**利用SonarQube实现代码静态扫描及问题总结**

*Code analyzing practice with SonarQube*

19 Jul 2016 in [Devops](https://yaowenjie.github.io/categories/index.html" \l "DevOps" \o ")  4 minutes read

[SonarQube(Sonar)](http://www.sonarqube.org/)是一个用于管理代码质量的开源平台。SonarQube目前已支持超过20种主流编程语言，它管理的代码质量主要涉及7个维度:架构与设计、重复、单元测试、复杂度、潜在的bug、代码标准、注释。

  本文，笔者将围绕搭建SonarQube这样的代码质量管理平台这个主题展开，结合java代码实例一步步讲述具体的过程，其中涉及Sonar的下载安装、创建对应Mysql数据库以及运行和管理，并对实践过程中出现的一些问题进行了分析和解决。

*注：本文中所有的实践都是在Ubuntu虚拟机（系统具体版本为Ubuntu 12.04 LTS）下进行，但目测同样适用于各个平台。*

**1. 安装SONAR**

  从[SonarQube官方网站](http://www.sonarqube.org/)下载对应的安装包<http://www.sonarqube.org/downloads/>，下载并解压至任意目录。

**2. 安装MYSQL，创建SONAR的用户和数据库**

  Sonar支持SQL Server、MySQL、Oracle、Postgres等多种数据库，这里我们以MySQL为例，如果你没有安装MySQL的话，可以使用诸如以下的指令（Ubuntu下）安装:

sudo apt-get install mysql-server

sudo mysql\_secure\_installation

  接着创建一个用于创建对应sonar用户和sonar的数据表的SQL文件，名为**create\_database.sql**:

**CREATE** **DATABASE** sonar CHARACTER **SET** utf8 **COLLATE** utf8\_general\_ci;

**CREATE** **USER** 'sonar' IDENTIFIED **BY** 'sonar';

**GRANT** **ALL** **ON** sonar.\* **TO** 'sonar'@'%' IDENTIFIED **BY** 'sonar';

**GRANT** **ALL** **ON** sonar.\* **TO** 'sonar'@'localhost' IDENTIFIED **BY** 'sonar';

FLUSH **PRIVILEGES**;

  运行：

mysql -u root -p < create\_database.sql

  正常情况下，你便在MySQL中创建sonar的用户和数据库。你可以使用sonar用户登录查看是否成功创建了一个名为sonar的数据库:

mysql -u sonar -p

2.1 创建数据库时出现错误（第二步成功请无视这一步）

  笔者在创建sonar数据库的时候由于“手残”等诸多原因，导致数据库创建失败/中断，这时候需要我们手动的Drop掉（注意一定是Drop掉，不能只是删除）对应的数据库和用户，并重新执行第二步创建数据库的操作:

mysql> **DROP** **DATABASE** sonar;

mysql> **DROP** **USER** 'sonar'@'%';

**3 修改SONAR配置并启动**

  再启动之前，需要修改第一步解压的安装包下**conf/sonar.properties**文件，去掉这两行前面的注释符号，可能需要填充具体的username和password（前文创建数据库时用到的username和password）：

sonar.jdbc.username=sonar

sonar.jdbc.password=sonar

  再去掉这行前面的注释符号：

sonar.jdbc.url=jdbc:mysql://localhost:3306/sonar?useUnicode=**true**&characterEncoding=utf8&rewriteBatchedStatements=**true**&useConfigs=maxPerformance

  然后运行如下指令启动Sonar：

{Your\_Sonar\_Path}/bin/linux-x86-64/sonar.sh start

  如果系统为32位的，你需要将上方路径改为“bin/linux-x86-32/sonar.sh”，否决启动将会失败。当然，把该路径加入环境变量也不失为一种方便的举措。  
  启动成功后，在浏览器中访问：**http://localhost:9000**，你将看到类似这样的SonarQube的Home页面（首次Project应该是空的）：

3.1 SONAR启动后异常停止

  笔者在正常启动Sonar后，遇到过两种异常停止的情况，由于控制台看不到具体的log信息，可以在sonar的解压包路径下的**logs/sonar.log**里寻找到具体信息。

**SONAR与MYSQL版本不匹配**

  这种情况下，可以在log里面看到类似如下这样的内容:

2016.05.18 15:17:37 INFO web[o.a.c.h.Http11NioProtocol] Starting ProtocolHandler ["http-nio-0.0.0.0-9000"]

2016.05.18 15:17:37 INFO web[o.s.s.a.TomcatAccessLog] Web server is started

2016.05.18 15:17:37 INFO web[o.s.s.a.EmbeddedTomcat] HTTP connector enabled on port 9000

2016.05.18 15:17:37 WARN web[o.s.p.ProcessEntryPoint] Fail to start web

java.lang.IllegalStateException: Webapp did not start

at org.sonar.server.app.EmbeddedTomcat.isUp(EmbeddedTomcat.java:84) ~[sonar-server-5.5.jar:na]

at org.sonar.server.app.WebServer.isUp(WebServer.java:48) [sonar-server-5.5.jar:na]

at org.sonar.process.ProcessEntryPoint.launch(ProcessEntryPoint.java:105) ~[sonar-process-5.5.jar:na]

at org.sonar.server.app.WebServer.main(WebServer.java:69) [sonar-server-5.5.jar:na]

  这里没有明显的错误，但是[Google之](http://stackoverflow.com/questions/37296804/java-lang-illegalstateexception-webapp-did-not-start-at-sonarqube)才发现与版本有关，笔者一开始使用的SonarQube 5.6并不支持MySQL 5.5。所以需要将SonarQube降到5.4，当然也可以升级MySQL，笔者选择了前者。

**虚拟机内存不够**

2016.07.18 22:58:26 ERROR web[o.a.c.c.StandardContext] One or more listeners failed to start. Full details will be found in the appropriate container log file

2016.07.18 22:58:26 ERROR web[o.a.c.c.StandardContext] Context [] startup failed due to previous errors

2016.07.18 22:58:26 WARN web[o.a.c.l.WebappClassLoaderBase] The web application [ROOT] appears to have started a thread named [Abandoned connection cleanup thread] but has failed to stop it. This is very likely to create a memory leak. Stack trace of thread:

java.lang.Object.wait(Native Method)

java.lang.ref.ReferenceQueue.remove(ReferenceQueue.java:143)

com.mysql.jdbc.AbandonedConnectionCleanupThread.run(AbandonedConnectionCleanupThread.java:43)

2016.07.18 22:58:26 WARN web[o.a.c.l.WebappClassLoaderBase] The web application [ROOT] appears to have started a thread named [Timer-0] but has failed to stop it. This is very likely to create a memory leak. Stack trace of thread:

java.lang.Object.wait(Native Method)

java.util.TimerThread.mainLoop(Timer.java:552)

java.util.TimerThread.run(Timer.java:505)

  如果出现这样的log信息，那是因为SonarQube运行需要的内存不够的原因，缺啥补啥，笔者便将使用的虚拟机运存从512MB增加到1024MB，问题便消失了。

**4. 使用SONARQUBE-SCANNER扫描分析具体代码**

  Sonar正常运行后，就需要添加/扫描/分析具体的代码了，SonarQube提供了支持多种工具的扫描器(SonarQube Scanner)，其中包括针对MSBuild、Ant、Maven、Gradle这样构建工具以及Jenkins这样CI工具的插件支持之外，还有一个可以直接运行的独立Scanner。这里就以一个[简单的基于Gradle构建的Java项目](https://github.com/Yaowenjie/Cucumber-Demo)为例，通过添加对应的gradle插件，实现对该项目代码的代码分析。

  首先，从github上clone/下载这个工程：<https://github.com/Yaowenjie/Cucumber-Demo>，然后在**build.gradle**中添加sonarqube插件（这种方式要求gradle的版本为2.1+）：

plugins {

id "org.sonarqube" version "2.0.1"

}

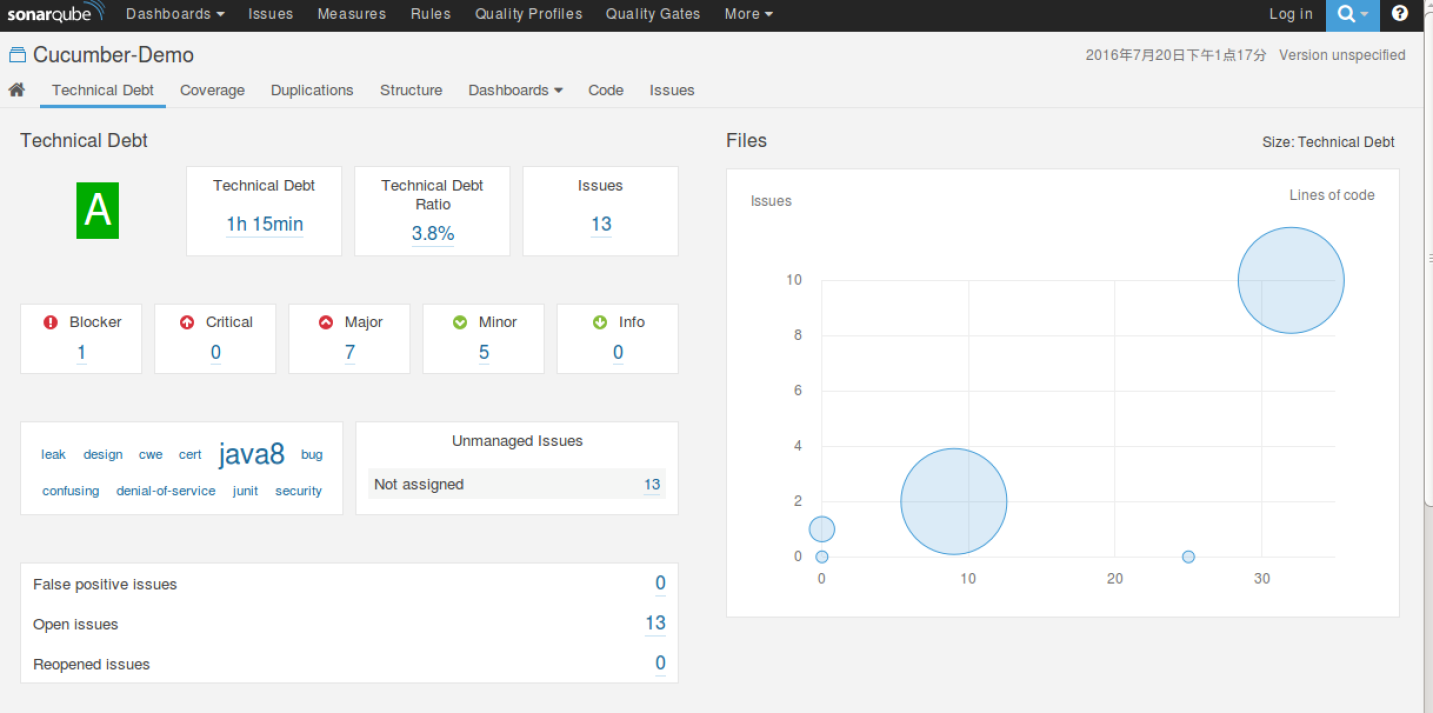
  接着，运行如下：

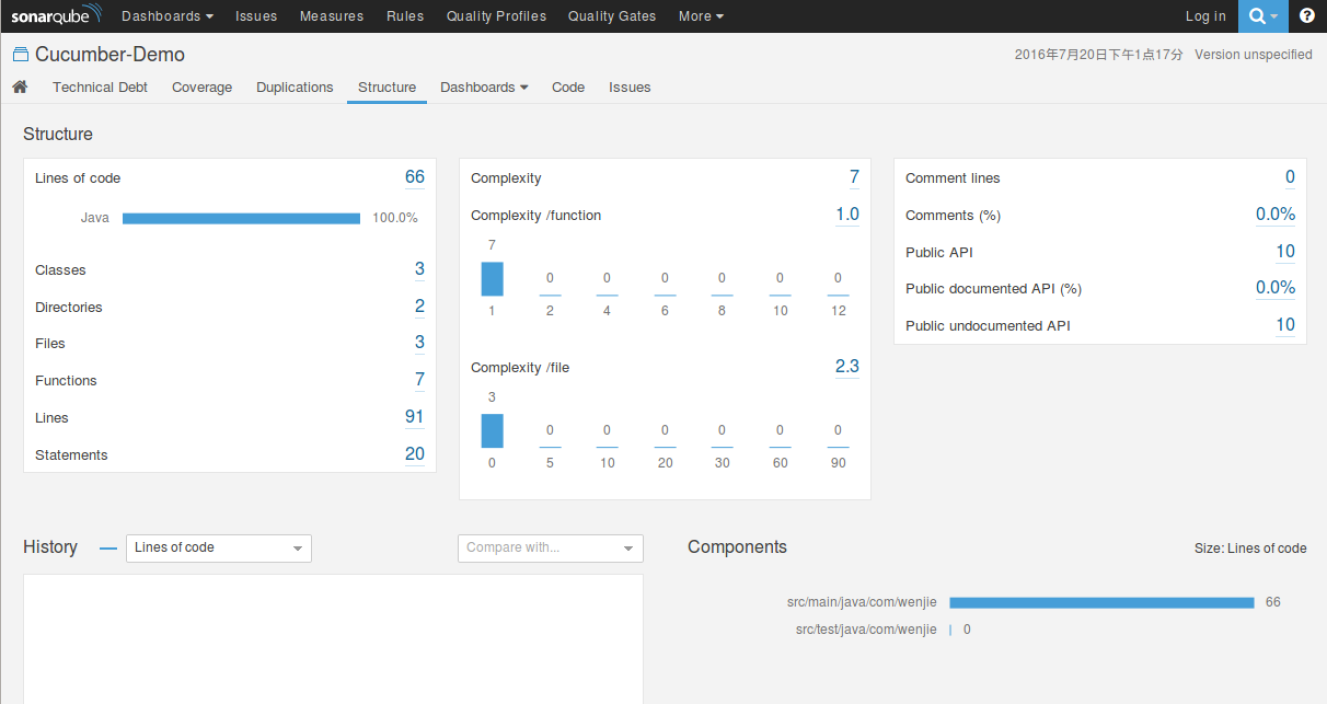
./gradlew sonar

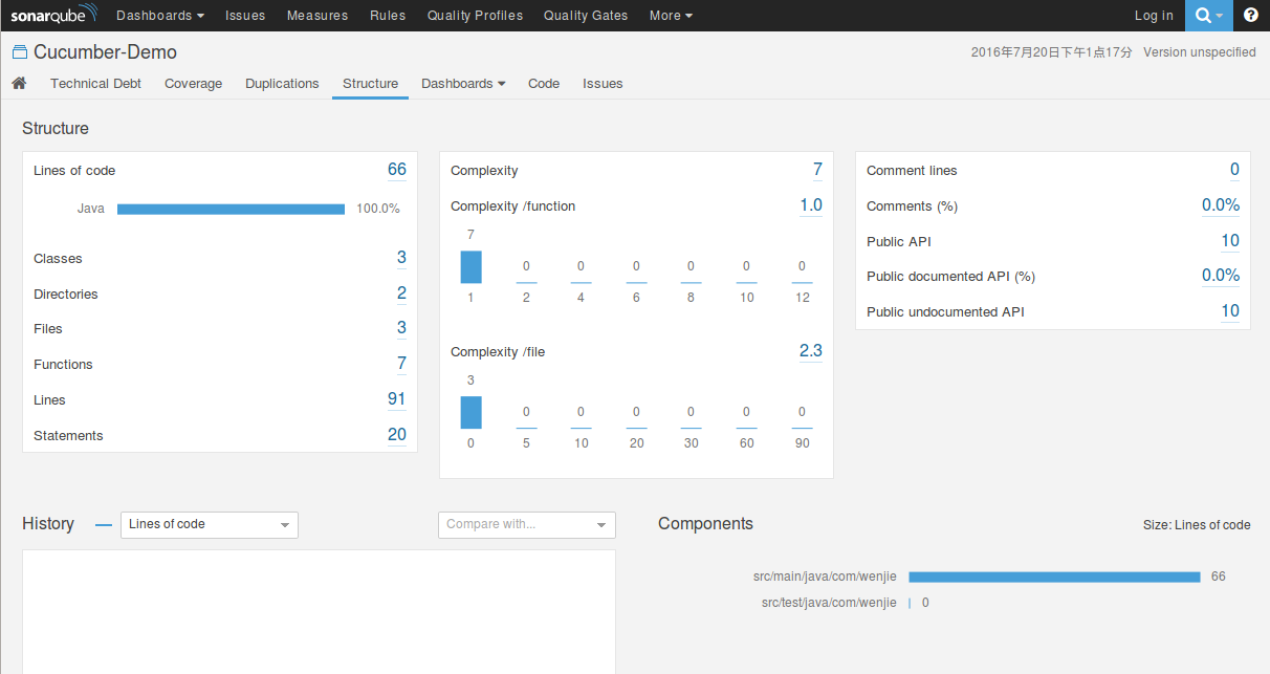
  如果你运行test的时候报错了的话，请在build.gradle内的test里排除掉BaseFlowTest：

exclude '\*\*/BaseFlowTest\*'

  成功执行后，在浏览器中访问**http://localhost:9000**，会发现新增了一个名为Cucumber-Demo的Project，点击进入可以看到详细的代码分析数据和图表。



技术债细节： 

项目结构信息：

至此，整个过程便完成了。当然，Sonar可以展示和管理的内容远远不止这些，这里只是一个简单但暂且还算全面的Demo，更多内容请访问[SonarQube官网](http://www.sonarqube.org/)。