Al Imam Mohammad Ibn Saud Islamic University
College of Computer and Information Sciences

Computer Science Department

CS292 Software Engineering2

Alinma's Bank application

Second semester 1443/2021

Date: 4/11/2021

Section: 371

Group members' names	Students IDs	Students' emails
Manar Mofid Zahid Yabrak	440027466	mmzyabrak@sm.imamu.edu.sa
Shouq Khalid Almutairi	440021077	skaalmutairi@sm.imamu.edu.sa
Noor Hossam Elmasry	440027219	nelmasry@sm.imamu.edu.sa
Reema Mohammed Aloqayli	440025060	rmnaloqayli@sm.imamu.edu.sa

Instruction: Dr. Lamees Alhazzaa

Contents

Figures

1 Introduction

1.1 Purpose

Banking applications and their vulnerabilities are a growing concern. This report discusses a static analysis of Alinma Bank application. Specifically, it discusses the design analysis, code structure, security aspects, and documentation.

We examined the security of the application using a vulnerability assessment, bug analysis, and code smell analysis along with the way the source code was written and its comments.

1.2 Goals

1. Perform a static analysis on Alinma Bank application.
2. Explain and discuss the finding of the analysis.
3. Provide a solution for the issues found by the analysis.

1.3 Lessons Learned

During the workshop, we learned how to use and install MobSF, SonarQube and PMD tools, as well as identify issues regarding security, design, clean code, and code documentation.

2 Mobile App analysis results

2.1 Installation

This is the steps we took to install MobSF:

1. Installing/checking the requirement from MobSF'S documentation page https://mobsf.github.io/docs/#/requirements.
2. Used APKCOMBO to download Alinma's Bank APK.
3. Installing MobSF tool from GitHub using the command:

```
git clone https://github.com/MobSF/Mobile-Security-Framework-MobSF.git
cd Mobile-Security-Framework-MobSF
./setup.sh
```

This is the steps we took to install SonarQube:

1. Installing/checking the requirement from SonarQube's documentation page https://docs.sonarqube.org/latest/requirements/requirements/.
2. Download SonarQube.
3. install Apache Maven.
4. Setup the JDK.
5. Edit the configuration to JDK path and the correct proprieties.
6. Run the server's setup script.

This is the steps we took to install PMD:

1. Installing/checking the requirement from PMD's documentation page https://pmd.github.io/pmd-6.40.0/.
2. Download PMD through GitHub using the command:

```
curl -OL https://github.com/pmd/pmd/releases/download/pmd_releases%2F6.40.0/pmd-bin-6.40.0.zip
```

2.2 Issues

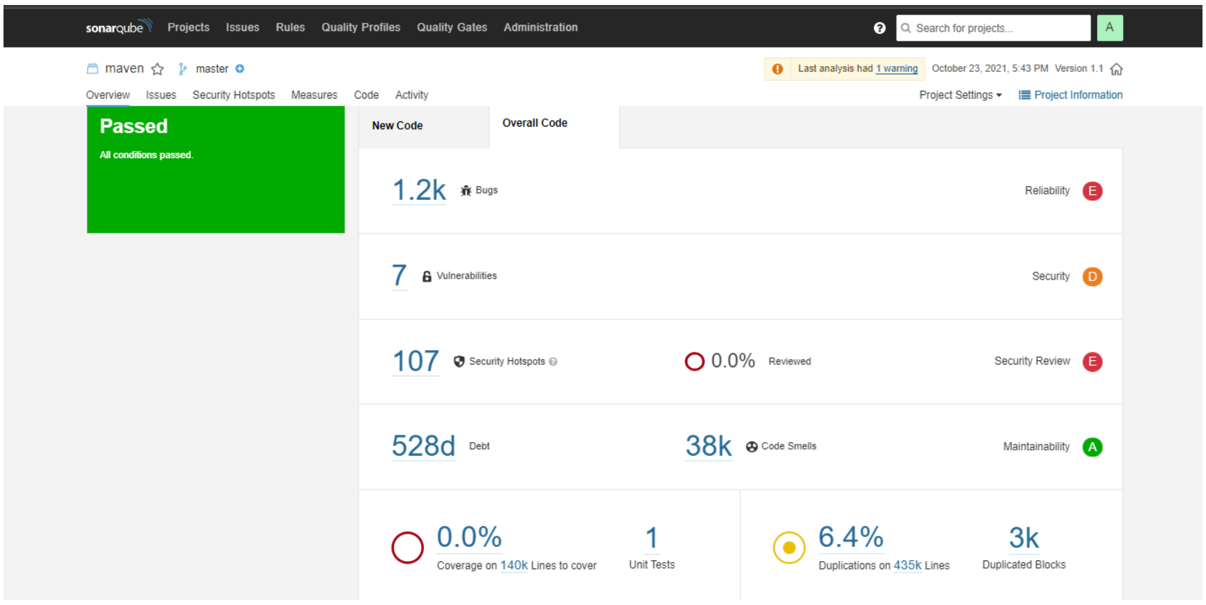
Many of APK downloaders didn't have Alinma's Bank APK or had the wrong version.
One of the members couldn't install MobSF so she opted to use a website called Codacy.
For the security issues MobSF didn't analyze the application thoroughly so we used another analysis tool called SonarQube.
For the design analysis we used PMD since MobSF didn't show any issues in that regards.

3 Discussion

3.1 Security Issues

In this part of the analysis, we will talk about some of the security issues that we found in the system, including Bugs, Vulnerabilities, and Code Smells. And as you can see in the figure below, we found thousands of bugs and code smells, some of the were major issues and some of them were minor, and we generally talked about the most important ones.

Table showing the security issues in Alinma Bank application



Last analysis had 1 warning October 23, 2021, 5:43 PM Version 1.1

Passed
All conditions passed.

1.2k Bugs — Reliability E

7 Vulnerabilities — Security D

107 Security Hotspots — 0.0% Reviewed — Security Review E

528d Debt — 38k Code Smells — Maintainability A

0.0% Coverage on 140k Lines to cover — 1 Unit Tests

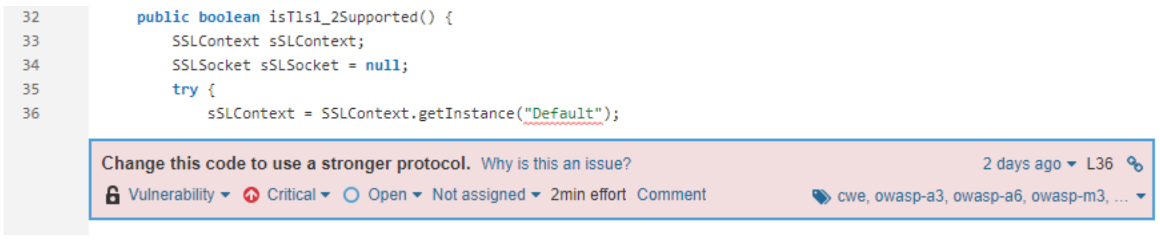
6.4% Duplications on 435k Lines — 3k Duplicated Blocks

Figure 1 Overall Security Issues in Alinma Bank Application

3.1.1 Vulnerability

1. **Insecure SSL/TLS protocols should be avoided.**

   When an unsafe TLS protocol version is utilized, this rule causes a problem.



```
    public boolean isTls1_2Supported() {
        SSLContext sSLContext;
        SSLSocket sSLSocket = null;
        try {
            sSLContext = SSLContext.getInstance("Default");
```

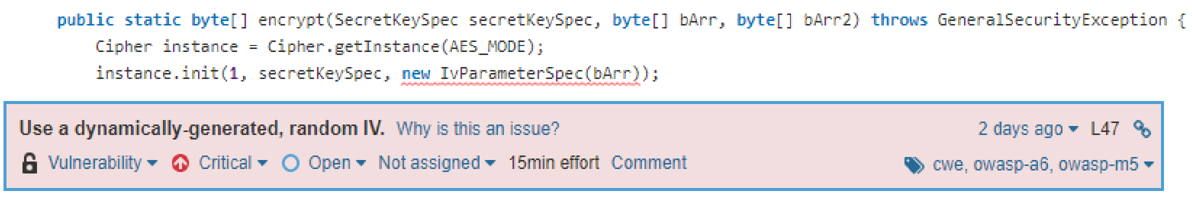
Change this code to use a stronger protocol. Why is this an issue? 2 days ago L36
Vulnerability Critical Open Not assigned 2min effort Comment cwe, owasp-a3, owasp-a6, owasp-m3, ...

Figure 2 Insecure Protocols

   File: com/alinma/tls/TlsCheckerPlugin.java

2. **The Cipher Block Chaining IVs should be unpredictably unpredictable.**

   When using the Cipher Block Chaining (CBC) method to encrypt data, an Initialization Vector (IV) is utilized to randomize the encryption, which means that the same plaintext does not always yield the same ciphertext under the same key. The IV does not have to be secret, but it should be unpredictable in order to avoid a "Chosen-Plaintext Attack." NIST recommends using a secure random number generator to produce Initialization Vectors.



```
public static byte[] encrypt(SecretKeySpec secretKeySpec, byte[] bArr, byte[] bArr2) throws GeneralSecurityException {
    Cipher instance = Cipher.getInstance(AES_MODE);
    instance.init(1, secretKeySpec, new IvParameterSpec(bArr));
```

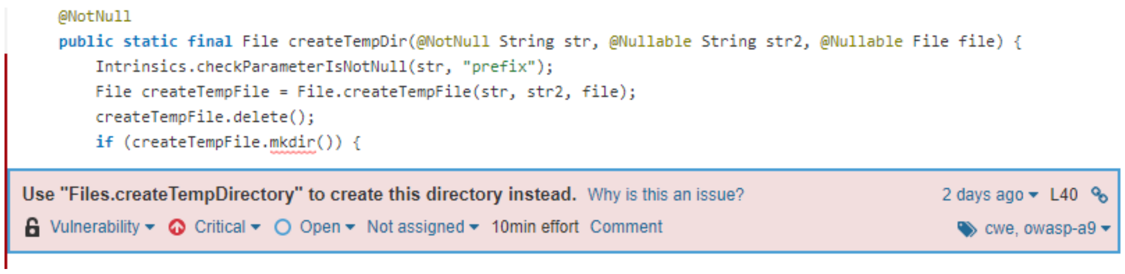
Use a dynamically-generated, random IV. Why is this an issue? 2 days ago L47
Vulnerability Critical Open Not assigned 15min effort Comment cwe, owasp-a6, owasp-m5

Figure 3 Random IV

   File: com/scottyab/aescrypt/AESCrypt.java

3. **Insecure techniques of creating temporary files should not be employed.**

   Using File.createTempFile as the initial step in generating a temporary directory result in a race situation, making it fundamentally unstable and insecure.



```
@NotNull
public static final File createTempDir(@NotNull String str, @Nullable String str2, @Nullable File file) {
    Intrinsics.checkParameterIsNotNull(str, "prefix");
    File createTempFile = File.createTempFile(str, str2, file);
    createTempFile.delete();
    if (createTempFile.mkdir()) {
```

Use "Files.createTempDirectory" to create this directory instead. Why is this an issue? 2 days ago L40
Vulnerability Critical Open Not assigned 10min effort Comment cwe, owasp-a9

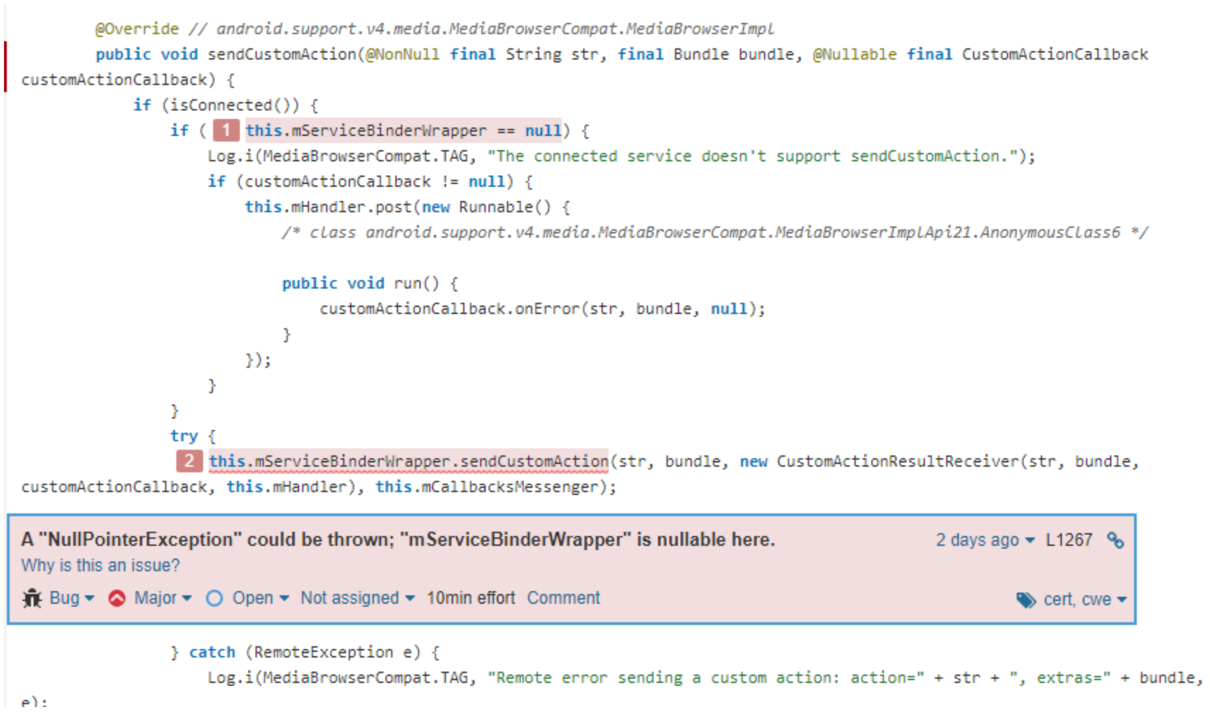
Figure 4 Insecure Temporary File

   File: kotlin/io/FilesKt__UtilsKt.java

3.1.2 Bugs

1. **Null pointers should not be dereferenced**

   Null references should never be dereferenced or accessed. This will result in a NullPointerException being raised. At best, such an exception will result in the sudden termination of the program. At worst, it could leak debugging information that an attacker could use, or it could allow an attacker to circumvent security safeguards.



```
        @Override // android.support.v4.media.MediaBrowserCompat.MediaBrowserImpl
        public void sendCustomAction(@NonNull final String str, final Bundle bundle, @Nullable final CustomActionCallback
customActionCallback) {
            if (isConnected()) {
                if ( 1 this.mServiceBinderWrapper == null) {
                    Log.i(MediaBrowserCompat.TAG, "The connected service doesn't support sendCustomAction.");
                    if (customActionCallback != null) {
                        this.mHandler.post(new Runnable() {
                            /* class android.support.v4.media.MediaBrowserCompat.MediaBrowserImplApi21.AnonymousClass6 */

                            public void run() {
                                customActionCallback.onError(str, bundle, null);
                            }
                        });
                    }
                }
                try {
                   2 this.mServiceBinderWrapper.sendCustomAction(str, bundle, new CustomActionResultReceiver(str, bundle,
customActionCallback, this.mHandler), this.mCallbacksMessenger);
```

A "NullPointerException" could be thrown; "mServiceBinderWrapper" is nullable here.   2 days ago L1267
Why is this an issue?
Bug Major Open Not assigned 10min effort Comment   cert, cwe

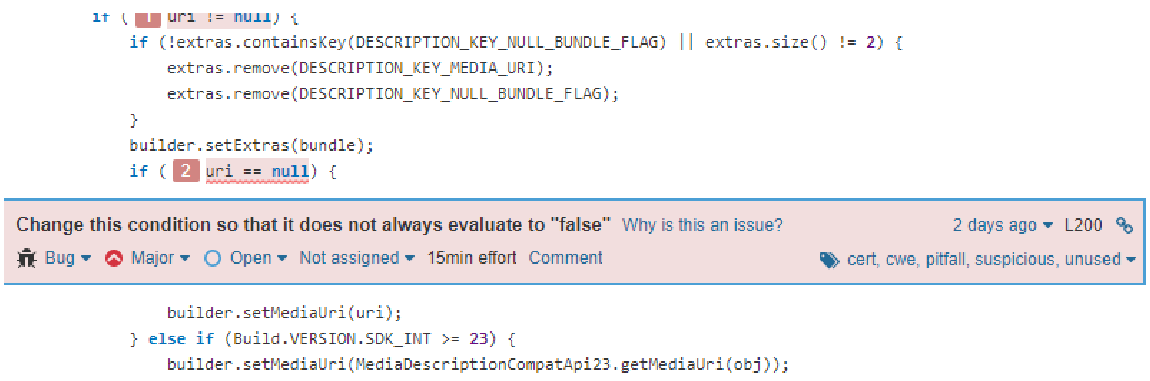
```
                } catch (RemoteException e) {
                    Log.i(MediaBrowserCompat.TAG, "Remote error sending a custom action: action=" + str + ", extras=" + bundle,
e);
```

Figure 5 Null Pointer Exception

File: android/support/v4/media/MediaBrowserCompat.java

2. **Conditionally run code must be accessible.**

   Conditional expressions that are either always true or false might result in dead code. Such code is always prone to bugs and should never be used in production.



```
if ( 1 uri != null) {
    if (!extras.containsKey(DESCRIPTION_KEY_NULL_BUNDLE_FLAG) || extras.size() != 2) {
        extras.remove(DESCRIPTION_KEY_MEDIA_URI);
        extras.remove(DESCRIPTION_KEY_NULL_BUNDLE_FLAG);
    }
    builder.setExtras(bundle);
    if ( 2 uri == null) {
```

Change this condition so that it does not always evaluate to "false"   Why is this an issue?   2 days ago   L200

Bug   Major   Open   Not assigned   15min effort   Comment   cert, cwe, pitfall, suspicious, unused

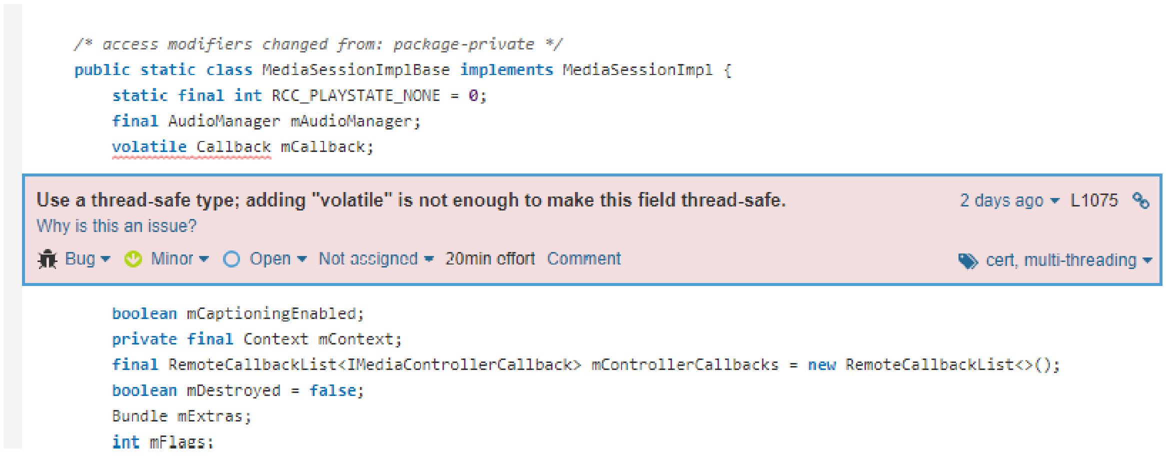
```
        builder.setMediaUri(uri);
    } else if (Build.VERSION.SDK_INT >= 23) {
        builder.setMediaUri(MediaDescriptionCompatApi23.getMediaUri(obj));
```

Figure 6 Accessible Condition

File: android/support/v4/media/MediaDescriptionCompat.java

3. **Non-primitive fields should not be considered "volatile."**

   When an array is marked volatile, it means that the array will always be read fresh and will never be thread cached, but the objects in the array will not be. Likewise, designating a mutable object field volatile indicates that the object reference is volatile but the object itself is not, and other threads may not observe updates to the object state. This can be avoided with arrays by using the appropriate AtomicArray class, such as AtomicIntegerArray. For changeable objects, the volatile should be removed, and another technique, such as synchronization or ThreadLocal storage, should be used to assure thread safety.



```
/* access modifiers changed from: package-private */
public static class MediaSessionImplBase implements MediaSessionImpl {
    static final int RCC_PLAYSTATE_NONE = 0;
    final AudioManager mAudioManager;
    volatile Callback mCallback;
```

Use a thread-safe type; adding "volatile" is not enough to make this field thread-safe.   2 days ago   L1075
Why is this an issue?

Bug   Minor   Open   Not assigned   20min effort   Comment   cert, multi-threading

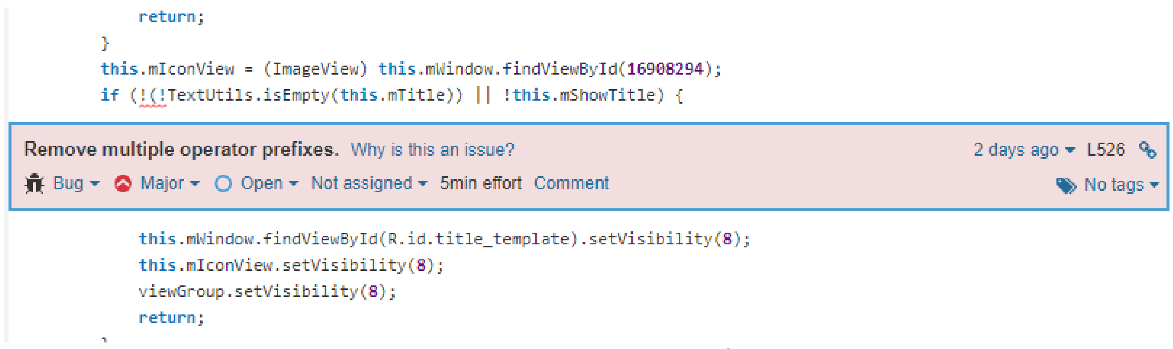
```
    boolean mCaptioningEnabled;
    private final Context mContext;
    final RemoteCallbackList<IMediaControllerCallback> mControllerCallbacks = new RemoteCallbackList<>();
    boolean mDestroyed = false;
    Bundle mExtras;
    int mFlags;
```

Figure 7 Thread Safety

File: android/support/v4/media/session/MediaSessionCompat.java

4. Unary prefix operators should not be used more than once.

A mistake is frequently caused by the superfluous repeating of an operator. There's no need to write “!!!i” when “!i” will do.



```
            return;
        }
        this.mIconView = (ImageView) this.mWindow.findViewById(16908294);
        if (!(!TextUtils.isEmpty(this.mTitle)) || !this.mShowTitle) {
```

Remove multiple operator prefixes. Why is this an issue? 2 days ago L526
Bug Major Open Not assigned 5min effort Comment No tags

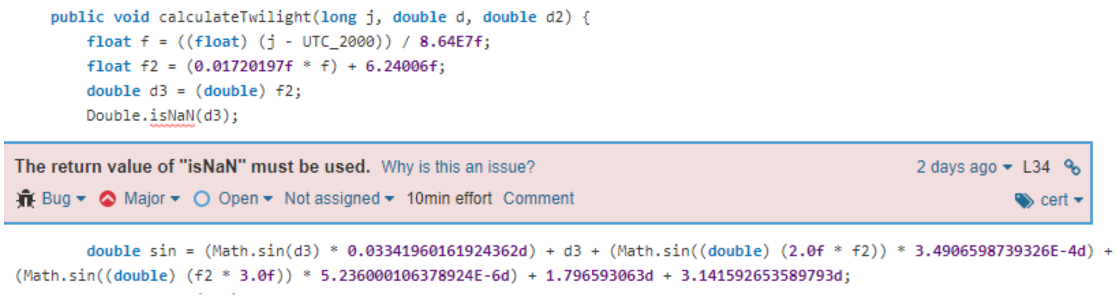
```
            this.mWindow.findViewById(R.id.title_template).setVisibility(8);
            this.mIconView.setVisibility(8);
            viewGroup.setVisibility(8);
            return;
```

Figure 8 Multiple Operator Prefixes

File: androidx/appcompat/app/AlertController.java

5. Functions with no side effects should not have their return values ignored.

What is the point of calling a method if the results are disregarded when there are no side effects? In this scenario, whether the call is superfluous and must be removed, or the source code does not behave as expected.



```
    public void calculateTwilight(long j, double d, double d2) {
        float f = ((float) (j - UTC_2000)) / 8.64E7f;
        float f2 = (0.01720197f * f) + 6.24006f;
        double d3 = (double) f2;
        Double.isNaN(d3);
```

The return value of "isNaN" must be used. Why is this an issue? 2 days ago L34
Bug Major Open Not assigned 10min effort Comment cert

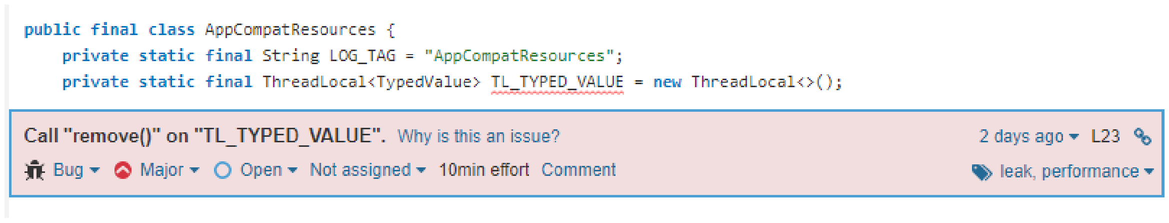
```
        double sin = (Math.sin(d3) * 0.03341960161924362d) + d3 + (Math.sin((double) (2.0f * f2)) * 3.4906598739326E-4d) +
(Math.sin((double) (f2 * 3.0f)) * 5.236000106378924E-6d) + 1.796593063d + 3.141592653589793d;
```

Figure 9 Ignored Return Value

File: androidx/appcompat/app/TwilightCalculator.java

6. **When "ThreadLocal" variables are no longer in use, they should be cleaned up.**

   ThreadLocal variables are designed to be trash collected when the holding thread dies. Memory leaks can arise when holding threads are re-used, which is common on application servers that use a thread pool.



```
public final class AppCompatResources {
    private static final String LOG_TAG = "AppCompatResources";
    private static final ThreadLocal<TypedValue> TL_TYPED_VALUE = new ThreadLocal<>();
```

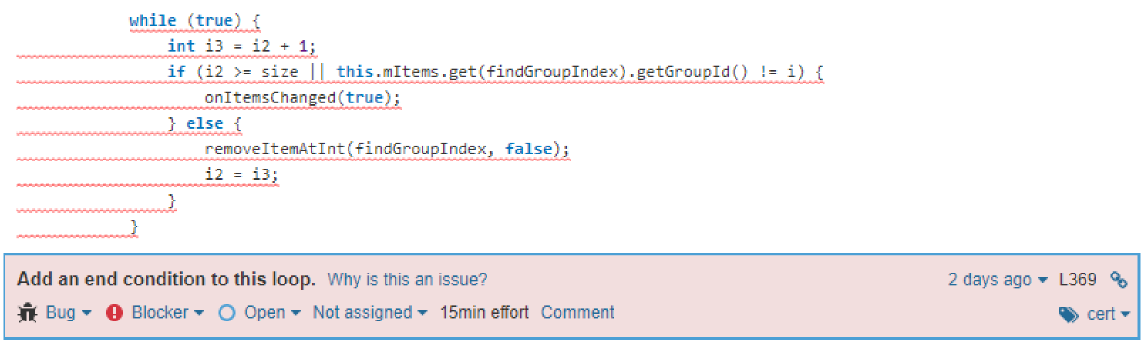
Call "remove()" on "TL_TYPED_VALUE". Why is this an issue? 2 days ago L23
Bug Major Open Not assigned 10min effort Comment leak, performance

Figure 10 Clean up Variables

File: androidx/appcompat/content/res/AppCompatResources.java

7. **Infinite loops should be avoided.**

   An infinite loop is one that will never terminate while the program is running; in order to exit the loop, you must destroy the program. Every loop should have an end condition, whether it is met by achieving the loop's end condition or by using a break.



```
        while (true) {
            int i3 = i2 + 1;
            if (i2 >= size || this.mItems.get(findGroupIndex).getGroupId() != i) {
                onItemsChanged(true);
            } else {
                removeItemAtInt(findGroupIndex, false);
                i2 = i3;
            }
        }
```

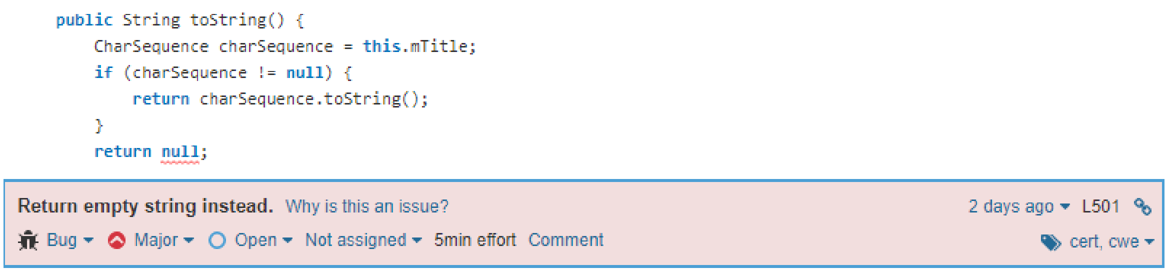
Add an end condition to this loop. Why is this an issue? 2 days ago L369
Bug Blocker Open Not assigned 15min effort Comment cert

Figure 11 Avoid Infinite loop

File: androidx/appcompat/view/menu/MenuBuilder.java

8. **The methods "toString()" and "clone()" should not return null.**

   When you call toString() or clone() on an object, it should always return a string or another object. Instead, returning null violates the method's implicit contract.



```
public String toString() {
    CharSequence charSequence = this.mTitle;
    if (charSequence != null) {
        return charSequence.toString();
    }
    return null;
```

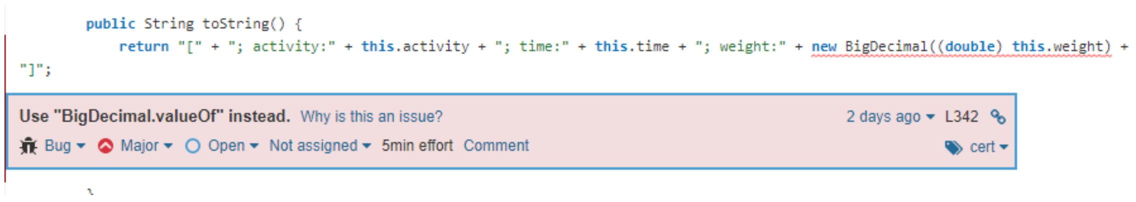
Return empty string instead. Why is this an issue? 2 days ago L501
Bug Major Open Not assigned 5min effort Comment cert, cwe

Figure 12 Don't Return Null

File: androidx/appcompat/view/menu/MenuItemImpl.java

9. **The phrase "BigDecimal(double)" should not be used.**

   Because of floating point imprecision, the BigDecimal(double) constructor is unlikely to return the expected value.



```
        public String toString() {
            return "[" + "; activity:" + this.activity + "; time:" + this.time + "; weight:" + new BigDecimal((double) this.weight) +
"]";
```

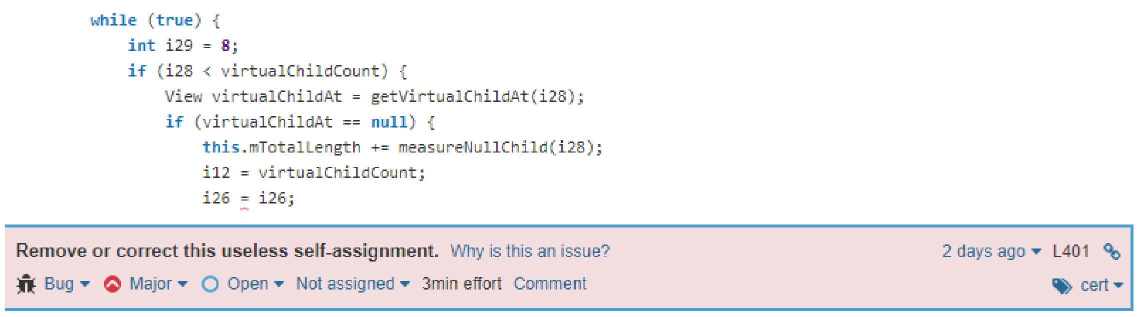
Use "BigDecimal.valueOf" instead. Why is this an issue? 2 days ago L342
Bug Major Open Not assigned 5min effort Comment cert

Figure 13 Don't Use BigDecimal

File: androidx/appcompat/widget/ActivityChooserModel.java

10. Variables should not be allocated to themselves.

There's no purpose to reassign a variable to itself. This statement is either unnecessary and should be removed, or the re-assignment is an error, and another value or variable should have been assigned instead.



```
while (true) {
    int i29 = 8;
    if (i28 < virtualChildCount) {
        View virtualChildAt = getVirtualChildAt(i28);
        if (virtualChildAt == null) {
            this.mTotalLength += measureNullChild(i28);
            i12 = virtualChildCount;
            i26 = i26;
```

Remove or correct this useless self-assignment. Why is this an issue? 2 days ago L401
Bug Major Open Not assigned 3min effort Comment cert

Figure 14 Useless Self-Assignment

File: androidx/appcompat/widget/LinearLayoutCompat.java

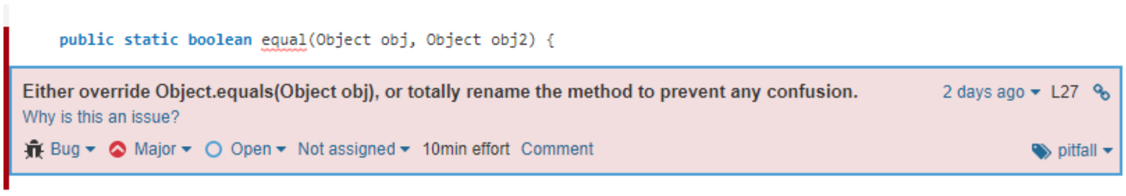
11. Methods should not be named "tostring", "hashcode" or "equal"

A method named tostring(), hashcode(), or equal() is either:

• An error in the form of a typo. The program does not behave as expected when overriding toString(), Object.hashCode() notice the camelCasing, or Object.equals (note the's' at the end).

• Intentionally done. The name, on the other hand, may confuse every other developer, who may not notice the naming change or will mistake it for a problem.

The method should be renamed in both circumstances.



```
public static boolean equal(Object obj, Object obj2) {
```

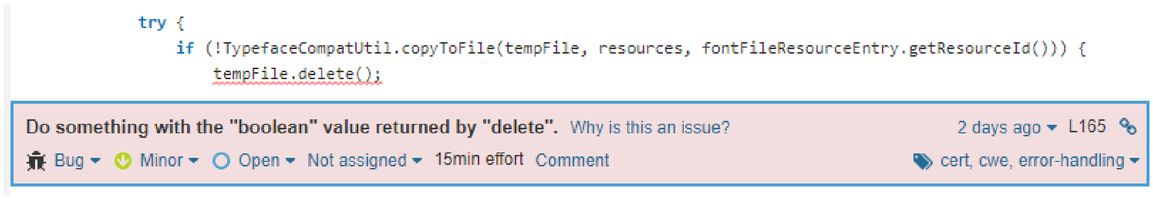
Either override Object.equals(Object obj), or totally rename the method to prevent any confusion. 2 days ago L27
Why is this an issue?
Bug Major Open Not assigned 10min effort Comment pitfall

Figure 15 Rename the Method

File: androidx/collection/ContainerHelpers.java

12. Return values containing the operation status code should not be ignored.

When the return value of a function call includes the operation status code, this value should be validated to ensure that the operation was successful.



```
try {
    if (!TypefaceCompatUtil.copyToFile(tempFile, resources, fontFileResourceEntry.getResourceId())) {
        tempFile.delete();
```

Do something with the "boolean" value returned by "delete". Why is this an issue? 2 days ago L165

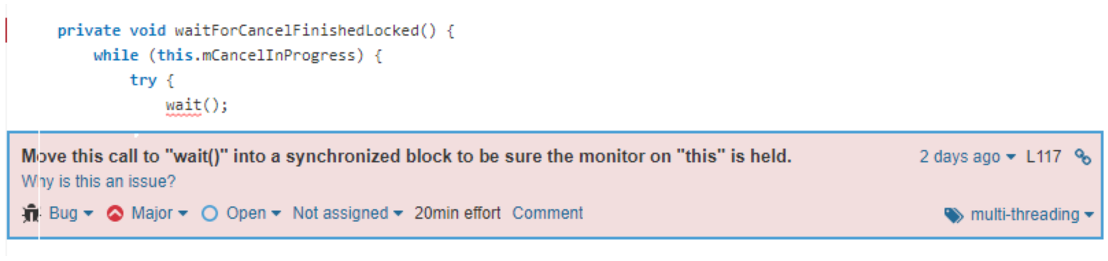
Bug Minor Open Not assigned 15min effort Comment cert, cwe, error-handling

Figure 16 Don't Ignore Return Value

File: androidx/core/graphics/TypefaceCompatApi21Impl.java

13. "wait," "notify," and "notifyAll" should be used only when an item clearly has a lock on it.

The methods Object.wait(...), Object.notify(), and Object.notifyAll() must be called by the thread that owns the object's monitor. An IllegalMonitorStateException exception is thrown if this is not the case. This rule enforces this constraint by requiring that one of these methods be called only within a synchronized method or statement.



```
private void waitForCancelFinishedLocked() {
    while (this.mCancelInProgress) {
        try {
            wait();
```

Move this call to "wait()" into a synchronized block to be sure the monitor on "this" is held. 2 days ago L117

Why is this an issue?

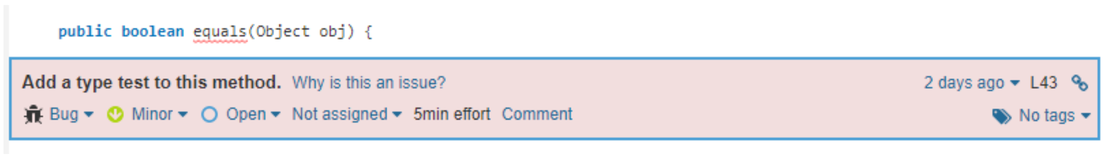
Bug Major Open Not assigned 20min effort Comment multi-threading

Figure 17 Call "wait()" Function

File: androidx/core/os/CancellationSignal.java

14. "equals (Object obj)" should be used to test the argument type.

Because the equals method accepts a generic Object as a parameter, it can accept any type of object. The method should not be written with the expectation that it would only be used to test objects of the same class type. It must instead examine the type of the parameter.



```
    public boolean equals(Object obj) {
```

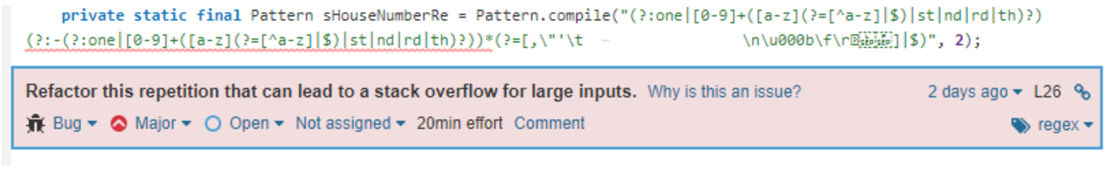
Add a type test to this method. Why is this an issue? 2 days ago L43
Bug Minor Open Not assigned 5min effort Comment No tags

Figure 18 Add Type Test

File: androidx/core/os/LocaleListPlatformWrapper.java

15. Regular expressions should not cause the stack to overflow.

To perform backtracking, the Java regex engine employs recursive method calls. Attempting to match a regular expression that comprises numerous pathways on large inputs, (i.e., the body of the repetition comprises an alternation (|), an optional element, or another repetition) might cause a stack overflow. This does not occur when a possessive quantifier is used (for example, *+ instead of *) or when a character class is used within a repeat (for example, [ab]* instead of (a|b)*).



```
    private static final Pattern sHouseNumberRe = Pattern.compile("(?:one|[0-9]+([a-z](?=[^a-z]|$)|st|nd|rd|th)?)
(?:-(?:one|[0-9]+([a-z](?=[^a-z]|$)|st|nd|rd|th)?))*(?=[,\"'\t              \n\u000b\f\r]|$)", 2);
```

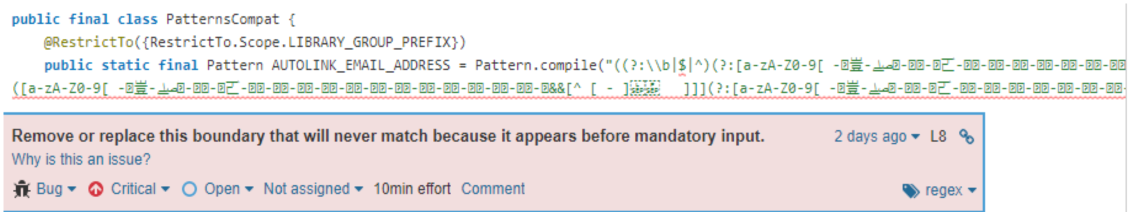
Refactor this repetition that can lead to a stack overflow for large inputs. Why is this an issue? 2 days ago L26
Bug Major Open Not assigned 20min effort Comment regex

Figure 19 Stack Overflow

File: androidx/core/text/util/FindAddress.java

16. **Regex borders should not be used in such a way that they cannot be matched.**

Regular expression borders and A can only match at the beginning of the input (or, if combined with the MULTILINE flag, at the beginning of the line) and $, Z, and z can only match at the end. These patterns can be abused by inadvertently switching and $, for example, to produce a pattern that will never match.



```
public final class PatternsCompat {
    @RestrictTo({RestrictTo.Scope.LIBRARY_GROUP_PREFIX})
    public static final Pattern AUTOLINK_EMAIL_ADDRESS = Pattern.compile("((?:\\b|$|^)(?:[a-zA-Z0-9[ -[illegible]
([a-zA-Z0-9[ -[illegible]&&[^ [ - ][illegible]  ]]](?:[a-zA-Z0-9[ -[illegible]
```

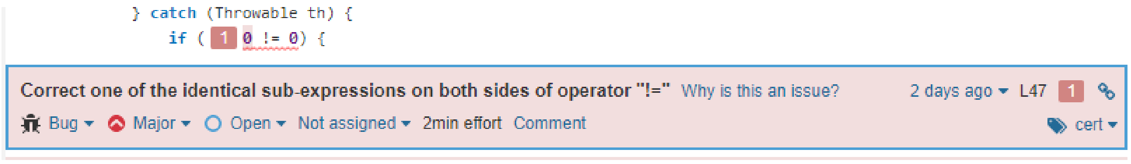
Remove or replace this boundary that will never match because it appears before mandatory input. 2 days ago L8
Why is this an issue?
Bug Critical Open Not assigned 10min effort Comment regex

Figure 20 Remove Boundary

File: androidx/core/util/PatternsCompat.java

17. **Before assigning math operands, they should be cast.**

When you perform arithmetic on integers, the result is always an integer. You can use automatic type conversion to convert that result to a long, double, or float, but because it originated as an int or long, the outcome will most likely not be what you anticipate. For example, if the result of int division is assigned to a floating-point variable, precision is lost prior to the assignment. Similarly, if the multiplication result is assigned to a long, it may already have overflowed prior to the assignment. In any situation, the outcome will be unexpected. Instead, before the operation, at least one operand should be cast or promoted to the final type.



```
        } catch (Throwable th) {
            if ( 1 0 != 0) {
```

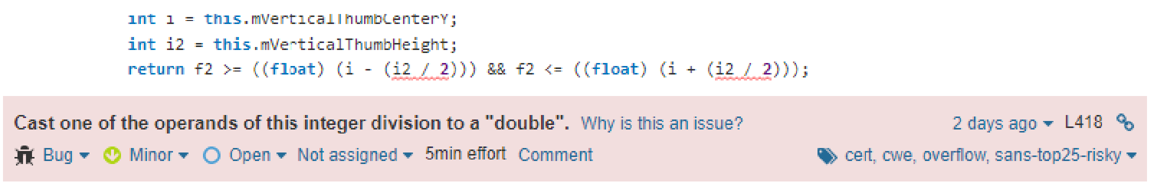
Correct one of the identical sub-expressions on both sides of operator "!=" Why is this an issue? 2 days ago L47 1
Bug Major Open Not assigned 2min effort Comment cert

Figure 21Avoid Both Side Binary Operator

File: androidx/recyclerview/widget/FastScroller.java

18. Identical phrases on both sides of a binary operator should be avoided.

It is nearly always a mistake to use the same value on both sides of a binary operator. In the case of logical operators, it is either a copy/paste error and hence a problem, or it is just unnecessary code that should be streamlined. Having the same value on both sides of a bitwise operator and most binary mathematical operators provides predictable results and should be simplified.



```
int i = this.mVerticalThumbCenterY;
int i2 = this.mVerticalThumbHeight;
return f2 >= ((float) (i - (i2 / 2))) && f2 <= ((float) (i + (i2 / 2)));
```

Cast one of the operands of this integer division to a "double". Why is this an issue? 2 days ago L418
Bug Minor Open Not assigned 5min effort Comment cert, cwe, overflow, sans-top25-risky

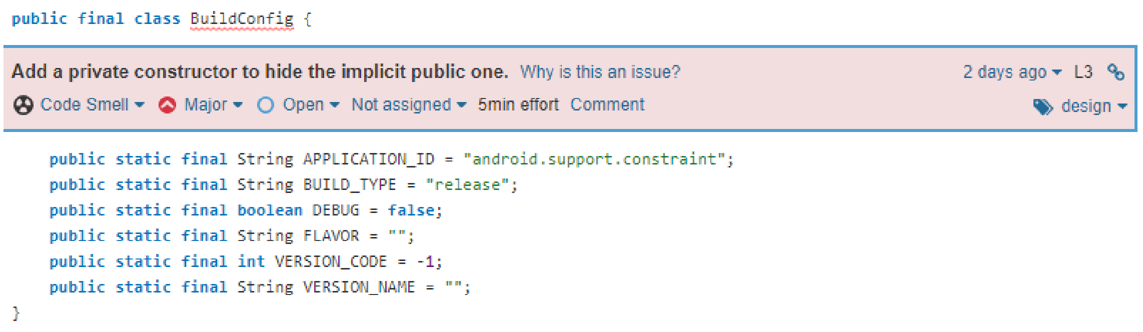
Figure 22 Cast Math Operand

File: androidx/vectordrawable/graphics/drawable/AnimationUtilsCompat.java

3.1.3 Code Smells

1. There should be no public constructors in utility classes.

Utility classes are collections of static members that are not intended to be instantiated. Public constructors should be avoided even in abstract utility classes that can be enhanced. Java adds an implicit public constructor to every class that does not explicitly declare at least one. As a result, at least one non-public constructor must be defined.



```
public final class BuildConfig {
```

Add a private constructor to hide the implicit public one. Why is this an issue? 2 days ago L3
Code Smell Major Open Not assigned 5min effort Comment design

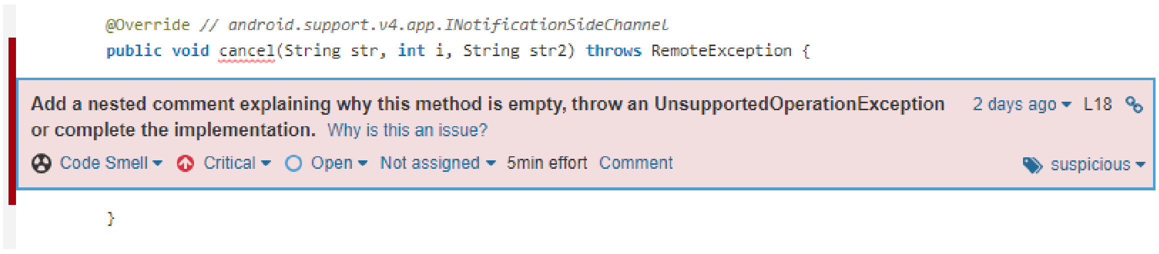
```
    public static final String APPLICATION_ID = "android.support.constraint";
    public static final String BUILD_TYPE = "release";
    public static final boolean DEBUG = false;
    public static final String FLAVOR = "";
    public static final int VERSION_CODE = -1;
    public static final String VERSION_NAME = "";
}
```

Figure 23 No Public Constructor

File: android/support/constraint/BuildConfig.java

2. **Methods should not be left blank.**

There are various reasons why a method lacks a method body:
• It is an inadvertent omission that should be corrected to avoid unexpected behavior in production.
• It is not yet, and will never be, supported. An UnsupportedOperationException should be thrown in this situation.
• The method is a purposefully blank override. A nested remark should explain the reason for the blank override in this scenario.



```
@Override // android.support.v4.app.INotificationSideChannel
public void cancel(String str, int i, String str2) throws RemoteException {
```

Add a nested comment explaining why this method is empty, throw an UnsupportedOperationException or complete the implementation. Why is this an issue? 2 days ago L18

Code Smell Critical Open Not assigned 5min effort Comment suspicious

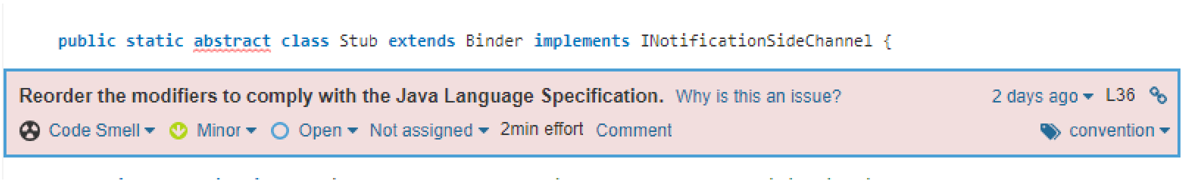
```
}
```

Figure 24 Don't Leave a Blank Method

File: android/support/v4/app/INotificationSideChannel.java

3. **Modifiers must be declared in the proper sequence.**

In Java, a static modifier should be declared after an abstract modifier, thus they are not following this convention in our situation, which has no technical impact but will degrade code readability because most developers are used to the conventional order.



```
public static abstract class Stub extends Binder implements INotificationSideChannel {
```

Reorder the modifiers to comply with the Java Language Specification. Why is this an issue? 2 days ago L36

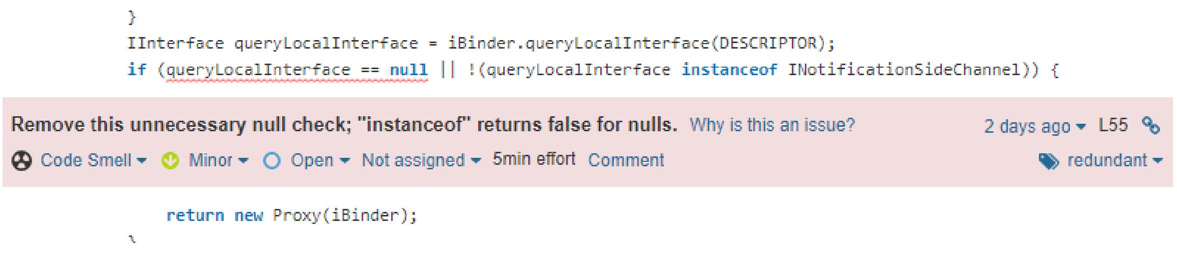
Code Smell Minor Open Not assigned 2min effort Comment convention

Figure 25 Modifiers with Java Language Specification

File: android/support/v4/app/INotificationSideChannel.java

4. "instanceof" should not be used with null tests.

There is no need to use null test with an instanceof test. Because null is not an instance of anything, a null check is unnecessary.



```
}
IInterface queryLocalInterface = iBinder.queryLocalInterface(DESCRIPTOR);
if (queryLocalInterface == null || !(queryLocalInterface instanceof INotificationSideChannel)) {
```

Remove this unnecessary null check; "instanceof" returns false for nulls. Why is this an issue? 2 days ago L55
Code Smell Minor Open Not assigned 5min effort Comment redundant

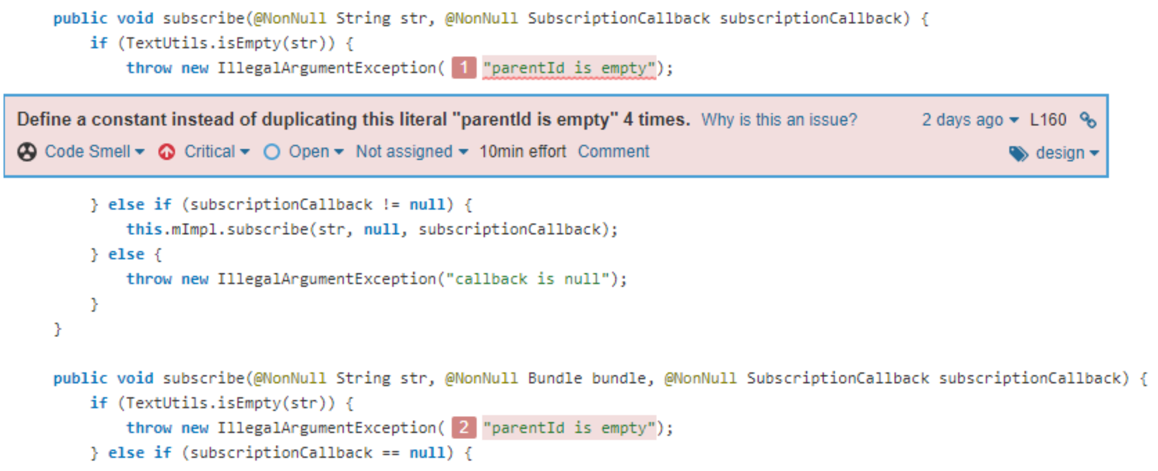
```
    return new Proxy(iBinder);
```

Figure 26 Remove Unnecessary Null Check

File: android/support/v4/app/INotificationSideChannel.java

5. String literals should not be reused.

Duplicated string literals make refactoring more error-prone because you must ensure that all occurrences are updated. Constants, on the other hand, can be referenced from multiple places but only need to be modified once.



```
public void subscribe(@NonNull String str, @NonNull SubscriptionCallback subscriptionCallback) {
    if (TextUtils.isEmpty(str)) {
        throw new IllegalArgumentException( 1 "parentId is empty");
```

Define a constant instead of duplicating this literal "parentId is empty" 4 times. Why is this an issue? 2 days ago L160
Code Smell Critical Open Not assigned 10min effort Comment design

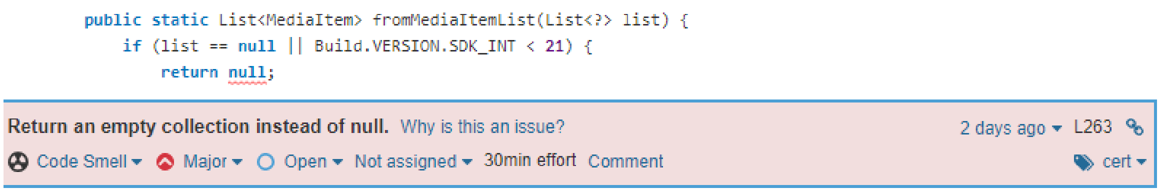
```
    } else if (subscriptionCallback != null) {
        this.mImpl.subscribe(str, null, subscriptionCallback);
    } else {
        throw new IllegalArgumentException("callback is null");
    }
}

public void subscribe(@NonNull String str, @NonNull Bundle bundle, @NonNull SubscriptionCallback subscriptionCallback) {
    if (TextUtils.isEmpty(str)) {
        throw new IllegalArgumentException( 2 "parentId is empty");
    } else if (subscriptionCallback == null) {
```

Figure 27 Don't Duplicate Strings

File: android/support/v4/media/MediaBrowserCompat.java

6. **Instead of returning null, empty arrays and collections should be returned.**

   Returning null instead of an array, collection, or map causes method caller to explicitly test for nullity, making them more complex and difficult to read. Furthermore, null is frequently used as a synonym for empty.



```
public static List<MediaItem> fromMediaItemList(List<?> list) {
    if (list == null || Build.VERSION.SDK_INT < 21) {
        return null;
```

Return an empty collection instead of null. Why is this an issue? 2 days ago L263

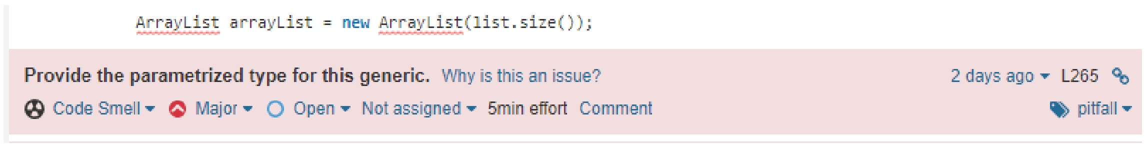
Code Smell Major Open Not assigned 30min effort Comment cert

Figure 28 Return Empty Instead Of Null

File: android/support/v4/media/MediaBrowserCompat.java

7. **No raw types should be utilized.**

   Generic types should not be used in variable declarations or return values in their raw form (without type parameters). This avoids generic type checking and postpones the detection of dangerous code until runtime.



```
ArrayList arrayList = new ArrayList(list.size());
```

Provide the parametrized type for this generic. Why is this an issue? 2 days ago L265

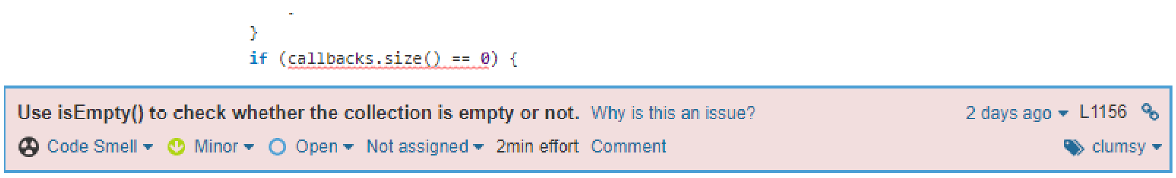
Code Smell Major Open Not assigned 5min effort Comment pitfall

Figure 29 Don't Use Raw Type

File: android/support/v4/media/MediaBrowserCompat.java

8. To check for emptiness, use Collection.isEmpty().

It is possible to test for emptiness with Collection.size() but using Collection.isEmpty() makes the code clearer and may be more performant. Any implementation of the isEmpty() method should have a time complexity of O(1), although certain size() implementations can have a time complexity of O(2) (n).



```
}
if (callbacks.size() == 0) {
```

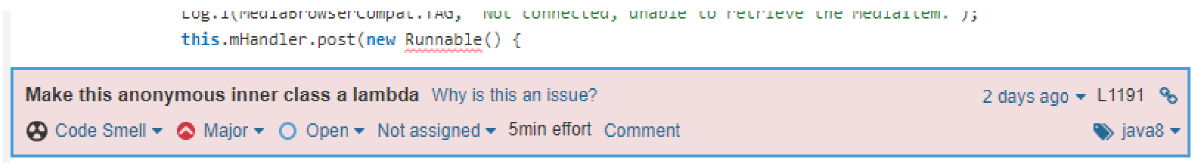
Use isEmpty() to check whether the collection is empty or not. Why is this an issue? 2 days ago L1156
Code Smell Minor Open Not assigned 2min effort Comment clumsy

Figure 30 Use isEmpty() to Check

File: android/support/v4/media/MediaBrowserCompat.java

9. Anonymous inner classes with only one method should be renamed lambdas.

Prior to Java 8, the only way to implement closures in Java was to use anonymous inner classes. However, the syntax of anonymous classes may appear cumbersome and ambiguous. To dramatically increase source code readability, most uses of anonymous inner classes in Java 8 should be replaced by lambdas.



```
Log.i(MediaBrowserCompat.TAG, "Not connected, unable to retrieve the MediaItem.");
this.mHandler.post(new Runnable() {
```

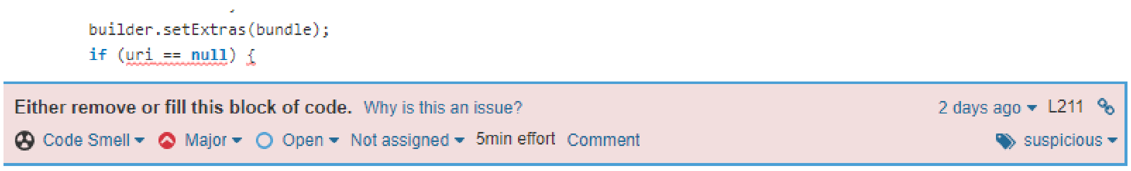
Make this anonymous inner class a lambda Why is this an issue? 2 days ago L1191
Code Smell Major Open Not assigned 5min effort Comment java8

Figure 31 Anonymous Inner Class

File: android/support/v4/media/MediaBrowserCompat.java

10. Nested code blocks should not be left unfilled.

When a piece of code is truly missing, most of the time a block of code is empty. As a result, such an empty block must be either filled or eliminated.



```
builder.setExtras(bundle);
if (uri == null) {
```

Either remove or fill this block of code. Why is this an issue? 2 days ago L211

Code Smell Major Open Not assigned 5min effort Comment suspicious

Figure 32 Don't leave Empty Block

File: android/support/v4/media/MediaDescriptionCompat.java

3.2 Clean Code

clean code is the code that any developer can read and change easily.

We will discuss the cleanliness of the code for the Alinma Bank application code

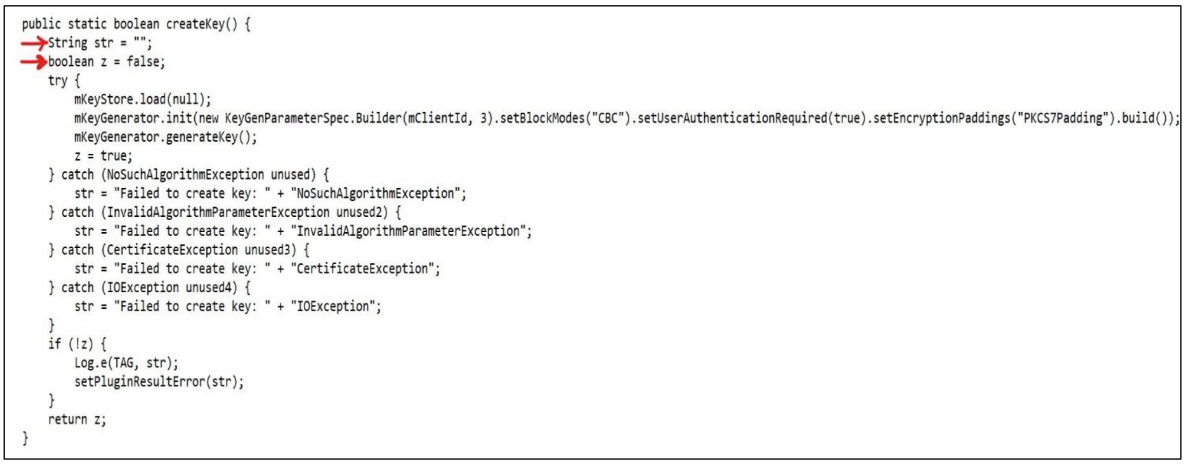
1. **Use coherent indentation:**

   By reviewing the code, we see that it uses coherent indentation, which is good. This makes it easier for other programmers to read the code.

2. **The identifier should be meaningful:**

   There are many meaningless identifiers, it is not telling the reader why it exists, what it does, and how it is used.

   To make clean code, we should select a meaningful name for identifier, the name that you specify in your code should reveal its intent. It should specify what is the purpose of a variable.



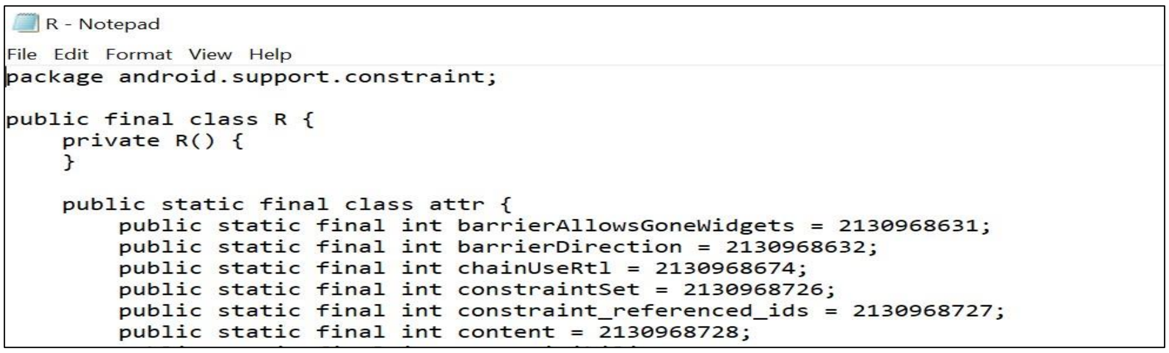
```
public static boolean createKey() {
    String str = "";
    boolean z = false;
    try {
        mKeyStore.load(null);
        mKeyGenerator.init(new KeyGenParameterSpec.Builder(mClientId, 3).setBlockModes("CBC").setUserAuthenticationRequired(true).setEncryptionPaddings("PKCS7Padding").build());
        mKeyGenerator.generateKey();
        z = true;
    } catch (NoSuchAlgorithmException unused) {
        str = "Failed to create key: " + "NoSuchAlgorithmException";
    } catch (InvalidAlgorithmParameterException unused2) {
        str = "Failed to create key: " + "InvalidAlgorithmParameterException";
    } catch (CertificateException unused3) {
        str = "Failed to create key: " + "CertificateException";
    } catch (IOException unused4) {
        str = "Failed to create key: " + "IOException";
    }
    if (!z) {
        Log.e(TAG, str);
        setPluginResultError(str);
    }
    return z;
}
```

Figure 33 Identifier Should be Meaningful

File: corova/plugin/android/fingerprintauth/FingerprintAuth.java

3. The class name should be meaningful:

There are many classes whose name are meaningless; it does not tell the reader it was written for what purpose which this affects the quality of the code.


R - Notepad

File Edit Format View Help

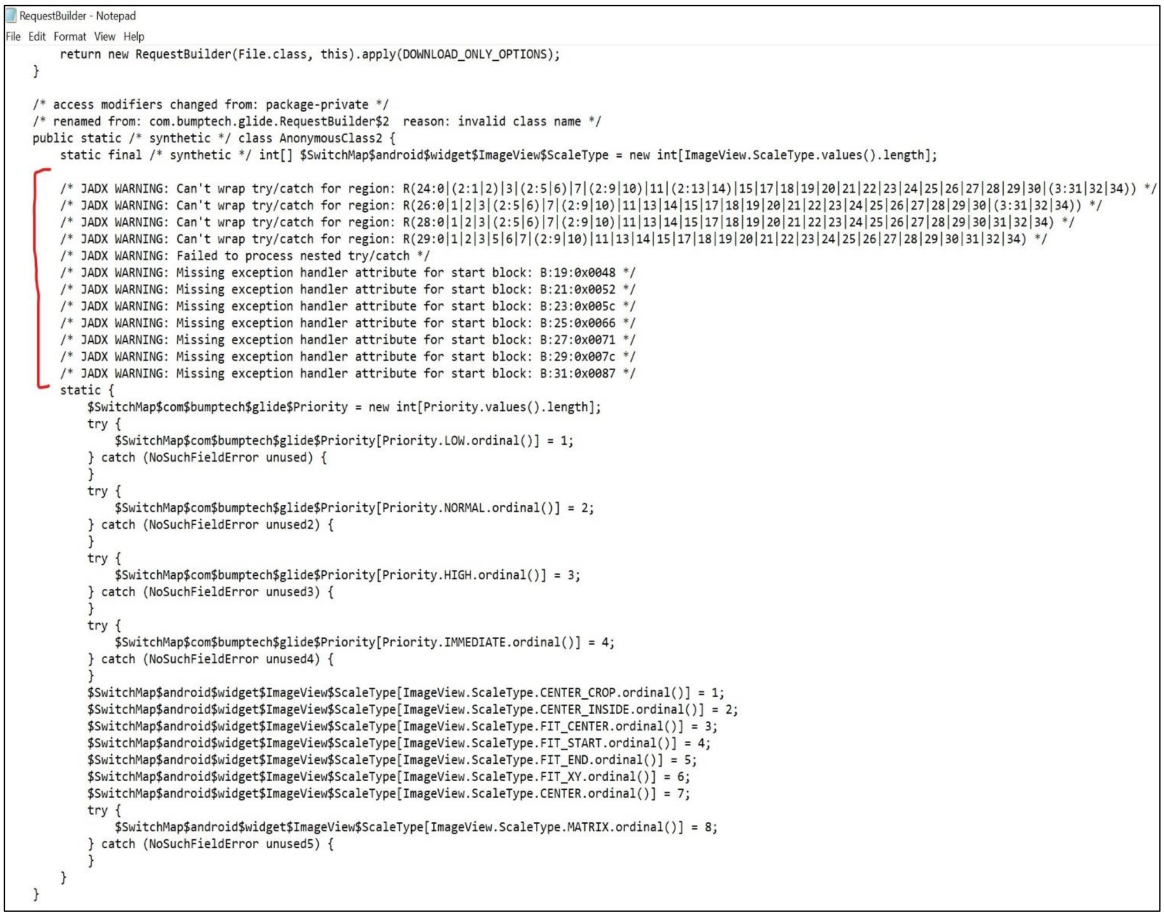
```
package android.support.constraint;

public final class R {
    private R() {
    }

    public static final class attr {
        public static final int barrierAllowsGoneWidgets = 2130968631;
        public static final int barrierDirection = 2130968632;
        public static final int chainUseRtl = 2130968674;
        public static final int constraintSet = 2130968726;
        public static final int constraint_referenced_ids = 2130968727;
        public static final int content = 2130968728;
```

Figure 34 Class Name Should be Meaningful

File: android/support/constraint/R.java

4. Vertical Density:

vertical density means lines of code that close association it should appear vertically dense, close to each other.
In the figure below because of the presence of the comment, it was not easy to read, it was taken a little of time. I have to move my eyes and my head to understand. It is better to make it interconnected, not separated by a comment.



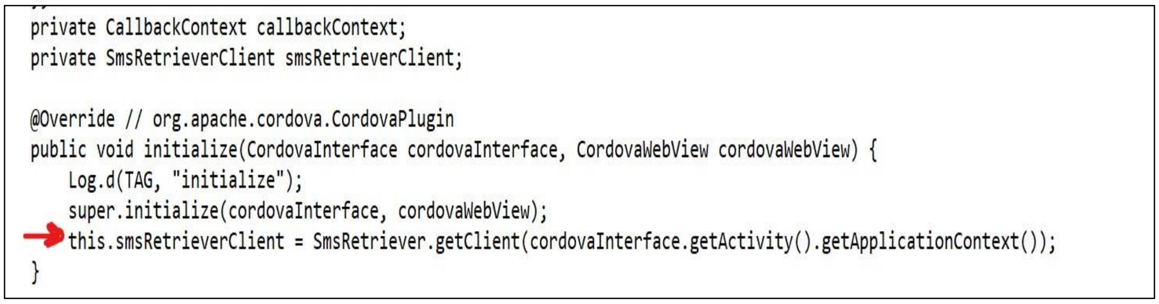
```
        return new RequestBuilder(File.class, this).apply(DOWNLOAD_ONLY_OPTIONS);
    }

    /* access modifiers changed from: package-private */
    /* renamed from: com.bumptech.glide.RequestBuilder$2  reason: invalid class name */
    public static /* synthetic */ class AnonymousClass2 {
        static final /* synthetic */ int[] $SwitchMap$android$widget$ImageView$ScaleType = new int[ImageView.ScaleType.values().length];

        /* JADX WARNING: Can't wrap try/catch for region: R(24:0|(2:1|2)|3|(2:5|6)|7|(2:9|10)|11|(2:13|14)|15|17|18|19|20|21|22|23|24|25|26|27|28|29|30|(3:31|32|34)) */
        /* JADX WARNING: Can't wrap try/catch for region: R(26:0|1|2|3|(2:5|6)|7|(2:9|10)|11|13|14|15|17|18|19|20|21|22|23|24|25|26|27|28|29|30|(3:31|32|34)) */
        /* JADX WARNING: Can't wrap try/catch for region: R(28:0|1|2|3|(2:5|6)|7|(2:9|10)|11|13|14|15|17|18|19|20|21|22|23|24|25|26|27|28|29|30|31|32|34) */
        /* JADX WARNING: Can't wrap try/catch for region: R(29:0|1|2|3|5|6|7|(2:9|10)|11|13|14|15|17|18|19|20|21|22|23|24|25|26|27|28|29|30|31|32|34) */
        /* JADX WARNING: Failed to process nested try/catch */
        /* JADX WARNING: Missing exception handler attribute for start block: B:19:0x0048 */
        /* JADX WARNING: Missing exception handler attribute for start block: B:21:0x0052 */
        /* JADX WARNING: Missing exception handler attribute for start block: B:23:0x005c */
        /* JADX WARNING: Missing exception handler attribute for start block: B:25:0x0066 */
        /* JADX WARNING: Missing exception handler attribute for start block: B:27:0x0071 */
        /* JADX WARNING: Missing exception handler attribute for start block: B:29:0x007c */
        /* JADX WARNING: Missing exception handler attribute for start block: B:31:0x0087 */
        static {
            $SwitchMap$com$bumptech$glide$Priority = new int[Priority.values().length];
            try {
                $SwitchMap$com$bumptech$glide$Priority[Priority.LOW.ordinal()] = 1;
            } catch (NoSuchFieldError unused) {
            }
            try {
                $SwitchMap$com$bumptech$glide$Priority[Priority.NORMAL.ordinal()] = 2;
            } catch (NoSuchFieldError unused2) {
            }
            try {
                $SwitchMap$com$bumptech$glide$Priority[Priority.HIGH.ordinal()] = 3;
            } catch (NoSuchFieldError unused3) {
            }
            try {
                $SwitchMap$com$bumptech$glide$Priority[Priority.IMMEDIATE.ordinal()] = 4;
            } catch (NoSuchFieldError unused4) {
            }
            $SwitchMap$android$widget$ImageView$ScaleType[ImageView.ScaleType.CENTER_CROP.ordinal()] = 1;
            $SwitchMap$android$widget$ImageView$ScaleType[ImageView.ScaleType.CENTER_INSIDE.ordinal()] = 2;
            $SwitchMap$android$widget$ImageView$ScaleType[ImageView.ScaleType.FIT_CENTER.ordinal()] = 3;
            $SwitchMap$android$widget$ImageView$ScaleType[ImageView.ScaleType.FIT_START.ordinal()] = 4;
            $SwitchMap$android$widget$ImageView$ScaleType[ImageView.ScaleType.FIT_END.ordinal()] = 5;
            $SwitchMap$android$widget$ImageView$ScaleType[ImageView.ScaleType.FIT_XY.ordinal()] = 6;
            $SwitchMap$android$widget$ImageView$ScaleType[ImageView.ScaleType.CENTER.ordinal()] = 7;
            try {
                $SwitchMap$android$widget$ImageView$ScaleType[ImageView.ScaleType.MATRIX.ordinal()] = 8;
            } catch (NoSuchFieldError unused5) {
            }
        }
    }
```

Figure 35 Vertical Density

File: com/bumptech/glide/RequestBuilder.java

5. The Law of Demeter:

The Law of Demeter for methods attempts to minimize coupling between classes. They should write a code with “loosely coupled” classes and reduce dependencies which make reusing the classes easier, this will lead fewer errors.



```
private CallbackContext callbackContext;
private SmsRetrieverClient smsRetrieverClient;

@Override // org.apache.cordova.CordovaPlugin
public void initialize(CordovaInterface cordovaInterface, CordovaWebView cordovaWebView) {
    Log.d(TAG, "initialize");
    super.initialize(cordovaInterface, cordovaWebView);
    this.smsRetrieverClient = SmsRetriever.getClient(cordovaInterface.getActivity().getApplicationContext());
}
```

Figure 36 Law of Demeter

File: com/andreszs/smsretriever/SMSRetriever.java

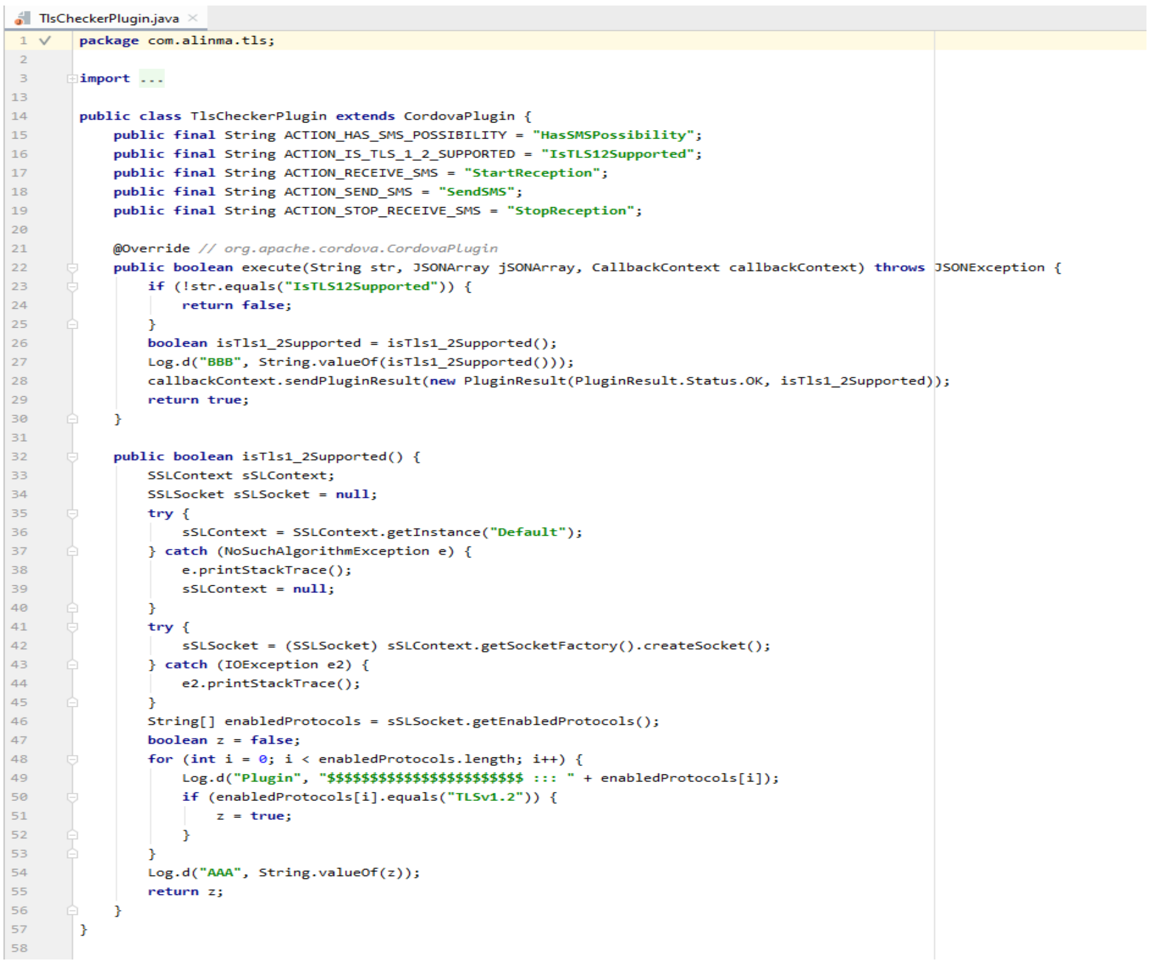
3.3 Code Documentation

Code documentation is the thing that helps a person to understand the code. And not refer to the comments on the code only, it can come as a separate file containing a description of the code, how code works, what the code constraints, what is the input and output and several examples about how to run it like README file.

To help the developer and save his time, multiple of the tools appear that can create the document that can be published to public people, but these tools put some simple standards to write the comments on the code.

We checked the Alinma Bank application code after converting the APK of the application, we found most of the code has no comments and it didn't follow any community standards of writing the comments on the code.

And below are examples of the code without comments:

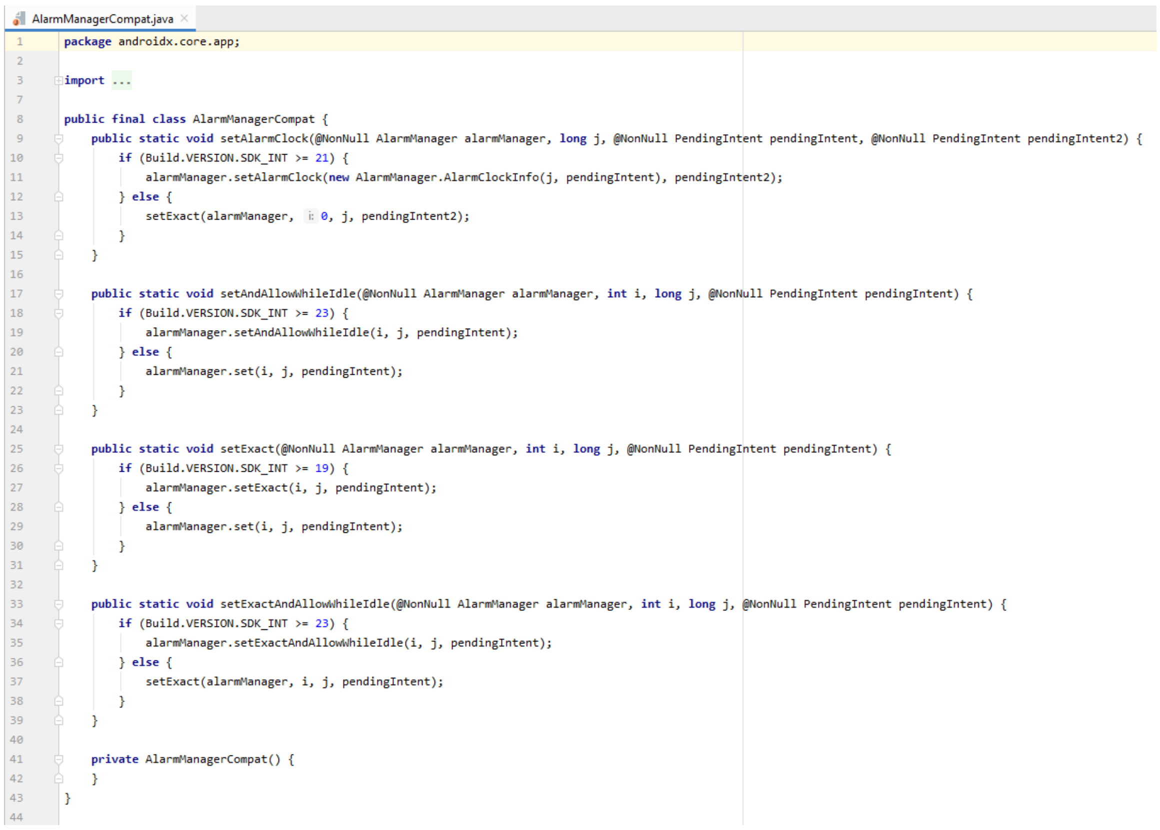


TlsCheckerPlugin.java

```java
package com.alinma.tls;

import ...

public class TlsCheckerPlugin extends CordovaPlugin {
    public final String ACTION_HAS_SMS_POSSIBILITY = "HasSMSPossibility";
    public final String ACTION_IS_TLS_1_2_SUPPORTED = "IsTLS12Supported";
    public final String ACTION_RECEIVE_SMS = "StartReception";
    public final String ACTION_SEND_SMS = "SendSMS";
    public final String ACTION_STOP_RECEIVE_SMS = "StopReception";

    @Override // org.apache.cordova.CordovaPlugin
    public boolean execute(String str, JSONArray jSONArray, CallbackContext callbackContext) throws JSONException {
        if (!str.equals("IsTLS12Supported")) {
            return false;
        }
        boolean isTls1_2Supported = isTls1_2Supported();
        Log.d("BBB", String.valueOf(isTls1_2Supported()));
        callbackContext.sendPluginResult(new PluginResult(PluginResult.Status.OK, isTls1_2Supported));
        return true;
    }

    public boolean isTls1_2Supported() {
        SSLContext sSLContext;
        SSLSocket sSLSocket = null;
        try {
            sSLContext = SSLContext.getInstance("Default");
        } catch (NoSuchAlgorithmException e) {
            e.printStackTrace();
            sSLContext = null;
        }
        try {
            sSLSocket = (SSLSocket) sSLContext.getSocketFactory().createSocket();
        } catch (IOException e2) {
            e2.printStackTrace();
        }
        String[] enabledProtocols = sSLSocket.getEnabledProtocols();
        boolean z = false;
        for (int i = 0; i < enabledProtocols.length; i++) {
            Log.d("Plugin", "$$$$$$$$$$$$$$$$$$$$$$$$ ::: " + enabledProtocols[i]);
            if (enabledProtocols[i].equals("TLSv1.2")) {
                z = true;
            }
        }
        Log.d("AAA", String.valueOf(z));
        return z;
    }
}
```

Figure 37 Example of the Code Without Comments #1

File: com\alinma\tls


AlarmManagerCompat.java

```java
package androidx.core.app;

import ...

public final class AlarmManagerCompat {
    public static void setAlarmClock(@NonNull AlarmManager alarmManager, long j, @NonNull PendingIntent pendingIntent, @NonNull PendingIntent pendingIntent2) {
        if (Build.VERSION.SDK_INT >= 21) {
            alarmManager.setAlarmClock(new AlarmManager.AlarmClockInfo(j, pendingIntent), pendingIntent2);
        } else {
            setExact(alarmManager, i: 0, j, pendingIntent2);
        }
    }

    public static void setAndAllowWhileIdle(@NonNull AlarmManager alarmManager, int i, long j, @NonNull PendingIntent pendingIntent) {
        if (Build.VERSION.SDK_INT >= 23) {
            alarmManager.setAndAllowWhileIdle(i, j, pendingIntent);
        } else {
            alarmManager.set(i, j, pendingIntent);
        }
    }

    public static void setExact(@NonNull AlarmManager alarmManager, int i, long j, @NonNull PendingIntent pendingIntent) {
        if (Build.VERSION.SDK_INT >= 19) {
            alarmManager.setExact(i, j, pendingIntent);
        } else {
            alarmManager.set(i, j, pendingIntent);
        }
    }

    public static void setExactAndAllowWhileIdle(@NonNull AlarmManager alarmManager, int i, long j, @NonNull PendingIntent pendingIntent) {
        if (Build.VERSION.SDK_INT >= 23) {
            alarmManager.setExactAndAllowWhileIdle(i, j, pendingIntent);
        } else {
            setExact(alarmManager, i, j, pendingIntent);
        }
    }

    private AlarmManagerCompat() {
    }
}
```

Figure 38 Example of the Code Without Comments #2

File: androidx\core\app

Since they write the code with Java language, we advise them to follow a standards of comments Javadoc to be easy for them to extract a nice document that explains their code, or at least they can write a README file that contains an overview of the code.

3.4 Design Analysis

3.4.1 External design:

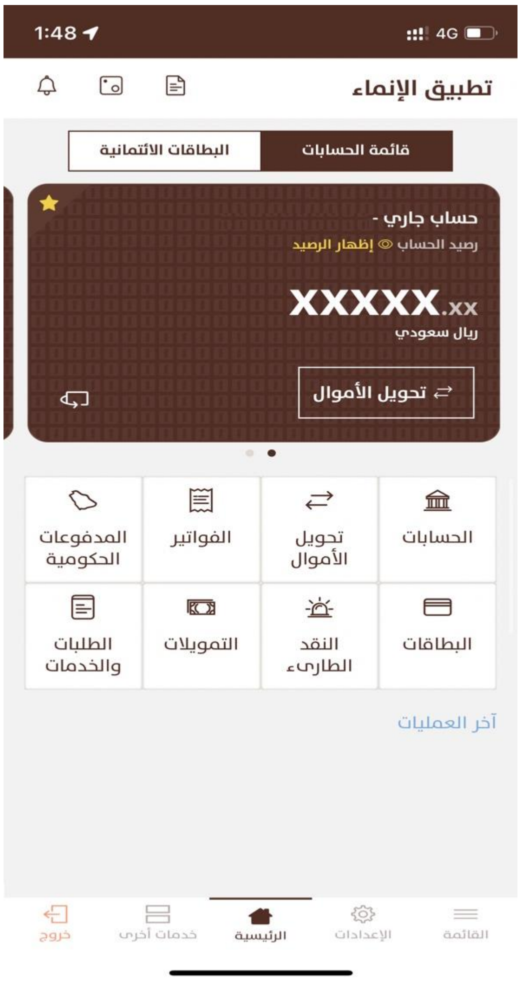


Figure 39 Alinma Bank Interface

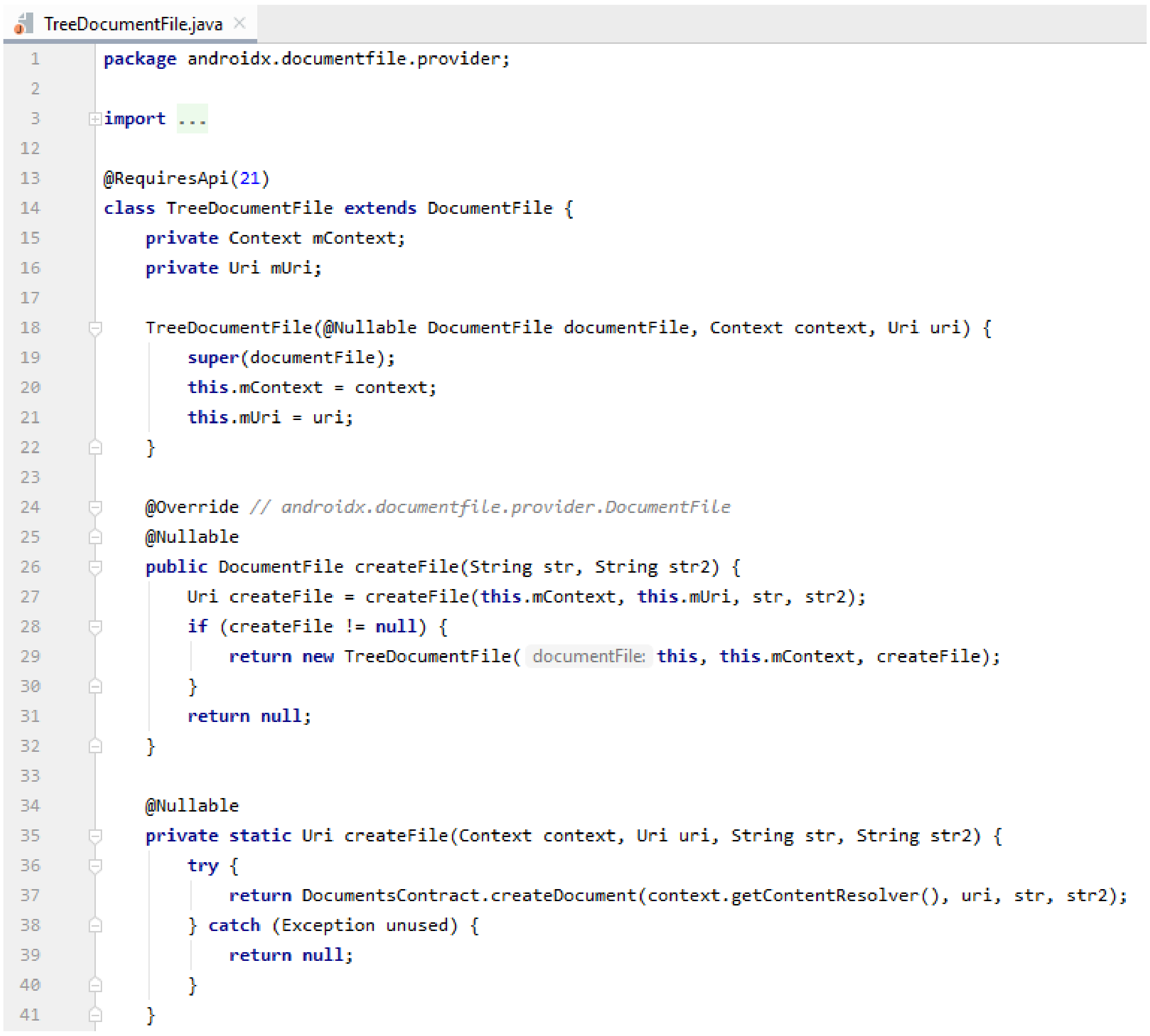
A user's experience depends on the design of the interface, navigation, and features of an app.

This is evident in Alinma bank application. Each button on the application is clearly labeled and the interface is intuitive. In addition, the most important information can be viewed on the front, making it easier to navigate and more accessible. And the account balance can be hidden with the app, which is another cool feature.

3.4.2 Internal design

According to our analysis, some of the classes have a high chance of being "god classes" which means they would have a high coupling problem meaning there is a lot of responsibilities on that class, making enhancement, unit testing, and re-usability challenging. One of them was file: “androidx/constraintlayout/widget/ConstraintLayout.java”

Another example of low coupling:


TreeDocumentFile.java

```java
package androidx.documentfile.provider;

import ...

@RequiresApi(21)
class TreeDocumentFile extends DocumentFile {
    private Context mContext;
    private Uri mUri;

    TreeDocumentFile(@Nullable DocumentFile documentFile, Context context, Uri uri) {
        super(documentFile);
        this.mContext = context;
        this.mUri = uri;
    }

    @Override // androidx.documentfile.provider.DocumentFile
    @Nullable
    public DocumentFile createFile(String str, String str2) {
        Uri createFile = createFile(this.mContext, this.mUri, str, str2);
        if (createFile != null) {
            return new TreeDocumentFile( documentFile: this, this.mContext, createFile);
        }
        return null;
    }

    @Nullable
    private static Uri createFile(Context context, Uri uri, String str, String str2) {
        try {
            return DocumentsContract.createDocument(context.getContentResolver(), uri, str, str2);
        } catch (Exception unused) {
            return null;
        }
    }
```

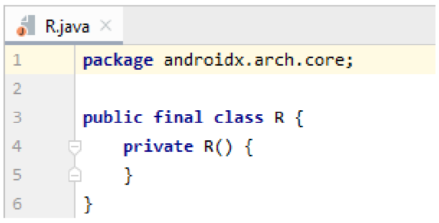
Figure 40 Example of Low Coupling

File: androidx\documentfile\provider

Another thing we found is the high and low cohesion. High cohesion is one of the good practices in writing any code since each class is going to focus on only one purpose, and this will make the class be easy to manage and maintain.

Furthermore, we also found an implementation of SRP through a "Data class", which leads to high cohesion.

Examples of high cohesion:

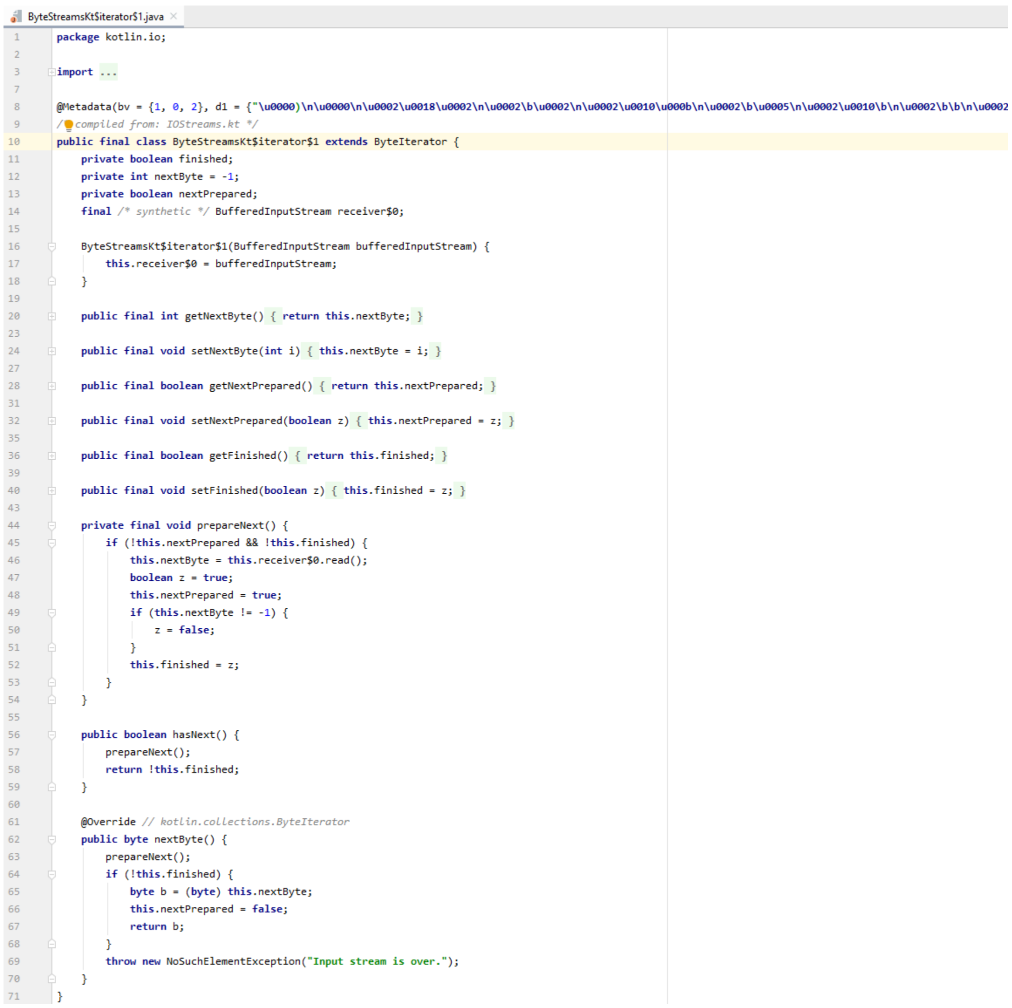


R.java

```java
package androidx.arch.core;

public final class R {
    private R() {
    }
}
```

Figure 41 Example of High Cohesion #1

File: androidx\arch\core



ByteStreamsKt$iterator$1.java

```java
package kotlin.io;

import ...

@Metadata(bv = {1, 0, 2}, d1 = {"\u0000)\n\u0000\n\u0002\u0018\u0002\n\u0002\b\u0002\n\u0002\u0010\u000b\n\u0002\b\u0005\n\u0002\u0010\b\n\u0002\b\b\n\u0002
/* compiled from: IOStreams.kt */
public final class ByteStreamsKt$iterator$1 extends ByteIterator {
    private boolean finished;
    private int nextByte = -1;
    private boolean nextPrepared;
    final /* synthetic */ BufferedInputStream receiver$0;

    ByteStreamsKt$iterator$1(BufferedInputStream bufferedInputStream) {
        this.receiver$0 = bufferedInputStream;
    }

    public final int getNextByte() { return this.nextByte; }

    public final void setNextByte(int i) { this.nextByte = i; }

    public final boolean getNextPrepared() { return this.nextPrepared; }

    public final void setNextPrepared(boolean z) { this.nextPrepared = z; }

    public final boolean getFinished() { return this.finished; }

    public final void setFinished(boolean z) { this.finished = z; }

    private final void prepareNext() {
        if (!this.nextPrepared && !this.finished) {
            this.nextByte = this.receiver$0.read();
            boolean z = true;
            this.nextPrepared = true;
            if (this.nextByte != -1) {
                z = false;
            }
            this.finished = z;
        }
    }

    public boolean hasNext() {
        prepareNext();
        return !this.finished;
    }

    @Override // kotlin.collections.ByteIterator
    public byte nextByte() {
        prepareNext();
        if (!this.finished) {
            byte b = (byte) this.nextByte;
            this.nextPrepared = false;
            return b;
        }
        throw new NoSuchElementException("Input stream is over.");
    }
}
```

Figure 42 Example of High Cohesion #2

File: kotlin/io/ByteStreamsKt$iterator$1.java

4 Conclusion

Overall, the results of Alinma Bank's application show an alarming statement, such as a CVSS score of Medium and a security score of High.

This should not be the case for a banking application since it's a critically important application. It is necessary to resolve issues such as logging of confidential information that shouldn't necessarily be logged or relying on cryptographic algorithms such as RC2/RC4/MD4/MD5 that are known to be insecure. The hashing can be performed by SHA-256 or another method.

As for the code itself, it should be documented to help the developers in the company to identify the purpose of the methods. It will help eliminate some of the issues found in code smells and bugs, such as redundant lines of code or empty methods as well as to decrease the coupling between classes.