相关工具学习报告

一、Git

1. 什么是git

git是用于文件版本管理的分布式系统。通过git，可以保留文件的历史版本以便随时恢复到指定时间点。

2. git的安装

在linux下输入sudo apt-get install git即可完成安装.

在Windows下需要从[https://git-for-windows.github.io](https://git-for-windows.github.io/" \t "_blank)下载.

3. 创建版本库

版本库是一个在git版本管理之下的目录。版本库中每个文件的增删查改都将被git跟踪.

创建方法:

在bash或者windows下的git bash输入以下命令

$ mkdir testgit

$ cd testgit

$ git init

这样就在当前目录建立了一个版本库.

4. git的工作流程

创建新版本: 修改文件 -> 将修改后的文件添加到暂存区 -> 提交.

如上操作之后，就为版本库添加了一个可以恢复到的新状态.

合并分支: 添加分支 -> 在新分支中提交代码 -> ...(不断提交代码） -> 合并分支

合并分支可用于同时进行多个功能的开发. 使用分支功能后，这些开发工作可以在一个稳定版本的基础上分别进行，最后只要合并分支，就能将对代码的修改合并到一起.

5. 具体命令列表

（1) 版本控制

将code.java添加到暂存区

$ git add code.java

将code.java从暂存区删除

$ git rm --cache code.java

提交修改

$ git commit -m "message"

查询暂存区

$ git status

文件状态对比(将当前文件状态与上次状态对比)

$ git diff code.java

删除文件

$ rm code.java

$ git rm code.java

从上一个版本恢复误删的文件

$ git checkout --test.txt

（2）分支管理

查看分支：git branch

创建分支：git branch <name>

切换分支：git checkout <name>

创建+切换分支：git checkout -b <name>

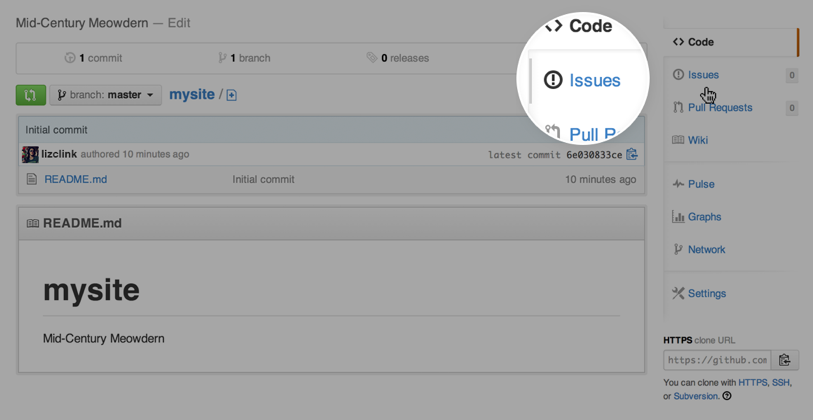
合并某分支到当前分支：git merge <name>

删除分支：git branch -d <name>

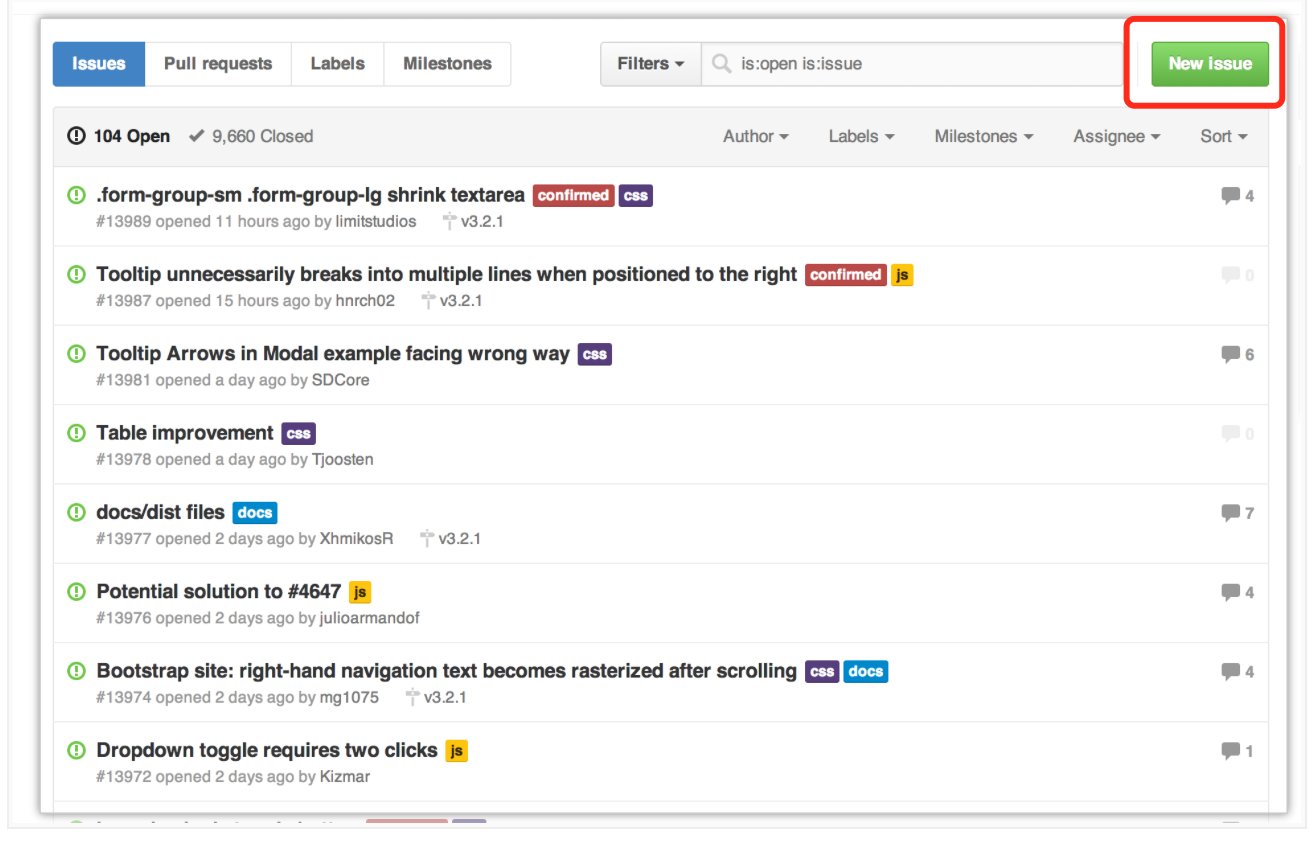
二、Github Bug Tracker

本次项目我们使用github的Issues 进行BUG管理.操作流程如下

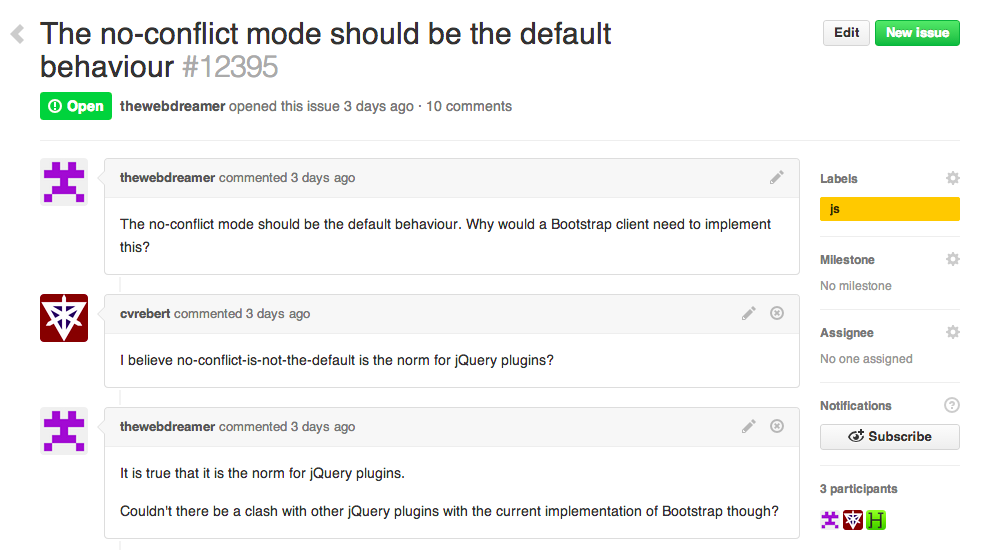
在github中打开Issue界面:



点击添加新Issues：



填入相关信息后提交，得到Issues详情如下



其中Title 和Description用于描述BUG. label和milestone用于对Bug进行分类. assignee用于指派bug的处理者，comments用于其他人review并评论.

三、Jenkins 持续集成环境

1.简介

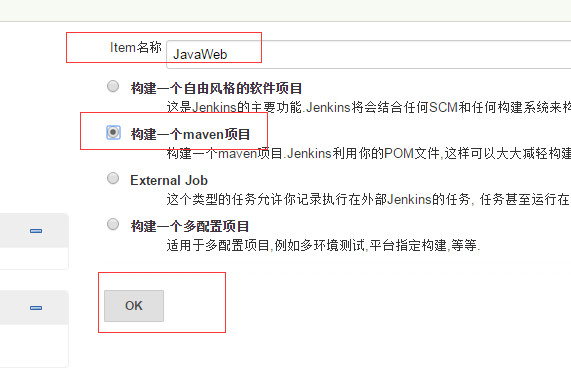
Jenkins是一个持续开发环境。在现实中，开发环境和用户使用的环境常常有所区别，如可能会出现程序使用了一些开发环境的依赖库，而用户环境并没有安装。Jenkins可以在每次提交代码之后实现用户环境下程序测试的自动化。

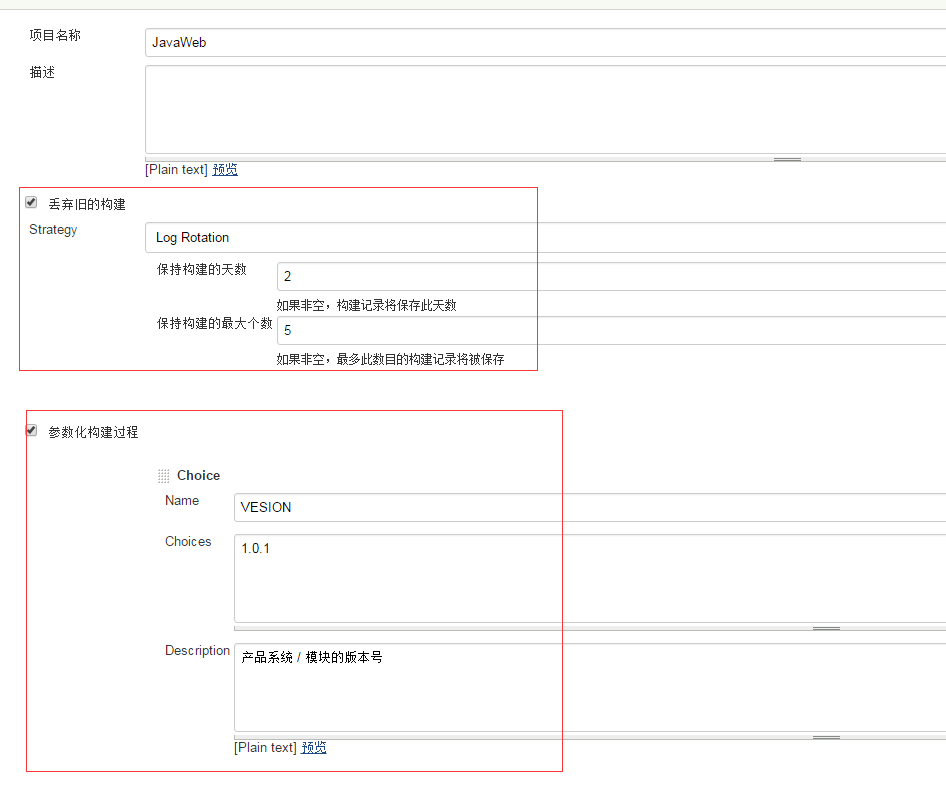
2. 部署方法

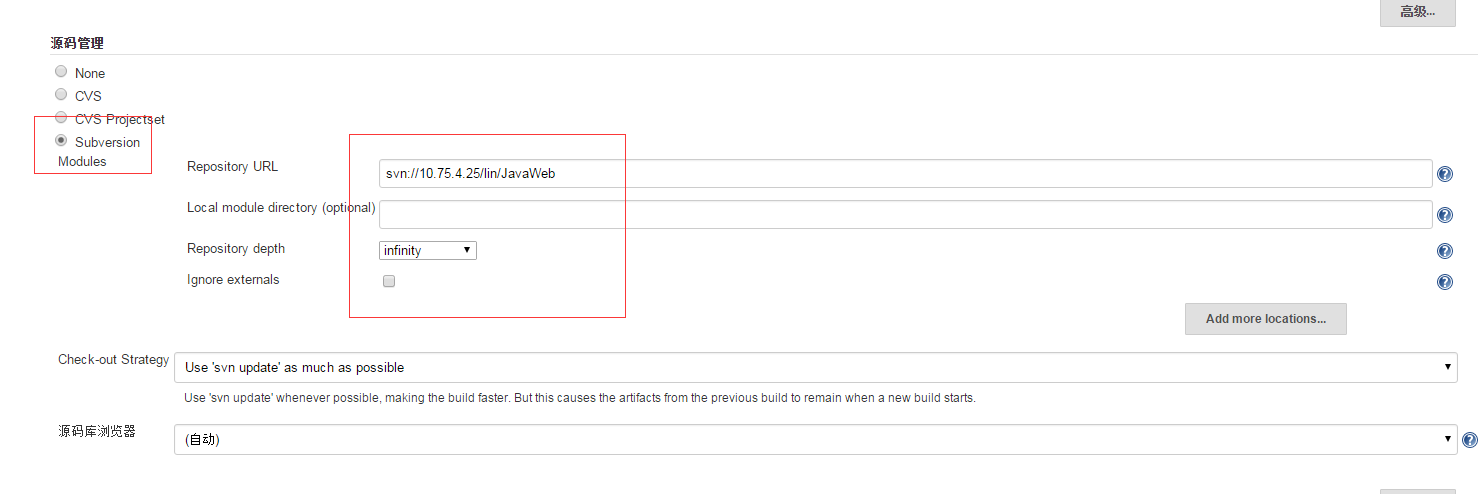
（1）安装Jenkins

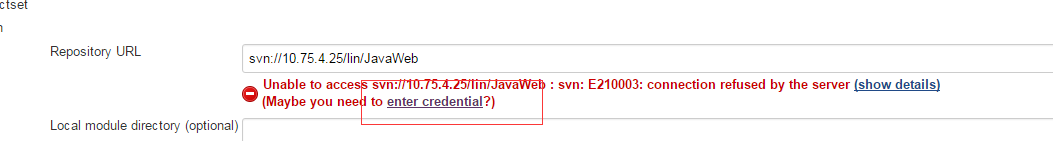
在官网下载war包，然后解压，把得到的war包直接扔到tomcat下，启 动tomcat，Jenkins就安装完毕，访问ttp://localhost:8080，即可进入Jenkins 控制目录

2. 新建一个Job，设置好相关信息

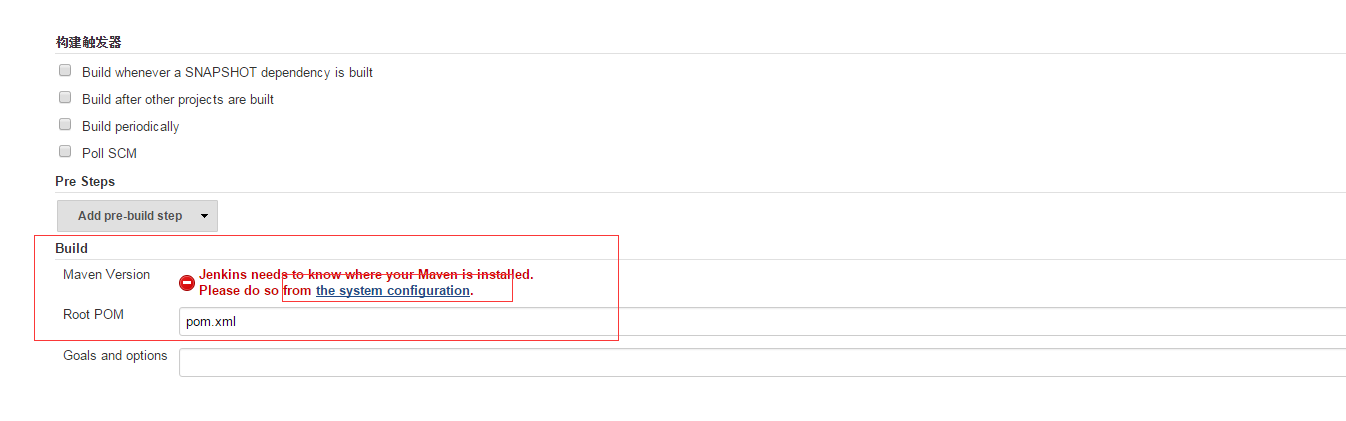
 

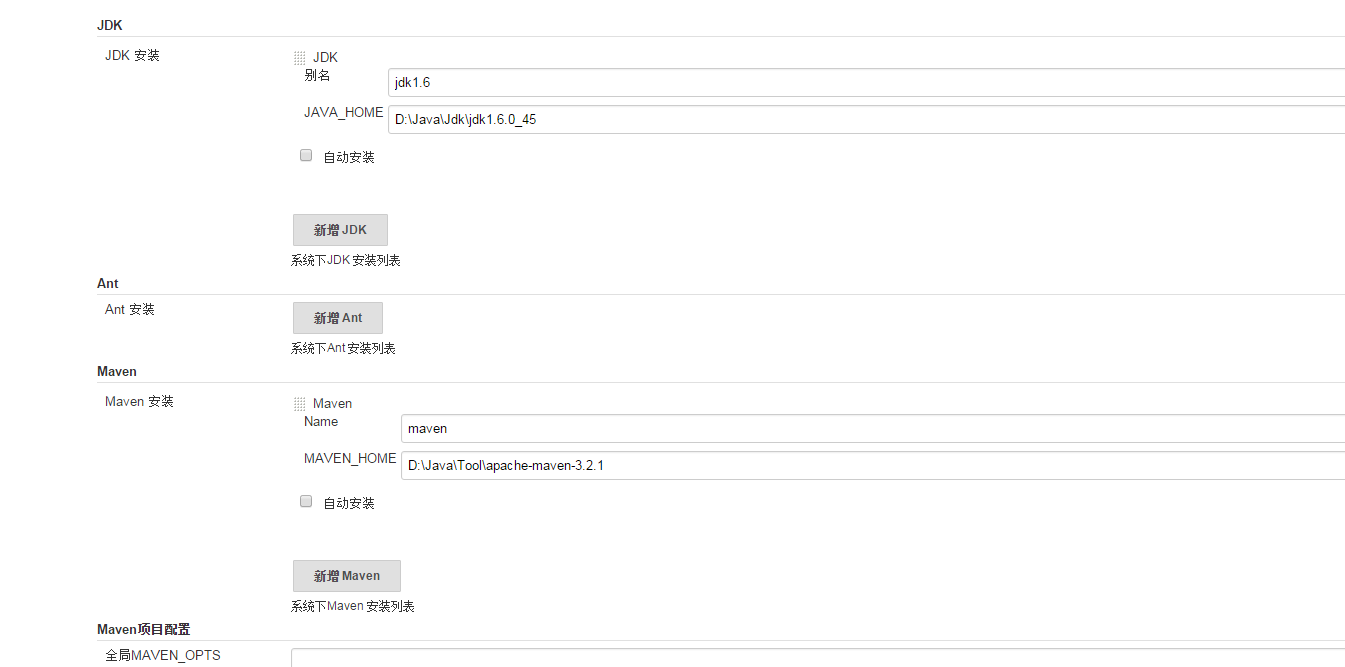






3. 配置JDK和Maven

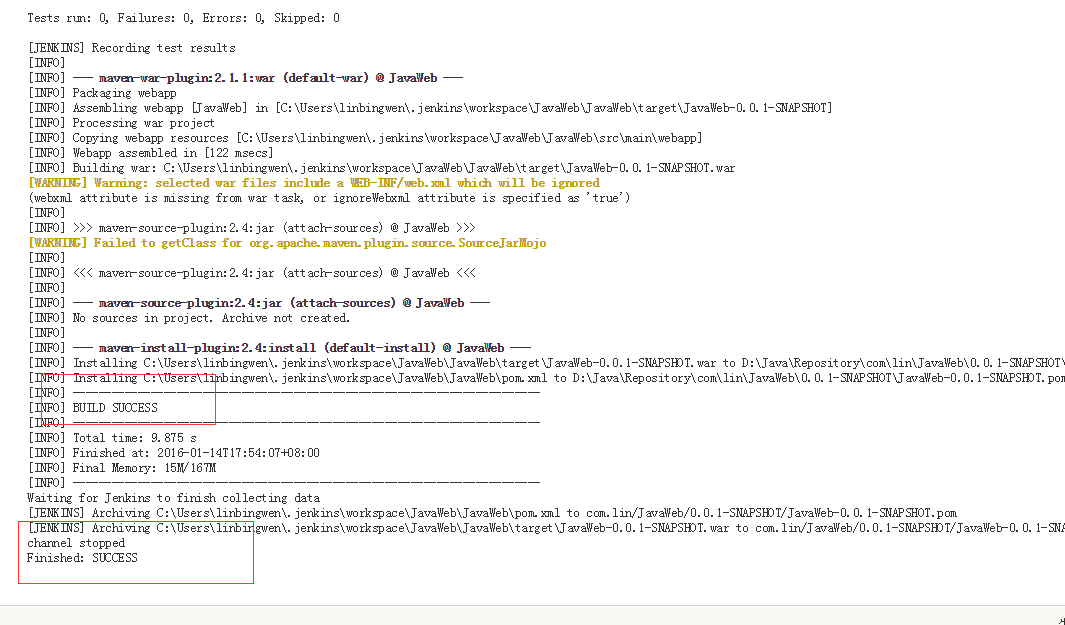




4. 构造项目



成功输出如下



5. 可以在工作目录找到.war文件。 这就是输出. 可见编译是成功的.

