

Jenkins + Gradle + SonarQube 项目持续集成并分析环境搭建

jenkins简单介绍

Jenkins 是一个可扩展的持续集成引擎。

主要用于

1. 持续、自动地构建/测试软件项目。
2. 监控一些定时执行的任务。

优势

1. 软件构建自动化：配置完成后，CI系统会依照预先制定的时间表，或者针对某一特定事件，对目标软件进行构建。
2. 构建可持续的自动化检查：CI系统能持续地获取新增或修改后签入的源代码，也就是说，当软件开发团队需要更新代码时，CI系统会自动地检查代码，并执行测试。
3. 构建可持续的自动化测试：构建检查的扩展部分，构建后执行预先制定的一套测试规则，完成后触发通知 (Email, SMS, etc.)。
4. 生成后后续过程的自动化：当自动化检查和测试成功完成，软件构建的周期中可能也需要一些额外的任务，诸如部署、打包、发布等。

jenkins安装

前往[官网](#),选择其中的.war文件进行下载。

安装方式 | 必须安装jdk环境

1. 必须事先安装Tomcat服务器，然后将.war文件放到webapps目录下，重启Tomcat就可以了。
2. 直接进入cmd，使用命令

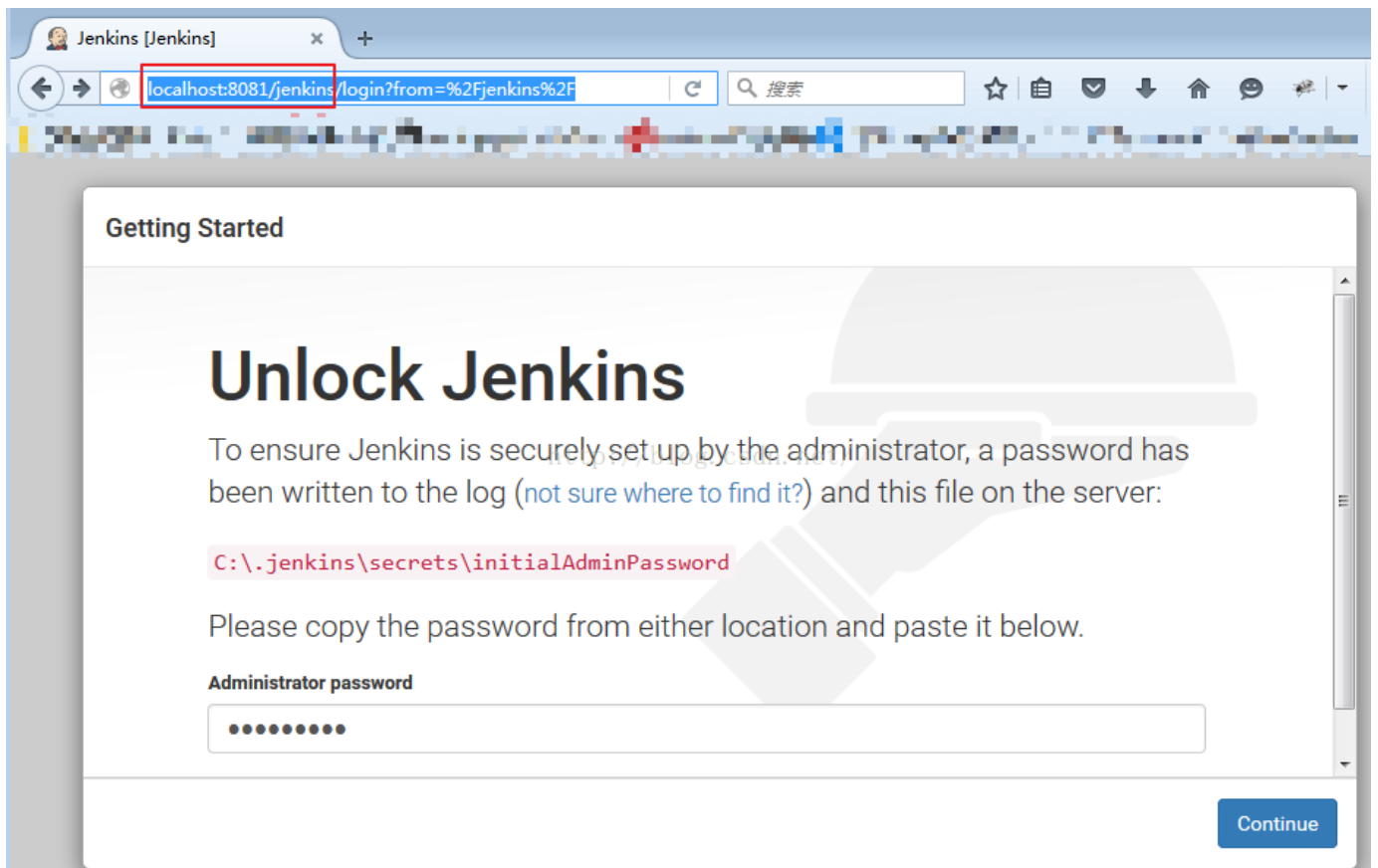
```
java -jar jenkins.war
```

使用这种方式进行安装，使用的是jenkins内置的服务器。

安装成功（cmd会出现类似jenkins is running）后，然后在浏览器输入<http://localhost:8080>

如果出现“网络无法运作”，请检查当前是否使用的代理服务器，应该转为“直接连接”

网页加载成功后，出现一下界面



在框中输入控制台输出的密钥

```
*****
Jenkins initial setup is required. A security token is required to proceed.
Please use the following security token to proceed to installation:

41d2b60b0e4cb5bf2025d33b21cb
*****
```

如果是通过Tomcat安装，密钥在webapp.jenkins\secrets\initialAdminPassword文件中
之后就是注册用户名
注册成功并登陆后，便出现一下界面

安装完成后，进行环境的配置


1. 点击系统管理

- 新建
- 用户
- 任务历史
- 项目关系
- 检查文件指纹
- 系统管理
- My Views
- Credentials

构建队列

2. 点击Global Tool Configuration

[Configure Credentials](#)
Configure the credential providers and types

[Global Tool Configuration](#)
Configure tools, their locations and automatic installers.

[读取设置](#)
放弃当前内存中所有的设置信息并从配置文件中重新读取 仅用于当您手动修改配置文件时重新读取设置。

[管理插件](#)
添加、删除、禁用或启用Jenkins功能扩展插件。

3. 配置jdk， 配置git

JDK

JDK 安装

JDK

别名

jdk

JAVA_HOME

C:\Program Files\Java\jdk1.8.0_101

☐ 自动安装

删除 JDK

新增 JDK

系统下JDK 安装列表

Git

Git installations

Git

Name

git

Path to Git executable

C:\Program Files\Git\bin\git.exe

☐ 自动安装

4. 配置gradle

Gradle

Gradle 安装

Gradle name	gradle2.14.1
GRADLE_HOME	C:\Users\peiyu_wang\gradle\wrapper\dists\gradle-2.14.1-all\8bnwg5hd3w55iofp58khbp6yv\gradle-2.14.1
<input type="checkbox"/> 自动安装	

Gradle name	gradle3.3
GRADLE_HOME	C:\Users\peiyu_wang\gradle\wrapper\dists\gradle-3.3-all\55gk2rcmfc6p2dg9u9ohc3hw9\gradle-3.3
<input type="checkbox"/> 自动安装	

Gradle name	gradle2.10
GRADLE_HOME	C:\Users\peiyu_wang\gradle\wrapper\dists\gradle-2.10-all\4w5fzrkeut1ox71xslb49gst\gradle-2.10
<input type="checkbox"/> 自动安装	

jenkins集成的项目使用的gradle版本，这里必须配置，因为自动构建需要和项目使用相同版本的gradle

5.点击Save

6.安装Android SDK

点击系统设置

管理Jenkins



[系统设置](#)
全局设置&路径



[Configure Global Security](#)
Secure Jenkins; define who is allowed to access/use the system.

7.配置Android Home全局属性，全局属性对所有项目生效

全局属性

☒ Environment variables

键值对列表

键 ANDROID_HOME

值 C:\Android_SDK

增加

8.点击Save

9.安装Android Line插件，点击管理插件



[Configure Credentials](#)
Configure the credential providers and types



[Global Tool Configuration](#)
Configure tools, their locations and automatic installers.



[读取设置](#)
放弃当前内存中所有的设置信息并从配置文件中重新读取 仅用于当您手动修改配置文件时重新读取设置。



[管理插件](#)
添加、删除、禁用或启用Jenkins功能扩展插件。

创建Job

1.

点击新建

-  新建
-  用户
-  任务历史
-  项目关系
-  检查文件指纹
-  系统管理
-  My Views
-  Credentials

构建队列

2.填写名称，选择自由风格的软件项目

Enter an item name

» This field cannot be empty, please enter a valid name



构建一个自由风格的软件项目

这是Jenkins的主要功能.Jenkins将会结合任何SCM和任何构建系统来构建你的项目, 甚至可以构建软件以外的系统.



构建一个maven项目

构建一个maven项目.Jenkins利用你的POM文件,这样可以大大减轻构建配置.

3.简单介绍下面内容

General 源码管理 构建触发器 构建环境 构建 构建后操作

项目名称 Load

描述

[Plain text] 预览

☐ GitHub project

☐ Throttle builds

☒ 丢弃旧的构建

Strategy Log Rotation

保持构建的天数 1

如果非空，构建记录将保存此天数

保持构建的最大个数 5

如果非空，最多此数目的构建记录将被保存

☐ 参数化构建过程

☐ 关闭构建

☐ 在必要的时候并发构建

高级...

高级...

1. 丢弃旧的构建

1.1 保持构建的天数，表示构建的内容在1天内不被删除，操作1天后删除

1.2 表示允许保持构建的最大数量，当超过这个数量时，最前面的构建会被删除

2. 参数化构建，就是在构建时传入一下参数，这些参数会被配置到项目中，并且可以影响后面的构建步骤。详情请点击有点的？

4. 配置项目Git地址，可以选择构建的分支

☐ None

☒ Git

Repositories

Repository URL

Please enter Git repository.

Credentials - none - Add

高级...

Add Repository

Branches to build

Branch Specifier (blank for 'any') */*master

Add Branch

源码库浏览器 (自动)

Additional Behaviours Add

☐ Subversion

5. 点击Add填写Git账号和密码

Add Credentials

Domain: Global credentials (unrestricted)

Kind: Username with password

Scope: Global (Jenkins, nodes, items, all child items, etc)

Username: wpy

Password: ...

ID:

Description:

Add Cancel

6. 构建触发器，就是什么时候触发构建

构建触发器

☐ 触发远程构建 (例如,使用脚本)

☐ Build after other projects are built

☒ Build periodically

日程表: H 2 ***

Would last have run at 2017年7月25日 星期二 上午02时56分40秒 CST; would next run at 2017年7月26日 星期三 上午02时56分40秒 CST.

☐ GitHub hook trigger for GITScm polling

☐ Poll SCM

有多种触发形式

这里介绍第三种，定时构建

可以规定每天，或者某个时候进行定时构建。日程表中的内容是安装一个格式编写的，详情可以点击右边的？

7. 点击增加构建步骤，选择gradle script

☐ Use secret text(s) or file(s)

构建

增加构建步骤

- Execute Windows batch command
- Execute shell
- Invoke Ant
- Invoke Gradle script
- Invoke top-level Maven targets
- Run with timeout

8. 可以选择invoke Gradle,此时是从刚才配置的Gradle版本选择一个，但是必须和项目是同一个版本,或者选择Use Gradle Wrapper，此时的Wrapper location,就是项目的根目录，有build.gradle,gradlew的根目录。Tasks就是clean build这类gradle命令，可以连着写。

推荐在命令后加上

```
--stacktrace --debug
```


例如

```
clean assembleRelease --stacktrace --debug
```

Invoke Gradle script

☒ Invoke Gradle

Gradle Version:

☐ Use Gradle Wrapper

Tasks:

Switches:

System properties:

Pass all job parameters as System properties: ☐

Project properties:

Pass all job parameters as Project properties: ☐

Root Build script:

Build File:

Specify Gradle build file to run. Also, [some environment variables are available to the build script](#)

Force GRADLE_USER_HOME to use workspace: ☐

但是当我们要使用SonarQube时，只能选择Use Gradle Wrapper
配置如下，其中Root Build script就是要构建的根目录。此时选择的时andfix模块

General 源码管理 构建触发器 构建环境 **构建** 构建后操作

☐ Invoke Gradle

☒ Use Gradle Wrapper

Make gradlew executable ☒

Wrapper location

Tasks

Switches

System properties

Pass all job parameters as System properties ☐

Project properties

Pass all job parameters as Project properties ☐

Root Build script

If your workspace has the top-level build.gradle in somewhere other than the module root directory, specify the path (relative to the module root) here, such as \${workspace}/parent/ instead of just \${workspace}. If left empty, defaults to build.gradle (from [Gradle Plugin](#))

Build File

保存 Apply

Specify Gradle build file to run. Also, [some environment variables are available to the build script](#)

9. 设置构建完成之后的操作

构建后操作

E-mail Notification

Recipients

Whitespace-separated list of recipient addresses. May reference build parameters like \$PARAM. E-mail will be sent when a build fails, becomes unstable or returns to stable.

☒ 每次不稳定的构建都发送邮件通知

☐ 单独发送邮件给对构建造成不良影响的责任人

增加构建后操作步骤

设置邮件提醒。但是使用邮件还得配置其他，这里不做介绍。

10. 点击Save

在主界面点击刚才创建的项目

返回面板

状态

修改记录

工作空间

立即构建

删除 Project

配置

Project PeiyuApplication

测试jenkins

工作区

最新修改记录

Build History	构建历史
find	X
#18	2017-7-25 下午5:32
#17	2017-7-25 下午4:28
#16	2017-7-25 下午3:25
#15	2017-7-25 下午3:11
#14	2017-7-25 下午3:09

相关连接

- [Last build\(#18\).2 小时 34 分之前](#)
- [Last stable build\(#18\).2 小时 34 分之前](#)
- [Last successful build\(#18\).2 小时 34 分之前](#)
- [Last completed build\(#18\).2 小时 34 分之前](#)

点击立即构建。

如果出现失败，点击#*，进入后点击Console Output查看控制台信息

一般错误：

1.Gradle版本不一致，更改Gradle版本

2.项目使用的SDK版本在刚才配置的sdk路径中没有，安装相应的sdk版本

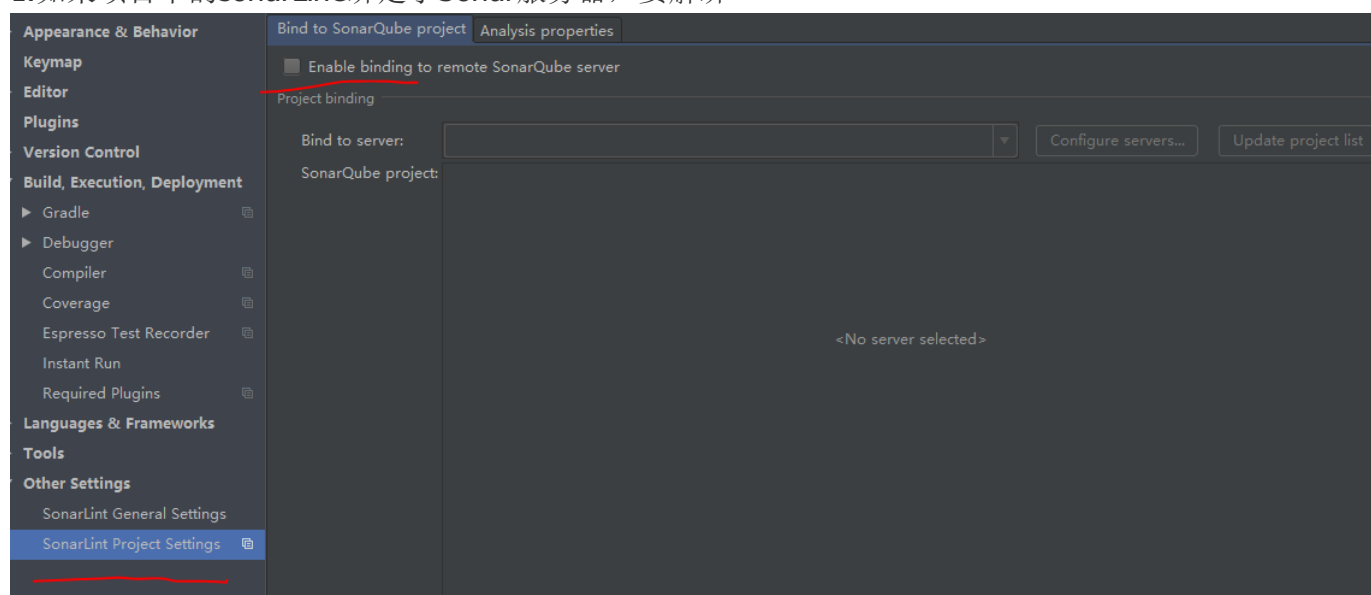
3.项目中使用的sdk路径找不到，这时可以更改项目根目录下的local.properties文件，将sdk路径改为正确的路径

构建成功后，图片时蓝色的。

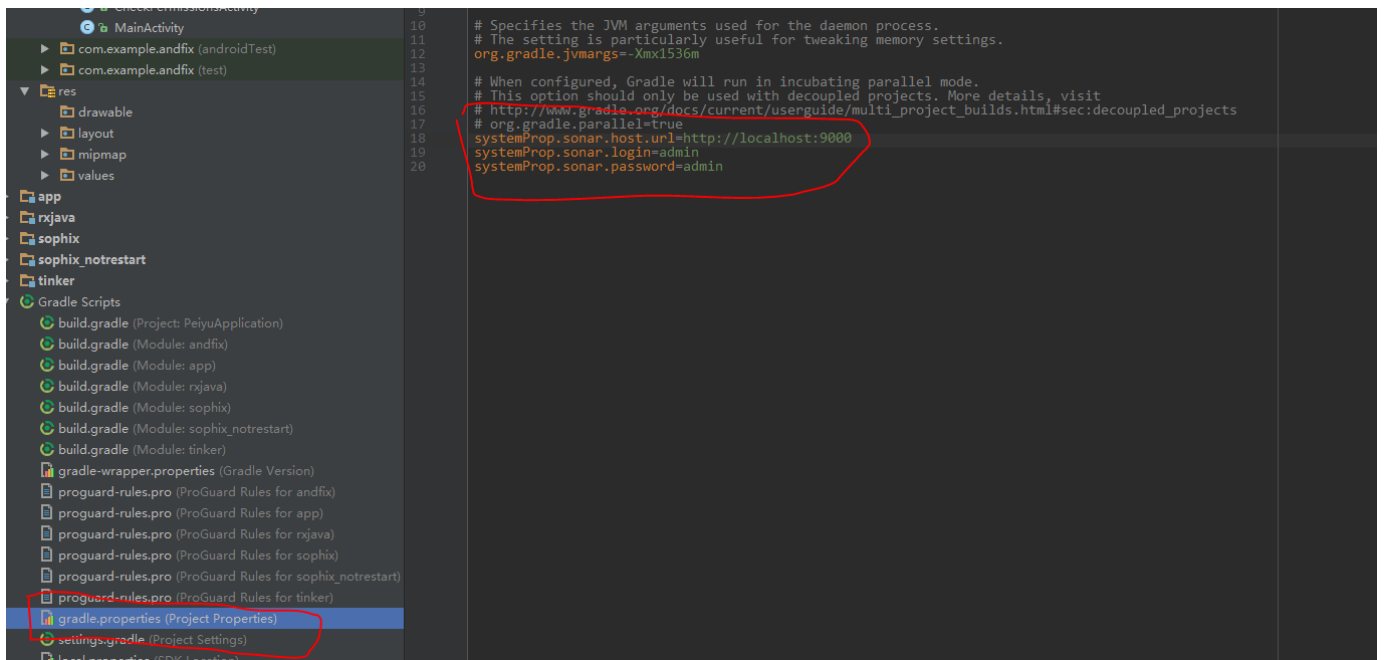
然后进入sonarQube服务器，便可以看到刚才自动构建的项目。

注意

1.如果项目中的sonarLine绑定了Sonar服务器，要解绑



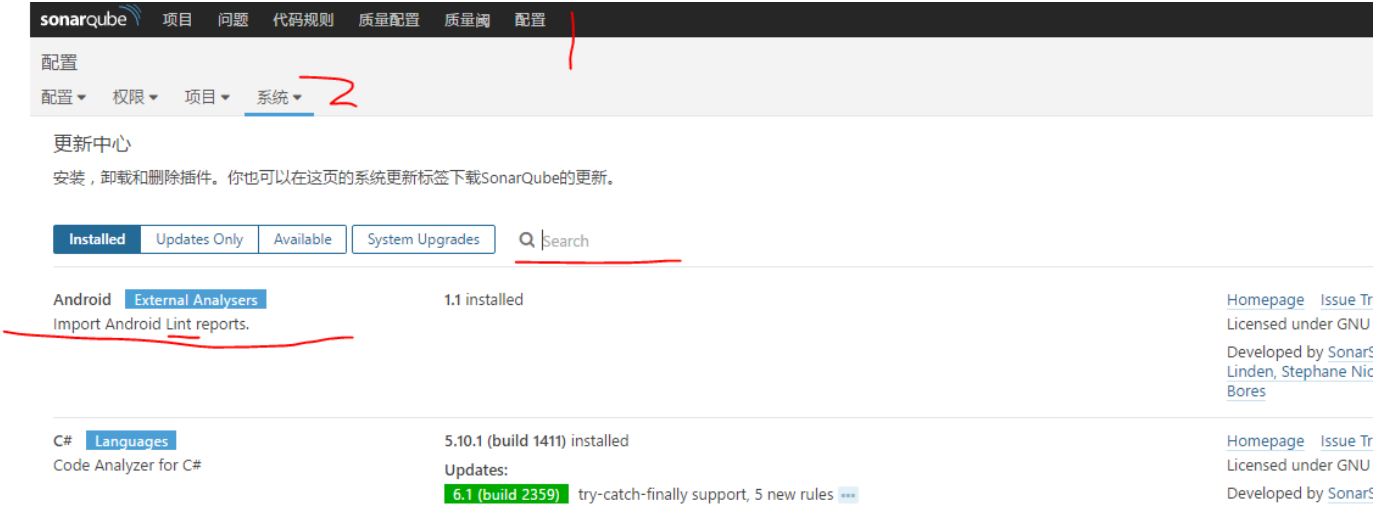
2.配置项目的gradle.properties



3.确保git上的项目有2中的配置

SonarQube分析配置

1. 安装Android Lint



2. 设置Android Lint的父类为Sonar Way



此时Android Lint就有了Android Lint检测规则，也有了原来Sonar Way的检测规则。

3.设置Android Lint为默认的质量配置，之后的自动构建使用Android Lint

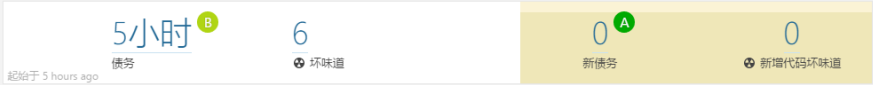
4.更新的项目情况，在SonarQube可以清晰的看到

质量阈 正常

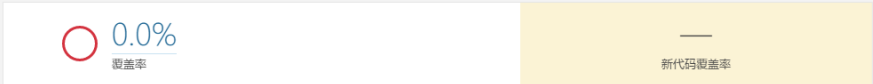
Bugs & 漏洞



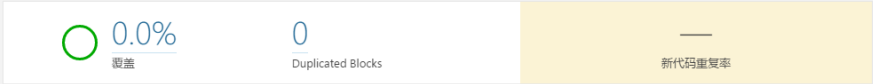
坏味道



覆盖率



重复



XS 120
代码行

Java 120

无标签

质量阈
(默认) [SonarQube way](#)

质量配置
(Java) [Android Lint](#)

关键字

活动
July 25, 2017
[unspecified](#)
质量配置: Use 'Android Lint' (Java)
质量配置: Stop using 'Sonar way' (Java)

July 25, 2017
质量配置: Changes in 'Sonar way' (Java)

July 25, 2017
质量配置: Changes in 'Sonar way' (Java)

July 25, 2017
已分析项目

July 25, 2017
已分析项目
[更多](#)