Android Platform Introduction

ICT2105 Mobile Application Development

Jeannie S. Lee

Spring 2020

Overview

History

OS Structures (Recap)

Android Operating System Stack

Virtual Machines (Recap)

Application Framework Components


Google Building 44 (Circa 2013)


#Android9


OREO

All About Android

(Mostly) Open source software stack

AOSP http://source.android.com

Open source software stack for wide array of mobile devices

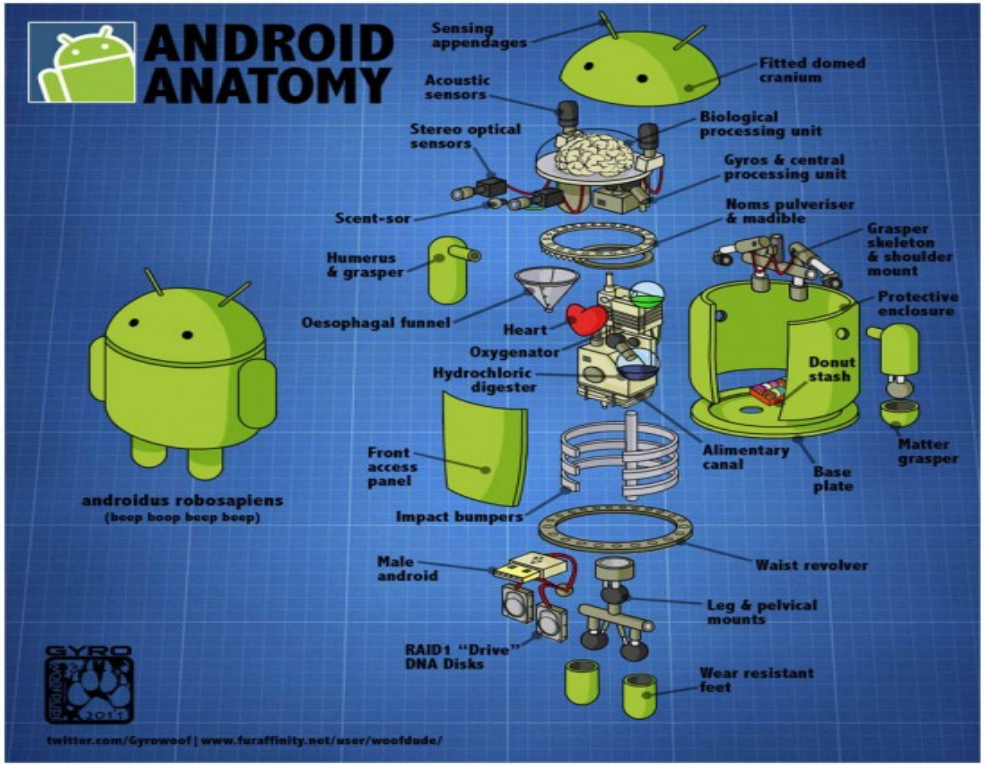
Linux-based, but lacks many of the commonly known executables

Licensed under Apache 2.0

ARM (32-bit or 64 bit), x86, MIPS processor architecture

Yearly version releases

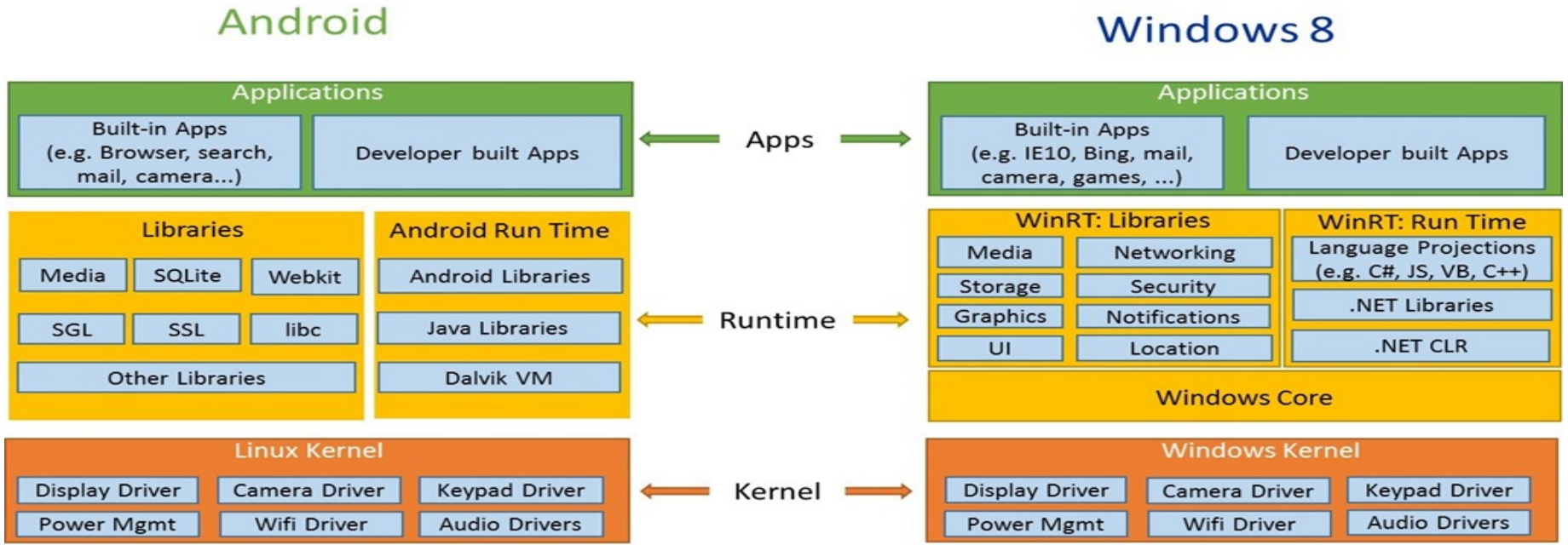
Code changes rapidly!



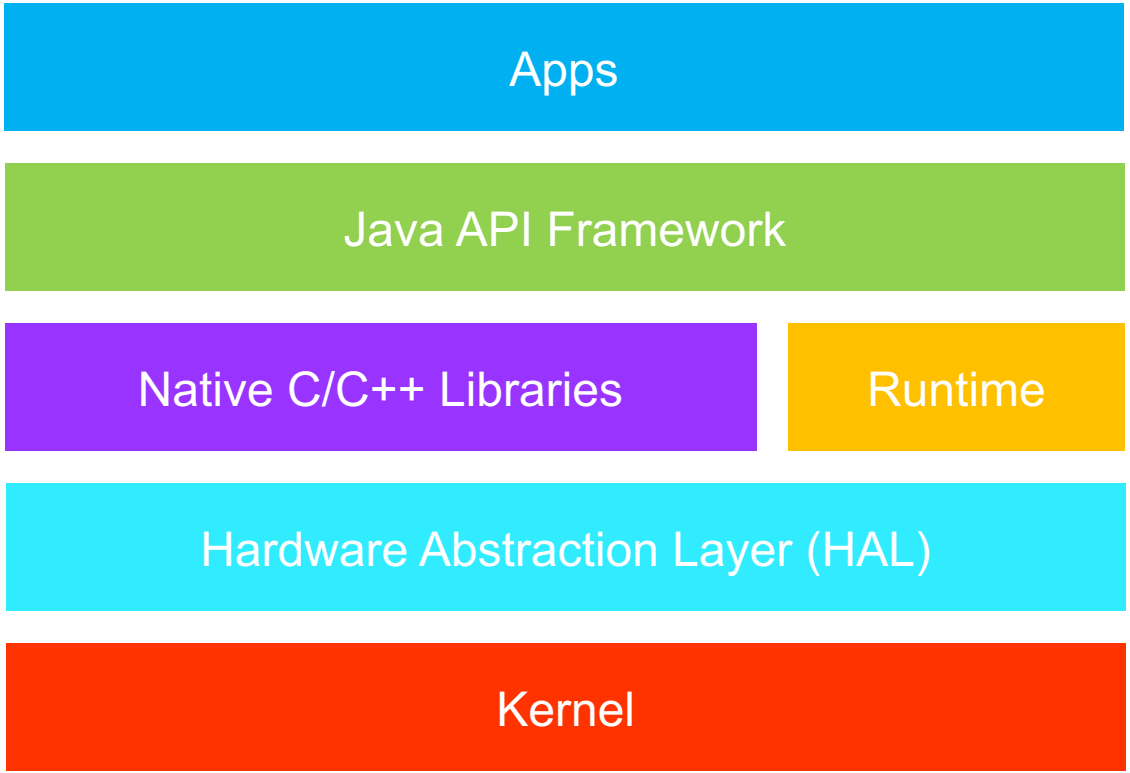
OS Structures (Recap)

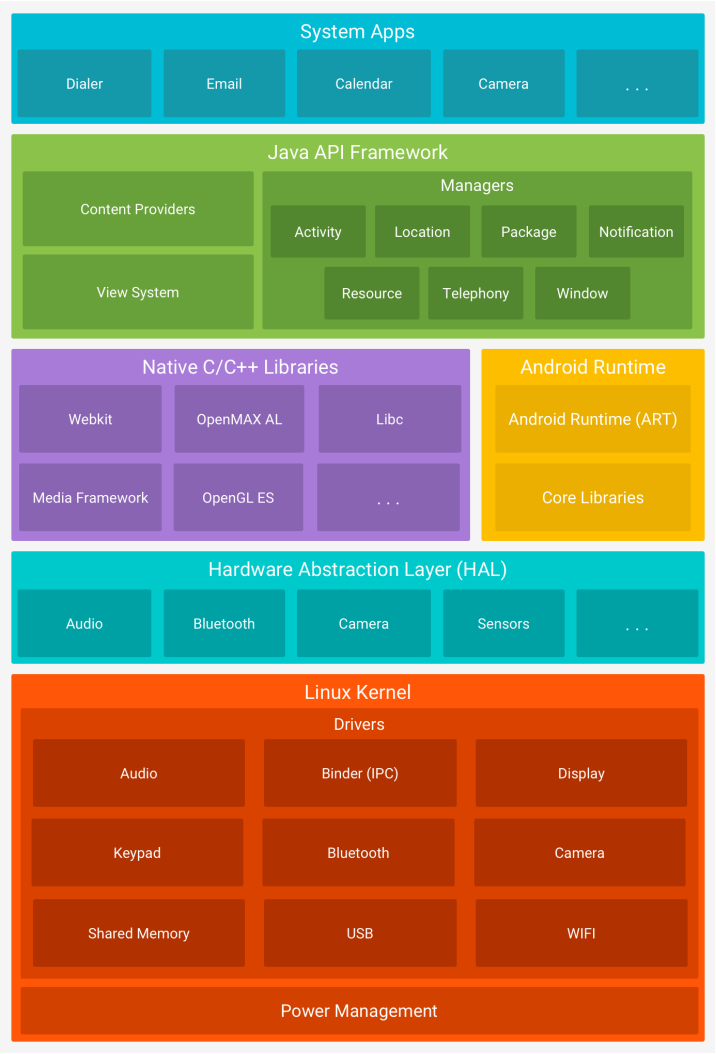
How An Operating System Is Put Together

A LAYERED STRUCTURE:

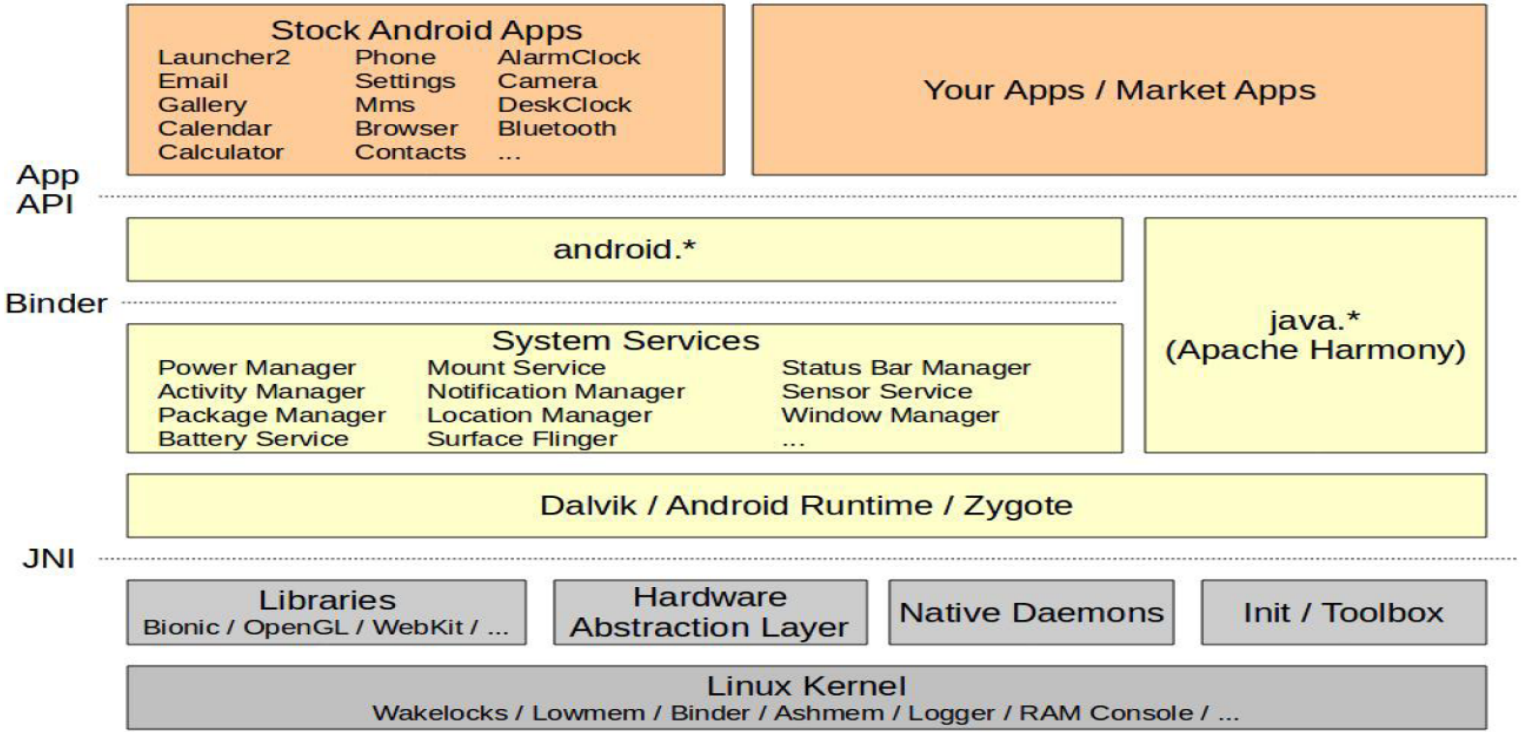


Source: WPI CS 502 Operating Systems, & Silberschatz, Gagne, Galvin

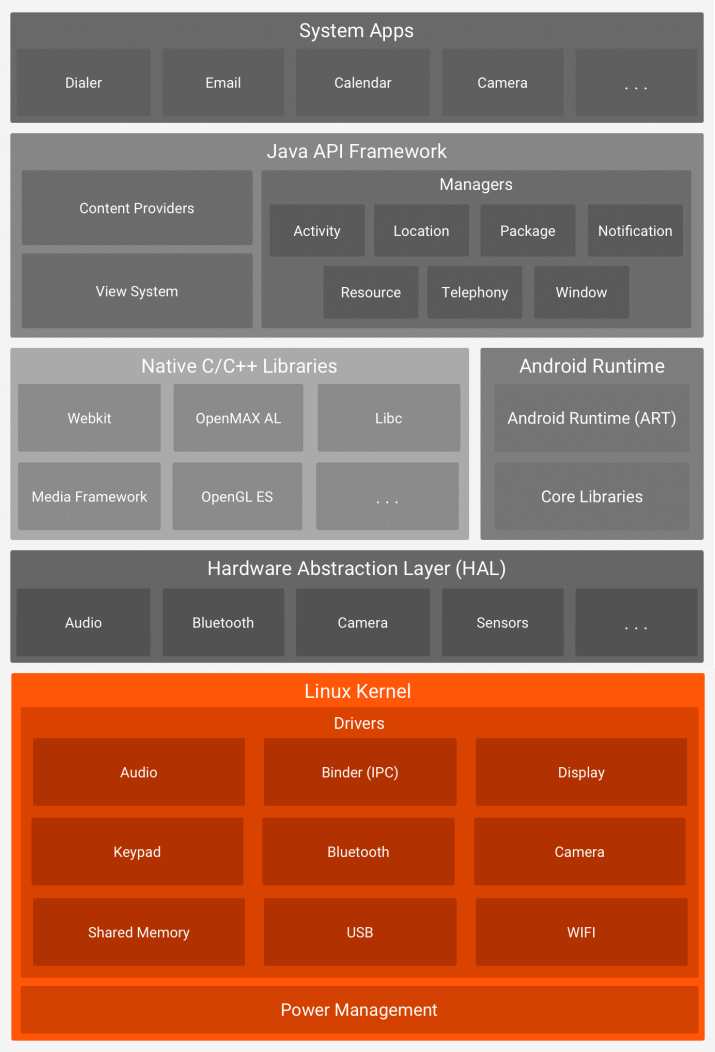

Apps
Java API Framework
Native C/C++ Libraries
Runtime
Hardware Abstraction Layer (HAL)
Kernel


System Apps
Dialer
Email
Calendar
Camera
. . .
Java API Framework
Content Providers
View System
Managers
Activity
Location
Package
Notification
Resource
Telephony
Window
Native C/C++ Libraries
Webkit
OpenMAX AL
Libc
Media Framework
OpenGL ES
. . .
Android Runtime
Android Runtime (ART)
Core Libraries
Hardware Abstraction Layer (HAL)
Audio
Bluetooth
Camera
Sensors
. . .
Linux Kernel
Drivers
Audio
Binder (IPC)
Display
Keypad
Bluetooth
Camera
Shared Memory
USB
WIFI
Power Management

Android OS Stack (Detailed)



Kernel



Kernel

Abstraction layer between the hardware and rest of the software

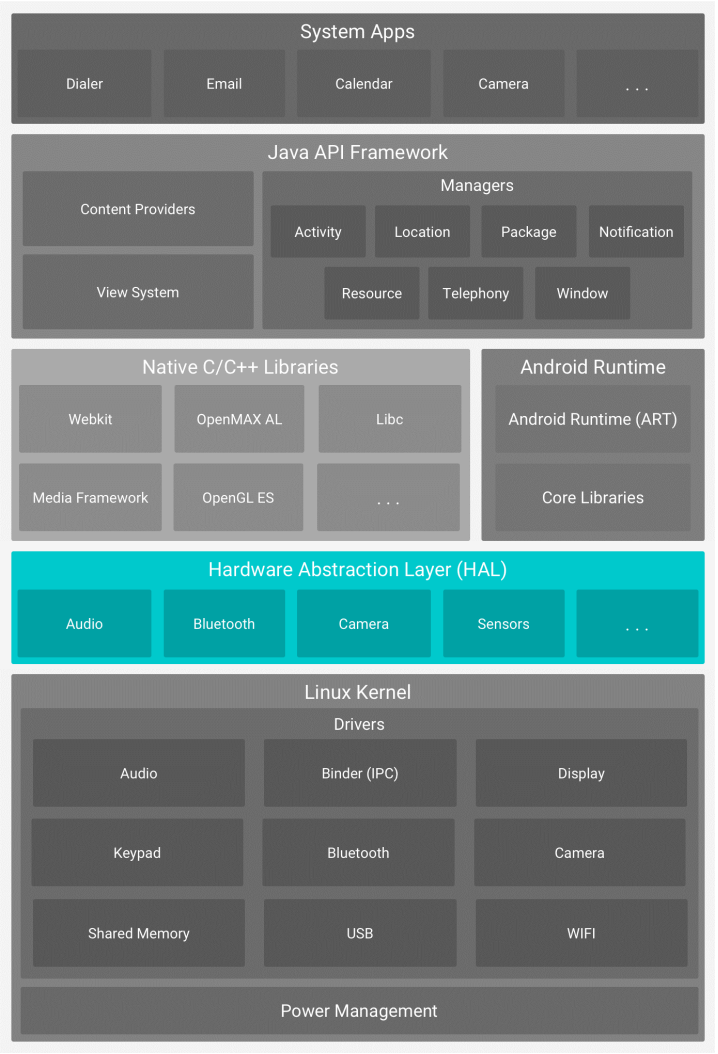
Android/Linux, NOT GNU/Linux

Linux kernel for core system services

Heavy modifications/enhancements to linux kernel

Alarm driver, ashmem, power manager, low memory killer, binder driver (IPC interface), kernel debugger, logger, etc

Hardware Abstraction Layer (HAL)



Hardware Abstraction Layer (HAL)

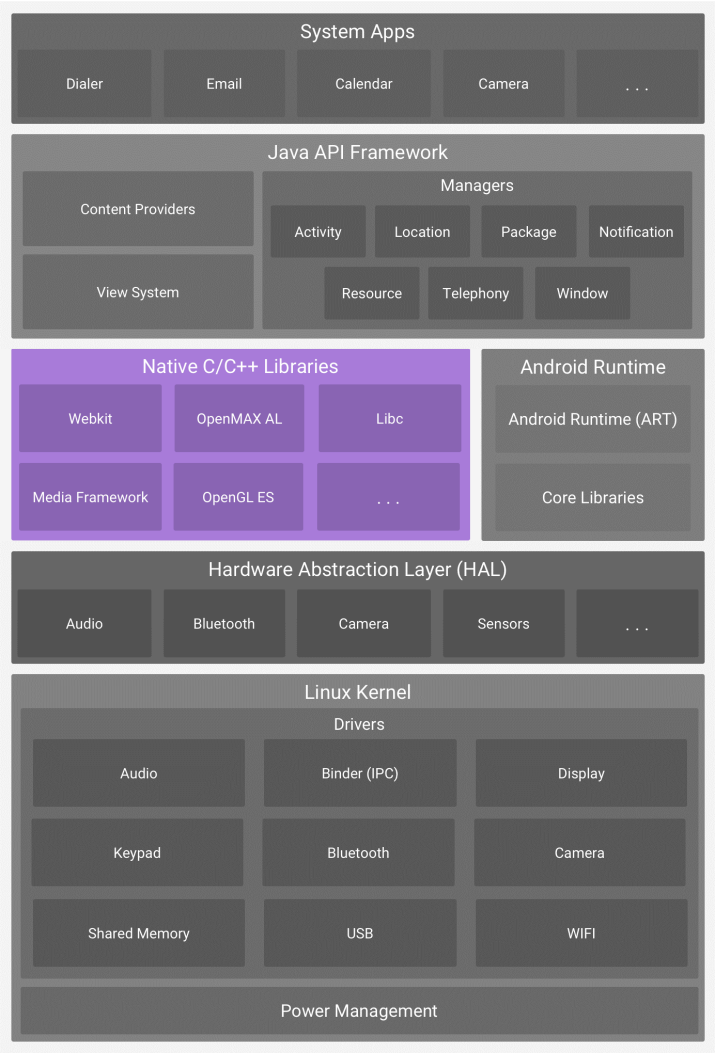
Android interface to hardware

Allows for kernel drivers or user-space drivers

Supports drivers without standard APIs

Lies between the kernel space driver and the user space Android native library

Native Libraries



Native Libraries

C/C++ libraries used by various components of the Android system

libc: C Standard Library

Google developed “Bionic” for memory constrained platforms

Lacks features in full libc implementation

SSL: Secure Socket Layer

SGL: 2D image engine

OpenGL ES: 3D image engine (Vulkan introduced in Nougat 7.0)

Media Framework: audio, video, image decode and playback

SQLite: Embedded database

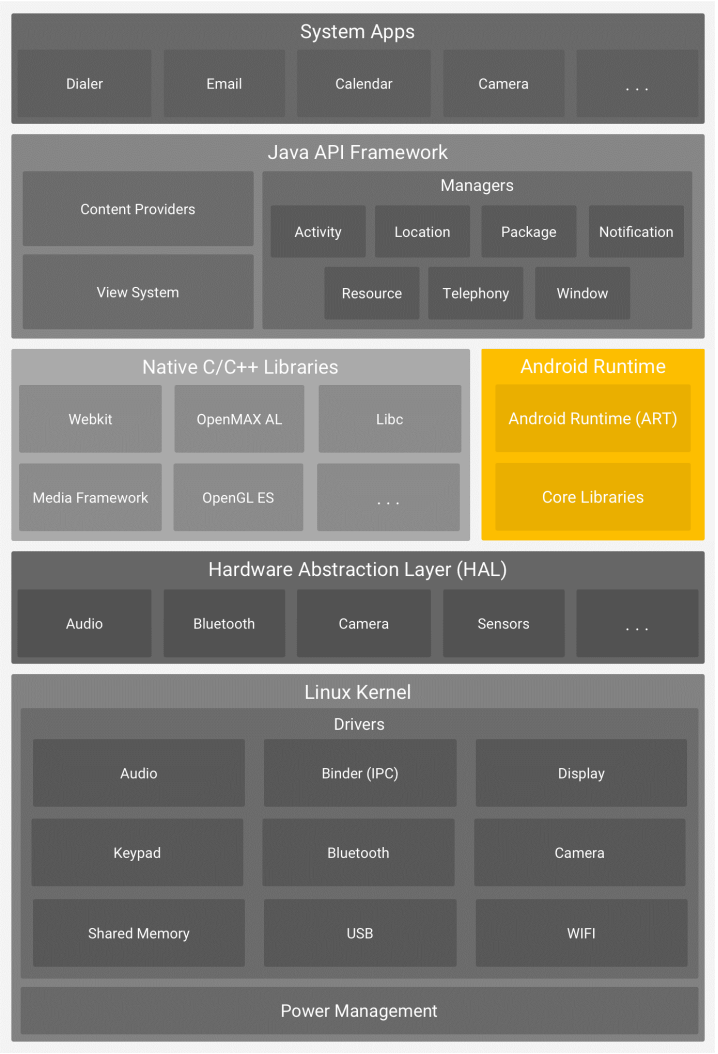
WebKit: browser layout engine

FreeType: Font rasterization engine

Surface Manager: Management of windows in different apps

Linux daemons managing system features such as telephony, networking, sensors, etc

Android Runtime



Android Runtime

Core libraries that provide Java language functionality

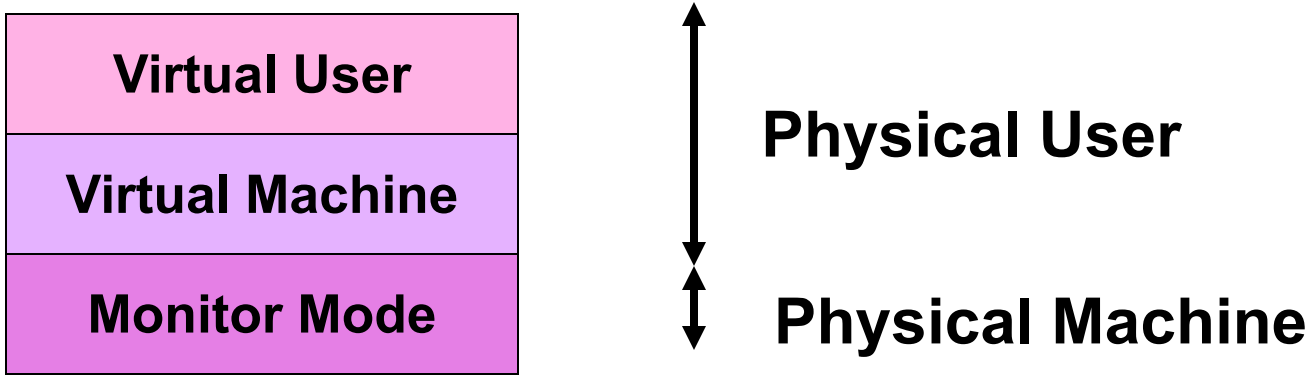
App runs in its own process, with own instance of VM, sandboxing of applications

A device can run multiple VMs!

Virtual Machine (Recap)

In a Virtual Machine - each process "seems" to execute on its own processor with its own memory, devices, etc.

- The resources of the physical machine are shared. Virtual devices are sliced out of the physical ones. Virtual disks are subsets of physical ones.
- Useful for running different OS simultaneously on the same machine.



Source: WPI CS 502 Operating Systems, & Silberschatz, Gagne, Galvin

Virtual Machine (Recap)

Provides an interface identical to the underlying bare hardware

Virtual machine implementation intercepts operations and interprets them

Several virtual machines may share the resources of a physical computer:

CPU scheduling: create illusion that users have their own processor

Virtual disks with virtual file systems on physical disk / file system

A normal user time-sharing terminal serves as the virtual machine operator’s console

Possible to run multiple and different OS’s on the same physical computer

Android Runtime (ART)

New runtime introduced in Android Lollipop (2014)

Ahead Of Time (AOT) app compilation

- ART compiles bytecode to native machine code
- When app launched, no interpretation is required for execution
- OAT files instead of DEX files as the stored executable format

Improved garbage collection (GC), reduced execution launch time, improved system responsiveness, longer battery life

Cross platform: ARM, x86, MIPS

Dalvik (Old Runtime)

Alternative bytecode representation

Not Java VM, Google's own Java-based VM

Register based, optimization for low memory requirements

Executes files in Dalvik Executable Format (.dex), translates bytecode instructions to machine instructions

Just In Time (JIT) compilation

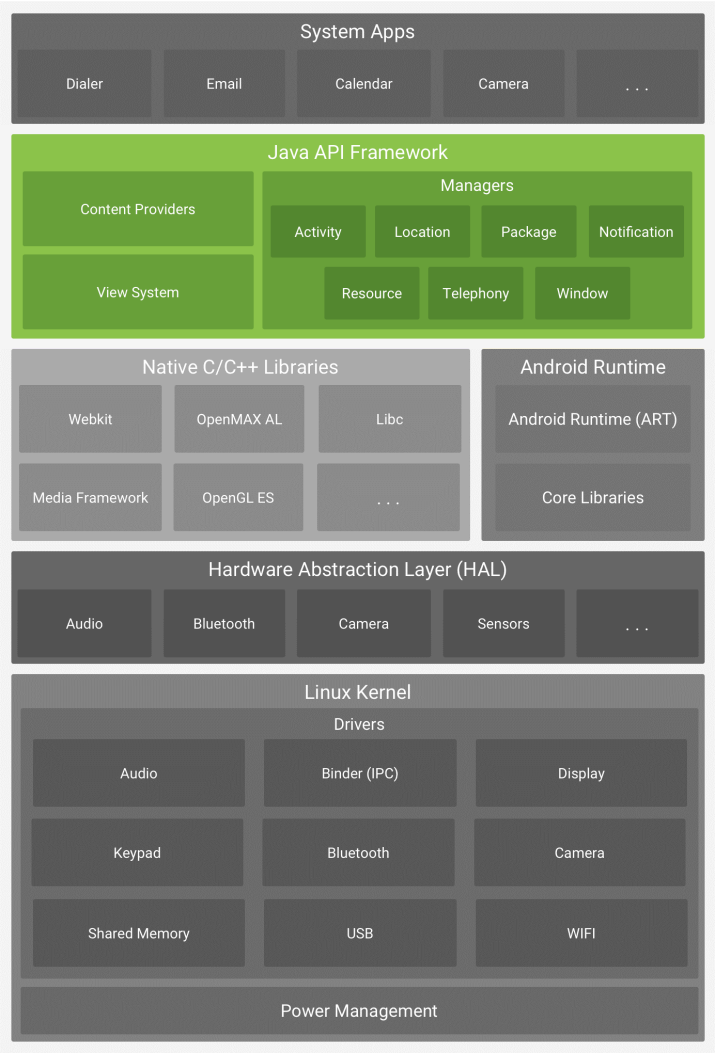
DX tool converts classes to .dex

Provides isolation among processes while running multiple simultaneous VM instances

Does not provide sandboxing to isolate a process from filesystem access

Responsibility of the OS and filesystem privilege control

Java API Framework



API Framework

Implementation of exposed APIs used by applications

Core Java libraries

Most standard Java SE packages: java.*, javax.*

Wrapper classes for native libraries e.g. SQLite

Auxiliary packages e.g. JSON, XML parsers, Apache HTTP Client

Classes for implementing application components

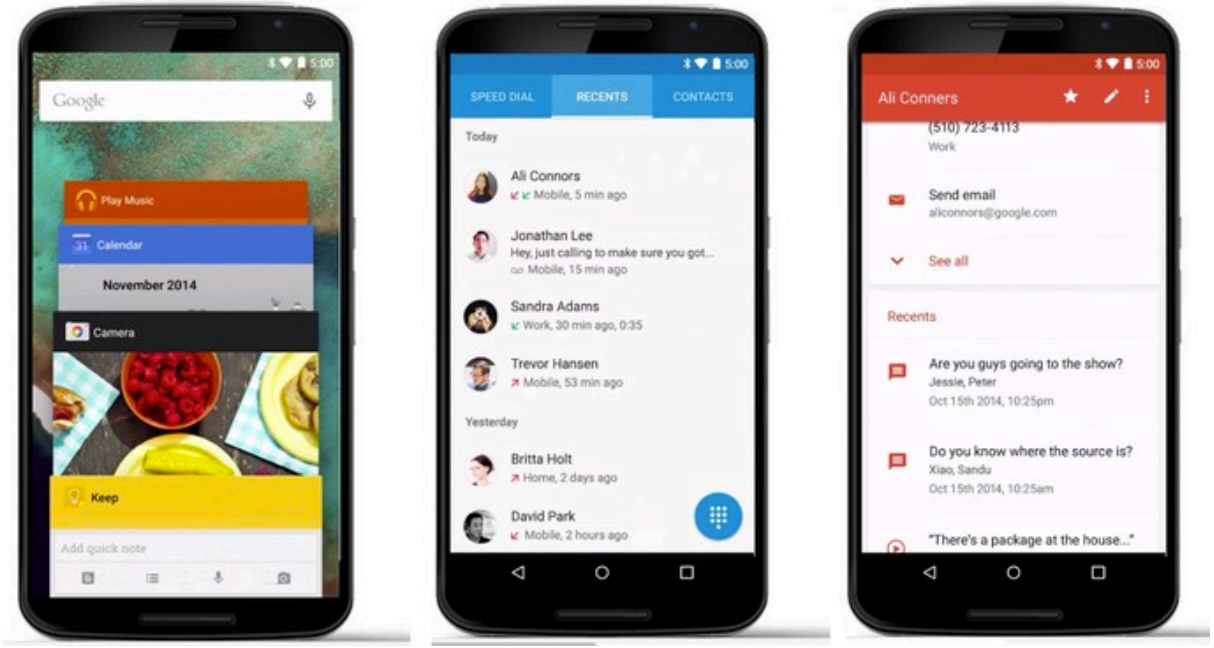
android.*

API Framework

View System

Provides common UI elements

Text boxes, buttons, icons, etc.

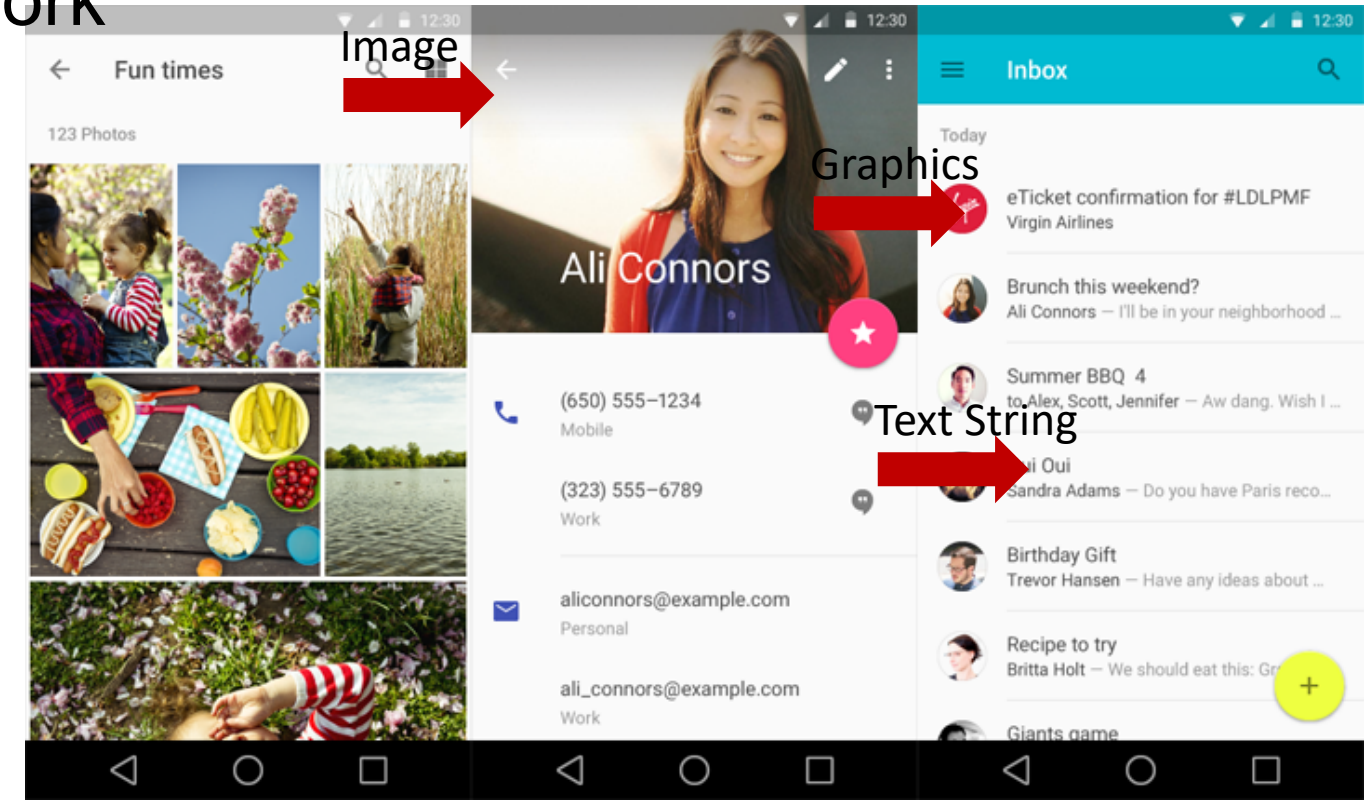


API Framework

Resource Manager

Handles non-compiled resources

Strings, graphics, layout files

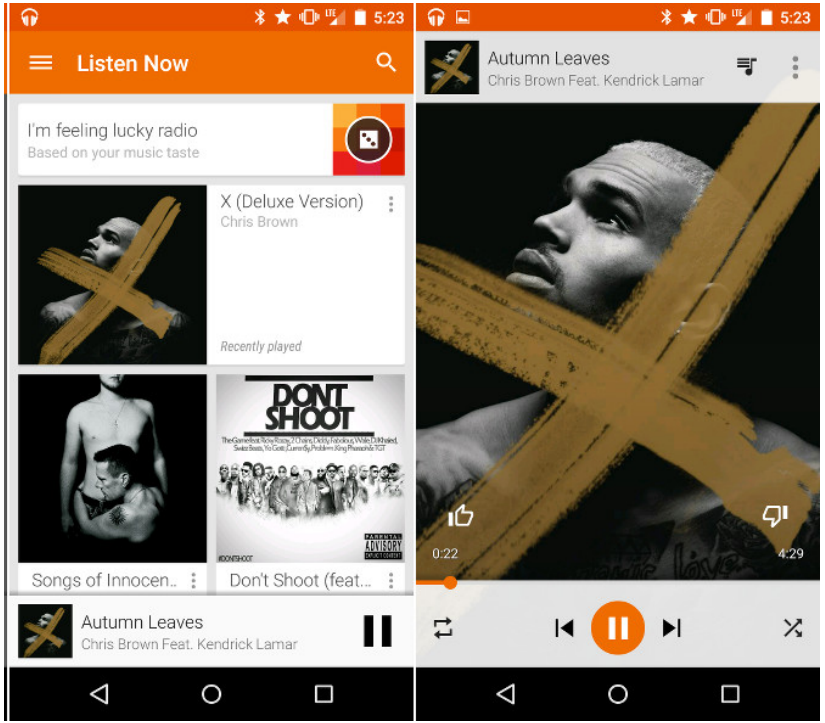


API Framework

Activity Manager

Manages app lifecycle and navigation stack

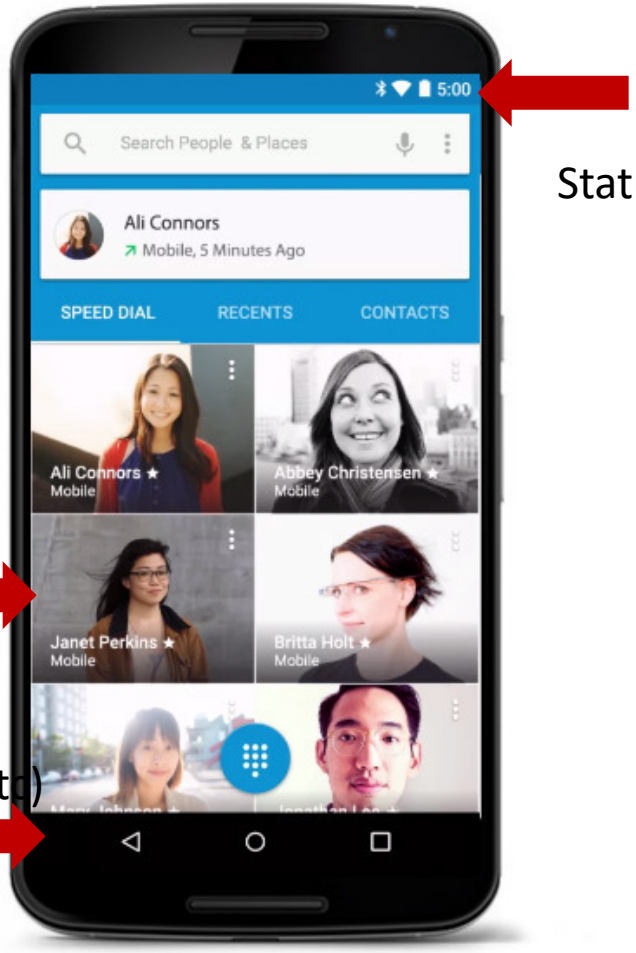
Activity: App component that provides a single focused task for a user



API Framework

Window Manager

Manages windows comprising the app



Status Bar (Notifications)

Window

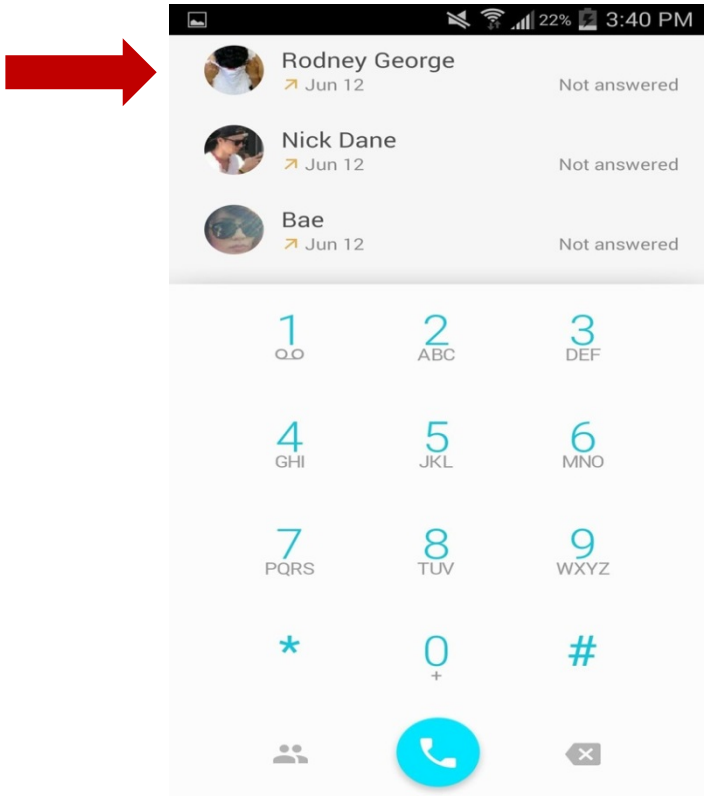
Nav Bar (Back, Home, etc)

API Framework

ContentProvider

Handles data sharing between applications

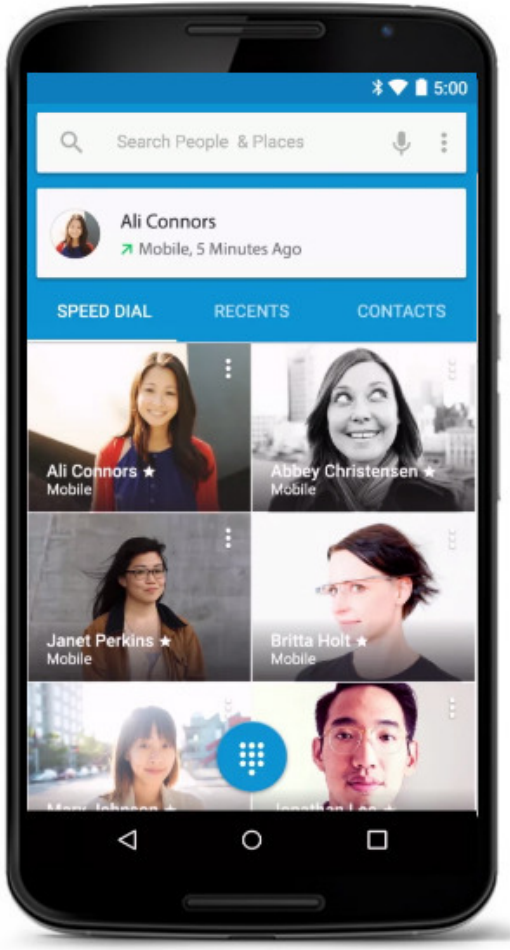
Phonebook



API Framework

Package Manager

Keeps track of app packages on the device

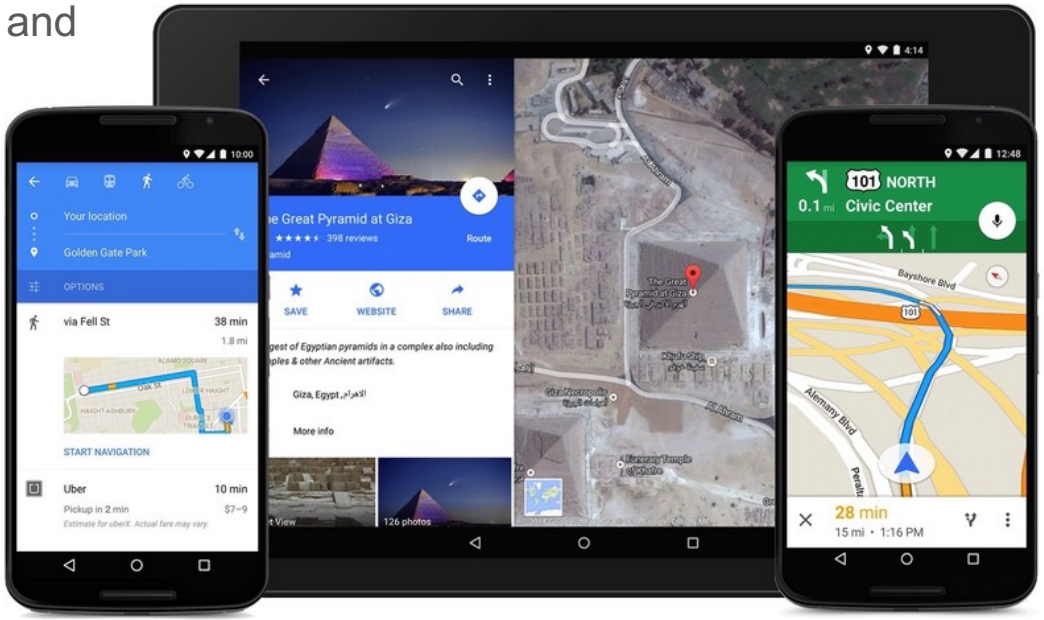


API Framework

Location Manager

Keeps track and provides location and movement information

GPS, wifi location, etc.

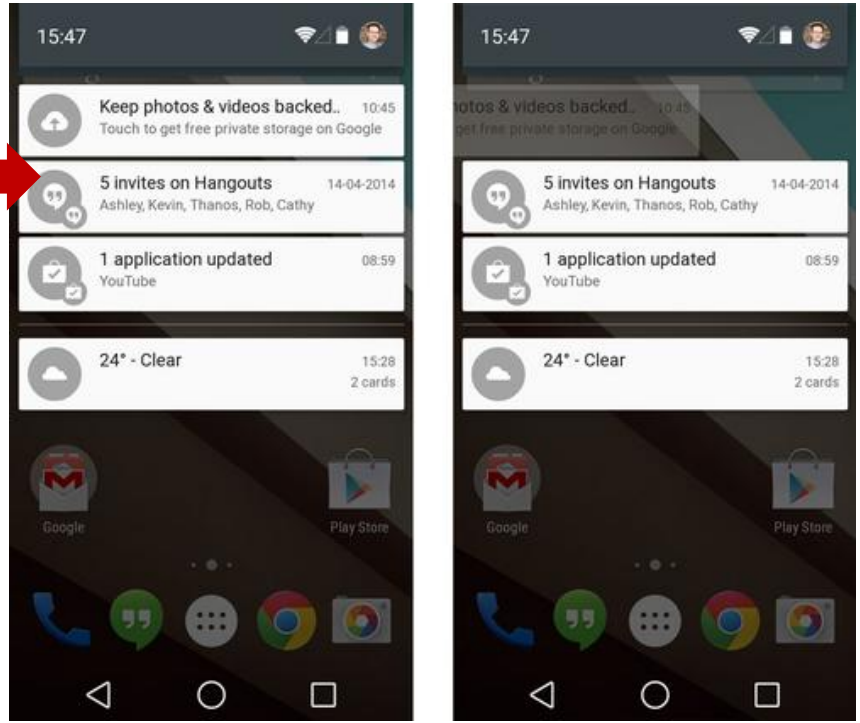


API Framework

Notification Manager

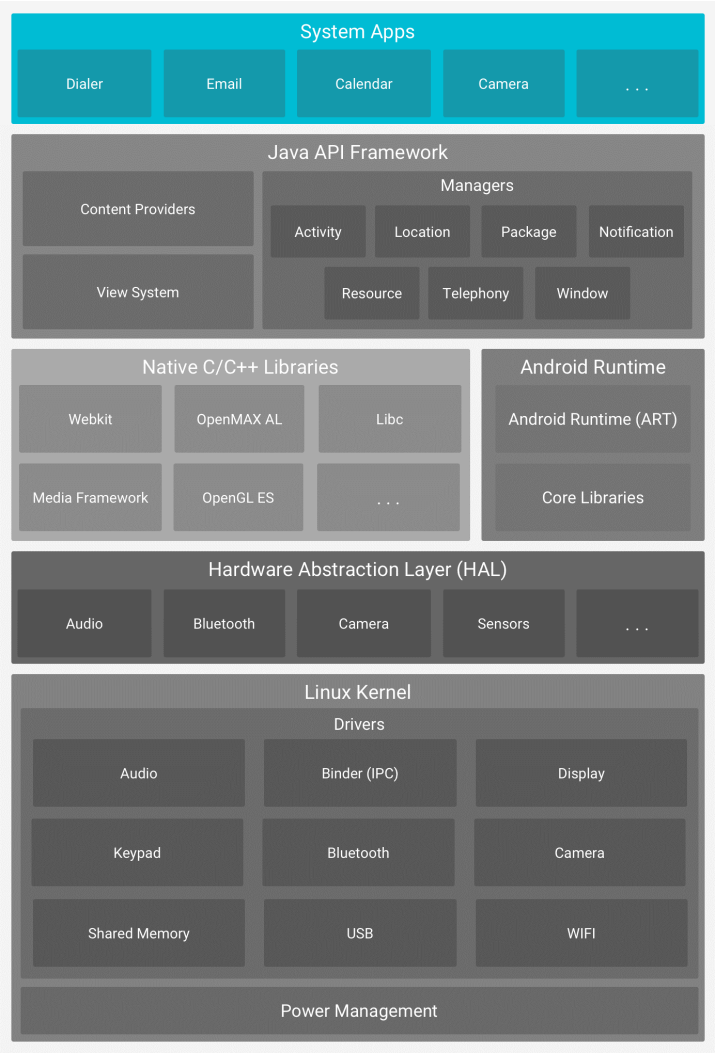
Handles notification icons and messages in the status bar when events occur

Notifications



System Apps

(and regular applications)



Applications

Consist of “system” and “user installed” apps

Ships with set of core “system” applications

Email client, SMS program, calendar, maps, browser, contacts, etc.

Can’t uninstall system apps!

System and user apps have same structure and use the same app framework layer

System Apps

Home/Launcher

Main home screen, replaceable!

Contacts

Phonebook/people database

Phone

Dialer for phone numbers

Browser

Surf web pages

Email

Read and compose emails

App Workflow

Most apps written in Java

Compile to Java bytecode files

DX tool converts java bytecode files into a single bytecode .dex file

Dalvik VM executes .dex bytecode file

In Android 5.0 and above, DEX files are converted into OAT files at installation time (new ART/Android Runtime)

App Structure

Distribute app as a single file, known as an ***application package*** or *APK*

Dalvik (VM) executable code

Compiled from the Java/Kotlin source code

Resources, such as images, audio/video clips, XML files describing layouts, etc.

Optional native shared libraries (.so files)

Native shared libraries are compiled from C/C++ code using NDK

APKs must be signed with a security certificate before they can be installed on a device

Platform Versions

API Level is an integer value that uniquely identifies the framework API revision offered by a version of the Android platform.

http://developer.android.com/guide/topics/manifest/uses-sdk-element.html#ApiLevels

http://developer.android.com/about/dashboards/index.html

Platform Version	API Level	VERSION_CODE
Android 10.0	29	Q
Android 9	28	P
Android 8.1	27	O_MR1
Android 8.0	26	O
Android 7.1.1 Android 7.1	25	N_MR1
Android 7.0	24	N
Android 6.0	23	M
Android 5.1	22	LOLLIPOP_MR1
Android 5.0	21	LOLLIPOP
Android 4.4W	20	KITKAT_WATCH
Android 4.4	19	KITKAT
Android 4.3	18	JELLY_BEAN_MR2
Android 4.2, 4.2.2	17	JELLY_BEAN_MR1
Android 4.1, 4.1.1	16	JELLY_BEAN

64-bit?!

Android is based on a Linux kernel

Linux has supported 64-bit for ages

Android Lollipop and above is 64-bit

Theoretical support for Exabytes of RAM

Need to be careful of what the guy at the mobile phone shop tells you

Summary

Open source platform based on the Linux kernel

Complete stack of operating system, middleware and apps

Application framework is the implementation of exposed APIs used by app developers

A process is a running instance of the app code