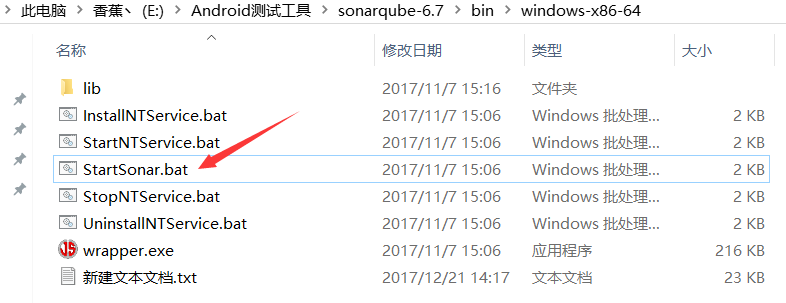
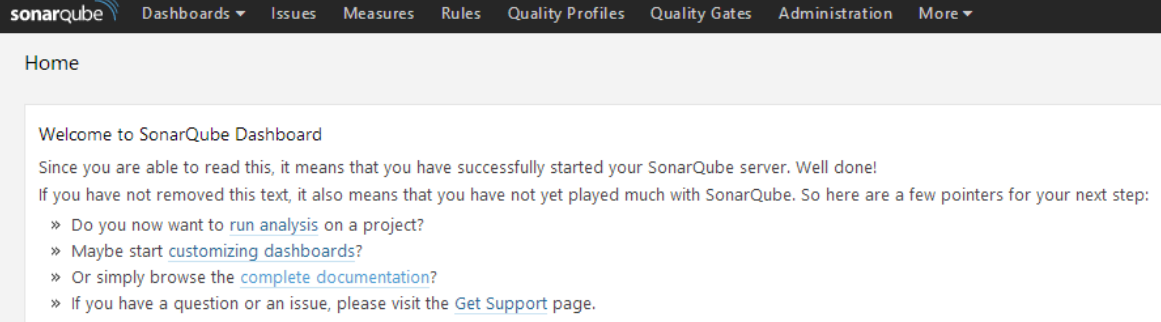
产品分析与代码监管之工具使用

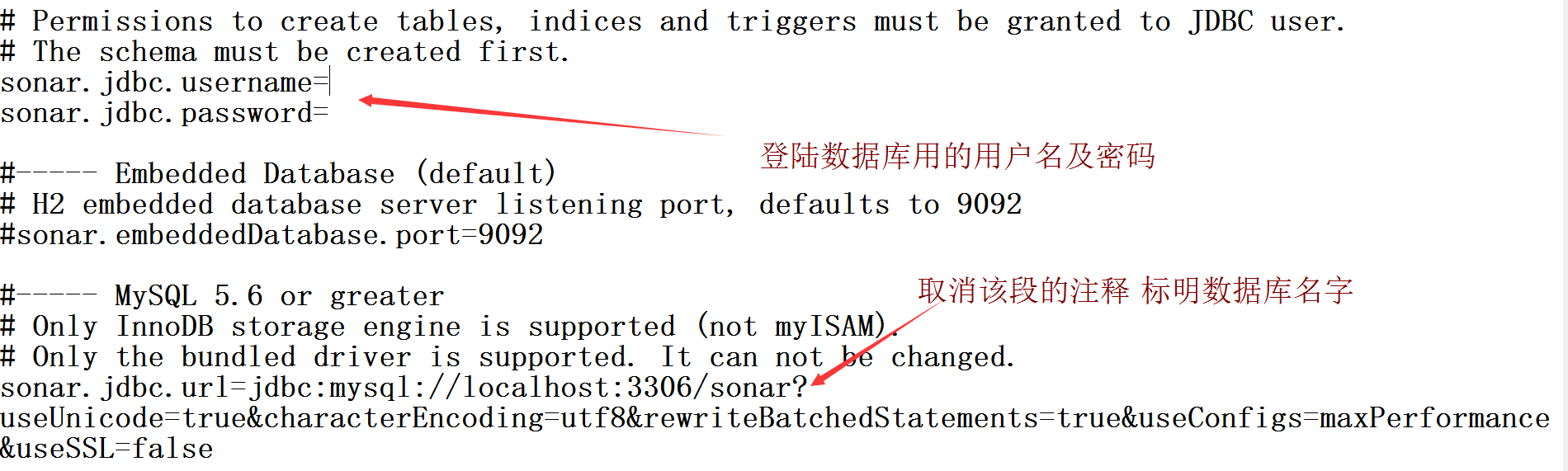
1. Sonarqube--代码检查工具
2. Sonarqube的安装
3. 下载sonarqube之后，解压打开bin目录，找到对应OS目录下的StartSonar。我电脑是win64位系统，则打开的是：



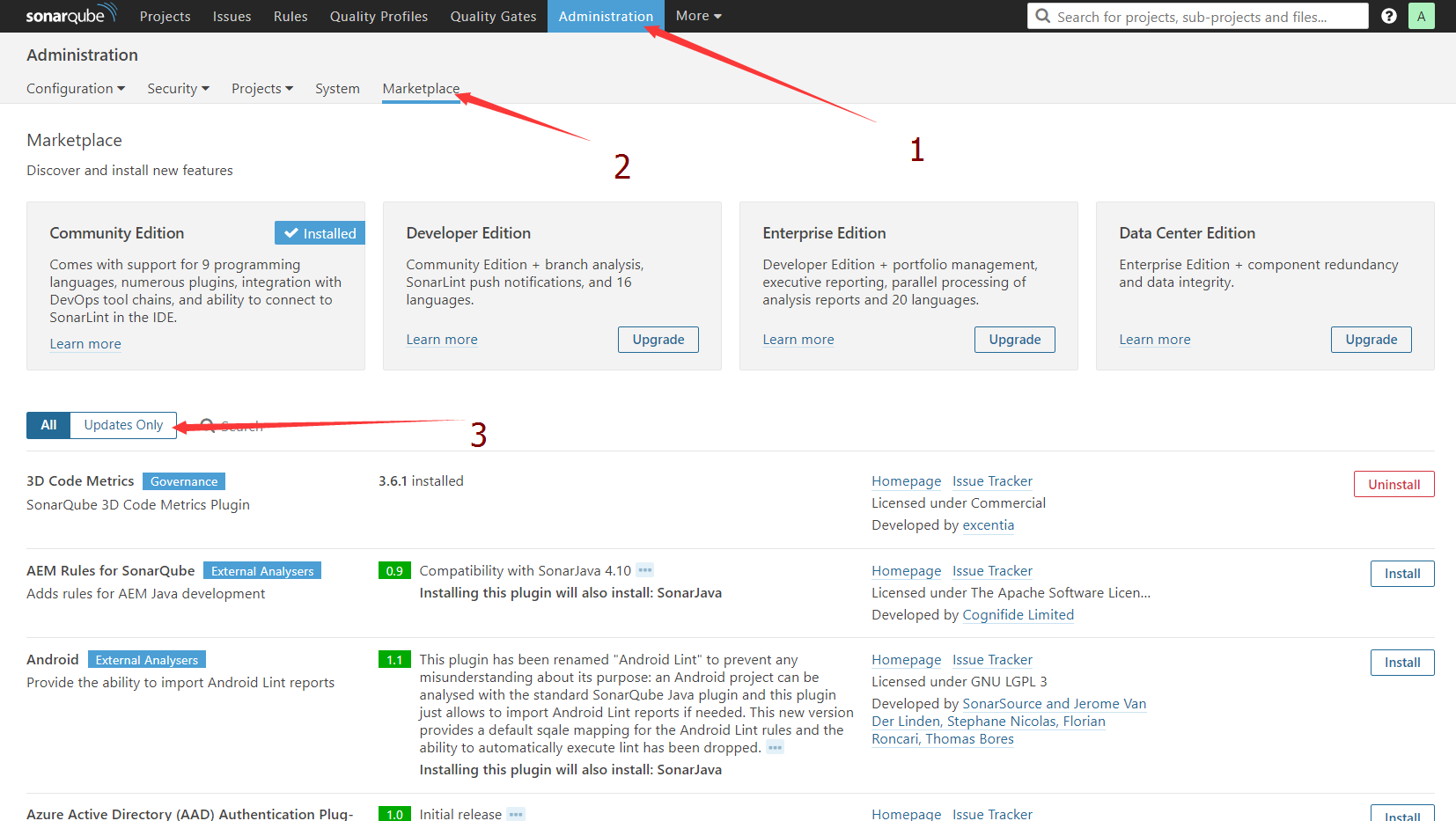
1. 启动浏览器，访问http://localhost:9000，如出现下图则表示安装成功。



1. Sonarqube的配置
2. 首先数据库的配置，这次我使用的是mysql。那么在mysql中新建一个数据库。
3. 打开安装目录下/conf/sonar.properties文件



1. 重启sonarqube服务，再次访问http://localhost:9000，会稍微有点慢，因为要初始化数据库信息。
2. 插件的顺序按图中顺序点击查找插件安装。



1. Sonarqube的使用
2. Sonar系统变量



1. 打开path，输入%SONAR\_RUNNER\_HOME%\bin
2. 打开项目的build.gradle里面添加

buildscript {  
 repositories {  
 maven {  
 url "https://plugins.gradle.org/m2/"  
 }  
 }  
 dependencies {  
  
 classpath "org.sonarsource.scanner.gradle:sonarqube-gradle-plugin:2.6.1"  
 }  
}  
  
apply plugin: "org.sonarqube"  
subprojects {  
 sonarqube {  
 properties {  
 property "sonar.host.url", "http://127.0.0.1:9000/" //我本地SonarQube平台的配置  
 property "sonar.projectName", project.name //projectname  
 property "sonar.projectKey", "56fae775a70d1b3dc743153fb6c4fa073e54fb9b" //projectkey  
 property "sonar.language", "java" //语言  
 property "sonar.sourceEncoding", "UTF-8" //编码  
 property "sonar.sources", android.sourceSets.main.java.srcDirs //源码，写这个就行  
 property "sonar.projectVersion", "1.0.0" //版本，随意  
 }  
 }  
 task wrapper(type: Wrapper) {  
 gradleVersion = '4.2.1'//我自己的gradle版本号，各位随意  
 }  
}

1. 项目根目录下新建sonar.properties，里面：

# must be unique in a given SonarQube instance

sonar.projectKey=my:project

# this is the name displayed in the SonarQube UI

sonar.projectName=apiautocore

sonar.projectVersion=1.0

# Path is relative to the sonar-project.properties file. Replace "\" by "/" on Windows.

# Since SonarQube 4.2, this property is optional if sonar.modules is set.

# If not set, SonarQube starts looking for source code from the directory containing

# the sonar-project.properties file.

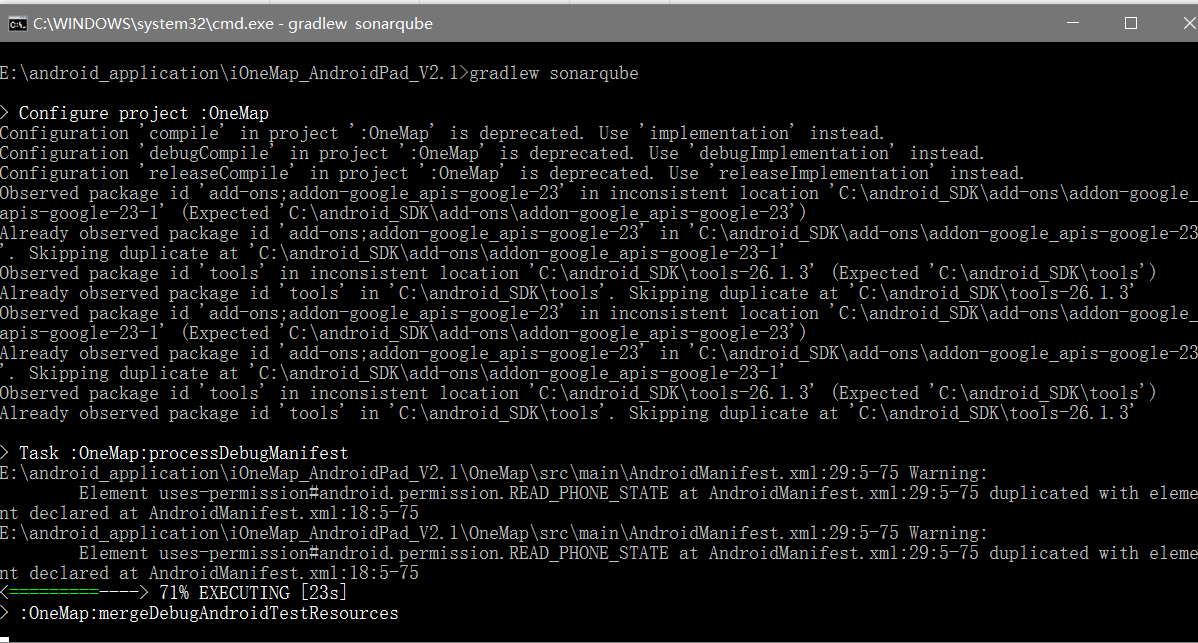
sonar.sources=src

# Encoding of the source code. Default is default system encoding

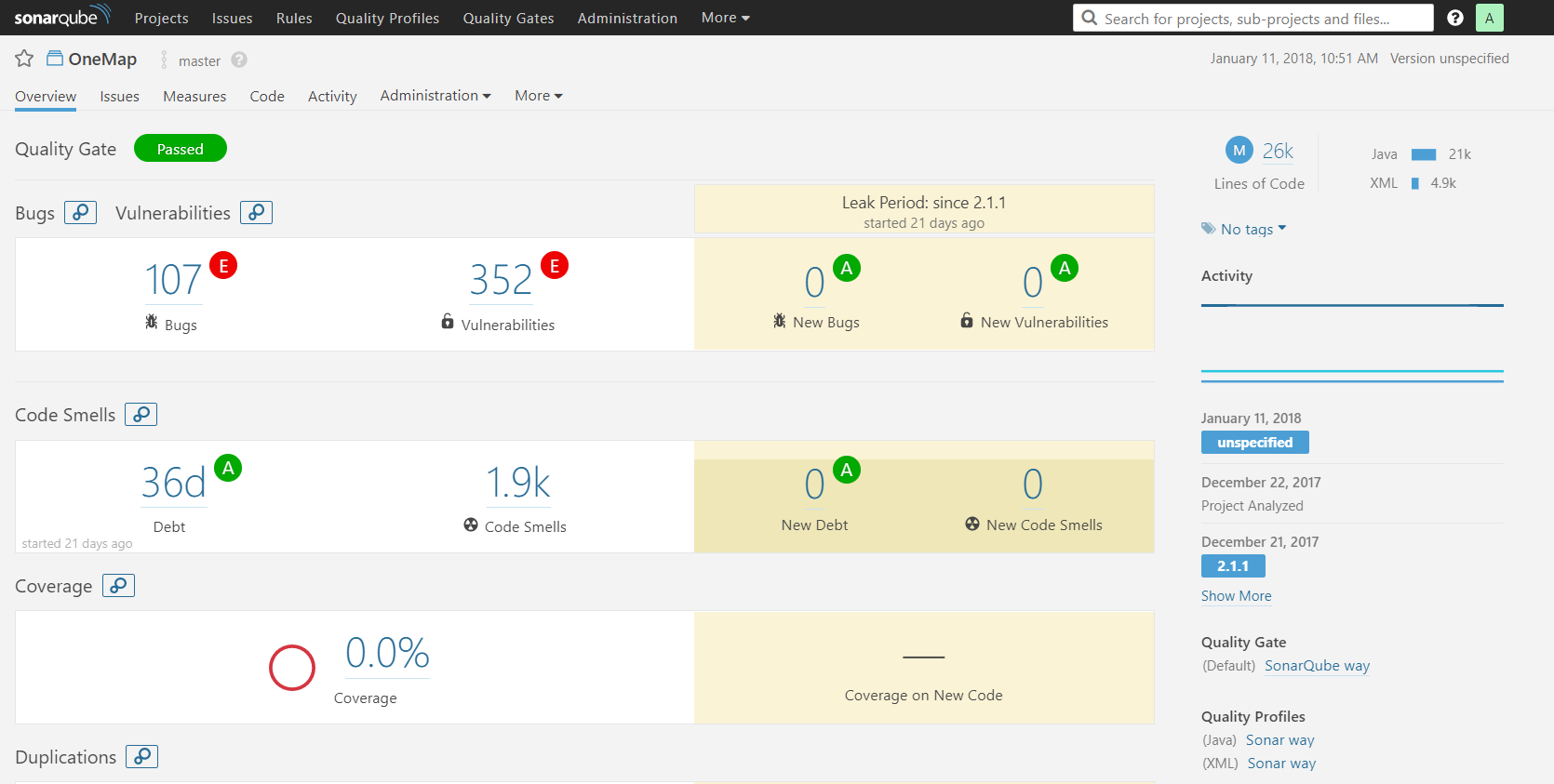
#sonar.sourceEncoding=UTF-8

其中：projectName是项目名字，sources是源文件所在的目录。

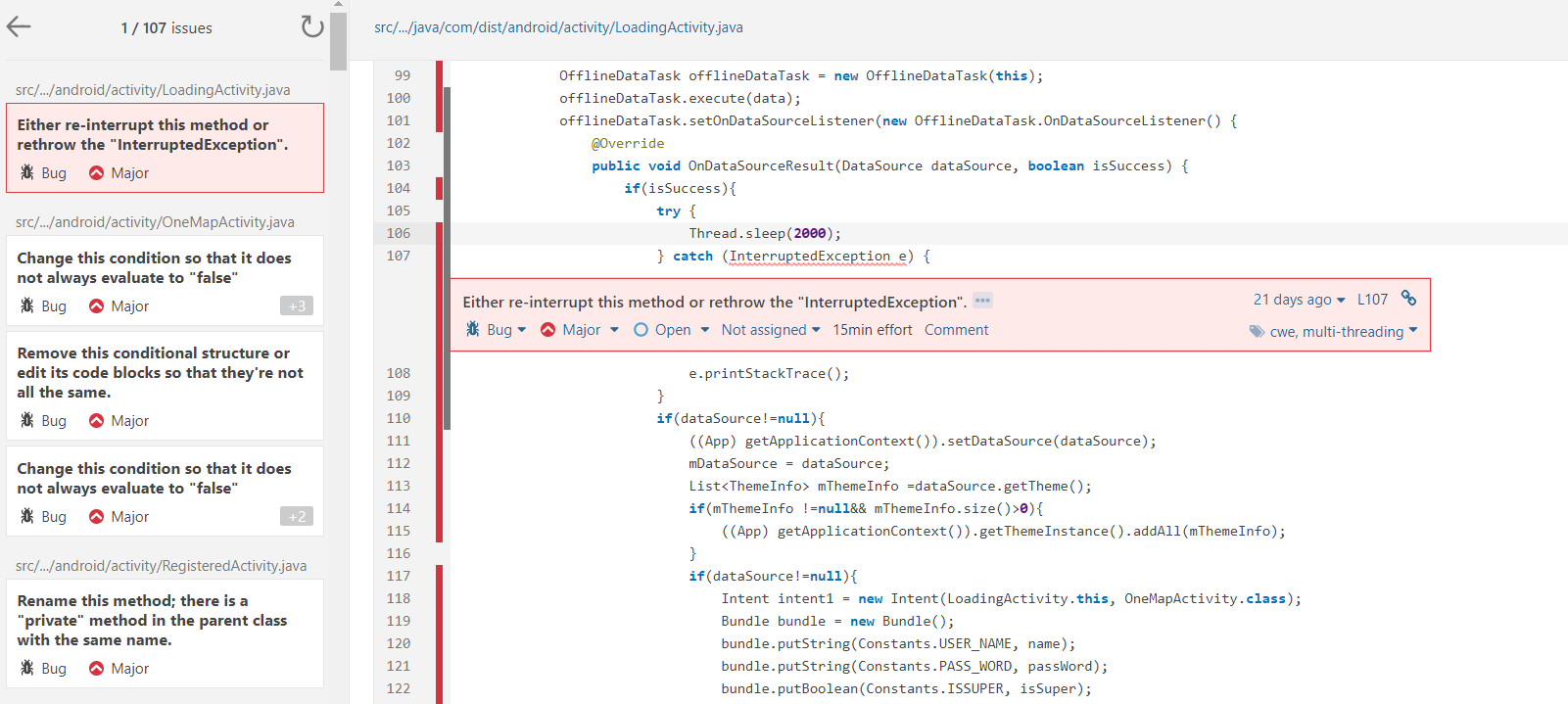
1. 完成之后，启动sonarqube，cmd中进入项目坐在目录，用指令：gradlew sonarqube，便会开始检查，如下图。



1. 然后进入localhost:9000，便可查看分析结果（包括bug与Vulnerabilities）。



1. 点开可查看bug或者Vulnerabilities详情



1. 听云—性能检测工具

由于听云为在线产品，所以只需要下载所需的探针，并配置到项目中即可。

1. 听云的配置
2. 听云的地址：<http://www.tingyun.com/>，进去登陆之后新建APP，如图：



1. 下载探针，添加到项目依赖中。
2. 工程相关依赖构建

1. 打开项目根目录下的build.gradle（Project）文件

2. 在buildscript 模块中加入代码

classpath 'com.networkbench.newlens.agent.android:agent-gradle-plugin:2.7.1'

3. 打开项目工程主模块下的build.gradle（Module）文件

4. 在文件中引入mavencentral()

repositories {

mavenCentral()

}

5. 在dependencies模块中添加代码

compile "com.networkbench.newlens.agent.android:nbs.newlens.agent:2.7.1"

6. 添加听云符号表插件

apply plugin: 'newlens'

1. 配置应用权限

构建完成后，请在待监测的App工程的AndroidMainfest.xml文件中增加以下的权限：

<uses-permission android:name="android.permission.ACCESS\_NETWORK\_STATE" />

<uses-permission android:name="android.permission.ACCESS\_WIFI\_STATE" />

<uses-permission android:name="android.permission.READ\_PHONE\_STATE"/>

<uses-permission android:name="android.permission.READ\_LOGS" />

<uses-permission android:name="android.permission.INTERNET" />

1. 插入初始化探针代码

1. 在嵌码项目工程的启动页面“MainActivity”中import NBSAppAgent类

import com.networkbench.agent.impl.NBSAppAgent;

2. 在“MainActivity”中的onCreate()方法（如未找到该方法请新增onCreate()）中初始化Android SDK

NBSAppAgent.setLicenseKey("7282dad857db436891ac22309c7ca13c").withLocationServiceEnabled(true).start(this.getApplicationContext());

3. 在“Application”中的onCreate()方法（如未找到该方法请新增onCreate()）中初始化Android SDK（可选）

注：该配置仅仅只限于有多进程性能监控需求的应用。

NBSAppAgent.setLicenseKey("7282dad857db436891ac22309c7ca13c").withLocationServiceEnabled(true).startInApplication(this.getApplicationContext());

4. 若无需采集地理位置，使用以下配置

NBSAppAgent.setLicenseKey("7282dad857db436891ac22309c7ca13c").start(this.getApplicationContext());

1. 配置混淆

1. 发布前在proguard混淆配置文件中增加以下内容，以免tingyunSDK不可用

# ProGuard configurations for NetworkBench Lens

-keep class com.networkbench.\*\* { \*; }

-dontwarn com.networkbench.\*\*

-keepattributes Exceptions, Signature, InnerClasses

# End NetworkBench Lens

2. 若需要保留行号信息，在proguard.cfg 中添加以下内容

-keepattributes SourceFile,LineNumberTable

1. 听云的使用

首先可以验证嵌码的完整性

i. 数据收集服务器校验

嵌码完成后可通过“LogCat”查看听云SDK日志输出结果，用以进行数据收集服务器校验TAG为NBSAgent，标准日志输出结果如下所示：

com.dist.android.jxOneMap I/NBSAgent: NBSAgent start.

com.dist.android.jxOneMap I/NBSAgent: NBSAgent enabled.

com.dist.android.jxOneMap I/NBSAgent: NBSAgent V2.7.1

com.dist.android.jxOneMap I/NBSAgent: connect success

ii. 数据功能完整性校验

嵌码完成后可通过“LogCat”查看听云SDK日志输出结果，用以进行数据功能完整性校验TAG为TingYun，标准日志输出结果如下所示：

com.dist.android.jxOneMap D/TingYun: Crash switch is true

com.dist.android.jxOneMap D/TingYun: webView switch is true

com.dist.android.jxOneMap D/TingYun: ANR monitor switch is true

com.dist.android.jxOneMap D/TingYun: UserAction Switch is true

com.dist.android.jxOneMap D/TingYun: cdnSwitch Switch is false

配置好之后运行项目，App会自动把运行的各个参数指标传到听云，我们可以直观的查看性能。

