


marakana  
●●●

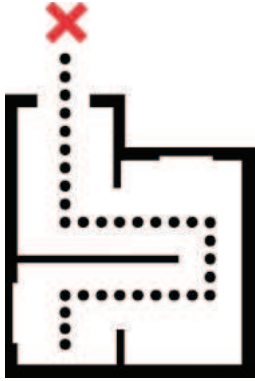
## Android: A 9,000-foot Overview

Marko Gargenta  
Marakana



## Agenda

- Android History
- Android and Java
- The Stack
- Android SDK
- Hello World!
- Main Building Blocks
- Android User Interface
- Debugging
- Summary



## History

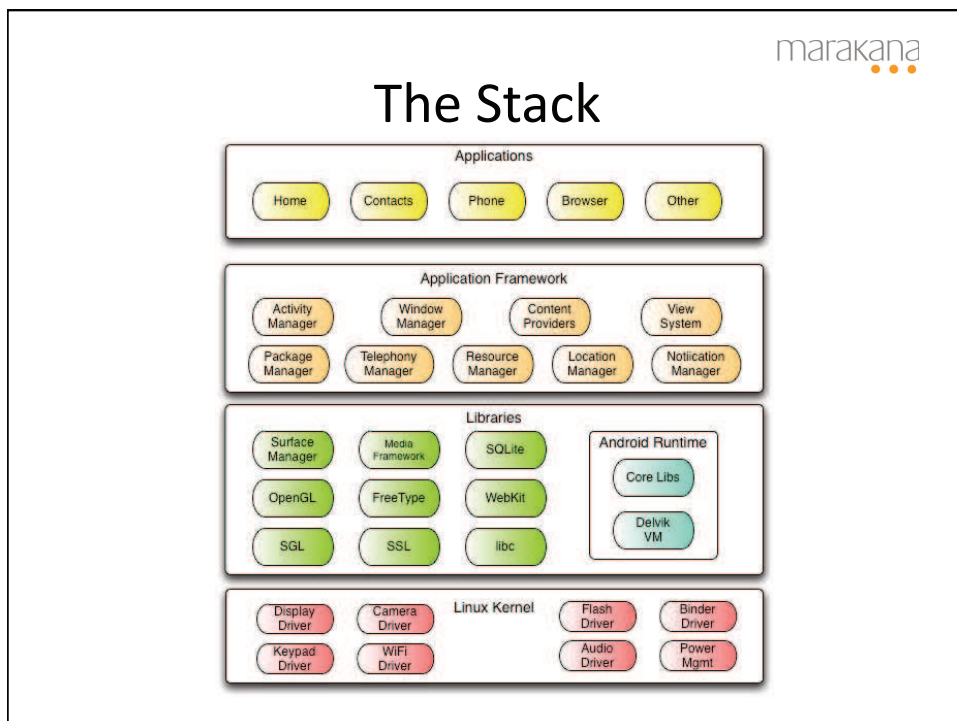
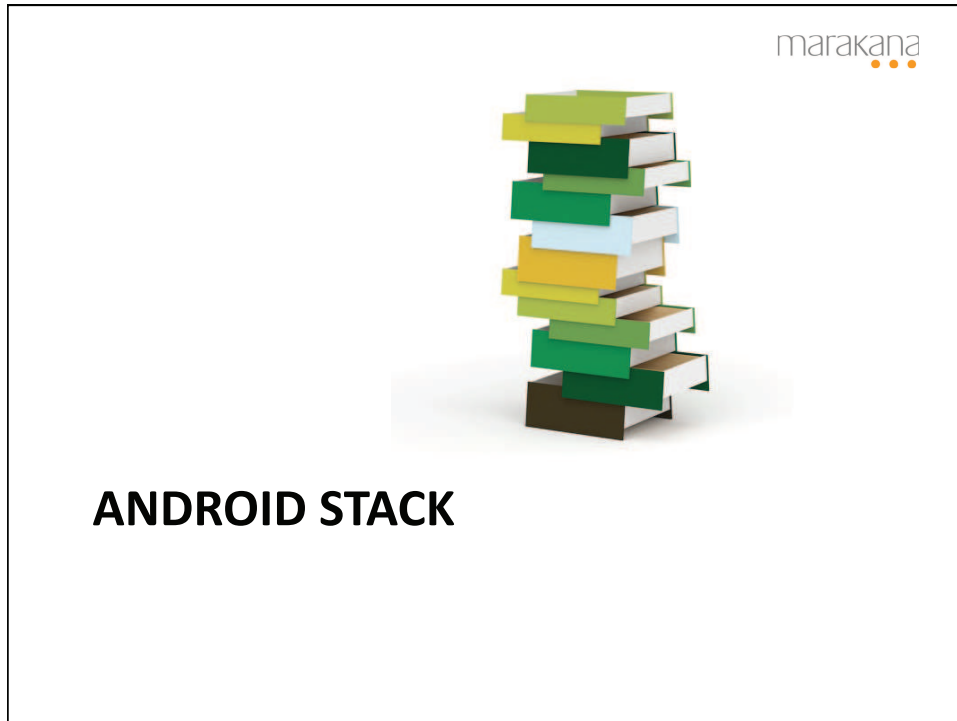
2005	Google buys Android, Inc. Work on Dalvik starts
2007	OHA Announced Early SDK
2008	G1 Announced SDK 1.0 Released
2009	G2 Released Cupcake, Donut, Eclair

## Android and Java

Android Java =  
Java SE –  
AWT/Swing +  
Android API

```

graph TD
    subgraph Java_SE [Java SE]
        JS1[Java Source Code] -- Java Compiler --> JB1[Java Byte Code]
        JB1 -.-> JVM[Java VM]
    end
    subgraph Android [Android]
        JS2[Java Source Code] -- Java Compiler --> JB2[Java Byte Code]
        JB2 -- Dex Compiler --> DB[Dalvik Byte Code]
        DB -.-> DE[Dalvik Executable]
        DE -.-> DVM[Dalvik VM]
    end
    JB1 -.-> JB2
    
```



## Linux Kernel


marakana

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



Applications

Home    Contacts    Phone    Browser    Other

---

Application Framework

Activity Manager    Window Manager    Content Providers    View System  
 Package Manager    Telephony Manager    Resource Manager    Location Manager    Notification Manager

---

Libraries

Surface Manager    Media Framework    SQLite    Android Runtime  
 OpenGL    FreeType    WebKit    Core Libs  
 SGL    SSL    libc    Dalvik VM

---

Linux Kernel

Display Driver    Camera Driver    Flash Driver    Binder Driver  
 Keypad Driver    WiFi Driver    Audio Driver    Power Mgmt

## Native Libraries

marakana

**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

Applications

Home    Contacts    Phone    Browser    Other

---

Application Framework

Activity Manager    Window Manager    Content Providers    View System  
 Package Manager    Telephony Manager    Resource Manager    Location Manager    Notification Manager

---

Libraries

Surface Manager    Media Framework    SQLite    Android Runtime  
 OpenGL    FreeType    WebKit    Core Libs  
 SGL    SSL    libc    Dalvik VM

---


Linux Kernel

Display Driver    Camera Driver    Flash Driver    Binder Driver  
 Keypad Driver    WiFi Driver    Audio Driver    Power Mgmt

marakana

## 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

## 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

## Android SDK - What's in the box

- SDK
- Tools
- Docs
- Platforms
  - Data
  - Skins
  - Images
  - Samples
- Add-ons
  - Google Maps



## The Tools

adb	hprof-conv
android	Jet
apkbuilder	layoutopt
ddms	lib
dmtracedump	mksdcard
draw9patch	sqlite3
emulator	traceview
hierarchyviewer	zipalign

Tools are important part of the SDK. They are available via Eclipse plugin as well as command line shell.

```

Cabotools markoS ls
Jet
NOTICE.txt
adb
android
apkbuilder
ddms
dmtracedump
draw9patch
emulator
Cabotools markoS
hierarchyviewer
hprof-conv
layoutopt
lib
mksdcard
source.properties
sqlite3
traceview
zipalign
    
```

## HELLO WORLD!

marakana

## Create New Project

Use the Eclipse tool to create a new Android project.

Here are some key constructs:

Project	Eclipse construct
Target	minimum to run
App name	whatever
Package	Java package
Activity	Java class

marakana

marakana

## The Manifest File

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.marakana"
    android:versionCode="1"
    android:versionName="1.0">
    <application android:icon="@drawable/icon"
        android:label="@string/app_name">
        <activity android:name=".HelloAndroid"
            android:label="@string/app_name">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
    <uses-sdk android:minSdkVersion="5" />
</manifest>
```

marakana



## The Layout Resource

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    >
<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="@string/hello"
    />
</LinearLayout>
```



## The Java File

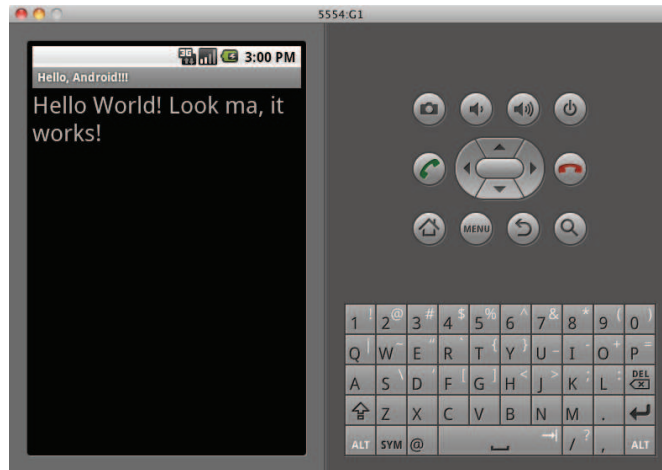
```
package com.marakana;

import android.app.Activity;
import android.os.Bundle;

public class HelloAndroid extends Activity {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
    }
}
```



## Running on Emulator



## MAIN BUILDING BLOCKS

marakana

## Activities

Activity is to an application what a web page is to a website. Sort of.

Android Application

Main Activity

Another Activity

Another Activity

marakana

## Activity Lifecycle

Activities have a well-defined lifecycle. The Android OS manages your activity by changing its state. You fill in the blanks.

```

    graph TD
      Starting([Starting]) -- "(1) onCreate()  
(2) onStart()  
(3) onRestoreInstanceState()  
(4) onResume()" --> Running([Running])
      Running -- "(1) onSaveInstanceState()  
(2) onPause()" --> Paused([Paused])
      Paused -- "onResume()" --> Running
      Paused -- "(1) onSaveInstanceState()  
(2) onStop()" --> Stopped([Stopped])
      Stopped -- "(3) onResume()  
(2) onStart()  
(1) onRestart()" --> Running
      Paused -- "<process killed>" --> Destroyed([Destroyed])
      Stopped -- "onDestroy()  
or  
<process killed>" --> Destroyed
  
```

marakana

## Intents

Intents are to Android apps what hyperlinks are to websites. They can be implicit and explicit. Sort of like absolute and relative links.

```

    graph TD
      subgraph App1 [Android Application]
        direction LR
        MA1(Main Activity) -- Intent --> AA1(Another Activity)
      end
      subgraph App2 [Android Application]
        direction RL
        AA2(Another Activity) -- Intent --> MA2(Main Activity)
      end
      AA1 -- Intent --> AA2
  
```

marakana

## Services

A service is something that can be started and stopped. It doesn't have UI. It is typically managed by an activity. Music player, for example

## Service Lifecycle

marakana

Service also has a lifecycle, but it's much simpler than activity's. An activity typically starts and stops a service to do some work for it in the background. Such as play music, check for new tweets, etc.

```

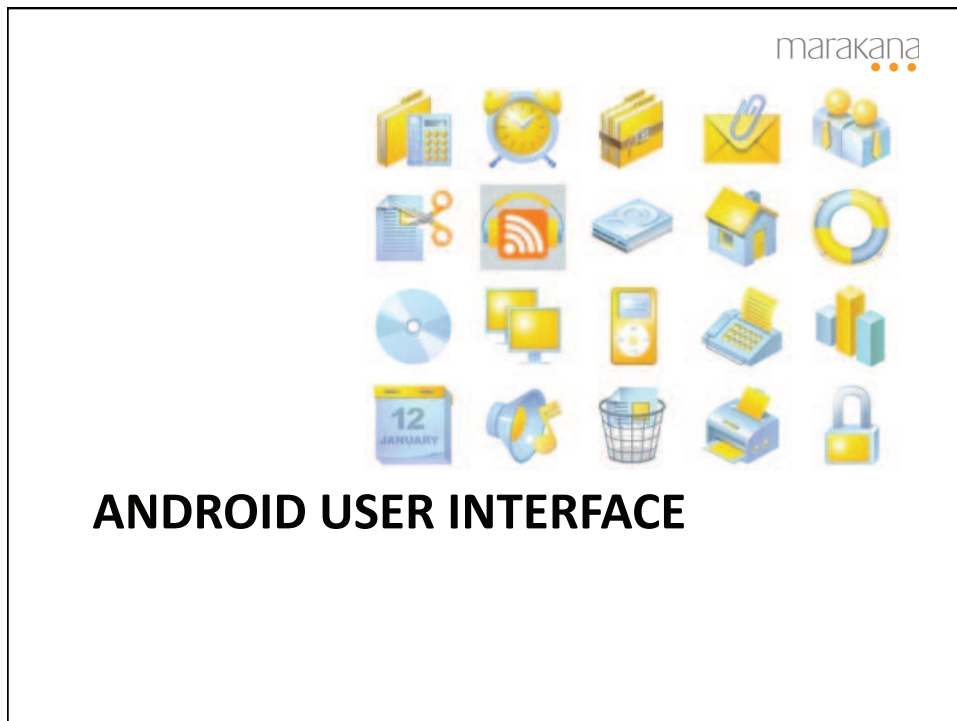
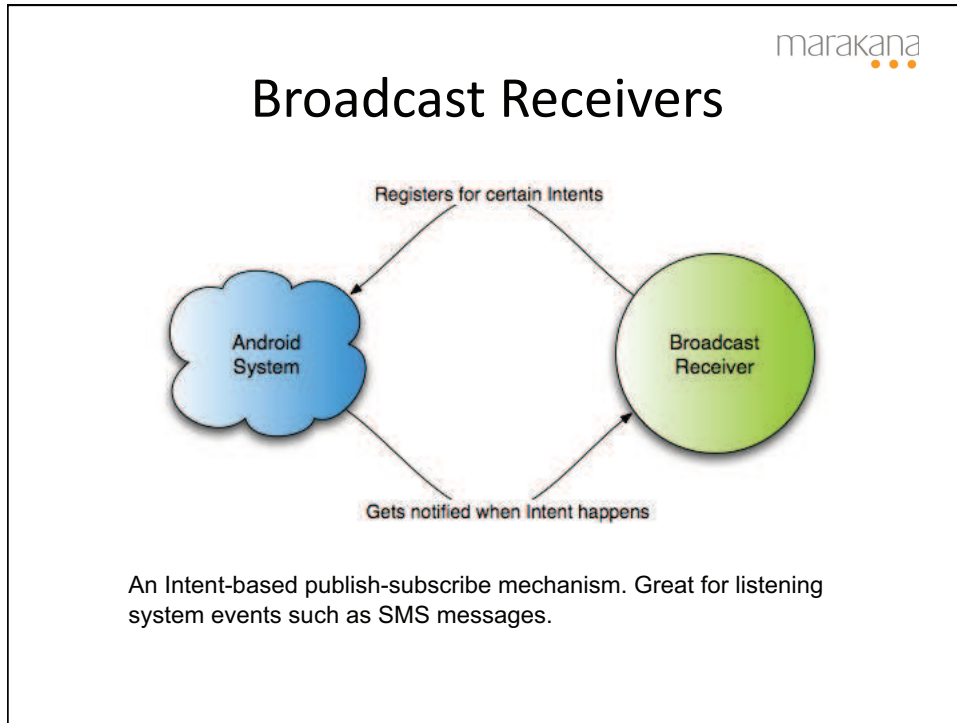
    graph TD
        Starting([Starting]) -- "(1) onCreate()  
(2) onStart()" --> Running([Running])
        Running -- "onStop()" --> Stopped([Stopped])
        Stopped -- "onStart()" --> Running
        Stopped -- "onDestroy()  
or  
<process killed>" --> Destroyed([Destroyed])
    
```

## Content Providers

marakana

Content Providers share content with applications across application boundaries. Examples of built-in Content Providers are: Contacts, MediaStore, Settings and more.

The diagram shows a green cylinder labeled "Content Provider" with a list of methods on the left: Content URI, insert(), update(), delete(), and query().

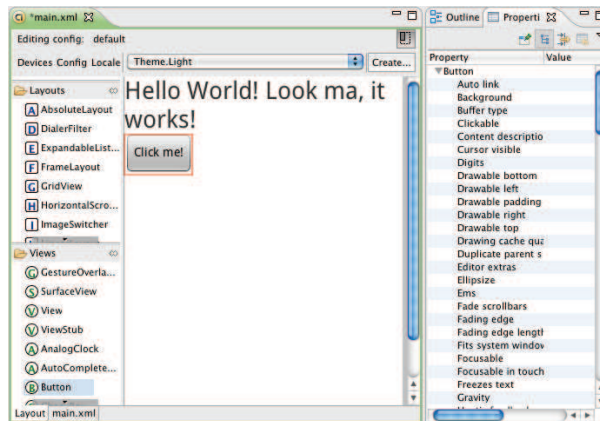


## Two UI Approaches

Procedural	Declarative
You write Java code Similar to Swing or AWT	You write XML code Similar to HTML of a web page

You can mix and match both styles.  
Declarative is preferred: easier and more tools

## XML-Based User Interface

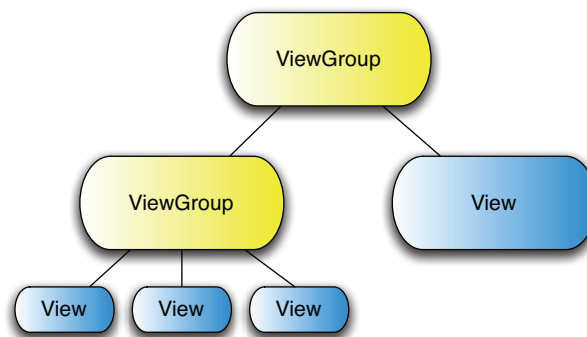


Use WYSIWYG tools to build powerful XML-based UI.  
Easily customize it from Java. Separate concerns.

## Dips and Sps


<b>px</b> (pixel)	Dots on the screen
<b>in</b> (inches)	Size as measured by a ruler
<b>mm</b> (millimeters)	Size as measured by a ruler
<b>pt</b> (points)	1/72 of an inch
<b>dp</b> (density-independent pixel)	Abstract unit. On screen with 160dpi, 1dp=1px
<b>dip</b>	synonym for dp and often used by Google
<b>sp</b>	Similar to dp but also scaled by users font size preference

## Views and Layouts



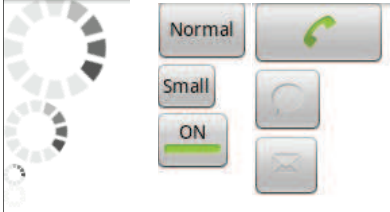

ViewGroups contain other Views but are also Views themselves.




marakana 

## Common UI Components

Android UI includes many common modern UI widgets, such as Buttons, Tabs, Progress Bars, Date and Time Pickers, etc.

marakana 

## Selection Components

Some UI widgets may be linked to zillions of pieces of data. Examples are ListView and Spinners (pull-downs).

Action	<input type="checkbox"/>
Adventure	<input type="checkbox"/>
Animation	<input type="checkbox"/>
Children	<input checked="" type="checkbox"/>
Comedy	<input checked="" type="checkbox"/>
Documentary	<input type="checkbox"/>
Drama	<input type="checkbox"/>

marakana

## Adapters

Abbaye de Belloc
Abbaye du Mont des Cats
Abertam
Abondance
Ackawi
Acorn
Adelocst

```

graph LR
    Adapter(Adapter) --- DS[(Data Source)]
    
```

To make sure they run smoothly, Android uses Adapters to connect them to their data sources. A typical data source is an Array or a Database.

marakana

## Complex Components

Certain high-level components are simply available just like Views. Adding a Map or a Video to your application is almost like adding a Button or a piece of text.

## marakana

# Menus and Dialogs

If you want to choose another menu resource, go back and re-run this activity.

First most often	Middle most often
Last most often	First least often
Middle least often	Last least often

## marakana

# Graphics & Animation

Android has rich support for 2D graphics.  
 You can draw & animate from XML.  
 You can use OpenGL for 3D graphics.

**Default**

**Custom**

Left  
Center  
Right

Positioned  
Positioned  
Positioned

Along a path  
Along a path

## Multimedia

**MediaPlayer** lets you simply specify the audio resource and play it.

**VideoView** is a View that you can drop anywhere in your activity, point to a video file and play it.

**XML:**

```
<VideoView
    android:id="@+id/video"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:layout_gravity="center" />
```



**Java:**

```
player = (VideoView) findViewById(R.id.video);
player.setVideoPath("/sdcard/samplevideo.3gp");
player.start();
```

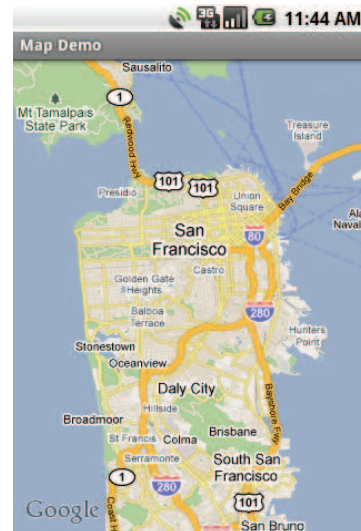
## Google Maps

Google Maps is an add-on in Android. It is not part of open-source project.

However, adding Maps is relatively easy using **MapView**.

**XML:**

```
<com.google.android.maps.MapView
    android:id="@+id/map"
    android:clickable="true"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:apiKey="0EfLSgdSCWIN...A" />
```





## 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.

# Debugger

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

# TraceView

Name	Incl %	Inclusive	Excl %	Exclusive	Time/Call
217 android/database/sqlite/SQLiteDatabase.native_setLocale (Ljava/lang/String;I)V	0.5%	8.235	0.5%	8.235	1+0
164 android/database/sqlite/SQLiteDatabase.setLocale (Ljava/util/Locale;V)	100.0%	8.235			1/1
218 java/lang/BootClassLoader.loadClass (Ljava/lang/String;ZLjava/lang/Class;)	0.5%	8.222	0.1%	1.633	19+0
219 android/util/SparseArray.binarySearch (II)I	0.5%	8.219	0.5%	8.219	133+0
220 android/os/Binder.execTransact (II)I	0.5%	8.207	0.0%	0.498	2+0
221 android/view/View.getDrawState ()I	0.5%	8.177	0.2%	3.367	60+0
222 android/graphics/drawable/TransitionDrawable\$TransitionState.newDrawable (L	0.5%	8.135	0.0%	0.205	2+0
223 android/graphics/drawable/NinePatchDrawable.getConstantState ()LAndroid/gra	0.5%	8.107	0.4%	5.772	51+0
224 java/util/concurrent/locks/ReentrantLock.unlock ()V	0.5%	8.084	0.1%	1.018	12+0
225 android/content/Context.obtainStyledAttributes (II)LAndroid/content/res/Typed	0.5%	8.073	0.1%	1.013	7+0
226 android/view/View.setFlags (I)I	0.5%	8.057	0.3%	5.510	62+0

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.

## 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

- Android is based on Java.
- You write Java code, but technically don't run it.
- Java is augmented with XML, mostly for UI purposes.
- Android Java is mostly based on Java SE with replacement UI libraries.

Marko Gargenta, Marakana.com  
[marko@marakana.com](mailto:marko@marakana.com)  
 +1-415-647-7000

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