# Jenkins+soapUi+maven+svn

# 持续集成框架手册

**制作人：王绍坤**

**文档版本号：V1.0.1**

目录

[Jenkins+soapUi+maven+svn 1](#_Toc455405570)

[持续集成框架手册 1](#_Toc455405571)

[一、环境准备： 4](#_Toc455405572)

[二、jenkins+tomcat本地安装配置 8](#_Toc455405573)

[三、soapuiProject 13](#_Toc455405574)

[四、Jenlins-maven-soapuiproject 15](#_Toc455405575)

[五、Jenlins构建触发器 20](#_Toc455405576)

[六、发送邮件设置 21](#_Toc455405577)

#### 文档修订记录

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 版本编号 | 变化状态 | 简要说明 | 变更人 | 变更日期 |
| 1.0 | 所有章节 | 创建 | 王绍坤 | 2016.07.04 |
| 1.0.1 | soapuiProject | TestSuit重新修改datasource数据参数化模板，增加Datasink写入datasource | 王绍坤 | 2016.07.08 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

## ****一、环境准备：****

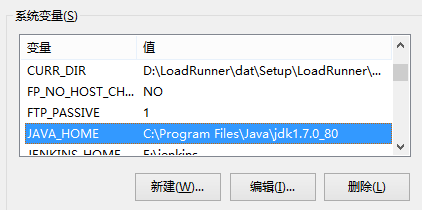
#### JDK：

建议安装JDK1.7

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

环境变量配置不再多说

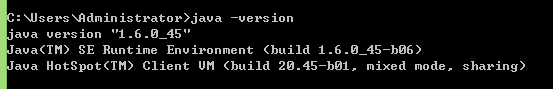
JAVA\_HOME: C:\Program Files\Java\jdk1.7.0\_80（不加“;”）



Path:%JAVA\_HOME%/bin;

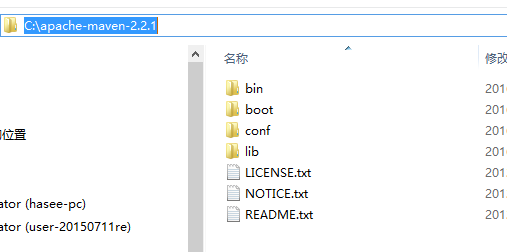
CLASSPATH：%JAVA\_HOME%\lib\tools.jar;%JAVA\_HOME%\lib\dt.jar;

命令行输入：java -version，出现如下提示说明安装成功



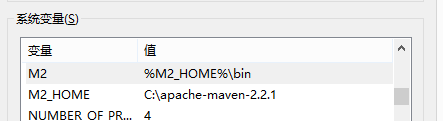
#### Maven安装与配置

下载apache-maven-2.2.1.rar解压到本地如C:\apache-maven-2.2.1，最好放在java同一硬盘，不然运行mvn –v会找不到jre



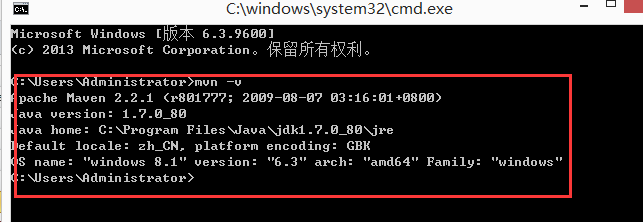
M2：%M2\_HOME%\bin

M2\_HOME：C:\apache-maven-2.2.1



Path：%M2%;

Cmd运行mvn –v如下结果说明maven配置成功



#### Soapui环境配置

这里使用pro版

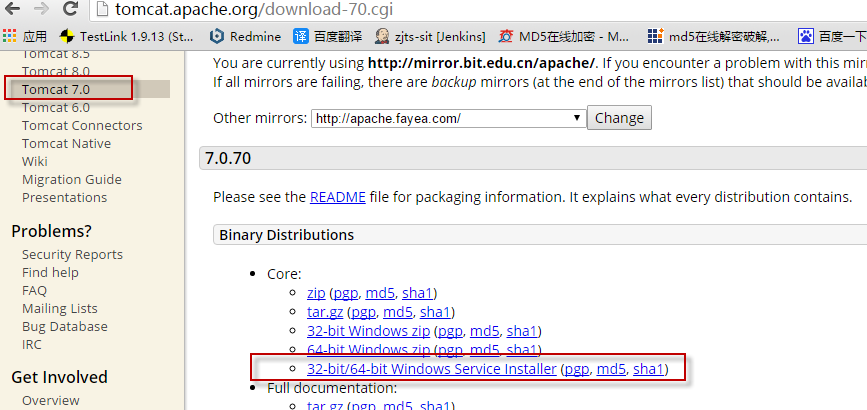
SOAPUI\_HOME：D:\Program Files (x86)\SmartBear\soapUI-Pro-4.5.2；

Path：%SOAPUI\_HOME%\bin;

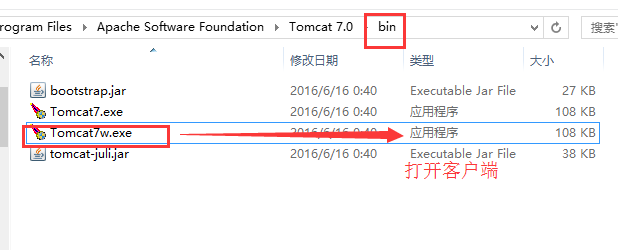
## ****二、jenkins+tomcat本地安装配置****

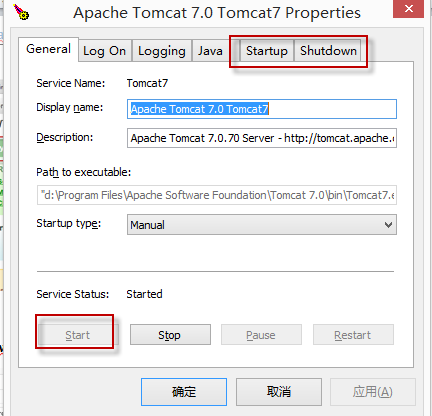
###### 1 、tomcat本地安装

<http://tomcat.apache.org/download-70.cgi> apache-tomcat-7.0.70.exe

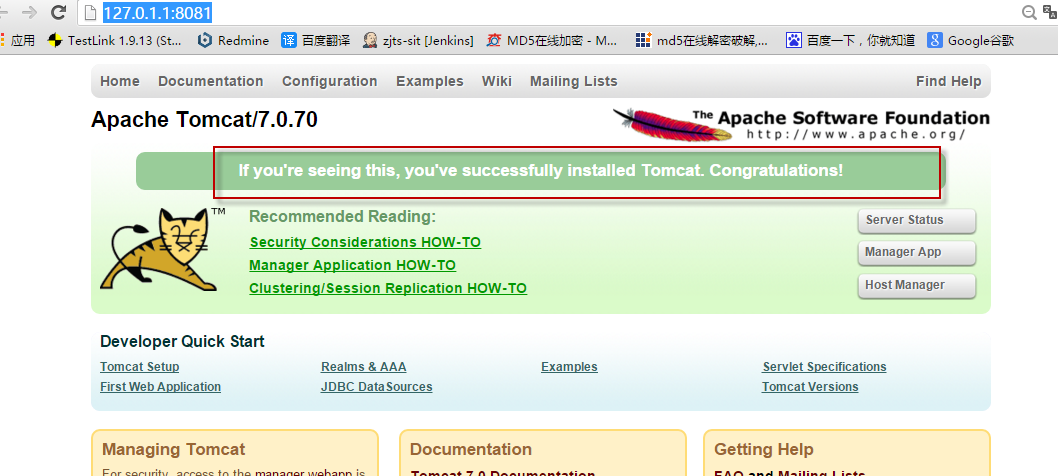
安装步骤可自行百度

启用或关闭服务

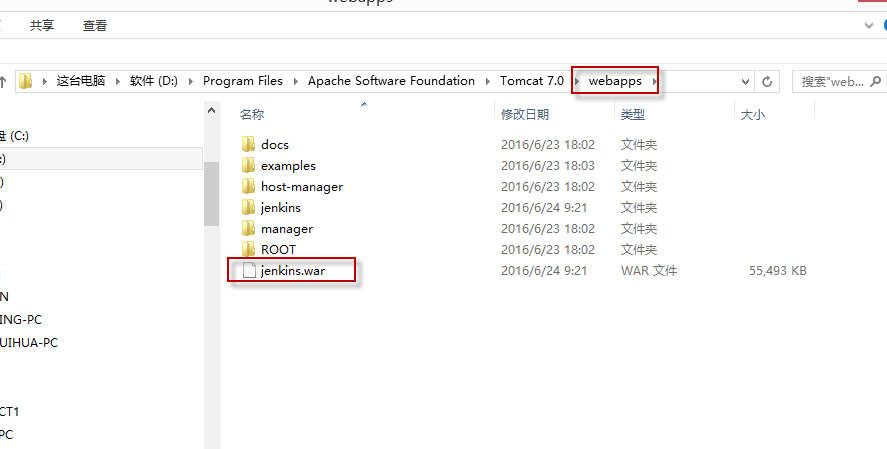




安装过程默认端口为8080，因为我本地安装jenkins已占用，所以把默认端口修改为8081或者其他。。，安装成功后浏览器访问<http://127.0.1.1:8081/> 成功会出现如下页面



###### 2 、下载jenkins.war包，下载成功后放到tomcat webapps目录

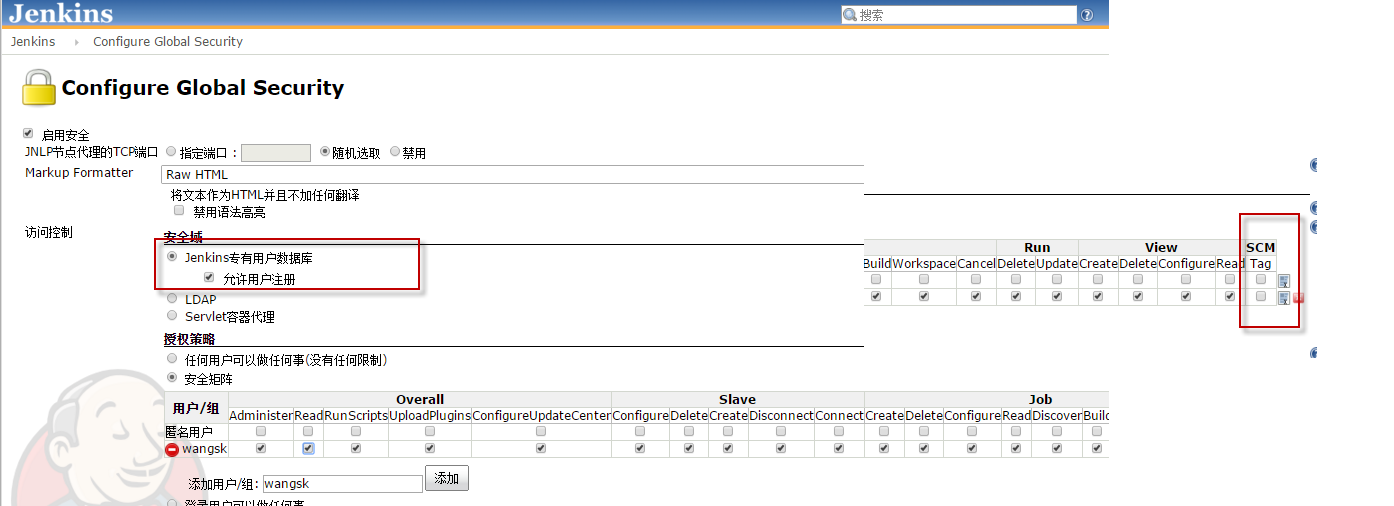


然后访问：<http://127.0.1.1:8081/jenkins/> jenkins环境部署成功如下图

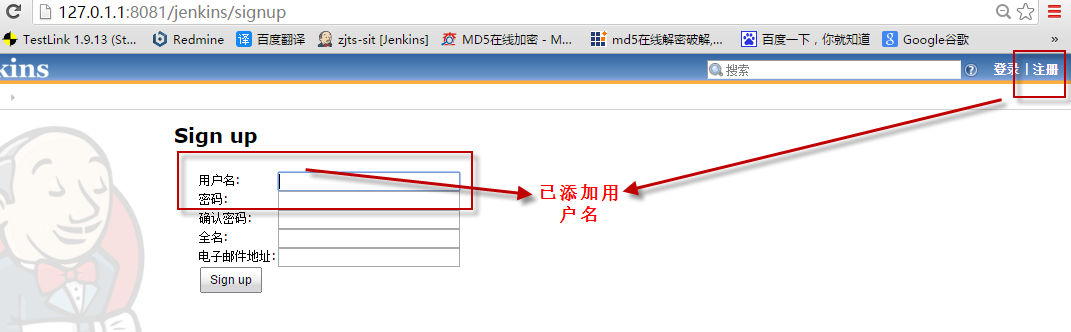


**Jenkins配置比起jmeter配置起来相对简单一些，如图配置即可运行**

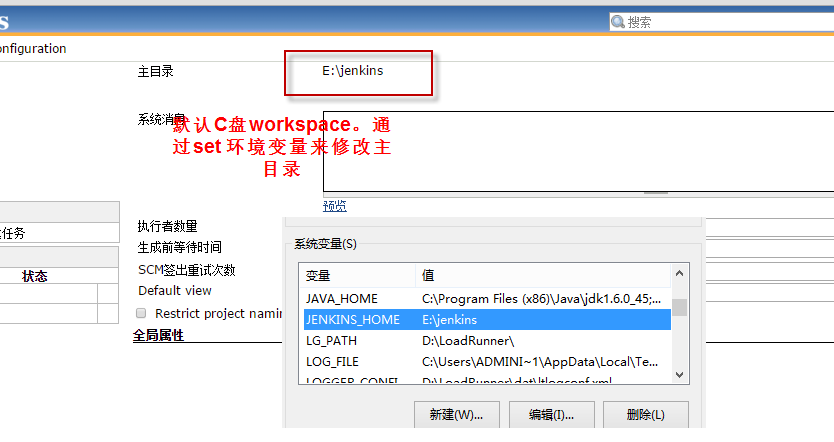
**首先系统设置启用安全，添加用户信息及权限**

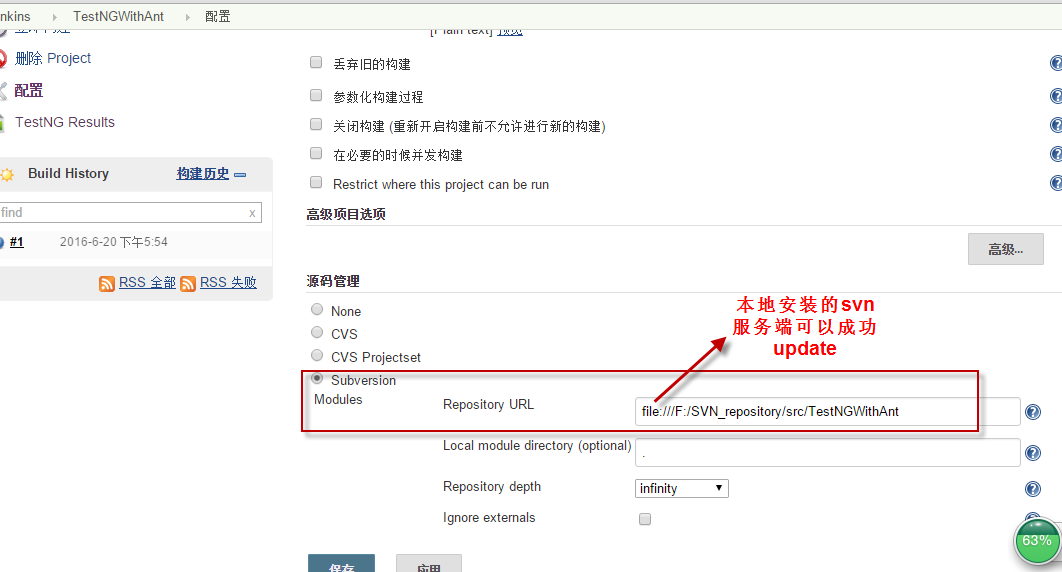


**注册用户**

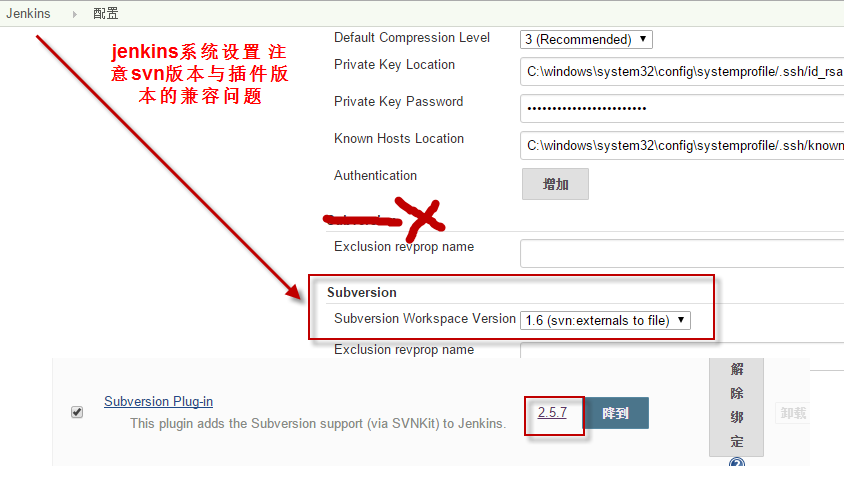


**1 job配置和主目录修改**

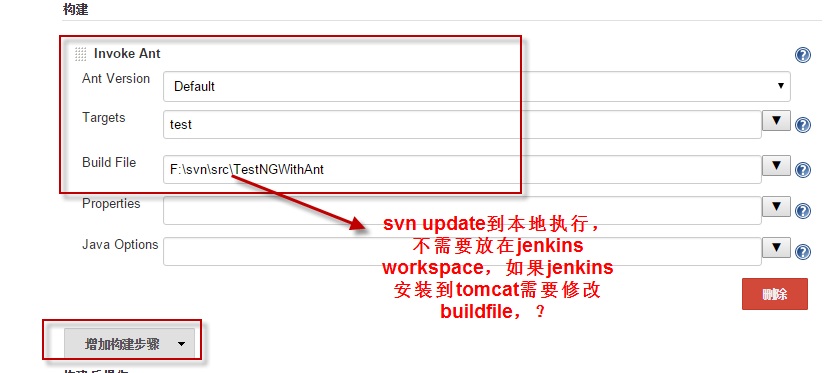




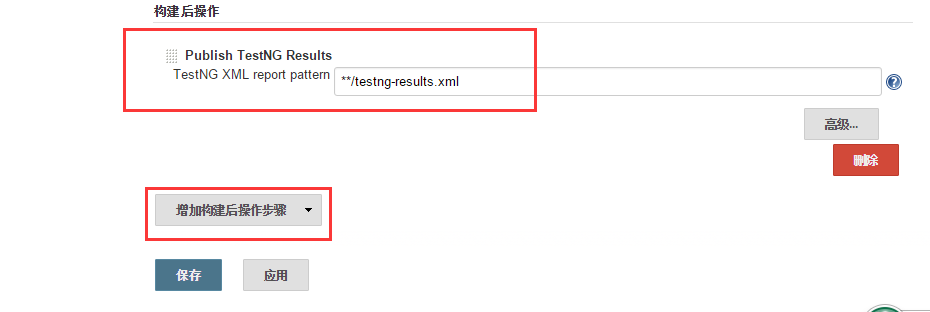
**注意svn版本与插件版本 否则会报错的**



**2**



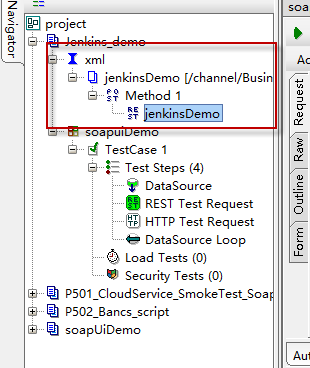
3



## ****三、soapuiProject****

~~Teststep说明~~

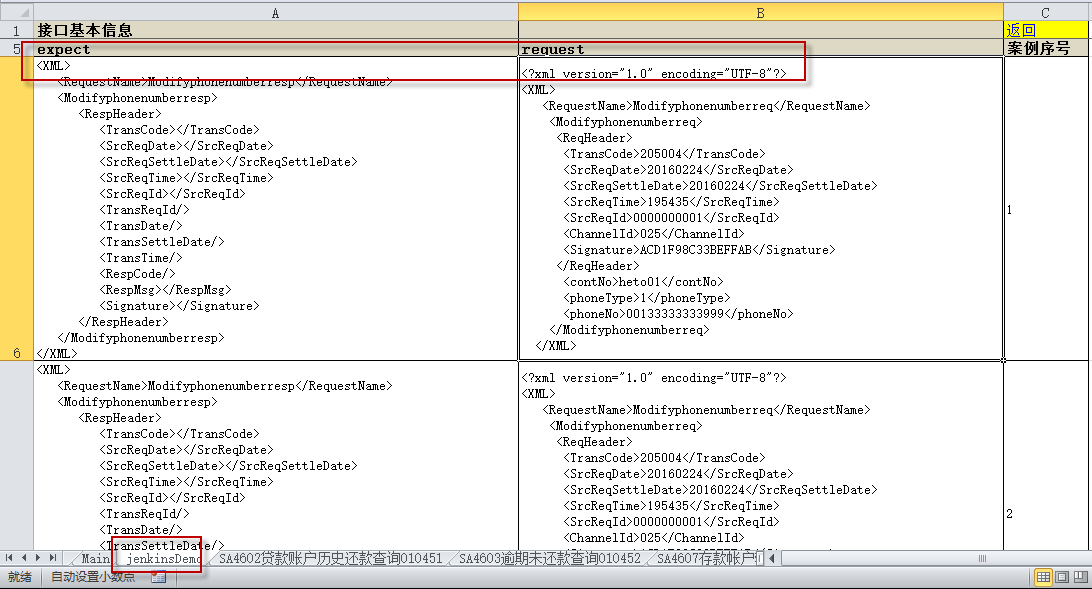
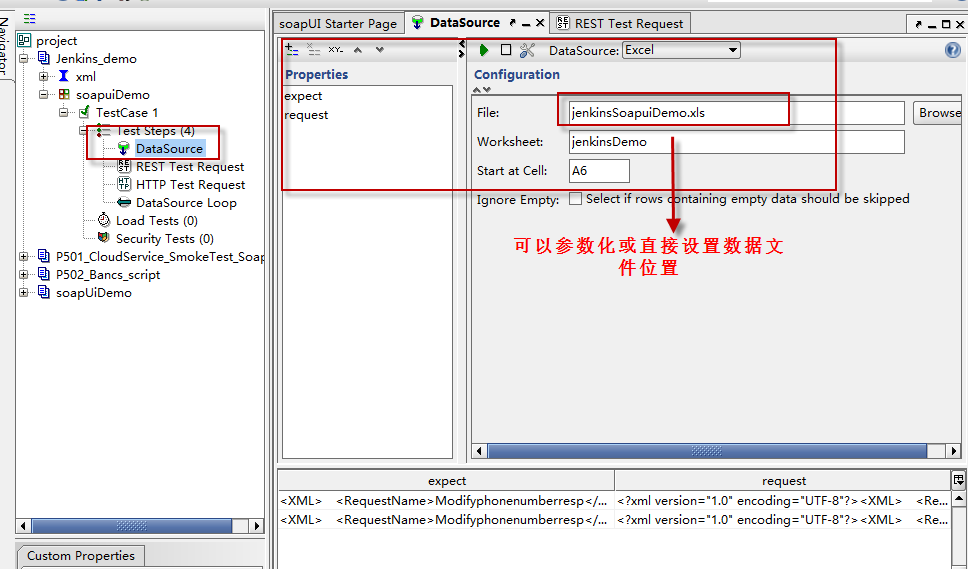
Soapui resource



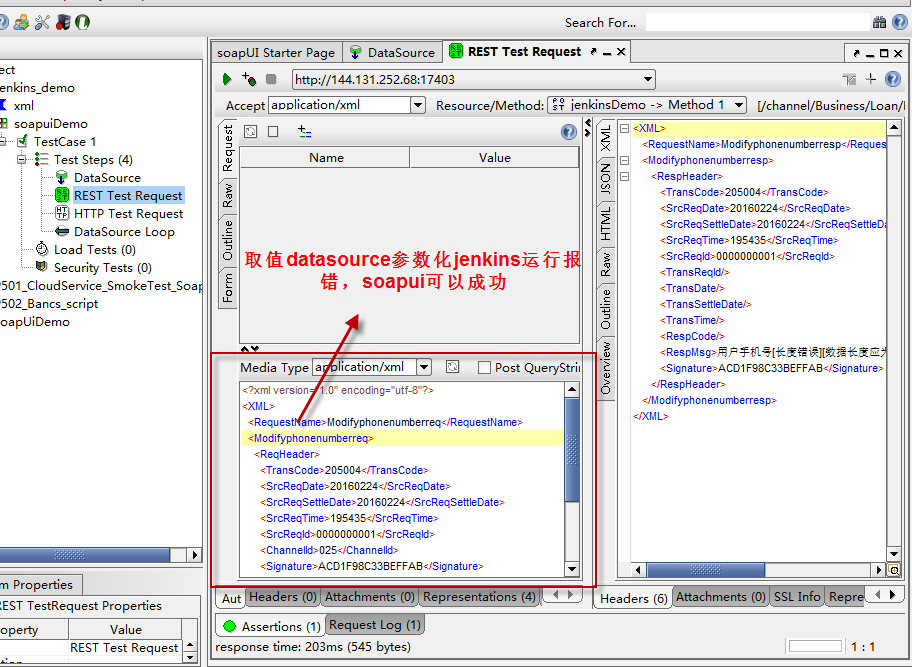
Datasource

${#Project#testdatafile}

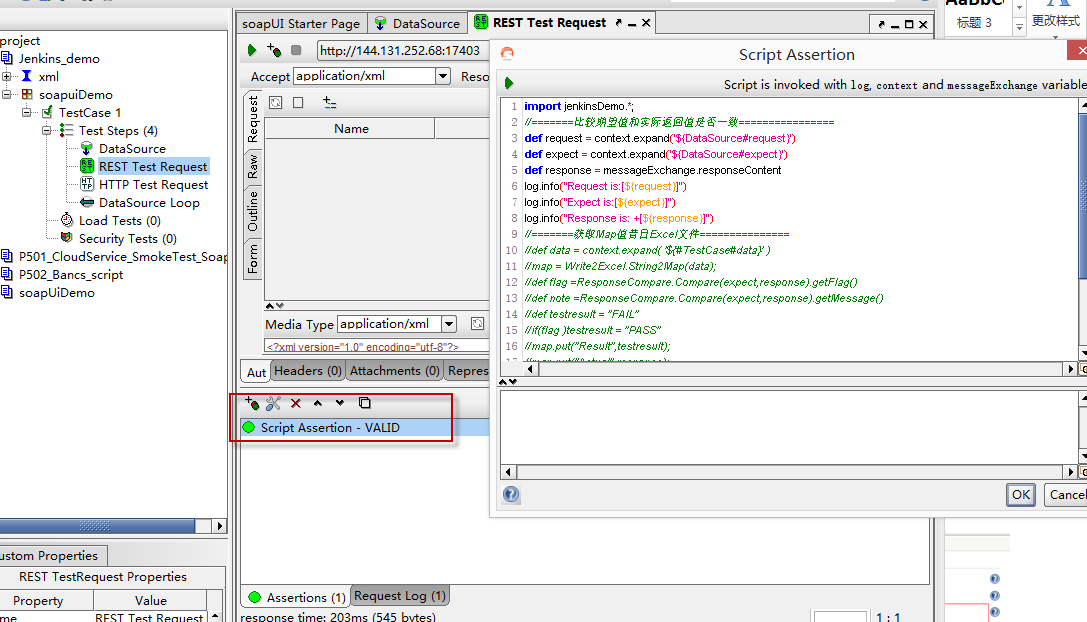
目前jenkins运行testsuit从datasource取数据抛异常估计与断言脚本有一定的关系。。

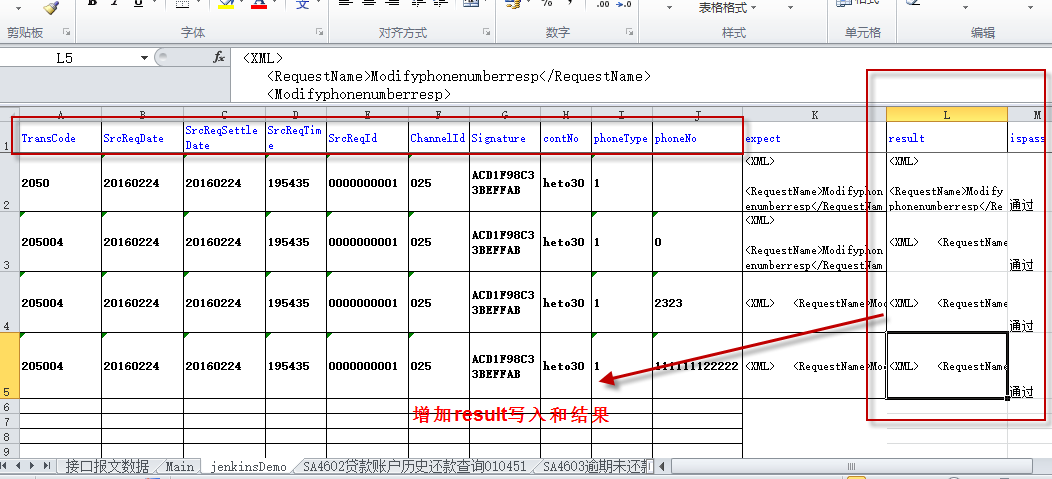
**testsuit request**

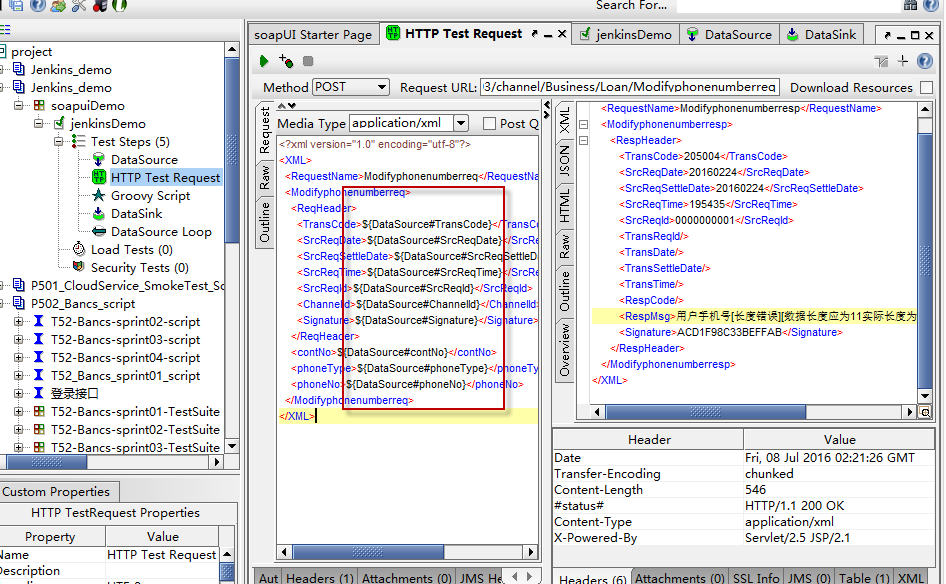
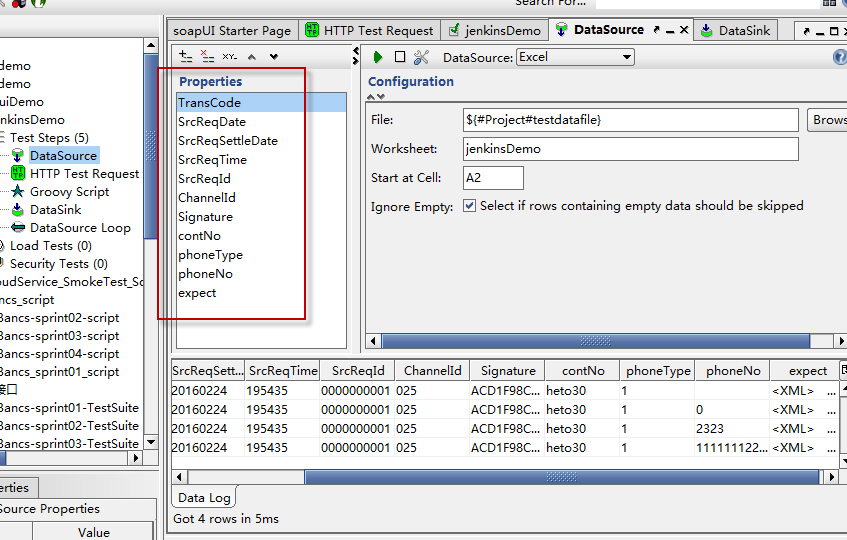


**断言**



**Soapui运行测试脚本没问题保存.xml文件**

**TestSuit重新修改datasource数据参数化模板，增加Datasink写入datasource**



import jenkinsDemo.\*;

//=======比较期望值和实际返回值是否一致================

def request = context.expand('${DataSource#request}')

def expect = context.expand('${DataSource#expect}')

def response = messageExchange.responseContent

log.info("Request is:[${request}]")

log.info("Expect is:[${expect}]")

log.info("Response is: +[${response}]")

//=======获取Map值昔日Excel文件===============

//def data = context.expand( '${#TestCase#data}' )

//map = Write2Excel.String2Map(data);

//def flag =ResponseCompare.Compare(expect,response).getFlag()

//def note =ResponseCompare.Compare(expect,response).getMessage()

//def testresult = "FAIL"

//if(flag )testresult = "PASS"

//map.put("Result",testresult);

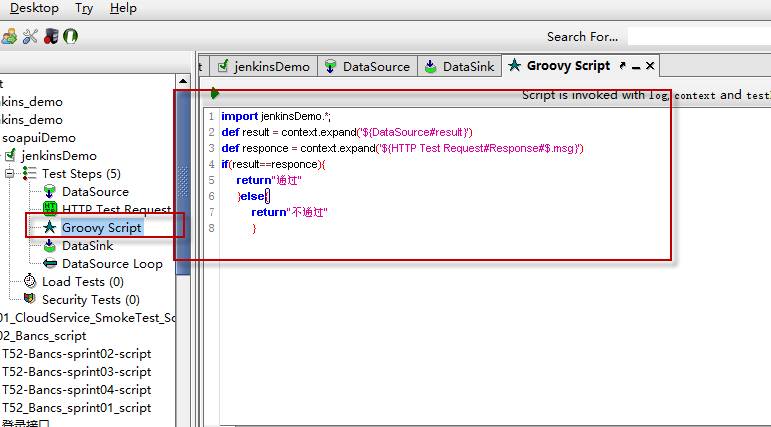
//map.put("Actual",response);

//map.put("Note",note);

//==============ExcelReport所显示的内容==============

//Write2Excel.ExcelReport(map)

//log.info(ResponseCompare.Compare(expect,response).getMessage())

//assert ResponseCompare.Compare(expect,response).getFlag()

import jenkinsDemo.\*;

def result = context.expand('${DataSource#result}')

def responce = context.expand('${HTTP Test Request#Response#$.msg}')

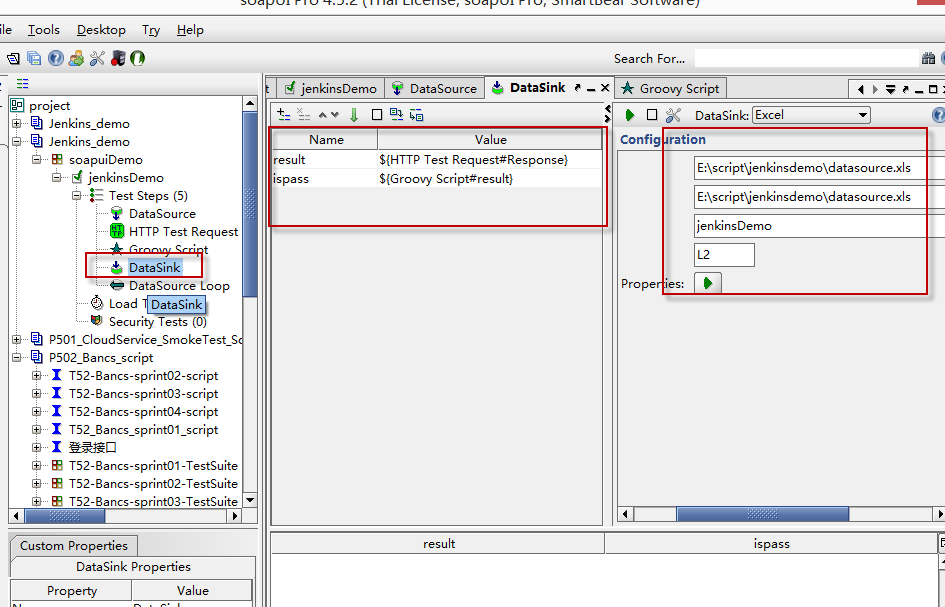
if(result==responce){

return"通过"

}else{

return"不通过"

}



## 四、Jenlins-maven-soapuiproject

##### 1、maven pom.xml

标黄为需要修改地方

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-v4\_0\_0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.test</groupId>

<artifactId>mytest</artifactId>

<packaging>jar</packaging>

<version>1.0-SNAPSHOT</version>

<name>mytest</name>

<url>http://maven.apache.org</url>

<dependencies>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>3.8.1</version>

<scope>test</scope>

</dependency>

</dependencies>

<pluginRepositories>

<pluginRepository>

<id>eviwarePluginRepository</id>

<url>http://www.eviware.com/repository/maven2/</url>

</pluginRepository>

</pluginRepositories>

<build>

<plugins>

<plugin>

<groupId>com.smartbear.soapui</groupId>

<artifactId>soapui-maven-plugin</artifactId>

<version>5.1.2-m-SNAPSHOT</version>

<configuration>

<!--soapUI project file location-->

<projectFile>F:\svn\jenkinsdemo\xml-soapui-project.xml</projectFile>

<!--output file location-->

<outputFolder>soapui-reports</outputFolder>

<!--soapui testsuite name-->

<testSuite>soapuiDemo</testSuite>

<junitReport>true</junitReport>

<exportwAll>true</exportwAll>

<printReport>true</printReport>

<testFailIgnore>true</testFailIgnore>

</configuration>

<executions>

<execution>

<id>soapUI</id>

<phase>test</phase>

<goals>

<goal>test</goal>

</goals>

</execution>

</executions>

</plugin>

</plugins>

</build>

<reporting>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-surefire-report-plugin</artifactId>

<version>2.18.1</version>

</plugin>

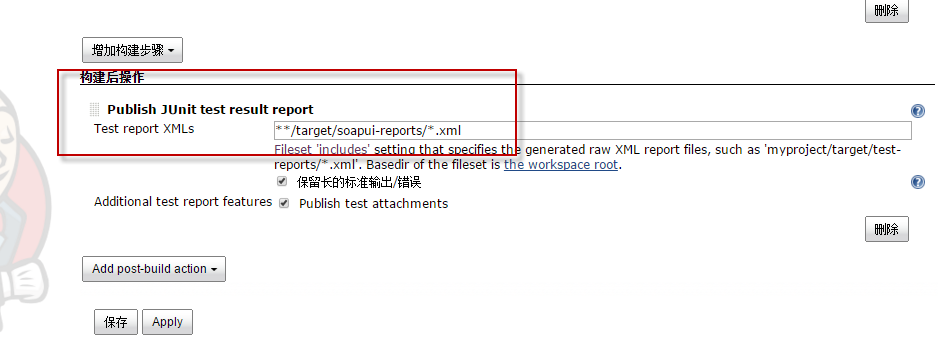
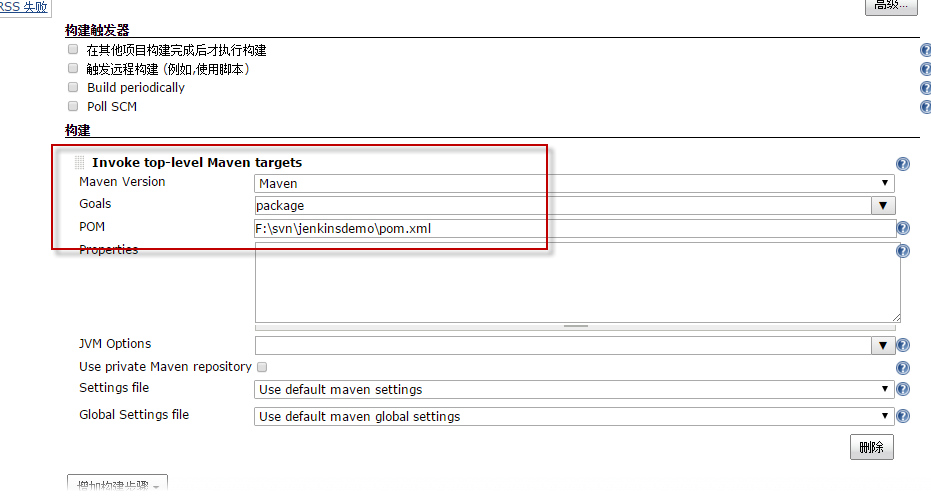
</plugins>

</reporting>

</project>

##### 2、jenkins-soapui

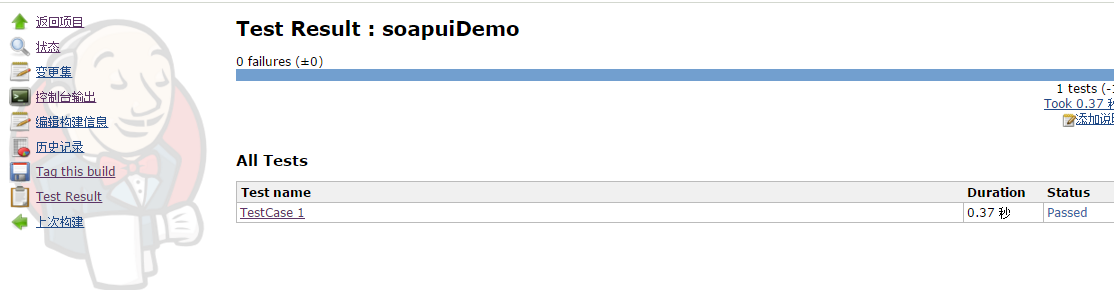
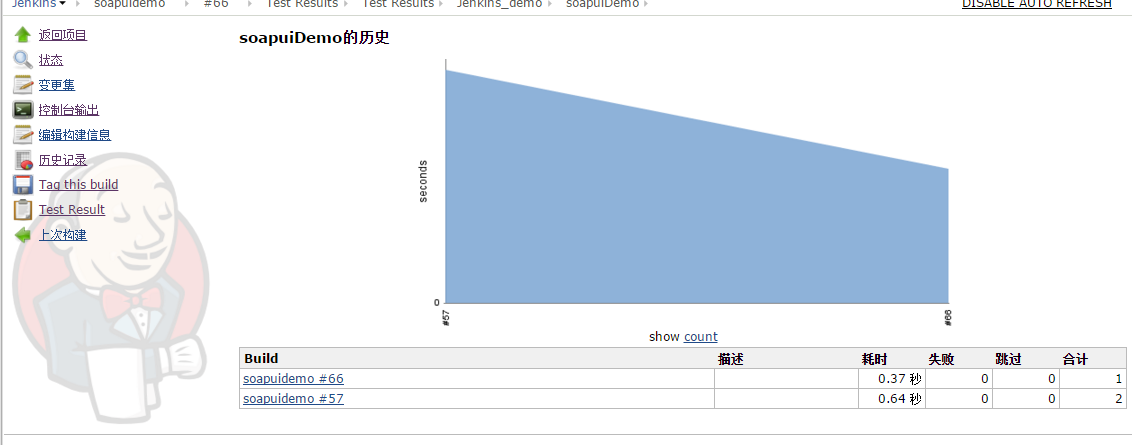
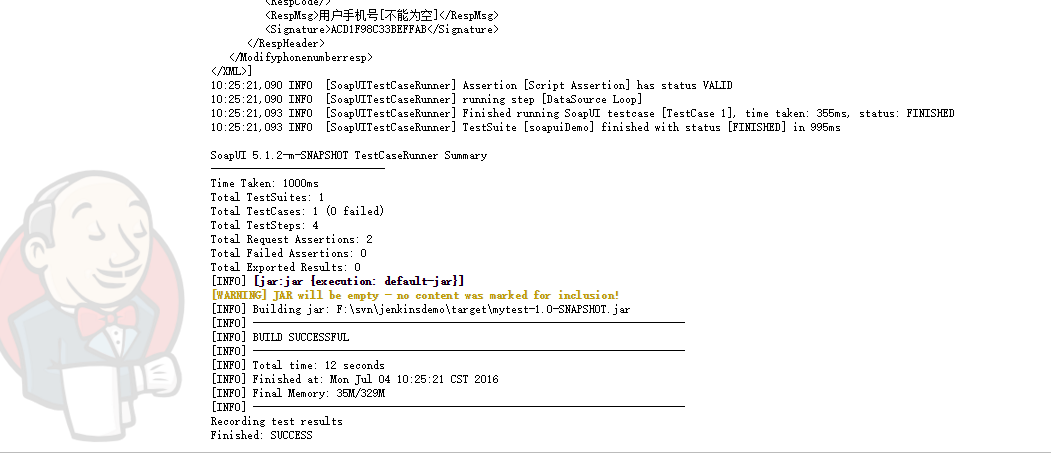
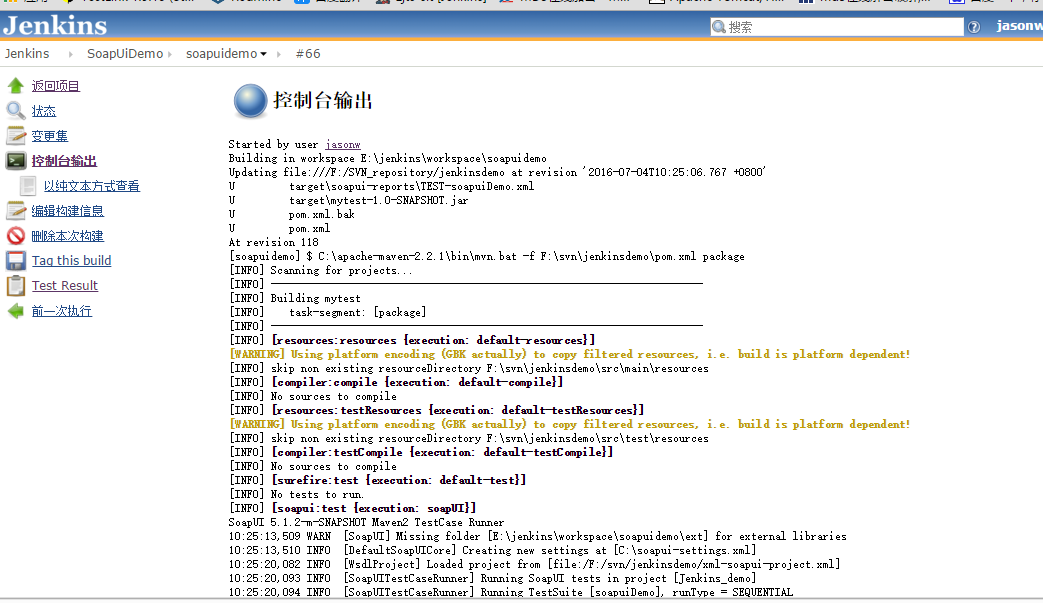
**构建一个自由风格的软件项目，调用maven pom.xml**



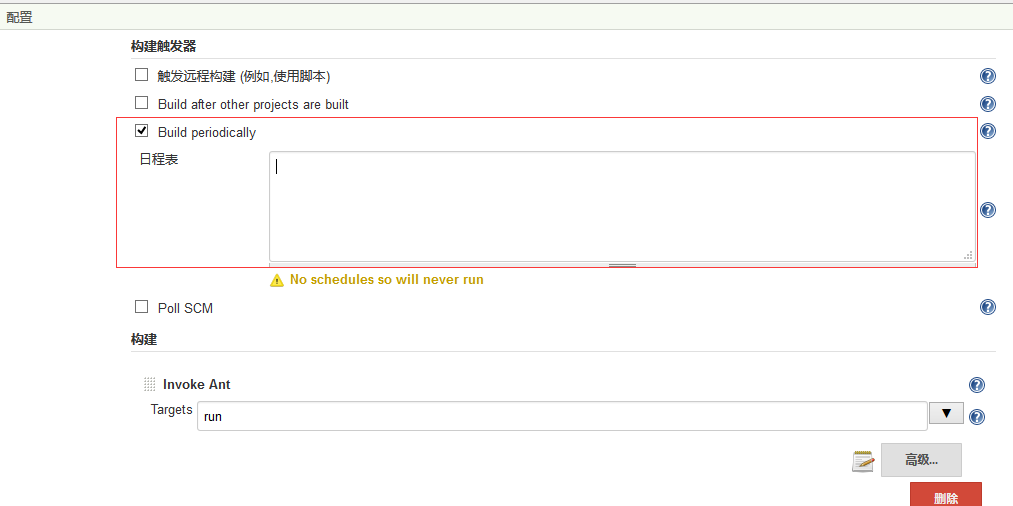
Report xmls 与pom.xml <outputFolder>target/soapui-reports</outputFolder>一致

后续可以调试生成html report

##### 3、构建控台输出



## 五、Jenlins构建触发器



如：设每一个小时我们需要重新构建一次。选择 Build periodically，在 Schedule 中填写 0 \* \* \* \*。

第一个参数代表的是分钟 minute，取值 0~59；

第二个参数代表的是小时 hour，取值 0~23；

第三个参数代表的是天 day，取值 1~31；

第四个参数代表的是月 month，取值 1~12；

最后一个参数代表的是星期 week，取值 0~7，0 和 7 都是表示星期天。

所以 0 \* \* \* \* 表示的就是每个小时的第 0 分钟执行一次构建。

**PS:**

**Poll SCM**：定时检查源码变更（根据SCM软件的版本号），如果有更新就checkout最新code下来，然后执行构建动作。我的配置如下：

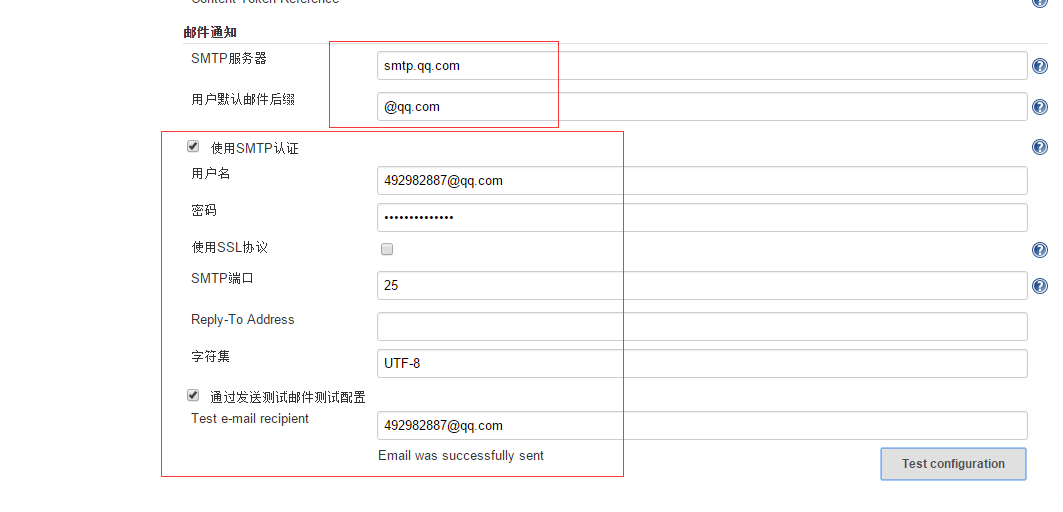
\*/5 \* \* \* \*  （每5分钟检查一次源码变化）

**Build periodically**：周期进行项目构建（它不care源码是否发生变化），我的配置如下：

0 2 \* \* \*  （每天2:00 必须build一次源码）

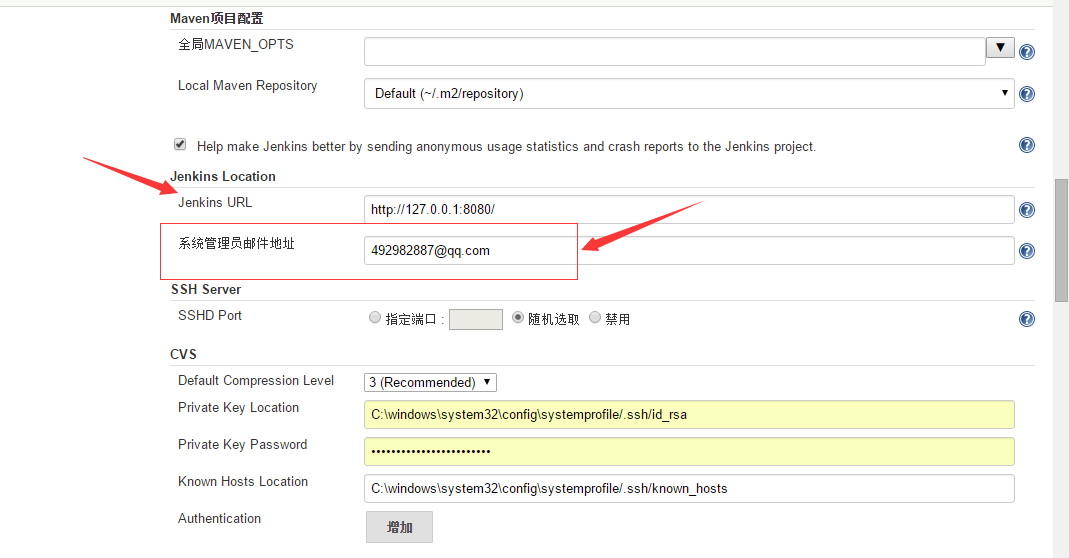
## 六、发送邮件设置

#### 发送邮件设置，插件管理安装[Email Extension Plugin](http://wiki.jenkins-ci.org/display/JENKINS/Email-ext+plugin)



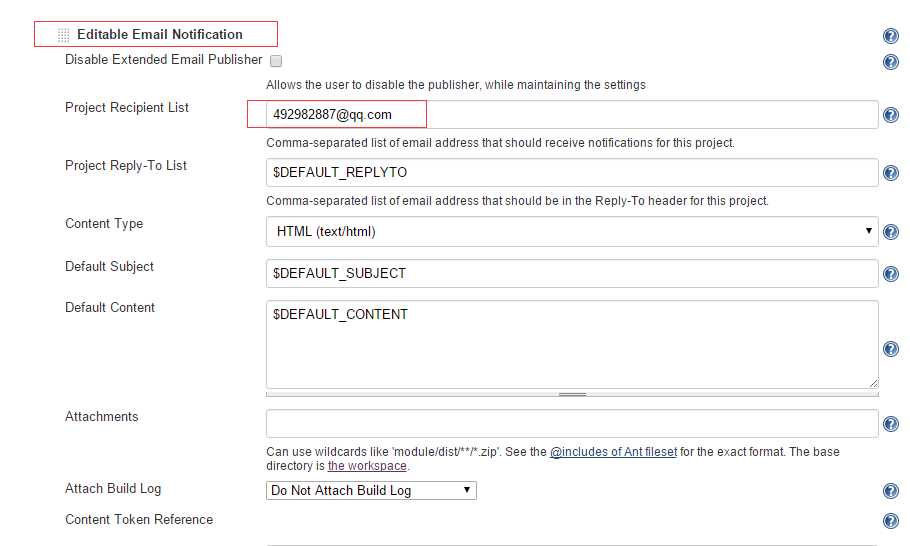
在设置Jenkins URL底下有一个文本框System Admin e-mail address，这里要设置发送者的邮箱地址，否则会报错

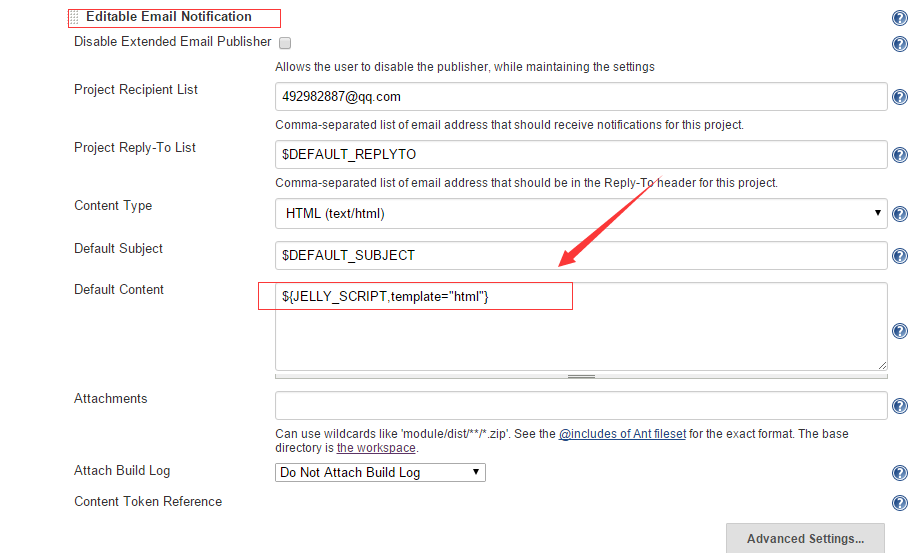
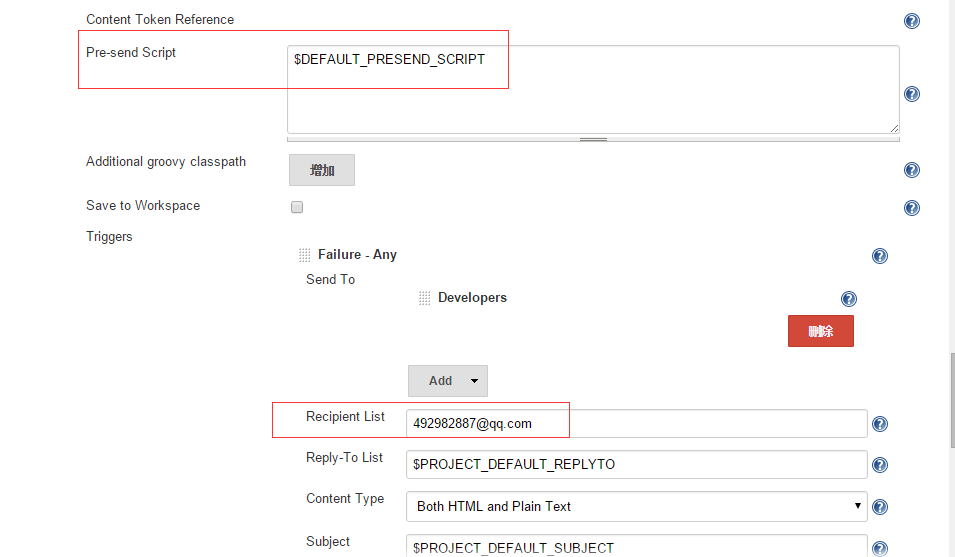
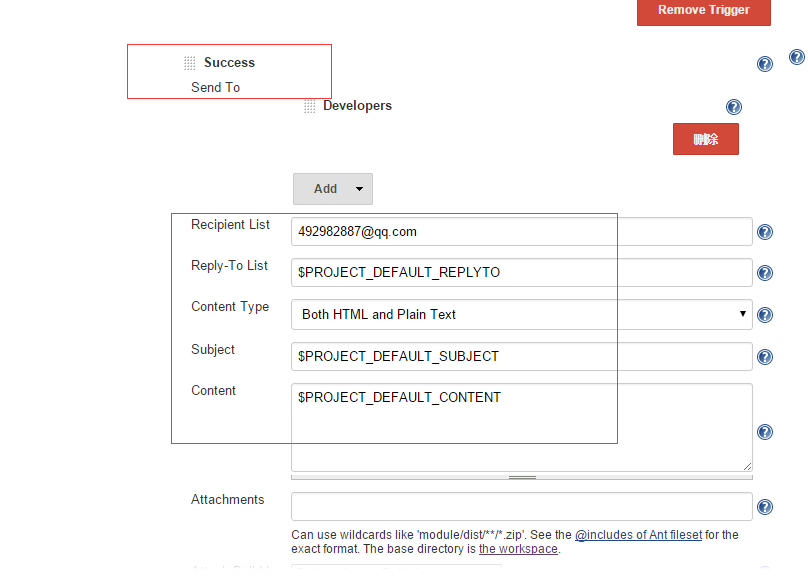
报错信息：  
Failed to send out e-mail  
  
com.sun.mail.smtp.SMTPSendFailedException: 501 mail from address must be same as authorization user;  nested exception is: com.sun.mail.smtp.SMTPSenderFailedException: 501 mail from address must be same as authorization user  
  
at hudson.util.PluginServletFilter$1.doFilter(PluginServletFilter.java:95)



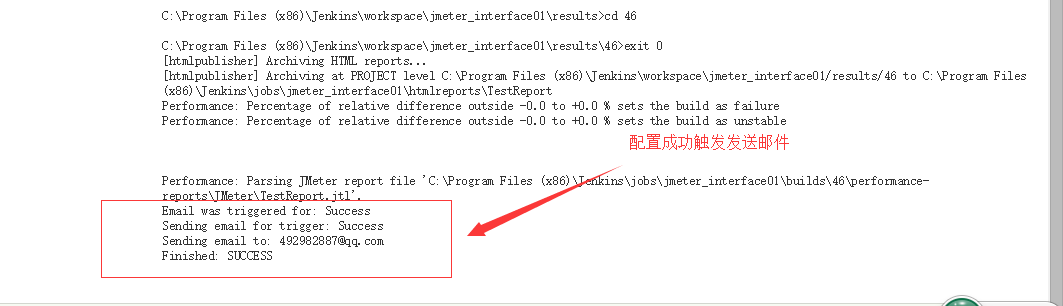
项目中增加构建步骤email notification





 ${JELLY\_SCRIPT,template="html"} JELLY是内置的插件，直接用即可  

Content Type选择html



收到邮件内容

