# Gitblit搭建及Git协作开发流程

版本：v1.0(2015年3月20日)

作者：南瓜剪子

## 概述

目前主流的是git作为自己代码管理，但是采用github需要付费才能够使用，如果不付费，代码需要公开。创业团队及小型开发团队都有必要搭建自己的代码服务器，自己摸索需要一定的时间，会赶不及项目进度。在此作者把自己的经验作为文档写下来，方便互联网各位创业者参考，能很快的搭建自己所需要的代码服务器。

同时作者也把最基本的开发流程，代码提交，代码上传，代码merge等流程用命令写出，方便大家参考。作者自己的团队开发以Maven+Java为主，使用eclipse作为开发工具，相关部分在此文档中都有详细的描述。

文档中使用的工具版本如下

**jdk1.8.0\_73**

**Gitblit 1.7.1**

**Eclipse MARS.1**

**PortableGit 2.7.1**

**Apache-maven-3.3.9**

注意： 由于互联网技术日新月异，以后工具如有更新，流程也许会变化，所以本文档中流程只作为参考。

并请准备一台Windows机器，并且在你的局域网内，开发者能够访问

**注意:** 如果你只是一个开发者，不需要关心Gitblit代码服务器搭建，请跳过下面章节

1. **Gitblit服务器搭建**

**第4章配置Gitblit用户**

**第5章创建第一个Gitblit版本库**

**第7章项目管理者上传项目最初始代码**

这几张是有关Gitblit部分，主要是代码管理员需要学习服务器管理。开发人员请从后面相关部分学习，**后面类容依然精彩**，会告诉你怎么下载代码，构建工程，还有上传代码，更新代码。

## 安装JDK

安装JDK这个过程我就不再过多描述

访问 <http://www.oracle.com/technetwork/java/javase/downloads>

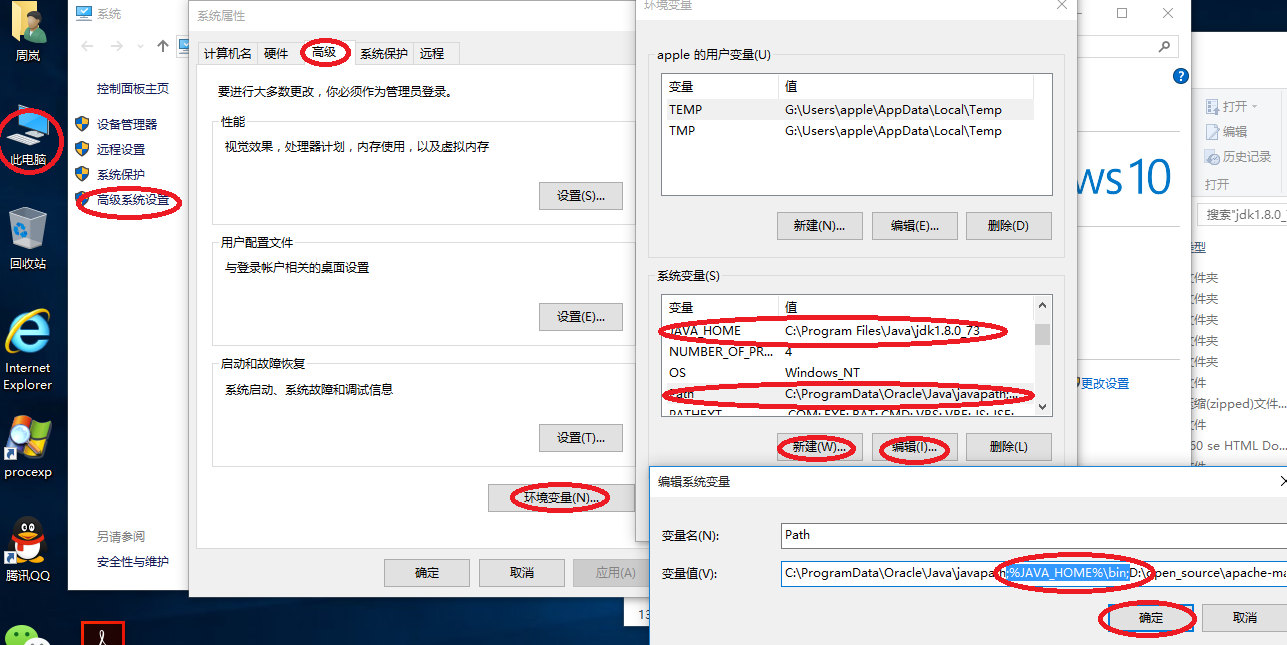
下载最新JDK,并按操作安装

安装完毕后最好设置环境变量JAVA\_HOME及PATH

新建**JAVA\_HOME=C:\Program Files\Java\jdk1.8.0\_73** (你的JDK安装目录，请核对你自己目录及版本)

并把(**;%JAVA\_HOME%\bin;**)加入到**PATH**

大致流程如下图所示



## Gitblit服务器搭建

访问地址：<http://gitblit.com/>

下载最新的Gitblit windows版本，由于小型团队一般以windows机器为主，在这里只介绍windows版本的搭建

Step1) 下载**gitblit-1.7.1.zip**

Step2) 解压到**D:\open\_source\gitblit-1.7.1**

Step3) 创建一个目录，用来存放代码库：**D:/GitRepositoies**

Step4) 修改data目录下的**gitblit.properties**

(**D:\open\_source\gitblit-1.7.1\data\gitblit.properties**)

加入以下红色内容

**#**

**# GITBLIT.PROPERTIES**

**#**

**# Define your custom settings in this file and/or include settings defined in**

**# other properties files.**

**#**

**# Include Gitblit's 'defaults.properties' within your configuration.**

**#**

**# NOTE: Gitblit will not automatically reload "included" properties. Gitblit**

**# only watches the 'gitblit.properties' file for modifications.**

**#**

**# Paths may be relative to the ${baseFolder} or they may be absolute.**

**#**

**# COMMA-DELIMITED**

**# SINCE 1.7.0**

**include = defaults.properties**

**#**

**# Define your overrides or custom settings below**

**#**

**# Base folder for repositories.**

**# This folder may contain bare and non-bare repositories but Gitblit will only**

**# allow you to push to bare repositories.**

**# Use forward slashes even on Windows!!**

**# e.g. c:/gitrepos**

**#**

**# SINCE 0.5.0**

**# RESTART REQUIRED**

**# BASEFOLDER**

**git.repositoriesFolder = D:/GitRepositoies**

**# Standard http port to serve. <= 0 disables this connector.**

**# On Unix/Linux systems, ports < 1024 require root permissions.**

**# Recommended value: 80 or 8080**

**#**

**# SINCE 0.5.0**

**# RESTART REQUIRED**

**server.httpPort = 1000**

**# Secure/SSL https port to serve. <= 0 disables this connector.**

**# On Unix/Linux systems, ports < 1024 require root permissions.**

**# Recommended value: 443 or 8443**

**#**

**# SINCE 0.5.0**

**# RESTART REQUIRED**

**server.httpsPort = 1001**

**# Specify the interface for Jetty to bind the standard connector.**

**# You may specify an ip or an empty value to bind to all interfaces.**

**# Specifying localhost will result in Gitblit ONLY listening to requests to**

**# localhost.**

**#**

**# SINCE 0.5.0**

**# RESTART REQUIRED**

**server.httpBindInterface =**

**# Specify the interface for Jetty to bind the secure connector.**

**# You may specify an ip or an empty value to bind to all interfaces.**

**# Specifying localhost will result in Gitblit ONLY listening to requests to**

**# localhost.**

**#**

**# SINCE 0.5.0**

**# RESTART REQUIRED**

**server.httpsBindInterface =**

注意：红色部分属性都是从同目录下**defaults.properties**拷贝过来，只是采用了我们自己的配置，其中为了安全及便利，我没有没有使用端口 80及443，用1000及1001代替(因为我团队的机器80端口用作其他用途了)。并把仓库设置为Step3中的目录。

Step5) 修改**D:\open\_source\gitblit-1.7.1\installService.cmd**下面红色部分

**@REM Install Gitblit as a Windows service.**

**@REM gitblitw.exe (prunmgr.exe) is a GUI application for monitoring**

**@REM and configuring the Gitblit procrun service.**

**@REM**

**@REM By default this tool launches the service properties dialog**

**@REM but it also has some other very useful functionality.**

**@REM**

**@REM http://commons.apache.org/daemon/procrun.html**

**@REM arch = x86, amd64, or ia32**

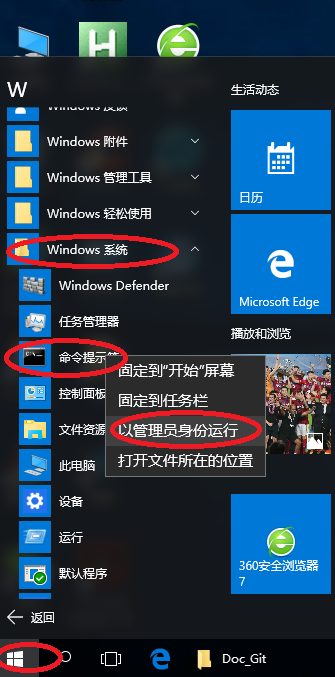
**SET ARCH=x86**

由于我的机器是x86-32位的，所以我改为**x86**,请对照你机器的架构选择

如果你的机器是64位，请选择**amd64**

Step6) 安装windows服务

用管理员权限打开CMD命令提示符,如下图所示



然后再命令行中输入一下命令

**C:\Windows\system32>d:**

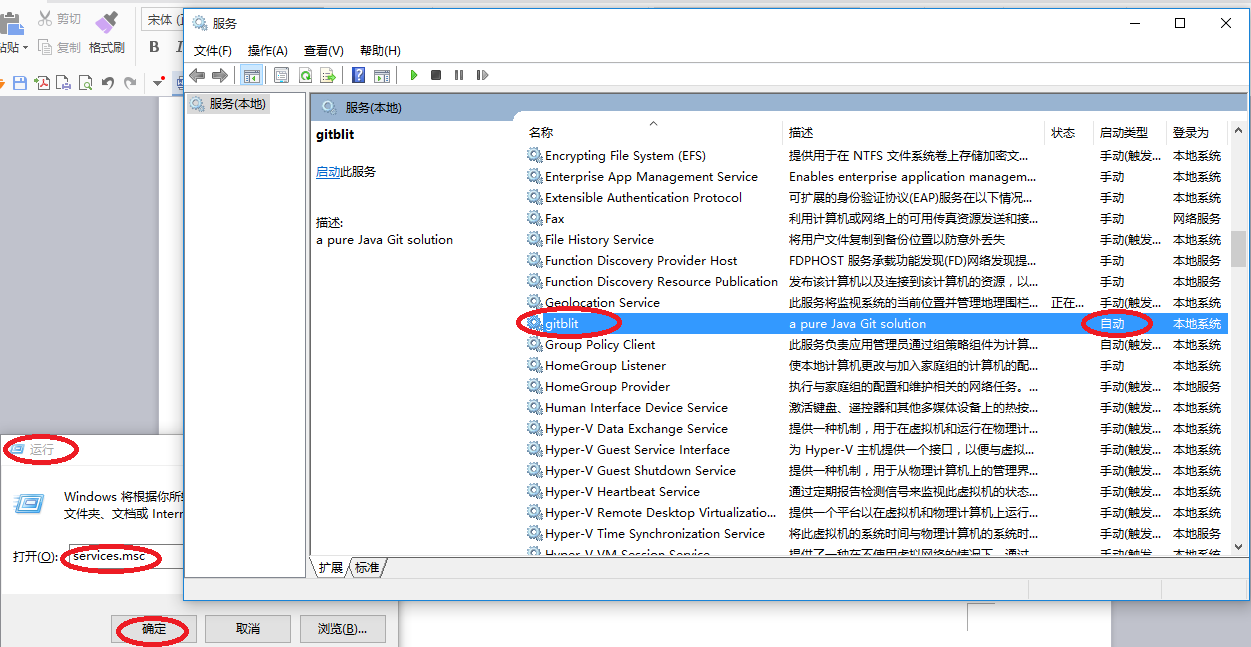
**D:\>cd D:\open\_source\gitblit-1.7.1**

**D:\open\_source\gitblit-1.7.1>installService**

Step7) 检查windows服务

开始菜单“**运行**”中，输入**services.msc**

并检查gitblit服务是否成功安装，由于开发团队每天都在用，确保为自动模式，每次windows启动都自动启动



右键点击gitblit, 并把**服务启动**起来



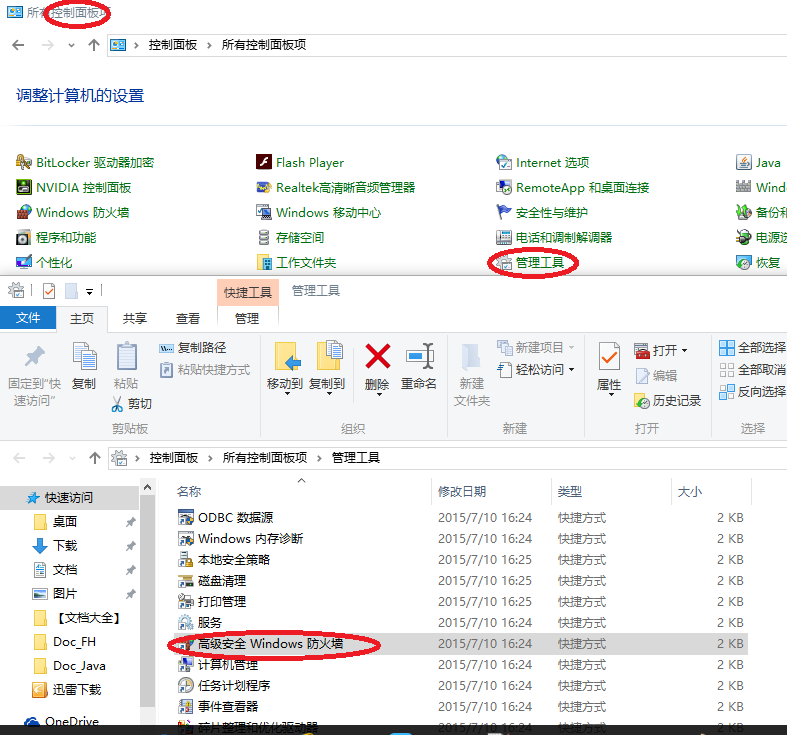
注意：如果要删除服务，请直接运行**D:\open\_source\gitblit-1.7.1**下面的**uninstallService.cmd**

Step8) 有可能服务器起来后，局域网其他机器无法访问，只能在本机浏览器中打开，这是由于防火墙的设置，你必须人工设置防火强规则。

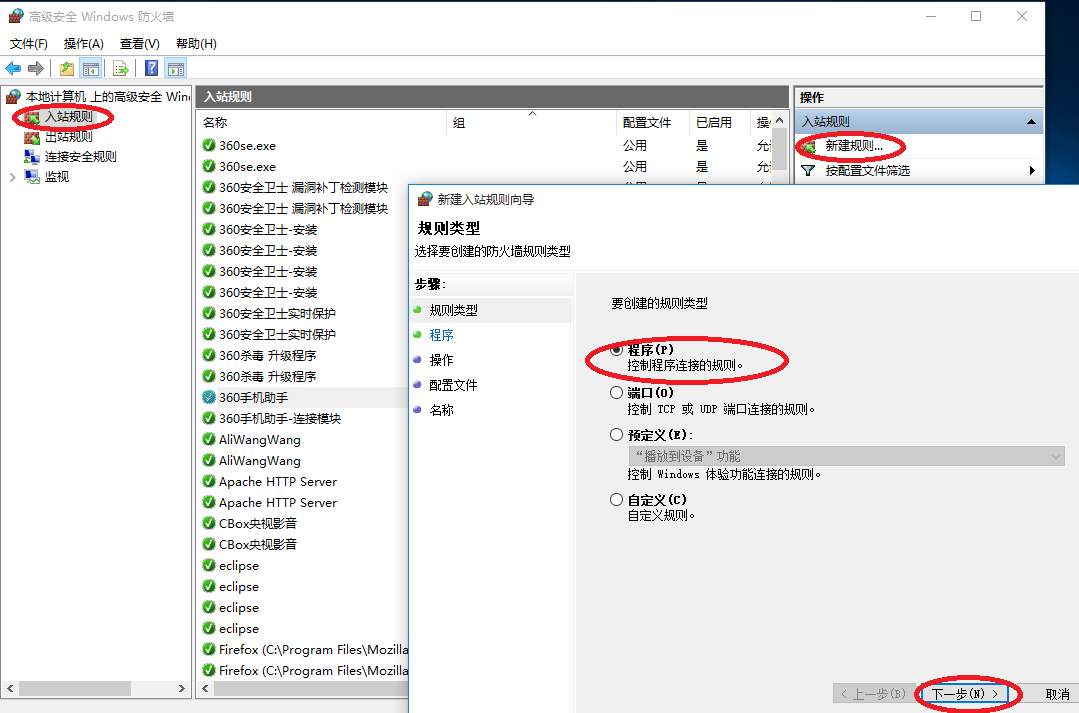
打开“**控制面板**”->“**管理工具**”->“**高级安全 Windows 防火墙**”

由于我演示用的机器是windows10,也许名字会不对，请点击“管理工具”里面的有关“防火墙”选项就行

如下图所示：

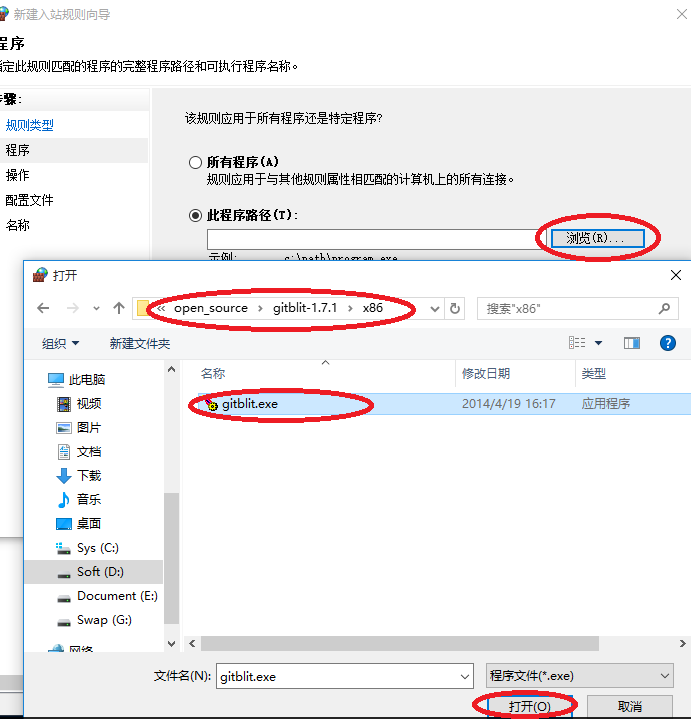


然后“**新建规则**”，选择“**程序**”

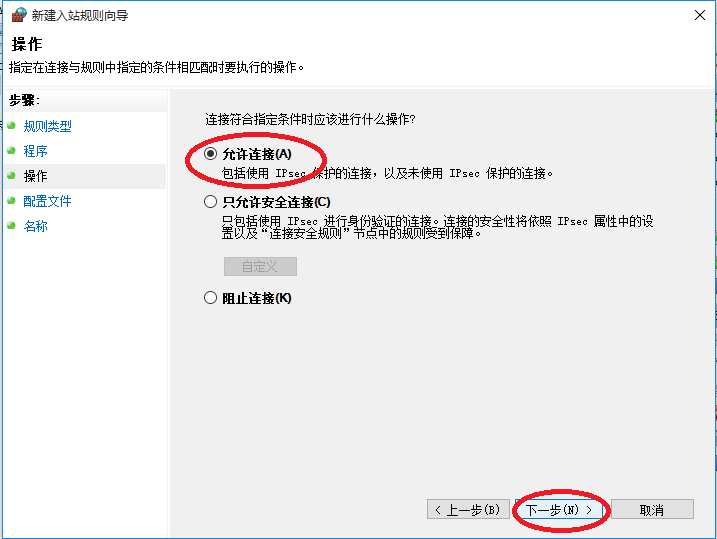


然后选择程序为**D:\open\_source\gitblit-1.7.1\x86\gitblit.exe**

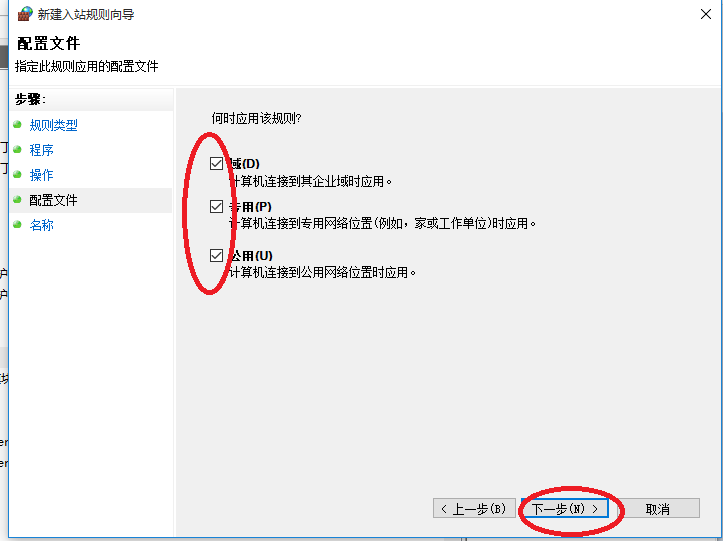
注意这里”**x86**”是你在step5)中选择的架构



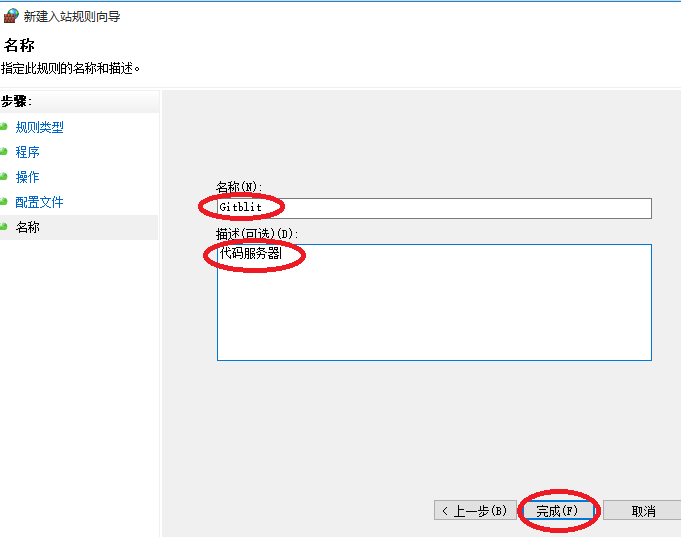
然后点下一步



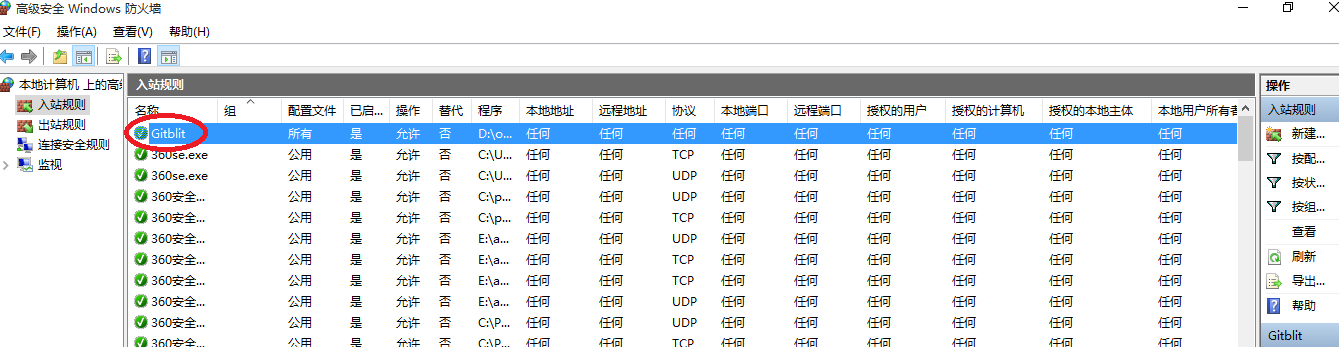
然后点下一步：



点下一步：

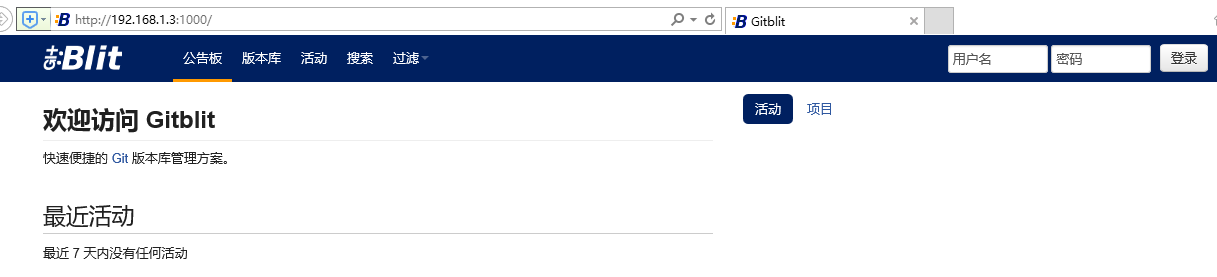


然后会看到有下面一条记录



Step8) 然后可以在IE中输入**http://192.168.1.3:1000/**

其中IP地址换成你服务器的地址，注意，这里我们使用端口为1000，在前面gitblit.properties里面配置



Gitblit安装完毕

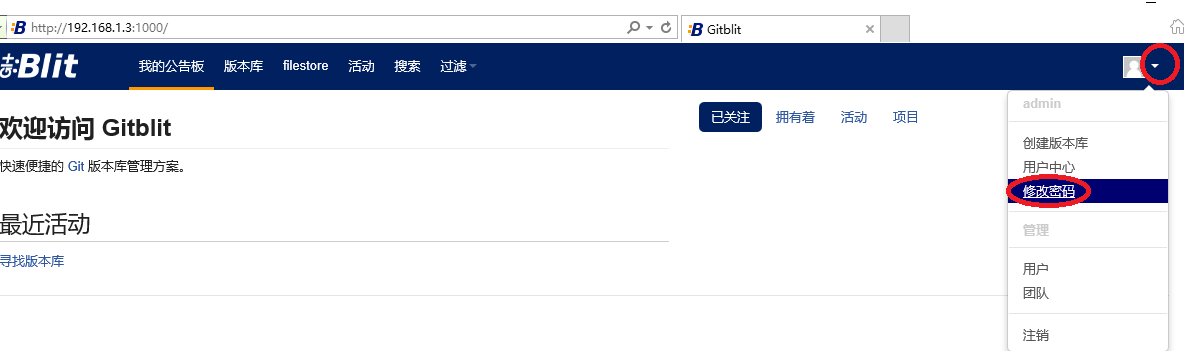
## 配置Gitblit用户

用IE登陆系统： **<http://192.168.1.3:1000/>**

Step1) 用管理员账号登录

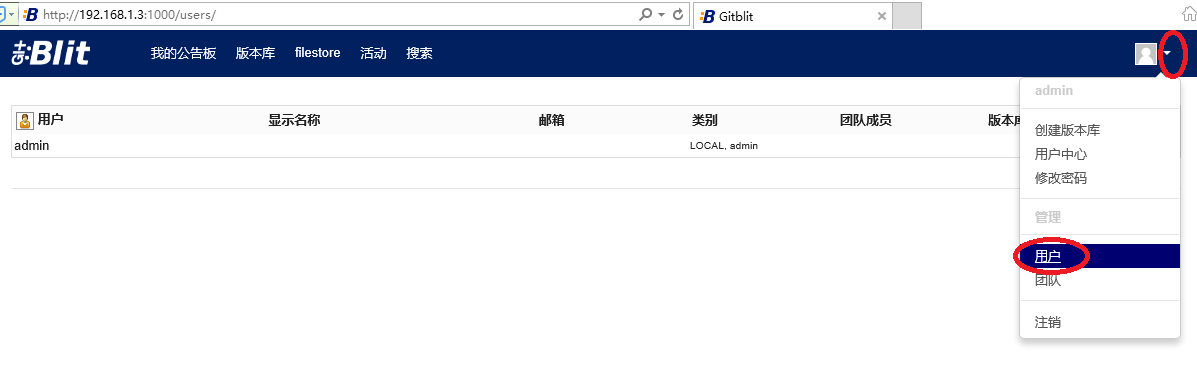
确实用户名/密码： admin/admin

Step2) 修改管理员账户密码



Step3) 添加用户

为你团队中的开发人员创建用户，只有授权用户才能进行相关操作



然后点击添加用户



然后填入成员基本信息，点击保存



然后把用户名及初始密码发送给团队成员，让他们登陆**http://192.168.1.3:1000/**进去修改密码

Step4) 创建项目组

也许你的创业团队有好几个项目，每个项目都由不同的人开发，你可以为每个项目创建个团队。

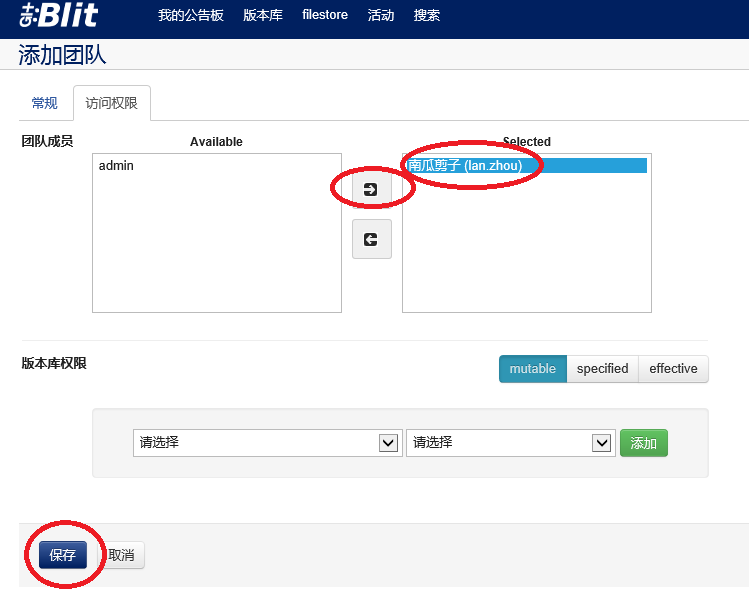


然后点击右上交“**添加团队**” 并填写以下信息:

例如我有一个项目名称叫“阿尔法猫”



然后选择团队成员，当前只有我一个，也可以选择多个，如下图



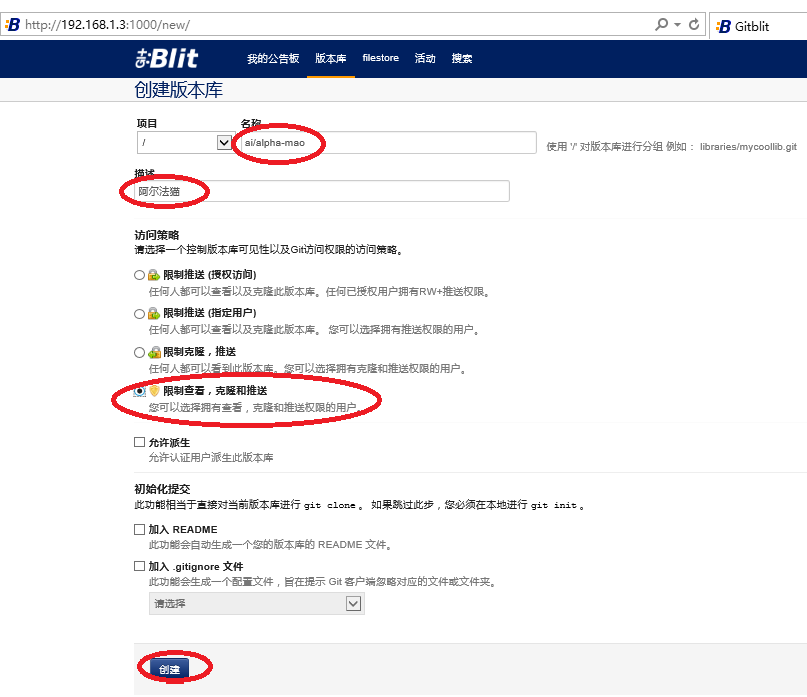
其他有关权限部分可以先不用关心，熟悉后请自行阅读<http://gitblit.com/>中文档

## 创建第一个Gitblit版本库： 阿尔法猫

Step1) 使用系统管理员账号登陆，并点击“创建版本库”



Step2) 然后按要求填入以下信息



可以给项目分组，那么在名称哪里填写：ai/alpha-mao

表示项目alpha-mao属于ai分组，

对于项目名称我们一般采用全小写，中间用分隔符“-”分离单词

注意，为了保证代码的安全，我们选择第四个： **限制查看，克隆及推送**，只有加入项目组的人才能够查看项目。 不像github，匿名用户可以查看克隆开源代码，一般创业团队的代码都是保密。

其他信息可以不用填写，下面会告诉你怎么推送新代码到服务器。

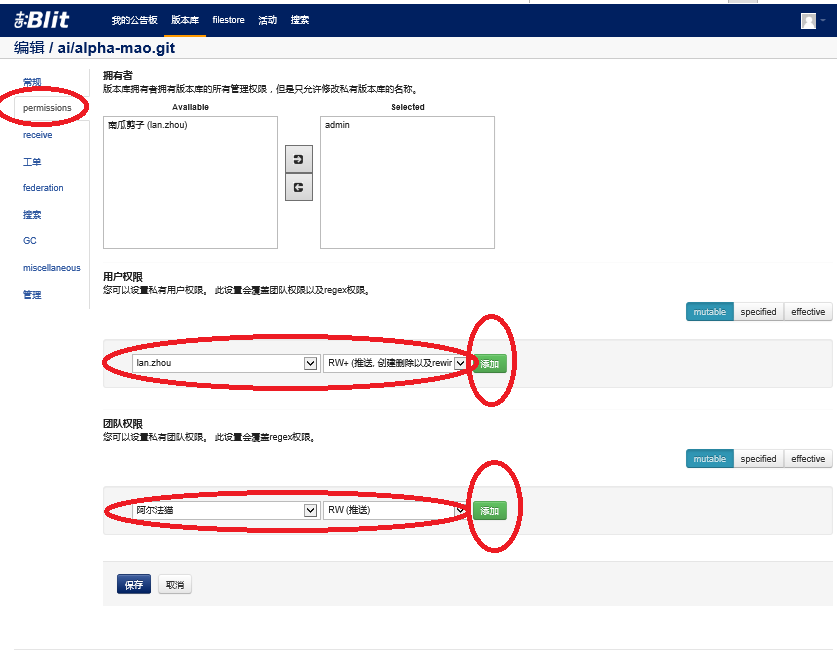
然后点击下面“创建”，

Step3) 设置项目权限

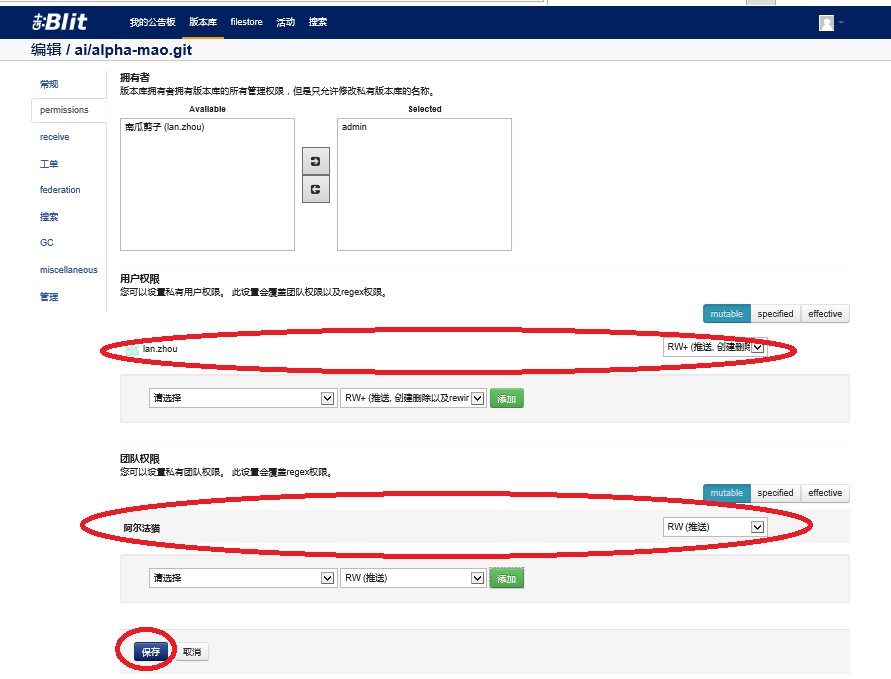
当项目创建后，点击版本库又上角的“编辑”按钮



然后设置团队及用户权限



注意：请一定要点击“添加”按钮



添加成功后，点击保存按钮

注意，团队权限只有能够查看克隆推送的权限，但是没有创建，修改分支(branch)的权限

我们这里设置了一个项目管理员：他除了有查看克隆推送的权限，还有创建分支的权限。

在后面我们会看到lan.zhou这个用户能够直接在服务器上创建**dev**分支。

到此我们创建了一个新的版本库，下面等待lan.zhou这个用户上传最初始的代码，然后团队里面的别人就可以克隆使用。

## 下载最新的git工具并安装

访问<https://git-scm.com/download/win>

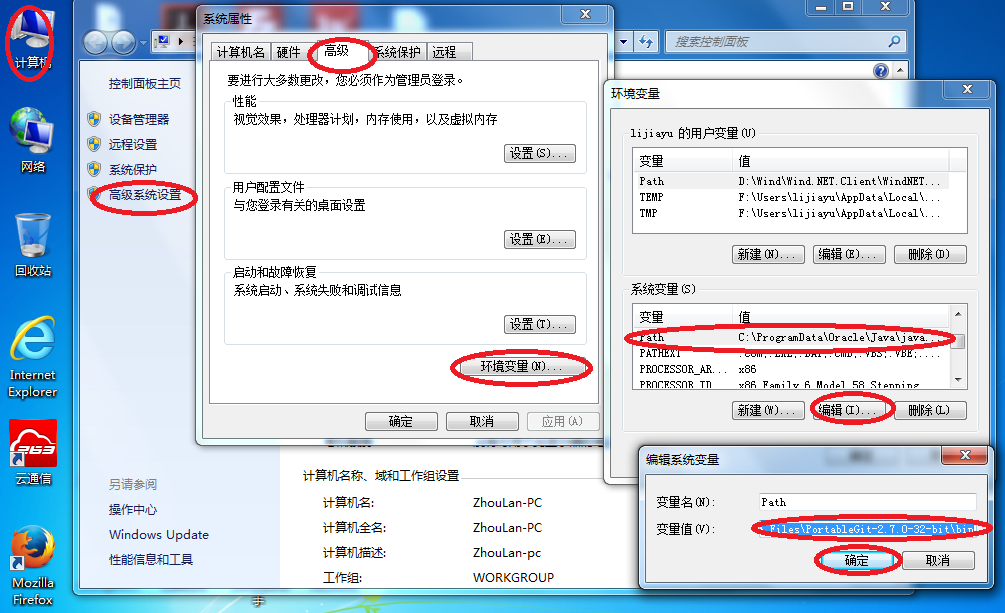
下载最新的git工具

一般“**[32-bit Git for Windows Portable](https://github.com/git-for-windows/git/releases/download/v2.7.2.windows.1/PortableGit-2.7.2-32-bit.7z.exe)”**就行，一般国外服务器比较难下载，可以找国内镜像。

安装步骤：

step1) 解压PortableGit-2.7.0-32-bit.7z.exe

step2) 按下图把bin目录(**;D:\Program Files\PortableGit-2.7.0-32-bit\bin**)加在系统环境变量PATH下



step3) 打开一个cmd(命令提示符)

运行下列命令

**git config --global user.email "nanguajianzi@163.com"**

**git config --global user.name "lan.zhou"**

注意：请把红色的用户名换成你自己的登陆名(详见第一章)

然后用下列命令检查是否配置成功：

**git config --list**

**core.symlinks=false**

**core.autocrlf=true**

**color.diff=auto**

**color.status=auto**

**color.branch=auto**

**color.interactive=true**

**pack.packsizelimit=2g**

**help.format=html**

**http.sslcainfo=/ssl/certs/ca-bundle.crt**

**diff.astextplain.textconv=astextplain**

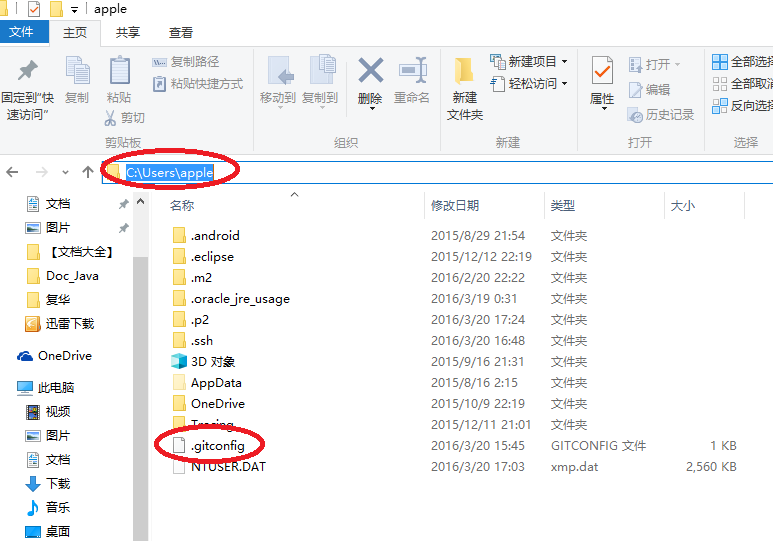
**rebase.autosquash=true**

**user.name=lan.zhou**

**user.email=nanguajianzi@163.com**

**core.editor='C:/Program Files/Notepad++/notepad++.exe' -multiInst -notabbar -nosession -noPlugin**

运行完上面两个config命令，在你的windows用户目录下(C:\Users\登陆用户名)就会生成一个文件**.gitconfig**



里面包含下面内容：

**[user]**

**email = nanguajianzi@163.com**

**name = lan.zhou**

## 项目管理者上传项目最初始代码

项目管理者请准备最初的一个项目，里面包含一些基本代码

请参考下面我准备的例子



下面进行本地的准备

Step1) 把上面文件点击右键“**保存到文件**”，然后解压到文件夹，比如

**D:\git\alpha-mao**

Step2) 打开一个windows命令提示符，运行下列命令

**C:\Users\apple>d:**

**D:\>cd D:\git\alpha-mao**

进入项目目录

Step3) 进行git 初始化

**D:\git\alpha-mao>git init**

**初始化空的 Git 版本库于 E:/apple/Documents/git/alpha-mao/.git/**

Step4) 把文件加入版本

**D:\git\alpha-mao>git add .gitignore**

**D:\git\alpha-mao>git add LICENSE**

**D:\git\alpha-mao>git add README.md**

**D:\git\alpha-mao>git add pom.xml**

**D:\git\alpha-mao>git add src/**

Step5) 第一次commit

**D:\git\alpha-mao>git commit -m "first commit"**

到此，项目在你本地已经准备好了，但是还没有上传到建立好的Gitblit服务器。

下面我们上传代码

Step1) 查看你的上传地址，

用项目管理者(前面建立的用户：lan.zhou，并不是系统管理员admin)登陆

<http://192.168.1.3:1000>

Step2) 查看刚才建立的新版本库



点击图中红圈标出的“复制到剪贴板”

这个就是用户lan.zhou对应的git服务器地址

Step3) 上传代码

在刚才的命令行中运行

**D:\git\alpha-mao>git remote add origin ssh://lan.zhou@192.168.1.3:29418/ai/alpha-mao.git**

**D:\git\alpha-mao>git push -u origin master**

**The authenticity of host '[192.168.1.3]:29418 ([192.168.1.3]:29418)' can't be established.**

**RSA key fingerprint is SHA256:yQE8FoZXgbMUR1ou9zl+CKLE9xqM1zwy09wU33A4CeM.**

**Are you sure you want to continue connecting (yes/no)? yes**

**Warning: Permanently added '[192.168.1.3]:29418' (RSA) to the list of known hosts.**

**Password authentication**

**Password:**

**Counting objects: 19, done.**

**Delta compression using up to 4 threads.**

**Compressing objects: 100% (9/9), done.**

**Writing objects: 100% (19/19), 5.60 KiB | 0 bytes/s, done.**

**Total 19 (delta 0), reused 0 (delta 0)**

**remote: Updating references: 100% (1/1)**

**To ssh://lan.zhou@192.168.1.3:29418/ai/alpha-mao.git**

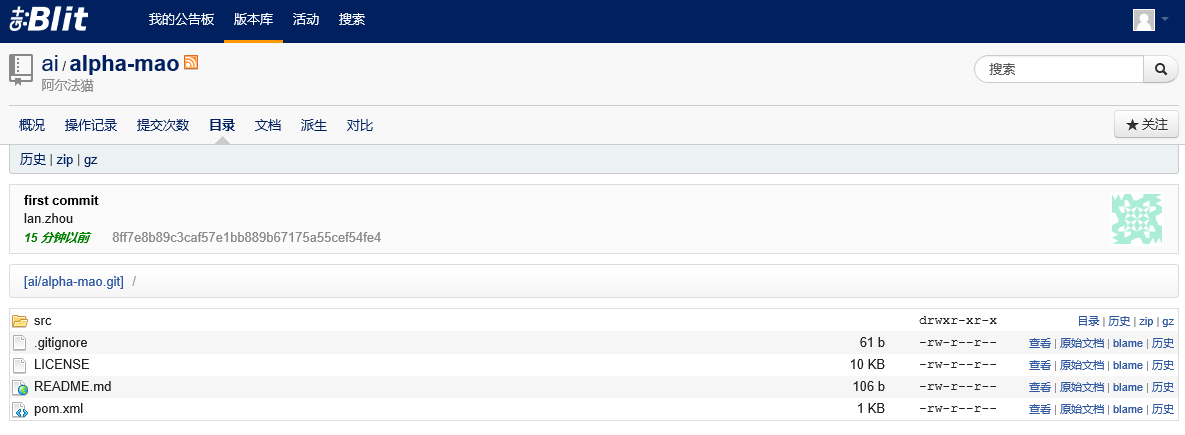
**\* [new branch] master -> master**

**分支 master 设置为跟踪来自 origin 的远程分支 master。**

在上图红色的部分先输入：yes,然后再输入你在Gitblit上设置的密码

到此，代码已经上传到服务器

Step4) 登陆到Gitblit可以看到相关代码



下面为了项目开发，我们一般务必要做两个事请

Step1) 创建初始tag，并上传到服务器

**D:\git\alpha-mao> git tag -a v0.1 -m "initial version"**

**D:\git\alpha-mao> git push origin v0.1**

然后你在 <http://192.168.1.3:1000> 上能够看到v0.1这个tag

Step2) 创建服务器的dev分支

一般master分支为稳定代码，小型团队平常开发一般在另一个分子dev上进行

我们必须在远端Gitblit服务器上创建dev分支。

由于Gitblit权限控制， 只有授权的人才能创建分子，详见**第4章配置Gitblit用户**。在这里lan.zhou这个用户拥有这个权限

运行下列命令在远端Gitblit服务器创建dev分支：

**D:\git\alpha-mao>git push origin master:dev**

Step3) 用下列命令查看分支

查看本地分支：

**D:\git\alpha-mao>git branch**

**\* master**

查看远端分支：

**D:\git\alpha-mao>git branch -r**

**origin/dev**

**origin/master**

Git环境搭建完毕，下面开始进入第二部分，开发部分

## 本地开发环境: eclipse

请到地址：<http://www.eclipse.org/downloads/>

下载最新版本的：

### [Eclipse IDE for Java EE Developers](http://www.eclipse.org/downloads/packages/eclipse-ide-java-ee-developers/mars2" \o "Tools for Java developers creating Java EE and Web applications, including a Java IDE, tools for Java EE, JPA, JSF, Mylyn, EGit and others.)

然后直接解压缩，右键点击里面的“eclipse.exe”在桌面创建快捷方式，

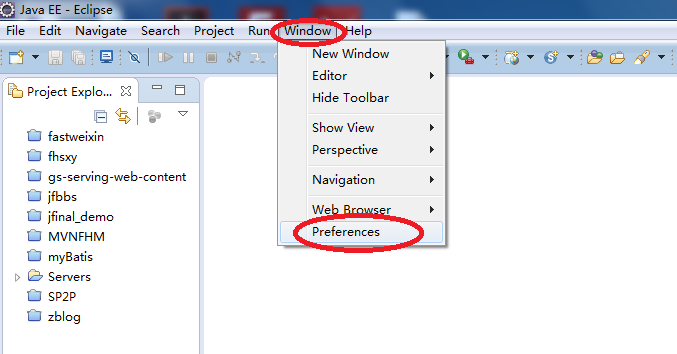
以后在桌面直接点击就可运行。

在本文中我们使用 MARS.1版本

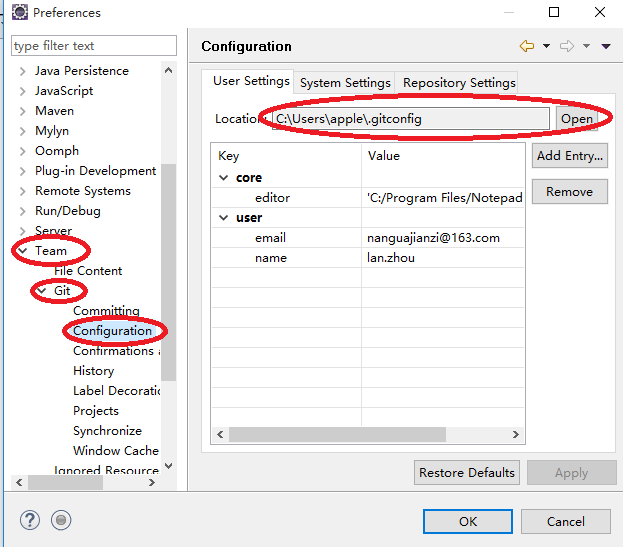
## 配置eclipse 的git功能

步骤

Step1)点击eclipse下列菜单



Step 2) 进入git配置选项卡



Step 3) 在上图中的location，点击Open,然后选中你的git配置文件: **C:\Users\apple\.gitconfig**

注意：其中apple是你的机器用户名

Step 4), 检查你的用户名与邮箱是否正确

## Eclipse中配置maven

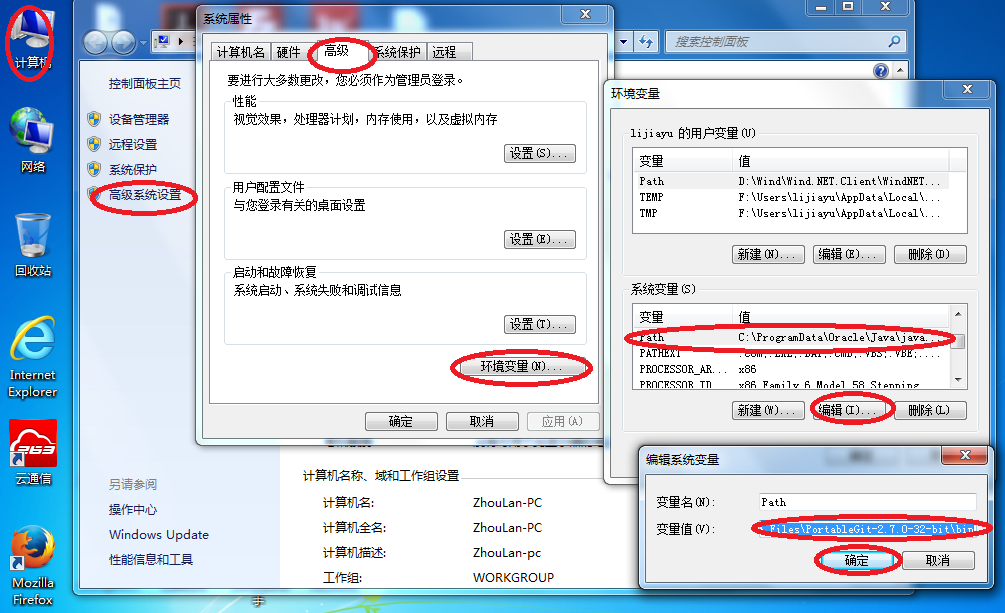
Step1) 下载maven

<http://maven.apache.org/download.cgi>

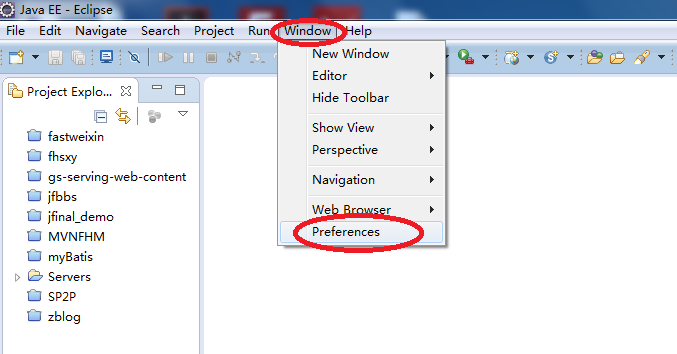
比如：apache-maven-3.3.9-bin.tar.gz

Step2) 同样解压后直接使用

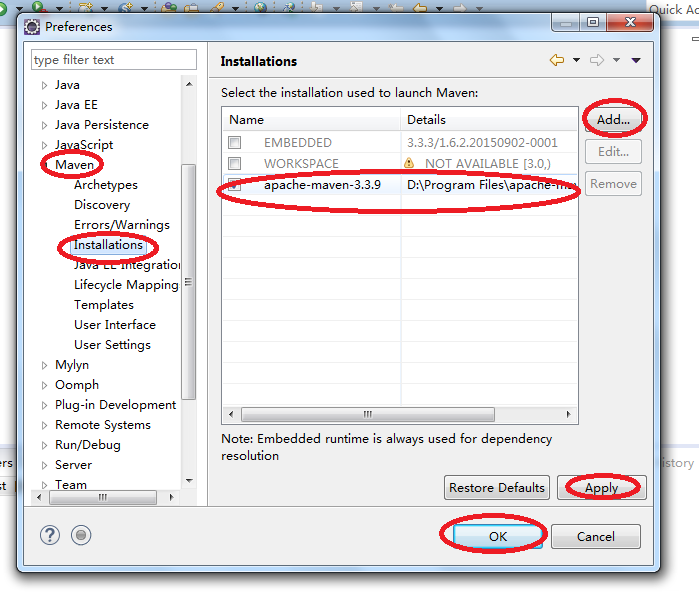
同时把bin目录(**;D:\Program Files\apache-maven-3.3.9\bin**)加入PATH目录



Step3) 设置eclipse

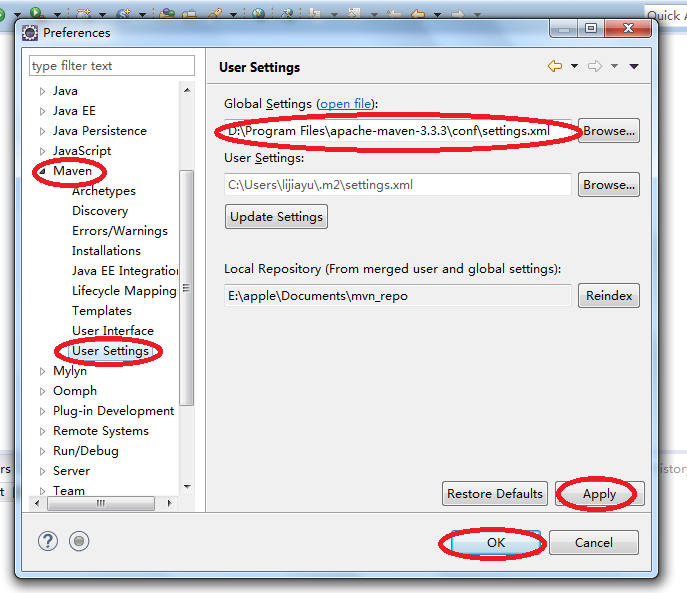


然后把刚才解压的目录加入



Step4) 选择配置文件，一定要选择你解压的maven目录下

**D:\Program Files\apache-maven-3.3.3\conf\settings.xml**



Step5) 更新配置文件

在文件D:\Program Files\apache-maven-3.3.3\conf\settings.xml

中加入下列**红色**的语句

**<mirrors>**

**<!-- mirror**

**| Specifies a repository mirror site to use instead of a given repository. The repository that**

**| this mirror serves has an ID that matches the mirrorOf element of this mirror. IDs are used**

**| for inheritance and direct lookup purposes, and must be unique across the set of mirrors.**

**|**

**<mirror>**

**<id>mirrorId</id>**

**<mirrorOf>repositoryId</mirrorOf>**

**<name>Human Readable Name for this Mirror.</name>**

**<url>http://my.repository.com/repo/path</url>**

**</mirror>**

**-->**

**<mirror>**

**<id>CN</id>**

**<name>OSChina Central</name>**

**<url>http://maven.oschina.net/content/groups/public/</url>**

**<mirrorOf>central</mirrorOf>**

**</mirror>**

**</mirrors>**

在这里我们引入了中国区的maven库，可以极大提高你的获取速度。

同时注意，你可以选择另外的目录作为maven库

**<localRepository>E:\apple\Documents\mvn\_repo</localRepository>**

这个目录可能会很大，选择合适的磁盘存放

## 团队其他成员在服务器端查看工程

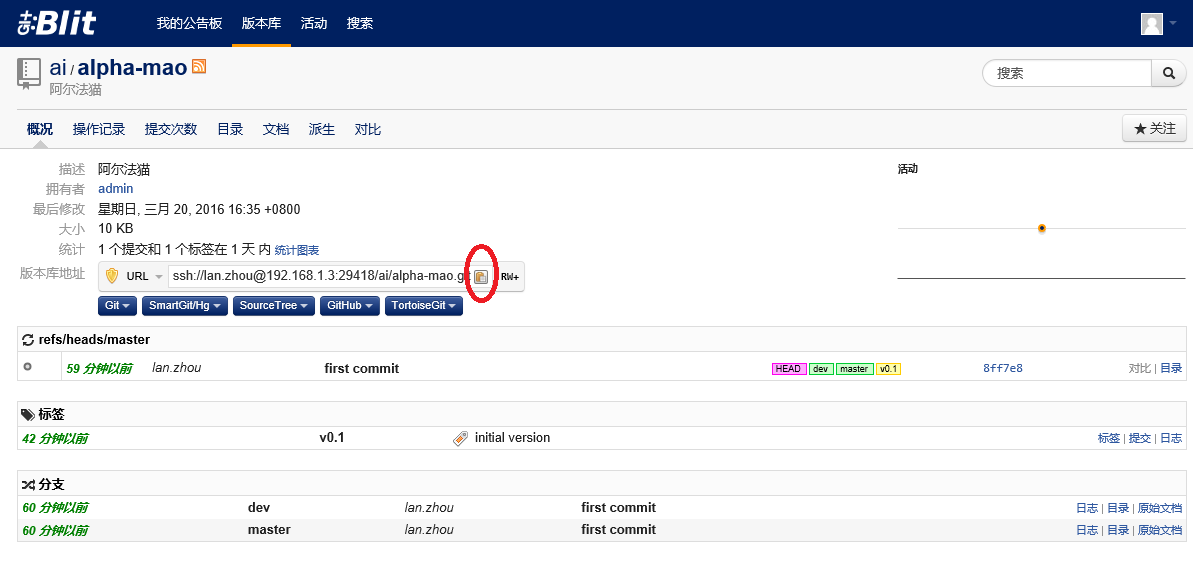
团队其他成员，这里我依然还是已自己(lan.zhou)为例，但是这里可以是属于项目阿尔法猫团队其他任何成员。

**请用浏览器访问地址：**<http://192.168.1.3:1000/>

Step1) 请点击下面的阿尔法猫项目



**Step2) 请取得项目的地址，点击下图中红色的标记出来的“复制到剪贴板”**



**你同时会看到我已经新建了两个分支：**

**master(主分支，用来存储当前稳定的代码）**

**dev(开发分支,我们平时每天用来提交代码的分支，如果多人同时提交，可能需要merge)**

**注意：不允许直接向master提交任何代码，只有管理员（余强）才能整合dev代码到master**

**同时你还会看到我已经建立一个第一个版本标签 v0.1，这是架构代码**

**我们将会很快开发v1.0版本**

## 团队其他成员第一次下载代码

Step1) 在你的电脑上建立一个目录，比如**E:\apple\Documents\git**

Step2) 打开一个命令行cmd，并运行下列两个命令

**C:\Users\apple>e:**

**E:\>cd apple\Documents\git**

进入**E:\apple\Documents\git**

Step3) 接着运行下列命令

**E:\apple\Documents\git> git clone ssh://lan.zhou@192.168.1.3:29418/ai/alpha-mao.git**

注意：真个连接是你在上面章节中step2中拷贝得到的链接，请完整复制，不用使用别人的链接，里面包含你的用户名

**然后进入项目目录**

**E:\apple\Documents\git>cd alpha-mao**

Step4) 可以熟悉简单的git命令

**> git status （查看当前你修改后还没有提交的代码）**

**On branch master**

**Your branch is up-to-date with 'origin/master'.**

**nothing to commit, working directory clean**

**> git branch （查看你本地的分支）**

**\* master**

**> git branch –r （查看远端服务器的分支）**

**origin/HEAD -> origin/master**

**origin/dev**

**origin/master**

注意，在服务器中我们已经有了origin/dev分支，我们将用这个分支进行开发

Step5) 创建你自己的本地开发分支

比如你分配到的任务是feature1, 或者你修改bug5540

你可以在你的本地创建名字为feature1或者bug5540的分支

但是对于我们一般代码开发，可以统一使用**localdev**

**> git branch localdev**

然后可以查看新建的localdev分支

**> git branch （查看你本地的分支）**

**localdev**

**\* master**

但是当前的工作分支仍然是master, 前面有一个\*号

Step 6) check out 当前的 localdev 分支

**> git checkout localdev**

**Switched to branch 'localdev'**

**> git branch （查看你本地的分支）**

**\* localdev**

**Master**

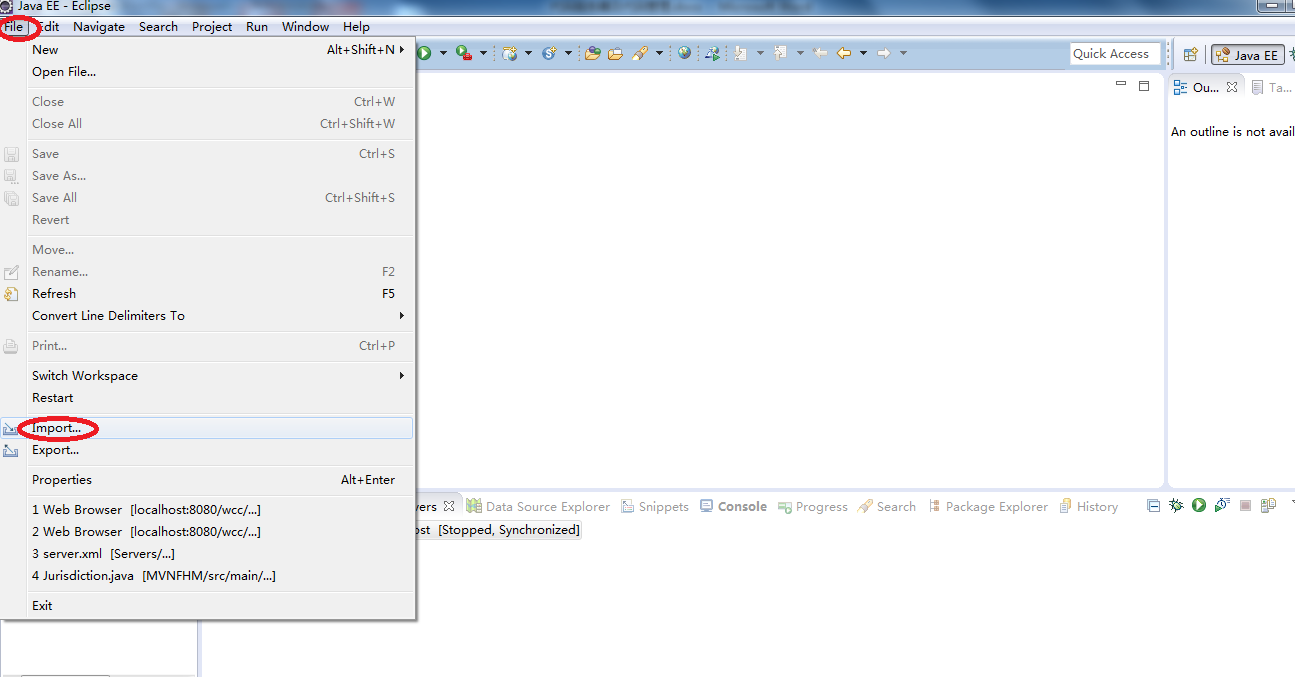
Step 7) merge remote dev分支到本地localdev分支

**> git merge origin/dev**

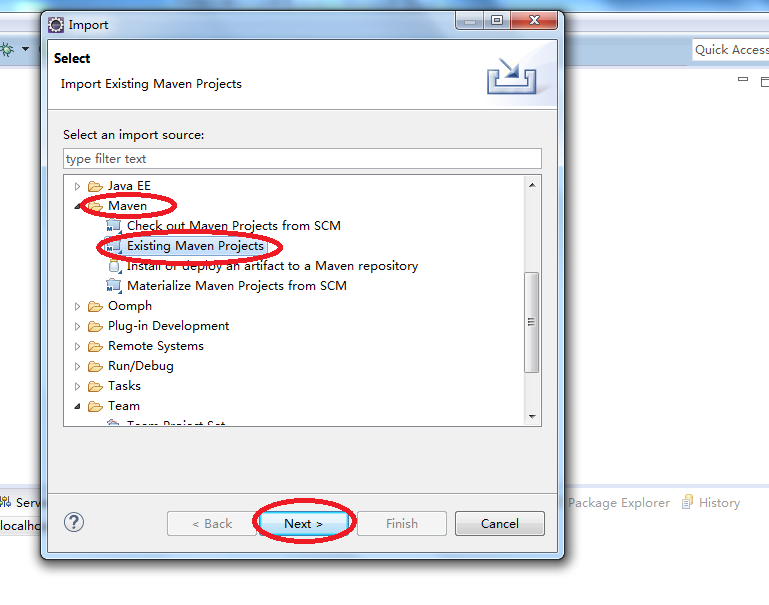
## eclipse导入工程

下面打开eclipse，导入工程

Step1),选择import

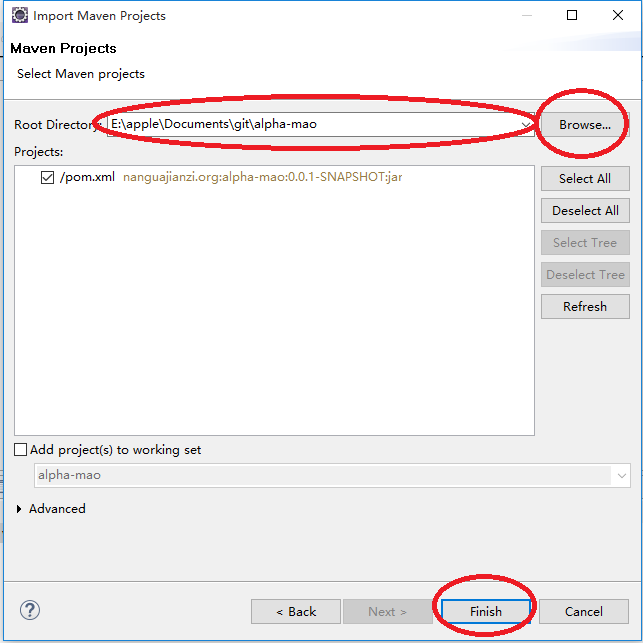
****

Step2)选择 “Existing Maven Projects”



Step3) 选择你在第6章中git下载的工程目录：D:\git\ **wei-customer-care**

然后点击finish



**Step4) 等待工程maven建立**

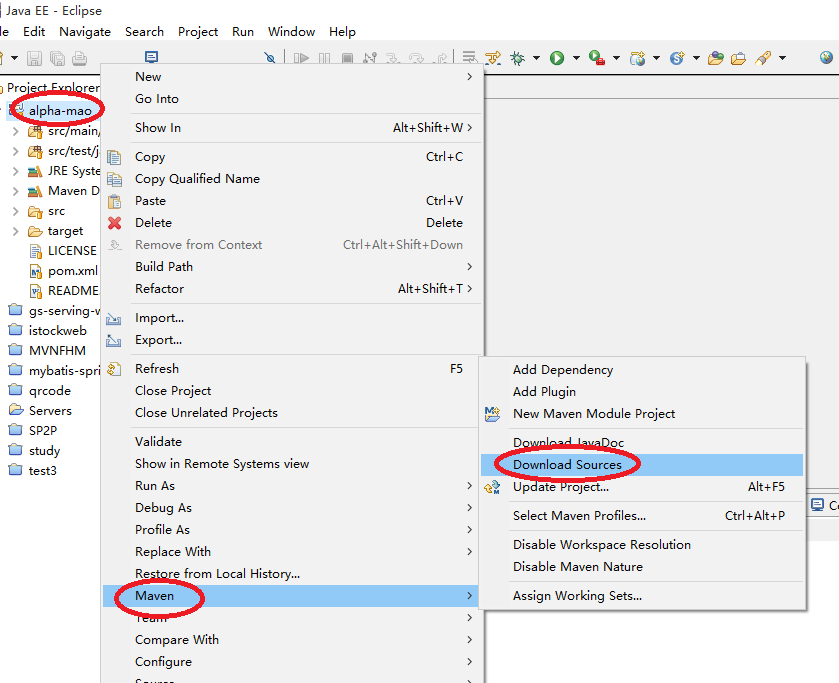
**这会是很长的时间，因为maven要下载所需要的所有依赖jar包**

## eclipse开发

现在起，你可以用eclipse进行开发

有些小tips

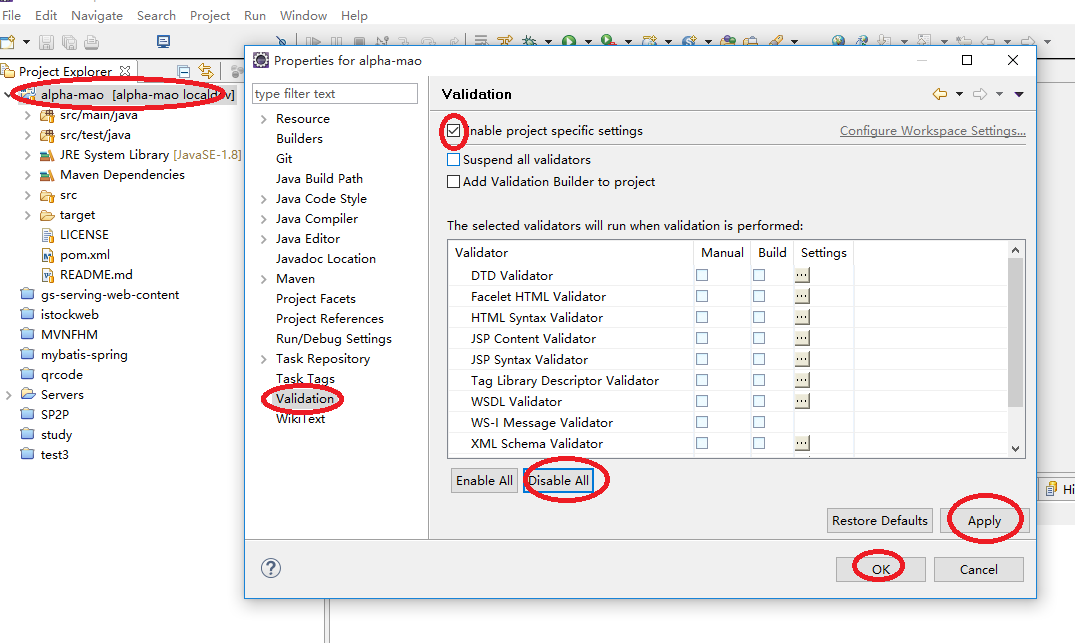
1. 下载第三方开源包源代码



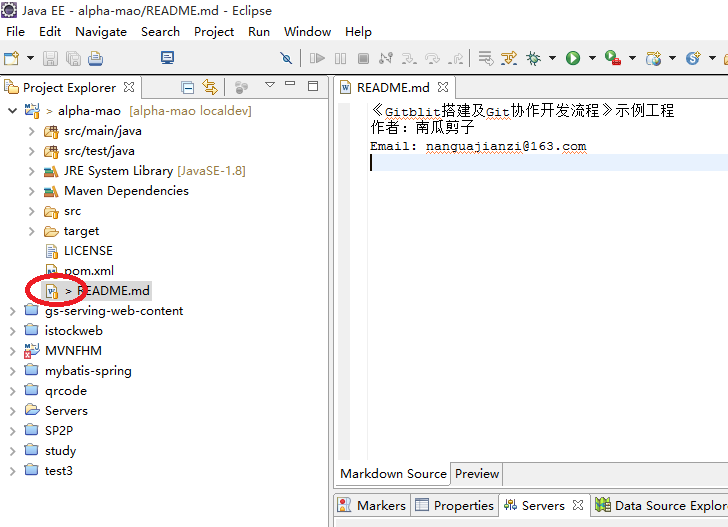
1. 由于有些项目有很多XML文件，eclipse默认是需要检查xml文件语法，这需要很多时间

可以关闭这些检查

选择项目，右键”Properties”



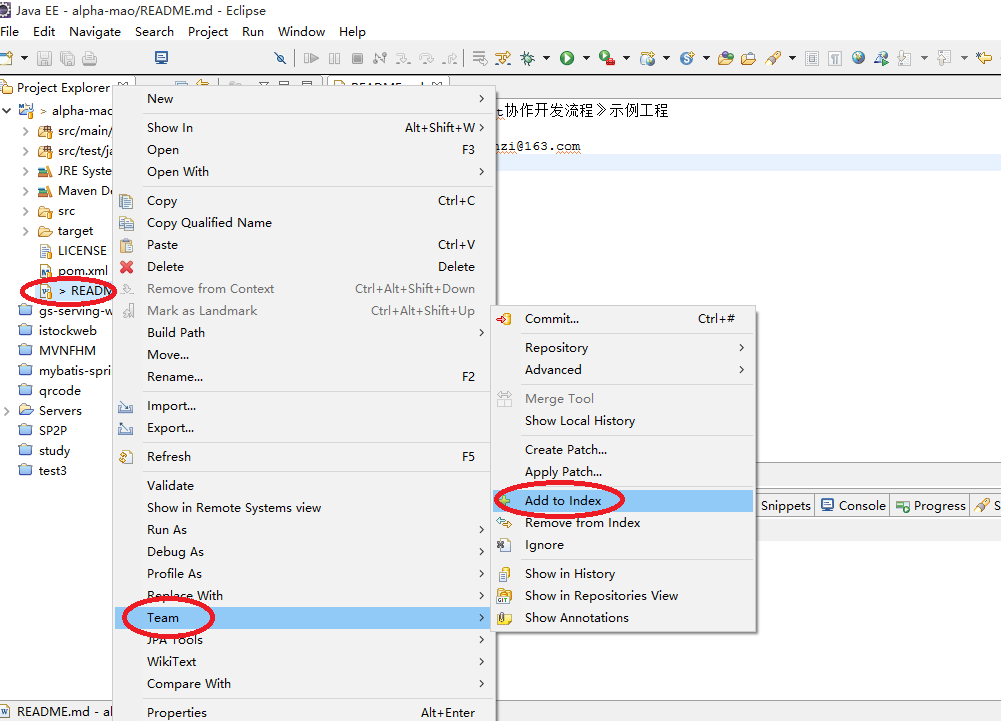
3)每次修改文件后，在左边文件名前都会有一个”>”符号



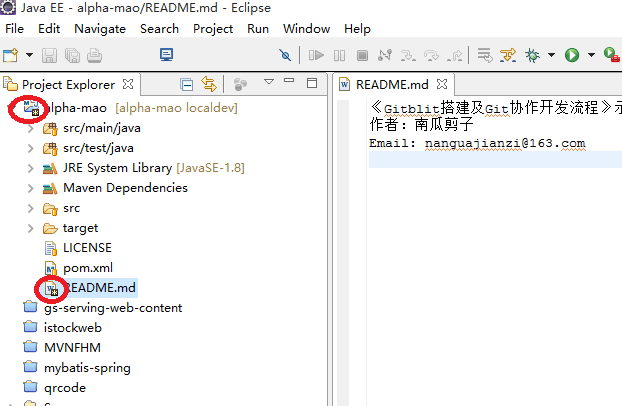
这表明你已经修改过文件了，但是还没有提交

4) 把你修改的文件放入待提交区域，

你该了多个文件，你可以一个一个把他们加入缓冲区(或者叫*index*)

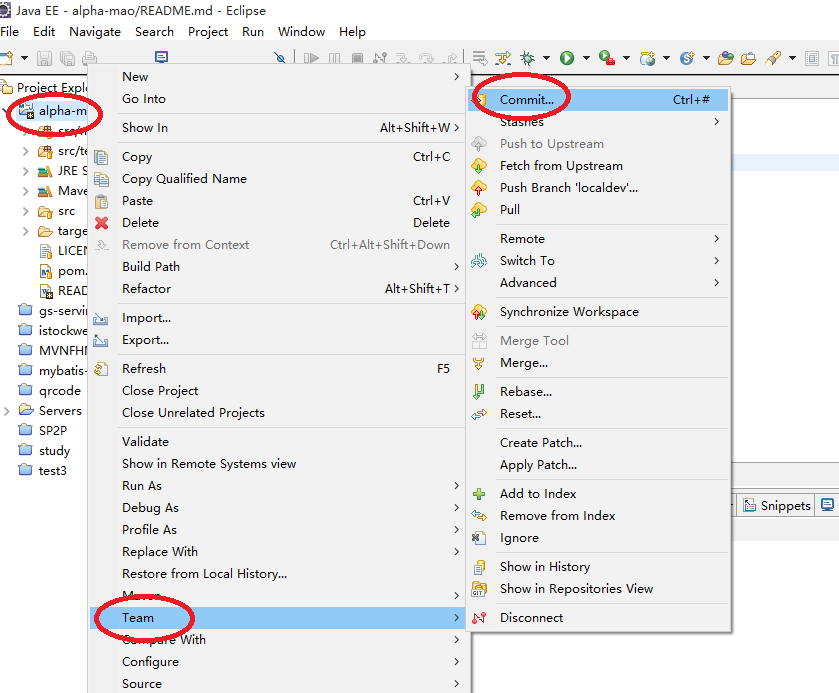


**然后你就会看到相关标记”\*” ,表示文件已经在缓冲区，还没有正式提交**

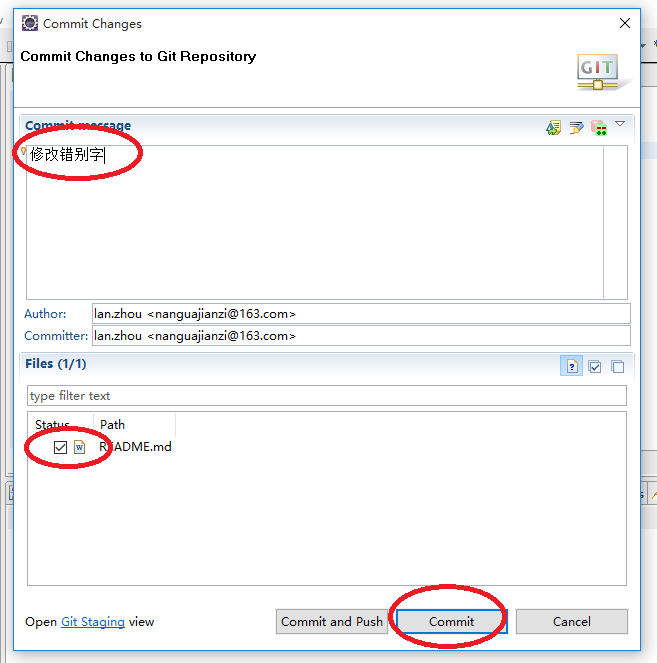


5）当每天工作完成后，提交当天代码

选择工程，右键，按下面操作



然后简单填写提交原因



记住，一定要点击commit,不能点击commit and push

注意，这个只是在你本地提交，并没有提交到服务器端

因为git是分布式的系统，为了上传代码，你必须继续下面的步骤

## 上传代码到服务器

我们强调每天上传代码，为了避免发生不必要损失

前面的操作，你实际上只是在本地更新代码，我们还需要上传到服务器

Step1) 从服务器取得最新代码

打开cmd命令提示符,并进入工程目录，比如：

**E:\apple\Documents\git\alpha-mao>**

然后运行下列命令

**git fetch origin**

**Password authentication**

**Password:**

上面的代码是从服务器取得最新的dev分支代码

Step2) merge服务器dev分支代码到我们的localdev分支代码

首先确保当前的工作分子是localdev

**> git branch （查看你本地的分支）**

**\* localdev**

**Master**

**> git merge origin/dev**

**Already up-to-date.**

(表示没有需要merge的地方，如果有冲突，别人也更改了同一处地方，你需要merge代码)

有关培训，余强会给大家演示一下

Step3) 更新代码到远程服务器

**>git push origin localdev:dev**

## 完毕

剩下的还需要大家自己研究，官方网站已经听过很多资料

<http://gitblit.com/>

[https://git-scm.com/](https://git-scm.com/download/win)

南瓜剪子(北京)