

# Updating SnapIntercept (and a general guide to Xposed + Snapchat reverse-engineering)

## Intro

SnapIntercept has a file called Obfuscator.java:

```
1 package local.interceptmod.modules;
2
3 interface Obfuscator {
4     // Snapchat version
5     int VersionCode = 1611;
6     String ExpectedVersion = "10.25.0.0";
7     // Snap event - everything here is in the file of SnapEventKlass
8     String SnapEventKlass = "qgt"; // find the file with "-FirstFrame" in it
9     String SnapEventGetCacheKey = "L"; // return L() + "-FirstFrame";
10    String SnapEventIsVideo = "i"; // in the (i()) in the if statement above: return rnv.SNAP_VIDEO_RECEIVED;
11    String SnapEventUsername = "aw"; // this.aw = str2;
12    String SnapEventTimestamp = "v"; // this.v = j2;
13    String SnapEventIsZipped = "ax"; // "isZipped", this.ax
14    // Media cache - everything here is in the file of MediaCacheEntryKlass
15    String MediaCacheEntryKlass = "qlj"; // find the file with ("mCache", this.a).a("mKey", this.b) in it
16    String MediaCacheEntryConstructorFirstParam = "svt"; // public qli(svs svs, String str) {this(svs, str, null, false);}
17    // Encryption
18    String CbcEncryptionAlgorithmKlass = "com.snapchat.android.framework.crypto.CbcEncryptionAlgorithm";
19    String CbcEncryptionAlgorithmDecrypt = "b";
20    String EncryptionAlgorithmInterface = "com.snapchat.android.framework.crypto.EncryptionAlgorithm";
21    // Root detection
22    String RootDetectorKlass = "rza"; // "/sbin/su", "/system/bin/su",
23    String RootDetectorFirst = "b";
24    String RootDetectorSecond = "c";
25    String RootDetectorThird = "d";
26    String RootDetectorForth = "e";
27 }
```

This interface maps obfuscated class/method names to ones that the mod can use, and basically makes it much easier to update the mod for Snapchat updates that don't change the logic of how incoming snaps are received/decrypted and just end up changing the obfuscation.

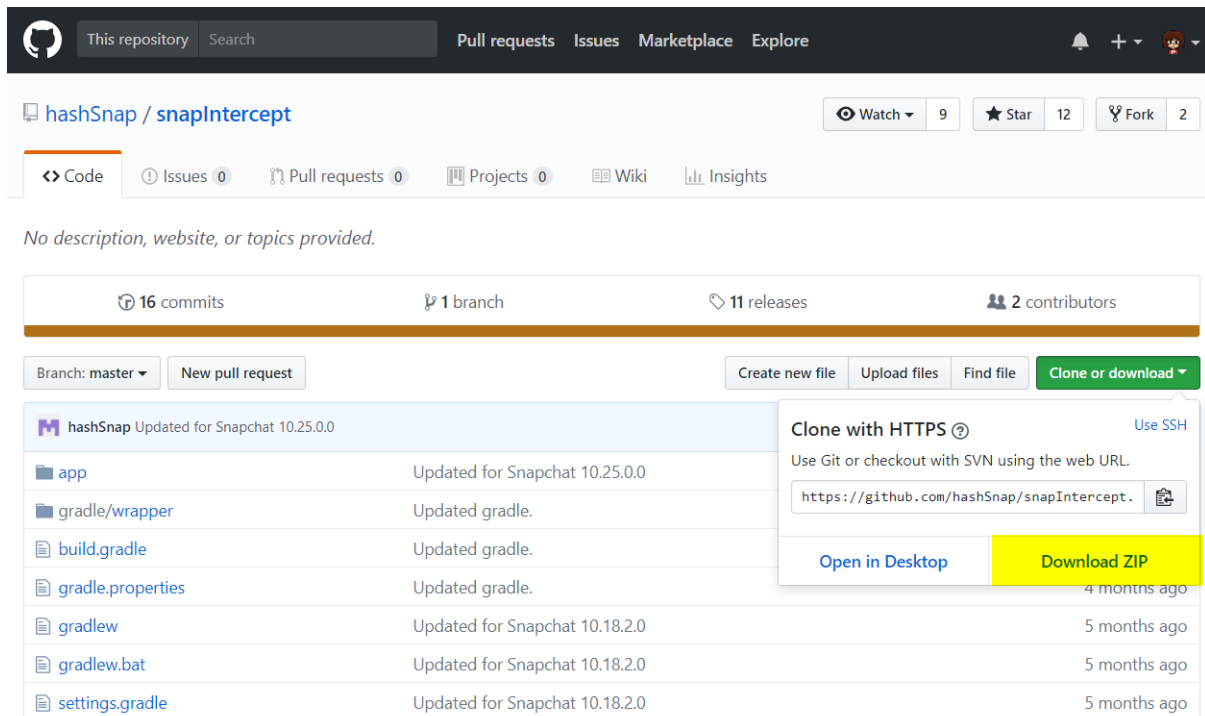
This file is what we aim to update when it comes to these Snapchat updates.

## What You Need

- [JADX](#) – this decompiles the Snapchat APK and lets us read it in understandable (but still obfuscated) Java code
- Snapchat APKs – you'll need both of the following:
  - The **latest version** (that you want to update to) – get this by updating Snapchat on your phone, then exporting the APK via something like Solid Explorer's Applications menu or something – this ensures you 100% have the correct new APK that phones will update to. Technically you could use APKMirror here but there are cases where they get the APK wrong.
  - The **version that works with the latest SnapIntercept** – if you don't have this on your phone, just grab it from [APKMirror](#).
- A text editor of some kind (I like [Sublime Text](#))
- [Android Studio](#) – to build and test your SnapIntercept update

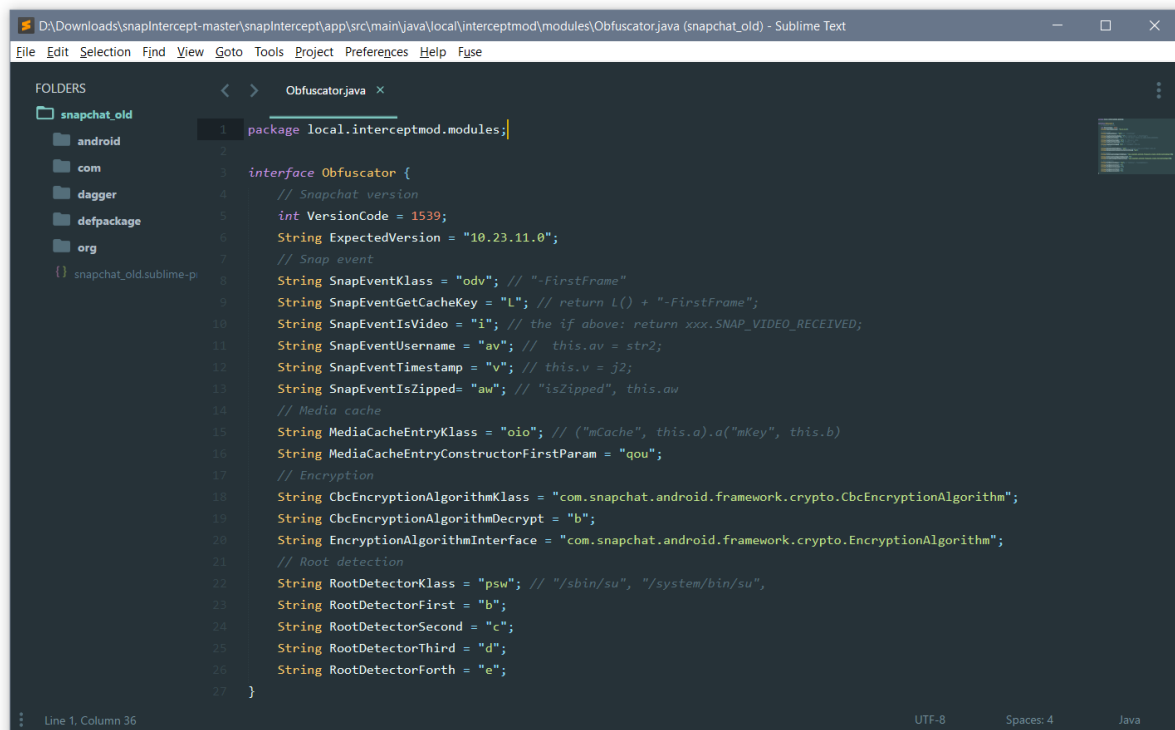
## Setting Up

First, go ahead and download the latest release version of SnapIntercept's source code:



*Note: you can also use Android Studio to directly clone this and work entirely in Android Studio without using Sublime, I just prefer Sublime.*

Go ahead and unzip that somewhere, then open up Obfuscator.java (in *SnapIntercept/app/src/main/java/local/interceptmod/modules*) in your text editor:









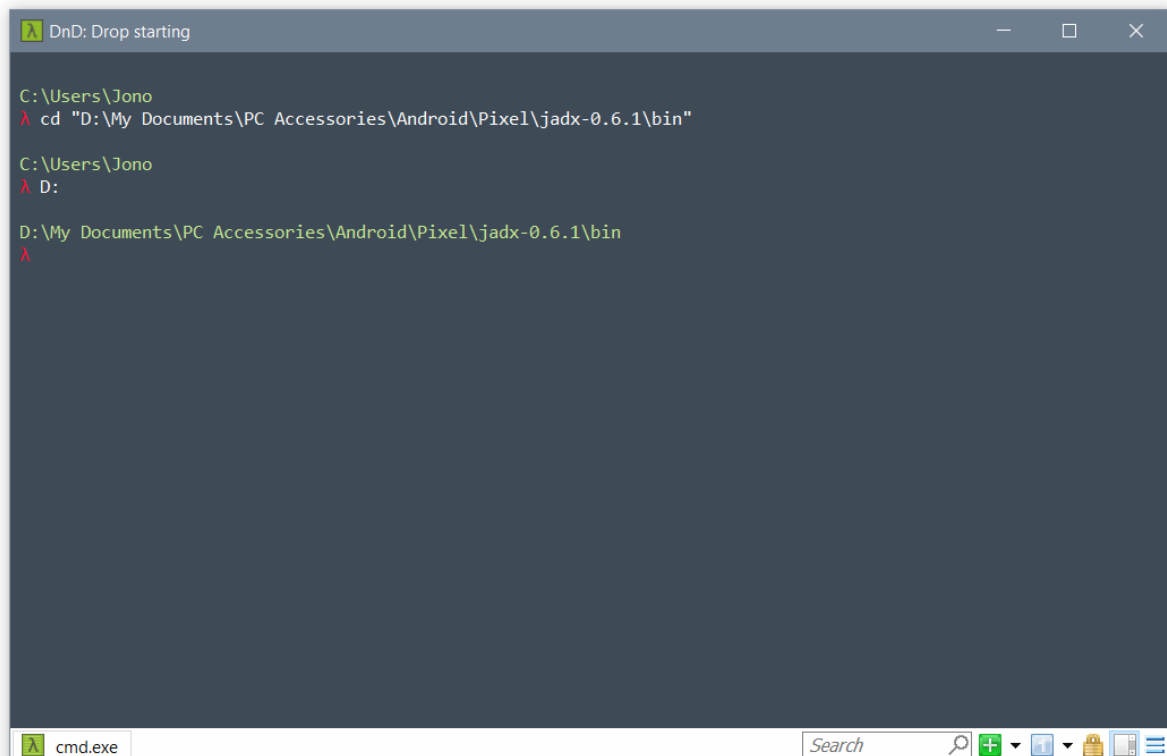
Next, go ahead and change ExpectedVersion and VersionCode to the version of the new Snapchat app. The easiest way to find VersionCode is to again use Solid Explorer's Applications feature, or you can also use Titanium Backup. In my case we went from 10.23.11.0 to 10.25.0.0 (and 1539 to 1611):

```
interface Obfuscator {
    // Snapchat version
    int VersionCode = 1611;
    String ExpectedVersion = "10.25.0.0";
}
```

### Decompiling Snapchat APKs

Our next step is to throw the snapchat APKs at JADX so we can get to the code. Rename the older version of snapchat to snapchat\_old.apk, and the newer version you're updating to snapchat.apk. Place them both in the same folder as jadx, and then open CMD and cd to the JADX folder:

<input type="checkbox"/> Name	Date modified	Type	Size
 snapchat.apk	15/02/2018 9:29 A...	APK File	65,330 KB
 snapchat_old.apk	10/02/2018 6:35 PM	APK File	65,330 KB
 jadx	5/12/2016 8:18 AM	File	6 KB
 jadx-gui	5/12/2016 8:18 AM	File	6 KB
 jadx.bat	5/12/2016 8:18 AM	Windows Batch File	3 KB
 jadx-gui.bat	5/12/2016 8:18 AM	Windows Batch File	3 KB



```
DnD: Drop starting

C:\Users\Jono
λ cd "D:\My Documents\PC Accessories\Android\Pixel\jadx-0.6.1\bin"

C:\Users\Jono
λ D:

D:\My Documents\PC Accessories\Android\Pixel\jadx-0.6.1\bin
λ
```

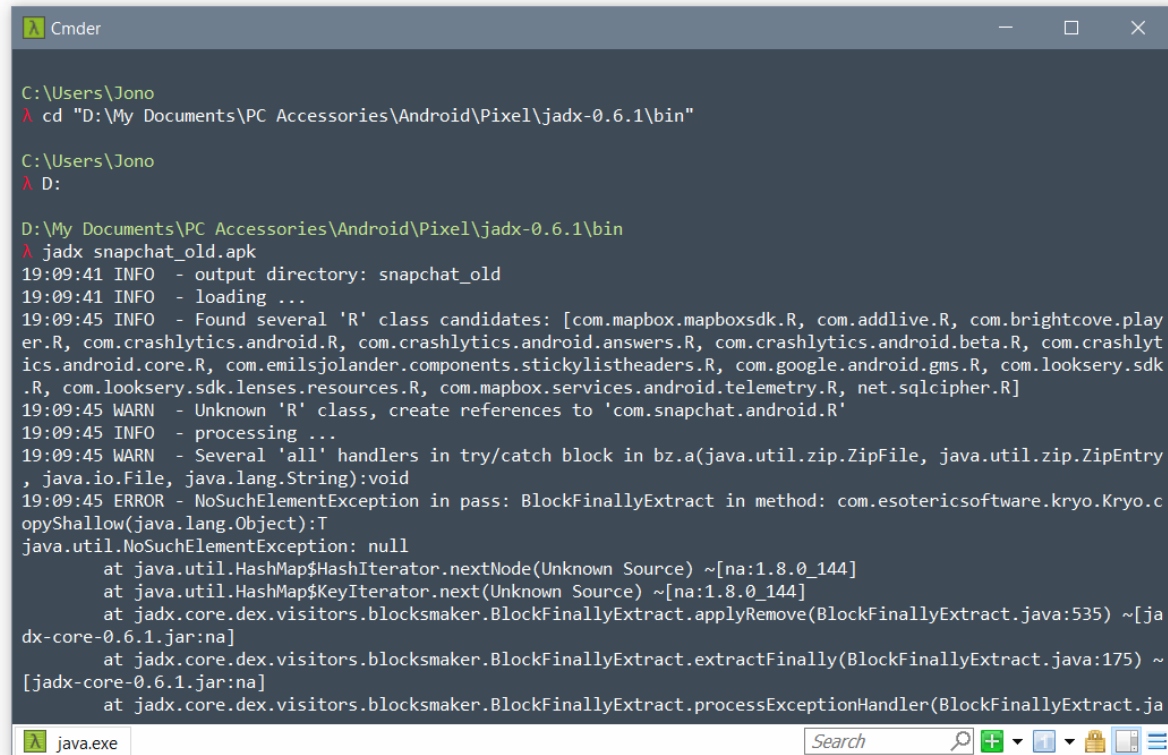
Now, in CMD run the following:

*jadx snapchat\_old.apk*

This will decompile the old version of snapchat (10.23.11.0 in my case) and place the java code in a folder called snapchat\_old. This can take a while, but usually it pulls out the bits you need within the first 5-10mins and you can cancel the batch job (Ctrl+C in cmd) after that.

Then, do the same with the new snapchat apk (10.25.0.0 in my case):









*jadx snapchat.apk*



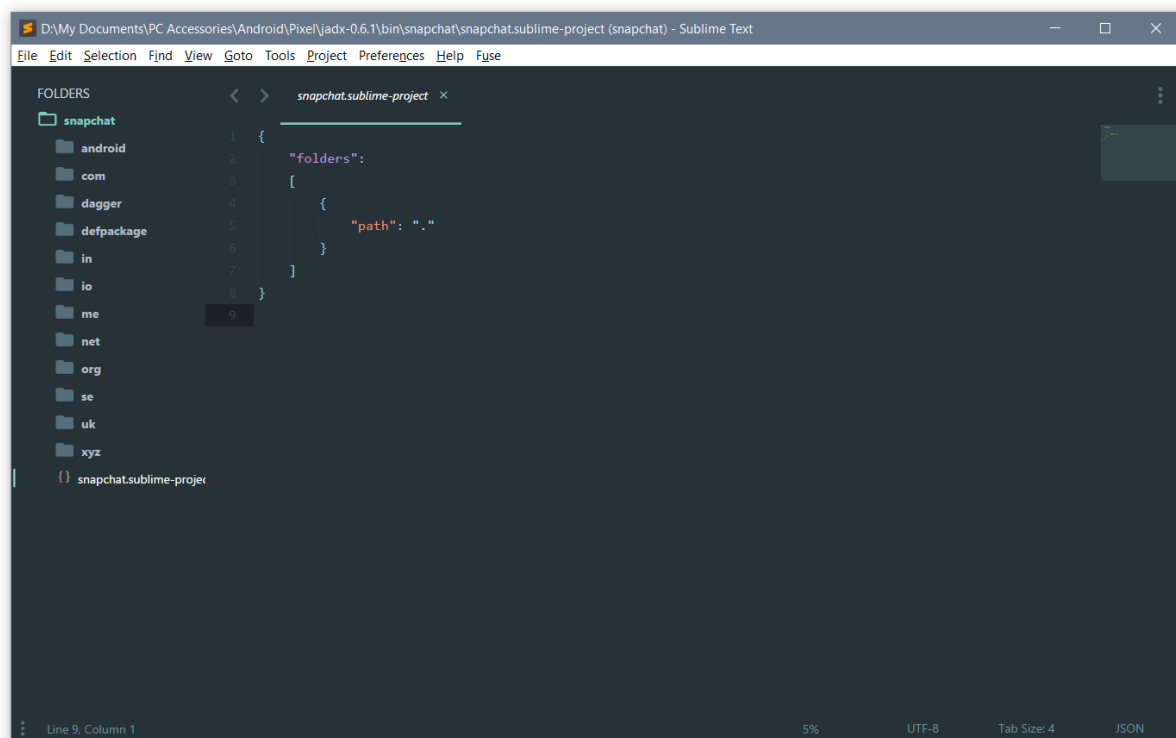
```
C:\Users\Jono
λ cd "D:\My Documents\PC Accessories\Android\Pixel\jadx-0.6.1\bin"

C:\Users\Jono
λ D:

D:\My Documents\PC Accessories\Android\Pixel\jadx-0.6.1\bin
λ jadx snapchat_old.apk
19:09:41 INFO - output directory: snapchat_old
19:09:41 INFO - loading ...
19:09:45 INFO - Found several 'R' class candidates: [com.mapbox.mapboxsdk.R, com.addlive.R, com.brightcove.play
er.R, com.crashlytics.android.R, com.crashlytics.android.answers.R, com.crashlytics.android.beta.R, com.crashlyt
ics.android.core.R, com.emilsjolander.components.stickylistheaders.R, com.google.android.gms.R, com.looksery.sdk
.R, com.looksery.sdk.lenses.resources.R, com.mapbox.services.android.telemetry.R, net.sqlcipher.R]
19:09:45 WARN - Unknown 'R' class, create references to 'com.snapchat.android.R'
19:09:45 INFO - processing ...
19:09:45 WARN - Several 'all' handlers in try/catch block in bz.a(java.util.zip.ZipFile, java.util.zip.ZipEntry
, java.io.File, java.lang.String):void
19:09:45 ERROR - NoSuchElementException in pass: BlockFinallyExtract in method: com.esotericsoftware.kryo.Kryo.c
opyShallow(java.lang.Object):T
java.util.NoSuchElementException: null
    at java.util.HashMap$HashIterator.nextNode(Unknown Source) ~[na:1.8.0_144]
    at java.util.HashMap$KeyIterator.next(Unknown Source) ~[na:1.8.0_144]
    at jadx.core.dex.visitors.blocksmer.BlockFinallyExtract.applyRemove(BlockFinallyExtract.java:535) ~[ja
dx-core-0.6.1.jar:na]
    at jadx.core.dex.visitors.blocksmer.BlockFinallyExtract.extractFinally(BlockFinallyExtract.java:175) ~
[jadx-core-0.6.1.jar:na]
    at jadx.core.dex.visitors.blocksmer.BlockFinallyExtract.processExceptionHandler(BlockFinallyExtract.java
```

<input type="checkbox"/> Name	Date modified	Type ^	Size
 snapchat	15/02/2018 9:42 A...	File folder	
 snapchat_old	15/02/2018 7:09 PM	File folder	
 snapchat.apk	15/02/2018 9:29 A...	APK File	65,330 KB
 snapchat_old.apk	15/02/2018 7:09 PM	APK File	63,992 KB
 jadx	5/12/2016 8:18 AM	File	6 KB
 jadx-gui	5/12/2016 8:18 AM	File	6 KB
 jadx.bat	5/12/2016 8:18 AM	Windows Batch File	3 KB
 jadx-gui.bat	5/12/2016 8:18 AM	Windows Batch File	3 KB

If you're using sublime, go ahead and chuck a *.sublime-project* file in both of those directories so you can open them up as separate projects. I call my projects snapchat and snapchat\_old:



You should now have two sublime projects up (in two separate windows), one for snapchat\_old and one for snapchat.

### Updating Obfuscator.java

Now, the fun/tedious part – updating the Obfuscator.java file.

Let's start with SnapEventClass. In Obfuscator.java, note the value of SnapEventClass – in this case it's "odv".

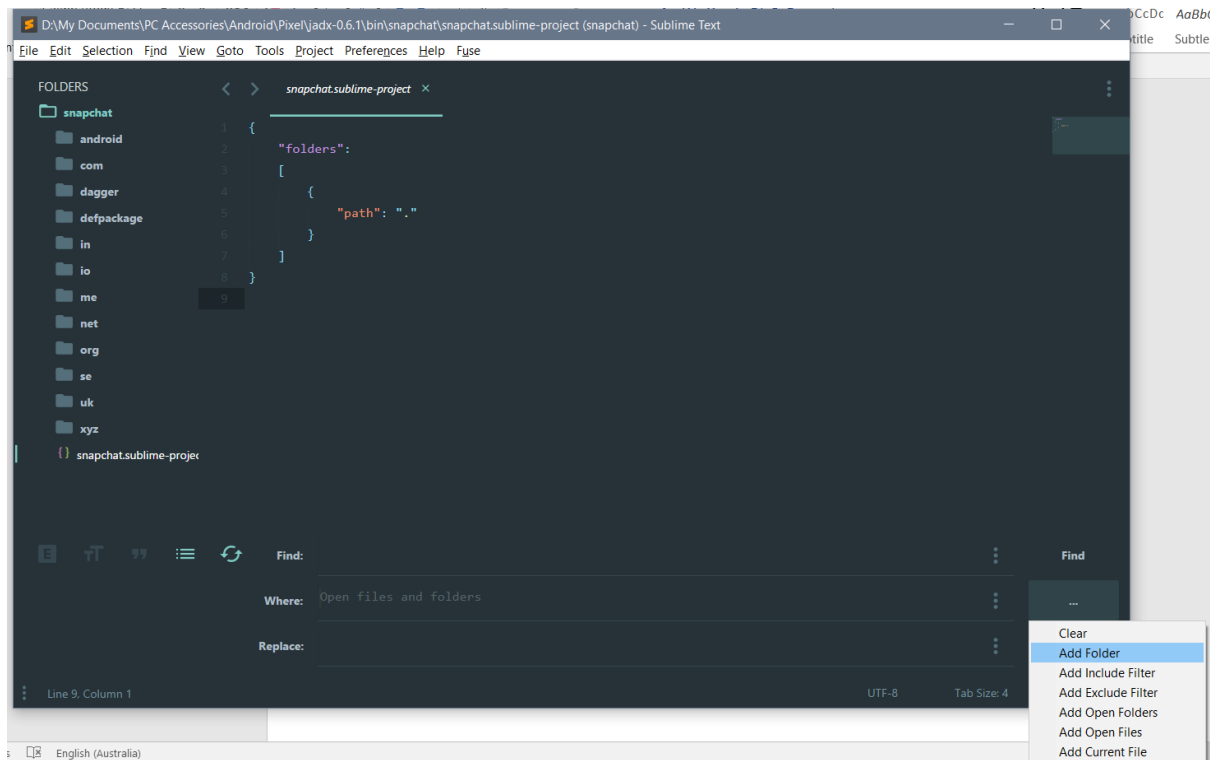
```
// Snap event
String SnapEventClass = "odv"; // "-FirstFrame"
String SnapEventGetCacheKey = "L"; // return L() + "-FirstFrame";
String SnapEventIsVideo = "i"; // the if above: return xxx.SNAP_VIDEO_RECEIVED;
String SnapEventUsername = "av"; // this.av = str2;
String SnapEventTimestamp = "v"; // this.v = j2;
String SnapEventIsZipped = "aw"; // "isZipped", this.aw
```

Now, go to your snapchat\_old Sublime window, go to the defpackage folder on the side, and then find odv.java and open it. If you can't find it, it's likely you didn't let jadx decompile the apk for long enough.

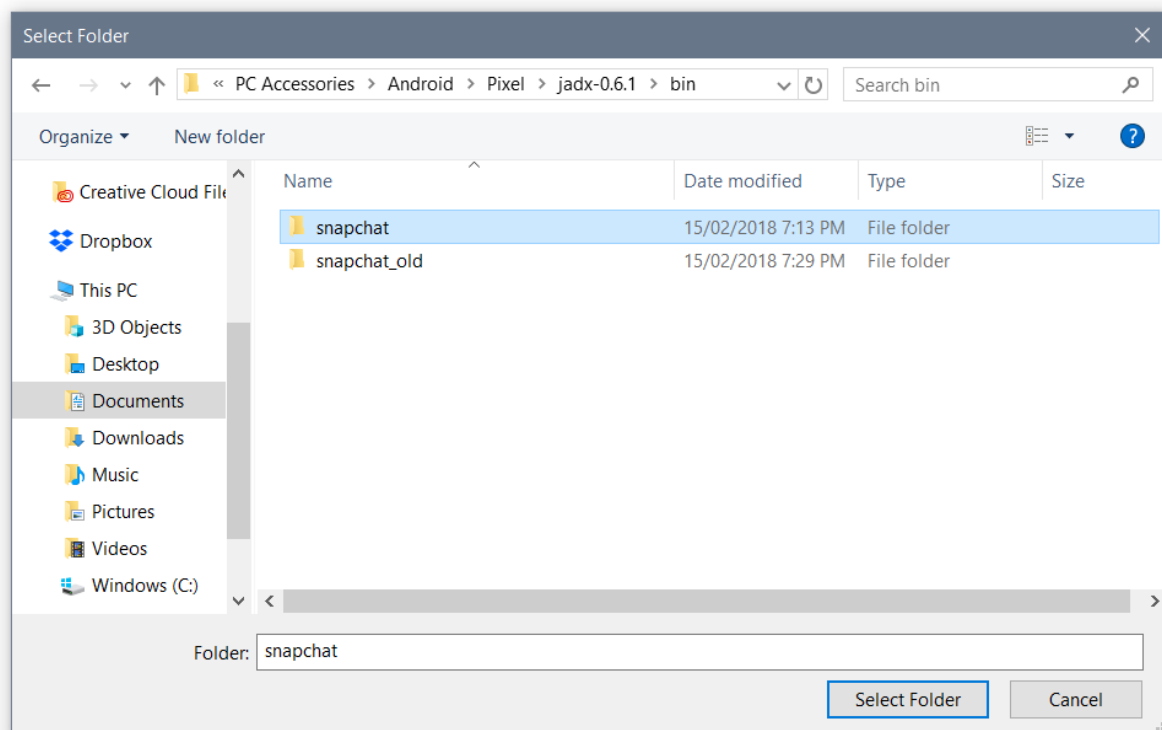
Now look for a line of code you think is pretty unique to the file and *doesn't contain an obfuscated variable* – in this case (as the comment says), go for "-FirstFrame".

```
209         public String at() {
210             return L() + "-FirstFrame";
211         }
```

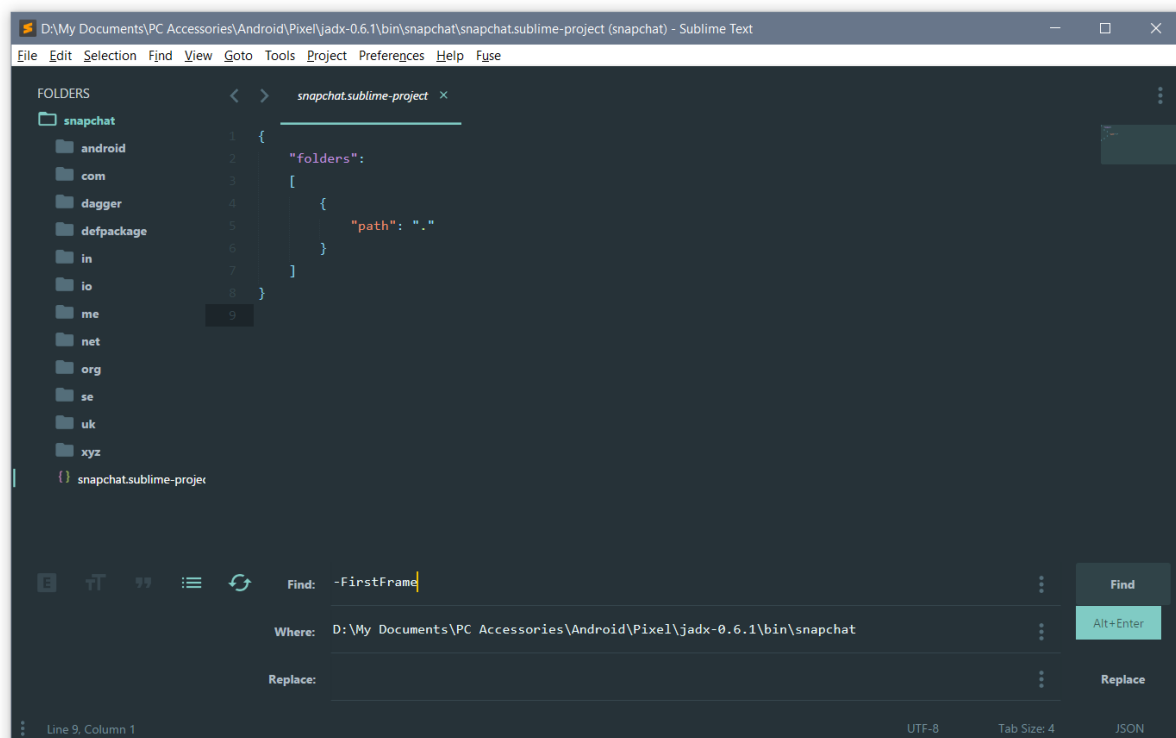
Now go to your new snapchat Sublime window, and press Ctrl+Shift+F to bring up the Find box, and click the ellipsis next to Where, then choose “Add Folder”.



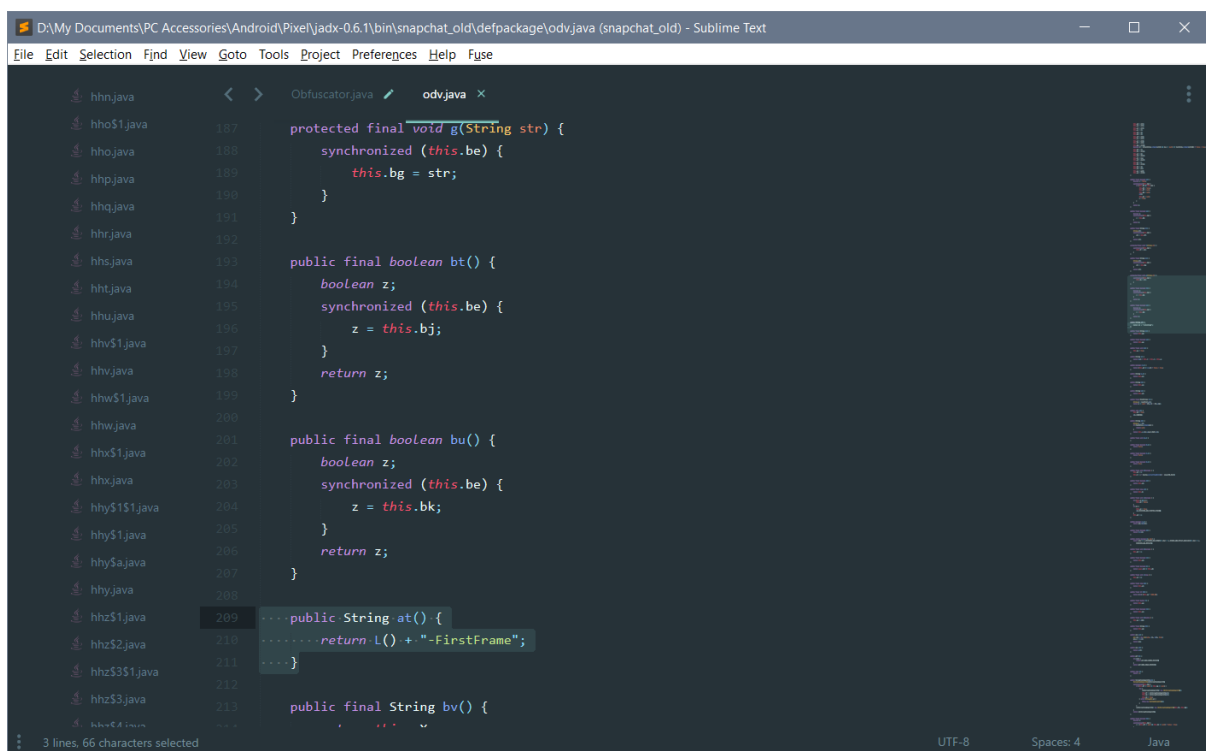
Then just choose the ‘snapchat’ folder containing the new apk’s code:

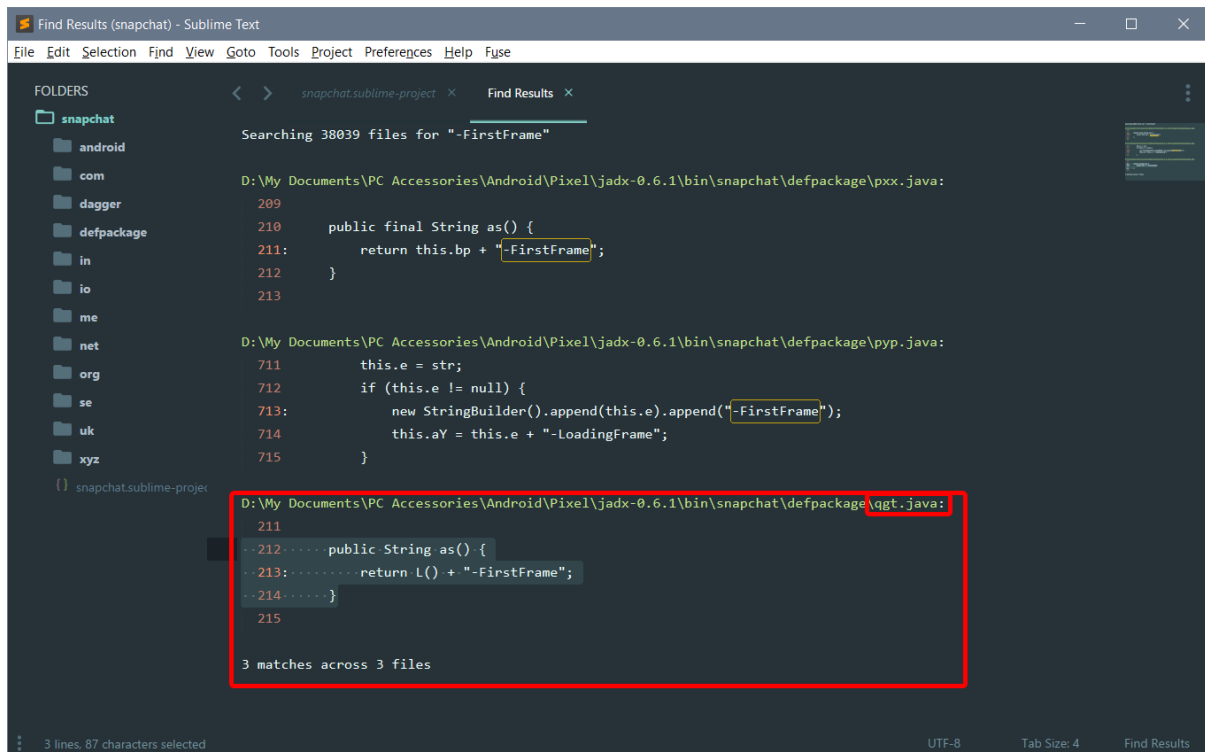


Now type your recognisable string from the old APK’s code (in this case *-FirstFrame*) into the Find box, and search for it:



This'll search the codebase of the new snapchat APK for any reference that matches the code snippet from the old APK! This is so we can find the new name of class. Wait a little while, and you'll get a few results. Pick the one that most closely resembles the code from the old apk:





Note the filename that contains this code – here it's qgt.java. Essentially what this means is that qgt.java is the same file as odv.java, it's just been renamed! So, we need to update our Obfuscator.java file to match this renaming. Go back to Obfuscator.java, and make the change!

```
// Snap event
String SnapEventKlass = "qgt"; // "-FirstFrame"
String SnapEventGetCacheKey = "L"; // return L() + "-FirstFrame";
String SnapEventIsVideo = "i"; // the if above: return xxx.SNAP_VIDEO_RECEIVED;
String SnapEventUsername = "av"; // this.av = str2;
String SnapEventTimestamp = "v"; // this.v = j2;
String SnapEventIsZipped = "aw"; // "isZipped", this.aw
```

Now do the same for all the other strings here. The difference is, all of the strings under the *// Snap event* comment will actually be found within the odv.java/qgt.java files! So, you don't actually need to go looking for a new file here, just search through qgt.java for the new string names. For example:

```
String SnapEventKlass = "qgt"; // "-FirstFrame"
String SnapEventGetCacheKey = "L"; // return L() + "-FirstFrame";
String SnapEventIsVideo = "i"; // the if above: return xxx.SNAP_VIDEO_RECEIVED;
```

Let's check odv.java...

```
public String at() {
    return L() + "-FirstFrame";
}
```

Now, let's search qgt.java for something that looks like this!



```
public String as() {
    return L() + "-FirstFrame";
}
```

So SnapEventGetCacheKey is 'L'. Here, we see that references the L() function in odv.java. When we look for that function in qgt.java, we find that it remains unchanged! It's still L()! So, we don't have to change SnapEventGetCacheKey!

It's the same for SnapEventIsVideo, which references the i() function above the code snippet "return pjf.SNAP\_VIDEO\_RECEIVED" in odv.java. Searching for SNAP\_VIDEO\_RECEIVED in qgt.java, we find that the function is still called i(), so don't change that!

odv.java:

```
public pjf b() {
    if (i()) {
        return pjf.SNAP_VIDEO_RECEIVED;
    }
    return pjf.SNAP_IMAGE_RECEIVED;
}
```

qgt.java:

```
public rnw b() {
    if (i()) {
        return rnw.SNAP_VIDEO_RECEIVED;
    }
    return rnw.SNAP_IMAGE_RECEIVED;
}
```

Now let's look at SnapEventUsername:

```
String SnapEventUsername = "av"; // this.av = str2;
String SnapEventTimestamp = "v"; // this.v = j2;
String SnapEventIsZipped = "aw"; // "isZipped", this.aw
```

The comment next to it is hinting that in odv.java, this string references a variable called av which has the value str2. So, let's try searching qgt.java for "= str2":

odv.java:

```
qkn$a.a;
this.av = str2;
this.aw = z;
```

qgt.java:

```
srk$a.a;  
this.aw = str2;  
this.ax = z;
```

Here's a change! In the new version of snapchat, *this.av* has become *this.aw*! So, go back to Obfuscator.java and make the change!

```
String SnapEventUsername = "aw"; // this.aw = str2;
```

Now use the same process for SnapEventTimestamp and SnapEventIsZipped, using the comments next to them as a hint for what to search for. You should get the following:

Before:

```
String SnapEventTimestamp = "v"; // this.v = j2;  
String SnapEventIsZipped= "aw"; // "isZipped", this.aw
```

After:

```
String SnapEventTimestamp = "v"; // this.v = j2;  
String SnapEventIsZipped= "ax"; // "isZipped", this.ax
```

*Note: obviously the strings you get will change depending on the versions of snapchat you're using, but if you're following along using the same versions as this tutorial, you should get these.*

When you get to MediaCacheEntryKlass, you're looking for a file again! So follow the same process you used to find qgt.java, and do the same here.

```
// Media cache  
String MediaCacheEntryKlass = "oio"; // ("mCache", this.a).a("mKey", this.b)  
String MediaCacheEntryConstructorFirstParam = "qou";
```

In my case, the old class was oio.java, and looking in there I searched the new snapchat codebase for "mCache" (again, as per the comment), and eventually found this snippet in qlj.java:

```
D:\My Documents\PC Accessories\Android\Pixel\jadx-0.6.1\bin\snapchat\defpackage\qlj.java:  
23  
24     public final String toString() {  
25:         return bct.a(this).a("mCache", this.a).a("mKey", this.b).a("mAlgorithm",  
this.c).a("mIsNewMediaCache", this.d).toString();  
26     }  
27 }
```

This seemed to match the snippet in oio.java:

```
public final String toString() {  
    return bcn.a(this).a("mCache", this.a).a("mKey", this.b).a("mAlgorithm", this.c).a("mIsNewMediaCache",  
        this.d).toString();  
}
```

So, I can safely assume qlj.java = oio.java. Applying the process we used for the SnapEventKlass before, I can search qlj.java to find what “qou” (MediaCacheEntryConstructorFirstParam) has changed to:

*oio.java:*

```
public oio(qou qou, String str, EncryptionAlgorithm encryptionAlgorithm, boolean z) {  
    this.e = ust.UNKNOWN;  
    this.a = qou;  
    this.b = str;  
    this.c = encryptionAlgorithm;  
    this.d = z;  
}
```

*qlj.java:*

```
public qlj(svt svt, String str, EncryptionAlgorithm encryptionAlgorithm, boolean z) {  
    this.e = wvl.UNKNOWN;  
    this.a = svt;  
    this.b = str;  
    this.c = encryptionAlgorithm;  
    this.d = z;  
}
```

So, we make the appropriate changes in Obfuscator.java:

*Before:*

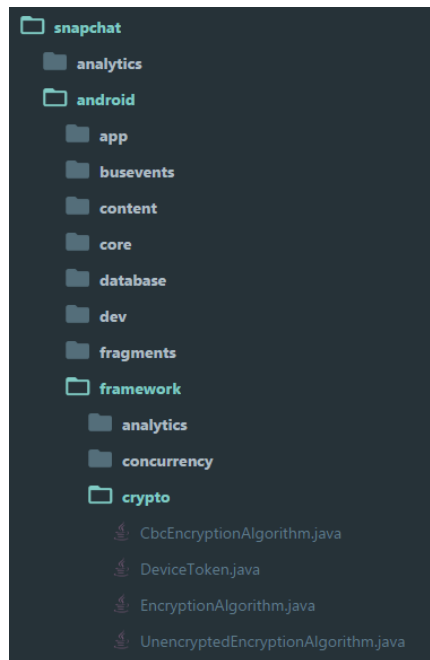
```
// Media cache  
String MediaCacheEntryKlass = "oio"; // ("mCache", this.a).a("mKey", this.b)  
String MediaCacheEntryConstructorFirstParam = "qou";
```

*After:*

```
// Media cache  
String MediaCacheEntryKlass = "qlj"; // ("mCache", this.a).a("mKey", this.b)  
String MediaCacheEntryConstructorFirstParam = "svt";
```

CbcEncryptionAlgorithmKlass is a lil' different. As the string suggests, it's not found in the defpackage folder like the rest of the classes, it's actually found in the /com/snapchat/android/framework/crypto path, as CbcEncryptionAlgorithm.java.

```
// Encryption  
String CbcEncryptionAlgorithmKlass = "com.snapchat.android.framework.crypto.CbcEncryptionAlgorithm";  
String CbcEncryptionAlgorithmDecrypt = "b";  
String EncryptionAlgorithmInterface = "com.snapchat.android.framework.crypto.EncryptionAlgorithm";
```



The EncryptionAlgorithmInterface references the EncryptionAlgorithm.java file in the same directory. These two probably won't change between versions of snapchat. What might change though, is CbcEncryptionAlgorithmDecrypt. This is found in CbcEncryptionAlgorithm.java by doing a search for "DECRYPT":

```
}  
  
public final InputStream b(InputStream inputStream) {  
    InputStream inputStream2 = null;  
    if (!e()) {  
        try {  
            inputStream2 = a(inputStream, a.DECRYPT);  
        } catch (GeneralSecurityException e) {  
        }  
    }  
    if (inputStream2 == null) {  
        return c(inputStream);  
    }  
    return inputStream2;  
}
```

In my case, it didn't change, but keep an eye out.

Finally, RootDetectorKlass. Use the same procedure as usual.

```
// Root detection
String RootDetectorKlass = "psw"; // "/sbin/su", "/system/bin/su",
String RootDetectorFirst = "b";
String RootDetectorSecond = "c";
String RootDetectorThird = "d";
String RootDetectorForth = "e";
```

Find the new name for psw.java (in my case, it's rza.java, found by searching for 'sbin/su'):

```
Searching 38039 files for "sbin/su"

D:\My Documents\PC Accessories\Android\Pixel\jadx-0.6.1\bin\snaphat\defpackage\rza.java:
7 public final class rza {
8     private static final rza a = new rza();
9:    private static final String[] b = new String[]{"sbin/su", "/system/bin/su", "/system/sbin/su", "/
data/local/sbin/su", "/data/local/bin/su", "/system/sd/sbin/su", "/system/bin/failsafe/su", "/data/local/
su"};
10
11    public static rza a() {

1 match in 1 file
```

Which matches this snippet in psw.java:

```
public final class psw {
    private static final psw a = new psw();
    private static final String[] b = new String[]{"_sbin/su", "/system/bin/su", "/system/sbin/su", "/data/local/sbin/su", "/data/local
/bin/su", "/system/sd/sbin/su", "/system/bin/failsafe/su", "/data/local/su"};

    public static psw a() {
        return a;
    }
}
```

The rest of the strings remained unchanged in my case. Make the final changes to Obfuscator.java, save the file, and we're ready to test!

*Before:*

```
// Root detection
String RootDetectorKlass = "psw"; // "/sbin/su", "/system/bin/su",
String RootDetectorFirst = "b";
String RootDetectorSecond = "c";
String RootDetectorThird = "d";
String RootDetectorForth = "e";
```

After:

```
// Root detection
String RootDetectorKlass = "rza"; // "/sbin/su", "/system/bin/su",
String RootDetectorFirst = "b";
String RootDetectorSecond = "c";
String RootDetectorThird = "d";
String RootDetectorForth = "e";
```

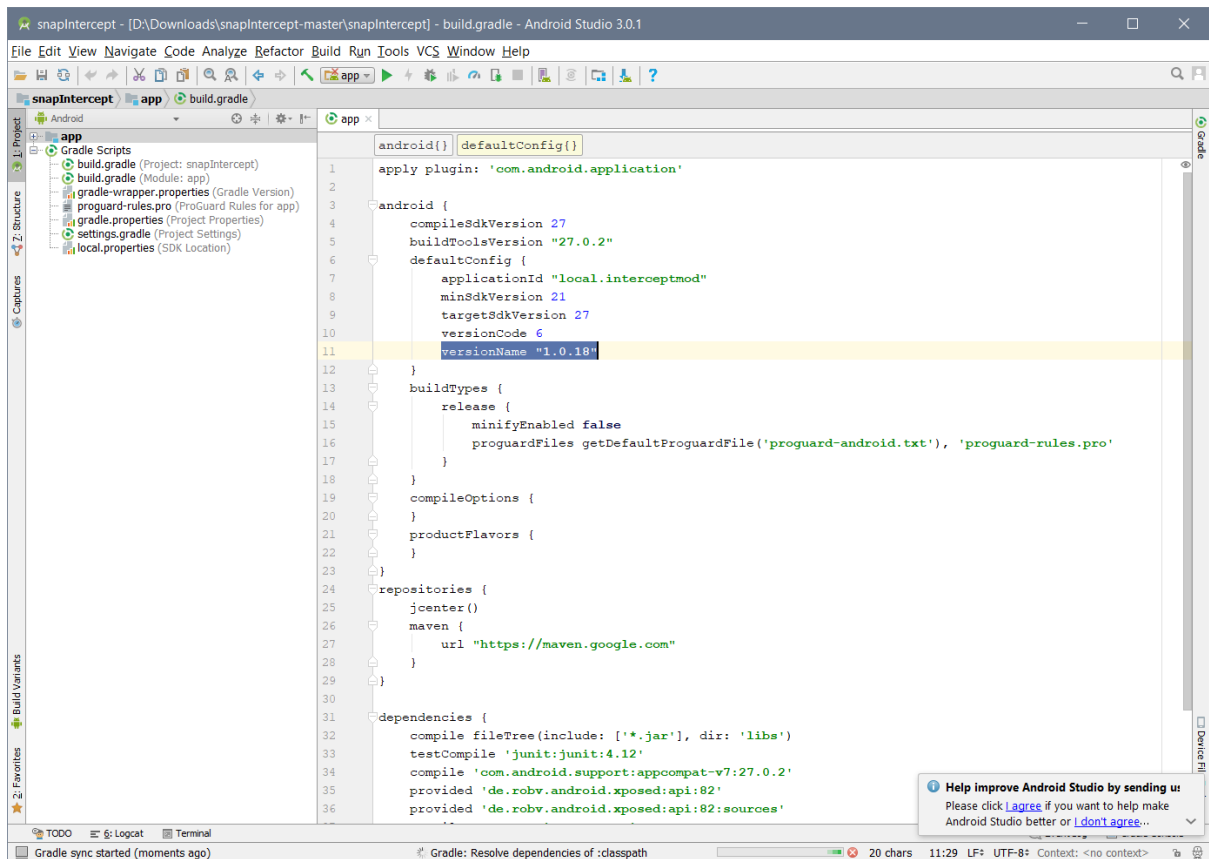
All the changes:

```
> qlj.java x Find Results x Obfuscator.java qgt.java
package local.interceptmod.modules;

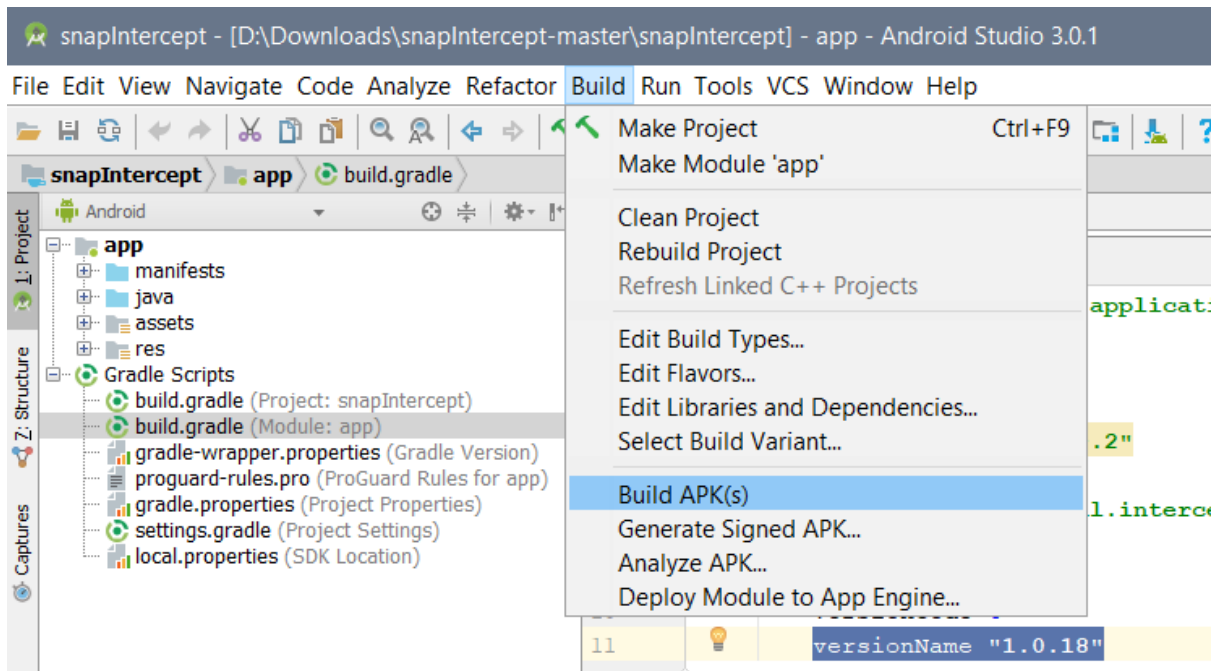
interface Obfuscator {
    // Snapchat version
    int VersionCode = 1611;
    String ExpectedVersion = "10.25.0.0";
    // Snap event
    String SnapEventKlass = "qgt"; // "-FirstFrame"
    String SnapEventGetCacheKey = "L"; // return L() + "-FirstFrame";
    String SnapEventIsVideo = "i"; // the if above: return xxx.SNAP_VIDEO_RECEIVED;
    String SnapEventUsername = "aw"; // this.aw = str2;
    String SnapEventTimestamp = "v"; // this.v = j2;
    String SnapEventIsZipped = "ax"; // "isZipped", this.ax
    // Media cache
    String MediaCacheEntryKlass = "qlj"; // ("mCache", this.a).a("mKey", this.b)
    String MediaCacheEntryConstructorFirstParam = "svt";
    // Encryption
    String CbcEncryptionAlgorithmKlass = "com.snapchat.android.framework.crypto.CbcEncryptionAlgorithm";
    String CbcEncryptionAlgorithmDecrypt = "b";
    String EncryptionAlgorithmInterface = "com.snapchat.android.framework.crypto.EncryptionAlgorithm";
    // Root detection
    String RootDetectorKlass = "rza"; // "/sbin/su", "/system/bin/su",
    String RootDetectorFirst = "b";
    String RootDetectorSecond = "c";
    String RootDetectorThird = "d";
    String RootDetectorForth = "e";
}
```

Testing your changes!

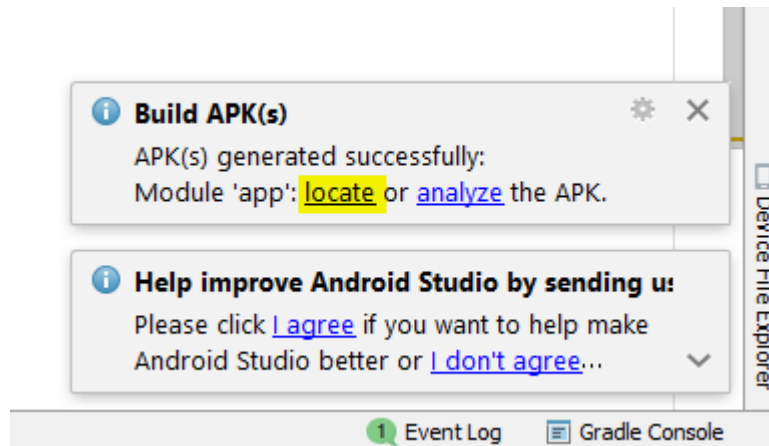
You can now exit Sublime and open up Android Studio. Point Studio to your SnapIntercept folder, and open up the build.gradle file with "(Module: app)" next to it. Increment the versionName (in my case I changed 1.0.17 to 1.0.18):



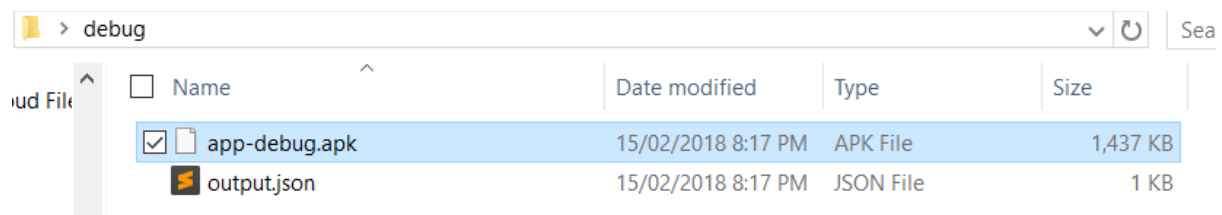
Save it, then go to Build → Build APK(s).



If you get a build error, make sure you've got the Android SDK installed for the target version – here it's Oreo (8.1.1). Now click 'locate':



This'll open up the folder containing the APK for your updated SnapIntercept APK! Usually this is in *SnapIntercept/app/build/outputs/apk/debug/*



Go ahead and chuck this APK on your phone, install it, activate the Xposed module, and restart your phone. Open up Snapchat, send yourself a snap, and check that SnapIntercept correctly saved that snap. If it's working, great job!! Go ahead and [open up a Pull Request](#) so that everyone else can benefit from your hard work! 😊