严苛模式主要检测两大问题，一个是线程策略，即TreadPolicy，另一个是VM策略，即VmPolicy

ThreadPolicy线程策略检测

线程策略检测的内容有

自定义的耗时调用 使用detectCustomSlowCalls()开启

磁盘读取操作 使用detectDiskReads()开启

磁盘写入操作 使用detectDiskWrites()开启

网络操作 使用detectNetwork()开启

VmPolicy虚拟机策略检测

Activity泄露 使用detectActivityLeaks()开启

未关闭的Closable对象泄露 使用detectLeakedClosableObjects()开启

泄露的Sqlite对象 使用detectLeakedSqlLiteObjects()开启

检测实例数量 使用setClassInstanceLimit()开启

常见用法

严格模式的开启可以放在Application或者Activity以及其他组件的onCreate方法。为了更好地分析应用中的问题，建议放在Application的onCreate方法中。

其中，我们只需要在app的开发版本下使用 StrictMode，线上版本避免使用 StrictMode，这里定义了一个布尔值变量DEV\_MODE来进行控制。

private boolean DEV\_MODE = true;

public void onCreate() {

if (DEV\_MODE) {

StrictMode.setThreadPolicy(new StrictMode.ThreadPolicy.Builder()

.detectCustomSlowCalls() //API等级11，使用StrictMode.noteSlowCode

.detectDiskReads()

.detectDiskWrites()

.detectNetwork() // or .detectAll() for all detectable problems

.penaltyDialog() //弹出违规提示对话框

.penaltyLog() //在Logcat 中打印违规异常信息

.penaltyFlashScreen() //API等级11

.build());

StrictMode.setVmPolicy(new StrictMode.VmPolicy.Builder()

.detectLeakedSqlLiteObjects()

.detectLeakedClosableObjects() //API等级11

.penaltyLog()

.penaltyDeath()

.build());

}

super.onCreate();

}

其中Android3.0引入的方法包括detectCustomSlowCalls()和noteSlowCode()，它们都是用来检测应用中执行缓慢代码的或者潜在的缓慢代码。

检测项

ThreadPolicy

detectAll 检测所有潜在的违例

detectCustomSlowCalls 自定义耗时操作

detectDiskReads 读磁盘

detectDiskWrites 写磁盘

detectNetwork 检查网络

detectResourceMismatches 检查资源类型是否匹配

VmPolicy

detectAll 检测所有潜在的

detectActivityLeaks 检测Activity的泄露

detectCleartextNetwork 检测明文的网络

detectFileUriExposure 检测file://或者是content://

detectLeakedClosableObjects 检查为管理的Closable对象

detectLeakedRegistrationObjects 检测需要注册类型是否解注

detectLeakedSqlLiteObjects 检测sqlite对象，如cursors

检测到违规项之后的表现形式

penaltyDeath crash，在所有表现形式最后运行，

penaltyDeathOnNetwork crash,在所有值钱，必须调用detectNetwork去允许这个。

penaltyDialog 弹出dialog

penaltyDropBox 将日志吸入到dropbox中

penaltyFlashScreen 屏幕闪烁

penaltyLog log日志

ThreadPolicy 详解

detectNetwork() 用于检查UI线程中是否有网络请求操作

detectDiskReads() 和 detectDiskWrites() 是磁盘读写检查

noteSlowCall针对执行比较耗时的检查

penaltyDeath()，当触发违规条件时，直接Crash掉当前应用程序。

penaltyDeathOnNetwork()，当触发网络违规时，Crash掉当前应用程序。

penaltyDialog()，触发违规时，显示对违规信息对话框。

penaltyFlashScreen()，会造成屏幕闪烁，不过一般的设备可能没有这个功能。

penaltyDropBox()，将违规信息记录到 dropbox 系统日志目录中（/data/system/dropbox），你可以通过如下命令进行插件：

adb shell dumpsys dropbox dataappstrictmode --print

1

permitCustomSlowCalls()、permitDiskReads ()、permitDiskWrites()、permitNetwork： 如果你想关闭某一项检测，可以使用对应的permit\*方法

VMPolicy 详解

detectActivityLeaks() 用户检查 Activity 的内存泄露情况

detectLeakedClosableObjects()用于资源没有正确关闭时提醒

detectLeakedSqlLiteObjects() 和

detectLeakedClosableObjects()的用法类似，只不过是用来检查 SQLiteCursor 或者 其他 SQLite

对象是否被正确关闭

detectLeakedRegistrationObjects() 用来检查 BroadcastReceiver 或者

ServiceConnection 注册类对象是否被正确释放

setClassInstanceLimit()，设置某个类的同时处于内存中的实例上限，可以协助检查内存泄露

注意事项

只在开发阶段启用StrictMode，发布应用或者release版本一定要禁用它。

严格模式无法监控JNI中的磁盘IO和网络请求。

应用中并非需要解决全部的违例情况，比如有些IO操作必须在主线程中进行。