sonar服务器扫描本本地工程代码质量用户使用手册

严格控制金融业IT工程质量，开源sonar是行业比较成熟的解决方案，本手册对sonarqube服务器的安装、使用进行了简单介绍，对于那些希望傻瓜式使用sonar完成代码质量扫描的用户，请直接查看第4部分：《**4 sonar-scanner 扫描工程**》自助完成工程代码的sonar扫描，完成任务后直接点击远程sonarqube的服务器地址查看结果。

目录

[1 sonarqube 3](#_Toc515453905)

[1.1 安装sonarqube服务 3](#_Toc515453906)

[1.2 基本配置 3](#_Toc515453907)

[1.2.1 start postgresql server 3](#_Toc515453908)

[1.2.2 配置sonar 的 database 3](#_Toc515453909)

[1.2.3 $SONAR\_HOME/sonar.properties 3](#_Toc515453910)

[1.2.4 $SONAR\_HOME/sonar-runner.properties 4](#_Toc515453911)

[1.2.5 $SONAR\_HOME/sonar-scanner.properties 4](#_Toc515453912)

[2 启动sonar服务器 4](#_Toc515453913)

[3 maven 工程使用sonarqube 7](#_Toc515453914)

[3. 1 pom.xml加入插件 7](#_Toc515453915)

[3.2 $MAVEN\_HOME/conf/settings.xml 7](#_Toc515453916)

[3.3对maven工程开始执行sonar扫描 8](#_Toc515453917)

[４sonar-scanner 扫描工程 9](#_Toc515453918)

[4.1 JDK1.8 9](#_Toc515453919)

[4.2 客户端 sonar-scanner-cli-3.1.0.1141 9](#_Toc515453920)

[4.3 配置工程的sonar-poject.properties 10](#_Toc515453921)

[4.4 conf/sonar-scanner.properties 11](#_Toc515453922)

[4.4 扫描本地工程并查看报告 11](#_Toc515453923)

[4.5 直接扫描源码潜在的问题： 13](#_Toc515453924)

[4.5.1需要提供 bytecode 13](#_Toc515453925)

[4.5.2 out of memory/HeapSpace 14](#_Toc515453926)

[5 Sonar 案例 16](#_Toc515453927)

[5.1 开源插件 16](#_Toc515453928)

[5.2 为写通用的rules 16](#_Toc515453929)

[5.1.1Create a  SonarQube plugin 16](#_Toc515453930)

[5.1.2 Put a dependency on the API of the language plugin for which you are writing coding rules. 18](#_Toc515453931)

[5.1.3 Create as many custom rules as required 18](#_Toc515453932)

[5.1.4 Generate the SonarQube plugin (jar file) 18](#_Toc515453933)

[5.1.5 Place this jar file in the SONARQUBE\_HOME/extensions/plugins directory 18](#_Toc515453934)

[5.1.6 Restart SonarQube server 18](#_Toc515453935)

[5.2 Custom Rules for java(write your own rule) 18](#_Toc515453936)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 起草 | 审核 | 时间 | 版本 | 密级 |
| 姜鹏 | 余强 | 2018/05/22 | 1.0 | 仅内部使用 |

# 1 sonarqube

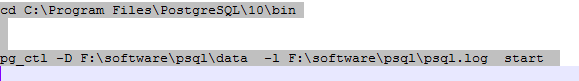
## 1.1 安装sonarqube服务

sonarqube-7.1

## 1.2 基本配置

sonar支持 Derby, H2等内嵌型数据库。(embedded database) ，教程以postgresql为例：

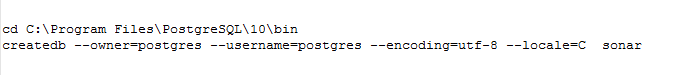
### 1.2.1 start postgresql server



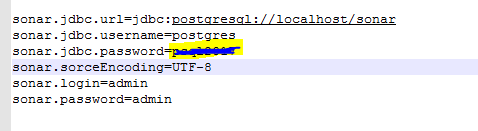
### 1.2.2 配置sonar 的 database

这儿，新建一个sonar 的数据库sonar

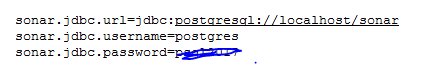
创建一个sonar database



### 1.2.3 $SONAR\_HOME/sonar.properties



### 1.2.4 $SONAR\_HOME/sonar-runner.properties



### 1.2.5 $SONAR\_HOME/sonar-scanner.properties

sonar.host.url=http://localhost:9000

sonar服务器已经准备就绪，随时可以开始使用。

# 2 启动sonar服务器

$SONAR\_HOME/bin/startSonar.sh

启动后，查看sonar数据库的表：

56个表已经生成。简单介绍一下基本的几张表。

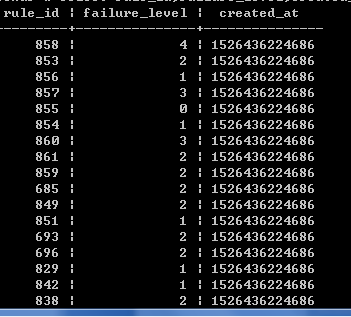
users 表： 存储管理员的账号信息



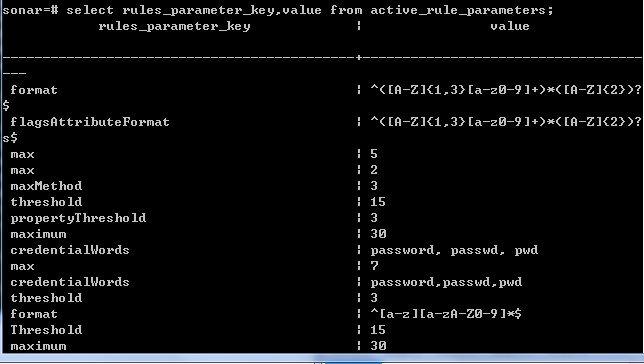
projects: 记录所有扫描的源文件，按项目目录迭代遍历



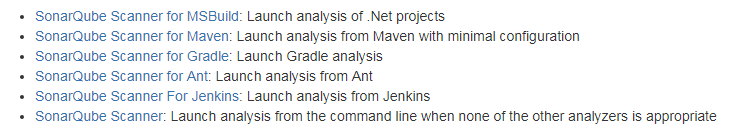
active\_rules： 记录扫描结果，failure\_level: Blocker, Critical, Issues 几个级别



active\_rule\_parameters：配置规则



从下图是可以了解到，sonar支持以下启动方式： .Net, maven, Gradle, Ant, Jekins，甚至soneqube cli 命令行启动扫描分析：



下面以流形的maven为例，介绍器使用。

# 3 maven 工程使用sonarqube

## 3. 1 pom.xml加入插件

<plugin>

<groupId>org.sonarsource.scanner.maven</groupId>

<artifactId>sonar-maven-plugin</artifactId>

<version>3.4.0.905</version>

</plugin>

## 3.2 $MAVEN\_HOME/conf/settings.xml

<settings>

<pluginGroups>

<pluginGroup>org.sonarsource.scanner.maven</pluginGroup>

</pluginGroups>

<profiles>

<profile>

<id>sonar</id>

<activation>

<activeByDefault>true</activeByDefault>

</activation>

<properties>

<!-- Optional URL to server. Default value is http://localhost:9000 -->

<sonar.host.url>

http://localhost:9000

</sonar.host.url>

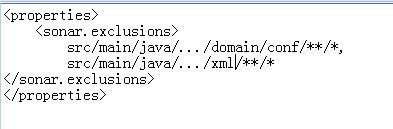
</properties>

</profile>

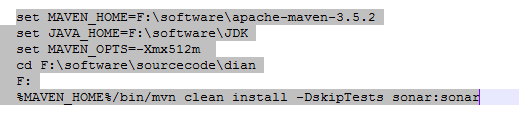
</profiles>

</settings>

我们还可以在maven中配置忽略规则，比如，我们不希望src/main/java/…/conf/..，src/main/java/…/conf/..进行扫描分析：



## 3.3对maven工程开始执行sonar扫描



以上白手起家(start from scrach) 介绍了如何从搭建soanr服务器，到maven工程的扫描。下面仅从用户快速使用的角度来介绍sonar的使用。

# 4 sonar-scanner 扫描工程

适用场景：

公司已有可用的sonar服务器，客户端只需要一个sonar-scnner的客户端即可。

针对汽融开发团队， 笔者量身定制了瘦客户端。汽融已经搭建有sonar服务器，

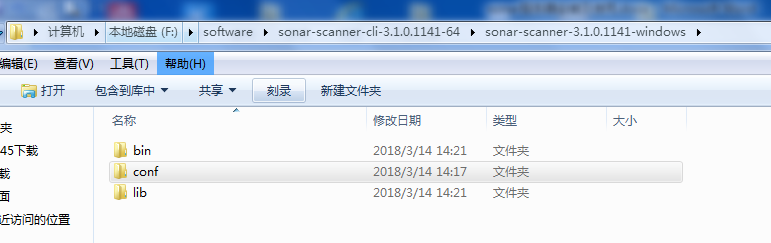
<http://10.14.200.67:9000>，目前处于正常运行状态。

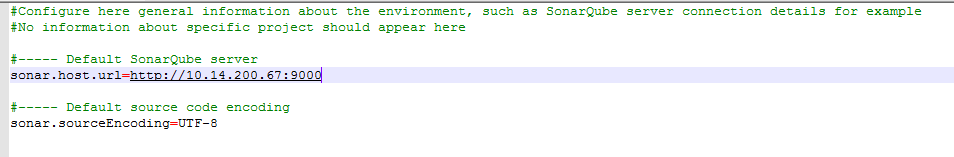
## 4.1 JDK1.8

如果本机没有安装，请 [\\apphq.pab.com.cn\信息科技部白名单软件发布\JDK路径安装JDK1.8](file:///\\apphq.pab.com.cn\信息科技部白名单软件发布\JDK路径安装JDK1.8)版本。

## 4.2 客户端 sonar-scanner-cli-3.1.0.1141

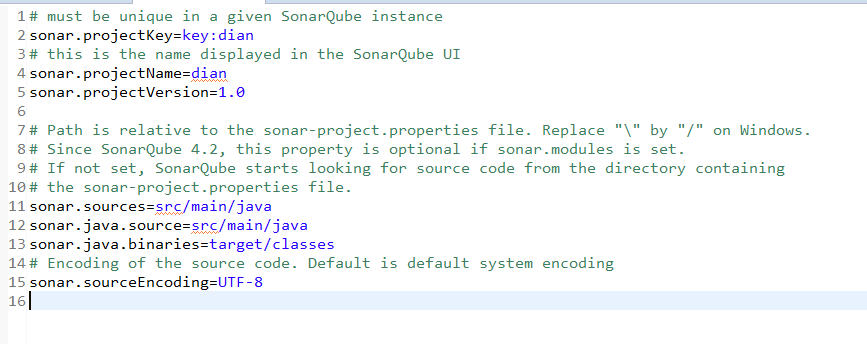
解压后，如图：

修改conf/sonar-scanner.properties



## 4.3 配置工程的sonar-poject.properties

在你需要扫描的工程下新建一个sonar-poject.properties



sonar.projectKey 请配置一个你自己的key, 如：

sonar.projectKey=key:cmis\_zhangsan

sonar.java.binarie 务必指定工程路径的编译后的目录，如：

sonar.java.binaries=target/classes

sonar.java.source请指定工程路径的java源文件的source folder，如：

sonar.java.source=src/main/java

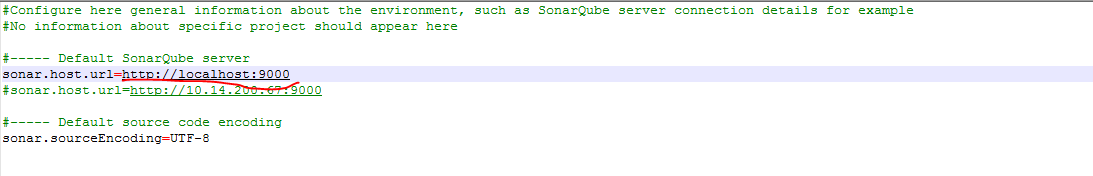
## 4.4 conf/sonar-scanner.properties

告诉sonar-scanner sonar服务器的位置，IP和端口，汽融暂时可以使用

sonar.host.url=http://10.14.200.67:9000

更多配置参数可以参考：

https://docs.sonarqube.org/display/SONAR/Analysis+Parameters



## 4.5 扫描本地工程并查看报告

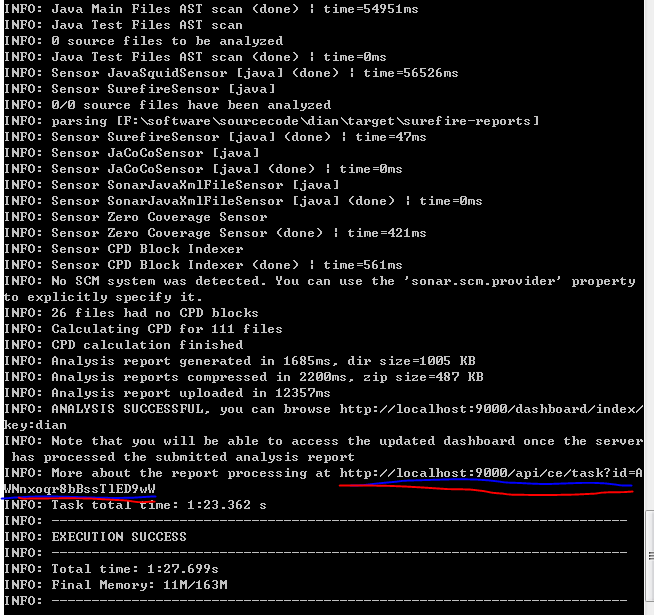
在DOS命令行下，

set PROJECT\_HOME=D:/java\_project/dian

cd %PROJECT\_HOME%

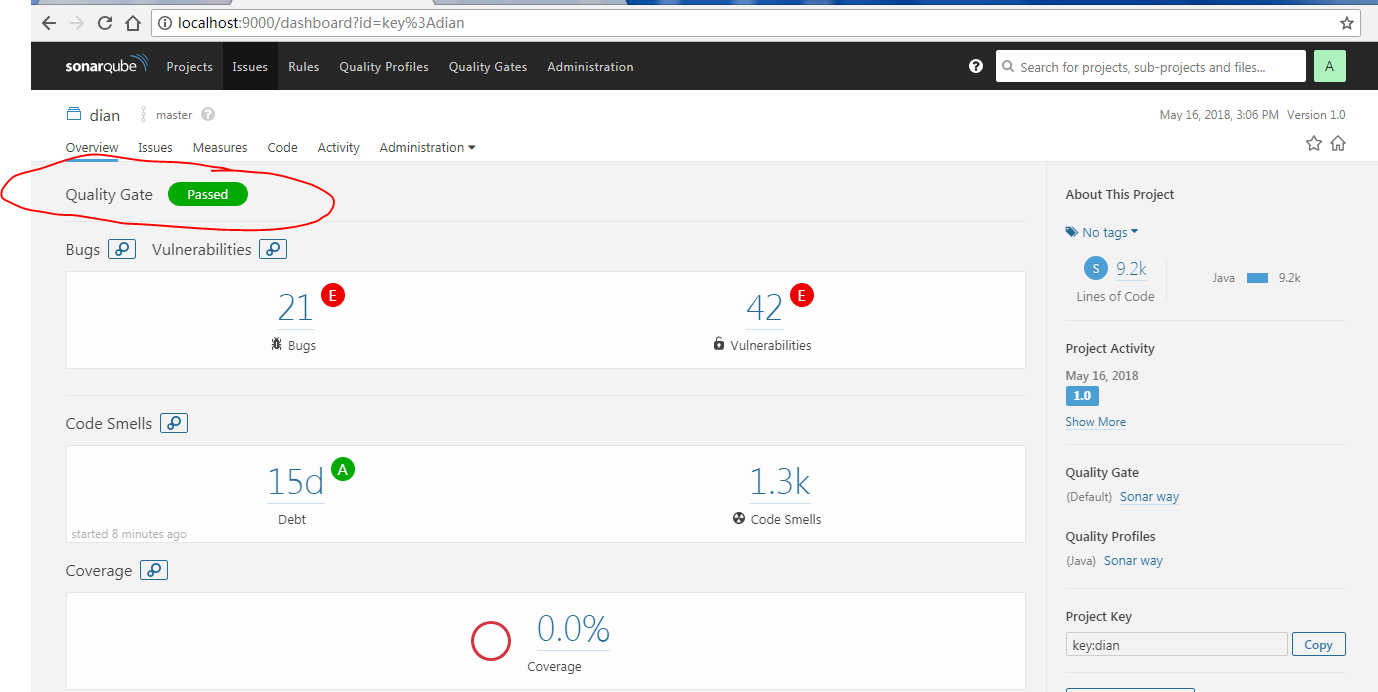
%SONAR\_SCANNER\_HOME/bin/sonar-scanner

扫描完成后，生成一个URL，点击进去查看结果



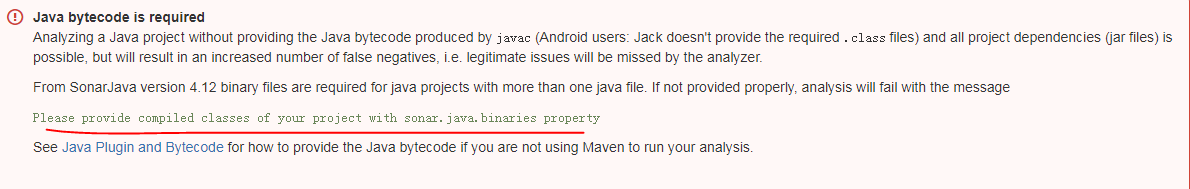
http://localhost:9000/api/ce/task?id=AWNnxoqr8bBssTlED9wW

sonar 质量“通过” passed



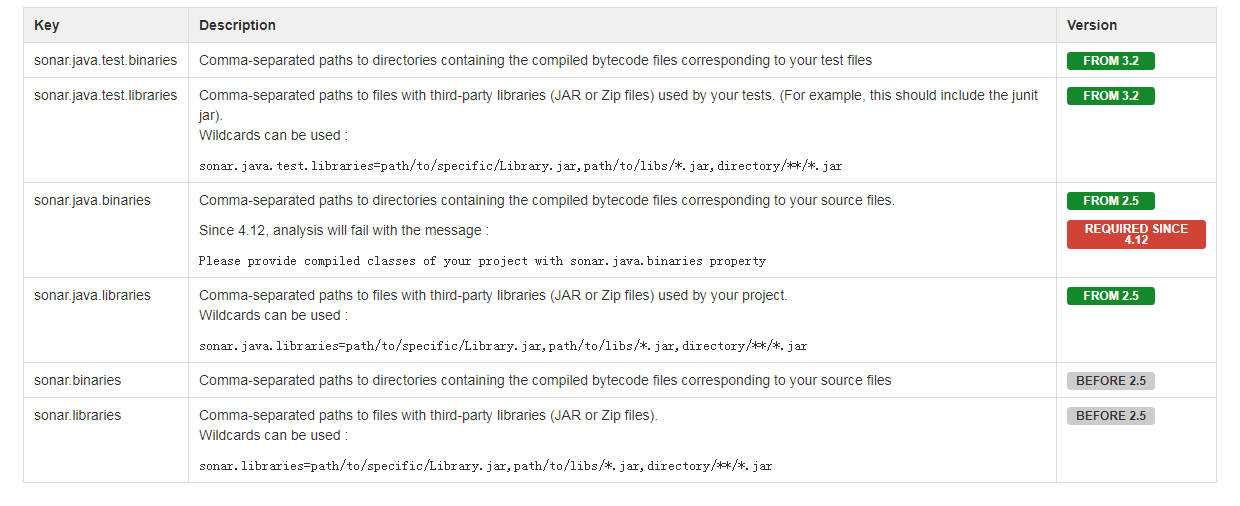
## 4.6 直接扫描源码潜在的问题：

### 4.6.1需要提供 bytecode

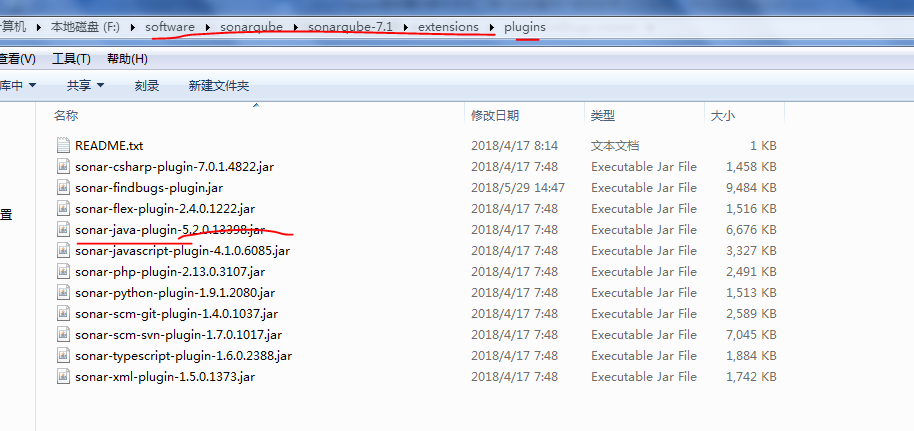


未使用 maven 扫描代码，提示错误，要求提供byecode。

<https://docs.sonarqube.org/display/PLUG/Java+Plugin+and+Bytecode>



查看 sonarqube 的 插件：



### 4.6.2 out of memory/HeapSpace

当使用SONAR-SCANNER扫描，默认的JVM经常会抛出out of memory /Heap Space 异常，下面给出示范：加上set SONAR\_SCANNER\_OPTS=-Xmx512m, 或者：

set SONAR\_SCANNER\_OPTS=-Xmx512m –Xss128k， 可以根据实际情况调节值。

cd F:\software\YMX\WeChat\examples

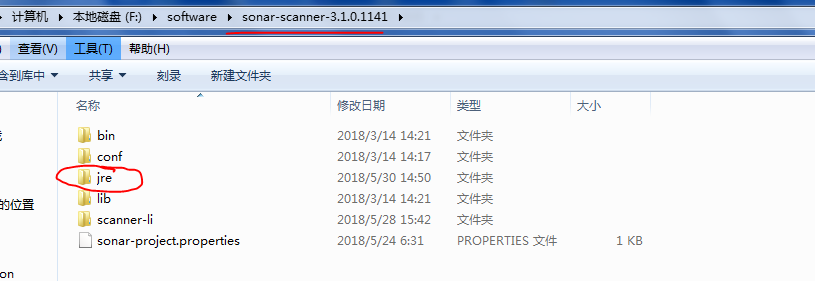
set JAVA\_HOME=F:\software\JDK

set SONAR\_SCANNER\_OPTS=-Xmx512m

F:\software\sonar-scanner-3.1.0.1141\bin\sonar-scanner

### 4.6.3 sonar-scanner找不到 java.exe

当使用SONAR-SCANNER扫描，如果遇到找不到java.exe, 请将JDK8的jre文件夹全部拷贝至$SONAR\_SCANNER\_HOME ,如下图所示：



# 5 Sonar 案例

<https://docs.sonarqube.org/>

## 5.1 开源插件

https://redirect.sonarsource.com/doc/plugin-library.html

## 5.2 为写通用的rules

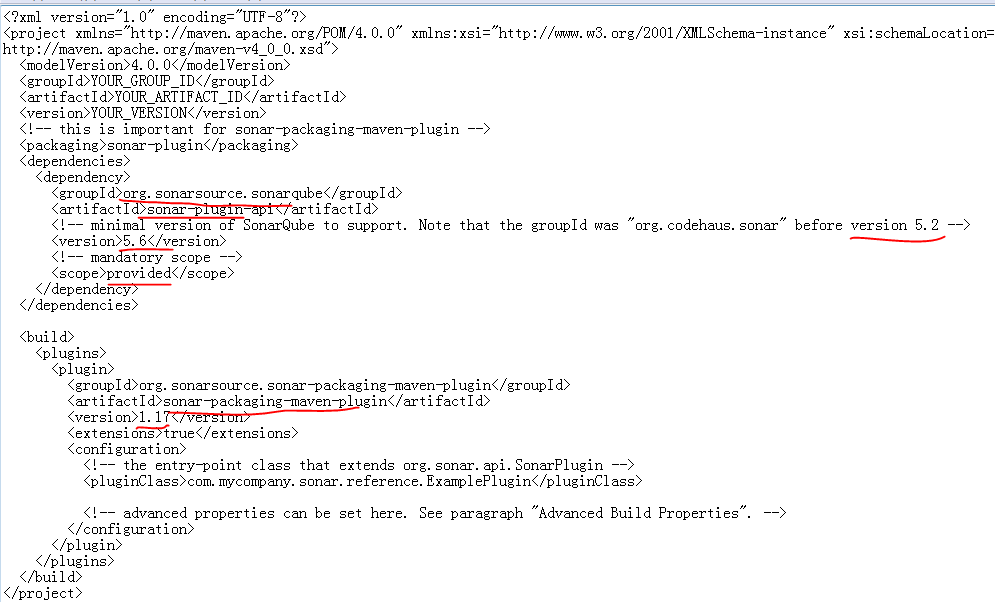
<https://docs.sonarqube.org/display/DEV/Adding+Coding+Rules>

<https://docs.sonarqube.org/display/DEV/Adding+Coding+Rules+using+Java>

SonarQube 插件写编码规则需要6步

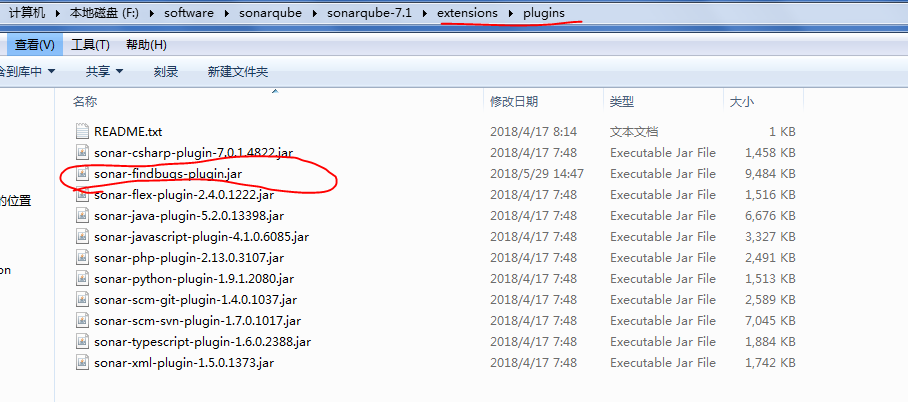
### 5.1.1Create a  [SonarQube plugin](https://docs.sonarqube.org/display/DEV/Developing+a+Plugin)

https://docs.sonarqube.org/display/DEV/Build+Plugin



官网下载一个sonar-findbugs-plugin maven 工程， 打包后放在

### 



将打包的插件拷贝至 $SONAR\_CUBE\_HOME/extensions/plugins/路径下，重新启动 sonarcube。

### 5.1.2 Put a dependency on the API of the language plugin for which you are writing coding rules.

### 5.1.3 Create as many custom rules as required

### 5.1.4 Generate the SonarQube plugin (jar file)

### 5.1.5 Place this jar file in the SONARQUBE\_HOME/extensions/plugins directory

### 5.1.6 Restart SonarQube server

## 5.2 Custom Rules for java(write your own rule)

白手起家定义一个sonar规则：

https://docs.sonarqube.org/display/PLUG/Writing+Custom+Java+Rules+101

从GitHub获取

https://github.com/SonarSource/sonar-custom-rules-examples/tree/master/java-custom-rules

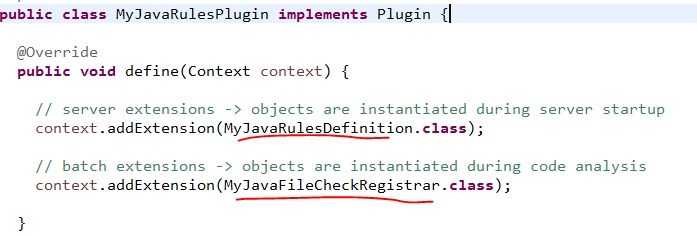
这个工程已经包含 custom rules ， 下面重新加入一个新的规则。

先分析pom.xml 文件：





查看<pluginClass>插件类 MyJavaRulesPlugin.java 源码:



https://github.com/SonarSource/sonar-custom-rules-examples/tree/master/java-custom-rules

待续…

如果您需要技术指导，请发邮件至作者的平安临时内部邮箱：EX\_WLJR\_JIANGPENG