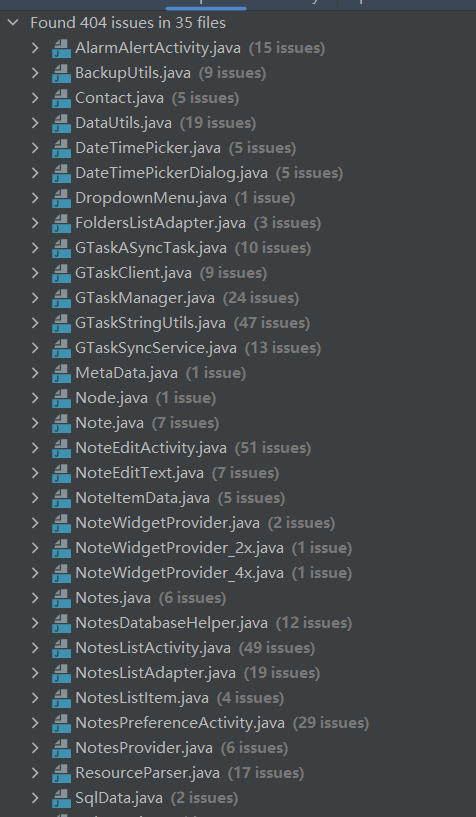
**自动质量分析：**

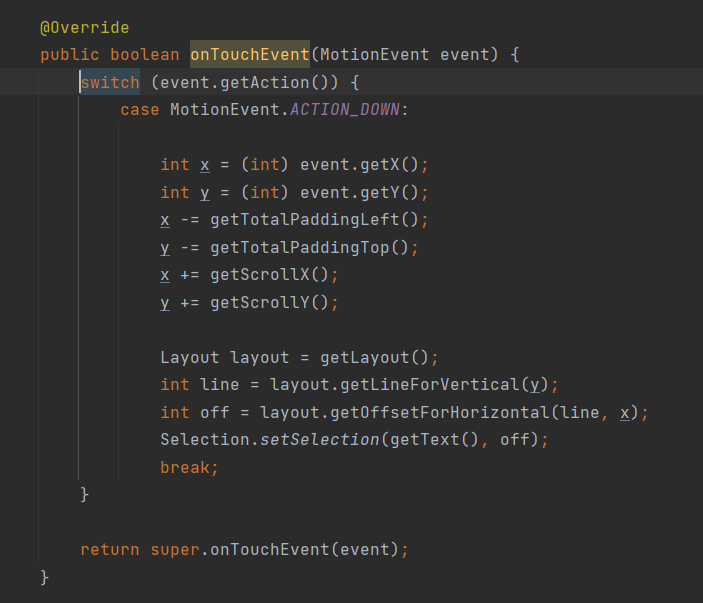


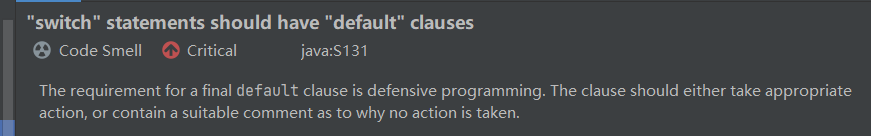
在35个文件中发现了一共404个问题。

（1）

ui/NoteEditText.java：

Critical型问题：





问题：

在这个函数里，调用了switch case选择分支，但是没有设置默认case（default情况），如果出现给定情况之外的情况，则容易出现问题。

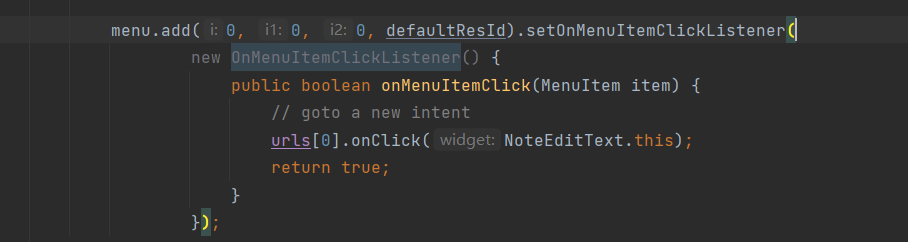
解决办法：

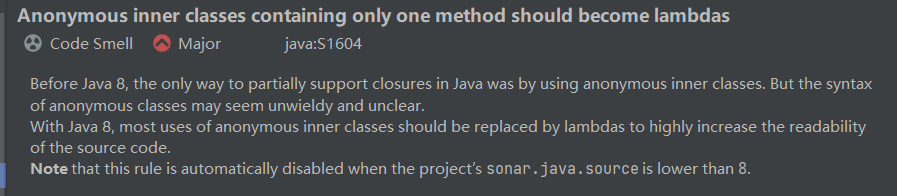
应该设置一个default case的情况来抛出异常，如果出现对应问题，则可以知道问题来源。

（2）

ui/NoteEditText.java：

Critical型问题：





问题：

使用了匿名内部类（anonymous inner classes），但是在JAVA 8的后续版本中，具有更高可读性的匿名函数（lambdas）显然是更好之选。

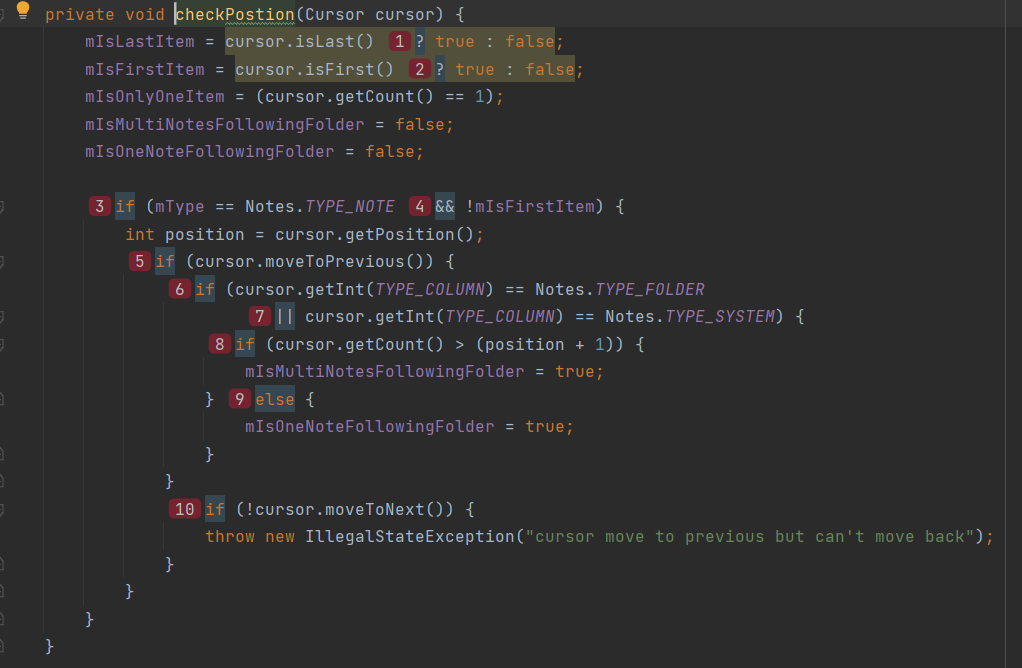
解决办法：

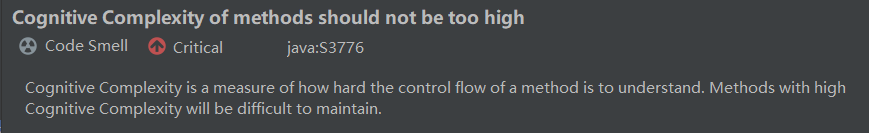
使用匿名函数来代替此处的匿名类

（3）

ui/NoteEditText.java：

Critical型问题：





问题：

设置的if判断过多，函数的认知复杂度太高，可读性较差。

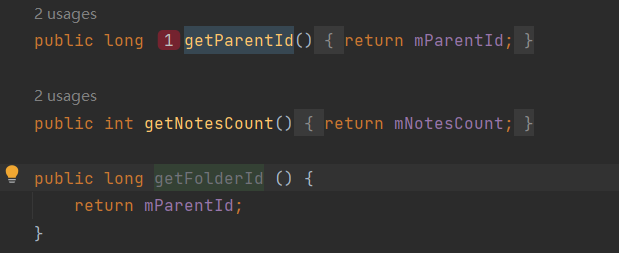
解决办法：

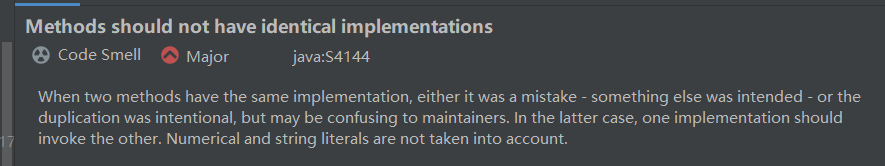
通过数学分析，来降低if判断的个数，提高判断质量，以降低认知复杂度。

（4）

ui/NoteEditText.java：

Critical型问题：





问题：

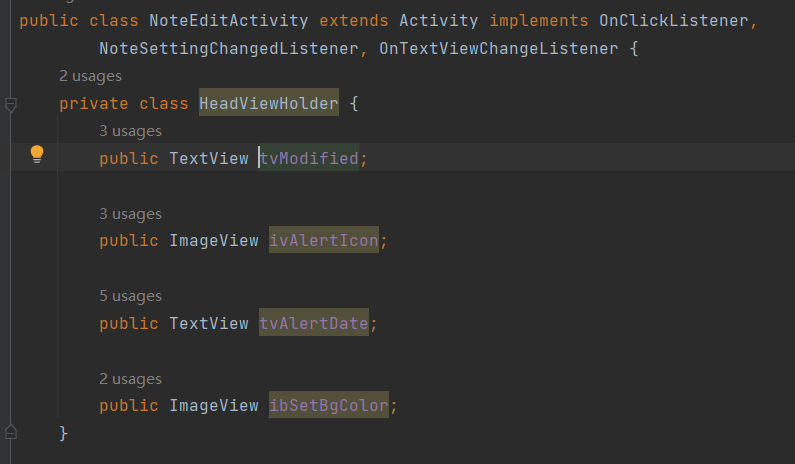
两个函数只有函数名是不同的，但是起到的作用是一致的，出现了冗余的现象。

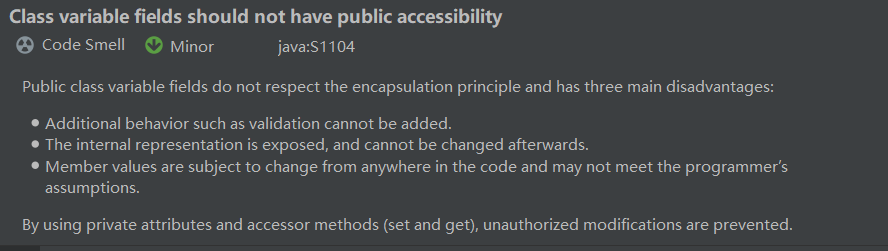
解决办法：

在调用getFolderId的地方调用getParentId即可，利用注释在调用处说明调用的目的。

（5）

ui/NoteEditActivity.java：





问题：

使用了public变量字段，而没有使用私有声明。封装不是很好。

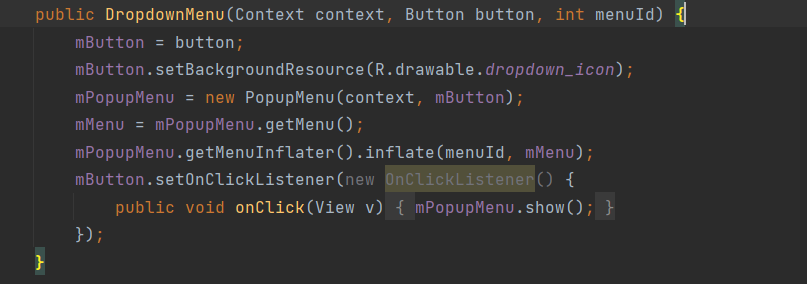
解决办法：

声明时声明为private字段，将变量字段封装起来。

**手动质量分析：**

（1）

ui/DropdownMenu.java：



问题：

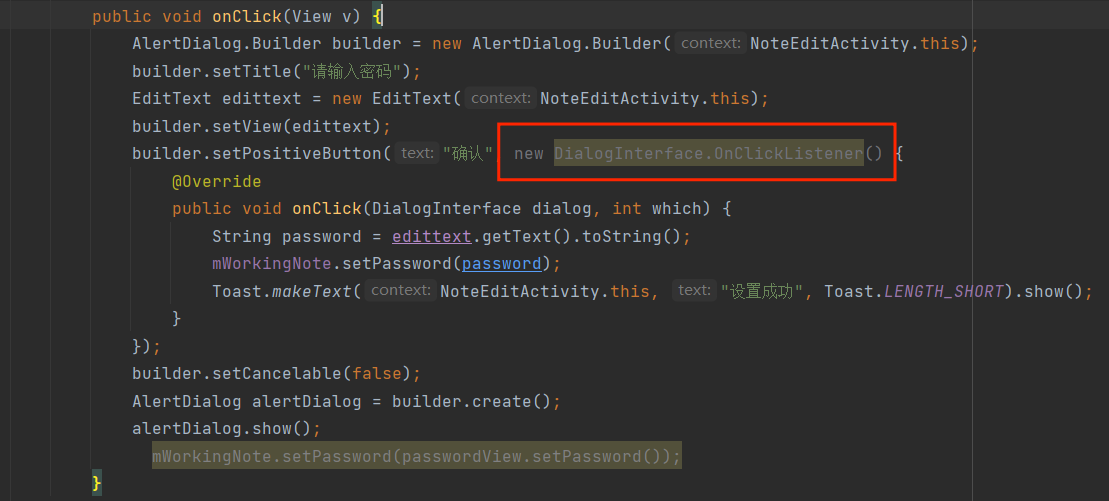
使用了匿名内部类，而没有使用可读性更好的匿名函数。

解决办法：

使用匿名函数代替匿名内部类。

（2）

ui/NoteEditActivity.java：



问题：

没有注释，可读性较差。且使用了匿名内部类，没有使用匿名函数。

解决办法：

对于红框内的部分，使用匿名函数类加以代替。

同时使用注释，向读者解释，这是在进行密码的设置。