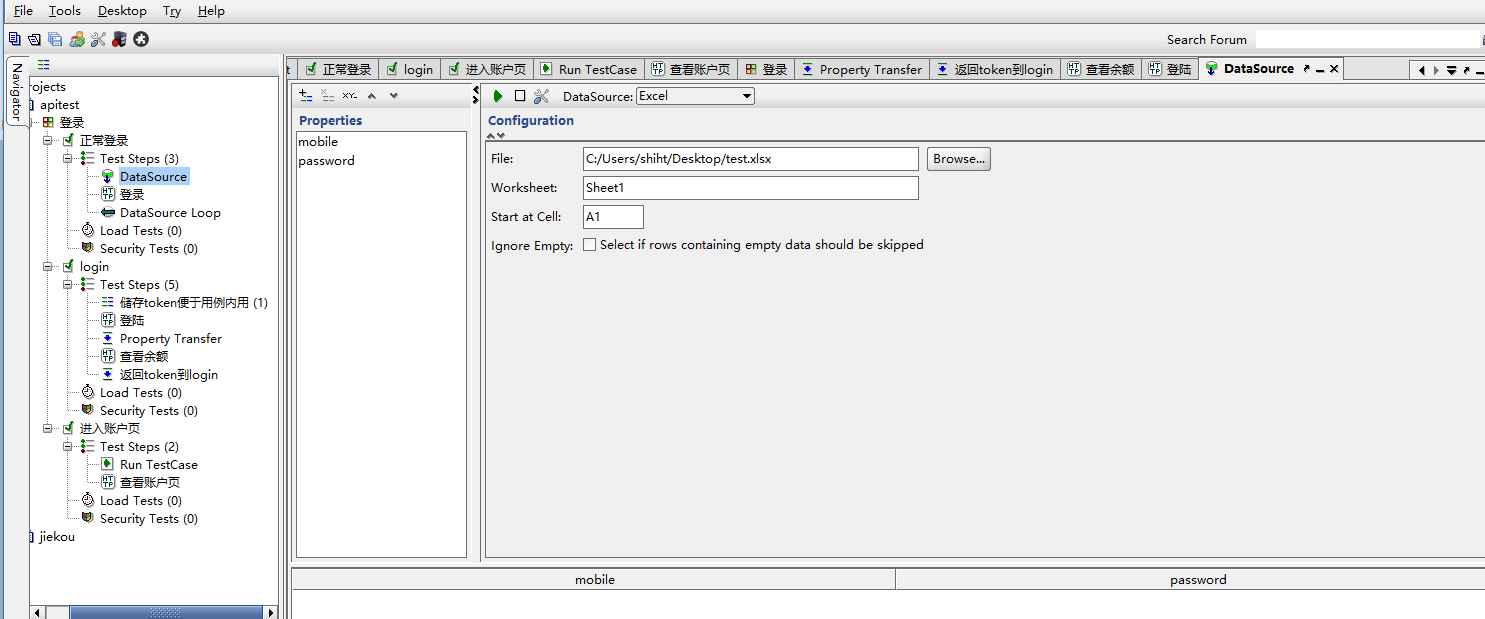
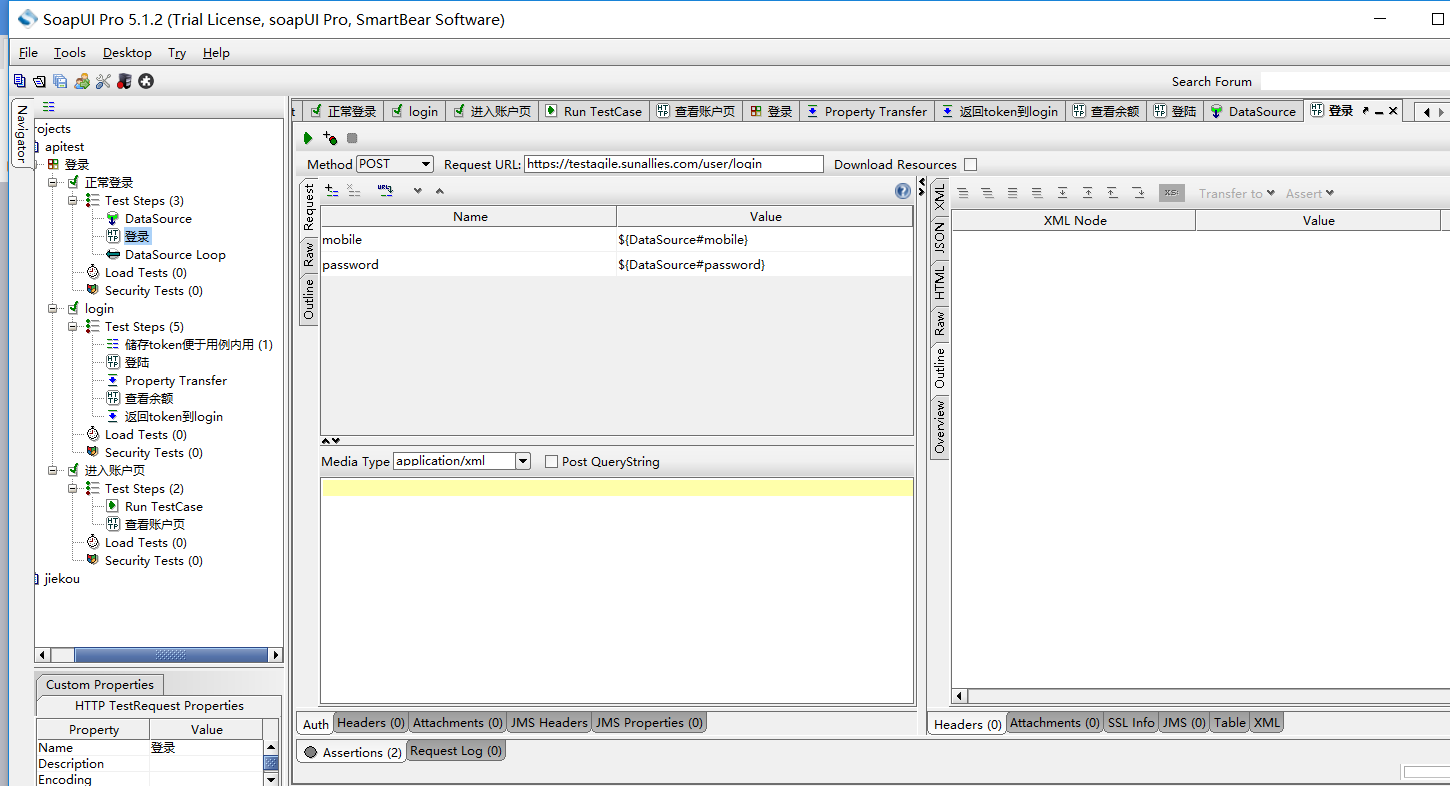
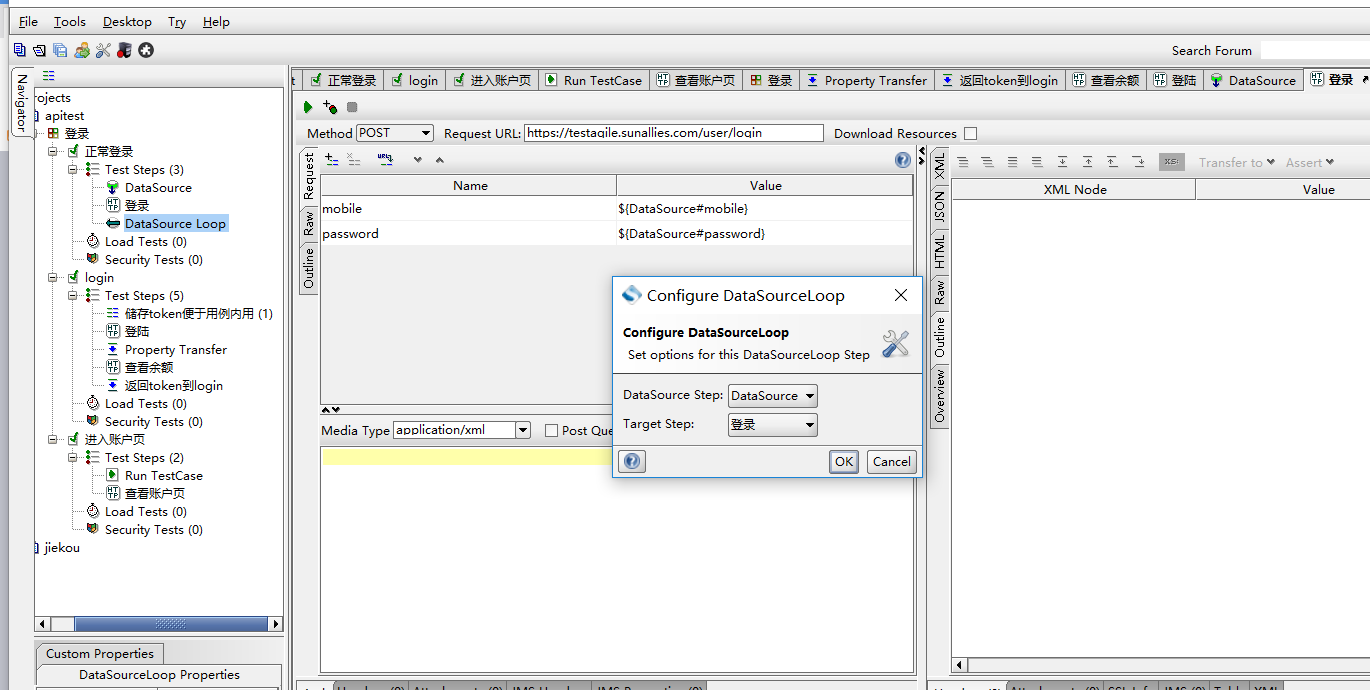
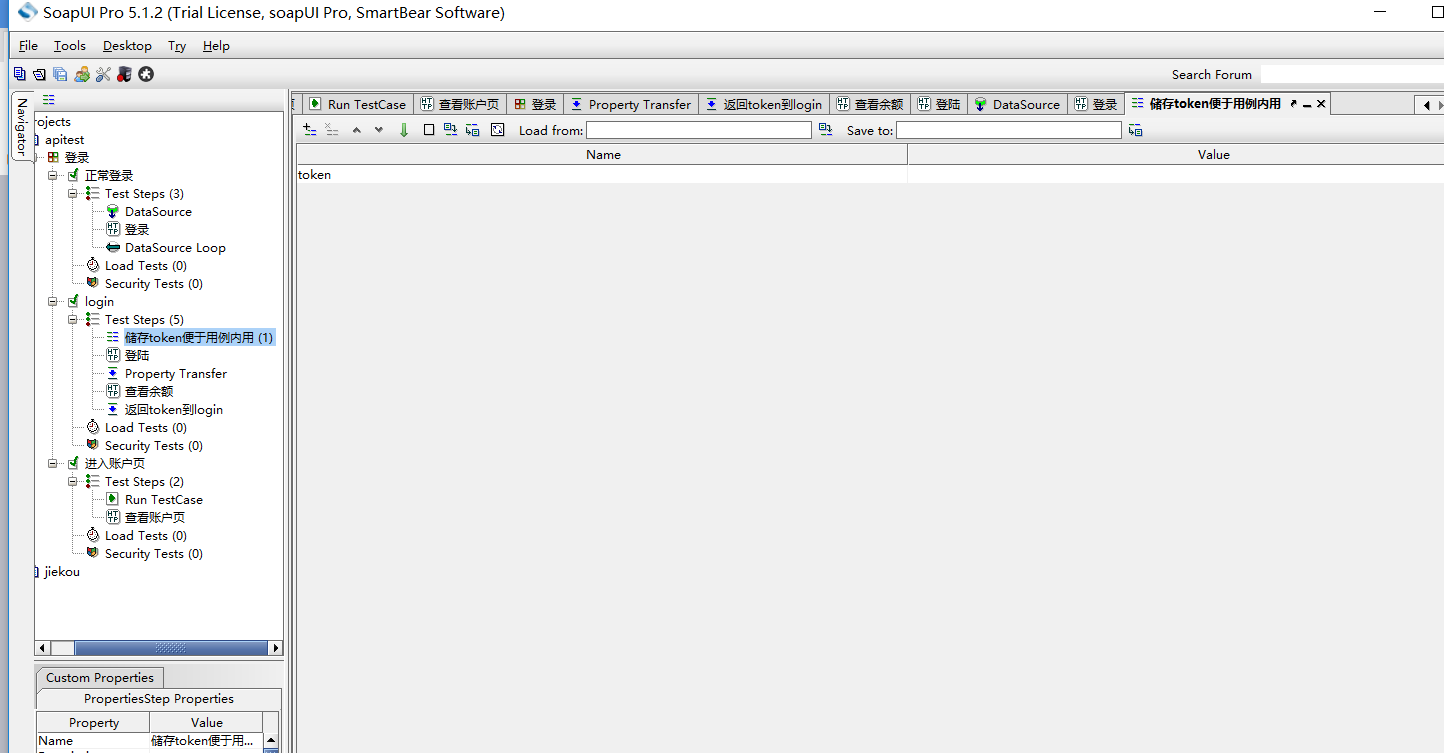
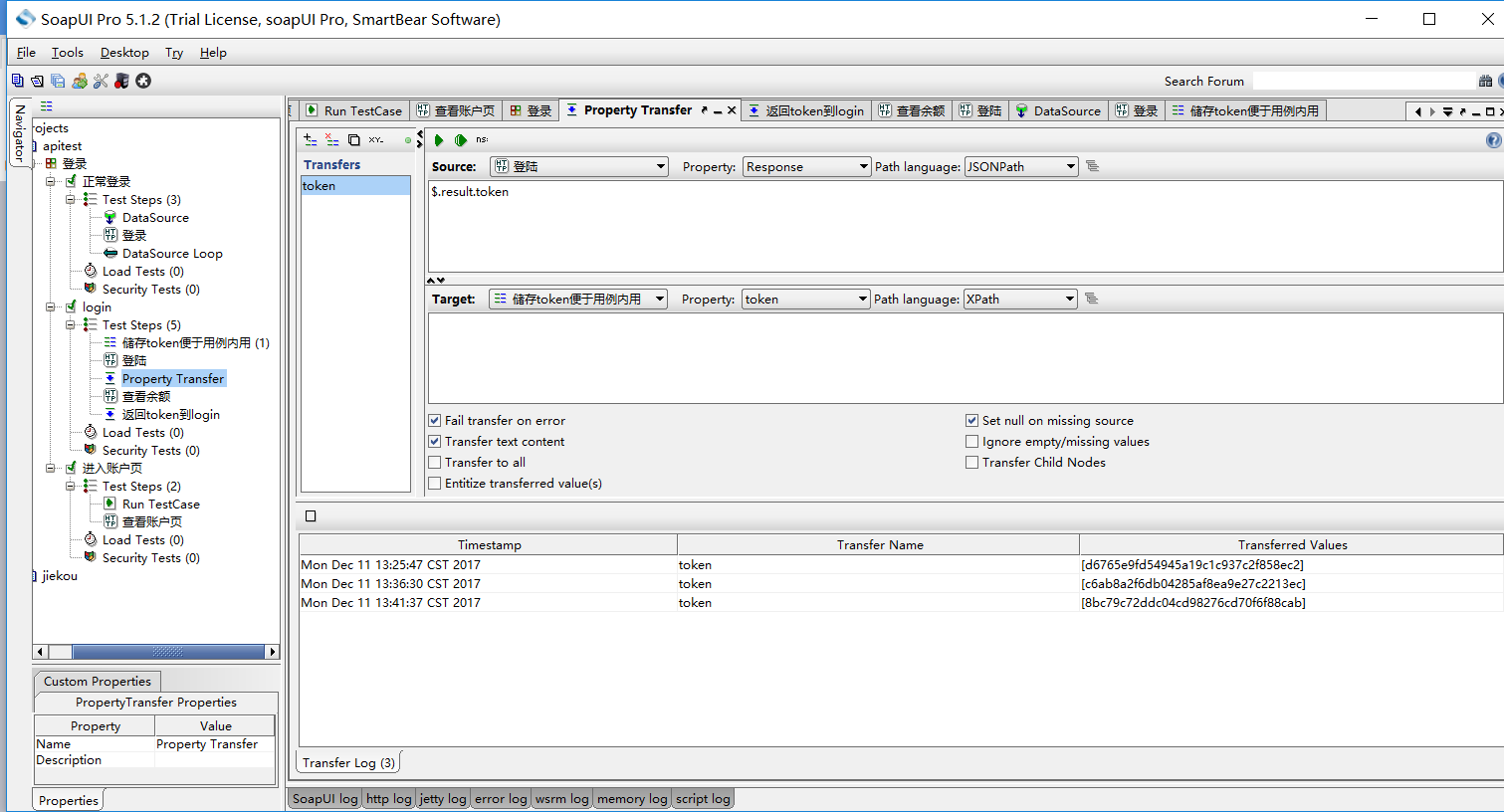
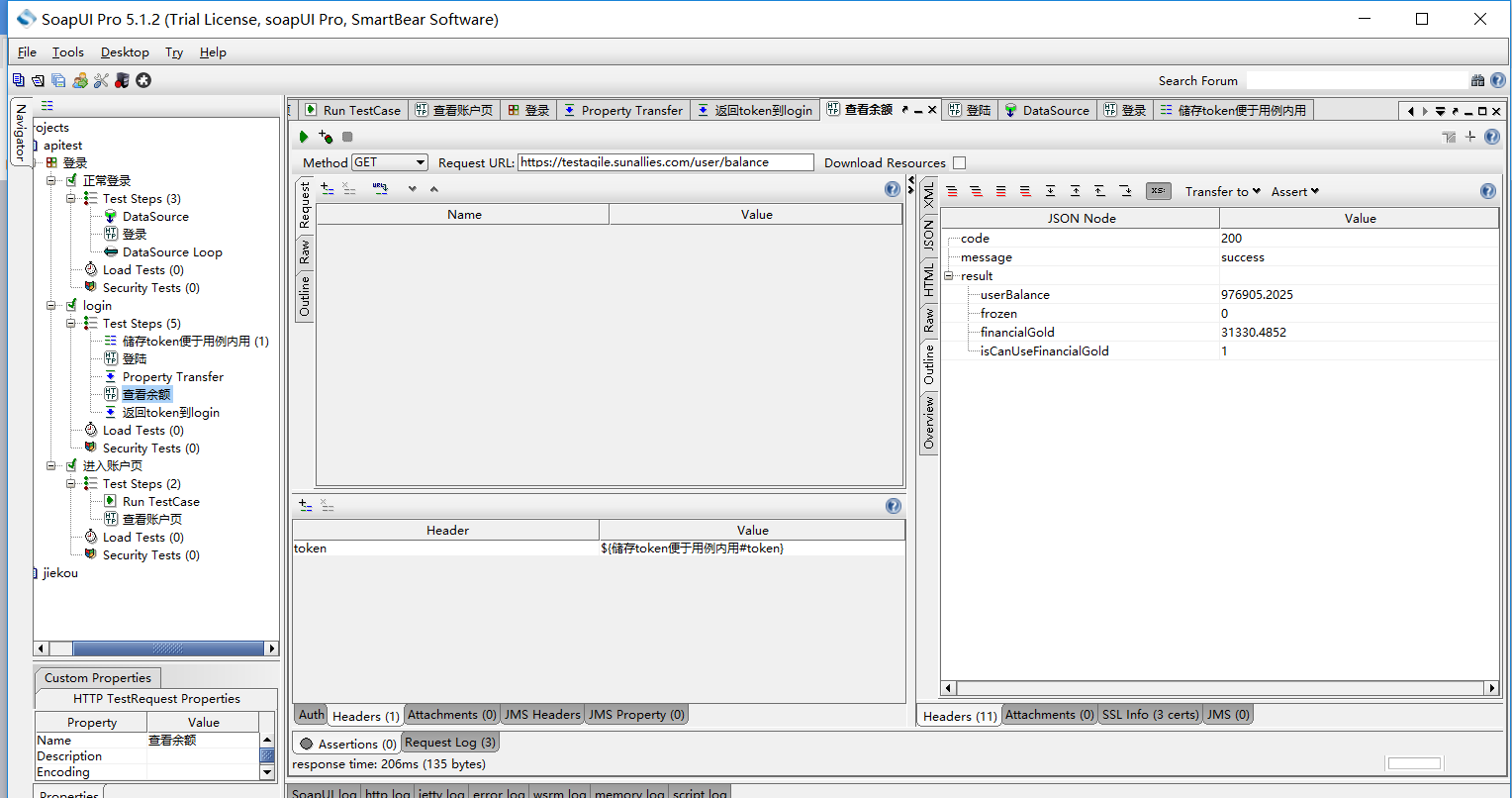
# 从excel中获取数据

Url中大小写不识别的问题，在左下角中输入就可以

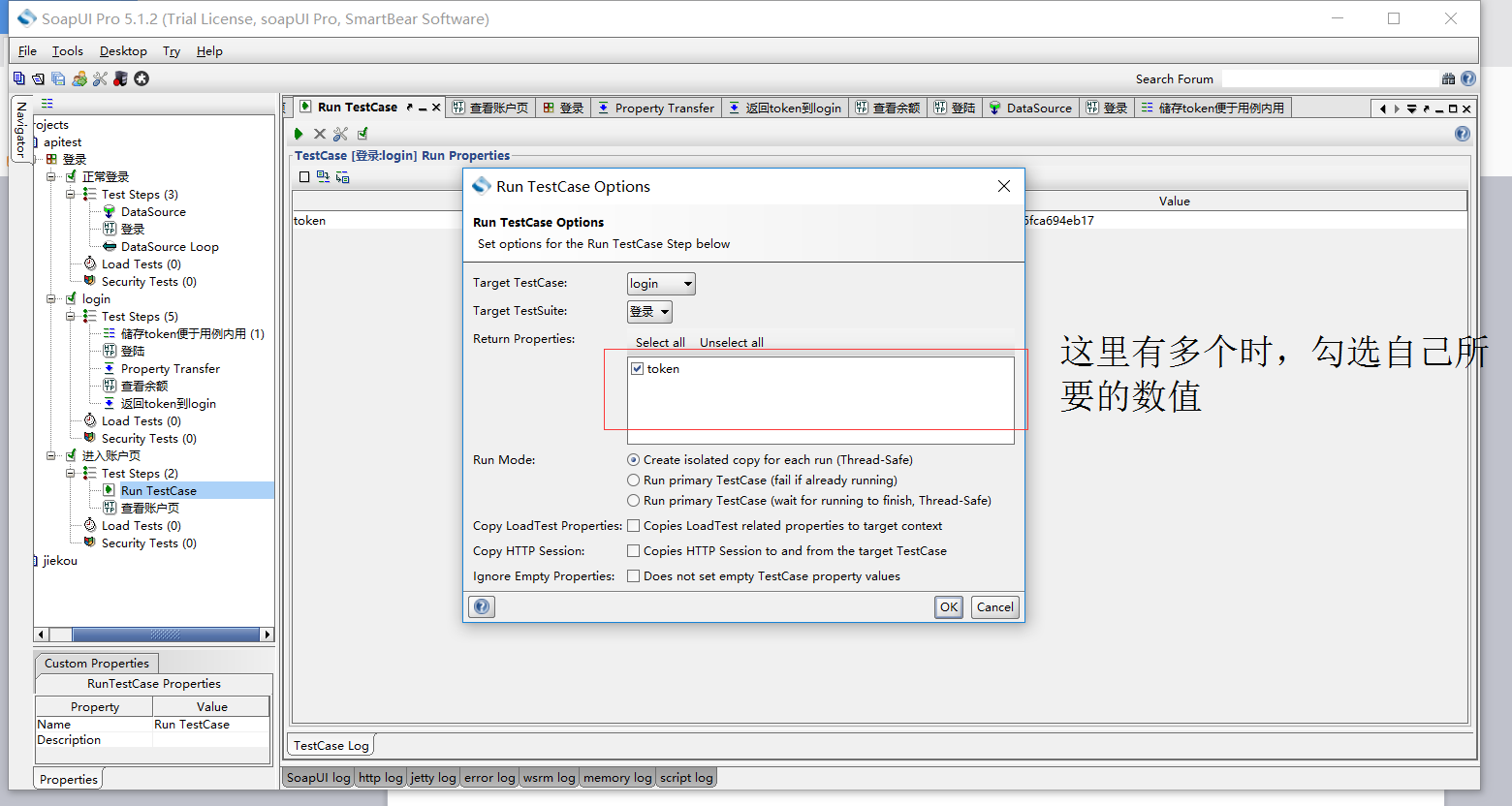
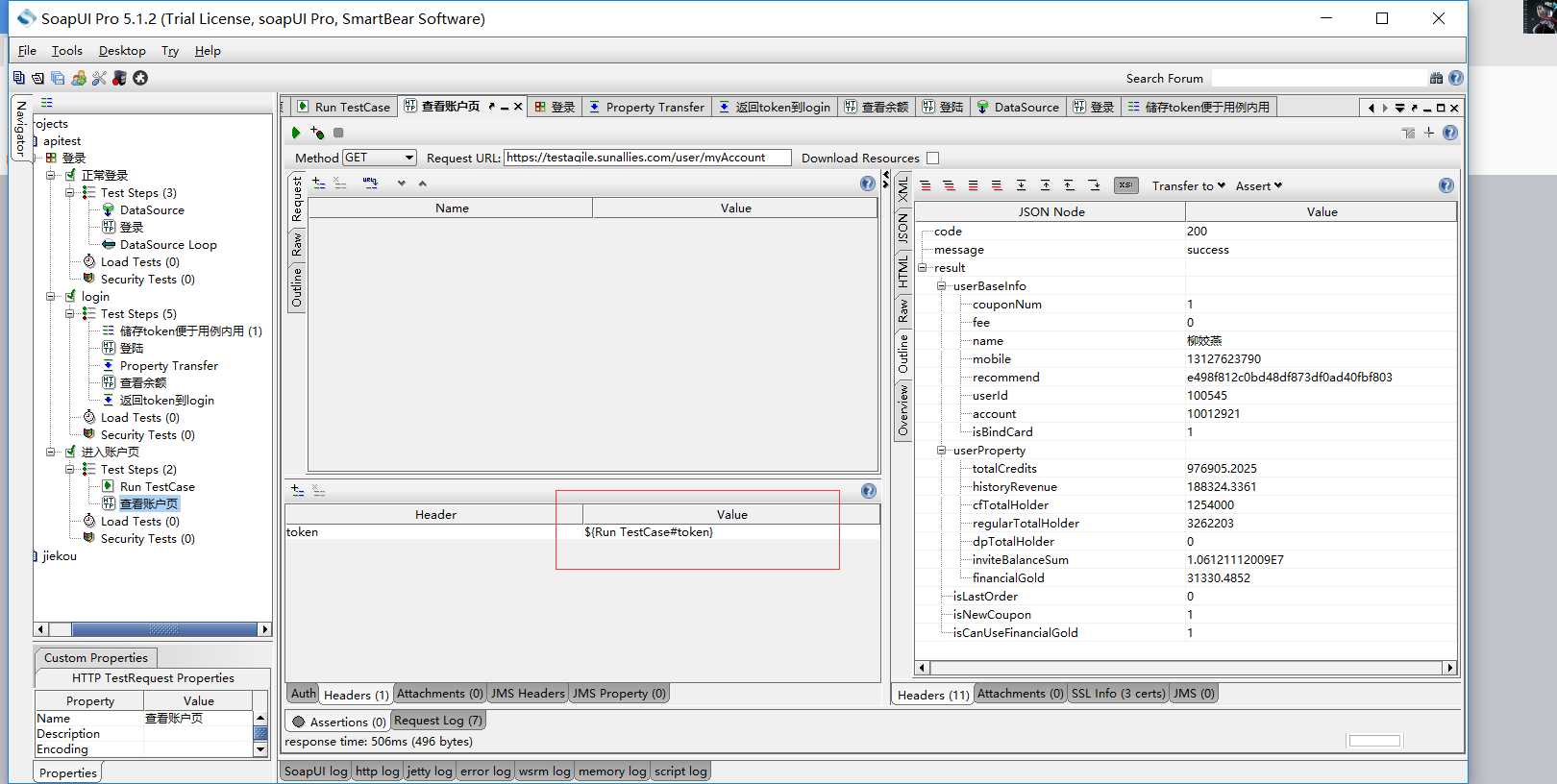
1. 从excel中读取数据：a、新建一个excel表格造需要的数据 b、在test case中新建datasource,将excel文件关联进去，如下图
2. 新建httprequest,设置value值从datasource中获取，如图
3. 新建datasourceloop,设置读取数据的循环，如图

# 一个test case用例间的数据传递

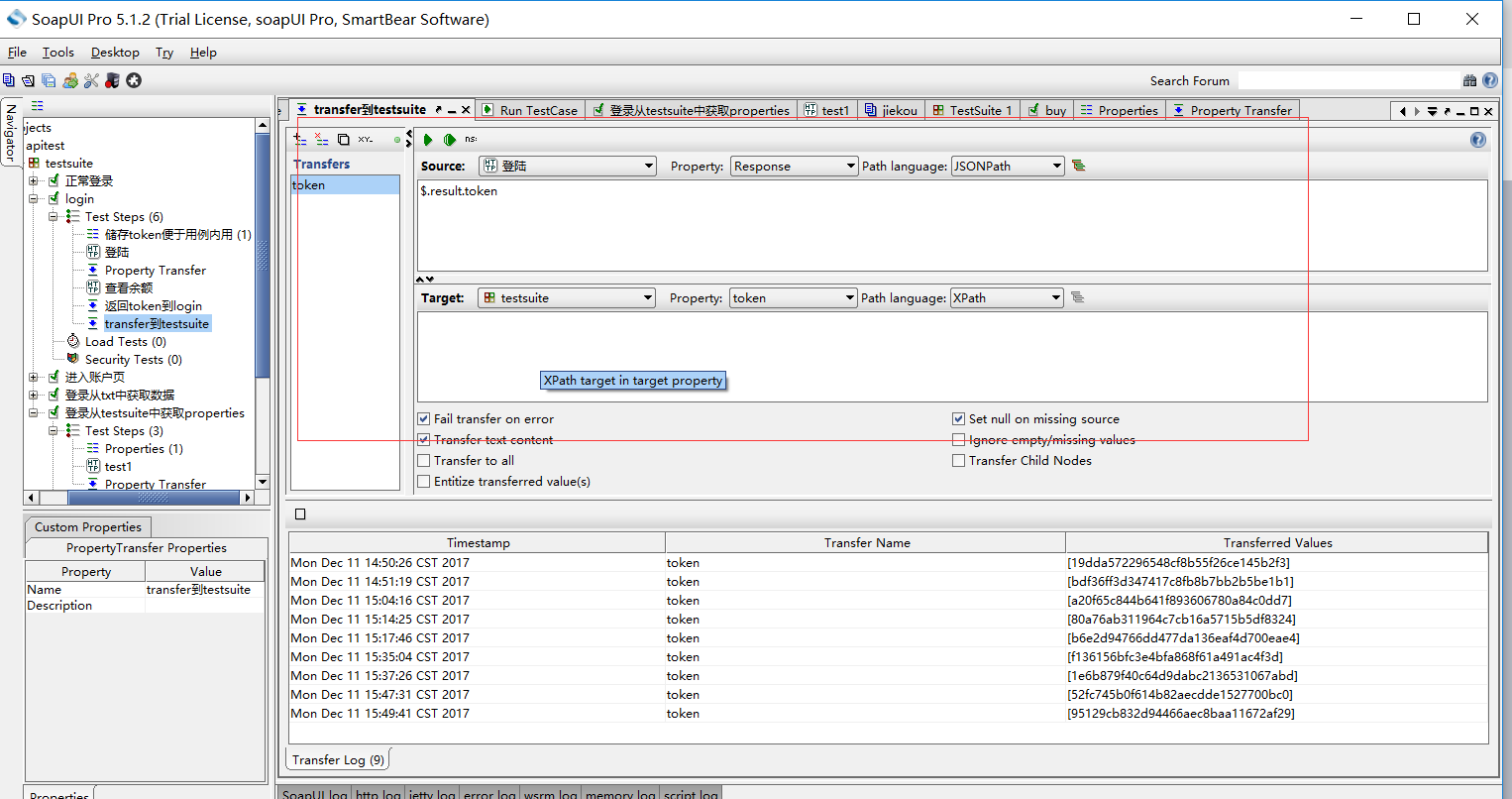
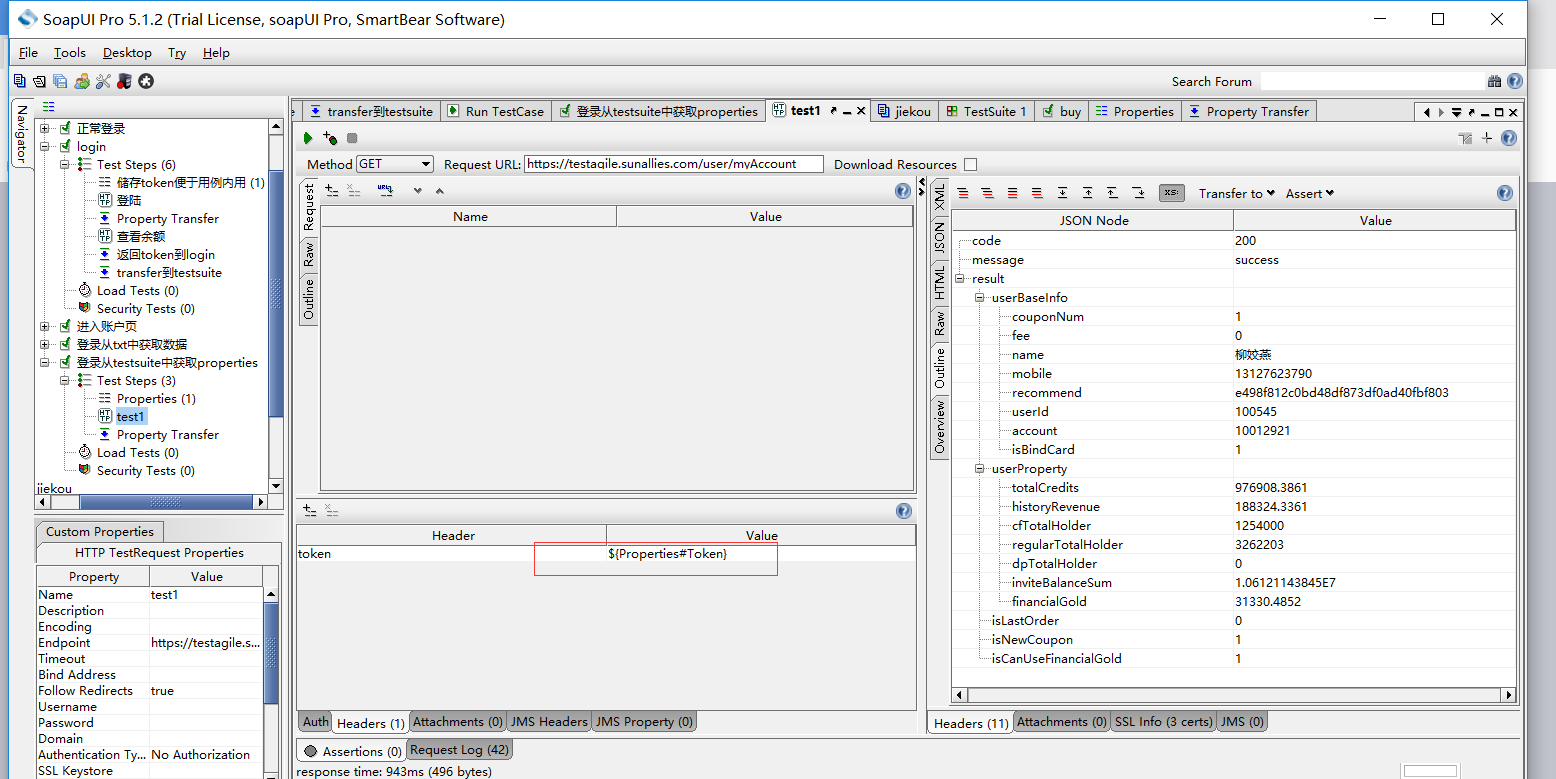
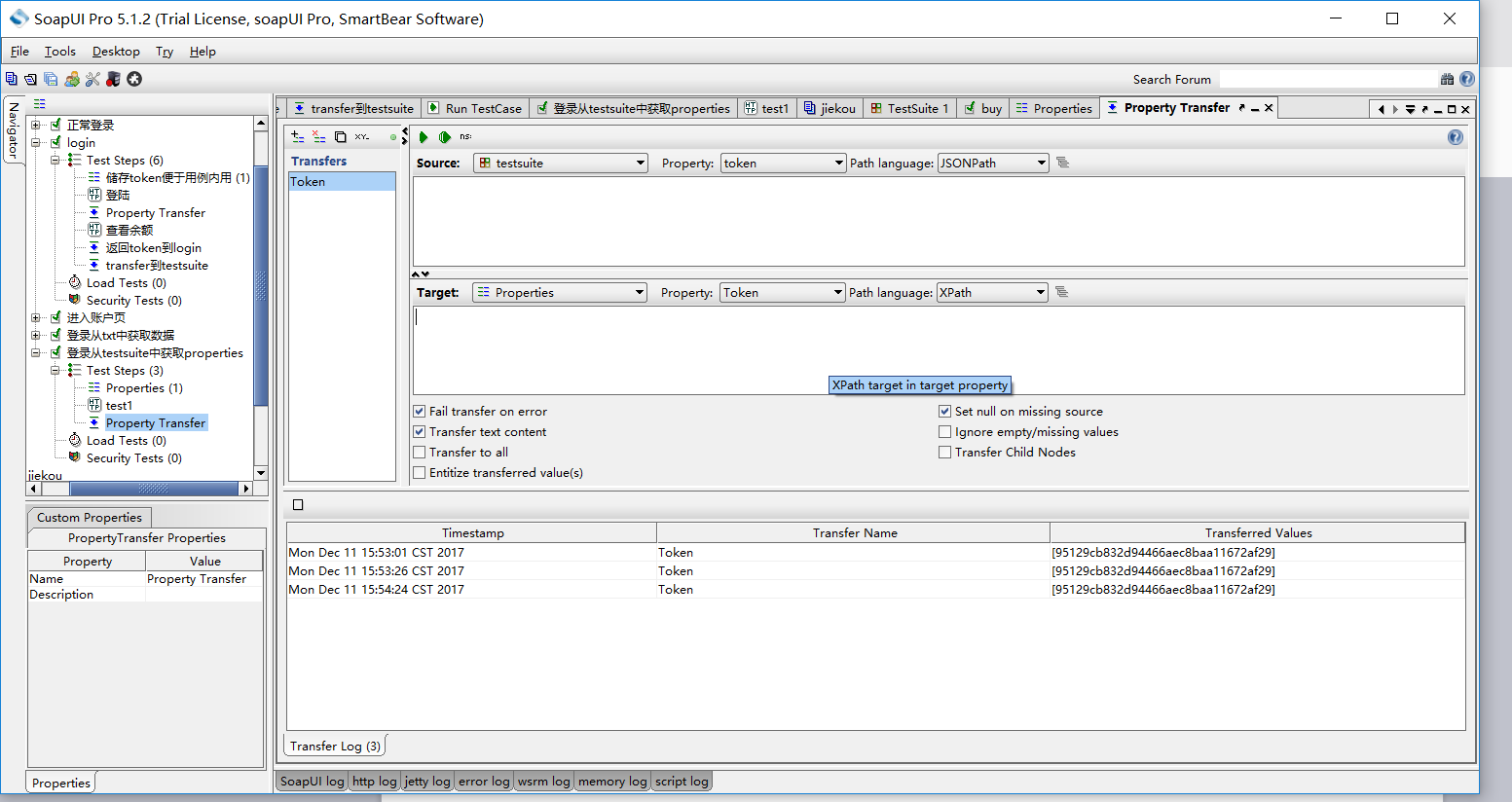
1. 新建http request A产生token
2. 在test case中新增一个Properties，用于存储token，如图
3. 新建一个Properties transfer 指定将数据从哪存储到哪，如图
4. 新建http request B ，并配置value从Properties中获取数据如图

注：get类型的请求参数应该写到headers中

# test case间的数据传递

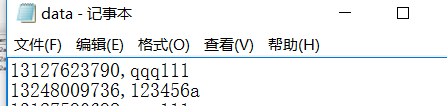
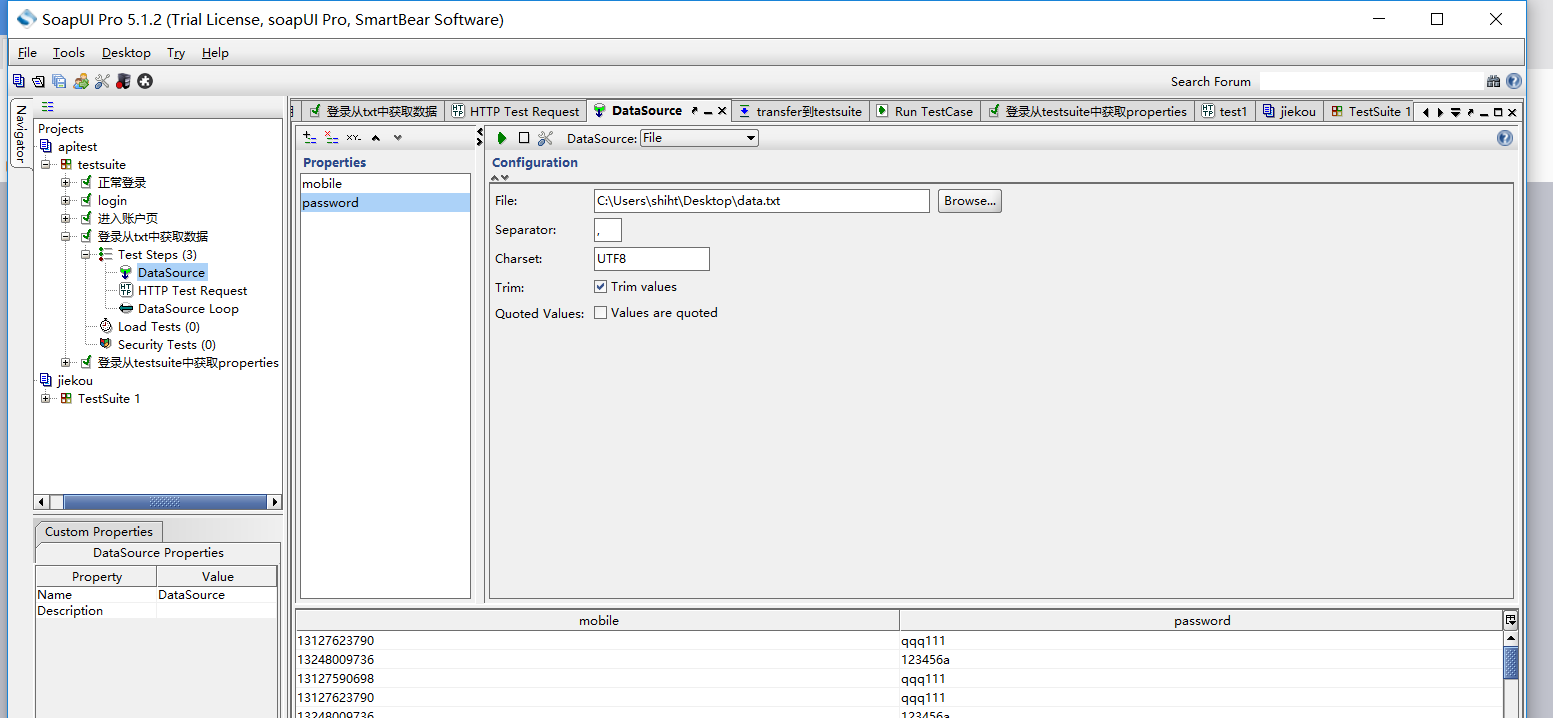
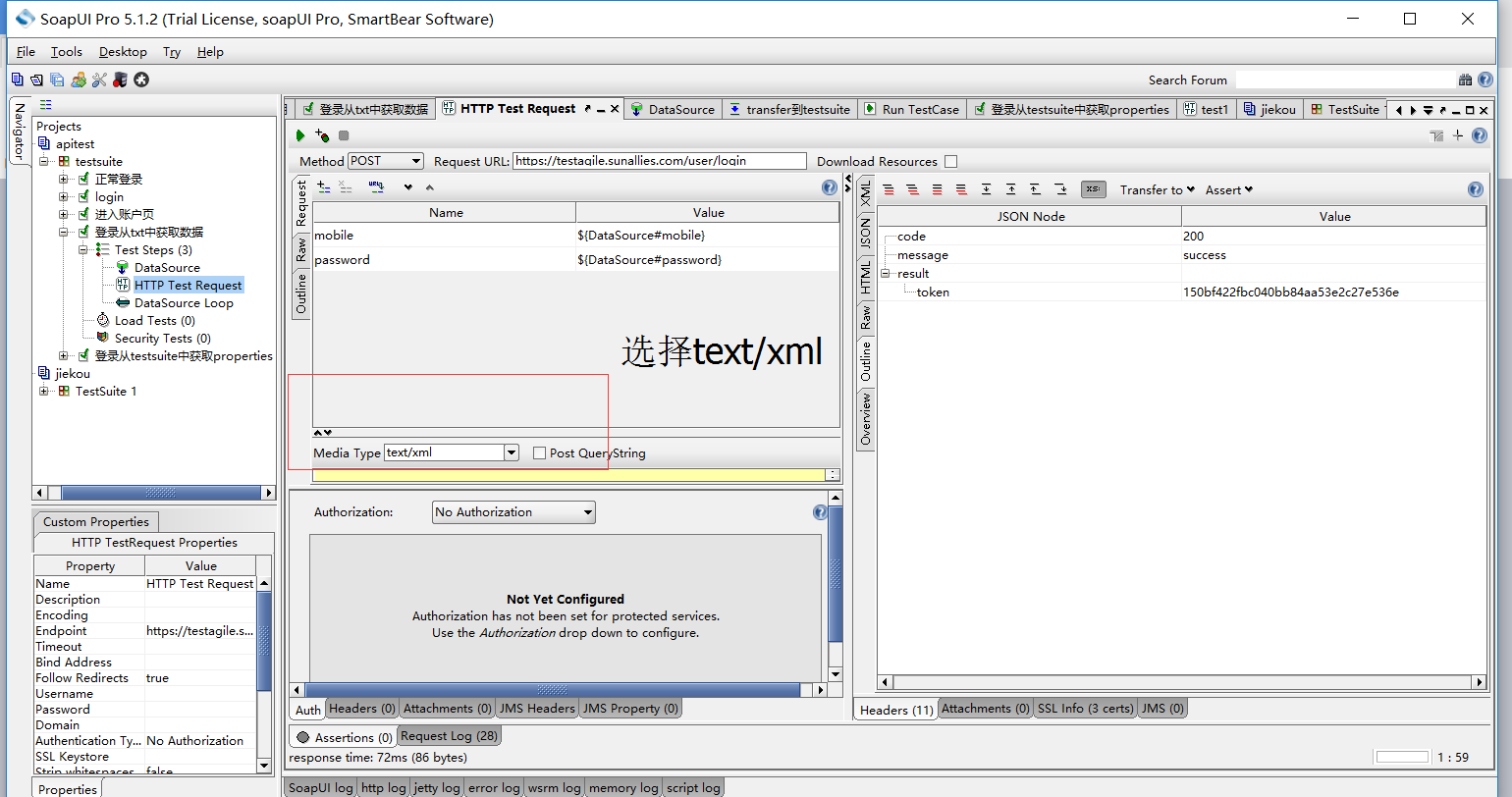
1. 先为test caseB新建一个Runtest case，设置从哪获取哪些数据，如图
2. 新建http request，指定value从上面的runtestcase中获取数据，如图

# 从testsuite获取数据到test case中

1. 在test suite中新建变量准备存储test caseA传过来的值
2. 在产生token的test caseA中新建一个properties transfer，将数据从test caseA传给test suite的properties，如图
3. 给test caseB新建一个properties
4. 新建test caseB，value从上面建properties里取值，如图
5. 新建一个properties transfer，将testsuite中的值传递给test caseB的properties

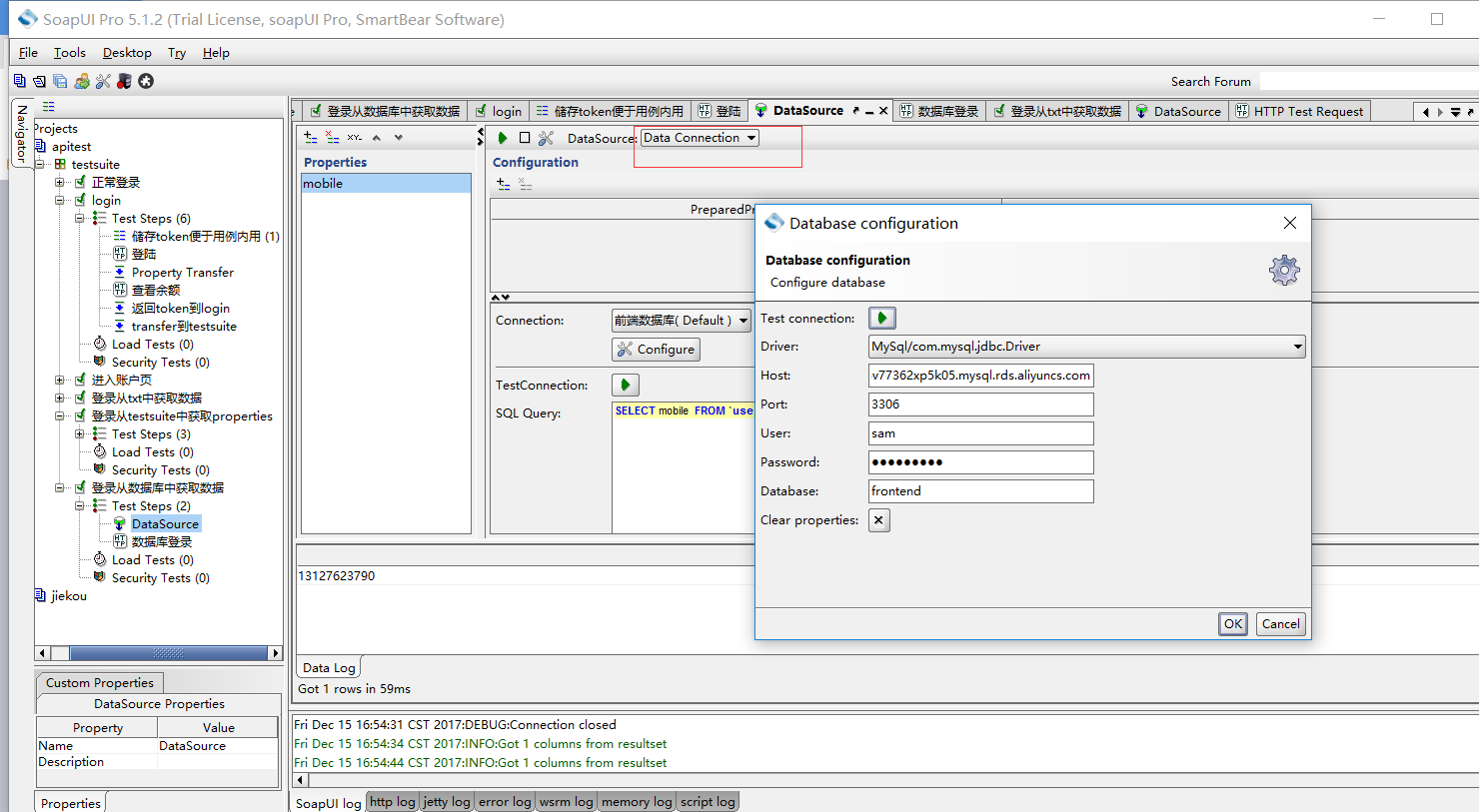
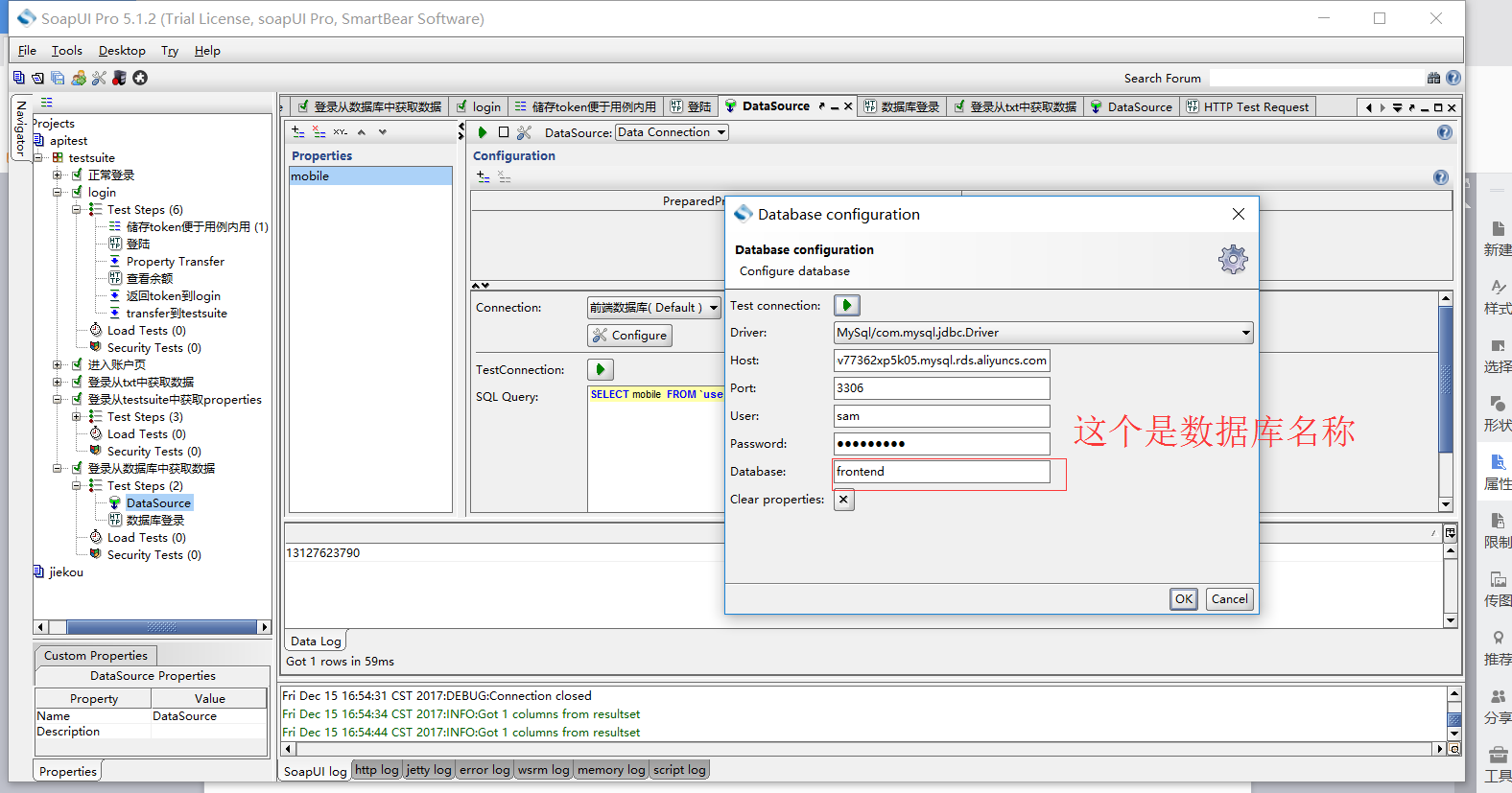
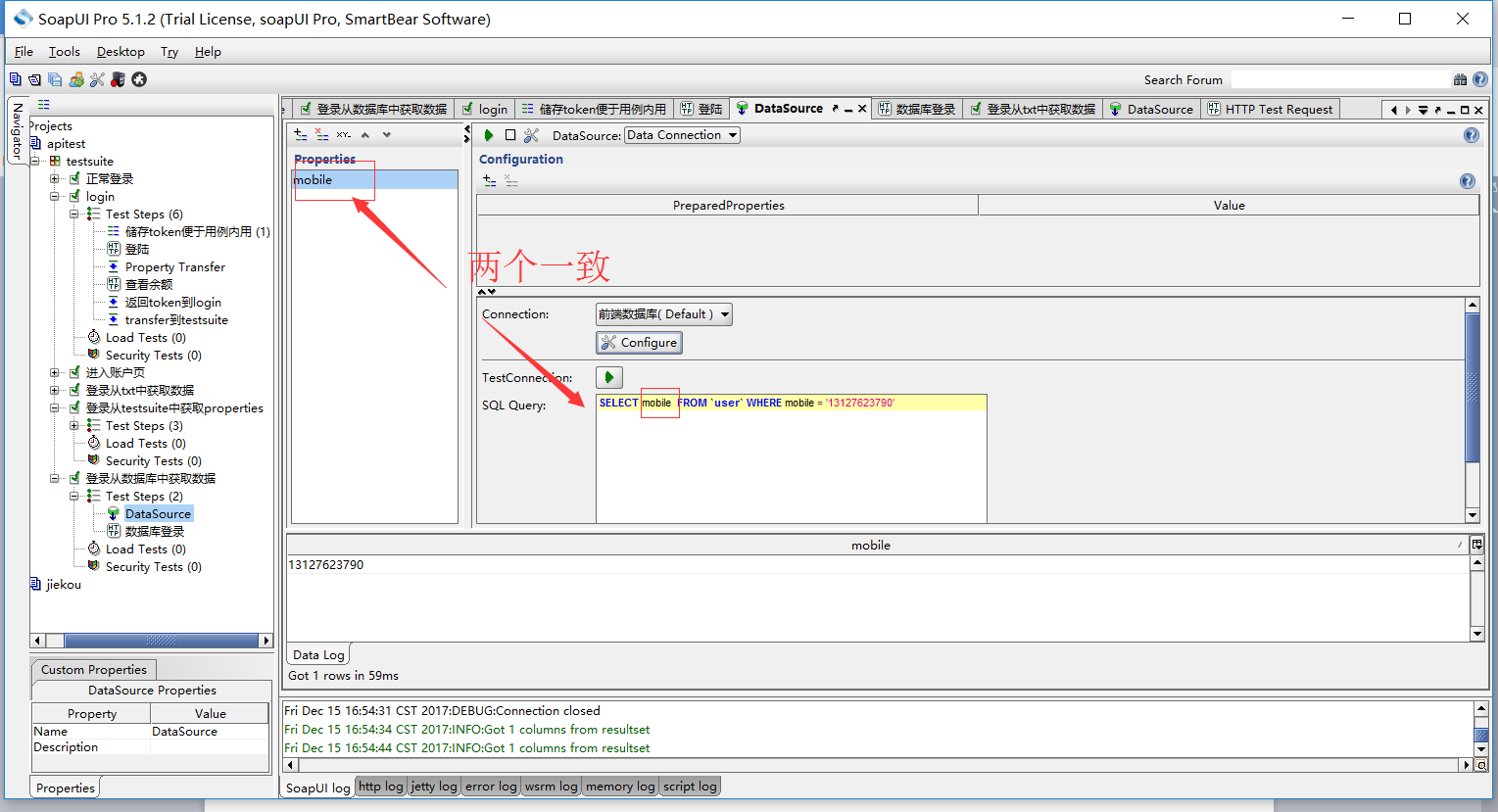
注：跨step间的数据传递需要通过properties来传递

# 从txt中获取数据

1. 新建txt文件，造数据，每一行以逗号隔开，如图
2. 新建一个datasource,选择文件格式是file，设置charset为UTF８，如图
3. 新建http request，赋值value值为datasource中的值，如图
4. 创建datasource loop设置循环

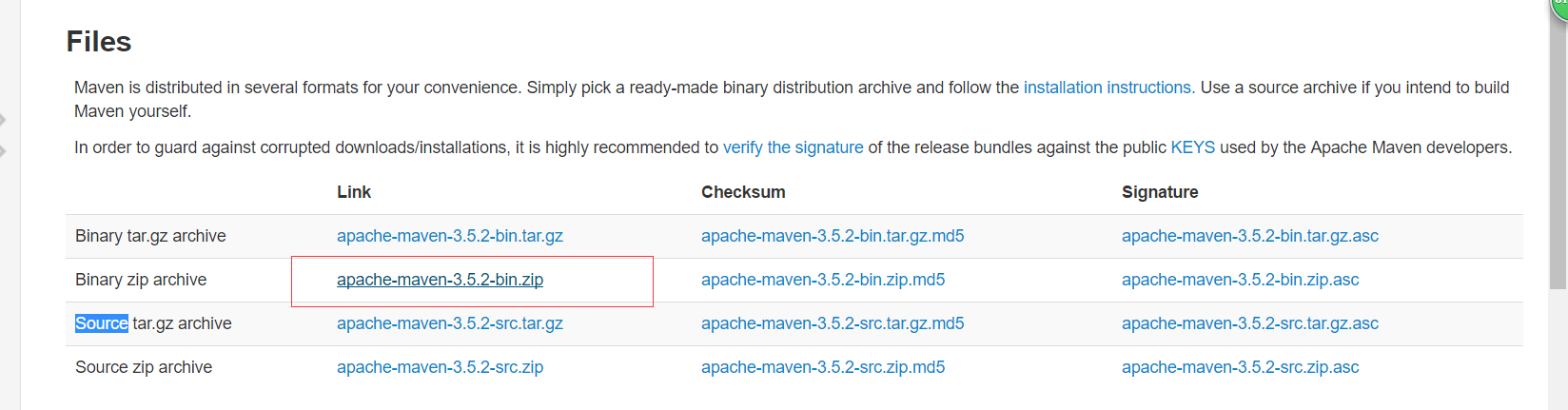
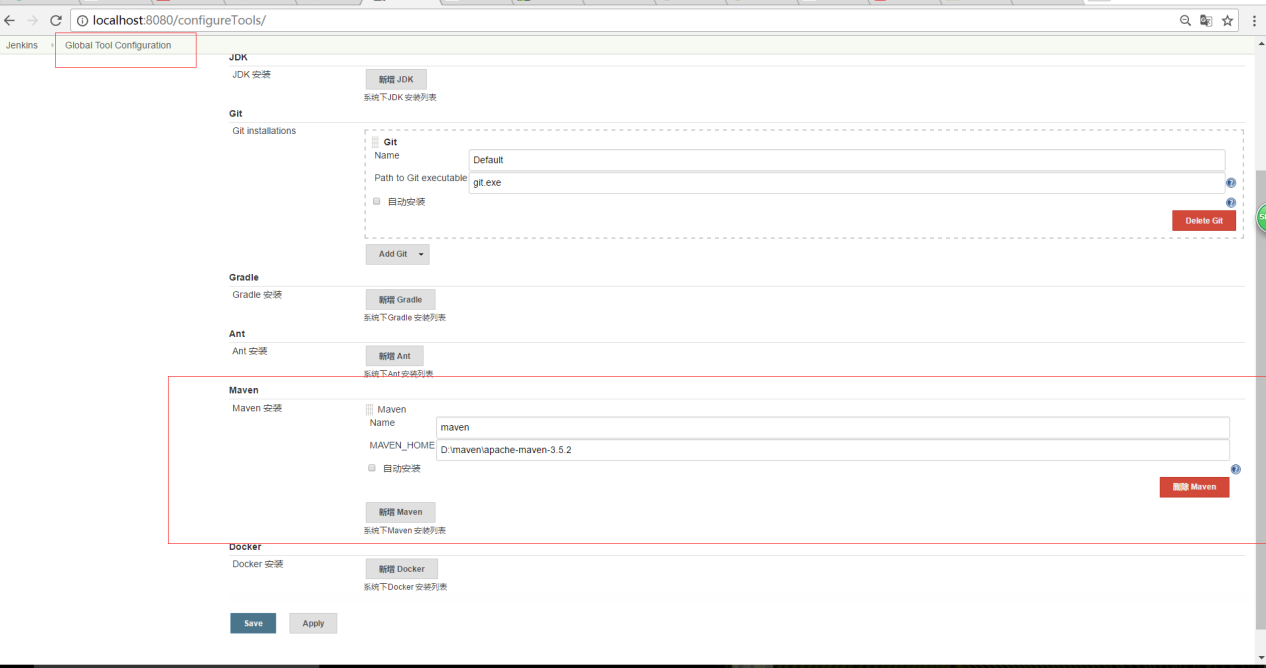
# 从数据库中获取数据

1、先下载个mysql的jdbc,放到soapui下的jre\lib\ext中如：mysql-connector-java-5.1.44-bin

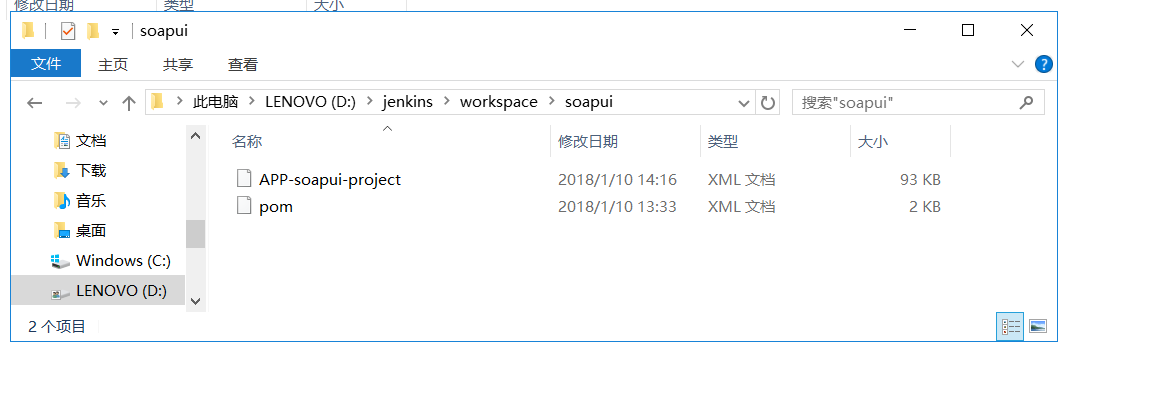
1. 新建一个datasource,选择datasource为data connection
2. 新建一个connection
3. 测试一下通过就行
4. Sql query 中写sql语句，其中properties中的名字必须和sql中获取数据的名称或者别名一样如图

# 新建一个http request，将value值设成sql 中取到的值

# Jenkins+maven+soapui集成接口测试

1. 安装maven，进入[Apache Maven](https://maven.apache.org/download.cgi" \t "http://www.yiibai.com/jenkins/_blank)下载链接二进制的 .zip 文件
2. 解压到指定文件，如D:/maven，然后在jenkins管理中进行如下配置

3、在jenkins插件中安装Maven Integration plugin，后就可以建立一个maven的项目了

项目点击一下构建后就可以创建一个workspace了，将soapui项目文件和pom文件都放在改maven项目下如图

Pom文件的内容如下：

<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/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>lg</groupId>

<artifactId>test</artifactId>

<packaging>jar</packaging>

<version>1.0-SNAPSHOT</version>

<pluginRepositories>

<pluginRepository>

<id>SmartBearPluginRepository</id>

<url>http://www.soapui.org/repository/maven2/</url>

</pluginRepository>

</pluginRepositories>

<build>

<plugins>

<plugin>

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

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

<version>5.1.2</version>

<dependencies>

<dependency>

<groupId>org.reflections</groupId>

<artifactId>reflections</artifactId>

<version>0.9.9-RC1</version>

</dependency>

<dependency>

<groupId>org.apache.poi</groupId>

<artifactId>poi-ooxml</artifactId>

<version>3.10-FINAL</version>

</dependency>

</dependencies>

<executions>

<execution>

<phase>test</phase>

<goals>

<goal>test</goal>

</goals>

<configuration>

<projectFile>test-soapui-project.xml</projectFile>

<projectProperties>

<value>message=Hello World!</value>

</projectProperties>

<junitReport>true</junitReport>

<exportAll>true</exportAll>

<outputFolder>target/surefire-reports</outputFolder>

<printReport>true</printReport>

</configuration>

</execution>

</executions>

</plugin>

</plugins>

</build>

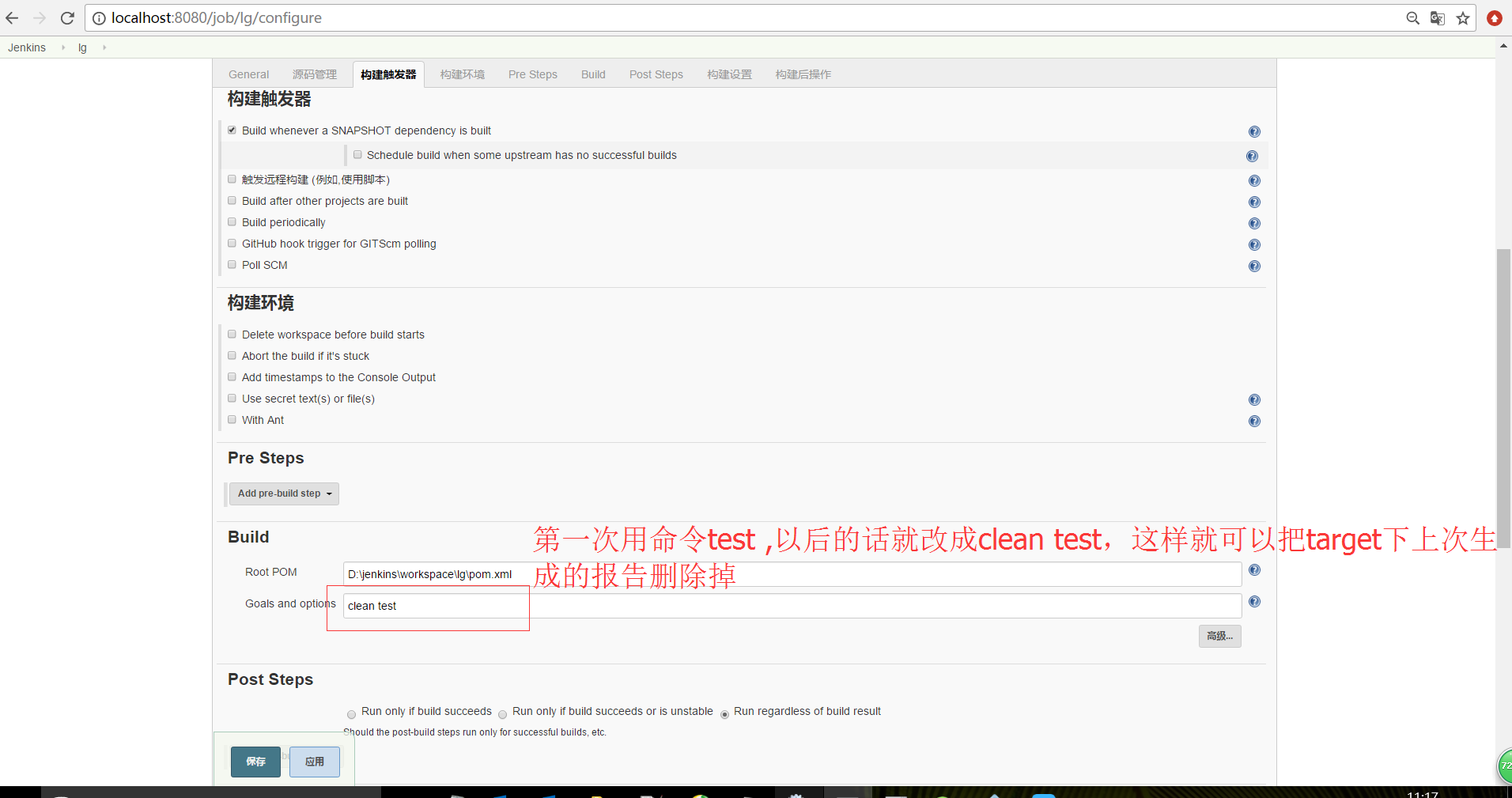
</project>

在D:\jenkins\workspace\soapui下创建src/main/resources放置soapui-settings.xml 文件

在D:\jenkins\workspace\soapui下创建src/main/resources放置default-soapui-workspace.xml

在D:\jenkins\workspace\soapui下创建target文件夹

4、然后配置该maven项目



5、下载soapui-pro-maven-plugin-5.1.2.jar 这是目前最新版本：

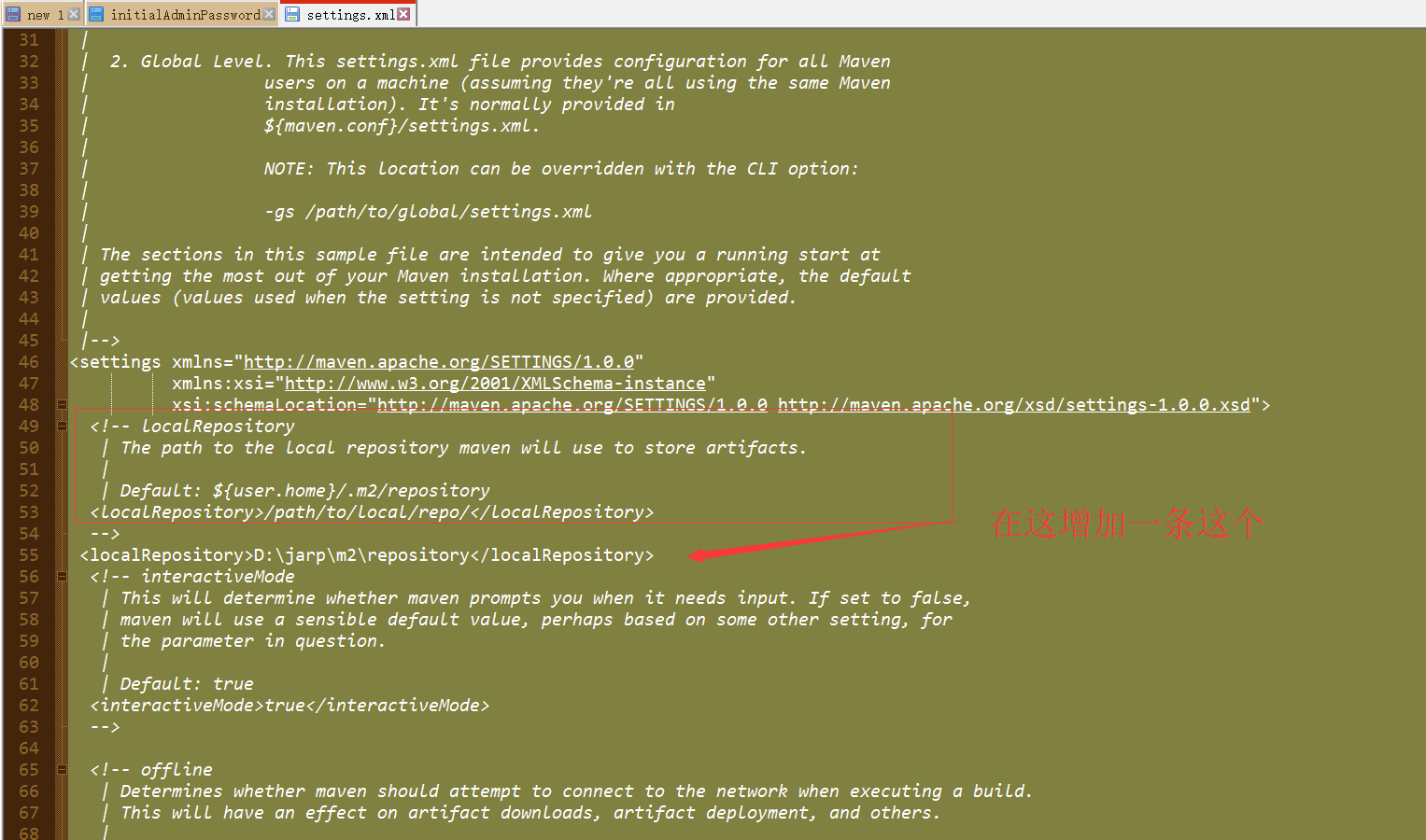
[http://smartbearsoftware.com/repository/maven2/com/smartbear/soapui/soapui-pro-maven-plugin/](http://smartbearsoftware.com/repository/maven2/com/smartbear/soapui/soapui-pro-maven-plugin/" \t "http://appletest.lofter.com/post/_blank)



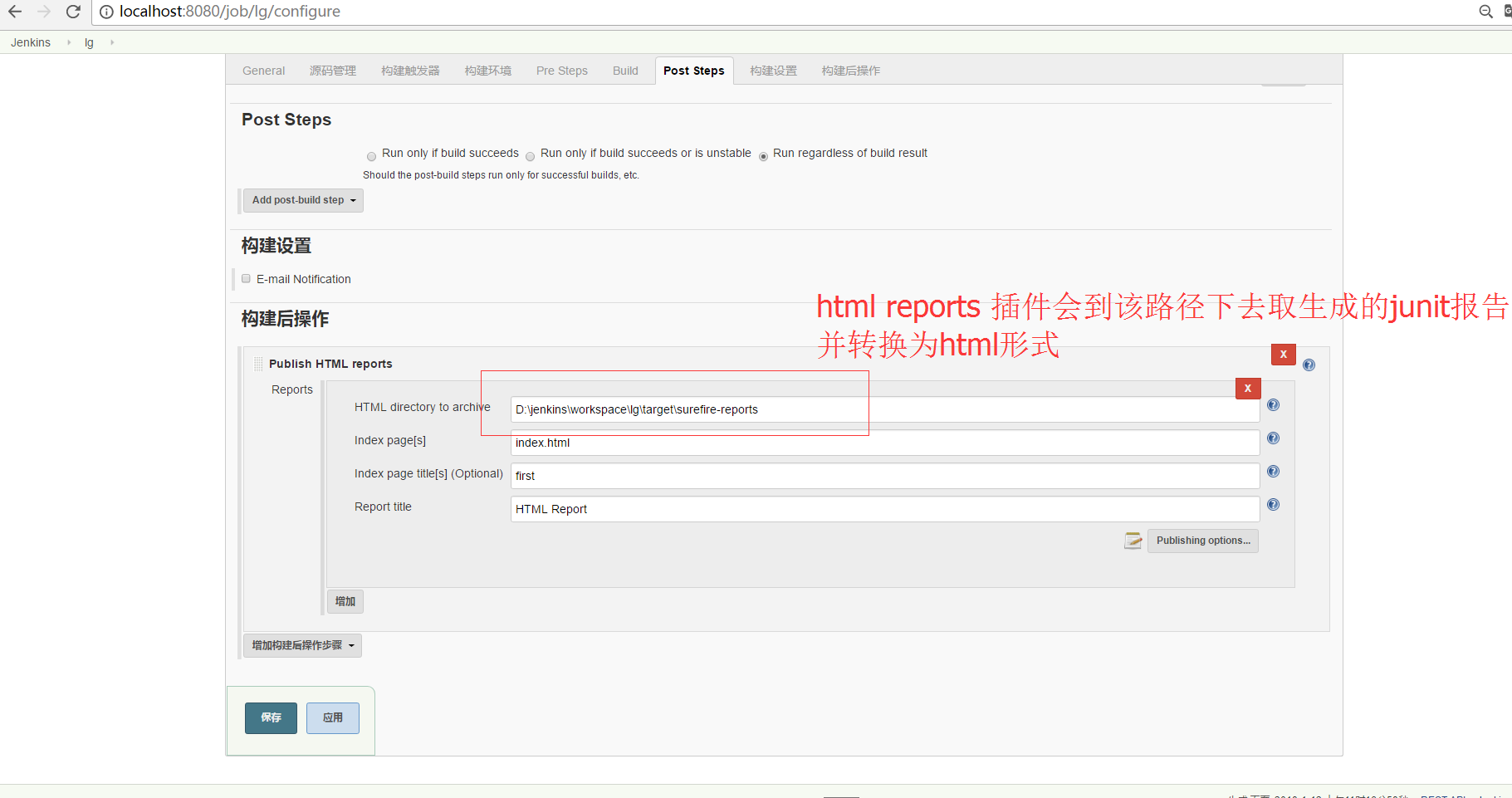
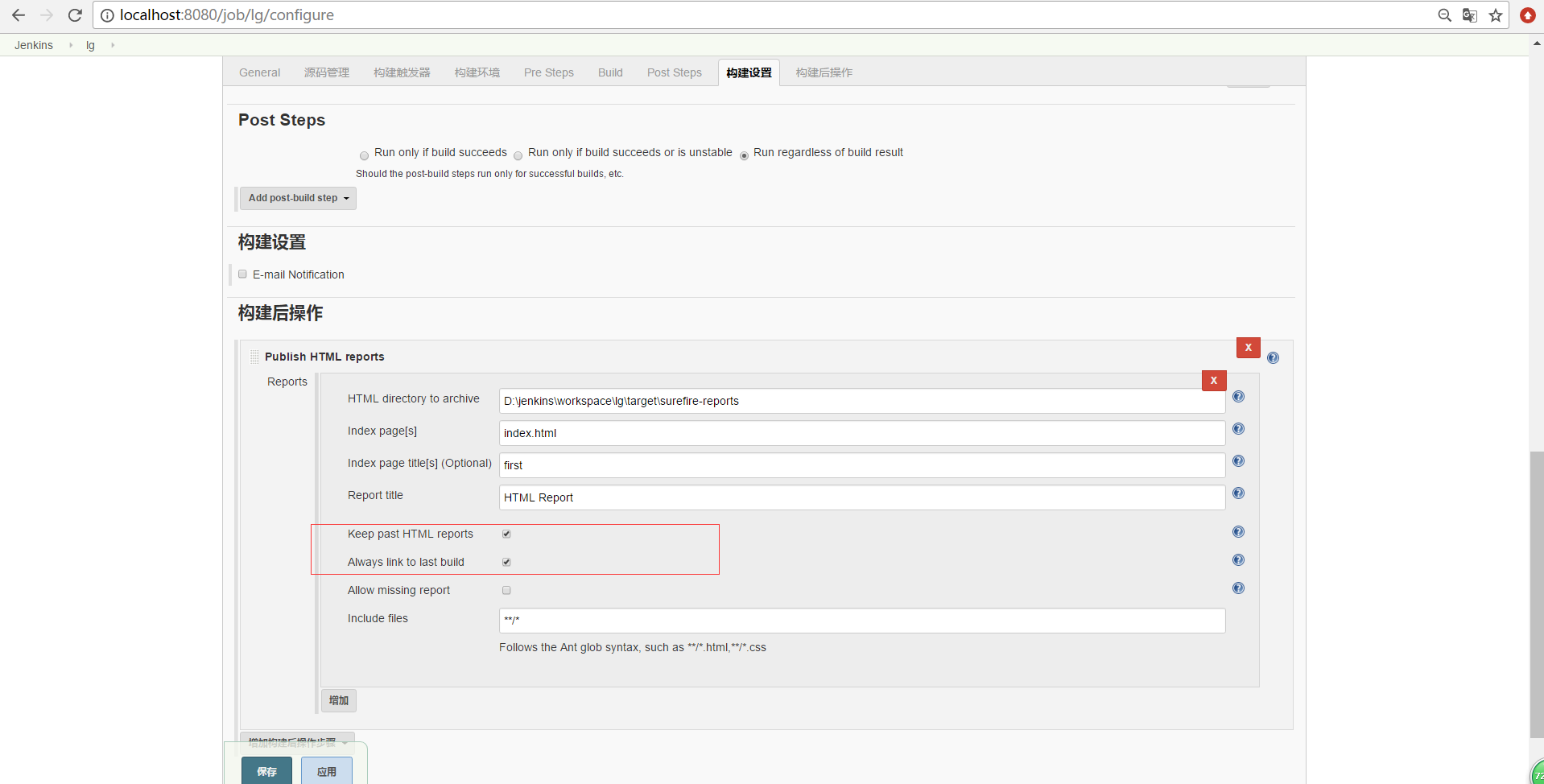
仅下载jar包

6、用命令安装插件包：mvn install:install-file -Dfile=D:/soapui-pro-maven-plugin-5.1.2.jar -DgroupId=com.smartbear.soapui -DartifactId=soapui-pro-maven-plugin-5.1.2.jar -Dpackaging=jar -Dversion=5.1.2

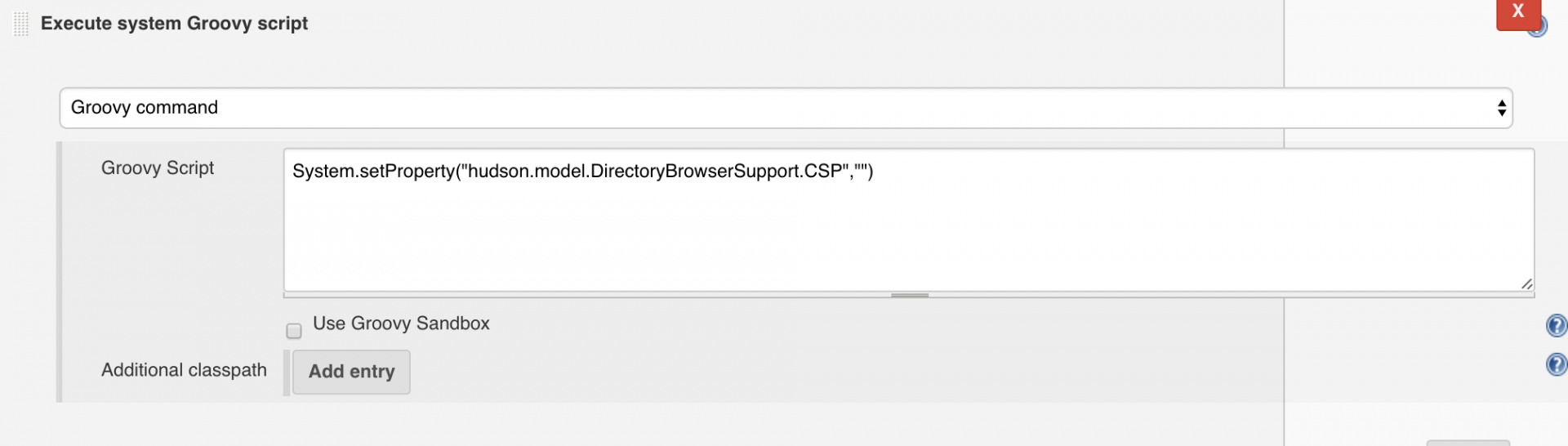
mvn install:install-file -Dfile=D:/soapui-pro-maven-plugin-5.1.2-m-SNAPSHOT.jar -DgroupId=com.smartbear.soapui -DartifactId=soapui-pro-maven-plugin-5.1.2-m-SNAPSHOT.jar -Dpackaging=jar -Dversion=5.1.2-m

上面的jar包默认是安装到C:\Users\shiht\.m2\文件下的（此时可能没有权限写入，只需给该文件夹赋予写的权限就可以了），如果要改变安装路径的话，可以先创建如D:/jarp/m2/repository仓库文件，并将maven/conf下的settings.xml文件进行修改修改如下

然后复制修改后的settings.xml 文件到D:/jarp/m2/下。改完路径后再安装上述的jar。如果要重新装上面的jar包时可以直接将D:/jarp/m2/repository下的文件全部删除就好。

1. 下载HTML Publisher plugin，并在maven项目下进行设置
2. 点击构建（第一次会构建成功，再次点击构建时会报test reports文档已经存在，将报告文件全部删除后又可构建成功，解决方法是将之前的mvn test命令改成mvn clean test ,该命令会先将target文件下的编译文档删除然后再执行test命令）也可以在命令行中进入到D:\jenkins\workspace\lg后执行命令mvn com.smartbear.soapui:soapui-pro-maven-plugin:5.1.2:test
3. 设置保留之前的html reports，并且每次链接到最新的一次报告文件，如下图
4. 查看html reports时可能会出现报告显示不全的问题，解决办法如下
5. 在jenkins系统管理中输入以下脚本运行（jenkins重启时会失效）  
   System.setProperty("hudson.model.DirectoryBrowserSupport.CSP", "")

B、下面就介绍下永久解决的方法：  
jenkins上安装 Groovy 插件，用来执行系统 Groovy 脚本，安装完成后，在增加构建步骤里添加：  
Execute system Groovy scripts

[](https://testerhome.com/uploads/photo/2017/fcf46851-6647-4a2b-82d6-17bf8a1d2fd9.png!large)

执行脚本：System.setProperty("hudson.model.DirectoryBrowserSupport.CSP", "")

# Rest project接口测试

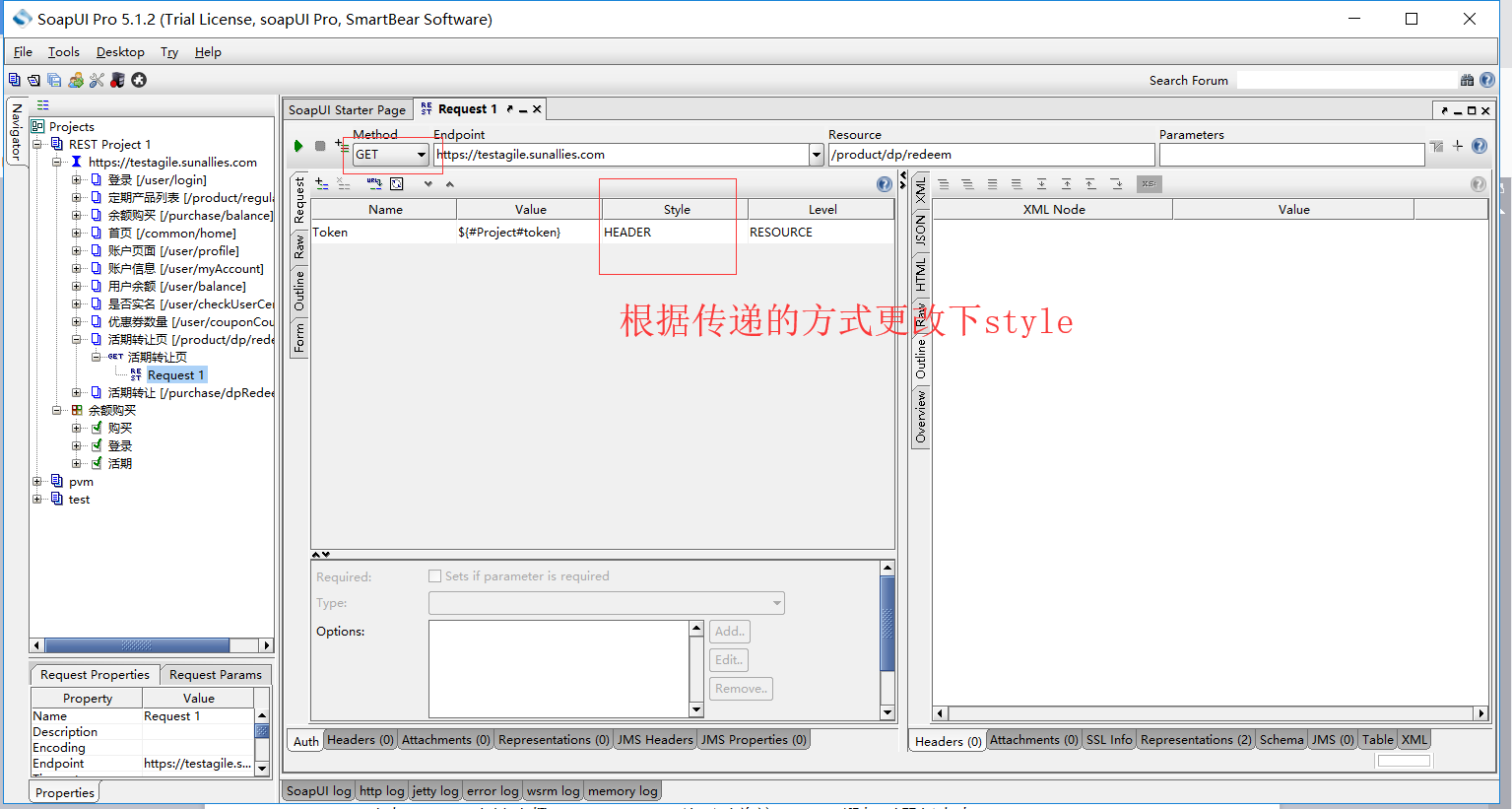
新建一个 Rest project

然后创建new resources

用rename重命名

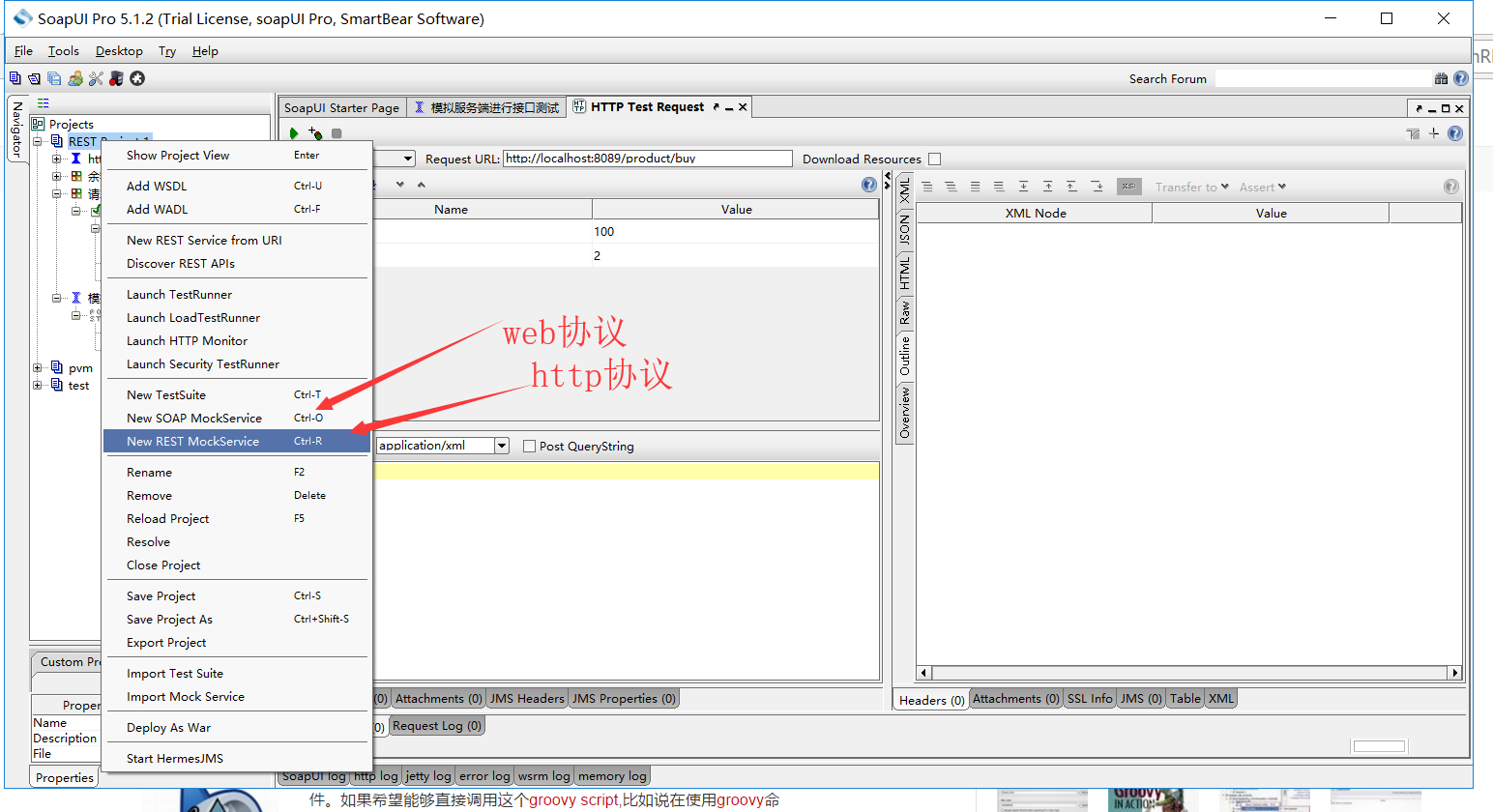
传递头部信息的时候将request中的style改成header

选中request右键选择add to testcase 就可以将该request添加到用例中去

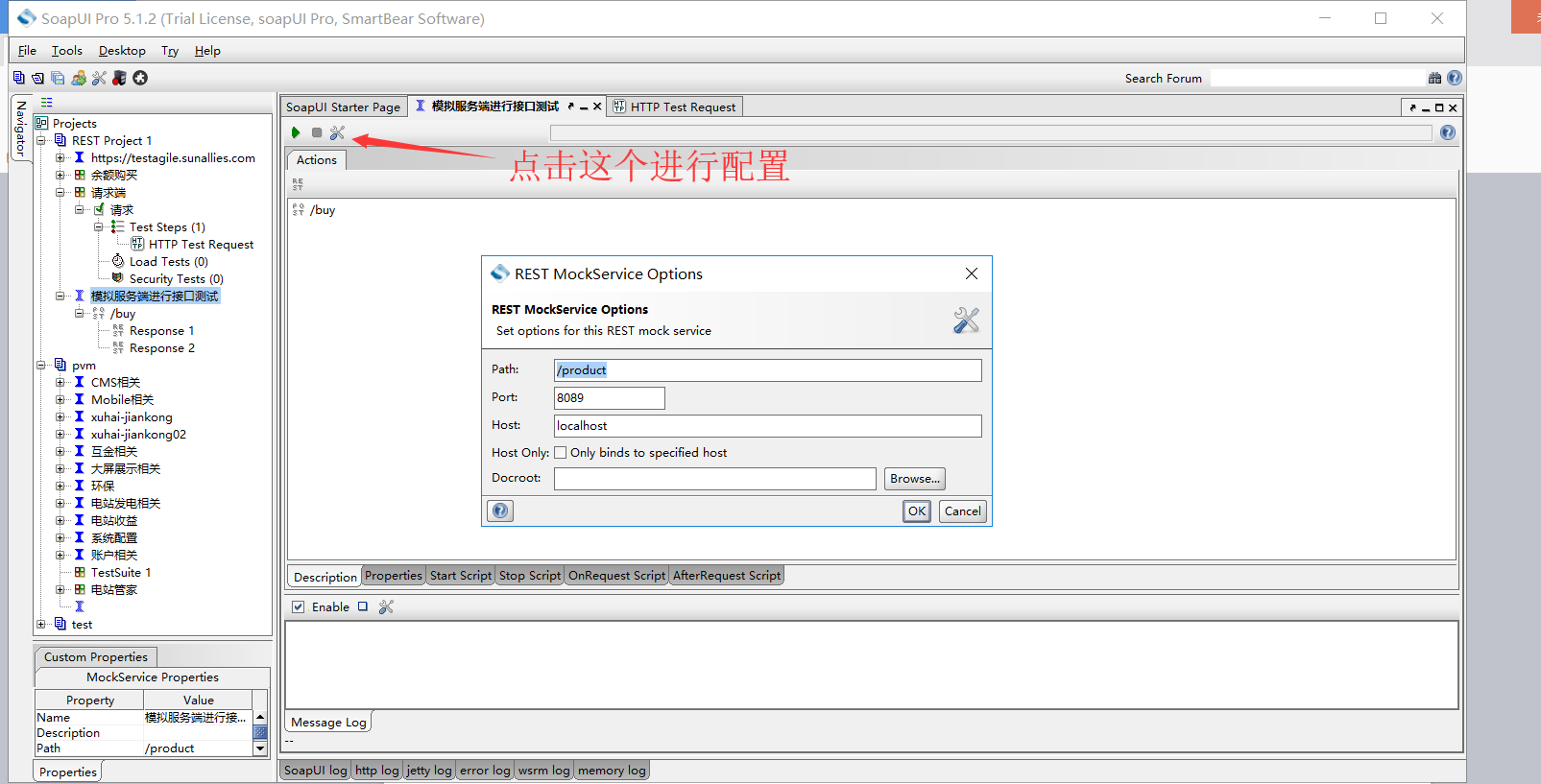


# Soapui模拟服务端进行接口测试

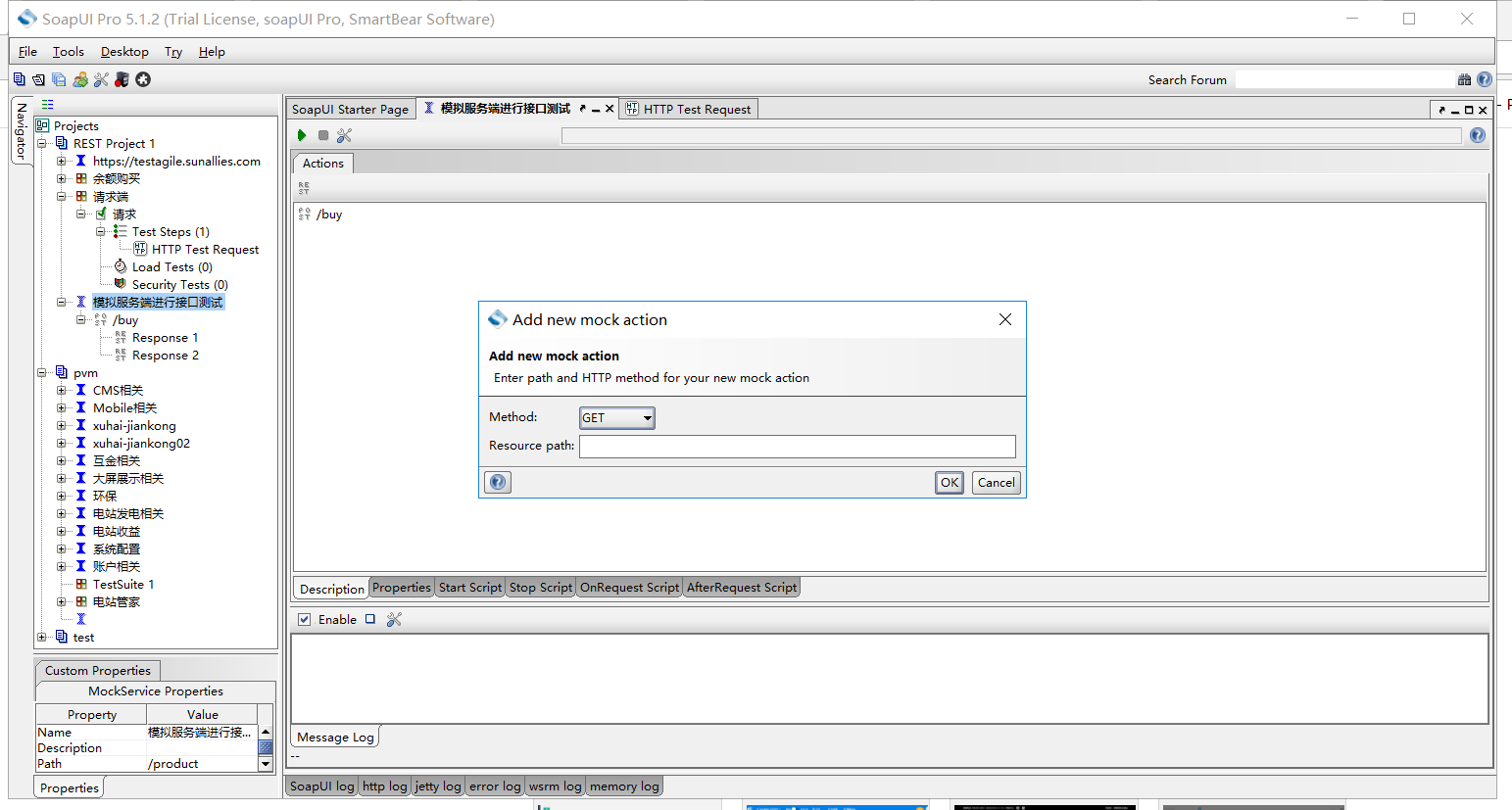
1. 新建一个Rest Mockservices

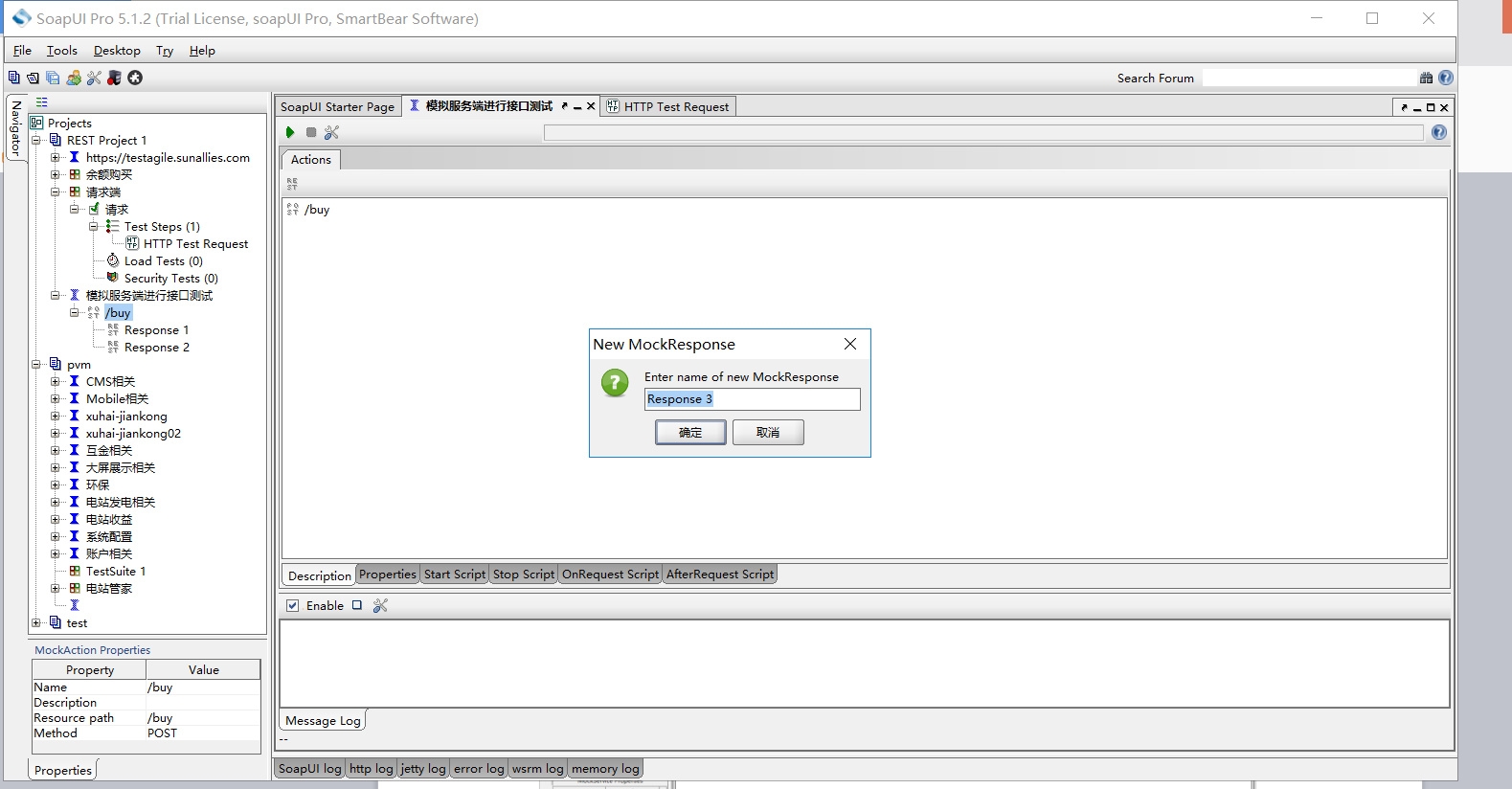


2、点击新建的mock service的配置，进行配置

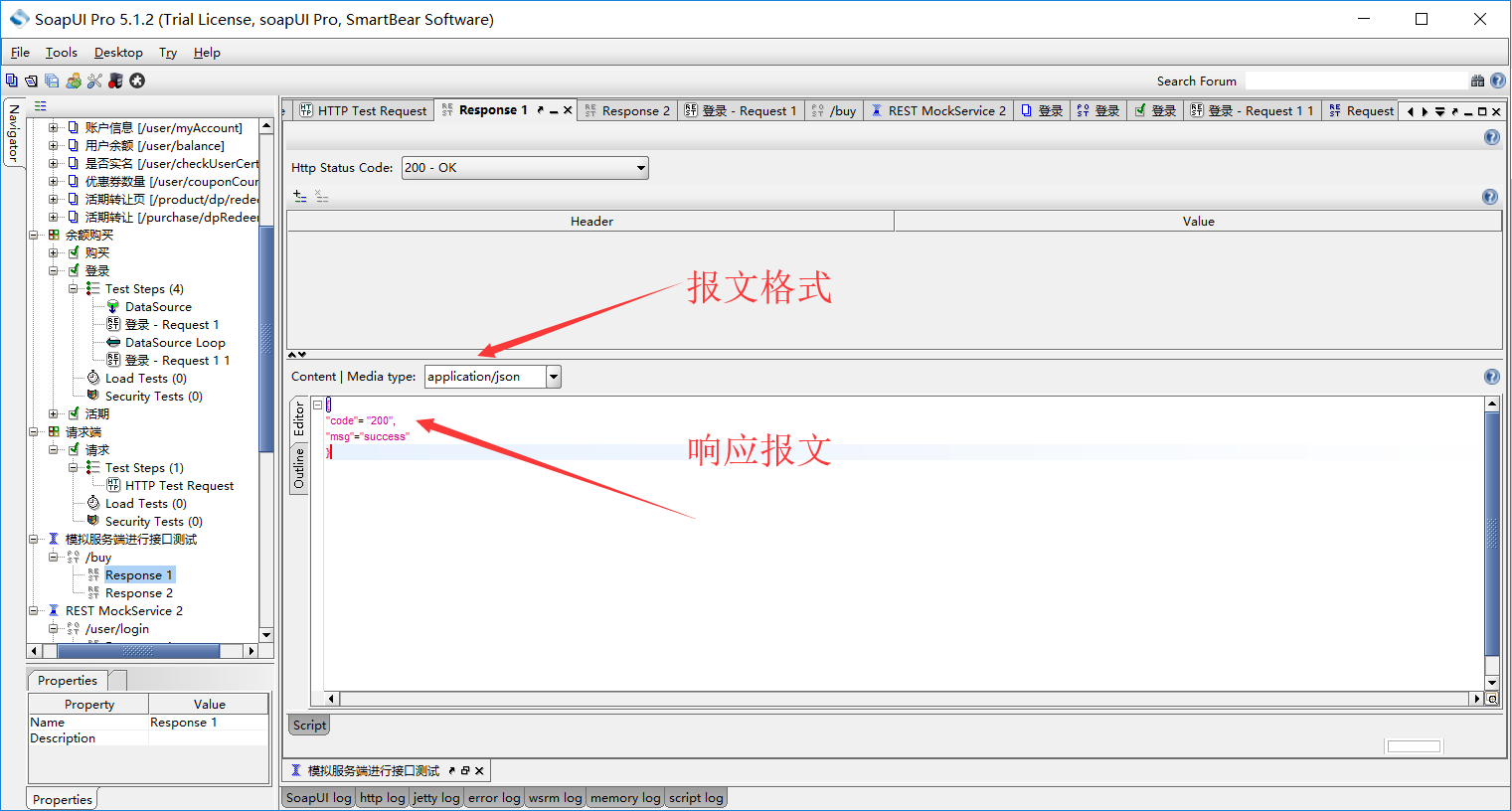


3、新建一个mock action，配置方式和相对路径

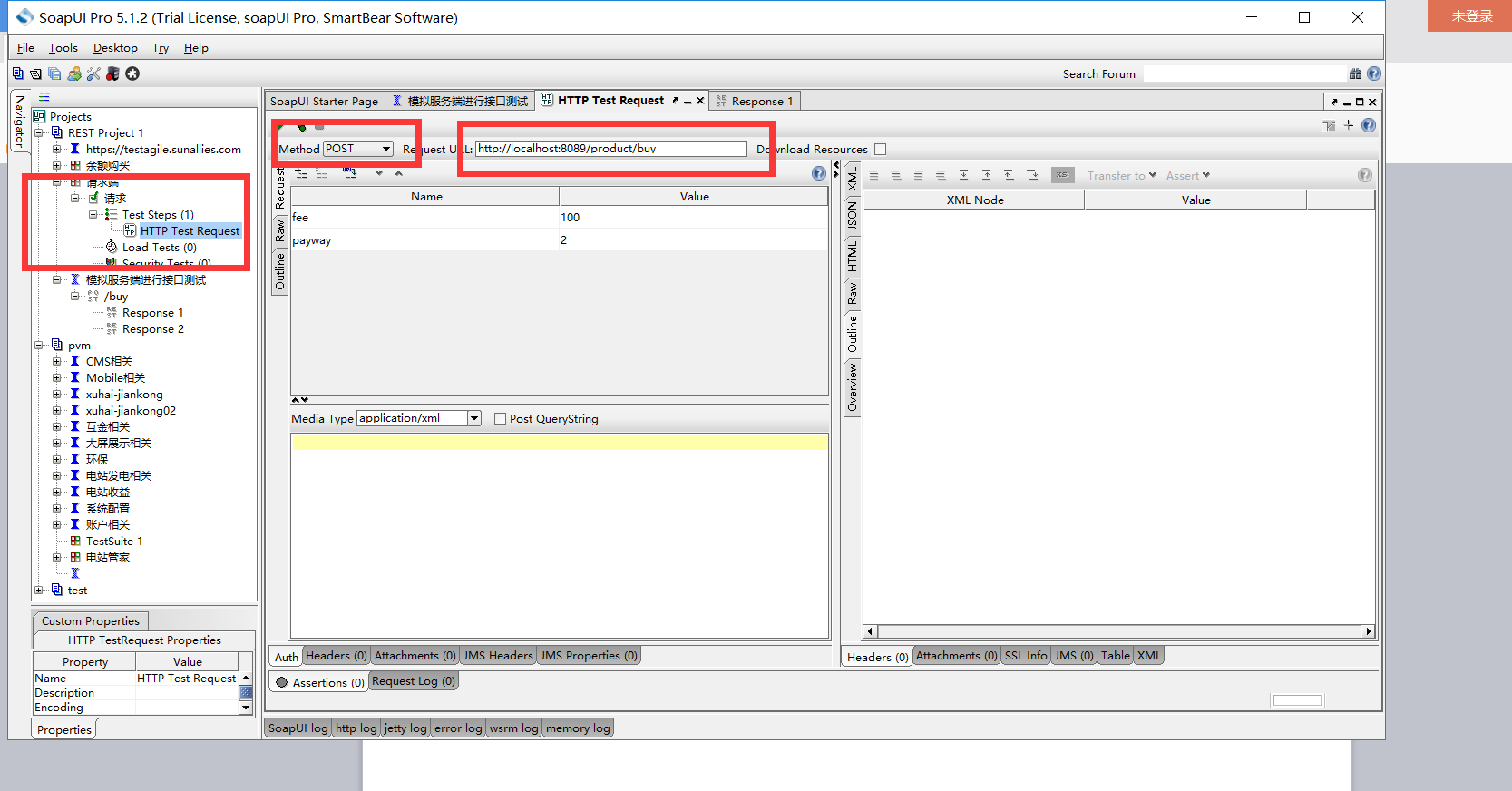
4、给新建的action建一个response



1. 进入到response中进行配置

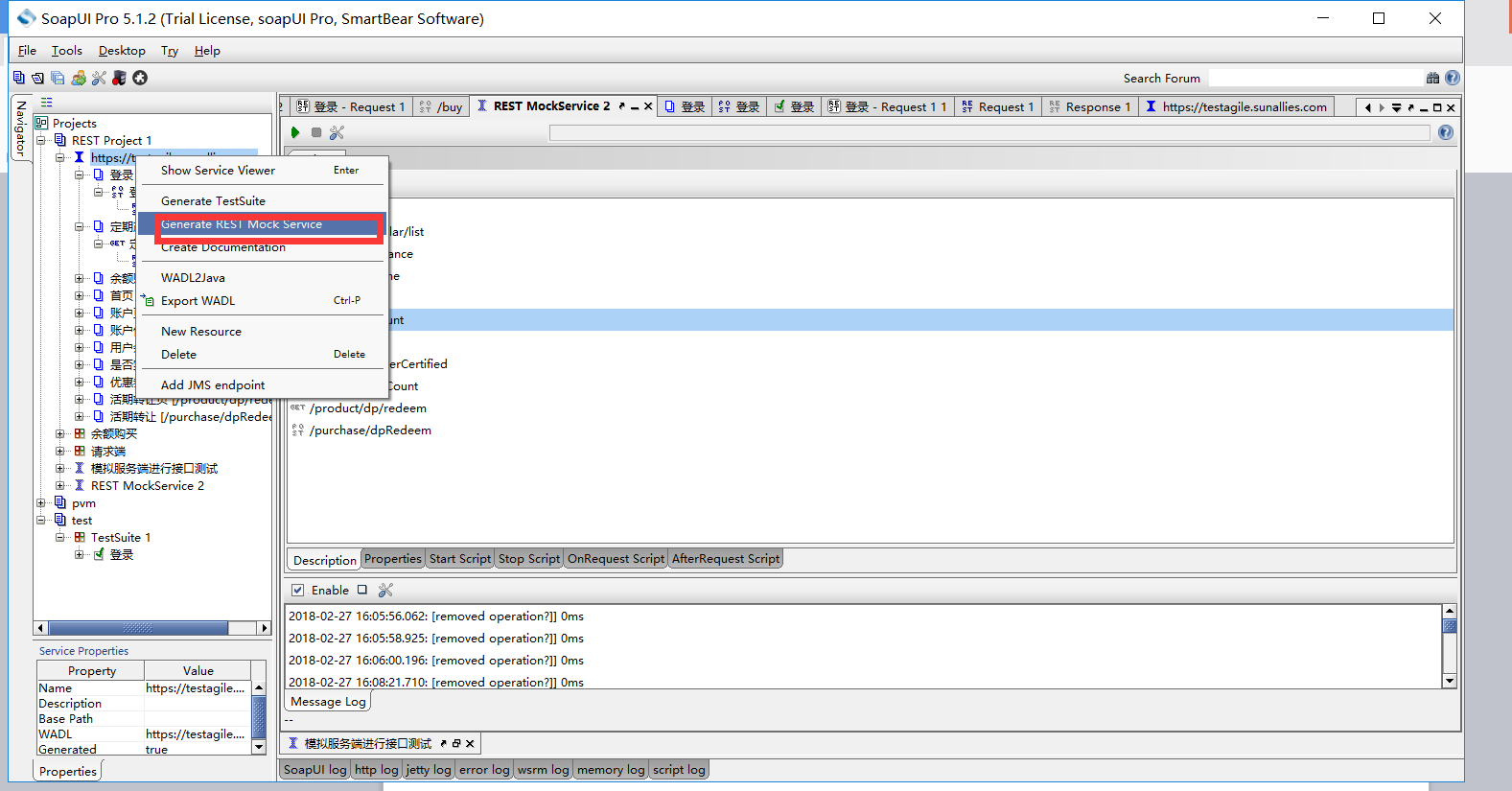


1. 将mock service启动
2. 新建一个request，请求地址和方式跟mock service保持一致

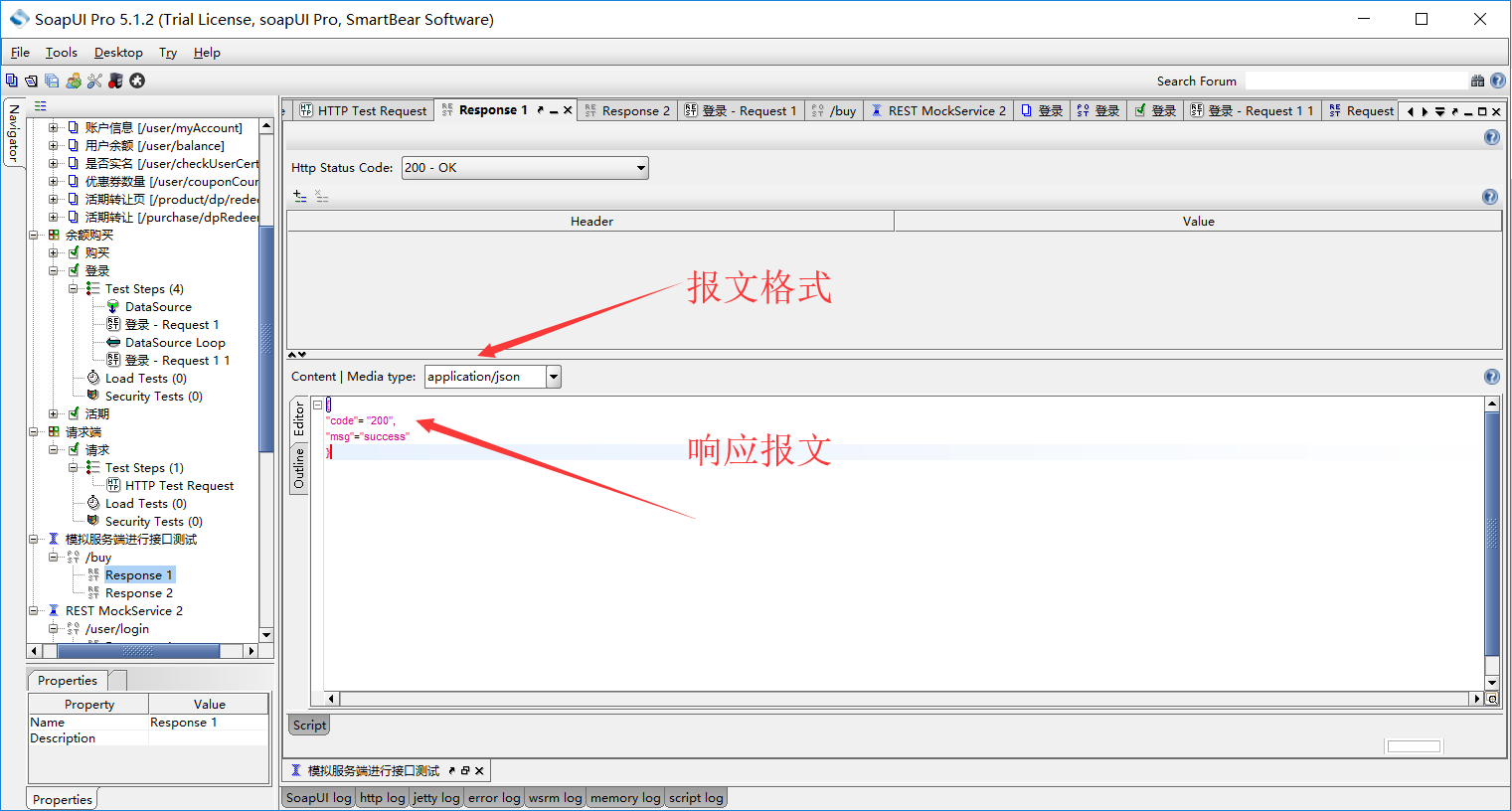


# Soapui模拟服务端进行接口测试方法2

1. 右键Rest Project直接生成mockservice



1. 进入服务端配置报文格式和响应报文



（一般情况下前端只是需要一个正确的结果，以便做页面，异常情况下的接口返回需要做二次开发）