# Robot FrameWork介绍

1、介绍

2、安装pip install robotframework

3、好处

1、python编写的自动化测试框架

2、自持关键字驱动和数据驱动模式

3、内置库丰富，可以直接调用

4、扩展库，AppiumLibrary HTTP library (Requests) Selenium2Library

5、很多工具有rf插件

4、命令行执行 pybot xxxx

输出3个文件

report.html 测试报告

log.html 日志文件

output.xml 详细的输出信息

# Robot FrameWork基本语法

## 文件结构

5、用例格式

项目—套件—用例—关键字

内部基本模块

\*\*\* Settings \*\*\* 与配置相关

Documentation 测试的文档说明

Library 测试导入的库

Suite Setup 执行测试套件前的关键字

Suite Teardown 执行测试套件后的关键字

Test Setup 执行测试用例前的关键字

Test Teardown 执行测试用例后的关键字

Test Template 测试模板--数据驱动模式

针对相同的工作流，执行不同的输入数据

使用一个高级的用户关键字，定义工作流，然后使用不同的输入和输出数据

测试相同的场景每个测试中重复使用同一个关键字

Resourse 导入的外部资源文件---主要是关键字

\*\*\* Variables \*\*\* 测试套件中的公共变量

${变量名} 变量值

\*\*\* Test Case \*\*\*

用例名

[Documentation]

[Tags] 标签

[Setup]

[Teardown]

关键字 参数

## 变量

地址<https://blog.csdn.net/huapingqi/article/details/81140996>

或者百度搜索RobotFramework框架系列

## 基础库

流程控制语句

循环语句

## 扩展库

Http Requests

selenium2Library

appium

## 开发关键字

利用Robot Framework编写测试用例，往往需要开发自己的关键字，有的关键字需要通过自己编写python代码来实现。这在rf中，就需要自己定义python库。这个过程其实不复杂，本文来介绍下。

1、在python安装目录下的 Lib\site-packages 目录下 新建一个目录，目录名就是库名，如 NewLibrary

2、在新建的NewLibrary目录下创建一个python文件，任何合法的文件名即可，扩展名为py。如：myclass.py

其内容如下

[IMG_256](https://www.cnblogs.com/51kata/p/javascript:void(0);)

class MyClass(object):

def \_\_init\_\_(self):

pass

def printMsg(self,msg):

print "hello "+msg

[IMG_257](https://www.cnblogs.com/51kata/p/javascript:void(0);)

上面代码定义了一个python类，其中除构造方法 \_init\_ 外，还定义了一个printMsg方法，该方法其实就是rf中的关键字，可以直接在rf中使用。下面我们介绍。

3、在新建的NewLibrary目录下创建 \_\_init\_\_.py 文件，文件名必须固定这样。该文件的功能，就是把关键字类暴露出来，让rf能知道。  
其内容如下：

from myclass import MyClass

class NewLibrary(MyClass):

ROBOT\_LIBRARY\_SCOPE = 'GLOBAL'

4、通过前面三个步骤，一个最简单的自定义的NewLibrary库就创建好了，在rf工具中就可以使用了  
1）先要导入 NewLibrary 库

2）使用关键字printMsg。需要说明的是，在rf中引用时，既可以写成 printMsg，也可写成 print msg。一般习惯用空格把单词分开。因为rf关键字不区分大小写。

\*\*\* Test Cases \*\*\*

successsd

print msg god

printMsg ss

printmsg yy

5、增加新的关键字  
增加新的关键字很简单，只需在MyClass类中增加新的方法即可。如：

[IMG_258](https://www.cnblogs.com/51kata/p/javascript:void(0);)

class MyClass(object):

def \_\_init\_\_(self):

pass

def printMsg(self,msg):

print "hello "+msg

def join\_info(self,a,b):

print a+b

[IMG_259](https://www.cnblogs.com/51kata/p/javascript:void(0);)

这里定义了 join\_info方法。在rf中使用时，可以写成 join  info 用空格分开。

**上面介绍了最简单的自定义模块，虽然简单，但实际可以满足大部分自己开发的需求了。**

# 接口测试用例设计

Level1：最能反应该接口功能的测试用例

Level2：参数的正常校验和异常校验

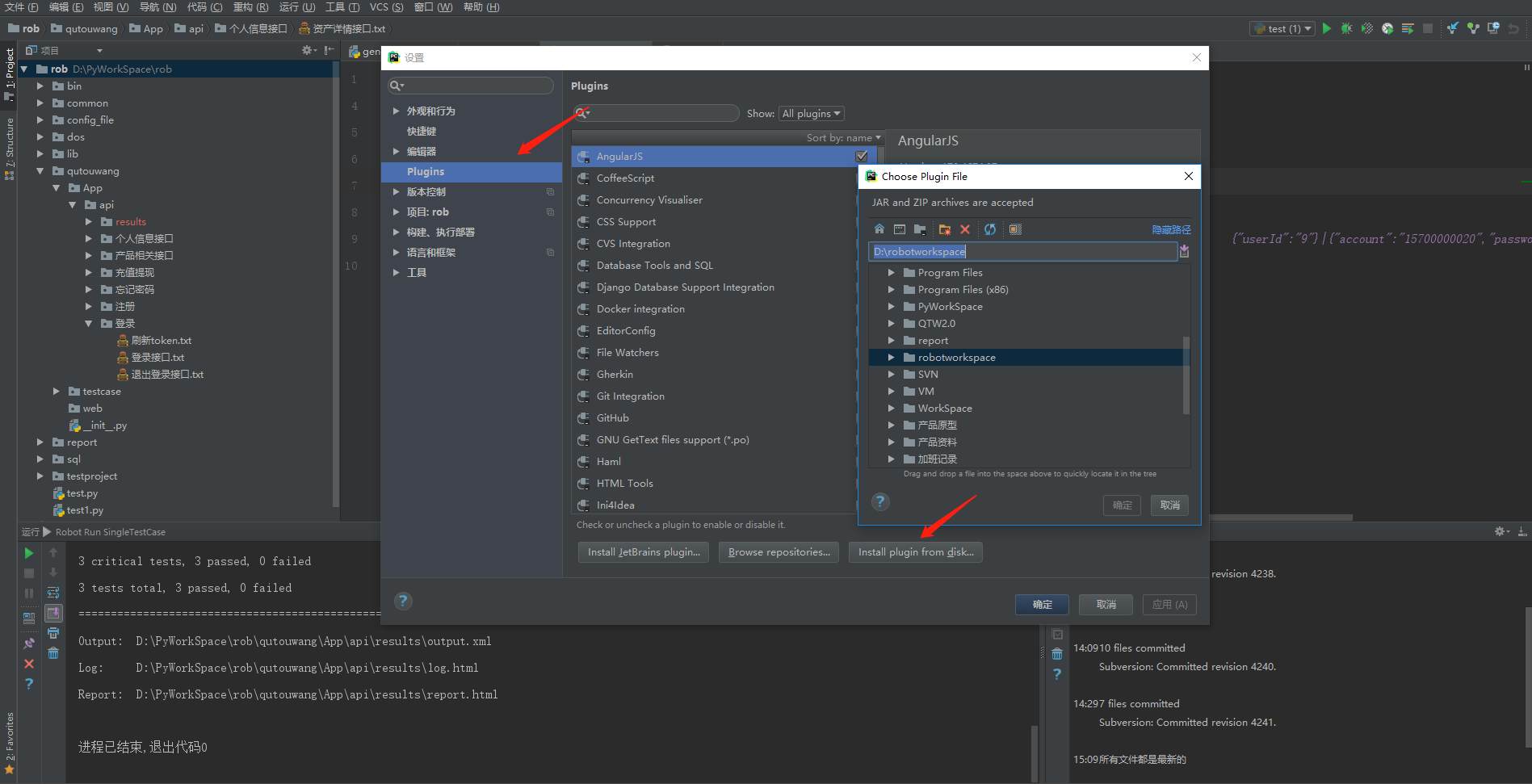
Level3：字段是否必传



# Pycharm安装Robot FrameWork插件

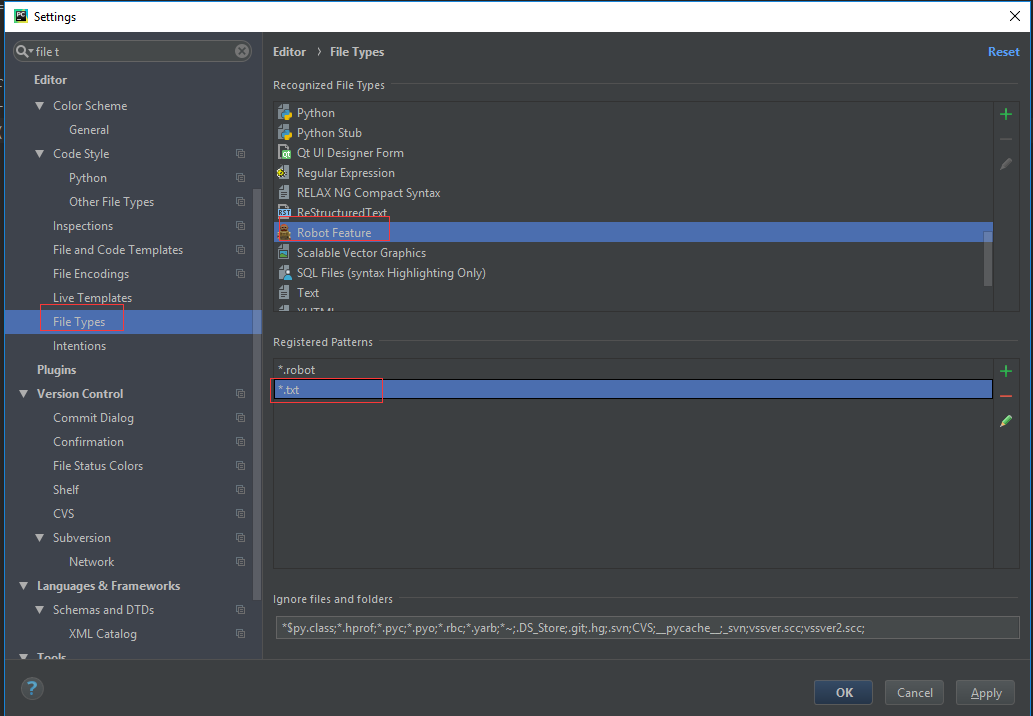
1、下载IntelliBot

2、打开PyCharm，单击 左上角file→settings →Plugins → 【install plugin from dink】



3、配置

点击File>点击setting>点击editor>点击file types,点击右侧的+号配置一下文件类型即可

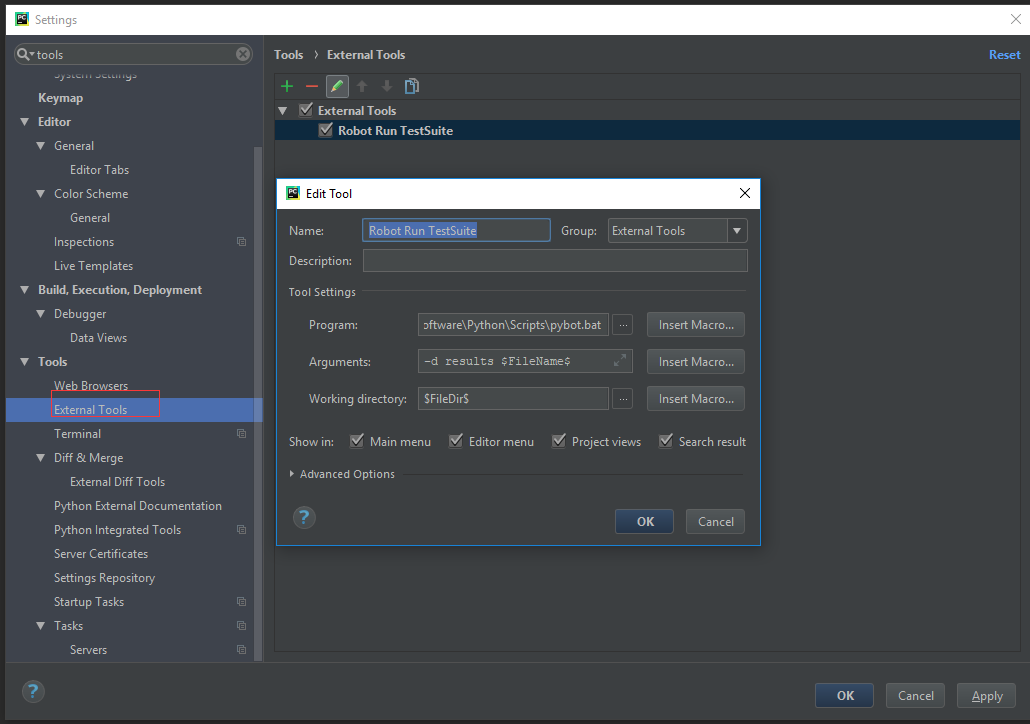


4. 至此已经可以在pycharm中编辑脚本了，但是还不能进行脚本的执行，接下来配置如何执行测试用例  
 点击File>点击setting>点击Tools>点击External tools，配置执行测试套件，可以执行一个测试套件内的所有case。

Robot Run TestSuite

-d ../../../../report $FileName$

$FileDir$

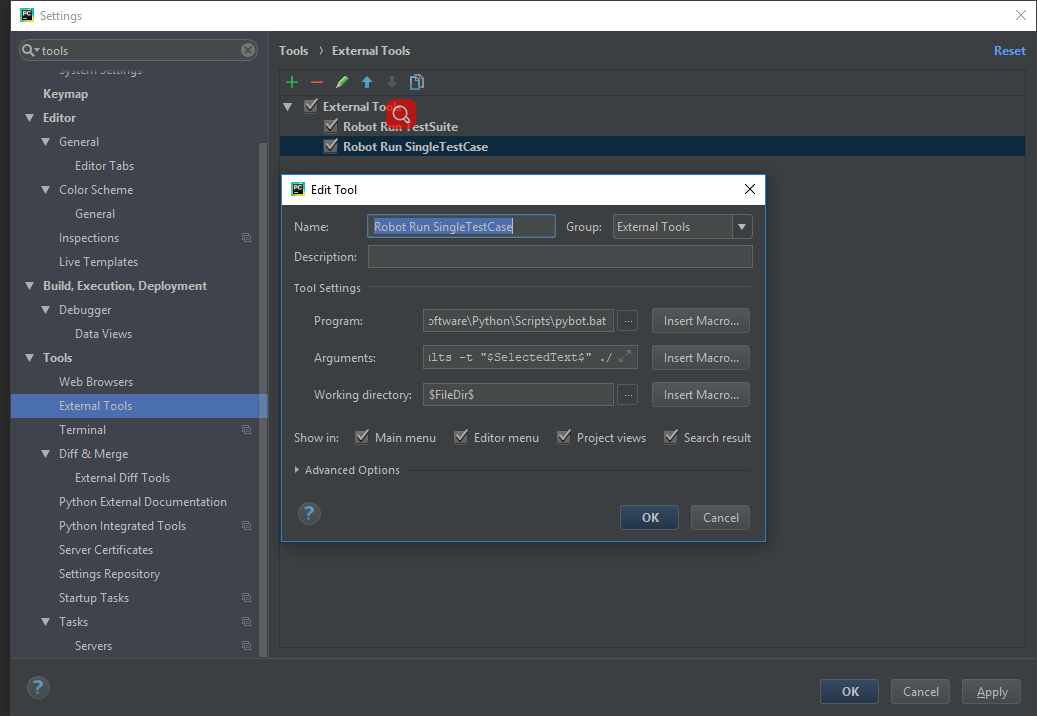


5. 添加执行单个用例的配置

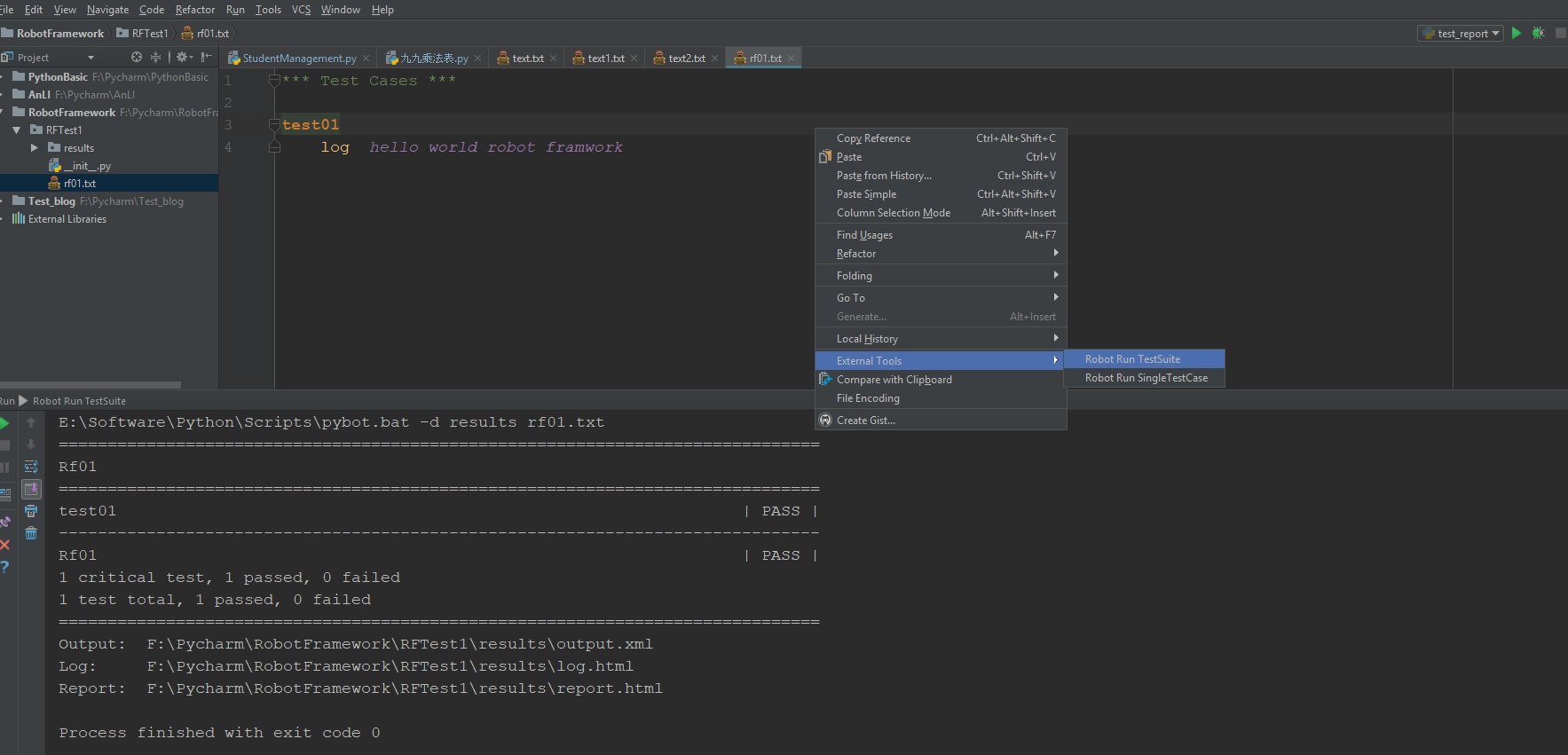
Robot Run SingleTestCase

-d ../../../../report -t \*$SelectedText$\* ./ $FileName$

$FileDir$



6. 执行实例



# rob框架介绍

## 辅助工具生成测试用例

使用bin目录下面generatecase.py文件中的Generatecase类将excel中设计的case生成到.txt

文件中。

注意：excel文件的名称反应的是生成的txt文件所在的路径

## 环境切换

目前编写的rob\_frame框架需要兼容测试环境和自动化环境，自动化环境的脚本执行每日定时构建，而测试环境的脚本在代码构建完成之后执行，达到一个冒烟测试的效果。

1、使用config\_file目录下的env.ini和Data.txt文件来控制环境切换

2、在使用python rob.py –e test/auto XXXXX执行时，会根据传入的test/auto值，重置env.ini和Data.txt文件，来达到控制环境切换的目的。

## 用户关键字lib

1、将lib文件放到PYTHONPATH中

查看PYTHONPATH  
import sys  
sys.path  
增加.pth文件，推荐！  
在site-packages添加一个路径文件，如mypkpath.pth，必须以.pth为后缀，写上你要加入的模块文件所在的目录名称就是了。  
2、lib文件中包含了目前所有的关键字

## 扩展断言

对于一些接口返回信息的检查比较复杂，光靠是否有返回信息已经不足以判断，例如交易流水，需要检查倒叙排序，还有是否为同一类型

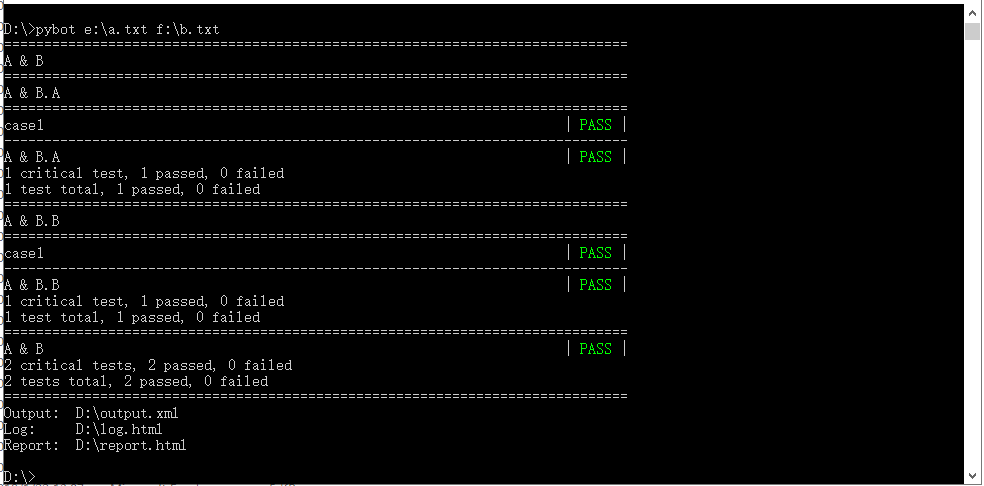
在HttpApiLib.py这个文件中，编写关键字对self.r.text进行处理

## 执行报错的脚本

具体实现是，将报告中的xml中的信息读取出来，获取是那些txt文件报错

然后可以按照下面的方法执行。。。

在用例执行完成之后，可以读取xml文件从中筛选出报错的txt文件，然后按照途中的方法，执行所有的报错的txt文件，来做到将错误的txt文件重新执行一遍而不需要全部执行



# 配置Jenkins

## linux下安装部署Jenkins

1. 安装JDK
2. 安装Tomcat

解压tar -zxvf apache-tomcat-8.0.53.tar.gz

放在usr/local/tomcat下所以要创建tomcat文件夹并将解压文件移动到tomcat下

mkdir /usr/local/tomcat

mv apache-tomcat-8.0.53 /usr/local/tomcat/

进入tomcat安装bin目录并启动

cd /usr/local/tomcat/apache-tomcat-8.0.53/bin/

启动 ./startup.sh

关闭 ./shutdown.sh

成功启动后浏览器输入<http://192.168.1.31:8090/>

修改端口

查看tomcat是否启动

ps -ef |grep tomcat

假设tomcat所在目录为/usr/local/apache-tomcat/

1、打开tomcat配置文件  
#vi /usr/local/apache-tomcat/conf/server.xml

2、打开Tomcat配置文件之后按 /8080  快速检索

3、修改端口，按 i 进行，将8080 修改为 80

4、按:wq 保存并退出

5、 重启tomcat

3、安装Jenkins

首次安装启动Jenkins，

jenkins初次登陆卡在SetupWizard界面

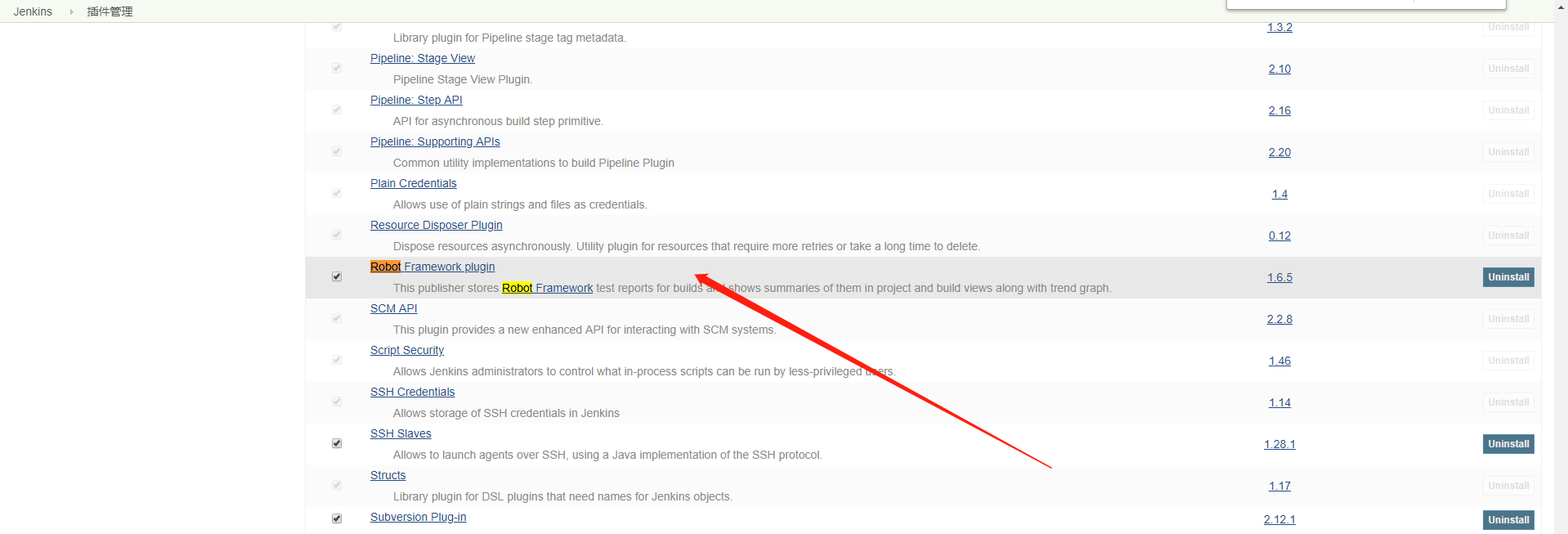
赋777权限给jenkins,并用谷歌浏览器打开，（本人亲测ok）

admin

123456

## 安装Jenkins的robot framework插件

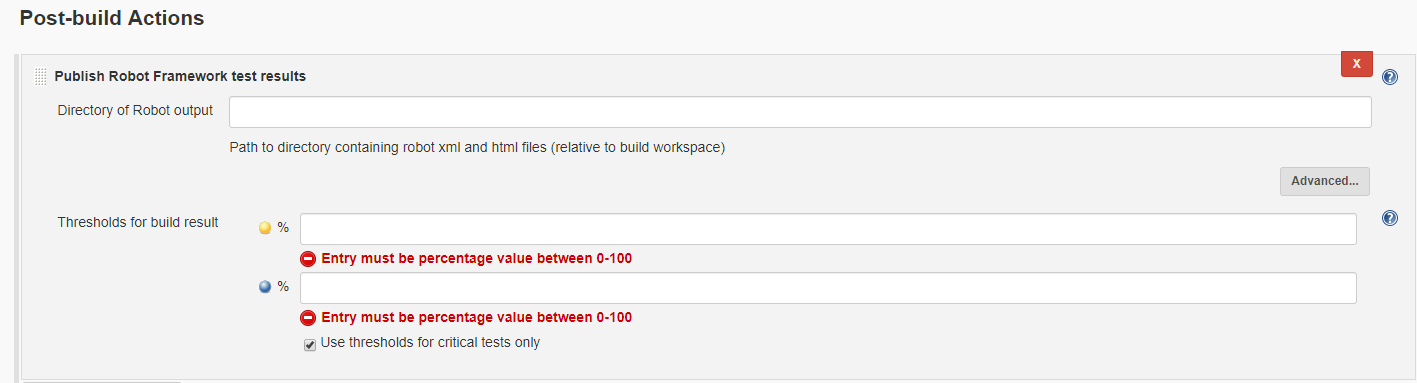
1、进入jenkins主页面，系统管理-管理插件-可选插件；在过滤栏输入“robot”,选择安装如下图的插件【Robot Framework plugin】

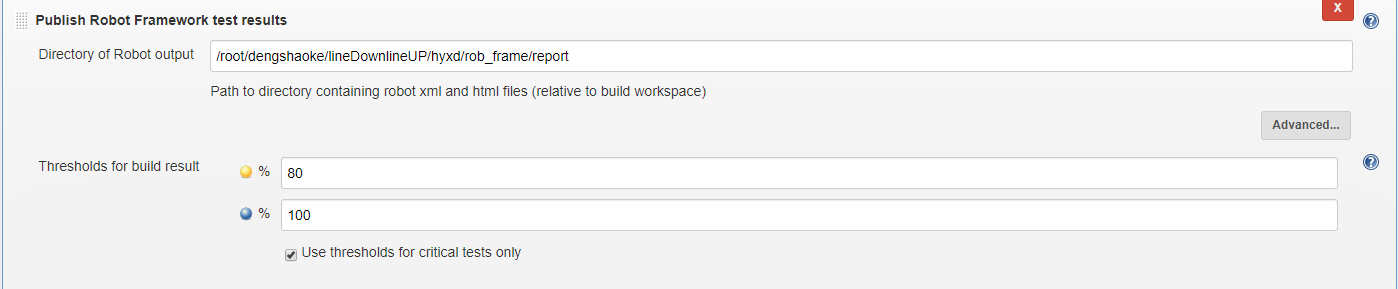


2、构建后的操作，选择“Publish Robot Framework test results”；

\*Directory of Robot output 填一个本地路径，要根据你的output文件放在哪里，可以不填

\*Thresholds for build result 测试用例执行成功率和通过率的设置，如80%和100%





## 执行构建

1、qtw\_interfacetest\_autoenv\_NoSendEmail

（1）测试环境自动化环境

（2）执行全部用例Level1/Level2/Level3

（3）不发送邮件

python /root/dengshaoke/lineDownlineUP/hyxd/rob\_frame/bin/rob.py -e auto -p pybot -d /root/dengshaoke/lineDownlineUP/hyxd/rob\_frame/report -f /root/dengshaoke/lineDownlineUP/hyxd/rob\_frame/qutouwang/App

2、qtw\_interfacetest\_testenv\_NoSendEmail

（1）测试环境qtw测试环境

（2）在去投网2.0测试//qtw\_invest\_api\_test执行构建机构后进行该构建

（3）sleep 600s延时构建，等待系统启动

（4）执行Level1/Level2/Level3全部用例

（5）不发送邮件

sleep 300

python /root/dengshaoke/lineDownlineUP/hyxd/rob\_frame/bin/rob.py -e test -p pybot -d /root/dengshaoke/lineDownlineUP/hyxd/rob\_frame/report -f /root/dengshaoke/lineDownlineUP/hyxd/rob\_frame/qutouwang/App

3、qtw\_interfacetest\_autoenv\_SendEmail

（1）测试环境自动化环境

（2）定时构建，每日执行

（3）执行用例等级为Level1的用例

（4）发送邮件

python /root/dengshaoke/lineDownlineUP/hyxd/rob\_frame/bin/rob.py -e auto -m YES -p pybot -d /root/dengshaoke/lineDownlineUP/hyxd/rob\_frame/report -f /root/dengshaoke/lineDownlineUP/hyxd/rob\_frame/qutouwang/App

4、qtw\_interfacetest\_error\_case

（1）执行上次构建报错的case

## 即时构建

在构建完成【去投网2.0测试】下面的[qtw-invest-api-test](http://192.168.1.46:8080/view/%E5%8E%BB%E6%8A%95%E7%BD%912.0%E6%B5%8B%E8%AF%95/job/qtw-invest-api-test/)任务后，会即时构建qtw\_interfacetest\_testenv\_NoSendEmail，达到冒烟测试的目的，有问题及时反馈出来。

java -jar jenkins-cli.jar -s http://192.168.1.31:8090/jenkins/ build qtw\_interfacetest\_testenv\_NoSendEmail --username admin --password 123456

参考资料：<https://www.cnblogs.com/honeybee/p/6525902.html>

1、下载jenkins-cli

jenkins-->系统管理-->Jenkins CLI：如下图



2、命令行执行

<http://192.168.1.31:8090/jenkins/>

查看jenkins-cli的帮助

java -jar jenkins-cli.jar -s <http://192.168.1.31:8090/jenkins/> help

查看job列表

java -jar jenkins-cli.jar -s <http://192.168.1.31:8090/jenkins/> list-jobs

## 其他遇到的问题

1、Jenkins打不开robotframework测试报告，报Opening Robot Framework report failed错误

参考资料：<https://blog.csdn.net/huashao0602/article/details/53925492>



2、Jenkins开放用户注册和匿名查看功能，可以不登陆而查看登录Jenkins

<https://blog.csdn.net/boonya/article/details/77362603>

3、jenkins的工作目录

cd /root/.jenkins

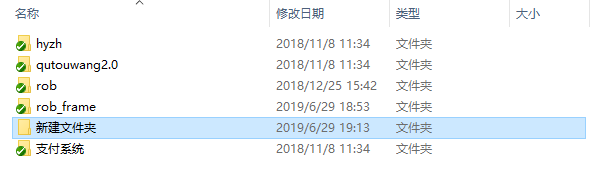
cd workspace

cd job

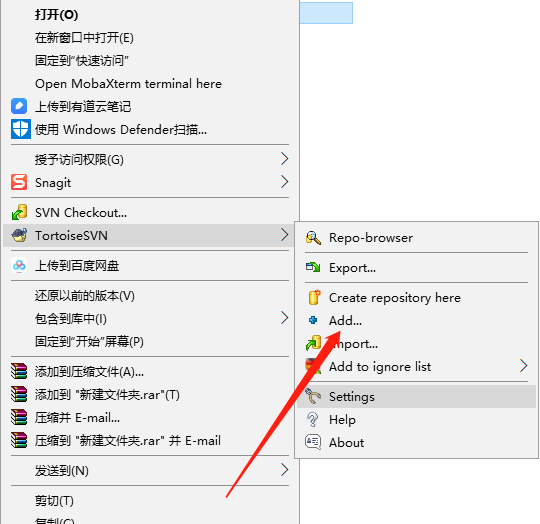
# SVN管理

## 添加项目到SVN

1、在SVN目录下新建文件夹



2、svn add，等待add成功后点击OK



3、svn commit



## linux下SVN常用命令

1、将文件checkout到本地

svn checkout path(path是服务器的目录)

2、更新到某个版本

svn update -r 版本号 path

svn update 更新当前目录以及子目录下的所有文件到最新版本

svn upate -r 200 test.cpp 将版本库中的test.cpp还原到版本200

简写 svn up

# 测试服务器

1、地址

192.168.1.31

2、账号

root/root123

3、python安装位置

/usr/local/lib/python3.6