

Lint Report


Check performed at Fri Oct 14 21:15:18 BRT 2016.


0 errors and 7 warnings found:

Security

1  AllowBackup: AllowBackup/FullBackupContent Problems

Performance

1  LogConditional: Unconditional Logging Calls

1  Overdraw: Overdraw: Painting regions more than once

3  UnusedResources: Unused resources

Usability

1  GoogleAppIndexingWarning: Missing support for Firebase App Indexing

Disabled Checks (18)

Security

AllowBackup: AllowBackup/FullBackupContent Problems

../src/main/AndroidManifest.xml:7: On SDK version 23 and up, your app data will be automatically backed up and restored on app install. Consider adding the attribute `android:fullBackupContent` to specify an `@xml` resource which configures which files to backup. More info: <https://developer.android.com/training/backup/autosyncapi.html>

```
4
5     <uses-permission android:name="android.permission.INTERNET"/>
6
7     <application
8         android:name=".App"
9         android:allowBackup="true"
```

Note: This issue has an associated quickfix operation in Android Studio/IntelliJ & Eclipse/ADT

Priority: 3 / 10

Category: Security

Severity: Warning

Explanation: AllowBackup/FullBackupContent Problems.

The allowBackup attribute determines if an application's data can be backed up and restored. It is documented at <http://developer.android.com/reference/android/R.attr.html#allowBackup>

By default, this flag is set to true. When this flag is set to true, application data can be backed up and restored by the user using adb backup and adb restore.

This may have security consequences for an application. adb backup allows users who have enabled USB debugging to copy application data off of the device. Once backed up, all application data can be read by the user. adb restore allows creation of application data from a source specified by the user. Following a restore, applications should not assume that the data, file permissions, and directory permissions were created by the application itself.

Setting allowBackup="false" opts an application out of both backup and restore.

To fix this warning, decide whether your application should support backup, and explicitly set android:allowBackup=(true|false)".

If not set to false, and if targeting API 23 or later, lint will also warn that you should set android:fullBackupContent to configure auto backup.

More info:

- <https://developer.android.com/training/backup/autosyncapi.html>
- <http://developer.android.com/reference/android/R.attr.html#allowBackup>

To suppress this error, use the issue id "AllowBackup" as explained in the [Suppressing Warnings and Errors](#) section.

Performance

LogConditional: Unconditional Logging Calls

../src/main/java/br/com/catbag/giffluxsample/asyncls/files/FileWriter.java:37: The log call Log.d(...) should be conditional: surround with if (Log.isLoggable(...)) Or if (BuildConfig.DEBUG) { ... }

```
34         }
35         outputStream.write(fileReader, 0, read);
36         fileSizeDownloaded += read;
37         Log.d(TAG, "file download: " + fileSizeDownloaded + " of " + fileSize);
```

```

38     }
39

```

Priority: 5 / 10

Category: Performance

Severity: Warning

Explanation: Unconditional Logging Calls.

The BuildConfig class (available in Tools 17) provides a constant, "DEBUG", which indicates whether the code is being built in release mode or in debug mode. In release mode, you typically want to strip out all the logging calls. Since the compiler will automatically remove all code which is inside a "if (false)" check, surrounding your logging calls with a check for BuildConfig.DEBUG is a good idea.

If you really intend for the logging to be present in release mode, you can suppress this warning with a `@SuppressWarnings` annotation for the intentional logging calls.

More info:

To suppress this error, use the issue id "LogConditional" as explained in the [Suppressing Warnings and Errors](#) section.

Overdraw: Overdraw: Painting regions more than once

[../src/main/res/layout/activity_gif_list.xml:11](#): Possible overdraw: Root element paints background `@color/notWatched` with a theme that also paints a background (inferred theme is `@style/AppTheme`)

```

8         android:paddingLeft="@dimen/activity_horizontal_margin"
9         android:paddingRight="@dimen/activity_horizontal_margin"
10        android:paddingTop="@dimen/activity_vertical_margin"
11        android:background="@color/notWatched"
12        tools:context="br.com.catbag.giffluxsample.ui.GifListActivity">
13

```

Priority: 3 / 10

Category: Performance

Severity: Warning

Explanation: Overdraw: Painting regions more than once.

If you set a background drawable on a root view, then you should use a custom theme where the theme background is null. Otherwise, the theme background will be painted first, only to have your custom background completely cover it; this is called "overdraw".

NOTE: This detector relies on figuring out which layouts are associated with which activities based on scanning the Java code, and it's currently doing that using an inexact pattern

matching algorithm. Therefore, it can incorrectly conclude which activity the layout is associated with and then wrongly complain that a background-theme is hidden.

If you want your custom background on multiple pages, then you should consider making a custom theme with your custom background and just using that theme instead of a root element background.

Of course it's possible that your custom drawable is translucent and you want it to be mixed with the background. However, you will get better performance if you pre-mix the background with your drawable and use that resulting image or color as a custom theme background instead.

More info:

To suppress this error, use the issue id "Overdraw" as explained in the [Suppressing Warnings and Errors](#) section.

UnusedResources: Unused resources

.././src/main/res/values/colors.xml:3: The resource R.color.colorPrimary appears to be unused

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <resources>
3   <color name="colorPrimary">#3F51B5</color>
4   <color name="colorPrimaryDark">#303F9F</color>
5   <color name="colorAccent">#FF4081</color>
```

.././src/main/res/values/colors.xml:4: The resource R.color.colorPrimaryDark appears to be unused

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <resources>
3   <color name="colorPrimary">#3F51B5</color>
4   <color name="colorPrimaryDark">#303F9F</color>
5   <color name="colorAccent">#FF4081</color>
6   <color name="notWatched">#11DE11</color>
```

.././src/main/res/values/colors.xml:5: The resource R.color.colorAccent appears to be unused

```
2 <resources>
3   <color name="colorPrimary">#3F51B5</color>
4   <color name="colorPrimaryDark">#303F9F</color>
5   <color name="colorAccent">#FF4081</color>
6   <color name="notWatched">#11DE11</color>
7   <color name="watched">#00FFFFFF</color>
```

Priority: 3 / 10

Category: Performance
Severity: Warning
Explanation: Unused resources.

Unused resources make applications larger and slow down builds.

More info:

To suppress this error, use the issue id "UnusedResources" as explained in the [Suppressing Warnings and Errors](#) section.

Usability

GoogleAppIndexingWarning: Missing support for Firebase App Indexing

../src/main/AndroidManifest.xml:7: App is not indexable by Google Search; consider adding at least one Activity with an ACTION-VIEW intent filter. See issue explanation for more details.

```
4
5     <uses-permission android:name="android.permission.INTERNET"/>
6
7     <application
8         android:name=".App"
9         android:allowBackup="true"
```

Note: This issue has an associated quickfix operation in Android Studio/IntelliJ

Priority: 5 / 10

Category: Usability

Severity: Warning

Explanation: Missing support for Firebase App Indexing.

Adds URLs to get your app into the Google index, to get installs and traffic to your app from Google Search.

More info: <https://g.co/AppIndexing/AndroidStudio>

To suppress this error, use the issue id "GoogleAppIndexingWarning" as explained in the [Suppressing Warnings and Errors](#) section.

Disabled Checks

The following issues were not run by lint, either because the check is not enabled by default, or because it was disabled with a command line flag or via one or more lint.xml configuration files in the project directories.