

# Practice of Android Reverse Engineering

Jim Huang ( 黃敬群 )

Developer, 0xlab

[jserv@0xlab.org](mailto:jserv@0xlab.org)

July 23, 2011 / HITcon

# Rights to copy

© Copyright 2011 **0xlab**

<http://0xlab.org/>

[contact@0xlab.org](mailto:contact@0xlab.org)



## Attribution – ShareAlike 3.0

### You are free

- to copy, distribute, display, and perform the work
- to make derivative works
- to make commercial use of the work

### Under the following conditions

- **BY:** **Attribution.** You must give the original author credit.
- **Share Alike.** If you alter, transform, or build upon this work, you may distribute the resulting work only under a license identical to this one.
- For any reuse or distribution, you must make clear to others the license terms of this work.
- Any of these conditions can be waived if you get permission from the copyright holder.

**Your fair use and other rights are in no way affected by the above.**

License text: <http://creativecommons.org/licenses/by-sa/3.0/legalcode>

Corrections, suggestions, contributions and translations are welcome!

Latest update: July 23, 2011



# Myself

was a Kaffe Developer

- Threaded Interpreter, JIT, AWT for embedded system, robustness

was a GCJ (Java Frontend for GCC) and GNU Classpath Developer

is an AOSP (Android Open Source Project) contributor

- 30+ patches are merged officially
- bionic libc, ARM optimizations



# Not Only for Cracking

- (1) Sometimes, it takes \_\_time\_\_ to obtain source code than expected. → Taiwanese ODM
- (2) Post-optimizations over existing Android applications
- (3) “Borrow” something good to produce “goods”



# Background Knowledge

(and Thank you!)

- The Code Injection and Data Protection of Android, Thinker Li @HITcon2011
- Reversing Android Malware, Mahmud ab Rahman @HITcon2011
- My focus would be the practice.
  - Hack Android applications for Beginners

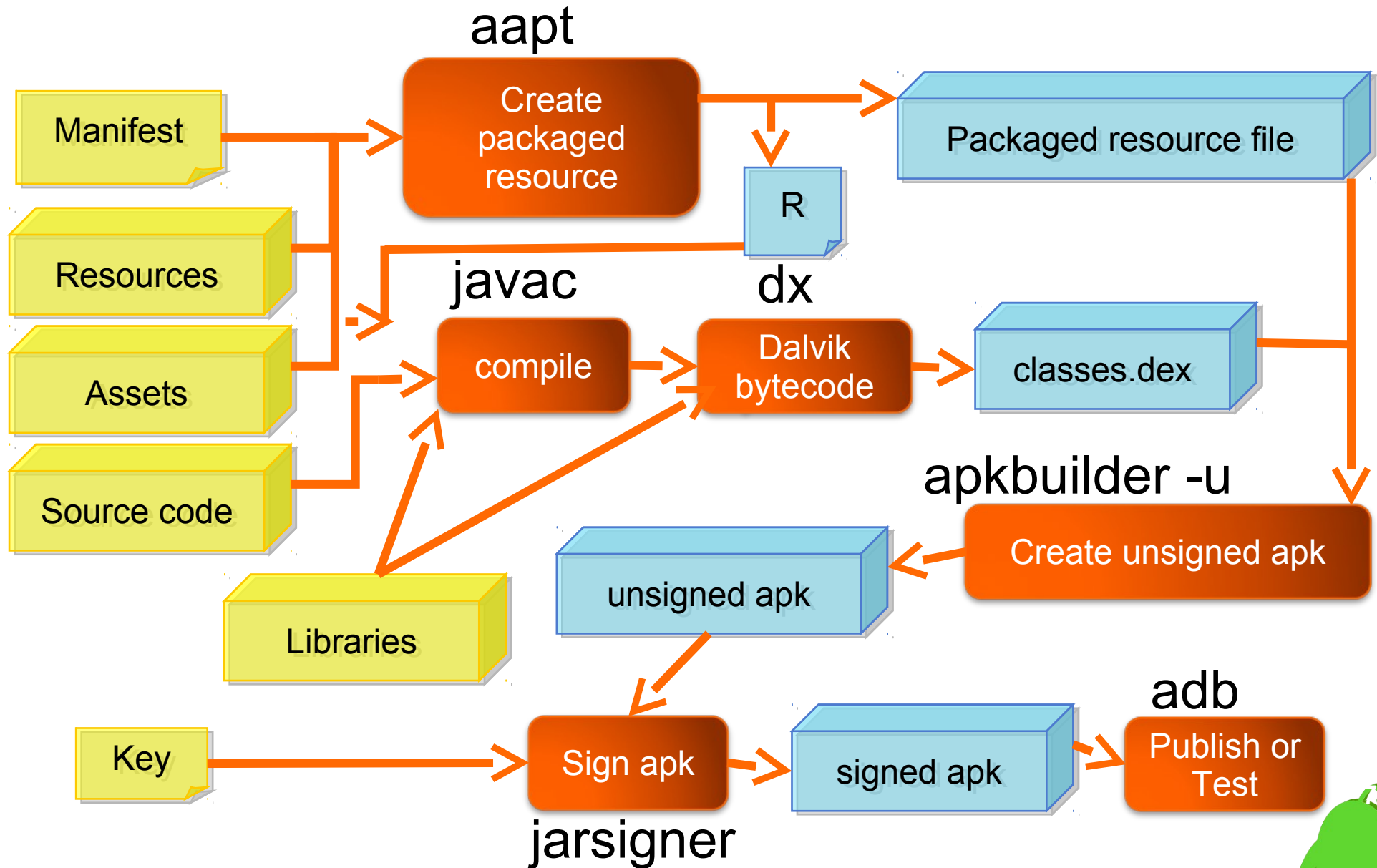


# Agenda

- (1) Development Flow
- (2) Reverse Practice
- (3) Real world tasks



# Android Application Development Flow



# APK content

```
$ unzip Angry+Birds.apk
```

```
Archive:  Angry+Birds.apk
```

```
...
```

```
  inflating: AndroidManifest.xml
```

```
 extracting: resources.arsc
```

```
 extracting: res/drawable-hdpi/icon.png
```

```
 extracting: res/drawable-ldpi/icon.png
```

```
 extracting: res/drawable-mdpi/icon.png
```

```
  inflating: classes.dex Dalvik DEX
```

```
  inflating: lib/armeabi/libangrybirds.so JNI
```

```
  inflating: lib/armeabi-v7a/libangrybirds.so
```

```
  inflating: META-INF/MANIFEST.MF
```

```
  inflating: META-INF/CERT.SF
```

```
  inflating: META-INF/CERT.RSA
```

**manifest +  
signature**



# APK content

```
$ unzip Angry+Birds.apk  
Archive:  Angry+Birds.apk
```

```
...
```

```
  inflating:  AndroidManifest.xml
```

```
Name:  classes.dex
```

```
SHA1-Digest:  I9Vne//i/5Wyzs5HhBVu9dIoHDY=
```

```
Name:  lib/armeabi/libangrybirds.so
```

```
SHA1-Digest:  pSdb9FYauyfjDUxM8L6JDmQk4qQ=
```

```
  inflating:  classes.dex
```

```
  inflating:  lib/armeabi/libangrybirds.so
```

```
  inflating:  lib/armeabi-v7a/libangrybirds.so
```

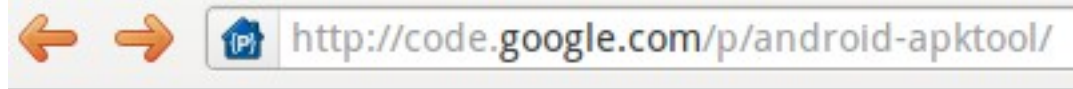
```
  inflating:  META-INF/MANIFEST.MF
```

```
  inflating:  META-INF/CERT.SF
```

```
  inflating:  META-INF/CERT.RSA
```



# AndroidManifest



```
$ unzip Angry+Birds.  
Archive:  Angry+Bird  
...  
...
```

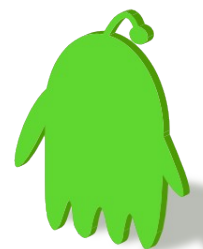


android-apktool  
A tool for reengineering Android apk files

```
    inflating: AndroidManifest.xml  
extracting:  resources.arsc
```

```
$ file AndroidManifest.xml  
AndroidManifest.xml: DBase 3 data file (2328 records)  
  
$ apktool d ../AngryBirds/Angry+Birds.apk  
I: Baksmaling...  
I: Loading resource table...  
...  
I: Decoding file-resources...  
I: Decoding values*/* XMLs...  
I: Done.  
I: Copying assets and libs...  
$ file Angry+Birds/AndroidManifest.xml  
Angry+Birds/AndroidManifest.xml: XML document text
```

50



Before performing reverse engineering, let's observe how Android system works



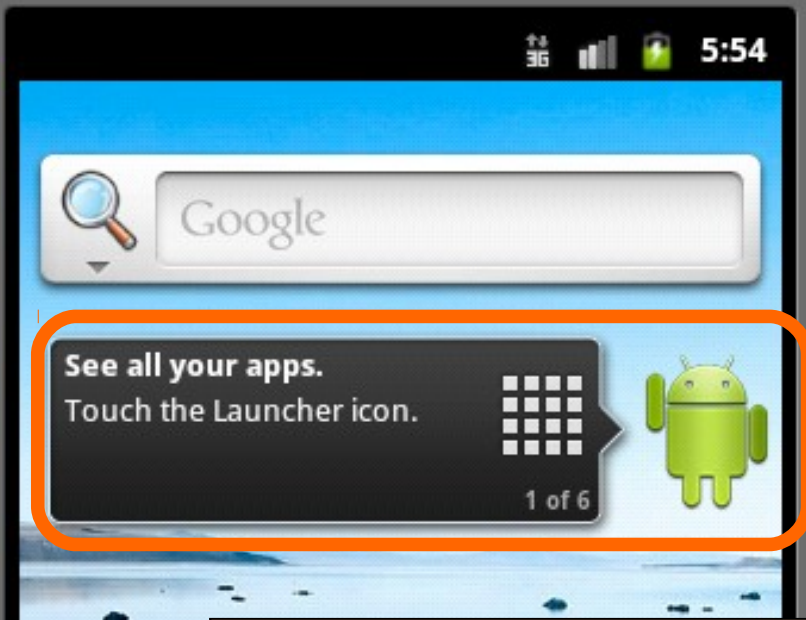
Virtual devices  
Installed packages  
Available packages

List of existing Android Virtual Devices located at /home/iscny/android/avd

AVD Name	Target Name	Platform	API Level	New...
✓ android2_3_3	Android 2.3.3	2.3.3	10	Edit

In this presentation, Android platform 2.3.3 is selected.

5554:android2\_3\_3



Widget

How can Launcher find widgets/activities and invoke them?



Android Launcher



# When installing FrozenBubble.apk

```
$ adb logcat -c
```

```
$ adb install -r FrozenBubble.apk
```

```
1222 KB/s (499568 bytes in 0.399s)
```

```
pkg: /data/local/tmp/FrozenBubble.apk
```

```
Success
```

```
$ adb logcat
```

```
D/AndroidRuntime( 329):
```

```
D/AndroidRuntime( 329): >>>>>>
```

```
AndroidRuntime START
```

```
com.android.internal.os.RuntimeInit <<<<<<
```

```
D/PackageParser( 60): Scanning
```

```
package: /data/app/vmdl110628918.tmp
```

```
...
```



# APK Installation Procedure

```
D/AndroidRuntime( 329):  
D/AndroidRuntime( 329): >>>>> AndroidRuntime START com.android.internal.os.RuntimeInit <<<<<<  
D/PackageParser( 60): Scanning package: /data/app/vmdl10628918.tmp  
I/PackageManager( 60): Removing non-system package:org.jfedor.frozenbubble  
I/ActivityManager( 60): Force stopping package org.jfedor.frozenbubble uid=10034  
D/PackageManager( 60): Scanning package org.jfedor.frozenbubble  
I/PackageManager( 60): Package org.jfedor.frozenbubble codePath changed from  
/data/app/org.jfedor.frozenbubble-2.apk to /data/app/org.jfedor.frozenbubble-1.apk; Retaining data and  
using new  
I/PackageManager( 60): Unpacking native libraries for /data/app/org.jfedor.frozenbubble-1.apk  
D/installd( 34): DexInv: --- BEGIN '/data/app/org.jfedor.frozenbubble-1.apk' ---  
D/dalvikvm( 340): DexOpt: load 54ms, verify+opt 137ms  
D/installd( 34): DexInv: --- END '/data/app/org.jfedor.frozenbubble-1.apk' (success) ---  
W/PackageManager( 60): Code path for pkg : org.jfedor.frozenbubble changing from  
/data/app/org.jfedor.frozenbubble-2.apk to /data/app/org.jfedor.frozenbubble-1.apk  
W/PackageManager( 60): Resource path for pkg : org.jfedor.frozenbubble changing from  
/data/app/org.jfedor.frozenbubble-2.apk to /data/app/org.jfedor.frozenbubble-1.apk  
D/PackageManager( 60): Activities: org.jfedor.frozenbubble.FrozenBubble  
I/ActivityManager( 60): Force stopping package org.jfedor.frozenbubble uid=10034  
I/installd( 34): move /data/dalvik-cache/data@app@org.jfedor.frozenbubble-1.apk@classes.dex ->  
/data/dalvik-cache/data@app@org.jfedor.frozenbubble-1.apk@classes.dex  
D/PackageManager( 60): New package installed in /data/app/org.jfedor.frozenbubble-1.apk  
I/ActivityManager( 60): Force stopping package org.jfedor.frozenbubble uid=10034  
I/installd( 34): unlink /data/dalvik-cache/data@app@org.jfedor.frozenbubble-2.apk@classes.dex  
D/AndroidRuntime( 329): Shutting down VM  
D/jdwp ( 329): abdb disconnected
```



# APK Installation Procedure

D/AndroidRuntime( 329)

Android Runtime performs init

D/AndroidRuntime( 329)

D/PackageParser( 60):

Package Manager detects APK and installs

I/PackageManager( 60)

I/ActivityManager( 60): Force stopping package org.jfedor.frozenbubble uid=10034

D/PackageManager( 60): Scanning package org.jfedor.frozenbubble

I/PackageManager( 60): Package org.jfedor.frozenbubble codePath changed from /data/app/org.jfedor.frozenbubble-2.apk to /data/app/org.jfedor.frozenbubble-1.apk; Retaining data and using new

I/PackageManager( 60): Unpacking native libraries for /data/app/org.jfedor.frozenbubble-1.apk

D/installd( 34): DexInv: --- BEGIN '/data/app/org.jfedor.frozenbubble-1.apk' ---

D/dalvikvm( 340): DexOpt:

DexOpt

(verify and optimize all of the classes in the DEX file)

D/installd( 34): DexInv: ---

W/PackageManager( 60): Resource path for pkg : org.jfedor.frozenbubble changing from /data/app/org.jfedor.frozenbubble-2.apk to /data/app/org.jfedor.frozenbubble-1.apk

W/PackageManager( 60): Resource path for pkg : org.jfedor.frozenbubble changing from /data/app/org.jfedor.frozenbubble-2.apk to /data/app/org.jfedor.frozenbubble-1.apk

D/PackageManager( 60): Activities: org.jfedor.frozenbubble.FrozenBubble

I/ActivityManager( 60): Force stopping package org.jfedor.frozenbubble uid=10034

I/installd( /data/dalvi

**Activities: org.jfedor.frozenbubble.FrozenBubble**

D/PackageManager( 60): New package installed in /data/app/org.jfedor.frozenbubble-1.apk

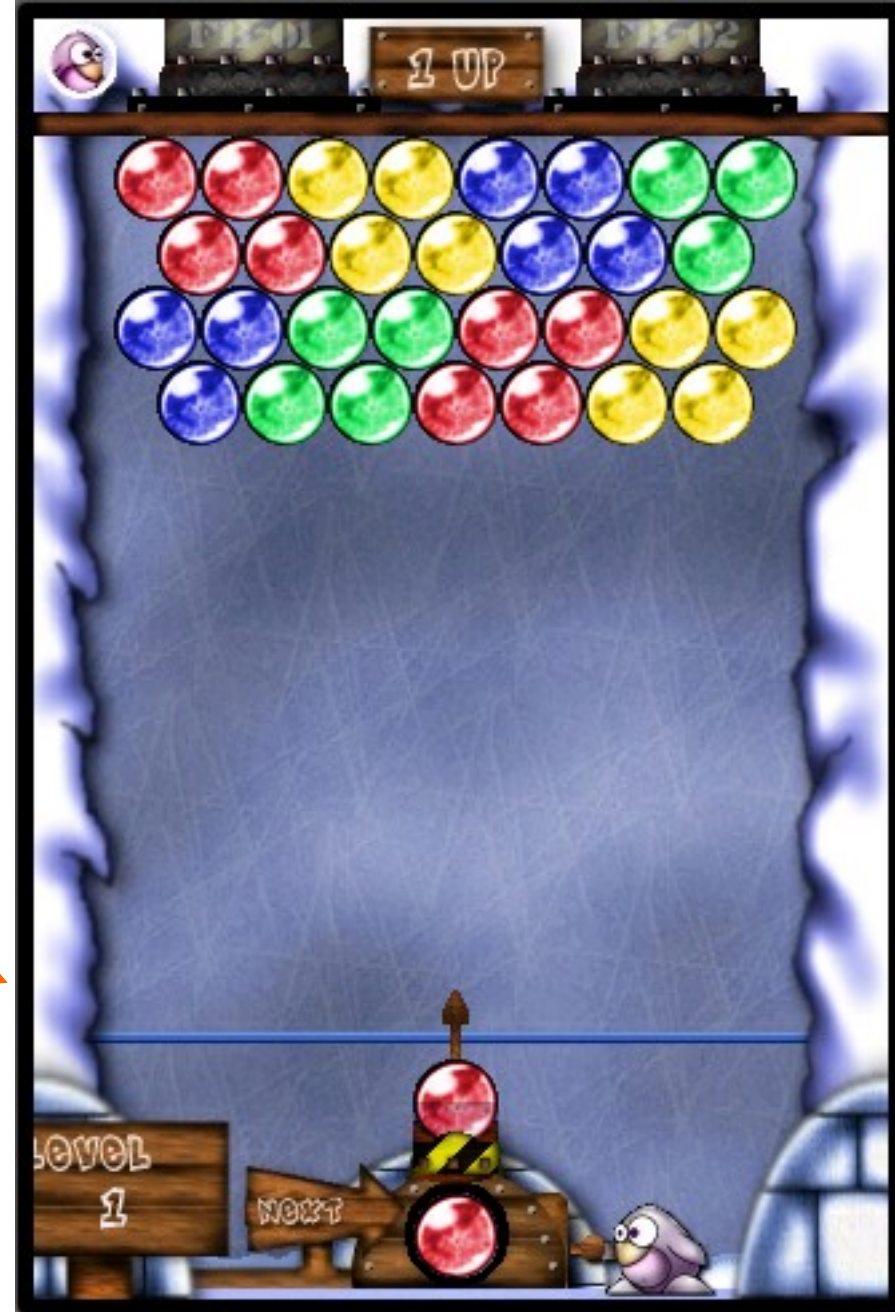
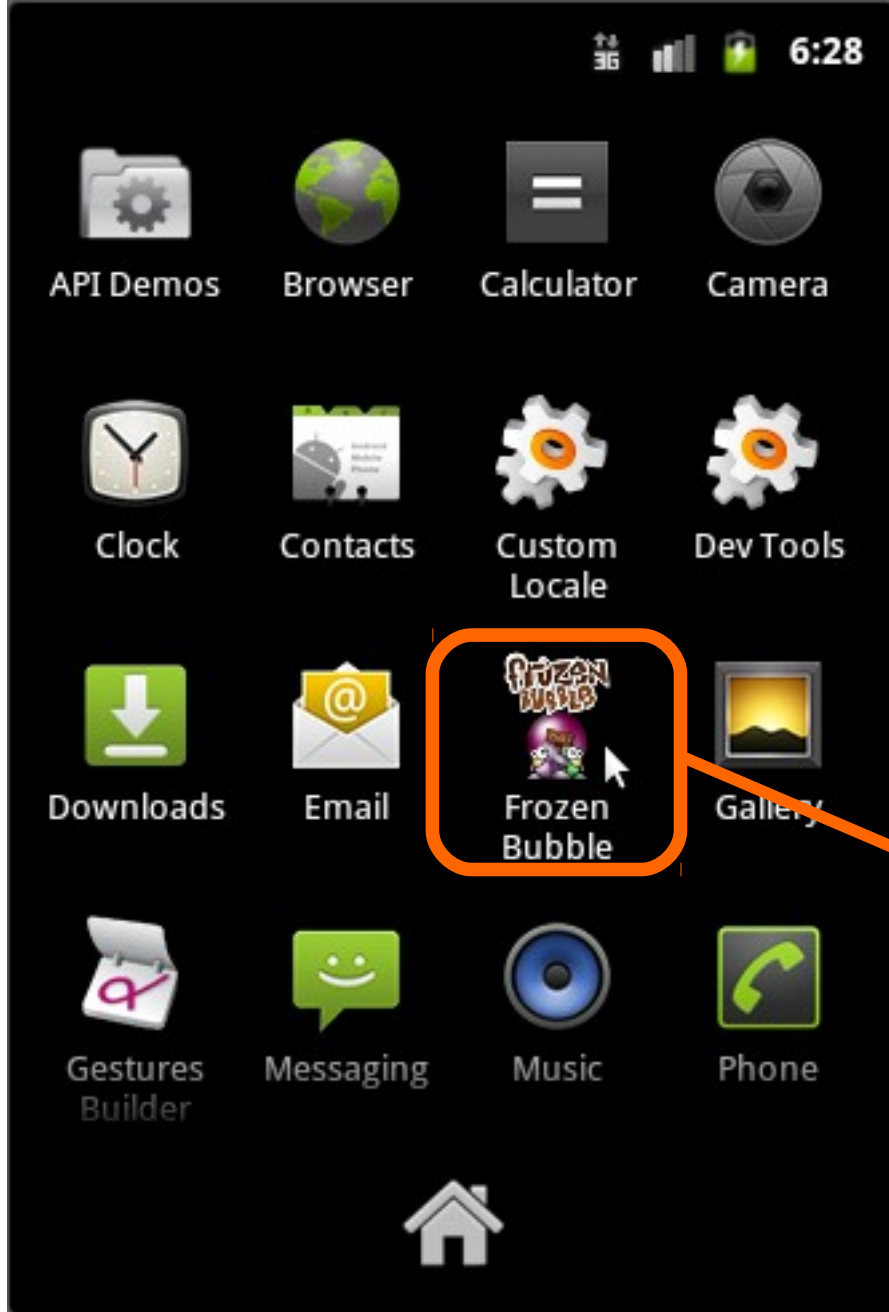
I/ActivityManager( 60): Force stopping package org.jfedor.frozenbubble uid=10034

I/installd( 34): unlink /data/dalvik-cache/data@app@org.jfedor.frozenbubble-2.apk@classes.dex

D/AndroidRuntime( 329): Shutting down VM

D/jdwp ( 329): abdb disconnected





```
I/ActivityManager( 60): Start proc org.jfedor.frozenbubble for activity  
org.jfedor.frozenbubble/.FrozenBubble: pid=356 uid=10034 gids={}
```

```
I/ActivityManager( 60): Displayed org.jfedor.frozenbubble/.FrozenBubble: +2s899ms
```

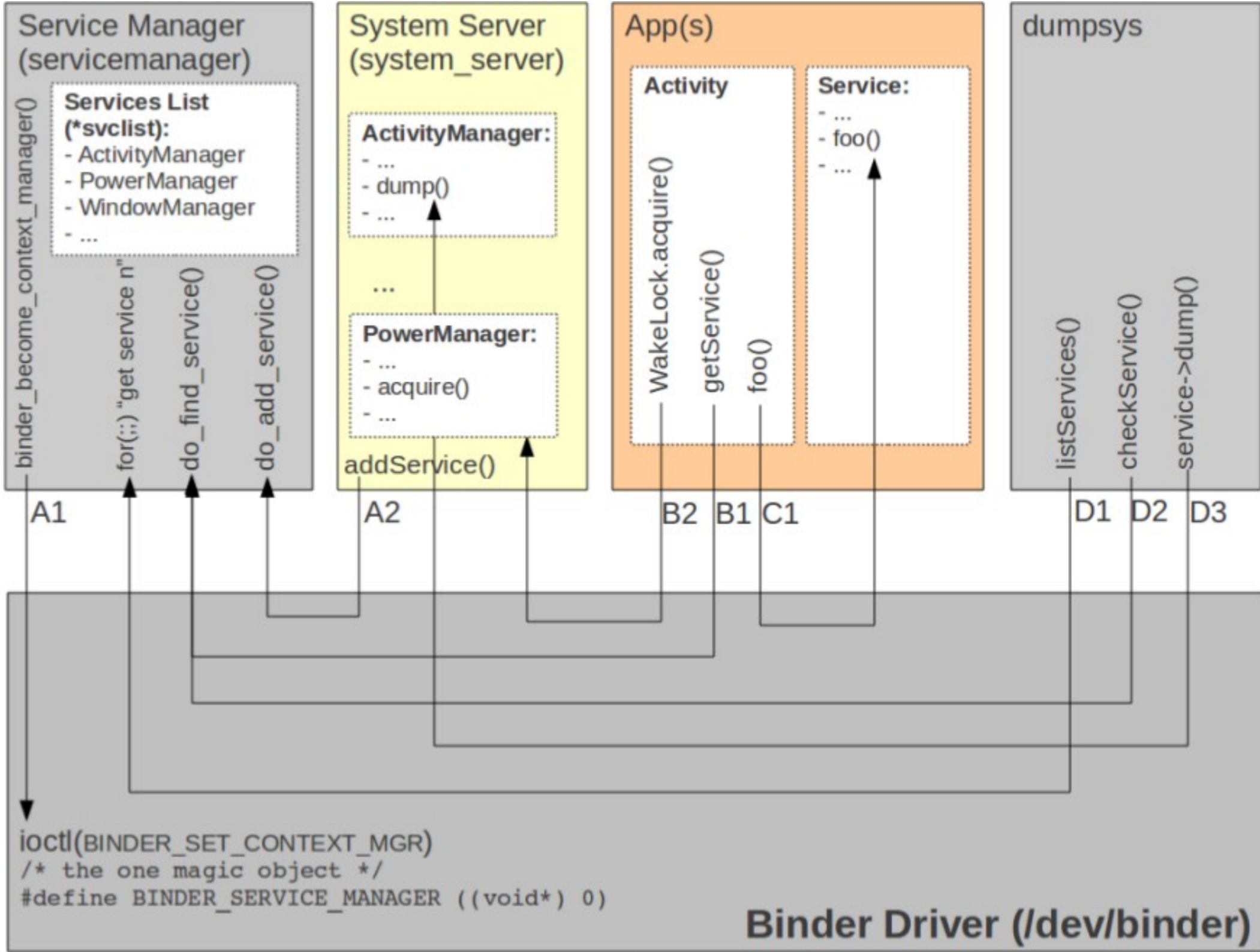


# Execute FrozenBubble from Android Launcher

```
$ adb shell am start \  
-e debug true \  
-a android.intent.action.MAIN \  
-c android.intent.category.LAUNCHER \  
-n org.jfedor.frozenbubble/.FrozenBubble
```

```
Starting: Intent  
{ act=android.intent.action.MAIN  
cat=[android.intent.category.LAUNCHER]  
cmp=org.jfedor.frozenbubble/.FrozenBubble (has extras)
```





# Execute FrozenBubble

```
$ adb shell dumpsys | grep -i bubble
```

```
name=org.jfedor.frozenbubble/org.jfedor.frozenbubble.FrozenBubble
```

```
  Intent { act=android.intent.action.PACKAGE_ADDED  
  dat=package:org.jfedor.frozenbubble flg=0x10000000 (has  
  extras) }
```

```
* TaskRecord{40744ad0 #4 A org.jfedor.frozenbubble}  
  affinity=org.jfedor.frozenbubble  
  intent={act=android.intent.action.MAIN  
  cat=[android.intent.category.LAUNCHER] flg=0x10200000  
  cmp=org.jfedor.frozenbubble/.FrozenBubble}  
  realActivity=org.jfedor.frozenbubble/.FrozenBubble
```

```
...
```



# ActivityManager

- Start new Activities and Services
- Fetch Content Providers
- Intent broadcasting
- OOM adj. Maintenance
- ANR (Application Not Responding)
- Permissions
- Task management
- Lifecycle management



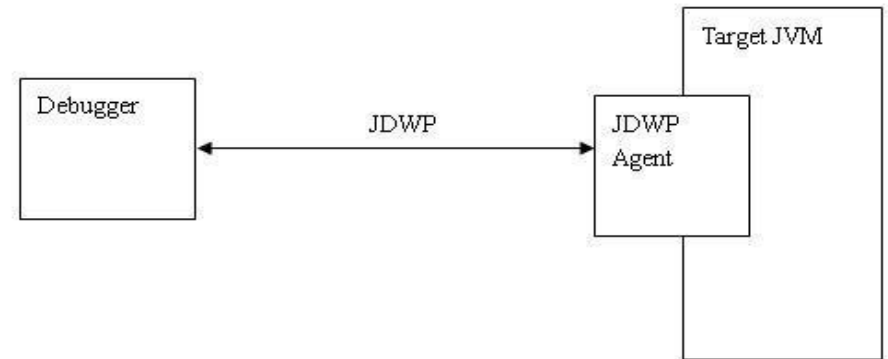
# ActivityManager

- starting new app from Launcher:
  - onClick(Launcher)
  - startActivity
  - <Binder>
  - ActivityManagerService
  - startViaZygote(Process.java)
  - <Socket>
  - Zygote



# Use JDB to Trace Android Application

```
#!/bin/bash
adb wait-for-device
adb shell am start \
  -e debug true \
  -a android.intent.action.MAIN \
  -c android.intent.category.LAUNCHER \
  -n org.jfedor.frozenbubble/.FrozenBubble &
debug_port=$(adb jdwp | tail -1);
adb forward tcp:29882 jdwp:$debug_port &
jdb -J-Duser.home=. -connect \
  com.sun.jdi.SocketAttach:hostname=localhost,port=29882 &
```



In APK manifest, `debuggable="true"`

JDWP: Java Debug Wire Protocol



# JDB usage

> **threads**

Group system:

```
(java.lang.Thread)0xc14050e388 <6> Compiler cond. Waiting
(java.lang.Thread)0xc14050e218 <4> Signal Catcher cond. waiting
(java.lang.Thread)0xc14050e170 <3> GC cond. waiting
(java.lang.Thread)0xc14050e0b8 <2> HeapWorker cond. waiting
```

Group main:

```
(java.lang.Thread)0xc14001f1a8 <1> main running
(org.jfedor.frozenbubble.GameView$GameThread)0xc14051e300
<11> Thread-10 running
(java.lang.Thread)0xc14050f670 <10> SoundPool running
(java.lang.Thread)0xc14050f568 <9> SoundPoolThread running
(java.lang.Thread)0xc140511db8 <8> Binder Thread #2 running
(java.lang.Thread)0xc140510118 <7> Binder Thread #1 running
```

> **suspend 0xc14051e300**

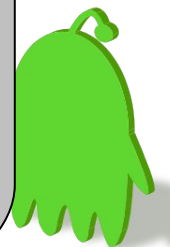
> **thread 0xc14051e300**

<11> Thread-10[1] **where**

[1] android.view.SurfaceView\$3.internalLockCanvas (SurfaceView.java:789)

[2] android.view.SurfaceView\$3.lockCanvas (SurfaceView.java:745)

[3] org.jfedor.frozenbubble.GameView\$GameThread.run (GameView.java:415)





Name

com.android.phone	114	⚙️
com.android.systemui	117	⚙️
com.android.launcher	121	⚙️
com.android.settings	158	⚙️
android.process.acore	177	⚙️
com.android.deskclock	187	⚙️
android.process.media	206	⚙️
com.android.mms	220	⚙️
com.android.email	240	⚙️
com.android.quicksearchbox	252	⚙️
com.android.music	263	⚙️
com.android.protips	274	⚙️
org.jfedor.frozenbubble	293	⚙️

Info Threads VM Heap Allocation Tracker Sysinfo

DDM-aware? yes  
 App description: org.jfedor.frozenbubble  
 VM version: Dalvik v1.4.0  
 Process ID: 293  
 Supports Profiling Control: Yes  
 Supports HPROF Control: Yes



DDMS = Dalvik Debug Monitor Server

Name		
emulator-5554	Online	
system_process	60	
jp.co.omronsoft.openwnn	110	
com.android.phone	114	
com.android.systemui	117	
com.android.launcher	121	
com.android.settings	158	
android.process.acore	177	
com.android.deskclock	187	
android.process.media	206	
com.android.mms	220	
com.android.email	240	
com.android.quicksearchbox	252	
com.android.music	263	
com.android.protips	274	
org.jfedor.frozenbubble	293	

Info	Threads	VM Heap	Allocation Tracker	Sysinfo	Emulator Control
ID	Tid	Status	utime	stime	Name
*5	298	running	11	10	JDWP
*6	299	vmwait	20	19	Compiler
7	300	native	0	0	Binder Thread #1
8	301	native	0	0	Binder Thread #2
9	303	native	1	2	SoundPoolThread
10	302	native	0	0	SoundPool
11	314	native	19831	743	Thread-10

Refresh Sat Jul 23 09:13:31 CST 2011

Class	Method	File
android.view.Surface	lockCanvasNative	Surface.java
android.view.Surface	lockCanvas	Surface.java
android.view.SurfaceView	internalLockCanvas	SurfaceView.java
android.view.SurfaceView	lockCanvas	SurfaceView.java
org.jfedor.frozenbubble	run	GameView.java

```
(JDB)
> thread 0xc14051e300
<11> Thread-10[1] where
[1] android.view.SurfaceView$3.internalLockCanvas (SurfaceView.java:789)
[2] android.view.SurfaceView$3.lockCanvas (SurfaceView.java:745)
[3] org.jfedor.frozenbubble.GameView$GameThread.run (GameView.java:415)
```



# hierarchyviewer: Traverse widgets

The screenshot displays an Android application window with the title "5554: Hierarchy Viewer". The application shows a game scene with a grid of colorful spheres (red, yellow, blue, green) at the top, a wooden sign with "1 UP" above it, and a character at the bottom. The Hierarchy Viewer is open on the right, showing a tree of widgets. The root widget is a `FrameLayout` with ID `@4051cba8`. It contains a `GameView` widget with ID `@4051d6c8` and ID `id/game`. The `GameView` widget is highlighted with a green border. A tooltip for the `GameView` widget shows the following performance metrics:

- 1 view
- Measure: 0.059 ms
- Layout: 0.097 ms
- Draw: 0.676 ms

The widget hierarchy is as follows:

```
graph TD; Root[FrameLayout @4051cba8] --- GameView[GameView @4051d6c8 id/game];
```

The image shows a side-by-side comparison of an Android application's runtime behavior and its visual hierarchy. On the left, a gallery application displays a message "No media found." with an icon of a camera and photo. On the right, the Hierarchy Viewer tool displays the underlying view structure:

- TextView** (@40588f08, id/title) is connected to the root.
- RelativeLayout** (@4058a3c0, id/root) is the root view, containing:
  - GridView** (@4058aa00, id/albums) with three green dots.
  - RelativeLayout** (@40598208, id/no\_images) with one red, one green, and one yellow dot. An orange arrow points from this view to the text box below.
- The inner **RelativeLayout** (@40598208) contains:
  - ImageView** (@40598878, id/no\_pictures\_image) with three green dots.
  - TextView** (@40598d70) with three yellow dots.

Figure out the association between APK resources and runtime behavior.

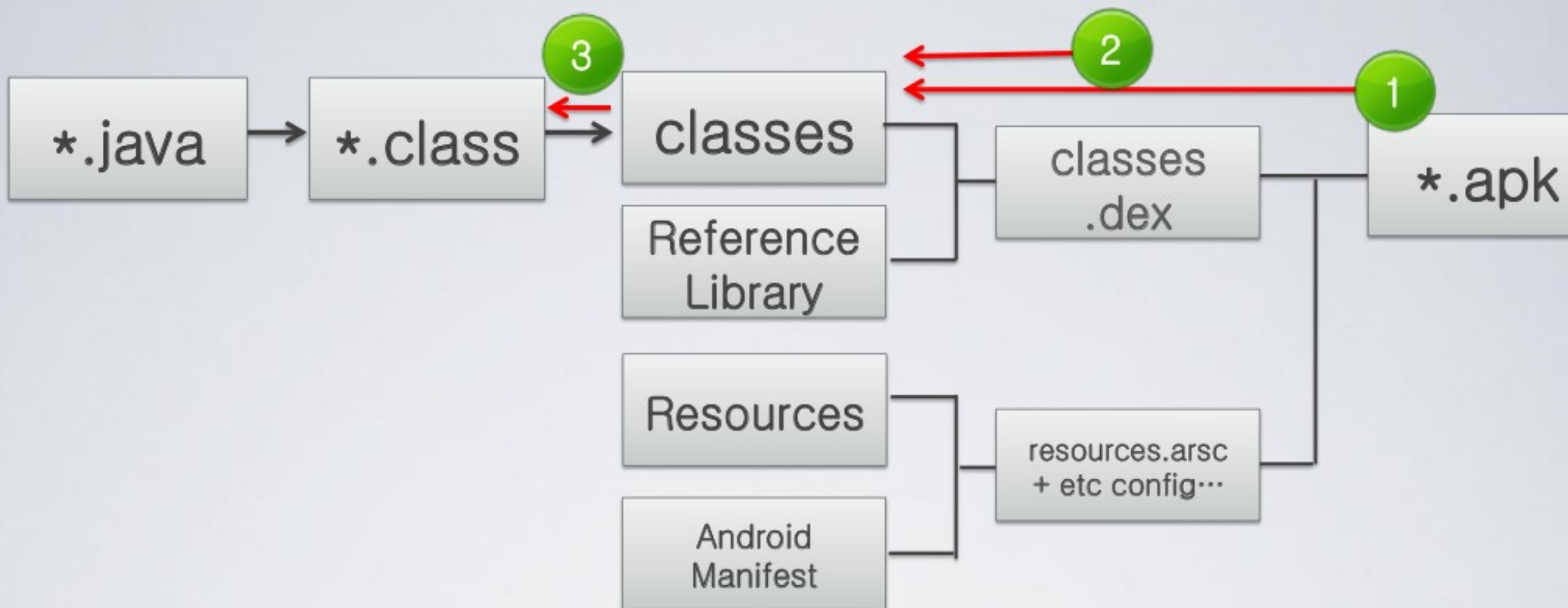


Decompile / Disassembly





- apktool: <http://code.google.com/p/android-apktool/>
- dex2jar: <http://code.google.com/p/dex2jar/>
- Jad / jd-gui: <http://java.decompiler.free.fr/>



smali : assembler/disassembler for Android's dex format

- <http://code.google.com/p/smali/>
- smali: The assembler
- baksmali: The disassembler
- Fully integrated in apktool

```
$ apktool d ../AngryBirds/Angry+Birds.apk
I: Baksmaling...
I: Loading resource table...
...
I: Decoding file-resources...
I: Decoding values*/* XMLs...
I: Done.
I: Copying assets and libs...
```



# Java bytecode vs. Dalvik bytecode

```
public int method( int i1, int i2 ) {  
    int i3 = i1 * i2;  
    return i3 * 2;  
}
```

(stack vs. register)

```
.var 0 is "this"  
.var 1 is argument #1  
.var 2 is argument #2
```

```
this: v1 (Ltest2;)  
parameter[0] : v2 (I)  
parameter[1] : v3 (I)
```

```
method public method(II)I  
    iload_1  
    iload_2  
    imul  
    istore_3  
    iload_3  
    iconst_2  
    imul  
    ireturn  
.end method
```

Java

```
.method public method(II)I  
    mul-int v0,v2,v3  
    mul-int/lit-8 v0,v0,2  
    return v0  
.end method
```

Dalvik

# Dalvik Register frames

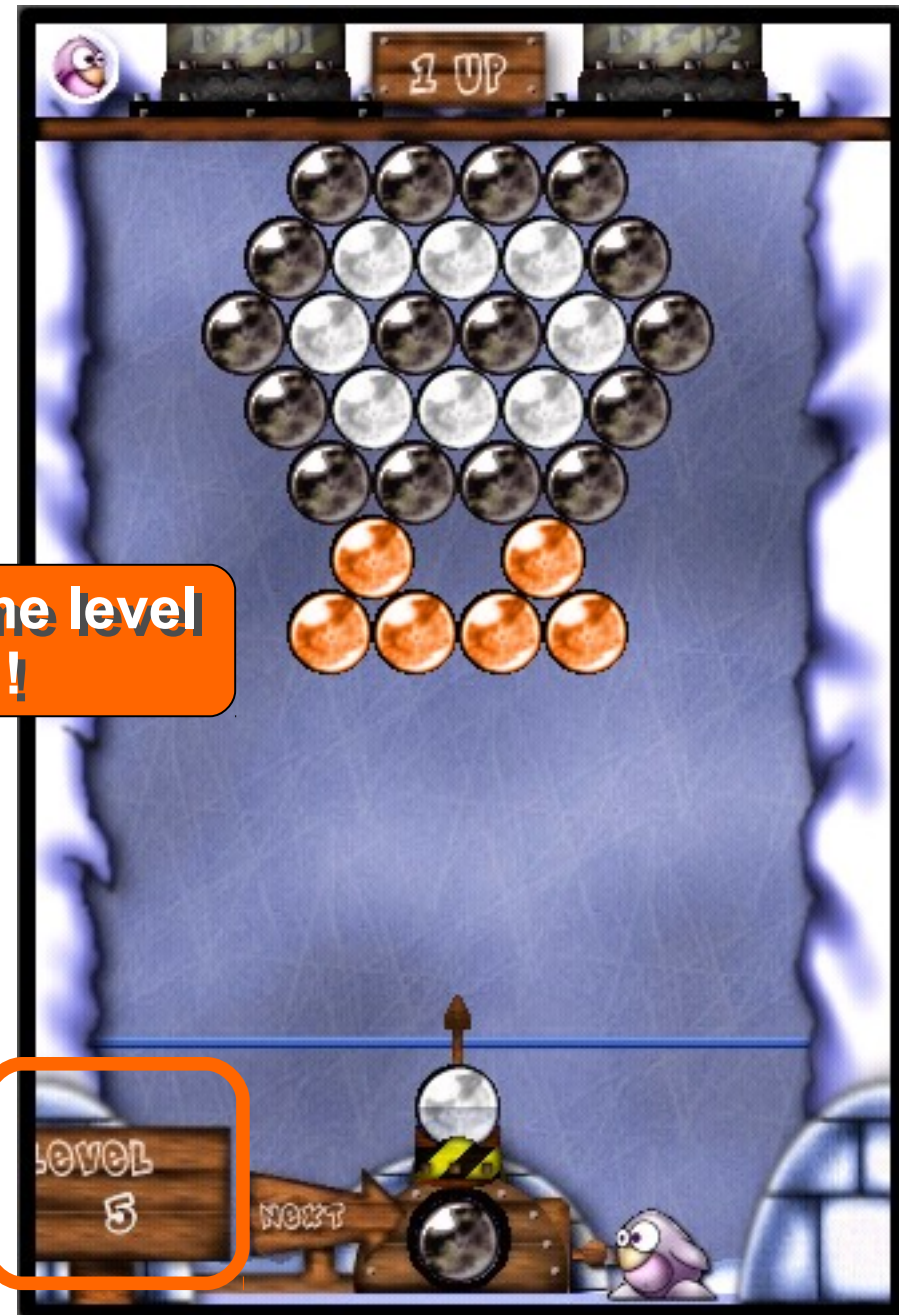
- Dalvik registers behave more like local variables
- Each method has a fresh set of registers.
- Invoked methods don't affect the registers of invoking methods.



# Practice: Level Up



**Change initial game level  
From 1 to 5 !**



# Disassembly

```
$ mkdir workspace smali-src
$ cd workspace
$ unzip ../FrozenBubble-orig.apk
Archive:  ../FrozenBubble-orig.apk
  inflating: META-INF/MANIFEST.MF
  inflating: META-INF/CERT.SF
  inflating: META-INF/CERT.RSA
  inflating: AndroidManifest.xml
...
extracting: resources.arsc
$ bin/baksmali -o smali-src workspace/classes.dex
```

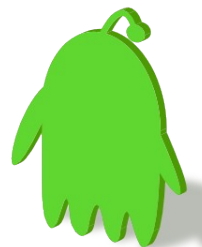


`org.jfedor.frozenbubble/.FrozenBubble`

```
smali-src$ find
```

```
./org/jfedor/frozenbubble/FrozenBubble.smali  
./org/jfedor/frozenbubble/R$id.smali  
./org/jfedor/frozenbubble/GameView.smali  
./org/jfedor/frozenbubble/SoundManager.smali  
./org/jfedor/frozenbubble/LaunchBubbleSprite.smali  
./org/jfedor/frozenbubble/Compressor.smali  
./org/jfedor/frozenbubble/R$attr.smali  
./org/jfedor/frozenbubble/BubbleFont.smali  
./org/jfedor/frozenbubble/PenguinSprite.smali  
./org/jfedor/frozenbubble/GameView$GameThread.smali  
./org/jfedor/frozenbubble/LevelManager.smali  
./org/jfedor/frozenbubble/BubbleSprite.smali  
./org/jfedor/frozenbubble/R$string.smali  
...
```

Generated  
from resources



```
smali-src$ grep "\.method"  
org/jfedor/frozenbubble/LevelManager.smali  
.method public constructor <init>([BI)V  
.method private getLevel(Ljava/lang/String;)[[B  
.method public getCurrentLevel()[[B  
.method public getLevelIndex()I  
.method public goToFirstLevel()V  
.method public goToNextLevel()V  
.method public restoreState(Landroid/os/Bundle;)V  
.method public saveState(Landroid/os/Bundle;)V
```

List the methods implemented in class LevelManager



```
.method private getLevel(Ljava/lang/String;) [[B  
    → private byte[][] getLevel(String data)  
  
.method public goToNextLevel() V  
    → public void goToNextLevel();
```

- Base types
  - I : int / J : long / S : short
  - Z : boolean
  - D : double / F : float
  - C : char
  - V : void (when return value)
- Classes: Ljava/lang/Object;
- Arrays: [I, [Ljava/lang/Object;, [[I



# Dalvik::Methods

- Rich meta-information is assigned to Dalvik methods
- Method meta-information:
  - Signature
  - Try-catch information
  - Annotations
  - Number of registers used
  - Debug information
    - Line numbers
    - Local variable lifetime



```
smali-src$ grep -r goToFirstLevel *  
org/jfedor/frozenbubble/GameView$GameThread.smali:  
invoke-virtual {v2},  
    Lorg/jfedor/frozenbubble/LevelManager;->goToFirstLevel()V  
org/jfedor/frozenbubble/LevelManager.smali:  
.method public goToFirstLevel()V
```

That the first argument of the method invocation is “this” as this is a non-static method.



# GameView\$GameThread.smali

```
.method public newGame()V
    . . .
    move-object/from16 v0, p0

    iget-object v0, v0,
Lorg/jfedor/frozenbubble/GameView$GameThread;-
>mLevelManager:Lorg/jfedor/frozenbubble/LevelManager;

    move-object v2, v0

    invoke-virtual {v2},
Lorg/jfedor/frozenbubble/LevelManager;->goToFirstLevel ()V
```

Equals to Java:

```
objLevelManager.goToFirstLevel();
```



# LevelManager.smali

```
.method public goToFirstLevel()V
```

```
    .registers 2
```

```
    .prologue
```

```
    .line 175
```

```
    const/4 v0, 0x0
```

```
    iput v0, p0,
```

```
Lorg/jfedor/frozenbubble/LevelManager; ->currentLevel:I
```

```
    .line 176
```

```
    return-void
```

```
.end method
```

Equals to Java:

```
public class LevelManager {  
    ...  
    public void goToFirstLevel() {  
        currentLevel = 0;  
    }  
    ...  
}
```

Equals to Java:

```
currentLevel = 0;
```

Constants to registers: const/4, const/16, const, const/high16, const-wide/16, const-wide/32, const-wide, const-wide/high16, const-string, const-class



# Modify constructor of GameView::GameThread()

- Look up output in GameView\$GameThread.smali

```
.class Lorg/jfedor/frozenbubble/GameView$GameThread;
```

```
.super Ljava/lang/Thread;
```

```
.annotation system Ldalvik/annotation/InnerClass;
```

```
    accessFlags = 0x0
```

```
    name = "GameThread"
```

```
.end annotation
```

```
# direct methods
```

```
.method public constructor
```

```
<init>(Lorg/jfedor/frozenbubble/GameView;Landroid/view/SurfaceHolder;[B)V
```



# Modify constructor of GameView::GameThread()

- Look up output in GameView\$GameThread.smali

# direct methods

.method public constructor

```
<init>(Lorg/jfedor/frozenbubble/GameView;Landroid/view/SurfaceHolder;[B)V
```

Equals to Java:

```
class GameView ??? {  
    class GameThread extends Thread {  
        public GameThread(SurfaceHolder s,  
                           byte[] b,  
                           int I) {
```



- Look up output in GameView.smali

```
.class Lorg/jfedor/frozenbubble/GameView;
.super Landroid/view/SurfaceView;
# interfaces
.implements Landroid/view/SurfaceHolder$Callback;
```
- Look up output in GameView\$GameThread.smali

```
.class Lorg/jfedor/frozenbubble/GameView$GameThread;
.super Ljava/lang/Thread;
```

Equals to Java:

```
class GameView extends SurfaceView
    implements SurfaceHolder.Callback {
    class GameThread extends Thread {
```

# Implementation of GameView::GameThread()

- Check GameView::public GameThread(SurfaceHolder s, byte[] b, int l)

```
const-string v3, "level"
```

```
const/4 v4, 0x0
```

```
move-object/from16 v0, v25
```

```
move-object v1, v3
```

```
move v2, v4
```

```
invoke-interface {v0, v1, v2},  
Landroid/content/SharedPreferences;-  
>getInt(Ljava/lang/String;I)I
```

```
move-result p4
```

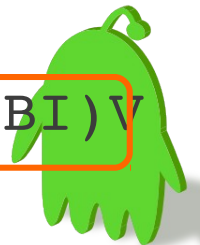
Invoke constructor of LevelManager

```
new-instance v3, Lorg/jfedor/frozenbubble/LevelManager;
```

```
move-object v0, v3
```

```
move-object/from16 v1, v22
```

```
move/from16 v2, p4 invoke-direct {v0, v1, v2},  
Lorg/jfedor/frozenbubble/LevelManager;-><init>([BI)V
```



# Register v1 related code

```
const-string v3, "level"
```

```
const/4 v4, 0x0
```

```
move-object/from16 v0, v25
```

```
move-object v1, v3
```

```
move v2, v4
```

```
invoke-interface {v0, v1, v2},  
Landroid/content/SharedPreferences;-  
>getInt(Ljava/lang/String;I)I
```

```
move-result p4
```

```
new-instance v3,  
Lorg/jfedor/frozenbubble/LevelManager;
```

```
move-object v0, v3
```

```
move-object/from16 v1, v22
```

```
move/from16 v2, p4
```

```
invoke-direct {v0, v1, v2},  
Lorg/jfedor/frozenbubble/LevelManager;-><init>([BI)V
```



# Register v2 related code

```
const-string v3, "level"
```

```
const/4 v4, 0x0
```

```
move-object/from16 v0, v25
```

```
move-object v1, v3
```

```
move v2, v4
```

"0x0" is passed to LevelManager's constructor as parameter

```
invoke-interface {v0, v1, v2},  
Landroid/content/SharedPreferences;-  
>getInt(Ljava/lang/String;I)I
```

```
move-result p4
```

```
new-instance v3,  
Lorg/jfedor/frozenbubble/LevelManager;
```

```
move-object v0, v3
```

```
move-object/from16 v1, v22
```

```
move/from16 v2, p4
```

```
invoke-direct {v0, v1, v2},  
Lorg/jfedor/frozenbubble/LevelManager;-><init>([BI)V
```



# Recall the grep results

```
smali-src$ grep "\.method"  
org/jfedor/frozenbubble/LevelManager.smali  
.method public constructor <init>([BI)V  
.method private getLevel(Ljava/lang/String;)[[B  
.method public getCurrentLevel()[[B  
.method public getLevelIndex()I  
.method public goToFirstLevel()V  
.method public goToNextLevel()V  
.method public restoreState(Landroid/os/Bundle;)V  
.method public saveState(Landroid/os/Bundle;)V
```

Equals to Java:

```
public class LevelManager {  
    ...  
    public LevelManager(byte[] b, int i)
```

# Register v2 related code

```
const-string v3,  
const/4 v4, 0x0
```

p4 reserve the result after method invocation.

```
move-object/from16 v0, v25
```

```
move-object v1, v3
```

```
move v2, v4
```

```
invoke-interface {v0, v1, v2},  
Landroid/content/SharedPreferences;  
>getInt(Ljava/lang/String,I)I
```

```
move-result p4
```

```
new-instance v3,  
Lorg/jfedor/frozenbubble/LevelManager;
```

Therefore, v2 has return value of method  
android.content.SharedPreferences.getInt()

```
move-object v0, v3
```

```
move-object/from16 v1, v22
```

```
move/from16 v2, p4
```

```
invoke-direct {v0, v1, v2},  
Lorg/jfedor/frozenbubble/LevelManager; -><init>([BI)V
```



# Modify!!!

- Check GameView::public GameThread(SurfaceHolder s, byte[] b, int l)

```
const-string v3, "level"
```

Change value from 0x0 to 0x4

```
const/4 v4, 0x0
```

```
move-object/from16 v0, v25
```

```
move-object v1, v3
```

```
move v2, v4
```

Remove!

```
invoke-interface {v0, v1, v2},  
Landroid/content/SharedPreferences;->getInt(Ljava/lang/String;I)I
```

```
move-result p4
```

```
new-instance v3, Lorg/jfedor/frozenbubble/LevelManager;
```

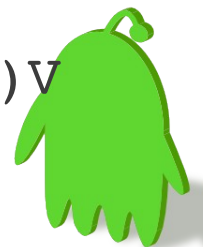
```
move-object v0, v3
```

```
move-object/from16 v1, v22
```

Remove!

```
move/from16 v2, p4
```

```
invoke-direct {v0, v1, v2},  
Lorg/jfedor/frozenbubble/LevelManager;-><init>([BI)V
```



# Real World Tasks



- ODEX (Optimized DEX)
  - platform-specific optimizations:
    - specific bytecode
    - vtables for methods
    - offsets for attributes
    - method inlining
- JNI
  - JNIEnv
- Native Activity
- Key signing



# DEX Optimizations

- Before being executed by Dalvik, DEX files are optimized.
  - Normally it happens before the first execution of code from the DEX file
  - Combined with the bytecode verification
  - In case of DEX files from APKs, when the application is launched for the first time.
- Process
  - The dexopt process (which is actually a backdoor of Dalvik) loads the DEX, replaces certain instructions with their optimized counterparts
  - Then writes the resulting optimized DEX (ODEX) file into the /data/dalvik-cache directory
  - It is assumed that the optimized DEX file will be executed on the same VM that optimized it. ODEX files are NOT portable across VMs.



# dexopt: Instruction Rewritten

- Virtual (non-private, non-constructor, non-static methods)

**invoke-virtual** <symbolic method name> → **invoke-virtual-quick** <vtable index>

Before:

```
invoke-virtual  
{v1,v2},java/lang/StringBuilder/append;append(Ljava/lang/String;)Ljava/lang/StringBuilder;
```

After:

```
invoke-virtual-quick {v1,v2},vtable #0x3b
```

- Frequently used methods

**invoke-virtual/direct/static** <symbolic method name> → **execute-inline** <method index>

– Before:

```
invoke-virtual {v2},java/lang/String/length
```

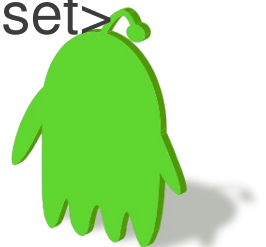
– After:

```
execute-inline {v2},inline #0x4
```

- instance fields: **iget/iput** <field name> → **iget/iput-quick** <memory offset>

– Before: `iget-object v3,v5,android/app/Activity.mComponent`

– After: `iget-object-quick v3,v5,[obj+0x28]`



# Meaning of DEX Optimizations

- Sets byte ordering and structure alignment
- Aligns the member variables to 32-bits / 64-bits
- boundary (the structures in the DEX/ODEX file itself are 32-bit aligned)
- Significant optimizations because of the elimination of symbolic field/method lookup at runtime.
- Aid of Just-In-Time compiler



JNI specificities can ease reversing

- 1- get the function signature in Java
- 2- use IDA to generate a TIL file from jni.h
- 3- assign the structure to the right variable
- 4- see function calls directly
- 5- do the same in Hex-Rays

# Reverse: JNI

```
.text:0000173C      MOV     R8, R3
.text:0000173E      MOVS   R3, 0x2A4
.text:00001742      LDR    R3, [R2, R3]
.text:00001744      MOV    R9, R1
.text:00001746      MOVS   R2, #0
.text:00001748      LDR    R1, [SP, #0x1D0+var_1BC]
.text:0000174A      MOVS   R7, R0
.text:0000174C      BLX    R3
```

1

```
.text:0000173E      MOVS   R3, 0x2A4
.text:00001742      LDR    R3, [R2, R3]
.text:00001744      MOV    R9, R1
.text:00001746      MOVS   R2, #0
.text:00001748      LDR    R1, [SP, #0x1D0+var_1BC]
.text:0000174A      MOVS   R7, R0
.text:0000174C      BLX    R3
.text:0000174E      LDR    R3, =0x1010100100
```

2

Chart of xrefs from  
[JNIInterface.GetStringUTFChars  
Use standard symbolic constant

676	H
01244	
0b1010100100	B

```
.text:0000173C      MOV     R8, R3
.text:0000173E      MOVS   R3, JNIInterface.GetStringUTFChars
.text:00001742      LDR    R3, [R2, R3]
.text:00001744      MOV    R9, R1
.text:00001746      MOVS   R2, #0
.text:00001748      LDR    R1, [SP, #0x1D0+var_1BC]
.text:0000174A      MOVS   R7, R0
.text:0000174C      BLX    R3
```

3

VSSI

## Reverse: JNI with Hex-Rays

```
v13 = strlen(v23);
8j3zIX(&v25, v23, v13, 0);
8j3zIX(&v25, v36, 32, 0);
sd1Hj(&v25, &v32);
v22 = &v32;
((void (__fastcall *)(JNIEnv *, int, _DWORD, signed int))(*jnienv_)->SetByteArrayRegion)(jnienv_, v24, 0, 32);
i = 0;
do
{
    v26[i] = v33[i] ^ v36[i & 7];
    ++i;
}
while ( i != 17 );
v27 = v26[0] ^ 0x31;
v28 = v26[1] ^ 0x2C;
v29 = v26[2] ^ 0x59;
v30 = v26[3] ^ 0x2F;
v15 = ((int (__fastcall *)(JNIEnv *, unsigned __int8 *))(*jnienv_)->FindClass)(jnienv_, v26);
```

```
(*jnienv_)->SetByteArrayRegion)(jnienv_, v24, 0, 32);
```

# Further Considerations

- Optimizing, Obfuscating, and Shrinking your Android Applications with ProGuard  
<http://www.androidengineer.com/2010/07/optimizing-obfuscating-and-shrinking.html>
- Missions:
  - Obfuscation
  - Optimizing
- ProGuard

```
<target name="-dex" depends="compile,optimize">
<target name="-post-compile">
  <antcall target="optimize"/>
</target>
```
- Google's License Verification Library (LVL)  
-keep class com.android.vending.licensing.ILicensingService



# Reference

- Analysis of Dalvik Virtual Machine and Class Path Library, Security Engineering Research Group, Institute of Management Sciences Peshawar, Pakistan (2009)  
<http://serg.imsciences.edu.pk>
- Android: forensics and reverse engineering, Raphaël Rigo – ANSSI (2010)
- <http://code.google.com/p/dex2jar/>
- <http://java.decompiler.free.fr/>





<http://0xlab.org>