简介

SonarQube(原名Sonar)是一个开源的代码质量管理系统,可以扫描项目中重复代码、编码标准、单元测试、代码覆盖率、代码复杂度、潜在Bug、注释等.

支持Java、C/C++、C#、PHP、Flex、Groovy、JavaScript、Python等多种语言.

代码扫码的好处在于通过配置规则扫描代码设计缺陷和代码优化从而提高代码的质量.

本文介绍使用docker技术搭建SonarQube环境并扫描代码.

环境搭建

快速启动

这种方式启动sonarqube是没有数据存储的,docker重新加载后数据不会reload数据.

```
docker run -d --name sonarqube -p 9000:9000 -p 9092:9092
sonarqube:lts

浏览器打开:0.0.0.0:9000
帐号密码:admin、admin
```

使用docker logs查看是否启动成功

```
docker logs --tail=100
6a2bac9532c247fb85ff1b292709adff5d9cd107a1359ddd8344d63612b89ae7
```

插件挂载

sonar-l10n-zh-plugin-1.16.jar
backelite-sonar-objective-c-plugin-0.6.3.jar

把jar包放到/Users/xinxi/Desktop/SonarQube/extensions/plugins目录下.

```
docker run -d --name sonarqube -p 9000:9000 -p 9092:9092
-v
/Users/xinxi/Desktop/SonarQube/extensions:/opt/sonarqube/extensions
sonarqube:lts
```

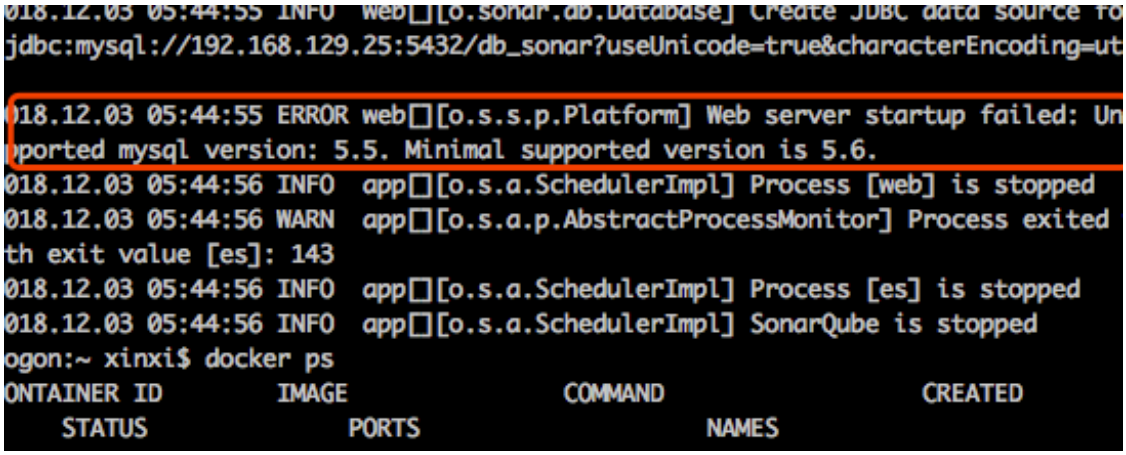
全配置启动

docker启动mysql

使用mysql作为数据存储,通过如下命令启动:

```
docker run --name mysql-5.6 -v /Users/xinxi/mysql:/var/lib/mysql -e
MYSQL_ROOT_PASSWORD=123321
-p 3306:3306 -d mysql:5.6
```

sonarqube不支持mysql5.5版本,最低支持5.6版本



docker启动sonarqube

需要注意两点:

- db_sonar数据库需要手动先创建,否则会报找不到
- 本地搭建ip地址不能写localhost,需要写内网ip

启动命令如下:

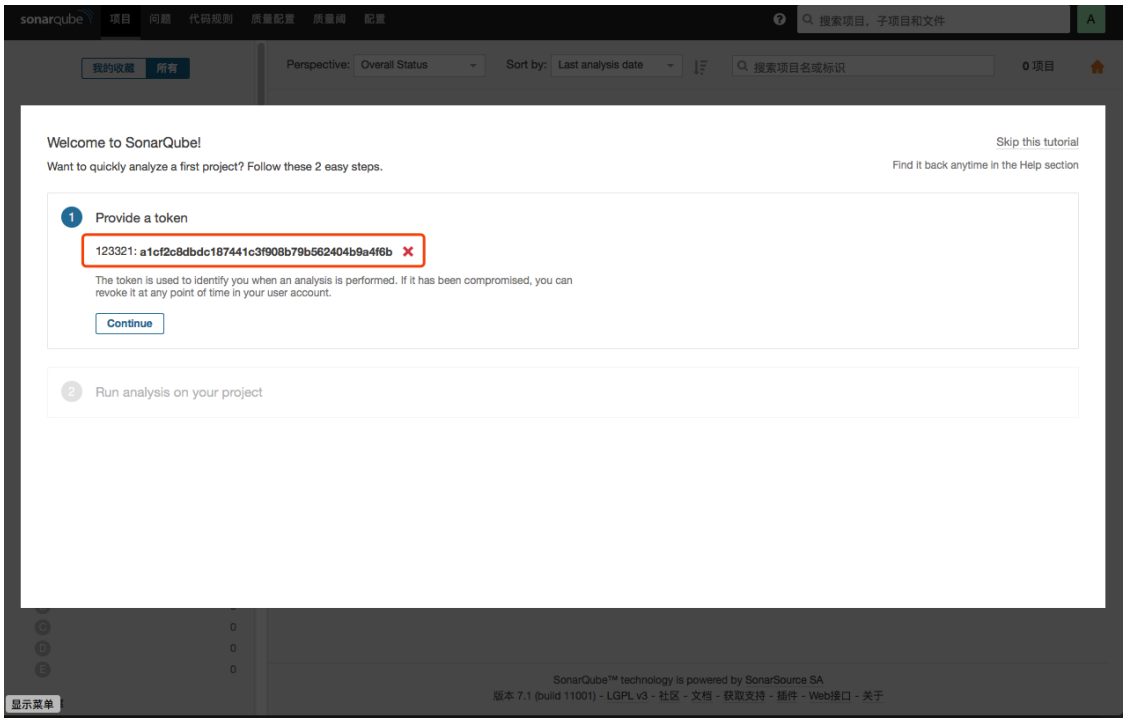
```
docker run -d --name Sonarqube -p 8185:9000 -p 8186:9092
-e "SONARQUBE_JDBC_USERNAME=root"
-e "SONARQUBE_JDBC_PASSWORD=123321"
-e "SONARQUBE_JDBC_URL=jdbc:mysql://192.168.1.109:3306
/db_sonar?useUnicode=true&characterEncoding=utf8"
-d sonarqube:lts
```

docker-compose启动

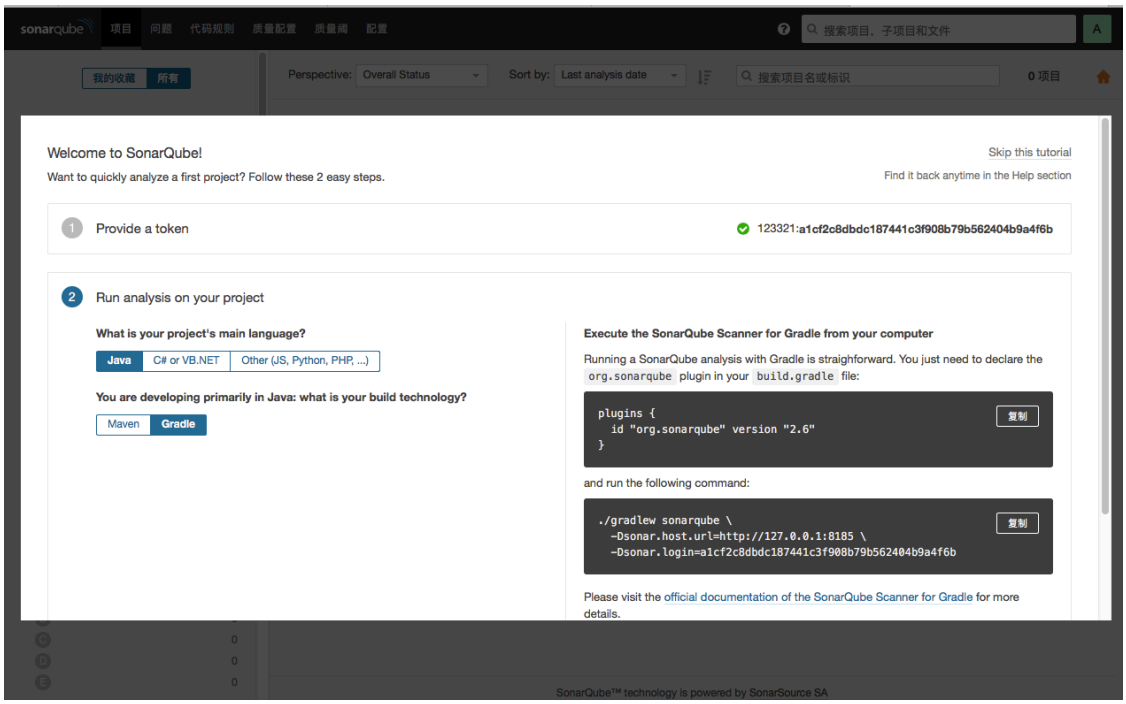
请参考:https://github.com/xinxi1990/SonarQube_Docker

效果

启动成功以后,第一次页登录成功后会设置token,这个token是以后被扫描代码工程中需要配置的,用于通过token的方式连接Sonarqube平台传数据结果



Android项目配置



项目地址:https://github.com/xinxi1990/FakeBiliBili

Android项目是使用gradle管理插件,所以需要配置gradle文件.
在项目根目录的gradle配置如下:

```
buildscript {
  repositories {
    maven {
      url "https://plugins.gradle.org/m2/"
    }
  }
  dependencies {
    classpath "org.sonarsource.scanner.gradle:sonarqube-gradle-
plugin:2.5"
  }
}

plugins {
  # 添加插件信息
  id "org.sonarqube" version "2.6-rc1"
}

apply plugin: "org.sonarqube"
```

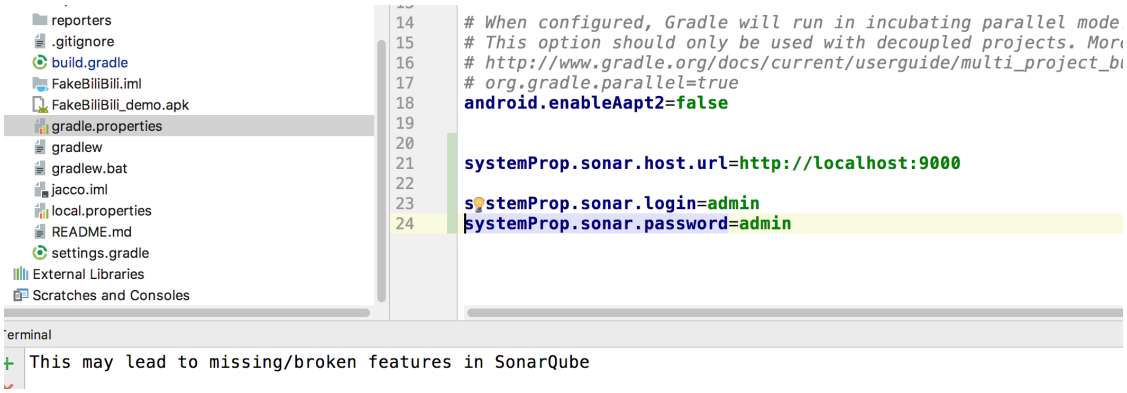
参数介绍:

```
Dsonar.host.url:sonarqube的ip+端口
Dsonar.login:sonarqube的token
```

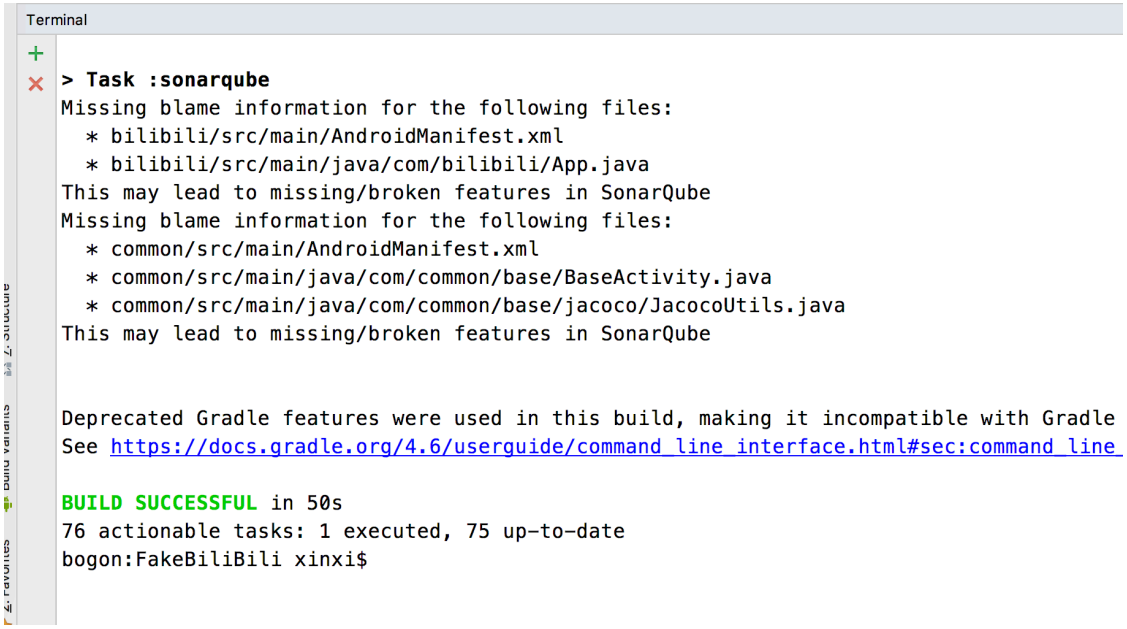
执行命令:

```
gradle sonarqube \
  -Dsonar.host.url=http://localhost:8186 \
  -Dsonar.login=a1cf2c8dbdc187441c3f908b79b562404b9a4f6b
```

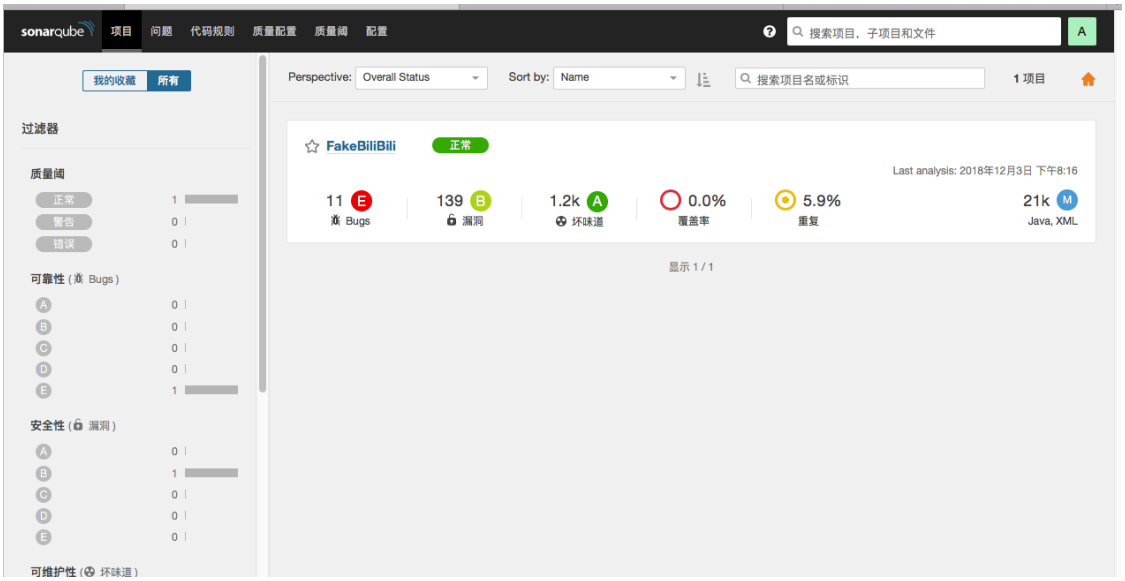
另外配置方式是在gradle.properties配置文件

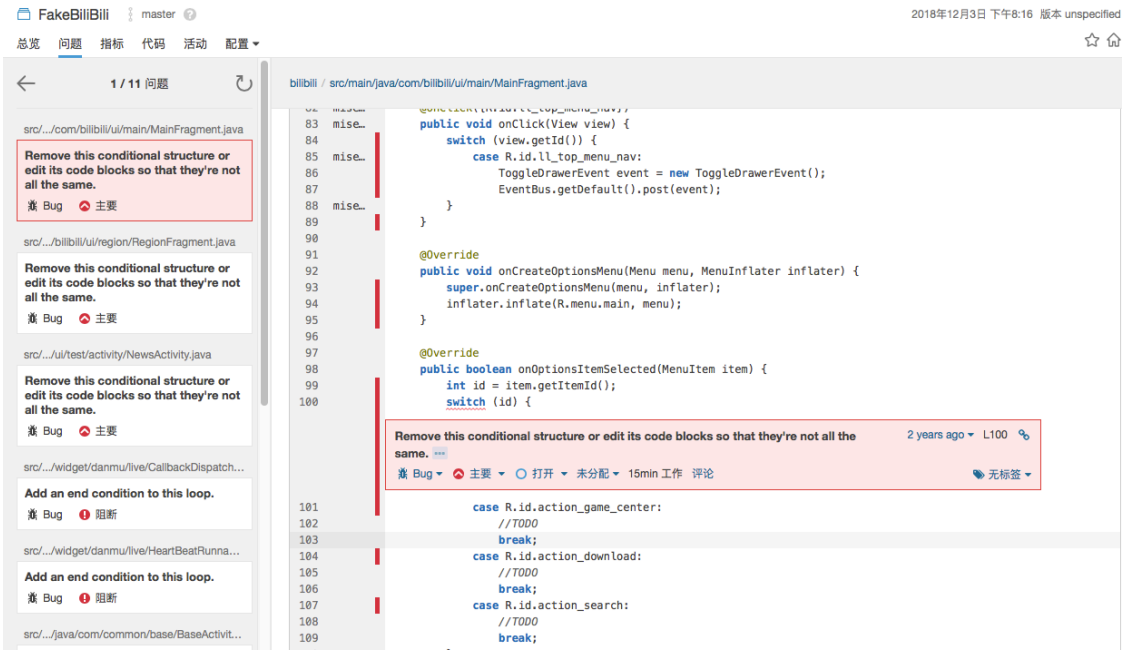

reporters
.gitignore
build.gradle
FakeBiliBili.iml
FakeBiliBili_demo.apk
gradle.properties
gradlew
gradlew.bat
jacco.iml
local.properties
README.md
settings.gradle
External Libraries
Scratches and Consoles
# When configured, Gradle will run in incubating parallel mode
# This option should only be used with decoupled projects. Mor
# http://www.gradle.org/docs/current/userguide/multi_project_b
# org.gradle.parallel=true
android.enableAapt2=false
systemProp.sonar.host.url=http://localhost:9000
systemProp.sonar.login=admin
systemProp.sonar.password=admin
Terminal
This may lead to missing/broken features in SonarQube

上传结果成功


Terminal
> Task :sonarqube
Missing blame information for the following files:
* bilibili/src/main/AndroidManifest.xml
* bilibili/src/main/java/com/bilibili/App.java
This may lead to missing/broken features in SonarQube
Missing blame information for the following files:
* common/src/main/AndroidManifest.xml
* common/src/main/java/com/common/base/BaseActivity.java
* common/src/main/java/com/common/base/jacoco/JacocoUtils.java
This may lead to missing/broken features in SonarQube
Deprecated Gradle features were used in this build, making it incompatible with Gradle
See https://docs.gradle.org/4.6/userguide/command_line_interface.html#sec:command_line
BUILD SUCCESSFUL in 50s
76 actionable tasks: 1 executed, 75 up-to-date
bogon:FakeBiliBili xinxi$

扫描结果





mvn项目配置

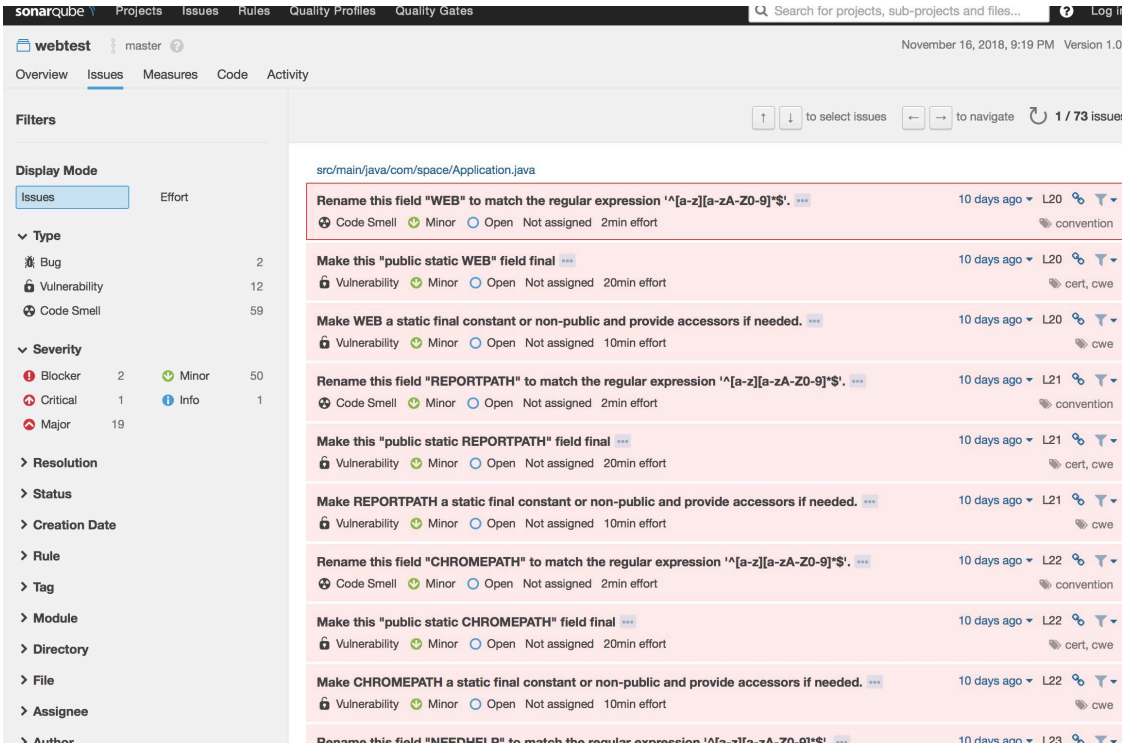
项目地址:https://github.com/xinxi1990/AppiumDemo

运行命令

```
mvn sonar:sonar   -Dsonar.host.url=http://localhost:9000   -
Dsonar.login=c8ce928f1497f1fa5591cdcf5357aa4e44920796
```

扫描结果

```
[INFO] 14/16 files analyzed
[WARNING] Missing blame information for the following files:
[WARNING]   * pom.xml
[WARNING]   * src/test/java/AllureTest/Demo.java
[WARNING] This may lead to missing/broken features in SonarQube
[INFO] 2 files had no CPD blocks
[INFO] Calculating CPD for 12 files
[INFO] CPD calculation finished
[INFO] Analysis report generated in 197ms, dir size=156 KB
[INFO] Analysis reports compressed in 118ms, zip size=73 KB
[INFO] Analysis report uploaded in 387ms
[INFO] ANALYSIS SUCCESSFUL, you can browse http://localhost:9000/dashboard/index/appiumdemo:appiumdemo
[INFO] Note that you will be able to access the updated dashboard once the server has processed the submitted analy
 report
[INFO] More about the report processing at http://localhost:9000/api/ce/task?id=AWeyKFbWGvIog0jZGQOq
[INFO] Task total time: 12.655 s
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
```



iOS项目配置

安装

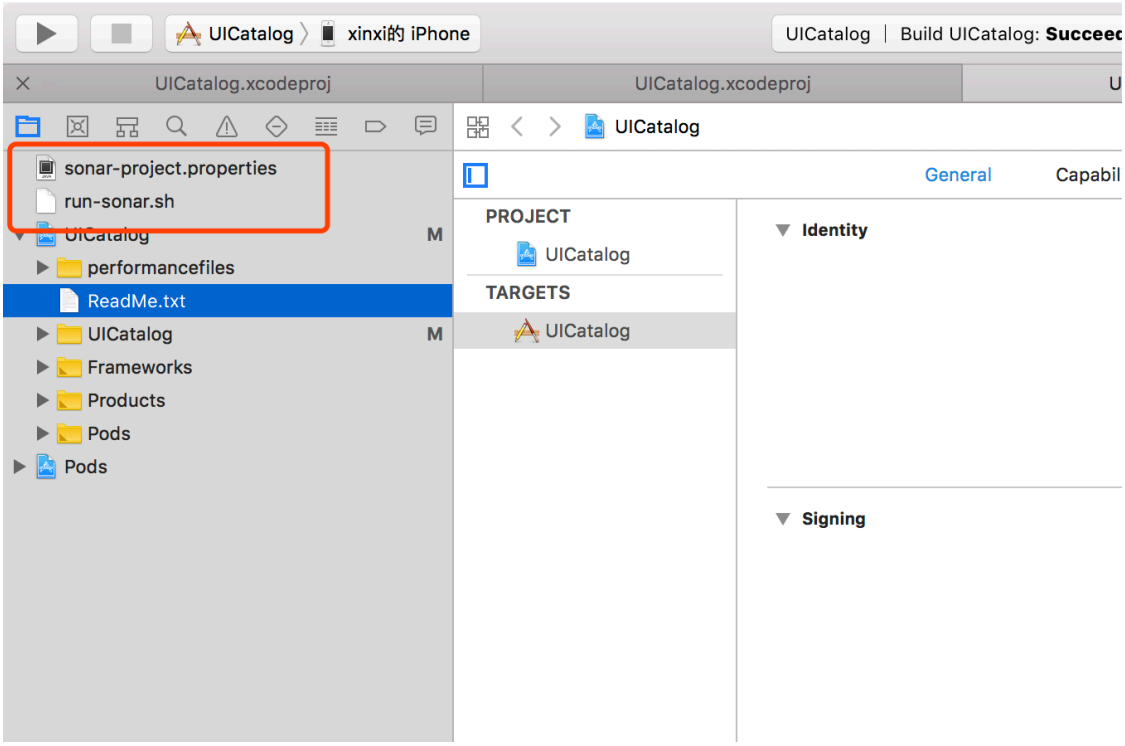
先需要安装如下工具

```
brew tap oclint/formulae
brew install oclint
brew install sonar-scanner
brew install gcovr
```

项目配置

需要在项目根目录增加run-sonar.sh和sonar-project.properties两个文件

具体配置请参考Demo代码

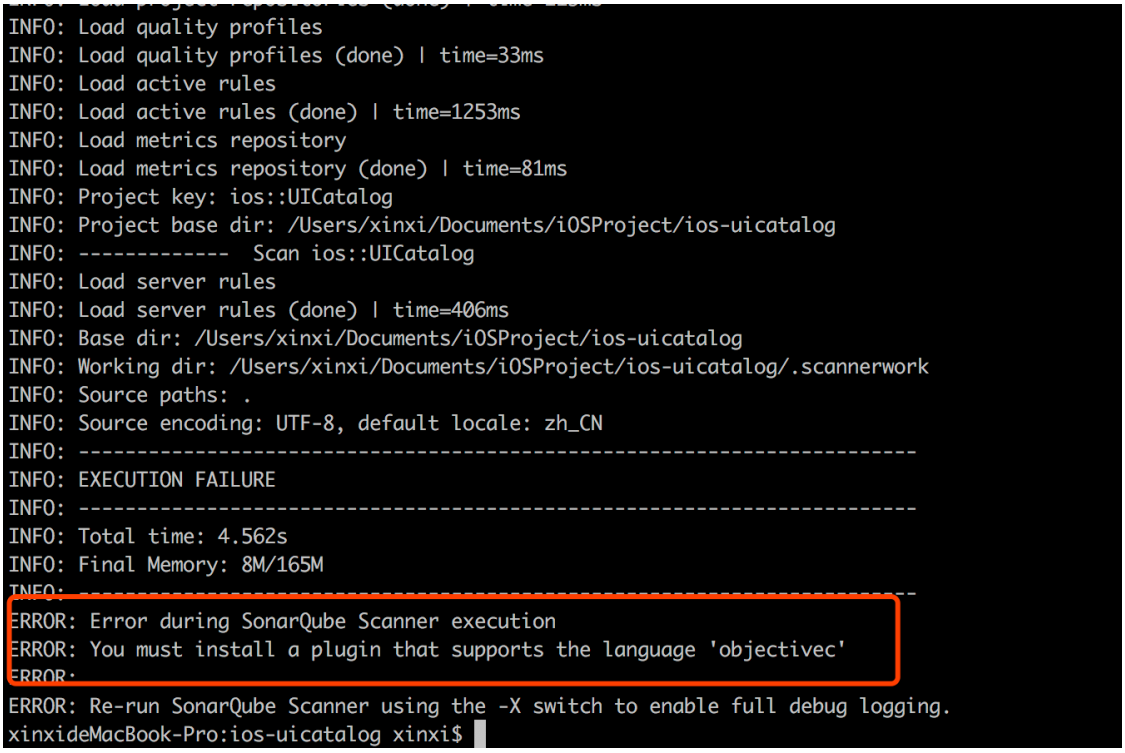


运行命令

在项目根目录下,sh run-sonar.sh

踩坑

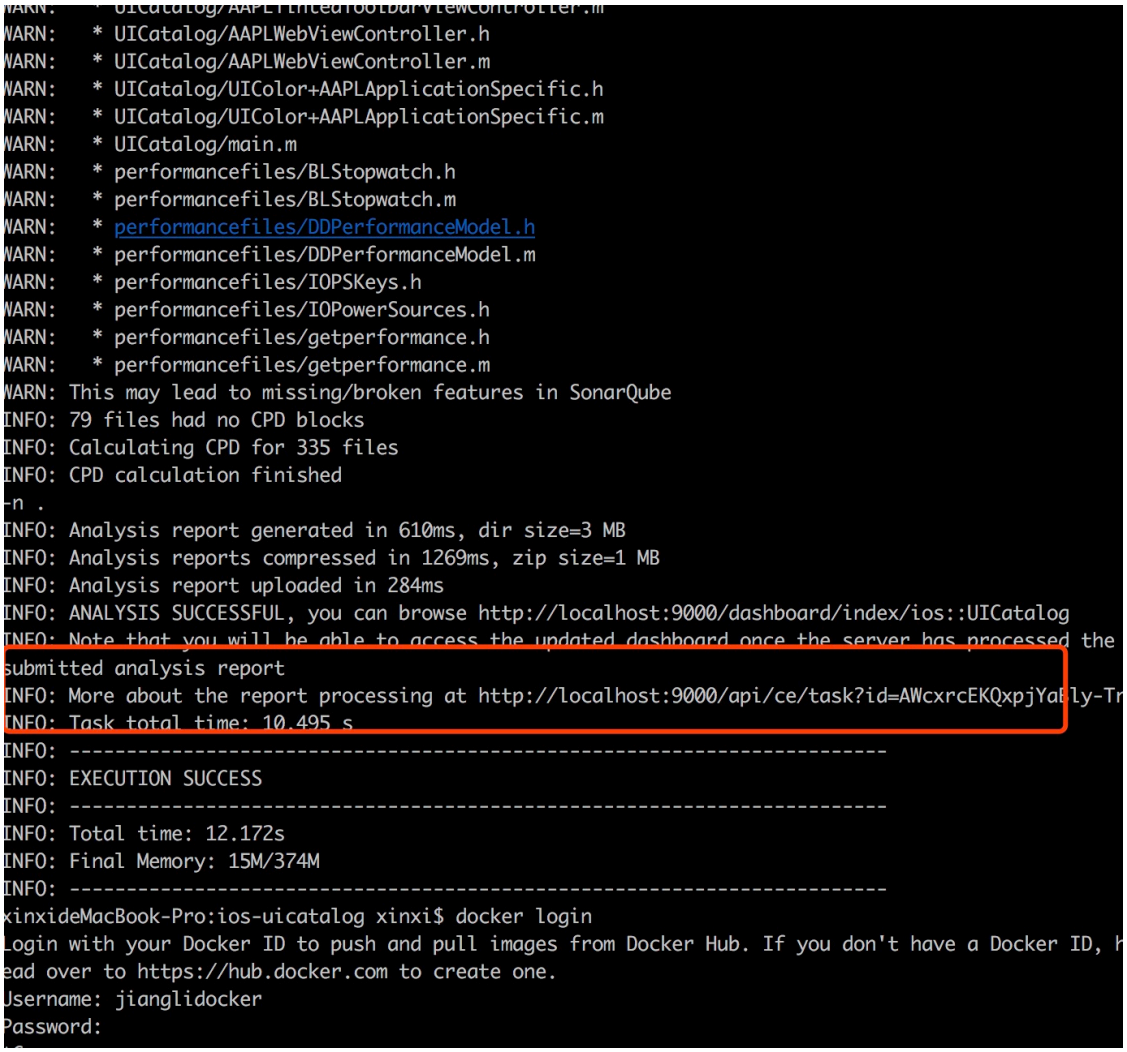
提示没有oc插件

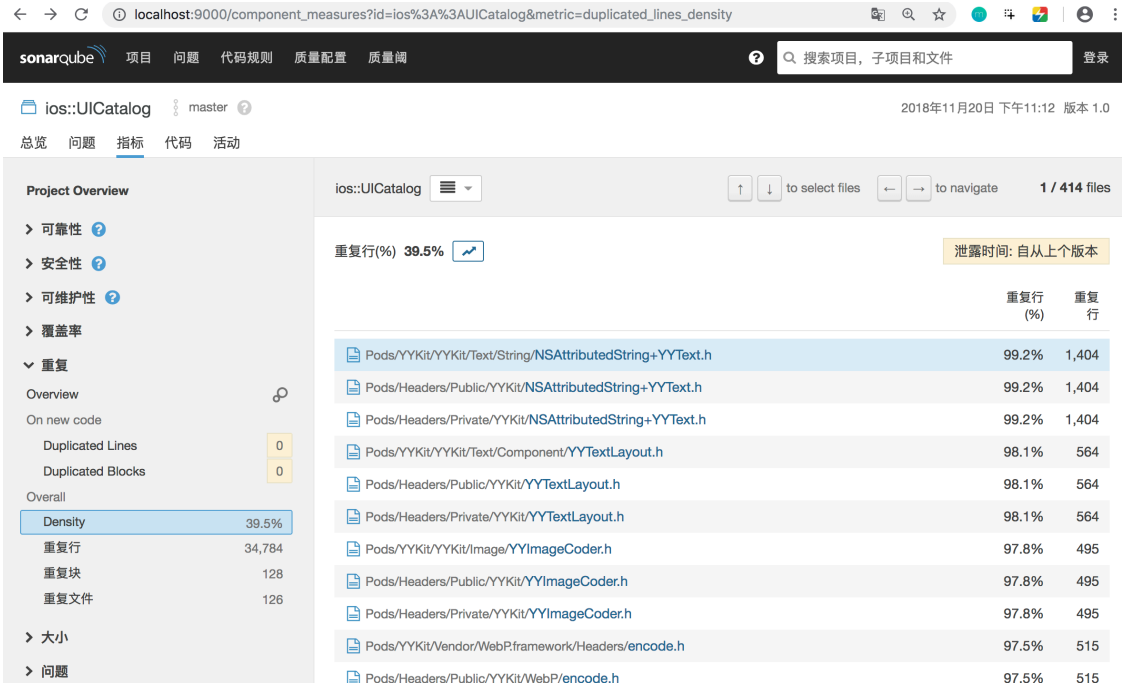


通过sonarqube安装的oc插件是收费的,下载免费的oc插件.
https://github.com/Backelite/sonar-objective-c

然后把backelite-sonar-objective-c-plugin-0.6.3.jar放到sonarqube的extensions/plugins中,然后重启镜像.

效果展示


WARN: * UICatalog/AAPLTintedToolbarViewController.m
WARN: * UICatalog/AAPLWebViewController.h
WARN: * UICatalog/AAPLWebViewController.m
WARN: * UICatalog/UIColor+AAPLApplicationSpecific.h
WARN: * UICatalog/UIColor+AAPLApplicationSpecific.m
WARN: * UICatalog/main.m
WARN: * performancefiles/BLStopwatch.h
WARN: * performancefiles/BLStopwatch.m
WARN: * performancefiles/DDPerformanceModel.h
WARN: * performancefiles/DDPerformanceModel.m
WARN: * performancefiles/IOPSKeys.h
WARN: * performancefiles/IOPowerSources.h
WARN: * performancefiles/getperformance.h
WARN: * performancefiles/getperformance.m
WARN: This may lead to missing/broken features in SonarQube
INFO: 79 files had no CPD blocks
INFO: Calculating CPD for 335 files
INFO: CPD calculation finished
-n .
INFO: Analysis report generated in 610ms, dir size=3 MB
INFO: Analysis reports compressed in 1269ms, zip size=1 MB
INFO: Analysis report uploaded in 284ms
INFO: ANALYSIS SUCCESSFUL, you can browse http://localhost:9000/dashboard/index/ios::UICatalog
INFO: Note that you will be able to access the updated dashboard once the server has processed the
submitted analysis report
INFO: More about the report processing at http://localhost:9000/api/ce/task?id=AWcxrcEKQxpjYa
INFO: Task total time: 10.495 s
INFO: --
INFO: EXECUTION SUCCESS
INFO: --
INFO: Total time: 12.172s
INFO: Final Memory: 15M/374M
INFO: --
xinxideMacBook-Pro:ios-uicatalog xinxi$ docker login
Login with your Docker ID to push and pull images from Docker Hub. If you don't have a Docker ID, h
ead over to https://hub.docker.com to create one.
Username: jianglidocker
Password:



制作镜像

sonarqube原生不支持中文、并且没有oc插件.可以把汉化中文包和oc插件打包成新的docker镜像

把sonar-l10n-zh-plugin-1.16.jar、backelite-sonar-objective-c-plugin-0.6.3.jar和dockerfile放到一个目录下

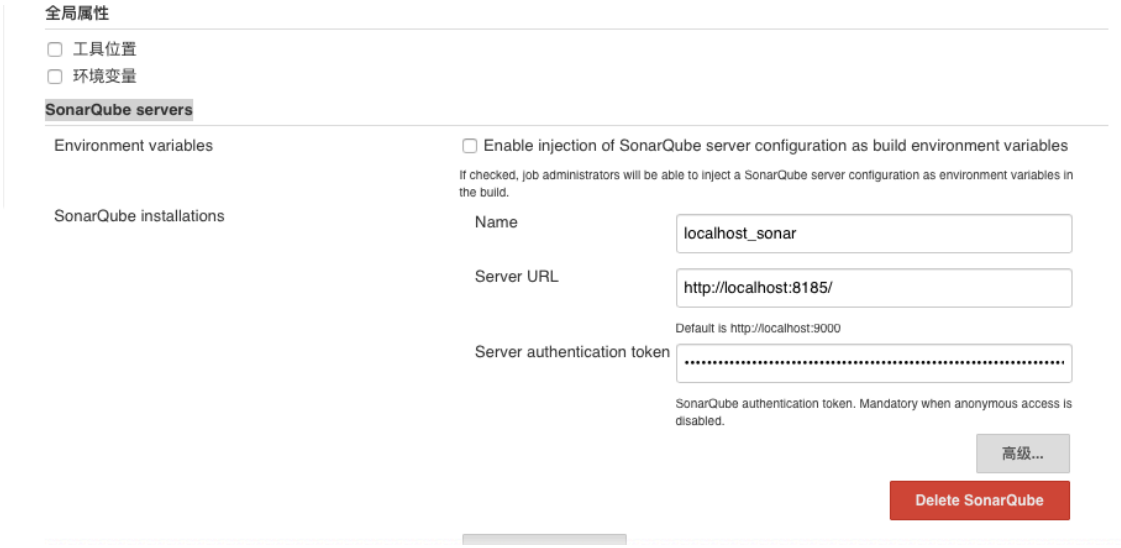
dockerfile如下:

```
dockfile:
FROM sonarqube
ADD sonar-l10n-zh-plugin-1.16.jar
/opt/sonarqube/extensions/plugins/
ADD backelite-sonar-objective-c-plugin-0.6.3.jar
/opt/sonarqube/extensions/plugins/
```

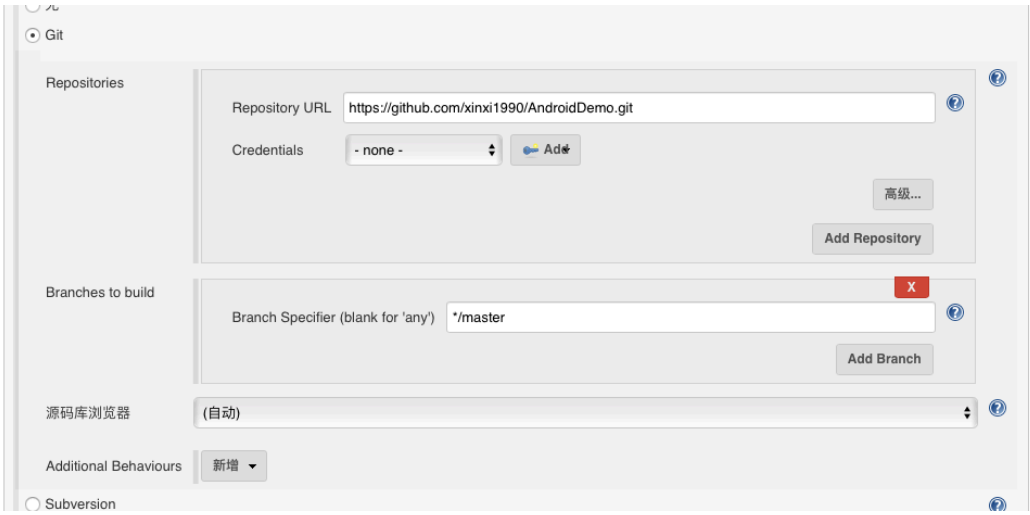
执行:docker build -t sonarqube:zh .

jenkins持续集成

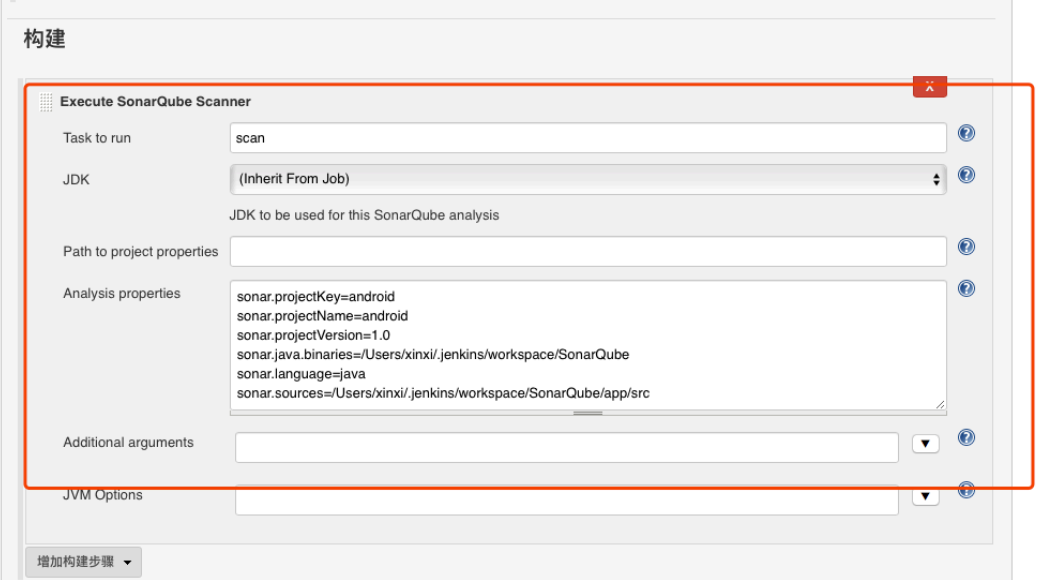
使用Android项目为例,首先需要在Jenkins中安装sonarqube插件,然后在系统设置中配置SonarQube的servers地址



新建job并配置git地址



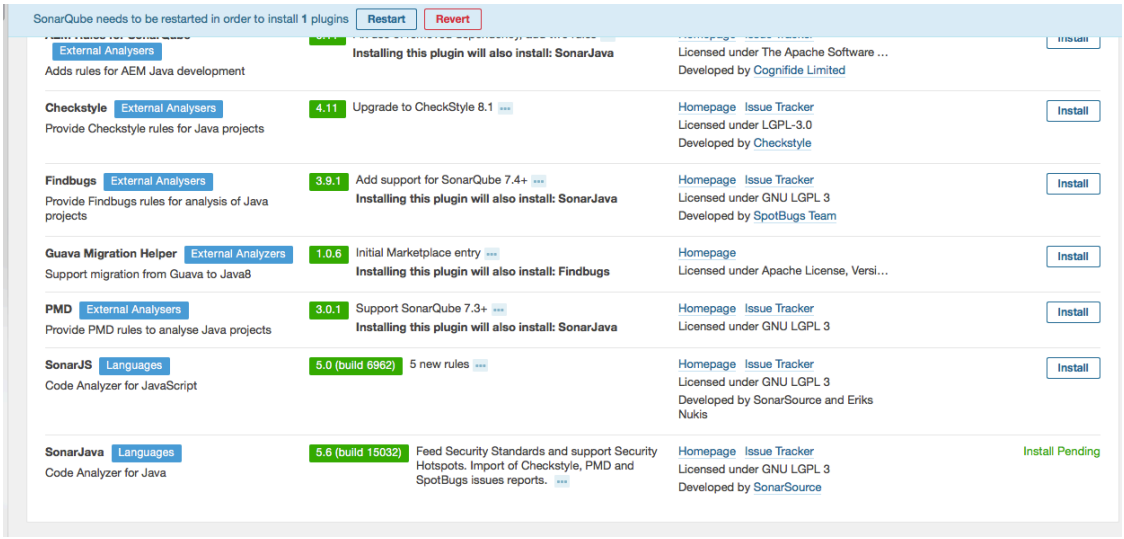
在构建处选择Execute SonarQube Scanner中配置如下



提示需要java插件

```
INFO: Process project properties
INFO: Load project repositories
INFO: Load project repositories (done) | time=147ms
INFO: Load quality profiles
INFO: Load quality profiles (done) | time=72ms
INFO: Load active rules
INFO: Load active rules (done) | time=351ms
INFO: Load metrics repository
INFO: Load metrics repository (done) | time=32ms
WARN: SCM provider autodetection failed. No SCM provider claims to support this project. Please use
sonar.scm.provider to define SCM of your project.
INFO: Project key: android
INFO: Project base dir: /Users/xinxi/.jenkins/workspace/SonarQube
INFO: ------------  Scan android
INFO: Load server rules
INFO: Load server rules (done) | time=49ms
INFO: Base dir: /Users/xinxi/.jenkins/workspace/SonarQube
INFO: Working dir: /Users/xinxi/.jenkins/workspace/SonarQube/.scannerwork
INFO: Source paths: app/src
INFO: Source encoding: UTF-8, default locale: zh_CN
INFO: ------------------------------------------------------------------------
INFO: EXECUTION FAILURE
INFO: ------------------------------------------------------------------------
INFO: Total time: 2.888s
INFO: Final Memory: 11M/169M
INFO: ------------------------------------------------------------------------
ERROR: Error during SonarQube Scanner execution
ERROR: You must install a plugin that supports the language 'java'
ERROR:
ERROR: Re-run SonarQube Scanner using the -X switch to enable full debug logging.
WARN: Unable to locate 'report-task.txt' in the workspace. Did the SonarScanner succedeed?
ERROR: SonarQube scanner exited with non-zero code: 1
Notifying upstream projects of job completion
Finished: FAILURE
```

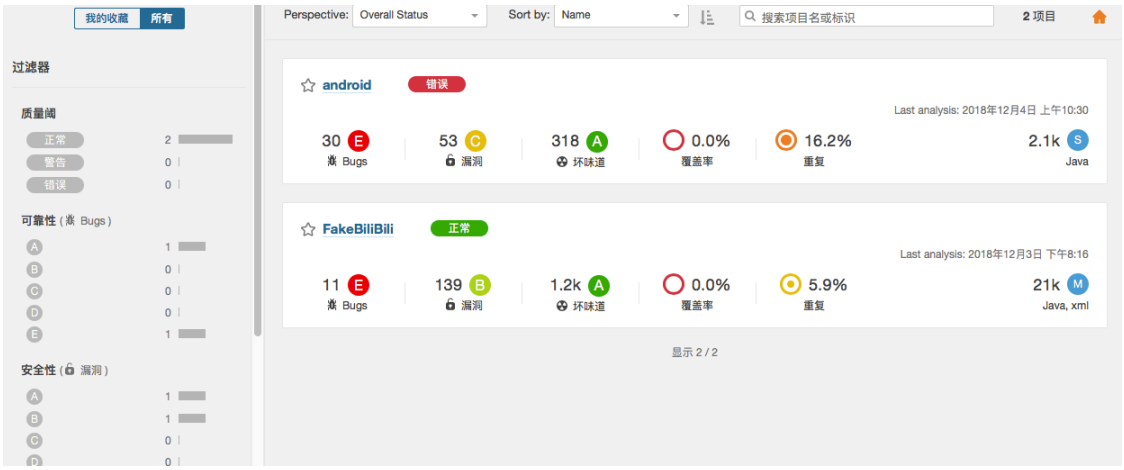
在SonarQube中安装java插件并且重启



扫描代码完成并且上传扫码结果

```
INFO: Sensor SurefireSensor [java] (done) | time=2ms
INFO: Sensor JaCoCoSensor [java]
INFO: Sensor JaCoCoSensor [java] (done) | time=0ms
INFO: Sensor SonarJavaXmlFileSensor [java]
INFO: Sensor SonarJavaXmlFileSensor [java] (done) | time=1ms
INFO: Sensor Zero Coverage Sensor
INFO: Sensor Zero Coverage Sensor (done) | time=55ms
INFO: Sensor CPD Block Indexer
INFO: Sensor CPD Block Indexer (done) | time=100ms
INFO: No SCM system was detected. You can use the 'sonar.scm.provider' property to explicitly specify it.
INFO: 9 files had no CPD blocks
INFO: Calculating CPD for 23 files
INFO: CPD calculation finished
INFO: Analysis report generated in 165ms, dir size=287 KB
INFO: Analysis reports compressed in 209ms, zip size=137 KB
INFO: Analysis report uploaded in 79ms
INFO: ANALYSIS SUCCESSFUL, you can browse http://localhost:8185/dashboard/index/android
INFO: Note that you will be able to access the updated dashboard once the server has processed the submitted
analysis report
INFO: More about the report processing at http://localhost:8185/api/ce/task?id=AWd3lyWuwi3YDBgzxplL
INFO: Task total time: 9.652 s
INFO: ------------------------------------------------------------------------
INFO: EXECUTION SUCCESS
INFO: ------------------------------------------------------------------------
INFO: Total time: 11.305s
INFO: Final Memory: 17M/543M
INFO: ------------------------------------------------------------------------
Notifying upstream projects of job completion
Finished: SUCCESS
```

SonarQube查看扫描结果



规则

Bugs

坏味道

检查验证标识符名称中的缩写（连续大写字母）长度，还允许执行骆驼案例命名

漏洞

覆盖率

重复

结语

SonarQube是一款优秀代码扫描工具,可以通过静态扫码代码的方式发现编码问题,代码扫描是一种低成本高收益的方式,在持续集成中是必不可少的环节.

学习帖

SonarQube的安装与使用
https://blog.imyxiao.com/docker/sonarqube.html

SonarQube 中文插件安装
https://www.jianshu.com/p/6cc4632628b1

使用Jenkins进行Android自动打包及SonarQube代码自动检测
http://blog.51cto.com/536410/2052972

oclint官方
http://oclint.org

IOS-Sonar代码质量监控
https://blog.csdn.net/helloworld_junyang/article/details/53836001

基于Sonar的iOS代码质量检测系统
https://blog.csdn.net/hualusiyu/article/details/79349025

iOS Sonar集成流程详解
https://www.jianshu.com/p/74bee59fef1c

Docker构建SonarQube检测代码质量平台
https://blog.csdn.net/OneZhous/article/details/80527953

SonarQube + Jenkins Pipeline配置
https://blog.csdn.net/liuxinghao/article/details/77967158

https://www.cnblogs.com/zishi/p/6766994.html
Sonar项目主要指标以及代码坏味道详解

docker-compose教程（安装，使用, 快速入门）
https://blog.csdn.net/pushiqiang/article/details/78682323