Jenkins配置Sonar检测构建项目

**文档更新记录**

|  |  |  |  |
| --- | --- | --- | --- |
| **日期** | **修改内容概要** | **作者** | **版本** |
| 2017.10.23 | [1、添加多模块中单独发布单模块的配置](#moresingle)  [2、添加单元测试中初始化问题解决方案](#question10)  [3、pom文件配置了跳过测试的解决方案](#delpomskip) | 常博 | 1.0 |
|  |  |  |  |
|  |  |  |  |

目录

[Jenkins配置Sonar检测构建项目 1](#_Toc495933516)

[一、引言 3](#_Toc495933517)

[1、编写目的 3](#_Toc495933518)

[2、适用范围 3](#_Toc495933519)

[二、构建项目 3](#_Toc495933520)

[1、Java工程的配置 3](#_Toc495933521)

[2、相关配置 4](#_Toc495933522)

[2.1、Jenkins系统配置 4](#_Toc495933523)

[3、Jenkins新建项目 6](#_Toc495933524)

[4、Jenkins构建项目 13](#_Toc495933525)

[5、常见问题 15](#_Toc495933526)

[5.1、Sonar报告没有显示覆盖率或为零 15](#_Toc495933527)

# 一、引言

## 1、编写目的

本文档主要编写了如何使用Jenkins创建Java项目，并配置SonarQuber对工程进行代码质量检测工作。

## 2、适用范围

使用 Maven创建的Java项目。

# 二、构建项目

## 1、Java工程的配置

在Java工程中，只需要在pom.xml添加如下信息：

<plugin>

<groupId>org.codehaus.mojo</groupId>

<artifactId>cobertura-maven-plugin</artifactId>

<version>2.6</version>

<configuration>

<formats>

<format>xml</format>

</formats>

**<instrumentation>**

**<excludes>**

**<exclude>com/company/demo/entity/\*</exclude>**

**<exclude>com/company/demo/utils/\*</exclude>**

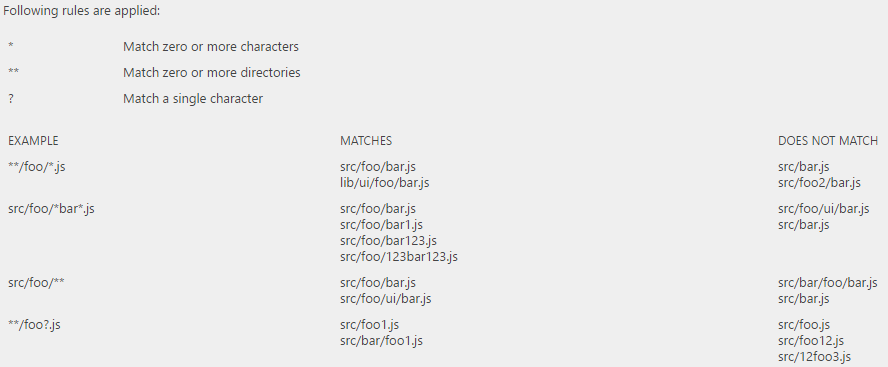
**</excludes>**

**</instrumentation>**

</configuration>

</plugin>

红色字体部分为不进行单元测试覆盖率统计的文件路径，如没有则删除该部分代码。参考下图配置忽略的文件路径：



## 2、相关配置

### 2.1、Jenkins系统配置

1）下载安装SonarQube插件及Cobertura插件；

如下图2.1中，选择系统管理🡪管理插件。



图2.1

选择后进入图2.2，选择可选插件找到SonarQube插件和Cobertura插件选中，点击“直接安装”按钮安装。

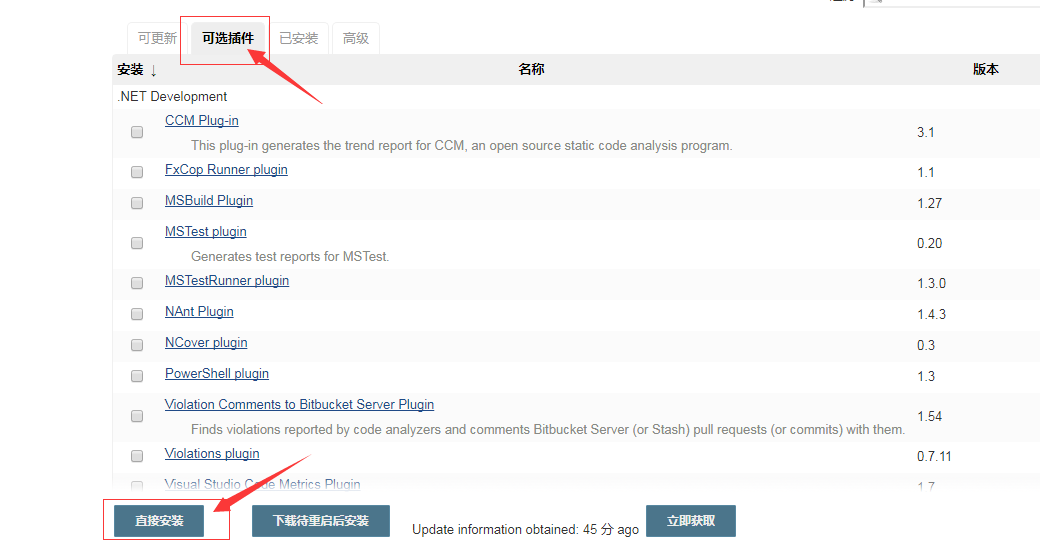


图2.2

2）SonarQube平台相关配置；

完成插件安装后，如上图2.1中，选择系统管理后，选择，找到SonarQube平台的配置项，如下图2.3。根据SonarQube平台的信息配置相关的信息，主要是SonarQube平台的访问路径，版本及对应账号的token。

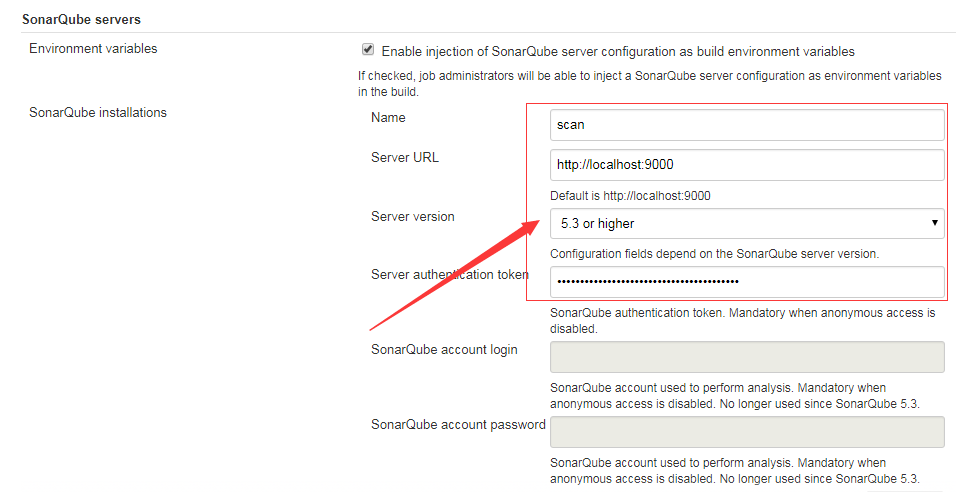


图2.3

3）Sonar-Scanner相关配置；

如上图2.1中，选择系统管理后，选择，找到SonarQube Scanner 配置选项，根据SonarQube-Scanner安装路径，添加配置信息如下图2.3：

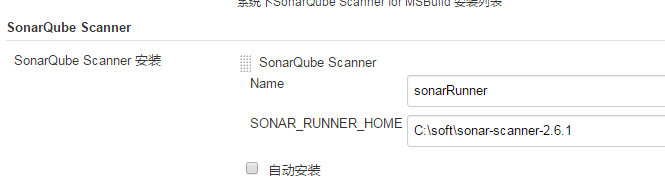


图2.4

## 3、Jenkins新建项目

点击页面左边栏的新建，开始创建新的项目。如图3.1：

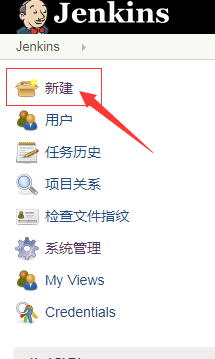


图3.1

填写创建的项目名称及风格，点击OK，成功创建项目，这里我们一般选择构建一个maven项目。如图3.2：

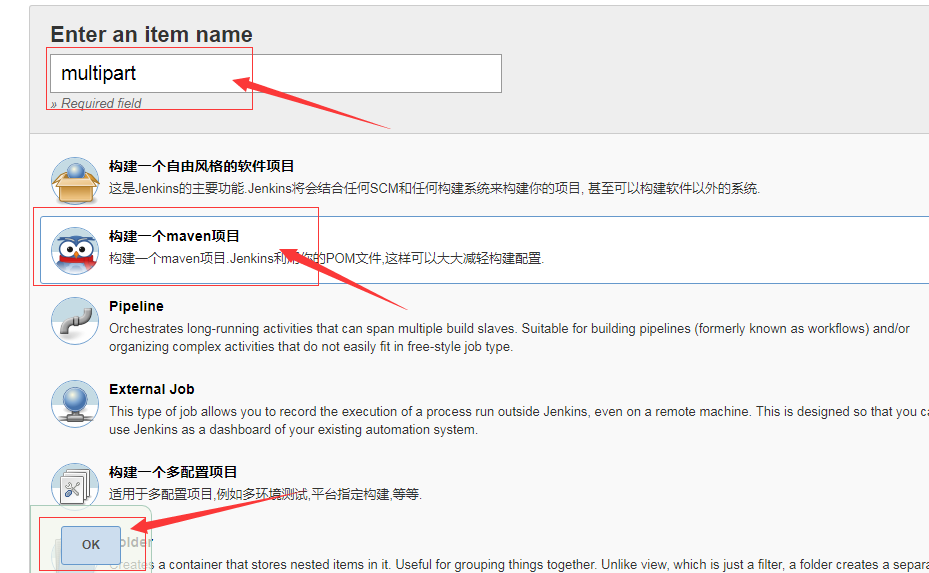


图3.2

项目创建成功后，直接转入项目的配置页面，该页面退出后，可在项目中选择配置进入。配置页面如图3.3

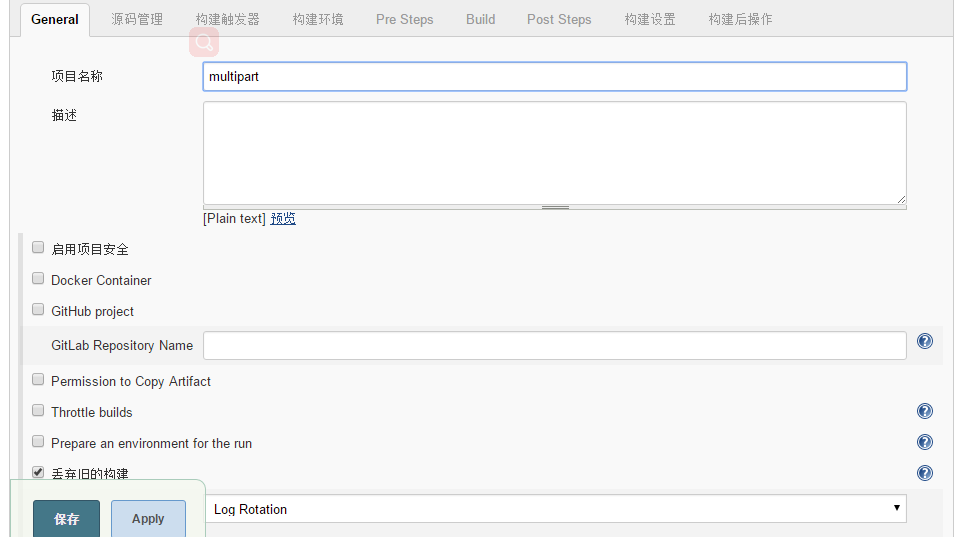


图3.3

配置页面中，项目名称即是我们在图3.2中填写的名称。在综合配置中，选择参数化构建过程，点击添加参数按钮，选择Git Parameter配置，如图3.4：

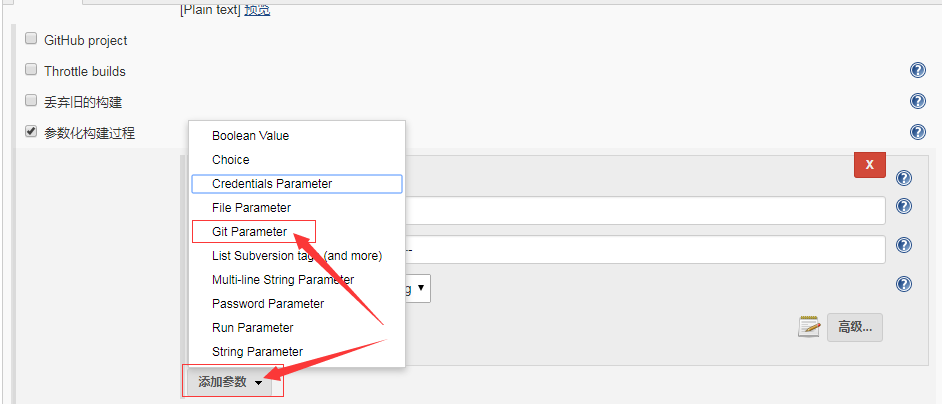


图3.4

Git参数配置如下图3.5，添加Git参数配置后，可以在构建页面快速的选择Git的分支或标签。

|  |  |
| --- | --- |
| **主要参数名称** | **功能** |
| Name | 名称，其他配置调用使用 |
| Description | 描述信息，在构建项目时显示提示 |
| Parameter Type | 构建时，Git显示分支或者标签，或全部 |
| Branch Filter | 用于过滤分支 |
| Tag Filter | 用于过滤标签 |

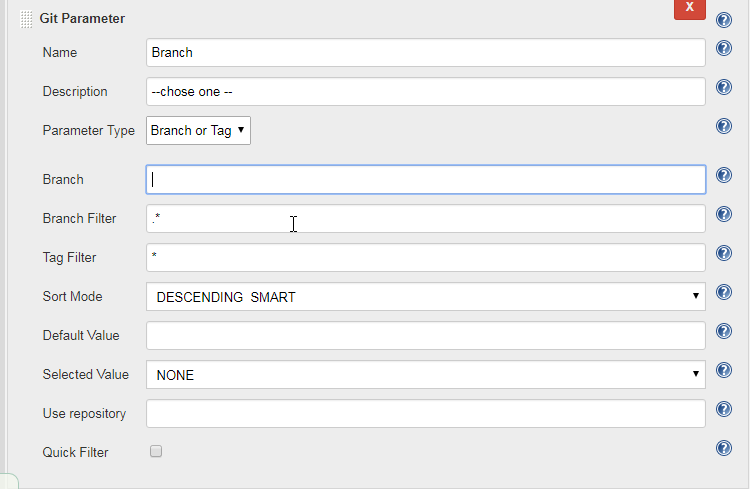


图3.5

在源码管理中，选择Git，填写构建项目的下载路径和账号，如图3.6。图中箭头所示创建分支的参数值即为上图3.5中配置的Name。

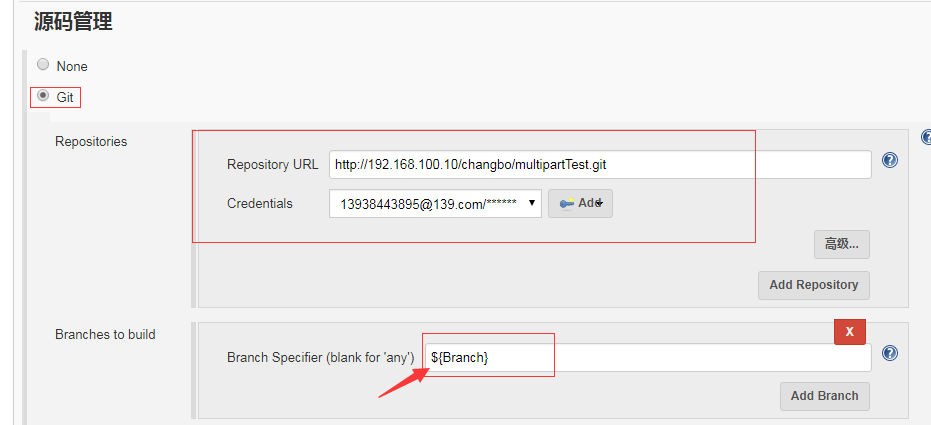


图3.6

找到Build，添加配置文件的名称和Maven的命令，如图3.7。如箭头所示查看单元测试覆盖率需要添加单元测试的语句“**cobertura:cobertura**”，并且不能有“-**Dmaven.test.skip=true**”，该语句将跳过单元测试。

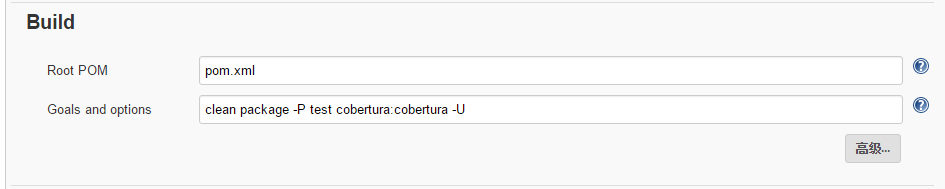


图3.7

点击增加构建步骤按钮，选择Execute shell，如图3.8。

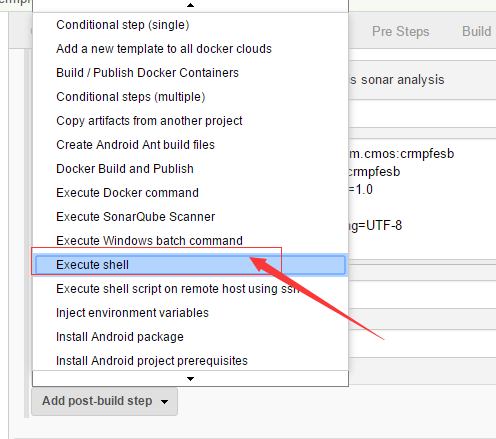


图3.8

选择后填写对应部署系统的命令操作Maven打包后的Git项目到相应的目录结构并启动。如下图3.9。图中有3个单点选项，分别是：

1. 只在maven构建成功后执行；
2. 在maven操作成功或者不稳定状态后执行；
3. 无论构建结果如何，都执行；

该配置需要管理人员去查看发布后的状态，如果选择错误，可能会导致后续的配置没有执行，无法正常发布成功。

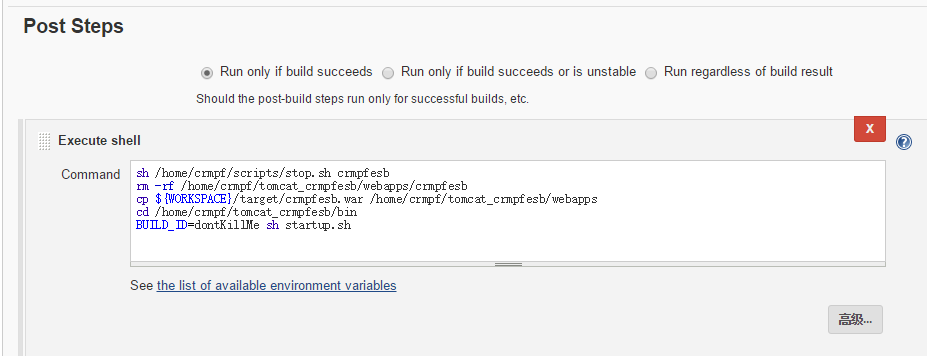


图3.9

再次点击增加构建步骤按钮，选择添加SonarQube Scanner的配置，如图3.10。需要注意的是，必须先配置**Maven**，然后再配置**SonarQube Scanner**，否则无法显示单元测试覆盖率。

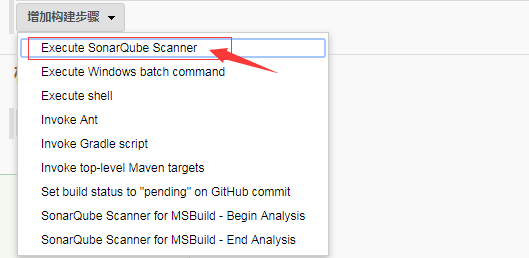


图3.10

SonarQube Scanner配置如下图3.11所示，填写SonarQube Scanner的名称（**Jenkins系统配置SonarQube时的名称**），选择JDK版本及填写配置信息。由于SonarQube需求，选择JDK版本为1.8。

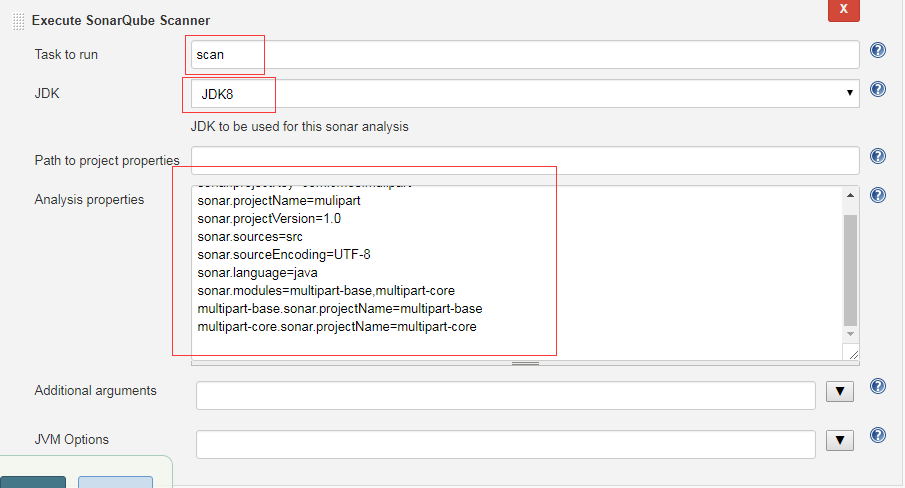


图3.11

SonarQube配置信息在单模块项目和多模块项目中的配置基本相同，需要修改部分相关的配置代码。

单模块配置如下：

**sonar.projectKey=com.cmos:crmpfcore //必须sonar中的唯一**

**sonar.projectName=crmpfcore //项目名称**

**sonar.projectVersion=1.0 //项目版本**

**sonar.sources=src //sonar扫描目录**

**sonar.sourceEncoding=UTF-8 //编码**

**sonar.language=java**  **//Sonar扫描的语言**

多模块配置如下：

**sonar.projectKey=com.cmos:crmpfcore //必须sonar中的唯一**

**sonar.projectName=crmpfcore //项目名称**

**sonar.projectVersion=1.0 //项目版本**

**sonar.sources=src //sonar扫描目录**

**sonar.sourceEncoding=UTF-8 //编码**

**sonar.language=java //Sonar扫描的语言**

**sonar.modules=Module1,Module2,Module3,… //pom中全部模块名称**

**Module1.sonar.projectName=module 1 //sonar中每个模块名称**

**Module2.sonar.projectName=module 2**

**…**

多模块项目每个模块单独发布：

**sonar.projectKey=Sonar-zpay-main-core**

**sonar.projectName=zpay-main-core**

**sonar.projectVersion=1.0**

**sonar.sources=zpay-main-core/src**

**sonar.cobertura.reportPath=zpay-main-core/target/site/cobertura/coverage.xml**

**sonar.sourceEncoding=UTF-8**

**sonar.language=java**

**需要注意，上面标红的名称必须相同**

至此，Jenkins新建项目完成。

## 4、Jenkins构建项目

在新建项目中，点击下图中左边栏箭头指示部分，进入Git的分支或标签选择页面。



图4.1

在下图4.2中，选择要发布的分支或标签后，点击开始构建按钮即可。

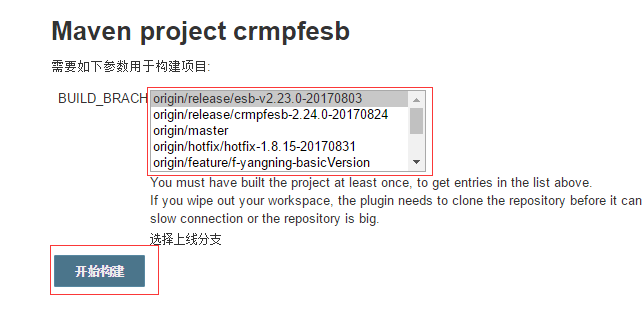


图4.2

项目构建成功后，点击如下图4.3红色部分，可直接跳转至SonarQube平台查看项目的代码质量检测结果，图4.4。



图4.3

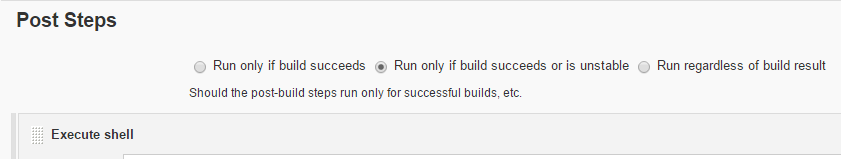


图4.4

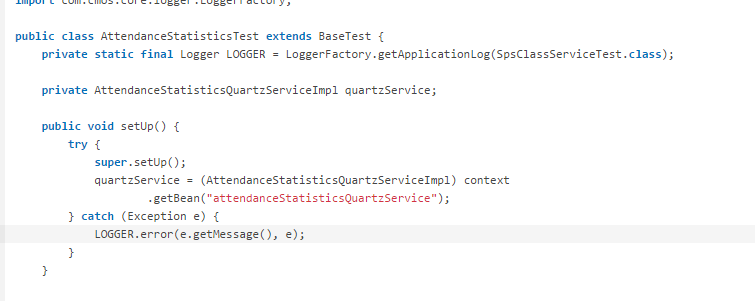
## 5、常见问题

### 5.1、Sonar报告没有显示覆盖率或为零

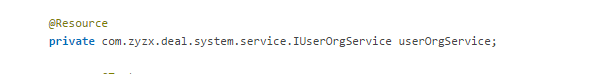
1. 查看maven的pom文件是否添加了插件的配置信息。
2. 查看Jenkins的maven配置信息，删除Dmaven.test.skip=true语句。确保maven启动时，启动单元测试。
3. maven配置信息是否添加cobertura:cobertura，是否添加了SonarQube Scanner配置信息。
4. 查看发布的项目是否使用JUnit插件编写单元测试。
5. Jenkins启动失败，查看是否为单元测试时报错。如果有报错，将报错的单元测试注释掉或者代码内部使用try catch捕获处理。
6. 当单元测试内容过多或者引入过多的第三方插件时，会出现内存溢出的错误。为启动的JVM添加内存参数，增大内存的值可解决。
7. 项目下有多个模块，并在pom文件中已经配置的。需要根据多模块的示例配置，否则不显示覆盖率。
8. 在jenkins配置中，maven的配置要在sonar scanner配置前。
9. 在jenkins中，**某些项目发布时状态为不稳定**。需要在问题8中，两个配置中间的选项选择如下图所示。



1. 部分单元测试案例中，初始化对象时读取配置文件，使用注解依赖注入的方式，造成无法初始化对象。这时需要使用setUp方法（另外也可以直接使用new的方式），如下图：



而不是使用：



1. 部分跳过单元测试直接配置在pom文件中，去掉下面部分：

