使用as进行分析,对小米便签全部源码（41个文件）自动完成质量审查，得到如下代码质量分析报告：

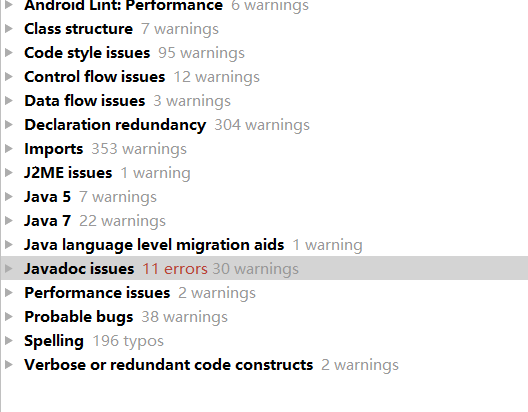


图10 as分析结果

在小米便签软件的源代码中，共审查出35个文件中的558个问题，经过人工审查，其中大量的问题在于规范和代码的简洁性要求，例如有308个Critical Issues为R.java类中的常量命名问题，所以总体来说，小米便签软件的质量较为不错。

以 Android 开头的组，例如

· Android > Lint > Correctness (可能影响程序正确性)

· Android > Lint > Performance (可能影响程序性能)

· Android > Lint > Security (可能影响程序安全性)

· 等等

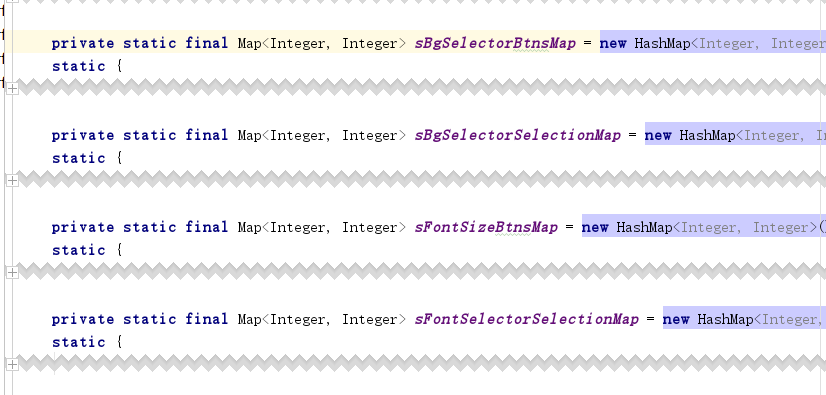
·  Class structure 组：指出类的设计上可能存在的问题

·  Code style issues 组：有助于提供代码书写规范

·  Probable bugs 组：有助于发现隐藏的问题

1. Android > Lint > Performance
2. HashMap可以替换为SparseArray，对于键类型为integer的映射，使用Android SparseArray API通常效率更高。此检查确定您可能需要考虑使用SparseArray而不是HashMap以获得更好性能的场景。当值类型是INTS的原始类型时，这是特别有用的，在这里您可以使用SaleSnTayRayand避免将int值自动装箱到整数。如果您需要构造一个HashMap，因为您需要在需要映射的控件之外调用一个API，那么您可以使用@SuppressLint注释来抑制此警告。

Noteeditactivity.java:

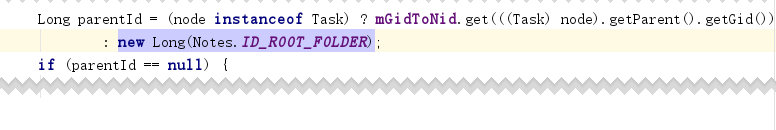


Notaslistadapter.java:



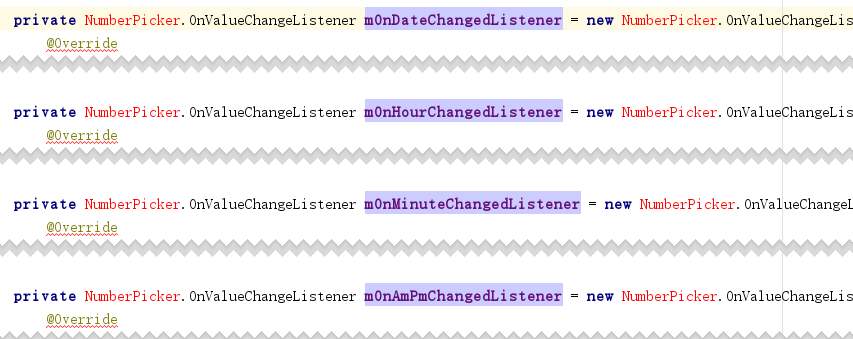
1. 应该使用valueOf而不是new，不应该直接调用包装类的构造函数，例如“new Integer（42）”。而是调用工厂方法的valueOf，例如Integer.valueOf（42）。这通常使用较少的内存，因为0和1等常用整数将共享一个实例。

Gtaskmanger.java

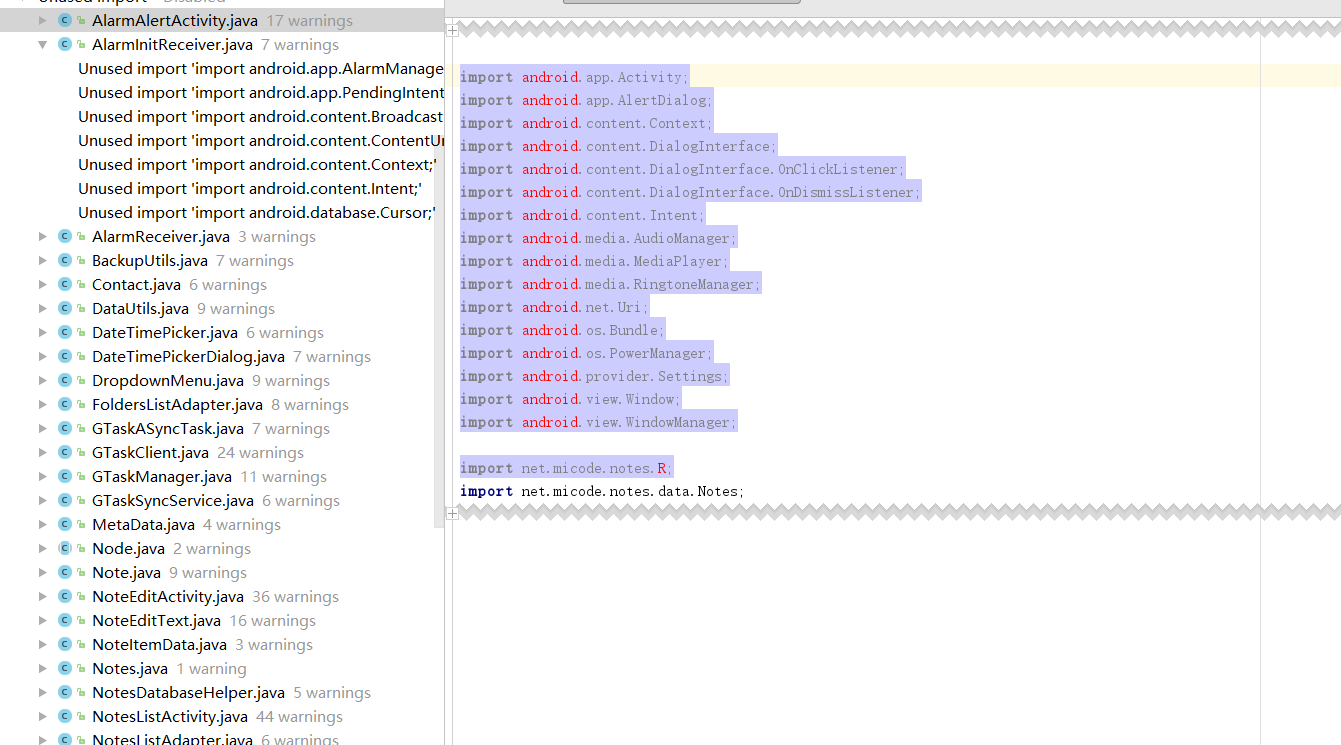


1. Class structure
2. 此检查搜索可以用局部变量替换的冗余类字段。如果一个字段的所有局部用法前面都有对该字段的赋值，则可以删除该字段，并用局部变量替换其用法。

datetimepicker

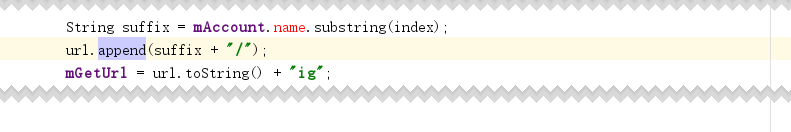


1. Import
2. 报告任何未使用的导入语句。由于IDEA可以使用其“Optimize Imports”命令自动检测和修复此类语句，因此此检查对于基于您不打算更改的代码的脱机报告非常有用

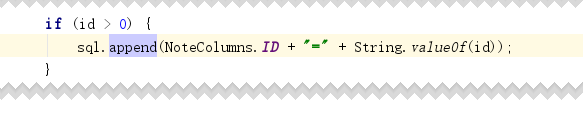


1. Performance issues

1.报告用作StringBuffer.append（）、StringBuilder.append（）或Appendable.append（）参数的字符串连接。这样的调用可以在现有的StrugBuff/Builder／AppEdable中变成链式的附加调用，从而节省额外的StringBuffer / Builder分配的成本。

此检查忽略编译时计算的字符串连接，当转换为链式追加调用时，这只会降低性能。

Gtaskclient.java



Notesprovider.java

1. Probable bugs

1.此检查分析方法控制和数据流，以报告始终为真或假的可能条件、静态证明其值为常量的表达式以及可能导致违反可空性合约的情况。

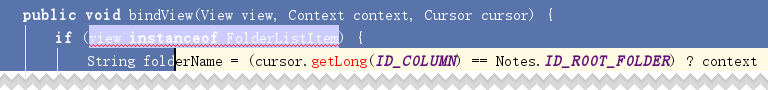
标记为@Nullable或@not null的变量、方法参数和返回值被视为可为null（或分别不为null），并在分析期间用于检查可为null的协定，例如报告可能产生的NullPointerException（NPE）错误。

可以使用@Contract注释定义更复杂的合同，例如：

@Contract（“\unull->null”）-如果第二个参数为null@Contract（“\unull->null；”，则方法返回null！空->！null”）-如果方法的第二个参数为null，则返回null，否则不返回null@Contract（“true->fail”）-一个典型的assertFalse方法，如果传递了true，则抛出异常

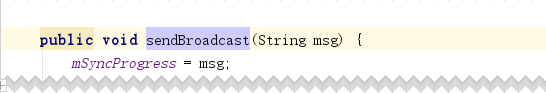
可以将检查配置为使用custom@Nullable

@NotNull注释（默认使用annotations.jar中的注释）



Folderslistadapter.java

1. 报告必须无限递归或引发异常的方法。本次检验报告的方法不能正常返回



Stasksyncservice.java

3.报告对忽略调用结果的特定方法的任何调用。检查的方法调用是：

调用简单的getter（除了返回一个字段之外什么也不做）

调用此检查设置中指定的方法

调用用org.jetbrains.annotations.Contract注释的方法（pure=true）

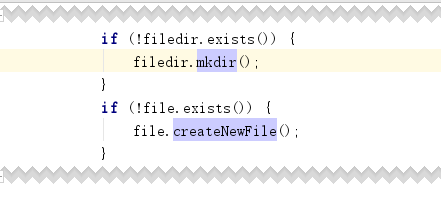
调用用javax.annotation.CheckReturnValue注释的方法

调用位于用javax.annotation.CheckReturnValue注释的类中的方法

调用位于用javax.annotation.CheckReturnValue注释的包中的方法

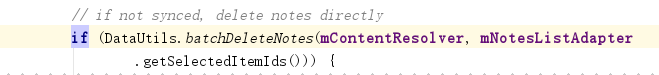
可选地所有对非库方法的调用

对于许多方法的调用，忽略结果是完全合法的，但对于某些方法，这几乎肯定是一个错误。忽略调用结果很可能是错误的方法示例包括java.io.inputStream.read（），它返回实际读取的字节数和java.lang.String或java.math.biginger上的任何方法，因为所有这些方法都是无副作用的，因此如果忽略这些方法就没有意义。



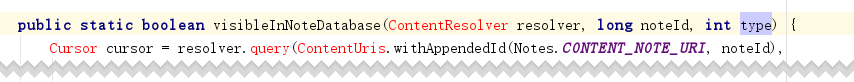
Backuputils.java:

4.报告具有空正文的if、while、do、for和switch语句。虽然有时是有意的，但这种结构是令人困惑的，而且常常是打字错误的结果。



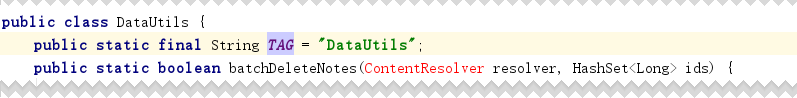
Noteslistactivtivity.java

1. Declaration redundancy
2. 此检查报告传递给特定参数的值似乎总是相同常量的方法。



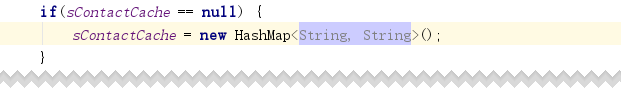
Datautils.java

1. 此检查报告在指定检查范围内找到的所有字段、方法或类，这些字段、方法或类的访问修饰符可能已缩小。

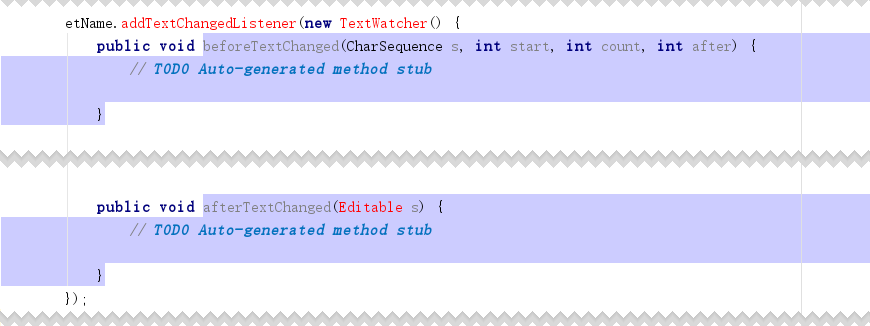


Datautils.java

1. 此检查报告在指定检查范围内找到的所有字段、方法或类，这些字段、方法或类的声明中可能添加了最终修饰符。

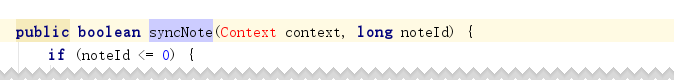


Backuputils.java

1. 此检查报告以下方法：-method为空或-interface method的所有实现为空或-method本身为空且仅由空方法重写注意，仅包含super（）调用并传递其自身参数的方法也被视为空。对于使用特殊注释（例如EJB annotations javax.EJB.Init和javax.EJB.Remove）注释的方法，此检查将自动取消。
2. 

Noteslistactivtivity.java

1. 报告其返回值在调用时从不使用的方法。此类方法的返回类型可以设为void。



Note.java

1. 此检查报告返回值似乎总是相同常量的方法和方法层次结构。