects: 24, done.
Counting objects: 100% (24/24), done.
Delta compression using up to 4 threads
Compressing objects: 100% (19/19), done.
Writing objects: 100% (23/23), 6.60 KiB | 675.00 KiB/s, done.
Total 23 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2), done.
To https://github.com/yuanhaipeng594226632/rep190123.git
   a639470..4df4f65  master -> master

Administrator@PC201703191131 MINGW64 ~/Desktop/bb/rep190123 (master)
$
```

4.11 idea+git

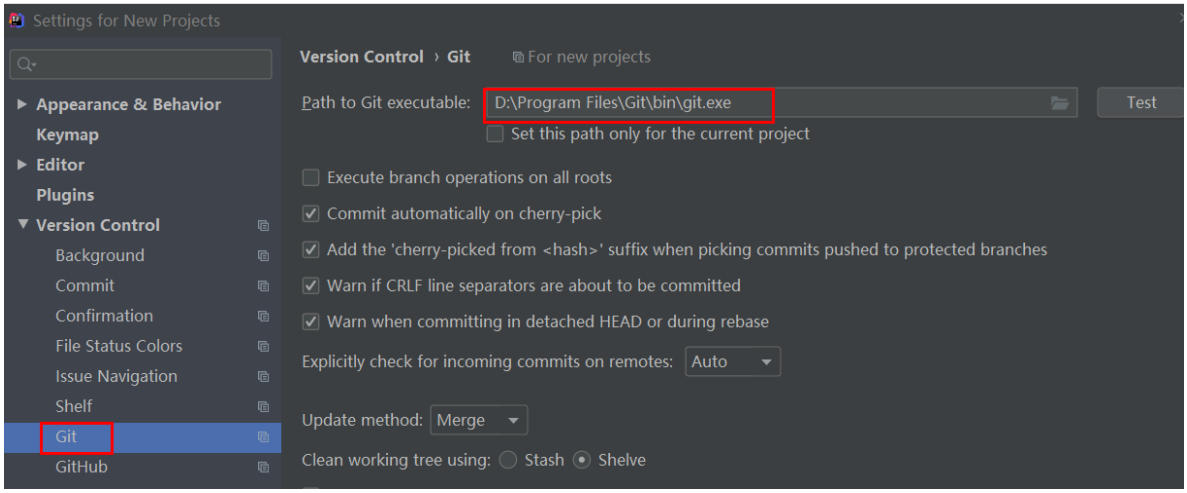
Idea关联git

Idea自身路径需要在英文目录

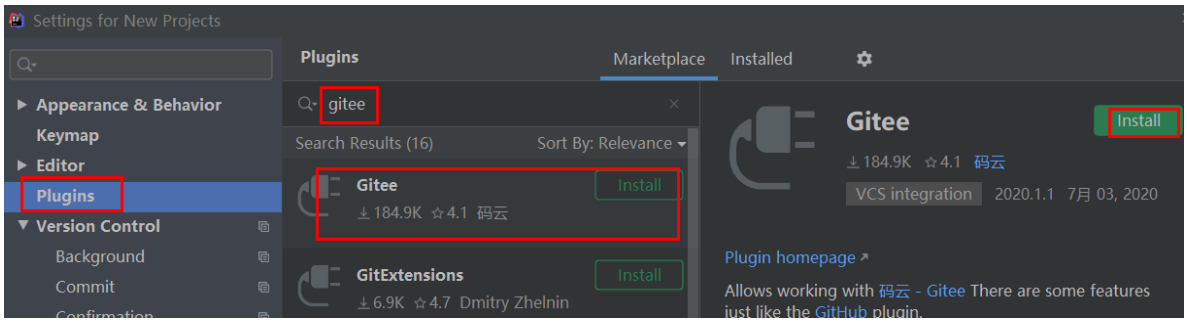
1)Idea配置Git客户端:

File — Settings---Version Control — Git关联Git安装目录下的

bin/git.exe执行文件(这个就是git的客户端指令，类似svn.exe)

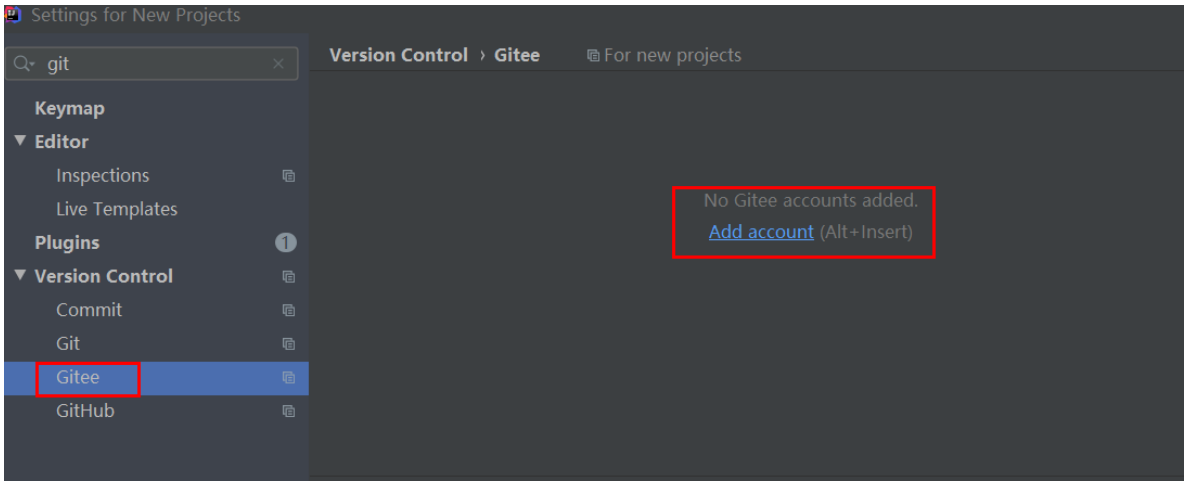


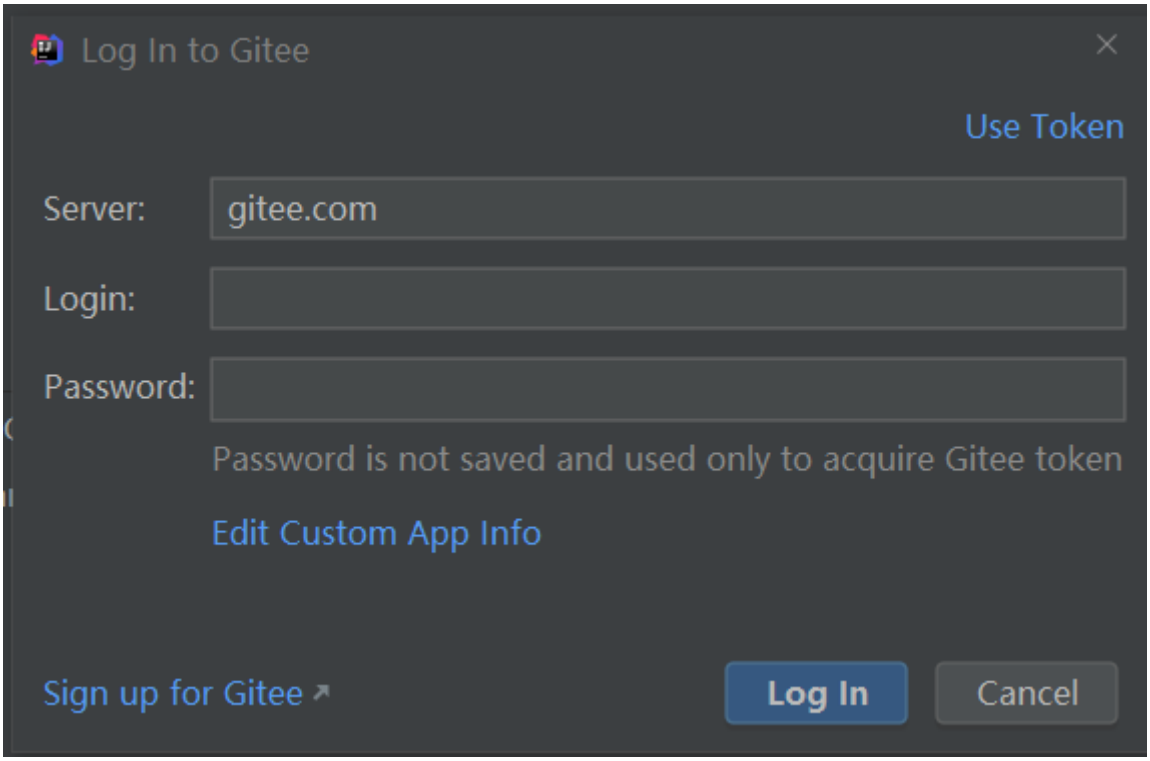
2)下载gitee插件



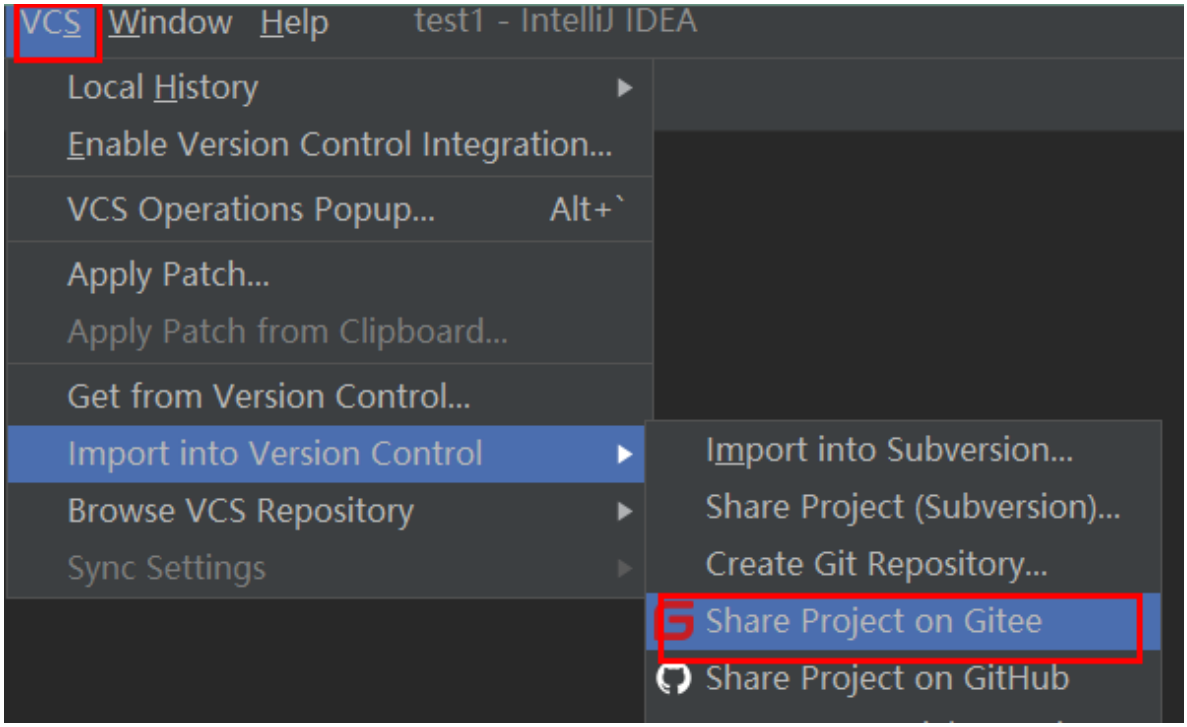
3)添加信息

注意:登录时，使用邮箱登录

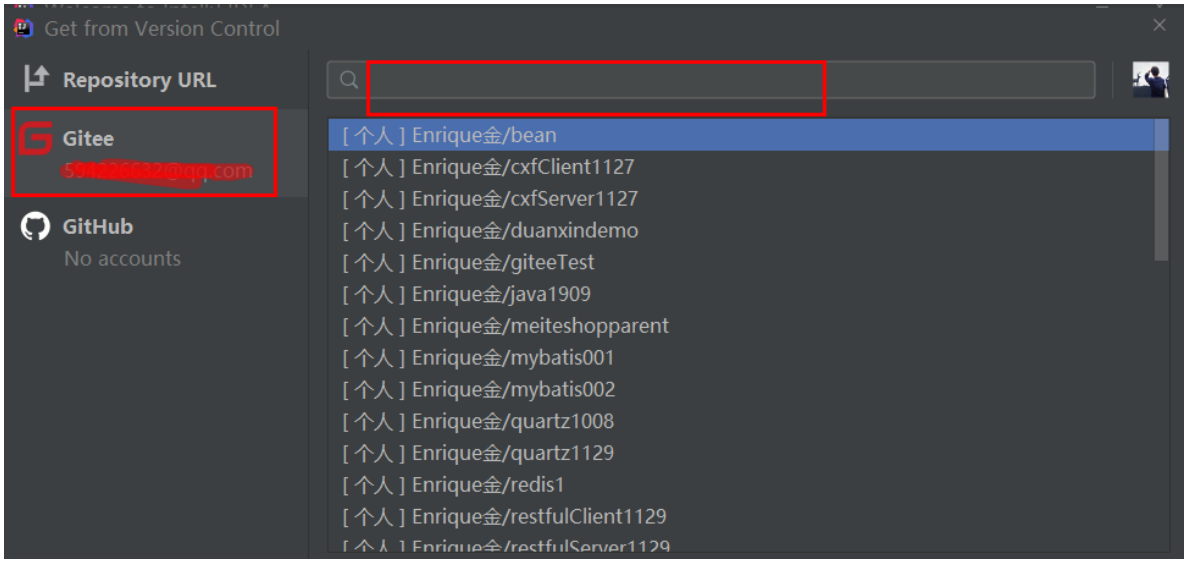




4)本地项目上传到服务器



5)服务器项目下载到本地


Get from Version Control
Repository URL
Gitee
GitHub
No accounts
[个人] Enrique金/bean
[个人] Enrique金/cxfClient1127
[个人] Enrique金/cxfServer1127
[个人] Enrique金/duanxindemo
[个人] Enrique金/giteeTest
[个人] Enrique金/java1909
[个人] Enrique金/meiteshopparent
[个人] Enrique金/mybatis001
[个人] Enrique金/mybatis002
[个人] Enrique金/quartz1008
[个人] Enrique金/quartz1129
[个人] Enrique金/redis1
[个人] Enrique金/restfulClient1129