## 安装jenkins ,Jenkins管理中安装testlink,robotframework插件

**<https://www.cnblogs.com/yangxia-test/p/4354328.html>**

## 二．slave挂载：slave称为node

<https://blog.csdn.net/liuchunming033/article/details/52025541>

<https://blog.csdn.net/liuchunming033/article/details/48652097>

没有Launch agent via Java Web Start时处理办法：

https://blog.csdn.net/e295166319/article/details/54946966

**挂载slave方法：**

**2.1：**通过ssh连接slave.(需要安装ssh slaves plugin插件) #master主动连接slave

2.1.1：通过username/password

2.1.2：通过username/private\_key

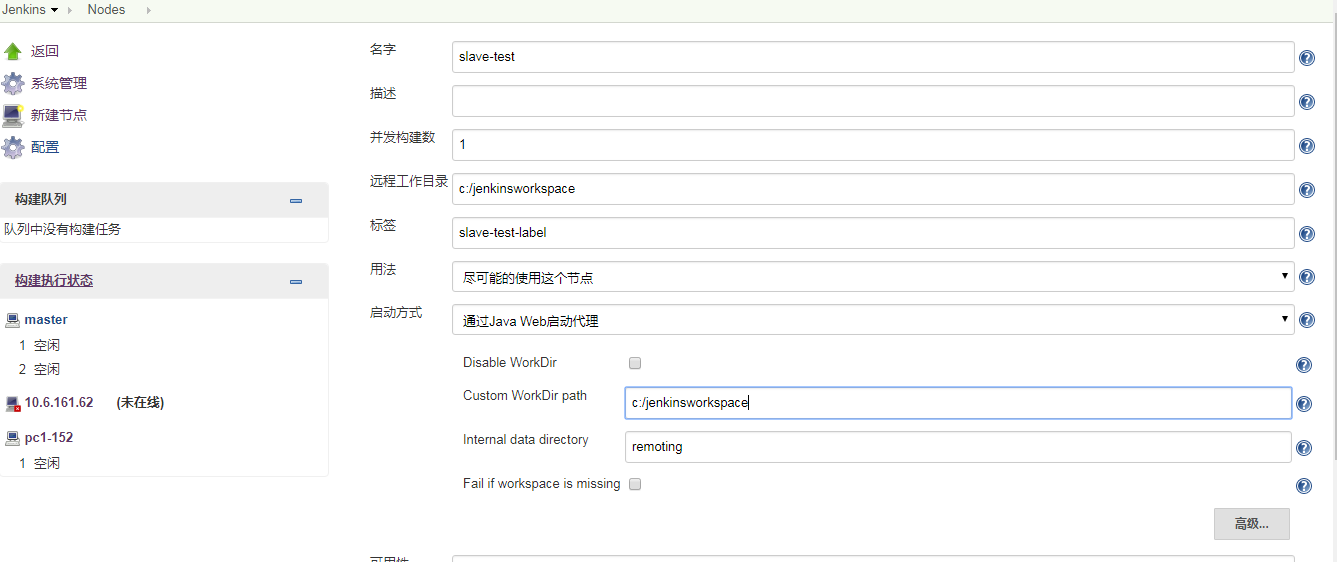
**2.2:** 通过JNLP连接node

不需要master能ssh上slave，只需要能ping通即可。Slave还可以作为服务运行在slave的机器上. #slave主动连接master

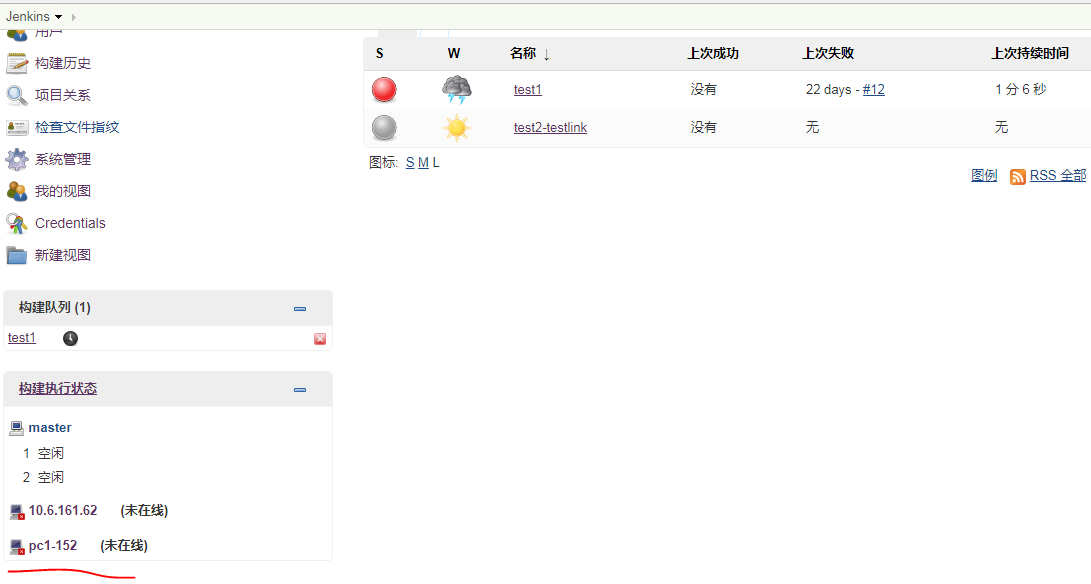
## 三．Windows上安装slave方法：

### 1.jenkins上创建节点：

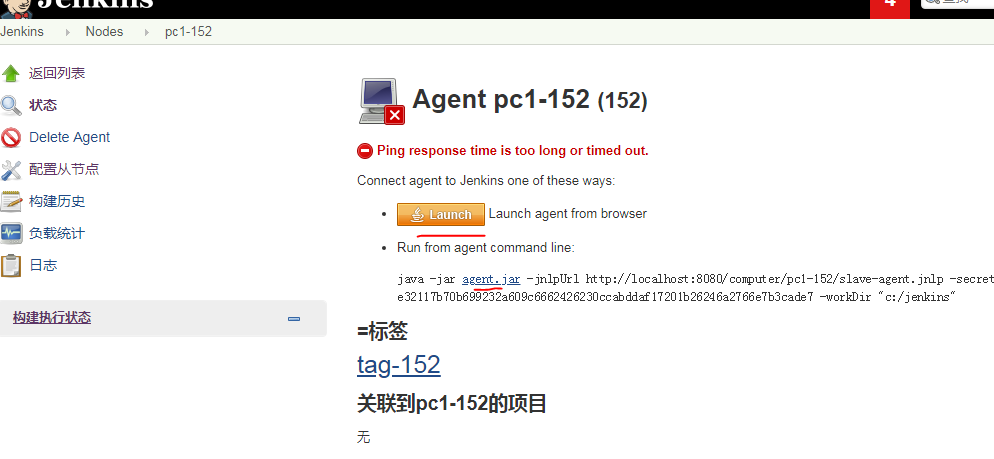
系统管理--管理节点--新建节点



### 2.slave环境必备条件：安装java jdk并配置好环境变量

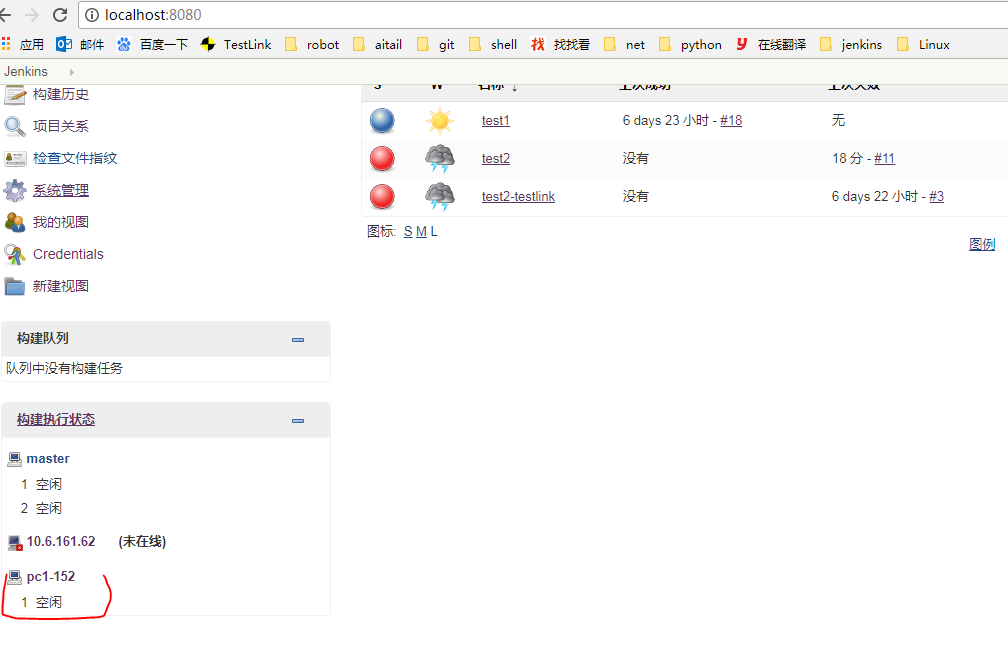


2.在slave所在的机器登录jenkins master，点击已创建好的windows节点名，



3.如上图，点击launch（有时可能需要升级java）; agent.jar也需要下载到本地（后面启动需要的文件）

**文件下载成功后按提示命令启动服务**

4.成功挂载后可以在jenkins页面看到slave有未在线状态转为空闲状态则挂载成功

## 四．创建job：

<https://www.cnblogs.com/zz27zz/p/7366575.html>

1. 创建任务
2. 填写必要的项目：

2.1 general/restrict where this project can be run(填写执行此任务的slave的标签)

2.2构建/填写对应的testlink 与任务信息

2.3构建填写要在slave上执行的命令（pybot -x junit.xml --include add\_site\_ap c:\yangjian\project\UI-Auto\AddUser\_To\_Altaicare\ ），

此命令在slave 的执行目录为：**工作目录\workspace\job名**（例：c:\jenlins\workspace\test1\）

2.4构建后操作：publish robot framework test result,填写robot输出output.xml的目录用于jenkins解析（jenkins的robot插件主要目的为解析目录）(robot没有制定output.xml生成目录时，会在命令执行的地方生成output.xml;log.html;report.html)

1. 日志查看：1.jenkins上查看日志；2.slave上执行的log会上传到master上的jenkins安装目录/jobs/job名下

## 五．Jenkins链接testlink (未成功)

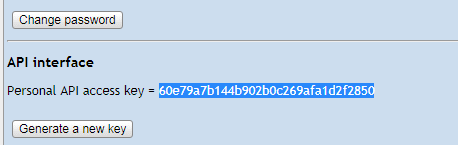
Link1:<https://blog.csdn.net/talorswfit20111208/article/details/53112885>

Link2:<https://www.cnblogs.com/bonjov1/p/4763349.html>

jenkins/系统设置/TestLink



Developer key是testlink中api interface /personal api access key:



Testlink中建立工程：

1. 再就是在job里配置Testlink了，在这之前需要先在Testlink里面建好测试工程、测试计划、测试组件、测试用例等信息。

Testlink中case与robot中case一一对应?

结果回填：

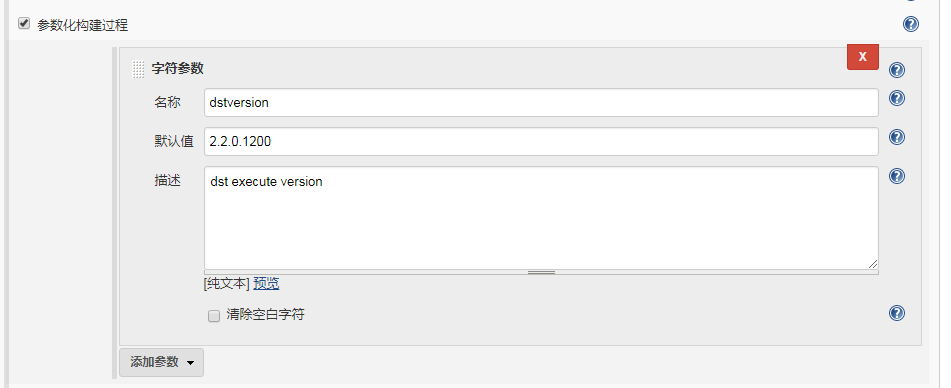
方法1：jenkins与testlink对接

方法2：testlink的python接口回填，定义成关键字

<https://www.cnblogs.com/-brenda/p/8697411.html>

<https://www.cnblogs.com/bonjov1/p/4763349.html>

## 参数化构建过程



自定义构建所需的变量

变量引用说明：不同的操作环境变量引用方式不同

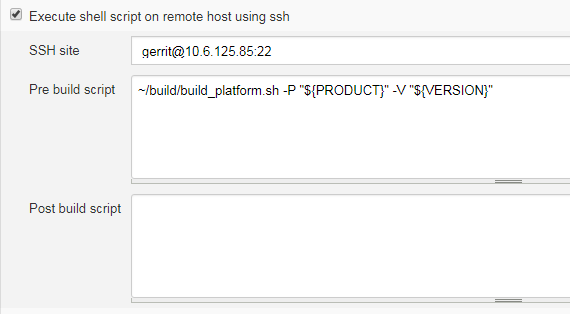
Shell -- $dstversion

Dos -- %dstversion%

Git源码管理-- ${dstversion}

## 构建环境

7.1远程执行脚本：

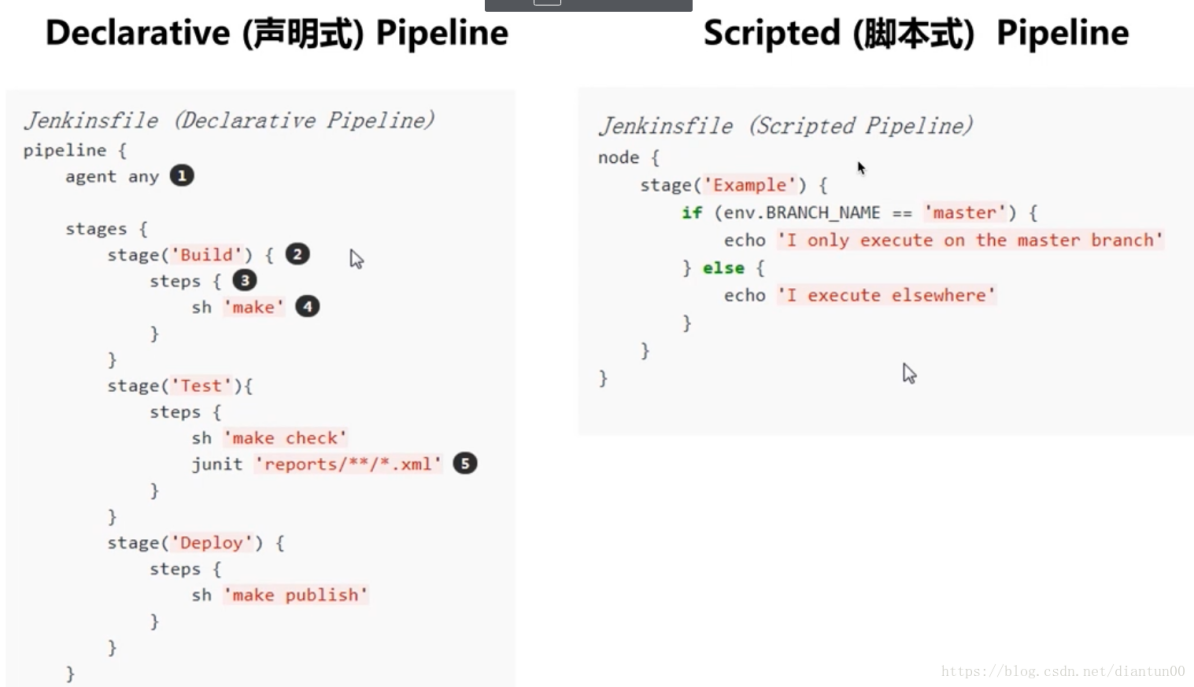


Pre build script---执行build之前执行的脚本

Post build script --执行build之后执行的脚本

## 流水线

### 8.1 pipeline:声明式与脚本式:



pipeline {  
 agent {label ‘slave1’} #指定agent

environment { CC = 'clang' bb=’123’} #环境变量，引用”${CC}”  
 tools {  
 maven 'M3'   
 }  
 stages {  
 stage('Preparation') {  
 steps {  
 git 'https://github.com/jglick/simple.git' #拉取代码  
 sh ‘./test.sh’ #执行shell

sh ‘ps -ef | grep “${CC}”’ #引用环境变量

sh ''' #多行shell

ps -ef | grep java;

ps -ef | grep "${version}";

ps -ef | grep "${CC}";

pwd

'''

}   
 }  
 stage('Build') {  
 steps {  
 script{  
 *if* (isUnix()) { #执行命令  
 sh 'mvn -Dmaven.test.failure.ignore clean package'  
 } *else* {  
 bat 'mvn -Dmaven.test.failure.ignore clean package'  
 }  
 }  
 }  
 }  
 stage('git action'){  
 steps{  
 dir(path: "./project"){ #切换目录

git(

branch: "master",

  credentialsid: "xxxx", #已添加到jenkins的凭证

并且此凭证由权限拉取此代码

  url : "http://xxx.git",

  changelog: true

      )

sh """git status

    git checkout v1.0

    git branch

   """

  }  
 }  
 }

stage('run other job'){ #此stage执行其他job  
 steps{  
 echo “this stage build other job”

build job: ‘demojob2’ #执行其他job

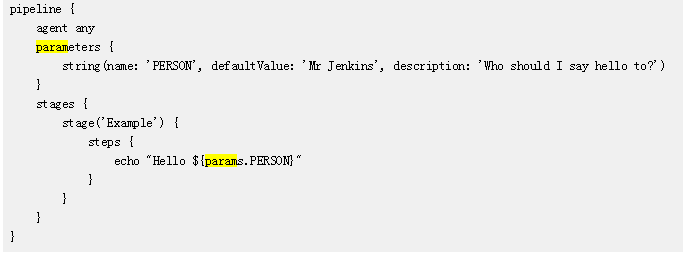
parameters:[

String(name:’myname’,value:’myvalue’)

]  
 }  
 }  
 }  
}

### 8.2参数化构建 VS pipeline内部变量

参数化构建：jenkins页面添加参数==》在pipeline中



### 8.3 pipeline中拉取代码的方式

使用Git插件拉取：

git(

branch: "master",

  credentialsid: "xxxx", #已添加到jenkins的凭证名并且此凭证由权限拉取此代码

  url : "http://xxx.git",

  changelog: true

      )

1. Linux上生成私钥+公钥==》公钥与代码端用户绑定==》私钥添加jenkins凭证（ssh name+key+创建私钥时输入的密码）==>pipeline中使用凭证拉取
2. Git插件只支持branch不支持tag的解决方法：

git插件拉取完后使用shell ：git checkout tag来切换tag

### 8.4 页面pipeline与pipeline from scm

页面pipeline直接在页面写pipeline

From scm 可用不同路径加载jenkinsfile (常用git上拉取)

### 8.5 jenkinsapi使用

pip install jenkinsapi

实例化，创建job，参数化构建job

self.server=Jenkins(*url*, username=*username*, password=*password*)

jk.server['pull\_webproject1'].get\_config()

self.server.create\_job(*config*["jobname"],str(xml))

self.server.build\_job(*jobname*,*params*)

## 源码管理

build trigger :构建触发器 ，为genkins提供一个可用于触发构建的特殊url

