



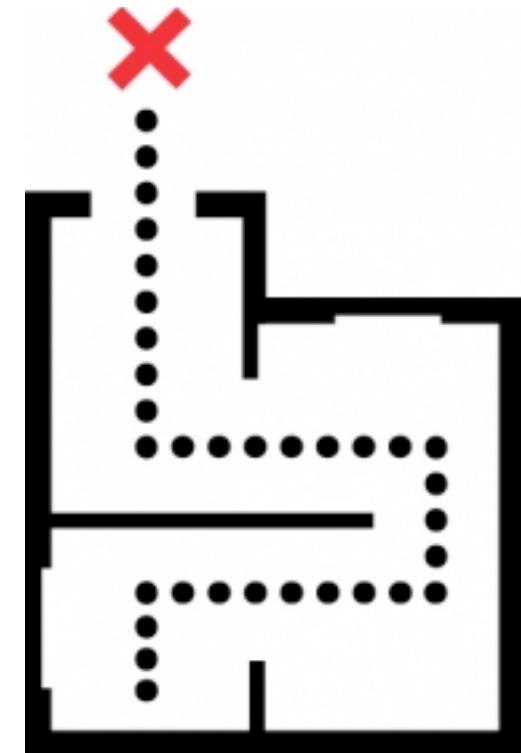
Android Internals

Marko Gargenta

Marakana

Agenda

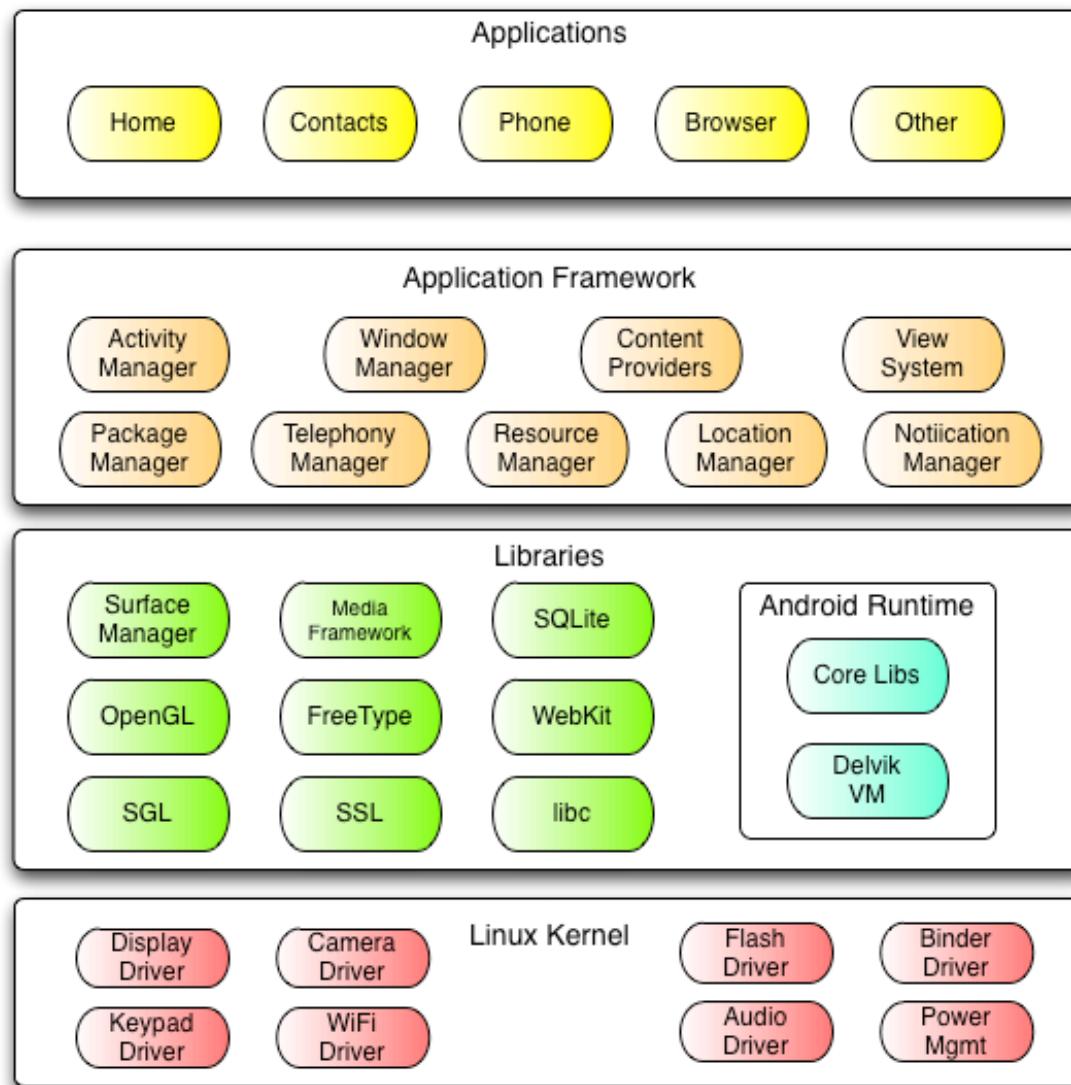
- Android Stack
- Operating System Features
- Working with Hardware
- Android Startup & Runtime
- Native Development Kit
- Debugging
- Summary





ANDROID STACK

The Stack



Linux Kernel

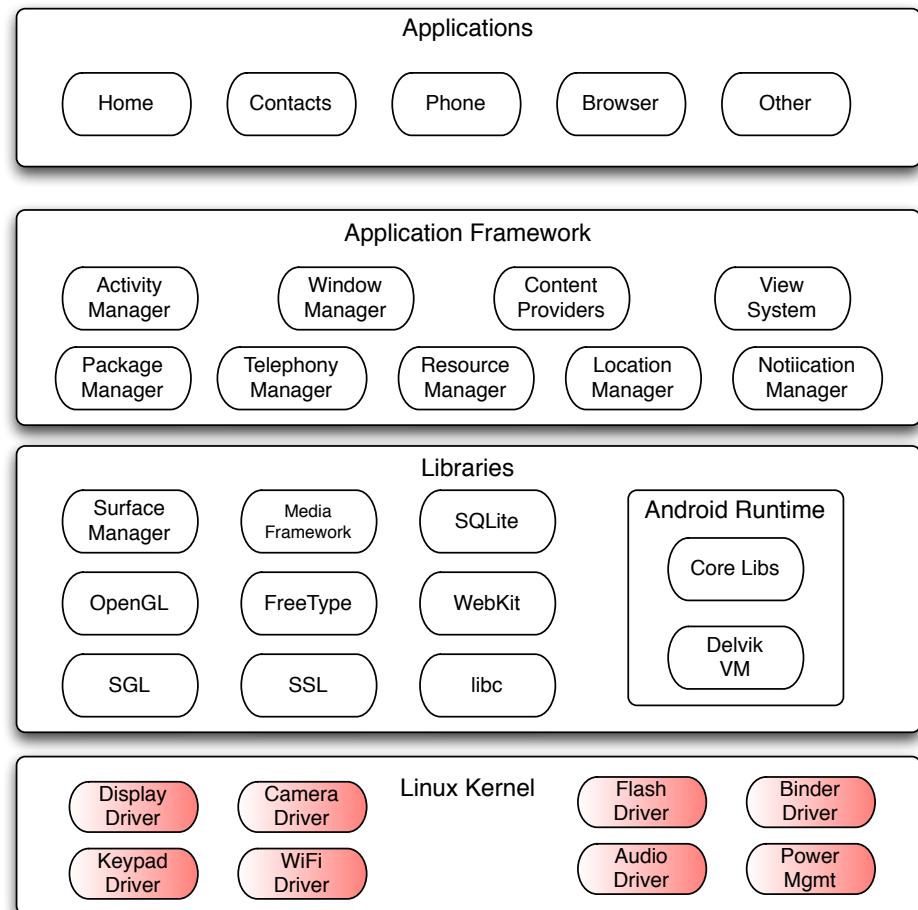
Android runs on Linux.

Linux provides as well as:

- Hardware abstraction layer
- Memory management
- Process management
- Networking

Users never see Linux sub system

The adb shell command opens
Linux shell



Native Libraries

Bionic, a super fast and small
GPL-based libc library optimized
for embedded use

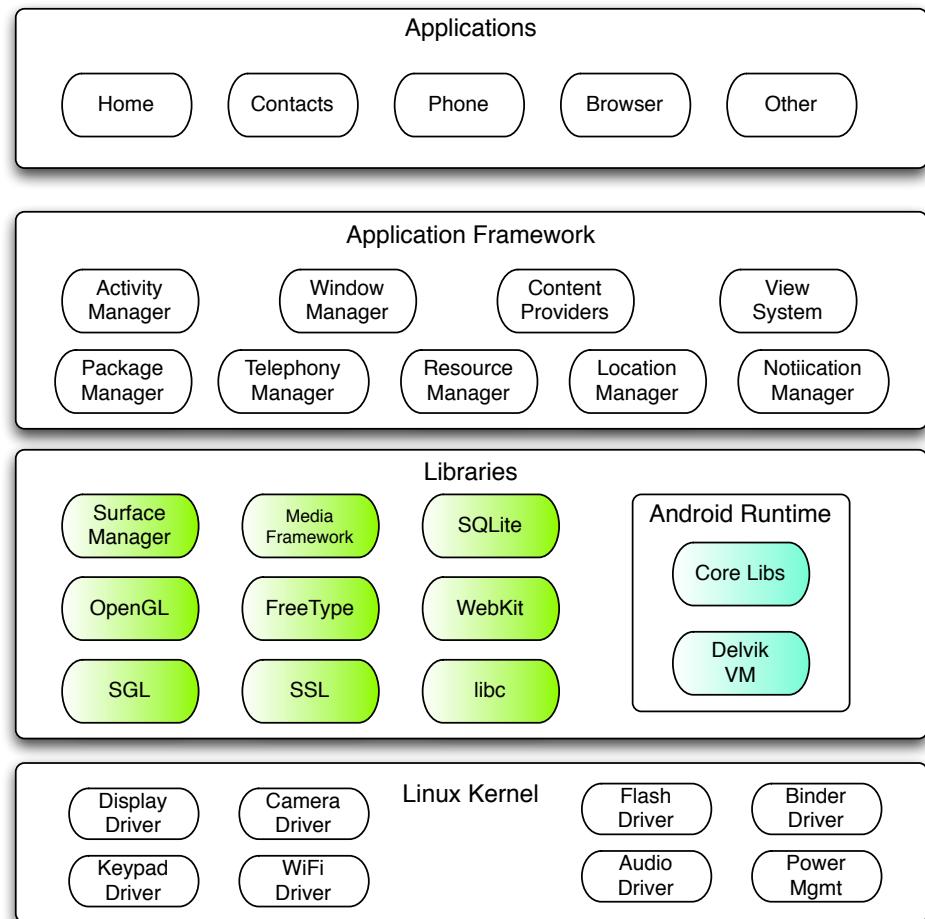
Surface Manager for composing
window manager with off-screen
buffering

2D and 3D graphics hardware
support or software simulation

Media codecs offer support for
major audio/video codecs

SQLite database

WebKit library for fast HTML
rendering



Dalvik

Dalvik VM is Google's implementation of Java

Optimized for mobile devices

Key Dalvik differences:

Register-based versus stack-based VM

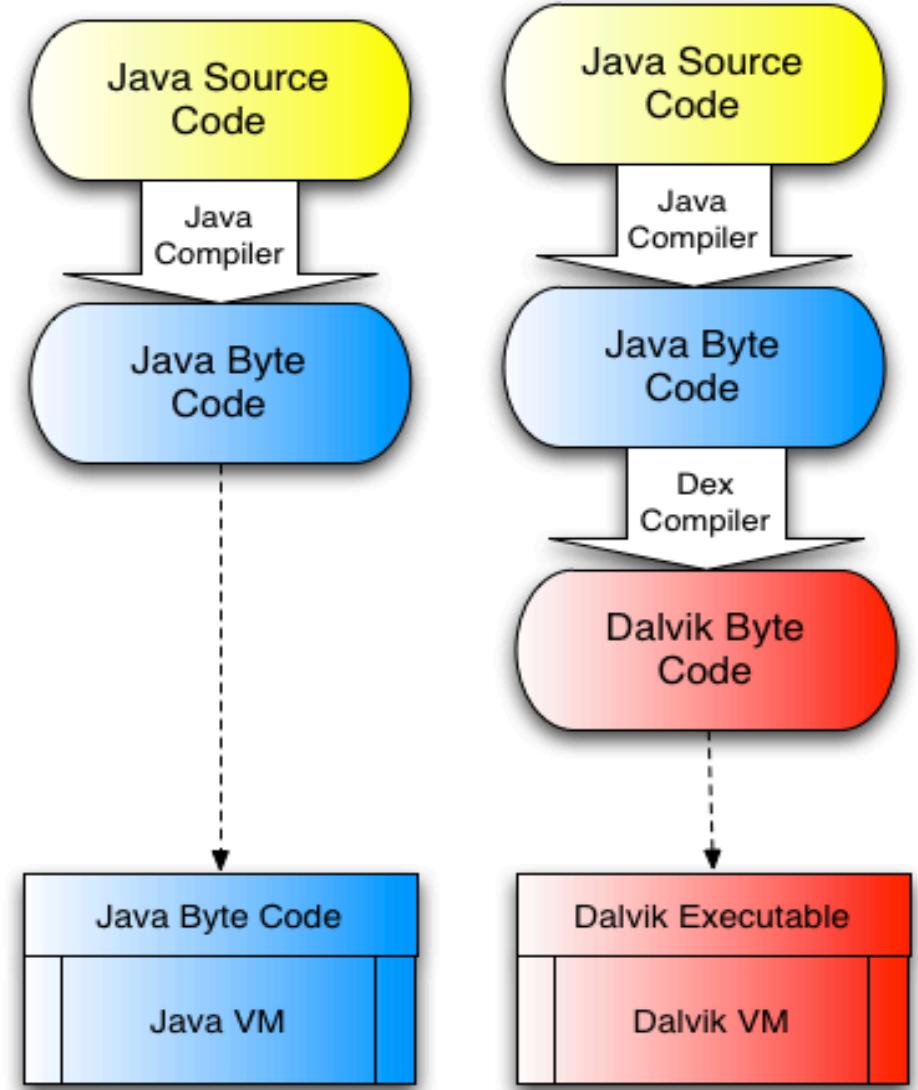
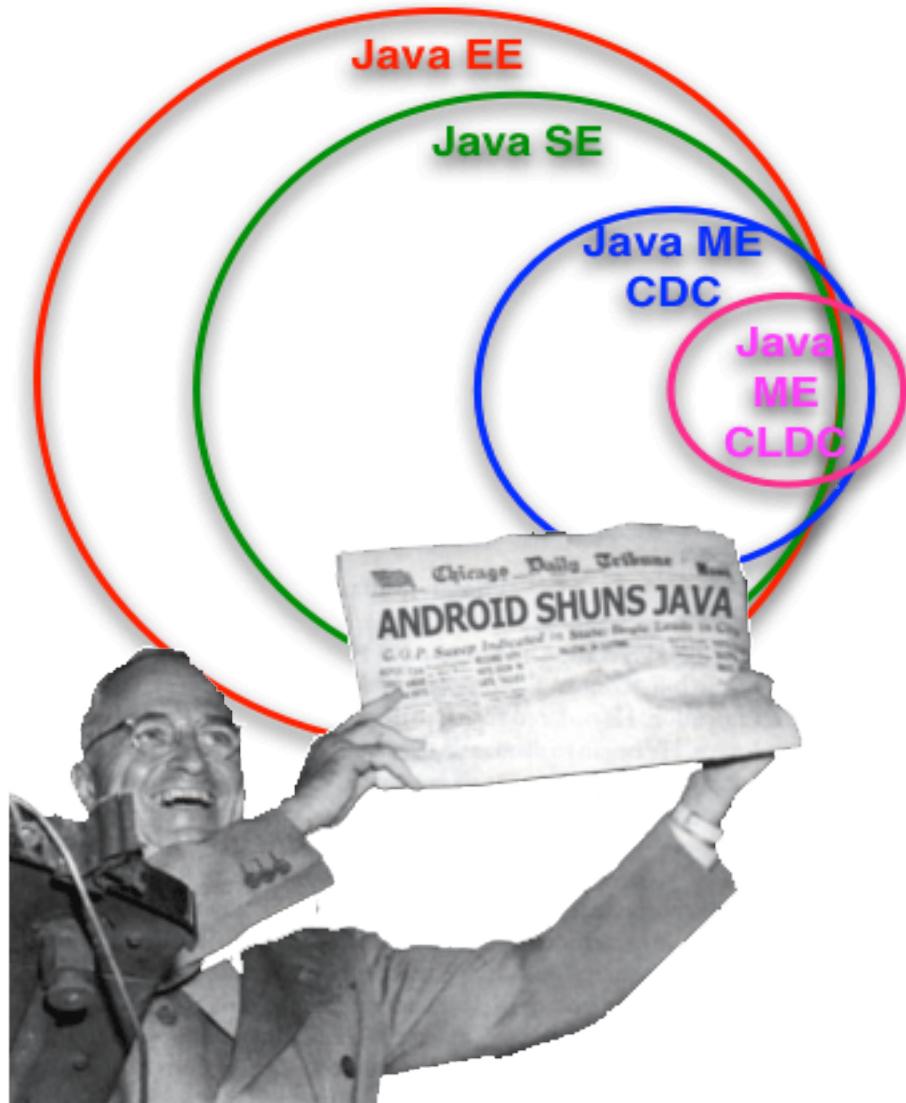
Dalvik runs .dex files

More efficient and compact implementation

Different set of Java libraries than SDK



Android and Java



Application Framework

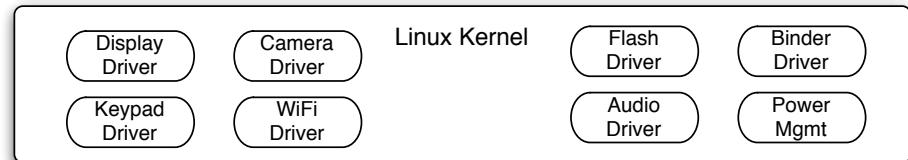
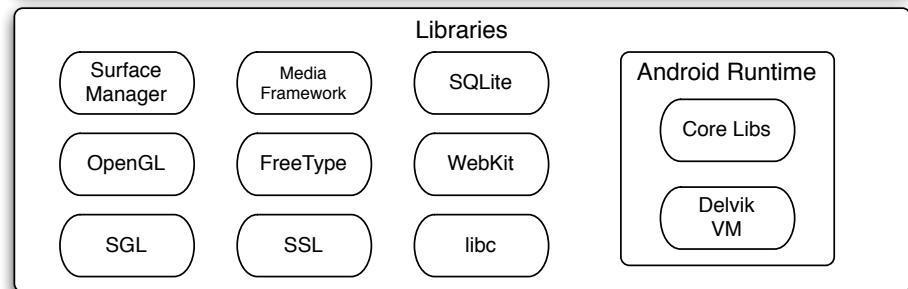
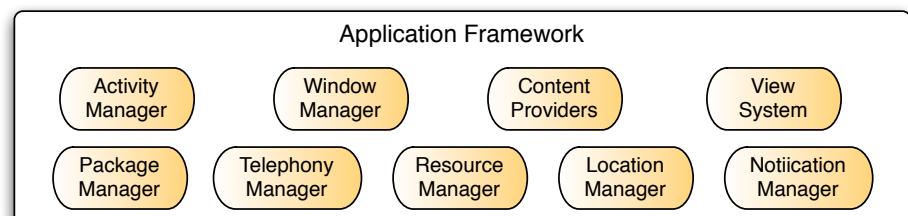
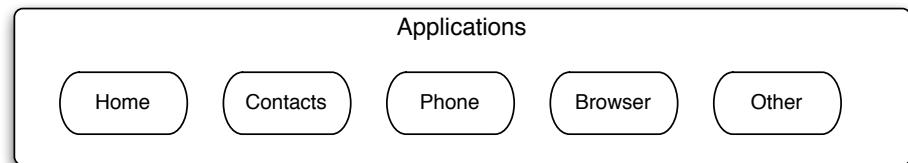
Activation manager controls the life cycle of the app

Content providers encapsulate data that is shared (e.g. contacts)

Resource manager manages everything that is not the code

Location manager figures out the location of the phone (GPS, GSM, WiFi)

Notification manager for events such as arriving messages, appointments, etc



Applications



Android Application APK

Dalvik
Exe

Resources

Native
Libs



OPERATING SYSTEM FEATURES

File System

The file system has three main mount points. One for system, one for the apps, and one for whatever.

Each app has its own sandbox easily accessible to it. No one else can access its data. The sandbox is in /data/data/com.marakana/

SDCard is expected to always be there. It's a good place for large files, such as movies and music.

Everyone can access it.

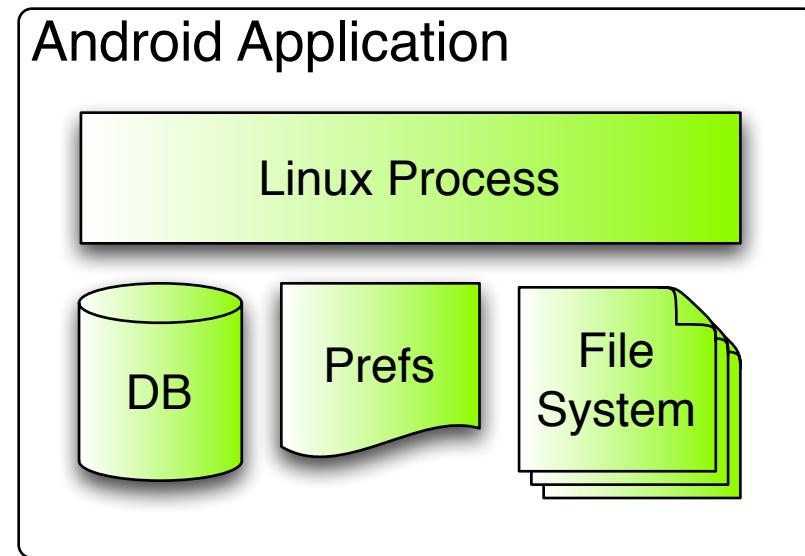
- ▼  data
 - ▶  anr
 - ▶  app
 - ▶  app-private
 - ▶  backup
 - ▶  dalvik-cache
 - ▶  data
 - ▶  dontpanic
 - ▶  local
 - ▶  lost+found
 - ▶  misc
 - ▶  property
 - ▶  system
- ▶  sdcard
- ▼  system
 - ▶  app
 - ▶  bin
 - ▶  build.prop
 - ▶  etc
 - ▶  fonts
 - ▶  framework
 - ▶  lib
 - ▶  lost+found
 - ▶  tts
 - ▶  usr
 - ▶  xbin

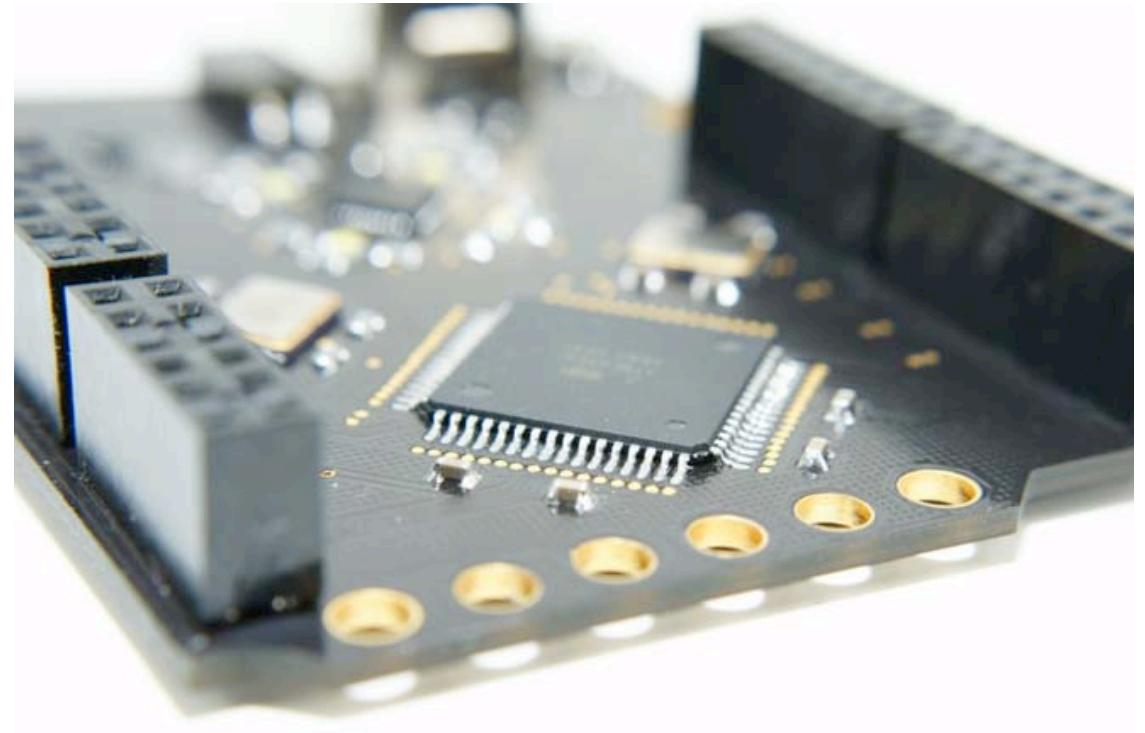
Security

Each Android application runs inside its own Linux process.

Additionally, each application has its own sandbox file system with its own set of preferences and its own database.

Other applications cannot access any of its data, unless it is explicitly shared.



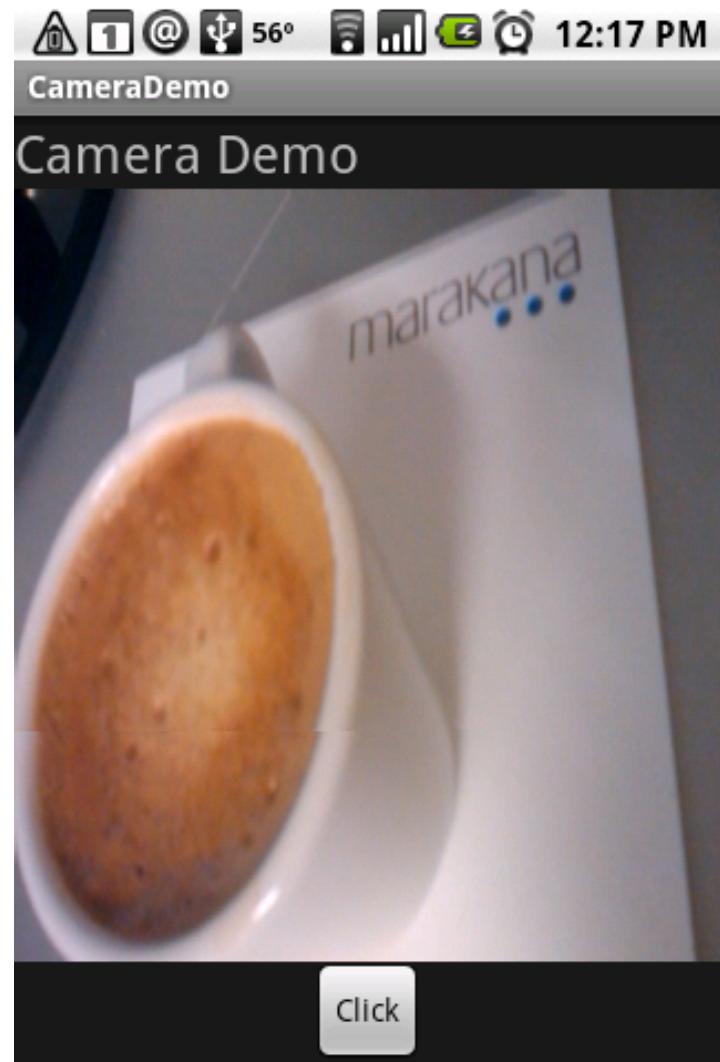


WORKING WITH HARDWARE

Camera

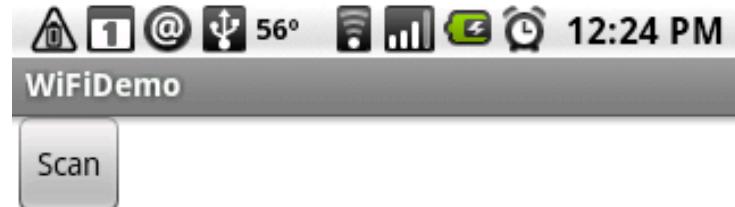
Android SDK supports access to built-in Camera and its preview.

You can access real-time frames, or get a callback when shutter is open. The photo data is passed back in either raw or jpeg format.



WiFi

WiFi API allows for managing your connection, scanning for active WiFi points and find out details about each.



WiFiDemo

WIFI Status: SSID: Marakana Network, BSSID: 00:14:bf:43:dc:0d, MAC: 00:23:76:0c:07:7d, Supplicant state: COMPLETED, RSSI: -200, Link speed: 54, Net ID: 6

- ID: 0 SSID: "dorado" BSSID: null PRIO: 8
KeyMgmt: NONE Protocols: WPA RSN
AuthAlgorithms:
PairwiseCiphers: TKIP CCMP
GroupCiphers: WEP40 WEP104 TKIP CCMP

ID: 1 SSID: "cpmcairnet" BSSID: null PRIO: 20
KeyMgmt: NONE Protocols: WPA RSN
AuthAlgorithms:
PairwiseCiphers: TKIP CCMP
GroupCiphers: WEP40 WEP104 TKIP CCMP

- ID: 2 SSID: "MeetingAccess" BSSID: null PRIO: 10
KeyMgmt: NONE Protocols: WPA RSN
AuthAlgorithms:

Telephony

With Telephony API, you can:

Make phone calls

Monitor phone state and activity

Access phone properties and status

Monitor data connectivity

Control the phone



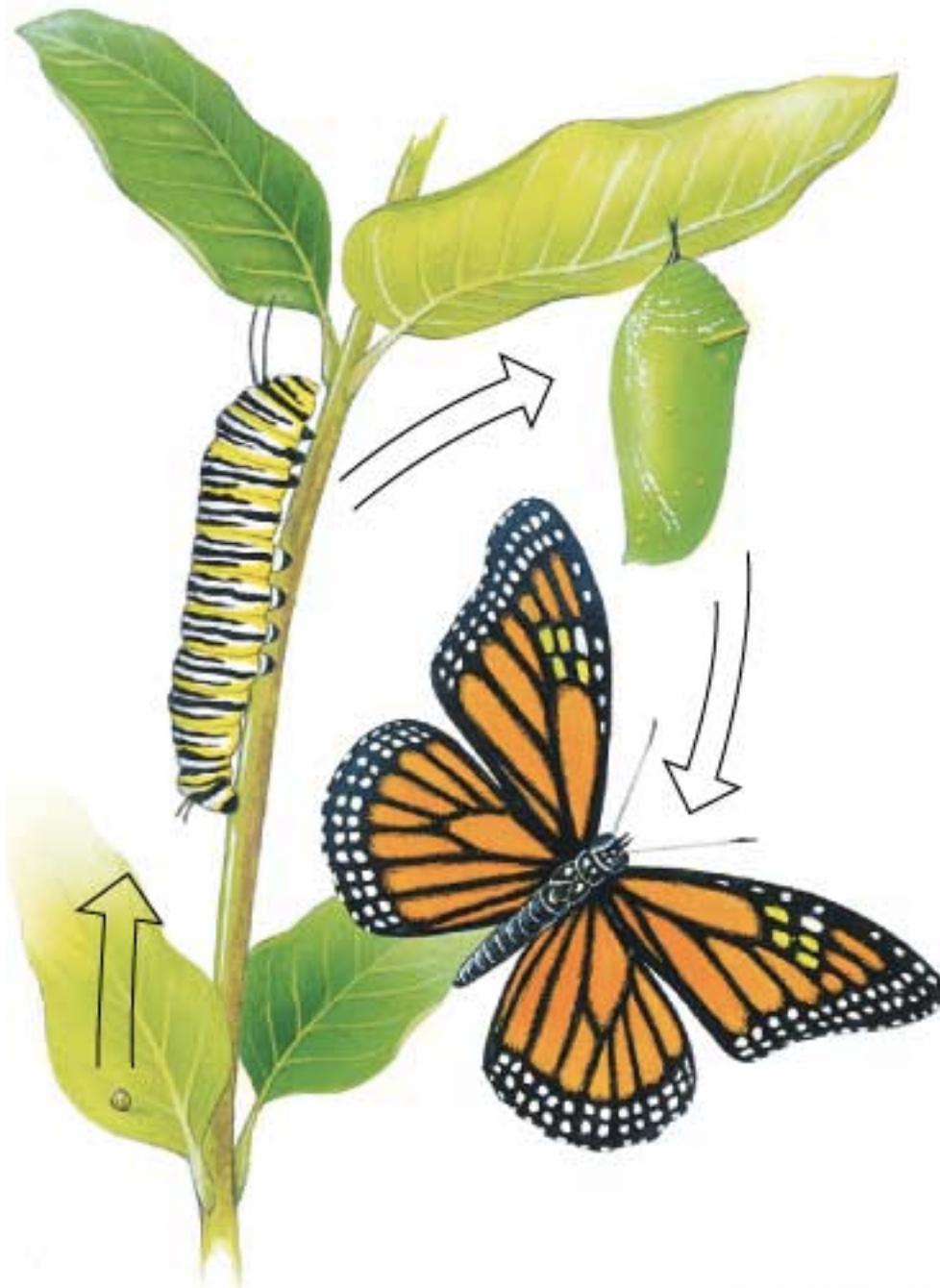
ConnectivityDemo

Connectivity Demo

NetworkInfo: type: WIFI[], state: CONNECTED/
CONNECTED, reason: (unspecified), extra: (none),
roaming: false, failover: false, isAvailable: true

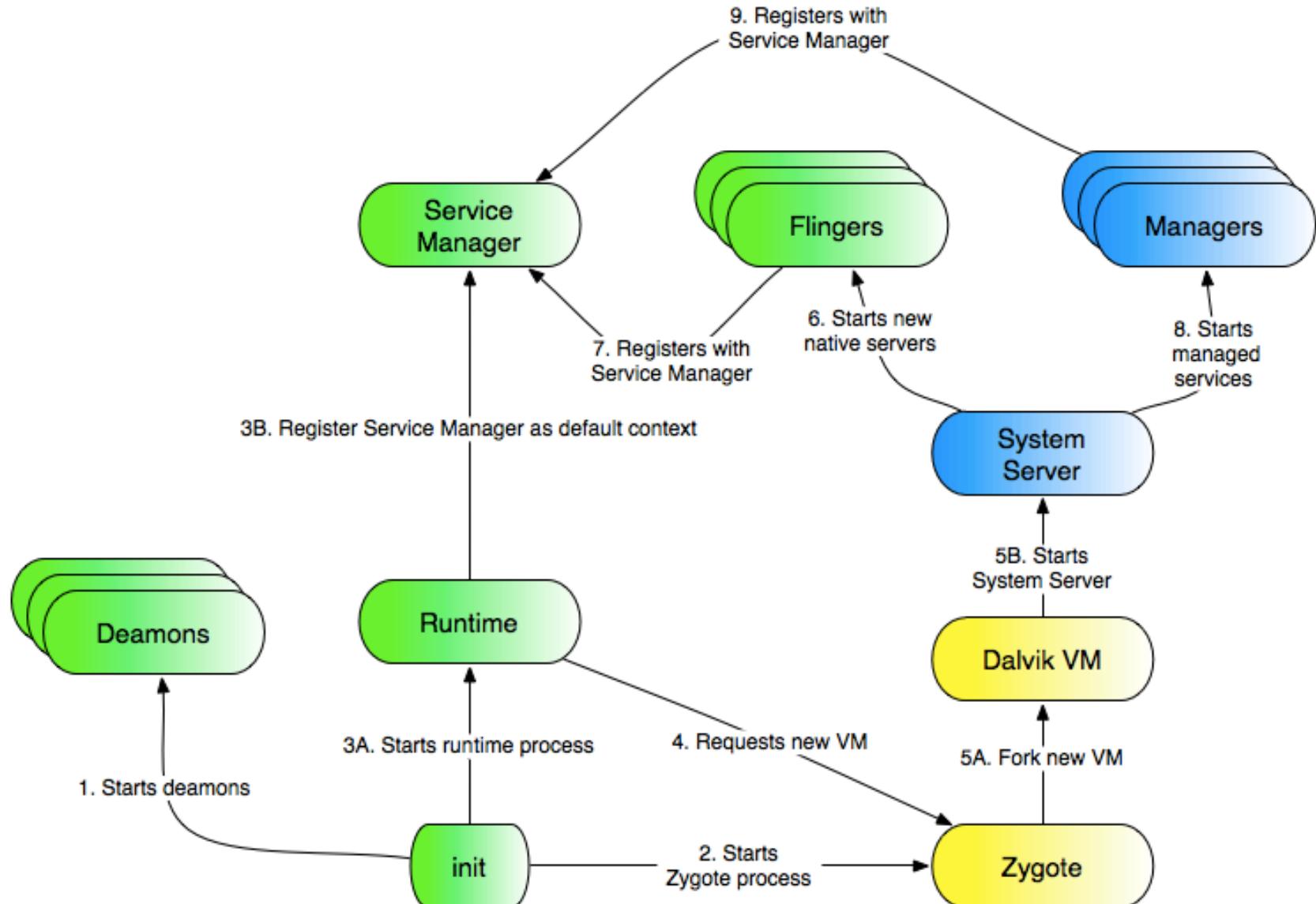
NetworkInfo: type: MOBILE[UMTS], state:
DISCONNECTED/DISCONNECTED, reason:
dataDisabled, extra: epc.tmobile.com, roaming:
false, failover: false, isAvailable: true

It is a simple yet powerful API

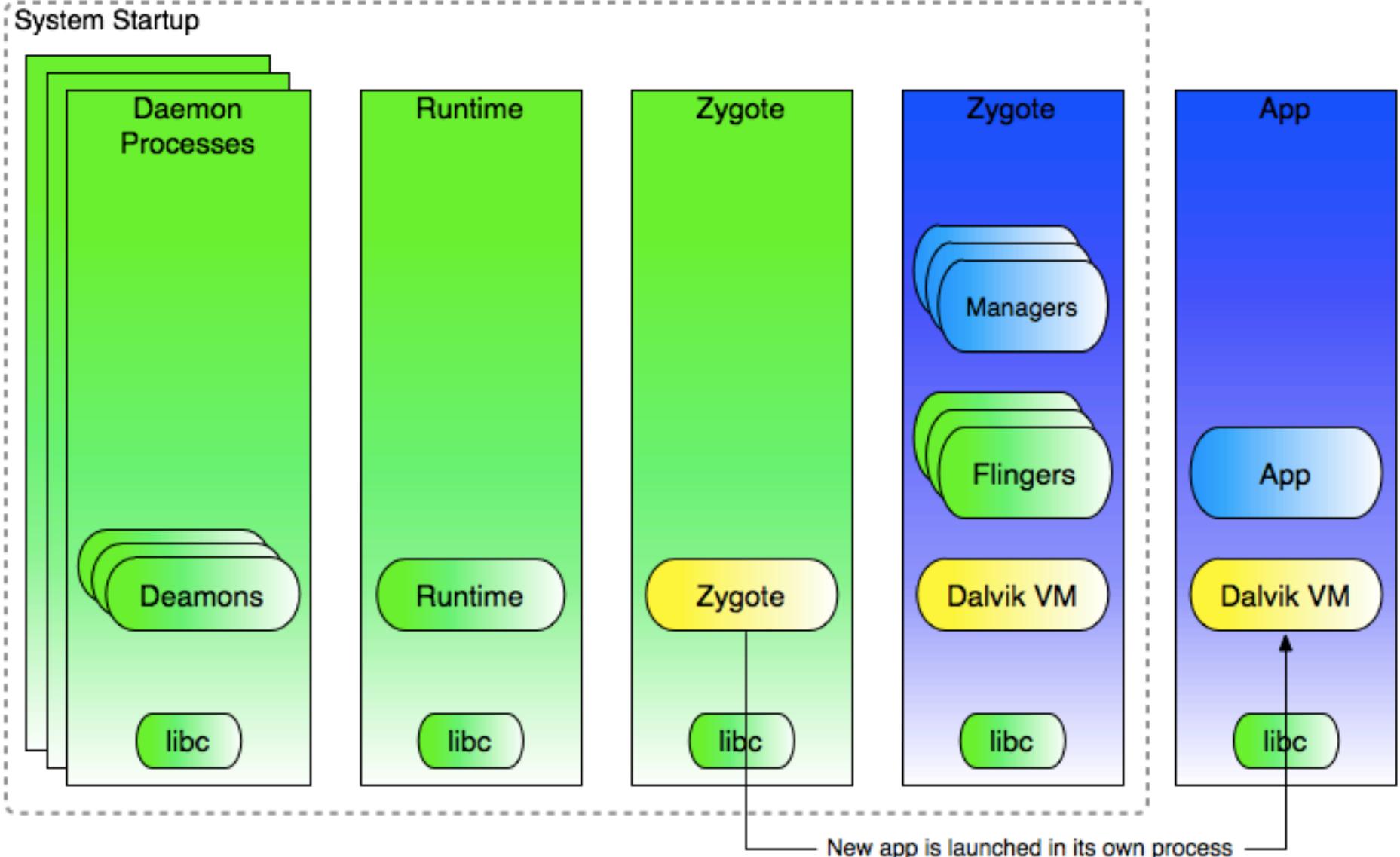


ANDROID STARTUP & RUNTIME

Startup Walkthrough



Runtime Overview

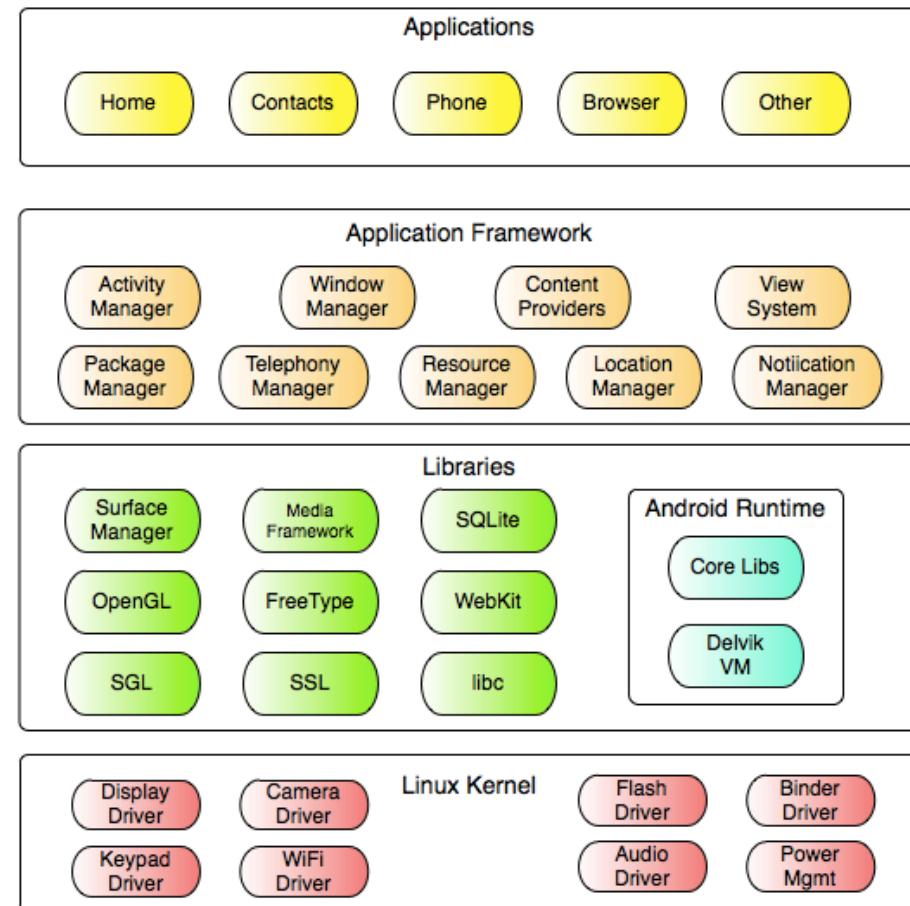


Layer Interactions

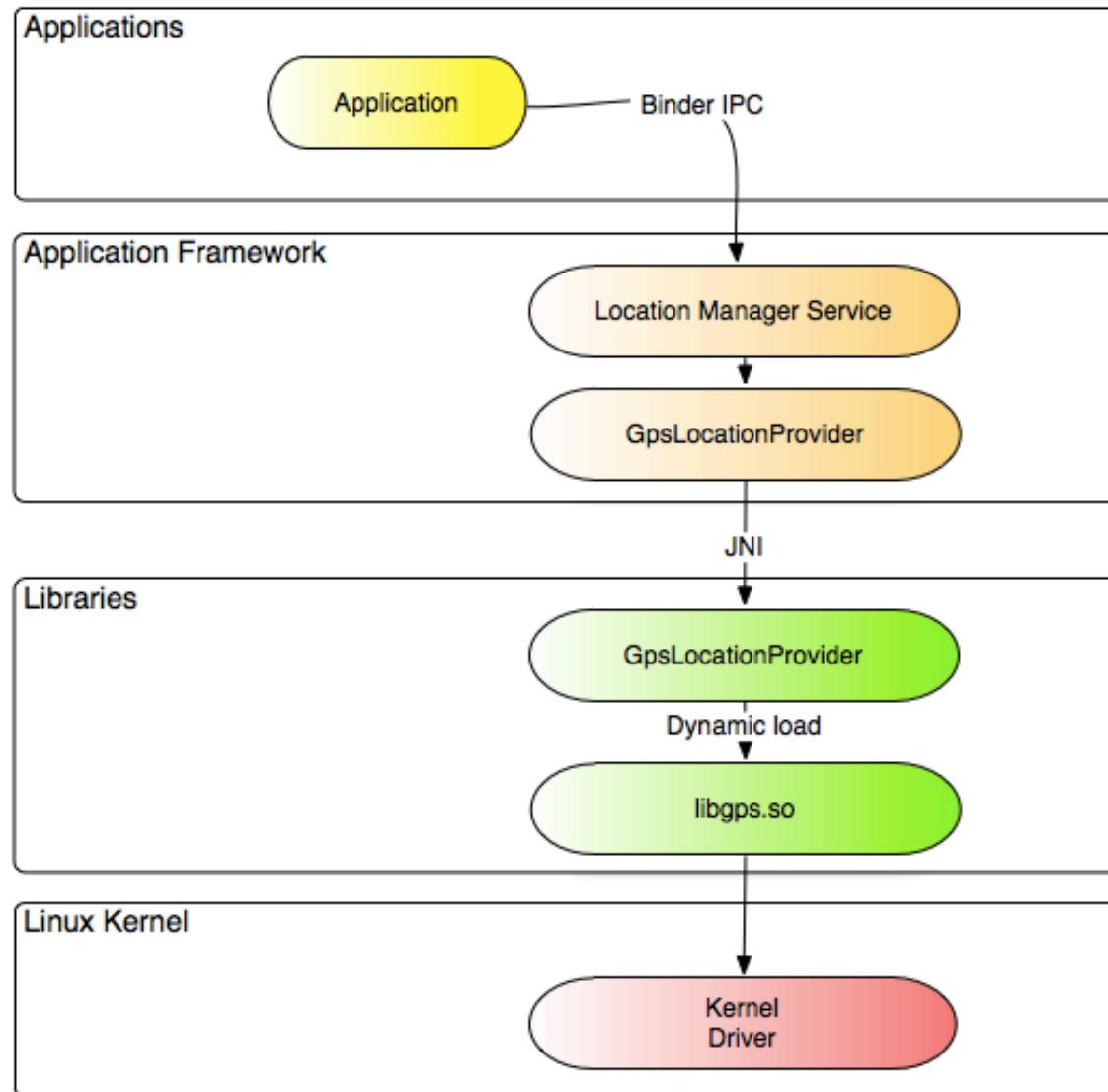
There are three main scenarios for your app to talk to native library:

- Directly
- Via native service
- Via native daemon

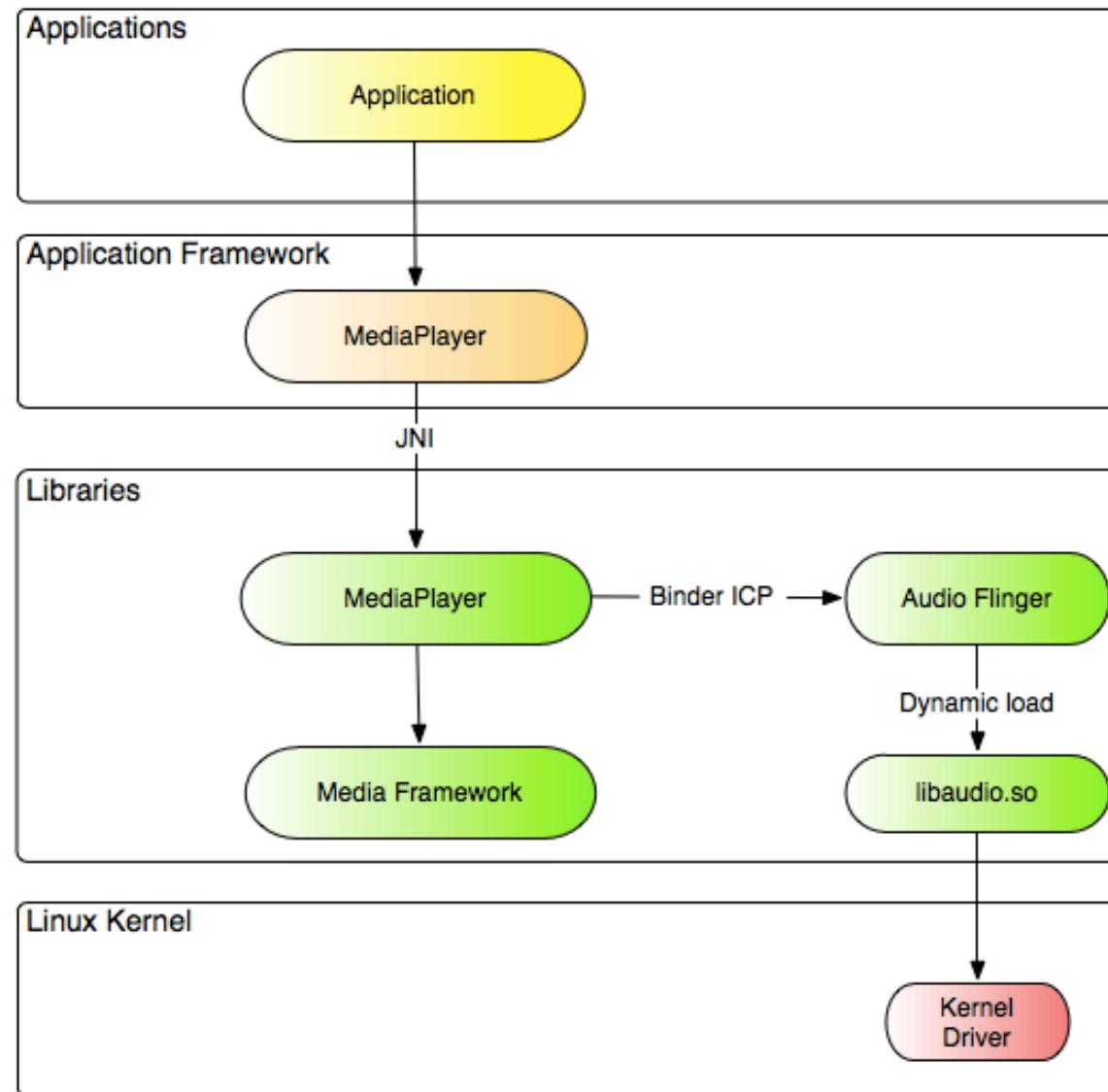
It will depend on the type of app and type of native library which method works best.



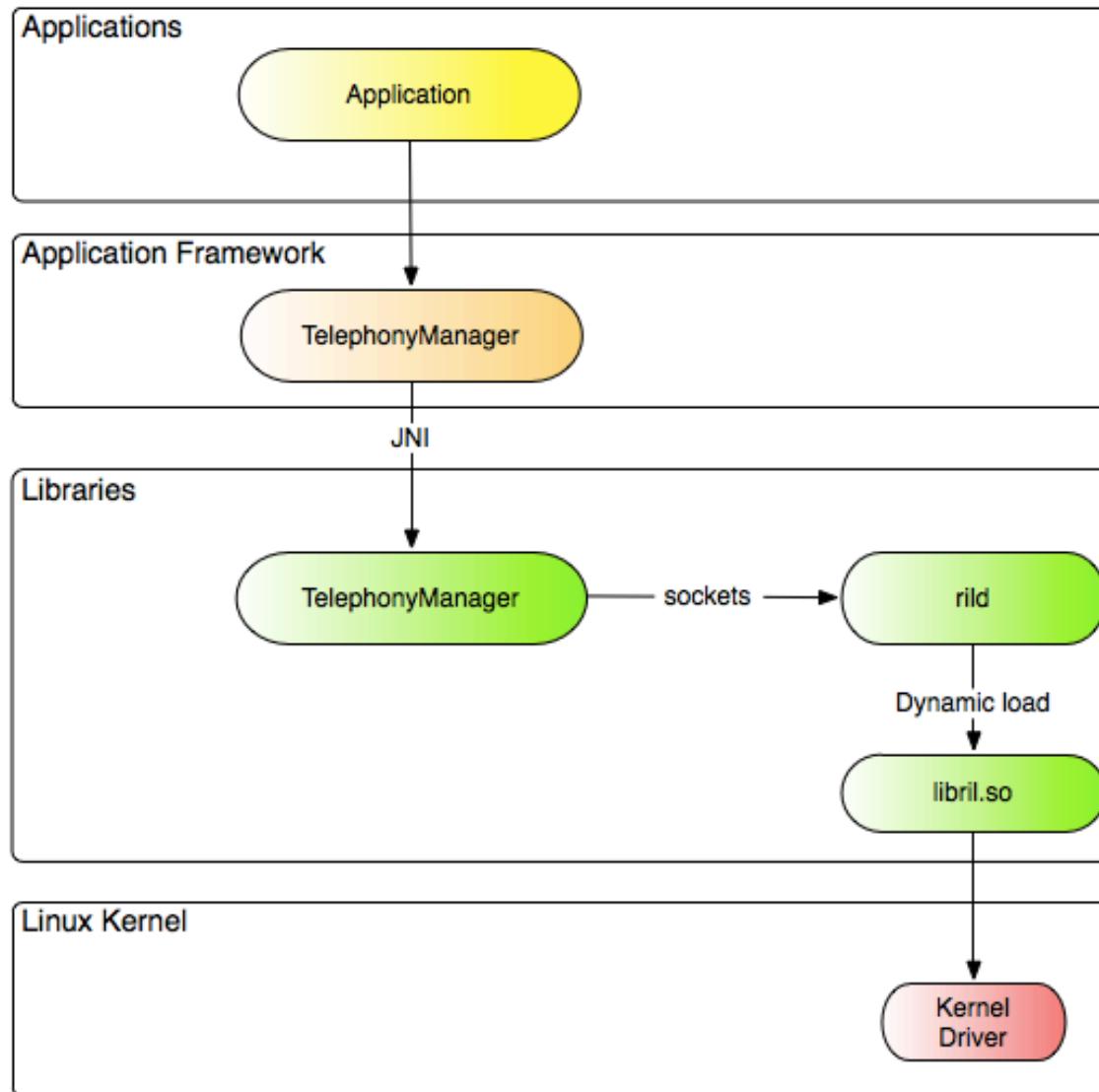
App – Runtime Service - Lib



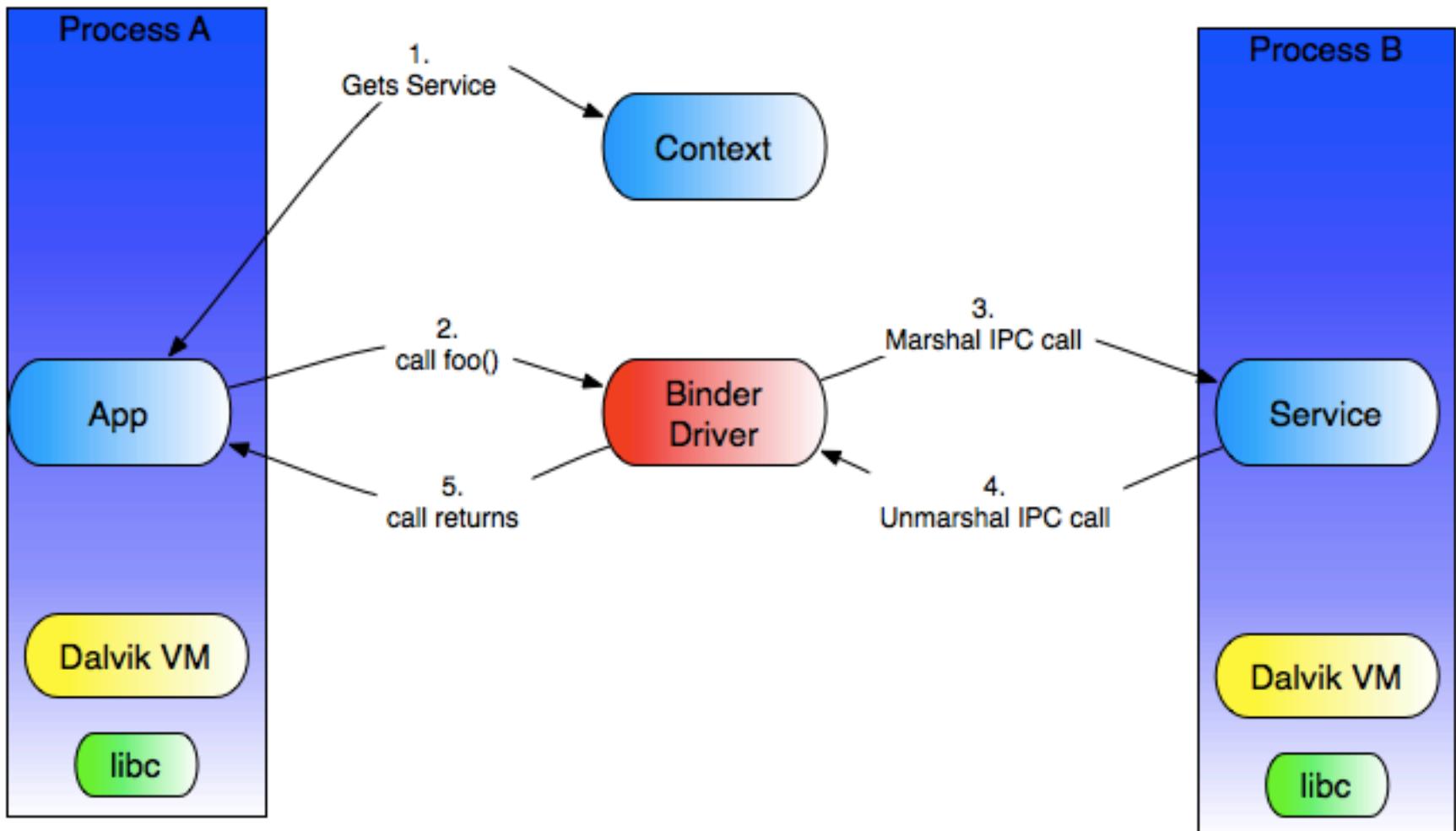
App – Runtime-Native Service-Lib



App–Runtime–Native Daemon–Lib



Binder IPC

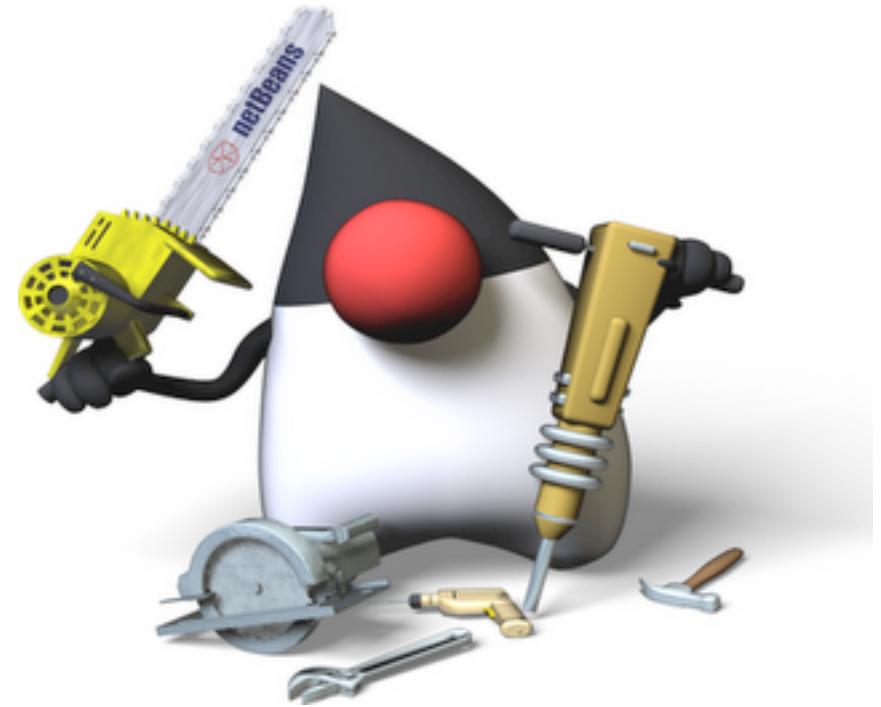


High-performance IPC: shared memory, per-process thread pool, synchronous

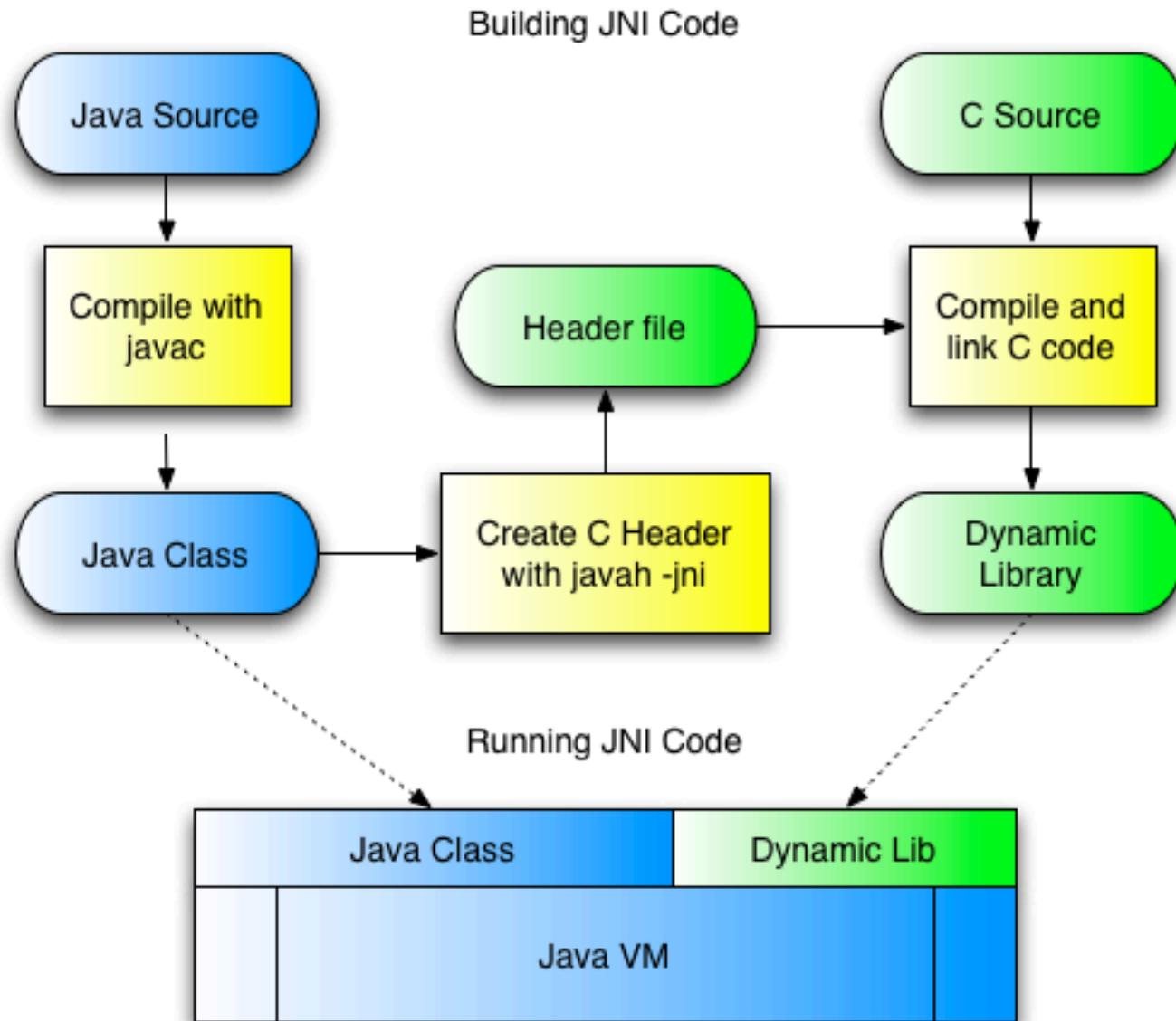
Java Native Interface

JNI defines naming and coding convention so that Java VM can find and call native code.

JNI is built into JVM to provide access to OS I/O and others.



Building and Running JNI Code





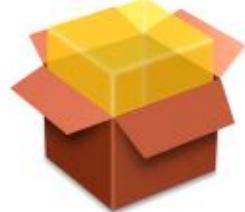
marakana

NATIVE DEVELOPMENT KIT

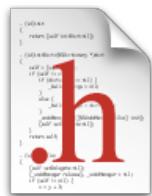
What's in NDK?



Tools to build and compile your native code for the device architecture (such as ARM)



A way to package your library into the APK file so you can distribute your application easily



A set of native system headers that will be supported for the future releases of Android platform (libc, libm, libz, liblog, JNI headers, some C++ headers, and OpenGL)



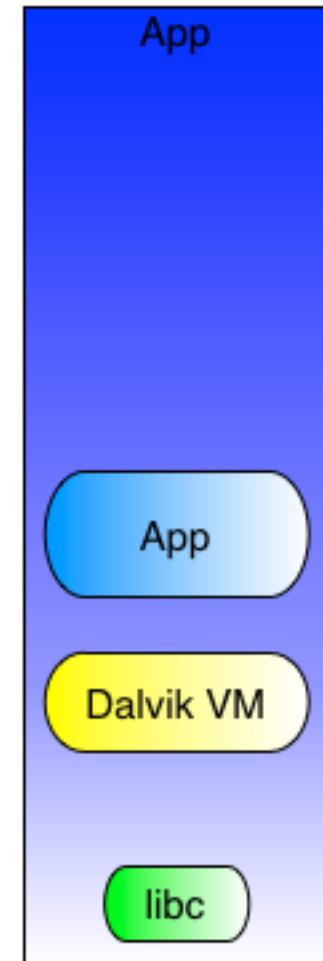
(some) documentation, sample code and examples

Why NDK?

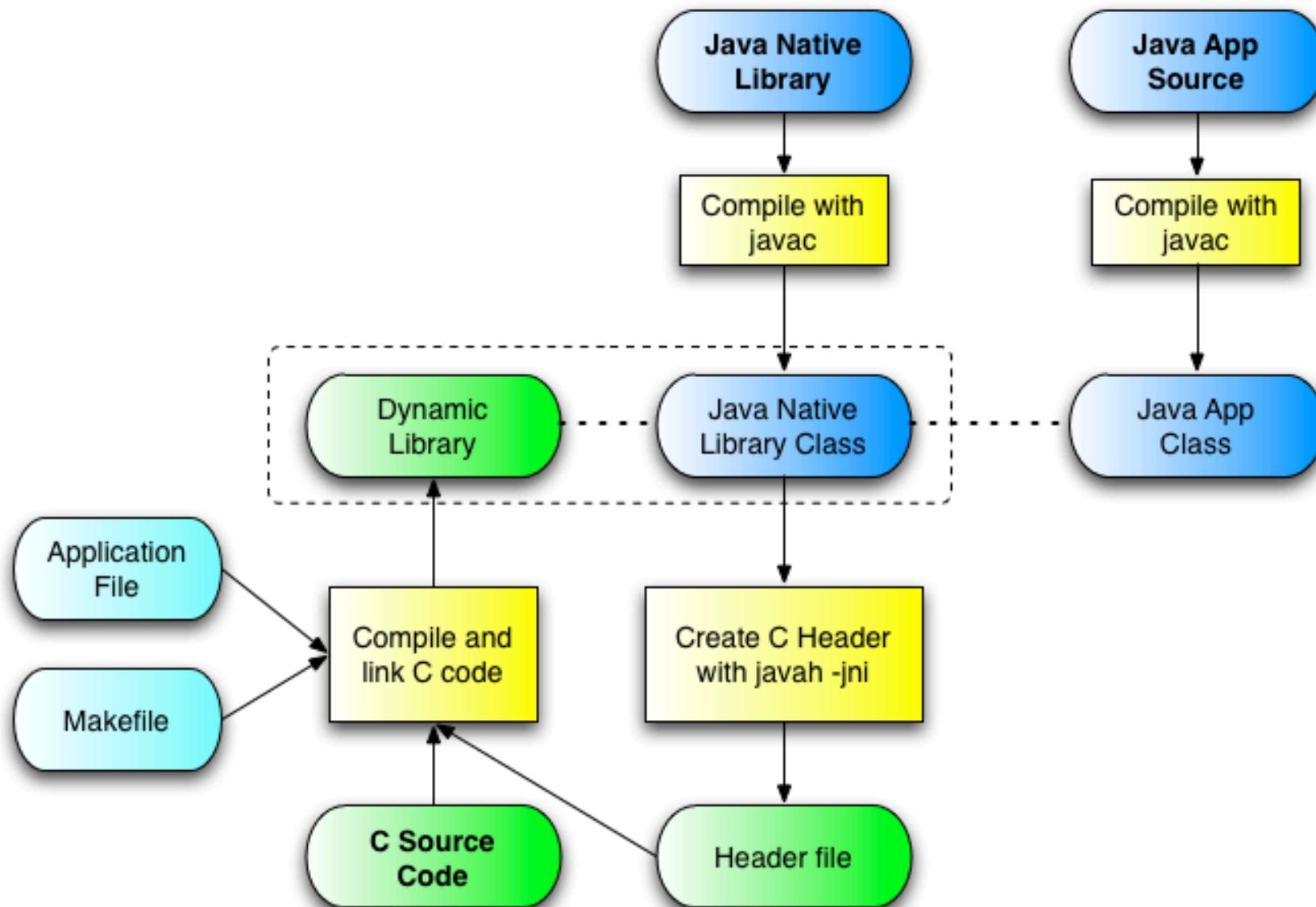
NDK allows you to develop parts of your Android application in C/C++.

You cannot develop native-only apps in NDK – your app is still subject to security sandboxing.

Main motivation for native code is performance.



Using NDK



DEBUGGING ANDROID APPS

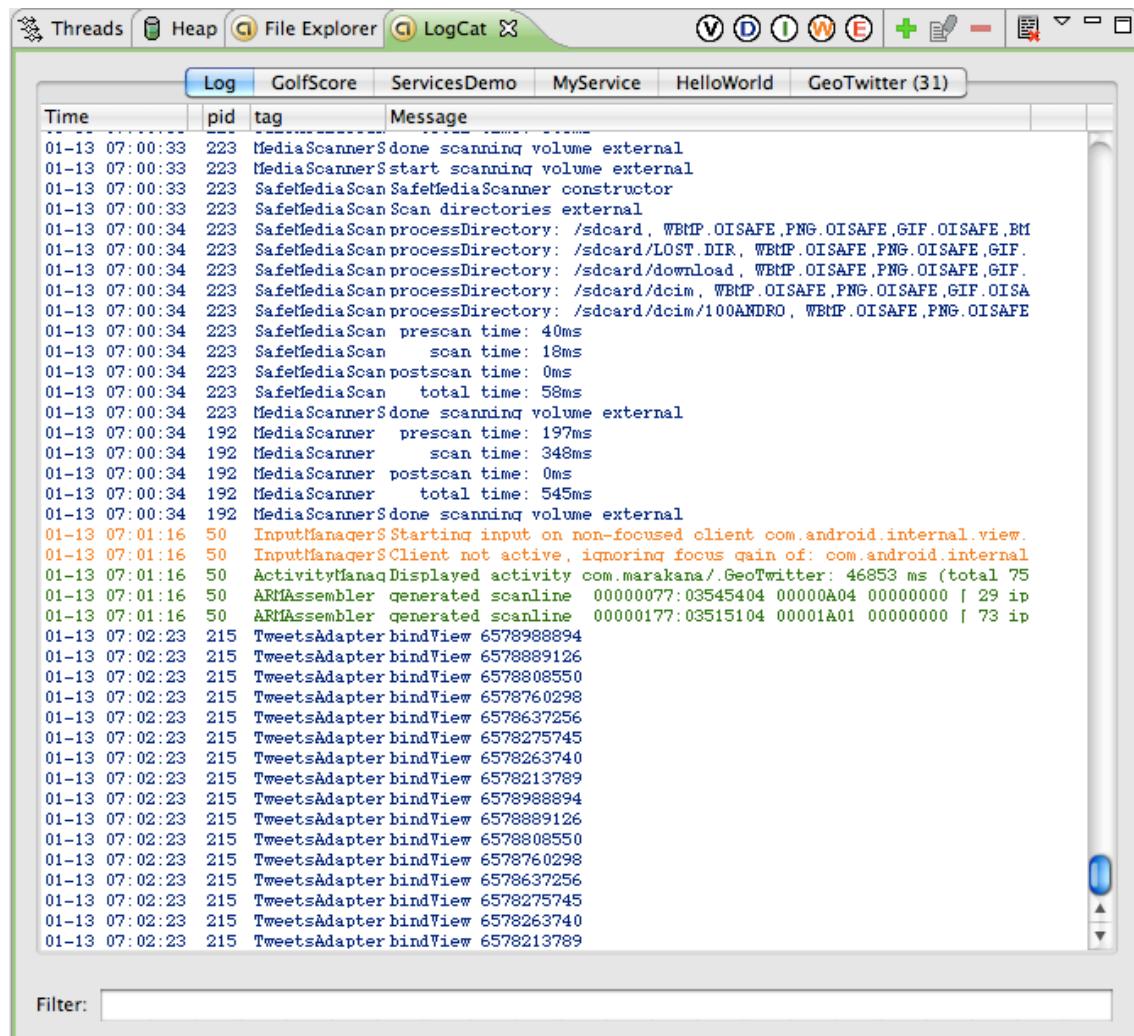


LogCat

The universal, most versatile way to track what is going on in your app.

Can be viewed via command line or Eclipse.

Logs can be generated both from SDK Java code, or low-level C code via Bionic libc extension.

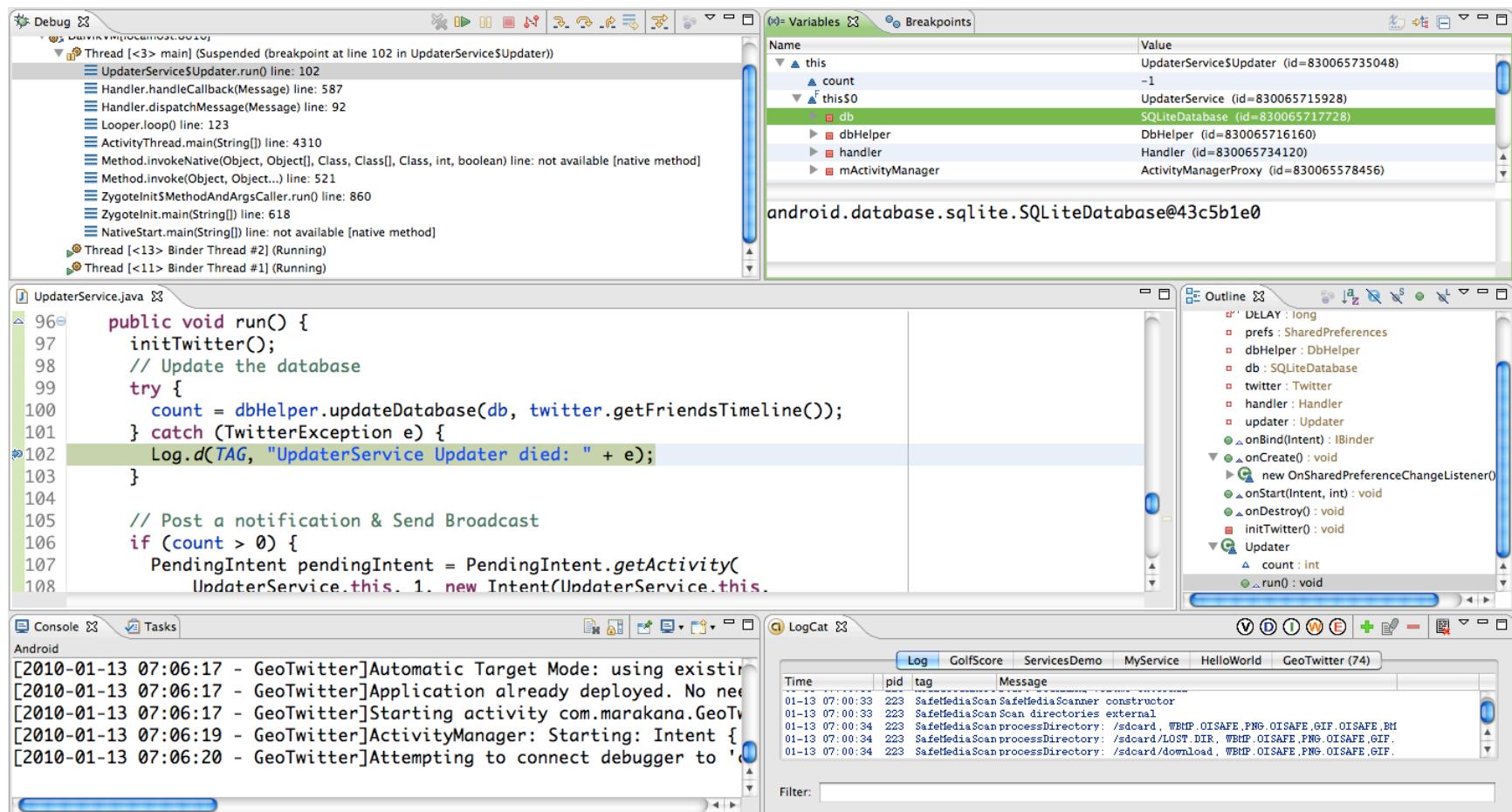


The screenshot shows the Eclipse IDE's LogCat view. The title bar includes tabs for Threads, Heap, File Explorer, and LogCat. Below the tabs is a toolbar with icons for selection, copy, paste, and other operations. The main area is a table with columns: Time, pid, tag, and Message. The table displays numerous log entries. Some entries are timestamped at 01-13 07:00:33 and 01-13 07:00:34, while others are at 01-13 07:01:16. Pids listed include 223, 192, and 50. Tags like 'MediaScanner', 'SafeMediaScan', and 'TweetsAdapter' are visible. The 'Message' column contains detailed log messages such as 'done scanning volume external', 'start scanning volume external', and 'bindView' calls with specific IDs like 6578988894 and 6578889126.

Time	pid	tag	Message
01-13 07:00:33	223	MediaScanner	done scanning volume external
01-13 07:00:33	223	MediaScanner	start scanning volume external
01-13 07:00:33	223	SafeMediaScan	SafeMediaScanner constructor
01-13 07:00:33	223	SafeMediaScan	Scan directories external
01-13 07:00:34	223	SafeMediaScan	processDirectory: /sdcard/WBMP.OISAFE.PNG.OISAFE.GIF.OISAFE.BM
01-13 07:00:34	223	SafeMediaScan	processDirectory: /sdcard/LOST.DIR, WBMP.OISAFE.PNG.OISAFE.GIF.
01-13 07:00:34	223	SafeMediaScan	processDirectory: /sdcard/download, WBMP.OISAFE.PNG.OISAFE.GIF.
01-13 07:00:34	223	SafeMediaScan	processDirectory: /sdcard/dcim, WBMP.OISAFE.PNG.OISAFE.GIF.OISAF
01-13 07:00:34	223	SafeMediaScan	processDirectory: /sdcard/dcim/100ANDRO, WBMP.OISAFE.PNG.OISAFE
01-13 07:00:34	223	SafeMediaScan	processDirectory: /sdcard/dcim/dcim, WBMP.OISAFE.PNG.OISAFE.GIF.OISAF
01-13 07:00:34	223	SafeMediaScan	prescan time: 40ms
01-13 07:00:34	223	SafeMediaScan	scan time: 18ms
01-13 07:00:34	223	SafeMediaScan	postscan time: 0ms
01-13 07:00:34	223	SafeMediaScan	total time: 58ms
01-13 07:00:34	223	MediaScanner	done scanning volume external
01-13 07:00:34	192	MediaScanner	prescan time: 197ms
01-13 07:00:34	192	MediaScanner	scan time: 348ms
01-13 07:00:34	192	MediaScanner	postscan time: 0ms
01-13 07:00:34	192	MediaScanner	total time: 545ms
01-13 07:00:34	192	MediaScanner	done scanning volume external
01-13 07:01:16	50	InputMethodManager	Starting input on non-focused client com.android.internal.view
01-13 07:01:16	50	InputMethodManager	Client not active, ignoring focus gain of: com.android.internal
01-13 07:01:16	50	ActivityManager	Displayed activity com.marakana/.GeoTwitter: 46853 ms (total 75
01-13 07:01:16	50	ARMAssembler	generated scanline 00000077:03545404 00000A04 00000000 29 ip
01-13 07:01:16	50	ARMAssembler	generated scanline 00000177:03515104 00001A01 00000000 73 ip
01-13 07:02:23	215	TweetsAdapter	bindView 6578988894
01-13 07:02:23	215	TweetsAdapter	bindView 6578889126
01-13 07:02:23	215	TweetsAdapter	bindView 6578808550
01-13 07:02:23	215	TweetsAdapter	bindView 6578760298
01-13 07:02:23	215	TweetsAdapter	bindView 6578637256
01-13 07:02:23	215	TweetsAdapter	bindView 6578275745
01-13 07:02:23	215	TweetsAdapter	bindView 6578263740
01-13 07:02:23	215	TweetsAdapter	bindView 6578213789
01-13 07:02:23	215	TweetsAdapter	bindView 6578988894
01-13 07:02:23	215	TweetsAdapter	bindView 6578889126
01-13 07:02:23	215	TweetsAdapter	bindView 6578808550
01-13 07:02:23	215	TweetsAdapter	bindView 6578760298
01-13 07:02:23	215	TweetsAdapter	bindView 6578637256
01-13 07:02:23	215	TweetsAdapter	bindView 6578275745
01-13 07:02:23	215	TweetsAdapter	bindView 6578263740
01-13 07:02:23	215	TweetsAdapter	bindView 6578213789

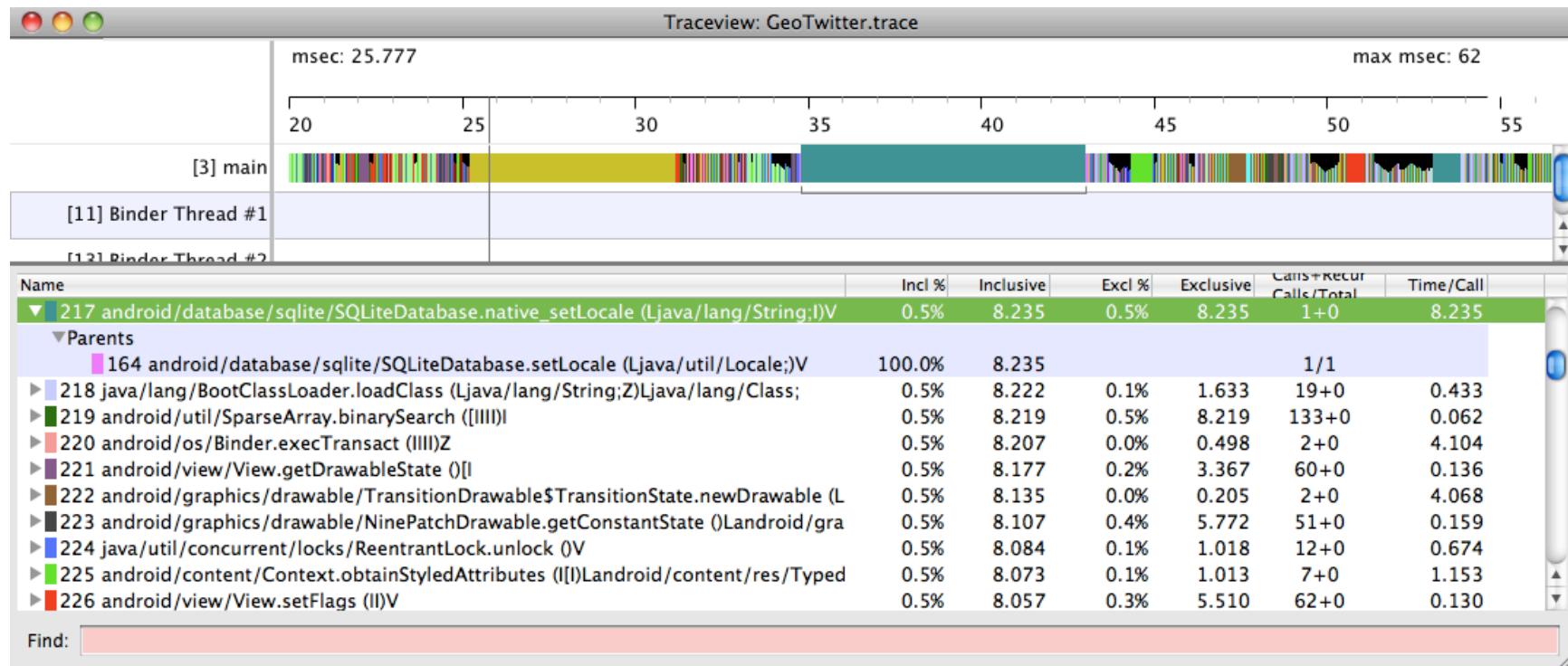
Filter:

Debugger



Your standard debugger is included in SDK, with all the usual bells & whistles.

TraceView

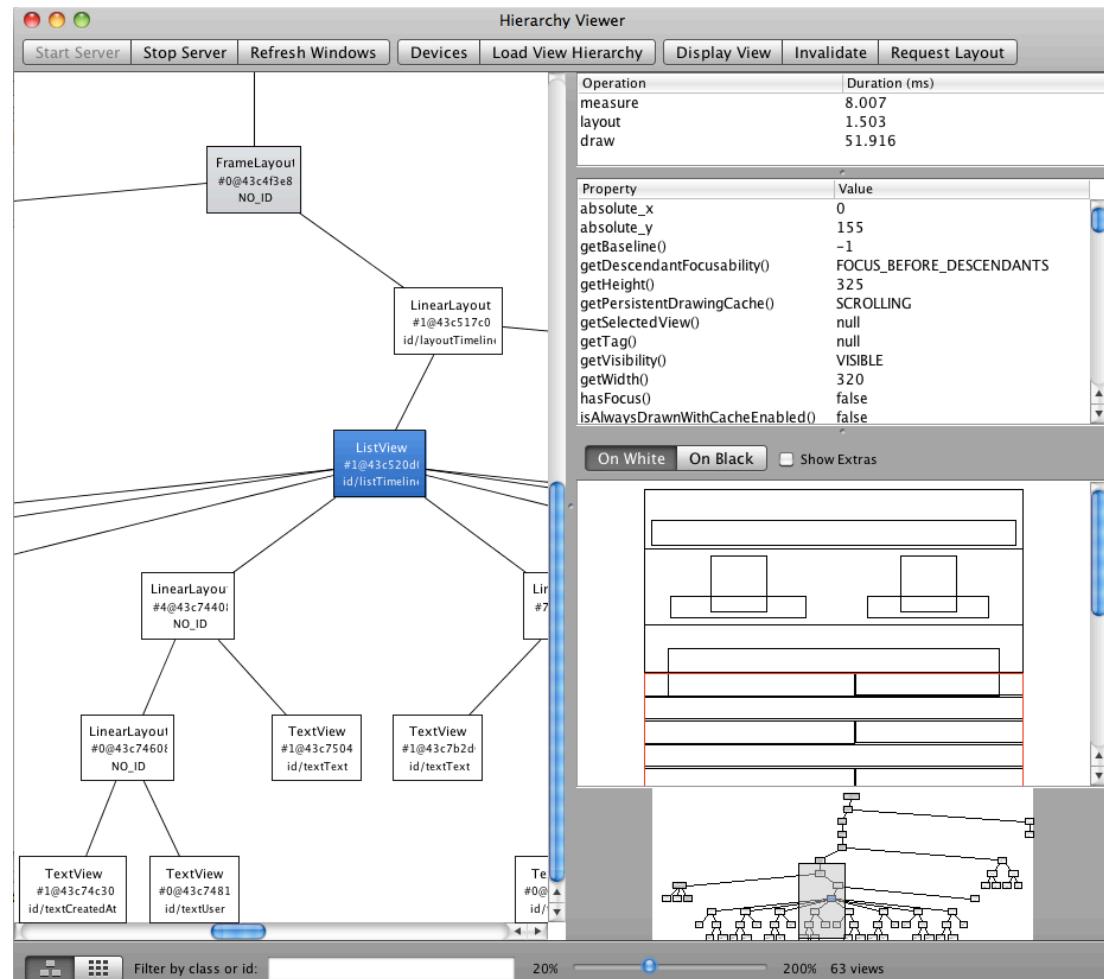


TraceView helps you profile your application and find bottlenecks. It shows execution of various calls through the entire stack. You can zoom into specific calls.

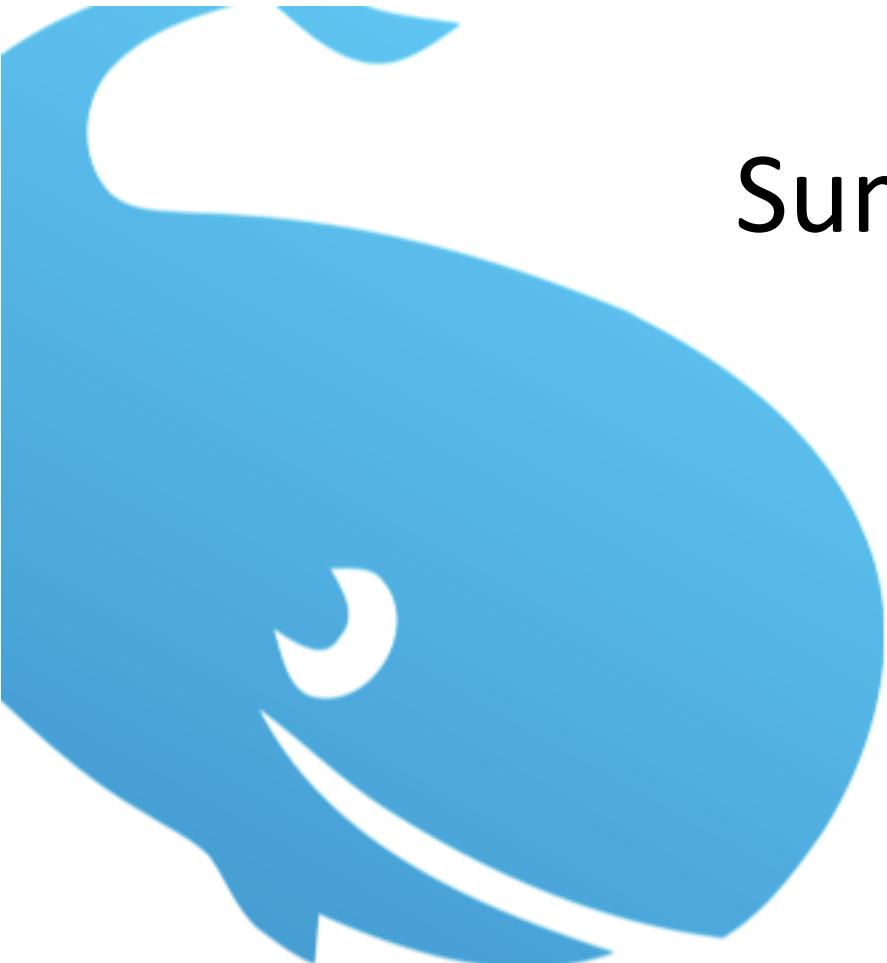
Hierarchy Viewer

Hierarchy Viewer helps you analyze your User Interface.

Base UI tends to be the most “expensive” part of your application, this tool is very useful.



Summary



For most applications, you will just need Android SDK to develop apps.

Sometimes you may need NDK to make parts of your app run faster.

Ultimately, you can do whatever you want by compiling Android platform from source.



Marko Gargenta
Marakana.com

ANDROID

Licensed under Creative Commons License (cc-by-nc-nd). **Please Share!**

