marakana

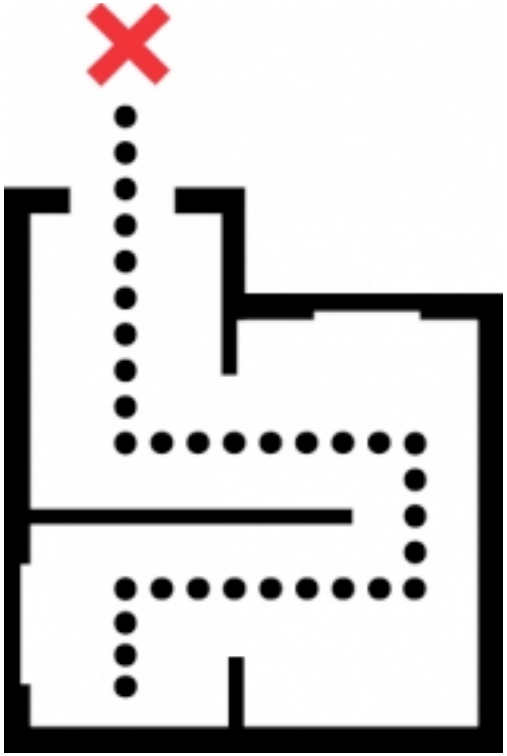
Android Internals

Marko Gargenta
Marakana

marakana

Agenda

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

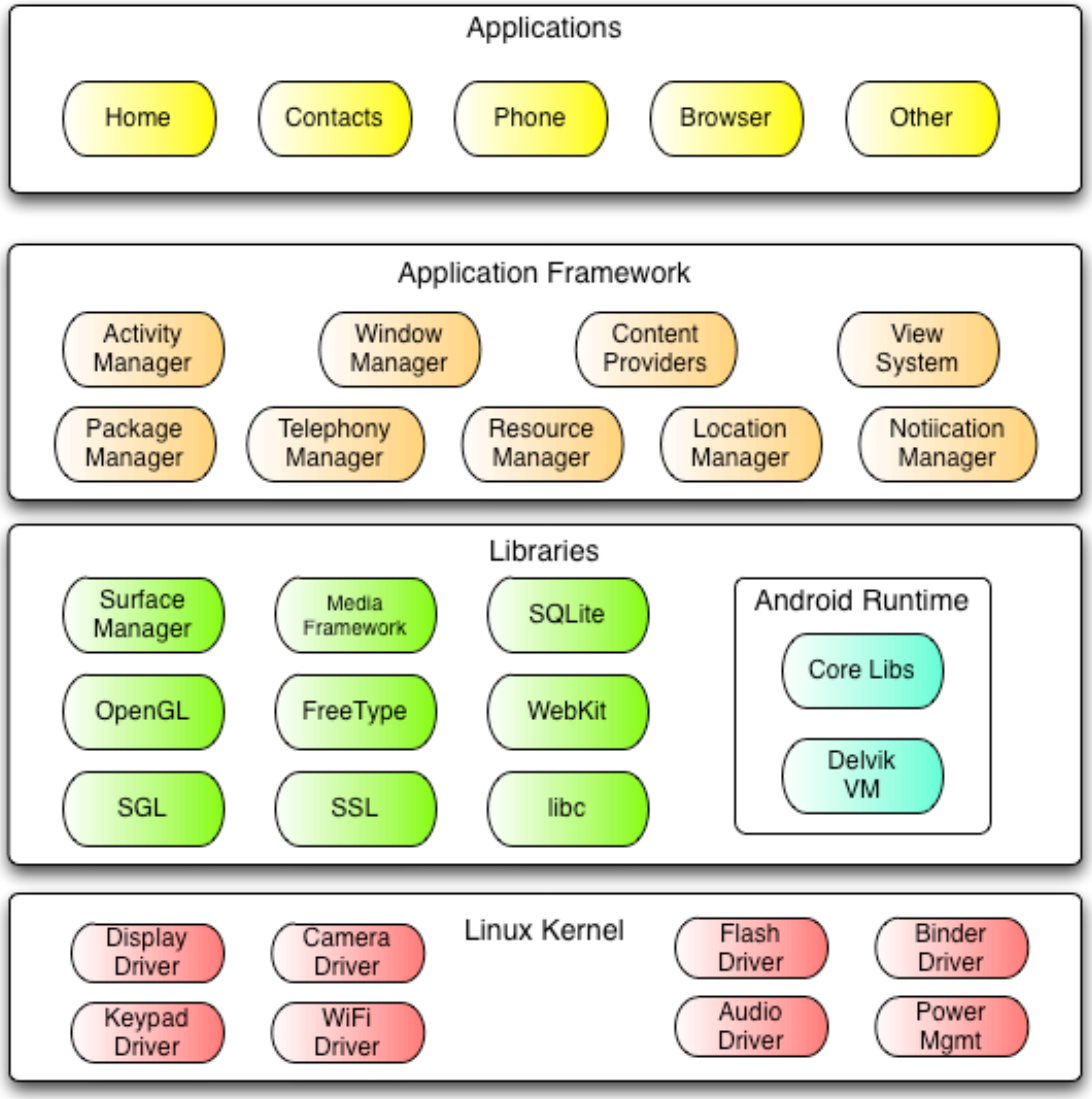


marakana

ANDROID STACK

marakana

The Stack



marakana

Linux Kernel

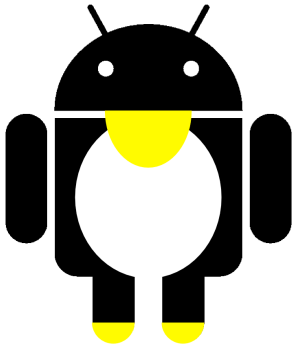
Android runs on Linux.

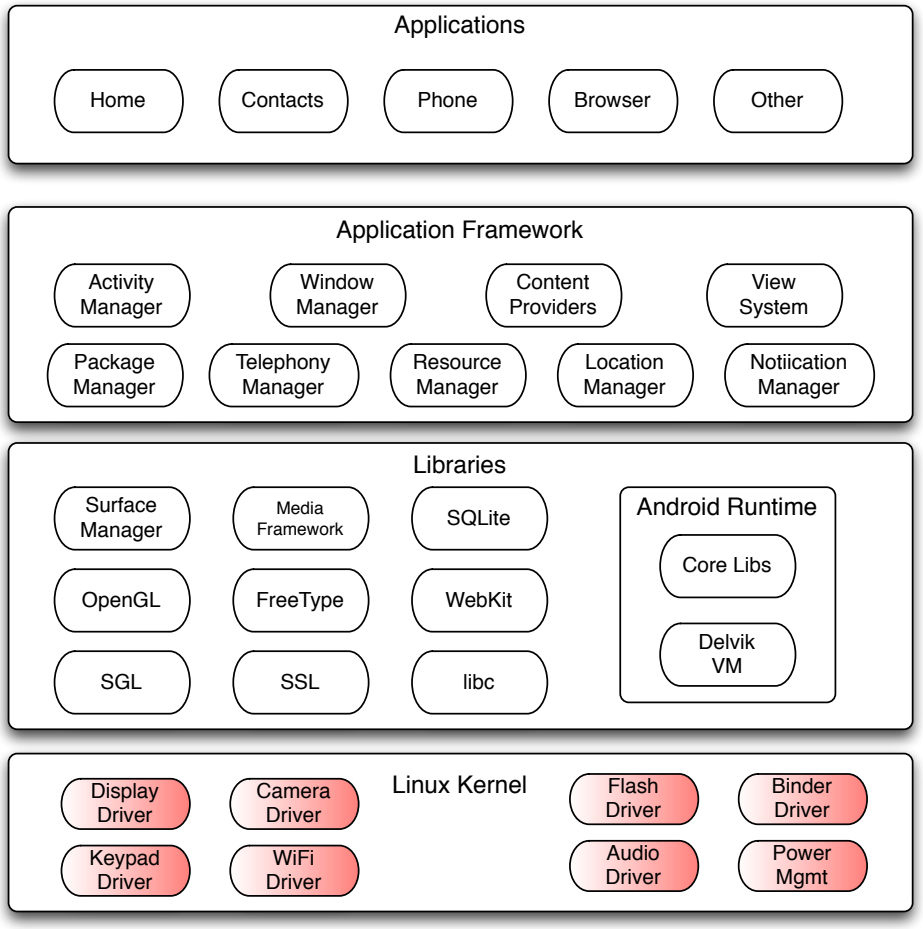
Linux provides as well as:

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

Users never see Linux sub system

The adb shell command opens Linux shell





marakana

Native Libraries

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

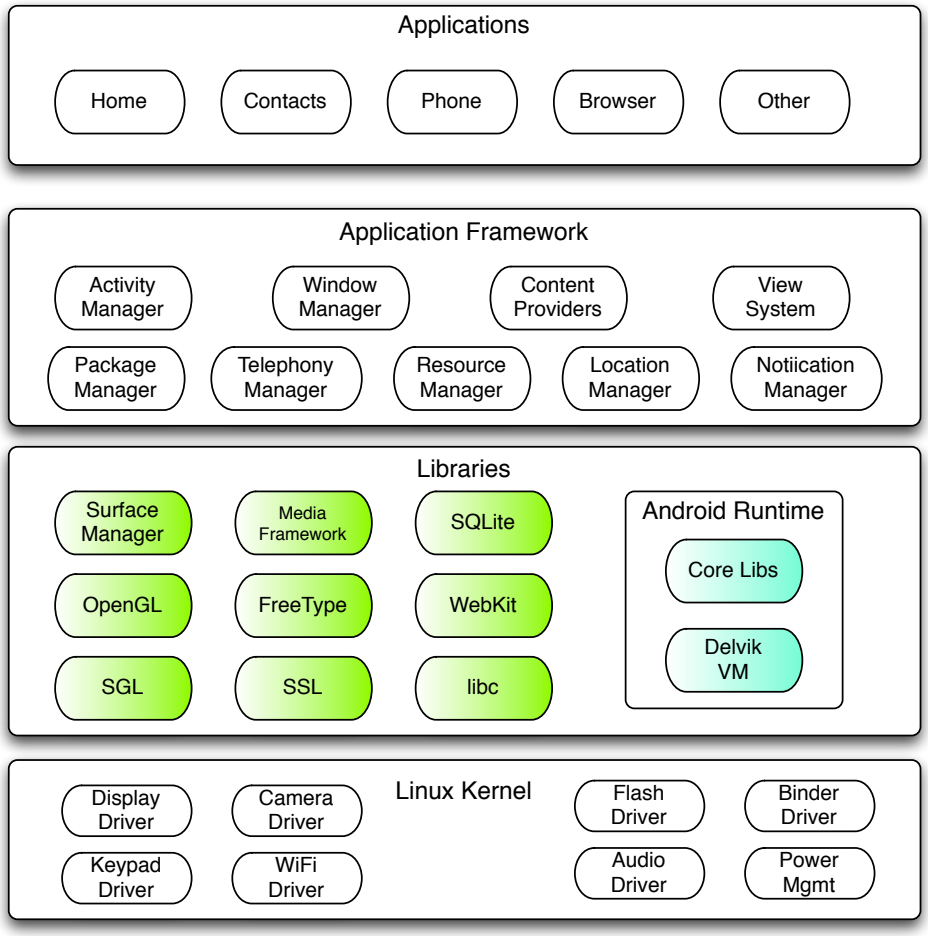
Surface Manager for composing window manager with off-screen buffering

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

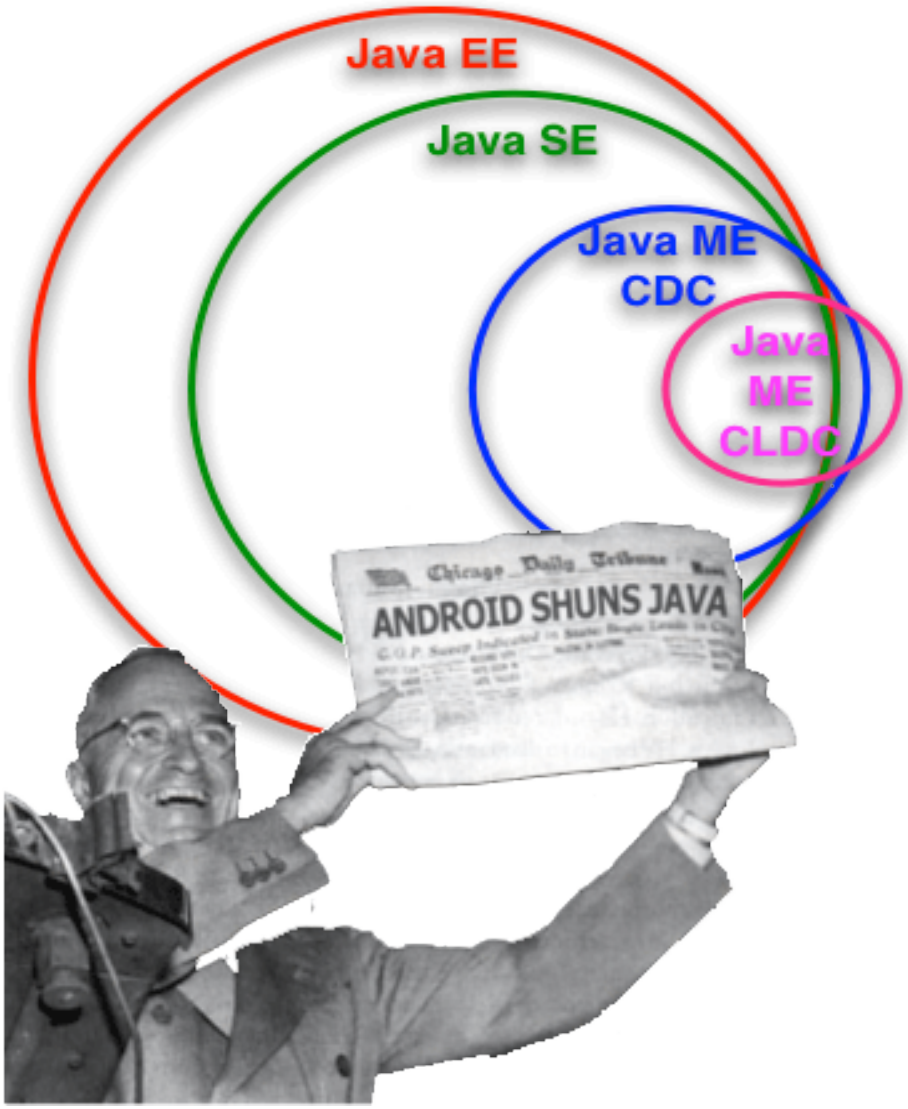
Optimized for mobile devices

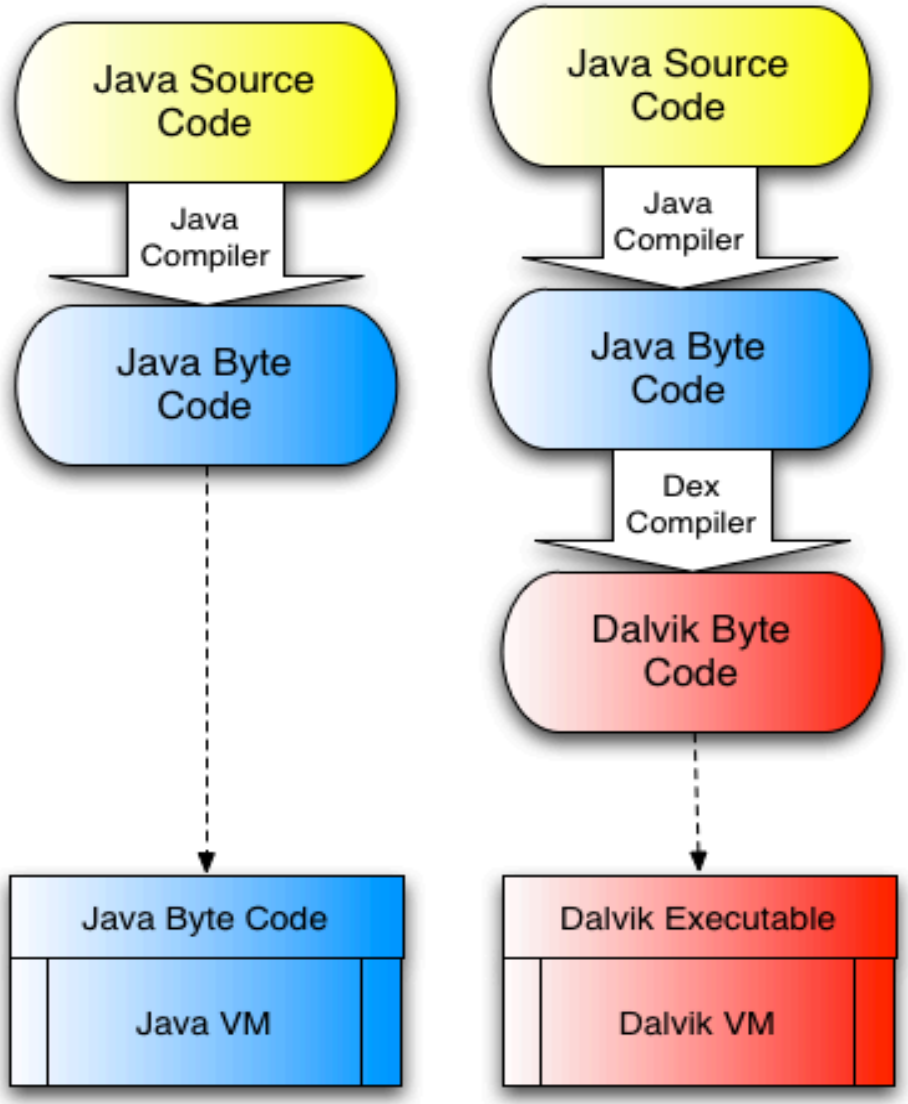
Key Dalvik differences:

- Register-based versus stack-based VM
- Dalvik runs .dex files
- More efficient and compact implementation
- Different set of Java libraries than SDK

marakana

Android and Java





marakana

Application Framework

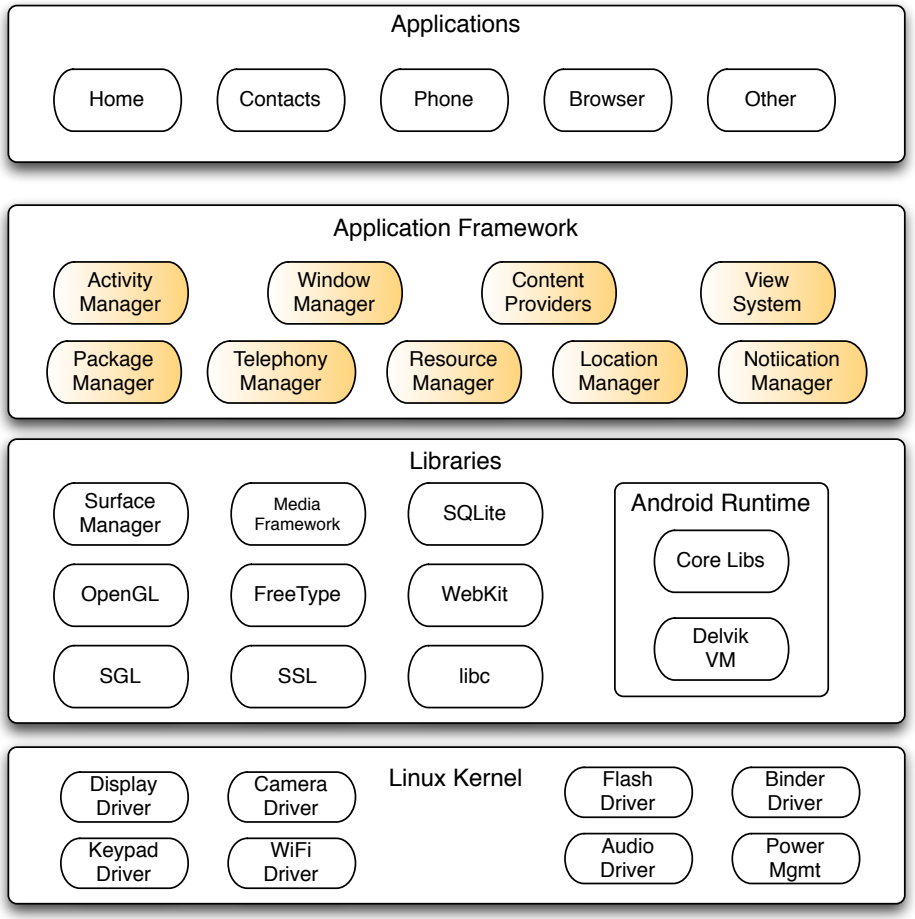
Activation manager controls the life cycle of the app

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

Resource manager manages everything that is not the code

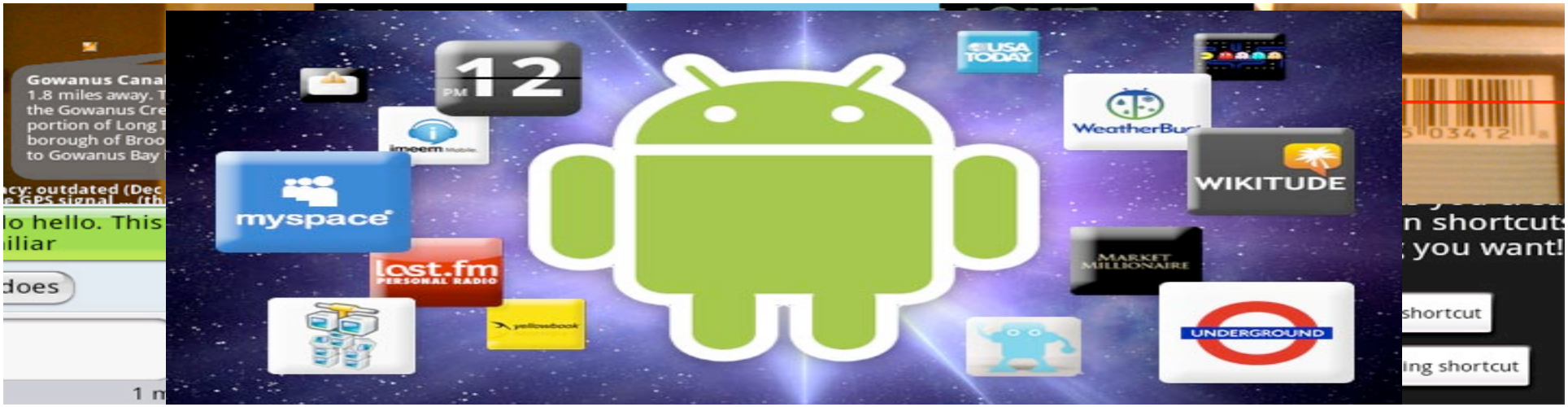
Location manager figures out the location of the phone (GPS, GSM, WiFi)

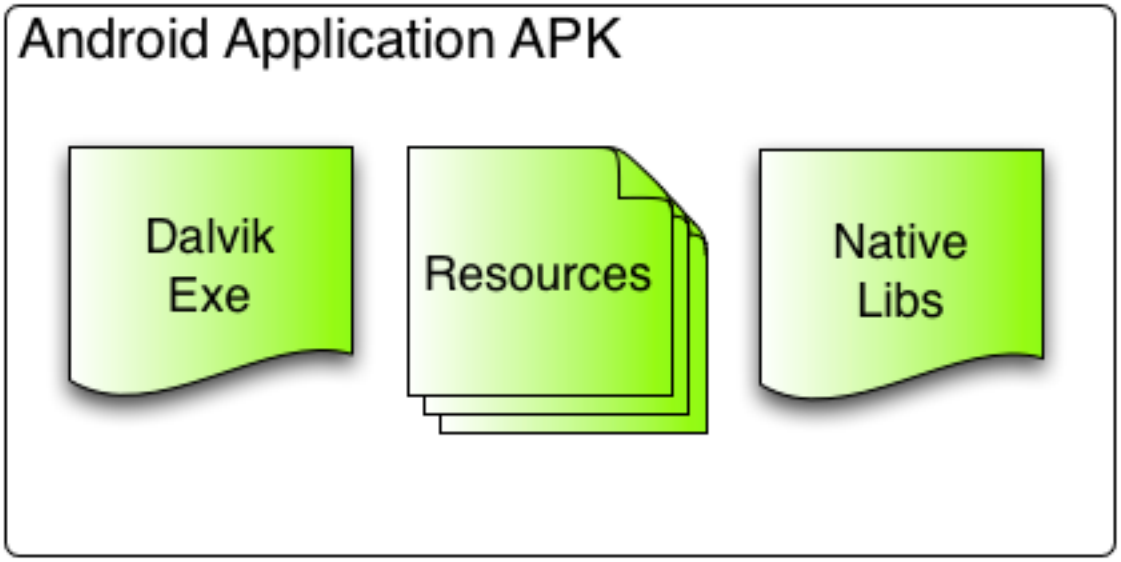
Notification manager for events such as arriving messages, appointments, etc



marakana

Applications





marakana

OPERATING SYSTEM FEATURES

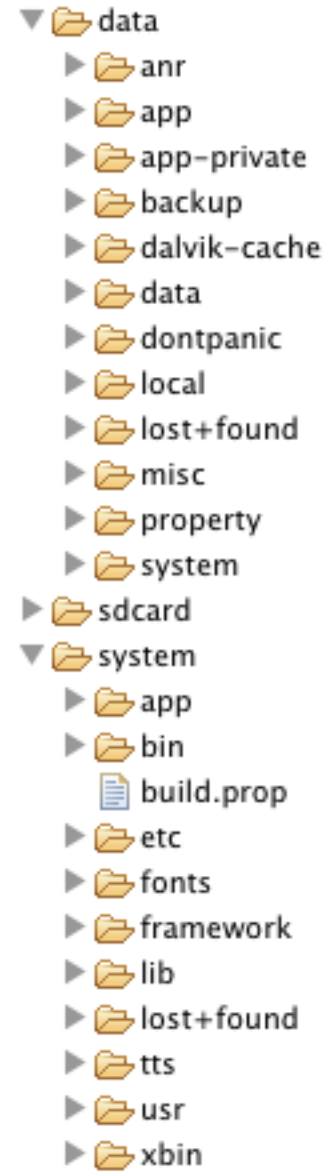
marakana

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.



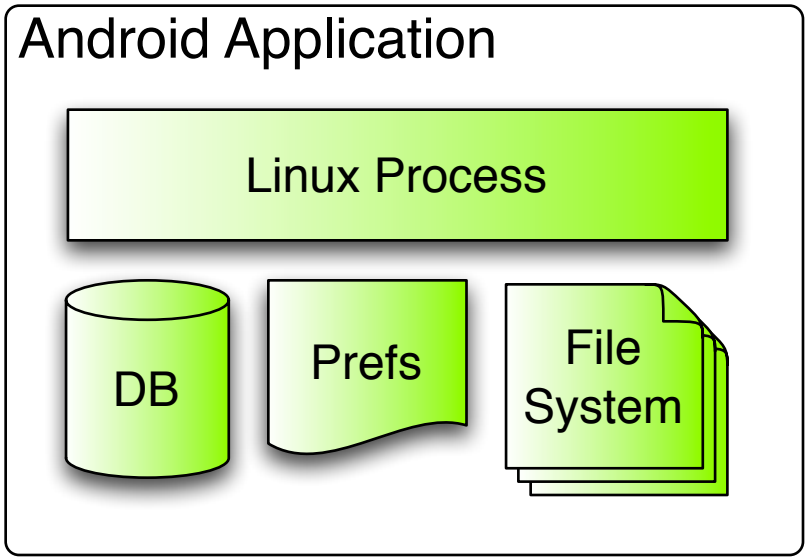
marakana

Security

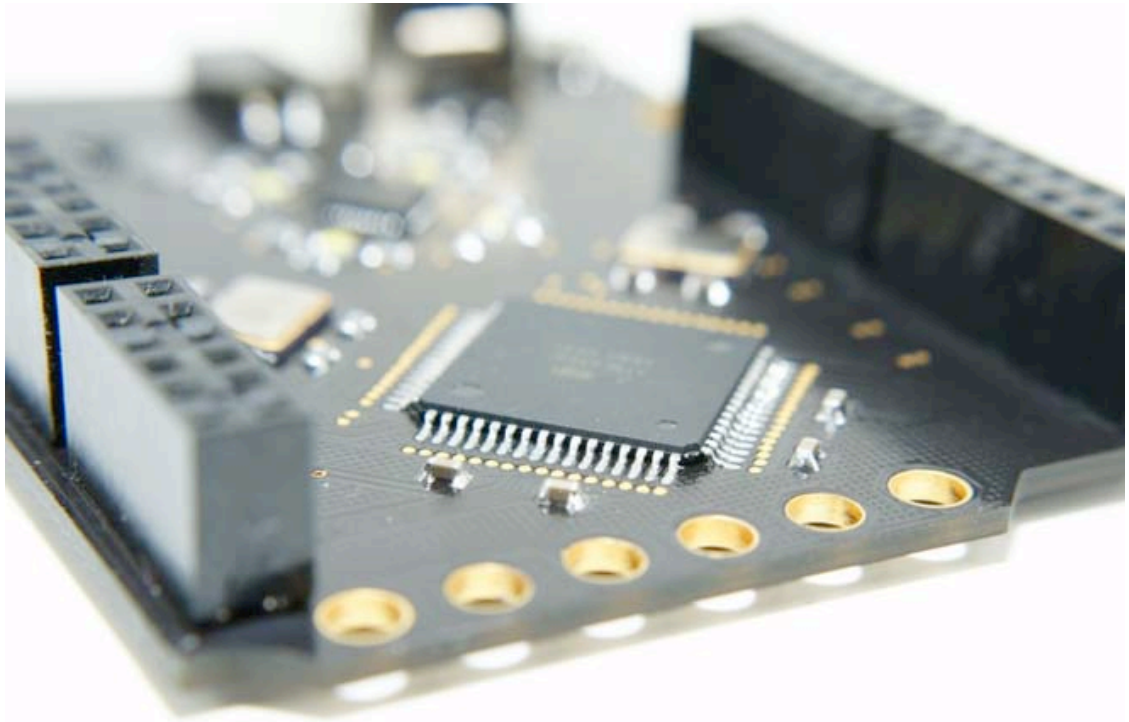
Each Android application runs inside its own Linux process.

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.



marakana



WORKING WITH HARDWARE

marakana

Camera

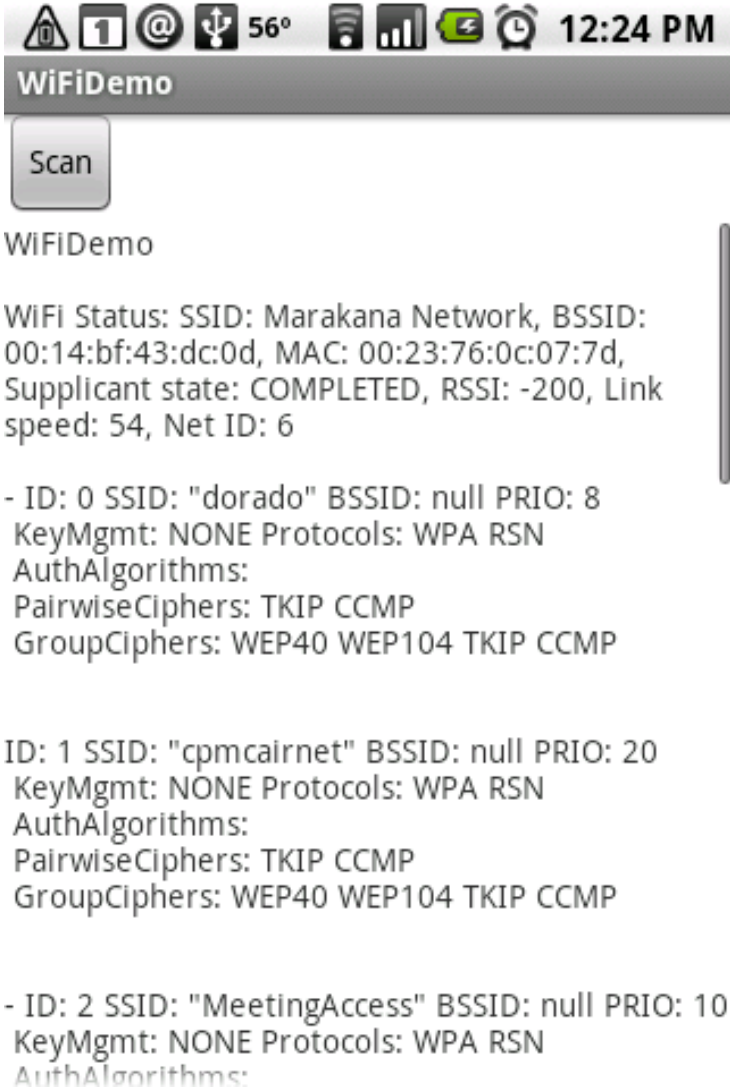
Android SDK supports access to built-in Camera and its preview.

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.

marakana

WiFi

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



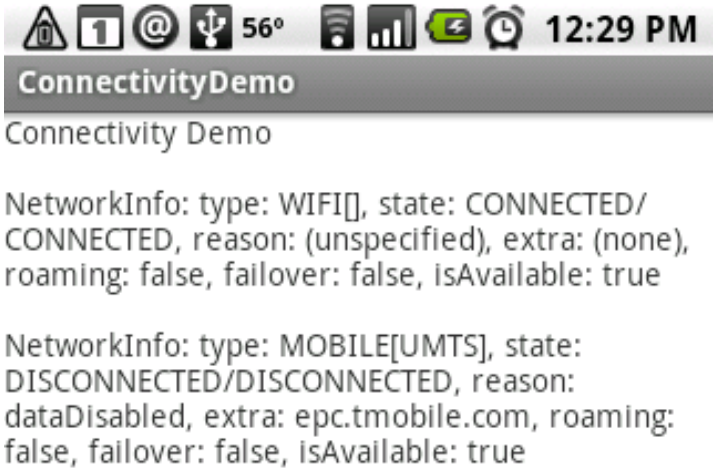
marakana

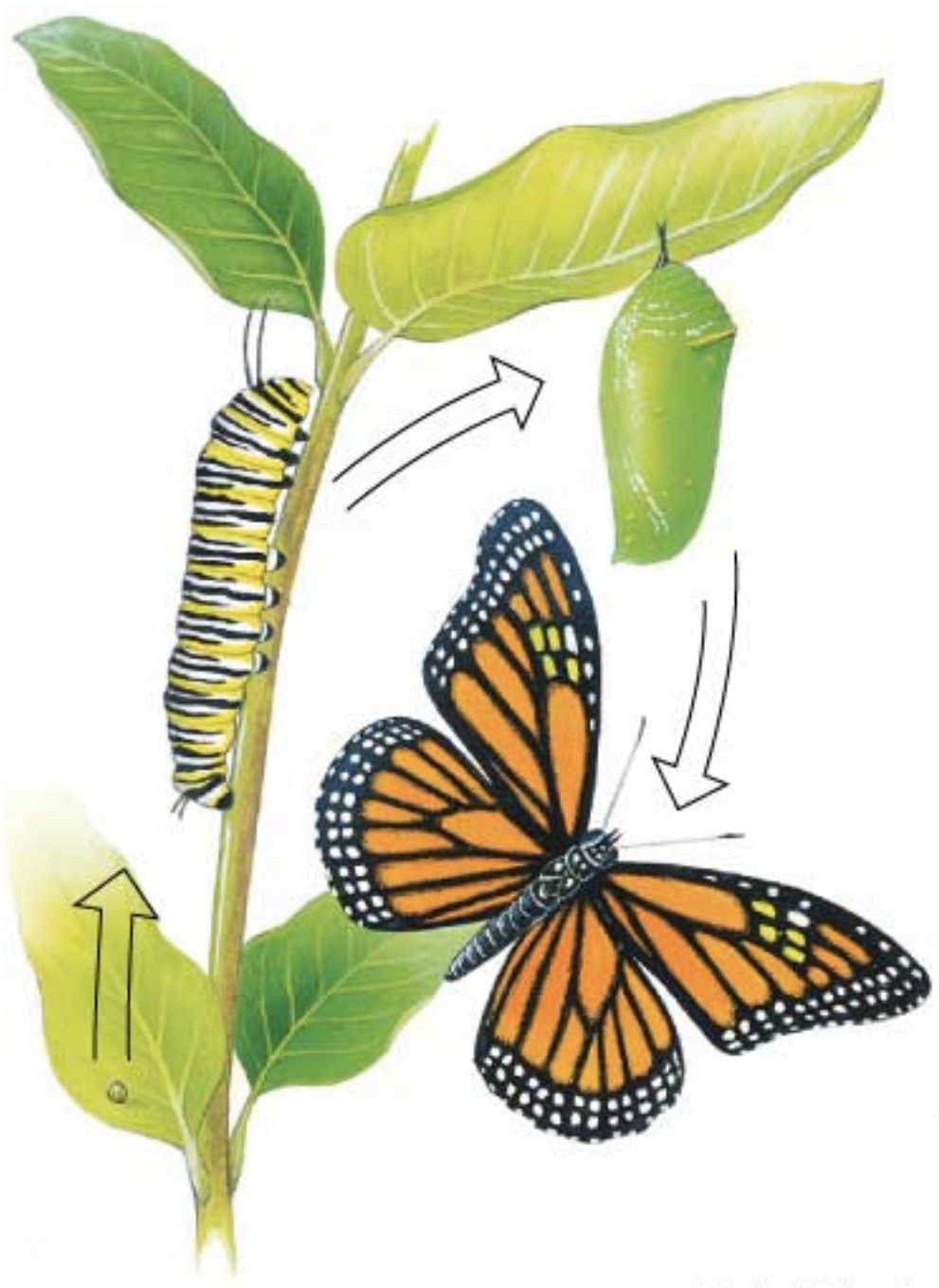
Telephony

With Telephony API, you can:

Make phone calls
Monitor phone state and activity
Access phone properties and status
Monitor data connectivity
Control the phone

It is a simple yet powerful API



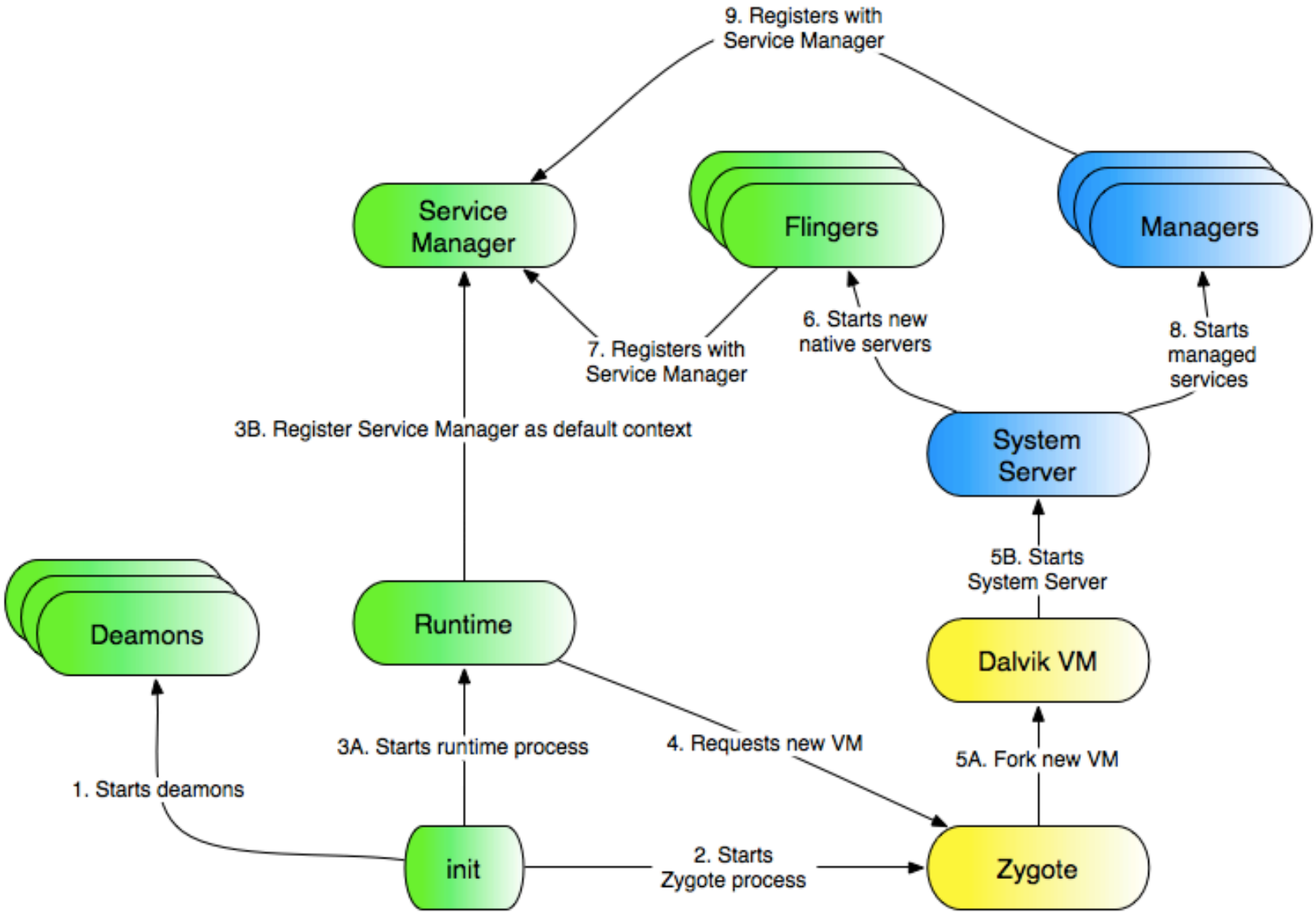


marakana

ANDROID STARTUP & RUNTIME

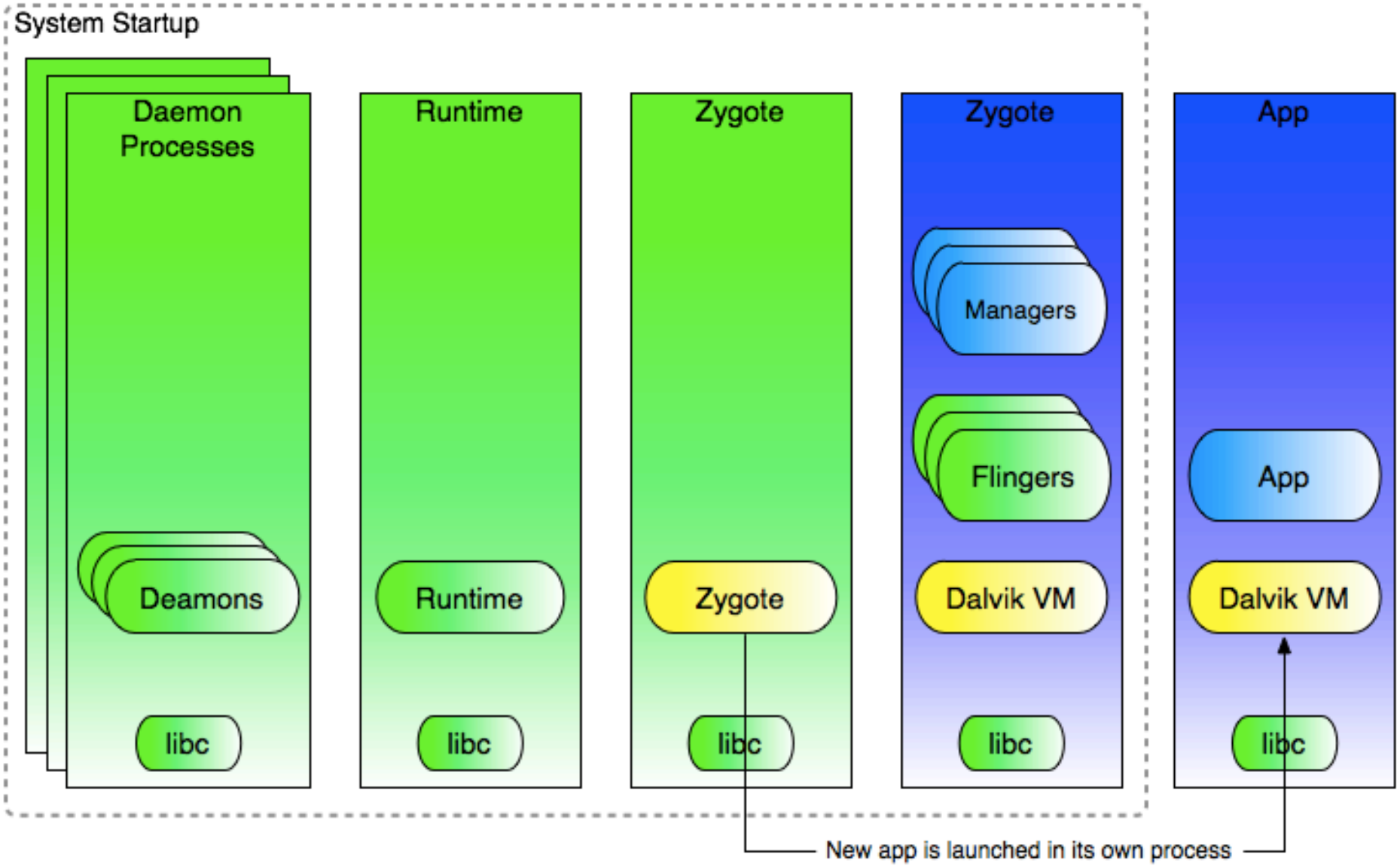
marakana

Startup Walkthrough



marakana

Runtime Overview



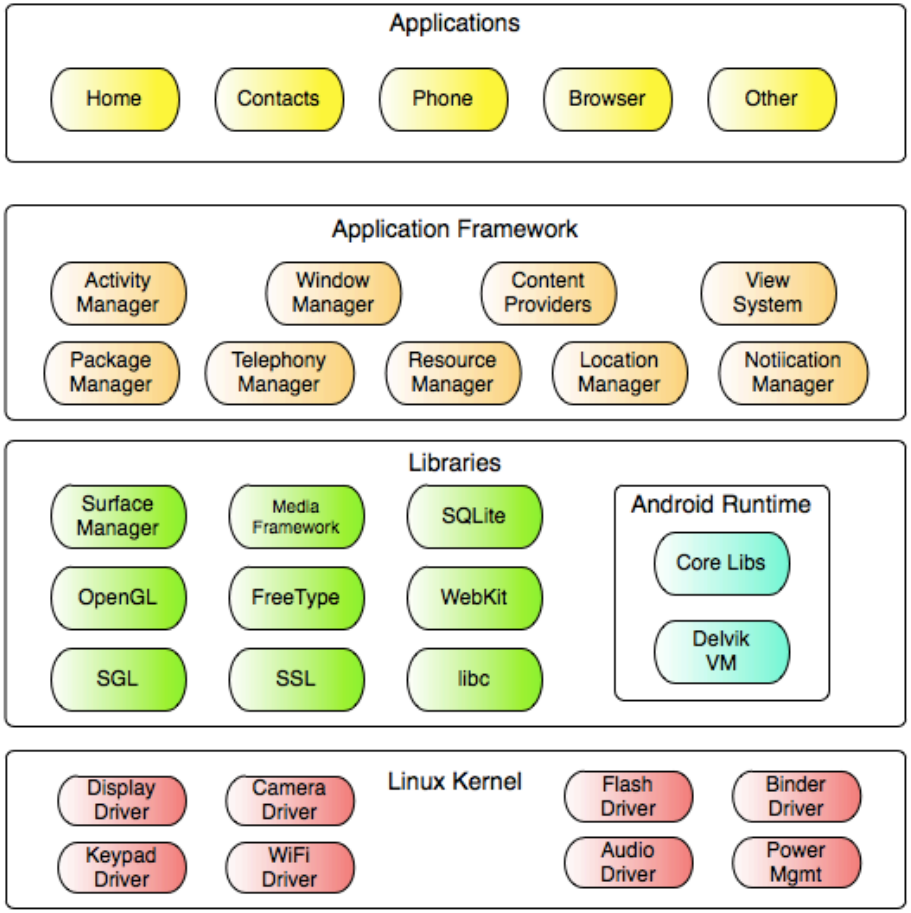
marakana

Layer Interactions

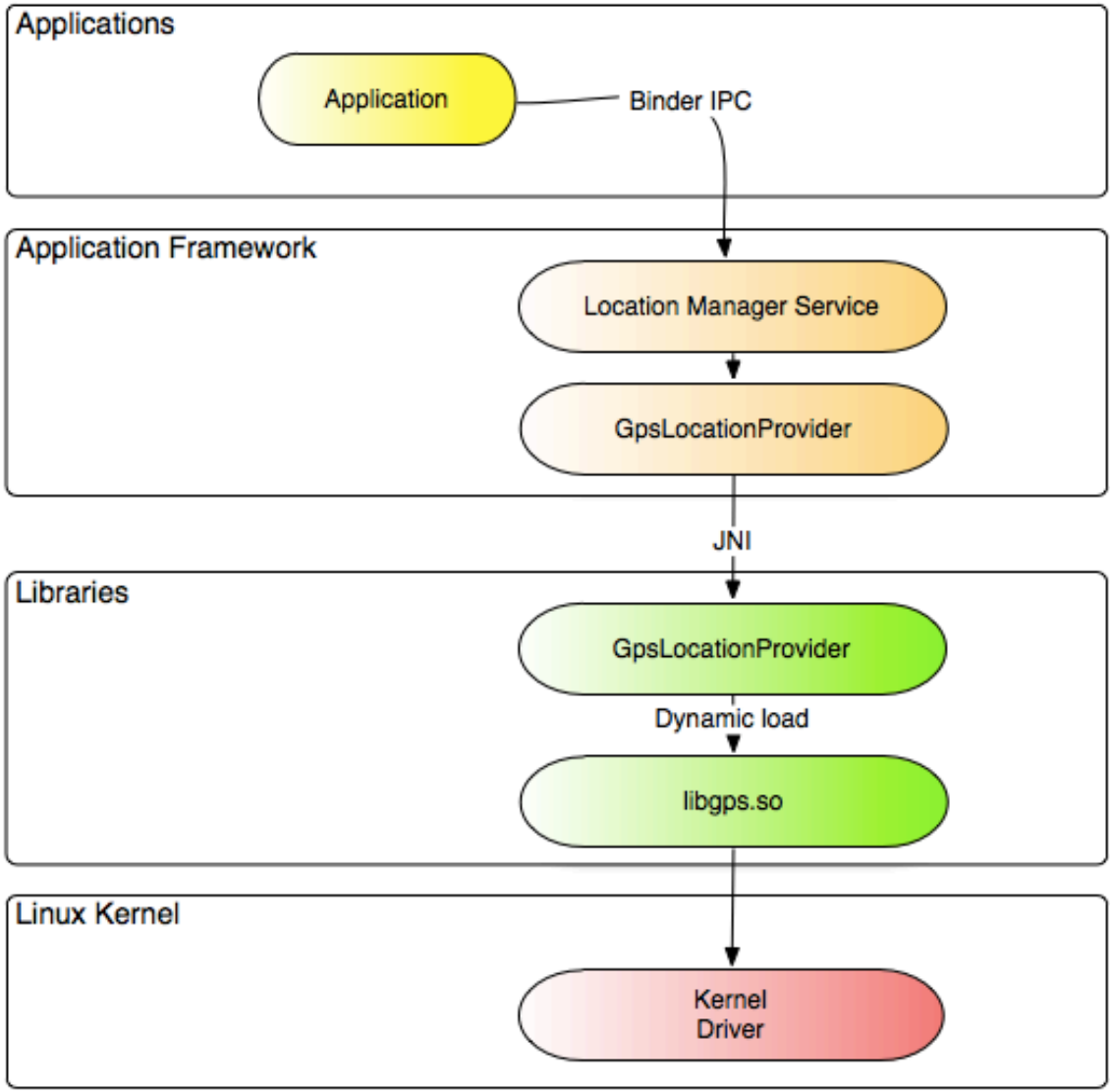
There are three main scenarios for your app to talk to native library:

- Directly
- Via native service
- Via native daemon

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

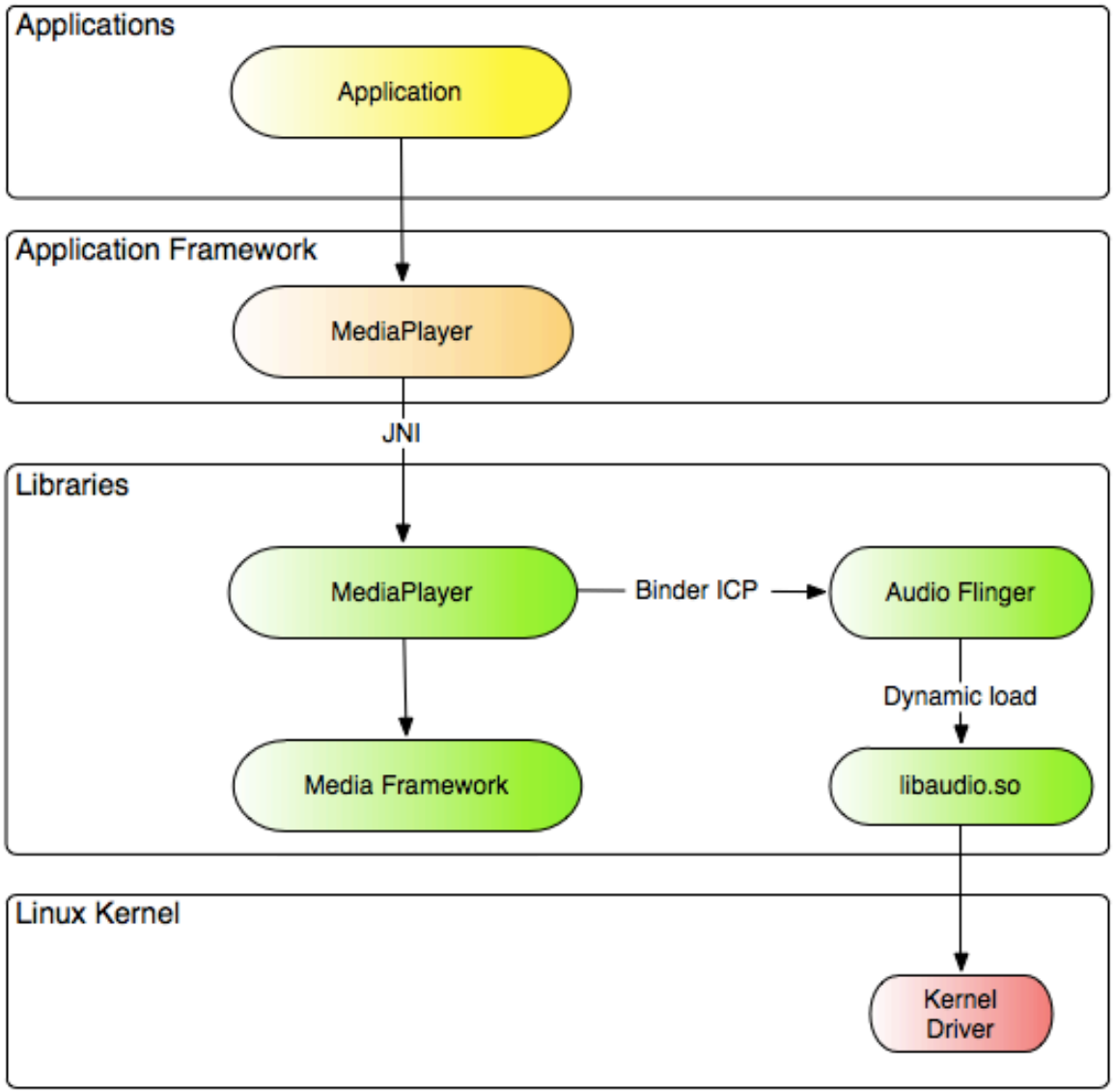


App – Runtime Service - Lib



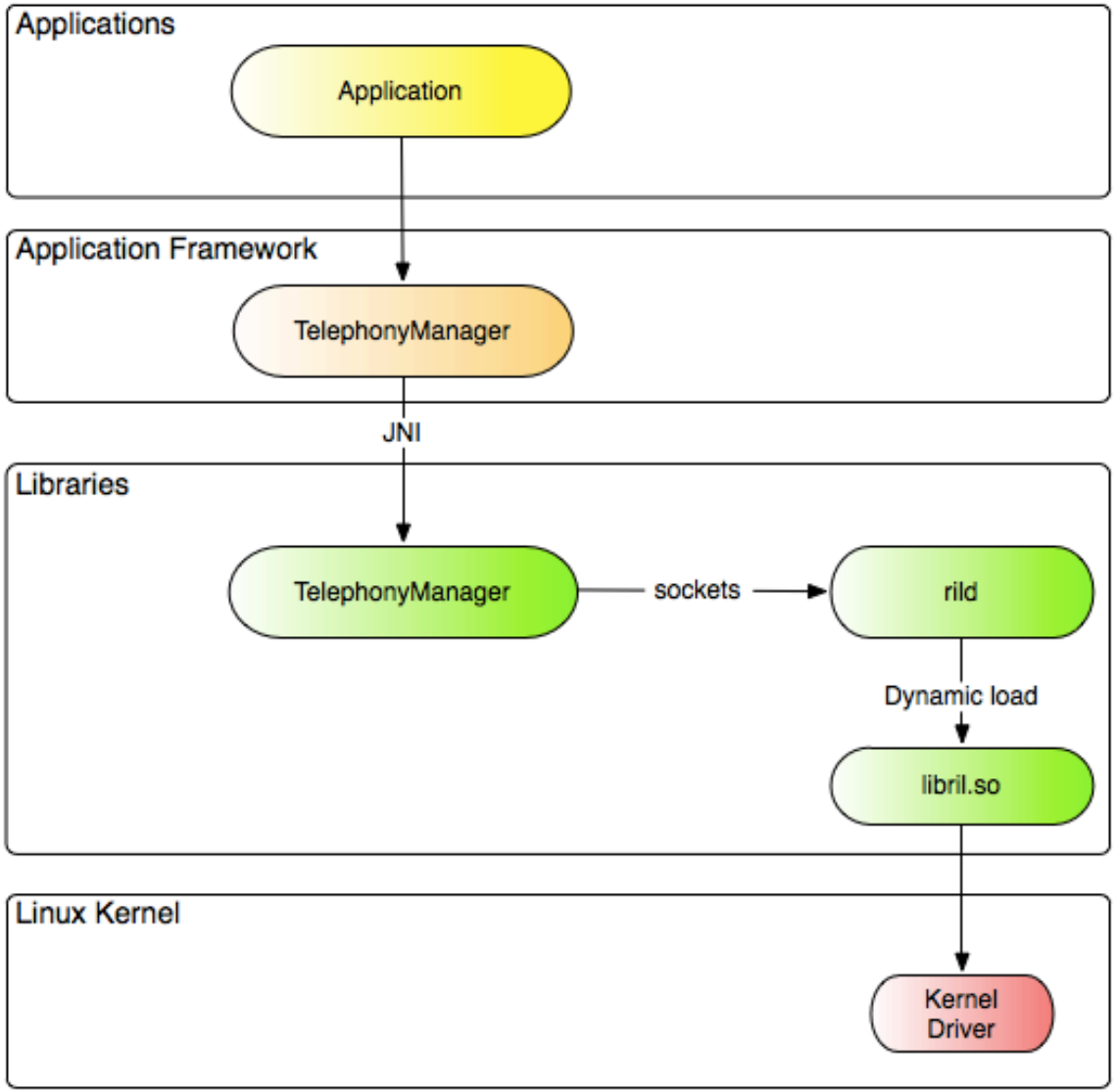
marakana

App – Runtime-Native Service-Lib



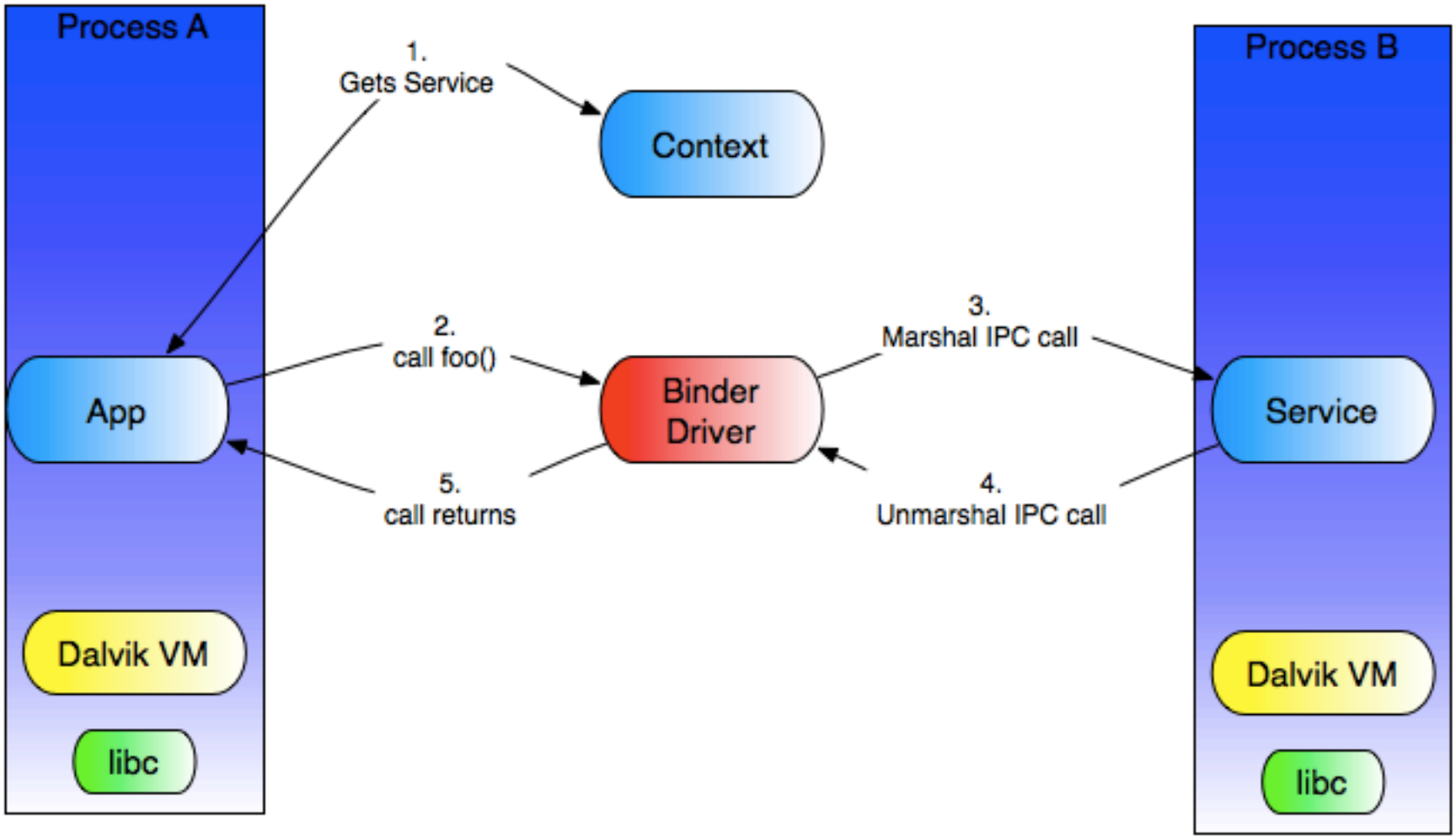
marakana

App–Runtime–Native Daemon-Lib



marakana

Binder IPC



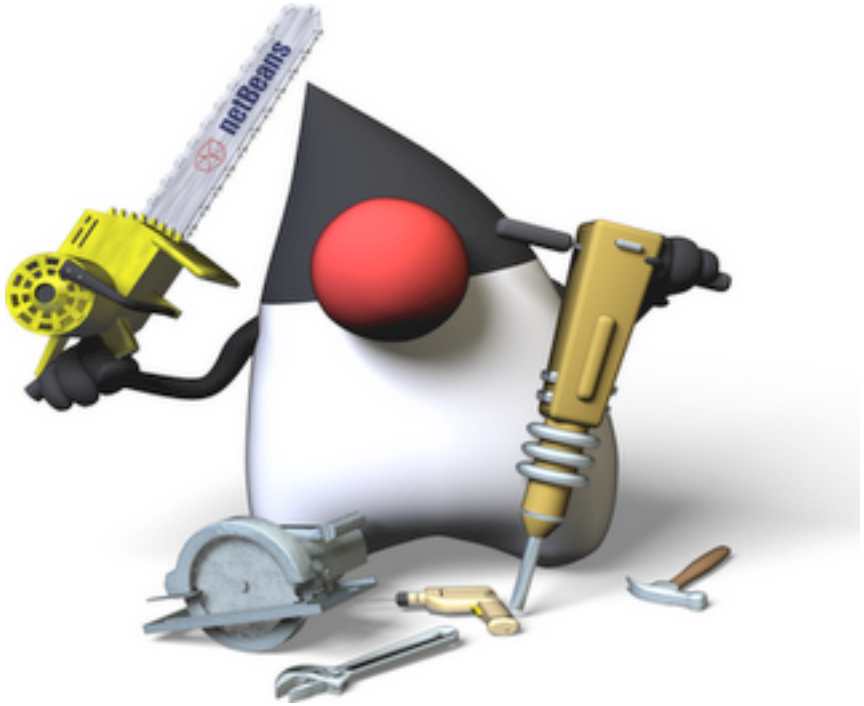
High-performance IPC: shared memory, per-process thread pool, synchronous

marakana

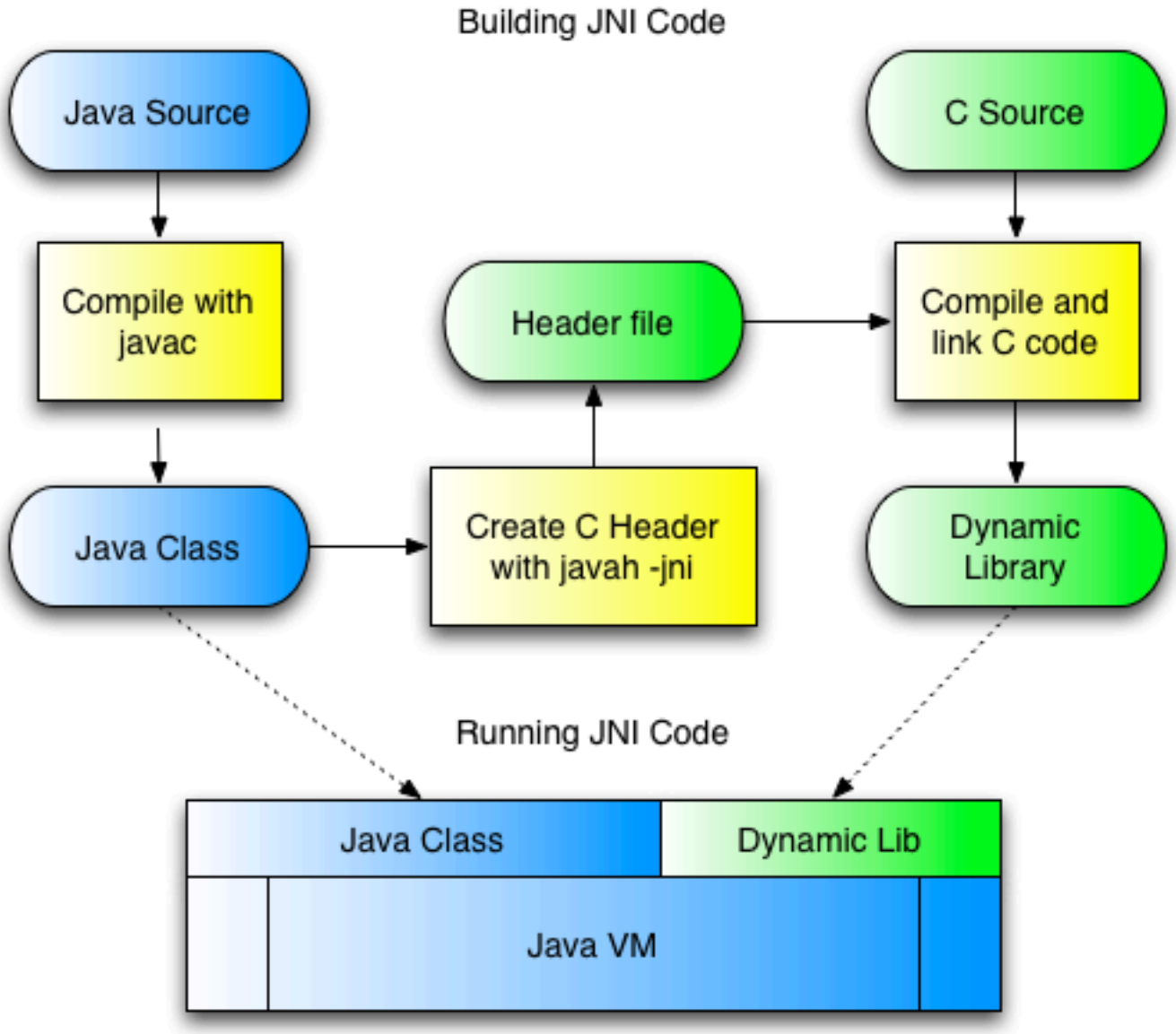
Java Native Interface

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

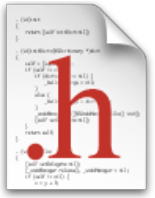
NATIVE DEVELOPMENT KIT

marakana

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

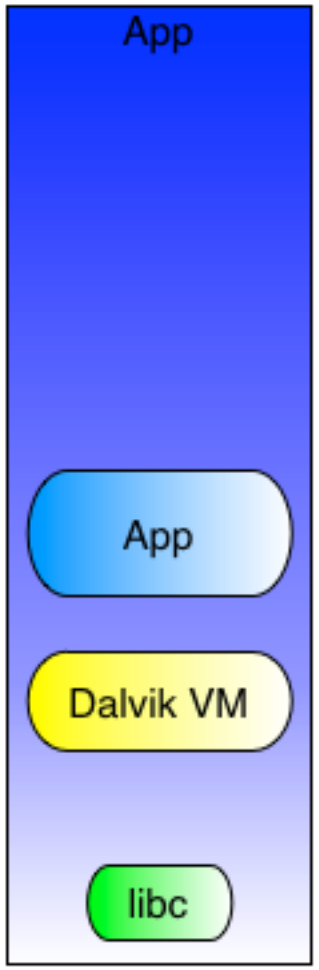
marakana

Why NDK?

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

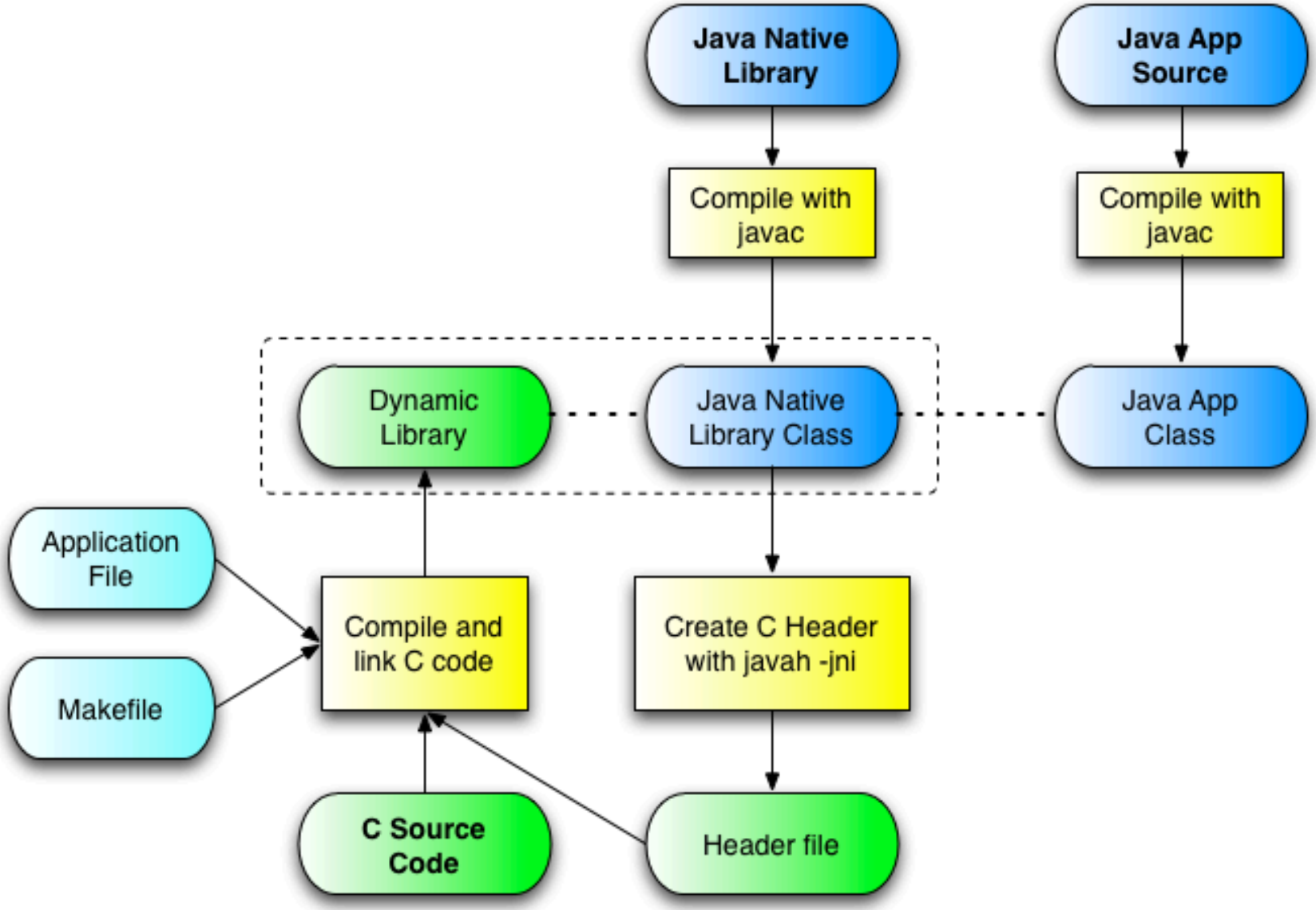
You cannot develop native-only apps in NDK – your app is still subject to security sandboxing.

Main motivation for native code is performance.



marakana

Using NDK



DEBUGGING ANDROID APPS

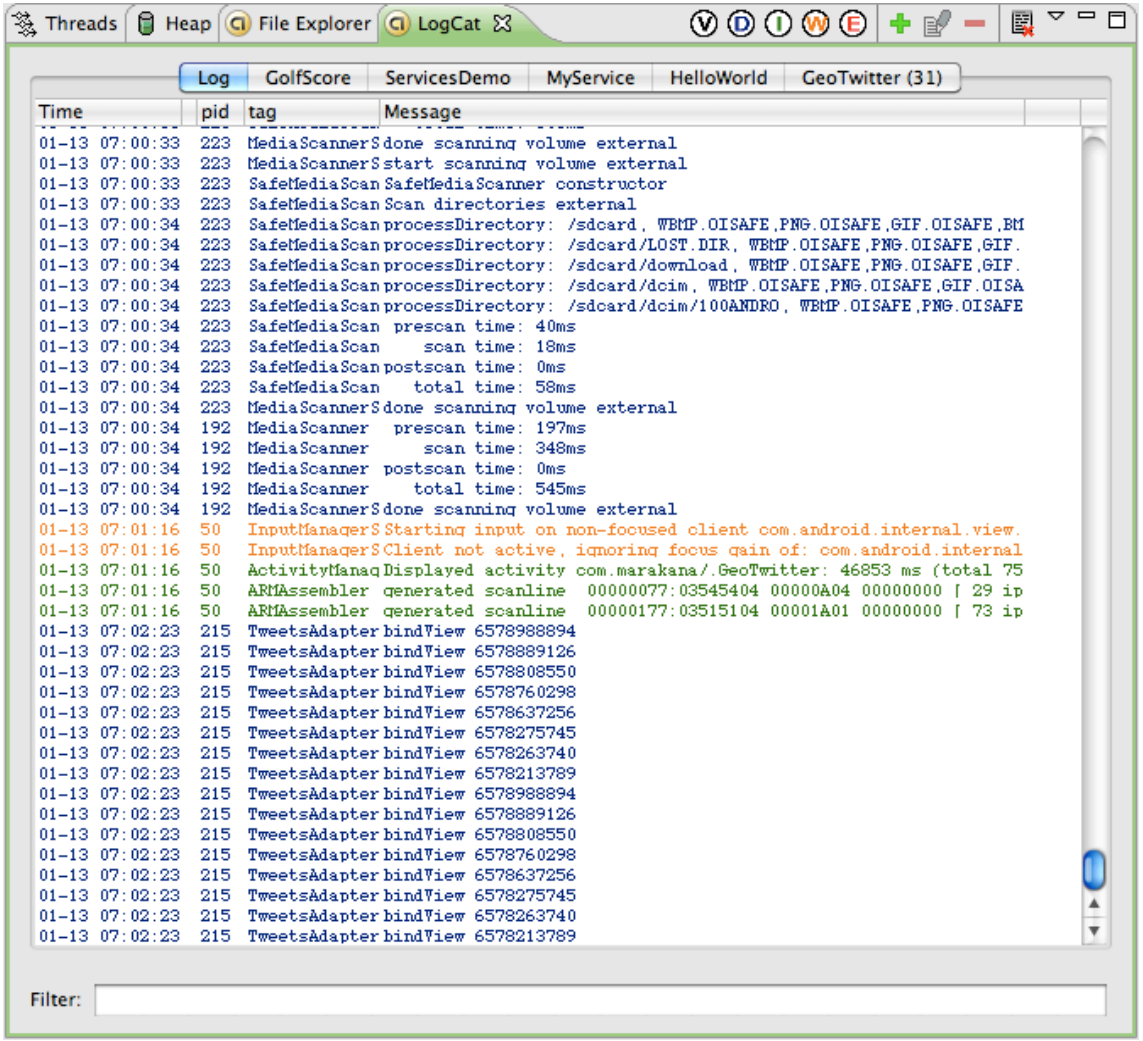
marakana

LogCat

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

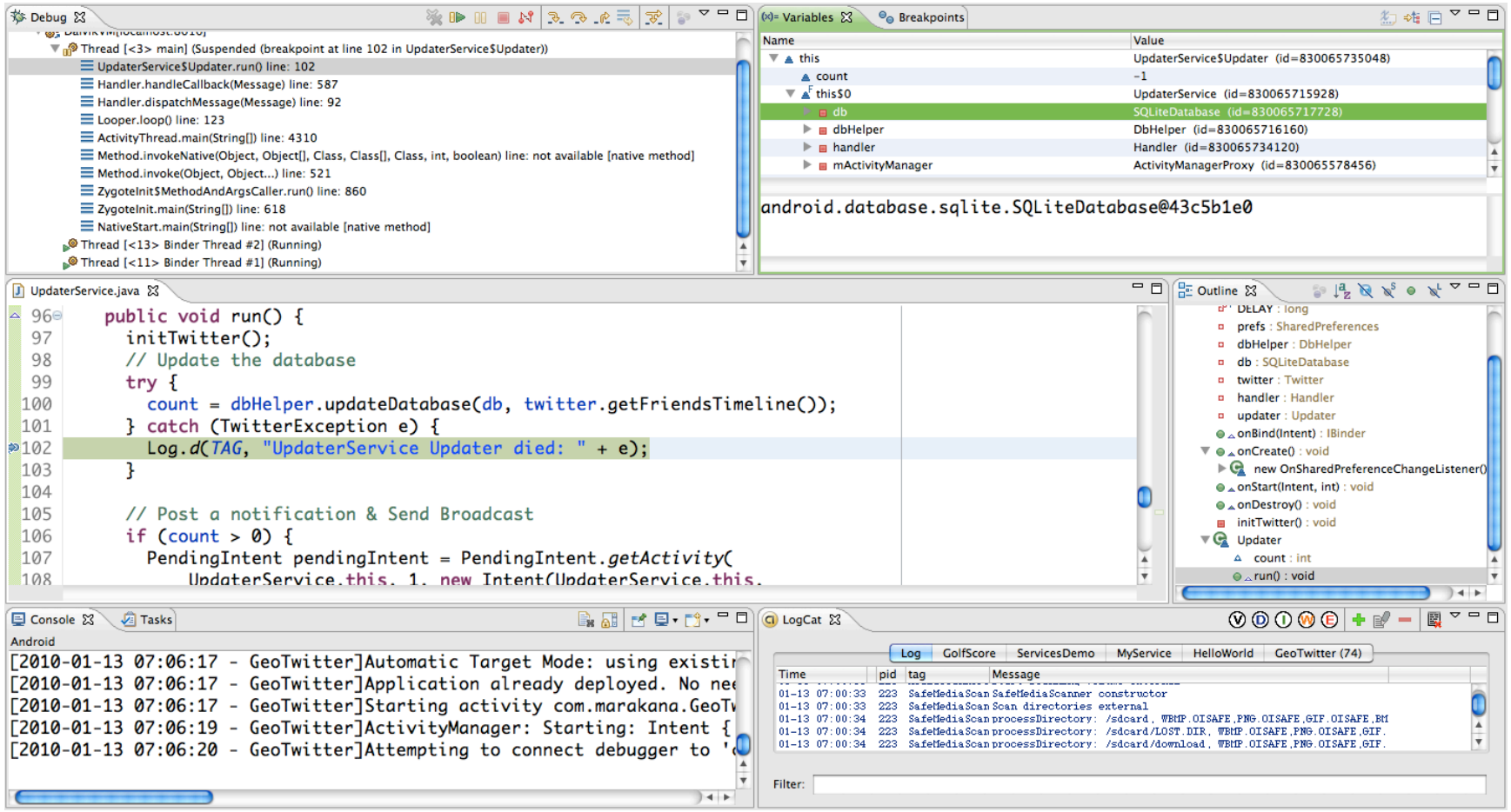
Can be viewed via command line or Eclipse.

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



marakana

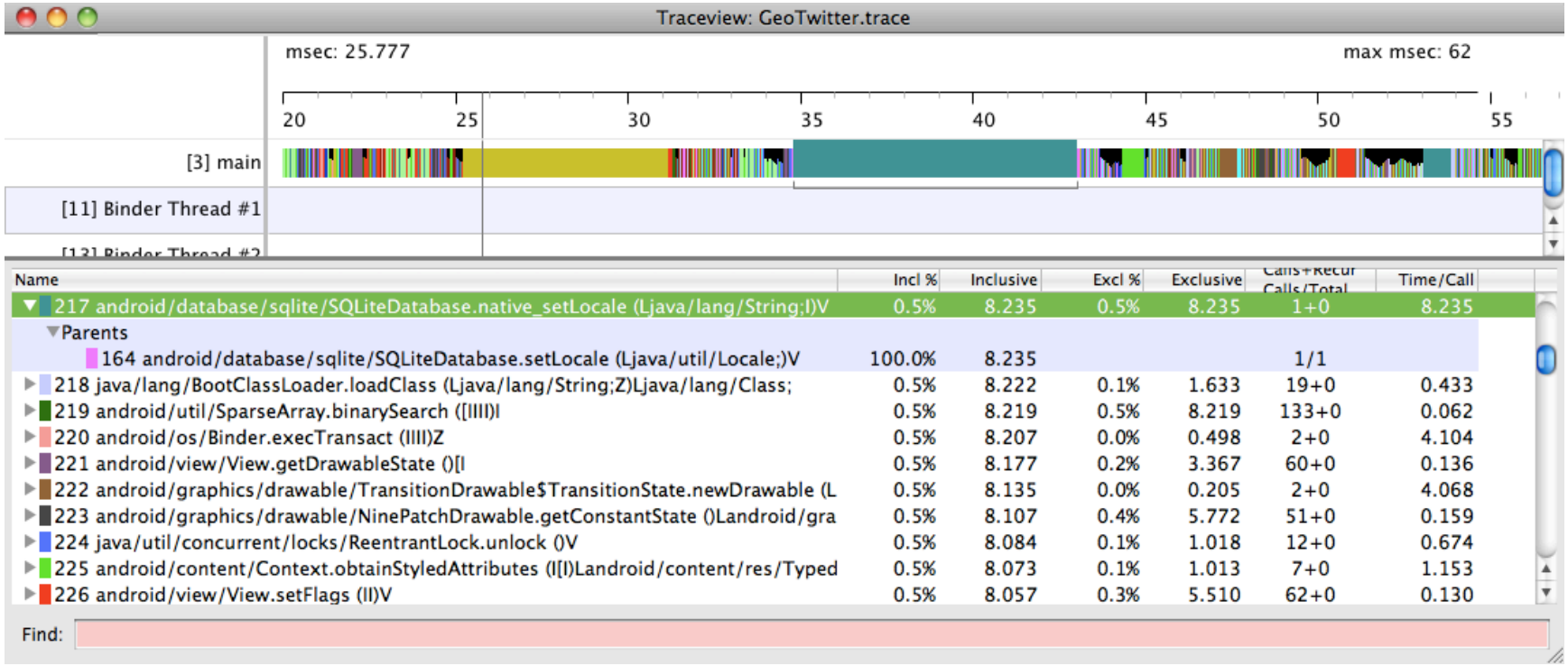
Debugger



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

marakana

TraceView



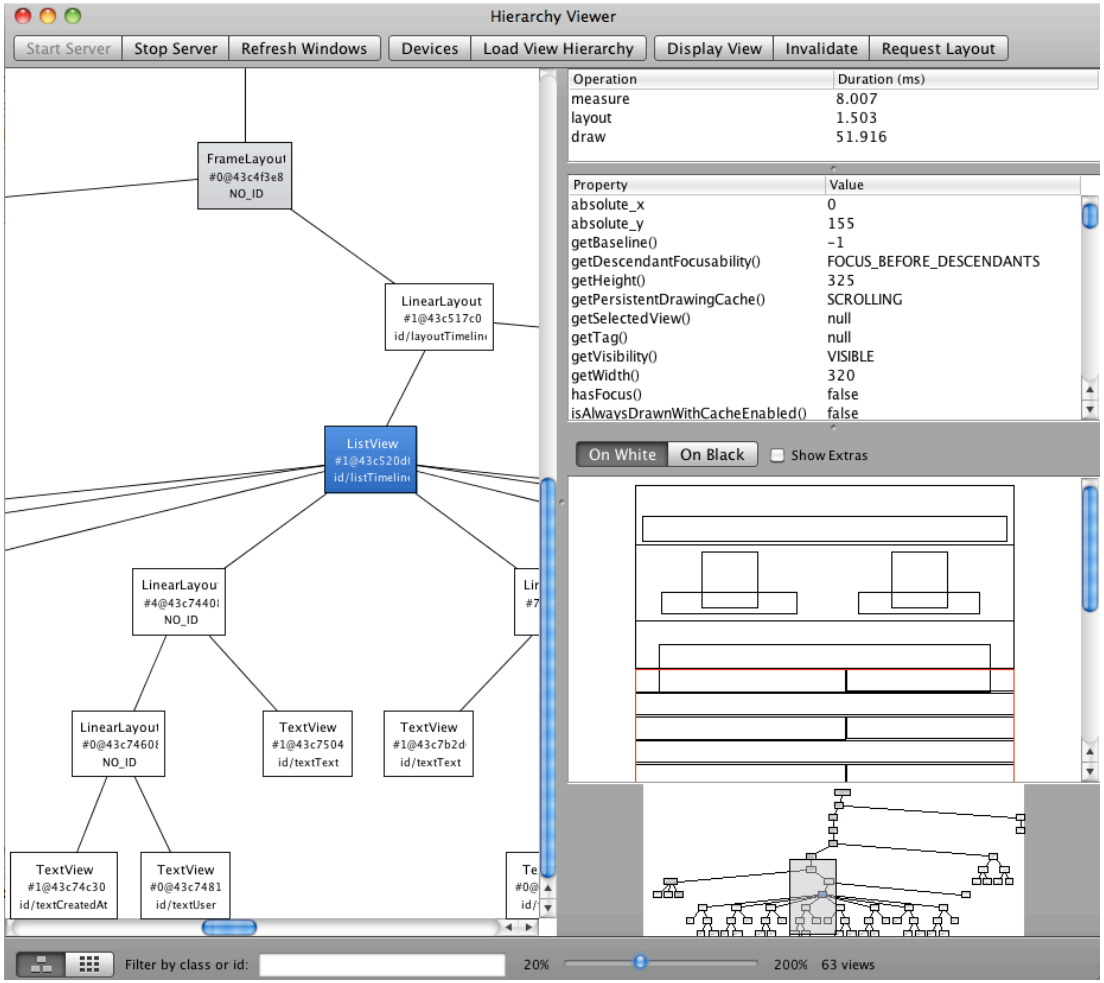
TraceView helps you profile you application and find bottlenecks. It shows execution of various calls through the entire stack. You can zoom into specific calls.

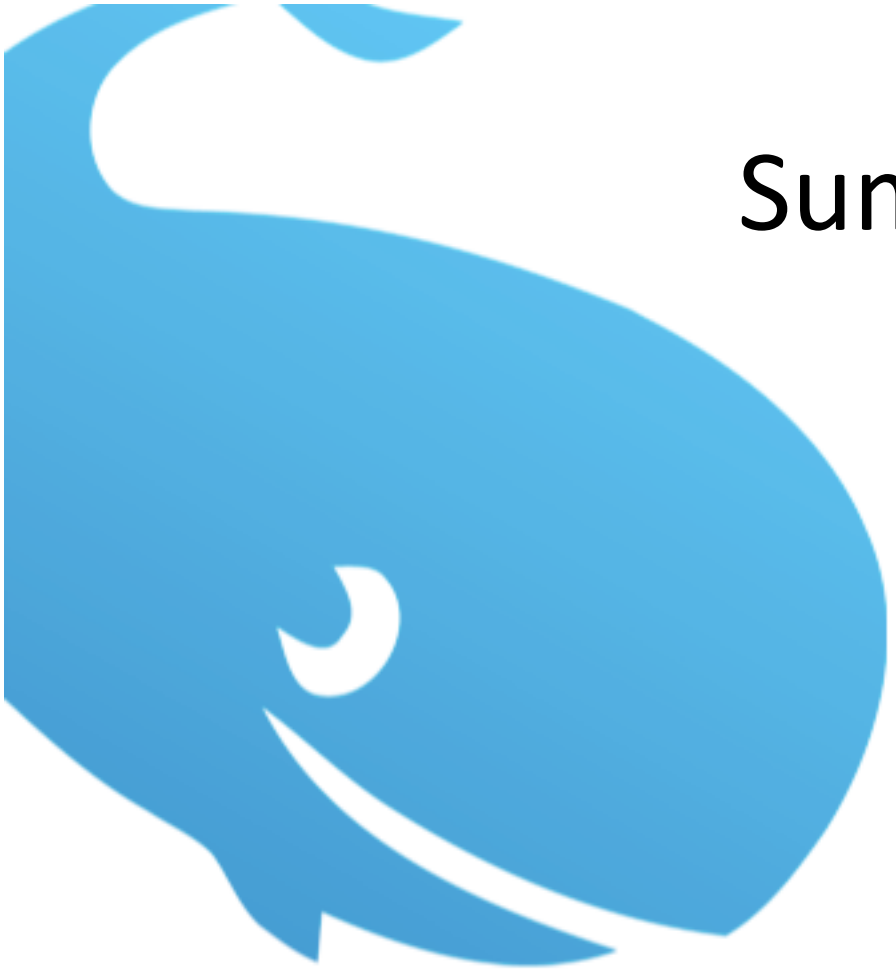
marakana

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.





marakana

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